



# Algorithms, Programming and Software for High Petascale Computing

**Serge G. Petiton**

**[Serge.petiton@inria.fr](mailto:Serge.petiton@inria.fr)**



# Outline

- Introduction : Post Petaflop computing, on the road to Exascale computing
- Algorithms (linear algebra), Languages and toward a Global Programming Paradigm for High Petascale Computing,
- Our past Japanese-French Research Collaborations
- Future ANR-JST project (tentative, work in progress)



# Outline

- **Introduction : Post Petaflop computing, on the road to Exascale computing**
- Algorithms (linear algebra), Languages and toward a Global Programming Paradigm for High Petascale Computing,
- Our past Japanese-French Research Collaborations
- Future ANR-JST project (tentative, work in progress)



# Several Correlated Challenges

- **Large scale**, several dozens of hundreds of thousand of processors
- Nodes with intra-communication problems
- **Many-core** processors (dozen of thousands, ...millions)
- SIMD or pipelined (or others) **accelerators**
- **Low power** processor (electrical power optimization)
- From Grid-level to Core-level communications
- Integration of parallel computing and distributed computing
- toward a large scale global computing challenge
- Data communication optimization (**scheduling**)
- **Data storage** and “data ranking”, convergence between cloud and HPC?
- Network : from worldwide latency to optical network for intra-core communications
- **Multi-arithmetic** and accurate normalize elementary functions



## *Hierarchical Computing, from worldwide GRID to one thread of one core*

- GRID of supercomputer
- Supercomputer : clusters (of clusters) of nodes (dozen)
- Nodes (dozen or hundred of thousand)
- Processors (dozen, or less)
- Cores (dozen, hundreds)
- Threads (dozens)
- VLIW
- SIMD
- Accelerators
- K-adic operations.....
  
- ***Key issues : fault tolerance, and electrical power optimization, and arithmetic accuracies***



# We will have to program such machine!

End-users will have to develop scientific software

- **High level languages** have to be proposed to end-users
- High efficiency versus **time to “solution”**
- End-users and scientists have to be able to **give expertise** from the methods to the BLAS choice and scheduling strategies
- They have to be able to describe workflow between computing tasks, data migration, visualization and post-treatment process
- They have to use updated component computing and developing frameworks and technologies
- The programming have to be **independent from the hardware** (P2P platform to Top10 supercomputers), and the middleware.



## The Future Exaflop barrier : not only another flop counting frontier coming after the Petaflops

- Sustained Petascale applications on an unique computer exist since a few months
- Next frontier : Exascale computing (and GigaWatts???)
- Nevertheless, **many challenges would emerge as soon as with the announced 10 Petaflop computers and beyond.**
- We have to be able to anticipate solutions to be able to educate scientists as soon as possible to the future programming.
- We have to use the existing emerging platforms and prototype to imagine the future language, systems, algorithms,.....
- We have to structure a new area of HPC (**IESP initiative**)



# Outline

- Introduction : next challenge is 10 Petaflop computing, toward Exascale computing
- **Algorithms (linear algebra), Languages and toward a Global Programming Paradigm for High Petascale Computing, example :**
  - **YML, experimentation on GRID5000**
  - **Hybrid restarted iterative methods (sparse matrices)**
  - **Sparse compression format and GPU performances**
  - **Krylov computation and accuracy on IBM Cell and GPU**
- Our past Japanese-French Research Collaborations
- Future ANR-JST project (tentative, work in progress)



# YML

[yml.prism.uvsq.fr](http://yml.prism.uvsq.fr)

- High level graph description language
- Independant of Middleware, hardware and librairies
- A backend for each middleware (or set of middleware)
- Expertise may be propose by end-users
- May use existing components
  
- Deployed in INRIA, LIFL, PRISM, Tsukuba (Japan), Hohai (Chine), MONS (Belgique), UCD (Irlande)
- Collaboration initialized last summer with Stanford Univ (LISTZ and PPL project)



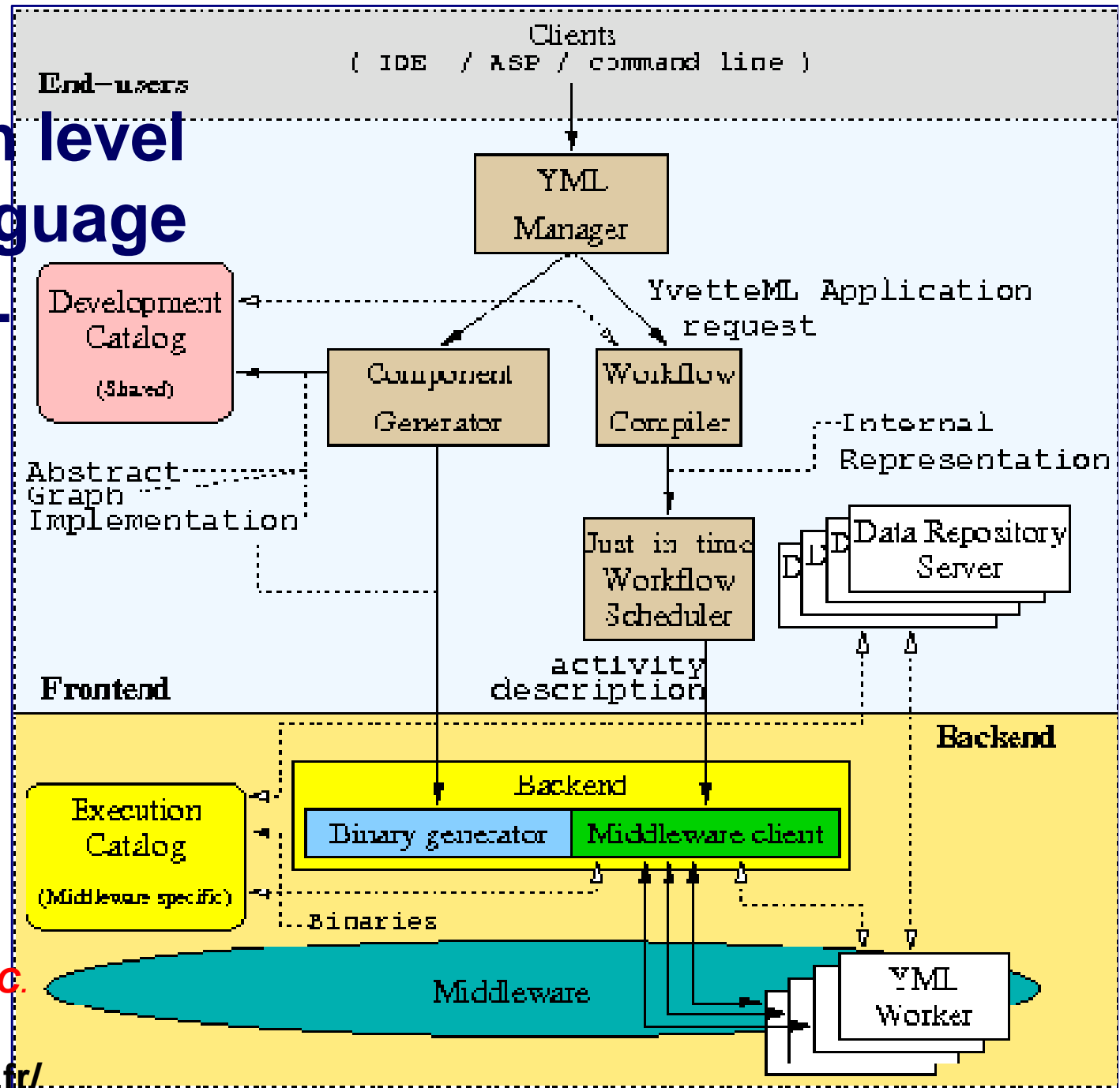
# High level Language YML

Front end :  
Depends only of the applications

Back end : depends of middleware.

Ex. XtremWeb (XW, F),  
OmniRPC (Jp), and  
One hybrid XW-omniRPC.

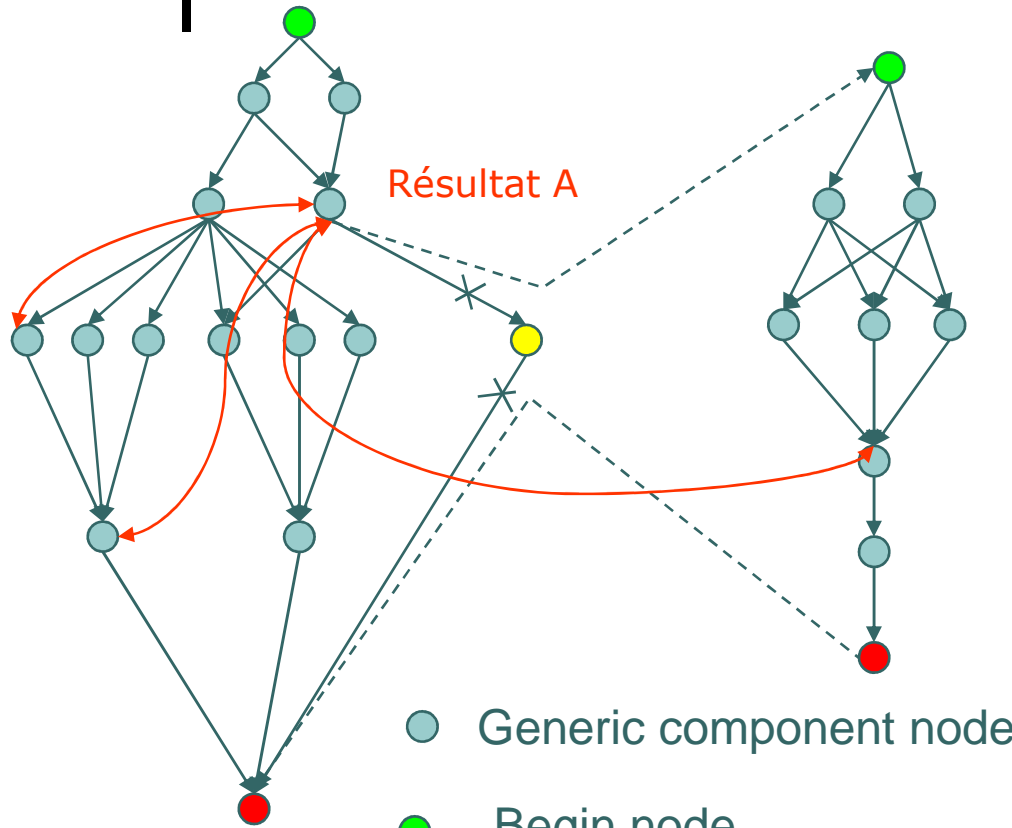
<http://yml.prism.uvsq.fr/>



SP10  
SP11  
SP12



# Components/Tasks Graph Dependency



- Generic component node
- Begin node
- End node
- Graph node
- Dependence

```

par
  compute tache1(..);
  signal(e1);
//
  forall i=1,n
    compute tache2(i,..); migrate
matrix(..);
  signal(e2(i));
  end forall
//
wait(e1 and e2(1);
Par
  compute tache3(..);
  signal(e3);
//
  compute tache4(..);
  signal(e4);
//
  compute tache5(..); control robot(..);
  signal(e5); visualize mesh(...);
end par
//
wait(e3 and e4 and e5);
compute tache6(..);
compute tache7(..);
end par

