IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 24, Issue 3, Ser. III (May. –June. 2022), PP 51-55
www.iosrjournals.org

Problem Classes: P Vs NP

Mr. Rupak Kumar Gogoi[1],Mr. Abinash Borah[2] , Ms Chandrani Borah[3]
Assistant Professor,
Jorhat Engineering College, Jorhat

Abstract:*This paper reviews recent developments made to solve the P* ***vs.*** *NP problem. Consider different ways to find a solution to this problem. This paper covers the complexity oftheevidence and many other aspects that shedlightonpreviouswork in thisarea. A deep and thorough analysis of different works by different authors around the world can conclude that the problem hasnotyetbeenresolved, but there is stillplenty of roomfor further investigation.*

Keywords:*Clique, cloning, brute force.*

Date of Submission: 11-06-2022 Date of Acceptance: 27-06-2022

IINTRODUCTION

As we use more computer power and cleverer methods to address more and more complicated problems, the ones we can't solve begin to stand out.

The notion of NP-completeness aids our understanding of these restrictions, and the P versus NP dilemmas begin to loom large not only as a fascinating theoretical subject in computer science, but as a fundamental principle that pervades all fields.

While we don't expect the P versus NP problem to be solved anytime soon, it has sparked research in a variety of areas to help us better understand, handle, and even exploit the hardness of certain computer challenges.

We'll look at how numerous studies have attempted to answer the P vs NP dilemma, as well as how this question has shaped so much study in computer science and beyond.We'll look at different techniques to dealing with NP-complete problems, as well as the theory that has emerged from them.We illustrate how the constraints of approximation algorithms were caused by a new sort of interactive proof system.

We investigate whether a quantum computer can tackle problems that are NP-complete.We conclude by describing a new long-term initiative that will use algebraic-geometric techniques to separate P from NP.

IIP VS NP Problem

Assume we have a huge group of students to pair up with in order to work on a presentation.We know which kids are compatible with one another, and we want to place them in two-person groups.We may have almost 300 billion trillion possible pairings if we looked at all conceivable partnerships for just 40 pupils.

However, many similar issues lack an efficient algorithm. What if we wished to divide students into three groups of three, each with a compatible pair of students (Partition Into Triangles)? What if we wanted to locate a huge group of students who are all compatible with one another?

What if we intended to seat the students around a large round table (Hamiltonian Cycle) with no incompatible students seated next to each other? What if we divided the pupils into three groups, each of which contained solely his or her compatibles (3-Coloring)?

P = NP denotes that we can discover an efficient solution to any problem that has an efficiently verifiable solution.

We call the very hardest NP problems which includes:

- ➔ Partition into triangles
- ➔ Clique
- ➔ Hamiltonian Cycle
- ➔ 3-Coloring

NP-complete, which means that if we find an effective solution for one of them, we can find an efficient algorithm for all of them, and indeed for any NP problem.

Computation began to play a prominent role in practically every academic subject, particularly the sciences, as computers became cheaper and more powerful.The more scientists learn about computers, the more

they discover that some problems appear to be computationally challenging.NP-completeness is found in many of these fundamental problems.

IIIWHAT IF P = N P?

To appreciate the significance of the P versus NP problem, consider a world in which P = NP.

We could have P = NP technically but not have practical algorithms for most NP-complete problems.But suppose we do have very fast algorithms for all of these problems.

Many people focus on the negative, claiming that if P = NP, public-key cryptography is impossible.True, but the benefits of P = NP will make the internet look like a footnote in history. Everything will be much more efficient as all NP-complete optimization problems become easy. Transportation in all forms will be optimally scheduled to move people and goods more quickly and cheaply. Manufacturers can improve their production processes to increase speed and reduce waste.

And this is only the beginning.

P = NP would have far-reaching implications in mathematics.

Short fully logical proofs for theorems can be found, but these proofs are usually extremely long.

