Inria Sophia-Antipolis
Sept. 23, 2015

Some progresses on hybrid solvers toward extreme scale

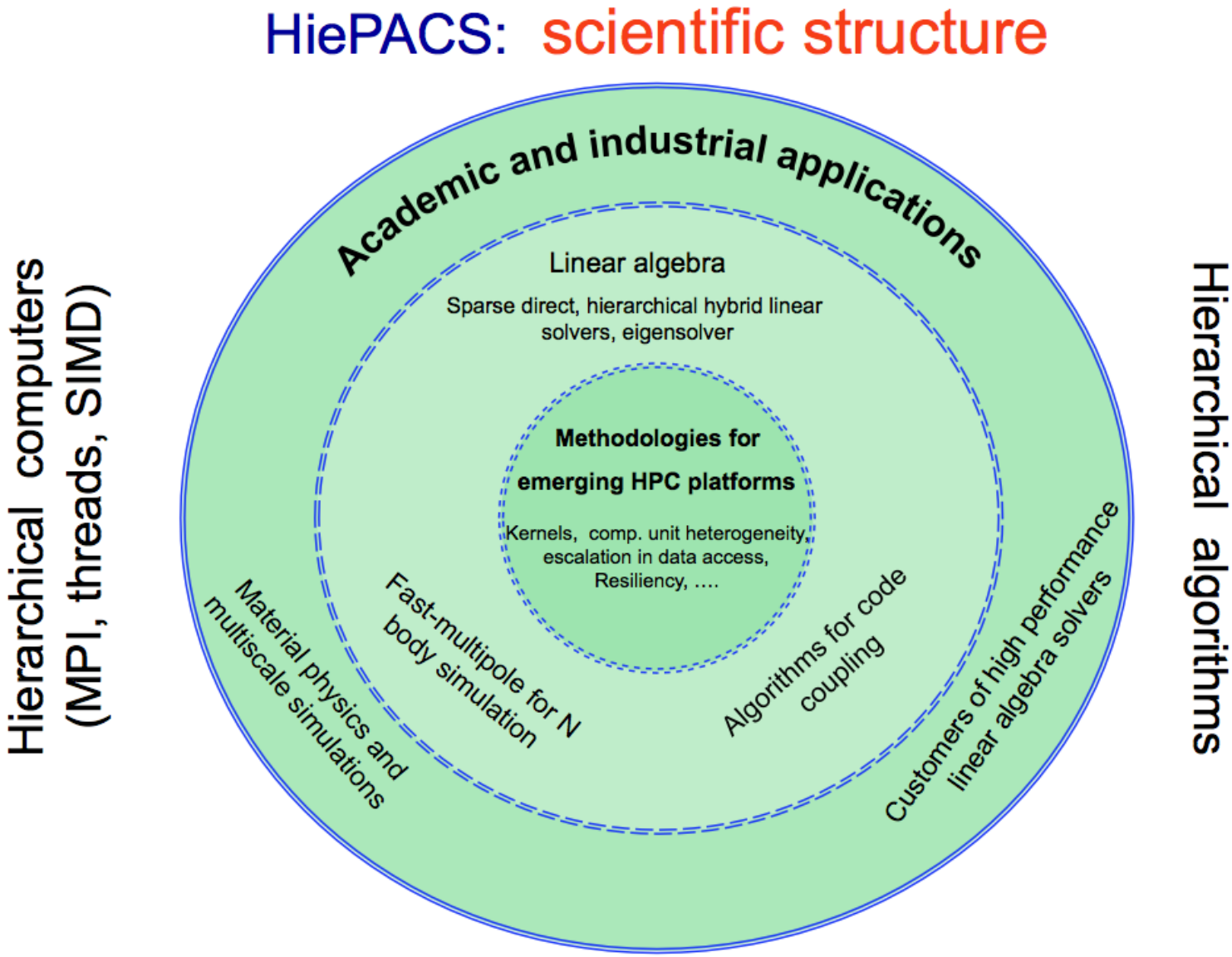
Luc Giraud

joint work with
Inria HiePACS project members

HiePACS Inria Project
INRIA Bordeaux Sud-Ouest
Joint Inria-CERFACS lab

Forewords

HiePACS objectives: Contribute to the design of effective tools for frontier simulations arising from challenging research and industrial multi-scale applications towards extreme computing



Team members

- Permanent Researchers
  - E. Agullo - Inria
  - O. Coulaud - Inria
  - A. Esnard - Bordeaux University
  - M. Faverge - Bordeaux INP
  - L. Giraud - Inria - team leader
  - A. Guermouche - Bordeaux University
  - P. Ramet - Bordeaux University
  - J. Roman - Inria (Bdx INP)
- Post-Doctoral Fellows
  - R. Garnier - Inria-Région Aquitaine
  - Y. Harness - Inria-Région Aquitaine
  - M. Khun - RAPID DGA
  - E. F. Yetkin - G8-ECS/FP7 Exa2CT
- Technical Staff
  - Q. Khan - Carnot funds
  - J. Pédron - Labex CPU
  - C. Piacibello - DGA Hi-Box
  - F. Pruvost - Inria ADT HPC-Collective
- PhD Students
  - P. Blanchard - ENS Cachan
  - B. Bramas - Airbus-Inria- R. Aquitaine
  - A. Casadei - French Ministry
  - J.M. Couteyen - ASTRIUM/ANRT
  - A. Etcheverry - ANR OPTIDIS
  - A. Falco - Airbus-Inria- R. Aquitaine
  - C. Fournier - CERFACS
  - S. Nakov - TOTAL
  - L. Poirel - ANR DEDALES
  - G. Pinchon - DGA/Inria
  - M. Predari - Inria-Région Aquitaine
  - F. Rozar - CEA and MdS
- Research scientist (partners)
  - P. Brenner - Airbus Defence and Space
  - G. Latu - CEA Cadarache
  - G. Sylvand - Airbus Group Innovation

Current collaborations witin Associate Teams: MORSE (UTK, ICL, UCL, Kaust), FASTLA (LNBL, Stanford), IPL C2S@Exa and industrial partners

Sparse linear solver

Goal: solving $\mathcal{A}x = b$, where $\mathcal{A}$ is sparse

Usual trades off

Direct

- Robust/accurate for general problems
- BLAS-3 based implementations
- Memory/CPU prohibitive for large $3D$ problems
- Limited weak scalability

Iterative

- Problem dependent efficiency / accuracy
- Sparse computational kernels
- Less memory requirements and possibly faster
- Possible high weak scalability

Hybrid direct-iterative solver, ours: MaPHyS

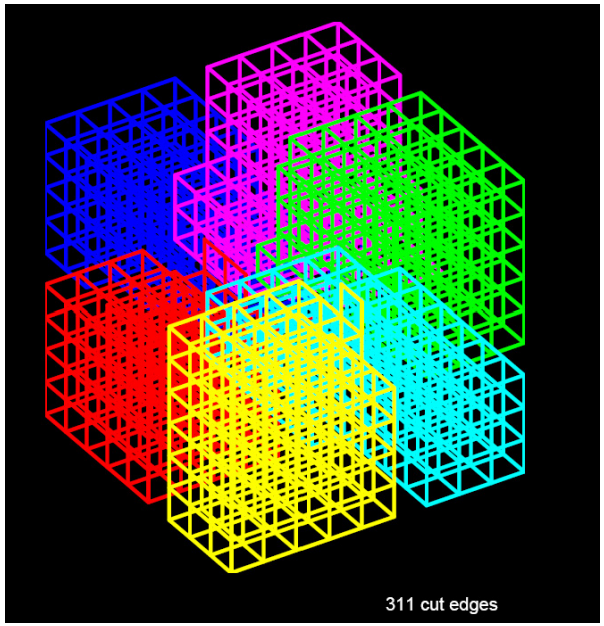
Hybrid Linear Solvers

Develop robust scalable parallel hybrid direct/iterative linear solvers

- Exploit the efficiency and robustness of the sparse direct solvers
- Develop robust parallel preconditioners for iterative solvers
- Take advantage of the natural scalable parallel implementation of iterative solvers

