Experience with Heuristics, Benchmarks & Standards for Cylindrical Algebraic Decomposition

Matthew England & James Davenport
Matthew.England@coventry.ac.uk J.H.Davenport@bath.ac.uk
http://www.sc-square.org[1]
Coventry University University of Bath

24 September 2016

[1] SC^2: H2020-FETOPEN-2016-2017-CSA project 712689

Motivation

In the SC^2 motivational paper [Á15], the author identified the use of sophisticated heuristics as a technique that the Satisfiability Checking community excels in and from which it is likely the Symbolic Computation community could learn and prosper.
That author was undoubtedly right here: not so much that heuristics are unknown in Symbolic Computation, as that the Symbolic Computation *literature* largely ignores heuristics, so that they are hidden secrets.
So the present authors thought they should open up.

Reality of systems (as opposed to literature)

The top-level commands of most computer algebra systems are often heuristics.

The speaker cut his teeth on making integration algorithmic: nevertheless

- Maple's definite integrator [Inc16] contains a list of eight methods, which are currently tried in a fixed order;
- A purely rule-based integrator [JR16] produces "better" (Size of integral (as an expression); Continuity; Real versus complex; Aesthetics) results than algorithms.

Cylindrical Algebraic Decomposition

A workhorse of Real {Nonlinear Arithmetic, Algebraic Geometry} with numerous approaches for problems in *n* variables

Projection/Lifting [Col75, Eng13]

Regular Chains [CM14a]

But there are other approaches to Quantifier Elimination

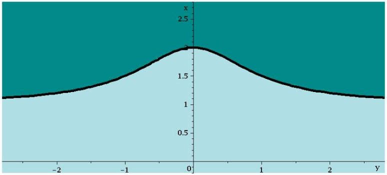
Virtual Term Substitution [KSD16]

Comprehensive Gröbner Bases [FIS16]

Every approach has in fact many choices within it, which can affect both the running time *and* the actual answer: [BD07] shows the size of the answer can vary doubly-exponentially (in *n*).

Variable Order

The obvious one (within the constraints of the problem).



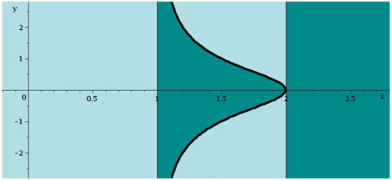


Figure: CADs under different variable orderings: 3 cells versus 11 cells

It's also the [BD07] doubly-exponential difference case

Various heuristics (somewhat documented! PL-CAD)

Brown [Bro04, Section 5.2]. Use the following criteria, starting with the first and breaking ties with rest:
(1) Eliminate variable if lowest overall degree.
(2) Eliminate variable if lowest (maximum) total degree in terms in which it occurs.
(3) Eliminate variable if smallest number of terms contains it.

sotd [DSS04] For all admissible orderings, calculate the projection set and choose the one with smallest *sum of total degrees* for each of the monomials in each of the polynomials

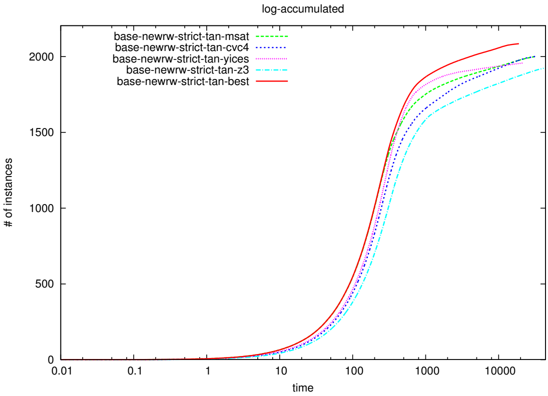
greedy [DSS04] As above, but choose the first variable only, then the second . . .

ndrr [BDEW13] construct the full projection set and choose the ordering whose set has the least *number of distinct real roots* of the univariate polynomials.

Compared in [HEW+14], across 7001 3-variable problems from NLSAT.

