History of P versus NP Problem

Mamta

Abstract— In this paper a brief history of famous P versus NP problem is discussed. Does P = NP? emerges as great challenge of science. The research work done by prominent experts of the field and theories given by them are concluded. This paper is prepared for non-specialists so I try to keep it as non technical as possible. Further predictions done by some great mathematicians are also given.

Index Terms— Alan Turing , Alonzo Church, Cobham-Edmonds, Computational complexity, David Hilbert, June Hartmanis , Kurt Godel, Lenoid Levin, NP class, NP-complete, P class, Richard Karp, Richard Stearns, Stephen Cook, Turing machine .

1. INTRODUCTION

The P versus NP problem is a major unsolved problem in computer science and is considered by many to be the most important open problem in the field. It is one of the seven Millennium Prize Problems selected by the Clay Mathematics Institute to carry a US$1,000,000 for the first correct proof. Informally, it asks whether every problem whose solution can be quickly verified by a computer, can also be quickly solved by a computer. According to Computational complexity theory, the class P consists of all those decision problems that can be solved on a deterministic sequential machine in an amount of time that is polynomial in the size of the input; the class NP consists of all those decision problems whose positive solutions can be verified in polynomial time given the right information, or equivalently, whose solution can be found in polynomial time on a non-deterministic machine.

2. HISTORY

It started in 1928, when David Hilbert posed a challenge, named Entscheidungsproblem (German for "decision problem") which asks for an algorithm that takes as input a statement of first-order logic and answers 'Yes' or 'No' according to whether the statement is universally valid, i.e., valid in every structure satisfying the axioms. By the completeness theorem of first-order logic, a statement is universally valid if and only if it can be deduced from the axioms, so the Entscheidungsproblem can also be viewed as asking for an algorithm to decide whether a given statement is provable from the axioms using the rules of logic. The origin of the Entscheidungsproblem goes back to Gottfried Leibniz, who in the seventeenth century, after having constructed a successful mechanical calculating machine, dreamt of building a machine that could manipulate symbols in order to determine the truth values of mathematical statements. He realized that the first step would have to be a clean formal language, and much of his subsequent work was directed towards that goal.

In 1936, Alonzo Church and Alan Turing published independent papers showing that a general solution to the Entscheidungsproblem is impossible. This assumption is now known as the Church-Turing thesis. Alan Turing argued that "whenever there is an effective method for obtaining the values of a mathematical function, the function can be computed by a Turing machine". While according to Church's thesis, "A function of positive integers is effectively calculable only if recursive." If attention is restricted to functions of positive integers then Turing's Thesis and Church's thesis are equivalent, and thus the term Church-Turing thesis was first introduced by Kleene. Thus the thesis can be stated as follows "Every effectively calculable function is a computable function." Since 1966, the Turing award has been given annually by the Association for Computing Machinery for technical or theoretical contributions to the computing community. It is widely considered to be the computing world's highest honour, equivalent to the Nobel Prize.

On 20 March 1956, Kurt Godel in his letter to Neumann wrote about this problem. Few lines from the letter are "One can obviously easily construct a Turing machine, which for every formula F in first order predicate logic and every natural number n, allows one to decide if there is a proof of F of length n (length = number of symbols). Let $\psi(F \cdot n)$ be the number of steps the machine requires for this and let $\varphi(n) = \max(F, \psi(F \cdot n))$. The question is how fast $\varphi(n)$ grows for an optimal machine. One can show that $\varphi(n) \geq k \cdot n$. If there really were a machine with $\varphi(n) \sim k \cdot n$ (or even $\sim k \cdot n^2$), this would have consequences of the greatest importance." However, this letter was discovered only in the 1989 and was publicized when Juris Hartmanis published a translation and commentary.

Alan Cobham's 1965 paper entitled "The intrinsic computational difficulty of functions" is one of the earliest mentions of the concept of the complexity class P, consisting of problems decidable in polynomial time. The Cobham-Edmonds thesis named after Alan Cobham and Jack Edmonds (also known as the extended Church-Turing thesis) states that any reasonable model of computation can be simulated on any other with a slowdown that is at most polynomial in the size of the input.

- *Mamta is currently pursuing master of technology degree program in computer science engineering in Gurgaon Institute of Technology, Gurgaon affiliated to Maharshi Dayanand University,Rohtak, India, PH-7876860777. E-mail:mamta100492@gmail.com*

Cobham theorized that this complexity class was a good way to describe the set of feasibly computable problems. Any problem that cannot be contained in P is not feasible, but if a real-world problem can be solved by an algorithm existing in P, generally such an algorithm will eventually be discovered.

Edmonds was the recipient of the 1985 John von Neuman Theory Prize.

In 1965, June Hartmanis with Richard Stearns published a Turing-award winning paper "On the Computational Complexity of Algorithms." It provides a precise definition of the complexity of an algorithm, defines the concept of a complexity class, and shows that there is an infinite sequence of distinct complexity classes and therefore an infinite sequence of increasingly hard problems. They defined a complexity class as the set of all problems solvable within a specified time bound. Their paper shows that there is an infinite hierarchy of complexity classes (for example, problems for which the fastest algorithm takes a time proportional to n, $n \log n$, n^2, n^3, 2^n, and so on) where a small increase in the time bound enables more problems to be solved. A second paper of Hartmanis with Philip M. Lewis showed that a similar hierarchy exists when the complexity is defined in terms of the amount of memory space required (as a function of input size) to solve the problem on a Turing machine. Changing the simple model of a Turing Machine by separating the "input tape" from the "work tape" allowed sub-linear space-bounded complexity classes to be defined.

