Marijn J. H. Heule
Department of Computer Science
University of Texas at Austin, USA
marijn@cs.utexas.edu

Trip to the Proof

Brute trust

Marijn Heule describes the developments in automated reasoning that led to his solutions of the Boolean Pythagorean Triples problem and the Schur number five problem and to improvements in computing the chromatic number of the plane.

An important trade-off in automated reasoning is efficiency versus correctness. Research and development of fully automatic tools focus primarily on performance, while the interactive theorem proving community deeply cares about the trusted core of their tools. Interactive tools have been successful in constructing a formal proof of famous problems, such as the four color theorem. Fully automatic tools, which are frequently used in industry to find bugs in hardware or software, have become significantly more powerful in the last two decades, thereby allowing to solve long-standing open problems. However, their effectiveness also raised questions whether we can trust these results as computer-generated solutions typically cannot be understood by humans.

My roots lie in highly automated tools for propositional logic, known as satisfiability (SAT) solvers. These solvers determine whether there exists a satisfying assignment (or, equivalently, a solution) for a propositional formula. My interest in correctness originates from my post-doc with Armin Biere at the Johannes Kepler University in Linz, Austria. His solvers have been among the strongest and most reliable ones in the community for over a decade. However, we worried about the correctness of one of the techniques in his top solver. This technique, called blocked clause addition, can remove solutions while ensuring that at least one solution remains (if the initial formula has one). Armin considered the implementation of the technique 'experimentally correct' as he tested the solver on a million small problems without encountering a discrepancy. However, after a few weeks of studying the code, I was able to manually produce a formula with a solution, while his implementation of blocked clause addition would claim there is none.

The bug turned out to be a deep conceptual error. In their quest to further improve performance, developers of state-of-the-art SAT solvers have been adding techniques that go beyond the classical proof system for propositional logic, known as resolution. Examples of such techniques are symmetry breaking, Gaussian elimination, and the earlier mentioned blocked clause addition. The ability to remove solutions makes validation challenging: instead of checking whether no solution is removed (as in resolution), one needs to check whether not all remaining solutions are removed.

In the following years I have been working with various colleagues on new proof systems for propositional logic that allow compact expression of all techniques used in state-of-the-art solvers — including those that cannot be succinctly expressed using resolution. These proof systems reason about the *absence* of facts. We call them *interference-based proof systems*, as learning one fact may block learning another one. In contrast, most proof systems for propositional logic, including resolution, reason about the *presence* of facts.

The design goal of the new proof system was to have a single redundancy criterion that covers all techniques and is computable in polynomial time. This does not imply that checking is cheap since the criterion is more complex and general compared to most reasoning techniques used in SAT solvers. However, we were able to implement a checker that is reasonably efficient. The SAT community started to use the proof system in various settings. For example, participants in the international SAT competitions are required since 2013 to certify that a claim that a problem has no solutions. Such a claim is only considered correct if the certificate, known as a proof of unsatisfiability, can be checked. Also, when Boris Konev and Alexei Lisitsa solved the Erdős discrepancy problem using SAT, they produced and checked a proof of unsatisfiability of their result.

In 2015 I wanted to show that proofs of unsatisfiability are a viable option to show correctness of SAT solving results no matter how many computational resources were required to solve the problem. Initially, I looked at the Schur number five problem. This problem dates back to the early 20th century, when Issai Schur asked whether coloring the positive integers with a finite number of colors would result in a monochromatic solution of the equation $a+b=c$. Schur proved in 1916 that this is indeed the case. Schur number $S(k)$ denotes the largest positive integer such that the numbers $[1,S(k)]$ can be colored with k colors such that there is no monochromatic solution of the equation $a+b=c$. Early on, the first three Schur numbers were determined. However, it took five decades to compute that $S(4)=65$ (by Leonard Baumert in 1965).

Victor Marek from the University of Kentucky has been encouraging me to compute $S(5)$ since the day we met at a workshop in Baltimore back in 2008. Over the years I have been trying to compute this number, but it appeared too hard. Victor suggested to tackle a related challenge: Will any coloring of the positive integers

with two colors result in a monochromatic solution of the equation $a^2+b^2=c^2$? In 1980, Ronald Graham offered an award of $\$100$ for the first person to solve this *Boolean Pythagorean Triples problem*. We teamed up with Oliver Kullmann from Swansea University and determined that the numbers $[1,7824]$ can be colored with two colors while avoiding a monochromatic solution of $a^2+b^2=c^2$, while this is impossible for $[1,7825]$.