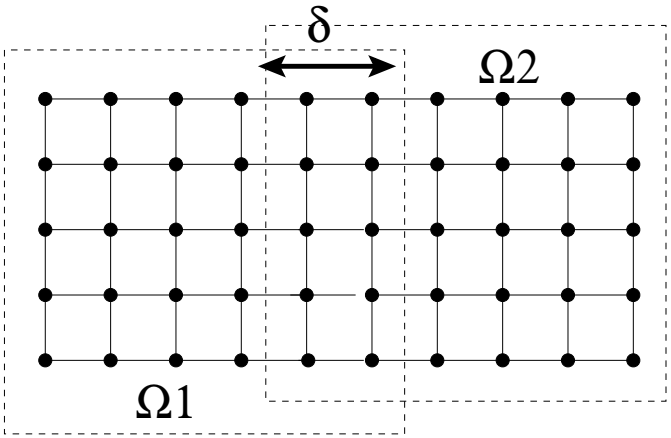
Domain Decomposition (DD)

- Natural approach for PDE's
- Extend to general sparse matrices
- Partition the problem into subdomains, subgraphs
- Use a direct solver on the subdomains
- Robust preconditioned iterative solver



Overlapping Domain Decomposition

Classical Additive Schwarz preconditioners



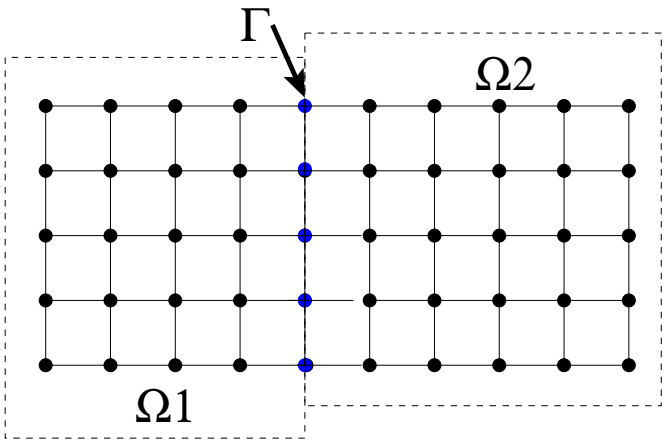
- *Goal:* solve linear system $\mathcal{A}x = b$
- Use iterative method
- Apply the preconditioner at each step
- The convergence rate deteriorates as the number of subdomains increases

$$\mathcal{A} = \begin{pmatrix} \mathcal{A}_{1,1} & \mathcal{A}_{1,\delta} & \\ \mathcal{A}_{\delta,1} & \mathcal{A}_{\delta,\delta} & \mathcal{A}_{\delta,2} \\ & \mathcal{A}_{\delta,2} & \mathcal{A}_{2,2} \end{pmatrix} \Longrightarrow \mathcal{M}^{\delta}_{\mathcal{AS}} = \begin{pmatrix} \boxed{\begin{matrix} \mathcal{A}_{1,1} & \mathcal{A}_{1,\delta} \\ \mathcal{A}_{\delta,1} & \mathcal{A}_{\delta,\delta} \end{matrix}}^{-1} & & \\ & \boxed{\begin{matrix} \mathcal{A}_{\delta,\delta} & \mathcal{A}_{\delta,2} \\ \mathcal{A}_{\delta,2} & \mathcal{A}_{2,2} \end{matrix}}^{-1} \end{pmatrix}$$

Classical Additive Schwarz preconditioners N subdomains case

$$\mathcal{M}^{\delta}_{AS} = \sum_{i=1}^{N} \left(\mathcal{R}^{\delta}_i\right)^T \left(\mathcal{A}^{\delta}_i\right)^{-1} \mathcal{R}^{\delta}_i$$

Non-overlapping Domain Decomposition



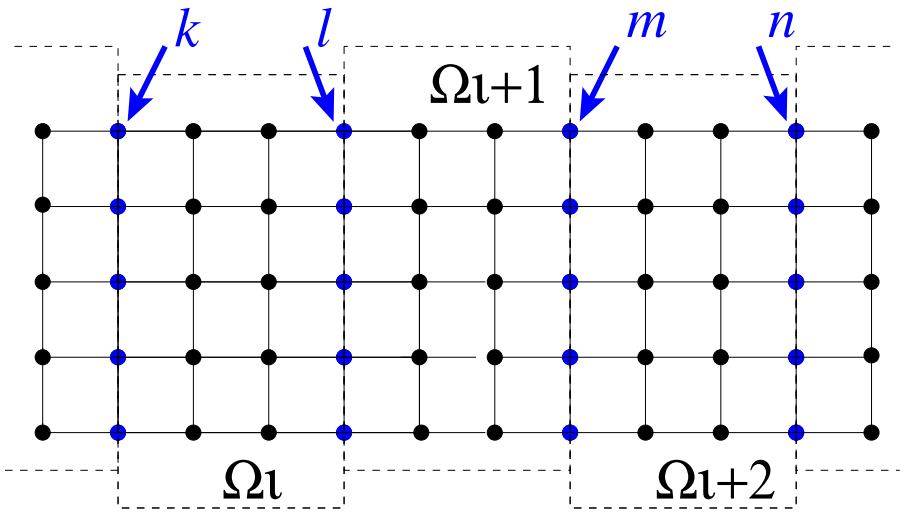
- *Goal:* solve linear system $\mathcal{A}x = b$
- Apply partially Gaussian elimination
- Solve the reduced system $\mathcal{S}x_\Gamma = f$
- Then solve $\mathcal{A}_i x_i = b_i - \mathcal{A}_{i,\Gamma} x_\Gamma$

$$\begin{pmatrix} \mathcal{A}_{1,1} & 0 & \mathcal{A}_{1,\Gamma} \\ 0 & \mathcal{A}_{2,2} & \mathcal{A}_{2,\Gamma} \\ 0 & 0 & \mathcal{S} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_\Gamma \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_\Gamma - \sum_{i=1}^{2} \mathcal{A}_{\Gamma,i}\mathcal{A}_{i,i}^{-1} \quad b_i \end{pmatrix}$$

Solve $\mathcal{A}x = b \Longrightarrow$ solve the reduced system $\mathcal{S}x_\Gamma = f \Longrightarrow$ then solve $\mathcal{A}_i x_i = b_i - \mathcal{A}_{i,\Gamma} x_\Gamma$

where $\mathcal{S} = \mathcal{A}_{\Gamma,\Gamma} - \sum_{i=1}^{2} \mathcal{A}_{\Gamma,i}\mathcal{A}_{i,i}^{-1}\mathcal{A}_{i,\Gamma}$, and $f = b_\Gamma - \sum_{i=1}^{2} \mathcal{A}_{\Gamma,i}\mathcal{A}_{i,i}^{-1} \quad b_i$.

Distributed Schur complement



$$\Gamma = k \cup \ell \cup m \cup n$$

$$\overbrace{\begin{pmatrix} \mathcal{S}_{kk}^{(\iota)} & \mathcal{S}_{k\ell} \\ \mathcal{S}_{\ell k} & \mathcal{S}_{\ell\ell}^{(\iota)} \end{pmatrix}}^{\Omega_\iota} \quad \overbrace{\begin{pmatrix} \mathcal{S}_{\ell\ell}^{(\iota+1)} & \mathcal{S}_{\ell m} \\ \mathcal{S}_{m\ell} & \mathcal{S}_{mm}^{(\iota+1)} \end{pmatrix}}^{\Omega_{\iota+1}} \quad \overbrace{\begin{pmatrix} \mathcal{S}_{mm}^{(\iota+2)} & \mathcal{S}_{mn} \\ \mathcal{S}_{nm} & \mathcal{S}_{nn}^{(\iota+2)} \end{pmatrix}}^{\Omega_{\iota+2}}$$

In an assembled form: $\mathcal{S}_{\ell\ell} = \mathcal{S}_{\ell\ell}^{(\iota)} + \mathcal{S}_{\ell\ell}^{(\iota+1)} \Longrightarrow \mathcal{S}_{\ell\ell} = \sum_{\iota \in adj} \mathcal{S}_{\ell\ell}^{(\iota)}$

