Laboratoire
d'Informatique
Parallélisme
Réseaux
Algorithmique
Distribuée

UNIVERSITÉ DE VERSAILLES ST-QUENTIN-EN-YVELINES

université PARIS-SACLAY

MAISON DE LA SIMULATION

HIGH PERFORMANCE DATA ANALYTICS AND SOME APPLICATIONS

Nahid.Emad@uvsq.fr

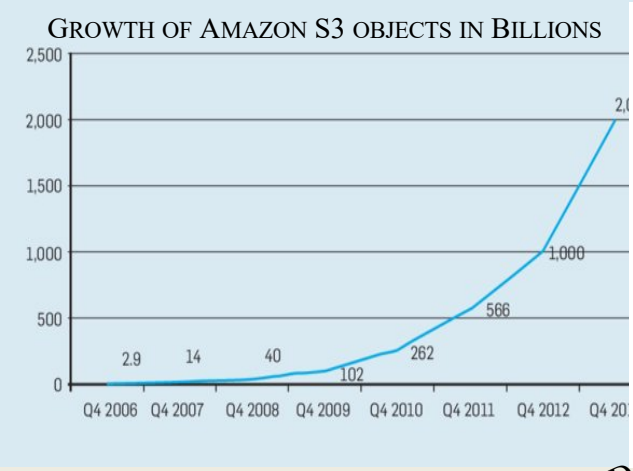
Workshop on Latest Advances in Scalable Algorithms for Large-Scale Systems

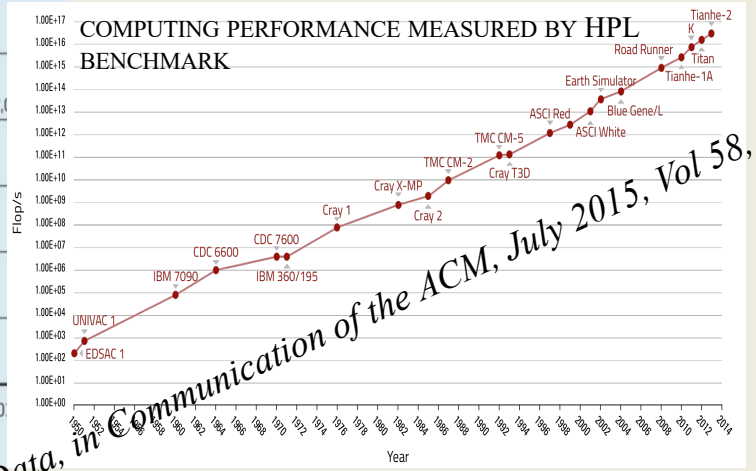
November 12, 2020

BIG DATA ANALYSIS & HPC

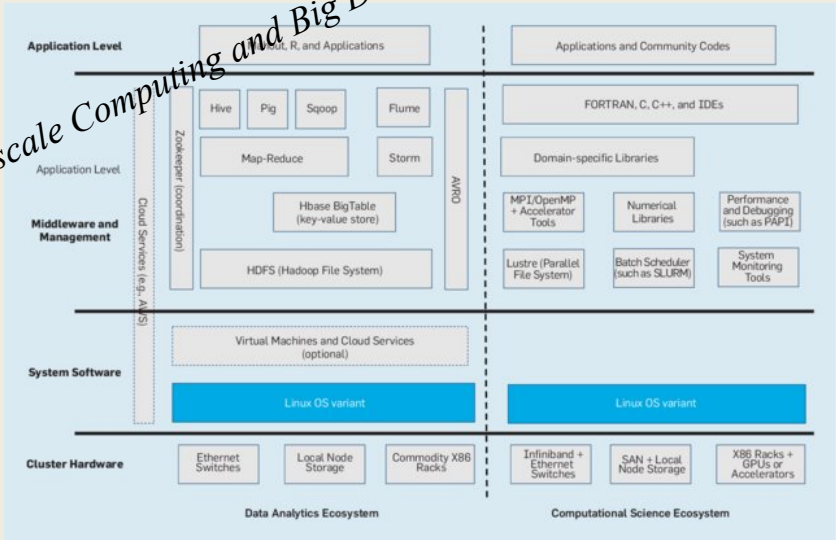
The Fourth Paradigm: ***Data-Intensive Scientific Discovery***, Tony Hey, Stewart Tansley, Kristin Tolle, Published by Microsoft Research | October 2009, ISBN: 978-0-9825442-0-4

Big Data Analysis & HPC Evolutions





D. A. Reed and J. Dongarra , Exascale Computing and Big Data, in Communication of the ACM, July 2015, Vol 58, N°7.



The tools and cultures of HPC and big data analytics diverged, to the detriment of both; unification is essential to address a spectrum of major research domains.

BIG DATA ANALYSIS & HPC

- Emerging Exascale supercomputers : What programming paradigms ? What methods for what applications? How efficiently program such architectures exploiting mixed arithmetics (and 16, 32, 64 bits,...), etc. ? How to manage the convergence of distributed and parallel computing in these architectures?
- New programming paradigms have to be proposed for this extreme computational and data sciences programming.
- New methods have to be developed (involving applied math, graph theory, Bayesian network, statistic, linear algebra, game theory, …..)
- Big Data analysis and HPC would converge to develop new applications on those platforms/supercomputers (mixing computational science and data science)
- This convergence is crucial to propose future machine learning algorithm for Post-Petascale platforms and supercomputers

OUTLINE

- Some application challenges
- Programming models and frameworks
- Unite and conquer approach
- Concluding remarks and perspectives

OUTLINE

- Some application challenges
- Programming models and frameworks
- Unite and conquer approach
- Concluding remarks and perspectives

