The Millennium Problem About Computational Intractability

Celina de Figueiredo[1]

Abstract

The Clay Mathematics Institute has selected seven Millennium Problems to motivate research on important classic questions that have resisted solution over the years. Among them is the major unsolved problem in theoretical computer science: the P versus NP problem, which aims to classify the possible existence of efficient solutions to combinatorial and optimization problems. The main goal is to determine whether there exist problems whose solutions can be quickly verified but require an impractically long time to solve by any direct method.

Keywords Graph theory · Graph algorithms · Theory of computing · Analysis of algorithms and problem complexity

Mathematics Subject Classification 05C85 · 68Q25

1 A Gift to Mathematics from Computer Science

One of the seven Millennium Problems selected by the Clay Mathematics Institute is the P versus NP problem, the major unsolved problem in theoretical computer science: Are there questions whose answer can be quickly checked, but which require an impossibly long time to solve by any direct procedure?

✉ Celina de Figueiredo
celina@cos.ufrj.br

1 Programa de Engenharia de Sistemas e Computação, Instituto Alberto Luiz Coimbra de Pós-Graduação e Pesquisa de Engenharia, Universidade Federal do Rio de Janeiro, Rio de Janeiro, Brazil

Given *n* cities to visit by car, how can one plan a route that visits each city exactly once without repetition? If you provide a specific sequence of the cities, I can easily verify whether it constitutes a valid route. However, there is no known efficient method to find such a valid permutation. Given a map with *n* countries, how can one assign three colors so that no two neighboring countries share the same color? If you give me a colored map, I can easily check that the solution is correct. But no general method is known to find a proper coloring with three colors so easily. In both cases, the solution must be found among a vast number of possible configurations. As *n* increases, the number of possible solutions grows exponentially, making brute-force search computationally infeasible. The P versus NP problem asks whether, in general, a method which avoids brute force exists.

The 2023 ACM Turing award for foundational contributions to the theory of computation for Avi Wigderson follows his 2021 Abel prize, together with László Lovász, for their foundational contributions to theoretical computer science and discrete mathematics, and their leading role in shaping them into central fields of modern mathematics.

The unprecedented recognition given to Avi Wigderson, who now holds both the Abel prize, often referred to as the Nobel prize for Mathematics, and the Turing award, often referred to as the Nobel prize for Computer Science, gives attention to the P versus NP problem, considered by Avi Wigderson "a gift to Mathematics from Computer Science".

1.1 Abel and Turing

The Abel prize is directly modeled after the Nobel prizes and was proposed by the King of Norway since Alfred Nobel had not included Mathematics, but the dissolution of the union between Sweden and Norway in 1905 postponed the prize to 2002, to celebrate the two-hundredth anniversary of the Norwegian mathematician Niels Henrik Abel (1802–1829). The laureates for fundamental contributions to discrete mathematics and theoretical computer science include in 2012 Endre Szemerédi, a coauthor of the Brazilian Yoshiharu Kohayakawa of University of São Paulo [1], and more recently in 2021 László Lovász and Avi Wigderson [7].

Today, it is more difficult to distinguish between pure and applied mathematics, and the careers of those two pioneers of computation theory, who both received prestigious mathematics awards in 2021, exemplify the close relationship between mathematics and computer science. László Lovász (1948, Budapest) grew up a talented child competing at solving hard problems inspired by Paul Erdős, the most prolific mathematician of the modern era. László Lovász focused on the mathematics of discrete objects, interested in basic research as well as in its applications, and worked as a full-time researcher at Microsoft for seven years in between academic positions. Avi Wigderson (1956, Haifa) on the other hand studied Computer Science in Israel and the United States and held various academic positions before moving to the Institute for Advanced Study, Princeton, in 1999, where he is ever since. Avi Wigderson contributed to practically all areas of computer science, in which he attacked any problem with whatever mathematical tools he could find, even from distant fields of study.

The ACM Turing Award, also an annual prize, is given since 1966 by the Association for Computing Machinery (ACM) and is named after the British mathematician Alan Mathison Turing (1912–1954). The laureates in theoretical computer science include in 1974 Donald Knuth, a coauthor of the Brazilian Jayme Szwarcfiter of the Federal University of Rio de Janeiro [15], for his contributions to the analysis of algorithms, in 1982 Stephen Cook for the understanding of the complexity of computation, in 1985 Richard Karp for contributions to the theory of algorithms, polynomial-time computability and NP-completeness, and 1986 Robert Tarjan for the design and analysis of algorithms and data structures. In 2023, the mathematician Avi Wigderson was nominated for harnessing randomness and for reshaping our understanding of the role of randomness in computation.

