

# So the problem has poor complexity: what next?

James Davenport  
masjhd@bath.ac.uk

University of Bath  
Partially supported by EPSRC under grant EP/T015713

12 September 2023

# Complexity Theory is wonderful

Though we talk about complexity of problems, we should be more precise and talk about the (problem, encoding) pair.

## Definition (upper bound)

If  $\exists$  algorithm  $A$  which solves all instances of problem  $P$  whose  $E$ -encoding takes at most  $n$  bits, taking at most  $f(n)$  operations of type  $T$ , then the  $T$ -complexity of  $(P, E)$  is at most  $f(n)$ .

$A$  is a witness to this upper bound.  $f$  is often a  $\mathcal{O}$  expression.

## Definition (lower bound)

If we can prove that any such algorithm  $A$  (which solves all instances of problem  $P$  whose  $E$ -encoding takes at most  $n$  bits) must take at least  $g(n)$  operations of type  $T$  on one such instance  $I(n)$ , then the  $T$ -complexity of  $(P, E)$  is at least  $g(n)$ .

$I(n)$  is the witness.  $g$  is often a  $\Omega$  expression.  
Implicitly, these are *worst case* bounds.

# Strong/Weak points of Complexity Theory: [AL17]

“Depending on context and tradition, a computational problem can mean something practical that begs to be solved as efficiently as possible, or a mathematical object in its own right, to be analysed, classified, and understood. In the first sense, the aim is to develop methods that work well on problems of interest, while in the second, complexity-theoretic sense, algorithms are merely devices used to show that a problem can be solved within certain resource constraints, e.g., in a certain complexity class or with a running time bounded by some function of the input size. Needless to say, complexity-theoretic results are often only weakly correlated with practical experience; a typical example is the simplex method<sup>1</sup>. This is particularly true for numerical problems, where often a condition number serves as a proxy to computational complexity”. Another example is SAT-Solving: the quintessential NP-Complete problem [Coo66], yet since about 2000 every German car has been manufactured with the aid of a SAT solver (say 100M such solutions) [KS00].

<sup>1</sup>But see [Bix15, slide 31].

In general, non-trivial lower bounds are hard to achieve.

- 1 Reduction to a previous problem.  
Notably the theory of NP-complete problems [Coo66]
  - 2 The number of possibilities (sorting being  $\Omega(n \log_2 n)$ , for example)
  - 3 The size of the output (matrix multiplication being  $\Omega(n^2)$ )
- + “CAD is doubly exponential [DH88, BD07]” is of this flavour.  
The ingenuity consists in finding a bad family  $I(n)$ .

# Beyond simple bit-length

We are already used in computer algebra (and elsewhere) at looking at finer descriptions than the bit-length. Indeed we rarely look at the raw bit length. We can have “dense univariate polynomials of degree  $d$  and coefficients bounded by  $2^H$ ”, where the bitsize is  $(d + 1)(H + 1)$ , but often the behaviour for  $d$  and  $H$  is different.

If we move to (dense) polynomials in  $n$  variables, the bitsize is now  $(d + 1)^n(H + 1)$  (slightly different if  $d$  is total degree bound). Kronecker-substitutions (replacing  $x_2$  by  $x_1^{d+1}$  etc.) can give a  $d/n$  tradeoff. Sometimes a  $d/H$  tradeoff as well.

If we look at sparse polynomials with  $t$  terms, the bitsize is  $\log(d)t(H + 1)$  (or  $n \log(d)t(H + 1)$  for multivariates).



Note that dependence on  $n$  is now linear: of course  $t$  may be exponential in  $n$ .

# Output-sensitive bounds

With sparse polynomials another problem arises: that of connecting output size to input size. For sparse problems there is often a bad worst-case connection, e.g.  $\frac{x^n-1}{x-1} = x^{n-1} + \dots + x + 1$  with  $n$  terms.

The first breakthrough was [BOT88] in interpolation, which was polynomial in  $T$  a bound for output number of terms (and polynomial in  $\log d$ ).