Some application: Gamma ray detection

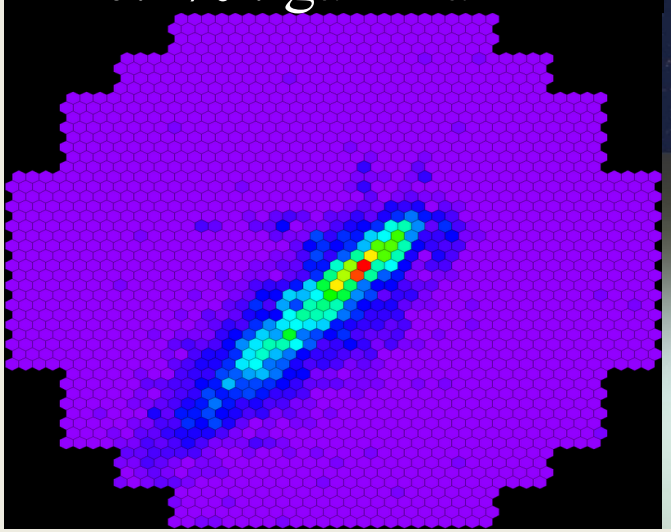
Gamma Rays

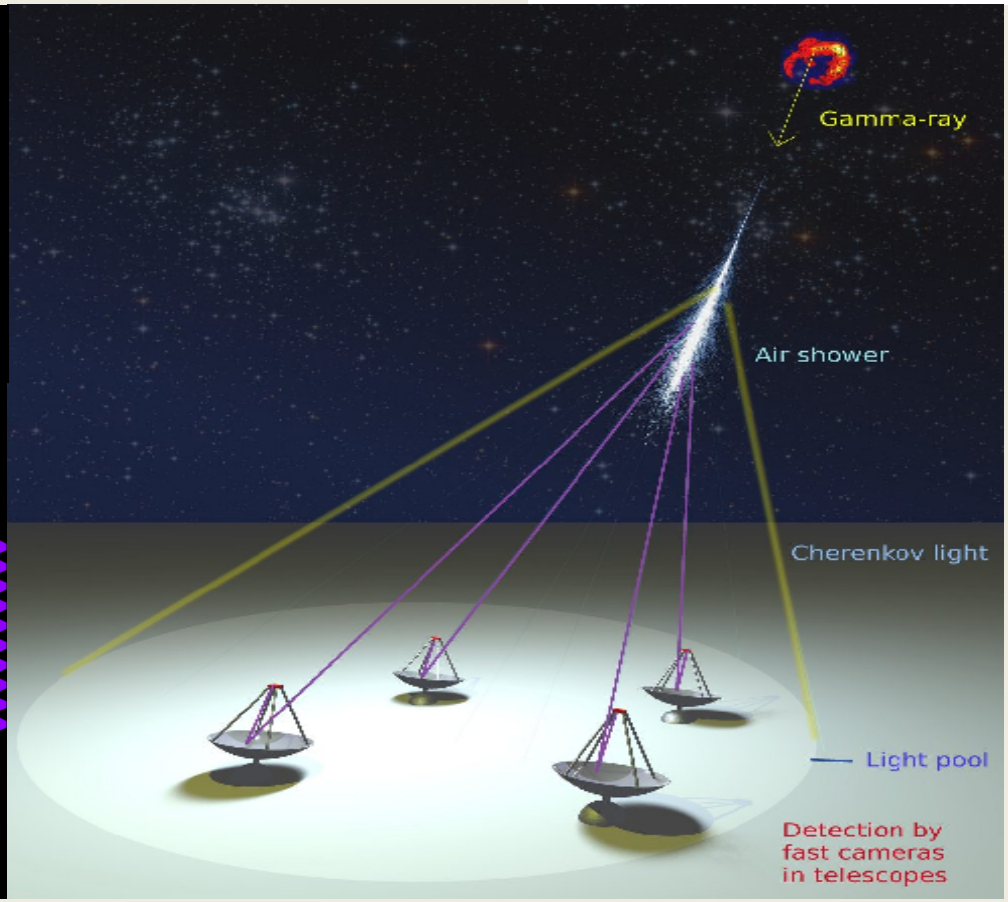
90% : Protons

10% : heavy kernels

1% : electrons

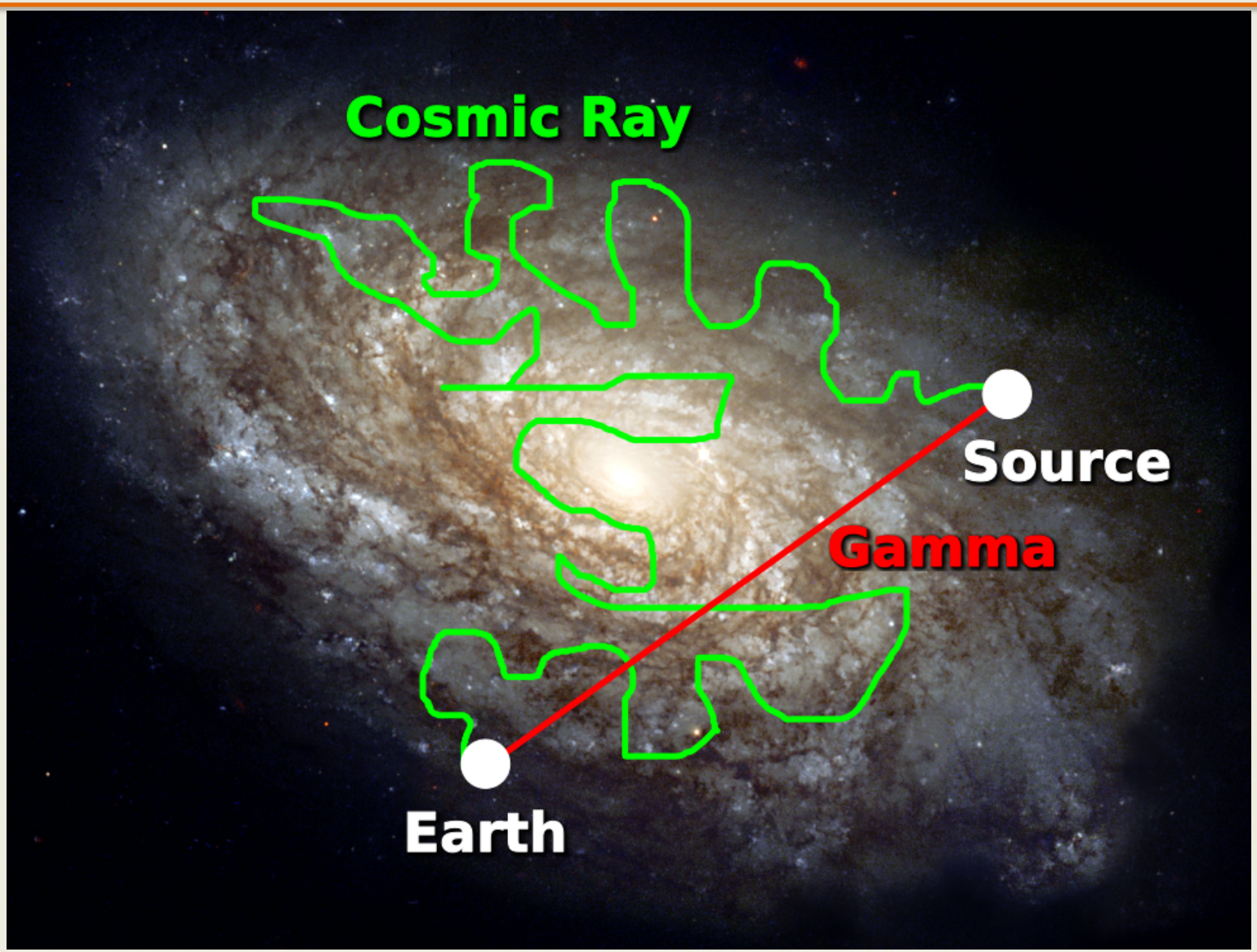
0.1% : gamma



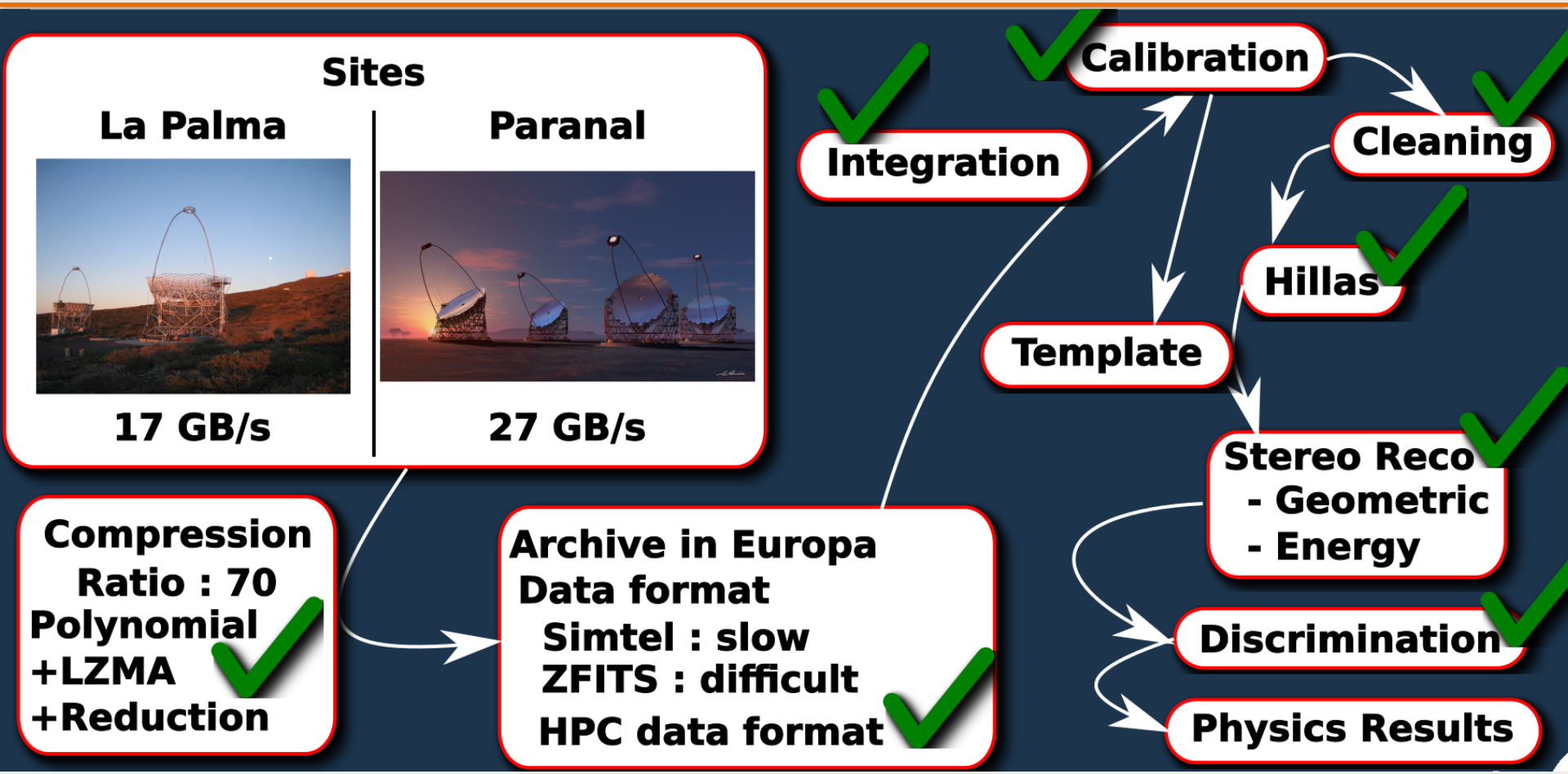


P. Aubert, N. Emad, F. Gaté J. Jacquemier, G. Lamanna G. Maurin, T. Vuillaume,

Some application: Gamma ray detection



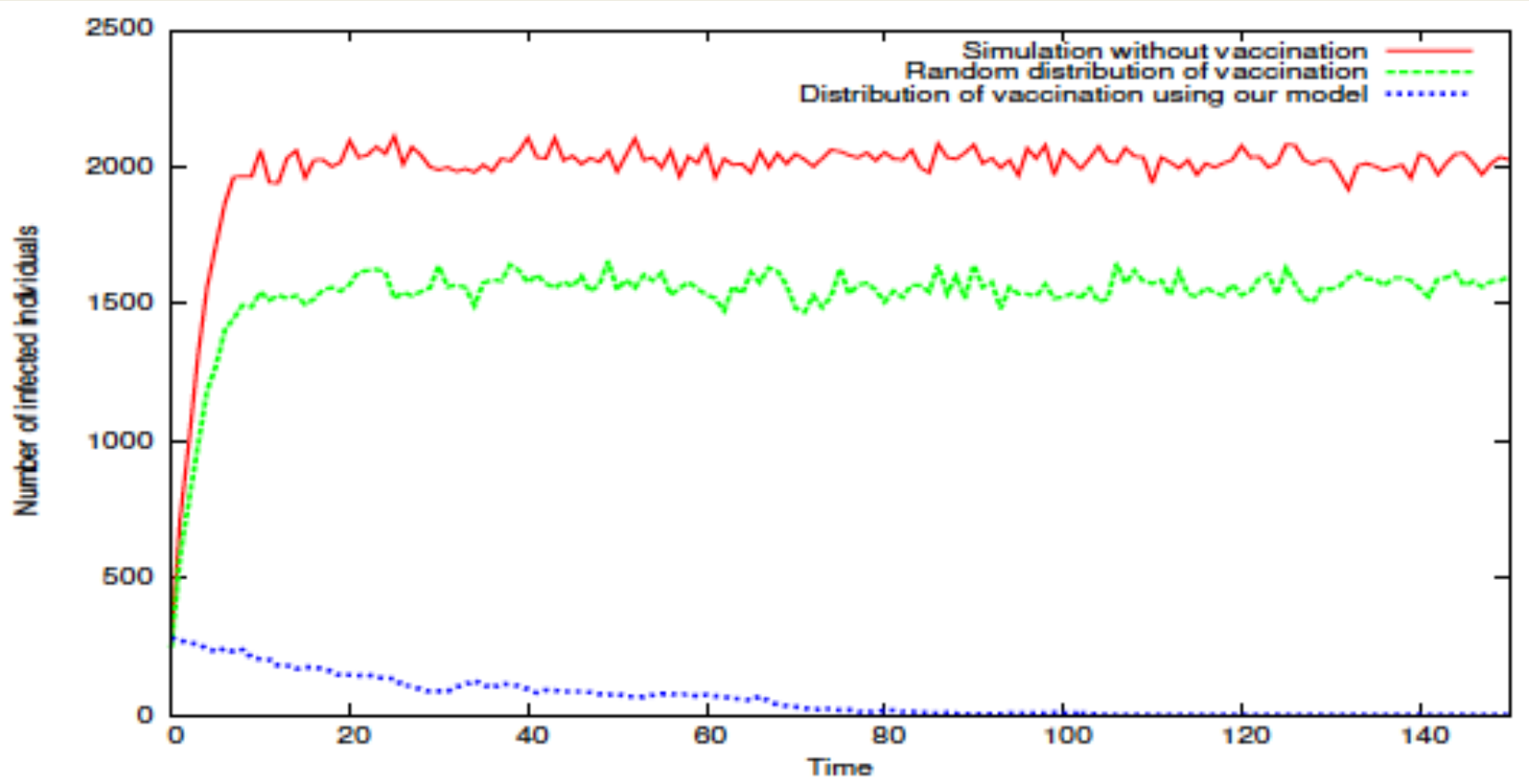
Some application: Gamma ray detection (**SVD**)



- ***Polynomial data compression for large-scale physics experiments***. P. Aubert, T. Vuillaume, G. Maurin, J. Jacquemier, G. Lamanna, and N. Emad. CoRR, abs/1805.01844, 2018.
- ***High Performance Computing algorithms for Imaging Atmospheric Cherenkov Telescopes***. T. Vuillaume, P. Aubert, G. Maurin, J. Jacquemier, G. Lamanna, and N. Emad. Proceeding of Science, ICRC2017-771, 2017.
- ***Data Analysis with SVD for Physical Expriments, Application to the Cherenkov Telescope Array***. P. Aubert, T. Vuillaume, F. Gaté, G. Maurin, J. Jacquemier, N. Emad, and G. Lamanna. In the Proceedings of SIAM Conference on Parallel Processing for Scientific Computing, Tokyo, Japan, March 7-10 2018.

