Solving mathematical problems with SAT

Oliver Kullmann

Computer Science Department
Swansea University

ASL 2017 North American Meeting, Boise, March 2017
Computer-Aided Proofs, March 20, 2017

Harnessing SAT

The power of solving propositional logic, most importantly

- determining a propositional formula to be **satisfiable** or **unsatisfiable**
- resp. to be **falsifiable** or **tautological**,

has increased dramatically over the last two decades.

Especially industrial applications (safety, correctness) are impressive.

Also apparently every modern ATP-solver nowadays uses SAT.

Standard use of SAT in ATP

The typical use of SAT in ATP

is based on **abstraction**:
abstract away the parts of the problem
beyond propositional logic.

This leads to two main characteristic features:

1. **heavy interaction** between the non-propositional parts of the solver and the SAT part;
2. the SAT problems are **relatively small**.

A similar approach one finds in SMT ("SAT modulo theory"), and, based on the computer algebra, in `MathCheck` (Zulkoski, Bright, Heinle, Kotsireas, Czarnecki, and Ganesh [13]).

Heavier use of SAT

In the direction of Herbrand's Theorem, it is also possible to have

SAT solving as the main part of the solving process.

Basically

1. "**all instantiations**" of quantifiers are considered (reducing the problem to infinite propositional logic),
2. and **compactness** is applied (creating infinitely many finite problems).

This results naturally in a much heavier use of SAT.

Now very big SAT problems are to be solved.

Our endeavours are going in this direction.

SAT solving different for easy and hard

Not only are the SAT problems arising in this heavy use of SAT very big,

but they are typically also **very hard**.

The success of SAT solving over the last two decades however

concentrates on problems which are
(very) big, but indeed **(relatively) easy** —
these methods go aggressively for the **"easy success"**,
not for the **worst case**.

So new SAT tools are needed:

integrating aggressive solving and worst-case planning.

This talk

In this talk I

focus on the heaviest use of SAT:
the problem is **directly expresses as a SAT problem**
(for efficiency of encoding).

This then needs also

improved SAT solving.

But that's not everything ...
Since we are PROVING something,

big big **propositional proofs** need to be handled.

Outline

Ramsey theory

- Many problems of Ramsey theory can be understood easily,
- and they yield great problems for SAT.

The fundamental theme is:

> For infinite sets X and certain simple "structures" on X:
> how resilient is this structure against finite partitioning?

A fundamental problem I

Consider

- a finite set $\mathbb{P}$ of polynomials $P(x_1, \ldots, x_k)$ over k variables,
- with integer coefficients.

Let

$$N(\mathbb{P}) := \left\{ \{x_1, \ldots, x_k\} \subset \mathbb{N} : \forall\, P \in \mathbb{P} : P(x_1, \ldots, x_k) = 0 \right\}$$

be the set of (common) zeros (Nullstellen) of $\mathbb{P}$ over **the natural numbers**, where for convenience we don't take the tuple, but just the set of components of the zero-vector.

$$\mathbb{N} = \{1, 2, \ldots\}$$

A fundamental problem II

Definition

$\mathbb{P}$ is called *m*-**regular** for some $m \in \mathbb{N}$, if the hypergraph $N(\mathbb{P})$ is not m-colourable, that is:
For every partition $X = X_1 \cup \cdots \cup X_m$ of X into m subsets there is i and $\vec{x} \in N(\mathbb{P})$ with $\vec{x} \subseteq X_i$.
$\mathbb{P}$ is called **regular** if it is m-regular for all $m \geq 1$.

A central problem of Ramsey theory is:

Determine the (m-)regular systems $\mathbb{P}$.

The linear case was completely solved by Rado [9], generalising work by Schur [10] and van der Waerden [12]; a recent overview was given in Baglini [2].

A fundamental problem III

The general case is wide open.

- By Hilbert's Tenth Problem the case "1-regularity" is undecidable.
- But for all $m \geq 2$ the complexity of "m-regularity" is completely open (from linear-time to undecidable).
- And so is "regularity".

