Proof of Contradiction Towards Hierarchy Theorem of Complexity Classes

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

ABSTRACT

We give the full contradiction proof of hierarchy theorem for complexity classes as later we gained new results which are re-described in this work.

Keywords: contradiction, complexity classes, hierarchy, proof.

INTRODUCTION

The beginning is a foundations of "P versus NP" theorem [1], hierarchy is a subset inclusion [2, 3, 4] of major complexity classes like polynomial (P), non-polynomial (NP) and exponential (EXPTIME/EXPSPACE).

In our prior results [5], we have shown that subset construction for Schneider's canonical forms [6] can be verified and parsed in time *O(1)* and *O(n)* respectively, the solution is a new class of fixed-input automata (FIA), which solves the problem of exponential growth of states in deterministic finite automaton (DFA) when producing it from non-deterministic finite automataon (NFA) by using subset construction.

PROOF

Since from hierarchy theorem we know that considered complexity classes form a strict hierarchy, we proved later that FIA solve the problem in linear time, we, thus, get the contradiction and in our corollary P = EXPTIME, i.e. polynomial complexity class is equal to exponential.

We know that for Schneider's canonical forms we can construct a solution verifier operating in time *O(1)*, which requires linear construction time O(n) to determine the parameter "t", the solution, thus gives the same answer towards the membership problem for the given regular expression:

$$f(r)=\{r(t)='b'\Rightarrow accept, reject\ otherwise\}.$$

Thus, our function for regular expression "r" is either true or false according to the verifying condition.

Let's assume that P is a strict subset of EXPTIME complexity class, then:

$$P\subset EXPTIME\Rightarrow O(f(r))=O(2^{|r|}).$$

However, as we know from our established fact:

$$O(f(r))=O(1+|(r)|)\neq O(2^{|r|})\Rightarrow P\not\subset EXPTIME.$$

The above output is a strict contradiction towards hierarchy theorem and, with some assumption, towards the inequivivalence of complexity classes P and EXPTIME.

For the next, we will show that these complexity classes are equal according to the complexity notation of the solution verifier.

So this follows "as is", let's first assume the following fact:

$$P \subseteq NP \subseteq EXPTIME.$$

From this fact it follows that:

$$O(P) = O(|r|) \leq O(NP) \leq O(EXPTIME) = O(|2^{|r|}|).$$

However, as we have devised the unary and linear solution verifier, it follows that:

$$O(f(P)) = O(|r|) \leq O(f(NP)) \leq O(f(EXPTIME)) = O(1 + |r|) \Rightarrow O(P) = O(NP) = O(EXEXPTIME).$$

The above relation is a contradiction, thus:

$$P = NP = EXPTIME.$$

For our final and non formal proof, we state that if the exponential problem can be solved in linear time and verified in unary, then, it follows that there's no differentiating relation between complexity classes as they collapse towards the singular point of complexity with measure *O(1)*.

CONCLUSION

The obtained results give the indisprovable results for the relation complexity classes as we have disproved the hierarchy theorem which is wrong due to the final relation leading to EXPTIME and EXPSPACE complexity classses.

REFERENCES

1. Cook, S. (2003). The importance of the P versus NP question. *Journal of the ACM (JACM)*, *50*(1), 27-29.
2. Chang, Y. J., & Pettie, S. (2019). A time hierarchy theorem for the LOCAL model. *SIAM Journal on Computing*, *48*(1), 33-69.
3. Kristiansen, L. (2006, June). Complexity-theoretic hierarchies. In *Conference on Computability in Europe* (pp. 279-288). Berlin, Heidelberg: Springer Berlin Heidelberg.
4. Canonne, C. L., & Gur, T. (2018). An adaptivity hierarchy theorem for property testing. *computational complexity*, *27*, 671-716.
5. Syzdykov, M. (2023). On Successful P versus NP within Finite Automata and Regular Expressions. *Advanced Technologies and computer science*, *1*(2), 4-8.
6. Schneider, K., Shabolt, J., & Taylor, J. G. (2004). *Verification of reactive systems: formal methods and algorithms*. Heidelberg: Springer.