Solving Polynomial Systems in the Cloud with Polynomial Homotopy Continuation

Jan Verschelde
joint with Nathan Bliss, Jeff Sommars, and Xiangcheng Yu

University of Illinois at Chicago
Department of Mathematics, Statistics, and Computer Science
https://kepler.math.uic.edu

Algebraic Statistics Seminar, Illinois Institute of Technology
8 September 2015, Chicago

Outline

1. Polynomial Homotopy Continuation
   - solving systems with deformations
   - software packages
   - a blackbox solver
2. Solving in the Cloud
   - a web interface to phc
   - what we currently have
3. Classifying the Solved Systems
   - searching a database of polynomial systems
   - the graph isomorphism problem
   - benchmarking the canonization of systems

Solving Polynomial Systems in the Cloud

1. Polynomial Homotopy Continuation
   - solving systems with deformations
   - software packages
   - a blackbox solver
2. Solving in the Cloud
   - a web interface to phc
   - what we currently have
3. Classifying the Solved Systems
   - searching a database of polynomial systems
   - the graph isomorphism problem
   - benchmarking the canonization of systems

polynomial homotopy continuation methods

$\mathbf{f}(\mathbf{x}) = \mathbf{0}$ is a polynomial system we want to solve,
$\mathbf{g}(\mathbf{x}) = \mathbf{0}$ is a start system ($\mathbf{g}$ is similar to $\mathbf{f}$) with known solutions.

A homotopy $\mathbf{h}(\mathbf{x}, t) = (1 - t)\mathbf{g}(\mathbf{x}) + t\,\mathbf{f}(\mathbf{x}) = \mathbf{0}$, $t \in [0, 1]$,
to solve $\mathbf{f}(\mathbf{x}) = \mathbf{0}$ defines solution paths $\mathbf{x}(t)$: $\mathbf{h}(\mathbf{x}(t), t) \equiv \mathbf{0}$.

Numerical continuation methods track the paths $\mathbf{x}(t)$, from $t = 0$ to 1, applying predictor-corrector algorithms.

Polynomial homotopy continuation methods are hybrid:

- Symbolic: definition of a homotopy $\mathbf{h}(\mathbf{x}, t) = \mathbf{0}$.
  We exploit the sparse structure of $\mathbf{f}$ when constructing $\mathbf{g}$.
- Numeric: path tracking with predictor-corrector algorithms.
  Verify approximate solutions with special care for diverging paths.

polyhedral homotopy methods

The Newton polytope P of a polynomial f with support A is the convex hull of A, formally: $f(x) = \sum_{\mathbf{a} \in A} c_{\mathbf{a}} \mathbf{x}^{\mathbf{a}}$, $c_{\mathbf{a}} \neq 0$, $P = \mathrm{conv}(A)$.

Theorem (Bernstein 1977)

Let $\mathbf{f}(\mathbf{x}) = 0$ *be a polynomial system* $\mathbf{f} = (f_1, f_2, \ldots, f_n)$, $\mathbf{x} = (x_1, x_2, \ldots, x_n)$, *with Newton polytopes* $\mathbf{P} = (P_1, P_2, \ldots, P_n)$. *Denote* $\mathbb{C}^* = \mathbb{C} \setminus \{0\}$.

1. *The mixed volume* V *of* $\mathbf{P}$ *is a generically sharp bound on the number of isolated solutions in* $(\mathbb{C}^*)^n$.
2. *If an initial form system* $\mathrm{in}_{\mathbf{v}}(\mathbf{f})(\mathbf{x}) = \mathbf{0}$ *has a solution in* $(\mathbb{C}^*)^n$, *then* $\mathbf{f}(\mathbf{x}) = \mathbf{0}$ *has fewer than* V *isolated solutions.*

Polyhedral homotopies are ***optimal*** for general sparse systems: no solution paths diverge to infinity.

deflation to restore quadratic convergence

Newton's method breaks when the Jacobian matrix $J_{\mathbf{f}}$ is rank deficient.

