

Computer Algebra and Satisfiability Modulo Theories

Invited Talk at ACA 2022

James Davenport

University of Bath
(supported by EPSRC under EP/T015713)

- I have worked in computational algebra, especially computer algebra/symbolic computation, for over 50 years: [SDSD⁺75] contains results from 1970–72, and my thesis [Dav81] is still occasionally cited.
- Since the inspirational talk of Erika Ábrahám at ISSAC 2015 [Á15] I have been interested in the interface between Symbolic Computation and Satisfiability Checking.
- This has led to an EU-funded project [ABB⁺16], and a sequence of workshops
<http://www.sc-square.org/workshops.html>.
- Initially this seemed like a dialogue of the deaf [Dav17], but things are improving.

Complexity in Computer Algebra (1)

When it comes to dense polynomials/matrices, we have found good algorithms, and apart from the gap in matrix multiplication etc. exponent ($2 + o(1) \leq \omega \leq 2.3728596$ [AW21]) we have upper bounds that match lower bounds for most basic problems.

When it comes to sparse polynomials, there is a major problem: lack of correlation between input size and output size. Consider

$$\begin{aligned}\frac{x^n - 1}{x - 1} &= x^{n-1} + x^{n-2} + \dots + 1 \\ \gcd(x^{pq} - 1; x^{p+q} - x^p - x^q + 1) &= x^{p+q-1} - x^{p+q-2} \pm \dots - 1\end{aligned}$$

(the second example is due to [Sch03]). There has been significant progress here since [DC09]: [GGdCR22] has algorithms that are nearly linear in $\max(|\text{Input}|, |\text{Output}|)$.

[ETCT22] is an interesting development in the area of polynomial root isolation: their algorithm is $\tilde{O}_B(d^2 + d\tau)$ **on average** whereas the “best” algorithm has a **worst case** complexity of $\tilde{O}_B(d^3 + d^2\tau)$

Complexity isn't all

But many problems do not have as neat a correlation between worst-case complexity and average-case, even if one could prove a result about “average complexity”, which often has not been done.

- Polynomial Factorisation: the answer may be “irreducible”, but this may be easily found, or be very difficult [SD69]. The time may also depend on random choices for evaluations.
- Gröbner bases: the answer may be $\{1\}$, but this may require a great deal of computation. Note that there are doubly-exponential Gröbner bases [Chi09, MR13], but these seem to be “rare”.
- Cylindrical Algebraic Decomposition. Again, there are doubly-exponential examples [BD07, DH88], but these seem to be “rare”.

Can we quantify “rare”? See [AL17], who propose to ignore exponentially rare doubly exponential examples.

Figure: ISSAC Proceedings, from [vdH22]

Study	1997	2004	2012	2017	2021
None	53%	47%	27%	28%	37%
Benchmarks	17%	30%	22%	24%	24%
Complexity analysis	22%	9%	27%	39%	29%
Both	8%	14%	24%	9%	10%

Table 1: Statistics for the number of papers in ISSAC proceedings that contain benchmarks, a complexity analysis, nothing of this kind, or both.

Note that (apart from 2004!) there was significantly more attention on complexity than benchmarks.

However, by and large, every benchmark set is different.

SAT is the quintessential NP-complete problem which is stunningly easy in practice (much of the time). Since 2000, every car made by a German manufacturer is a result of SAT-solving [?].

Hence the SAT and SMT communities collect large, centrally-curated, collections of benchmarks. There is still a question of how representative these are, and new sub-collections are being contributed [UDE22].

[Dav21] is a plea for Computer Algebra to follow the tradition of the SAT and SMT communities, and collect large, centrally-curated, collections of benchmarks. **Computer Algebra is bad at this.**

Since this is a SIGSAM-sponsored conference, we might have a discussion on how SIGSAM could help with this.

Such Benchmark sets allow contests

SAT contests are here: <http://www.satcompetition.org>. They have been run since 2002. In the early years, there were distinct tracks for Industrial/Handmade/Random problems: this has been abandoned. SMT contests now have Cloud and Parallel tracks. The methodology is that the organisers accept submissions (from contestants¹ and others), then produce a list of problems (in DIMACS, a standard format) and set a time (and memory) limit, and see how many of the problems the submitted systems can solve on the contest hardware.

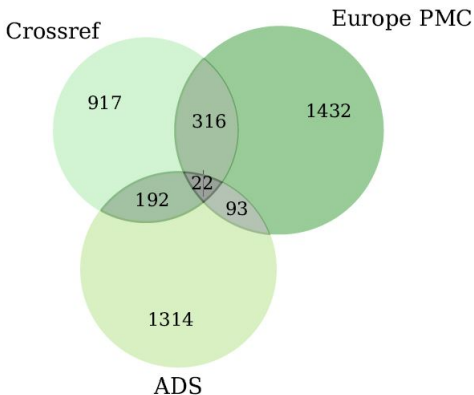
SAT is easy to certify (the solver just produces a list of values of the x_i). Verifying UNSAT is much harder, but since 2013 the contest has required proofs.

