Digital Collections of Examples in Mathematical Sciences

James Davenport[1]

University of Bath

9 December 2021

see `https://arxiv.org/abs/2107.12908` and [Dav18]

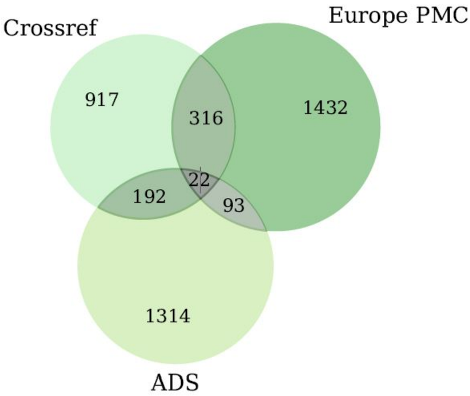
[1]Partially Supported by EPSRC Grant EP/T015713/1

Plan of Talk

1. Data Citation
2. Important Collections in Pure Mathematics
3. Important Test Suites in SAT/SMT
4. The Lack of Test Suites elsewhere
5. Complexity Theory and its weaknesses
6. Way Forward?

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

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.

Important Databases in Pure Mathematics

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.

SAT Solving

SAT solving, normally seen as solving a Boolean expression written in CNF. Given a 3-literals/clause CNF satisfiability problem,

$$\underbrace{(l_{1,1} \vee l_{1,2} \vee l_{1,3})}_{\text{Clause 1}} \wedge (l_{2,1} \vee l_{2,2} \vee l_{2,3}) \wedge \cdots \wedge (l_{N,1} \vee l_{N,2} \vee l_{N,3}),$$

where $l_{i,j} \in \{x_1, \overline{x_1}, x_2, \overline{x_2}, \ldots\}$, is it satisfiable? In other words, is there an assignment of $\{T, F\}$ to the x_i such that all the clauses are *simultaneously* true.

3-SAT: the quintessential NP-complete problem [Coo66]. 2-SAT is polynomial, and k-SAT for $k > 3$ is polynomial-transformable into 3-SAT. In practice we deal with SAT — i.e. no limitations on the length of the clauses.

Let n be the number of i such that x_i (and/or $\overline{x_i}$) actually occur. Typically n is of a similar size to N.

SAT Solving

Despite being NP-complete, nearly all examples are easy (e.g. [KS00] for the automotive industry),
either easily solved (SAT) or easily proved insoluble (UNSAT) and for random problems there seems to be a distinct phase transition between the two: [GW94, AP04, AP06].
This means that constructing difficult examples is itself difficult, and a research area in itself: [Spe15, BC18].
SAT solving has many applications, e.g. new ways of multiplying matrices [HKS21], so we want effective solvers for “real” problems, not just “random” ones.

Fundamental question: what does this mean?

SAT Contests: `http://www.satcompetition.org`

Been run since 2002. In the early years, distinct tracks for Industrial/Handmade/Random problems: this has been abandoned.
The methodology is that the organisers accept submissions (from contestants and others), then produce a list of problems (in 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 (just produce a list of values), UNSAT is much harder, but since 2013 the contest has required proofs of UNSAT for the UNSAT track, and since 2020 in all tracks, in DRAT: a specified format (some have been $>$ 100GB).
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 [Bix15], chips $\times 1.41$ [Moo75].

SAT in practice

Although SAT, and k-SAT, are reducible to 3-SAT, *we shouldn't do this in practice*. Indeed, some solvers look for this reduction and undo it.

A significant practical development was "Two Watched Literals" [MMZ+01], which means that, most of the time, we don't look at a whole clause.

> *In the worst case it is no more efficient than the algorithm it replaced, in practice it is hundreds of times more efficient. The "not moving back" feels like it should save at most half the time, it in fact speeds things up by often 10 times.*
> `https://news.ycombinator.com/item?id=18236555`

In particular it is much more cache-efficient on modern processors. This is essentially orthogonal to complexity theory as we know it.

SMT: life beyond SAT

Consider a theory T, with variables y_j, and various Boolean-valued statements in T of the form $F_i(y_1, \ldots, y_n)$, and a CNF with $F_i(y_1, \ldots, y_n)$ rather than just x_i.
Then the SAT/UNSAT question is similar ($\exists$ values of y_i ...), and the community runs SMT Competitions (`https://smt-comp.github.io/2020/`), but a separate track for each theory, as the problems will be different.
The SMTLIB format [BFT17] provides a standard input format.
UNSAT is in general unsolved (but see [KAED21] for one example).
There is substantial progress in SMT-solving over the years, possibly similar to SAT.