SOME APPLICATION: FLU EPIDEMIC

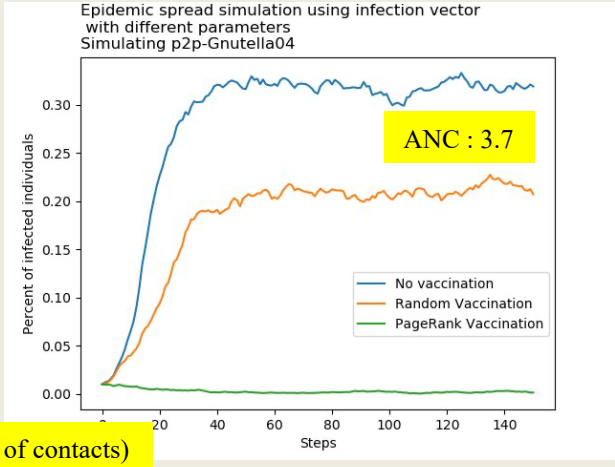
Stochastic simulation using the infection vector (**EVD of transition matrix**)

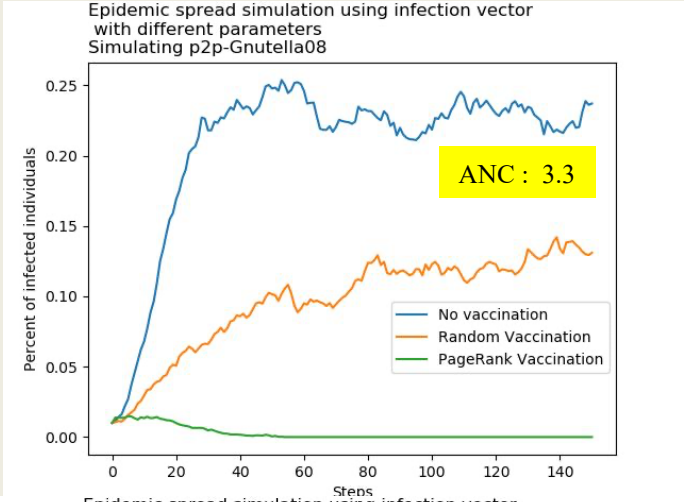


Time series of infection in an 7010-node power-law social graph ba, with $\nu = 0.2$, $\delta = 0.24$ and $x=5$

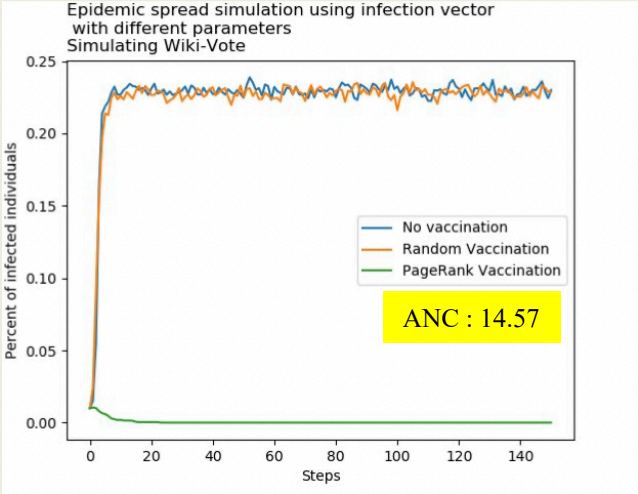
SOME APPLICATION: FLU EPIDEMIC

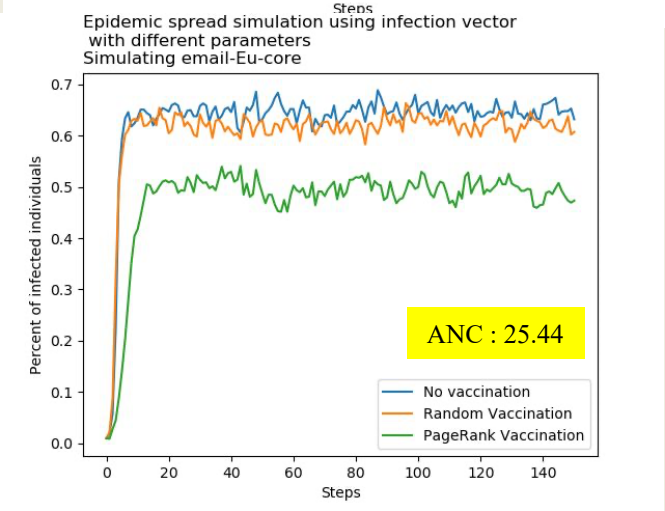
Stochastic simulation using the infection vector (**EVD of transition matrix**)





ANC (average number of contacts)





Simulation by ISTY/UVSQ engineering students L. ARMAND and R. COUZINET

SOME APPLICATION : THE CLUSTERING PROBLEM

Example: detect relevant groups based on frequent co-purchasing on Amazon.com

Data: V. Krebs. 2004

Visualization: M. Bastian, S. Heymann, and M. Jacomy. “Gephi: An Open Source Software for exploring and manipulating networks” 2009

An application: the clustering problem

Pink Liberal

Yellow Neutral

Green Conservative

Data: V. Krebs. 2004

Visualization: M. Bastian, S. Heymann, and M. Jacomy. "Gephi: An Open Source Software for exploring and manipulating networks" 2009

Clustering techniques

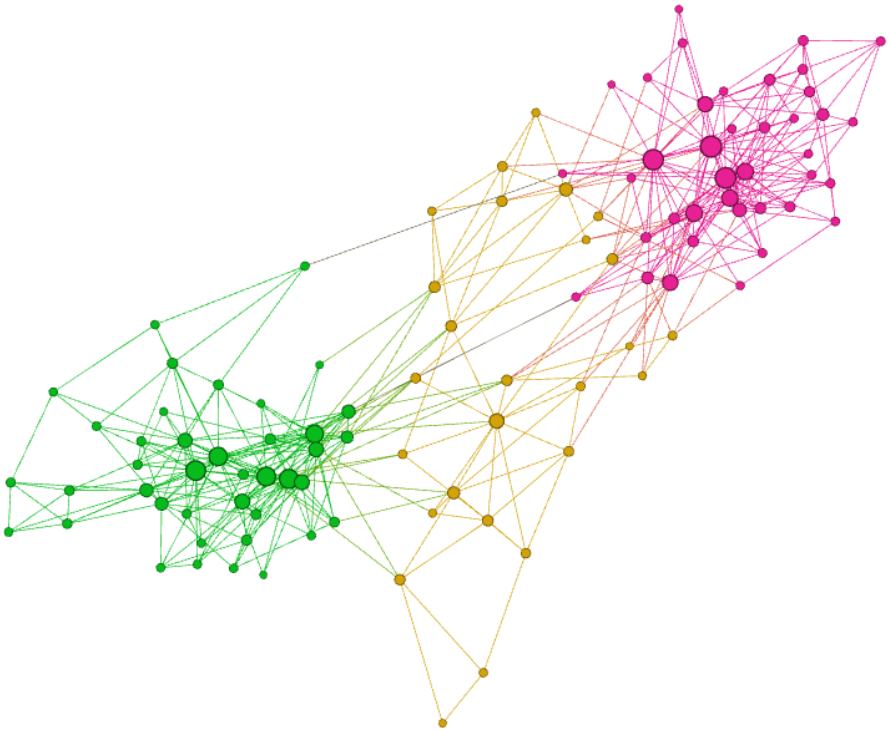
Partitioning vertices V of a graph $G = (V, E)$ into a set of clusters $S_k \subseteq V$ such that: $\cup_{k=1}^{p} S_k$.

- **Modularity maximization allows to compute the difference between vertices assigned into clusters for a graph $G = (V, E)$ versus a random graph $R = (V, F)$. The issued optimization problem is NP-Complete. It can be approximated by the largest eigenpairs of the modularity matrix: $B = A - \frac{1}{2\omega} v.v^T$ where $v^T = (v_1, \cdots, v_n)$ is the volume vector, v_i the volume of the node i and ω is the number of edges of G.**
- **Minimum balanced cut permits to minimize the number of edges between clusters. This problem is also an NP-Complete problem, which can be approximated by the smallest eigenpairs of the Laplacian : $L = D - A$ where D is the degree matrix of G.**

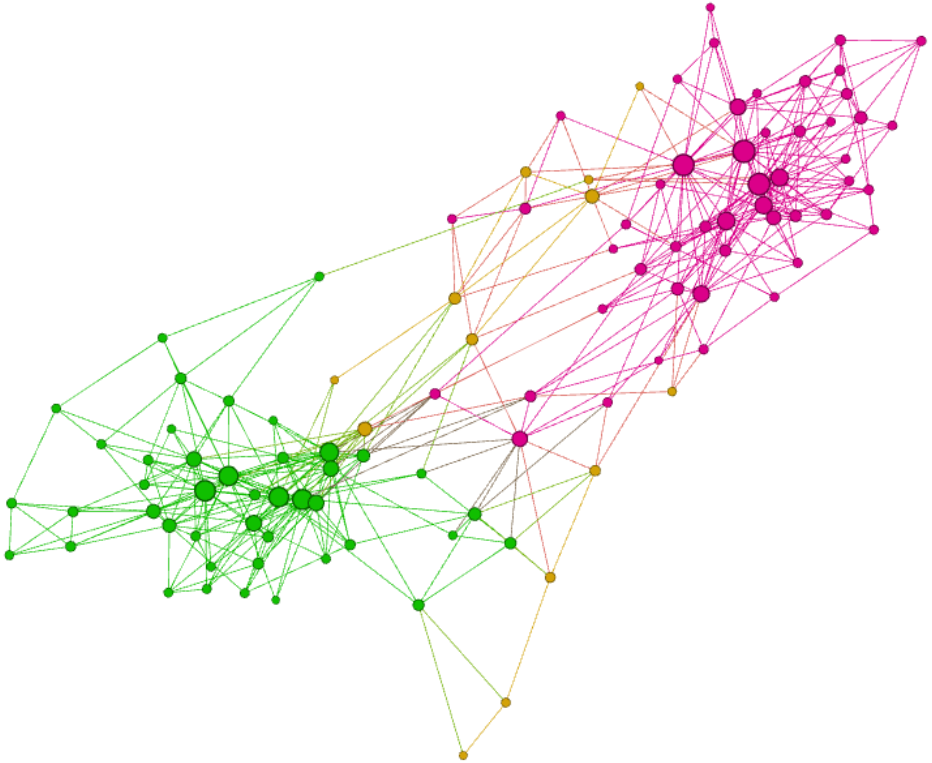
MODULARITY MAXIMIZATION CLUSTERING

84% hit rate

Spectral Modularity maximization



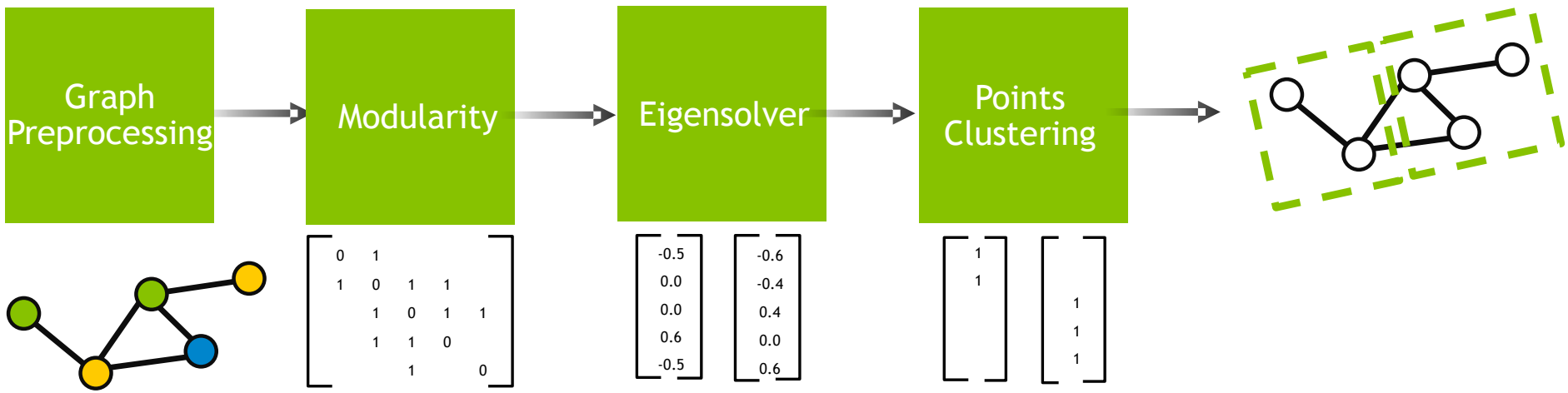
Ground truth



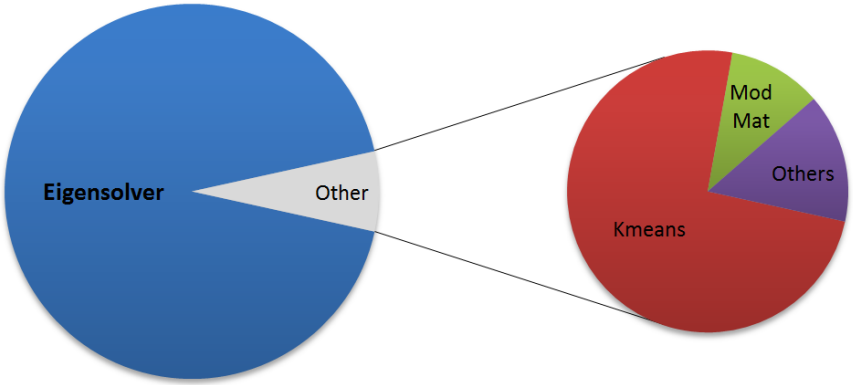
A. Fender, N. Emad, S. Petiton, M. Naumov, ***Parallel Modularity Clustering****, Procedia Computer Science, Volume 108, 2017, Pages 1793-1802*

Algorithm: Modularity maximization clustering

Let $G = (V, E)$ be an input graph and A be its weighted adjacency matrix.
Let p be the number of desired clusters.
Set the modularity matrix $B = A - \frac{1}{2\omega}\mathbf{v}\mathbf{v}^T$.
Find p largest eigenpairs $BU = U\Sigma$, where $\Sigma = \mathrm{diag}(\lambda_1, ..., \lambda_p)$.
Scale eigenvectors U by row or by column (optional).
Run clustering algorithm, such as k-means, on points defined by rows of U.

