Parallel Homotopy Algorithms to Solve Polynomial Systems

Jan Verschelde

Department of Math, Stat & CS
University of Illinois at Chicago
Chicago, IL 60607-7045, USA

email: `jan@math.uic.edu`

URL: `http://www.math.uic.edu/~jan`

Joint work with Anton Leykin and Yan Zhuang.

2nd International Congress on Mathematical Software

1-3 September 2006, Castro Urdiales, Spain.

Outline of the Talk

A. homotopy continuation methods are "**pleasingly parallel**"

jumpstarting homotopies for **memory** *efficiency*

B. parallel implementations of **polyhedral homotopies**

static and dynamic load **balancing**

C. a **numerically stable** simplicial solver

instabilities *appeared only in large systems* $(n = 12)$

D. applications from **mechanical design**

design of serial chains requires $>$ **100,000 paths**

parallel PHCpack

phc -b (blackbox solver) works well for systems of medium size,
about 1,000 solution paths.

implement "pleasingly parallel" homotopies:
with Yusong Wang (HPSEC'04): Pieri homotopies;
with Anton Leykin (HPSEC'05): monodromy breakup;
with Yan Zhuang (HPSEC'06): polyhedral homotopies.

software development on personal cluster computer
(1 manager + 13 workers at 2.4Ghz) built by Rocketcalc.

Runs done on UIC supercomputer argo, NCSA machines
Platinum IA32 Cluster and IBM pSeries 690 system copper.

Goal: solve systems which require > 100,000 paths well.

Other Parallel Homotopy Solvers

T. Gunji, S. Kim, K. Fujisawa, and M. Kojima:
PHoMpara – parallel implementation of the Polyhedral Homotopy continuation Method for polynomial systems. *Computing* 77(4):387–411, 2006.

H.-J. Su, J.M. McCarthy, M. Sosonkina, and L.T. Watson:
Algorithm 8xx: **POLSYS_GLP**: A parallel general linear product homotopy code for solving polynomial systems of equations. To appear in *ACM Trans. Math. Softw.*

Numerical Algebraic Geometry

A.J. Sommese and C.W. Wampler: The Numerical Solution of Systems of Polynomials Arising in Engineering and Science. *World Scientific Press*, Singapore, 2005.

Jumpstarting Homotopies

Problem: huge #paths (e.g.: $> 100{,}000$),

undesirable to store all start solutions in main memory.

Solution: (assume manager/worker protocol)

1. The manager reads start solution from file "**just in time**" whenever a worker needs another path tracking job.
2. For total degree and linear-product start systems, it is **simple to compute** the solutions whenever needed.
3. As soon as worker reports the end of a solution path back to the manager, the solution is **written to file**.

Indexing Start Solutions

The start system $\left\{ \begin{array}{l} x_1^4 - 1 = 0 \\ x_2^5 - 1 = 0 \\ x_3^3 - 1 = 0 \end{array} \right.$ has $4 \times 5 \times 3 = 60$ solutions.

Get 25th solution via decomposition: $24 = 1(5 \times 3) + 3(3) + 0$.

Verify via lexicographic enumeration:

$000 \rightarrow 001 \rightarrow 002 \rightarrow 010 \rightarrow 011 \rightarrow 012 \rightarrow 020 \rightarrow 021 \rightarrow 022 \rightarrow 030 \rightarrow 031 \rightarrow 032 \rightarrow 040 \rightarrow 041 \rightarrow 042$

$100 \rightarrow 101 \rightarrow 102 \rightarrow 110 \rightarrow 111 \rightarrow 112 \rightarrow 120 \rightarrow 121 \rightarrow 122 \rightarrow \boxed{130} \rightarrow 131 \rightarrow 132 \rightarrow 140 \rightarrow 141 \rightarrow 142$

