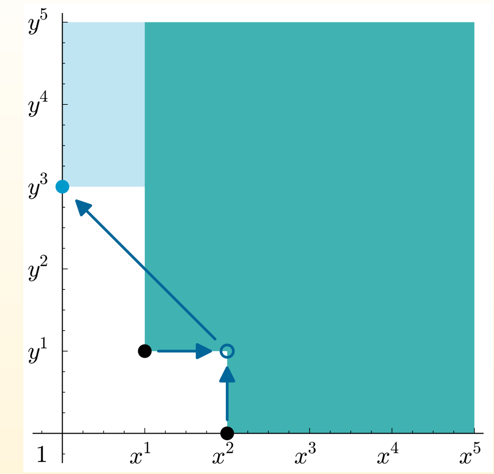
Reducing the number and size of linear programs in a dynamic Gröbner basis algorithm

THE UNIVERSITY OF SOUTHERN MISSISSIPPI

John Perry



Abstract

The dynamic algorithm to compute a Gröbner basis is nearly twenty years old, yet it seems to have arrived stillborn; aside from two initial publications, there have been no published followups. One reason for this may be that, at first glance, the added overhead seems to outweigh the benefit; the algorithm must solve many linear programs with many constraints. This paper describes two methods of reducing the cost substantially.

Dynamic Algorithm

Idea

Gritzmann and Sturmfels, 1993 [4]

- seek "optimal" ordering while computing basis
- measure "optimality" using Hilbert function

Pseudocode

inputs F, generators of polynomial ideal I
outputs

- σ, monomial ordering
- G, Gröbner basis of I with respect to σ

do

1. Let $G = \{\}$, $P = \{(f, 0) : f \in F\}$, σ any ordering
2. **repeat while** $P \neq \emptyset$
   (a) Select $(p, q) \in P$ and remove it
   (b) Let r be some σ-normal form of $\operatorname{spoly}(p, q)$ modulo G
   (c) **if** $r \neq 0$
      i. Add (g, r) to P for each $g \in G$
      ii. Add r to G
   (d) Select an ordering τ
   (e) Add to P any (p, q) such that $p, q \in G \wedge \operatorname{lm}_\sigma(p) \neq \operatorname{lm}_\tau(p)$
   (f) Let $\sigma = \tau$
3. **return** G, σ

First implementation, improvements

Caboara, 1993 [1]

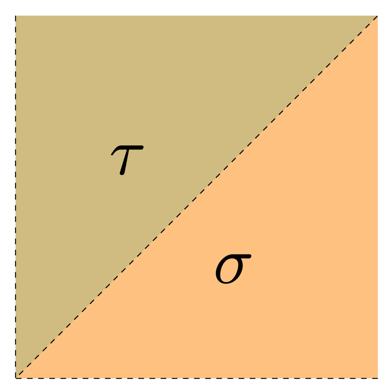
- Compute feasibility, ordering w/linear program:
  $$t > u \Rightarrow \omega(\mathbf{t} - \mathbf{u}) > 0$$
- Consider only mutually indivisible monomials of r
- Keep previously computed leading monomials invariant
  ⇝ Eliminate step 2e, allow discarding useless pairs
  BUT! can increase effort in this case [3]
- Linear programs can grow unwieldy
  ⇝ Skip step 2d "after some time"
  BUT! when?

Problem: *How can we minimize number, size of linear programs?*

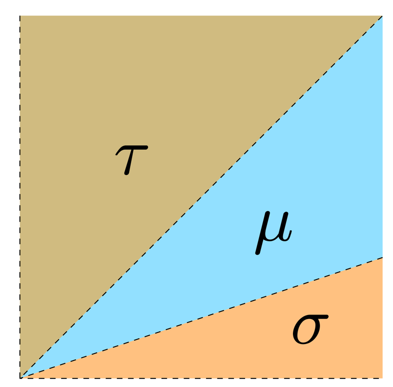
Better living through geometry

Geometry of monomial orderings

- Orderings ⟷ cones in positive orthant (Gröbner fan) [5]
- Add polynomials? split some cones



$g_1 = x^2 + y^2 - 4$
$g_2 = xy - 1$
$\operatorname{lm}_\sigma(G) = (x^2, xy)$
$\operatorname{lm}_\tau(G) = (y^2, xy)$



