Parallelization of Triangular Decompositions

Marc Moreno Maza & Yuzhen Xie

Ontario Research Centre for Computer Algebra (ORCCA)
University of Western Ontario, Canada

SHARCNET Fall Workshop 2006, Waterloo

Solving polynomial systems symbolically . . .

- Polynomial systems :
  - systems of non-linear algebraic (or differential) equations,
  - solving them is a fundamental problem in mathematical sciences,
  - which is hard for both numerical and symbolic approaches.
- Symbolic solving :
  - provides exact answers,
  - but suffers from expression swell.
- Applications of symbolic solving :
  - increasing number of applications (cryptology, robotics, geometric modeling, dynamical systems in biology, . . .)
  - can now compete with numerical solving (real solving)
  - sometimes, this is the only way to go (parametric solving, solving over finite fields).

Solving polynomial systems . . .

$$\begin{cases} x^2 + y + z &= 1 \\ x + y^2 + z &= 1 \\ x + y + z^2 &= 1 \end{cases}$$

The output with phc the symb.-num. software of J. Verschelde:

```
solution 1 :    start residual :  3.968E-12   #iterations : 1   success
 x :  9.99999695984909E-01   4.13938269379988E-07
 y :  3.04015091103714E-07  -4.13938269379988E-07
 z :  3.04015090976779E-07  -4.13938269379988E-07
== err :  2.154E-06 = rco :  1.197E-07 = res :  9.920E-13 = complex regular ==
solution 2 :    start residual :  1.388E-16   #iterations : 1   success
 x :  4.14213562373095E-01   2.35098870164458E-38
 y :  4.14213562373095E-01  -1.67507944992176E-37
 z :  4.14213562373095E-01   1.29304378590452E-37
== err :  7.517E-16 = rco :  6.017E-02 = res :  5.551E-17 = real regular ==
```

```
solution 3 :    start residual :  2.400E-12   #iterations : 1   success
 x :  1.80048038888678E-08   4.29782537417684E-07
 y :  9.99999981995196E-01  -4.29782537417684E-07
 z :  1.80048038262633E-08   4.29782537417684E-07
== err :  1.344E-06 = rco :  7.463E-08 = res :  5.995E-13 = complex regular ==
```

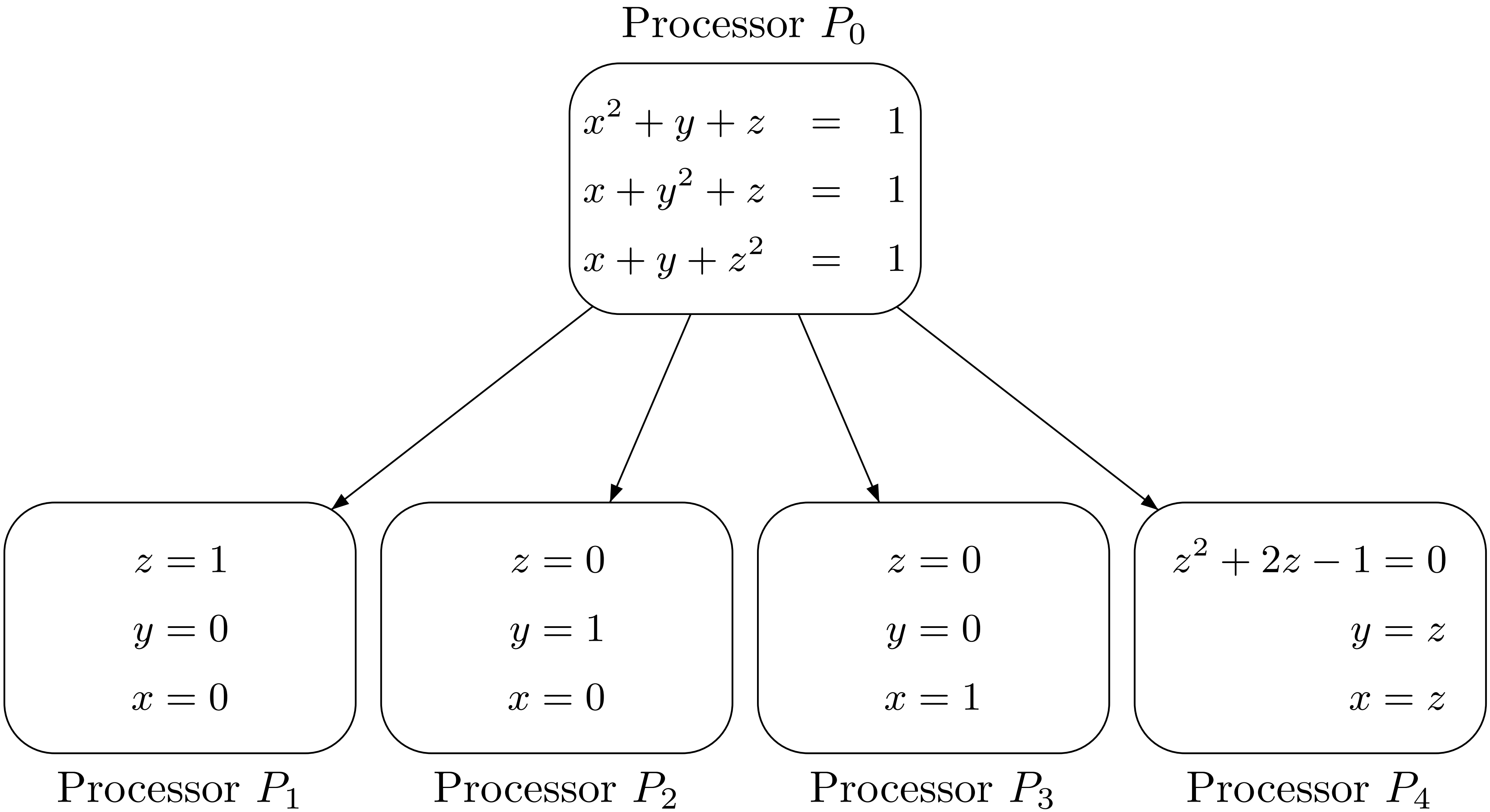
```
solution 4 :    start residual :  9.614E-13   #iterations : 1   success
 x :  1.00000024904061E+00  -3.93267692590196E-08
 y : -2.49040612161639E-07   3.93267692590197E-08
 z : -2.49040612108234E-07   3.93267692590197E-08
== err :  8.657E-07 = rco :  4.806E-08 = res :  2.400E-13 = complex regular ==
solution 5 :    start residual :  2.745E-12   #iterations : 1   success
 x :  3.58839953269127E-07   1.89357516639334E-07
 y :  3.58839953269127E-07   1.89357516639334E-07
 z :  9.99999641160047E-01  -1.89357516639334E-07
== err :  1.645E-06 = rco :  7.071E-08 = res :  6.863E-13 = complex regular ==
solution 6 :    start residual :  1.744E-34   #iterations : 1   success
 x : -2.41421356237309E+00   0.00000000000000E+00
 y : -2.41421356237309E+00   0.00000000000000E+00
 z : -2.41421356237309E+00  -1.00577224408752E-106
== err :  3.611E-35 = rco :  4.142E-01 = res :  6.868E-106 = real regular ==
solution 7 :    start residual :  1.112E-12   #iterations : 1   success
 x : -2.64786238552867E-07  -4.67724648385200E-08
 y : -2.64786238552867E-07  -4.67724648385200E-08
 z :  1.00000026478624E+00   4.67724648385200E-08
== err :  9.341E-07 = rco :  4.530E-08 = res :  2.779E-13 = complex regular ==
solution 8 :    start residual :  2.045E-12   #iterations : 1   success
 x :  1.42636460554469E-07  -3.16738323586431E-07
 y :  9.99999857363539E-01   3.16738323586431E-07
 z :  1.42636460467758E-07  -3.16738323586431E-07
== err :  1.378E-06 = rco :  7.656E-08 = res :  5.117E-13 = complex regular ==
==========================================================================
A list of 8 solutions has been refined :
Number of regular solutions   : 8.
Number of singular solutions  : 0.
Number of real solutions      : 2.
Number of complex solutions   : 6.
Number of clustered solutions : 0.
Number of failures            : 0.
```