Let R be the rank of the Jacobian matrix $J_{\mathbf{f}}$ at $\mathbf{z}$:
a random selection of $R+1$ columns of $J_{\mathbf{f}}(\mathbf{z})$ is linearly dependent.
We introduce $R+1$ multipliers $\boldsymbol{\lambda}$ in a deflation operator, and apply Newton's method on an augmented system:

$$\left\{ \begin{array}{rcl} \mathbf{f}(\mathbf{x}) & = & \mathbf{0} \\ J_{\mathbf{f}}(\mathbf{x},\lambda)B\boldsymbol{\lambda} & = & \mathbf{0} \\ \mathbf{h}\boldsymbol{\lambda} & = & 1 \end{array} \right.$$

where B is a random matrix and $\mathbf{h}$ a random vector.

Theorem (Leykin, Verschelde, Zhao, 2006)

The number of deflations needed to restore the quadratic convergence of Newton's method converging to an isolated solution is strictly less than the multiplicity of the isolated solution.

numerical irreducible decomposition

We compute *generic points* on positive dimensional solution sets, adding random hyperplanes and slack variables to $\mathbf{f}(\mathbf{x}) = \mathbf{0}$:

$$E_k(\mathbf{f})(\mathbf{x},\mathbf{z}) = \left\{ \begin{array}{rcl} \mathbf{f}(\mathbf{x}) + B\mathbf{z} & = & 0 \\ c_{1,0} + c_{1,1}x_1 + \cdots + c_{1,n}x_n + z_1 & = & 0 \\ & \vdots & \\ c_{k,0} + c_{k,1}x_1 + \cdots + c_{k,n}x_n + z_k & = & 0 \end{array} \right.$$

where B is a random n-by-k matrix and random coefficients $c_{i,j}$.
A cascade of homotopies removes the slack variables:

$$\mathbf{h}(\mathbf{x},t) = (1-t)E_k(\mathbf{f}) + t \left(\begin{array}{c} E_{k-1}(\mathbf{f}) \\ z_k \end{array} \right) = \mathbf{0}.$$

Theorem (Sommese, Verschelde, 2001)

Solutions of $E_k(\mathbf{f})(\mathbf{x},\mathbf{z}) = \mathbf{0}$ with nonzero $\mathbf{z}$ are regular; and provide the start solutions in a cascade of homotopies to compute as many generic points as the degrees of the solution sets.

Solving Polynomial Systems in the Cloud

1. Polynomial Homotopy Continuation
   - solving systems with deformations
   - software packages
   - a blackbox solver
2. Solving in the Cloud
   - a web interface to phc
   - what we currently have
3. Classifying the Solved Systems
   - searching a database of polynomial systems
   - the graph isomorphism problem
   - benchmarking the canonization of systems

related software packages

Some related software packages:

1. HOM4PS (Chen, Gao, Lee, Li, Wu, Zeng)
   - \+ fast mixed volume calculators and polyhedral homotopies
   - – current versions are not free nor open source
2. pss (Malajovich)
   - \+ parallel mixed volume computation, released under the GNU GPL
3. Bertini (Bates, Hauenstein, Sommese, Wampler)
   - – no support for polyhedral methods
   - \+ adaptive multiprecision path trackers
4. NAG4M2 (Leykin)
   - \+ well integrated with Macaulay2
   - \+ does not reinvent the wheel:
     $\rightarrow$ interfaces to Bertini, HOM4PS, PHCpack.

PHCpack

PHCpack is a package for Polynomial Homotopy Continuation.
ACM Transactions on Mathematical Software archived version 1.0
as Algorithm 795, vol. 25, no. 2, pages 251–276, 1999.

blackbox solver:

`phc -b` computes all isolated solutions of a polynomial system.

External software package integrated in PHCpack:

- Fast mixed volume computation by MixedVol of Gao, Li, and Wu, Algorithm 846 of ACM TOMS, vol. 31, pages 555–560, 2005.
- Double double and quad double arithmetic with the QD Library of Hida, Li, and Bailey published in the 15th IEEE Symposium on Computer Arithmetic, pages 155–162. IEEE, 2001.