We considered the single equation $x^2 + y^2 = z^2$.

Finitisation

Partitioning into two sets, i.e., $m = 2$, can be directly encoded using boolean variables:

For $n \in \mathbb{N}$ let v_n be a boolean variable.
v_n `true` means the first part, v_n `false` means the second part.

Now $\mathbb{P}$ is 2-regular iff the (possibly) infinite disjunction

$$\bigvee_{\vec{x} \in N(\mathbb{P})} \left(\bigwedge_{x \in \vec{x}} v_x\right) \vee \left(\bigwedge_{x \in \vec{x}} \neg v_x\right)$$

is a tautology, which by compactness is true iff for some $n \in \mathbb{N}$ the **finite disjunction**

considering only the $\vec{x} \in N(\mathbb{P})$ with $\vec{x} \subseteq \{1, \ldots, n\}$

is a tautology.

The Boolean Pythagorean Triples Problem I

The question about the regularity of

$$x^2 + y^2 - z^2$$

is a long-standing open problem.

- Only 1-regularity was known (since there exist Pythagorean triples, e.g., $3^2 + 4^2 = 5^2$).
- Ron Graham asked in the 80s specifically whether 2-regularity holds (we call this the Boolean Pythagorean Triples Problem).
- There were some opinions (conjectures) that it is not 2-regular.

The Boolean Pythagorean Triples Problem II

We (together with Marijn Heule and Victor Marek)
showed via SAT the 2-regularity (Heule, Kullmann, and Marek [6]),
and got $100 for it.

Such a solution needs a **proof**, which can be automatically checked.

We provided a proof of 200 TB and checked it.
Meanwhile it has also been checked independently (Cruz-Filipe, Marques-Silva, and Schneider-Kamp [3]).

Concrete (practical) re-formulation I

Call three natural numbers $a, b, c \in \mathbb{N}$ a Pythagorean triple if $a^2 + b^2 = c^2$.

Is it possible to partition $\mathbb{N}$ into two sets $A \cup B = \mathbb{N}$, such that neither A nor B contains a Pythagorean triple?

Let's start with partitioning $\mathbb{N}$ into even and odd numbers. Remember:

- even * even = even, odd * odd = odd
- even + even = even, odd + odd = even.

Thus there can be no Pythagorean triple consisting only of odd numbers.
So we are fine on the odd part, but the even part contains e.g.

$$2 \cdot (3, 4, 5) = (6, 8, 10), \quad 6^2 + 8^2 = 10^2.$$

Concrete (practical) re-formulation II

Continuing with that attempt, we now needed to partition the even numbers further,

to destroy all Pythagorean triples.

For a more experimental approach, it would be easier

instead of partitioning $\mathbb{N}$,
to partition $\{1, \ldots, n\}$ for $n = 5, 6, 7 \ldots$

So we can first destroy all Pythagorean triples with hypotenuse $c \leq 5$, then $c \leq 6$, and so on. At the beginning it's pretty simple, since there are only few Pythagorean triples:

$$(3, 4, 5), (6, 8, 10), (5, 12, 13), (9, 12, 15), (8, 15, 17), \ldots$$

But slowly it gets harder ... will it ever end?!? (YES, as we showed.)

So what?

Alright,

- an apparently hard mathematical problem has been solved (open for more than 30 years),
- some people got excited about it,
- and we got some media attention (due to “AI”, “supercomputer” and “largest proof ever”; see links).

But 200 TB is hard to read,
and so do we now know anything more than just
“yes, it is true” ?!

The wheelbarrows

There is an old story about a worker suspected of stealing: every evening, as he leaves the factory, the wheelbarrow he rolls in front of him is carefully inspected. The guards can find nothing. It is always empty.

Finally, the penny drops: what the worker is stealing are the wheelbarrows themselves ...

Lesson: we should look not just at the message itself, but also at its environment/circumstances (which might actually be more important).