Solving polynomial systems symbolically . . .

$$\left\{\begin{array}{l} x^2 + y + z = 1 \\ x + y^2 + z = 1 \\ x + y + z^2 = 1 \end{array}\right. \quad \underline{\text{has Gröbner basis}} :$$

$$\left\{\begin{array}{r} z^6 - 4z^4 + 4z^3 - z^2 = 0 \\ 2z^2y + z^4 - z^2 = 0 \\ y^2 - y - z^2 + z = 0 \\ x + y + z^2 - 1 = 0 \end{array}\right. \quad \underline{\text{and triangular decomposition}} :$$

$$\left\{\begin{array}{l} z = 1 \\ y = 0 \\ x = 0 \end{array}\right. \bigcup \left\{\begin{array}{l} z = 0 \\ y = 1 \\ x = 0 \end{array}\right. \bigcup \left\{\begin{array}{l} z = 0 \\ y = 0 \\ x = 1 \end{array}\right. \bigcup \left\{\begin{array}{r} z^2 + 2z - 1 = 0 \\ y = z \\ x = z \end{array}\right.$$


Processor P_0
$x^2 + y + z = 1$
$x + y^2 + z = 1$
$x + y + z^2 = 1$
$z = 1$
$y = 0$
$x = 0$
Processor P_1
$z = 0$
$y = 1$
$x = 0$
Processor P_2
$z = 0$
$y = 0$
$x = 1$
Processor P_3
$z^2 + 2z - 1 = 0$
$y = z$
$x = z$
Processor P_4

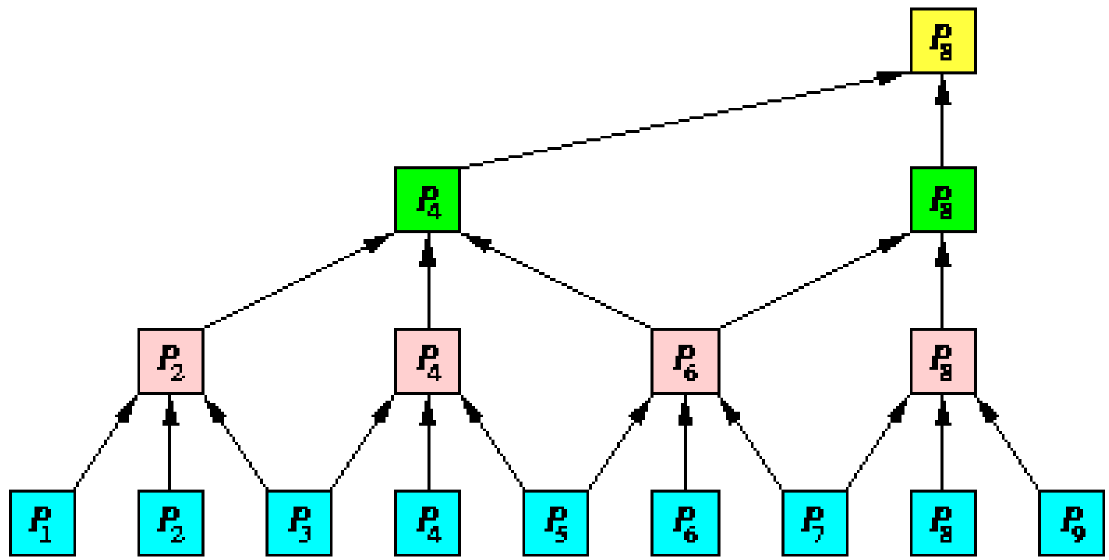
An example of efficient parallelization

Consider a tridiagonal linear system of order n:

$$
\begin{array}{lllllll}
\cdots & \cdots & \cdots & & & & \cdots \\
a_{i-2}x_{i-2} & + b_{i-1}x_{i-1} & + c_i x_i & & & = & e_{i-1} \\
& a_{i-1}x_{i-1} & + b_i x_i & + c_{i+1}x_{i+1} & & = & e_i \\
& & a_i x_i & + b_{i+1}x_{i+1} & + c_{i+2}x_{i+2} & = & e_{i+1} \\
& & & \cdots & \cdots & \cdots & \cdots
\end{array}
$$

For every even i replacing x_i with $-\frac{e_i - c_{i+1}x_{i+1} - a_{i-1}x_{i-1}}{b_i}$ leads to another tridiagonal system of order $n/2$:

$$
\begin{array}{lllllll}
\cdots & \cdots & \cdots & & & & \cdots \\
A_{i-3}x_{i-3} & + B_{i-1}x_{i-1} & + C_{i+1}x_{i+1} & & & = & E_{i-1} \\
& A_{i-1}x_{i-1} & + B_{i+1}x_{i+1} & + C_{i+3}x_{i+3} & & = & E_{i+1} \\
& & \cdots & \cdots & \cdots & & \cdots
\end{array}
$$



Observe that, on this example:

- the number of processors, here $p = n$, can be set such that
- the number of parallel steps, here $O(\log n)$, is known and small,
- processors activity (scheduling) is easy to organize,
- data-communication is not intensive.

