A Short Tour Through the Wonders of the Field of Computational Complexity

P vs. NP, Boolean satisfiability problem (SAT), and more!

Christopher Soto

Bridging Math and Computer Science

Queens College of the City University of New York
November 19th, 2019

Today's Talk

- Origin of the P versus NP problem
- Difference between P and NP
- Boolean satisfiability problem (SAT)
- Naive and non-naive approaches to solving the SAT problem
- Cook-Levin Theorem
- Equivalence between P versus NP and the SAT problem

Millennium Prize Problems

Solved:

- Poincaré Conjecture (solved in 2003 by Grigori Perelman)

Unsolved:

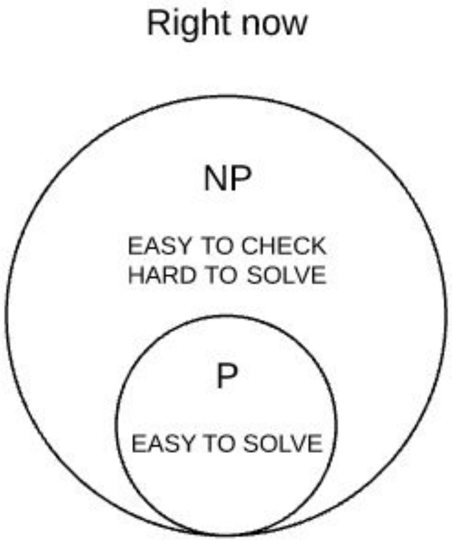
- Yang-Mills and Mass Gap
- Riemann Hypothesis
- **P vs NP Problem** (today's talk)
- Navier-Stokes Equation
- Hodge Conjecture
- Birch and Swinnerton-Dyer Conjecture

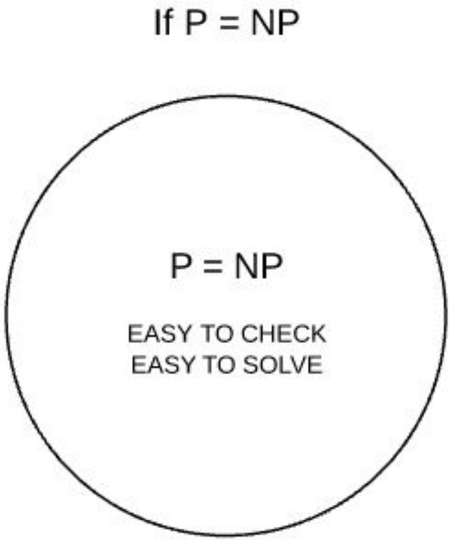
Seven problems in mathematics stated by the Clay Mathematics Institute on May 24th, 2000.

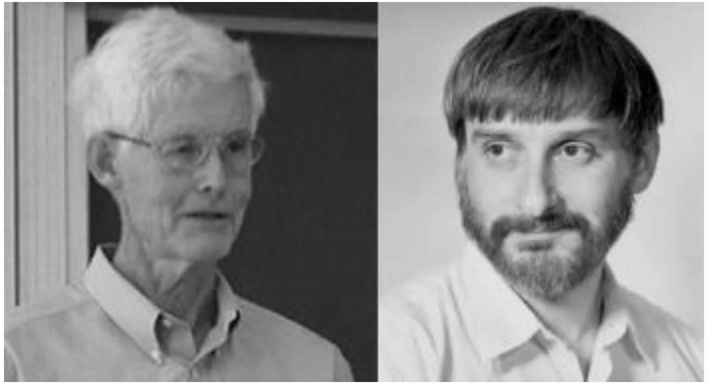
Each correct solution carries a US $1 million prize.

Origin of P versus NP

<u>The Question:</u> If it is <u>easy to check</u> that a solution to a problem is correct, is it also <u>easy to solve</u> the problem?







Stephen Cook and Leonid Levin formulated the P versus NP problem in independently in 1971.

What are P and NP Problems?