Interfaces to Macaulay2 (Gross, Petrovic), Maple (Leykin), MATLAB (Guan), Python (Piret), Sage (Hampton, Jokela, Stein).

Version 2.4 supports GPU accelerated path trackers (Yoffe, Yu).

Solving Polynomial Systems in the Cloud

1. Polynomial Homotopy Continuation
   - solving systems with deformations
   - software packages
   - a blackbox solver
2. Solving in the Cloud
   - a web interface to phc
   - what we currently have
3. Classifying the Solved Systems
   - searching a database of polynomial systems
   - the graph isomorphism problem
   - benchmarking the canonization of systems

what is a blackbox solver?

What to expect from a blackbox solver?

- The polynomials are the *only* input to the solver.
- Parameters that control the execution options are
  - set to *work well on a large class of examples*; and/or
  - tuned automatically during the solving process.
- The output contains various *diagnostics* and checks.

  The user should be
  - warned in case of ill conditioning and nearby singularities;
  - able to verify (or falsify) the computed results.

what does `phc -b` do?

Polyhedral homotopies are optimal for sparse polynomial systems:

- the root count (the mixed volume) is sharp for generic problems;
- every path in a polyhedral homotopy ends at an isolated root, except for systems that has special initial forms.

The blackbox solver was designed for *square* problems,
that is: as many equations as unknowns.

Special cases:

- linear systems and polynomials in one variable
- binomial systems (exactly two monomials in every polynomial)
  - isolated solutions determined by Hermite normal form;
  - positive dimensional solution sets are monomial maps.

more concretely, what does `phc -b` really do?

Stages in the solving process:

1. Parse and classify the polynomial system.
   - Parse: attempt to gracefully handling of syntax errors.
   - Classify: handle univariate, linear, and binomial directly.
2. If not univariate, linear, or binomial, then two cases remain:
   - the system is square (as many equations as unknowns),
   - the system is overdetermined or underdetermined.

The nonsquare case is still experimental (as is `phc -a`),
`phc -a` gives access to an equation-by-equation solver.

the square case

The original blackbox solver has its focus on isolated solutions.

For the square case, we have four types of start systems:

1. start system based on the total degree,
2. one multi-homogenous partition of the set of variables,
3. general linear-product start systems,
4. random coefficient start systems solved by polyhedral homotopies.

The start system with the lowest root count is selected and solved.

Path tracking to the target system gives solutions, to classify

1. detect path clustering, gather multiplicities;
2. frequency tables of forward errors, backward errors (residuals), and estimates for condition numbers.

Solving Polynomial Systems in the Cloud

1. Polynomial Homotopy Continuation
   - solving systems with deformations
   - software packages
   - a blackbox solver
2. Solving in the Cloud
   - a web interface to phc
   - what we currently have
3. Classifying the Solved Systems
   - searching a database of polynomial systems
   - the graph isomorphism problem
   - benchmarking the canonization of systems

motivation for a cloud service

In some disciplines, cloud computing has become the norm.

Benefits for the user (but there are risks as well):

- No installation is required, just sign up.

  Installing software can be complicated and a waste of time,
  especially if one want to perform a single experiment.
  The user should not worry about upgrading to newer versions.

- We offer a computing service.

  The web server is hosted by a powerful computer,
  which can be extended with the addition of compute servers.

- Files and data are stored and managed for the user.

  The input and output files are managed at the server.
  For larger problems, storage space can become an issue.

Macaulay2 in the cloud

`PHCpack.m2` is a package distributed with Macaulay2.

Macaulay2 runs in the cloud as well.