Why solving non-linear systems is much more difficult?

Let $F \subset \mathbb{K}[X]$ with $X = x_1 < \cdots < x_n$ and a coefficient field $\mathbb{K}$. Let d be the maximum (total) degree of a monomial in F.

Let $V(F) \subset \overline{\mathbb{K}}^n$ be the zero set of F, where $\overline{\mathbb{K}}$ is an algebraically closed field containing $\mathbb{K}$. For instance $\mathbb{K} = \mathbb{Q}$ and $\overline{\mathbb{K}} = \mathbb{C}$.

- $V(F)$ may consist of components of **different dimension**: points, curves, surfaces, $\ldots$,
- Even if $V(F)$ is finite, it may contain $O(d^n)$ points,
- The idea of *substitution* or *simplification* is much **more complicated** than in the linear case and leads to the notion of a *Gröbner basis*,
- **Large intermediate data**.

Solving polynomial systems symbolically and in parallel!

- Related work :
  - Parallelizing the computation of Gröbner bases (R. Bündgen, M. Göbel & W. Küchlin, 1994) (S. Chakrabarti & K. Yelick, 1993 - 1994) (G. Attardi & C. Traverso, 1996) (A. Leykin, 2004)
  - Parallelizing the computation of characteristic sets (I.A. Ajwa, 1998), (Y.W. Wu, W.D. Liao, D.D. Liu & P.S. Wang, 2003) (Y.W. Wu, G.W. Yang, H. Yang, H.M. Zheng & D.D. Liu, 2005)

What is a Gröbner basis?

• Assume F is a linear system. Then, a **solution** of F is a a **solved system** system S for $x_1 < \cdots < x_n$ which reduces to 0 (i.e. cancels) all polynomials in F. Moreover, up to trivial transformations, the set S is unique.

• Now, assume that F is not linear. Then, a **Gröbner basis** of F is a system B which which **reduces to** 0 all polynomials in the ideal generated by F. Moreover, up to trivial transformations, the set B is unique.

$$\left\{\begin{aligned} x^2 + y + z &= 1 \\ x + y^2 + z &= 1 \\ x + y + z^2 &= 1 \end{aligned}\right. \quad \underline{\text{has Gröbner basis}} : \left\{\begin{aligned} z^6 - 4z^4 + 4z^3 - z^2 &= 0 \\ 2z^2y + z^4 - z^2 &= 0 \\ y^2 - y - z^2 + z &= 0 \\ x + y + z^2 - 1 &= 0 \end{aligned}\right.$$

Parallelizing the computation of Gröbner bases

Input: $F \subset \mathbb{K}[X]$ and an admissible monomial ordering $\leq$.

Output: G a reduced Gröbner basis w.r.t. $\leq$ of the ideal $\langle F \rangle$ generated by F.

repeat

(S) $B :=$ MinimalAutoreducedSubset$(F, \leq)$

(R) $A :=$ S_Polynomials$(B) \cup F$;

$R :=$ Reduce$(A, B, \leq)$

(U) $R := R \setminus \{0\}$; $F := F \cup R$

until $R = \emptyset$

return B

To go further: triangular decompositions

- The zero set $V(F)$ admits a decomposition (unique when minimal)

$$V(F) = V(F_1) \cup \cdots \cup V(F_e),$$

s.t. $F_1, \ldots, F_e \subset \mathbb{K}[X]$ and every $V(F_i)$ **cannot** be decomposed further.

- Moreover, up to technical details, each $V(F_i)$ is the zero set of a **triangular system**, which can view as a **solved system**

$$\begin{cases} T_n(x_1, \ldots, x_d, x_{d+1}, x_{d+2}, \ldots, x_{n-1}, \mathbf{x_n}) = 0 \\ T_{n_1}(x_1, \ldots, x_d, x_{d+1}, x_{d+2}, \ldots, \mathbf{x_{n-1}}) = 0 \\ \vdots \\ T_{d+2}(x_1, \ldots, x_d, x_{d+1}, \mathbf{x_{d+2}}) = 0 \\ T_{d+1}(x_1, \ldots, x_d, \mathbf{x_{d+1}}) = 0 \\ h(x_1, \ldots, x_d) \neq 0 \end{cases}$$

The characteristic set method

Input: $F \subset \mathbb{K}[X]$.

Output: C an autoreduced characteristic set of F (in the sense of Wu).

repeat
(S) $B :=$ MinimalAutoreducedSubset(F, $\leq$)
(R) $A := F \setminus B$;
$R :=$ PseudoReduce(A, B, $\leq$)
(U) $R := R \setminus \{0\}$; $F := F \cup R$
until $R = \emptyset$
return B

- Repeated calls to this procedure computes a decomposition of $V(F)$.
- Cannot start computing the 2nd component before the 1st is completed.

