Towards a Theory of Algorithmic Proof Complexity*

Albert Atserias†
Universitat Politècnica de Catalunya
Centre de Recerca Matemàtica
Barcelona, Spain

March 2, 2023

Abstract

A possibly unexpected by-product of the mathematical study of the lengths of proofs, as is done in the field of propositional proof complexity, is, I claim, that it may lead to new polynomial-time algorithms. To explain this, I will first recall the origins of proof complexity as a field, and then explain why some of the recent progress in it could lead to some new algorithms.

As is well known, the formulation of the P vs. NP problem has its origins in mathematical logic. Cook's original interest on the satisfiability problem for logic formulas came from its applications to automated theorem proving; this is clearly reflected in the title of his 1971 paper "On the Complexity of Theorem Proving Procedures" [4]. By viewing the satisfiability of a formula and the tautologyhood of its negation as dual concepts, it can be argued that

*Synopsis of Invited Talk at 49th EATCS ICALP 2022, Paris, France, July 4-8, 2022

†Partially funded by Spanish State Research Agency, through the Severo Ochoa and María de Maeztu Program for Centers and Units of Excellence in R&D (CEX2020-001084-M) and Ministerio de Ciencia e Innovación (MICIN) through project PID2019- 109137GB-C22 (PROOFS).

Cook established the grounds of a general theory of duality for any problem in NP. In this general framework, satisfying assignments and formal proofs would play the roles of dual certificates, with the obvious caveat that proofs typically appear to be exponentially longer than their duals. The question whether this empirical observation about the lengths of proofs is a true fact called for a theory of proof complexity on which to build what later came to be known as Cook's Program [5, 6].

The development of Cook's Program since its formulation in the early 1970's led to many insights on the combinatorial intricacies of formal proofs. The strongest results in the area typically take the form of exponential lower bounds on the length of proofs for proof systems of practical use, including Resolution [9], Cutting Planes [13], and a few others [1, 12, 11]. To achieve this, a deep theory that explains what causes a propositional tautology to not have short proofs was developped. As part of this theory we find certain tight semantic characterizations of resource-bounded proofs [14, 2, 10] much in the same way that the completeness theorem of mathematical logic equalizes truth with proof.

In this talk I want to argue that the semantic study of proof complexity for these proof systems could also lead to new insights of algorithmic nature. Perhaps paradoxically, the starting point for this is the recent discovery that the problem of automating their proof-search is NP-hard [3, 8, 7]. In a nutshell, what these NP-hardness reductions show is that every instance of any problem in NP can be efficiently encoded into a propositional formula that is either a tautology that has a short proof, or is indistinguishable in a formal sense from a falsifiable formula. The existence of a short proof clearly comes with a short certificate. The main point is, however, that the indistinguishability from a falsifiable formula can often, but not always, be also certified efficiently, no less than through the tools of the theory that studies which formulas fail to have short proofs.

As applications, I will discuss how this phenomenon could be used, potentially, to develop new polynomial-time algorithms for a couple of well-known combinatorial problems not obviously related to propositional logic.

References

[1] Miklós Ajtai. The complexity of the pigeonhole principle. *Comb.*, 14(4):417–433, 1994. 2

[2] Albert Atserias and Víctor Dalmau. A combinatorial characterization of resolution width. *J. Comput. Syst. Sci.*, 74(3):323–334, 2008. Prelim. in CCC 2003. 2

[3] Albert Atserias and Moritz Müller. Automating resolution is np-hard. *J. ACM*, 67(5):31:1–31:17, 2020. Prelim. in FOCS 2019. 2

[4] Stephen A. Cook. The Complexity of Theorem-Proving Procedures. In *STOC*, pages 151–158. ACM, 1971. 1

[5] Stephen A. Cook and Robert A. Reckhow. On the Lengths of Proofs in the Propositional Calculus (Preliminary Version). In *Proceedings of the 6th Annual ACM Symposium on Theory of Computing, April 30 - May 2, 1974, Seattle, Washington, USA*, pages 135–148. ACM, 1974. 2

[6] Stephen A. Cook and Robert A. Reckhow. The Relative Efficiency of Propositional Proof Systems. *J. Symb. Log.*, 44(1):36–50, 1979. 2

[7] Susanna F. de Rezende, Mika Göös, Jakob Nordström, Toniann Pitassi, Robert Robere, and Dmitry Sokolov. Automating algebraic proof systems is NP-hard. In *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*, pages 209–222. ACM, 2021. 2

[8] Mika Göös, Sajin Koroth, Ian Mertz, and Toniann Pitassi. Automating cutting planes is NP-hard. In *Proccedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, STOC 2020, Chicago, IL, USA, June 22-26, 2020*, pages 68–77. ACM, 2020. 2

[9] Armin Haken. The Intractability of Resolution. *Theor. Comput. Sci.*, 39:297–308, 1985. 2

[10] Tuomas Hakoniemi. *Size bounds for algebraic and semialgebraic proof systems*. PhD thesis, Universitat Politècnica de Catalunya, 2022. 2

[11] Jan Krajícek, Pavel Pudlák, and Alan R. Woods. An Exponenetioal Lower Bound to the Size of Bounded Depth Frege Proofs of the Pigeonhole Principle. *Random Struct. Algorithms*, 7(1):15–40, 1995. 2

[12] Toniann Pitassi, Paul Beame, and Russell Impagliazzo. Exponential Lower Bounds for the Pigeonhole Principle. *Comput. Complex.*, 3:97–140, 1993. 2

[13] Pavel Pudlák. Lower Bounds for Resolution and Cutting Plane Proofs and Monotone Computations. *J. Symb. Log.*, 62(3):981–998, 1997. 2

[14] Pavel Pudlák and Samuel R. Buss. How to Lie Without Being (Easily) Convicted and the Length of Proofs in Propositional Calculus. In *Computer Science Logic, 8th International Workshop, CSL '94, Kazimierz, Poland, September 25-30, 1994, Selected Papers*, volume 933 of *Lecture Notes in Computer Science*, pages 151–162. Springer, 1994. 2