An example session with omitted output is below:

```
Macaulay2, version 1.7


i1 : loadPackage "PHCpack";
i2 : help(PHCpack);
i3 : help solveSystem;
i4 : R = CC[x,y,z];
i5 : S = {x+y+z-1, x^2+y^2, x+y-z-3};
i6 : solveSystem(S)
```

Solving Polynomial Systems in the Cloud

1. Polynomial Homotopy Continuation
   - solving systems with deformations
   - software packages
   - a blackbox solver
2. Solving in the Cloud
   - a web interface to phc
   - what we currently have
3. Classifying the Solved Systems
   - searching a database of polynomial systems
   - the graph isomorphism problem
   - benchmarking the canonization of systems

what we currently have

A web interface to the blackbox solver of `phc` is running at `https://kepler.math.uic.edu`.

1. The server `kepler` runs Red Hat Linux.
2. Apache is the web server.
3. Our database is MySQL.
4. Python is the scripting language.

All software is free and open source.

The web service was also deployed and tested on a Mac OS X.

In its current state, the setup of the web interface is minimal, but, most importantly: It works!

Apache

The web server runs Apache.

- One `index.html` leads to the login Python script.
- The `cgi-bin` directory contains all scripts.

The setup process is automatically executed at a reboot.

Registration is done automatically via a google email account.

five Python scripts

Less than 1,500 lines of Python code.

A simplified view of the original design:

- `cookie_login` prints first login screen
  - calls `register` for a first time user; or
  - calls `phc_solver`
- `register` sends email to first time user
- `activate` runs when user clicks in email
- `contact` is optional to send emails about the service
- `phc_solver` solves polynomial systems

MySQL

MySQL is called in Python through the module `MySQLdb`.

The database manages two tables:

- `users`: data about users, encrypted passwords;
- `polys`: references to systems and solutions.

Mathematical data are not stored in the database:

- Every user has a folder, a generated 40 character string.
- With every system there is another generated 40 character string.

Solving Polynomial Systems in the Cloud

1. Polynomial Homotopy Continuation
   - solving systems with deformations
   - software packages
   - a blackbox solver
2. Solving in the Cloud
   - a web interface to phc
   - what we currently have
3. Classifying the Solved Systems
   - searching a database of polynomial systems
   - the graph isomorphism problem
   - benchmarking the canonization of systems

the classification problem

If we have solved a system with the same structure,
then the solved system is the start system in a homotopy.

For example, the systems

$$\begin{cases} x^2 + xy^2 - 3 = 0 \\ 2x^2y + 5 = 0 \end{cases} \quad \text{and} \quad \begin{cases} 3 + 2ab^2 = 0 \\ b^2 - 5 + 2a^2b = 0 \end{cases}$$

are isomorphic to each other. There support sets are

$$\{\{(2,0),(1,2),(0,0)\},\{(2,1),(0,0)\}\}$$
$$\text{and} \quad \{\{(0,0),(1,2)\},\{(0,2),(0,0),(2,1)\}\}.$$

Definition

Two sets of support sets are *isomorphic* if there exists a permutation of their equations and variables so that the sets of support sets are identical.