Avi Wigderson started exploring the relationship between randomness and computers in the 1980s, before the internet existed, attracted to ideas he worked on by intellectual curiosity, rather than how they might be used.

One of the unexpected ways in which Wigderson's ideas are now widely used is on zero-knowledge proofs, interactive protocols used by a prover to convince a skeptical verifier without revealing the proof. Two people want to establish trust without sharing information, and zero-knowledge proofs detail ways of verifying information without revealing the information itself[1].

1.2 Hilbert, Gödel, Turing, and von Neumann

The Clay Institute was inspired by the set of twenty-three problems proposed by the mathematician David Hilbert in Paris in 1900 as a program to drive the progress of mathematics in the twentieth century. The Clay Millennium Prize Problems were announced in Paris in 2000. Unlike Hilbert's problems, the seven prize problems selected by the Clay Institute were already renowned among professional mathematicians, with many actively working towards their resolution. Only one problem selected by the Clay Institute belonged to Hilbert's list: the Riemann hypothesis. Only one problem selected by the Clay Institute has no associated mathematician: the P versus NP problem.

David Hilbert had a two-part dream: everything that is true in Mathematics is provable, and everything that is provable can be automatically computed. In 1931, Kurt Gödel proved his incompleteness theorems about the limits of what if provable, establishing that no matter how hard you try, your set of axioms will always be incomplete, they will not be sufficient to prove all true facts.

In 1936, Alan Turing proved that a general solution of Hilbert's Entscheidungsproblem is impossible, by introducing his Turing machine, which captures the idea of effective procedure and sets the standard of what can be computed. The Turing machine is a universal model: in a single machine you can have as part of the data the program you want to run. The unsolvable problem of determining from a description of an arbitrary computer program and its input whether the program will finish running is known as Turing's halting problem.

---

[1]Avi explains https://www.youtube.com/watch?app=desktop&v=5ovdoxnfFVc

Gödel and Turing proved that Hilbert's dreams were unattainable, but in doing so gave birth to the theoretical foundations of computation and algorithms. In 1956, Kurt Gödel wrote to John von Neumann about the time required on a Turing machine to test whether a formula in the predicate calculus has a proof of length *n*. In the 1940s and 1950s, Alan Turing and John von Neumann played a major role in the early development of computers.

The Turing machine is the standard computer model in computability theory, the necessary formal model of a computer to define the P versus NP problem precisely. Although the model was introduced before physical computers were built, it nevertheless continues to be accepted as the proper computer model for the purpose of defining the notion of computable function.

1.3 Cook, Karp, Knuth, and Wigderson

Stephen Cook and Richard Karp brought order into chaos by formalizing a very natural class of problems. In every undertaking that we embark on, we assume that, when we find what we are looking for, we know that we have found it. This is the very definition of NP: a problem is in NP exactly if you can check whether the solution you got is correct [5].

Alan Turing showed that testing whether an assertion has a proof is algorithmicaly unsolvable. The P versus NP question can be viewed as a more finitary version of the Entscheidungsproblem. Suppose we wish to test whether an assertion has a short proof. Of course this can be solved by a brute force algorithm, but can it be done efficiently?

Stephen Cook's 1971 paper "The complexity of theorem proving procedures" [3], followed by Richard Karp's 1972 paper "Reducibility among combinatorial problems" [13], formalized the complexity of problems through the definition of the classes of problems NP and P. The key complexity concept is the asymptotic behaviour of algorithms, given by a function describing the growth of the number of steps taken by an algorithm as a function of the size of an arbitrary input. The theory of NP-completeness restricts attention to decision problems: those having a yes/no solution.

An algorithm solves a problem in polynomial time if its associated complexity function is a polynomial, which says that its complexity growth is bounded by a fixed power of the size of the input. The class P consists of the problems that admit a polynomial-time algorithm. The class P refers to decision problems that can be solved by a deterministic polynomial-time algorithm that given an arbitrary input after a polynomial number of steps deterministically always returns the same yes/no correct answer.

A non-deterministic algorithm can exhibit different behaviours on different runs. We view a non-deterministic algorithm as being composed of two separate stages, the first being a guessing stage and the second a checking stage. Given a problem input, the first stage guesses some structure. We then provide both input and structure to the checking stage which proceeds to compute in a deterministic manner, either answering yes or no. A non-deterministic algorithm solves a problem in polynomial time if for every input there is some guessed structure that leads the deterministic checking

stage to answer yes in polynomial time. The class NP consists of the problems that admit a polynomial-time non-deterministic algorithm.

