

Literature Review - An Analysis of P vs N

Ultan Kearns

November 28, 2019

Contents

1	Introduction	2
1.1	P VS NP Problem - A Brief Introduction	2
1.2	An Explanation of Polynomial Time	2
1.3	Analysis of Project	2
1.4	Outline of Chapters	3
2	P VS NP - A History & Analysis	4
2.1	Brute Force Explained	4
2.2	Origins of P VS NP	4
2.3	P VS NP Real World Example	5
2.4	Analysis of Current State of Problem and Research	5
2.4.1	P = NP?	5
2.4.2	P != NP?	6
2.5	Analysis of Papers Reviewed	6
3	Importance of P VS NP	8
4	Conclusion	9

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 $P \neq NP$ (! = 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
- 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 disproven
- 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 Sipser in his paper The History and Status of the P versus NP Question, of brute force search appearing in western literature according to Sipser 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 algorithmic 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 Sipser 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 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 argument which the authors believe to be correct. The authors 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 authors do in this paper[13]. The authors 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 authors unproductive[13]. This does not bode well for the authors 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 authors 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[]. The authours claim that if $P \neq NP$ then digital physics[] 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 $P \neq NP$ implies that hypercomputational processes[] exist in our physical universe. They then go on to write that hypercomputational processes are never physically real.

2.4.2 $P \neq NP$?

2.5 Analysis of Papers Reviewed

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 Sipser has gone on to write well received books[] and is Dean of Science at MIT(Massachusetts Institute of Technology)[] . Michael Sipser was also inducted as an ACM(Association for Computing Machinery) fellow in 2017[]. This paper was invaluable to me when I was researching the P vs NP problem.

In the rather succinctly titled paper: $P = NP$ [13] two authors 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 an algorithm for an NP-Complete problem rather disheartening as the paper was titled $P = NP$, 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 believe 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 authors themselves stated a meta argument. If however it did provide an algorithm for an NP-complete problem then the paper would be conclusive evidence of $P = NP$.

In the paper titled

Chapter 3

Importance of P VS NP

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 disproven.

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. Sipser, “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 computational 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\(351jmbntvnsjtlaadkposzje\)\)/reference/ReferencesPapers.aspx?ReferenceID=2157543](https://www.scirp.org/(S(351jmbntvnsjtlaadkposzje))/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>.