Hence the solution is *output-sensitive bounds*, where we assume  $T$  bounds the number of terms in the inputs and outputs. There has been much good work in this area since our talk at my first SYNASC [DC09].

Note that linear in  $T$  is as good as one can get, and in 2009 we thought that polynomial in  $T$  would be a major improvement.

## Theorem ([GGdCR22, Theorem 1.3])

*There is a Monte Carlo randomized algorithm that, given two sparse polynomials  $f, g \in \mathbf{Z}[x_1, \dots, x_n]$  such that  $g$  divides  $f$  and a bound  $T$  on the sparsity of the quotient  $f/g$ , computes the sparse representation of  $f/g$  with probability at least  $2/3$ . It requires  $\tilde{O}((T + \#f + \#g)(n \log D + \log H))$  bit operations where  $D = \deg(f)$ , and  $H$  is a bound on the height of the three polynomials  $f$ ,  $g$  and  $f/g$ .*

Note that we have to supply  $T$ , but if we're wrong, we double and retry: doesn't affect  $\tilde{O}$ .

Should output-sensitive bounds be used more widely than basic sparse polynomial operations?

One obvious solution to the fact that “worst-case complexity” is precisely that, is to look at “average-case complexity”. The challenge then is to ask what metric we place on “average”. An obvious answer would be “all bit patterns of length  $n$  have probability  $2^{-n}$ ”, but this ignores the tedious (but generally not fundamental) issue that not all such bit patterns are valid for encoding  $E$ .

More seriously, not all such bit patterns may represent “interesting” objects. A classic case is (dense) polynomial factorisation. Here it is the case that almost all polynomials do not factor. In fact a stronger statement is true: almost all have Galois group  $S_N$  and so are easily proved irreducible [Mus75, LP97].

A more serious issue is that the representations may have essential equivalences, e.g.  $f(x)$ ,  $f(-x)$ ,  $f(x+1)$  . . . . We might wish to consider the number field, rather than the actual polynomial, and here the results are very different [Mal02]. So “which average” matters.



“Average case complexity” can be ruined by an exponentially small set of “bad cases” if they are exponentially worse than the typical case.

## Definition ([AL17, Definition 1.1.])

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

Perhaps we can also talk about “weak worst case” complexity as well, where the true worst case is exponentially rare.

# Parameterized Complexity [DF99]

This came on the scene long after computer algebra developed its own methodologies, and is rarely referred to. The idea is that, rather than just looking at the bitsize, we look at more refined parameters of the problem, which may or may not be evident.

**evident** We have already seen this with polynomials, where we look at  $d$  and  $H$  rather than  $n$ .

**retrospectively evident** this includes the “output sensitive” sparse polynomial algorithms, and there is probably more to be done here.

**not evident** in the traditional theory, many graph algorithms are exponential in the tree width, but polynomial in everything else for a fixed tree width.

**Gröbner** Solving zero-dimensional systems, where the complexity depends on the solving degree. This appears towards the end of the solving procedure, and its computation is believed to be essentially equivalent to complete solving.

## Conjecture (Distinctly Optimistic)

*The doubly-exponential behaviour of Gröbner bases [MR10] is exponentially rare, i.e. the weak average complexity of Gröbner bases is singly exponential in the number of variables.*

## Conjecture (Distinctly Optimistic)

*The same is true for Comprehensive Gröbner systems.*

## Conjecture (Even More Optimistic)

*Real Quantifier Elimination by Comprehensive Gröbner Systems [Wei98] has weak average complexity singly exponential in the number of variables (but doubly exponential in the number of alternations).*

Otherwise “How well does it do on interesting examples”: often subtly “the examples which interest me”, from which it’s a small step to “the examples which put my ideas in a good light”, often unconsciously.

Here we should learn from the SAT and SMT communities: large curated collections of “interesting” problems. This isn’t trivial. [HHP<sup>+</sup>23] point out that the original MetiTarski dataset, often used in the real geometry area, wasn’t symmetric in variables, to the point where “choose  $x_3$ ” was a disproportionately effective strategy. A different problem is that the MetiTarski dataset, even before its augmentation to solve the bias issue, was far larger than the other sources of examples in SMTLIB.