The class P is symmetric because having a fast algorithm to determine whether an object possesses a property C implies having a fast algorithm to determine whether it does not.

The class NP is asymmetric because it requires a short certificate to prove that an object has property C, but not necessarily a short certificate to prove that it does not. For the map coloring problem, a proper coloring with three colors is a short certificate but no short certificate for the inexistence of such a coloring in known. For the route problem, a correct permutation of the cities such that from one city we are able to visit the next city without visiting any city twice is also a short certificate but no short certificate for the inexistence of such a correct permutation is known.

The four countries Bolivia, Brazil, Paraguay, and Argentina share borders with each other, which is a short certificate for the inexistence of a proper coloring with three colors for the map of South America. There are no four states in Brazil which share borders with each other, and yet no proper coloring with three colors for the map of Brazil exists. No short certificate for the inexistence of a proper coloring with three colors is known.

The polynomial-time non-deterministic algorithm captures the notion of polynomial-time verifiability. For a given input, a non-deterministic algorithm always produces the same yes or no answer, regardless of the sequence of choices made during its execution. By definition, a deterministic polynomial-time algorithm is a particular case of a non-deterministic polynomial-time algorithm, which implies that the class NP contains the class P. The Millennium Prize problem asks whether NP and P are equal classes of problems.

A polynomial-time reduction is a method for solving one problem using another. Stephen Cook introduced the concept of polynomial-time reduction to compare the complexities of problems within the NP class. He proved that Satisfiability, a problem in logic, is a complete problem by proving that every problem is NP admits a polynomial-time reduction to Satisfiability. This means that if a polynomial-time algorithm is found for Satisfiability, then every problem in the NP class can also be solved in polynomial time. Subsequently, Richard Karp proved that 21 well known problems are also complete in the class NP, by defining for each of the 21 problems a polynomial-time reduction from the complete problem Satisfiability. This means that if a polynomial-time algorithm is found for one of Cook–Karp's complete problems, then P = NP.

In 1974 Donald Knuth [14] asked the community for a suitable name for a problem at least as difficult to solve in polynomial time as those of Cook–Karp class NP. Knuth suggested the names Herculean, Formidable, and Arduous, but the winning write-in vote was NP-hard and NP-complete, put forward by several people at Bell Labs. Soon after, two of them, Michael R. Garey and David S. Johnson, wrote the book "Computers and Intractability – A Guide to the Theory of NP-Completeness" [9].

We now know thousands of problems that we want to solve, in logic, in number theory, in combinatorics, in optimization, and so on, that are equivalent, because they are all proved to be NP-complete. We have these two classes that seem separate, the class P of problems that admit a polynomial-time algorithm, and the class NP-com-

plete of problems that admit a non-deterministic polynomial algorithm and such that every problem in NP is polynomial-time reduced to it. There is an almost clear-cut division: most problems of interest are either in P or are NP-complete. Determining whether these two classes are equal is the essence of the P versus NP question; finding a polynomial-time algorithm for any NP-complete problem would resolve the question.

To mark the 40th anniversary in 2012 of the publication of Karp's paper, where the wide applicability of the NP-completeness concept was established, Johnson wrote the paper "A Brief History of NP-Completeness" [12]. The year 2012 also marked the 100th birthday of Alan Turing, whose Turing machine is the basic model for computation used to define the classes P and NP [11].

A Brazilian perspective on the 40th anniversary in 2019 of the publication of the NP-completeness guide [9], unanimously considered by many in the computational complexity community as its most important book, was given as an homage to David S. Johnson [2].

To mark the 50th anniversary in 2021 of the publication of Cook's paper, Lance Fortnow argued in the paper "Fifty years of P vs. NP and the possibility of the impossible" [8] how advances in algorithms, machine learning, and hardware can help tackle many NP-hard problems once thought impossible. Vijaya Ramachandran in her Millennium Lecture about P versus NP remarks that NP-complete does not mean that the problem is intractable, although it was often seen as a barrier to the existence of efficient algorithms[2]. The theory that proposes the millennium problem helps to understand the fundamental computational limitations. A question of theoretical interest explains the practical difficulty of problems formulated throughout the scientific community.

Knuth and Wigderson have opposed views regarding the possible equality of the classes of problems P and NP. On one hand, Wigderson has the intuition that P is not equal to NP, since one may easily check whether a given proof of a theorem is correct and to actually find the proof may be much harder. On the other hand, Knuth has the intuition that P is equal to NP, since there are proofs of existence of algorithms that are not constructive, the proof of existence do not necessarily provide the algorithm. In a recent interview[3], Knuth questions that to actually find an algorithm is not necessarily easier than to prove that the algorithm does not exist.

