Speeding up Cylindrical Algebraic Decomposition by Means of Gröbner Bases.

James Davenport (with David Wilson, Russell Bradford)[1]
University of Bath; J.H.Davenport@bath.ac.uk

13 July 2012

[1]Special Mention: Matthew England

Otherwise Known As

"If all you have is a hammer, all your problems look like nails."

"I have this hammer (Cylindrical Algebraic Decomposition): which window should I break?"

More prosaically: "Issues in Problem Formulation".

Illustration: Linear Algebra

To solve linear system, first put them in upper triangular form: Sure, but with

$$\begin{pmatrix} 9 & 0 & 0 & 0 & 0 & 0 \\ 8 & 7 & 0 & 0 & 0 & 0 \\ 7 & 6 & 5 & 0 & 0 & 0 \\ 6 & 5 & 4 & 3 & 0 & 0 \\ 5 & 4 & 3 & 2 & 1 & 0 \\ 4 & 3 & 4 & 3 & 2 & 1 \end{pmatrix}$$

would you really do this? Certainly not "by hand".

Quantifier Elimination

Given a real Tarski formula:

$$(Q_k x_k)(Q_{k+1} x_{k+1}) \cdots (Q_n x_n) \Phi(x_1, \ldots, x_n) \qquad (P)$$

where each Q_i is either $\forall$ or $\exists$ and Φ is quantifier free, produce a quantifier free equivalent formula $\Psi(x_1, \ldots, x_{k-1})$.
Note that $\forall x \forall y \equiv \forall y \forall x$ so we are really only interested in **blocks** of quantifiers.
Known to be doubly exponential in number of blocks [DH88]

Cylindrical Algebraic Decomposition (CAD) — Collins1975

Input: $P(x_1, \ldots, x_n)$

- **project** x_n to get $P_{n-1}(x_1, \ldots, x_{n-1})$, then $x_{n-1}, \ldots, x_2$
- **base** solve resulting equations (UP) in x_1 alone, with N roots and $N+1$ intervals
- **lift x_2** N 1-D slices and $N+1$ 2-D cylinders, each partitioned by polynomials(x_1, x_2)
- **keep lifting** $x_3, \ldots, x_n$
- **analyse** result, to get regions $(x_1, \ldots, x_{k-1})$ where formula is true.

The cost is in the lifting, but the control is in the projection.
$x_1, \ldots, x_{k-1}$ can be in any order, and order within blocks doesn't matter.

Pretty Under-determined?

Indeed so. What should we do?

Notation

For $p = \sum_{(i_1,\ldots,i_n)\in I} a_{i_1,\ldots,i_n} x_1^{i_1} \ldots x_n^{i_n}$, *define the sum of total degrees* $\texttt{sotd}(p) = \sum_{(i_1,\ldots,i_n)\in I} i_1 + \cdots + i_n$.

Observation (1)

For a given problem (P), *the time, and number of regions, for different orders is closely correlated with* $\texttt{sotd}(UP)$*. [DSS04].*

Therefore one could try all possible (legal) projections, and lift the one with least `sotd`. Note that this parallelises well

Better still

Observation (2)

For a given problem (P), *As we project* $(P_{n-1}), \ldots, (P_2), (UP)$, *the best* $\texttt{sotd}(P_k)$ *tends to come from the best* $\texttt{sotd}(P_{k+1})$*. [DSS04].*

Hence a greedy algorithm: pick the best x_n, in terms of $\texttt{sotd}(P_{n-1})$, then best x_{n-1}, ...
$O(n^2)$ projection operations, as opposed to $O(n!)$ for previous slide and $O(n)$ for the "no choice" variant.
Very effective in practice.

Cylindrical Algebraic Decomposition: Special Case

Suppose we are given a problem, which we may formulate as

$$\text{quantified variables } e_1 = 0 \wedge \cdots \wedge e_k = 0 \wedge B(f_1, \ldots, f_l), \quad (P)$$

where B is a Boolean combination of conditions $= 0, \neq 0, < 0$ etc. on some polynomials f_j. Examples

- A branch cut $\Im(f(z)) = 0 \wedge \Re(f(z)) < 0$

see ISSAC 2010 poster

- An obstacle in robotics

then we may be able, by applying Gröbner techniques to the e_j , producing $e_j^{(i)}$, and then reducing the f_j, to produce various alternative formulations

$$\text{q.v. } e_1^{(i)} = 0 \wedge \cdots \wedge e_{k^{(i)}}^{(i)} = 0 \wedge B(f_1^{(i)}, \ldots, f_l^{(i)}), \qquad (P^{(i)})$$

Which $(P^{(i)})$ should we pick?