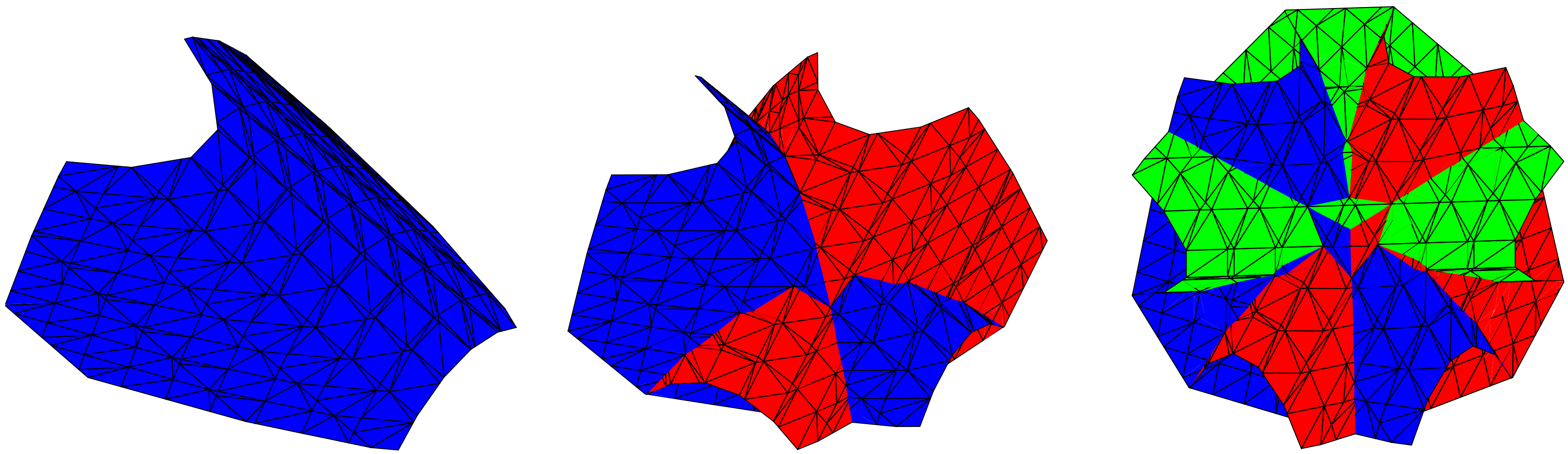
Continue solving polynomial systems symbolically and in parallel!

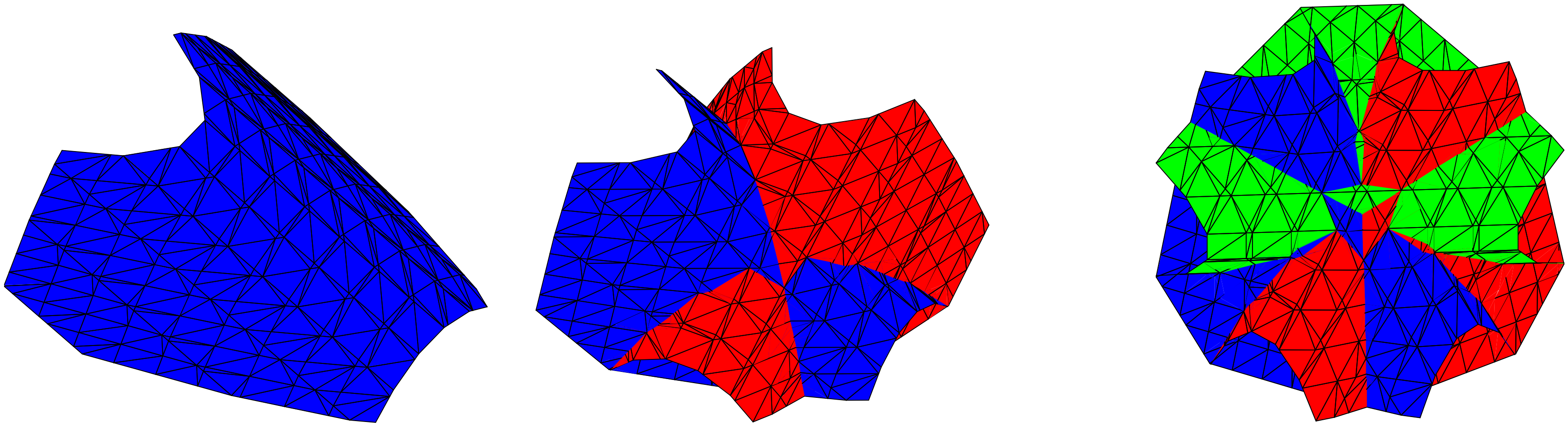
- **New motivations** :
  - revival of parallelism,
  - sharper complexity results for polynomial system solving,
  - new algorithms, modular triangular decompositions, offering better opportunities for parallel execution.
- **Our goal** :
  - to develop a solver for which the number of processes in use depends on the geometry of the solution set (= its intrinsic complexity) of the input system,
  - as a complement to other approaches for parallelizing symbolic computations.

Main approach 1

Incremental solving: by solving one equation after the other, leads to a **more geometric approach**

$$\left\{ x^2+y+z=1 \right. \qquad \left\{ \begin{array}{l} x^2+y+z=1 \\ x+y^2+z=1 \end{array} \right. \qquad \left\{ \begin{array}{l} x^2+y+z=1 \\ x+y^2+z=1 \\ x+y+z^2=1 \end{array} \right.$$





$$\left\{\ x^2 + y + z = 1\right.$$

$$\left\{\begin{array}{lll} x & + & y^2 + z = 1 \\ y^4 & + & (2z-2)y^2 \\ & + & y - z + z^2 = 0 \end{array}\right.$$

$$\left\{\begin{array}{r} x + y = 1 \\ y^2 - y = 0 \\ z = 0 \end{array}\right.$$

$$\left\{\begin{array}{r} 2x + z^2 = 1 \\ 2y + z^2 = 1 \\ z^3 + z^2 - 3z = -1 \end{array}\right.$$

Main approach 2

A task manager algorithm for triangular decompositions: Triade (M. Moreno Maza, 2000)

- A *task* is any couple $[F, T]$ where $F \subset \mathbb{K}[X]$ and $T \subset \mathbb{K}[X]$ is a triangular system.
  - if $F = \emptyset$ the task is *solved*,
  - otherwise we aim at *solving* $[F, T]$, that is, computing triangular systems $T_1, \ldots, T_\ell$ representing $Z(F, T)$ the common zeros of F and T.
- In fact, we shall *solve* $[F, T]$ *lazily* that is, computing tasks $[F_1, T_1], \ldots, [F_\ell, T_\ell]$ such that
  - each $[F_i, T_i]$ is more solved than $[F, T]$,
  - $Z(F_1, T_1) \cup \cdots \cup Z(F_\ell, T_\ell)$ represents $Z(F, T)$,
  - for all i we have $F_i = \emptyset$ whenever T_i has maximum dimension.

Triade **Top level**

Input: $F \subset \mathbb{K}[X]$.

Output: $\mathcal{T}$ a triangular decomposition of $V(F)$.

