P vs NP

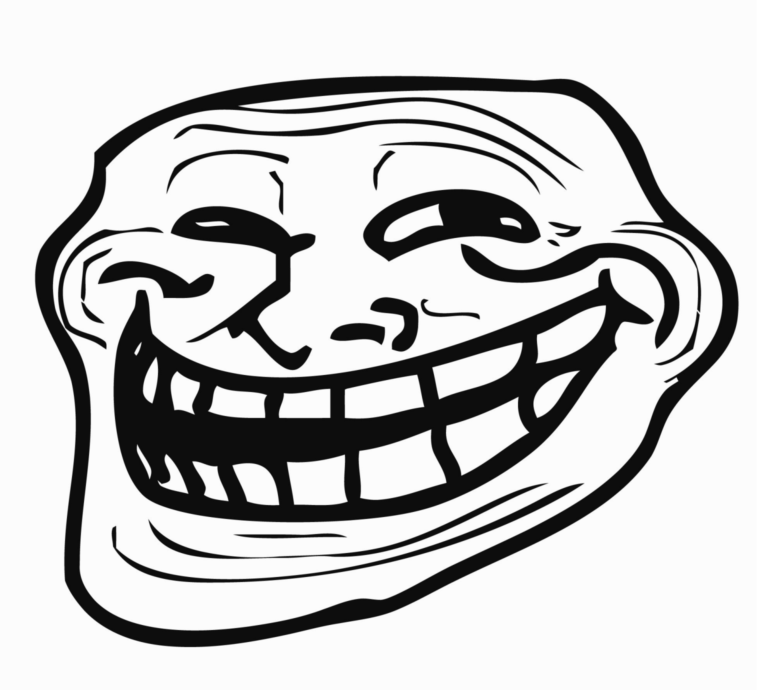
By: Handy Cho, Rachel Kaper and Anna Steinfeld

In Class Exercise:

You have 24 minutes and 30 seconds to prove (or disprove)

P = NP.

...Just Kidding



What is P vs NP?

- P stands for polynomial time which means that the complexity of the algorithm (number of operations it would take as a function of the number of data items) is O(n^k), for some constant variable k

- P = NP means that if an equation takes polynomial time on a non-deterministic model then a deterministic model would solve the same equation also in polynomial time. P vs NP is the question whether these two models are identical

History of P vs NP

- Started in 1928 when David Hilbert proposed a challenge called Entscheidungs problem
- 1936 Alonzo Church and Alan Turing published papers known as Church-Turing thesis (“Every effectively calculable function is a computable function.”)
- 1965 Alan Cobham published paper (The intrinsic computational difficulty of functions). Cobham-Edmond Thesis: reasonable model of computation can be simulated on any other model with a method that is polynomial in input size
- 1972 Richard Karp published famous paper “Reducibility Among Combinatorial Problems”

Facts From Experts

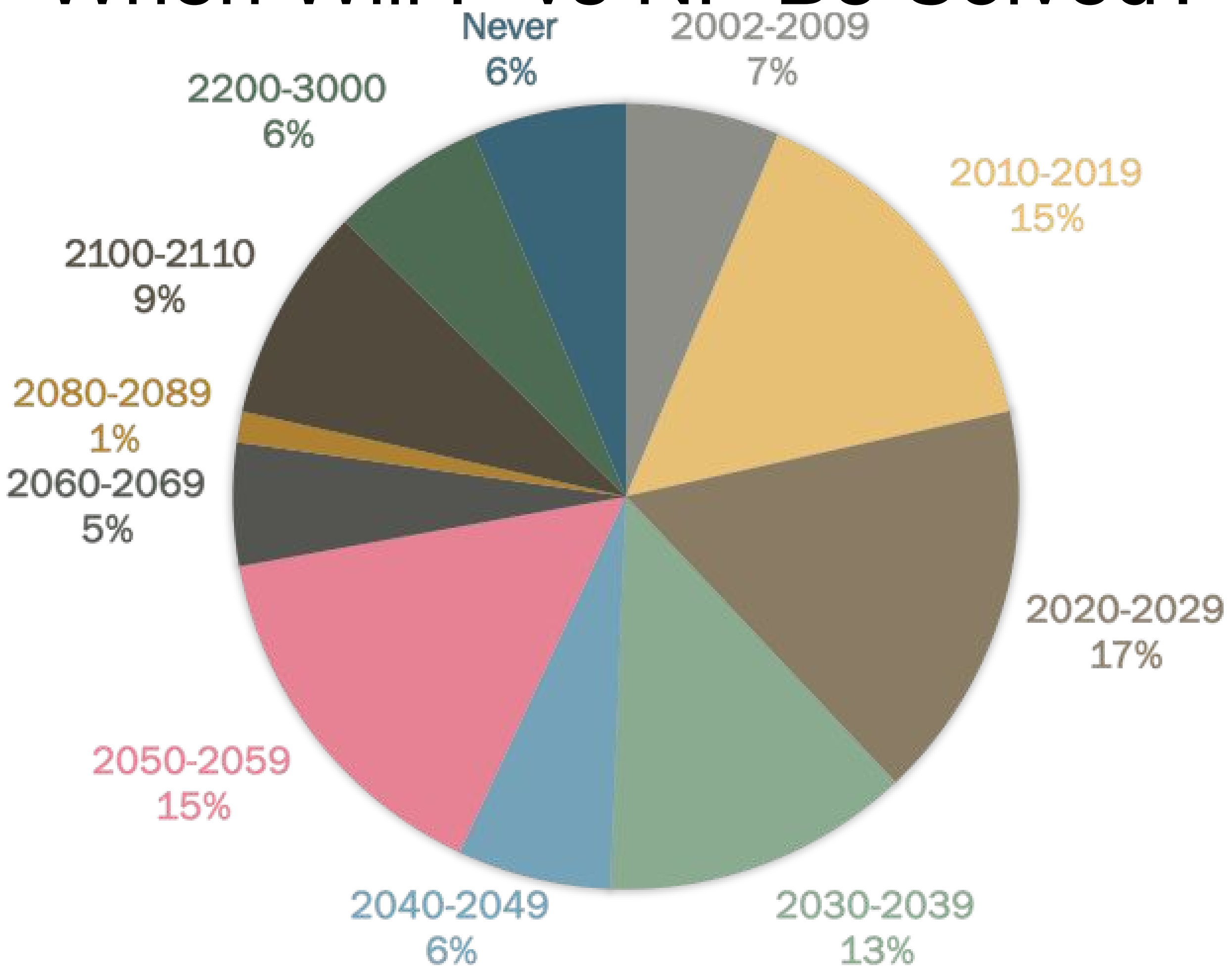
- Poll completed in 1996 and 1997
- 100 experts from various fields questioned about this problem
- Appeared in SIGACT News Complexity Theory