$200 \rightarrow 201 \rightarrow 202 \rightarrow 210 \rightarrow 211 \rightarrow 212 \rightarrow 220 \rightarrow 221 \rightarrow 222 \rightarrow 230 \rightarrow 231 \rightarrow 232 \rightarrow 240 \rightarrow 241 \rightarrow 242$

$300 \rightarrow 301 \rightarrow 302 \rightarrow 310 \rightarrow 311 \rightarrow 312 \rightarrow 320 \rightarrow 321 \rightarrow 322 \rightarrow 330 \rightarrow 331 \rightarrow 332 \rightarrow 340 \rightarrow 341 \rightarrow 342$

Using Linear-Product Start Systems Efficiently

- Store start systems in their linear-product product form, e.g.:

$$g(\mathbf{x}) = \left\{ \begin{array}{c} (\cdots)\cdot(\cdots)\cdot(\cdots)\cdot(\cdots) = 0 \\ (\cdots)\cdot(\cdots)\cdot(\cdots)\cdot(\cdots)\cdot(\cdots) = 0 \\ (\cdots)\cdot(\cdots)\cdot(\cdots) = 0 \end{array} \right.$$

- Lexicographic enumeration of start solutions,

  $\longrightarrow$ as many candidates as the total degree.

- Eventually store results of incremental LU factorization.

  $\longrightarrow$ prune in the tree of combinations.

a problem from electromagnetics

posed by Shigetoshi Katsura to PoSSo in 1994:

a family of $n-1$ **quadrics** and one linear equation;
#solutions is 2^{n-1} ($=$ Bézout bound).

$n = 21$**: 32 hours and 44 minutes** to track 2^{20} paths by 13 workers at 2.4Ghz, producing output file of 1.3Gb.

tracking about 546 paths/minute.

verification of output:

1. parsing 1.3Gb file into memory takes 400Mb and 4 minutes;
2. data compression to quadtree of 58Mb takes 7 seconds.

Polyhedral Homotopies

D.N. Bernshteĭn. *Functional Anal. Appl.* 1975.

B. Huber and B. Sturmfels. *Math. Comp.* 1995.

T.Y. Li. *Handbook of Numerical Analysis. Volume XI.* 2003.

T. Gao, T.Y. Li, and M. Wu. Algorithm 846: **MixedVol**: A software package for mixed volume computation. *ACM Trans. Math. Softw.* 31(4):555–560, 2005.

T. Gunji, S. Kim, M. Kojima, A. Takeda, K. Fujisawa, and T. Mizutani. **PHoM** – a polyhedral homotopy continuation method for polynomial systems. *Computing* 73(4):55–77, 2004.

G. Jeronimo, G. Matera, P. Solernó, and A. Waissbein. Deformation techniques for sparse systems. `arXiv:math.CA/0608714 v1 29 Aug 2006`.

3 stages to solve a polynomial system $f(\mathbf{x}) = \mathbf{0}$

1. Compute the mixed volume (aka the BKK bound) of the Newton polytopes spanned by the supports A of f via a **regular mixed-cell configuration** Δ_ω.

2. Given Δ_ω, solve a generic system $g(\mathbf{x}) = \mathbf{0}$, using polyhedral homotopies. Every cell $C \in \Delta_\omega$ defines one homotopy

$$h_C(\mathbf{x}, s) = \sum_{\mathbf{a} \in C} c_{\mathbf{a}} \mathbf{x}^{\mathbf{a}} + \sum_{\mathbf{a} \in A \setminus C} c_{\mathbf{a}} \mathbf{x}^{\mathbf{a}} s^{\nu_{\mathbf{a}}}, \quad \nu_{\mathbf{a}} > 0,$$

   tracking as many paths as the mixed volume of the cell C, as s goes from 0 to 1.

3. Use $(1-t)g(\mathbf{x}) + tf(\mathbf{x}) = \mathbf{0}$ to solve $f(\mathbf{x}) = \mathbf{0}$.

Stages 2 and 3 are **computationally most intensive** $(1 \ll 2 < 3)$.