The general feeling is that these contests have really pushed the development of SAT solvers, roughly speaking $\times 2$ /year. For comparison, Linear Programming has done $\times 1.8$ over a greater timeline and with more rigorous documentation [Bix15].

¹In 2020, contestants were required to submit at least 20 problems, as well as a solver.

We need to improve Data Citation

- Data Citation is a mess in practice [vdSNI⁺19]: only 1.16% of dataset DOIs in Zenodo are cited (and 98.5% of these are self-citation).
- Is poorly harvested: [vdSNI⁺19, Figure 5].



so there
are between
4,000–20,000
data sets waiting
to be harvested

We need to improve Data Citation

- Data Citation is a mess in practice [vdSNI⁺19]: only 1.16% of dataset DOIs in Zenodo are cited (and 98.5% of these are self-citation).
- Is poorly harvested: [vdSNI⁺19, Figure 5].
- Is still a subject of some uncertainty: [MN12, KS14]
- Changes are still being proposed [DPS⁺20]
- *de facto* people cite a paper if they can find one.
- ? A rôle for *Communications in Computer Algebra*.

OEIS Online Encyclopedia of Integer Sequences [Slo03];

Long time at a personal site: <http://www.research.att.com/~njas/sequences>; now at <https://oeis.org/>.

- * Recommended citation: “N. J. A. Sloane, editor, The On-Line Encyclopedia of Integer Sequences, published electronically at <https://oeis.org>, [date]”.



But you have to search the website to find it!

- + Large toolset around it.

Group Theory (as an example)

- The Classification of Finite Simple Groups
 - The Transitive Groups acting on n points: [BM83] ($n \leq 11$); [Roy87] ($n = 12$); [But93] ($n = 14, 15$); [Hul96] ($n = 16$); [Hul05] ($17 \leq n \leq 31$); [CH08] ($n = 32$).
 - These are in GAP (and MAGMA), except that $n = 32$ isn't in the default build.
- + These are a really great resource (if that's what you want)
- How do you cite them? “[The21, GAP transgrp library]”?

Also Other libraries such as primitive groups



Group Theory is “easy”: for a given n there are a finite number and we “just” have to list them.

SMT can learn from CA: Finite Fields

A new trend at SMT 2022 was having the theory T be finite fields (sometimes \mathbf{F}_p for small p , sometimes \mathbf{F}_p for large p , and sometimes \mathbf{F}_{q^n}).

Note that if we actually want solutions in \mathbf{F}_p rather than its algebraic closure, we need to add the field equation $x^p - x$: a lesson that the SMT community is learning.

\mathbf{F}_p small The usual approach is “bit-blasting”, i.e. representing a number $< 2^n$ by n booleans representing its bits. But both JHD ($p = 7$) and [Had22] ($p = 3, 13$) have had success with direct representations (one Boolean for each value).

\mathbf{F}_p large This was described at SMT 2022, but the field equations are a major bottleneck.

\mathbf{F}_{q^n} As above.

The key trick with the field equations for large fields is to ignore them at first. After we have a system of equations, via Gröbner or SMT, then if the equations are zero-dimensional there is (possibly after FGLM [FGLM93]) a polynomial p_n in x_n only. Then by repeated squaring and reduction we compute $x_n^p - x_n$ modulo p_n . Even if the equations are not zero-dimensional, this is often a better approach. JHD has had some (limited) success with

- 1 Compute Gröbner Basis (without field equations)
- 2 Add the reduction of $x_n^p - x_n$ modulo the whole Gröbner Basis
- 3 Recompute the Gröbner Basis and repeat as necessary.

A problem for both fields, but where the SMT community is probably more advanced, is the question of the accuracy of the systems. Both Computer Algebra and SMT systems are large, complex, and often multi-generational, systems. The multi-generational aspect is probably more acute in Computer Algebra: JHD still gets queries (and bug reports) on his thesis [Dav81]. Maple is over 40 years old, SageMath may appear new, but incorporates Macsyma which is 55 years old. Practically no computer algebra system can “explain” its results: [Dav81] can produce a trace of decisions, but that is a long way from a certificate.

GCD Verifying “common divisor” is easy, “**greatest**” is NP-hard for sparse polynomials [Pla84].

GCD and SAT! The NP-hardness results of [Pla84] rely on encoding a SAT-formula W in x_1, \dots, x_n as $P_M(W)$, which vanishes at the $M = \prod_{i=1}^n p_i$ th roots of unity corresponding to satisfying assignments for W .

