MAPHYS or the development of a parallel algebraic domain decomposition solver in the course of the Solstice project

Emmanuel AGULLO, Luc GIRAUD, Abdou GUERMOUCHE,
Azzam HAIDAR, Yohan LI-TIN-YIEN, Jean ROMAN

HiePACS project - INRIA Bordeaux Sud-Ouest
joint INRIA-CERFACS lab. on High Performance Computing

CERFACS Sparse Days

Toulouse, June 2010

Outline

Motivations

$$\mathcal{A}x = b$$

The "spectrum" of linear algebra solvers

Direct

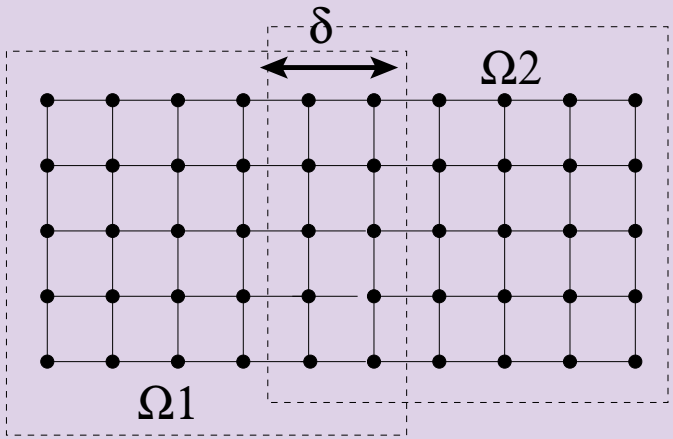
- Robust/accurate for general problems
- BLAS-3 based implementations
- Memory/CPU prohibitive for large $3D$ problems
- Limited parallel scalability

Iterative

- Problem dependent efficiency/controlled accuracy
- Only mat-vect required, fine grain computation
- Less memory computation, possible trade-off with CPU
- Attractive "build-in" parallel features

Overlapping Domain Decomposition

Classical Additive Schwarz preconditioners



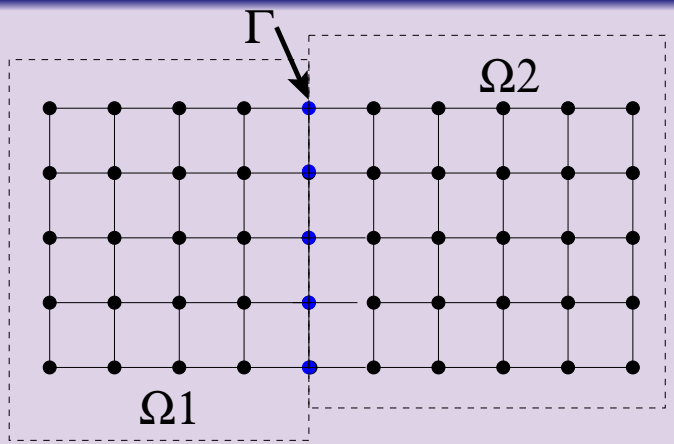
- *Goal:* solve linear system $\mathcal{A}x = b$
- Use iterative method
- Apply the preconditioner at each step
- The convergence rate deteriorates as the number of subdomains increases

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

Classical Additive Schwarz preconditioners N subdomains case

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

Schur complement reduced system



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

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

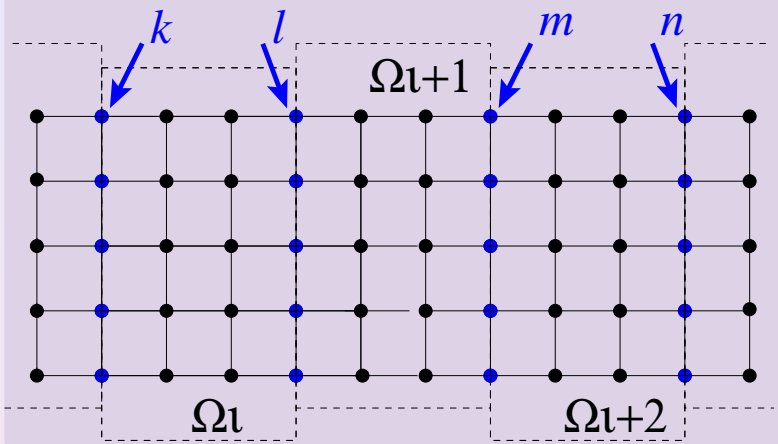
Solve $\mathcal{A}x = b \Longrightarrow$ solve the reduced system $\mathcal{S}x_\Gamma = f \Longrightarrow$ then solve $\mathcal{A}_i x_i = b_i - \mathcal{A}_{i,\Gamma} x_\Gamma$

