Theoretical Foundations of Applied SAT Solving (14w5101)

Albert Atserias (Universitat Politecnica de Catalunya, Barcelona),
Armin Biere (Johannes Kepler University),
Samuel Buss (University of California, San Diego),
Antonina Kolokolova (Memorial University of Newfoundland),
Jakob Nordström (KTH Royal Institute of Technology),
Karem Sakallah (University of Michigan and Qatar Computing Research Institute)

January 19, 2014 – January 24, 2014

1 Overview of the Field and Purpose of the Workshop

Proving logic formulas is a problem of immense importance both theoretically and practically. On the one hand, it is believed to be intractable in general, and deciding whether this is so is one of the famous million dollar Clay Millennium Problems [20], namely the P vs. NP problem originating from the ground-breaking work of Cook [12]. On the other hand, today so-called SAT solvers based on *conflict-driven clause learning (CDCL)* [3, 18, 22] are routinely and successfully used to solve large-scale real-world instances in a wide range of application areas (such as hardware and software verification, electronic design automation, artificial intelligence research, cryptography, bioinformatics, operations research, and railway signalling systems, just to name a few examples — see, e.g., [8] for more details).

During the last two decades, there have been dramatic — and surprising — developments in SAT solving technology that have improved performance by many orders of magnitude. But perhaps even more surprisingly, the best SAT solvers today are still at the core based on relatively simple methods from the early 1960s [14, 15] (albeit with many clever optimizations), searching for proofs in the so-called resolution proof system [9]. While such solvers can often handle formulas with millions of variables, there are also known tiny formulas with just a few hundred variables that cause even the very best solvers to stumble (see, e.g., theoretical work in [10, 17, 28] and experimental results in [19, 29]). The fundamental question of when SAT solvers perform well or badly, and what underlying mathematical properties of the formulas influence SAT solver performance, remains very poorly understood. Other crucial SAT solving issues, such as how to optimize memory management and how to exploit parallelization on modern multicore architectures, are even less well studied and understood from a theoretical point of view.

Another intriguing fact is that although other mathematical methods of reasoning are known that are much stronger than resolution in theory, in particular methods based on algebra [11] and geometry [13], attempts to harness the power of such methods have conspicuously failed to deliver any significant improvements in practical performance. And while resolution is a fairly well-understood proof system, even very basic questions about these stronger algebraic and geometric methods remain wide open.

The purpose of this workshop was to gather leading researchers in applied and theoretical areas of SAT and proof complexity research and to stimulate an increased exchange of ideas between these two communities. To the best of our knowledge, this was the first large-scale workshop aimed specifically at bringing together practitioners and theoreticians from the two fields. We believe that proof complexity can shed light on the power and limitations on current and possible future SAT solving techniques, and that problems encountered in SAT solving can spawn interesting new areas in theoretical research. We see great opportunities for fruitful interplay between theoretical and applied research in this area, and believe that a more vigorous interaction between the two has potential for major long-term impact in computer science and mathematics, as well for applications in industry. Judging from the feed-back from the participants, the workshop provided a powerful stimulus in this direction.

2 Presentation Highlights and Workshop Themes

As this workshop brought together different communities many talks at the workshop were designed to be tutorials and surveys, intended to bridge the gaps between the diverse backgrounds of the participants. Specifically, the following five "mini-tutorials" were presented:

1. Marijn Heule: Conflict-driven clause learning (CDCL).
2. Sam Buss: Proof complexity.
3. Matti Järvisalo: Preprocessing.
4. Jakob Nordström Weak proof systems and connections to SAT solving.
5. Albert Atserias: Semialgebraic proof systems.

The presentations by Heule and Järvisalo covered more practical aspects of SAT solver design, the tutorials by Buss and Atserias presented relevant theoretical (proof complexity) background, and the tutorial by Nordström outlined the nature of relationship between the proof complexity and SAT solving.

In addition to the mini-tutorials, there were survey talks on integrating cutting planes/pseudo-Boolean reasoning in CDCL solvers (Daniel Le Berre), parameterized complexity and SAT (Stefan Szeider), satisfiability modulo theories (Albert Oliveras), and QBF solving (Martina Seidl).