$ToDo := [[F, \emptyset]; \mathcal{T} := [\,]$

repeat

(S) $Tasks :=$ Select$(ToDo)$

(R) $Results :=$ LazySolve$(Tasks)$

(U) $(ToDo, \mathcal{T}) :=$ Update$(Results, ToDo, \mathcal{T})$

until $ToDo = \emptyset$

return $\mathcal{T}$

A Triade example

$$\begin{cases} f_1 = x - 2 + (y-1)^2 \\ f_2 = (x-1)(y-1) + (x-2)y \\ f_3 = (x-1)z \end{cases}$$

Factorizing f_3 leads to two sub-systems:

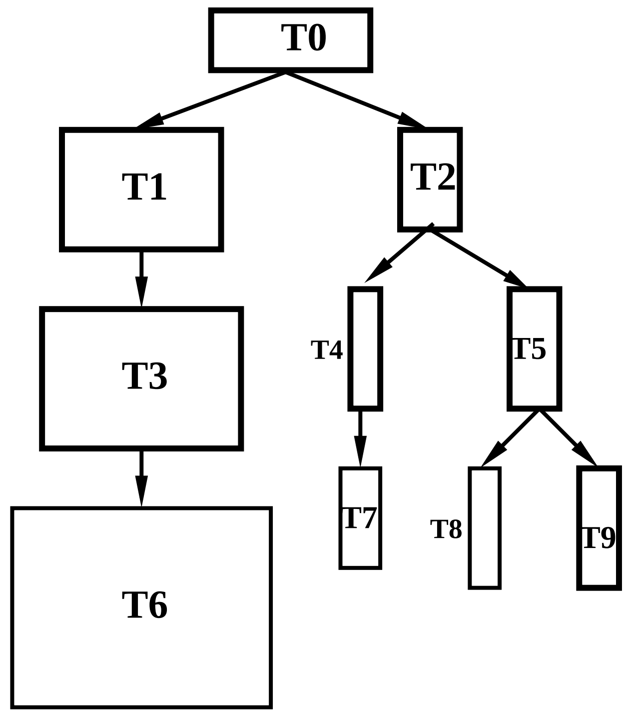
$$S_1 : \begin{cases} y = 0 \\ x = 1 \end{cases} \quad \text{and} \quad S_2 : \begin{cases} x - 1 + y^2 - 2y = 0 \\ (2y-1)x + 1 - 3y = 0 \\ z = 0 \end{cases}$$

The sub-system S_1 is solved. Continuing with S_2 leads finally to

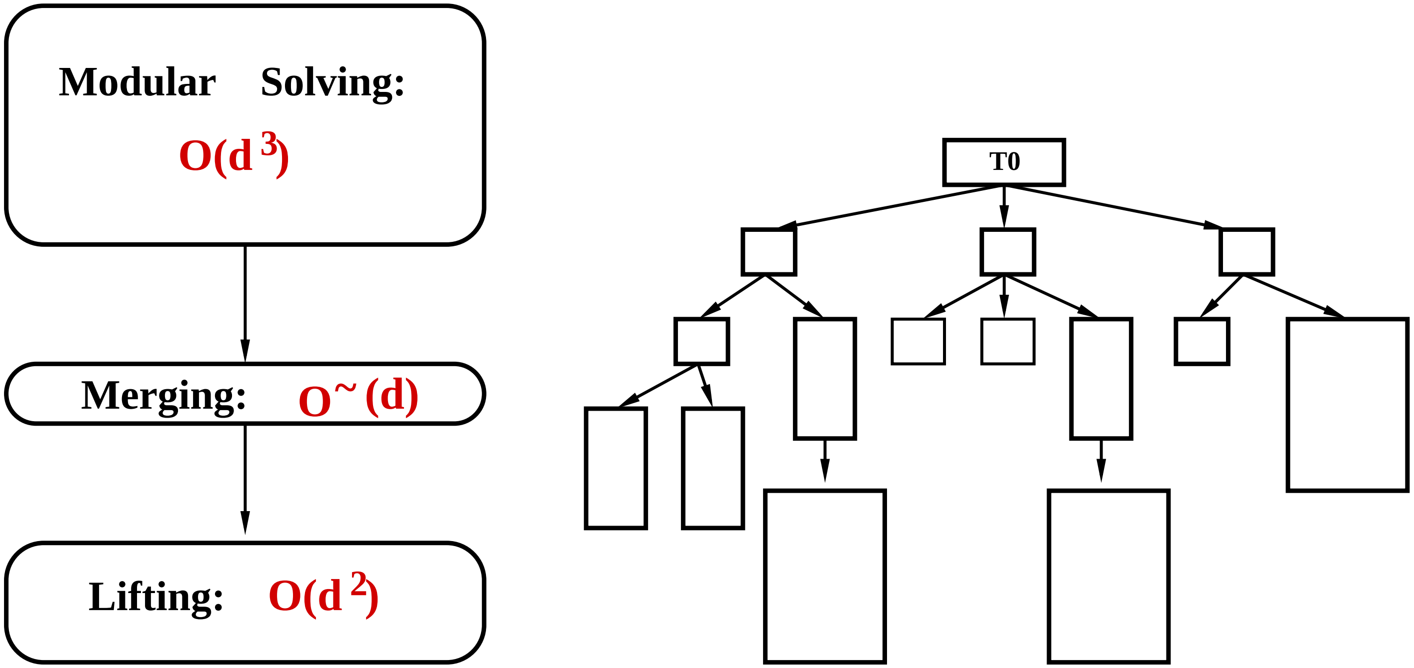
$$\begin{cases} z = 0 \\ y = 0 \\ x = 1 \end{cases}, \begin{cases} z = 0 \\ y = 1 \\ x = 2 \end{cases} \quad \text{and} \quad \begin{cases} z = 0 \\ 2y = 3 \\ 4x = 7 \end{cases}$$

Main difficulties for parallel execution

- Very irregular tasks (CPU time, memory, data-communication)
- For most systems with rational number coefficients, in theory and in practice, the solution set consists of one main piece and possibly a few tiny pieces.