the isomorphism problem of polynomials

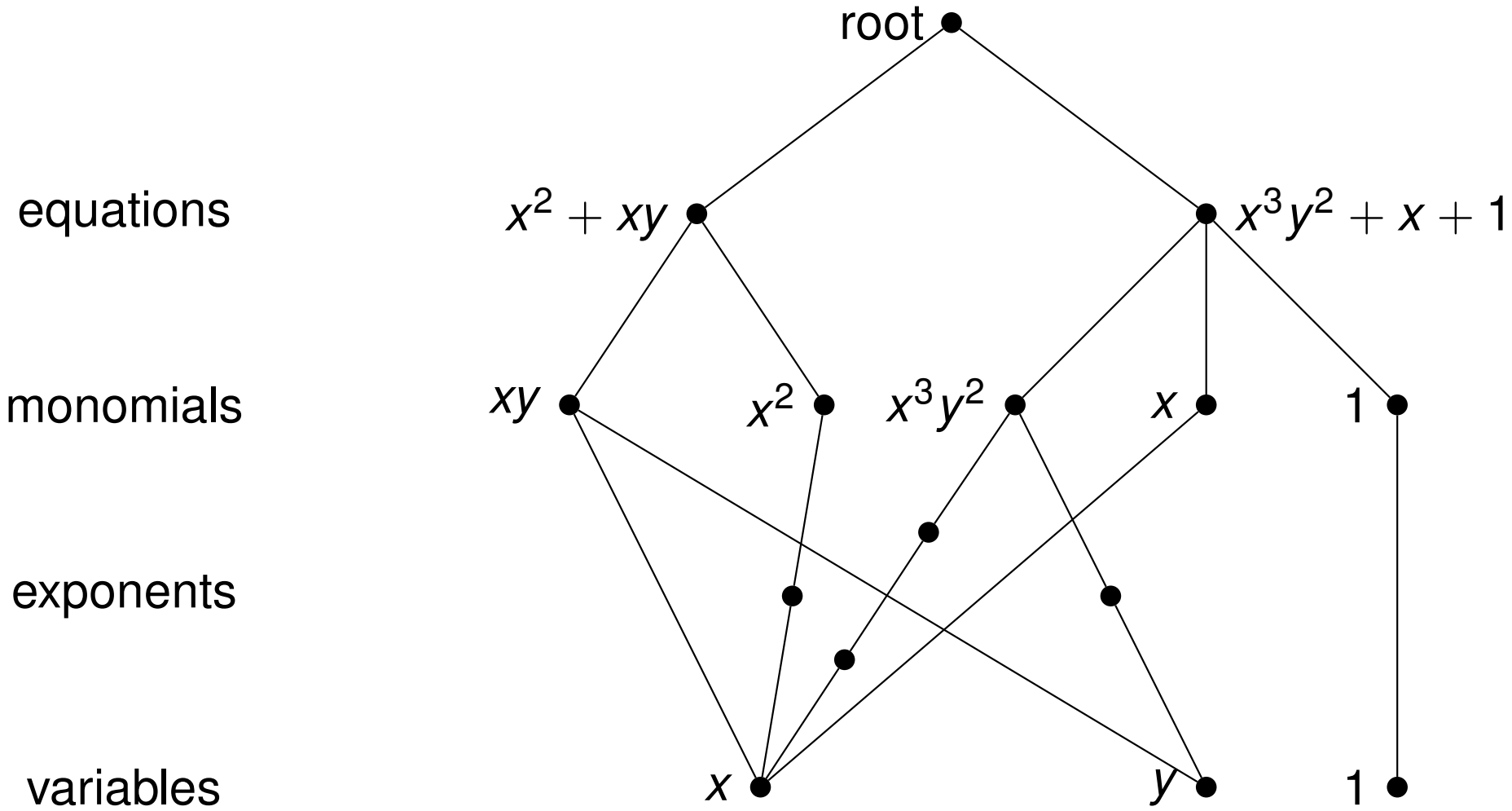
Related work occurs in multivariate cryptography.

- J. Patarin. **Hidden fields equations (HFE) and isomorphism of polynomials (IP): Two new families of asymmetric algorithms.** In U. Maurer, editor, *Advances in Cryptology - EUROCRYPT'96*, volume 1070 of *Lecture Notes in Computer Science*, pages 33–48. Springer-Verlag, 1996.
- J.-C. Faugère and L. Perret. **Polynomial equivalence problems: Algorithmic and theoretical aspects.** In S. Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *Lecture Notes in Computer Science*, pages 30–47. Springer-Verlag, 2006.
- C. Bouillaguet, P.-A. Fouque, and A. Véber. **Graph-theoretic algorithms for the "Isomorphism of Polynomials" problem.** In T. Johansson and P. Nguyen, editors, *EUROCRYPT 2013*, volume 7881 of *Lecture Notes in Computer Science*, pages 211–227. Springer-Verlag, 2013.

Solving Polynomial Systems in the Cloud

1. Polynomial Homotopy Continuation
   - solving systems with deformations
   - software packages
   - a blackbox solver
2. Solving in the Cloud
   - a web interface to phc
   - what we currently have
3. Classifying the Solved Systems
   - searching a database of polynomial systems
   - the graph isomorphism problem
   - benchmarking the canonization of systems

representing a system as a graph



the graph isomorphism problem

Definition

The *graph isomorphism problem* asks whether for two undirected graphs F, G there is a bijection ϕ between their vertices that preserves incidence;
i.e.: if a and b are vertices connected by an edge in F,
then $\phi(a)$ and $\phi(b)$ are connected by an edge in G.