The workshop talks and discussions can be grouped into several thematic areas. These include:

- *Practical and theoretical issues in the algorithmic design of SAT solvers.* A number of the talks discussed the design details of how the most successful SAT solvers are implemented. This included Heule's talk on CDCL, Järvisalo's talk on clause preprocessing, Simon's talk on the power of so-called *glue clauses* [1], and Biere's talk on the SAT solver *Lingeling* [6] and other related solvers. The last two presentations were able to delve into implementation details that have not been discussed well in the literature. On the other hand, there was also a general feeling that, in spite of the successes of SAT solvers, we still have a very limited understanding of when and why different techniques work well or poorly. There was a lot of discussion on this topic throughout the week as well as at a special Thursday evening meeting. Many participants expressed the opinion that SAT implementations need better benchmarking and better testing in order to develop an improved understanding of why the techniques currently in use work effectively and how they interact, as well as in order to make future progress on improving SAT solvers.

- *Theoretical issues for SAT solvers.* Several presentations highlighted gaps between our theoretical understanding of resolution and the implementation of SAT solvers. Since extended resolution is known to be much more powerful than resolution, one much discussed question was how the extension rule can speed up CDCL solvers. Buss's tutorial suggested heuristics for introducing extension variables, Goultiaeva suggested studying whether CDCL SAT solvers without restarts can polynomially simulate resolution, and Van Gelder gave examples of the power of extended resolution for several tautologies. In related topics, Nordström's, Lauria's and Johannsen's talks addressed the memory usage requirements for resolution proofs, and Szeider discussed parameterized complexity of SAT. These measures of proof complexity may also have impact on the effectiveness of SAT algorithms.

- *Parallelizability.* With the increasing availability of parallel computing, it is natural to try to harness this resource by implementing parallel algorithms for SAT. Sabharwal and Manthey both discussed parallel implementations of satisfiability algorithms and presented a wealth of graphical data analysing their performance. The problems encountered when implementing parallel solvers, especially what concerns choosing branching variables and sharing learned clauses, further highlight the gap between our practical algorithms for satisfiability and our understanding of why different techniques work well. For this reason, there was a lot of discussion on the topics of these talks, both on how sequential algorithms can be parallelized and on how the experiences with parallel algorithms can inform sequential algorithms.

- *Non-Boolean SAT solvers.* Quite a few talks addressed non-Boolean extensions of SAT solvers. This started with Atserias's tutorial talk on semi-algebraic proof systems. Galesi discussed space complexity of algebraic proof systems. Le Berre and Kullman discussed cutting planes, XOR constraints, and pseudo-Boolean constraints. Kalla spoke on Groebner basis methods. Marques-Silva gave a presentation about using SAT oracles to solve more general problems, such as counting. Beame talked about exact model counting. Olivera and Sinz discussed SMT (satisfiability modulo theories) systems, which are extensions of SAT solvers to more powerful logics. Sakallah discussed how detecting and breaking symmetries can make SAT solving more efficient.

  These extensions to SAT solvers are all very powerful, at least in the right situations. However, for general instances of the SAT problem it still seems that nothing can really beat SAT solvers based on conflict-driven clause learning. The reasons for this are a little mysterious; however, in all implementations to date, the extra power awarded by, e.g., algebraic or pseudo-Boolean reasoning cannot quite compensate for the extra time needed for performing such more advanced reasoning. Nonetheless, there is some prospect that the next major breakthrough in SAT solving may come from some kind of non-Boolean reasoning.

- *Quantified Boolean Formulas (QBF).* Determining the satisfiability of a quantified Boolean formula is a much harder problem than ordinary satisfiability. In spite of this, recent years have seen surprisingly good progress in this area. Seidl and Narodytska gave talks describing practical implementations of QBF solvers, along with theoretical justifications. Santhanam presented a new asymptotic improvement for QBF solving.

3 Open Problems

We solicited open problems from the participants prior to the workshop, and, additionally, held an open problem session during the first evening. Below follows a selection of open problems and conjectures proposed by participants of the workshop.[1]

- *Norbert Manthey:* An important open problem is that extended resolution is easy in principle to integrate into SAT solver (its a few lines of code for Riss [25], Glucose [16] or MiniSat [21]), but so far it is not used. The reason seems to be that there are no working heuristics known that give a hint which extension to perform, and when to perform it. Good heuristic suggestions would seem to be very valuable, and so if there are some proof related measurements that could be used to decide whether a learned clause should be kept or not (or for guiding the decision heuristic), this would definitely be worth knowing.

