# Workpackage 5:
# High Performance Mathematical Computing

Second OpenDreamKit Project review

Luxembourg, October 30, 2018

# High performance mathematical computing

## Mathematical computing

Computing with a large variety of objects

- $\mathbb{Z}, \mathbb{Q}, \mathbb{Z}/p\mathbb{Z}, \mathbb{F}_q,$ 17541718814389012164632

for applications where all digits matter.

# High performance mathematical computing

## Mathematical computing

Computing with a large variety of objects

- $\mathbb{Z}, \mathbb{Q}, \mathbb{Z}/p\mathbb{Z}, \mathbb{F}_q,$ $\qquad$ $175417188143890121646 32$
- Polynomials over $\mathbb{Z}, \mathbb{Q}, \mathbb{Z}/p\mathbb{Z}, \mathbb{F}_q,$ $\qquad$ $\frac{2}{5}x^3 + x^2 - \frac{1}{19}x + 2$

for applications where all digits matter.

# High performance mathematical computing

## Mathematical computing

Computing with a large variety of objects

- $\mathbb{Z}, \mathbb{Q}, \mathbb{Z}/p\mathbb{Z}, \mathbb{F}_q,$ $\quad 175417188143890121646432$
- Polynomials over $\mathbb{Z}, \mathbb{Q}, \mathbb{Z}/p\mathbb{Z}, \mathbb{F}_q,$ $\quad \frac{2}{5}x^3 + x^2 - \frac{1}{19}x + 2$
- Matrices over $\mathbb{Z}, \mathbb{Q}, \mathbb{Z}/p\mathbb{Z}, \mathbb{F}_q,$ $\quad \begin{bmatrix} 27 & 3 & -1 \\ 9 & 0 & 2 \end{bmatrix}$

for applications where all digits matter.

# High performance mathematical computing

## Mathematical computing

Computing with a large variety of objects

- $\mathbb{Z}, \mathbb{Q}, \mathbb{Z}/p\mathbb{Z}, \mathbb{F}_q,$ $\quad 17541718814389012164632$
- Polynomials over $\mathbb{Z}, \mathbb{Q}, \mathbb{Z}/p\mathbb{Z}, \mathbb{F}_q,$ $\quad \frac{2}{5}x^3 + x^2 - \frac{1}{19}x + 2$
- Matrices over $\mathbb{Z}, \mathbb{Q}, \mathbb{Z}/p\mathbb{Z}, \mathbb{F}_q,$ $\quad \begin{bmatrix} 27 & 3 & -1 \\ 9 & 0 & 2 \end{bmatrix}$
- Matrices of polynomials over $\mathbb{Z}, \mathbb{Q}, \mathbb{Z}/p\mathbb{Z}, \mathbb{F}_q,$ $\quad \begin{bmatrix} 3x^2+3 & 2x^2+3 \\ 4x^2+1 & x^2+4x+4 \end{bmatrix}$

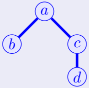
for applications where all digits matter.

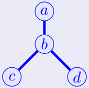
# High performance mathematical computing

## Mathematical computing

Computing with a large variety of objects

- $\mathbb{Z}, \mathbb{Q}, \mathbb{Z}/p\mathbb{Z}, \mathbb{F}_q,$ $\quad 17541718814389012164632$
- Polynomials over $\mathbb{Z}, \mathbb{Q}, \mathbb{Z}/p\mathbb{Z}, \mathbb{F}_q,$ $\quad \frac{2}{5}x^3 + x^2 - \frac{1}{19}x + 2$
- Matrices over $\mathbb{Z}, \mathbb{Q}, \mathbb{Z}/p\mathbb{Z}, \mathbb{F}_q,$ $\quad \begin{bmatrix} 27 & 3 & -1 \\ 9 & 0 & 2 \end{bmatrix}$
- Matrices of polynomials over $\mathbb{Z}, \mathbb{Q}, \mathbb{Z}/p\mathbb{Z}, \mathbb{F}_q,$ $\quad \begin{bmatrix} 3x^2+3 & 2x^2+3 \\ 4x^2+1 & x^2+4x+4 \end{bmatrix}$

$\frac{3q^2-q^5}{q^5+2q^4+3q^3+3q^2+2q+1}$ 

$+ \frac{2q}{q^4+q^3+2q^2+q+1}$ 


- Tree algebra

for applications where all digits matter.