- **Brown** Is the best heuristic most often
- **sotd** makes the most savings

So what is the best definition of "best"? Survivor plots?



Other choices: Which equational constraint to pick

- It does matter: [EBD15] shows $\times 16$ variation in cell count
- No cheap heuristic (as the polynomials are the same)
- sotd and ndrr both do reasonably, but these are expensive heuristics
- What's the metric/comparison?

Same problem for Truth-Table Invariant CAD

Various heuristics (documented) RC-CAD

Brown As before

Triangular Simlar

sotd You do the P/L projections, work out the variable order, then throw the projections away

ndrr ditto

New [EBDW14] Degrees as in Brown, tiebreak by calculating principal next-stage polynomials (and possibly refined tie-break)

On a small sample New+ is best

Incremental Algorithms

Both the tme, and indeed the results, for such methods (e.g.

[CM14b] for CAD by regular chains (see [EBC+14])

[BK15] for constructing a single cell)

can easily be dependent on the *order* in which we present the polynomials. In general it is not obvious what heuristics to use here: lowest degree first seems obvious.

[EBC+14] suggested doing the complete complex projection for each order, then refining the one with lowest sotd (one could also consider ndrr): again expensive.

The same problem, four orders [EBC+14]

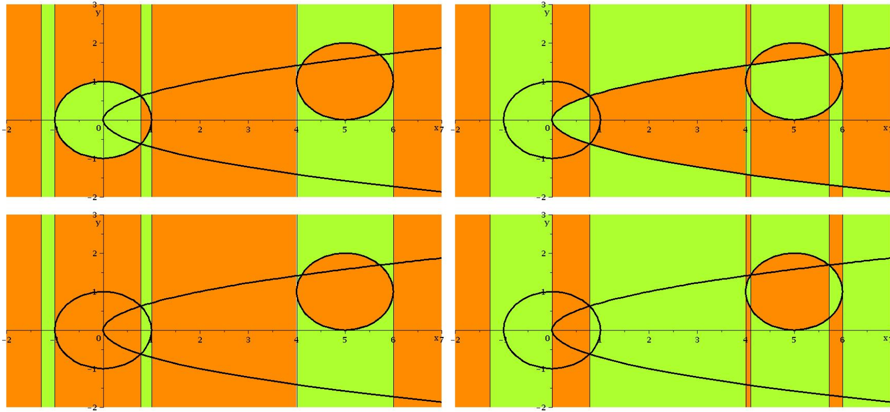


Figure:

37	81
25	45

cells

Gröbner preconditioning for CAD

Two flavours: basic and enhanced (both work with n formula, simple cases shown)

1. Replace $f_1 = 0 \wedge f_2 = 0 \wedge \cdots$ by $\bigwedge_{f \in GB(f_1, f_2)} f = 0 \wedge \cdots$ [BH91]

1991 10 examples: sped up 6, slowed 2 (one very slow GB), and 2 impossible

2012 GB was much faster, but similar conclusions

2. Also, replace $\bigwedge_{f \in G} \wedge g < 0 \wedge \cdots$ by $\bigwedge_{f \in G} \wedge \widehat{g} < 0 \wedge \cdots$ where $g \rightarrow_G \widehat{g}$ [WBD12]

So when should we do it?

Machine Learning?

See [HEDP16] and Matthew England's talk tomorrow 10:10.

Note that we have been using Machine Learning as a meta-heuristic: to decide which existing heuristic to apply.

But machine learning needs large benchmark sets, and these need to have standard representation.

E. Ábrahám.
Building Bridges between Symbolic Computation and Satisfiability Checking.
In D. Robertz, editor, *Proceedings ISSAC 2015*, pages 1–6, 2015.

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.

R.J. Bradford, J.H. Davenport, M. England, and D.J. Wilson.
Optimising Problem Formulation for Cylindrical Algebraic Decomposition.
In J. Carette *et al.*, editor, *Proceedings CICM 2013*, pages 19–34, 2013.