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

Nonoverlapping Domain Decomposition

Schur complement reduced system



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

Distributed Schur complement

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

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]

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

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

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

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

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

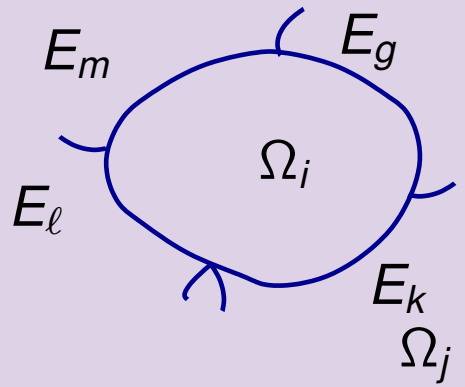
$$\bar{\mathcal{S}}^{(i)} = \mathcal{S}^{(i)} + \sum_{\iota \in adj} \mathcal{S}_{\ell\ell}^{(\iota)}$$

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

Parallel preconditioning features

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

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



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

Assembled local Schur complement

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

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

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

Algebraic Additive Schwarz preconditioner

Main characteristics in $2D$

- The ratio interface/interior is small
- Does not require large amount of memory to store the preconditioner
- Computation/application of the preconditioner are fast
- They consist in a call to LAPACK/BLAS-2 kernels

Main characteristics in $3D$

- The ratio interface/interior is large
- The storage of the preconditioner might not be affordable
- The construction of the preconditioner can be computationally expensive
- Need cheaper Algebraic Additive Schwarz form of the preconditioner

How to alleviate the preconditioner construction

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 - [INRIA PhyLeas - A. Haidar, L.G., Y.Saad - 10]

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

Mixed arithmetic strategy

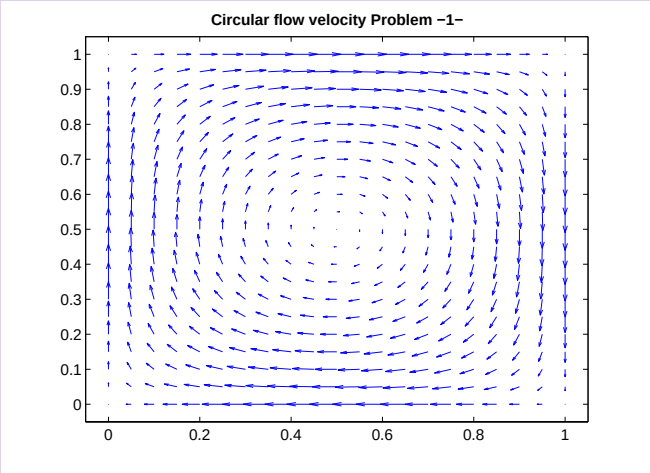
- Compute and store the preconditioner in 32-bit precision arithmetic
- Remarks: the backward stability result of GMRES indicates that it is hopeless to expect convergence at a backward error level smaller than the 32-bit accuracy [C.Paige, M.Rozložník, Z.Strakoš - 06]
- Idea: To overcome this limitation we use FGMRES [Y.Saad - 93; Arioli, Duff - 09]