```

## Diapositive 11

---

**SP10** Serge Petiton; 18/06/2004

**SP11** Serge Petiton; 18/06/2004

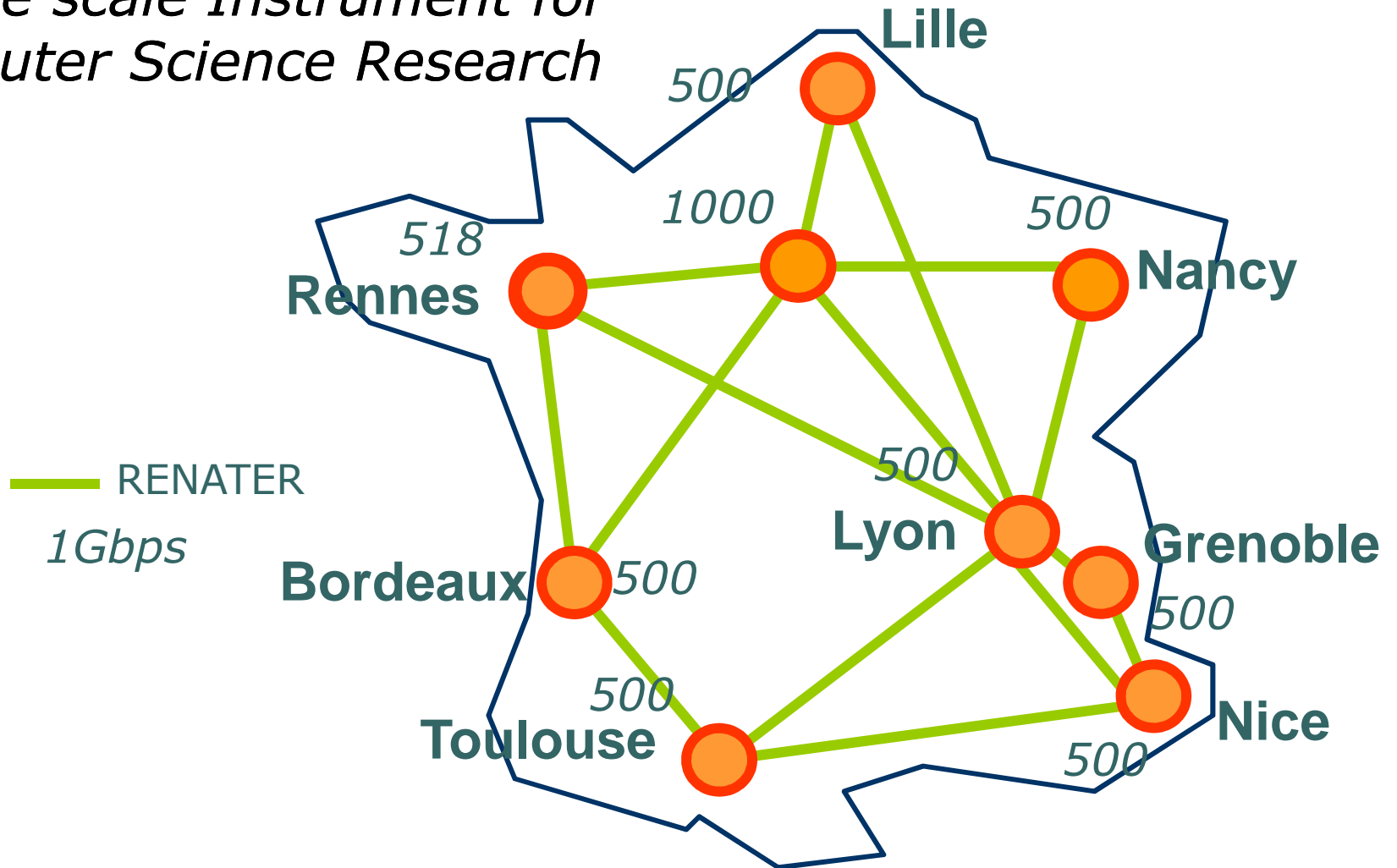
**SP12** Serge Petiton; 18/06/2004



# YML experimentations on

Grid 5000: use as a cluster of cluster

*A large scale Instrument for  
Computer Science Research*





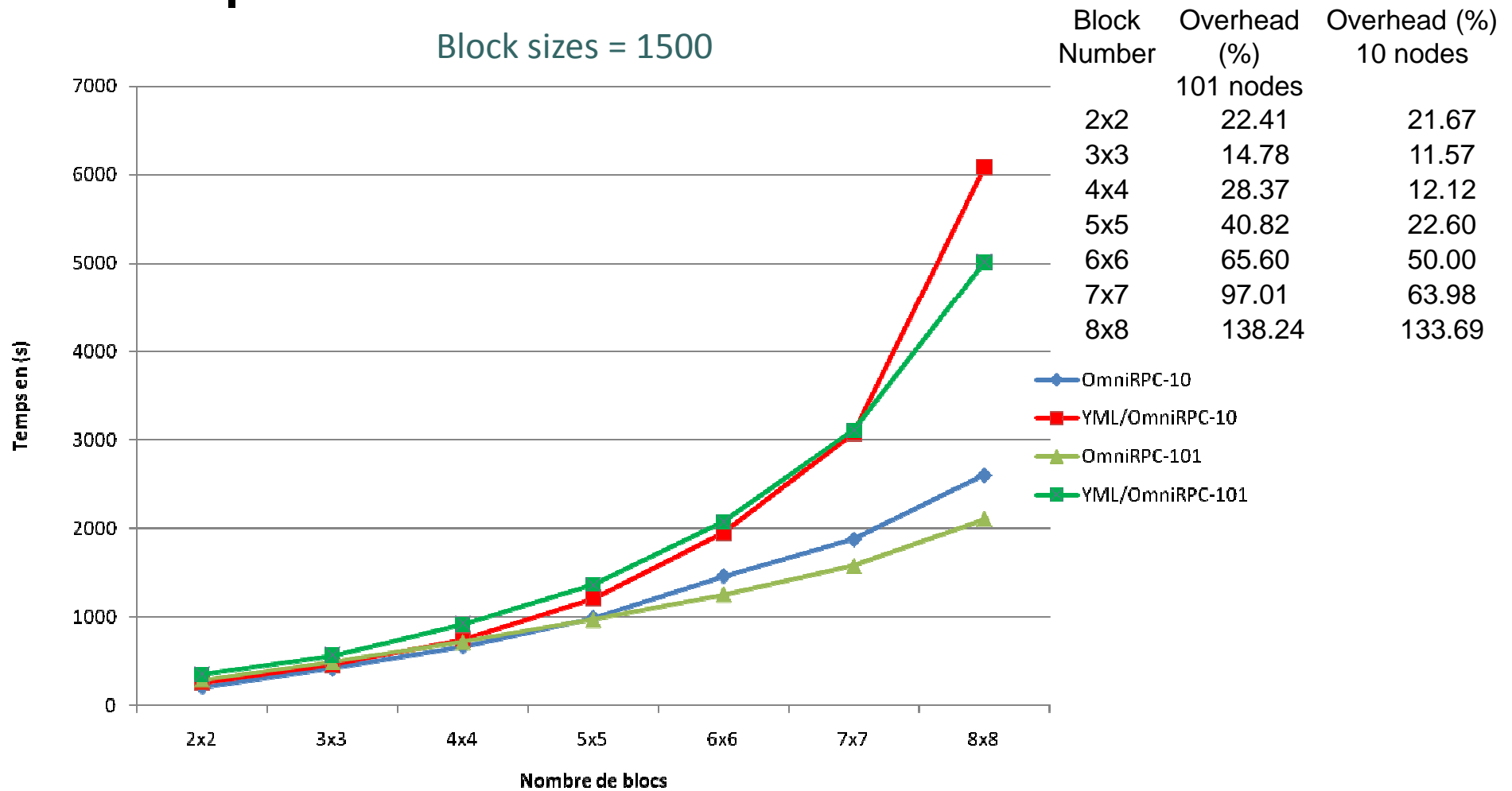
# YML experimentations with Maximes Hugues, TOTAL and LIFL

Concerned GRID500 ressource

Site	Number of nodes	CPU	Memory
Nancy	120	2 x Dual Core INTEL Xeon, 1.6GHz	2GB
	47	2 x AMD64 Opteron, 2.0GHz	2GB
Bordeaux	51	2 x Dual Core Intel Xeon, 3.0GHz	2GB
Sophia	56	2 x Dual Core AMD64 Opteron, 2.2GHz	4GB
Rennes	99	2 x AMD64 Opteron, 2.0GHz	2GB
	33	2 x Dual Core Intel Xeon, 2.33GHz	8GB

# GRID 5000, BGJ, 10, 101 nœuds, OmniRPC versus YML/OmniRPC

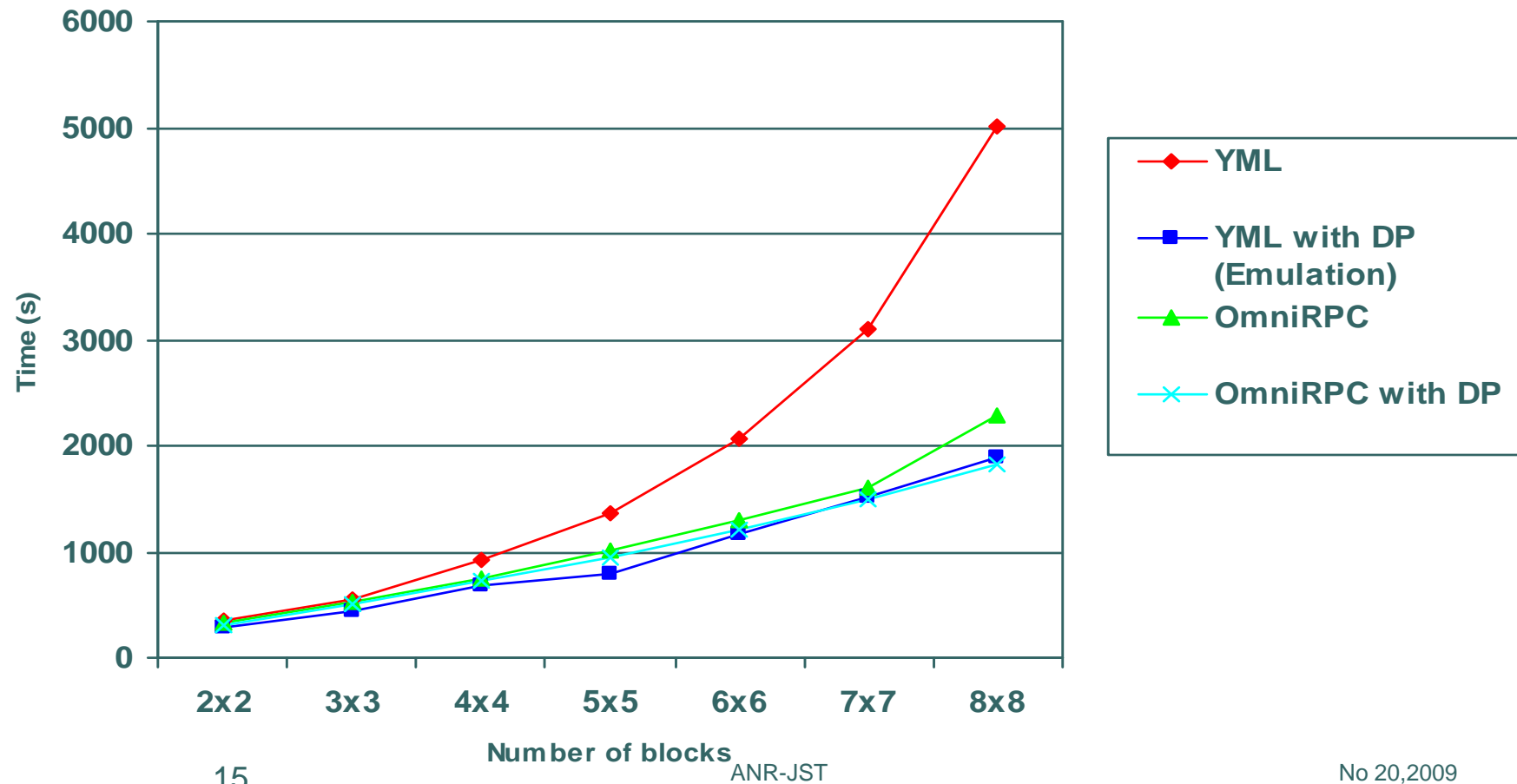
Block sizes = 1500





# Block Gauss-Jordan, 200 processors Cluster, Grid'5000; YML/OmniRPC vs OmniRPC (with TOTAL and LIFL)

Time of execution with and without data persistence on Nancy,  
block size = 1500