Algebraic Additive Schwarz preconditioner

[L.Carvalho, L.G., G.Meurant - 01]

$$\mathcal{S} = \sum_{i=1}^{N} \mathcal{R}_{\Gamma_i}^T \mathcal{S}^{(i)} \mathcal{R}_{\Gamma_i}$$

$$\mathcal{S} = \begin{pmatrix} \ddots & & & & \\ & \mathcal{S}_{kk} & \mathcal{S}_{k\ell} & & \\ & \mathcal{S}_{\ell k} & \mathcal{S}_{\ell\ell} & \mathcal{S}_{\ell m} & \\ & & \mathcal{S}_{m\ell} & \mathcal{S}_{mm} & \mathcal{S}_{mn} \\ & & & \mathcal{S}_{nm} & \mathcal{S}_{nn} \end{pmatrix} \Rightarrow \mathcal{M} = \begin{pmatrix} \ddots & & & & \\ & \boxed{\mathcal{S}_{kk}} & \boxed{\mathcal{S}_{k\ell}} & \boxed{-1} & \\ & \boxed{\mathcal{S}_{\ell k}} & \boxed{\mathcal{S}_{\ell\ell}} & \mathcal{S}_{\ell m} & \boxed{-1} \\ & & \boxed{\mathcal{S}_{m\ell}} & \mathcal{S}_{mm} & \mathcal{S}_{mn} \\ & & & \boxed{\mathcal{S}_{nm}} & \boxed{\mathcal{S}_{nn}} \end{pmatrix}$$

$$\mathcal{M} = \sum_{i=1}^{N} \mathcal{R}_{\Gamma_i}^T (\bar{\mathcal{S}}^{(i)})^{-1} \mathcal{R}_{\Gamma_i}$$

Similarity with Neumann-Neumann preconditioner [J.F Bourgat, R. Glowinski, P. Le Tallec and M. Vidrascu - 89] [Y.H. de Roek, P. Le Tallec and M. Vidrascu - 91]

where $\bar{\mathcal{S}}^{(i)}$ is obtained from $\mathcal{S}^{(i)}$

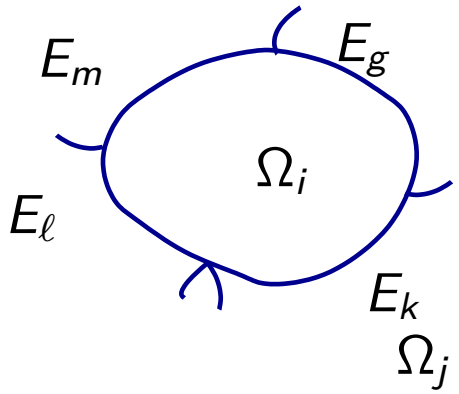
$$\underbrace{\mathcal{S}^{(i)} = \begin{pmatrix} \mathcal{S}_{kk}^{(\iota)} & \mathcal{S}_{k\ell} \\ \mathcal{S}_{\ell k} & \mathcal{S}_{\ell\ell}^{(\iota)} \end{pmatrix}}_{\text{local Schur}} \Rightarrow \underbrace{\bar{\mathcal{S}}^{(i)} = \begin{pmatrix} \mathcal{S}_{kk} & \mathcal{S}_{k\ell} \\ \mathcal{S}_{\ell k} & \mathcal{S}_{\ell\ell} \end{pmatrix}}_{\text{local assembled Schur}}$$

$$\sum_{\iota \in adj} \mathcal{S}_{\ell\ell}^{(\iota)}$$

Parallel preconditioning features

$$S^{(i)} = A^{(i)}_{\Gamma_i\Gamma_i} - A_{\Gamma_i I_i} A^{-1}_{I_i I_i} A_{I_i \Gamma_i}$$

$$M_{AS} = \sum_{i=1}^{\#domains} R_i^T (\bar{S}^{(i)})^{-1} R_i$$



$$\bar{S}^{(i)} = \begin{pmatrix} S_{mm} & S_{mg} & S_{mk} & S_{m\ell} \\ S_{gm} & S_{gg} & S_{gk} & S_{g\ell} \\ S_{km} & S_{kg} & S_{kk} & S_{k\ell} \\ S_{\ell m} & S_{\ell g} & S_{\ell k} & S_{\ell\ell} \end{pmatrix} \qquad S^{(i)} = \begin{pmatrix} S^{(i)}_{mm} & S_{mg} & S_{mk} & S_{m\ell} \\ S_{gm} & S^{(i)}_{gg} & S_{gk} & S_{g\ell} \\ S_{km} & S_{kg} & S^{(i)}_{kk} & S_{k\ell} \\ S_{\ell m} & S_{\ell g} & S_{\ell k} & S^{(i)}_{\ell\ell} \end{pmatrix}$$

Assembled local Schur complement

local Schur complement

$$S_{mm} = \sum_{j \in adj(m)} S^{(j)}_{mm}$$

Parallel implementation

- Each *subdomain* $\mathcal{A}^{(i)}$ is handled by one *processor*

$$\mathcal{A}^{(i)} \equiv \begin{pmatrix} \mathcal{A}_{\mathcal{I}_i\mathcal{I}_i} & \mathcal{A}_{\mathcal{I}_i\Gamma_i} \\ \mathcal{A}_{\mathcal{I}_i\Gamma_i} & \mathcal{A}^{(i)}_{\Gamma\Gamma} \end{pmatrix}$$

- Concurrent partial factorizations are performed on each processor to form the so called "local Schur complement"

$$\mathcal{S}^{(i)} = \mathcal{A}^{(i)}_{\Gamma\Gamma} - \mathcal{A}_{\Gamma_i\mathcal{I}_i}\mathcal{A}^{-1}_{\mathcal{I}_i\mathcal{I}_i}\mathcal{A}_{\mathcal{I}_i\Gamma_i}$$

- The reduced system $\mathcal{S}x_\Gamma = f$ is solved using a distributed Krylov solver
  - One matrix vector product per iteration each processor computes $\mathcal{S}^{(i)}(x^{(i)}_\Gamma)^k = (y^{(i)})^k$
  - One local preconditioner apply $(\mathcal{M}^{(i)})(z^{(i)})^k = (r^{(i)})^k$
  - Local neighbor-neighbor communication per iteration
  - Global reduction (dot products)
- Compute simultaneously the solution for the interior unknowns

$$\mathcal{A}_{\mathcal{I}_i\mathcal{I}_i}x_{\mathcal{I}_i} = b_{\mathcal{I}_i} - \mathcal{A}_{\mathcal{I}_i\Gamma_i}x_{\Gamma_i}$$

What tricks exist to construct cheaper preconditioners

- Sparsification strategy through dropping

$$\widehat{s}_{k\ell} = \begin{cases} \bar{s}_{k\ell} & \text{if} \quad \bar{s}_{k\ell} \geq \xi(|\bar{s}_{kk}| + |\bar{s}_{\ell\ell}|) \\ 0 & \text{else} \end{cases}$$

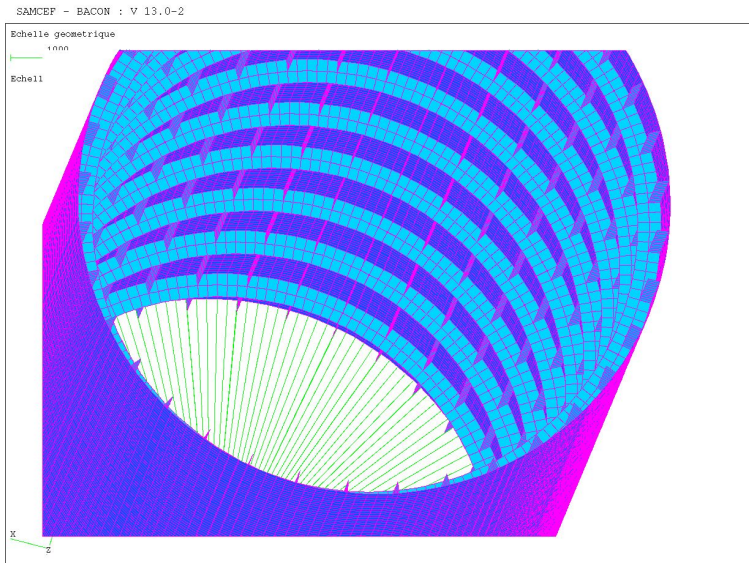
- Approximation through ILU - [A. Haidar, L.G., Y.Saad - 10]

$$pILU\left(A^{(i)}\right) \equiv pILU \begin{pmatrix} A_{ii} & A_{i\Gamma_i} \\ A_{\Gamma_i i} & A^{(i)}_{\Gamma_i\Gamma_i} \end{pmatrix} \equiv \begin{pmatrix} \tilde{L}_i & 0 \\ A_{\Gamma i}\tilde{U}_i^{-1} & I \end{pmatrix} \begin{pmatrix} \tilde{U}_i & \tilde{L}_i^{-1}A_{i\Gamma} \\ 0 & \tilde{S}^{(i)} \end{pmatrix}$$

- Mixed arithmetic strategy
  - Compute and store the preconditioner in 32-bit precision arithmetic Is accurate enough?
  - Remarks: the backward stability result of GMRES indicates that it is hopeless to expect convergence at a backward error level smaller than the 32-bit accuracy [C.Paige, M.Rozložník, Z.Strakoš - 06]
  - Idea: To overcome this limitation we use FGMRES [Y.Saad - 93]
- Data sparse preconditioner use $\mathcal{H}$-matrix idea to form and apply a data sparse preconditioner.

