Proof Search and the Structure of Solution Spaces

Albert Atserias, IMTech, Universitat Politècnica de Catalunya, Barcelona

January 9, 2021

This note was published in the "Research focus" section of the Newsletter 01 of the Institut de Matemàtiques (IMTech) of UPC whose function is to publicise the research activities and achievements of its faculty members.

In 2019, Moritz Müller and I resolved a longstanding open question on the computational complexity of proof search. The preliminary version of the article "Automating Resolution is NP-Hard" was the co-winner of the Best Paper Award at the 60th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2019). In its final version, the paper was invited for submission to the Journal of the ACM, and published in 2020 [1]. Why does the proof that *yet another problem is NP-hard* deserve such an important award? In this note I try to explain why. Let me begin with the historical background.

This year is the 50th anniversary of Steve Cook's article "The Complexity of Theorem Proving Procedures" [5]. Cook's seminal work established that the satisfiability problem for propositional logic is NP-complete. For the first time, the generic task of searching a solution within an exponentially big space of candidate solutions was reduced to that of a concrete, well-defined, and combinatorially simple-to-state problem. The importance of Cook's Theorem was quickly recognized. Only one year later, Richard Karp wrote another landmark paper where he established that "a large number of classic unsolved [computational] problems of covering, matching, packing, routing, assignment and sequencing are equivalent". All of them, Karp showed, are equivalent to the problem of recognizing the satisfiable formulas of propositional logic [7].

The fact that hundreds, if not thousands, of NP-complete problems were discovered since then raises an intriguing question. Might any of these equivalent problems admit sufficient structure to allow for efficient search algorithms? Might the beautiful theory of matchings on graphs be useful to find Hamilton cycles? Might the powerful tools of field theory help in solving systems of polynomial equations? In a sense, what the theory of NP-completeness says is that any attempt to give computationally useful structure to the search spaces of such problems is bound to fail. Or at least fail as much as it has failed for the problem of determining truth in the algebra of propositions since its conception by George Boole more than one and a half centuries ago [3].

Proof theory is the branch of mathematical logic that links the syntactic concept of proof with the semantic notion of truth. A formula is unsatisfiable if and only if its negation is a tautology, and tautologies have proofs in any one of the many complete proof systems for propositional logic. Thought this way, the proof theory of propositional logic can be seen as giving structure to the satisfiability problem. Just like an assignment that satisfies a formula is a certificate that a solution exists, a proof of its negation is a certificate that a solution *does not* exist. The trouble is that while truth assignments are always short, proofs could be exponentially longer. But what if we focus on short proofs? Can reasonably-sized proofs be found in reasonable time? This is the question of *automatability* of a proof system.

In his 1971 article, Cook asked to study "the complexity of theorem proving procedures", including Resolution, which is the object of our 2019 article, and the rest of this note.

A formula in conjunctive normal form, a CNF, is a conjunction of a set of clauses, where a clause is a disjunction of variables or negations of variables. If each clause has at most k literals, then we call it a k-CNF. For example, $(p \vee q \vee r) \wedge (\neg p \vee \neg q) \wedge (\neg p \vee \neg r) \wedge (\neg q \vee \neg r)$ is a 3-CNF with three variables and four clauses. Resolution is a proof system to establish the unsatisfiability of CNFs by deriving new clauses from old. The goal is to end up proving the empty, trivially unsatisfiable clause. Such proofs are called *refutations*. The single inference rule of Resolution cannot be simpler: from two clauses of the form $A \vee x$ and $B \vee \neg x$, derive the clause $A \vee B$. It is obvious that if an assignment satisfies the two premises in the inference, then it must also satisfy the conclusion. In particular, if the empty clause can be proved, then the initial formula must be unsatisfiable. It is an entertaining exercise to show that the converse is also true in quantitative terms: if a CNF is unsatisfiable, then the empty clause can be proved in no more than 2^n inference steps, where n is the number of variables. Needless to say, if shorter proofs exist, then those would be preferable since a proof of length 2^n, even for $n = 500$, would not fit in the observable universe.

Unlike the space of satisfying assignments of a formula, which appears unstructured, it is a remarkable fact that the space of *short* Resolution proofs appears to have non-trivial structure. An important result due to Ben-Sasson and Wigderson [2] states that if a 3-CNF formula with n variables has a Resolution refutation of length s, then it also has one in which all its clauses have $O(\sqrt{n \log s})$ many literals. For polynomially small s, the number of such clauses is weakly exponential, i.e., of the form $\exp(O(n^{1/2}(\log n)^{3/2}))$, which means that an algorithm of this run-time exists that finds a proof. In summary: when polynomially short proofs exist, weakly exponentially long ones can be found.