15

ANR-JST

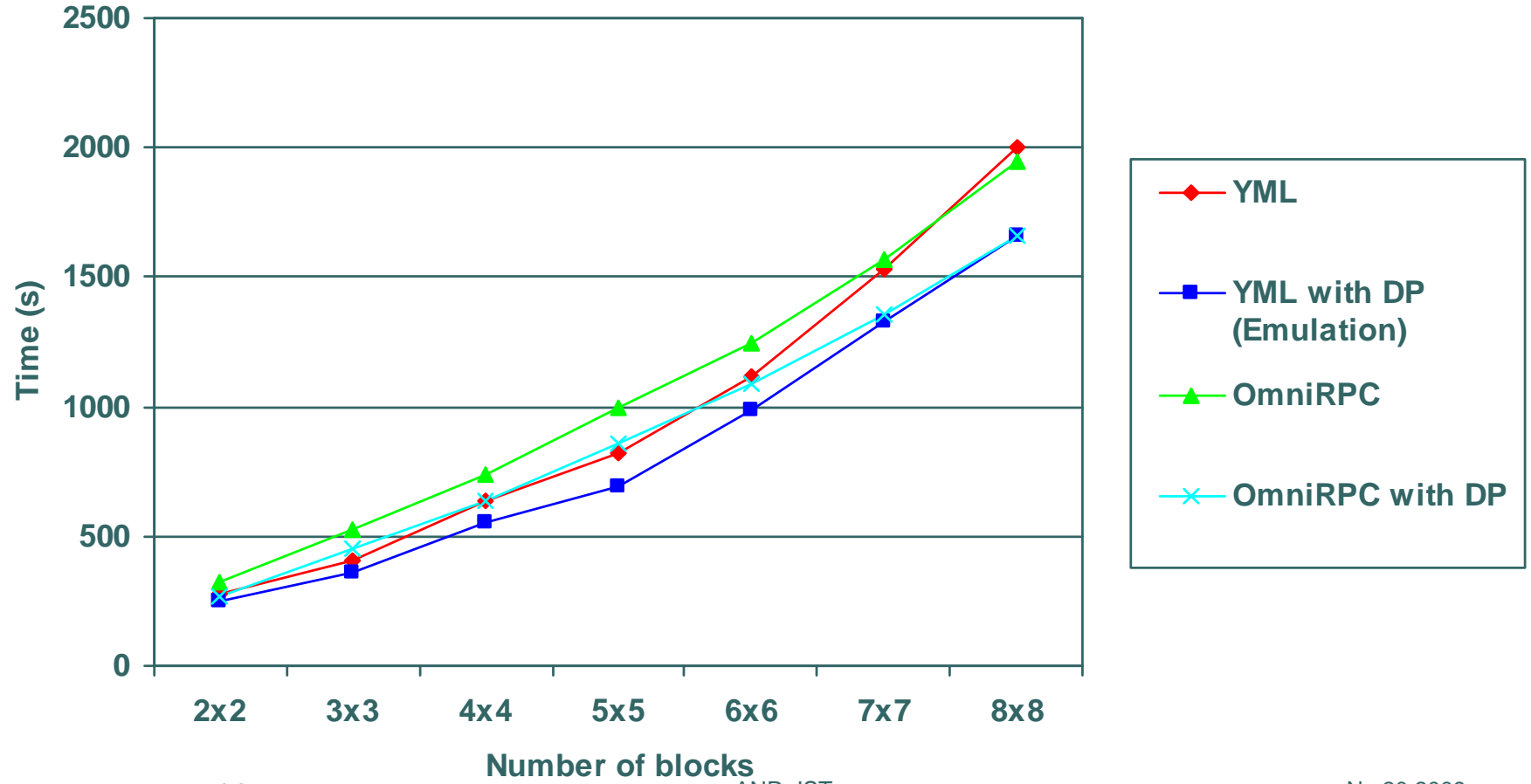
No 20,2009

\*DP: Data Persistence



# BGJ, 200 processors Cluster, Grid'5000; YML/OmniRPC vs OmniRPC (TOTAL and LIFL)

Time of execution with and without data persistence on Rennes,  
block size = 1500

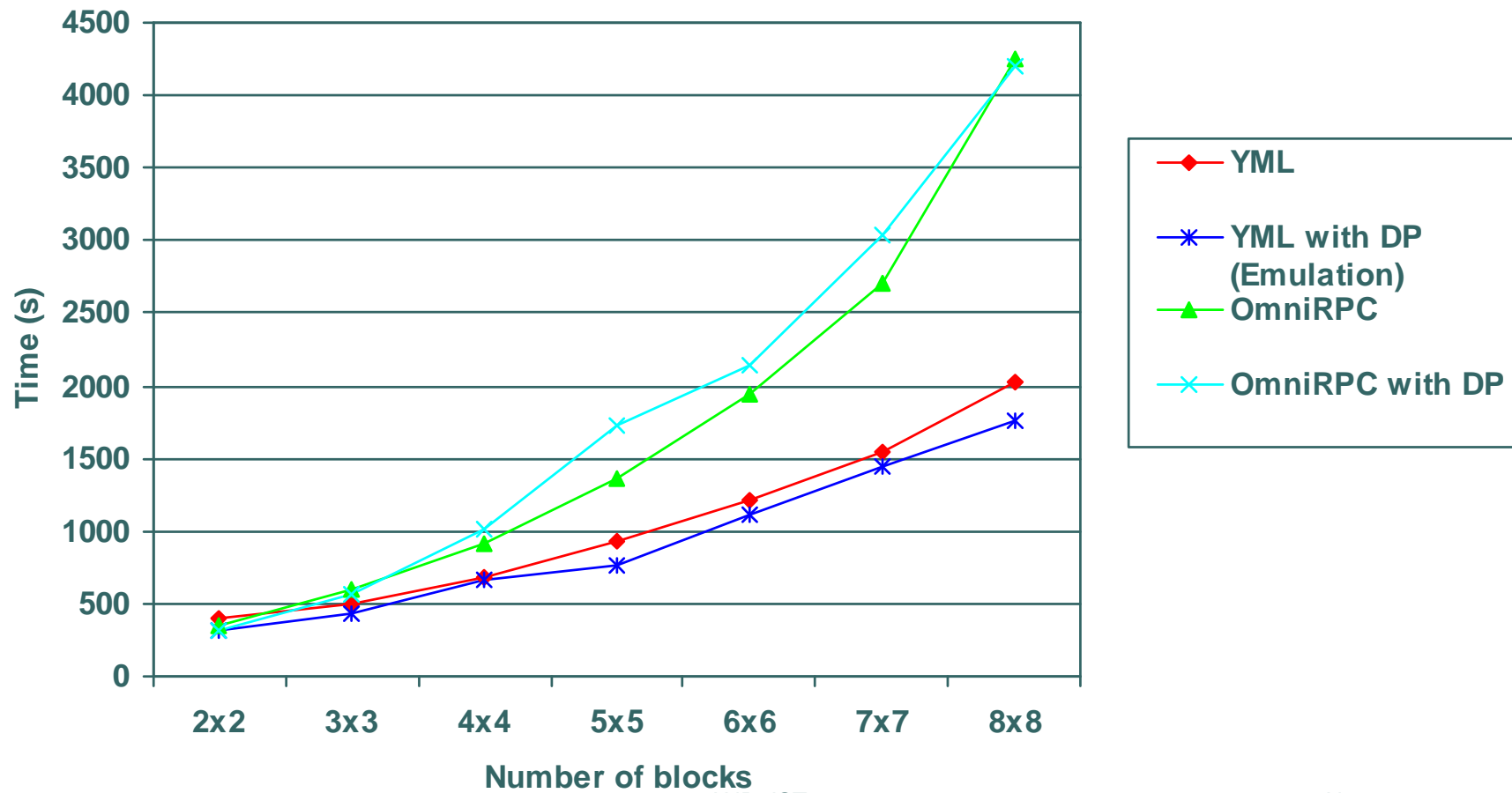


\*DP: Data Persistence



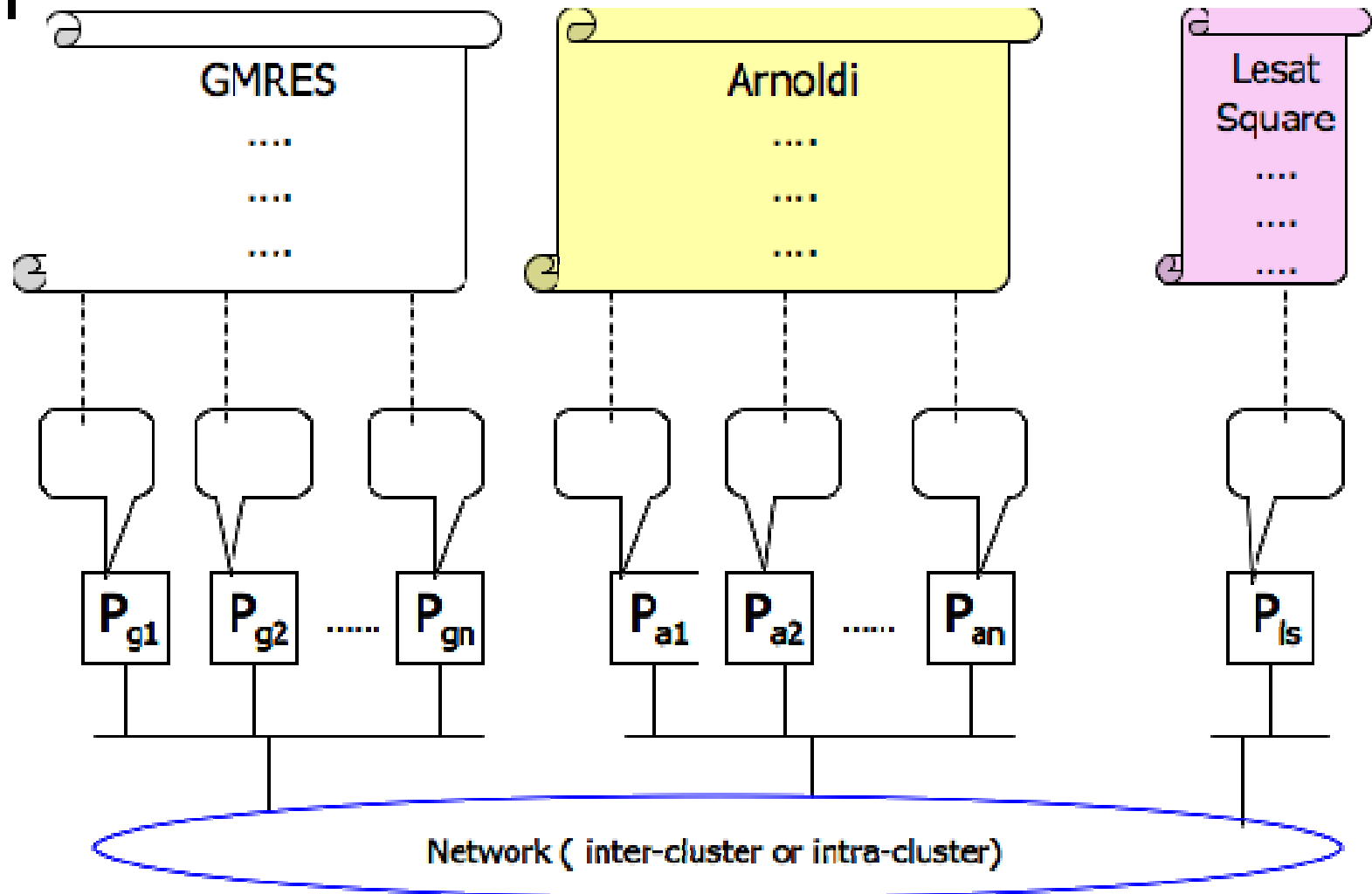
# BGJ, 200 processors distributed over 4 Clusters, Grid'5000; YML/OmniRPC vs OmniRPC (with Maxime Hugues (TOTAL and LIFL))