Indefinite systems in structural mechanics S.Pralet, SAMTECH

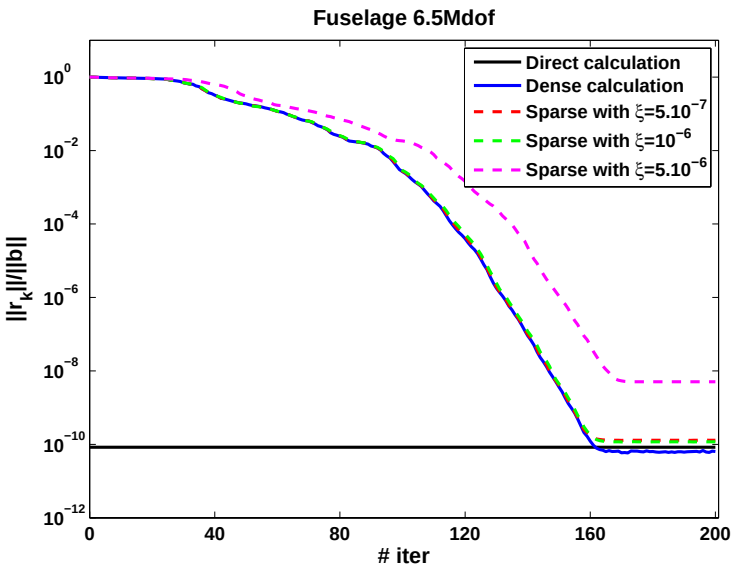
Fuselage of 6.5 Mdof



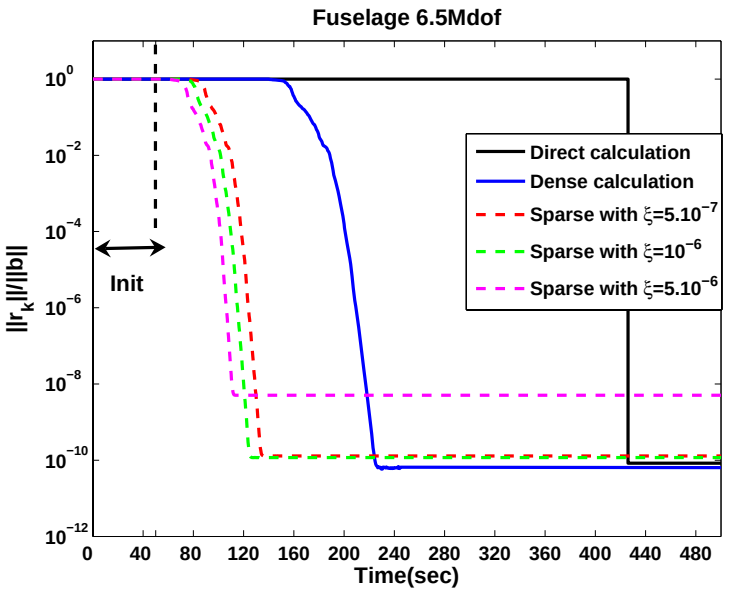
- Linear elastricity
- Composed of its skin, stringers and frames
- Midlinn shell elements are used
- Each node has 6 unknowns
- One extremity is fixed
- On the other extremity a rigid body element is added
- A force perpendicular to the axis is applied

Numerical behaviour of sparse preconditioners

Convergence history



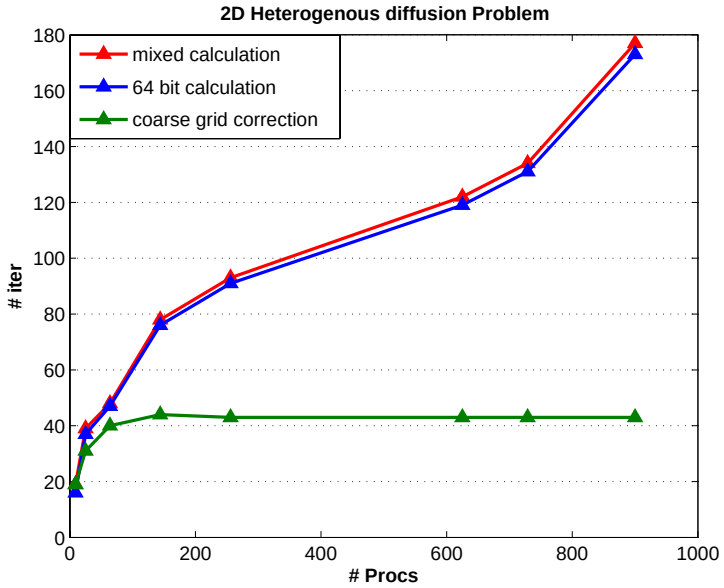
Time history



- Fuselage problem of 6.5 Mdof dof mapped on 16 processors
- The sparse preconditioner setup is 4 times faster than the dense one (19.5 v.s. 89 seconds)
- In term of global computing time, the sparse algorithm is about twice faster
- The attainable accuracy of the hybrid solver is comparable to the one computed with the direct solver

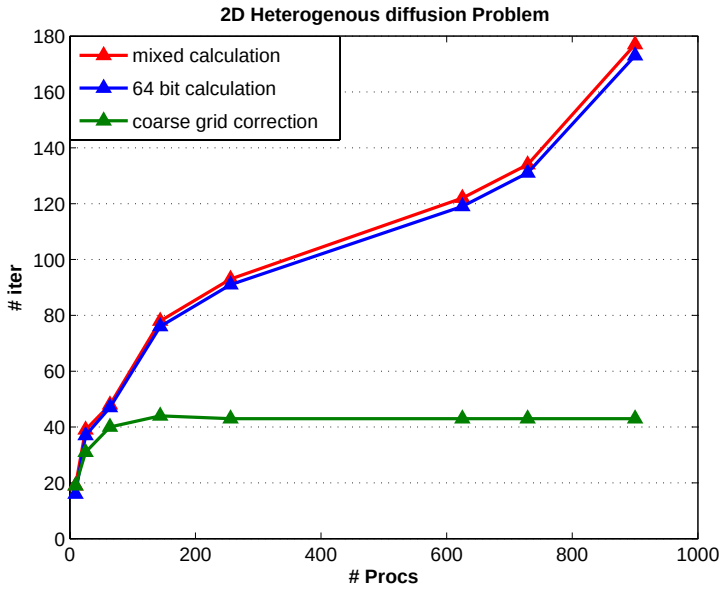
Scalability bottleneck: numerical limitations