# High performance mathematical computing

**Need for High performance:** applications where size matters:

Experimental maths: testing conjectures

- larger instances give higher confidence

# High performance mathematical computing

**Need for High performance:** applications where size matters:

Experimental maths: testing conjectures

- larger instances give higher confidence

Algebraic cryptanalysis: security = computational difficulty

- key size determined by the largest solvable problem

**Example**

Breaking RSA by integer factorization: $n = pq$. Last record:

- $n$ of 768 bits
- linear algebra in dimension $192\,796\,550$ over $\mathbb{F}_2$ (105Gb)
- About 2000 CPU years

# High performance mathematical computing

**Need for High performance:** applications where size matters:

Experimental maths: testing conjectures

- larger instances give higher confidence

Algebraic cryptanalysis: security = computational difficulty

- key size determined by the largest solvable problem

**Example**

Breaking RSA by integer factorization: $n = pq$. Last record:

- $n$ of 768 bits
- linear algebra in dimension $192\,796\,550$ over $\mathbb{F}_2$ (105Gb)
- About 2000 CPU years

3D data analysis, shape recognition:

- via persistent homology
- large sparse matrices over $\mathbb{F}_2$, $\mathbb{Z}$

# Goal: delivering high performance to maths users

**Systems :** GAP, PARI/GP, SageMath, Singular

**Components :** FLINT, MPIR, LinBox, PPL, NumPy

# Goal: delivering high performance to maths users

## Harnessing modern hardware ⇝ parallelisation

- in-core parallelism (SIMD vectorisation)
- multi-core parallelism
- distributed computing: clusters, cloud

**Systems :** GAP, PARI/GP, SageMath, Singular

**Components :** FLINT, MPIR, LinBox, PPL, NumPy

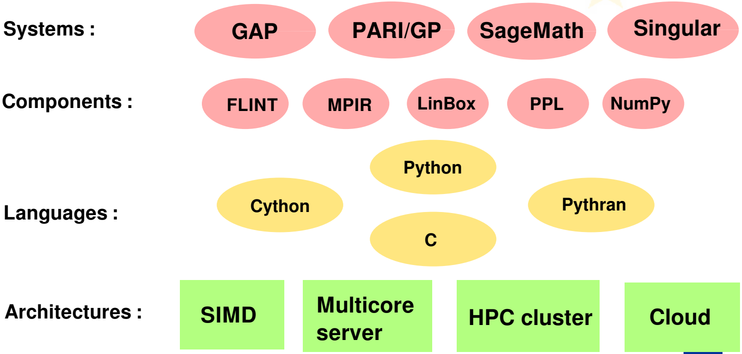
**Architectures :** SIMD, Multicore server, HPC cluster, Cloud

# Goal: delivering high performance to maths users

## Languages

- Computational Maths software uses high level languages (e.g. Python)
- High performance delivered by languages close to the metal (C, assembly)

$\rightsquigarrow$ compilation, automated optimisation



# High performance mathematical computing

## Goal:

- Improve/Develop parallelization of software components
- Expose them through the software stack
- Offer High Performance Computing to VRE's users

## Milestone M8: Seamless use of parallel computing architecture in the VRE (proof of concept)

*Astrid wants to run compute intensive routines involving both dense linear algebra and combinatorics. She has access through a JupyterHub-based VRE to a high end multi-core machine which includes a vanilla SAGE installation. She automatically benefits from the HPC features of the underlying specialized libraries (LinBox, ...). This is a proof of concept of the overall framework to integrate the HPC advances of specialized libraries into a general purpose VRE. It will prepare the final integration of a broader set of such parallel features for the end of the project*

# Addressing recommendations of review 1

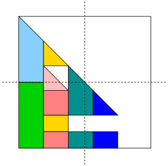
**Recommendation 10:** *Regarding WP5,* **make contacts** *with HPC community in order to ascertain current state-of-the-art. The work in this WP needs to be* **nearer the leading edge**.