## Time of execution on a Cluster of Clusters



# Asynchronous Iterative Restarted Methods

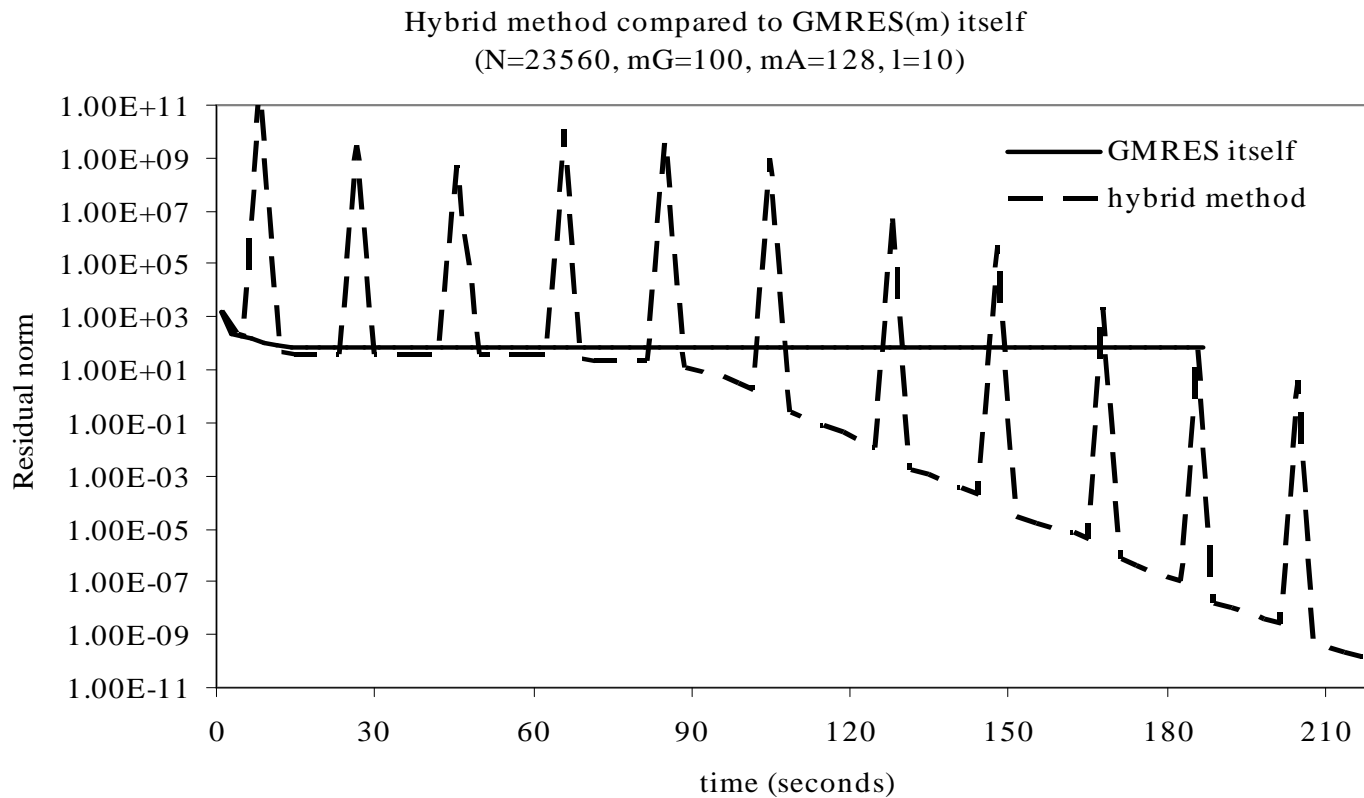
## INRIA, LIFL



# HYBRID METHODS

## Numeric Results and Analysis

advantage over GMRES itself (II) (difficult convergence case)

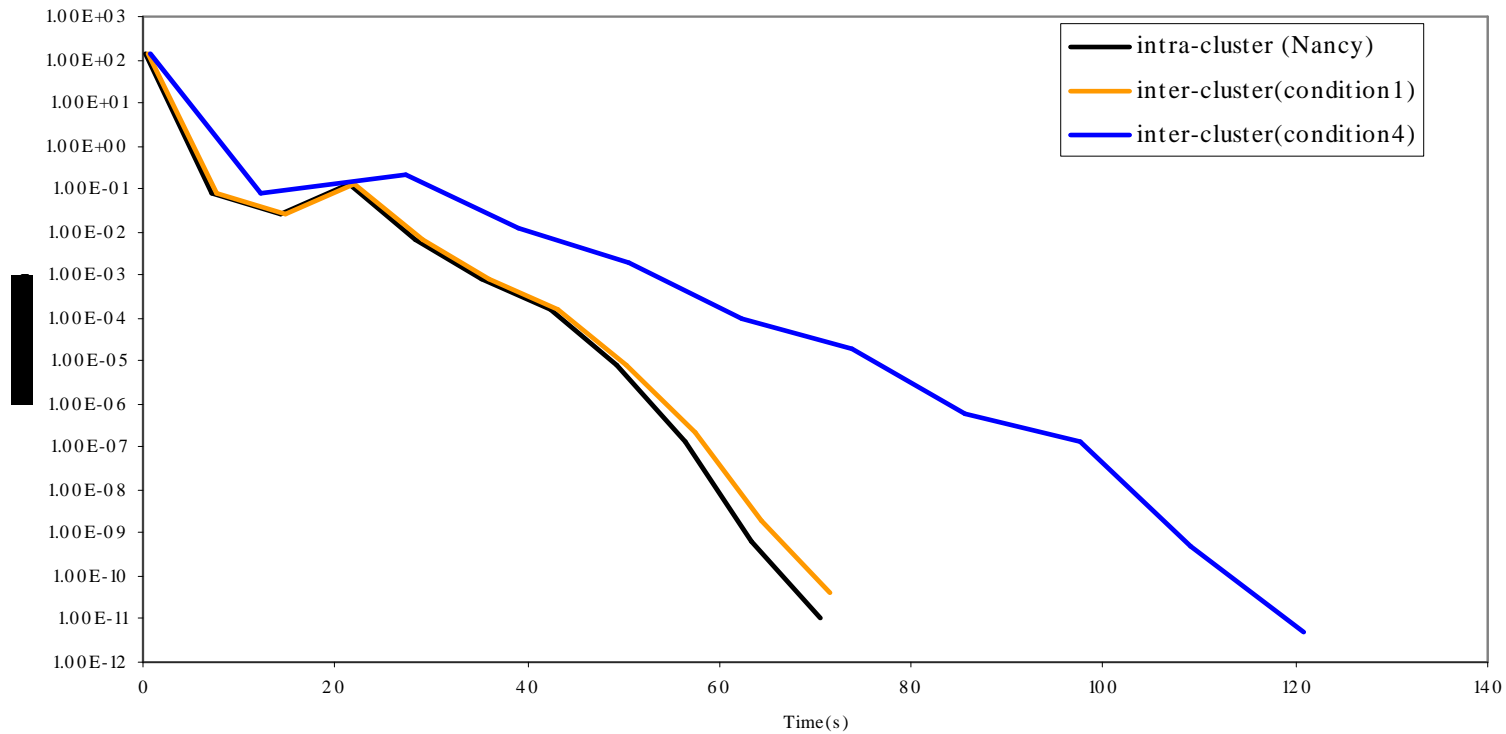




# Numeric Results and Analysis

comparison between different network configurations on Grid5000

Implementation intra-cluster and inter-cluste (17000)

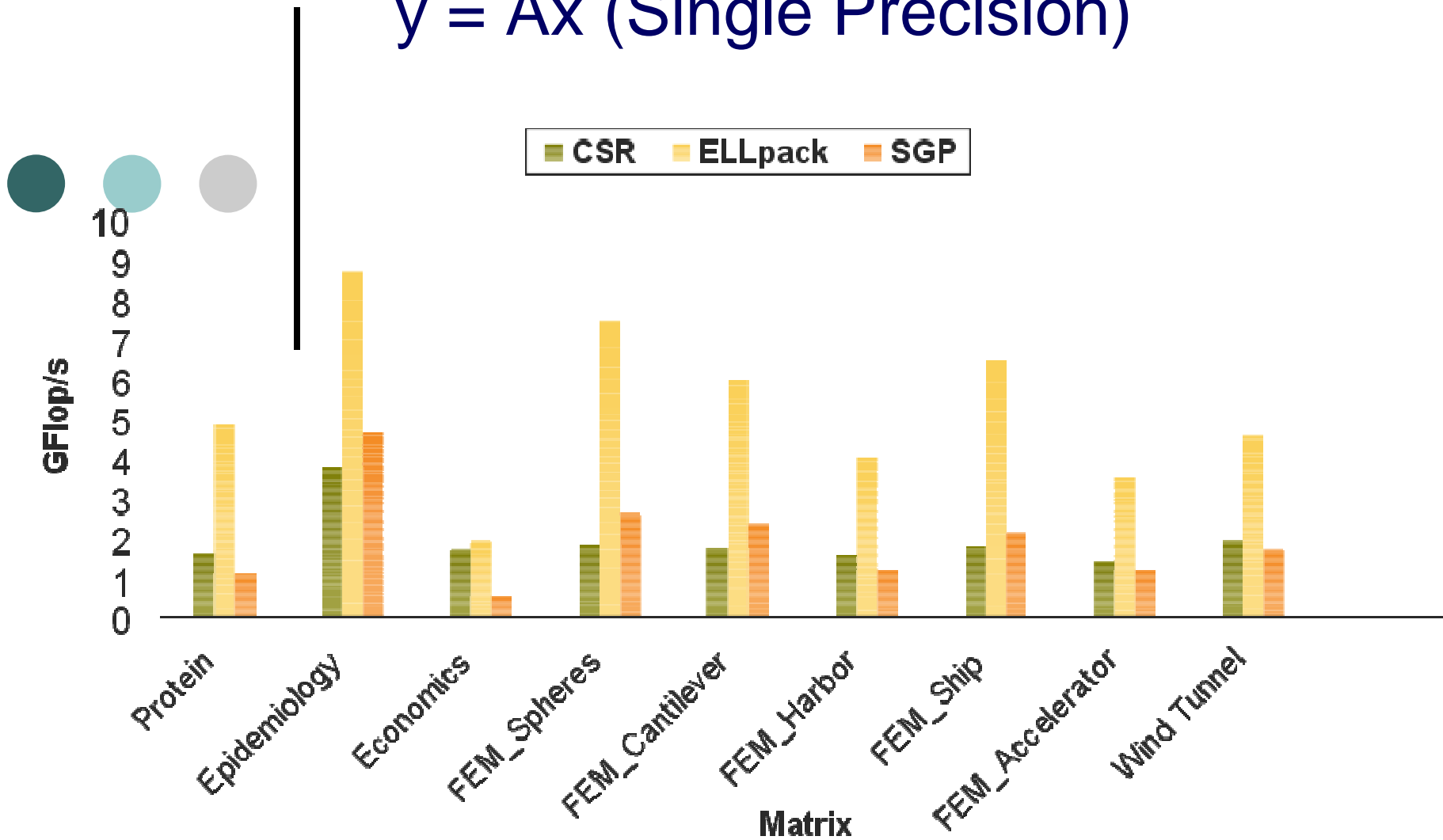


# Sparse Matrix Computation with TOTAL and LIFL



- One GPGPU of a Tesla S0170
- Evaluation of CSR, ELLpack and SGP format
- Each thread computes
  - One row for CSR and ELLpack format
  - One column for SGP format
- Different type of matrix from Tim Davis