However, the Occam razor principle can be used to recognise and verify mathematical proofs as they are typically written in journals.We can then find proofs of theorems with reasonably long proofs, say less than 100 pages. A person who proves P = NP would leave the Clay Institute with seven million dollars (actually six, since the Poincare Conjecture appears to be solved).

IVAPPROACHES TO DEMONSTRATION OF P 6= N P

4.1 DIAGONALIZATION

Is it possible to create an NP language L that is expressly designed to cause every polynomial-time algorithm to fail to calculate L correctly on certain input? Using a technique called as diagonalization, Cantor [8] demonstrated that the real numbers are uncountable.Cantor demonstrated how to produce a new real number that was not on the list given a countable list of reals. Turing used a similar strategy to argue that the Halting issue is not computable in his fundamental paper on computation [39]. Complexity theorists used diagonalization in the 1960s to illustrate that if you had more time or memory, you can solve more problems.

Why not divide NP and P using diagonalization?

Diagonalization requires simulation. Also a diagonalization proof would likely relativize, i.e., work even if all machines involved have access to the same additional information.

VCIRCUIT COMPLEXITY

It suffices to show that some NP-complete issue cannot be solved by relatively modest circuits of AND, OR, and NOT gates (the number of gates bounded by a fixed polynomial in the input size) to show P 6= NP.

Saxe and Sipser demonstrated that tiny circuits with a set number of layers of gates cannot solve the parity function. Razborov demonstrated in 1985 that the NP-complete issue of finding a large clique does not have short circuits if just AND and OR gates are allowed (no NOT gates). If Razborov's result is extended to generic circuits, P 6=NP is proven. Razborov later demonstrated that allowing NOT gates would cause his tactics to fail spectacularly.Razborov and Rudich introduce the concept of "natural" proofs and show that our current circuit complexity techniques can't be pushed much farther.In fact, no substantial new circuit lower bounds have been discovered in the last two decades.

VIComplexity Proof

Consider a set of Tautologies, which are Boolean formulas for variables over ANDs, ORs, and NOTs that make true when the variables are set to True or False, for example, the formula

(a AND b) OR (NOT a) OR (NOT b): A literal is a variable or its negation, i.e. x or NOT x. A formula, like the one above, is in Disjunctive Normal Form (DNF) if it is the OR of ANDs of one or more literals.

If a formula is not a tautology, we can easily demonstrate this by displaying a variable assignment that results in false.But, if it were a tautology, we wouldn't anticipate short proofs.

If it were possible to prove that there are no short tautological proofs, P 6= NP would follow. In 1985, Haken [22] showed that tautologies that encode the pigeonhole principle (n + 1 pigeons in n holes means some hole has more than one pigion) do not have short resolution proofs.

Since then, complexity theorists have discovered similar flaws in a variety of other proof systems, including cutting planes, algebraic proof systems based on polynomials, and restricted versions of proofs based on the Frege axioms, which are the fundamental axioms taught in an introductory logic course.

To prove P 6= NP, however, we must demonstrate that tautologies cannot have short proofs in an arbitrary proof system.Even a breakthrough finding that tautologies lack short general Frege proofs would not suffice in distinguishing NP from P.

VIIHandling Hardness

So you've got an NP-complete problem to solve.If, as we believe, P 6= NP, you will never find a general algorithm that will always solve your problem correctly and accurately.However, there are times when you must solve the problem in any way. Not all hope is lost.This section describes some of the tools available for NP-complete problems, as well as how computational complexity theory examines these approaches.When confronted with NP-complete problems in the real world, it is common to need to combine several of these approaches.

VIIIBRUTE FORCE

Computers have become more powerful.A brute force search through all possibilities can solve many moderate-sized problems.The NP-complete traveling salesperson problem can be solved using cutting plane method extensions. Although 3 SAT remains NP-complete, the best algorithms can solve SAT problems with up to 100 variables.It is preferable to search all possibilities for satisfiability on general formulae.Because the running times of all of these algorithms grow exponentially, even a small increase in problem size can kill an efficient algorithm.NP-complete problems cannot be solved using brute force alone.