Wheelbarrow I: the length is the message

From an “ordinary” mathematical point of view, the length of the proof is a hindrance, is in the way of meaning — but perhaps

the meaning is its meaninglessness ?!

More precisely, we consider computational complexity theory:

1. Here we are interested in *inherent complexity*.
2. The questions we are considering are investigated as examples of inherent high complexity.

Wheelbarrow II: the brutishness is the message

To add insult to injury:

SAT is actually (some kind of) brute-force!

So the proof we delivered is a Proof by Exhaustion.

Case distinctions — a “blind spot” of mathematics?

Is there a Science of brute-force ?!
(See our forthcoming CACM article Heule and Kullmann [4] on this subject.)

Wheelbarrow III: the method is the message

Some speak of “clever/inspired/intelligent brute-force”. We prefer to see it as

$$\text{SAT} = \text{brute-force} + \text{brute reason}.$$

“Brute reason” organises the brute-force search.

> *I can stand brute force, but brute reason is quite unbearable. There is something unfair about its use. It is hitting below the intellect.* *O. Wilde (The Picture of Dorian Gray)*

The SAT revolution

Via SAT we can solve problems by “brute-force” which seem completely hopeless otherwise.

So very different from e.g. the Four Colour Theorem — we can’t just tick off possibilities, but **magic** is needed.

- Modern microprocessors not possible without SAT (Electronic design automation (EDA)).
- Major applications in verification (railway industry).

Perhaps just a mechanically operated lure?

Perhaps the mathematical problem here is just useful as a kind of sparring partner, to help developing the methods?

The specific SAT technique here,
Cube-and-Conquer ([5, 6],
has been developed in the context of solving
such purely mathematical problems ([1]),
and is now one of the strongest methods
to solve industrial problems!

Further motivations I

Knowing that something is true might make it easier to find an "ordinary" (possibly more general) proof.

- An example could be the Erdös Discrepancy Problem (open since 1932).
- In 2014, finally a special case was solved by SAT solving Konev and Lisitsa [7], yielding an extracted proof of 13 GB.
- While the general statement was proven 1 1/2 years later in Tao [11].

Furthermore, the ATP-proof might yield *numerical details* (in our case: 7825).

Further motivations II

The formal proof found by SAT might be investigated:

- Perhaps we can find structure in the proof.
- Via some form of proof mining, we might extract more specific information on the result (the proof shows more than a mere existence-result).
- We might even attempt “method mining”, studying the interactions between the methods used to find the proof and the problem itself.

Yet these are speculations, but hopefully in the future we gain more understanding.

An important aspect here is
to lift (part of) mathematics to the meta-level,
to integrate the logical foundations.

Further motivations III

Proof mining (here in a concrete, direct sense – mining the 200 TB) could also become interesting in the "negative sense":

- The "traditional" interest is to search for a "short proof".
- But perhaps the question, why there isn't one, or

  what makes the problem hard,

  is the real question here?!

SAT solving

The previous DNF is negated, yielding a CNF, and the task is to show

unsatisfiability (i.e., inconsistency).

SAT solvers solve CNFs.

- The hybrid SAT-solving method Cube-and-Conquer, whose idea we developed in the context of applications to Ramsey theory, was adopted to the task (various heuristics optimised).
- Due to the nature of "C&C", whether performed on a single computer or on a cluster doesn't matter.

C&C: old and new

SAT is “solving CNFs by brute-force, guided by brute reason”.

- Two main paradigms for “brute reason” have been developed.
- The first and older one is about logical inference and systematic case distinctions (“systematic and slow”).
- The second and newer one (mainly responsible for the SAT revolution in industrial applications) is about making mistakes quickly and learning from them (“quick and dirty, but with magic local cleverness”).

C&C combines the two:

> First we build a systematic (and clever) big(!) case distinction.
> Then we solve the cases by the quick method (independently!).

Actually **proving** it

Just claiming to have solved it — that's not much.