Computer Algebra: where are we

Obviously, Group Theory (etc.) are part of computer algebra: what about the rest?

Much traditional complexity theory is for dense polynomials/matrices/..., but real life is often about sparse ones.

In general the problems have a bad worst-case complexity, and we want effective solvers for "real" problems, not just "random" ones.

The question is "what does this mean?".

Format No common standard. We do have OpenMath [BCC+17], but it's not as widely supported as we would like.

Contests None. Could SIGSAM organise them?

Problem Sets No independent ones. Each author chooses his own.

Archive Not really.

Polynomial g.c.d.

- NP-hard (for sparse polynomials, even univariate) [Pla84].
- Can be challenging for multivariates
  - No standard database: trawl previous papers (and often need to ask the authors)

Verification is a challenge: one can check that the result is *a common divisor*, but verifying *greatest* is still NP-hard [Pla84].

Polynomial g.c.d.: However

The NP-hardness results of [Pla84] rely on encoding a SAT-formula W in $x_1, \ldots, x_n$ 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.
There are also blow-up results [Sch03]

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

[DC10] asks whether these problems are limited to cyclotomics, and [CD10] asks about explicitly encoding cyclotomics, so this would be $\gcd(C_{pq}, C_p C_q) = \frac{C_p C_q}{C_1}$. **Both are open problems**.

Polynomial Factorisation

1. Standard algorithm [Zas69] has exponential worst-case behaviour [SD69].
2. [Col79] conjectured polynomial average*-time.
3. Polynomial-time for dense encodings [LLL82], but a very bad exponent.
4. Exponentially larger output for sparse encodings: $x^p - 1 = (x-1)(x^{p-1} + \cdots + 1)$
5. presumably NP-hard (even in terms of output size) for sparse.

Verification is a challenge: one can check that the result is *a factorisation*, but checking completeness (i.e. that these factors are irreducible) seems to be as hard as the original problem (in the worst case).

Polynomial Factorisation: Reality

With probability 1, a random polynomial is irreducible, so what are the *interesting* problems?

* Therefore [Col79] conjecture needed a subtle definition of "average" time
* [ABD88] points out that "difficult" examples arise from factoring over algebraic number fields, which happen in "real life".

Hence An "engineering" task of switching between "exponential but generally quick" and "slow but guaranteed polynomial".

– No standard database: trawl previous papers (and often need to ask the authors)

So what constitutes a good benchmark for such tasks?

Gröbner Bases

- Doubly exponential (w.r.t. n) worst-case complexity [MR13], even if a prime ideal [Chi09].

\+ There is a collection [BM96]

– Very old (1996) and completely static.

– – Some examples only in PDF.

Again The generic case is not interesting: Shape Lemma [BMMT94].

? No concept of UNSAT, but it's not clear what a certificate might mean.

Real Algebraic Geometry (CAD)

- Doubly exponential (w.r.t. n) worst-case complexity [BD07]

$+$ There is a collection [Wil14]

$-$ Somewhat old (2014) and completely static.

$\surd$ The DEWCAD project [BDE$^+$21] might update this, but still issues of long-term conservation.

? Format: text, Maple and QEPCAD

? No concept of UNSAT (but see [KAED21]), and it's not clear what a "certificate" might mean.

Real Algebraic Geometry (CAD)

- Let a be the number of alternations of quantifiers.
- [BD07] has $a = n - 1$.
- There are (unimplemented) algorithms for quantifier elimination which are $d^{\mathrm{poly}(n)2^a}$, and implemented algorithms which *might* have similar complexity.
- The world of software, or cyber-physical systems, tends to produce examples with $a = 0$, i.e. $\exists$geometry : bad(geometry)

? Are there *natural* examples for quantifier elimination with a non-trivial?

Integration

- Complexity is essentially unknown (but certainly involves g.c.d., factorisation etc.)
- A new question here is the "niceness" of the output.
- "Paper" mathematics produced large databases, e.g. [GR07].
  – PDF, and the devil to scan.
- Current best database is described in [JR10].
- Algorithm-based software (e.g. [Dav81]) has an internal proof of UNSAT, but I know of no software that can exhibit it.