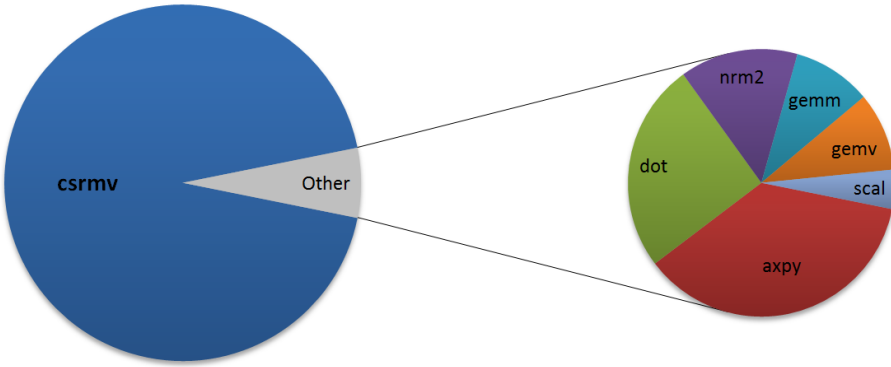


PROFILING: MODULARITY CLUSTERING



The eigensolver takes **90%** of the time

The sparse matrix vector multiplication takes **90%** of the time in the eigensolver



Joe Eaton (NVIDIA, USA), Alexandre Fender (UVSQ/NVIDIA), Maxim Naumov (NVIDIA, USA), Serge Petiton (Cristal/MDLS, Lille U)

OUTLINE

- Some application challenges
- Programming models and frameworks
- Unite and conquer approach
- Concluding remarks and perspectives

TRENDS FOR EMERGING SUPERCOMPUTERS

- Multi-core to many cores:
  Minimization of data movement but all communication has not the same cost and the latencies between distant cores are time consuming
- High degree of hierarchy (nodes, memory):
  Multi-level parallelism (coarse, medium, fine grain), multi-level scheduling strategies, …
- Convergence of parallel and distributed systems:
  Communication, energy consumption, fault tolerance

GOAL: Exploitation of different hierarchies of processing elements and shared and distributed memories of these evolutionary systems

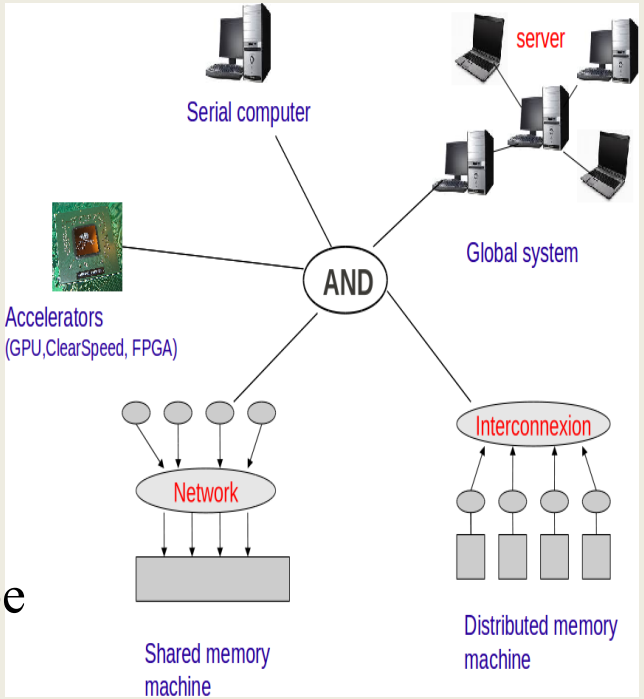
PROGRAMING PARADIGM

COMPONENT APPROACH :

- ✓ Interoperability, reusability, durability, …
- ✓ Making (re)use of existing libraries, ..., in the context of extreme scale computing

GRAPH OF COMPONENTS :

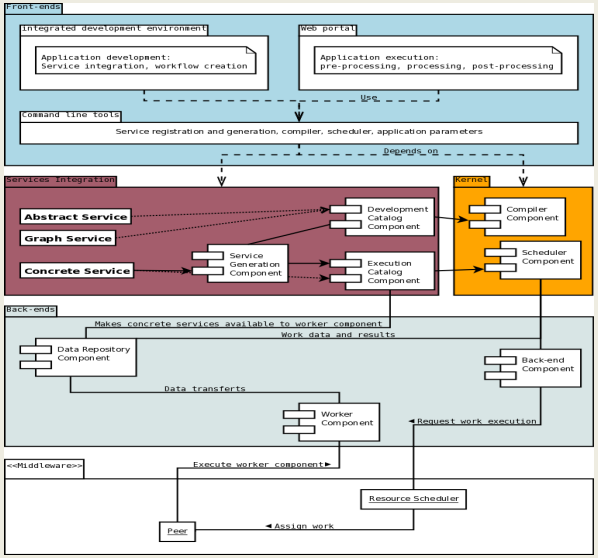
- ✓ Graph of very coarse grain components : data flow oriented SPMD, PGAS-like, data-parallelism
- ✓ A component can be itself a graph of tasks and can be descripted by SPMD PGAS-like model
  - limitation of communications to the cores allocated to such components
  - on each processor, we can program accelerators
  - on each core multithreaded optimization can be used



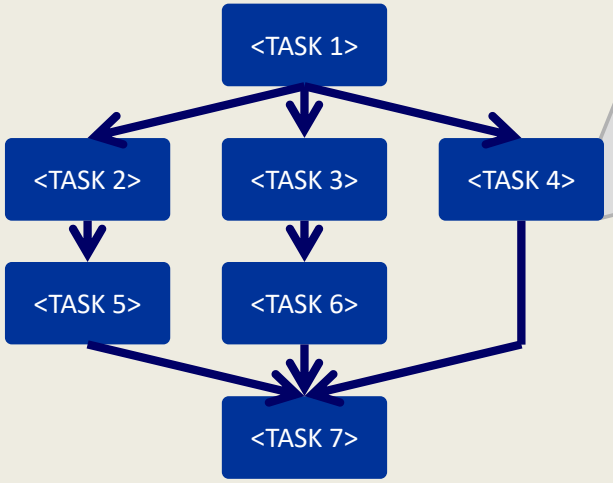
Users have to be able to give expertise to middleware, runtime system & schedulers

YML: An environment and high-level language (http://yml.prism.uvsq.fr/)

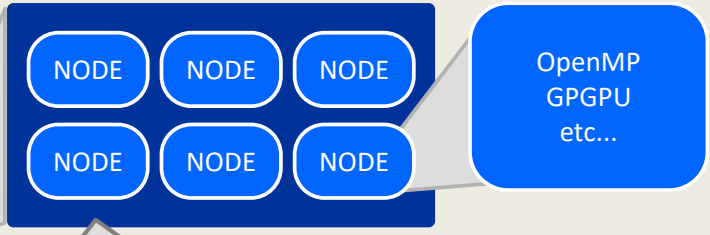
YML FRAMEWORK/LANGUAGE AND XMP LANGUAGE



S. Petiton (Lille and MDLS), O. Delanoy (UVSQ), T. Dufaud (UVSQ and MDLS), M. Sato and Miwako Tusji (U. Tsukuba then RIKEN), T. Boku (U. Tsukuba), B. Chapman (U. Houston, then U. Stony Brook and BNL), J. Protze, M. Müller and C. Terboven (RTWH, Aachen), M. Dandouna (NumeriX), L. Drummond, O. Marques (LBNL), L. Choy (U. Lille, U. Tsukuba/Japan,), M. Hugues (Total and Google/Asutin), S. Ling and Y. Zhang (U. Lille 1 and U. Hohai/China), H. Haiwu (U. Lille 1, then U. Hohai/China, the Chinese Academy of Science), Xinzhe Wu (MDLS/CNRS), J. Gurhen (U. Lille 1 and MDLS), ...



YML provides a workflow programming environment and high level graph description language called YvetteML




```
for(i=0;i<n;i++){
    for(j=0;j<n;j++){
        tmp[i][j]=0.0;
#pragma xmp loop (k) on t(k)
        for(k=0;k<n;k++){
            tmp[i][j]+=(m1[i][k]*m2[k][j]);
}}}
#pragma xmp reduction (+:tmp)
```

Several French/Japanese/German ANR/JST/DFG and SPPEXA projects based on YML and XMP

Each task is a parallel program over several nodes.
XMP language can be used to descript parallel program easily!

WHAT NUMERICAL METHODS?

Main characteristics based on proposed programing paradigm:

- Avoiding synchronous communication (such as large scalar products, overall synchronization)
- Promoting asynchronicity
- Taking into account heterogeneity
- Introducing fault tolerance
- Encouraging load balancing possibility
- Introducing auto-tuning, machine learning

OUTLINE

- Some application challenges
- Programming models and frameworks
- Unite and conquer approach
- Concluding remarks and perspectives

UNITE AND CONQUER APPROACH

UC/ Hybrid methods \ Components of hybrid methods (co-methods)				
Bi-Lanczos	+	Bi-Lanczos	+	...
ERAM	+	ERAM	+	...
ERAM	+	GMRES	+	...
Projection method	+	Convergence accelerator	+	...
...		...		

UC method : A combination of two or more iterative methods in order to accelerate the convergence of one of them.

Multiple iterative method A combination of two or more instances of the same iterative method in order to accelerate the convergence of one of them**.**

Saad (Chebyshev acceleration techniques for solving nonsymmetric eigenvalue; 1984), Brezinski (hybrid procedures for solving linear systems; 1994), Code coupling (in simulation), ...

RESTARTED KRYLOV SUBSPACE METHODS

Let (P_n) be an eigenproblem (or a linear system) with A a large and sparse n-order matrix.

The Arnoldi projection of (P_n) onto Krylov subspace $K_m(A,v)=span(v, Av,\ldots,A^{m-1}v)$ can be expressed as:

$$AV_m=V_mH_m+f_me^T{}_m.$$

Then, (P_m) :

$$H_my_i=\lambda_iy_i \;\; for\ i=1,\ \ldots m$$

can represent (P_n) in $K_m(A,v)$ subspace.