8.1 PARAMETERIZED COMPLEXITY

In a vertex cover problem, we identify a collection of K central people such that at least one of them is central for every compatible pair of people. We can efficiently assess whether a central set of people exists for tiny K, regardless of the overall number of persons n.Even with a small K, the Clique problem can be difficult to solve.

8.2 APPROXIMATION

We can come up with a good estimate.The distances in the travelling salesperson dilemma can be expressed as Euclidian Distance. Although the problem is still NP-complete, Arora [5] provides an efficient technique that comes very close to the best possible solution.The MAX-CUT problem entails splitting people into two groups in order to maximise the number of persons who are incompatible with each other.Semi-definite programming is used by Goemans and Williamson [18] to produce a division of persons only a.878567 factor of the best possible.

IXAVERAGE-CASE COMPLEXITY AND HEURISTICS

NP-completeness is the study of how algorithms perform with the worst possible inputs.
To address NPcomplete problems that come from their areas' specialised problems, many computer scientists use a variety of heuristics.The SAT formula is given more attention than any other Boolean formula.
The satisfiability of Boolean formula is straightforward and efficient in most natural NP-complete problems.
These SAT solvers can frequently resolve the satisfiability of formulas with a million variables in competition .Levin constructed a distributional variant of the P vs NP problem and developed a theory of efficient algorithms over a specified distribution.Some problems, such as lattice versions of the shortest vector issue or determining the permanent of a matrix, are difficult on average exactly when they are difficult in the worst-case scenario.

XUSING HARDNESS

In Section 3, we saw the lovely world that emerges when P = NP is assumed.
P 6= NP, on the other hand, is expected to hold in a very strong fashion.We can leverage high hardness assumptions to our advantage, especially when developing cryptographic protocols and reducing or even eliminating the use of random bits in probabilistic algorithms.

XICRYPTOGRAPHY

Public-key cryptography is impossible if P = NP. To acquire public-key protocols, we require strong average-case assumptions about the difficulty of factoring or similar issues, not just P 6= NP.
Using hard problems, we can perform a lot more than just public-key cryptography.On-line poker is normally played through a reputable website located in the Caribbean.Is it possible to play poker over the Internet without using a secure server? Not only poker, but any protocol that utilises a trusted party can be replaced with one that doesn't, and the players couldn't cheat or learn anything new beyond what they could accomplish with the trusted party, using the correct cryptographic assumptions.

XIIELIMINATING RANDOMNESS

There were probabilistic procedures for determining if a number was prime, for example.

It's either exceedingly difficult or impossible to create truly independent and uniform random bits.
Instead, computer algorithms produce a series of bits from a seed using pseudorandom generators.
The generators available on most computers operate well most of the time, although they occasionally provide inaccurate outputs, both in theory and in practise.We can theoretically improve pseudorandom generators in two ways: one using cryptography's strong hardness requirements, and the other using worst-case complexity assumptions.I'll concentrate on the second strategy.

XIIIARE QUANTUM COMPUTERS CAPABLE OF SOLVING NPCOMPLETE PROBLEMS?

While there are efficient randomised and non-randomized algorithms for determining whether a number is prime, these algorithms rarely provide us with the factors of a composite number.
Much of modern encryption is based on the notion that there are no efficient methods for factoring or comparable difficulties. So, may quantum computers solve NP-complete problems in the future?
Unlikely.Although Lov Grover discovered a quantum algorithm that works on generic NP problems, it only offers a quadratic speedup, and we have evidence that such techniques will not go any farther. Meanwhile, quantum cryptography, which uses quantum mechanics to create cryptographic protocols that do not rely on hardness assumptions, has had some success in both theory and practise.