Valid as it is, this result begs the question whether polynomial-size Resolution proofs are common at all. If they were a very rare exception, then the statement that short proofs can be found would be essentially void. With this perspective in mind, it is at least worrying that the early results on the proof complexity of Resolution seemed to indicate that polynomial-size Resolution proofs were indeed an exception. An influential result of Chvátal and Szemerédi, titled "Many Hard Examples for Resolution" [4] shows that, in a well-defined average-case sense, *most* unsatisfiable 3-CNF formulas are indeed exponentially hard for Resolution. Are we then flirting with the void?

One first answer to this is that just like the worst-case model for computational complexity has its weaknesses, any average-case model has its own problems. The formulas that appear in applications are not only not worst-case; even less are they drawn from the probability distribution for which they were shown exponentially hard. The spectacular success of so-called SAT-solvers, software packages that are able to find satisfying assignments or Resolution refutations of many formulas with many variables, illustrates the point. Consider the so-called Boolean Pythagorean Triples Problem: Can every initial segment $1, \ldots, n$ of the natural numbers be partitioned in two in such a way that all triples of the form $a^2 + b^2 = c^2$ are avoided in each part? In a major breakthrough, in 2016 a SAT-solver was able to show that the CNF formula that encodes the problem with n variables is satisfiable if, and only

if, $n \leq 7824$ [6]. This solved a Ramsey theory open problem posed by Graham in the 1980s. Despite being referred to as "Two-hundred-terabyte maths proof is largest ever" [8], it should be noted that, for $n = 7825$, the proof must be able to rule out 2^{7825} many partitions of $1, \ldots, 7825$. A 2×10^{14} bytes long proof is, in comparison, *microscopically* tiny.

A second answer to the question whether short Resolution proofs are common or rare exceptions, can be found in our 2019 result. I claim that this is, indeed, what makes it surprising. Formally, we say that Resolution is *automatable* if there is an algorithm that, given an unsatisfiable formula, finds a Resolution refutation in time polynomial in the length of the shortest refutation. After a long series of works on the theme going back two decades, we finally proved that the problem is NP-hard: Resolution is *not* automatable unless P = NP. Perhaps more importantly, concerning our question on the abundance of short proofs, here is what the result has to offer. As it goes, the NP-hardness proof establishes that every formula F can be efficiently converted into a new one, G, whose polynomially short Resolution refutations *correspond* to the satisfying assignments of F. More precisely, if F is satisfiable, then G is unsatisfiable and has a polynomially short Resolution refutation that can be easily constructed from a satisfying assignment of F. In contrast, when F is unsatisfiable, the technically hardest part of the NP-hardness proof shows that G does not even have weakly exponentially longer Resolution refutations. Among other consequences, this means that short Resolution proofs not only abound; despite exhibiting a certain amount of non-trivial structure, they are able to mirror the search space of solutions of *any* problem in NP.

Only a handful of other problems in NP without polynomial-time algorithms are known whose natural solution spaces are sufficiently structured to allow better-than-exponential algorithms. Perhaps the two most famous examples are the problem of factoring integers into primes, and the graph isomorphism problem. None of those, however, is even expected to be NP-hard. The problem of finding polynomially short Resolution proofs stands out, then, as a notable exception.

References

[1] Albert Atserias and Moritz Müller. Automating Resolution is NP-Hard. *J. ACM*, 67(5):31:1–31:17, 2020. Preliminary version in *Proc. FOCS 2019*.

[2] Eli Ben-Sasson and Avi Wigderson. Short proofs are narrow - resolution made simple. *J. ACM*, 48(2):149–169, 2001. Preliminary version in *Proc. STOC 1999*.

[3] George Boole. *The mathematical analysis of logic: being an essay towards a calculus of deductive reasoning*. Macmillan, Barclay, & Macmillan, Cambridge, 1847.

[4] Vasek Chvátal and Endre Szemerédi. Many Hard Examples for Resolution. *J. ACM*, 35(4):759–768, 1988.

[5] Stephen A. Cook. The complexity of theorem proving procedures. In *Proceedings of the Third Annual ACM Symposium*, pages 151–158, New York, 1971. ACM.

[6] Marijn J. H. Heule, Oliver Kullmann, and Victor W. Marek. Solving and Verifying the Boolean Pythagorean Triples Problem via Cube-and-Conquer. In *Proc. SAT 2016*, volume 9710 of *LNCS*, pages 228–245. Springer, 2016.

[7] Richard M. Karp. Reducibility Among Combinatorial Problems. In *Complexity of Computer Computations*, pages 85–103. Plenum Press, New York, 1972.

[8] Evelyn Lamb. Two-hundred-terabyte maths proof is largest ever. *Nature*, 534:17–18, 2016.