NP-complete Problems can be Solved and Verified in Polynomial Time

Mirzakhmet Syzdykov
Independent researcher
mirzakhmets@icloud.com

ABSTRACT

In this short paper we give the final resolution of “P versus NP” theorem according to our previous results obtained in the field of automaton implementation for extended regular expressions and generally intersection operator.

Keywords: P versus NP, proof, P = NP.

INTRODUCTION

The statement was given before by Stephen Cook [1] in his seminal work, where there’s a clear definition of oracle with ‘certificate’ and the problem solution itself defined as an automaton.
Prior to this ‘milestone work’, we have worked on narratives of experimental proof and showed that the equivalence function exists [2], this is, however, not a definitive answer with the full composition of the problem according to oracle and problem languages.
Before we have shown that intersection of languages can be simulated on finite automaton with the ‘witness state’ [3-5].

STATEMENT

Let’s define the language proposed by our solution as $L_{solution}(A)$, where A is the alphabet and this language defines the certificates for the oracle.
We are to show that P equals NP and, thus, the problem can be solved and verified in any polynomial time.

PROOF

Let’s assume that P not equals NP and thus, we have:

$$t\{L_{solution}(A)\} \gg |w|^{O(1)}.$$

The above statement means as originally that solution language cannot be defined by any polynomial automaton, however, there exists Aho-Corasick automaton which can encode any of the string of the fixed length |w| within the polynomial complexity.

For our proof we are assuming that the solution automaton is polynomial in size and simulation and, thus, any set of solutions can be found on this suffix tree automaton due to Aho and Corasick.

Thus it follows that complexity classes are equal on case of language $L_{solution}(A)$:

$$P = NP.$$

DISCUSSION

Generally speaking, we have used the ‘postpone’ technique in our proof, showing that the solution set of words is fixed and polynomial, thus, solution can be found in any polynomial time, since the verification of our word is polynomial, then what if we got the set of solutions, now we can construct

the automaton on which the solution can be found by reverse approach from the states which are already verified.
As before we have shown that the function *f(NP) = P* may exist, here we go with the same assumption, so that *g(f(NP)) = P*, even if *f(NP) = NP*, where *g(x)* is an Aho-Corasick automaton function on the set of solutions f(NP). Thus, more theoretical approach is more recommended, rather than pointing onto the question of finding the function *f(x)*.
Here in discussion section, we will give the proof by contradiction assuming *P <> NP*:

$$P \neq NP \Rightarrow f(NP) \neq f(P) \Rightarrow g(f(NP)) \neq [\, g(f(P)) = P \,].$$

The above statement is a contradiction, thus *P = NP* as there exist the second function by which we can simply represent our solution tree as an Aho-Corasick automaton.
We could say more about our previous functional hypothesis, however, there is the time to make concluding remarks as for this time, this is all what could be made to the present time.

CONCLUSION

We have shown the proof by ‘blind technique’ in which we propose the existence of the set of solutions and the corresponding automaton, on which we find this solution by reverse search from the verified finishing states or leafs or Aho-Corasick tree, then going to the starting state from the parent node onward.
Thus, if solution exists for the related language of certificates, then it can be effectively encoded and found by the reverse technique. All these give the proof by not finding exact algorithm, but rather the theoretical approach.
Finishing all the above, we can make an outlet which goes as “we can verify the problem as well as construct the solution decision tree in the same polynomial time asserting that solution exists” – this is the preamble of our proof method.

ACKNOWLEDGMENTS

We express our sympathy towards effective algorithms and their verification as it goes beyond ‘extended operators’ and, thus, gives a more broader sense towards proof of “P versus NP” theorem.

REFERENCES

1. Cook, S. (2003). The importance of the P versus NP question. *Journal of the ACM (JACM), 50*(1), 27-29.

2. Syzdykov, M., & Kardeis, Y. (2025). ON EXPERIMENTAL PROOF OF "P VERSUS NP" THEOREM. *Journal of Mathematics, Mechanics and Computer Science*, *127*(3). https://doi.org/10.26577/JMMCS202512735

3. Syzdykov, M. (2015). Algorithm to Generate DFA for AND-operator in Regular Expression. *International Journal of Computer Applications*, *124*(8).

4. Syzdykov, M. (2017). Deterministic automata for extended regular expressions. *Open Computer Science*, *7*(1), 24-28.

5. Syzdykov, M. (2022). Extended Regular Expressions in Finite Automata Revisited. *ADVANCED TECHNOLOGIES AND COMPUTER SCIENCE*, (1), 4-7.