Nevertheless, we shouldn’t let these challenges get in the way of doing much better than we currently do at Empirical Complexity.

This isn't easy, but we have some useful examples in the SAT and SMT communities. As said yesterday ([AZ23]), “benchmarking is difficult”. We can state some key points (see also [Dav18]).

- A large set of examples.



Too many computer algebra papers have single-figure numbers.

- Proper de-biasing ([HHP<sup>+</sup>23]), where we can probably also take lessons from “AI”, and balancing (this is a human decision, based on “representativeness”).
- Independently run (as in [AZ23]).
- Verified answers (and a penalty for errors).
- Ideally, a similar set of rules etc. from year to year.

# Doing well on subsets

This could be regarded as declaring whole subsets “interesting”, or at least “useful” (German cars). Here are some examples from quantifier elimination.

- examples with few alternations. Platzer claims that  $\forall\exists\forall$  is the most complex structure of interest to him, but each quantifier might have many variables.
- linearity (or multilinearity). It has been observed that QE is unreasonably (in terms of the number of variables) effective in examples coming from economics. This seems to be due to the fact that most of the expressions are multilinear.



[BD07] shows that linear with many alternations can still be doubly exponential.

- Virtual Term Substitution can be a powerful tool with variables occurring linearly. It makes a good pre-processor or integrated option [Ton21, DTU23].

Let  $a$  be the number of alternations, which we wish to treat as a parameter in the sense of Parametric Complexity. Let  $n$  be the number of actual variables (after unwinding any recycling). Then both [DH88]  $n \approx 5a$  and [BD07]  $n \approx 3a$  give  $a$  as a lower bound on the double exponent, whereas all cylindrical algebraic decomposition methods have  $n + O(1)$  as the upper bound.  $\Theta(n)$ , which we have, is pretty imprecise as a double exponent! Either we need to find better lower bounds, or conclude that the current cylindrical algebraic decomposition techniques need substantial improvement.

Then we can write this in disjunctive normal form, and commute  $\exists$  and  $\bigvee$

$$\bigvee_i \exists x_{k+1} \dots x_n \bigwedge_j P_{i,j}(x_1, \dots, x_n) \sigma_{i,j} 0.$$

This may cause exponential blowup in the DNF conversion.

Certainly if  $k = 0$  we can do better than cylindrical algebraic decomposition on a single  $\exists x_{k+1} \dots x_n \bigwedge_j P_{i,j}(x_1, \dots, x_n) \sigma_{i,j} 0$ .

However, if we pay the exponential blowup in the DNF conversion, there is a lot of commonality in the

$\exists x_{k+1} \dots x_n \bigwedge_j P_{i,j}(x_1, \dots, x_n) \sigma_{i,j} 0$  terms, which a cylindrical algebraic decomposition can exploit.

Complexity Theory isn't easy!



# So what do to? (I)

Obviously I can't tell you but you might wish to think about these.

① Have I got the right encoding?



“Sparse” is generally harder than “dense”, often done probabilistically

② If my problem is perverse examples with output blowup, should I be looking for output-sensitive complexity?



To do so, you must have a truly output-sensitive algorithm, i.e. it must be fast when the input and output are small.

③ Perhaps I should have a different encoding for input and output



That might work, depending on the problem. “Low degree factors of sparse polynomials” is a good case.

④ Should I be thinking of average case complexity?



This is generally harder, unless you have some good probability theory on your problem

## So what do to? (II)

- ⑤ Maybe I'll just do empirical complexity.
  - !! Fine, but try to do it well: collecting good problem sets is hard, and *do* try to archive them well (*not* just as PDF!).
  - ?? and maybe you'll discover other parameters, such as perfect elimination orderings [CP16]



D. Amelunxen and M. Lotz.

Average-case complexity without the black swans.

*J. Complexity*, 41:82–101, 2017.



H.S. Aliefendioglu and Z. Zafeirakopoulos.

Experimental comparison of real root isolation implementations.