- *Norbert Manthey:* Another open problem is how to generate proofs within parallel SAT solvers. For portfolio-style SAT solvers this problem might be solvable with less effort than for search space splitting parallel SAT solvers. However, currently I do not know any parallel SAT solver that is able to produce an UNSAT proof in any format. Thus, one can also not compare how far current systems are away from actual (shortest) proof (except considering the number of total learned clauses as the length, and the maximum learned clause size as the width).

- *Sharad Malik:* It may be helpful for the theory and practice groups to come together to see if they can provide some insight into how modern CDCL locality-based SAT solvers exploit the structure

[1] Most of the open problems below, as well as several others, can be found in the list of open problem on the pre-workshop webpage `www.csc.kth.se/~jakobn/BIRS_14w5101/`. They have been lightly edited here for context.

of instances arising in practice. This has been wide open in the community and could benefit from discussion within this diverse group.

- *Alasdair Urquhart:* Can it be shown for well-known formula families such as the pigeonhole principle (PHP) formulas or Tseitin formulas that regular resolution is optimal? This conjecture seems very plausible to me, but I do not see how to approach it at the moment. More generally, one can ask whether it is possible to give general conditions on a set of clauses that ensure that regular resolution is optimal? In general, the examples separating general and unrestricted resolution have a rather artificial appearance, where we add "spoiler variables" to mess up any regular refutation.

- *Karem Sakallah:* During the workshop Sakallah presented an empirical study made by himself and Marques-Silva [26] of various SAT heuristics, and asked for theoretical explanations of the observed behaviour.

- *Adrian Darwiche:* Rather than splitting on variables when making decisions in a SAT solver, one could split on formulas. But does it help? And how to do that in the best way?

- *Armin Biere:* Current practical successful paradigms for SAT solving all use control-dominated algorithms, like variants of CDCL, WalkSAT, or Look-Ahead based algorithms, and are thus hard to port to highly parallel computing architectures, which require memory locality and thus data-flow orientation. It is a challenge to come up with SAT algorithms which are organized around data-flow.

- *Armin Biere:* Portfolio based SAT solving is the dominating approach in the parallel application track of the SAT competition [27]. However, the improvements we saw in the last two years are apparently based on using better sharing schemes for learned clauses, thus kind of implicit work splitting. From a practical point of view it is first of all still unclear how much of the success of solvers like Penelope or Plingeling can be attributed to the portfolio idea and how much is due to splitting the work. As the number of compute units is increased it is conjectured that the relative contribution of the portfolio part will saturate. Does this happen and when?

- *Armin Biere:* The variable scoring scheme VSIDS (variable state independent decaying sum) introduced by Chaff [22] and its modern variants is crucial for the speed of CDCL solvers. There is almost no empirical investigation on how it really works, and further no theoretical explanation why it is working so well.

- *Alexandra Goultiaeva:* It has been proven that CDCL SAT solvers, viewed appropriately as a proof system, can polynomially simulate resolution [24]. The order of decisions is assumed to be non-deterministic, i.e., the proof shows that (if a short resolution proof exists) there exists a sequence of decisions that would allow the solver to find a short resolution proof. I am very much interested in seeing this extended to heuristics, especially VSIDS. That is, I would like to see a result that either:
  - shows that whenever a short resolution proof exists, there will always be a sequence of decisions that respects VSIDS ordering and allows the solver to find a short resolution proof, or
  - shows a counterexample where a short resolution proof exists but a solver respecting VSIDS ordering (regardless of tie-breaking) can never find a short proof.

- *Massimo Lauria:* Prove lower (and upper) bounds on proof size in resolution (and stronger proof systems) for k-clique formulas and other combinatorial principles.[2]

- *Paul Beame:* Could one make SAT solvers learn not just clauses/bad partial assignments but more general residual formulas, or other kinds of "reasons" for formulas being unsatisfiable? What would be good practical "reasons" of this kind?