A Static Distribution of the Workload

manager	worker 1	worker 2	worker 3
Vol(cell 1) = 5	#paths(cell 1) : 5		
Vol(cell 2) = 4	#paths(cell 2) : 4		
Vol(cell 3) = 4	#paths(cell 3) : 4		
Vol(cell 4) = 6	#paths(cell 4) : 1	#paths(cell 4) : 5	
Vol(cell 5) = 7		#paths(cell 5) : 7	
Vol(cell 6) = 3		#paths(cell 6) : 2	#paths(cell 6) : 1
Vol(cell 7) = 4			#paths(cell 7) : 4
Vol(cell 8) = 8			#paths(cell 8) : 8
total #paths : 41	#paths : 14	#paths : 14	#paths : 13

Since polyhedral homotopies solve a **generic** system $g(\mathbf{x}) = \mathbf{0}$,

we **expect** every path to take the same amount of work...

Results on the cyclic n-roots problem

Problem	#Paths	CPU Time
cyclic 5-roots	70	0.13m
cyclic 6-roots	156	0.19m
cyclic 7-roots	924	0.30m
cyclic 8-roots	2,560	0.78m
cyclic 9-roots	11,016	3.64m
cyclic 10-roots	35,940	21.33m
cyclic 11-roots	184,756	2h 39m
cyclic 12-roots	500,352	24h 36m

Wall time for start systems to solve the cyclic n-roots problems, using 13 workers, with static load distribution.

Dynamic versus Static Workload Distribution

	Static versus Dynamic on our cluster				Dynamic on argo	
#workers	Static	Speedup	Dynamic	Speedup	Dynamic	Speedup
1	50.7021	–	53.0707	–	29.2389	–
2	24.5172	2.1	25.3852	2.1	15.5455	1.9
3	18.3850	2.8	17.6367	3.0	10.8063	2.7
4	14.6994	3.4	12.4157	4.2	7.9660	3.7
5	11.6913	4.3	10.3054	5.1	6.2054	4.7
6	10.3779	4.9	9.3411	5.7	5.0996	5.7
7	9.6877	5.2	8.4180	6.3	4.2603	6.9
8	7.8157	6.5	7.4337	7.1	3.8528	7.6
9	7.5133	6.8	6.8029	7.8	3.6010	8.1
10	6.9154	7.3	5.7883	9.2	3.2075	9.1
11	6.5668	7.7	5.3014	10.0	2.8427	10.3
12	6.4407	7.9	4.8232	11.0	2.5873	11.3
13	5.1462	9.8	4.6894	11.3	2.3224	12.6

Wall time in seconds to solve a start system for the cyclic 7-roots problem.

A well conditioned polynomial system

just one of the 11,417 start systems generated by polyhedral homotopies

12 equations, 13 distinct monomials (after division):

$$
\left\{
\begin{array}{c}
b_1 x_5 x_8 + b_2 x_6 x_9 = 0 \\
b_3 x_2^2 + b_4 = 0 \\
b_5 x_1 x_4 + b_6 x_2 x_5 = 0 \\
c_1^{(k)} x_1 x_4 x_7 x_{12} + c_2^{(k)} x_1 x_6 x_{10}^2 + c_3^{(k)} x_2 x_4 x_8 x_{10} + c_4^{(k)} x_2 x_4 x_{11}^2 \\
+ c_5^{(k)} x_2 x_6 x_8 x_{11} + c_6^{(k)} x_3 x_4 x_9 x_{10} + c_7^{(k)} x_4^2 x_{12}^2 + c_8^{(k)} x_3 x_6 \\
+ c_9^{(k)} x_4^2 + c_{10}^{(k)} x_9 = 0, \quad k = 1, 2, \ldots, 9
\end{array}
\right.
$$

Random coefficients chosen on the complex unit circle.

Despite the high degrees, only 100 well conditioned solutions.