Presentation at SYNASC2023, 2023.



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.

doi:10.1145/1277548.1277557.



R.E. Bixby.

Computational Progress in Linear and Mixed Integer Programming.

*Presentation at ICIAM 2015, 2015.*

URL: [http:](http://staff.bath.ac.uk/masjhd/Others/Bixby2015a.pdf)

[//staff.bath.ac.uk/masjhd/Others/Bixby2015a.pdf](http://staff.bath.ac.uk/masjhd/Others/Bixby2015a.pdf).



M. Ben-Or and P. Tiwari.

A deterministic algorithm for sparse multivariate polynomial interpolation.

*In Proceedings 20th. Symp. Theory of Computing, pages 301–309, 1988.*



S.A. Cook.

*On the minimum computation time of functions.*

PhD thesis, Department of Mathematics Harvard University, 1966.



D. Cifuentes and P. Parrilo.

Exploiting chordal structure in polynomial ideals: A Grobner bases approach.

*SIAM Journal on Discrete Mathematics*, 30:1534–1570, 2016.



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-positioning-the-problem-has-poor-complexity-what-next?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, 2009.

# Bibliography IV

-  R.G. Downey and M.R. Fellows.  
Parameterized Complexity.  
*Springer*, 1999.
-  J.H. Davenport and J. Heintz.  
Real Quantifier Elimination is Doubly Exponential.  
*J. Symbolic Comp.*, 5:29–35, 1988.
-  J.H. Davenport, Z.P. Tonks, and A.K. Uncu.  
A Poly-algorithmic Approach to Quantifier Elimination.  
<https://arxiv.org/abs/2302.06814>, 2023.
-  Pascal Giorgi, Bruno Grenet, Armelle Perret du Cray, and Daniel S. Roche.  
Sparse Polynomial Interpolation and Division in Soft-linear Time.  
In Amir Hashemi, editor, *ISSAC '22: Proceedings of the 2022 International Symposium on Symbolic and Algebraic*

*Computation*, pages 459–468, New York, NY, USA, 2022.  
Association for Computing Machinery.



J. Hester, B. Hitaj, G. Passmore, S. Owre, N. Shankar, and E. Yeh.

An Augmented MetiTarski Dataset for Real Quantifier Elimination Using Machine Learning.

In Catherine Dubois and Manfred Kerber, editors, *Proceedings CICM 2023*, volume 14101 of *Springer Lecture Notes in Computer Science*, pages 297–302, 2023.



W. Küchlin and C. Sinz.

Proving Consistency Assertions for Automotive Product Data Management.

*J. Automated Reasoning*, 24:145–163, 2000.



T. Łuczak and L. Pyber.

On random generation of the symmetric group.

In *Proceedings Combinatorics geometry and probability*, pages 463–470, 1997.



G. Malle.

On the Distribution of Galois Groups.

*J. Number Theory*, 92:315–329, 2002.



E.W. Mayr and S. Ritscher.

Degree Bounds for Gröbner Bases of Low-Dimensional Polynomial Ideals.

In S.M. Watt, editor, *Proceedings ISSAC 2010*, pages 21–28, 2010.



D.R. Musser.

Multivariate Polynomial Factorization.

*J. ACM*, 22:291–308, 1975.





Z. Tonks.

*Poly-algorithmic Techniques in Real Quantifier Elimination.*

PhD thesis, University of Bath, 2021.

URL: [https:](https://researchportal.bath.ac.uk/en/studentTheses/poly-algorithmic-techniques-in-real-quantifier-elimination)

[//researchportal.bath.ac.uk/en/studentTheses/  
poly-algorithmic-techniques-in-real-quantifier-elimination](https://researchportal.bath.ac.uk/en/studentTheses/poly-algorithmic-techniques-in-real-quantifier-elimination)



V. Weispfenning.

A New Approach to Quantifier Elimination for Real Algebra.

In B.F. Caviness and J.R. Johnson, editors, *Quantifier Elimination and Cylindrical Algebraic Decomposition*, pages 376–392. Springer-Verlag, 1998.