Key solution 1



For solving $F \subseteq \mathbb{Q}[X]$ we use modular methods. Indeed, for a prime p:

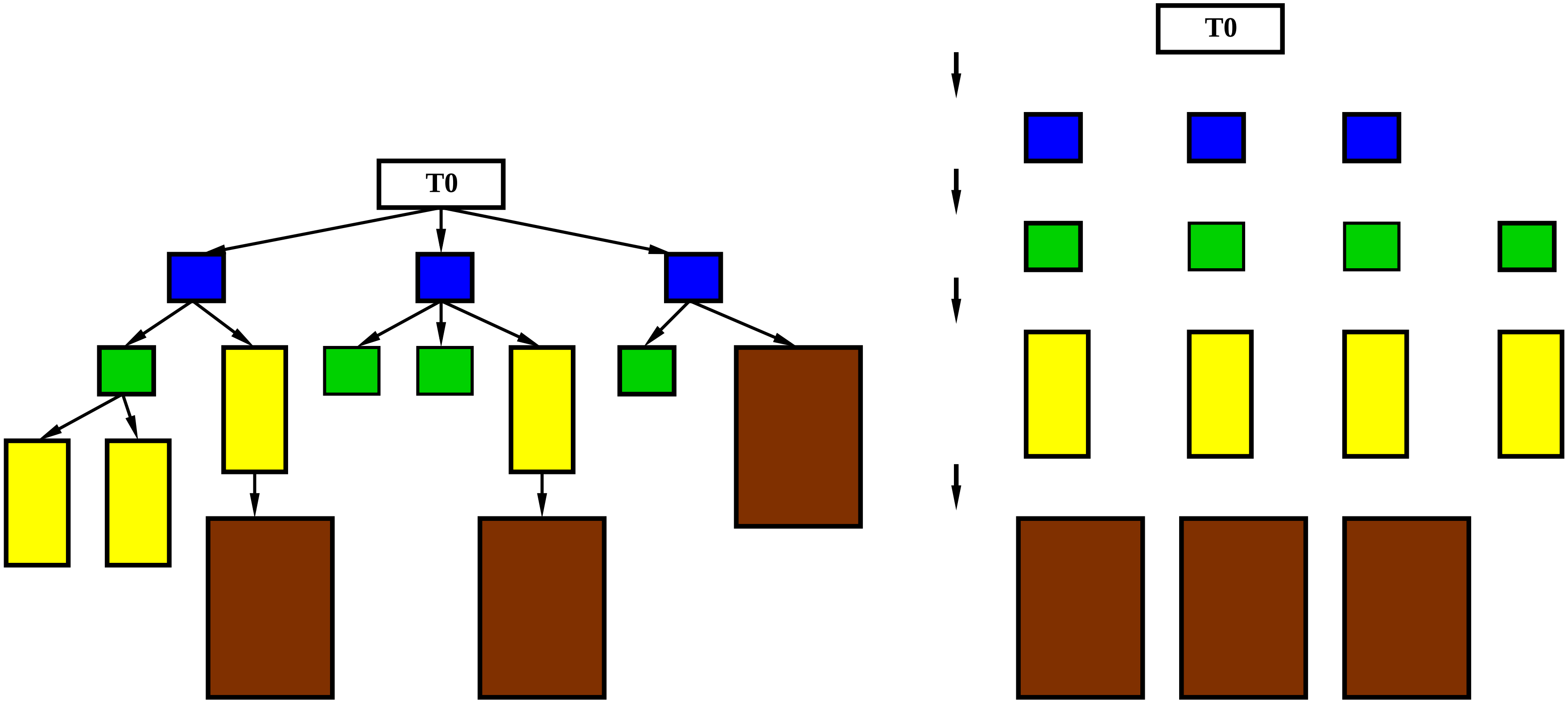
- irreducible polynomials in $\mathbb{Q}[X]$ are likely to factor modulo p,
- for p big enough, the result over $\mathbb{Q}$ can be recovered from the one over $\mathbf{Z}/p\mathbf{Z}[X]$.

(X. Dahan, M. Moreno Maza, É. Schost, W. Wu & Y. Xie, 2005)

Modular solving: some data

Sys	Name	n	d	p	sol. numb.	sol. sizes
1	eco6	6	3	105761	[1,1,2,4,4,4]	[56,57,721,1205,1293,1283]
2	Weispfenning-94	3	5	7433	[2,2,9,35,3,3]	[100,99,282,1048,134,135]
3	Issac97	4	2	1549	[4,7,3,2]	[561,749,458,334]
4	dessin-2	10	2	358079	[1,1,6,12,22]	[98,98,885,1100,1448]
5	eco7	7	3	387799	[1,1,1,1,4,2,	[67,72,73,76,4776,2603,
					4,4,4,4,4,2]	4770,4755,4770,4751,4764,2601]
6	Methan61	10	2	450367	[1,1,1,3,18,3]	[109,105,106,961,2307,957]
7	Reimer-4	4	5	55313	[1,1,1,1,4,4,24]	[35,35,35,35,350,352,868]
8	Uteshev-Bikker	4	3	7841	[1,1,1,1,2,30]	[16,27,32,27,472,2006]
9	gametwo5	5	4	159223	[14,19,11]	[2811,2987,2700]

Key solution 2



For solving $F \subseteq \mathbf{Z}/p\mathbf{Z}[X]$, we combine dimension theory and the lazy solving techniques of the Triade:

⇒ triangular systems are generated by decreasing order of dimension,

⇒ the hardest tasks to solve are postponed and can be solved in a single parallel step.

Challenges in the implementation

- dynamic process creation and management,
- scheduling of highly irregular tasks,
- complex data types, such as the polynomial data type,
- heavy data-communication,
- synchronization in heterogeneous environment.

