On The Nature, Current Status & Future Of The P vs NP Problem

Ultan Kearns

December 20, 2019

Contents

Chapter 1

Introduction

1.1 P VS NP Problem - A Brief Introduction

The P VS NP problem is a very famous problem in computer science. The problem can basically be described as following: if a computer is able to check the answer to a problem can that computer actually solve said problem?(In polynomial time)[1] P problems can be solved faster by computers than NP problems and are termed as "easy" problems, NP problems are "easy" for a computer to check but are not "easy" for a computer to solve.[1] If P ! = NP(! = means not equal) then problems in NP are harder to compute than to verify this means they could not be solved in polynomial time but could be checked in polynomial time. We will explore this more in later chapters.

1.2 An Explanation of Polynomial Time

Polynomial time is a term that is applied to algorithms if the number of steps to complete said algorithm for a given input is $O(n^k)$ where k is any non-negative integer and where n is the algorithmic complexity of the input to the algorithm [2] Everyday calculations such as all basic arithmetic and calculating digits of π and e are said to be done in polynomial time by a computer[2].

1.3 Analysis of Project

In this project I will compare various different papers by various authors and analyze and review their works. I will also cite various journals and professionals who have extensive experience in this area and with the P VS NP problem.

1.4 Outline of Chapters

- Chapter 1 - Introduction - This chapter introduces the problem and summarizes it
- Chapter 2 - P VS NP history - This chapter discusses the history of P VS NP and will give examples of such a problem. I will also critique select papers I have found which will show evidence both in favour and against P = NP. I used two academic papers which use references from various well-known computer scientists such as Michael Sisper as well as some mathmaticians such as Kurt Gödel.
- Chapter 3 - Why is P VS NP important - An analysis of the importance of this question and the ramifications it could bring to computer science if proven or disproved.
- Chapter 4 - Conclusion - This chapter will discuss the information I have gathered while researching this topic and the future of the P VS NP problem.

Chapter 2

P VS NP - A History & Analysis

2.1 Brute Force Explained

To understand the P VS NP problem we must first understand algorithmic complexity and the undesirability of the brute force methodology in algorithms. Brute force refers to an algorithm which is exhaustive and searches all possibilities before arriving at an answer, this is undesirable for many reasons such as: it has a high computing time, demands more computing power and can take up large amounts of memory when compared to non brute force algorithms. The first "lucid" account, according to Sisper in his paper The History and Status of the P versus NP Question, of brute force search appearing in western literature according to Sisper was by a researcher named Edmonds[3], he explained the problem of brute force search in the algorithmic method. The author also mentions that Von Neumann(famous computer scientist[4]) recognised the undesirability of brute force search in a computer program. There are several earlier papers which the author mentions in the chapter P and NP. He mentions papers by Rabin[5],Hartmanis & Stearns[6],Blum[7] which discussed the proposition of measuring the complexity of an algorithm in the number of steps required to solve it. The attempt to measure algorithmetic complexity gave rise to the P VS NP problem.

2.2 Origins of P VS NP

The P VS NP question originated due to developments in mathematical and electronic progressions in the mid twentieth century[3]. It's now a prominent question in the field of computer science and one of the Millennium Problems[8], these are a list of problems deemed most important by the Clay Mathematical Institute, which is a private operating foundation that aims to disseminate

mathematical knowledge[9]. In the first half of the 20th century mathematicians were working on the formal systems of mathematics[10], this led to appearances of problems in set theory such as Russell's Paradox[11] and Gödel's incompleteness theorem[12]. Such work on formalization led to the growing realization that certain algorithms are unsolvable[3].

2.3 P VS NP Real World Example

To use the example Sisper used in his research paper[3], imagine that you are searching for an object in a wide range of possibilities. Computers could expedite your search but in the case of finding the object in an extremely large space would require exhaustive searching even on the fastest machines. The P VS NP question in this example can be reduced down to the following: "Could there be some method to perform such an exhaustive search without having to resort to brute force?".

2.4 Analysis of Current State of Problem and Research

There has been much debate and speculation from researchers if the P VS NP problem is solvable, I have come across research papers that have claimed to solve the P VS NP problem such as[13] which was written by two students from The Rensselaer AI & Reasoning (RAIR) LabRensselaer Polytechnic Institute (RPI), and papers which claim to prove it is not solvable[14]. I will review both of these papers to analyze the current state of the problem and to critically evaluate the subject matter.