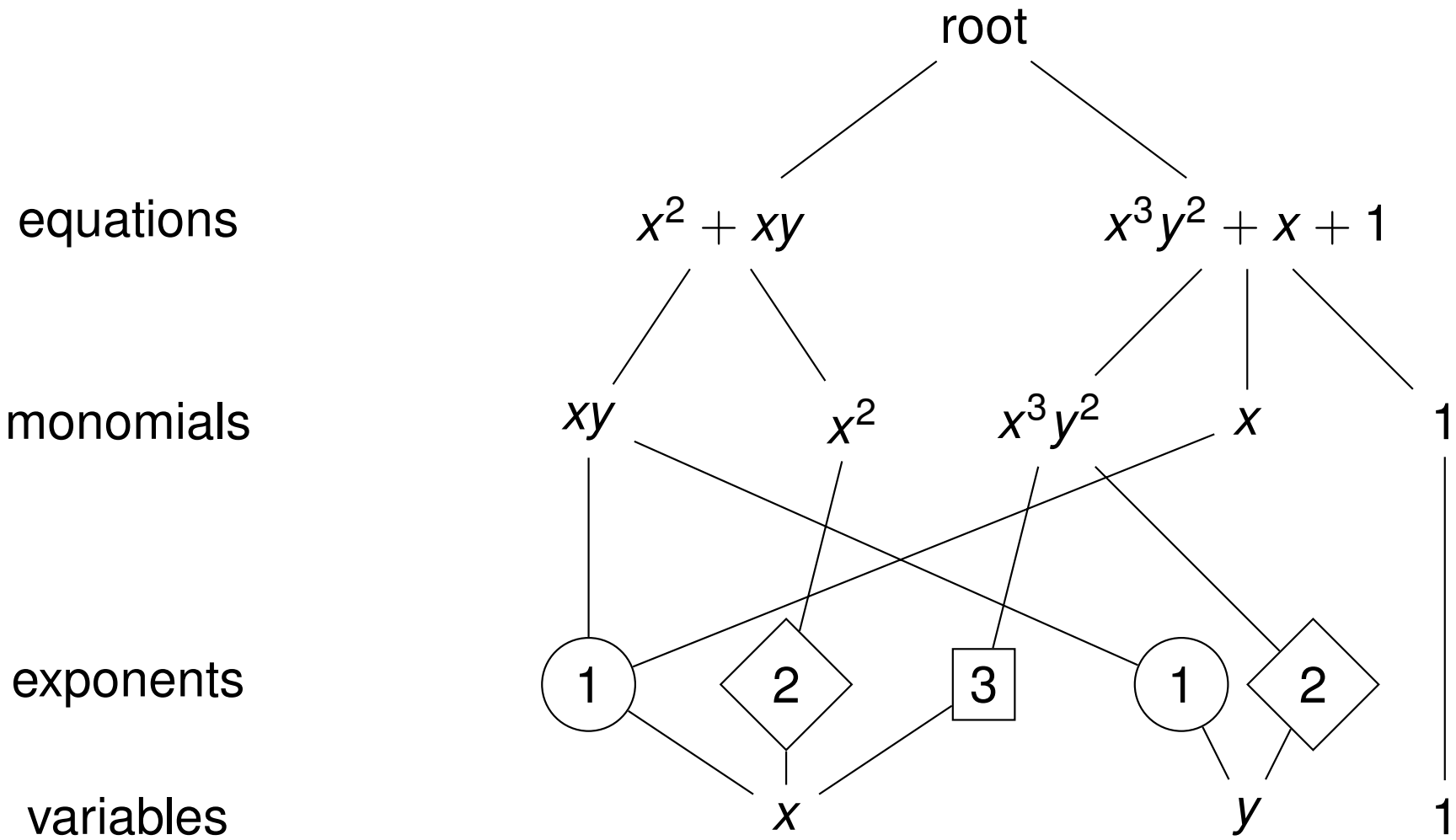
Proposition

The problem of determining whether two sets of support sets are isomorphic is equivalent to the graph isomorphism problem.