In addition to the Monday evening open problem session, we also organized a Thursday evening discussion session to provide a period for participants to raise questions, propose problems, and generally discuss and reflect on the issues raised during (the first four days of) the workshop. A selection of topics discussed during the Thursday evening session follows below.

[2] Code for generating these benchmarks in DIMACS format is available at `github.com/MassimoLauria/cnfgen`.

- *Karem Sakallah:* It is important to increase understanding. The SAT community has done a great job to produce good SAT solvers. It has done a worse job of explaining how they work. Competitions have been very important to drive progree but should now evolve. It is not important to have the fastest solver but to explain *why* the solver is fast. What happens when the knobs are turned? Which knobs are important? The communities are talking completely different languages. We need to invest in learning to understand each other. Finally, a question: Simple explanations need simple models. From where can we get such models?

- *Rahul Santhanam:* Can we use a scientific approach to understand hardness as an empirical phenomenon and make falsifiable claims? Possible hypotheses: backdoors, complexity measures, entropy, selection bias towards what works today — what do practitioners think about these hypotheses? What is the most general property Q such that formulas with property Q are easily solvable?

- *Moshe Vardi:* Here is a list of five central problems:
  - Go beyond resolution for SAT solvers.
  - Go beyond worst-case complexity in the analysis of SAT solver performance (by analogy, think of the smoothed analysis of Spielman-Teng as going beyond worst-case and average-case analysis for the simplex method in linear programming).
  - NP vs PSPACE vs EXPTIME: Is PSPACE really harder than NP?
  - Can we parallelize unit clause propagation / BCP?
  - Exact algorithms / computational complexity: Are any ideas from here relevant for real-world algorithms?

4 Opinions About the Workshop

Our impression from interacting with the participants during and after the workshop is that this was a highly successful event. The workshop was even considered significant enough that an editorial in the Communications of the ACM[3] was dedicated exclusively to this *"unusual workshop"* about which it said: *"Of course, one cannot expect robust bridges between two distinct technical communities to be erected in one week, but that should not diminish the tremendous value of such bridge-building workshops."*

We also collected feed-back from the participants after the workshop by sending out an e-mail message with questions to which almost half of the workshop participants responded. The questions together with a brief but representative selection of answers follow below.

4.1 What were a couple of good aspects of the workshop?

- *Having food together gave lots of opportunities for informal discussion, which was arguably more important than the presentations. The program had a good mix of practical and theoretical results, which started interesting discussion.*

- *My overall opinion of the workshop is very positive. I was able to get input from some of the best theoreticians and practitioners in the area of SAT, and learn about recent work in the area of SAT. The informal setting allowed interactions with many colleagues. Overall I believe this was a great experience!*

- *Most of the talks were very good. I especially enjoyed the tutorials. I also enjoyed the discussion session in which some selected people gave their opinions.*

- *I really liked the tutorials and surveys. I was hoping for an overview of the landscape of SAT solving, and I got exactly that. I also liked the fact that the theorists and practitioners made an attempt to communicate with each other.*

[3] Available at `cacm.acm.org/magazines/2014/3/172516-boolean-satisfiability/fulltext`.

- *The participants where among the best in their fields. The mix of theory and applied people was good, and the discussions between these groups was important. This also revealed some deficiency in the communication between these groups, and identifying a problem is the first step to a solution.*
- *The in-depth discussions, the broad coverage of a range of relevant issues — the agenda was very well planned.*
- *That it brought together people who do not normally get to interact; two communities that could help each other out by collaborating.*

4.2 What were some aspects that could have been better?

- *Presentation of open problems in applied SAT, and opportunities from the theory side could have made stronger, and earlier in the week. Communication across the field could have started faster (but I do not have any idea how to achieve this). Maybe, one should have tried to have some task oriented session, where somebody posts a problem, and an idea how to solve it (or without an idea, and another guy comes up with the idea), and then the whole group could decide whether this idea might work or not — again, I am not sure whether something like this would work.*
- *The interaction between the two communities was mainly done through presentations. Maybe some focused discussion on some specific topic would have been fun and productive (modulo a strong moderation).*
- *Maybe the schedule was tight, with not that much time for discussions between talks. But it is difficult to say, since people used to discuss together during breakfast, lunch or dinner.*