2.4.1 Evidence In Favour of P = NP

According to the paper this is a meta-level argument and not an object proof[13] this means that it's a philosophical arguement which the authours believe to be correct. The authours made the claim that a constructive proof shouldn't be necessary to win the Clay Mathematics $1000000[8], which seems quite suspicious from the outset as it stands to reason that if you had authentic proof you would not have to justify yourself in such a way as the authours do in this paper[13]. The authours make the point that exhibiting a polynomial-time algorithm for one or more of the approximately 1000 NP-complete algorithms would be too taxing and according to the authours unproductive[13]. This does not bode well for the authours as they seem to be trying to convince the reader such a proof is unnecessary and they claim they deserve the reward for solving the problem[13]. The authours discuss Turing Machines and discuss how every Turing Machine able to calculate M_n^i there is a corresponding purely mathematical Turing Machine which is able to carry out the same exact computation,

which they claim is an important point to remember in later sections of the paper. To prove the aforementioned point the authours reference the Steiner Tree Problem which is known to be NP-Complete they reference a book from Garey and Johnson confirming the fact[15] it is also proven in a paper by Alessandro Santuari[16]. The authours proceed to explain the Steiner Tree Problem which is the problem of connecting a number n points on a plane with a graph of a minimal overall length using junction points if deemed necessary[16]. The authours proceed to explain how this NP-complete problem can be done in polynomial time, that is $o(n^k)$ in algorithmic complexity. They proceed to describe a physical experiment[13] which goes as follows:

- Make 2 parallel glass plates
- Insert n pins between the plates which will represent the points
- Dip the structure into a soap solution
- Then the soap film will connect the n pins in a minimum Steiner Graph

The authours reference an earlier experiment by scientists Hiroki Iwamura, Masamichi Akazawa, Yoshihito Amemiya[17]. The authours claim that the limited steps involved show that the NP-Complete problem can be solved in $O(n^k)$ steps where k is a non-negative integer. They go on to say that this will not be enough to change peoples minds and prove once and for all that P does in fact equal NP but they encourage us to read further. The authours use first order logic to prove their reasoning by showing that a Turing Machine should be able to solve the Stein Graph Problem[16] in polynomial time. This logic only works if such premises are true, the authours claim that their logic is valid and will appear so using automated proof checkers, but their premises have yet to be proven, they move onto the "more sophisticated" proof in the next chapter[13] In the more sophisticated proof the authours reason that if a Turing Machine is able to solve the analogue example(see physical experiment above) in polynomial time then the Turing Machine should be able to solve the original problem in polynomial time. The authours prove this by using First Order Logic[18]. The authours claim that if P != NP then digital physics[19] is not correct. They reason that it has to be true that all physical phenomena can be modelled on information processing terms of some kind P != NP implies that hypercomputational processes(Hypercomputation refers to a surpassing the established Turing model)[20] exist in our physical universe. They then go on to write that hypercomputational processes are never physically real.

2.4.2 Evidence Against P = NP

In this section I have chosen to analyze a paper called On The P VS NP Question: A Proof of Inequality[14]. The authour briefly summarizes the paper in the opening paragraph, he claims that he has proven that an NP-Complete problem called the Core Function and has proven that the minimal cost of such

an implementation increases exponentially, since this is an NP-Complete problem, such a proof will undoutedly prove that P does not equal NP. The authour begins the introduction section of this paper by describing the P vs NP problem which I have discussed in the beginning of this paper. He then goes on to explain the problem derived from the SAT[21]. He describes the problem as such "Given a Boolean expression containing only the names of a set of variables (some of which may be complemented), the operators AND, OR and NOT, and parentheses, is there an assignment of TRUE and FALSE values to the variables which makes the entire expression TRUE?"[14]. The authour goes onto show us how the satisfiability problem will be written in this paper.
In the next section the authour discusses The Core Function. There is a diagram of a circuit which is decomposed into two layers, the Compatibility Layer and the Core Layer, this is for the sake of brevity. The first layer will take the input of the triplets of boolean values as well as a sign which is used for the NOT which is the negation operator. The Core layer will take the sum of the triplets T all the way up to $T-1, T; T, T$. I have chosen to skip ahead a few pages and into the proof as the paper is mathematically dense and is very long compared to the previous. The authour reasons that the sum of T prime implicants of The Core Function shows that the circuit he has designed in the paper must employ more than T gates since the number of prime implicants implemented by his circuit increases with n as $3n$ so the cost of the minimal implementation of The Core Function $CF(n)$ increases with n as $3n$. Since The Core Function is an NP-Complete problem it is equivalent to proving P != NP.