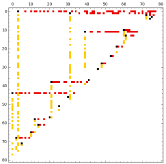
# Addressing recommendations of review 1

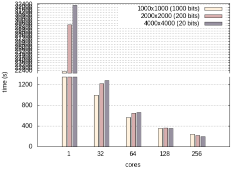
**Recommendation 10:** *Regarding WP5,* **make contacts** *with HPC community in order to ascertain current state-of-the-art. The work in this WP needs to be* **nearer the leading edge***.*

## Leading edge achievements in linear algebra

- symmetric factorization outperforms LAPACK implementation
- new non-hierarchical generator for quasiseparable matrices
- large scale parallelization of rational linear solver





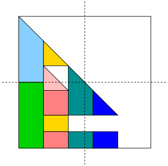



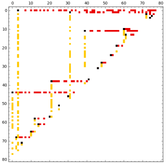
# Addressing recommendations of review 1

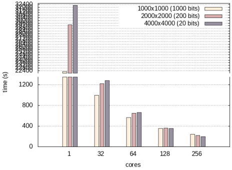
**Recommendation 10:** *Regarding WP5,* **make contacts** *with HPC community in order to ascertain current state-of-the-art. The work in this WP needs to be* **nearer the leading edge***.*

## Leading edge achievements in linear algebra

- symmetric factorization outperforms LAPACK implementation
- new non-hierarchical generator for quasiseparable matrices
- large scale parallelization of rational linear solver






## Interactions and contacts made:

- interaction with the `BLIS` group (vectorization and implementation of Strassen's algorithm)
- on-going collaboration with T. Mary (`Mumps`) and S. Chandrasekaran (UCSB) on quasiseparable matrix algorithmic

# Outline

## Deliverables under review for the period

- D5.12: Exact linear algebra algorithms and implementations.
- D5.11: Refactor and optimize Sage's Combinatorics

## Progress report on other deliverables

- T5.1: Pari
- T5.2: GAP
- T5.3: LinBox
- T5.4: Singular

## Workpackage management

- Milestone M8
- Strenghtening interactions with numerical HPC

# Outline

## Deliverables under review for the period

- D5.12: Exact linear algebra algorithms and implementations.
- D5.11: Refactor and optimize Sage's Combinatorics

## Progress report on other deliverables

- T5.1: Pari
- T5.2: GAP
- T5.3: LinBox
- T5.4: Singular

## Workpackage management

- Milestone M8
- Strenghtening interactions with numerical HPC

# Task 5.3: LinBox, High performance exact linear algebra

*Mathematics is the art of reducing any problem to linear algebra*

*– W. Stein*

## Linear algebra: a HPC building block

Similarly as in numercial HPC:

- central elementary problem to which others reduce to
- (rather) simple algorithmic
- high compute/memory intensity

## Specificities

- Multiprecision arithemtic $\Rightarrow$ lifting from finite precision ($\mathbb{F}_p$)
- Rank deficiency $\Rightarrow$ unbalanced dynamic blocking
- Early adopter of subcubic matrix arithemtic $\Rightarrow$ recursion

# D5.12: Exact linear algebra algorithms and implementations. Library maintenance and close integration in mathematical software for LinBox library

1. Algorithmic innovations:
   1.1 Rank deficient dense Gaussian elimination
   1.2 Quasiseparable matrices
   1.3 Outsourced computing security
2. Software releases and integration:
   2.1 LinBox ecosystem: `LinBox`, `fflas-ffpack`, `givaro`
   2.2 `SageMath` integration

# Rank deficient dense Gaussian elimination

[JSC'17] Fast computation of the rank profile matrix and the generalized Bruhat decomposition.