What Will the Verdict Be?

In 1996-1997 100 experts were polled for their opinions on P ?= NP:

- *61 thought P ≠ NP*
- *9 thought P = NP*
- *4 thought P is independent of NP*
- *3 stated it is NOT independent of Primitive Recursive Arithmetic*
- *1 said it depended on the model*
- *22 offered no opinion*

When Will P vs NP Be Solved?



What Techniques Will Be Used?

- Combinatorics and Complexity
- Logic
- Math
- Misc.
    - Computer assisted, non-constructive
    - New techniques
    - 36 people said technique is known

Additional Comments

- 13 people stated the solution would be hard
- 5 people thought the solution would be easy to follow
- 4 said the problem will be irrelevant due to large constraints/degree
- 2 fear a nonconstructive proof of P = NP
- 2 said question becomes irrelevant after quantum computing

Contributions and “Proofs:” P vs NP

- 115 “Proofs” of P vs NP
  - *62 Equal*
  - *50 $P \neq NP$*
  - *2 Unprovable*
  - *1 Undecidable*
- One paper appeared in peer-reviewed journal
  - *Mihalis Yannakakis (1988) proved using a symmetric linear program to express the traveling salesman requires exponential size*

Possible Outcomes of P vs NP

- $P = NP$
- $P \neq NP$
- Independent/Undecidable
- Unprovable
- $P \sim NP$

"If P=NP, then the world would be a profoundly different place than we usually assume it to be. There would be no special value in “creative leaps,” no fundamental gap between solving a problem and recognizing the solution once it’s found. Everyone who could appreciate a symphony would be Mozart; everyone who could follow a step-by-step argument would be Gauss; everyone who could recognize a good investment strategy would be Warren Buffett.“

-Scott Aaronson, MIT

How Would We Prove P ?= NP?

- To prove P = NP
  - *Give an algorithm that solves any NP complete problem in poly-time*
- To prove P ≠ NP
  - *Prove that no such algorithm exists for any NP complete problem*
  - *Much harder*
  - *Most computer scientists think P ≠ NP*
- Other computer scientists that do believe P = NP think the proof would be nonconstructive

What Happens if P = NP?

- RSA encryption would no longer be effective
- Leaps in Artificial Intelligent systems
- Programming would be greatly simplified
    - *Less code writing*
- Online transactions wouldn't be secure
- Mathematical leaps
    - *Proofs could be automatically generated/verified*

Attempts to Solve P = NP Using Clique

- 1996 Anatoly Plotnikov
  - *Polynomial-Time Partition of a Graph into Cliques*
  - Published in SouthWest Journal of Pure and Applied Mathematics
- 1997 Tang Pushan
  - *A polynomial algorithm for CLIQUE problem*
  - Claimed to have found an algorithm with polynomial time complexity for finding clique in a graph
- 2008 Zohreh O. Akbari
  - *A Deterministic Polynomial-time Algorithm for the Clique Problem and the Equality of P and NP Complexity Classes*

Other Attempts to Prove P = NP

- Hamiltonian Path
- 3-SAT
  - 2009 - Narendra S. Chaudhari ➜ O(n^13)
  - 2009 - Luigi Salemi ➜ O(n^11)
- 2010 - Changlin Wan
  - Crux - recursive definition of a Turing machine

Why Believe P = NP ?