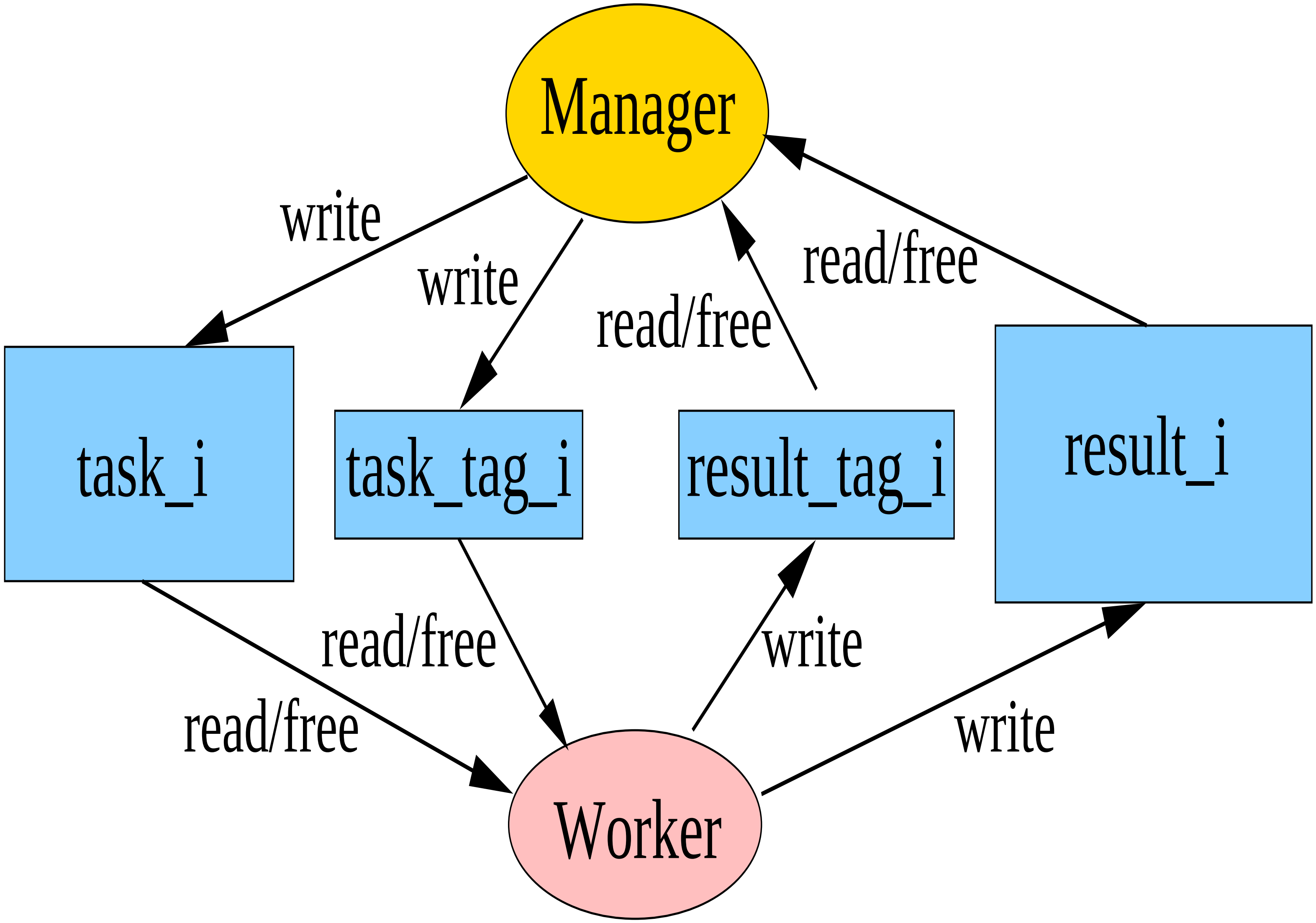
Preliminary implementation

- Environment:
  - in the ALDOR language (high-level language, designed for symbolic computations)
  - on an Authentic AMD with 4 CPUs (2390 MHz) and 8 GB total memory
  - using multi-processed parallelism.
  - using shared memory segments for data communication.
- Limitations in this implementation:
  - Only 4 CPUs ...
  - management of process workers quite rudimentory imposing heavy load on process creation and data communication
  - during the modular solving, computations are not split yet as much as they could.

Meet the challenges in the implementation

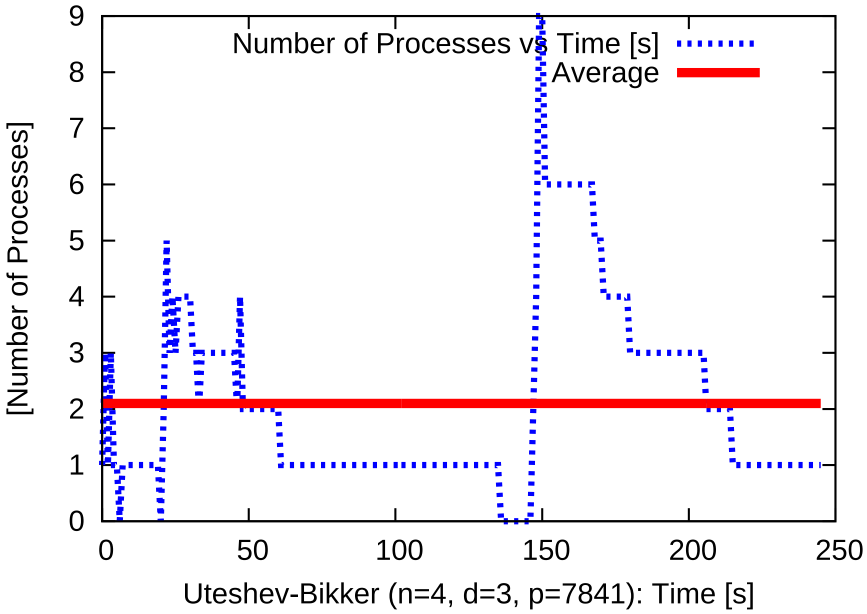
- The ALDOR language and available libraries do not provide support for parallelism
  - Thanks to its interoperability with C, we could implement an ALDOR type `SharedMemorySegment` for exchanging an array of machine integers.
- The Manager and Workers need to exchange Triade tasks $[F, T]$, that is collections of multivariate polynomials through `SharedMemorySegment`.
  - We use efficient conversions (based on Kronecker's substitution and others) between polynomials and C arrays of `int`.
  - Synchronization for data communication between each worker and the Manager is controlled by four `SharedMemorySegment`s and an access sequence defined by our protocol.