2.5 Analysis of Papers Reviewed

2.5.1 The History and Status of the P versus NP Question: An Analysis

In this paper titled: The History and Status of the P versus NP Question[3], Sisper outlines the history of the P VS NP problem. He gives real world examples such as the one referenced earlier in this chapter in the section titled: P VS NP Real World Example. The History and Status of the P versus NP Question gives the reader a complete history of the problem, it's formulation, and it's status at the time of the paper's publishing. The paper is well-referenced and written well and it's author Michael Sisper has gone on to write well received books[22] and is Dean of Science at MIT(Massachusetts Institute of Technology)[23]. Michael Sisper was also inducted as an ACM(Association for Computing Machinary) fellow in 2017[24]. This paper was invaluable to me when I was researching the P vs NP problem.

2.5.2 P = NP: An Analysis

In the rather succinctly titled paper: P = NP[13] two authours set about solving the P vs NP problem by showing that P does in fact equal NP. I found the

paper to be rather intriguing and the logic to be consistent, but I did have some contention with the fact that a lot of the evidence they provided was by use of first order logic. They make a convincing point with their logic which is very hard to debunk and very convincing, but I still find the fact that they did not provide a computational algorithm for an NP-Complete problem rather disheartining as the paper was title P = NP, the title heavily suggests that it would provide such an algorithm for an NP-complete problem. That being said it was a short read(6 pages in total including references) and fairly dense. This paper seemed to be well-referenced as well as written but I do not belive that it conclusively shows that P = NP due to its lack of providing a polynomial algorithm for an NP-complete problem. I will however say, that it has convinced me that P = NP, that being said I will follow this up with a warning that it does not prove so in a fashion that would be accepted by a majority of researchers, that is to say that the proof is as the authours themselves stated a meta argument. If however it did provide a computational algorithm for an NP-complete problem that ran in polynomial time, then the paper would be conclusive evidence of P = NP.

2.5.3 P != NP: An Analysis

In the paper tilted On the P vs NP Question: a proof of inequality the authour tries to prove that P != NP[14] The authour has based this paper on an NP-Complete problem known as the SAT or satisfiability problem[21]. The authour of this paper claims to have derived an NP-Complete problem from the SAT and that it's described by a Boolean function called "Core Function". The paper's aims according to the authour are to prove that the cost of minimal implementation of the core function is NP-Complete problem, if successful the authour will successfully prove that P != NP and give us a definitve answer to this problem. The authour introduces us to the P vs NP problem in his paper which I have already described in the introduction of this paper.

Chapter 3

Importance of P VS NP

If it is conclusively proven that P = NP then there exists an algorithm which can run in polynomial time for every NP-Complete problem that exists, as stated in the introduction of the paper P != NP[14]. Such a proof would revolutionize many fields in computer science, such as AI research and cryptography. The reason why such a question holds so much importance in the fields of computer science and mathematics, is that if solved it offers us proof that many difficult questions across many fields could be solvable in polynomial time. The overall effect this would have cannot be proven but one thing is for certain that if it is proven it could usher in a new era of scientific breakthroughs and offer us many solutions to complex mathematical problems.
If conversely it is disproved, this too will have an effect on a wide range of fields. It will prove once and for all that there is no algorithm that can run in polynomial time for NP-Complete problems which is a saddening thought and would have a drastic effect on many fields of research especially in cryptography, mathematics and computer science.
The P vs NP debate still rages on at the time this paper was written, and may indeed carry on for many decades or centuries after this paper has been written. In all this uncertainty only one thing remains clear, that this problem will entice many researchers, students, scientists and hobbyists for many years to come and its solution may entice many more into STEM fields and into the realm of abstract mathematics and computer science.