Solve a "binomial" system $\mathbf{x}^A = \mathbf{b}$ via Hermite

Hermite normal form of A: $MA = U$, $\det(M) = \pm 1$,
U is upper triangular, $|\det(U)| = |\det(A)| = \#$solutions.

Let $\mathbf{x} = \mathbf{z}^M$, then $\mathbf{x}^A = \mathbf{z}^{MA} = \mathbf{z}^U$, so solve $\mathbf{z}^U = \mathbf{b}$.

$n = 2$:

$$[z_1 \quad z_2] \begin{bmatrix} u_{11} & u_{12} \\ 0 & u_{22} \end{bmatrix} = [b_1 \quad b_2].$$

$$\left\{ \begin{array}{rcl} z_1^{u_{11}} & = & b_1 \\ z_1^{u_{12}} \; z_2^{u_{22}} & = & b_2 \end{array} \right. \qquad |b_k| = 1 \Rightarrow |z_i| = 1$$

numerically well conditioned

Reduce a "simplicial" system $C\mathbf{x}^A = \mathbf{b}$ via LU

$$\begin{array}{c} C = LU \\ \text{assume} \ \det(C) \neq 0 \end{array} \Rightarrow \begin{array}{lll} (1) & LU\mathbf{y} = \mathbf{b} & \text{linear system} \\ (2) & \mathbf{x}^A = \mathbf{y} & \text{binomial system} \end{array}$$

This is a numerically unstable algorithm!

Randomly chosen coefficients for C and $\mathbf{b}$ on complex unit circle,

but still, varying magnitudes in $\mathbf{y}$ do occur.

High powers, e.g.: 50, magnify the imbalance

$\rightarrow$ numerical underflow or overflow in binomial solver.

Separate Magnitudes from Angles

Solve $\mathbf{x}^A = \mathbf{y}$ via Hermite: $MA = U \Rightarrow \mathbf{x} = \mathbf{z}^M : \mathbf{z}^U = \mathbf{y}$.

$\mathbf{z} = |\mathbf{z}|\mathbf{e}_\mathbf{z}$, $\mathbf{e}_\mathbf{z} = \exp(i\theta_\mathbf{z})$, $\mathbf{y} = |\mathbf{y}|\mathbf{e}_\mathbf{y}$, $\mathbf{e}_\mathbf{y} = \exp(i\theta_\mathbf{y})$, $i = \sqrt{-1}$.

Solve $\mathbf{z}^U = \mathbf{y}$: $|\mathbf{z}|^U \mathbf{e}_\mathbf{z}^U = |\mathbf{y}|\mathbf{e}_\mathbf{y} \Leftrightarrow \left\{ \begin{array}{ll} \mathbf{e}_\mathbf{z}^U = \mathbf{e}_\mathbf{y} & \text{well conditioned} \\ |\mathbf{z}|^U = |\mathbf{y}| & \text{find magnitudes} \end{array} \right.$

To solve $|\mathbf{z}|^U = |\mathbf{y}|$, consider: $U \log(|\mathbf{z}|) = \log(|\mathbf{y}|)$.

Even as the magnitude of the values $\mathbf{y}$ may be extreme, $\log(|\mathbf{y}|)$ will be modest in size.

a numerically stable simplicial solver

We solve $C\mathbf{x}^A = \mathbf{b}$ by

1. LU factorization of $C \rightarrow \mathbf{x}^A = \mathbf{y}$, where $C\mathbf{y} = \mathbf{b}$.
2. Use Hermite normal form of A: $MA = U$, $\det(M) = \pm 1$, to solve binomial system $\mathbf{e}_\mathbf{z}^U = \mathbf{e}_\mathbf{y}$, $\mathbf{z} = |\mathbf{z}|\mathbf{e}_\mathbf{z}$, $\mathbf{y} = |\mathbf{y}|\mathbf{e}_\mathbf{y}$.
3. Solve upper triangular linear system $U \log(|\mathbf{z}|) = \log(|\mathbf{y}|)$.
4. Compute magnitude of $\mathbf{x} = \mathbf{z}^M$ via $\log(|\mathbf{x}|) = M \log(|\mathbf{z}|)$.
5. As $|\mathbf{e}_\mathbf{z}| = 1$, let $\mathbf{e}_\mathbf{x} = \mathbf{e}_\mathbf{z}^M$.