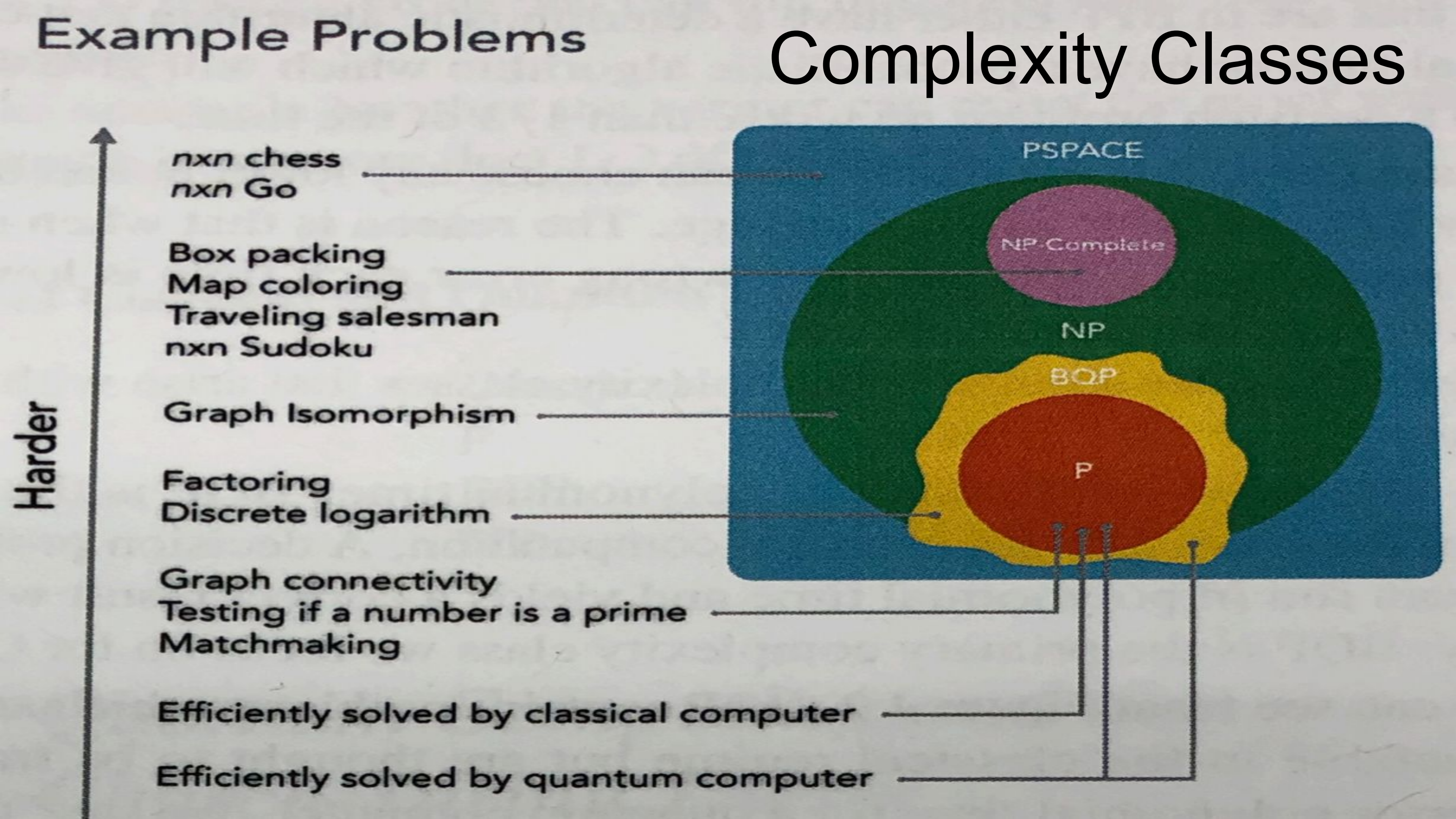
NP is the set of **problems that are easily checkable**, have proofs verifiable in polynomial time.

- Prime factorization
- Boolean satisfiability problem (SAT)
- Vehicle routing (TSP)

P is the set of **problems that are easily solvable**, problems that can be solved in polynomial time.

- Multiplication
- Sorting

Complexity Classes



Multiplication is a problem in the polynomial time class (P), can easily check this product:

37975227936943673922808872755

445627854565536638199

x

40094690950920881030683735292

761469389214899724061

What about the reverse? What are the factors of this product?
Prime factorization is a problem in the nondeterministic polynomial time class (NP)

1522605027922533360535618378
1326374297180681149613806886
5790849458012296325895289765
4000350692006139

This is RSA-100 (has 100 decimal digits) and took a few days using parallel computing in 1991 to be factored by Arjen K. Lenstra.

Boolean Satisfiability Problem (SAT)

- Given a boolean formula, determine whether the values of a given Boolean formula can be consistently replaced by the values TRUE or FALSE in such a way that the boolean formula evaluates to TRUE.