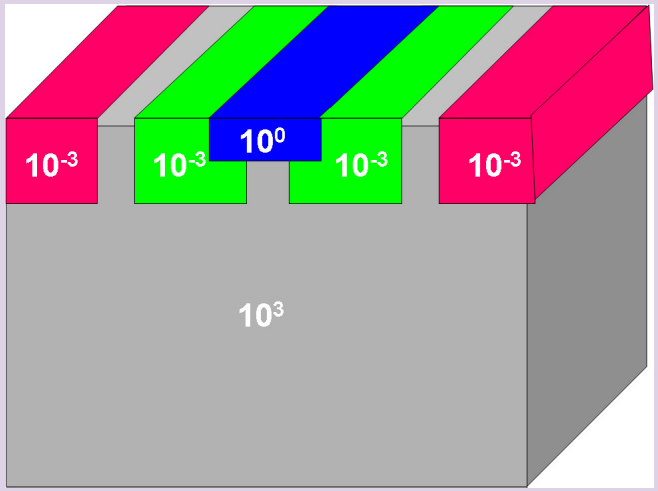
Exploit two levels of parallelism

Use a parallel sparse direct solver on each sub-domains/sub-graphs

Academic model problems

Problem patterns





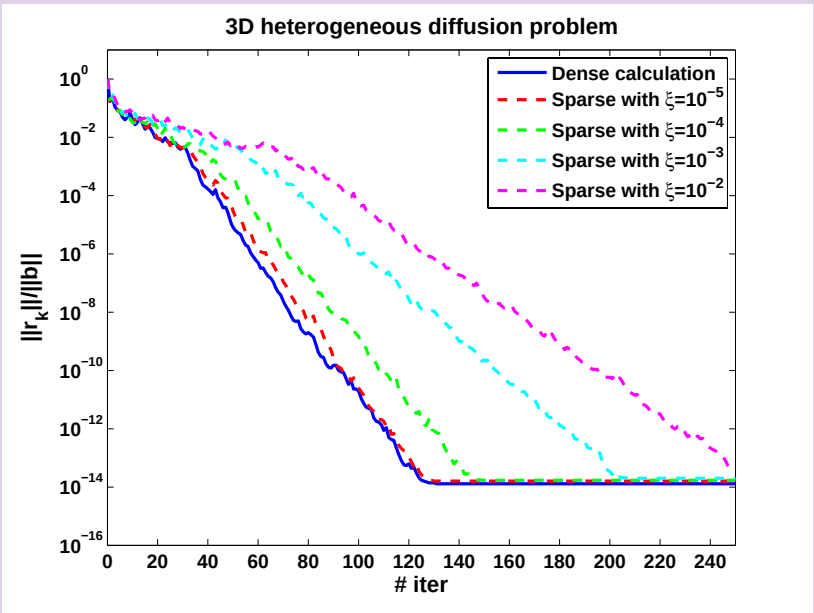
Diffusion equation ($\epsilon = 1$ and $v = 0$) and convection-diffusion equation

$$\begin{cases} -\epsilon \mathrm{div}(K.\nabla u) + v.\nabla u = f \quad \text{in} \quad \Omega, \\ u = 0 \quad \text{on} \quad \partial\Omega. \end{cases}$$

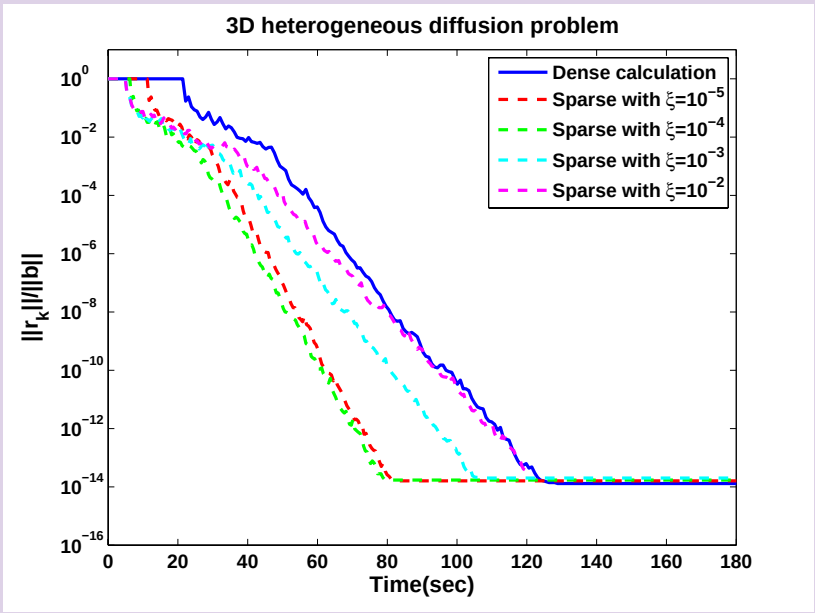
- Heterogeneous problems
- Anisotropic-heterogeneous problems
- Convection dominated term