2*D* Heterogenous diffusion

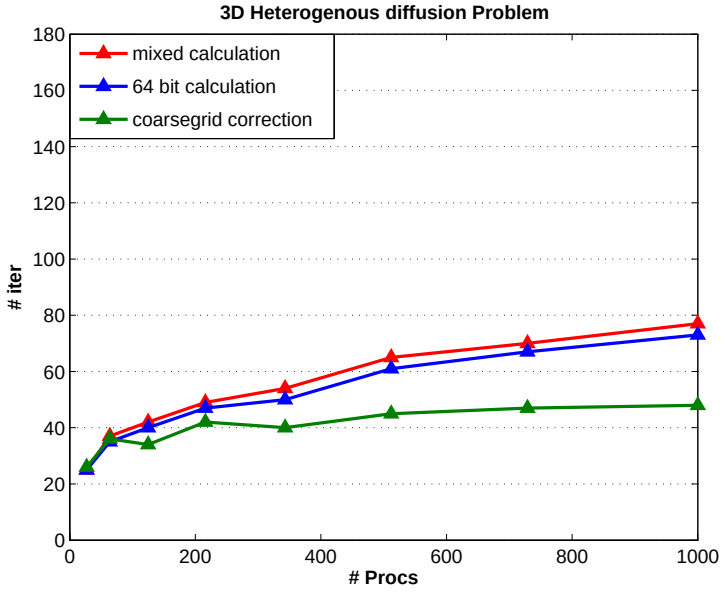


Scalability bottleneck: numerical limitations

2*D* Heterogenous diffusion

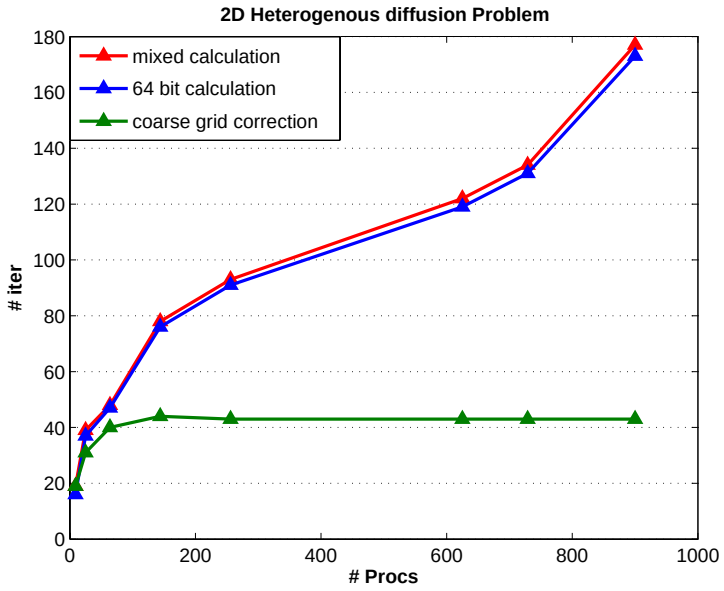


3*D* Heterogenous diffusion

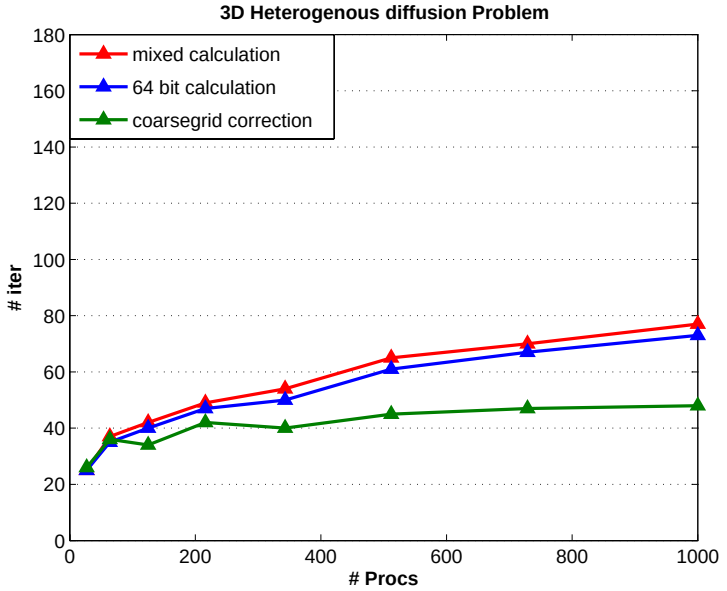


Scalability bottleneck: numerical limitations

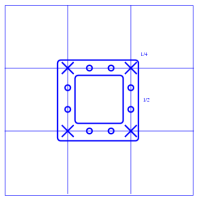
2D Heterogenous diffusion



3D Heterogenous diffusion



Domain based coarse space : $M = M_{AS} + R_O^T A_O^{-1} R_0$ where $A_0 = R_0 S R_O^T$



- "As many" dof in the coarse space as sub-domains
  [Carvalho, Giraud, Le Tallec, SISC, 01]
- Partition of unity : R_0^T simplest constant interpolation