4.3 Was there any area or topic which was missing or which you would have liked to hear more about?

- *The correlation between the proof complexity and the representation of the problem, hence: encodings, and high-level to low-level problem transformation.*
- *Maybe a little more from the area of algorithms for SAT (exact, parameterized, randomized. . .).*
- *I would have liked to hear more about Random SAT/Survey propagation and BDD-based approaches.*
- *Perhaps more on SMT (but the program was very very good).*
- *Obviously, some areas of proof complexity were not covered, e.g. there were quite a few talks on QBF solving, but nothing on the proof complexity side of it. In general, I believe that proof complexity of logics other than just classical propositional logic are interesting.*
- *I found all relevant topics covered. Maybe the machine learning perspective could have added another aspect.*
- *It would have been helpful to hear more about SAT through divide and conquer. Some of this came up in the discussion on parallel SAT solvers, but this is an important topic in its own right since it deals with information at the interface of problem partitions.*

4.4 Was there any area or topic that could have been de-emphasized or skipped?

- *Given the objectives of the workshop, and the audience, I would emphasize overview talks, summarizing relevant recent results, than talks focusing on specific topics or papers, either recently published or to be published. The 50 min talks allow covering a good amount of material, and so it was (and will be) fine to give overviews.*
- *Some of the specialized talks were hard to follow for the other part of the community.*

- *Not as far as I can think of. The program looked quite balanced to me... But, on the other hand, I do not have that much idea about the full plate on the other side. At the risk of offending someone, I thought QBF could be de-emphasized, although not removed.*

4.5 Would you personally come again if you were invited to a similar workshop in Banff? In Dagstuhl?

For Banff: 18 "yes" answers, 2 "maybe,", 0 "no." For Dagstuhl: 17 "yes" answers, 3 "maybe,", 0 "no."

4.6 And any other comments you might have, of course...

- *I feel like I learned more than at most conferences, maybe all.*
- *I am looking forward to results that originated during the workshop. It seems to be the case that some things already started.*
- *This was the most useful Banff workshop I've attended.*
- *Thank you, I think you did a wonderful job organizing.*
- *I'd like to see even more PhD students at these kind of workshops.*

5 Scientific Progress Made

This workshop was intended to foster new collaborations among researchers from the different communities surrounding the SAT problem. The main challenge when bringing together people from such different areas is to help them even understand each other's languages, and it seems clear from the feed-back of the participant that the workshop made a major contribution in this regard. In addition, the workshop also spawned a number of joint research projects, and some of these projects have already produced concrete results. Specifically, participants of the workshop have reported the following papers as resulting from the workshop, or at least being significantly influenced by it:

1. Gilles Audemard and Laurent Simon: *Lazy clause exchange policy for parallel SAT solvers* [2].
2. Paul Beame and Ashish Sabharwal: *Non-Restarting SAT Solvers With Simple Preprocessing Can Efficiently Simulate Resolution* [4].
3. Olaf Beyersdorff and Oliver Kullmann: *Unified Characterisations of Resolution Hardness Measures* [5].
4. Armin Biere, Daniel Le Berre, Emmanuel Lonca and Norbert Manthey: *Detecting Cardinality Constraints in CNF* [7].
5. Vijay Ganesh, Kaveh Ghasemloo and Jimmy Liang: *VSIDS Branching Heuristics in SAT Solvers and Timed Graph Centrality with Smoothing*, submitted.
6. Niklas Een, Alex Legg, Nina Narodytska, and Leonid Ryzhyk: *SAT-based Strategy Extraction in Reachability Games*, accepted to AAAI 2015.
7. Mladen Mikša and Jakob Nordström: *Long Proofs of (Seemingly) Simple Formulas* [19].
8. Zack Newsham, Vijay Ganesh, Sebastian Fischmeister, Gilles Audemard, and Laurent Simon: *Impact of Community Structure on SAT Solver Performance* [23].

Many other research project collaborations were initiated as a result of this workshop, although they have not yet led to published papers (we know of at least six such projects, and there are probably more). Several of these projects are between researchers from different communities who met as a result of this workshop.

One concrete project that we want to mention is that João Marques-Silva, Karem Sakallah, and Laurent Simon are conducting a more extensive empirical study of modern CDCL solvers to determine the impact of their many knobs on performance (i.e., a more comprehensive study than the preliminary on in [26] mentioned above). This is still work in progress.