- Connecting Rank Profile Matrix and row and column echelon forms
- $O\tilde{}(r^\omega + mn)$ probabilistic time
- generalization over arbitrary rings

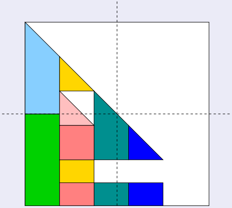
# Rank deficient dense Gaussian elimination

## [JSC'17] Fast computation of the rank profile matrix and the generalized Bruhat decomposition.

- Connecting Rank Profile Matrix and row and column echelon forms
- $O\tilde{}(r^\omega + mn)$ probabilistic time
- generalization over arbitrary rings

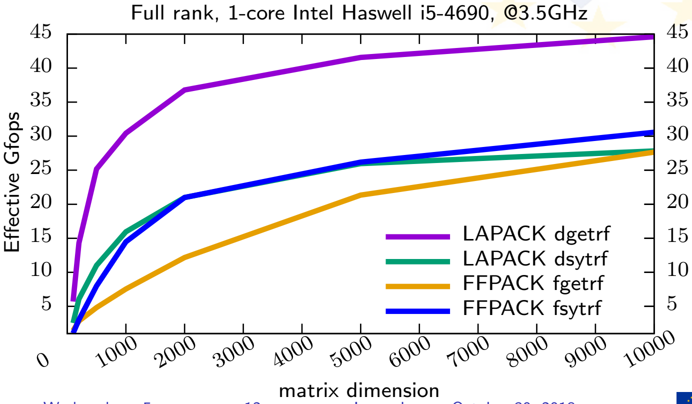
## [ISSAC'18] Symmetric triangular factorization

- First unconditional recursive algorithm
- Pivoting revealing the Rank Profile Matrix
- $O(n^2 r^{\omega-2})$ $(= 1/3 n^3$ with $\omega = 3, r = n)$



# LAPACK vs FFPACK modulo 8 388 593

| | LAPACK | | FFPACK | |
|---|---|---|---|---|
| $n$ | dgetrf (LU) | dsytrf (LDLT) | fgetrf (LU) | fsytrf (LDLT) |
| 5000 | 2.01s | 1.60s | 3.90s | 1.59s |
| 10000 | 14.95s | 11.98s | 24.12s | 10.90s |

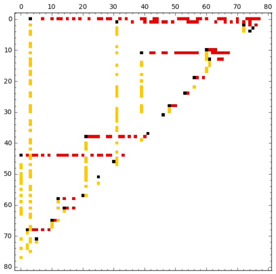


# Quasiseparable matrices

*Matrices with low off-diagonal rank*

## [ISSAC'16, JSC'18] New compact representation and algorithms

- Connection with rank profile matrix
- Matches the best space complexities: $O(ns)$
- Reduction to matrix multiplication: $O(ns^{\omega-1})$ for products
- Flat representation (non hierarchical)

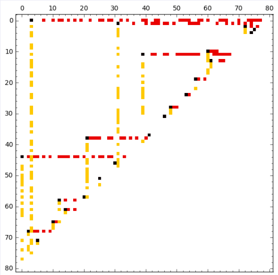


# Quasiseparable matrices

*Matrices with low off-diagonal rank*

## [ISSAC'16, JSC'18] New compact representation and algorithms

- Connection with rank profile matrix
- Matches the best space complexities: $O(ns)$
- Reduction to matrix multiplication: $O(ns^{\omega-1})$ for products
- Flat representation (non hierarchical)




Follow-up: on-going collaboration with numerical HPC experts:

- S. Chandrasekaran (UCSB)
- T. Mary (U. Manchester, `Mumps`)

# Outsourced computing security

## Exploratory aspect: Computation over the Cloud

Outsourcing computation on the cloud:

- trusted lightweight client computer
- untrusted powerful cloud server

$\Rightarrow$ need for certification protocols

Multiparty computation:

- each player contribute with a share of the input
- shares must remain private

# Outsourced computing security

## Exploratory aspect: Computation over the Cloud

Outsourcing computation on the cloud:

- trusted lightweight client computer
- untrusted powerful cloud server

$\Rightarrow$ need for certification protocols

Multiparty computation:

- each player contribute with a share of the input
- shares must remain private

## Contribution

ISSAC'17: Linear time certificates for LU, Det., Rank Profile matrix, etc

In submission: Secure multiparty Strassen's algorithm

# Software releases and integration

## LinBox ecosystem

`givaro`: field/ring arithmetic

`fflas-ffpack`: dense linear algebra over finite field

`LinBox`: exact linear algebra

Tightly integrated in `SageMath`

# Software releases and integration

## LinBox ecosystem

| | |
|---|---|
| `givaro`: field/ring arithmetic | 4 releases |
| `fflas-ffpack`: dense linear algebra over finite field | 6 releases |
| `LinBox`: exact linear algebra | 6 releases |
| Tightly integrated in `SageMath` | 13 tickets |

# Software releases and integration

## LinBox ecosystem

| | |
|---|---|
| `givaro`: field/ring arithmetic | 4 releases |
| `fflas-ffpack`: dense linear algebra over finite field | 6 releases |
| `LinBox`: exact linear algebra | 6 releases |
| Tightly integrated in `SageMath` | 13 tickets |

## Featuring

- Full functional implementations of new algorithmic contributions
- Improved vectorization and parallel routines
- Drastic improvement of reliability (continuous integration, test-suite coverage, randomized certificates, etc)

# Task 5.6: HPC infrastructure for Combinatorics

## General situation

Given a set $S$ of combinatorial objects we want to

- test conjectures: *e.g.* find an element of $S$ satisfying a certain property
- count or list the elements of $S$ having this property

## Specificities of combinatorics

Typically

- **huge**: $S$ does not fit in memory
- **embarassingly parallel**: $S = S_1 \cup \ldots \cup S_k$
- **unbalanced**: $S_i$ sizes are highly unbalanced

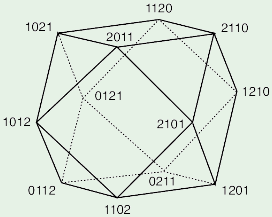
# D5.11: Refactor and optimise the existing combinatorics Sage code using the new developed Pythran and Cython features.

1. Algorithmic innovations, software integration and experimentations through examples
2. Software releases and integration:

# Concrete mathematical problem 1: polytopes

## Example

Some combinatorial objects can be encoded as integer vectors in polytopes.




- Efficient algorithms available (Double description, Barvinok)
- High performance libraries implementing them `PPL`, `Normaliz`
- Useful for many combinatorial applications

# High performance polytope code

## ODK outcomes

- `pplpy` library: Python interface to the high performance Parma Polyhedra Library (rational polytopes)
- `e-antic` library: C/C++ library for computations over embedded number fields. Building block for multithread polytope computations in Normaliz over number fields.
- *Sage Days 84* workshop

# Concrete mathematical problem 2: semigroups

## Example

Numerical semigroup are certain subsets of non-negative integers.

$$
\begin{tikzcd}[column sep=small]
 & \langle 1 \rangle \arrow[d, dash] & & \\
 & \langle 2,3 \rangle \arrow[dl, dash] \arrow[dr, dash] & & \\
\langle 2,5 \rangle \arrow[d, dash] & & \langle 3,4,5 \rangle \arrow[dl, dash] \arrow[d, dash] \arrow[dr, dash] & \\
\langle 2,7 \rangle & \langle 3,4 \rangle & \langle 3,5,7 \rangle & \langle 4,5,6,7 \rangle \\
\vdots & \vdots & \vdots & \vdots
\end{tikzcd}
$$

- Many mathematical open mathematical questions.
- Highly unbalanced tree.