Chapter 4

Conclusion

It remains unclear if the P vs NP problem will be solved in the future but what does remain clear is the fact that it will have a massive impact on the field of computer science if proven or disproved. At the time of writing this paper the debate rages on, many have tried and failed to solve this complex and intricate problem. The next generation of scientists, hobbyists, researchers and students have a difficult task ahead of them in the coming years as we try to solve this seemingly insurmountable problem. The vast majority of us will fail but it is better to have tried and failed than to never have tried at all.

Thank you for reading my paper, I wrote this paper to gain an understanding of an intricate and complex question in computer science that has interested me for many years. I hope that this paper offered you a concise and intriguing introduction to the P VS NP problem, its nature and its current status.

Bibliography

[1] L. Hardesty, "Explained: P vs. np." https://news.mit.edu/2009/explainer-pnp.

[2] D. Terr, "Polynomial time." http://mathworld.wolfram.com/PolynomialTime.html.

[3] M. Sisper, "The history and status of the p versus np question." https://www.win.tue.nl/~gwoegi/P-versus-NP/sipser.pdf.

[4] W. Poundstone, "John von neumann - american mathematician." https://www.britannica.com/biography/John-von-Neumann.

[5] M. O. Rabin, "Degree of difficulty of computing a function and a partial ordering of recursive sets." https://www.cs.toronto.edu/~sacook/homepage/rabin_thesis.pdf.

[6] J. H. . R. E. Stearns, "On the computationl complexity of algorithms." https://fi.ort.edu.uy/innovaportal/file/20124/1/60-hartmanis_stearns_complexity_of_algorithms.pdf.

[7] M. Blum, "A machine-independent theory of the complexity of recursive functions." http://port70.net/~nsz/articles/classic/blum_complexity_1976.pdf.

[8] C. M. Institute, "Millennium problems." http://www.claymath.org/millennium-problems.

[9] C. M. Institute, "The millennium prize problems." http://www.claymath.org/millennium-problems/millennium-prize-problems.

[10] M. I. . F. Rabe, "Formalizing foundations of mathematics." https://kwarc.info/people/frabe/Research/IR_foundations_10.pdf.

[11] A. D. Irvine and H. Deutsch, "Russell's paradox." https://plato.stanford.edu/archives/win2016/entries/russell-paradox/, 2016.

[12] P. Raatikainen, "Gödel's incompleteness theorems." https://plato.stanford.edu/archives/fall2018/entries/goedel-incompleteness/, 2018.

[13] S. B. . J. Taylor, "P=np." https://arxiv.org/pdf/cs/0406056.pdf.

[14] S. B. . J. Taylor, "On the p vs np question: A proof of inequality." https://core.ac.uk/reader/76531920.

[15] G. M. . J. D., "Garey m johnson d (1979),computers and intractability, w.h. freeman, new york, ny." https://www.scirp.org/(S(351jmbntvnsjt1aadkposzje))/reference/ReferencesPapers.aspx?ReferenceID=2157543.

[16] A. Santuari, "Steiner tree np-completeness proof." http://profs.sci.univr.it/~rrizzi/classes/Complexity/provette/Santuari/steiner.pdf.

[17] Y. A. Hiroki Iwamura, Masamichi Akazawa, "Single-electron majority logic circuits." https://www.semanticscholar.org/paper/Single-Electron-Majority-Logic-Circuits-Iwamura-Akazawa/62d47fab36d60d0af848091d4e001f98cd65bbd2.

[18] U. lecturer at MIT, "First-order logic." https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-825-techniques-in-artificial-intelligence-sma-5504-fall-2002/lecture-notes/Lecture5FinalPart1Save.pdf.

[19] K. Zuse, "Konrad zuse's rechnender raum(calculating space)." https://www.mathrix.org/zenil/ZuseCalculatingSpace-GermanZenil.pdf.

[20] B. Wells, "Hypercomputation by definition," *Theoretical Computer Science*, vol. 317, pp. 191–207, jun 2004.

[21] J. D. Ullman, "The satisfiability problem." http://infolab.stanford.edu/~ullman/ialc/spr10/slides/pnp2.pdf.

[22] M. Sisper, "Sisper bibliography." https://people.eng.unimelb.edu.au/adrianrp/COMP90054/biblio/Author/SIPSER-M.html.

[23] MIT, "Michael sipser." http://orgchart.mit.edu/dean-school-science.

[24] ACM, "Michael fredric sipser." https://dl.acm.org/author_page.cfm?id=81100333940.