~~Black swans~~ [AL17]

"Average-case complexity without the black swans": i.e. without an exponentially-rare family that is worse than exponentially bad.

Definition

For $k \in \mathbf{N}$ let (M_k, μ_k) be a probability space and let $T_k : M_k \to \mathbb{R}$ be a μ_k-measurable function. We say that the family $\{T_k\}$ has a weak expectation of $O(f(k))$ if there exists a family of sets of exceptional inputs, $E_k \subseteq M_k$, such that $\mu_k(E_k) = e^{-\Omega(k)}$ and the conditional expectation $E[T_k(x) | x \notin E_k]$ is bounded by $O(f(k))$.

- Condition numbers inversely proportional to a distance to a homogeneous algebraic set of ill-posed inputs;
- Renegar's condition number for conic optimization;
- The running time of power iteration for computing a leading eigenvector of a Hermitian matrix.

Guesses

Guess

Gröbner basis computation is weakly singly exponential in n.

Needs a measure on sets of polynomial inputs, or possibly a measure on ideals. Might need some further restrictions e.g. to radical ideals (but see [Chi09])

Guess

Real Quantifier Elimination is weakly singly exponential in n for any a/n ratio.

Needs a measure on inputs, I think.

Conclusions

1. We need to be more inventive around complexity theory
2. The field of computer algebra really ought to invest in the sort of contests that have stimulated the SAT and SMT worlds.
3. This requires much larger databases of “relevant” problems than we currently have, and they need to be properly curated.

+ Technology, e.g. wikis, or GitHub, has greatly advanced since [BM96].

4. These collections would allow much better benchmarking technology [BDG17].

Bibliography I

J.A. Abbott, R.J. Bradford, and J.H. Davenport.
Factorisation of Polynomials: Old Ideas and Recent Results.
In R. Janssen, editor, *Proceedings "Trends in Computer Algebra"*, pages 81–91, 1988.

D. Amelunxen and M. Lotz.
Average-case complexity without the black swans.
J. Complexity, 41:82–101, 2017.

D. Achlioptas and Y. Peres.
The threshold for random k-SAT is $2^k \log 2 - O(k)$.
J. Amer. Math. Soc., 17:947–973, 2004.

Bibliography II

D. Achlioptas and Y. Peres.
Random *k*-SAT: Two Moments Suffice to Cross a Sharp Threshold.
SIAM J. Comput., 36:740–762, 2006.

T. Balyo and L. Chrpa.
Using algorithm configuration tools to generate hard SAT benchmarks.
In *Eleventh Annual Symposium on Combinatorial Search*, pages 133–137, 2018.
URL: https://algo2.iti.kit.edu/balyo/papers/socs18.pdf.

Bibliography III

S. Buswell, O. Caprotti, D.P. Carlisle, M.C. Dewar, M. Gaëtano, M. Kohlhase, J.H. Davenport, and P.D.F. Ion.
The OpenMath Standard 2.0 Revision 1.
http://www.openmath.org, 2017.

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.

Bibliography IV

R.J. Bradford, J.H. Davenport, M. England, A. Sadeghimanesh, and A. Uncu.
The DEWCAD Project: Pushing Back the Doubly Exponential Wall of Cylindrical Algebraic Decomposition.
To appear in ACM Comm. Computer Algebra, 2021.
URL: https://arxiv.org/abs/2106.08740.

M.N. Brain, J.H. Davenport, and A. Griggio.
Benchmarking Solvers, SAT-style.
SC^2 2017 Satisfiability Checking and Symbolic Computation CEUR Workshop, 1974(RP3):1–15, 2017.
URL: http://ceur-ws.org/Vol-1974/RP3.pdf.

Bibliography V

E.R. Berlekamp.
Factoring Polynomials over Large Finite Fields.
Math. Comp., 24:713–735, 1970.

C. Barrett, P. Fontaine, and C. Tinelli.
The SMT-LIB Standard: Version 2.6.
http://smtlib.cs.uiowa.edu/papers/smt-lib-reference-v2.6-r2017-07-18.pdf, 2017.

R.E. Bixby.
Computational Progress in Linear and Mixed Integer Programming.
Presentation at ICIAM 2015, 2015.
URL: http://staff.bath.ac.uk/masjhd/Others/Bixby2015a.pdf.