Prior Technology [BH91]

They had much the same idea, and used state-of-the-art technology (for 1991): Gröbner bases and an early version of QEPCAD [Bro03].

We use current QEPCAD ($=_{\mathrm{Col}}$), also Maple [CMMXY09] ($<_{\mathbf{\Delta R}}$).

Unlike them, we never observed a case where the cost of Gröbner was significant compared to the CAD

Examples come from Wilson's example bank: `http://opus.bath.ac.uk/29503`.

Rerun with today's technology

Table: [BH91] Examples for full CADs

	$=_{\mathrm{Col}}$		$=_G/=_{\mathrm{Col}}$		$<_{\Delta\mathbf{R}}$		$=_G/<_{\Delta\mathbf{R}}$	
	Time	**Cells**	**Time**	**Cells**	**Time**	**Cells**	**Time**	**Cells**
I A	236	3723	99	273	29426	3763	2470	273
I B	212	3001	97	189	36262	2795	1482	189
R A	150	2101	110	105	17355	1267	570	165
R B	21091	7119	104	141	356670	7119	470	141
E A*	7390	114541	3214	53559	262623	28557	62496	14439
E B*	Error	?	Error	?	$> 1000s$	?	$> 1000s$	?
S A*	115	1751	104	297	16014	1751	2025	297
S B*	253	6091	105	243	43439	6091	1647	243
C A*	820	8387	Error	?	216028	7895	$> 1000s$	?
C B*	Error	?	Error	?	$> 1000s$	?	$> 1000s$	?

* indicates that the linear inequalities have been omitted in this version.

Now there's even more choice

Precisely which window should I use my hammer on?

- Do we Gröbner ($=_G$)?
- If so which order?
- How much reduction of inequalities etc. ($\stackrel{*}{\to}^{G}$) by the result of Gröbner?
- As well as the choice of order for CAD

Decisions, decisions, decisions!

More examples

Table: Examples from [CMMXY09]

	$<_{\Delta R}$		$=_G/<_{\Delta R}$			Ratio	
	Time	**Cells**	**Time**		**Cells**	**Time**	**Cells**
Cyclic–3	3136	381	$20 + 245 =$	265	21	11.83	18.14
Cyclic–4	$> 1000s$	?	$64 + 5813 =$	5877	621	?	?
2	2249	895	$22 + 1845 =$	1867	579	1.20	1.55
4	3225	421	$24 + 19738 =$	19762	1481	0.16	0.28
6	363	41	$20 + 918 =$	938	89	0.39	0.46
7	3667	895	$26 + 6537 =$	6563	1211	0.56	0.74
8	3216	365	$21 + 174 =$	195	51	16.49	7.16
13	14342	4949	$18 + 220 =$	238	81	60.26	61.10
14	334860	27551	$21 + 971 =$	992	423	337.56	65.13

You win some, you lose some!

sotd isn't that helpful

Table: [BH91]: degrees

	$<_{\Delta R}$			$=_G / <_{\Delta R}$		
	degrees	**Time**	**Cells**	degrees	**Time**	**Cells**
Intersection A	6 / 14	29426	3763	17 / 50	2470	273
Intersection B	6 / 14	36262	2795	15 / 41	1482	189
Random A	9 / 16	17355	1219	19 / 68	570	165
Random B	9 / 16	356670	7119	19 / 73	470	141
Ellipse A*	6 / 24	262623	28557	6 / 26	62496	14439
Ellipse B*	6 / 24	$> 1000s$	?	25 / 253	$> 1000s$	?
Solotareff A*	10 / 25	16014	1751	10 / 28	2025	297
Solotareff B*	10 / 25	43439	6091	21 / 69	1647	243
Collision A*	6 / 23	216028	7895	27 / 251	$> 1000s$	?
Collision B*	6 / 23	$> 1000s$	?	36 / 875	$> 1000s$	?