# Work stealing map-reduce

## A Python implementation

- Work stealing algorithm (Leiserson-Blumofe)
- Easy to use, easy to call from SageMath with many use cases
- Scale well with the number of CPU cores and reasonably efficient (given that it is Python code).

A typical speedup obtained for binary sequences

| # processors | 1 | 2 | 4 | 8 |
|---|---|---|---|---|
| Time (s) | 250 | 161 | 103 | 87 |

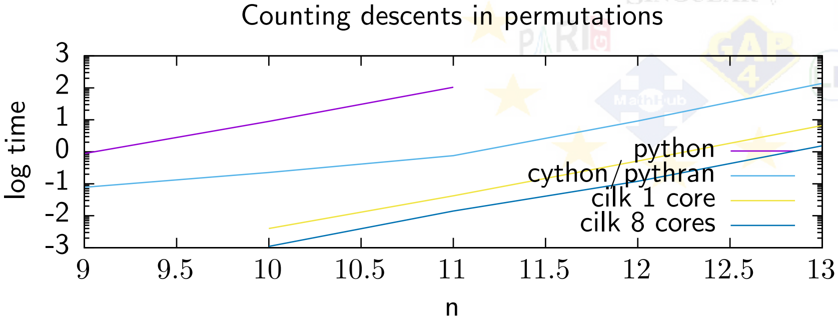
# Low-level parallelization for small combinatorial objects

## HPCombi: an experimental C++ library

- innovative SIMD code for combinatorics
- fine grained parallelization using Intel Cilk technology
- already in use for mathematical projects (numerical semigroups)

| Operation | Speedup |
|---|---|
| Sum of a vector of bytes | $\times 3.81$ |
| Sorting a vector of bytes | $\times 21.3$ |
| Inverting a permutation | $\times 1.97$ |
| Number of cycles of a permutation | $\times 41.5$ |
| Number of inversions of a permutation | $\times 9.39$ |
| Cycle type of a permutation | $\times 8.94$ |

# Exploration: Python, Cython, Pythran and Cilk (D5.8)




- Python (interpreted language) slow.
- Cython/Pythran (Python to C compilers) efficient.
- Cilk (SIMD and multithread parallelism) very promising.

# Software releases and integration

## Libraries releases and integration

`SageMath`: improvement of native code and libraries interfaces

`HPCombi`: C++ library for small combinatorial objects using SIMD instructions and fine grained parallelization (Cilk)

`e-antic` : C/C++ library for (arbitrary precision) embedded number field computations

`pplpy`: Python library interface to PPL library on Polytope

# Software releases and integration

## Libraries releases and integration

`SageMath`: improvement of native code and libraries interfaces — 5 tickets

`HPCombi`: C++ library for small combinatorial objects using SIMD instructions and fine grained parallelization (Cilk) — experimental

`e-antic` : C/C++ library for (arbitrary precision) embedded number field computations — planned release

`pplpy`: Python library interface to PPL library on Polytope — 6 releases

# Software releases and integration

## Libraries releases and integration

`SageMath`: improvement of native code and libraries interfaces — 5 tickets

`HPCombi`: C++ library for small combinatorial objects using SIMD instructions and fine grained parallelization (Cilk) — experimental

`e-antic` : C/C++ library for (arbitrary precision) embedded number field computations — planned release

`pplpy`: Python library interface to PPL library on Polytope — 6 releases

## Featuring

- SIMD instructions for combinatorial enumeration
- Generic map-reduce in SageMath
- Faster and more parallel code inside SageMath

# Outline

Deliverables under review for the period
- D5.12: Exact linear algebra algorithms and implementations.
- D5.11: Refactor and optimize Sage's Combinatorics

Progress report on other deliverables
- T5.1: Pari
- T5.2: GAP
- T5.3: LinBox
- T5.4: Singular