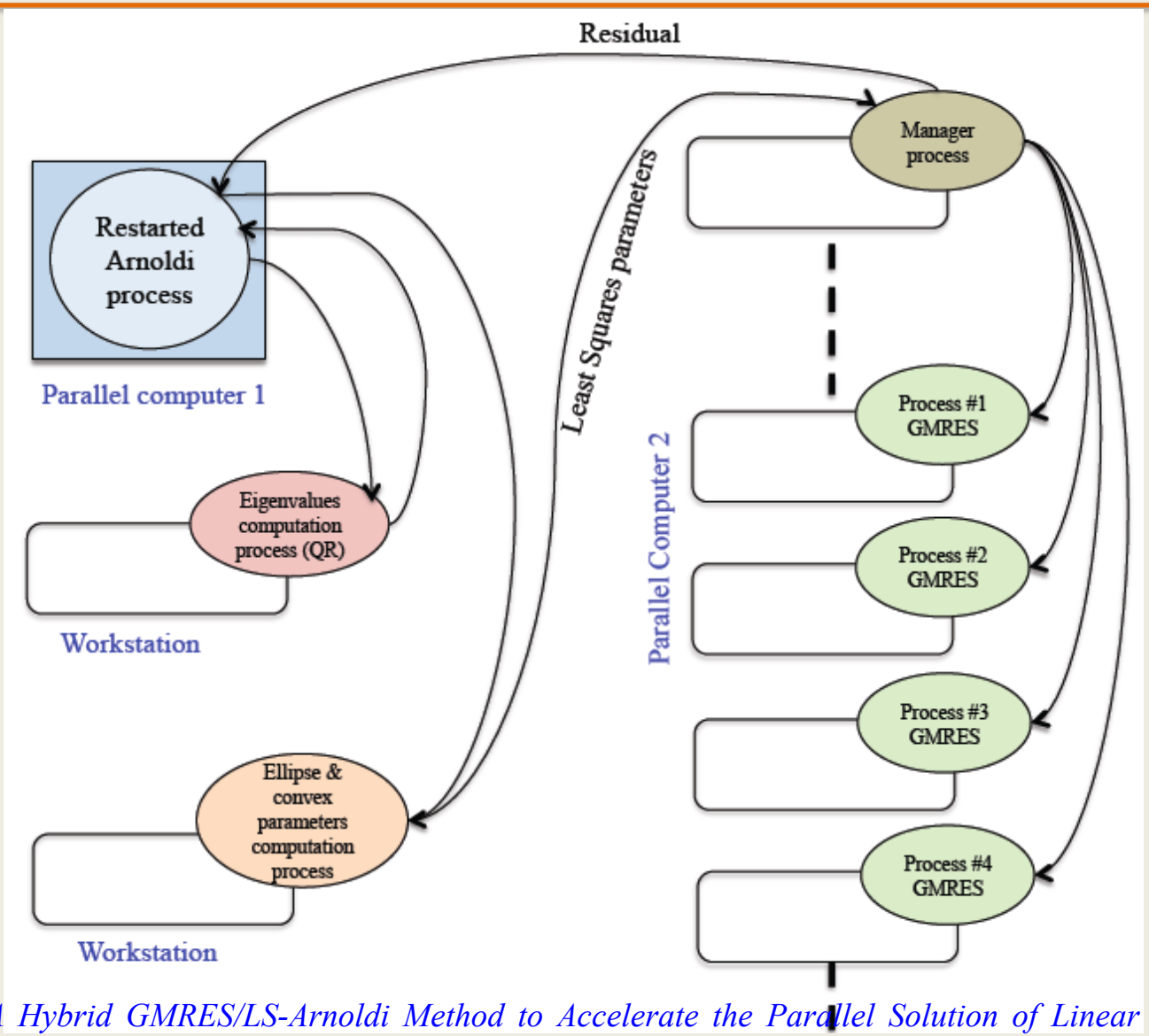
RESTARTED KRYLOV SUBSPACE METHODS

- *INIT: Choose k, m, v, tol, nbRC (eigenproblem)*
- *ITERATE:*
  a. *Arnoldi reduction :* $AV_m=V_mH_m+f_me^T{}_m$
  b. *Solve reduced problem:* $H_my_i=\lambda_iy_i$ *for* $i=1, \ldots m$
  c. *Compute Ritz-pairs* (Λ_k,U_k), *with* $\Lambda_k=(\lambda_1, \ldots, \lambda_k)$ *and* $U_k=[V_my_1,\ldots, V_my_k]$.
  d. *If (convergence) stop.*
  e. *Restart implicitly/explicitly ITERATE by defining an adapted restarting strategy* $f(\Lambda_k, U_k)$

What is the correlation between parameters *k, m, v, tol, nbRC,* $f(\Lambda_k, U_k)$?

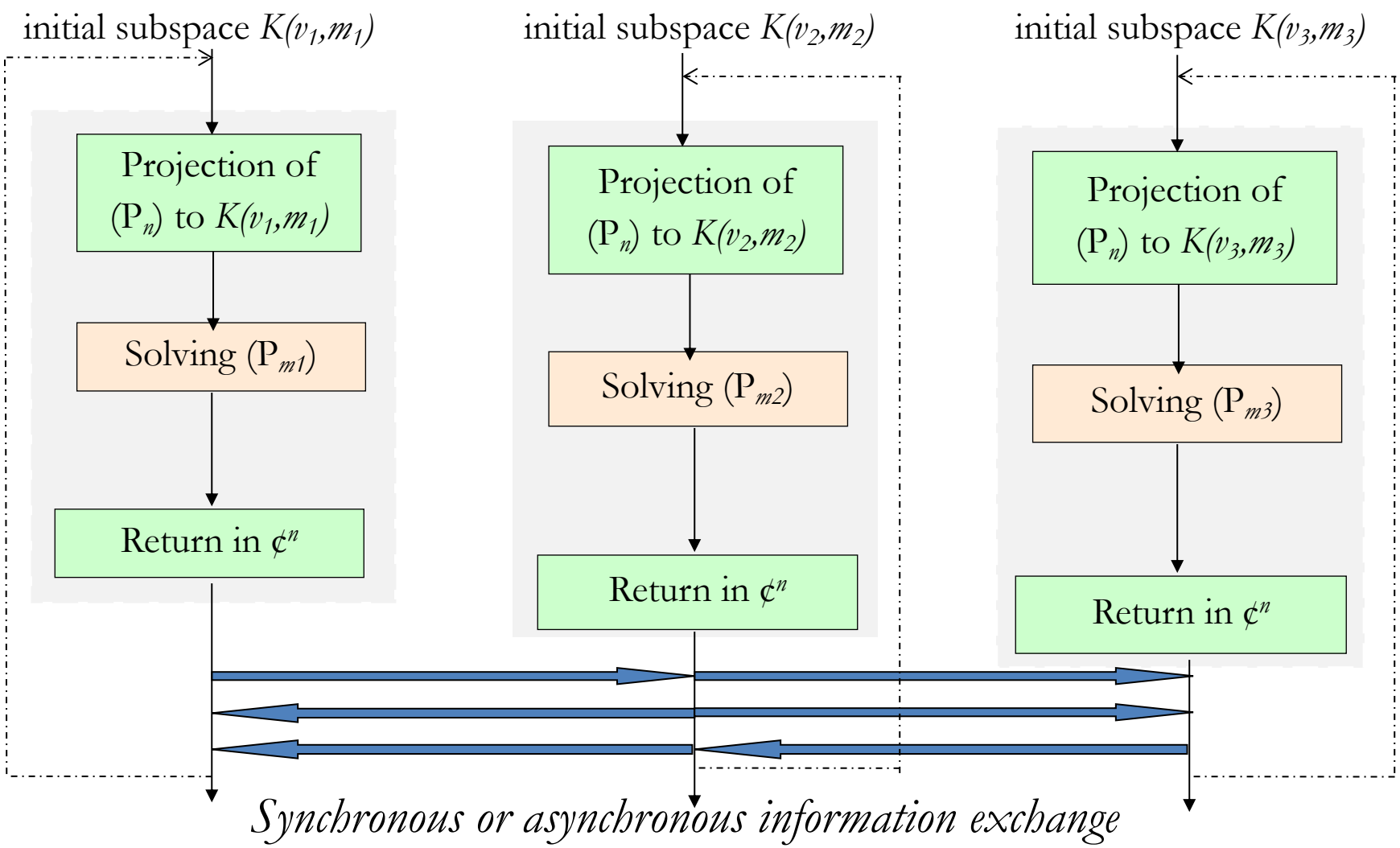
GMRES/LS-ARNOLDI FOR LINEAR SYSTEM

- Multi level parallelism: coarse & fine grain
- Asynchronous communication
- Fault tolerance
- Load balancing



Haiwu He, C. Bergere, S. Petiton,. A Hybrid GMRES/LS-Arnoldi Method to Accelerate the Parallel Solution of Linear Systems. Computers and Mathematics with Applications 51 (2006) 1647-1662.

MULTIPLE RKSM: A PARTICULAR CASE OF UC APPROACH



MULTIPLE METHOD: A PARTICULAR CASE OF UC APPROACH

A combination of two or more instances of the same RKSM for accelerating the convergence of one of them.

The instances can be defined by several Krylov subspaces:
$K_{mi} = span\{v_i, Av_i, \ldots, A^{mi-1}v_i\}$, for $i=1,2,\ldots,l$:

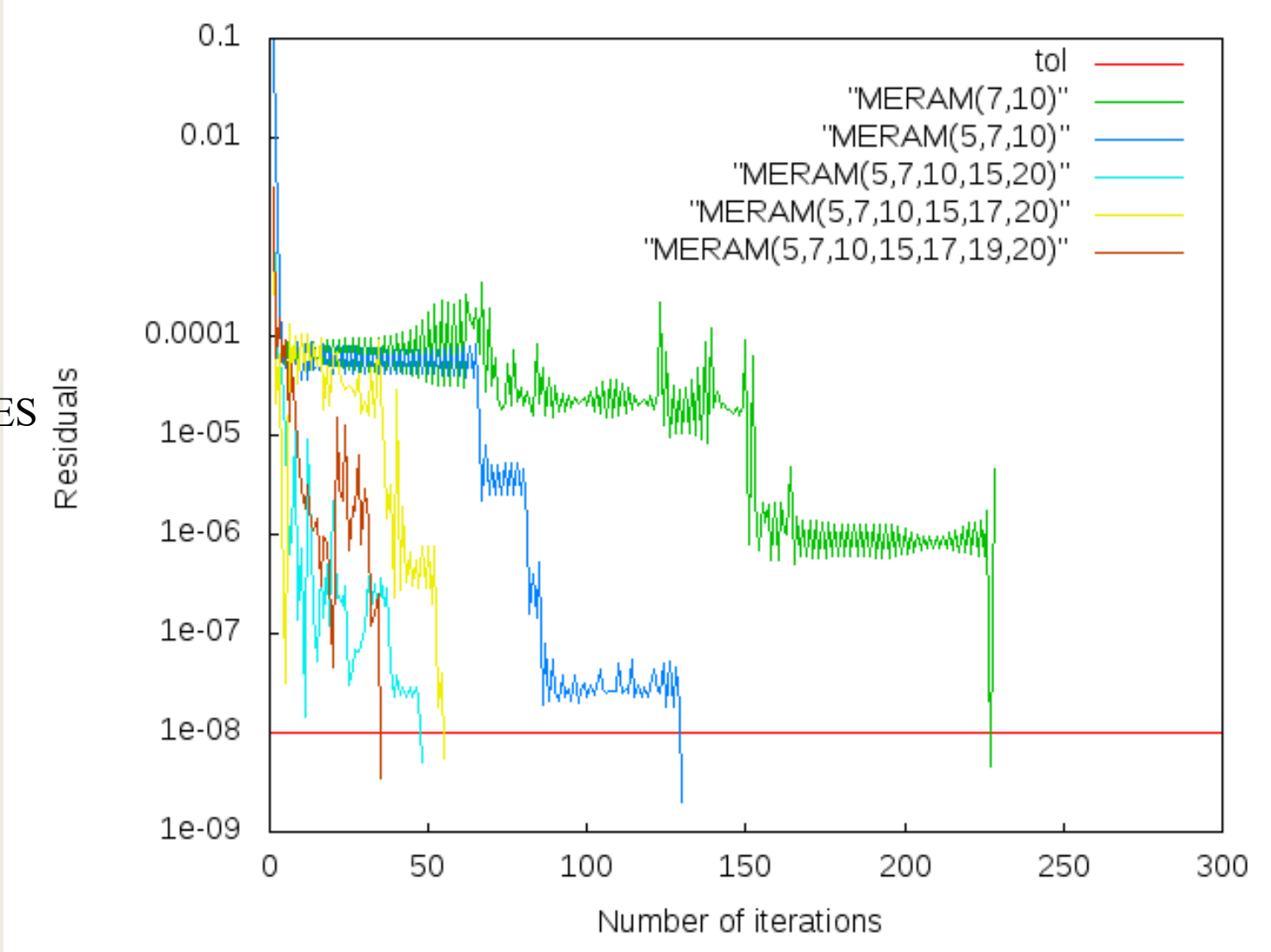
- ✓ Different subspaces :