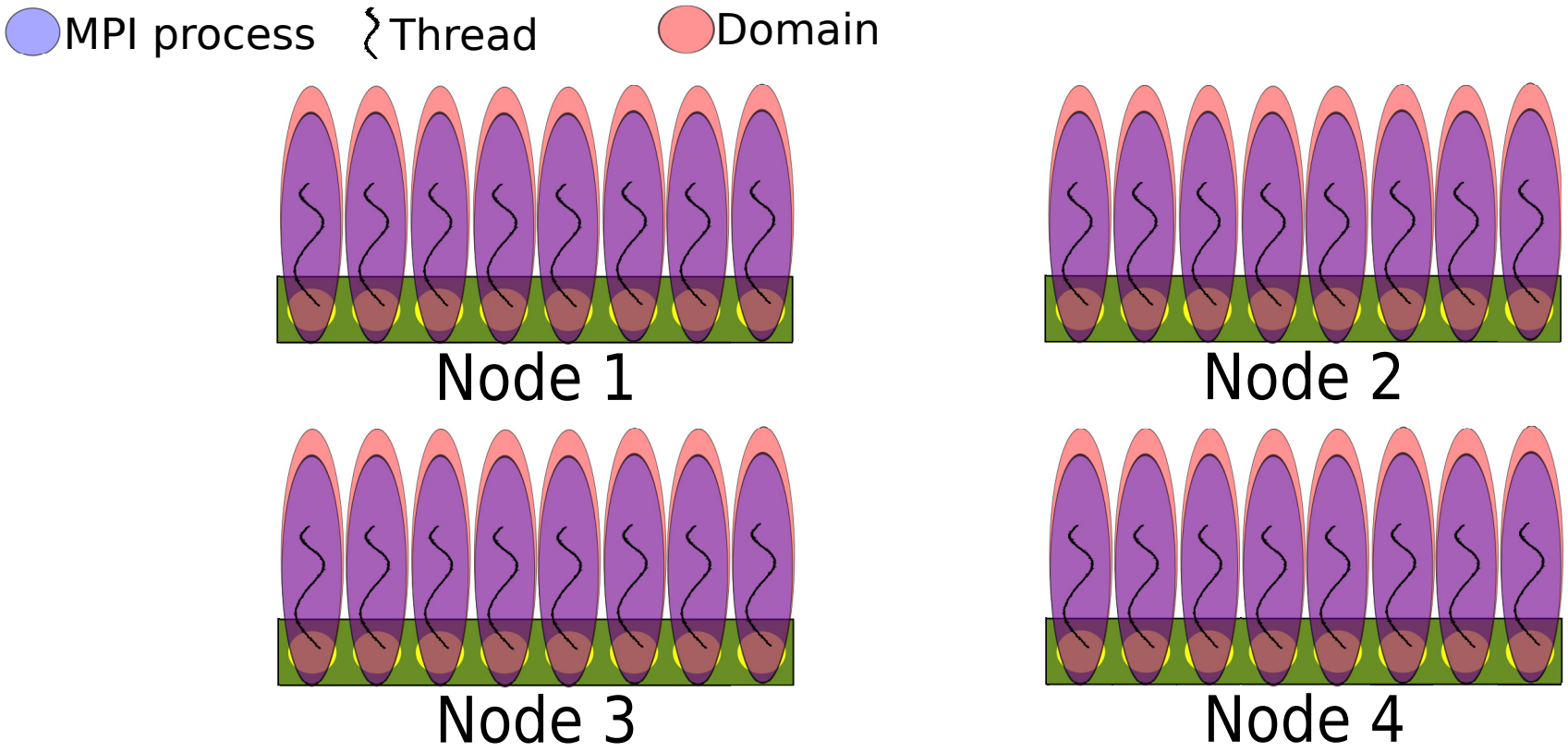
Ongoing PhD thesis: Louis Poirel

A computer science remedy: *2-levels* of parallelism

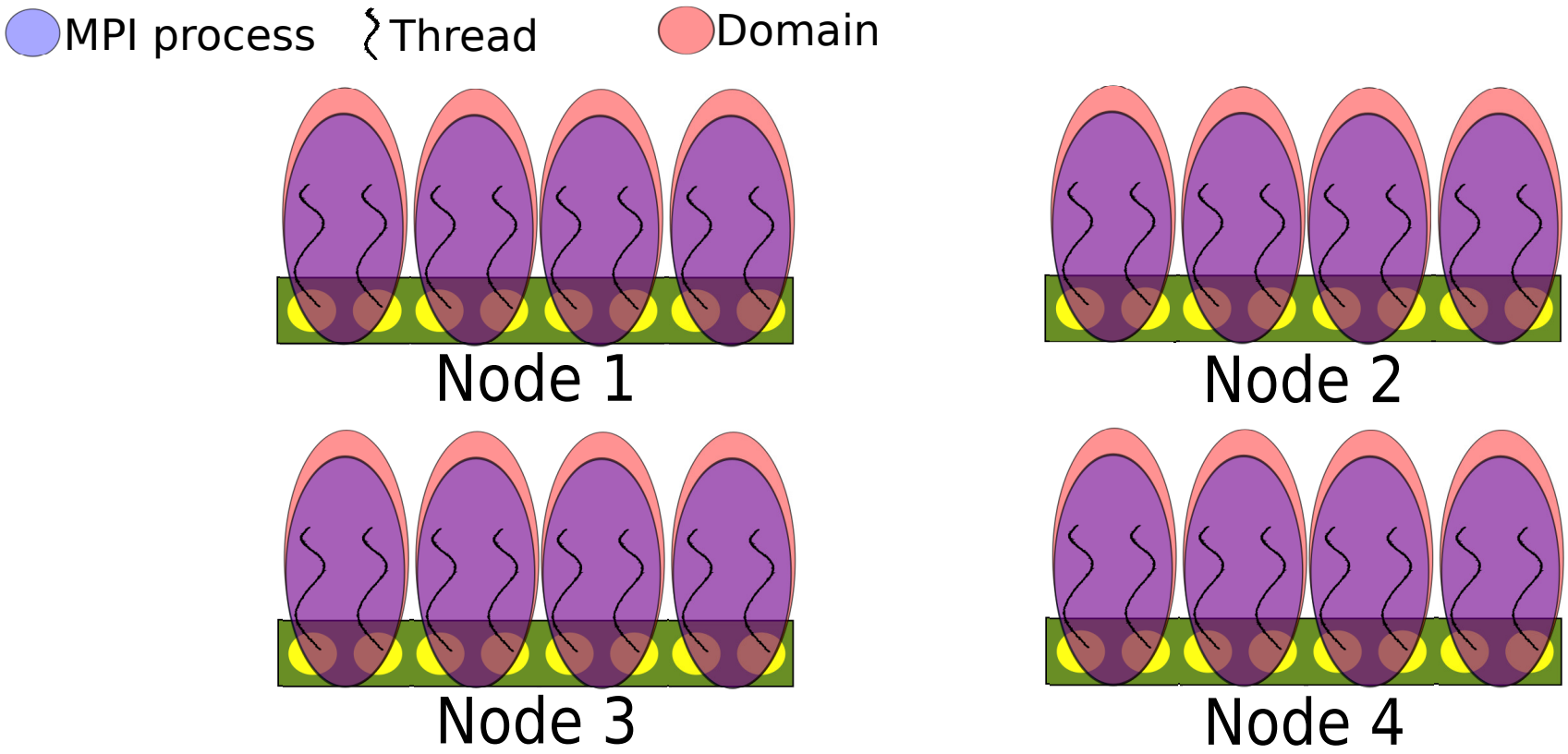
- "*The numerical improvement*"
  - Classical parallel implementations (*1-level*) of DD assign one subdomain per processor
  - Parallelizing means increasing the number of subdomains, often leads to increasing the number of iterations
  - To avoid this, one can instead of increasing the number of subdomains, keeping it small while handling each subdomain by more than one core introducing *2-levels* of parallelism
- "*The parallel performance improvement*"
  - Large $3D$ systems often require a huge amount of data storage
  - On SMP node: classical *1-level parallel* can only use a subset of the available cores
  - The "idle" cores might contribute to the computation and the simulation runs closer to the peak of per-node performance by using *2-levels* of parallelism

Finishing PhD thesis: Stojce Nakov

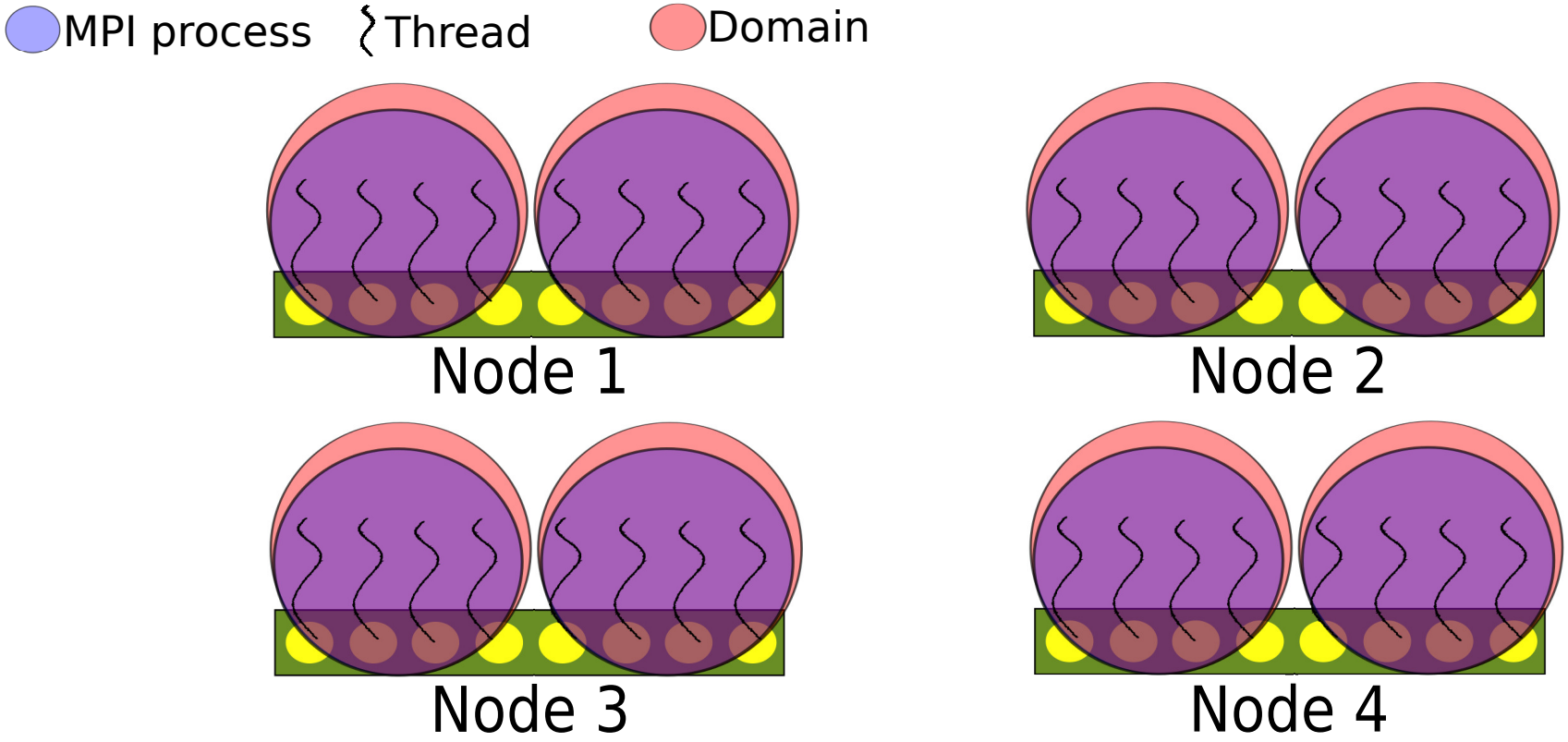
Flexibility to exploit entire multicore nodes