Workpackage management
- Milestone M8
- Strenghtening interactions with numerical HPC

# T5.1: Pari

## D5.16: Pari Suite release, fully supporting parallelization

- D5.10 (merged in D5.16):
  Generic parallelization engine is now mature (released since nov.2016).
  Support POSIX-threads and MPI.
- Current work: applying it throughout the library
  - Chinese remaindering
  - Rational linear algebra
  - Discrete logarithm
  - Resultants
  - APRCL primality testing

# T5.2: GAP

## D5.15: Final report of GAP development

- 8 releases were cut integrating contributions of D3.11 and D5.15
- Towards an integration of HPC-GAP: main release GAP-4.9
  - Build system refactoring
  - Ability to compile in HPC-GAP compatibility mode
- Work in progress:
  - Multithreaded linear algebra: at the level of the `Meataxe` library
  - Introspection functionalities: on-the-fly optimisation decision

# T5.3 LinBox

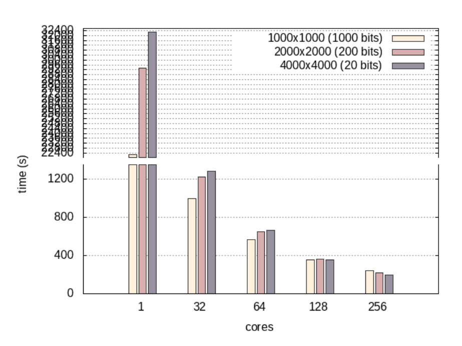
## D5.14: Distributed exact linear system solving

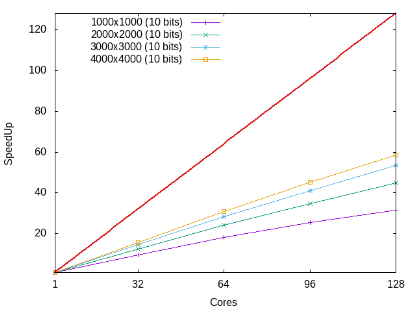
- 2 full time engineers
- Communication and serialization layer done
- Prototype MPI parallelization of Chinese remainder based solver.

# T5.3 LinBox

## D5.14: Distributed exact linear system solving

- 2 full time engineers
- Communication and serialization layer done
- Prototype MPI parallelization of Chinese remainder based solver.



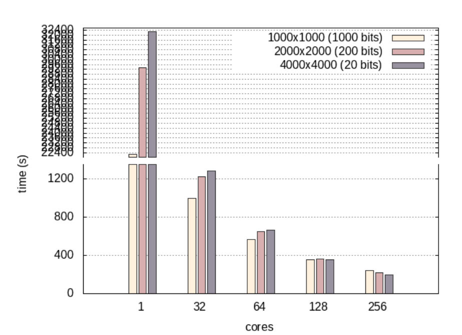



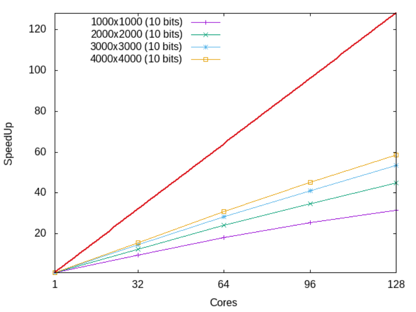
# T5.3 LinBox

## D5.14: Distributed exact linear system solving

Roadmap:

- Major refactorization of LinBox solver code under way
- Parallelization of Dixon-lifting solver
- Hybrid combination of CRT+Dixon.
- Hyrbid OpenMP-MPI implementation



# T5.4 Singular

## D5.13: Parallel sparse polynomial multiplication

`FLINT` now supports fast sparse multivariate polynomials:

- addition, subtraction, multiplication,
- division, division with remainder, GCD
- evaluation (and partial evaluation), composition

# T5.4 Singular

## D5.13: Parallel sparse polynomial multiplication

`FLINT` now supports fast sparse multivariate polynomials:

- addition, subtraction, multiplication,
- division, division with remainder, GCD
- evaluation (and partial evaluation), composition

**Parallelization of the (sparse) multiplication**

| threads | time (ms) | speedup |
|---|---|---|
| 1 | 148661 | x1.0 |
| 2 | 76881 | x1.9 |
| 3 | 54798 | x2.7 |
| 4 | 42855 | x3.4 |
| 5 | 37017 | x4.0 |
| 6 | 30892 | x4.8 |
| 7 | 28365 | x5.2 |
| 8 | 28048 | x5.3 |

# T5.4 Singular

## D5.13: Parallel sparse polynomial multiplication

`FLINT` now supports fast sparse multivariate polynomials:

- addition, subtraction, multiplication,
- division, division with remainder, GCD
- evaluation (and partial evaluation), composition

**Parallelization of the (sparse) multiplication**

| threads | time (ms) | speedup |
|---|---|---|
| 1 | 148661 | x1.0 |
| 2 | 76881 | x1.9 |
| 3 | 54798 | x2.7 |
| 4 | 42855 | x3.4 |
| 5 | 37017 | x4.0 |
| 6 | 30892 | x4.8 |
| 7 | 28365 | x5.2 |
| 8 | 28048 | x5.3 |

- Planned improvements to the memory manager $\Rightarrow$ closer to linear scaling
- Parallel division and GCD implementations are in progress.
- Integration into Factory/Singular remains to be done

# Outline

Deliverables under review for the period

D5.12: Exact linear algebra algorithms and implementations.

D5.11: Refactor and optimize Sage's Combinatorics

Progress report on other deliverables

T5.1: Pari

T5.2: GAP

T5.3: LinBox

T5.4: Singular

Workpackage management

Milestone M8

Strenghtening interactions with numerical HPC

# Addressing recommendations of review 1

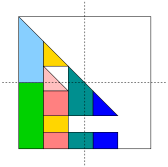
**Recommendation 10:** *Regarding WP5,* **make contacts** *with HPC community in order to ascertain current state-of-the-art. The work in this WP needs to be* **nearer the leading edge***.*

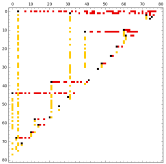
# Addressing recommendations of review 1

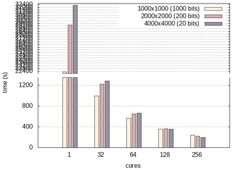
**Recommendation 10:** *Regarding WP5,* **make contacts** *with HPC community in order to ascertain current state-of-the-art. The work in this WP needs to be* **nearer the leading edge**.

## Leading edge achievements in linear algebra

- symmetric factorization outperforms LAPACK implementation
- new non-hierarchical generator for quasiseparable matrices
- large scale parallelization of rational linear solver





# Strengthening interactions with numerical HPC community

## Existing connection

- Dense linear algebra: numerical BLAS used for exact FFLAS for 17 years
- Pointwise interactions with J. Dongarra, L. Grigori, J-Y. L'Excellent, etc
- Publishing in major HPC venues: SIAM-PPSC, EuroPar, PMAA, ParCo
- Involvement in the French CNRS GDR-Calcul working group (Sci Comp)

# Strengthening interactions with numerical HPC community

## Existing connection

- Dense linear algebra: numerical BLAS used for exact FFLAS for 17 years
- Pointwise interactions with J. Dongarra, L. Grigori, J-Y. L'Excellent, etc
- Publishing in major HPC venues: SIAM-PPSC, EuroPar, PMAA, ParCo
- Involvement in the French CNRS GDR-Calcul working group (Sci Comp)

## Recently established

- With the `BLIS` group:
  - Report SIMD vectorization bug and share user experience
  - Implementation of Strassen's algorithm
- On-going collaboration with T. Mary (`Mumps`) and S. Chandrasekaran (UCSB) on quasiseparable matrix algorithmic