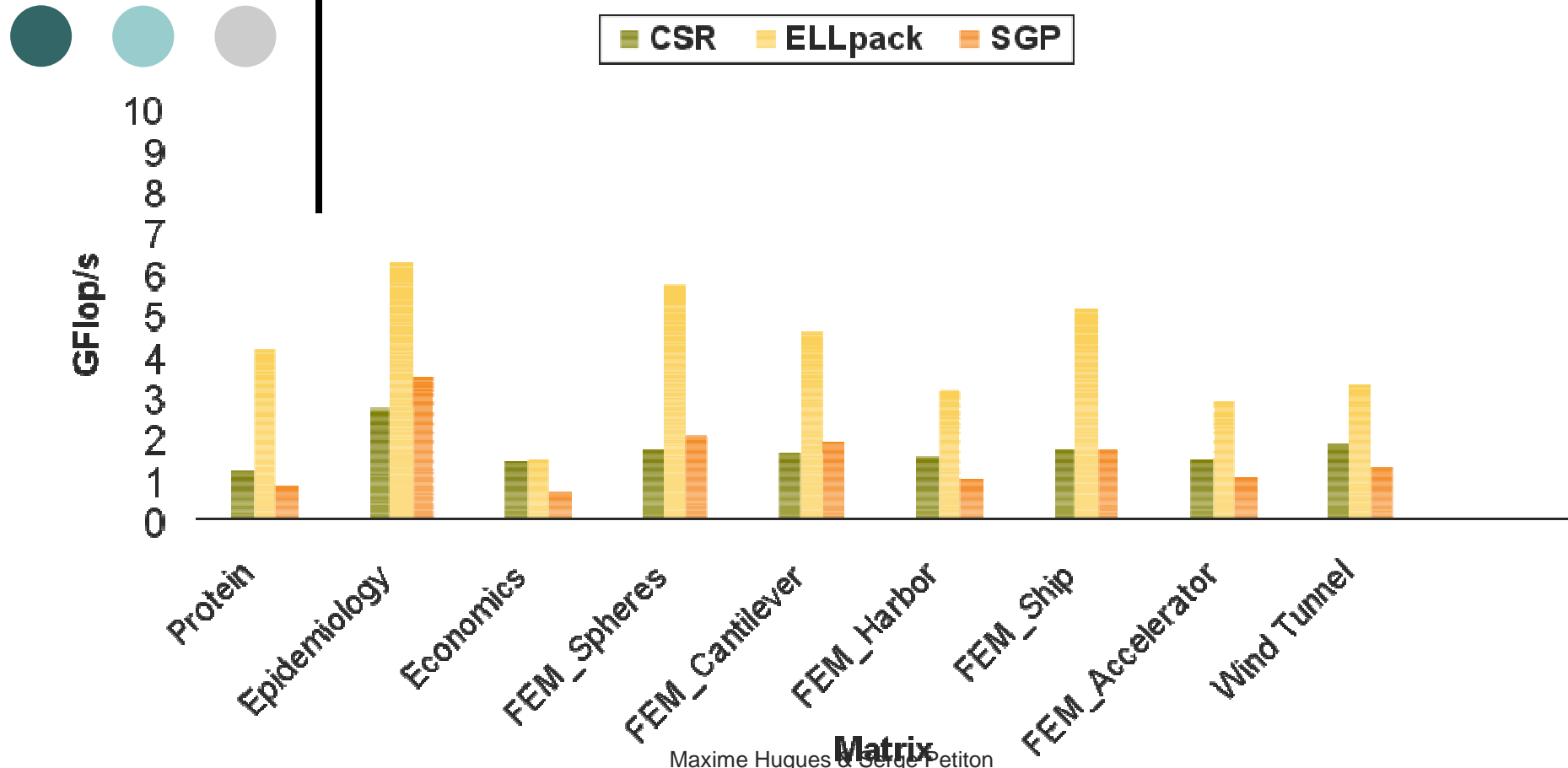
# Sparse Matrix Vector $y = Ax$ (Single Precision)



\*Matrix From Tim Davis

Maxime Hugues & Serge Petiton

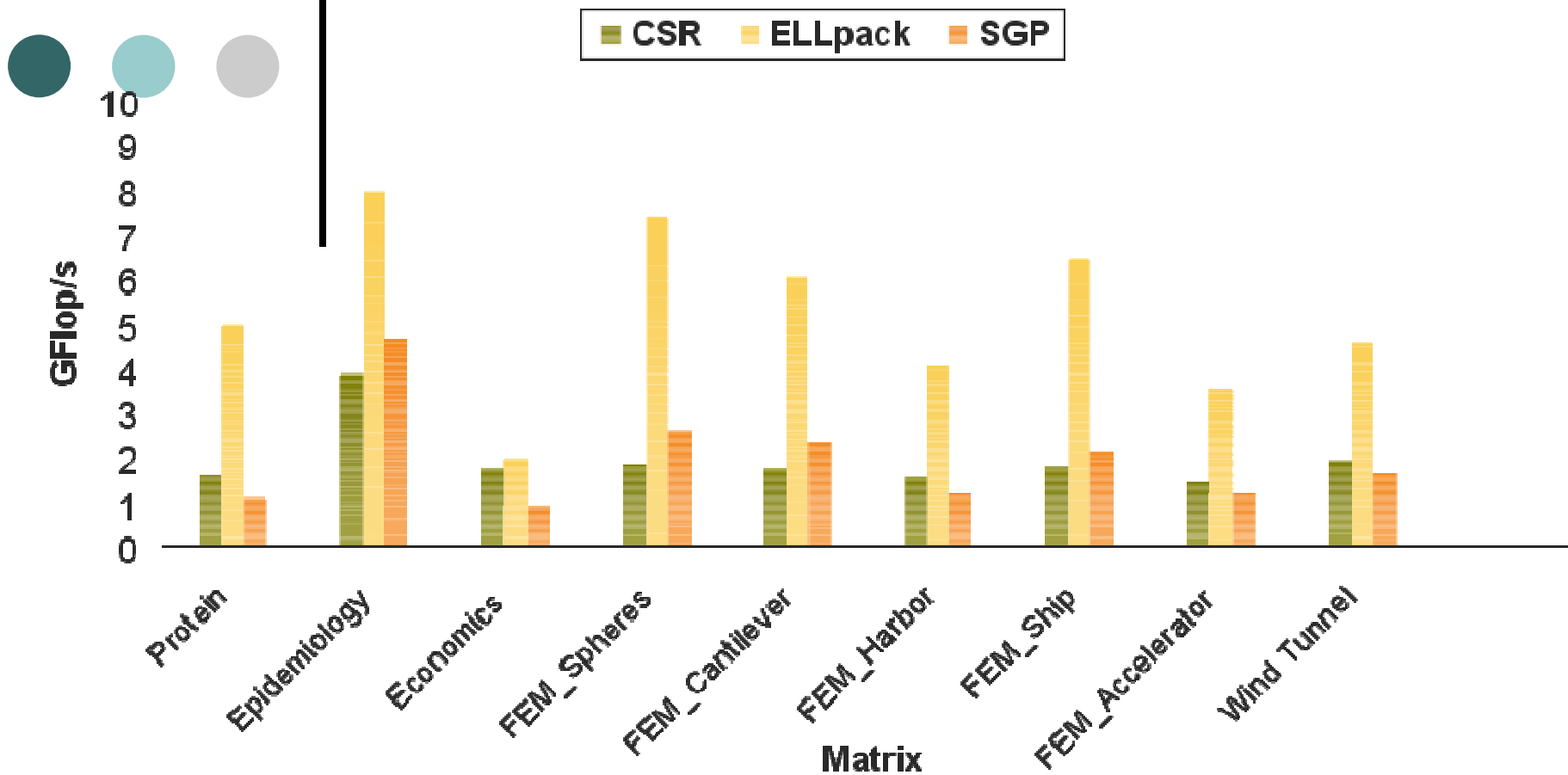
# Sparse Matrix Vector $y = Ax$ (Double Precision)



\*Matrix From Tim Davis

Maxime Hugues & Serge Petiton

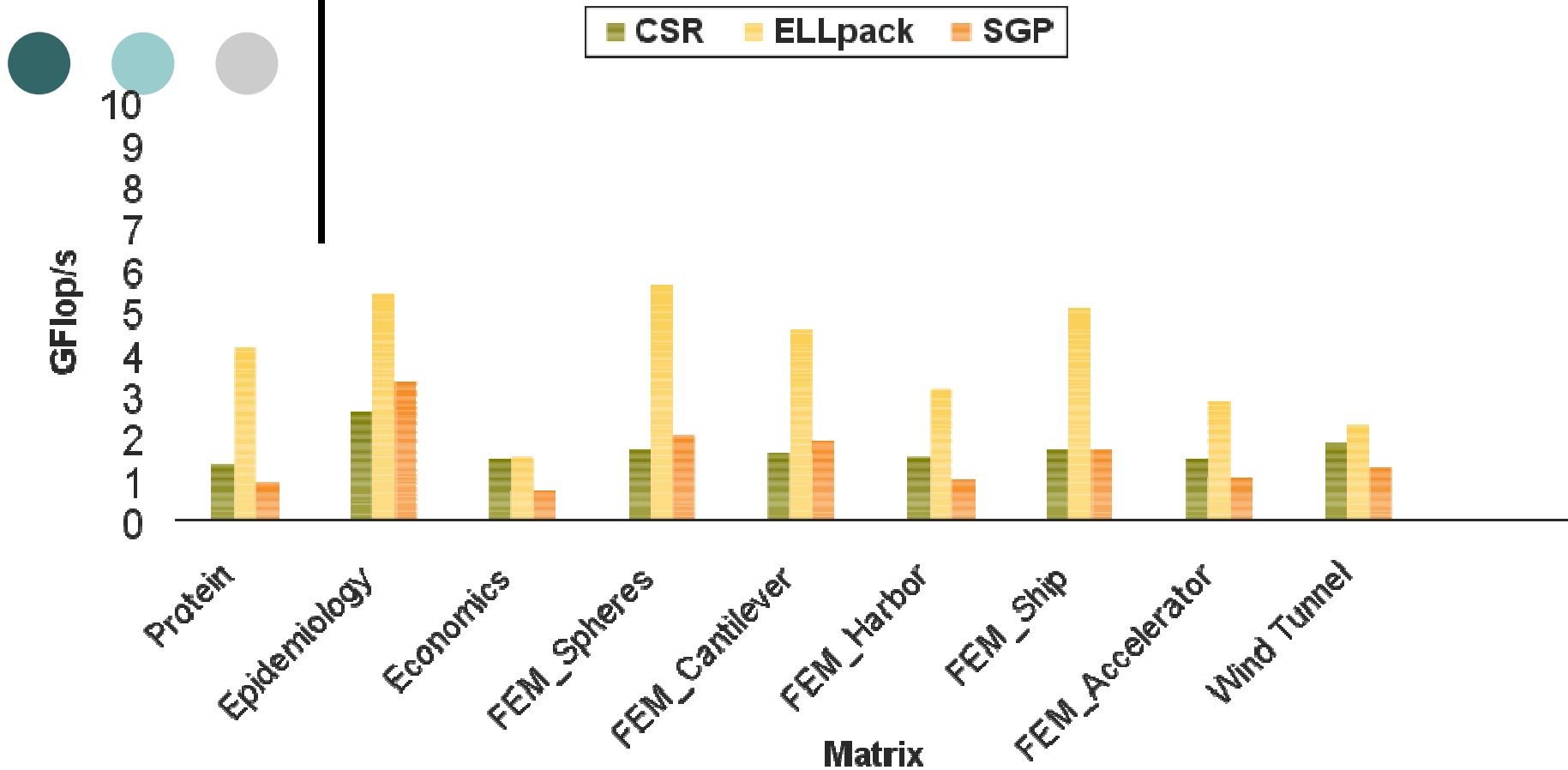
# Sparse Matrix Vector $y = A(Ax+x)+x$ (Single Precision)