Another ongoing project is that Marijn Heule and Jakob Nordström have initiated a study of combinatorial benchmark formulas and how different SAT solvers perform on these formulas when different SAT solver settings are varied (such as restart frequency or VSIDS decay factor). The goal is to find theoretical explanations for the sometimes dramatic differences in performance discovered so far even with fairly minor changes in parameters, and to draw conclusions that can hopefully guide the design of better SAT solving heuristics.

Additionally, Norbert Manthey designed a CNF formula generator for the clique coloring problem, submitted to the SAT competition 2014 and the Configurable SAT solver challenge 2014, based on the ideas from the workshop presentation by Allen van Gelder titled *Elementary short refutations for the clique-coloring principle and for Tseitin odd-charge graph formulas in extended resolution*. This is intended to create a reference benchmark for the development of extended resolution in SAT solvers.

6 Outcome of the Meeting

The explicit goal of this workshop was to increase the interaction and collaboration between experts in practical SAT solving and theoreticians in the related area of proof complexity. The tutorials and surveys were used to bring both communities on the same page with regards to advances and challenges in SAT solving and available proof complexity techniques that can be used to explain some of the observed behaviour. Thus, a first important outcome of the workshop was to provide both communities with extensive background knowledge of, respectively, theoretical and practical aspects of modern SAT solving.

By organizing two open problem/discussion sessions, we encouraged the participants to present challenges, insights and open problems in their respective areas. There are several collaborations that have started as a result of these discussions, and there are eight papers attributed, at least in part, to this workshop — see the previous section for more details.

Overall, this workshop has given a powerful impetus to a number of world-leading senior practitioners and theoreticians in SAT-related research, as well as to a selection of prominent junior researchers, to collaborate and explore the theoretical foundations of applied SAT solving. We fully anticipate more progress on the development and understanding of SAT solvers over the next several years.

An important next step to further stimulate research in this direction and strengthen the contacts between practitioners and theoreticians will be the Dagstuhl workshop *Theory and Practice of SAT Solving* in April 2015, explicitly intended as a follow-up on the workshop in Banff. We also hope that it will be possible to organize further workshops on this topic in Banff and/or Dagstuhl in future years to contitute to support these developments. We believe that both applied and theoretical researchers stand to benefit greatly from a more vigorous interaction between the two communities, and that this could have a major long-term impact for both academical research and industrial applications.

References

[1] Gilles Audemard and Laurent Simon. Predicting learnt clauses quality in modern SAT solver. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI '09)*, pages 399–404, July 2009.

[2] Gilles Audemard and Laurent Simon. Lazy clause exchange policy for parallel SAT solvers. In *Proceedings of the 17th International Conference on Theory and Applications of Satisfiability Testing (SAT '14)*, volume 8561 of *Lecture Notes in Computer Science*, pages 197–205. Springer, July 2014.

[3] Roberto J. Bayardo Jr. and Robert Schrag. Using CSP look-back techniques to solve real-world SAT instances. In *Proceedings of the 14th National Conference on Artificial Intelligence (AAAI '97)*, pages 203–208, July 1997.

[4] Paul Beame and Ashish Sabharwal. Non-restarting SAT solvers with simple preprocessing can efficiently simulate resolution. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence (AAAI '14)*, pages 2608–2615, July 2014.

[5] Olaf Beyersdorff and Oliver Kullmann. Unified characterisations of resolution hardness measures. In *Proceedings of the 17th International Conference on Theory and Applications of Satisfiability Testing (SAT '14)*, volume 8561 of *Lecture Notes in Computer Science*, pages 170–187. Springer, July 2014.

[6] Armin Biere. Lingeling, Plingeling, PicoSAT and PrecoSAT at SAT Race 2010. Technical Report 10/1, FMV Reports Series, Institute for Formal Models and Verification, Johannes Kepler University, August 2010.

[7] Armin Biere, Daniel Le Berre, Emmanuel Lonca, and Norbert Manthey. Detecting cardinality constraints in CNF. In *Proceedings of the 17th International Conference on Theory and Applications of Satisfiability Testing (SAT '14)*, volume 8561 of *Lecture Notes in Computer Science*, pages 285–301. Springer, July 2014.

