Global Scientific JOURNALS

GSJ: Volume 13, Issue 7, July 2025, Online: ISSN 2320-9186

www.globalscientificjournal.com

Theoretical and Formal Proof of "P versus NP" Theorem

Mirzakhmet Syzdykov
Satbayev University, Astana, Kazakhstan
mirzakhmets@icloud.com

ABSTRACT

After making the number of sufficient and successful experiments on account of "P versus NP" theorem, specifically according to the equivalence of complexity classes, we are giving the final and formal theoretical proof by contradiction in this paper summarizing all the results before and giving the definition of our functional hypothesis or conjecture.

Keywords: proof, P versus NP, theorem, complexity, class.

INTRODUCTION

The long-standing question of the relation between intractable or non-polynomial classes of complexity has gained attention more than fifty years ago, since it was first stated by Cook [1], in the letter by John Forbes Nash – he has addressed it to be one of the important questions in mathematics and computer science.

Since that the unsuccessful attempt was done by number of researchers from the past [2, 3]. We have stated that there could be the set of feasible functions according to which the Cook's equilibrium withholds according to the natural and mathematical notation [4]. We are to address the general breakthrough by seeing the statement of the problem according to which set of tractable polynomial class' problems is actually a subset of non-polynomials [5].

CONJECTURE

Let's define the *NP* as a class of non-polynomial problems and *P* is to be a class of polynomial problems, then there exist the function *f(x)* defined as follows:

$$f(np \in NP) = P \Leftrightarrow O(f(NP)) = O(P).$$

The above conjecture states that according to feasible and existent function *f(x)* both classes are equal, since any NP-complete problem in this class can be solved and approached by applying this function [4]. In the next section we show the proof by contradiction.

PROOF

Let's assume that function *f(x)* doesn't exist and classes "P" and "NP" are unequal:

$$\nexists f(np \in NP) \Rightarrow \nexists f(p \in P) : P \subseteq NP.$$

The above statement is a contradiction as problems in polynomial class are always tractable and feasible, thus, there exist any function which is defined on all its set:

$$\exists f(p \in P) \Leftrightarrow O(P) \in O(f \in F).$$

where *F* is a set of any arbitrary function, since polynomial problems can be solved in any observable amount of time.

Thus, we have that function *f(x)* is defined and exists:

$$\Rightarrow \exists f(NP) \Leftrightarrow O(f(NP)) = O(P) \Rightarrow NP = P.$$

RELATIONS

Let's define the following function *p(x)* - where *p(x)* is the number of square-free forms up to *x*. As we know the following holds true for any argument *x*:

$$p(x) \leq x \wedge p(x) \leq O\left(\sqrt{(x)}\right) \leq \int \sqrt{(x)} = \frac{2}{3} * x^{\frac{3}{2}}.$$

However, the integration above isn’t closed and converges to infinity, thus proving that the number square-free naturals is unbound and exists in infinity.

RE-WRITING ALGORITHM

Thus, we have the following statements for any assertion and extended operators:

(?=<a>)[<b>] = (<a>.) & (<b>),*
(?!<a>)[<b>] = (<b>) - (<a>.).*

Same is true for look-behind assertions:

[<b>](?<=<a>) = (.<a>)&(<b>),*
[<b>](?!<<a>) = (<b>) - (.<a>).*

ALGORITHM

Here we give the definition of the hash-based regular expression matching on deterministic or non-deterministic finite automata: it’s defined within the hash function *h(x)* and the function computed on the finite automaton, which can lead up to the optimal speed-up, the main property of the proposed function is that it gives the whole sum on both sides like the matching pattern and given automaton for regular expression. As before the hashing matching for regular expressions was studied in a more different context, thus our algorithm works in only two steps: a) compute the hash space for accepting states in automaton and hash function for matching string; b) if both are equal, then perform the direct matching – this approach gives a speedup up to the number of unsuccessful matches in input O(n*m). It’s necessary that hash function *h(x)* defined on automaton and pattern to be non-positional. The hashing function *h(x)* is defined as follows:

$$h(a \cdot b) = h(h(a) \times h(b)), h(a+b) = h(a) \cup h(b), h(a \wedge b) = h(a) \cap h(b),$$

$$h(a \in A) = h(a), h(e) = \{0\}, h() = \{\}, h(a * a) = \{0\} \cup \sum a^i, h(a-b) = h(a), h(\neg a) = A.$$

Thus, we get a speed-up for all cases including positive matchings and our algorithm has a minimal time complexity o(n + m) for the alphabet A and worst time complexity O(n*m), which is a great improvement over the practically supervised cases, when the non-matching occurs more often than positive result, for the above rules the hash function is defined as non-positional as:

CONCLUSION

We have given a strict, formal and theoretical proof towards the tractability and solution for any NP-complete problem within the time space and its measurement like big O-notation. The prior results also played the sufficient role, showing that both classes are equal and Cook’s equilibrium function exists.

We have shown the proof of the existence of square-free numbers in infinity.

We have made a step forward in exploring the set algebra for extended regular expressions.

We have also proposed the regular expression matching algorithm using hashing concept which improves the lower bound of matching to the theoretically minimal possible.

REFERENCES

1. Cook, S. (2003). The importance of the P versus NP question. *Journal of the ACM (JACM)*, *50*(1), 27-29.
2. Fortnow, L. (2009). The status of the P versus NP problem. *Communications of the ACM*, *52*(9), 78-86.
3. Sipser, M. (1992, July). The history and status of the P versus NP question. In *Proceedings of the twenty-fourth annual ACM symposium on Theory of computing* (pp. 603-618).
4. Syzdykov, M. (2021). Functional hypothesis of complexity classes. *Advanced technologies and computer science*, (3), 4-9.
5. Louiz, A. (2023). A Proof That the Set of NP-problems is Bigger Than the Set of P-problems by Using a Logical Consideration. *WSEAS Transactions on Computers*, *22*, 159-170.