Maxime Hugues & Serge Petiton

\*Matrix From Tim Davis

# Sparse Matrix Vector $y = A(Ax+x)+x$ (Double Precision)



Maxime Hugues & Serge Petiton

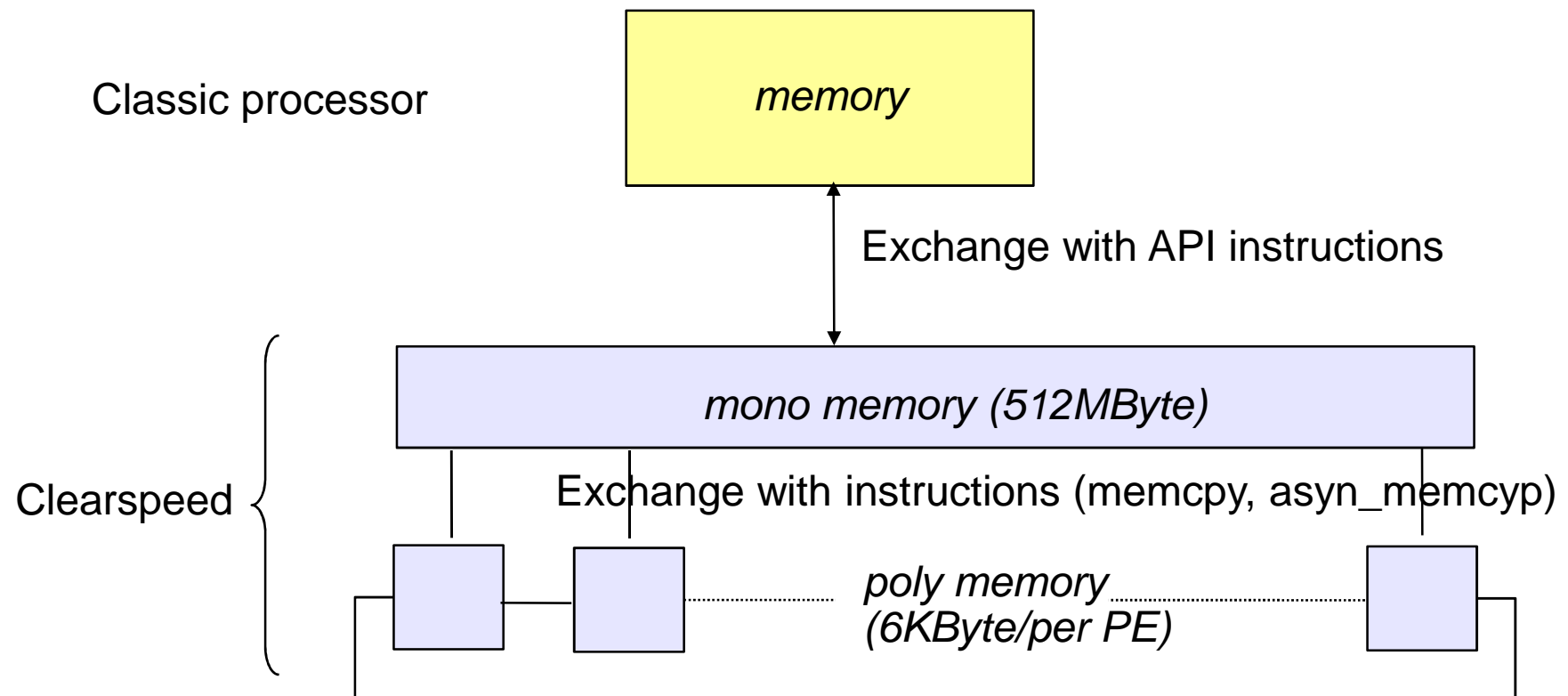
\*Matrix From Tim Davis



# GMRES and Hybrid method with accelerators

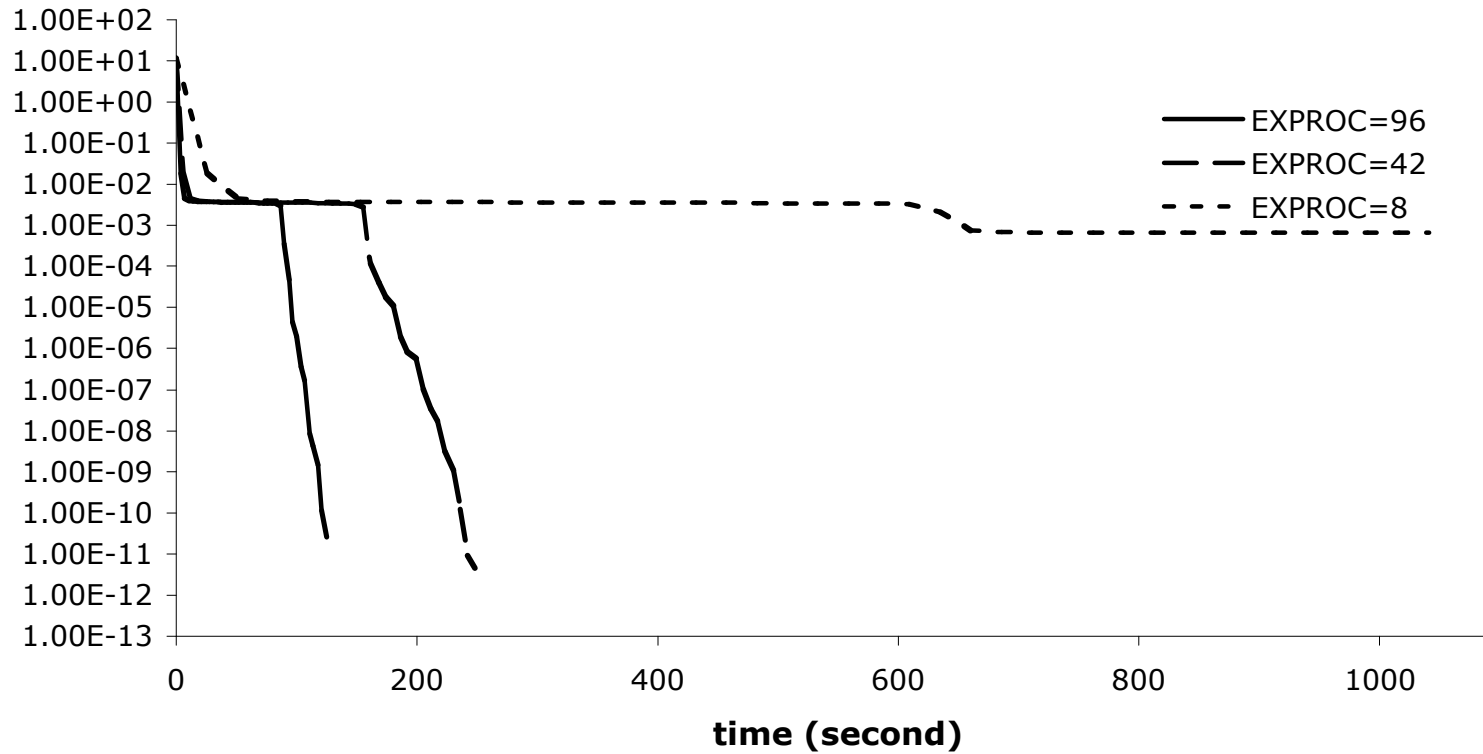
With Guy Bergère (LIFL) and Ye Zhang (LIFL); at TiTech, Japan

## *Levels of Memory on Clearspeed*



# ---- Parallelism degree limitation

- Best number of processors : total array (96 PEs)  
Example on Tsubame. Utm300 (N=300, mG=120)

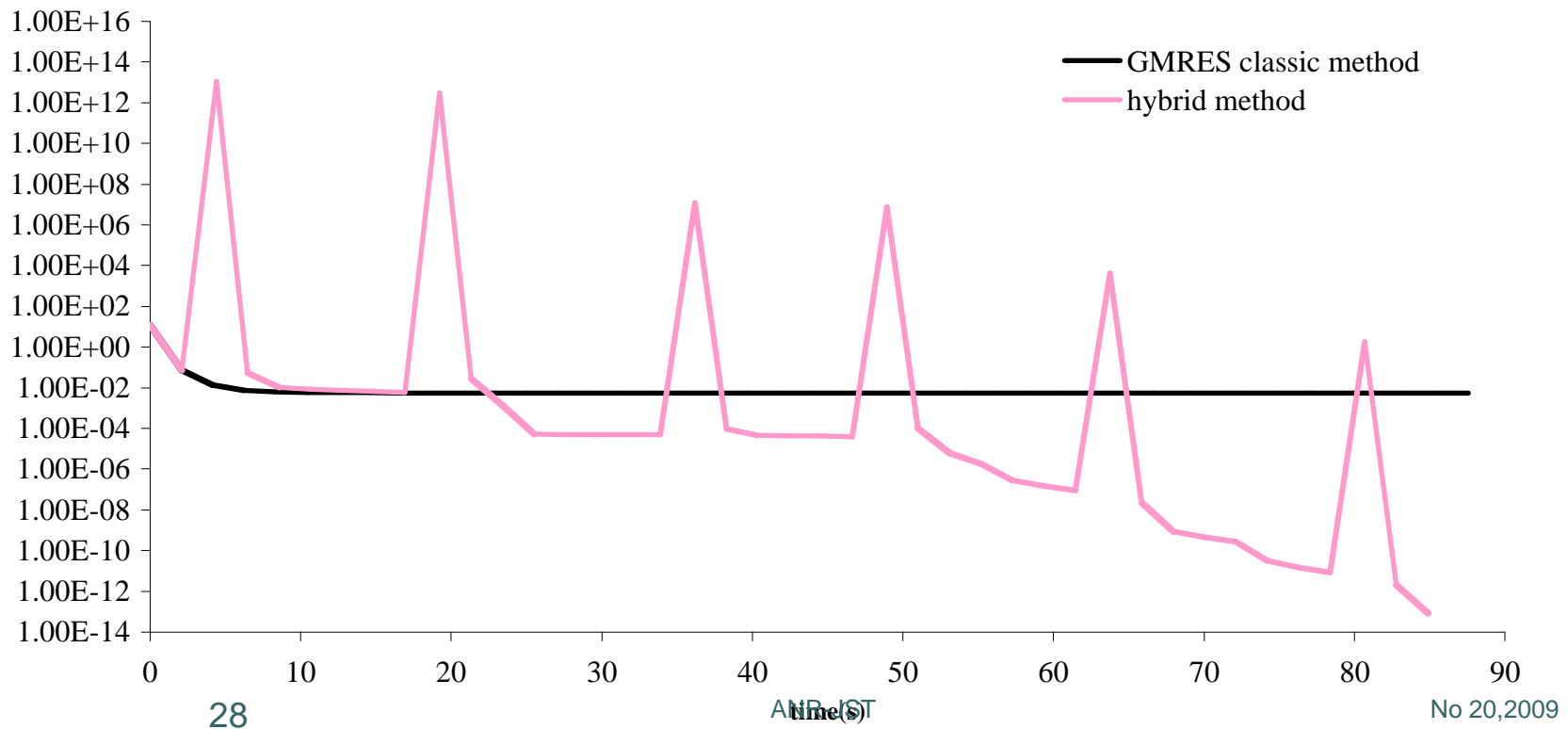




# Hybrid method on Tsubame-Clearspeed

- Example. Utm300 (N=300, mG=100, mA=128)