Even as $\mathbf{z}$ may be extreme, we deal with $|\mathbf{z}|$ at a logarithmic scale and never raise small or large number to high powers.

Only at the very end do we calculate $|\mathbf{x}| = 10^{\log(|\mathbf{x}|)}$ and $\mathbf{x} = |\mathbf{x}|\mathbf{e}_\mathbf{x}$.

Design of Serial Chains I

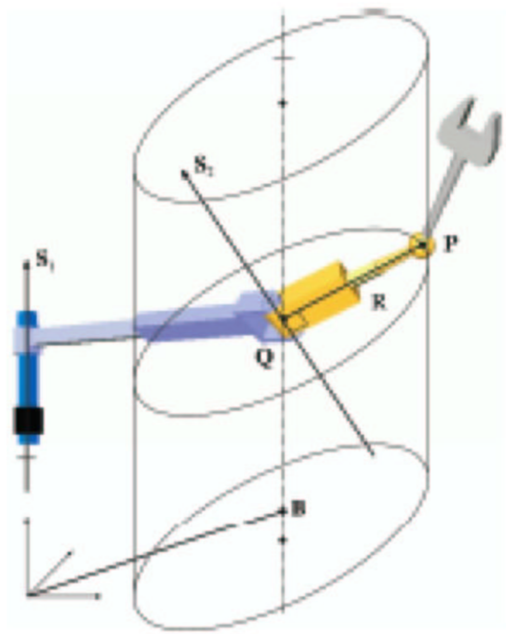

Figure 4.4: The elliptic cylinder reachable by a PRS serial chain.

H.J. Su. *Computer-Aided Constrained Robot Design Using Mechanism Synthesis Theory.* PhD thesis, University of California, Irvine, 2004.

Design of Serial Chains II

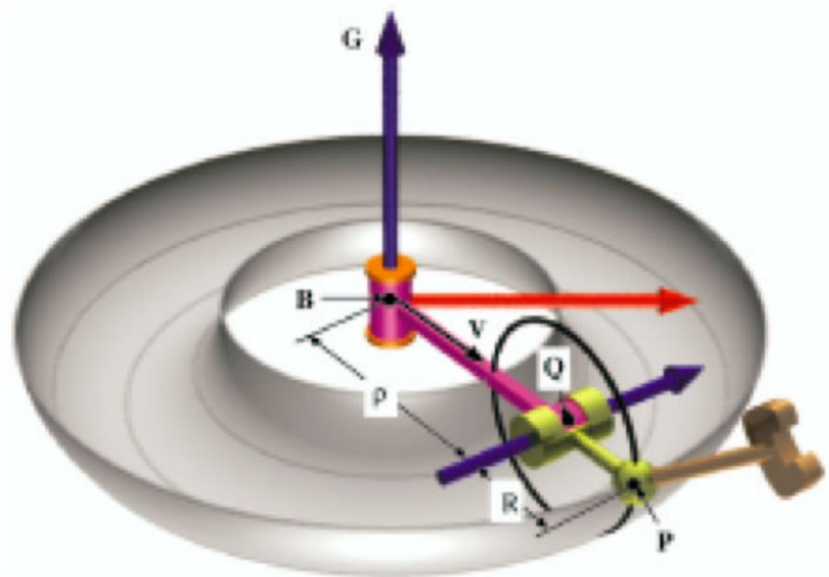

Figure 4.7: The circular torus traced by the wrist center of a "right" RRS serial chain.