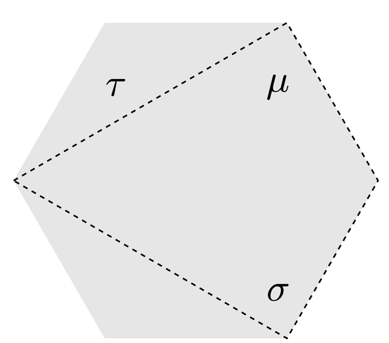
$g_3 = \operatorname{spoly}(g_1, g_2)$
$= y^3 + x - 4y$
$\operatorname{lm}_\sigma(G) = (x^2, xy, x)$
$\operatorname{lm}_\tau(G) = (y^2, xy, y^3)$
$\operatorname{lm}_\mu(G) = (x^2, xy, y^3)$

Corner vectors

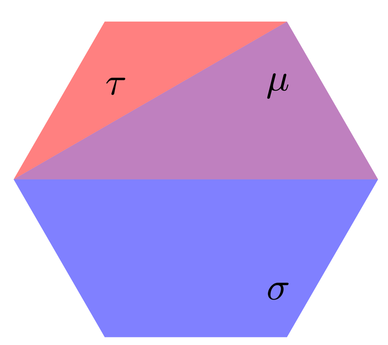
Theorem ([2]): *If we know corner vectors Ω of cone, we need constraints only for monomials u such that $\omega(\mathbf{t} - \mathbf{u}) > 0$ for all $\omega \in \Omega$.*

- ***BUT!*** hard to find all corners
  ⇝ find corners that maximize, minimize each x_i
- ***BUT!*** might miss some potential leading monomials
  ∴ $\operatorname{lm}(G)$ might change later, which is ***bad***
  ⇝ add constraints to revert changes

Example: (cross-section of 3d-cone)



- Select σ, but corner vectors miss τ
  ⇝ miss constraint $\operatorname{lm}_\sigma(f) > \operatorname{lm}_\tau(f)$



- New polynomial: cone splitting!
- τ preferred, but $\operatorname{lm}_\tau(G) \neq \operatorname{lm}_\sigma(G)$
- Add constraint $\operatorname{lm}_\sigma(f) > \operatorname{lm}_\tau(f)$
  ⇝ solution? compromise vector μ
  ⇝ infeasible? cannot choose τ

Experimental results

System	linear programs: prevented by…: cor vec's	trck	nmbr cmptd	max size	size of GB: dyn	stat
Caboara 1	24	0	11	22	34	239
Caboara 2	117	0	9	17	28	553
Caboara 4	84	0	6	22	10	13
Caboara 5	14	0	6	22	9	20
Caboara 6	3	0	6	15	7	7
Caboara 8	2	0	4	9	7	37
Cyc-6	38,421	762	1988	89	20	45
Cyc-7*	3,917,437	4,165	8,106	250	75	209
Cyc-6 hom.	2,042	6	83	54	34	99
Cyc-7 hom.	88,774	0	60	143	104	443
Kat-6	751	2	43	57	17	22
Kat-7	3,979	7	85	88	27	41
Kat-6 hom.	533	0	23	77	22	22
Kat-7 hom.	16,556	8	132	222	49	41

Observations and comments

- "cor vec's" + "trck" + "nmbr cmptd" = #lp's by divisibility
- substantial reduction in number and size of linear programs
  ⇝ determining feasibility, ordering no longer bottleneck
- Applied divisiblity criterion ($O(n^2)$ comparisons) before corner vectors ($O(n)$). Reversing increases "cor vec's" and efficiency.

The Fine Print:

- Normal strategy. Results sensitive to strategy, first polynomial.
- Sage-5.0 w/Cython (patched). C++ implementation planned.
- "trck" counts programs not computed b/c already rejected.
- Cyc-7 used min. degree strategy, corner vectors first.

Citations and Acknowledgments

[1] Massimo Caboara, *A dynamic algorithm for Gröbner basis computation*, ISSAC '93, ACM Press, 1993, pp. 275–283.
[2] Massimo Caboara and John Perry, *Reducing the number and size of linear programs in a dynamic Gröbner basis algorithm*, in preparation.
[3] Oleg Golubitsky, *Converging term order sequences and the dynamic Buchberger algorithm*, preprint received in private communication, in preparation.
[4] Peter Gritzmann and Bernd Sturmfels, *Minkowski addition of polytopes: Computational complexity and applications to Gröbner bases*, SIAM J. Disc. Math **6** (1993), no. 2, 246–269.
[5] Teo Mora and Lorenzo Robbiano, *The Gröbner fan of an ideal*, Journal of Symbolic Computation **6** (1988), 183–208.

Thanks to Nathann Cohen for help with linear programming in Sage, and to Massimo Caboara for insight and encouragement.