- $a^b \equiv c\ (p)$
- Primality Testing ϵ P
  - *Fermat's Little Theorem:*
    - 0 < a < p, a ϵ N
    - $a^{(p-1)} \equiv 1\ (p)$
  - *Wilson's Theorem*
    - n is prime **iff** $(n-1)! \equiv -1 \pmod{n}$

What Happens if P ≠ NP?

- Wouldn’t be a huge impact; most people believe this to be true anyways
- People would stop looking for poly-time solutions to NP-Hard problems
- Proof could lead to helpful insights

Why People Believe P ≠ NP

- Intuition - with all that P = NP would change, it just doesn't seem likely
- Computer scientists would have been able to solve it by now
- P = NP would imply a universal method
    - All NP complete problems are related but not equal

Attempts to solve $P \neq NP$

- Vinay Deolalikar published over an 100 page proof to prove P is not equal to NP
- Neil immerman proposed that there were flaws.

Failed Attempts to Prove P ≠ NP

- Circuit Complexity
- Quantum Computing
- Handicapping

Circuit Complexity

- Main Idea:
  - *NP-Complete problems cannot be solved by small circuits of AND, OR, and NOT gates*
  - *Small: some bound given by a fixed polynomial dependent on input size*

Circuit Complexity

- Successes:
  - *1985 Razborov showed NP-Complete problem of finding a large clique does not has small circuits if only AND and OR gates are considered*
  - *Extending these results to general circuits would prove P ≠ NP*
- Challenges
  - *Razborov later showed his technique cannot be extended to include NOT gates*
  - *Razborov and Rudich gave evidence that circuit complexity cannot be pushed much farther*

Quantum Computing

- Peter Shor mid 1990s factored using hypothetical quantum computer
  - *Very reliant on algebraic structures of numbers*
  - *Not seen in NP-Complete problems*
  - *Cannot apply algorithm to generic search problems*

Handicapping

- Main idea:
  - *Computers have a lot of computational ability*
  - *Handicap the computer and see if anything can be proved*
  - *If needed, handicap the computer further*
  - *May give insight into why searching is necessary*

Exercise:

Talk with your table to come up with some examples of handicapping in computer science and in life!

Current Approach: P ≠ NP

- Ketan Mulmuley and Milind Sohoni using Algebraic Geometry and dubbed Geometric Complexity Theory (GCT)
- Avoids previous problems, very complex
- Use high dimensional polygons that are based on group representations
- Would show Hamiltonian path has size at least $n^{\log(n)}$
- 3 complicated claims
- Mulmuley conjectures ~ 100 years

Thank You!

Questions?

DO YOUR COURSE EVALS (PLEASE)!

Sources

1. https://www.youtube.com/watch?v=msp2y_Y5MLE&t=219s
2. https://www.cs.umd.edu/~gasarch/papers/poll.pdf
3. https://rjlipton.wordpress.com/2009/09/18/why-believe-that-pnp-is-impossible/
4. http://www.win.tue.nl/~gwoegi/P-versus-NP.htm
5. http://cacm.acm.org/magazines/2009/9/38904-the-status-of-the-p-versus-np-problem/fulltext
6. https://primes.utm.edu/prove/prove4_3.html
7. http://people.cs.uchicago.edu/~fortnow/papers/pnp-cacm.pdf
8. http://cgi.csc.liv.ac.uk/~ped/teachadmin/COMP202/annotated_np.html
9. https://www.quantamagazine.org/20151214-graph-isomorphism-algorithm/
10. http://michaelnielsen.org/polymath1/index.php?title=Random_k-SAT
11. https://www.win.tue.nl/~gwoegi/P-versus-NP/Deolalikar.pdf
12. https://rjlipton.wordpress.com/2010/08/12/fatal-flaws-in-deolalikars-proof/
13. https://www.win.tue.nl/~gwoegi/P-versus-NP/sipser.pdf
14. http://www.ijser.org/researchpaper%5CHistory-of-P-versus-NP-Problem.pdf

Supplementary Material

Poly-time Algorithm —Primality

Theorem: Suppose that a and p are relatively prime integers with $p > 1$. p is prime if and only if

$$(x-a)^p = (x^p-a) \pmod{p}$$

Polytime Algorithm -- Primality

Input: Integer $n > 1$

if (n is has the form a^b with $b > 1$) then output COMPOSITE

$r := 2$

while ($r < n$) {

if (gcd(n,r) is not 1) then output COMPOSITE

if (r is prime greater than 2) then {

let q be the largest factor of r-1

if ($q > 4\text{sqrt}(r)\log n$) and ($n^{(r-1)/q}$ is not 1 (mod r)) then break

}

$r := r+1$

}

for $a = 1$ to 2sqrt(r)log n {

if ($(x-a)^n$ is not (x^n-a) (mod x^r-1,n)) then output COMPOSITE

}

output PRIME;

Interesting Problem: Graph Isomorphism (2015)

- László Babai (November 2015)
- Easier than NP
- Harder than P
- Moved problem closer to P
- Zero-Knowledge Proof
    - Blind Testing

NP-Complete Problems

- Graph Theory
  - Graph 3-Colorability
  - Longest Path