Blowup [Sch03]

$$\begin{aligned} \gcd(x^{pq} - 1, x^{p+q} - x^p - x^q + 1) &= \frac{(x^p - 1)(x^q - 1)}{x - 1} \\ &= \underbrace{x^{p+q-1} + x^{p+q-2} \pm \dots - 1}_{2 \min(p, q) \text{ terms}} \end{aligned}$$

Accuracy in Computer Algebra: Formal Proof

GCD As above.

Factorisation Verifying that these are the factors is easy: verifying that they are irreducible may require the trace of the original run in hard cases [SD69].

Integration $\int f = g$ is (generally) verifiable by differentiation, verifying unintegrability seems to require a trace, and proof of the relevant theorems.

Gröbner Bases Verifying that the output G is a Gröbner base is (relatively) trivial: All $S(g_i, g_j) \Rightarrow^G 0$. Verifying that it's the base of the input f_i is harder: $\forall i : f_i \Rightarrow^G 0$ and we need to track the linear algebra and end up with the $c_{i,j}$ such that each $g_i = \sum_j c_{i,j} f_j$.

Cylindrical Algebraic Decomposition This is really hard, and brighter people than me have failed [CM12] to prove the algorithm. There is some hope that we could prove specific instances [ADE⁺20], but this has not been tried yet.

[NPB22] report on an SMT fuzzer for testing the reliability of SMT solvers. It appears this is the latest of several such. It does seem pretty impressive: “more than 100 [errors] for cvc5 alone, and some of them critical”.

Since most of these SMT systems are written in C/C++, it was possible to use gcov, the standard GCC tool, to measure code coverage from fuzzing. Though not perfect, it is a measure of test validity.

This would be difficult/impossible for the “kernel plus written in self” model of most algebra systems.

Nevertheless, Computer Algebra systems, despite their age, could benefit from modern software engineering.



E. Ábrahám.

Building Bridges between Symbolic Computation and Satisfiability Checking.

In D. Robertz, editor, *Proceedings ISSAC 2015*, pages 1–6, 2015.



E. Ábrahám, B. Becker, A. Bigatti, B. Buchberger, A. Cimatti, J.H. Davenport, M. England, P. Fontaine, S. Forrest, D. Kroening, W. Seiler, and T. Sturm.

SC²: Satisfiability Checking meets Symbolic Computation (Project Paper).

In *Proceedings C1CM 2016*, pages 28–43, 2016.

 E. Ábrahám, J.H. Davenport, M. England, G. Kremer, and Z.P. Tonks.

New Opportunities for the Formal Proof of Computational Real Geometry?

SC²'20: Fifth International Workshop on Satisfiability Checking and Symbolic Computation CEUR Workshop Proceedings, 2752:178–188, 2020.

 D. Amelunxen and M. Lotz.




Average-case complexity without the black swans.

J. Complexity, 41:82–101, 2017.

 J. Alman and V.V. Williams.

A refined laser method and faster matrix multiplication.

In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms*, pages 522–539, 2021.

-  C.W. Brown and J.H. Davenport.
The Complexity of Quantifier Elimination and Cylindrical Algebraic Decomposition.
In C.W. Brown, editor, *Proceedings ISSAC 2007*, pages 54–60, 2007.
-  E.R. Berlekamp.
Factoring Polynomials over Large Finite Fields.
Math. Comp., 24:713–735, 1970.
-  R.E. Bixby.
Computational Progress in Linear and Mixed Integer Programming.
Presentation at ICIAM 2015, 2015.



G. Butler and J. McKay.

The transitive groups of degree up to 11.

Comm. Algebra, 11:863–911, 1983.



G. Butler.

The transitive groups of degree fourteen and fifteen.

J. Symbolic Comp., 16:413–422, 1993.



J.J. Cannon and D.F. Holt.

The transitive permutation groups of degree 32.

Experiment. Math., 17:307–314, 2008.



A.L. Chistov.

Double-exponential lower bound for the degree of any system of generators of a polynomial prime ideal.

St. Petersburg Math. J., 20:983–1001, 2009.



C. Cohen and A. Mahboubi.

Formal Proofs in Real Algebraic Geometry: From Ordered Fields to Quantifier Elimination.

Logical Methods in Computer Science, 8:1–40, 2012.



J.H. Davenport.

On the Integration of Algebraic Functions, volume 102 of *Springer Lecture Notes in Computer Science*.

Springer Berlin–Heidelberg–New York (Russian ed. MIR Moscow 1985), 1981.



J.H. Davenport.

SMT Nonlinear Real Arithmetic and Computer Algebra: a Dialogue of the Deaf?

SMT 2017 Satisfiability Modulo Theories CEUR Workshop 1889, pages 1–1, 2017.



