Understanding of P vs NP Problem: With Extension of the Continuum Hypothesis

Jihyeon Yoon*
(Mabangjin Software)

August 8, 2022

Abstract

P vs NP problem turns out to be P=NP with an extended context of Continuum Hypothesis and Category Theory.

1 Introduction

P vs NP problem is a long disputed problem of the computational science.[1][2] And I will show that it's P=NP in an extended context of Continuum Hypothesis and Category Theory.[3][4]

2 Continuum Hypothesis

The continuum hypothesis is a concentration representation between the following infinitives.[3]

$$2^{\aleph_n} = \aleph_{n+1}$$

Here, when n=0, a transformation occurs between the countable and uncountable infinity. Interpreter is for running programs based on specific source code. It's a concept that contrasts with compilers, but in fact, compilers are essentially more detailed interpreters that optimizes the source code in a minimal way.[5]

Interpreter runs the source code by the following procedure.

1. Interpreter creates the given source code first into a logical binary tree structure.
2. Interpreter cycles the binary tree structure contents in order to derive the execution result.

In a sense that every algorithm could be written in form of binary tree, every kind of binary tree could result in form of uncountable infinity.[6]

3 Category Theory

Next, it can be extended to the Continuum Hypothesis based on Category Theory[7]. As discussed earlier, the countable infinity is a binary tree corresponding to the binary code structure of the interpreter. And as a consequent, the interpreter's execution of a binary tree is a result of logical values. At this point, intuition through logic circuits can be derived: there are infinite uncountable number of logic circuits for the same outcomes with the same incomes, but there could be only one (or at least,

*Jihyeon Yoon is a korean medicine doctor. And he is a freelancer programmer also.
E-mail: flyingtext@nate.com
Yeongdeungpo-gu, Seoul, 07429,
Seoul, South Korea
ORCiD: https://orcid.org/0000-0001-9610-0994

countable) shortest optimized logical structure exists. (It's because all logical circuit structures could be reduced with two elements: NAND or NOR.[8])

$$
\begin{aligned}
&In\ Category\ \forall C,\\
&\exists(\ \cdot\ )\ = TRUE \Rightarrow\ NAND(\ \cdot\ ) = NAND(TRUE) = FALSE\\
&\exists(\ \cdot\ )\ = FALSE \Rightarrow\ NAND(\ \cdot\ ) = NAND(FALSE) = TRUE\\
&such\ that\ if\ NAND(\ \cdot\ ) = \neg(\ \cdot\ ) \Rightarrow (Evident\ Proposition) = TRUE
\end{aligned}
$$

1. All logical circuit structures can be reduced into two elements: NAND or NOR in classical computing. And they can be transformed into CNOT in quantum computing.[9] As a result, all logical circuit structures can be abbreviated to the shortest optimized single(or at least, countable) structure.

2. Additionally, despite the existence of the most optimized logical circuit structure, it is impossible to know the execution result value before execution. This is associated with Halting Problem.[10] One of the main arguments for Halting Problem is, "It's impossible to determine which Turing machine accepts what is self-evident or not."

To be more intuitive, the logical expression extended from "countable infinity" to "uncountable infinity" can be abbreviated to the most optimized structure in forms of NAND and NOR, and the optimizations to be revealed in classical computing are still remaining in questions. And in reality, the optimized structure in quantum computing in forms of CNOT can be evaluated by a quantum computer in polynomial running time. As a result, P vs NP problem turns out to be P=NP in quantum computing but remains questions of optimization in classical computing.

References

[1] S. Cook, "The P versus NP problem," *Clay Mathematics Institute*, vol. 2, 2000, publisher: Citeseer.

[2] L. Fortnow, "Fifty years of P vs. NP and the possibility of the impossible," *Communications of the ACM*, vol. 65, no. 1, pp. 76–85, Jan. 2022, publisher: Association for Computing Machinery (ACM). [Online]. Available: https://doi.org/10.1145/3460351

[3] G. H. Moore, "Early History of the Generalized Continuum Hypothesis: 1878–1938," *Bulletin of Symbolic Logic*, vol. 17, no. 4, pp. 489–532, Dec. 2011, publisher: Cambridge University Press. [Online]. Available: https://www.cambridge.org/core/journals/bulletin-of-symbolic-logic/article/abs/early-history-of-the-generalized-continuum-hypothesis-18781938/CFA1336806314CDC313F23B68FBD3C19

[4] B. J. Copeland, *The Essential Turing*. Clarendon Press, Sep. 2004, google-Books-ID: VlC5MkVIwqkC.

[5] T. H. Romer, D. Lee, G. M. Voelker, A. Wolman, W. A. Wong, J.-L. Baer, B. N. Bershad, and H. M. Levy, "The structure and performance of interpreters," in *Proceedings of the seventh international conference on Architectural support for programming languages and operating systems*, ser. ASPLOS VII. New York, NY, USA: Association for Computing Machinery, 1996, pp. 150–159. [Online]. Available: https://doi.org/10.1145/237090.237175

[6] J. Jones, "Abstract syntax tree implementation idioms," in *Proceedings of the 10th conference on pattern languages of programs (plop2003)*, 2003, pp. 1–10.

[7] J.-P. Marquis, "Category Theory," in *The Stanford Encyclopedia of Philosophy*, fall 2021 ed., E. N. Zalta, Ed. Metaphysics Research Lab, Stanford University, 2021. [Online]. Available: https://plato.stanford.edu/archives/fall2021/entries/category-theory/

[8] H. M. Sheffer, "A set of five independent postulates for Boolean algebras, with application to logical constants," *Transactions of the American Mathematical Society*, vol. 14, no. 4, pp. 481–488, 1913. [Online]. Available: https://www.ams.org/tran/1913-014-04/S0002-9947-1913-1500960-1/

[9] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information*, 1st ed. Cambridge ; New York: Cambridge University Press, Nov. 2000.

[10] C. Moore and S. Mertens, *The Nature of Computation*, Aug. 2011. [Online]. Available: https://academic.oup.com/book/25619