Bibliography VI

G. Butler and J. McKay.
The transitive groups of degree up to 11.
Comm. Algebra, 11:863–911, 1983.

D. Bini and B. Mourrain.
Polynomial test suite.
`http://www-sop.inria.fr/saga/POL/`, 1996.

E. Becker, M.G. Marinari, T. Mora, and C. Traverso.
The shape of the shape lemma.
In *Proceedings ISSAC 1994*, pages 129–133, 1994.

G. Butler.
The transitive groups of degree fourteen and fifteen.
J. Symbolic Comp., 16:413–422, 1993.

Bibliography VII

J. Carette and J.H. Davenport.
The Power of Vocabulary: The Case of Cyclotomic Polynomials.
http://arxiv.org/abs/1002.0012, 2010.

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.

Bibliography VIII

G.E. Collins.
Factoring univariate integral polynomials in polynomial average time.
In *Proceedings EUROSAM 79*, pages 317–329, 1979.

S.A. Cook.
On the minimum computation time of functions.
PhD thesis, Department of Mathematics Harvard University, 1966.

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.

Bibliography IX

J.H. Davenport.
The Role of Benchmarking in Symbolic Computation.
In *Proc. SYNASC 2018*, pages 275–279, 2018.
URL:
https://researchportal.bath.ac.uk/en/publications/
the-role-of-benchmarking-in-symbolic-computation-positi
doi:10.1109/SYNASC.2018.00050.

J.H. Davenport and J. Carette.
The Sparsity Challenges.
In S. Watt *et al.*, editor, *Proceedings SYNASC 2009*, pages 3–7, 2010.

Bibliography X

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.

I.S. Gradshteyn and I.M. Ryzhik.
Table of Integrals, Series and Products 7th edition (ed. A. Jeffrey and D. Zwillinger).
Academic Press, 2007.

I.P. Gent and T. Walsh.
The SAT phase transition.
ECAI, 94:105–109, 1994.

Bibliography XI

M.J. Heule, M. Kauers, and M. Seidl.
New ways to multiply 3×3-matrices.
Journal of Symbolic Computation, 104:899–916, 2021.

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

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

D.J. Jeffrey and A.D. Rich.
Reducing Expression Size Using Rule-Based Integration.
In S. Autexier *et al.*, editor, *Proceedings CICM 2010*, pages 234–246, 2010.

Bibliography XII

G. Kremer, E. Ábrahám, M. England, and J.H. Davenport.
On the Implementation of Cylindrical Algebraic Coverings for Satisfiability Modulo Theories Solving.
Proc. SYNASC 2021, pages 37–39, 2021.

W. Küchlin 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.

Bibliography XIII

A.K. Lenstra, H.W. Lenstra Jun., and L. Lovász.
Factoring Polynomials with Rational Coefficients.
Math. Ann., 261:515–534, 1982.

M.W. Moskewicz, C.F. Madigan, Y. Zhao, L. Zhang, and S. Malik.
Chaff: Engineering an Efficient SAT Solver.
In *Proceedings 38th Design Automation Conference*, pages 530–535, 2001.

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.

Bibliography XIV

G.E. Moore.
Progress in Digital Integrated Electronics.
International Electron Devices Meeting, pages 11–13, 1975.

E.W. Mayr and S. Ritscher.
Dimension-dependent bounds for Gröbner bases of polynomial ideals.
J. Symbolic Comp., 49:78–94, 2013.

D.A. Plaisted.
New NP-Hard and NP-Complete Polynomial and Integer Divisibility Problems.
Theor. Comp. Sci., 31:125–138, 1984.

Bibliography XV

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.

Bibliography XVI

N.J.A. Sloane.
The Online Encyclopedia of Integer Sequences.
Notices A.M.S., 50:912–915, 2003.

I. Spence.
Weakening Cardinality Constraints Creates Harder Satisfiability Benchmarks.
J. Exp. Algorithmics Article 1.4, 20, 2015.

The GAP Group.
GAP — Groups, Algorithms, and Programming, Version 4.11.1.
`https://www.gap-system.org`, 2021.

Bibliography XVII

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.

D.J. Wilson.
Advances in Cylindrical Algebraic Decomposition.
PhD thesis, University of Bath, 2014.

H. Zassenhaus.
On Hensel Factorization I.
J. Number Theory, 1:291–311, 1969.