J.H. Davenport.

Digital Collections of Examples in Mathematical Sciences.
To appear in Proc. EMS 2021, 2021.



J.H. Davenport and J. Carette.

The Sparsity Challenges.


In S. Watt *et al.*, editor, *Proceedings SYNASC 2009*, pages
3–7, 2009.



J.H. Davenport and J. Heintz.

Real Quantifier Elimination is Doubly Exponential.

J. Symbolic Comp., 5:29–35, 1988.

-  M. Daquino, S. Peroni, D. Shotton, G. Colavizza, B. Ghavimi, A. Lauscher, P. Mayr, M. Romanello, and P. Zumstein.
The OpenCitations Data Model.
International Semantic Web Conference 2020, pages 447–463, 2020.
-  Alperen Ergür, Josué Tonelli-Cueto, and Elias Tsigaridas.
Beyond Worst-Case Analysis for Root Isolation Algorithms .
In Hashemi [Has22], pages 139–148.
-  J.-C. Faugère, P. Gianni, D. Lazard, and T. Mora.
Efficient Computation of Zero-Dimensional Gröbner Bases by Change of Ordering.
J. Symbolic Comp., 16:329–344, 1993.

-  Pascal Giorgi, Bruno Grenet, Armelle Perret du Cray, and Daniel S. Roche.
Sparse Polynomial Interpolation and Division in Soft-linear Time.
In Hashemi [Has22], pages 459–468.
-  T. Hader.
Non-Linear SMT-Reasoning over Finite Fields.
Master's thesis, TU Wien, 2022.
-  Amir Hashemi, editor.
ISSAC '22: Proceedings of the 2022 International Symposium on Symbolic and Algebraic Computation, New York, NY, USA, 2022. Association for Computing Machinery.



A. Hulpke.

Konstruktion transitiver Permutationsgruppen.
PhD thesis, RWTH Aachen, 1996.



A. Hulpke.

Constructing transitive permutation groups.
J. Symbolic Comput., 39:1–30, 2005.



W. Kuchlin and C. Sinz.

Proving Consistency Assertions for Automotive Product Data Management.
J. Automated Reasoning, 24:145–163, 2000.



J. Kratz and C. Strasser.

Data publication consensus and controversies (version 3).
F1000Research Article 94, 3, 2014.



H. Mooney and M.P. Newton.

The Anatomy of a Data Citation: Discovery, Reuse, and Credit.

Journal of Librarianship and Scholarly Communication Article p.eP1035, 1, 2012.



E.W. Mayr and S. Ritscher.

Dimension-dependent bounds for Gröbner bases of polynomial ideals.

J. Symbolic Comp., 49:78–94, 2013.



A. Niemetz, M. Preiner, and C. Barrett.

Murxla: A Modular and Highly Extensible API Fuzzer for SMT Solvers.

International Conference on Computer Aided Verification, pages 92–106, 2022.



D.A. Plaisted.

New NP-Hard and NP-Complete Polynomial and Integer Divisibility Problems.

Theor. Comp. Sci., 31:125–138, 1984.



G.F. Royle.

The Transitive Groups of Degree Twelve.




J. Symbolic Comp., 4:255–268, 1987.






A. Schinzel.

On the greatest common divisor of two univariate polynomials, I.

In *A Panorama of number theory or the view from Baker's garden*, pages 337–352. C.U.P., 2003.

-  H.P.F. Swinnerton-Dyer.
Letter to E.R. Berlekamp.
Mentioned in [Ber70], 1969.
-  H.P.F. Swinnerton-Dyer, N.M. Stephens, J.H. Davenport,
J. Vélu, F.B. Coghlan, A.O.L. Atkin, and D.J. Tingley.
Numerical Tables on Elliptic Curves.
*Modular Functions of One Variable IV (Proceedings Antwerp
1972), pages 75–114, 1975.*
-  N.J.A. Sloane.
The Online Encyclopedia of Integer Sequences.
Notices A.M.S., 50:912–915, 2003.

-  The GAP Group.
GAP — Groups, Algorithms, and Programming, Version 4.11.1.
<https://www.gap-system.org>, 2021.
-  A.K. Uncu, J.H. Davenport, and M. England.
SMT-Solving Combinatorial Inequalities.
To appear in Proc. SCSC 2022, 2022.
-  Joris van der Hoeven.
On the Complexity of Symbolic Computation.
In Hashemi [Has22], pages 3–12.



S. van de Sandt, L.H. Nielsen, A. Ioannidis, A. Muench, E. Henneken, A. Accomazzi, C. Bigarella, J.B.G. Lopez, and S. Dallmeier-Tiessen.

Practice meets Principle: Tracking Software and Data Citations to Zenodo DOIs.

<https://arxiv.org/abs/1911.00295>, 2019.