  K_{mi} with $m_i \neq m_j$ *and* $v_i \neq v_j$ for $i,j \in [1,\ldots,l]$ and $i \neq j$
- ✓ Nested subspaces :

$$K_{m1} \subset K_{m2} \subset K_{m3} \subset \ldots \subset K_{ml}$$

SCALABILITY WRT THE NUMBER OF CO-METHODS

MERAM ON GRID'5000 PLATFORM USING120 CORES AND PETSC/SLEPC IN YML



CHARACTERISTICS OF UC METHODS

- Multi level parallelism (coarse grain and fine grain)
- Asynchronous communication
- Fault tolerance
- Great potential to dynamic load balancing
- Many parameters, many reuse software components
- Need well suited «standard» programming tools

well suited to peta-scale & exascale computing system

Implicitly Restarted Arnoldi method

- *INIT: Choose k, m=k+p, v, tol, nbRC (eigenproblem)*
- *ITERATE:*

  a. *Arnoldi reduction (AR) :* $AV_{k+p} = V_{k+p} H_{k+p} + f_{k+p} e^T_{k+p}$

  b. *Solve reduced problem:* $H_m y_i = \lambda_i y_i$ *for* $i=1, \ldots m$

  c. *Compute Ritz-pairs* (Λ_k, U_k), *with* $\Lambda_k = (\lambda_1, \ldots, \lambda_k)$ *and* $U_k = [V_m y_1, \ldots, V_m y_k]$.

  d. *If (convergence) stop.*

  e. *Implicitly restarting:*
  Do $p=m-k$ *implicitly shifted QR on* H_m $(H^+_m = QH_mQ)$:
  Update AR : $AV^+_{k+p} = V^+_{k+p} H^+_{k+p} + f_{k+p} e^T_{k+p} Q$ *(step a.)*
  Keep the first k *columns of both sides and go to a. and complete AR with* $p= m-k$ *additional steps.*

MIRAM WITH NESTED SUBSPACES ($K_{m1} \subset K_{m2} \subset \ldots \subset K_{ml}$)

$$AV_{m1}=V_{m1}H_{m1}+f_{m1}e^T_{m1} \Longrightarrow (\Lambda_k, U_k)_{m1}$$

$$AV_{m2}=V_{m2}H_{m2}+f_{m2}e^T_{m2} \Longrightarrow (\Lambda_k, U_k)_{m2}$$

$$\ldots \qquad \ldots$$

$$AV_{ml}=V_{ml}H_{ml}+f_{ml}e^T_{ml} \Longrightarrow (\Lambda_k, U_k)_{ml}$$

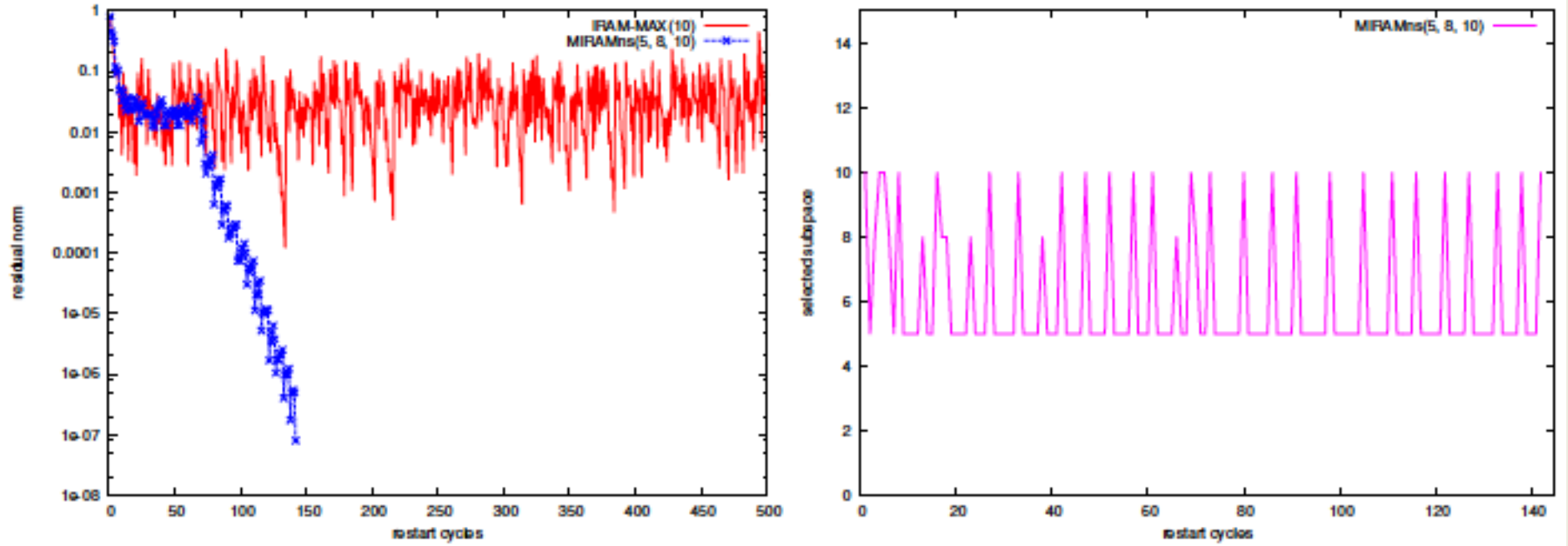
l times a-d steps of IRAM algorithm

$$\Longrightarrow (\Lambda_k, U_k)_{m\_best}$$

Let $m=m_{best}$ $p=p_{best}=m_{best}-k$

Do $p=m-k$ *implicitly shifted QR* ($H^+_m=QH_mQ$):
Update AR : $AV^+_{k+p}=V^+_{k+p}H^+_{k+p}+f_{k+p}e^T_{k+p}Q$ *(step a.)*
Keep the first k *columns of both sides and go to a. and complete AR with* $p= m-k$ *additional steps.*

AUTOTUNED IRAM AND EVOLUTION OF m_{BEST} WITHIN ARPACK



(a) MIRAMns(5, 8, 10) versus IRAM(10) (b) Evolution of m_{best} along cycles

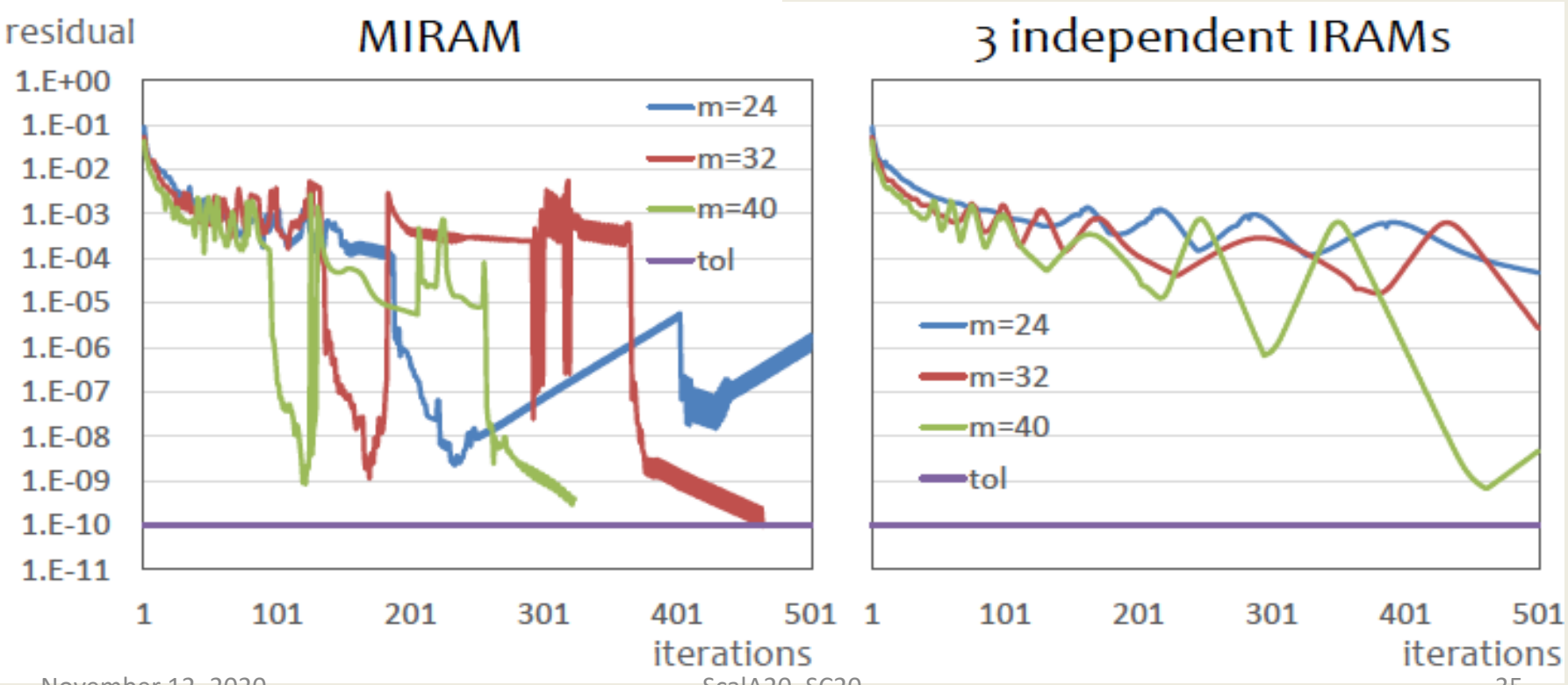
Bfw782a, k=2, tol=$10^{-8}$and a random initial guess

MIRAM with FP2C (XMP/YML) + PETSc/SLEPc/ARPACK: Experiments

- K-Computer
  - Node
    - SPARC64 VIIIfx (8core) CPU 128GFLOPS/node
    - 16 GB DDR3 SDRAM
  - System
    - 864 rack x 102 nodes x 1 CPU = 88,128 CPUs
    - Tofu: six-dimensional torus interconnect
    - FEFS (Fujitsu Exabyte File System) based on Lustre