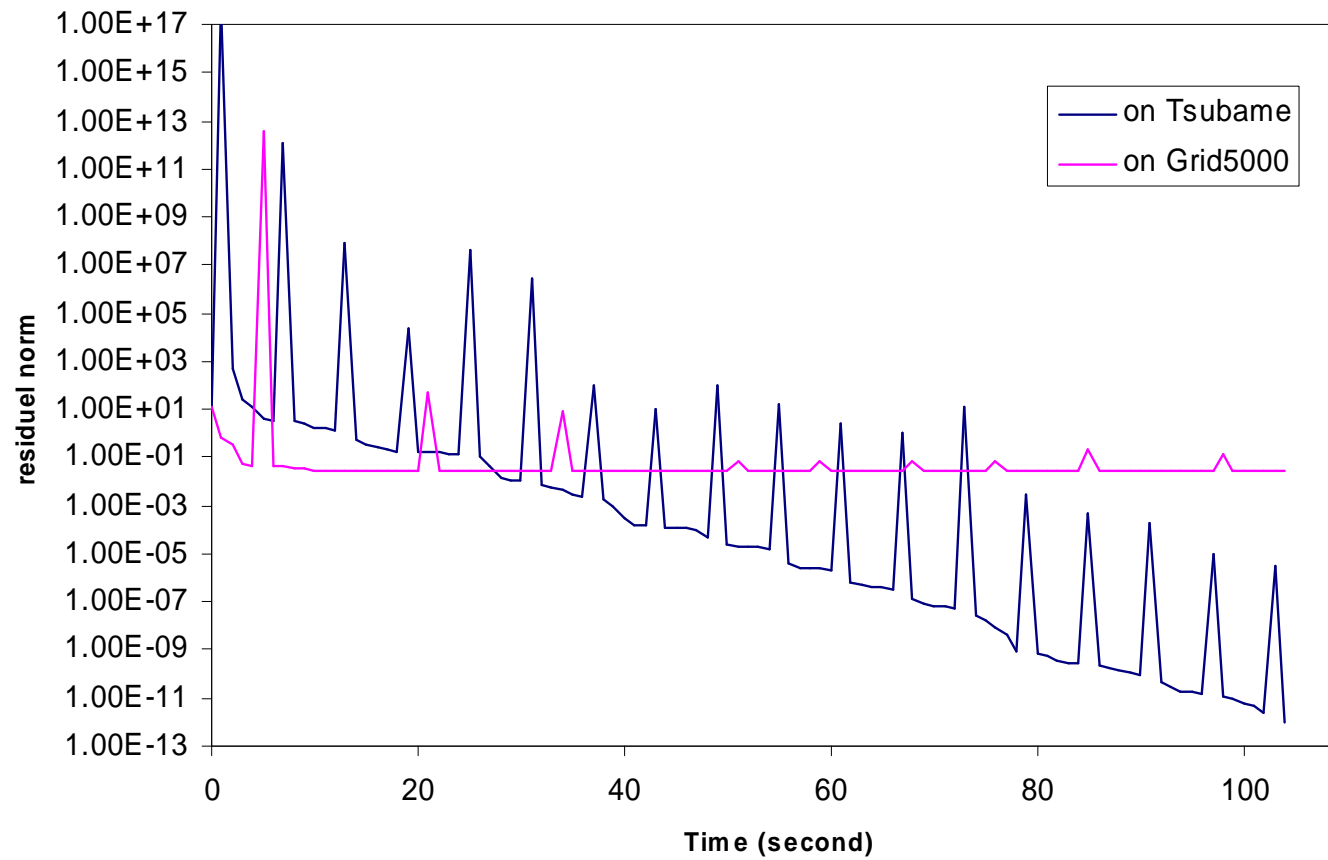
hybrid method compared with GMRES classic method  
utm300(N=300, mG=100, L=5, K=4)





# Hybrid method Compare with Grid5000

- Example. Utm300 (N=300, mG=50, mA=64, K=5, L=4)



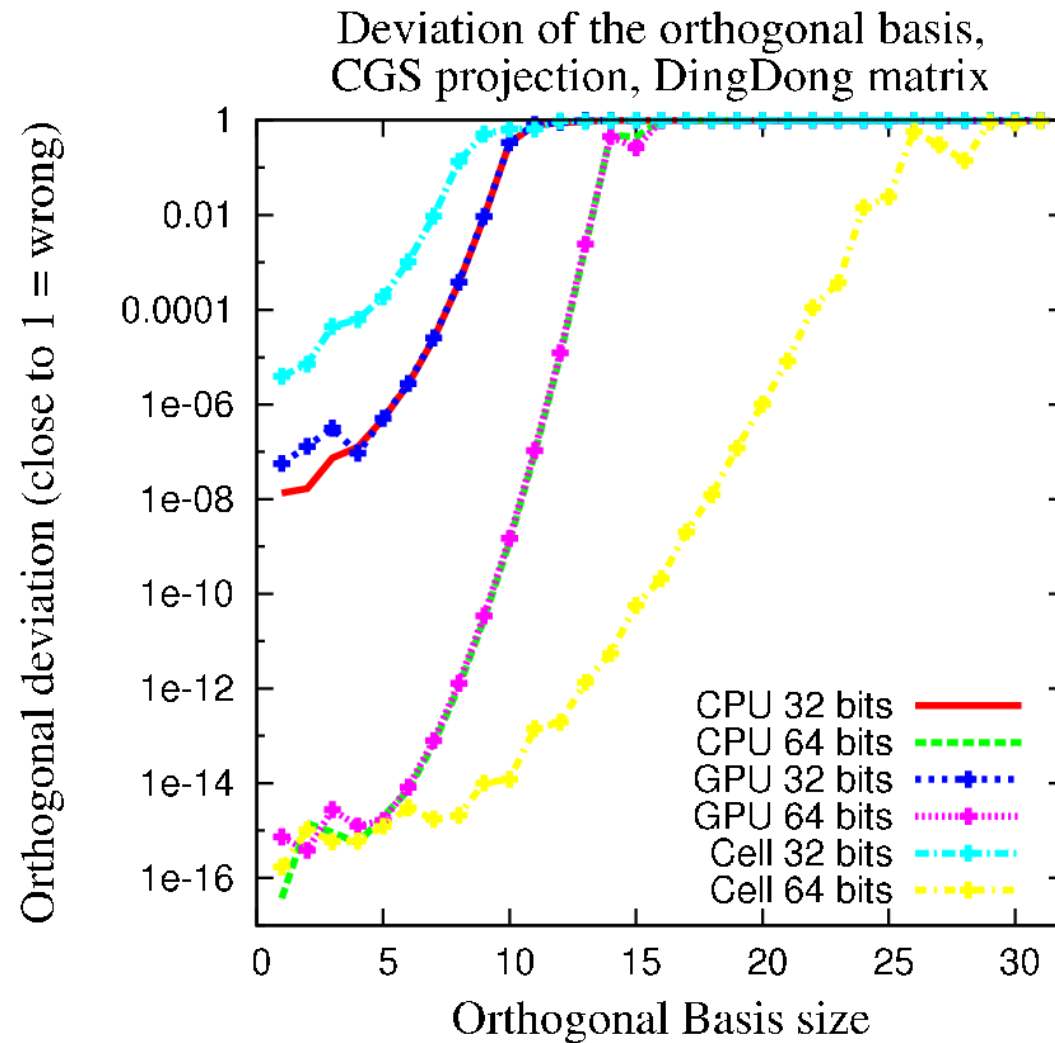


# Hybrid Multicore computing

with LIFL and CEA-DEN Saclay

- Work done with CNRS/LIFL and CEA
- Objectives : accelerate neutronics and radioprotection code, arithmetic accuracy analysis.
- Several steps
  - Study performance and accuracy of emerging computing architectures on some similar dense bandwidth bound numerical kernels
  - Try to accelerate the Minos core solver on some test case : IAEA benchmark
  - Generalize acceleration for other applications

# Orthogonalization : accuracy





# Outline

- Introduction : next challenge is 10 Petaflop computing, toward Exascale computing
- Algorithms (linear algebra), Languages and toward a Global Programming Paradigm for High Petascale Computing,
- **Our past Japanese-French Research Collaborations**
- Future ANR-JST project (tentative, work in progress)

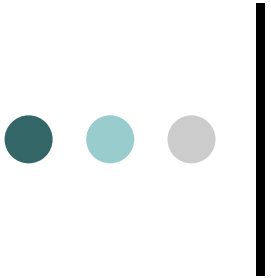


# Computer Science Japanese-French Research Collaborations on HPC and GRID

- INRIA associated teams
  - ex : Grand Large INRIA project (Franck Cappello) and University of Tsukuba (Mitsuhisa Sato), since 6 years
- Sakura projects (several on HPC and GRID)
- Japanese-French doctoral interships (Egide, MEA,...)
- CNRS-JST
  - NEGST (2006-2008) - (Paris06, Tokyo07, Toulouse08)
  - Redimps (2007-2009)
- French Delegation on “Superecomputer”, initiative of the French Embassy in Tokyo, January 2006
- PAAP workshop series (RIKEN07, Toulouse08, Kyoto09)

January 2009 - CNRS LIA Laboratory opening

**Japanese-French Laboratory in Informatics (JFLI) – HPC group**



## Our collaborations with the University of Tsukuba and TiTech

- **P2P programming on a Japanese (4 clusters in Tsukuba) French (200 PC) platform – Tsukuba**
- Low power linear algebra programming on cluster - Tsukuba
- YML\_OmniRPC framework and programming on Tsukuba Clusters and GRID5000
- Accelerator-multi-core programming - Titech
- **Sparse Matrix Hybrid Iterative methods on Tsubame – Titech**
- Eigenvalue programming on T2K – Tsukuba
- **YML-OmniRPC linear programming with data migration optimization based on smart scheduling, using GRID5000 – Tsukuba**

# NEGST, First workshop, June 23-24th, 2006, in Paris

[www2.lifl.fr/MAP/negst/](http://www2.lifl.fr/MAP/negst/)

## Japan

Kenichi Miura, NII, Naregi  
Mitsuhisa Sato, Univ. of Tsukuba  
Satoshi Matsuoka, Tokyo Inst. of Tech  
Tomohiro Kudoh, AIST  
Yutaka Ishikawa, AIST/Univ. of Tokyo  
Takashi Sasaki, KEK  
Setsuya Kawabata, KEK  
Osamu Tatebe, Univ. of Tsukuba

## France

Abdelkader Amar, ENS Lyon  
Gabriel Antoniu, IRISA (Friday)  
Dominique Boutigny, CNRS/IN2P3  
Franck Cappello, INRIA Futurs  
Yves Caniou, ENS Lyon  
Laurent Choy, CNRS/LIFL (Saturday)  
Paulo Concalvès, ENS Lyon (Friday)  
Michel Dayde, CNRS/IRIT  
Alexandre Denis, INRIA/LaBRI  
Nahid Emad, CNRS/PriSM (Friday)  
Francoise Genova, CNRS

Guillaume Huard, CNRS/IMAG  
Francois Le Diberder, CNRS/IN2P3  
Christian Perez, IRISA (Friday)  
Denis Perret-Gallix, IN2P3/CNRS  
Serge Petiton, CNRS/LIFL  
Pascale Primet, ENS Lyon  
Martin Quinson, CNRS/LORIA  
Dany Vandromme, Renater



## FRANCE

Laurent Baduel  
Christophe Calvi  
Franck Cappello  
Laurent Choy  
Philippe Codognet  
Pierre Dauchez  
Michel Dayde  
Nahid Emad  
Marie-Alice Foujols  
Patrice Klein  
Stephane Lanteri  
Pierre-Jean Martin  
Serge Petiton  
Olivier Richard  
Marie Rohmer

# PAAP, First workshop at RIKEN, 11/1-2, 2007

[www2.lifl.fr/MAP/paap/](http://www2.lifl.fr/MAP/paap/)

## JAPAN

Tadashi Watanabe	Daisuke Takahashi	Takahiro Katagiri	Mitsuhsa Sato	Yutaka Ishikawa
Hiroshi Nakamura	Hiroshi Nakashima	Satoshi Matusoka	Kenichi Miura	Makoto Taiji
Fumio Hirata	Hirufumi Tomita	Taisuke Boku	Satoshi Sekiguchi	Ryutaro Himeno
Toshikazu Takada	Mitsuo Yokokawa	Hiroko Furuno	Katsumi Tanaka	





# Outline

- Introduction : next challenge is 10 Petaflop computing, toward Exascale computing
- Algorithms (linear algebra), Languages and toward a Global Programming Paradigm for High Petascale Computing,
- Our past Japanese-French Research Collaborations
- **Future ANR-JST project (and other participating French projects)**