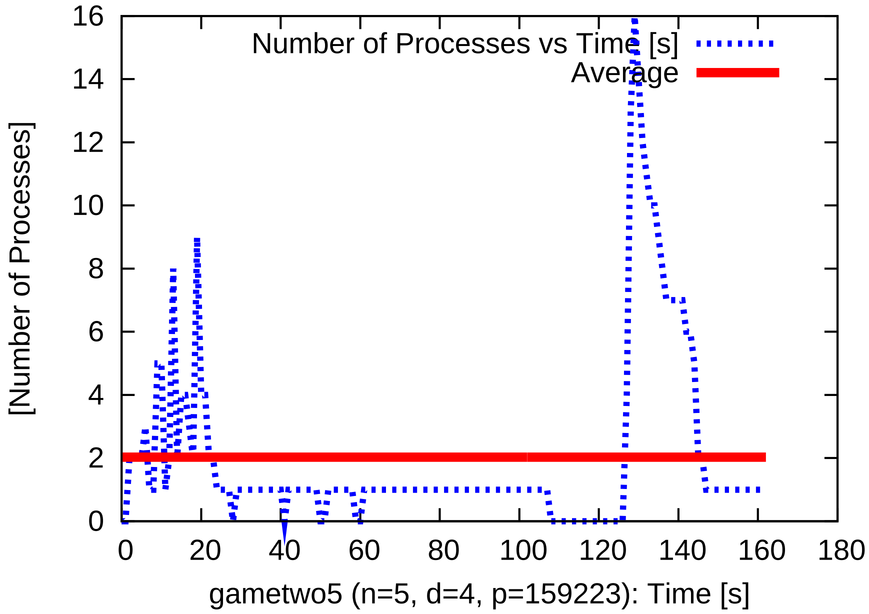
Implementation/Synchronization Scheme



Current results

- Promising experimental results: a speedup ratio of 1.5 to 2.0 w.r.t. the comparable sequential solver for some examples having 2 process workers on average through the entire solving process.





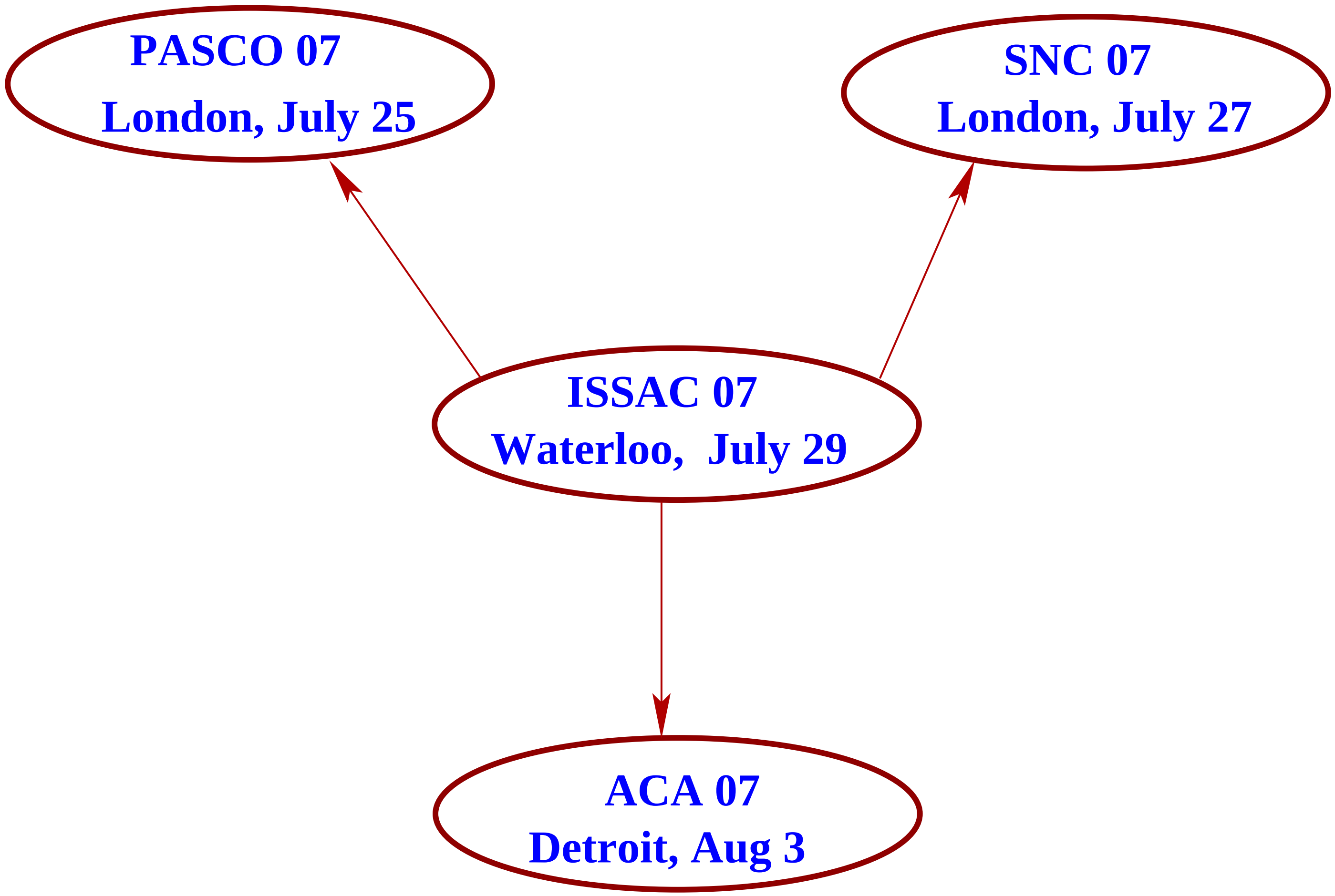
Conclusions and work in progress

- Conclusions :
  - Combining geometrical considerations (modular methods, dimension theory) and lazy evaluation techniques,
  - we have achieved successful coarse-grain parallelization of triangular decompositions,
  - such that the number of working processors depend on the geometry of the input system.
- Work in progress :
  - porting to machines with more processors, like Silky (64 bits 128-processor SMP)
  - implementing a finer scheduler (management of workers, data communication)
  - gaining more from the modular solving phasis.

Future plan

- develop a model for threads in Aldor to support parallelism for symbolic computations targeting SMP and multi-cores.
  - in particular, parametric types, such as polynomial data types, shall be properly treated.
- investigate multi-level parallel algorithms for triangular decompositions of polynomial systems.
  - coarse grained level for tasks to compute geometric of the solution sets.
  - medium/fine grained level for polynomial arithmetic such as multiplication, GCD/resultant, and factorization.
  - hence, to gain higher scalability.

Propaganda


PASCO 07
London, July 25
SNC 07
London, July 27
ISSAC 07
Waterloo, July 29
ACA 07
Detroit, Aug 3