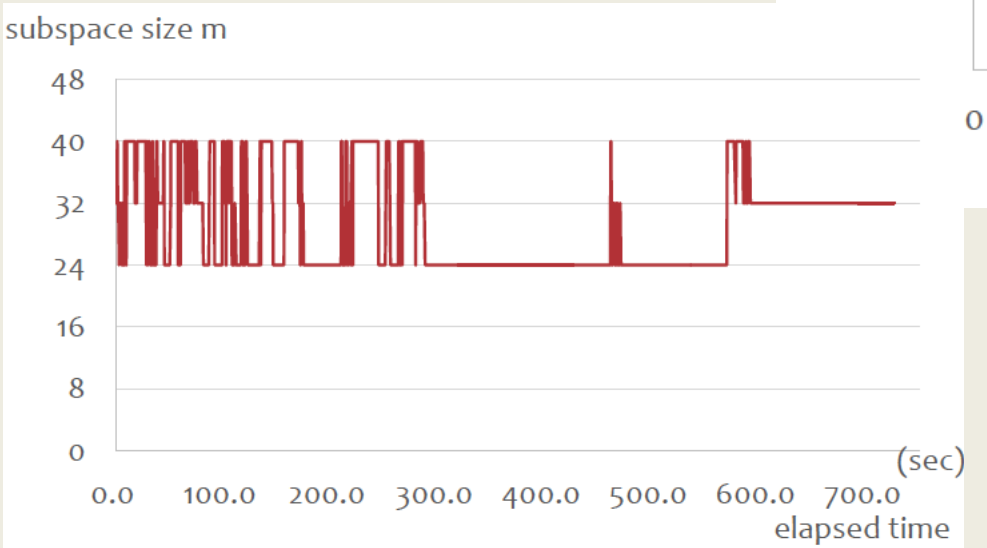
MIRAMns with YML+XMP on K Computer by Miwako TSUJI (RIKEN)

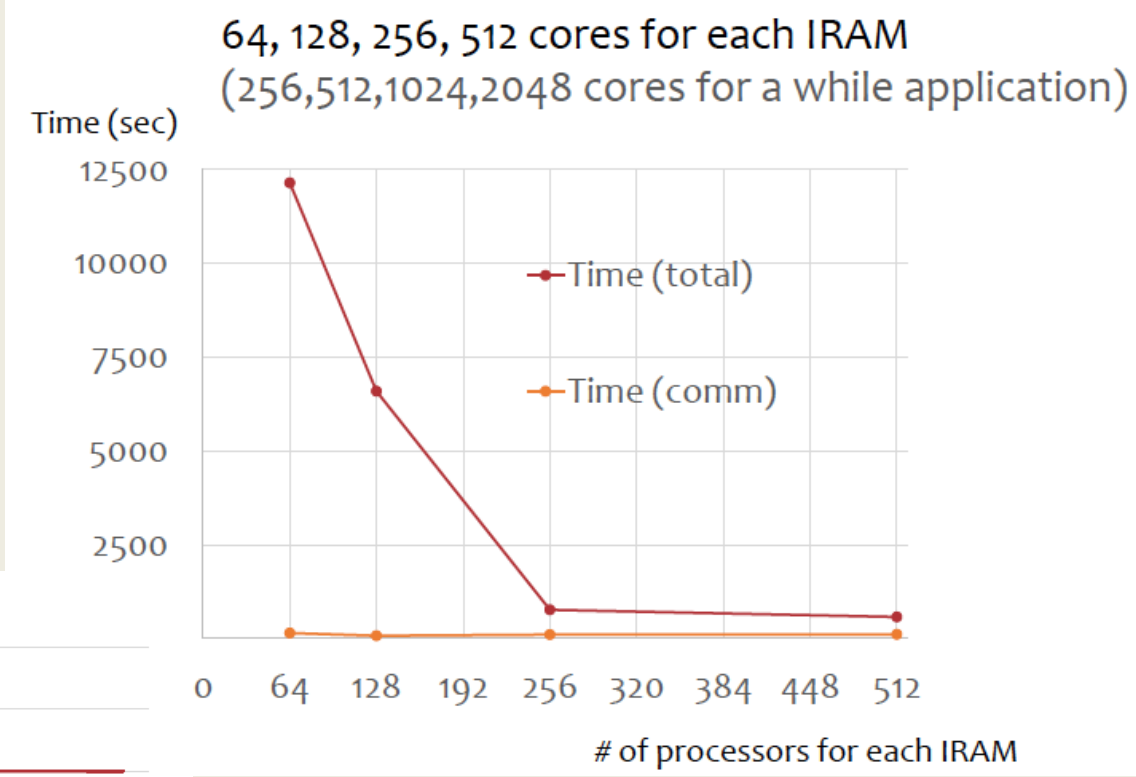
Schenk/nlpkkt240 in the UF Matrix Collection: n=27993600, k=10, tol=10^{-10}.



MIRAM with FP2C (XMP/YML) + PETSc/SLEPc/ARPACK: Experiments

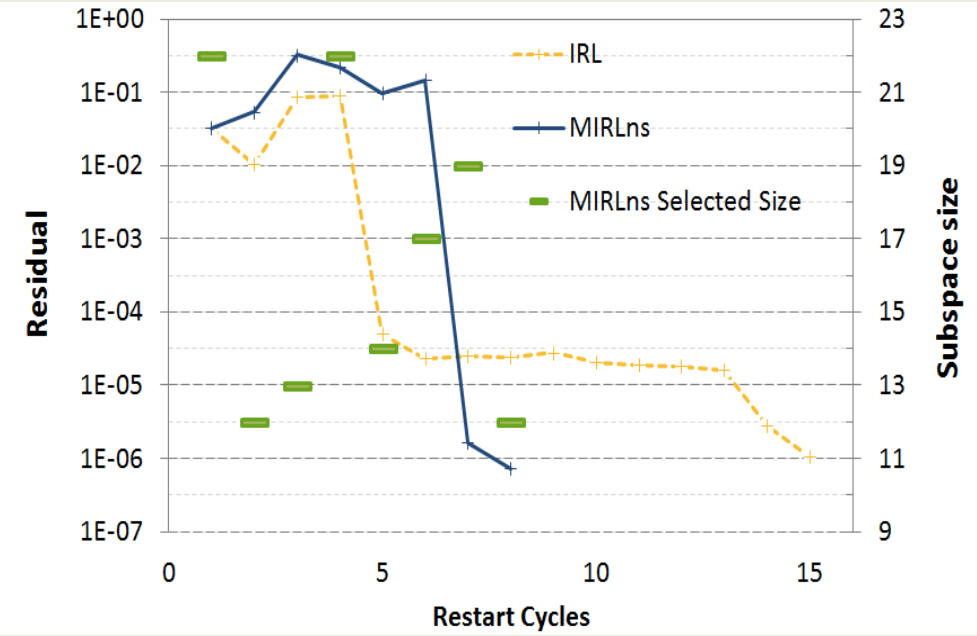
Evolution of the best subspace size along restarting cycles

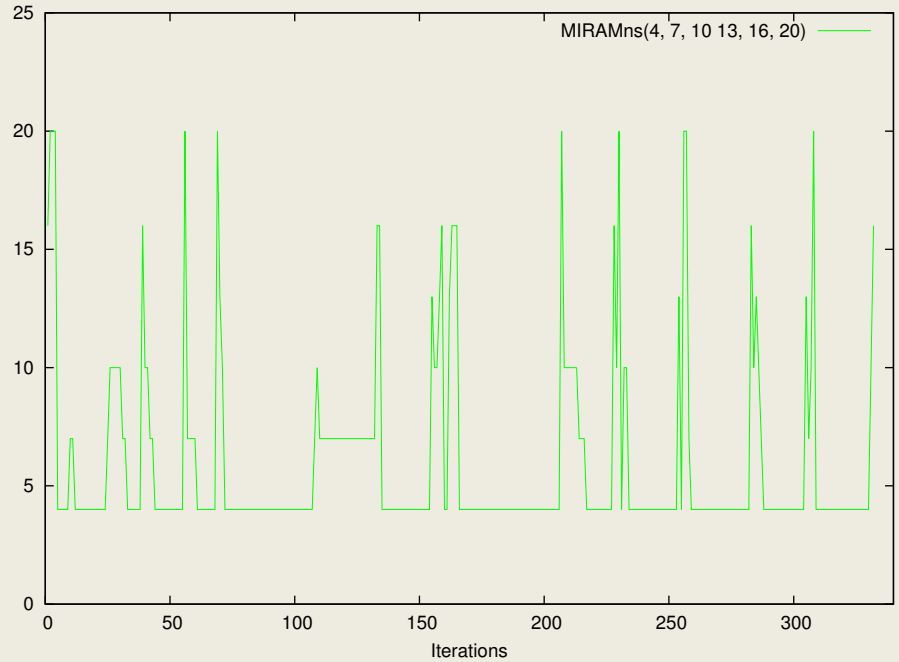




Speed-up execution time and Scalability wrt the number of cores:

EVOLUTION OF m_{BEST} IN AUTOTUNED IRAM





Matrix Tesla K20c, k=7

Evolution of m_{best} along cycles within MIRAM (4,7,10,13,16,20) with roadNet-PA matrix

IA in HPNC: ML to manage the many parameters of UCM?

Search of correlation between the parameters:

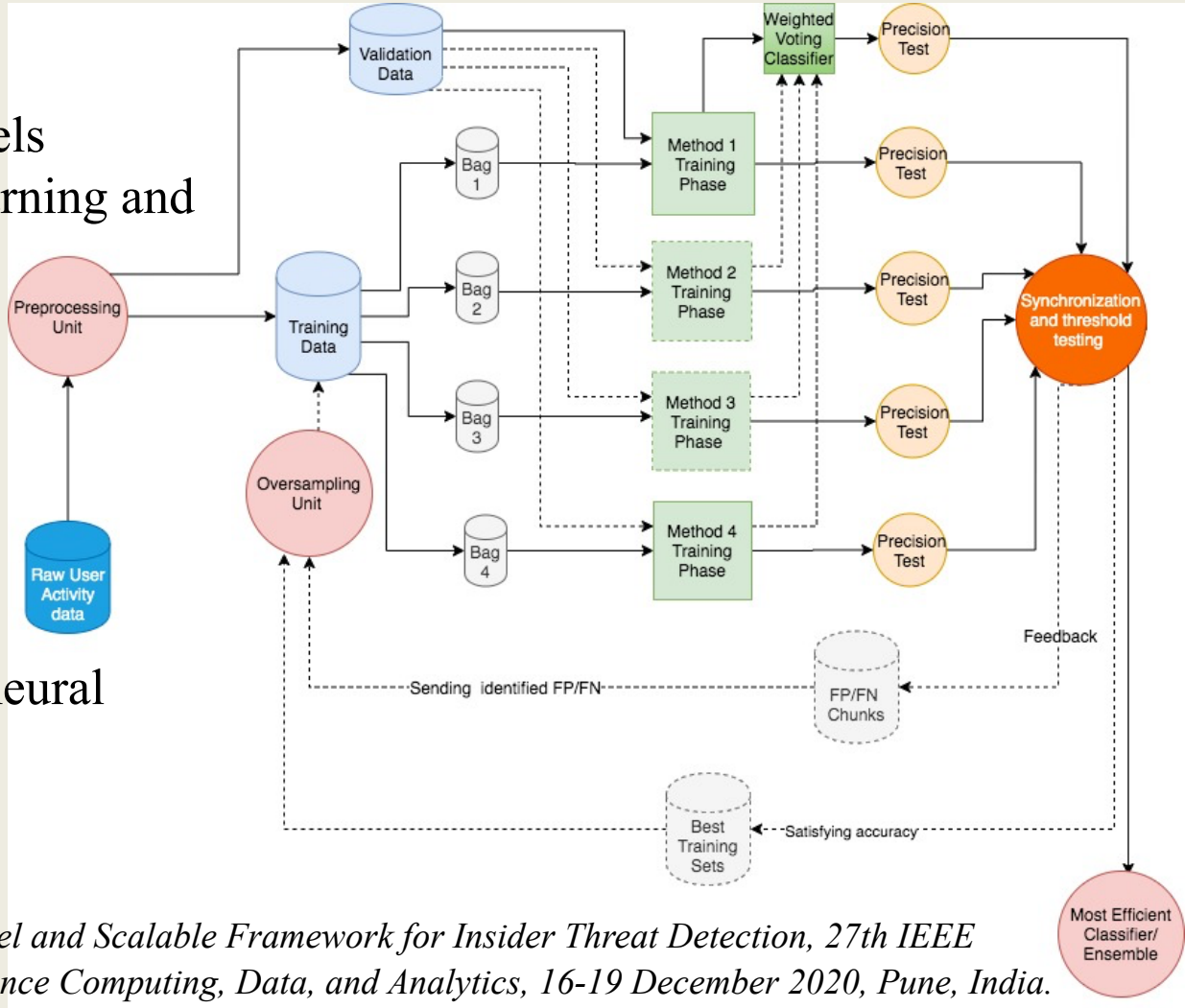
- ✓ k: the number of wanted eigenvalues,
- ✓ $M=[m_1, \ldots, m_l]$: the discrete interval of subspace sizes,
- ✓ $V=[v_1, \ldots, v_l]$: the initial guesses,
- ✓ tolerance: the desired accuracy,
- ✓ *nbRC:* the number of restart cycles,
- ✓ $f^1(\Lambda^1_k, U^1_k), \ldots, f^l(\Lambda^l_k, U^l_k)$: the restarting strategies for the instances,
- ✓ …

By using "big" data generated by so many experiments, ML can help to find fairly precise solution.