XIVFUTURE PERSPECTIVE

Geometric Complexity Theory, or GCT, was proposed by Ketan Mulmuley and Milind Sohoni as a solution to the P vs. NP problem using algebraic geometry.This approach appears to avoid the difficulties mentioned in Section 4, but it requires extensive mathematics that could take years or decades to complete. Although it is sufficient to demonstrate that Pn contains an integral point for all n, Mulmuley and Sohoni argue that this direct approach would be difficult.There are three major steps left under this approach:

Prove that the LP relaxation solves the integer programming problem for Pn in polynomial time, and then find an efficient, simple combinatorial algorithm for the integer programming problem for Pn that always returns yes." Although step (1) is challenging, Mulmuley and Sohoni have provided definite hypotheses based on reasonable mathematical analogies that would solve the problem (1).The path to completing steps (2) and (3), on the other hand, is less clear.Despite these remaining obstacles, solving the conjectures in (1) may provide some insight into the P versus NP problem.

XVCONCLUSION

The P versus NP problem has evolved from a logically interesting problem to perhaps the most fundamental and important mathematical question of our time, with increasing importance as computers become more powerful and widespread. Proving P 6= NP is not the end of the story; it simply shows that NP-complete problems lack efficient algorithms for all inputs, but many questions remain.

Cryptography, for example, would require that a problem like factoring (which is not thought to be NP-complete) be difficult for randomly generated composite numbers. Proving P 6= NP might not be the start of the story either. Weaker separations remain perplexingly di cult, for example showing that Boolean-formula Satisfiability cannot be solved in near-linear time or showing that some problem using a certain amount of memory cannot be solved using roughly the same amount of time. We have yet to fully comprehend the P versus NP problem; we have only begun to peel back the layers of this increasingly complex question.

Perhaps the P versus NP problem will be solved in the near future, but I almost hope not.The P versus NP problem continues to fascinate and perplex the mind, and further investigation of this problem will lead us to even more complexities in that truly mysterious process known as computation.

ACKNOWLEDGMENTS

Prof. Joydeep Kumar Sarmah is to be thanked for many useful discussions and comments.I would also like to take this opportunity to thank all of the department's faculty members for their kind assistance and cooperation throughout the study. Last but not least, I thank my friends for their assistance in completing the research work.

REFERENCES

1 The international SAT http://www.satcompetition.org. competitions web page.
2 S. Aaronson. Is P versus NP formally independent? Bulletin of the European Association for Theoretical Computer Science, 81, Oct. 2003.
3 M. Agrawal, N. Kayal, and N. Saxena. PRIMES is in P. Annals of Mathematics, 160(2):781{793, 2004.
4 D. Applegate, R. Bixby, V. Chvatal, and W. Cook. On the solution of traveling salesman problems.
5 DOCUMENTA MATHEMATICA, Extra Volume ICM III:645{656, 1998.
6 S. Arora. Polynomial time approximation schemes for Euclidean traveling salesman and other geometric problems. Journal of the ACM, 45(5):753{782, Sept. 1998.

7 S. Arora and B. Barak. Complexity Theory: A Modern Approach. Cambridge University Press, Cambridge, 2009.
8 T. Baker, J. Gill, and R. Solovay. Relativizations of the P = NP question. SIAM Journal on Computing, 4(4):431{442, 1975.
9 B. Cipra. This ising model is NP-complete. SIAM News, 33(6), July/August 2000.
10 V. Conitzer and T. Sandholm. New complexity results about Nash equilibria. Games and Economic Behavior, 63(2):621{641, July 2008.
11 S. Cook. The complexity of theorem-proving procedures. In Proceedings of the 3rd ACM Symposium on the Theory of Computing, pages 151{158. ACM, New York, 1971.

Mr. Rupak Kumar Gogoi. "Problem Classes: P Vs NP." *IOSR Journal of Computer Engineering (IOSR-JCE)*, 24(3), 2022, pp. 51-55.