Numerical behaviour of sparse preconditioners

Convergence history of PCG



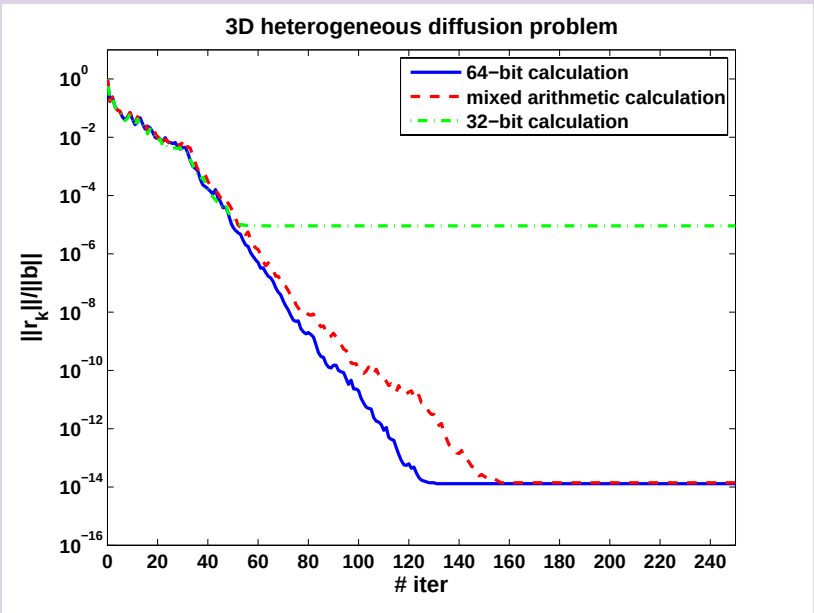
Time history of PCG



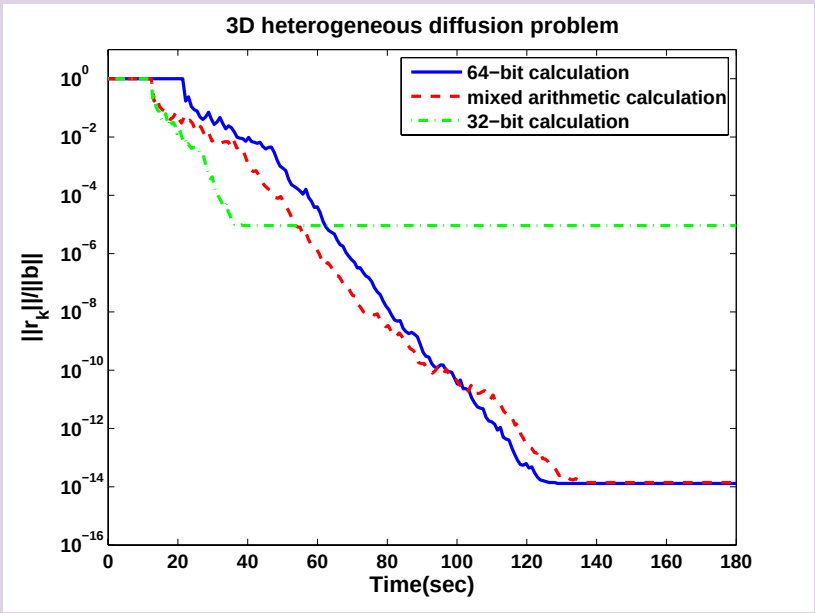
- $3D$ heterogeneous diffusion problem with 43 Mdof mapped on 1000 processors
- For ($\xi \lll$)the convergence is marginally affected while the memory saving is significant 15%
- For ($\xi \ggg$) a lot of resources are saved but the convergence becomes very poor 1%
- Even though they require more iterations, the sparsified variants converge faster as the time per iteration is smaller and the setup of the preconditioner is cheaper