H.J. Su. *Computer-Aided Constrained Robot Design Using Mechanism Synthesis Theory.* PhD thesis, University of California, Irvine, 2004.

Design of Serial Chains III

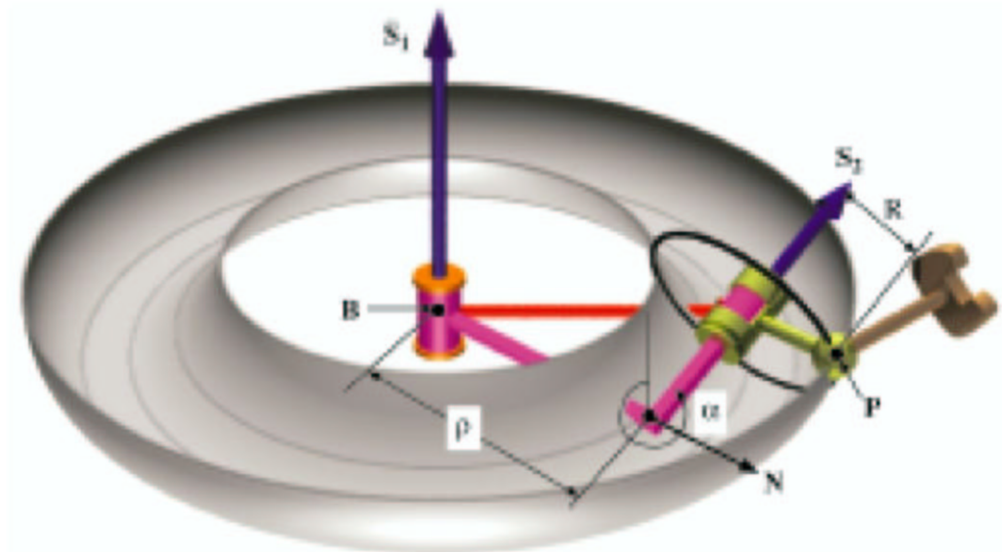

Figure 4.8: The general torus reachable by the wrist center of an RRS serial chain.

H.J. Su. *Computer-Aided Constrained Robot Design Using Mechanism Synthesis Theory*. PhD thesis, University of California, Irvine, 2004.

For more about these problems:

H.-J. Su and J. McCarthy. Kinematic synthesis of RPS serial chains. In the *Proceedings of the ASME Design Engineering Technical Conferences* (CDROM), Chicago, IL, Sep 2-6, 2003.

H.-J. Su, J. McCarthy, and L. Watson. Generalized linear product homotopy algorithms and the computation of reachable surfaces. *ASME Journal of Information and Computer Sciences in Engineering*, 4(3):226–234, 2004.

H.-J. Su, C. Wampler, and J. McCarthy. Geometric design of cylindric PRS serial chains. *ASME Journal of Mechanical Design*, 126(2):269–277, 2004.

Results on Mechanical Design Problems

Surface	Bounds on #Solutions			Wall Time	
	Bézout	linear-product	Mixvol	our cluster	on argo
elliptic cylinder	2,097,152	247,968	125,888	11h 33m	6h 12m
circular torus	2,097,152	868,352	474,112	7h 17m	4h 3m
general torus	4,194,304	448,702	226,512	14h 15m	6h 36m

Wall time for mechanism design problems on our cluster and argo.

- Compared to the linear-product bound, polyhedral homotopies cut the #paths about in half.
- The second example is easier (despite the larger #paths) because of increased sparsity, and thus lower evaluation cost.

Final Remarks

Three issues to improve performance of parallel homotopies

- Avoid storing all solutions in main memory.
- Numerical stability matters even more.
- Fast quality control of large solution lists.

An ambitious Swap of Letters:

PHC = Polynomial Homotopy Continuation

HPC = High Performance Computing

towards High Performance *Continuation*

the end