'degrees' is $\mathtt{td}(A_n)/\mathtt{sotd}(A_n)$.

$$\texttt{TNoI}(F) = \sum_{f \in F} \texttt{NoI}(f), \tag{1}$$

where $\texttt{NoI}(f)$ is the number of indeterminates present in a polynomial f.

Table: TNoI for Spheres

	$<_{\Delta \mathbf{R}}$			$=_G / <_{\Delta \mathbf{R}}$			$=_G / \xrightarrow{*}^G / <_{\Delta \mathbf{R}}$		
	TNoI	**Time**	**Cells**	TNoI	**Time**	**Cells**	TNoI	**Time**	**Cells**
S_1, S_2, C	8	8654	1073	5	905	267	4	270	99
S_2, S_3, C	8	189202	12097	6	5911	1299	6	499	213
S_3, S_4, C	8	248340	11957	7	8159	1359	7	580	213

And TNoI is helpful when sotd isn't

Table: TNoI for [BH91]

	$<_{\Delta R}$			$=_G / <_{\Delta R}$		
	TNoI	**Time**	**Cells**	TNoI	**Time**	**Cells**
Intersection A	8	29426	3763	7	2470	273
Intersection B	8	36262	2795	7	1482	189
Random A	9	17355	1219	5	570	165
Random B	9	356670	7119	5	471	141
Ellipse A*	7	262623	28557	6	62496	14439
Ellipse B*	7	$> 1000s$	?	21	$> 1000s$	?
Solotareff A*	9	16014	1751	8	2025	297
Solotareff B*	9	43439	6091	7	1647	243
Collision A*	7	216028	7895	18	$> 1000s$	?
Collision B*	7	$> 1000s$	?	22	$> 1000s$	?

And TNoI is helpful when sotd isn't, mostly (2)

Table: TNoI for [CMMXY09]

	$<_{\Delta R}$			$=_G / <_{\Delta R}$			
	TNoI	**Time**	**Cells**	TNoI	**Time**		**Cells**
Cyclic–3	9	3136	381	6	$20 + 245 =$	265	21
Cyclic–4	16	$> 1000s$	?	6	$64 + 5813 =$	5877	621
2	7	2249	895	14	$22 + 1845 =$	1867	579
4	6	3225	421	11	$24 + 19738 =$	19762	1481
6	4	363	41	5	$20 + 918 =$	938	89
7	8	3667	895	22	$26 + 6537 =$	6563	1211
8	6	3216	365	5	$21 + 174 =$	195	51
13	9	14342	4949	4	$18 + 220 =$	238	81
14	11	334860	27551	9	$21 + 971 =$	992	423

Why does TNoI work?

We don't know!

And it doesn't always: remember that false negative!

What causes TNoI to decrease?

- The number of polynomials goes down (clearly a win)
- ? But factoring a polynomial increases TNoI, even though it's generally a win.
- A polynomial ceases to involve a variable, so there are fewer/lower down resultants
- A polynomial gets replaced by several much simpler ones
- ! We can't really build a model of this, though.

Conclusions

- Gröbner has become (relatively) a lot faster, and is close to negligeable
- Generally $=_{\mathrm{Col}}$ (33 years after inception) has become a lot faster than $<_{\Delta\mathbf{R}}$ (3 years after inception)
- We have not found a transformation ($=_G$ or $\stackrel{*}{\to}^G$) which decreases `TNoI`, but makes the problem slower
- **But** there are examples where `TNoI` increases but the problem is faster
- Generalises "preconditioning" (ISSAC 2010 poster)
- Not only are there many formulations of the problem, there are many formulations of the answer

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

C.W. Brown.
QEPCAD B: a program for computing with semi-algebraic sets using CADs.
ACM SIGSAM Bulletin 4, 37:97–108, 2003.

C. Chen, M. Moreno Maza, B. Xia, and L. Yang.
Computing Cylindrical Algebraic Decomposition via Triangular Decomposition.
In *Proceedings ISSAC 2009*, pages 95–102, 2009.

J.H. Davenport and J. Heintz.
Real Quantifier Elimination is Doubly Exponential.
J. Symbolic Comp., 5:29–35, 1988.

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