- The total run-time for solving the problem was two days on a supercomputer, where we used roughly $1000 = 10^3$ cores.
- Without our novel techniques, just using standard SAT techniques, it would have needed say 1000 times more *time*.
- Still doable in principle (the supercomputer has 10^6 cores).
- But the point is the **extracted proof** (which we got *down* to "just" 200 TB).
- The total run-time must be small, **AND** the proof format must allow for good compression.
- Without the novel proof format, a blow-up of *space* of at least a factor of 1000 would have occurred (compared to run-time).

Remark: without SAT, the age of the universe would be nothing to solve it.

Independent verification

Recently, the first (published) independent verification took place (Cruz-Filipe et al. [3]):

- They used the 10^6 **main cases** (still highly complex), as computed by our SAT-solving system (altogether 68 GB).
- These 10^6 SAT problems were solved by some SAT solver.
- For each case extracting a proof using our tools for preprocessing.
- Finally on each of the 10^6 extracted proofs, a verified proof checker was run.
- This checker was extracted by the Coq interactive theorem prover (with some work-arounds for speed and memory).

Useful?!

A typical argument, as articulated in the Nature article ([8]):

> *If mathematicians' work is understood to be a quest to increase human understanding of mathematics, rather than to accumulate an ever-larger collection of facts, a solution that rests on theory seems superior to a computer ticking off possibilities.*

One can see here a widespread missing understanding of computer science: computers simply "tick off possibilities".

- The *systematic background* and motivation for e.g., ATP, SAT solving, formal methods, is not known.
- The *complexity issues* touched here might be far more interesting/relevant than the concrete result in Ramsey theory.
- Last, but not least: the "possibilities" are non-trivial, and simple algorithms might take forever.

Perhaps “meaningless” is the true meaning?!

- The “computer ticking off possibilities” is actually quite a sophisticated thing here, and is absolutely crucial for the analysis for example of the correctness of microprocessors.
- For some not yet understood reasons it seems that these “benchmarks” from the field of Ramsey theory are relevant for the perhaps most fundamental problem of computer science, the question what makes a problem hard (e.g. the P vs NP problem).

  It might thus well be, that at the direct level, the number 7,825 is a kind of “pure fact”, but the methods for determining this are highly relevant – perhaps it is precisely that the "7,825" has no meaning, which makes these computational(!) problems meaningful – the bugs in the designs of complicated artificial systems also have “no meaning”!

Likely not so for bigger minds ... so let’s consider aliens.

“Alien” versus “human” truth

Let’s call “alien” a true statement (best rather short) with only a very long proof.

- Already the question, whether we can **show** something (like our case) to be alien, is of highest relevance.
- But independently, such “alien truths” (or “alien questions”) arise in formal contexts, where large propositional formulas describe engineering systems, which in their complexity, especially what concerns “small” bugs, are perhaps beyond “understanding”.
- Mathematicians dislike “nitty-gritty details”, but prefer “the big picture” (handwaving).
- Moreover, typically here the social-economic situation(!) actually inhibits any form of understanding (even if possible) – this needs to go quickly, in a fully automated way, and likely also in a secretive fashion. (Thanks to Markus Roggenbach and his forthcoming book for pointing this out.)

More on “alien truths” (for the reader of the slides)

Here a concrete example of an alien truth: link
Some more details (see [4]):

- Short statement.
- Only very very long proof (best inherently!).
- But all kind of combinatorial counts (the number of so-and-so of size 1234 etc.) are only “weakly alien”; they are not truly awe-inspiring, since this is really just ticking off possibilities.
- Also a real proof is needed, not just a computation (like the minimum number of givens is 17 in general Sudoku; these examples typically are also only “weakly alien”).
- It must also not just involve mathematical reasoning, plus a derived list of possibly many, but simple cases. Our proof is “truly alien”: **no insights**! And **mysteriously** avoiding an enormous exponential effort.

What's to be understood

I ATP hopefully has still much more to gain via SAT.

II For hard problems the interplay between "old" and "new" in C&C seems crucial.

III Hopefully we gain much more such "alien insights".