A practical solution: the software package **nauty**.

B.D. McKay and A. Piperno. **Practical graph isomorphism, II.** *Journal of Symbolic Computation*, 60:94–112, 2014.

canonical graph labelings with nauty



Solving Polynomial Systems in the Cloud

1. Polynomial Homotopy Continuation
   - solving systems with deformations
   - software packages
   - a blackbox solver
2. Solving in the Cloud
   - a web interface to phc
   - what we currently have
3. Classifying the Solved Systems
   - searching a database of polynomial systems
   - the graph isomorphism problem
   - benchmarking the canonization of systems

benchmarking nauty on cyclic n-roots

$$\text{cyclic 3-roots:} \left\{ \begin{array}{r} x_1 + x_2 + x_3 = 0 \\ x_1x_2 + x_2x_3 + x_3x_1 = 0 \\ x_1x_2x_3 - 1 = 0. \end{array} \right.$$

Times in milliseconds for small values of n,
to compute the canonical form with `nauty`:

n	time	#nodes	#characters
4	0.006	29	526
6	0.006	53	1,256
8	0.006	85	2,545
10	0.007	125	5,121
12	0.007	173	8,761

Note: computing the root counts by `phc -b`
for the cyclic 10-roots problems takes 48.8 seconds.

cyclic n-roots, for large n

For larger values of the dimension of the cyclic n-root problem, times and sizes of the data start to grow exponentially.

n	time	#nodes	#characters
16	0.010	293	20,029
32	0.045	1,093	168,622
48	0.265	2,405	601,702
64	1.200	4,229	1,427,890
80	4.316	6,565	2,778,546
96	15.274	9,413	4,784,390
112	38.747	12,773	8,595,408
128	80.700	16,645	13,094,752

Note: computing all isolated solutions is no longer possible.
With GPU acceleration, tracking a limited number of paths is possible.

canonization of Nash equilibria

The system to compute all totally mixed Nash equilibria has supports invariant by the full permutation group.

n	time	#nodes	#characters
4	0.006	47	977
5	0.006	98	2,325
6	0.007	213	7,084
7	0.013	472	18,398
8	0.054	1,051	51,180
9	0.460	2,334	134,568
10	4.832	5,153	331,456
11	73.587	11,300	872,893
12	740.846	24,615	2,150,512

The systems are not sparse and the number of solutions groups as $n!$.

a system without symmetry

Another benchmark systems is formulated by Katsura.
Without symmetry in the support sets, the cost of the canonization increases not as fast in the dimension n.

n	time	#nodes	#characters
25	0.020	929	24,906
50	0.090	3,411	112,654
75	0.546	7,454	254,770
100	1.806	13,061	495,612
125	4.641	20,229	793,662
150	10.860	28,961	1,157,498
175	21.194	39,254	1,587,115
200	52.814	51,111	2,082,562
225	98.118	64,529	2,643,891

The number of solutions equals 2^n.

concluding remarks

Basic version of web interface to `phc -b` and `phc -p`.

- Built with LAMP stack, Python as scripting language.
- Classification of solved systems via the graph isomorphism.

In the future: automatic exploitation of permutation symmetry.

N. Bliss, J. Sommars, J. Verschelde, and Xiangcheng Yu.
Solving polynomial systems in the cloud with polynomial homotopy continuation.
In V.P. Gerdt, W. Koepf, W.M. Seiler, and E.V. Vorozhtsov, editors, *Computer Algebra in Scientific Computing, 17th International Workshop, CASC 2015, Aachen, Germany*, volume 9301 of *Lecture Notes in Computer Science*, pages 87–100. Springer-Verlag, 2015.

try it at **https://kepler.math.uic.edu**