For example:

Given: A v B

A = true
B = false

If either A is true or B is true, the boolean statement is than true, therefore:

A v B is therefore satisfiable!

Given: ¬B ∧ B

B = true?
B = false?

No value of B exists to make the boolean statement true, therefore:

¬B ∧ B is not satisfiable!

Naive and non-naive approaches to solving the SAT problem

- One naive approach of the boolean satisfiability problem would be to check each and every part of the statement.

- Recursively this would be exponential time complexity which is not what we want, we really want to find an algorithm to solve the SAT problem efficiently in polynomial time!

Given the following boolean statement, determine the a, b, c, and d values which make the statement satisfiable:

$$(a \lor \neg b \lor c) \land (\neg a \lor d) \land (a \lor d) \lor (\neg d \lor \neg c) \land (c \lor b) \land (c \lor \neg a)$$

...

a	b	c	d	(((a ∨ (¬b ∨ c)) ∧ ((¬a ∨ d) ∧ (a ∨ d))) ∨ ((¬d ∨ ¬c) ∧ ((c ∨ b) ∧ (c ∨ ¬a))))
F	F	F	F	**F**
F	F	F	T	**T**
F	F	T	F	**T**
F	F	T	T	**T**
F	T	F	F	**T**
F	T	F	T	**T**
F	T	T	F	**T**
F	T	T	T	**T**
T	F	F	F	**F**
T	F	F	T	**T**
T	F	T	F	**T**
T	F	T	T	**T**
T	T	F	F	**F**
T	T	F	T	**T**
T	T	T	F	**T**
T	T	T	T	**T**

Cook-Levin Theorem

- The concept of NP-completeness introduced in 1971 in the Cook-Levin theorem paper.
- Showed that the Boolean satisfiability problem (SAT) is NP-Complete.
- Any problem in NP can be reduced in polynomial time to an instance of the Boolean satisfiability problem (SAT).

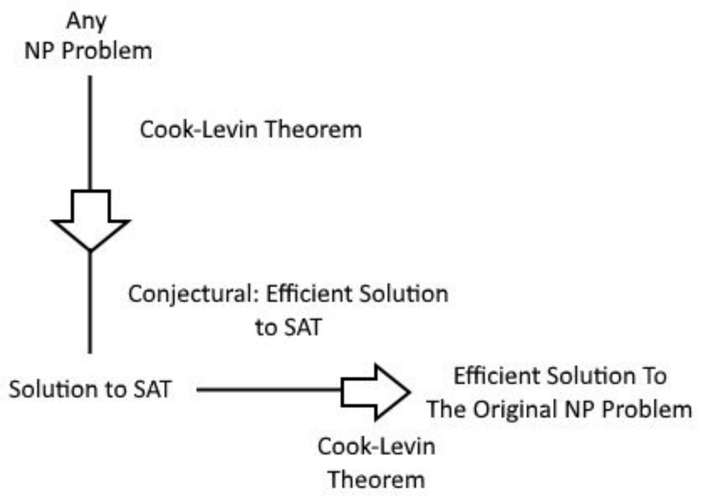
NP-Complete

- If a polynomial time algorithm exists for any of these problems, all problems in NP would be polynomial time solvable.

- Problems in NP whose individual complexity is related to that of the entire class.

Equivalence between P versus NP and the SAT Problem

- An efficient solution to the Boolean satisfiability problem (SAT) is a solution to any NP problem by the Cook-Levin theorem as any problem in NP can be reduced in polynomial time to an instance of the Boolean satisfiability problem.



How to show that P = NP

- Goal: Task of showing that there exists a polynomial-time algorithm to any NP-Complete problem

How to show that P != NP

- Goal: Show that there is no clever way to solve any NP-Complete problem.

One approach: Handicap the computer, try to prove something in a handicap state and see if it co-exists in a non-handicap state.

Second approach: Take infinite inputs to any NP-Complete problem and show it's easier for the machine to take a long-time.

Outcomes if P = NP or P != NP

If P = NP:

- Encryption schemes would be easy to crack.
- A magnitude of widespread benefits in multiple disciplines including medicine, mathematics, computer science, and more.
- More efficient markets (predicting markets, etc.)

If P != NP:

- Would allow one to show that many common problems cannot be solved efficiently.
- Shift the focus on research to solving partial solutions or solutions to other problems.

Thank you!
Any Questions?