[8] Armin Biere, Marijn J. H. Heule, Hans van Maaren, and Toby Walsh, editors. *Handbook of Satisfiability*, volume 185 of *Frontiers in Artificial Intelligence and Applications*. IOS Press, February 2009.

[9] Archie Blake. *Canonical Expressions in Boolean Algebra*. PhD thesis, University of Chicago, 1937.

[10] Vašek Chvátal and Endre Szemerédi. Many hard examples for resolution. *Journal of the ACM*, 35(4):759–768, October 1988.

[11] Matthew Clegg, Jeffery Edmonds, and Russell Impagliazzo. Using the Groebner basis algorithm to find proofs of unsatisfiability. In *Proceedings of the 28th Annual ACM Symposium on Theory of Computing (STOC '96)*, pages 174–183, May 1996.

[12] Stephen A. Cook. The complexity of theorem-proving procedures. In *Proceedings of the 3rd Annual ACM Symposium on Theory of Computing (STOC '71)*, pages 151–158, 1971.

[13] William Cook, Collette Rene Coullard, and Gyorgy Turán. On the complexity of cutting-plane proofs. *Discrete Applied Mathematics*, 18(1):25–38, November 1987.

[14] Martin Davis, George Logemann, and Donald Loveland. A machine program for theorem proving. *Communications of the ACM*, 5(7):394–397, July 1962.

[15] Martin Davis and Hilary Putnam. A computing procedure for quantification theory. *Journal of the ACM*, 7(3):201–215, 1960.

[16] The Glucose SAT solver. `http://www.labri.fr/perso/lsimon/glucose/`.

[17] Armin Haken. The intractability of resolution. *Theoretical Computer Science*, 39(2-3):297–308, August 1985.

[18] João P. Marques-Silva and Karem A. Sakallah. GRASP: A search algorithm for propositional satisfiability. *IEEE Transactions on Computers*, 48(5):506–521, May 1999. Preliminary version appeared in *ICCAD '96*.

[19] Mladen Mikša and Jakob Nordström. Long proofs of (seemingly) simple formulas. In *Proceedings of the 17th International Conference on Theory and Applications of Satisfiability Testing (SAT '14)*, volume 8561 of *Lecture Notes in Computer Science*, pages 121–137. Springer, July 2014.

[20] The Millennium Problems of the Clay Mathematics Institute, May 2000. See the webpage `http://www.claymath.org/millennium/`.

[21] The MiniSat page. `http://minisat.se/`.

[22] Matthew W. Moskewicz, Conor F. Madigan, Ying Zhao, Lintao Zhang, and Sharad Malik. Chaff: Engineering an efficient SAT solver. In *Proceedings of the 38th Design Automation Conference (DAC '01)*, pages 530–535, June 2001.

[23] Zack Newsham, Vijay Ganesh, Sebastian Fischmeister, Gilles Audemard, and Laurent Simon. Impact of community structure on SAT solver performance. In *Proceedings of the 17th International Conference on Theory and Applications of Satisfiability Testing (SAT '14)*, volume 8561 of *Lecture Notes in Computer Science*, pages 252–268. Springer, July 2014.

[24] Knot Pipatsrisawat and Adnan Darwiche. On the power of clause-learning SAT solvers as resolution engines. *Artificial Intelligence*, 175:512–525, February 2011. Preliminary version appeared in *CP '09*.

[25] Riss. `http://tools.computational-logic.org/content/riss.php`.

[26] Karem A. Sakallah and João Marques-Silva. Anatomy and empirical evaluation of modern SAT solvers. *Bulletin of the EATCS*, 103:96–121, 2011.

[27] The international SAT Competitions web page. `http://www.satcompetition.org`.

[28] Alasdair Urquhart. Hard examples for resolution. *Journal of the ACM*, 34(1):209–219, January 1987.

[29] Allen Van Gelder and Ivor Spence. Zero-one designs produce small hard SAT instances. In *Proceedings of the 13th International Conference on Theory and Applications of Satisfiability Testing (SAT '10)*, volume 6175 of *Lecture Notes in Computer Science*, pages 388–397. Springer, July 2010.