1.4 Discrete Mathematics, Theoretical Computer Science, and Randomized Algorithms

We can look at Avi Wigderson as following the steps of the computational complexity pioneers. We shall next describe the areas of Mathematics that Wigderson brought to the center of modern mathematics, and how he has revolutionized our understanding of the role of randomness in computation and the way we see the P versus NP problem.

---

[2] Watch https://www.claymath.org/lectures/p-versus-np/

[3] Watch https://www.youtube.com/watch?v=XDTOs8MgQfg

A change has occurred in the attitude of mainstream Mathematics towards the areas of Discrete Mathematics and Theoretical Computer Science. The notions of computational classes of P and NP became central; nowadays the whole of Mathematics can be viewed through effective computation and short proofs of existence.

In the Mathematics Subject Classification[4], the area of Discrete Mathematics appears as 68R, a subarea of Computer Science [16]. The area of Combinatorics appears as 05, a branch of mathematics that plays crucial role in computer science, since digital computers manipulate discrete, finite objects [18]. The subarea of Graph Theory, known as the mathematics of connectivity, appears as 05 C [19]. Combinatorial methods give analytical tools for the analysis of worst-case and expected performance of algorithms.

Concrete Mathematics, according to Graham, Knuth, and Patashnik [10], is a complement to abstract mathematics, a blend of continuous and discrete mathematics.

Theoretical Computer Science studies the power and limitations of computing and has two complementary sub-disciplines. Algorithm Design, which develops efficient methods for computational problems, and Computational Complexity, which shows limitations on the efficiency of algorithms. Discrete Mathematics and Theoretical Computer Science are allied fields: the study of graphs, strings, and permutations are central to Theoretical Computer Science [4].

Computing technology is made possible by algorithms, understanding the principles of powerful and efficient algorithms deepens our understanding of computer science, and also of the laws of nature since any natural process is an evolution which can be viewed as computation.

Computers are deterministic: the set of instructions of an algorithm applied to its input determines its computation and its output. The world we live in is full of random events that lack predictability or a well-defined pattern. Computer scientists allow algorithms to make random choices to improve their efficiency. A randomized algorithm flips coins to compute a solution that is correct with high probability [6].

Both non-deterministic and randomized algorithms are not deterministic, but in different ways. A non-deterministic algorithm always produces on a given input the same answer yes or no. A randomized algorithm on a given input will answer yes or no with a certain probability, so on one run it might answer yes and on another it might answer no.

Two paradigms of randomized algorithms can be seen through the fastest algorithms for the well-known problems of Sorting and Primality [4]. A Las Vegas randomized algorithm always returns a correct answer but its running time depends on the input. The Quicksort algorithm always returns a sorted sequence, but its time depends on the choice of a pivot.

A Monte Carlo randomized algorithm always returns an answer in the same time, but the answer may be incorrect with a certain small probability. The Primality Test that uses a necessary condition for a number to be prime always returns an answer in the same time, but the answer is definitely correct only if the input fails to satisfy the necessary condition (in that case, we definitely know the input is composite), oth-

---

[4] see https://mathscinet.ams.org/mathscinet/msc/msc2020.html

erwise if the input does satisfy the necessary condition the input may be composite with a small probability.

It is natural to believe that having randomness is much more powerful than not having it, but Avi Wigderson asked how necessary randomness was for efficient problem solving. Wigderson revolutionized our understanding of the role of randomness in computation by establishing that the need for randomness is tied to the computational difficulty of the problem [17]. He established sufficient conditions to ensure that every randomized polynomial-time algorithm can be efficiently derandomized, made fully deterministic, establishing a trade-off between hardness versus randomness: If there exists a hard enough problem, then randomness can be simulated by efficient deterministic algorithms; conversely, efficient deterministic algorithms even for specific problems with known randomized algorithms would imply that such a hard problem must exist.

1.5 Are You a Mathematician or a Computer Scientist?

Soon after Avi Wigderson was named in April 2024 the recipient of the 2023 ACM Turing Award, he was interviewed[5] by David Nirenberg, the director of the Institute for Advanced Study, Princeton, where Wigderson works at the School of Mathematics. The director holds Wigderson's book "Mathematics and Computation: A Theory Revolutionizing Technology and Science" [20], an introduction to computational complexity theory, the mathematical study of efficient computation that has evolved into a highly interdisciplinary field, its connections and interactions with mathematics, and its central role in the natural and social sciences, technology, and philosophy.