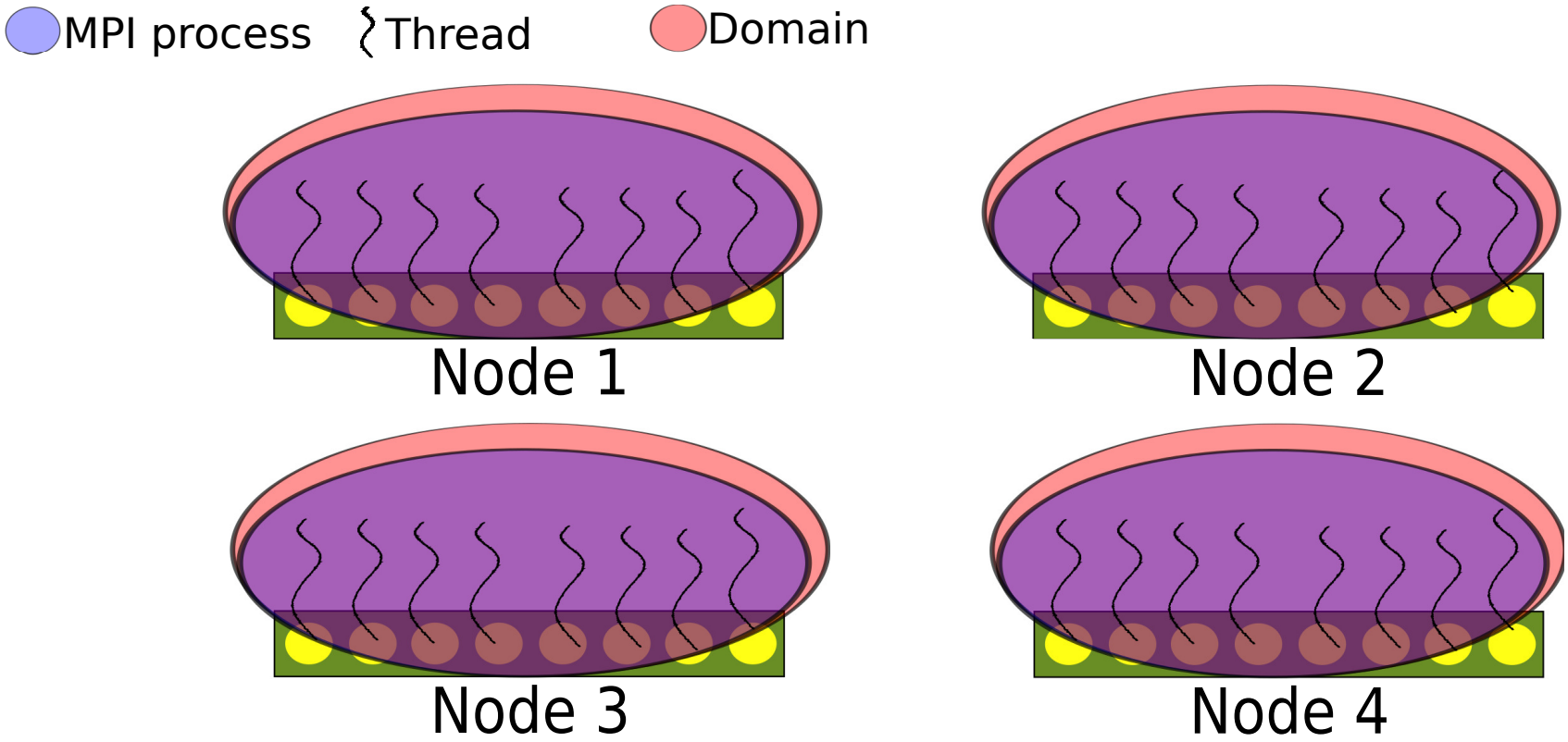
Flexibility to exploit entire multicore nodes



Flexibility to exploit entire multicore nodes



Flexibility to exploit entire multicore nodes



Software used in MaPHyS (this study)

Dense direct solver

- Multi-threaded Mkl library

Sparse direct solvers

- Mumps
- Multi-threaded PaStiX

Iterative Solvers

- Cg/Gmres/FGmres using multi-threaded Mkl library

Software used in MaPHyS (this study)

Dense direct solver

- Multi-threaded Mkl library

Sparse direct solvers

- Mumps
- Multi-threaded PaStiX

Iterative Solvers

- Cg/Gmres/FGmres using multi-threaded Mkl library

- Challenge
  - Composability
  - Performance

Experimental set up

Hopper - LBNL platform

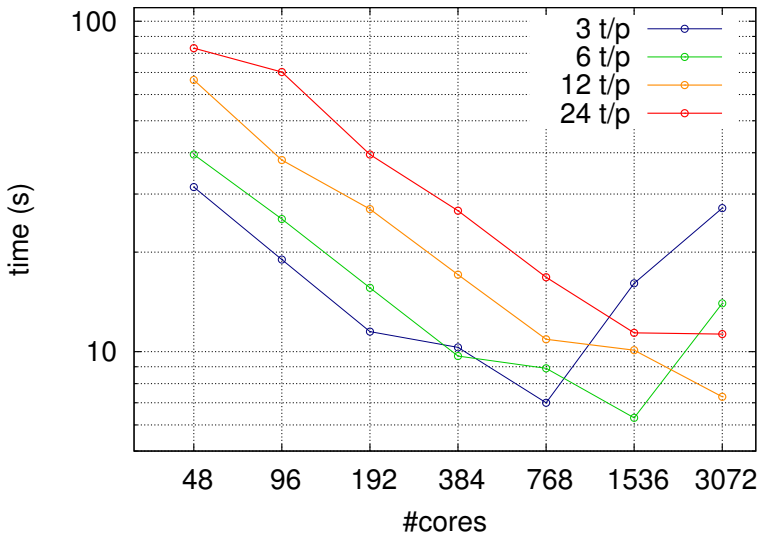
- Two twelve-core AMD 'MagnyCours' 2.1-GHz
- Memory: 32 GB GDDR3
- Double precision

Matrices

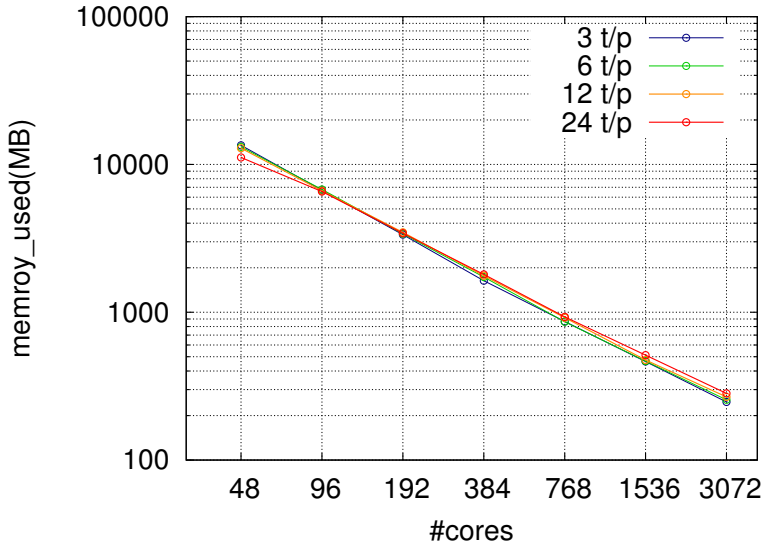
Matrix	Matrix211	Nachos4M
n	801K	4,147K
nnz	129,4M	256,4M
Preconditioner	dense	sparse02

Matrix211 matrix on the Hopper platform

All computation steps

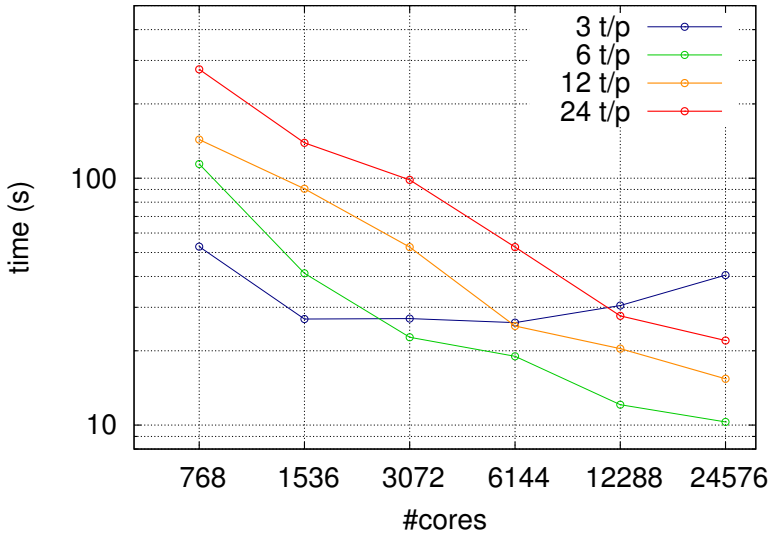


Memory per node

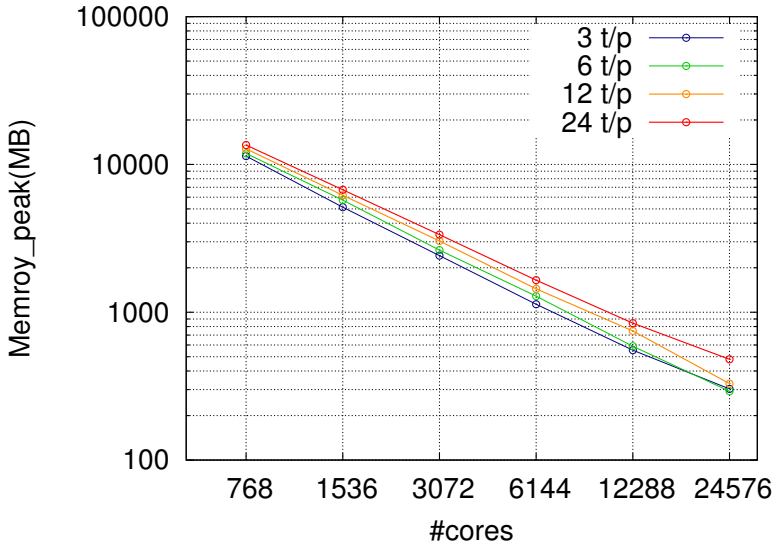


Nachos4M matrix on the Hopper platform

All computation steps



Memory per node



Preliminary comparison with PDSLin (LBNL)