## Energy-awareness in Petaflops machines (INRIA, project GREEN-NET)

- Today's work: **Towards autonomic energy-aware distributed systems**
  - Autonomic computing: “Autonomic Energy Management of Clustered Applications”
  - Task Allocation: “Energy-Aware Resource Allocation”
  - Platform: “The GREEN-NET Framework: Energy Efficiency in Large Scale Distributed Systems”
  - Networking: “*Energy Consumption of Residential and Professional Switches*”
- Project: **Study the potential for energy consumption reduction in Petascale machines**
  - Tuning hosts, networks, middleware and applications,
  - Allocating / scheduling tasks energy-wise
  - Measuring and modelling energy consumption of applications



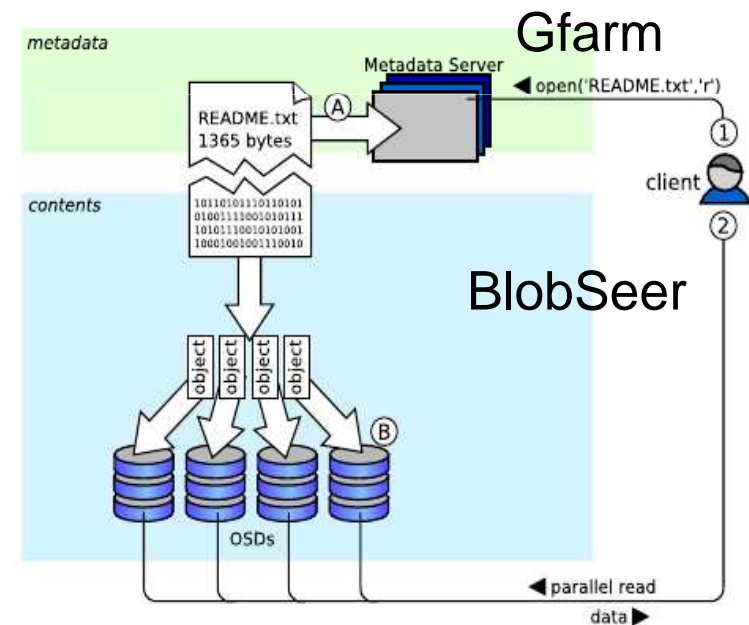
## New Challenges for Petascale Data Storage

- Scalable storage management for new-generation, data-oriented
- high-performance Petascale applications
  - Data-centric e-Science applications
  - Need to store, share and huge amounts of information
  - Massive data objects: BLOBs (Terabytes)
  - High concurrency for accessing massive shared data ( $10^3$  concurrent clients) at fine grains
- Goal: Integrated a scalable BLOB-based file system, able to cope with huge data under heavy access concurrency on Petascale architectures.

# New Challenges for Petascale Data (INRIA Rennes)

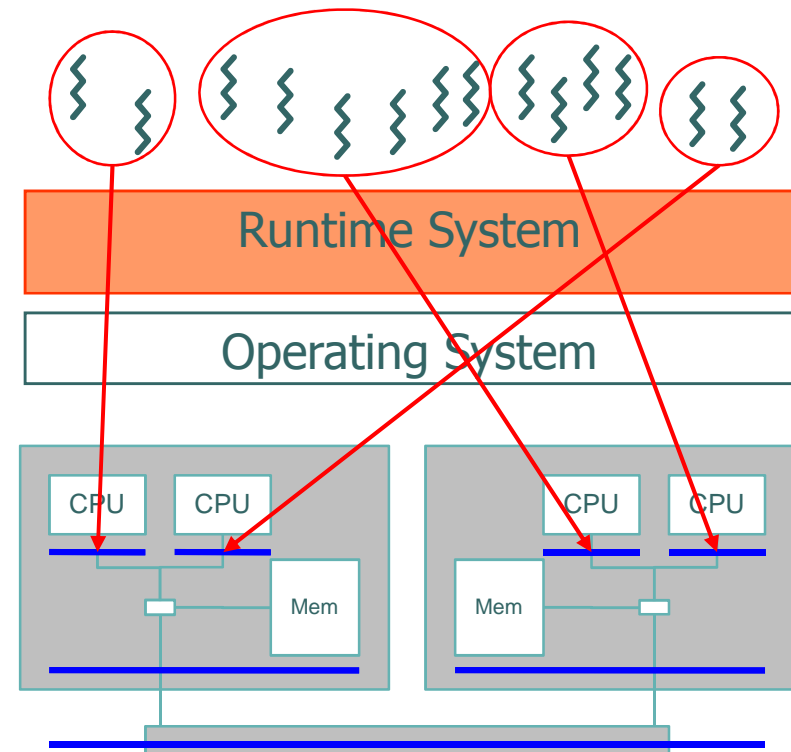
- Scalable storage management for new-generation, data-oriented
- high-performance Petascale applications
  - Data-centric e-Science applications
  - Need to store, share and huge amounts of information
  - Massive data objects: BLOBs (Terabytes)
  - High concurrency for accessing massive shared data ( $10^3$  concurrent clients) at fine grains

Goal: Build a scalable BLOB-based file system, able to cope with huge data under heavy access concurrency on Petascale architectures



# Thread Scheduling over Multicore Hierarchical Machines (INRIA Bordeaux)

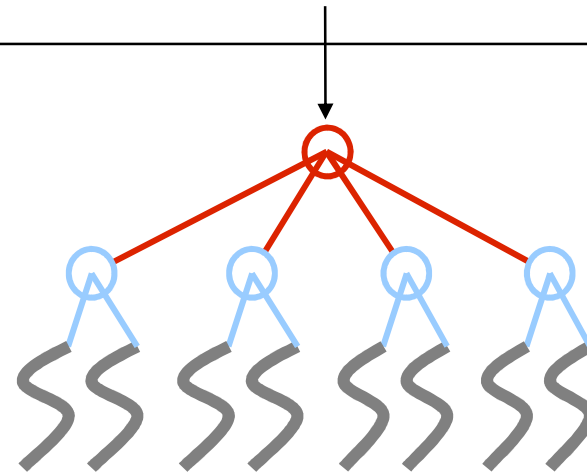
- The Bubble Scheduling concept
  - Capturing application's structure with nested bubbles
    - Accept scheduling directives
    - Extract the structure of parallelism
  - Scheduling = dynamic mapping trees of threads onto a tree of cores
- The BubbleSched platform
  - Designing portable NUMA-aware scheduling policies
    - Focus on algorithmic issues
  - Debugging/tuning scheduling algorithms
    - FxT tracing toolkit + replay animation



# Scheduling OpenMP programs using bubbles

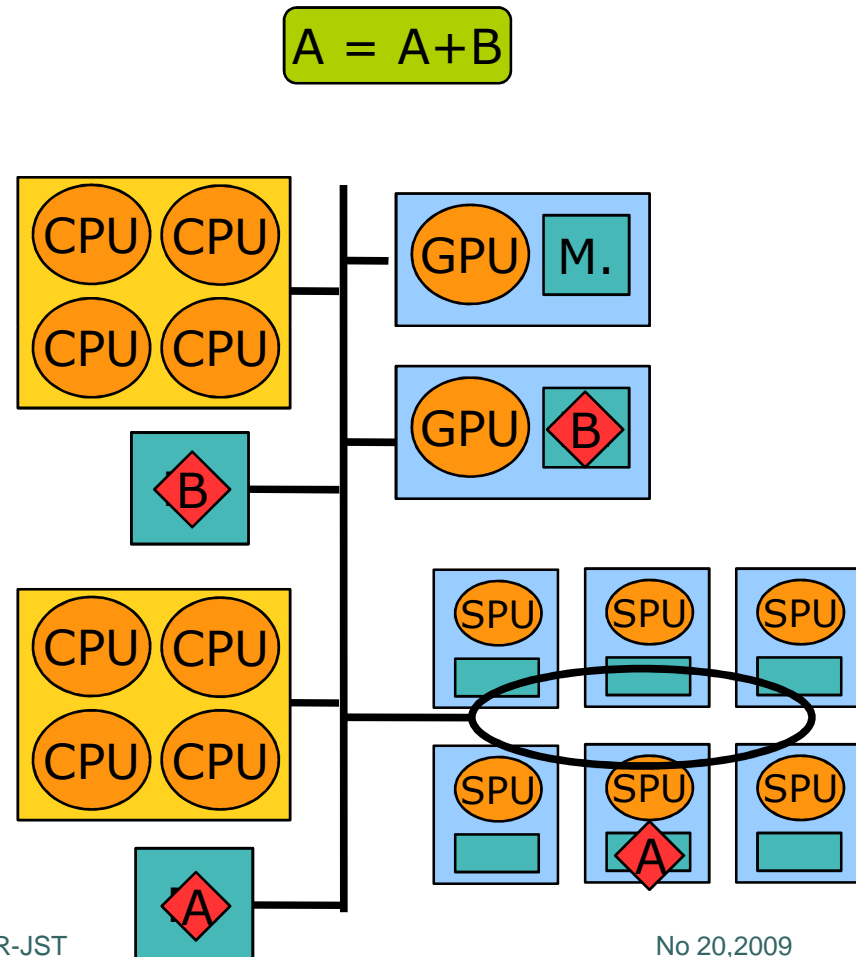
- Designing multicore-friendly programs with OpenMP
  - Parallel sections generate bubbles
  - Nested parallelism is welcome!
    - Lazy creation of threads
- The ForestGOMP platform
  - Extension of GNU OpenMP
    - Binary compliant with existing applications
  - Excellent speedups with irregular applications
    - Implicit 3D surface reconstruction
    - Tree depth > 15, more than 300,000 threads

```
void work()  
{  
    ...  
    #pragma omp parallel for  
    for (int i=0; i<MAX; i++)  
    {  
        ...  
        #pragma omp parallel for  
        num_threads (2)  
        for (int k=0; k<MAX; k++)  
        {  
            ...  
        }  
    }  
}
```



# Programming Hybrid Architectures

- The StarPU Runtime System
  - Mastering CPU+GPU+SPU = \*PU
- Challenge = exploiting all computing units simultaneously
  - Dynamically schedule tasks on all processing units
    - See a pool of heterogeneous cores
  - Avoid unnecessary data transfers between accelerators
    - Need to keep track of data copies





## Parallel Algorithms and Optimization (CNRS-IRIT)

- Design of sparse linear algebra kernels (full, sparse direct and iterative) on current HPC architectures
- Parallelization of large scale applications
- Nonlinear optimization and optimal control
- Data assimilation
- E.g. Developement MUMPS software in collaboration with CERFACS, LIP  
ENSL, . . .



## MUMPS: A MULTifrontal Massively Parallel Solver

- **MUMPS** solves large systems of linear equations of the form  $Ax=b$  by factorizing  $A$  into  $A=LU$  or  $LDL^T$
- Symmetric or unsymmetric matrices (partial pivoting)
- Parallel factorization and solution phases (uniprocessor version also available)
- Iterative refinement and backward error analysis
- Various matrix input formats
  - **assembled** format
  - **distributed** assembled format
  - sum of **elemental** matrices
- Partial factorization and Schur complement matrix
- Version for complex arithmetic
- Several orderings interfaced: AMD, AMF, PORD, METIS, SCOTCH



# High Post Petascale Computing (HP2C)

– name to be confirmed

- INRIA (Saclay, Rennes, Bordeaux, Lyon), CNRS (IRIT, PRISM), CEA (DEN Saclay), CNRS LIA JFLI at Tokyo.
- Tsukuba Univ, Today, Titech, Kyoto Univ.
- Languages would be the main goal of the project (YML-XcalableMP, Bubles )
- Back end on High Performance computers (T2K, TSUBAME, NG,...)
- Resilience, energy consumption optimization, and accuracy will be important key words along all the work-packages of the project (cf. French and Japanese project)
- New Linear algebra and optimization algorithm will be proposed (INRIA, CNRS-PRISM-IRIT, Tsukaba, Today)