IV And hopefully new true knowledges arises: at the direct level (deeper understanding through algorithms), and at the indirect levels (the methods, the reflection).

End

(references on the remaining slides).

For my papers see
http://cs.swan.ac.uk/~csoliver/papers.html.

Bibliography I

[1] Tanbir Ahmed, Oliver Kullmann, and Hunter Snevily. On the van der Waerden numbers $\mathrm{w}(2;3,t)$. *Discrete Applied Mathematics*, 174:27–51, September 2014. doi:10.1016/j.dam.2014.05.007.

[2] Lorenzo Luperi Baglini. Partition regularity of nonlinear polynomials: a nonstandard approach. *INTEGERS: Electronic Journal of Combinatorial Number Theory*, 14:23, 2014. URL http://www.integers-ejcnt.org/vol14.html. #A30.

[3] Luís Cruz-Filipe, Joao Marques-Silva, and Peter Schneider-Kamp. Efficient certified resolution proof checking. Technical Report arXiv:1610.06984v2 [cs.LO], arXiv, October 2016. URL https://arxiv.org/abs/1610.06984.

[4] Marijn J.H. Heule and Oliver Kullmann. The science of brute force. *Communications of the ACM*, 2017, to appear.

Bibliography II

[5] Marijn J.H. Heule, Oliver Kullmann, Siert Wieringa, and Armin Biere. Cube and conquer: Guiding CDCL SAT solvers by lookaheads. In Kerstin Eder, João Lourenço, and Onn Shehory, editors, *Hardware and Software: Verification and Testing (HVC 2011)*, volume 7261 of *Lecture Notes in Computer Science (LNCS)*, pages 50–65. Springer, 2012. doi:10.1007/978-3-642-34188-5_8. URL http://cs.swan.ac.uk/~csoliver/papers.html#CuCo2011.

[6] Marijn J.H. Heule, Oliver Kullmann, and Victor W. Marek. Solving and verifying the boolean Pythagorean Triples problem via Cube-and-Conquer. In Nadia Creignou and Daniel Le Berre, editors, *Theory and Applications of Satisfiability Testing - SAT 2016*, volume 9710 of *Lecture Notes in Computer Science*, pages 228–245. Springer, 2016. ISBN 978-3-319-40969-6. doi:10.1007/978-3-319-40970-2_15.

Bibliography III

[7] Boris Konev and Alexei Lisitsa. Computer-aided proof of Erdös discrepancy properties. *Artificial Intelligence*, 224:103–118, July 2015. doi:10.1016/j.artint.2015.03.004.

[8] Evelyn Lamb. Maths proof smashes size record: Supercomputer produces a 200-terabyte proof – but is it really mathematics? *Nature*, 534:17–18, June 2016. doi:10.1038/nature.2016.19990.

[9] Richard Rado. *Studien zur Kombinatorik*. PhD thesis, Philosophische Fakultät der Friedrich-Wilhelms-Universität, Berlin, 1933. URL http://eudml.org/doc/203249.

[10] Issai Schur. Über die Kongruenz $x^m + y^m = z^m \pmod{p}$. *Jahresbericht der Deutschen Mathematikervereinigung*, 25: 114–117, 1917. URL https://eudml.org/doc/145475.

[11] Terence Tao. The Erdös discrepancy problem. *Discrete Analysis*, 1:29, February 2016. doi:10.19086/da.609. URL http://arxiv.org/abs/1509.05363v5.

Bibliography IV

[12] B.L. van der Waerden. Beweis einer Baudetschen Vermutung. *Nieuw Archief voor Wiskunde*, 15:212–216, 1927.

[13] Edward Zulkoski, Curtis Bright, Albert Heinle, Ilias Kotsireas, Krzysztof Czarnecki, and Vijay Ganesh. Combining SAT solvers with computer algebra systems to verify combinatorial conjectures. *Journal of Automated Reasoning*, 58(3):313–339, March 2017. doi:10.1007/s10817-016-9396-y.