PDSLin a in a nutshell

$$\left(\begin{array}{cccc|c} \mathcal{A}_{\mathcal{I}_1\mathcal{I}_1} & & & & \mathcal{A}_{\mathcal{I}_1\Gamma} \\ & \mathcal{A}_{\mathcal{I}_2\mathcal{I}_2} & & & \mathcal{A}_{\mathcal{I}_1\Gamma} \\ & & \ddots & & \vdots \\ & & & \mathcal{A}_{\mathcal{I}_\mathcal{N}\mathcal{I}_\mathcal{N}} & \mathcal{A}_{\mathcal{I}_\mathcal{N}\Gamma} \\ \hline \mathcal{A}_{\Gamma\mathcal{I}_1} & \mathcal{A}_{\Gamma\mathcal{I}_2} & \dots & \mathcal{A}_{\Gamma\mathcal{I}_\mathcal{N}} & \mathcal{A}_{\Gamma\Gamma} \end{array}\right).$$

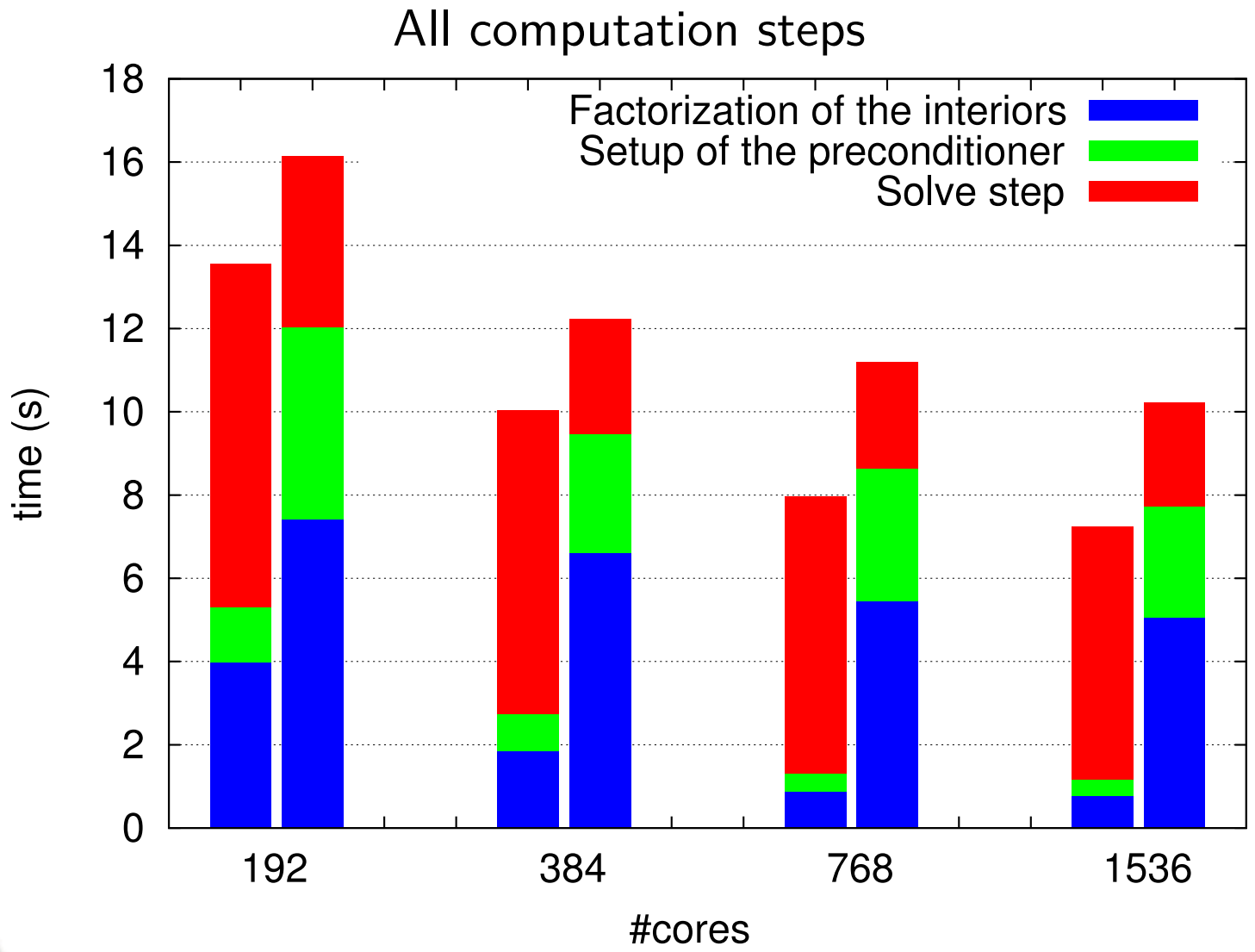
$$\begin{aligned} \mathcal{S} &= \mathcal{A}_{\Gamma\Gamma} - \sum_{\ell=1}^{\mathcal{N}} \mathcal{A}_{\Gamma\mathcal{I}_\ell} \mathcal{A}_{\mathcal{I}_\ell\mathcal{I}_\ell}^{-1} \mathcal{A}_{\mathcal{I}_\ell\Gamma} \\ &= \mathcal{A}_{\Gamma\Gamma} - \sum_{\ell=1}^{\mathcal{N}} (U_\ell^{-T} \mathcal{A}_{\Gamma\mathcal{I}_\ell}^T)^T (L_\ell^{-1} \mathcal{A}_{\mathcal{I}_\ell\Gamma}) = \mathcal{A}_{\Gamma\Gamma} - \sum_{\ell=1}^{\mathcal{N}} W_\ell G_\ell, \end{aligned}$$

$\tilde{T}_\ell = \tilde{G}_\ell \tilde{W}_\ell$ leading to $\hat{\mathcal{S}} = \mathcal{A}_{\Gamma\Gamma} - \sum_{\ell=1}^{\mathcal{N}} \tilde{T}_\ell$

Preliminary comparison with PDSLin (LBNL)

1. Concurrent parallel SuperLU_DIST instances $\mathcal{A}_{\mathcal{I}_\ell \mathcal{I}_\ell} = L_\ell U_\ell$,
2. Sparsification of W_ℓ and G_ℓ, to form $\tilde{W}_\ell$ and $\tilde{G}_\ell$
3. Parallel computation of $\hat{\mathcal{S}} = \mathcal{A}_{\Gamma\Gamma} - \sum_{\ell=1}^{\mathcal{N}} \tilde{T}_\ell$ and its sparsification
4. Parallel factorization $\tilde{\mathcal{S}}$ using an additional instance of SuperLU_DIST
5. Parallel iterative solution using a Krylov subspace method, GMRES for our experiments

Matrix211 on the Hopper platform



Concluding remarks

- Related ongoing efforts
  1. Partioning/balancing both interface and interior vertices (A. Cassadei - PhD Defence: Oct. 19)
  2. Parallel analysis and FEM API (M. Khun)
  3. Deflation/augmentation via local spectral calculation (L. Poirel)
  4. $\mathcal{H}$-arithmetic for local solve ($\mathcal{H}$-PasTiX) and preconditioner (A. Falco, G. Pichon, Y. Harness)
  5. Numerical resilience policies (M. Zounon)
- Future step: Task based implemenation on top of runtime systems (S. Nakov - PhD defence: Dec. 14)

Merci for your attention

Questions ?

https://team.inria.fr/hiepacs/