HPNC in AI: UC Approach in Machine Learning

UCEL: Unite and Conquer Ensemble Learning methods

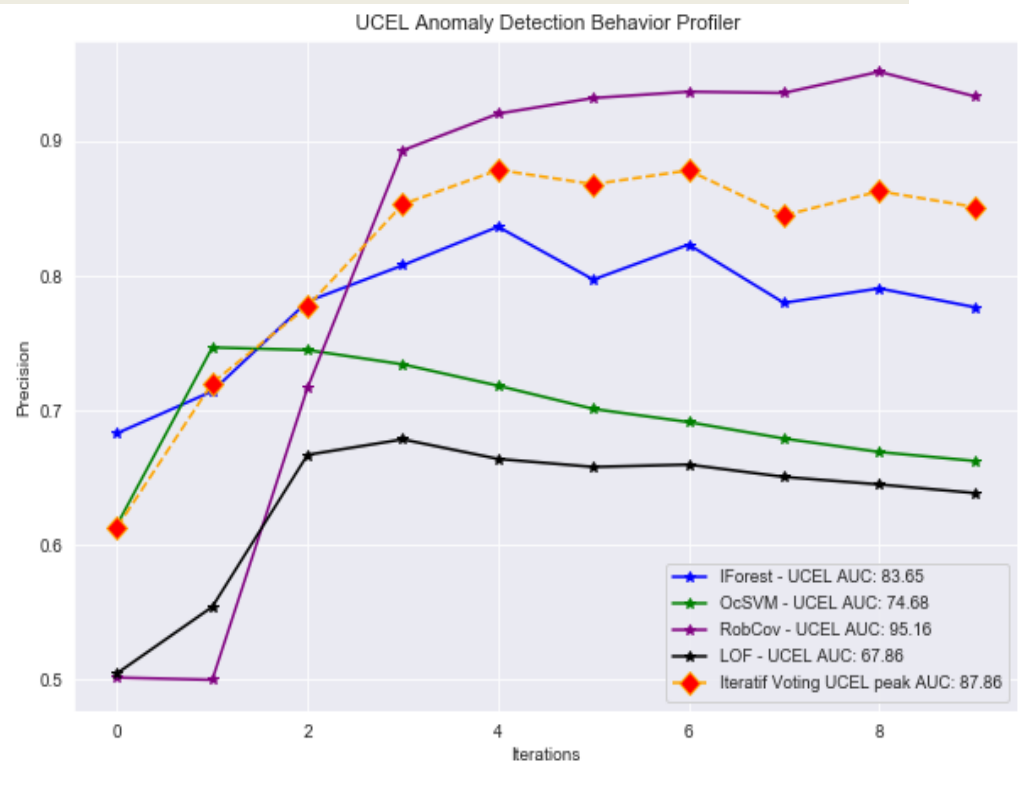
- User behavior classifier models
- Combination of ensemble learning and UC technics (UCEL)
  - -Bagging and Boosting
  - -UC approach
- Base methods family
  - -Anomaly detection
  - -Supervised methods
  - -*Graph based methods
  - -*Unsupervised recurrent neural network

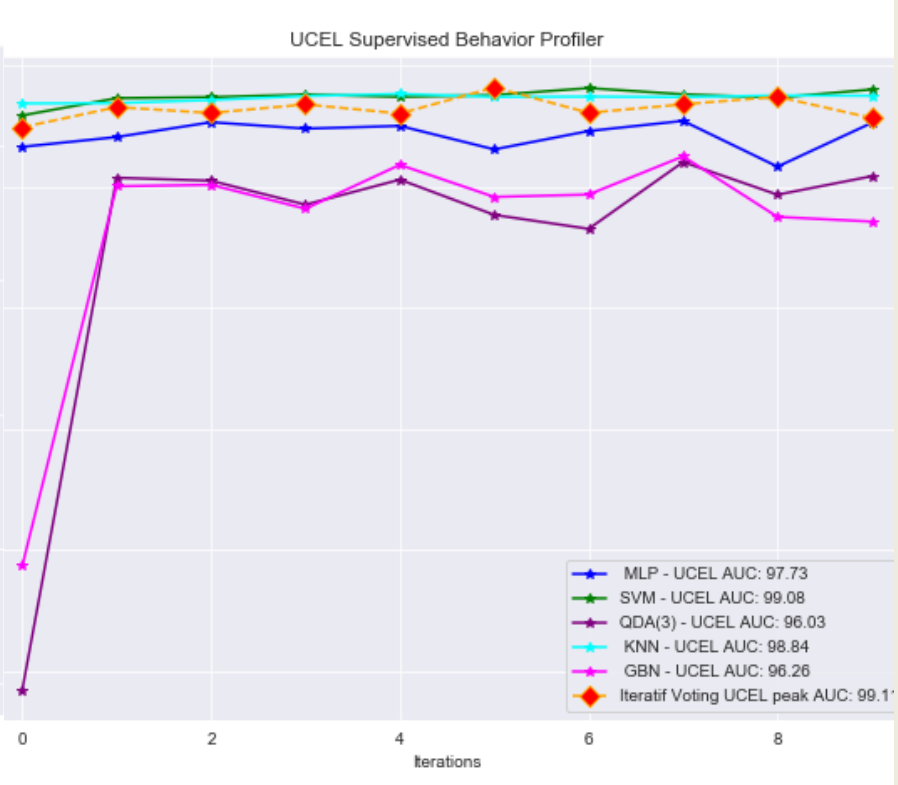


A.M. Diop, N. Emad and T. Winter, A Parallel and Scalable Framework for Insider Threat Detection, 27th IEEE International Conference on High Performance Computing, Data, and Analytics, 16-19 December 2020, Pune, India.

ANOMALY DETECTION WITH DIFFERENT CO-METHODS

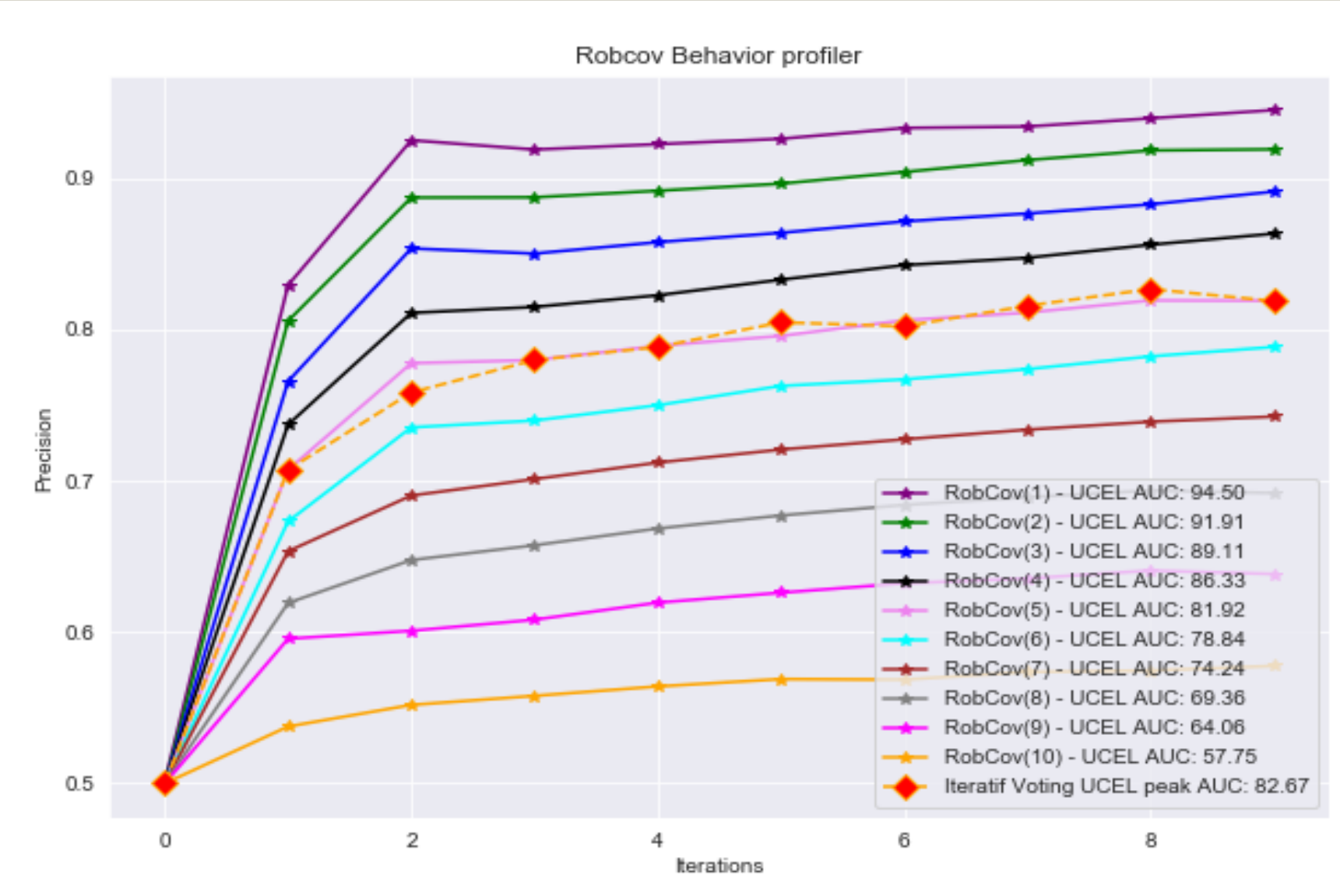
AUC-score evolution with 4 (left) and 5 (right) co-methods





A.M. Diop, N. Emad, T. Winter, A Unite and Conquer Based Ensemble Learning Method for User Behavior Modeling, 39th IEEE International Performance Computing and Communications Conference IPCCC 2020 November 6th – 8th, 2020, Austin, Texas, USA .

Anomaly Detection with Multiple Robcov (10): AUC-score evolution



PARALLEL UCEL ON GRID'5K (8 CO- METHODS RUN ON 9 NODES)