After congratulating his colleague, the director asks Wigderson, who now holds the highest recognitions for lifetime achievement in Mathematics and in Computer Science: "Are you a mathematician or a computer scientist?". Wigderson answers that he is both a mathematician and a computer scientist, since he studies the mathematical foundations of computing, he proves theorems to understand computation and computational processes also in nature. The director then asks: "After winning the Abel prize, which is considered the Nobel prize for Mathematics, and the Turing award, which is considered the Nobel prize for Computer Science, which natural science do you think is more amenable to give the Nobel prize for applications of innovations of computing?". Wigderson answers that it is conceivable that a Nobel prize goes to innovations of computing, since every evolution process in nature can be studied as a computational process[6].

Avi Wigderson feels lucky to be part of such a dynamic community, and I feel lucky to have had the opportunity given by the Bienal de Matemática 2024 to speak about his favorite problem, in the year his contributions have attracted so much attention, thanks to the prizes he has recently won.

---

[5] Watch https://www.youtube.com/watch?v=TK_vD-VnsFw

[6] Geoffrey Hinton, the Godfather of AI, was named, together with John J. Hopfield in October 2024, the recipient of the Nobel Prize in Physics https://www.nobelprize.org/all-nobel-prizes-2024/

References

1. Alon, N., Capalbo, M., Kohayakawa, Y., Rodl, V., Rucinski, A., Szemerédi, E.: Universality and tolerance. In Proceedings 41st Annual Symposium on Foundations of Computer Science. IEEE, (2000)
2. Buriol, L.S., de Figueiredo, C.M.H., Resende, M.G.C., Uchoa, E.: The guide to NP-completeness is 40 years old: an homage to David S. Johnson. Pesquisa Operacional **40**, e236329 (2020)
3. Cook, S.: The complexity of theorem proving procedures. In Proceedings of the Third Annual ACM Symposium on Theory of Computing, pages 151–158, (1971)
4. Cormen, T.H., Leiserson, C.E., Rivest, R., Stein, C.: Introduction to Algorithms. MIT Press, (2009)
5. de Figueiredo, C.M.H.: Resolver ou verificar? uma pergunta que vale um milhão de dólares. Ciência Hoje **287**, 42–46 (2011)
6. de Figueiredo, C.M.H., Lemos, M.J.M.S., da Fonseca, G.D., de Sá, V.G.P.: Introdução aos Algoritmos Randomizados. XXVI Colóquio Brasileiro de Matemática, IMPA (2007)
7. Dundas, B.I., Skau, C.F.: Abel interview 2021: László Lovász and Avi Wigderson. Not. Am. Math. Soc. **69**(5), 828–843 (2022)
8. Fortnow, L.: Fifty years of P vs. NP and the possibility of the impossible. Commun. ACM **65**, 76–85 (2021)
9. Garey, M.R., Johnson, D.S.: Computers and Intractability. A Guide to the Theory of NP-completeness, WH Freeman (1979)
10. Graham, R.L., Knuth, D.E., Patashnik, O.: Concrete Mathematics: A Foundation for Computer Science. Addison-Wesley, (1994)
11. Haeusler, E.H.: A celebration of Alan Turing's achievements in the year of his centenary. International Transactions in Operational Research **19**, 487–491 (2012)
12. Johnson, D.S.: A brief history of NP-completeness, 1954–2012. In Extra Volume: Optimization Stories, pages 359–376. Documenta Mathematica, (2012)
13. Karp, R.M.: Reducibility among combinatorial problems. In R. E. Miller and J. W. Thatcher, editors, Complexity of Computer Computations, pages 85–103. Plenum, (1972)
14. Knuth, D.E.: A terminological proposal. SIGACT News **6**, 12–18 (1974)
15. Knuth, D.E., Szwarcfiter, J.L.: A structured program to generate all topological sorting arrangements. Inf. Process. Lett. **19**, 153–157 (1974)
16. Lovász, L., Pelikán, J., Vesztergombi, K.: Discrete Mathematics: Elementary and Beyond. Springer, (2003)
17. Nisan, N., Wigderson, A.: Hardness vs randomness. J. Comput. Syst. Sci. **49**(2), 149–167 (1994)
18. Polya, G., Tarjan, R.E., Woods, D.R.: Notes on Introductory Combinatorics. Birkhäuser, (2014)
19. Szwarcfiter, J.L.: Teoria Computacional de Grafos: os Algoritmos. Elsevier, (2018)
20. Wigderson, A.: Mathematics and Computation: A Theory Revolutionizing Technology and Science. Princeton University Press, (2019)