In 1967, Manuel Blum developed n axiomatic complexity theory based on his axioms and proved an important result, the so called, speed-up theorem. Given a Blum complexity measure (ψ,φ) and a total computable function f with two parameters, then there exists a total computable predicate g (a Boolean valued computable function) so that for every program for g, there exists a program j for g so that for almost all x

$$F\big(x, \emptyset(x)\big) \leq \emptyset(x)$$

f is called the speedup function. The fact that it may be as fast-growing as desired (as long as it is computable) means that the phenomenon of always having a program of smaller complexity remains even if by "smaller" we mean "significantly smaller" (for instance, quadratically smaller, exponentially smaller).

The field really began to flourish when the US researcher Stephen Cook and working independently, Leonid Levin in the USSR, proved that there exist practically relevant problems that are NP-complete. In the US in 1971, Stephen Cook published his paper "The complexity of theorem proving procedures" in which he formalized the notions of polynomial-time reduction and NP-completeness, and proved the existence of a NP-complete problem by showing that the Boolean satisfiability problem is NP-complete. This theorem was proven independently by Leonid Levin, and has thus been given the name the Cook-Levin theorem. Levin's journal article on this theorem was published in 1973; he had lectured on the ideas in it for some years before that time though complete formal writing of the results took place after Cook's publication. Levin was awarded the Knuth Prize in 2012 for his discovery of NP-completeness and the development of average-case complexity.

In 1972, Richard Karp took this idea a leap forward with his landmark paper, "Reducibility among Combinatorial Problems", in which he showed that 21 diverse combinatorial and graph theoretical problems, each infamous for its computational intractability, are NP-complete. He received Turing award in 1985 for his continuous contribution to the theory of algorithms.

In August 2010, Vinay Deolalikar, who works at the research arm of Hewlett-Packard in Palo Alto, California, believes he has solved the riddle of P vs NP in a move that could transform mankind's use of computers. He claims to have proven that P, which refers to problems whose solutions are easy to find and verify, is not the same as NP. But a clear consensus has emerged that the proof, as it stands, is fatally flawed.

In 1996 and 1997, SIGACT News Complexity Theory Columns 14 and 15 collected various experts' opinions on the future of computational complexity. How Will it Be Resolved?

Out of 100 experts, 61 thought P≠NP; 9 thought P=NP; 4 thought that it is independent, while no particular axiom system was mentioned; 3 just stated that it is not independent of Primitive Recursive Arithmetic; 1 said it would depend on the model and 22 offered no opinion.

3. Conclusion

It is evident that every problem in *P* is also in *NP*. The correctness of the algorithm generating the solution automatically certifies that solution. But we are not sure about the converse. Although the P = NP? problem itself remains open despite a million-dollar prize and a huge amount of dedicated research to solve the problem has led to several new techniques. All known proof techniques, each of which is known to be insufficient to prove that P ≠ NP. But we don't have to lose the hope as R. M. Karp in March 2001 predicted that "the P versus NP problem will be resolved, one way or the other, by a mathematician under 30 using an approach that nobody has thought of yet." While on the other hand S. A. Cook predicted that "Someone will give a sound proof that P is not NP, sometime in the next 20 years."

4. Acknowledgment

Author would like to thank her husband, Pankaj Dhawan who has consistently encouraged writing this article. Author also expresses sincere gratitude to Mr. Sidharth Biswas for his helpful suggestions and constructive criticism of this work.

5. REFERENCES

[1] R.Karp. Reducibility among combinatorial problems, In R. Miller and J. Thatcher, editors, Complexity of Computer Computations, pages 85-103, Plenum Press, 1972.

[2] Hartmanis J. and R.E. Stearns, "On the Computational Complexity of Algorithms," *Transactions of the American Mathematical Society*, Vol.117, Issue 5 (May 1965), pp. 285-306.

[3] L. Levin. Average case complete problems. SIAM Journal on Computing, 15:285-286, 1986.

[4] A. Turing. On computable numbers, with an application to the Etscheidungs problem. Proceedings of the London Mathematical Society, 42:230{265, 1936.

[5] L. Fortnow and S. Homer. A short history of computational complexity. Bulletin of the European Association for Theoretical Computer Science, 80, June 2003. Computational Complexity Column.

[6] J. Edmonds. Paths, trees and owers. Canadian Journal of Mathematics, 17:449{467, 1965.

[7] L. Fortnow and W. Gasarch. Computational complexity. http://weblog.fortnow.com.

[8] S. Cook. The complexity of theorem-proving procedures. In Proceedings of the 3rd ACM Symposium on the Theory of Computing, pages151 {158. ACM, New York, 1971.

[9] A. M. Turing, On computable numbers, with applications to the Entscheidungs problem,Proc. London Math. Soc. (2) 42 (1937), 230-265.

[10] M. Davis, Computability and unsolvability, McGraw-Hill, New York, 1958.

[11] H. Yamada, Real-time computation and recursive functions not real-time computable, IRETrans. EC-11 (1962), 753-760.

[12] J. Myhill, Linear bounded automata, ^ADD Tech. Note 60-165, Rep. No. 60-22, Univ.of Pennsylvania, June, 1960.

[13] R. W. Ritchie, Classes of predictably computable functions, Trans. Amer. Math. Soc.106 (1963), 139-173.

[14] A.N.Chomsky, on certain formal properties of grammars, Information and Control 2(1959), 137-167.

[15] J. Hartmanis and R. E. Steams, Computational complexity of recursive sequences, Proc.

[16] Fifth Annual Sympos. On Switching Theory and Logical Design, Princeton, N. J. 1964.

[17] M. O. Rabin, Real-time computation, Israel J. Math. 1 (1963), 203-211.

[18] Computational Complexity: a conceptual perspective by Oded Goldreich.