The paper about the solution of the Boolean Pythagorean Triples problem was mostly a demonstration that

1. SAT solvers are now able to solve hard problems by linear time speedups even when using thousands of cores; and
2. that we can produce a proof of such hard problems that can be validated by third parties.

Quite unexpectedly, we were contacted by an editor of *Nature* regarding the solution and the proof. In the interview I tried to convince her of the importance of the main contributions. However, she appeared only interested (and worried about) the size of the proof: 200 terabytes. Her article on the 'Largest Math Proof Ever' focussed on the lack of understanding of the computer-generated solution. Moreover, according to the article, the clever algorithms were only 'ticking off possibilities'.

Yet there is no such thing as bad publicity. The aftermath of solving the Boolean Pythagorean Triples problem and the article in *Nature* was very positive. On a personal note, I had the honor of meeting several great mathematicians, including Ronald Graham (who gave me the $\$100$ check), Alfred Hales, Timothy Gowers and Tom Hales. Professionally, it was great that the interactive theorem proving community significantly improved the trust story by developing formally verified checkers of proofs of unsatisfiability. There are now verified checkers in three main theorem provers: ACL2, Coq, and Isabelle.

Meanwhile, I was finally able to solve Schur number five: $S(5)=160$. The Texas Advanced Computing Center made this possible by allowing me to use 2400 CPUs for weeks on end. The size of the resulting proof of unsatisfiability was 2 petabytes, roughly ten times larger than the proof of the Boolean Pythagorean Triples problem. The use of the new proof system was crucial as it allowed to compactly express symmetry breaking. Without it, the proof would have been $120\,(=5!)$ times larger. That would have made it impossible to check the proof, even with the vast number of resources at my disposal. The existence of efficient, formally verified proof checkers also raised the bar for validation. The proof was eventually verified using the ACL2 checker, which required 35 CPU years. Schur number five is arguably the hardest problem ever solved using SAT solvers. We can be confident that this immense proof is correct since it can and has been checked using highly trustworthy systems by independent parties.

Photo: Arjen van Lith

Marijn Heule, with in the background an illustration of the Boolean Pythagorean Triples problem

Recently, an unexpected application for proof checking tools emerged: computing the chromatic number of the plane. This problem asks how many colors are required to color all points of the plane such that no two points at distance 1 from each other have the same color. Early on, two elegant proofs were found to show that the number of colors is at least 4 and at most 7. However, hardly any progress has been made since the 1950's. A breakthrough was announced in April 2018: Aubrey de Grey found a 1581-vertex unit-distance graph with chromatic number 5, thereby improving the lower bound. A Polymath project was started to find a smaller graph with this property. Proof checking tools turned out to be useful here. To validate that a graph has chromatic number 5, one needs to show that there exist no valid 4-coloring. SAT solvers are arguably the fastest method to achieve this. The proof of unsatisfiability showing that no 4-coloring exists can be optimized by proof checking tools. From the optimized proof one can extract a subgraph that has chromatic number 5. This method is more successful than randomly dropping vertices that do not lower the chromatic number. The proof checking tools allowed me to find a unit-distance graph with chromatic number 5 that has only 553 vertices.

Although a graph with 553 vertices is still hard to comprehend for humans, this reduction is substantial and represents a step towards understanding why coloring the plane requires at least 5 colors. In fact, the techniques based on optimizing proofs of unsatisfiability may eventually produce the most elegant argument. This would be an interesting twist in the discussion on the usefulness of mechanized mathematics as computers might be able to give the shortest and clearest proofs of theorems. Some computer-generated proofs may be large, because there exists no short proof.

In the coming years, automated reasoning techniques are poised to solve hard problems that have been open for many decades. Mathematical challenges that may be feasible for an automated approach are Ramsey number five, the Collatz conjecture and the chromatic number of the plane. The proofs may reveal crucial insights that might otherwise be overlooked by mathematicians. However, even if the proofs do not provide any understanding, we can be confident that they are correct, as we have highly trustworthy systems that can validate them.