B. Buchberger and H. Hong.
Speeding-up Quantifier Elimination by Gröbner Bases.
Technical Report 91-06, 1991.

C. Brown and M. Košta.
Constructing a single cell in cylindrical algebraic decomposition.
J. Symbolic Computation, 70:14–48, 2015.

C.W. Brown.
Tutorial handout.
http://www.cs.usna.edu/~wcbrown/research/ISSAC04/handout.pdf, 2004.

C. Chen and M. Moreno Maza.
An Incremental Algorithm for Computing Cylindrical Algebraic Decompositions.
In Ruyong Feng, Wen-shin Lee, and Yosuke Sato, editors, *Computer Mathematics*, pages 199–221. Springer Berlin Heidelberg, 2014.

C. Chen and M. Moreno Maza.
An Incremental Algorithm for Computing Cylindrical Algebraic Decompositions.
In R. Feng, W.-S. Lee, and Y. Sato, editors, *Proceedings Computer Mathematics ASCM 2009 and 2012*, pages 199–221, 2014.

G.E. Collins.
Quantifier Elimination for Real Closed Fields by Cylindrical Algebraic Decomposition.
In *Proceedings 2nd. GI Conference Automata Theory & Formal Languages*, pages 134–183, 1975.

A. Dolzmann, A. Seidl, and Th. Sturm.
Efficient Projection Orders for CAD.
In J. Gutierrez, editor, *Proceedings ISSAC 2004*, pages 111–118, 2004.

M. England, R. Bradford, C. Chen, J.H. Davenport, M.M. Maza, and D.J. Wilson.
Problem formulation for truth-table invariant cylindrical algebraic decomposition by incremental triangular decomposition.
In S.M. Watt *et al.*, editor, *Proceedings CICM 2014*, pages 45–60, 2014.

M. England, R. Bradford, and J.H. Davenport.
Improving the Use of Equational Constraints in Cylindrical Algebraic Decomposition.
In D. Robertz, editor, *Proceedings ISSAC 2015*, pages 165–172, 2015.

M. England, R. Bradford, J.H. Davenport, and D.J. Wilson.
Choosing a Variable Ordering for Truth-Table Invariant Cylindrical Algebraic Decomposition by Incremental Triangular Decomposition.
In *Proceedings ICMS 2014*, pages 450–457, 2014.

M. England.
An Implementation of CAD in Maple Utilising McCallum Projection.
Technical Report 2013-02 Bath Computer Science, 2013.

R. Fukasaku, H. Iwane, and Y. Sato.
On the Implementation of CGS Real QE.
In *Proceedings ICMS 2016*, pages 165–172, 2016.

Z. Huang, M. England, J.H. Davenport, and L.C. Paulson.
Using Machine Learning to Decide When to Precondition Cylindrical Algebraic Decomposition With Groebner Bases.
https://arxiv.org/abs/1608.04219, 2016.

Z. Huang, M. England, D. Wilson, J.H. Davenport, L.C. Paulson, and J. Bridge.
Applying machine learning to the problem of choosing a heuristic to select the variable ordering for cylindrical algebraic decomposition.
In S.M.Watt *et al.*, editor, *Proceedings CICM 2014*, pages 92–107, 2014.

Maplesoft Inc.
A Discussion of the Practical Issues of Computing Integrals in Maple.
Presentation at ICMS 2016 (Berlin): see §3.8 http://staff.bath.ac.uk/masjhd/Meetings/JHDatICMS2016.pdf, 2016.

D.J. Jeffrey and A. Rich.
Recent Developments in the RUBI Integration Project.
Presentation at ICMS 2016 (Berlin): see §3.9 http://staff.bath.ac.uk/masjhd/Meetings/JHDatICMS2016.pdf, 2016.

M. Košta, T. Sturm, and A. Dolzmann.
Better answers to real questions.
J. Symbolic Comp., 74:255–275, 2016.

D.J. Wilson, R.J. Bradford, and J.H. Davenport.
Speeding up Cylindrical Algebraic Decomposition by Gröbner Bases.
In J. Jeuring *et al.*, editor, *Proceedings CICM 2012*, pages 279–293, 2012.