Numerical behaviour of mixed preconditioners

Convergence history of PCG



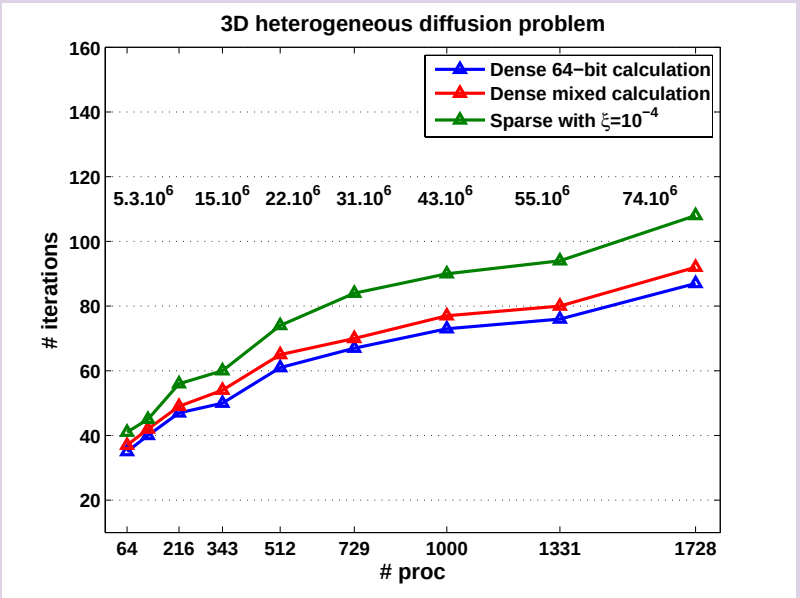
Time history of PCG



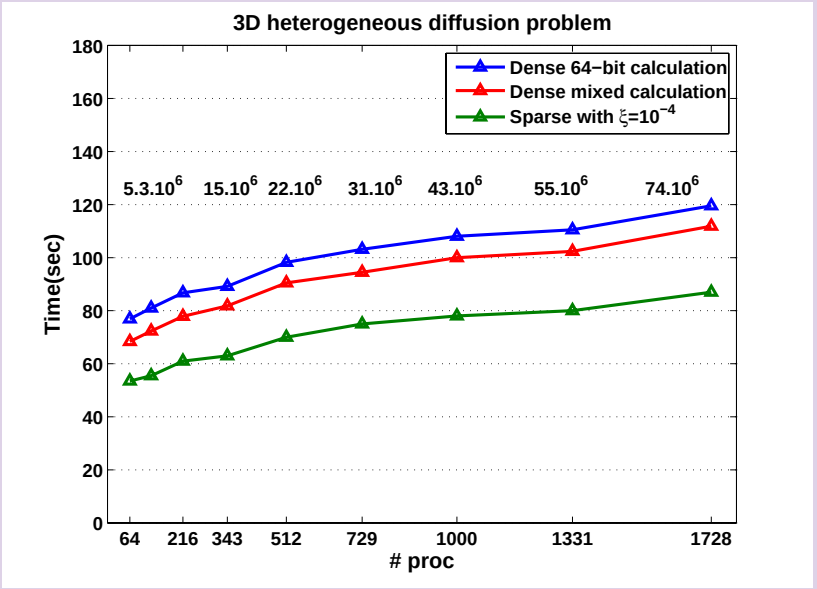
- 3*D* heterogeneous diffusion problem with 43 Mdof mapped on 1000 processors
- 64-bit and mixed computation both attained an accuracy at the level of 64-bit machine precision
- The number of iterations slightly increases
- The mixed approach is the fastest, down to an accuracy that is problem dependent

Scaled scalability on massively parallel platforms

Numerical scalability



Parallel performance



- The solved problem size varies from 2.7 up to 74 Mdof
- Control the grow in the # of iterations by introducing a coarse space correction
- The computing time increases slightly when increasing # sub-domains
- Although the preconditioners do not scale perfectly, the parallel time scalability is acceptable
- The trend is similar for all variants of the preconditioners using CG Krylov solver

Approximate Schur aproach: motivations joint work with Y. Saad

Exact vs. approximate Schur: memory saving (MB)

kept entries in factor	sub-domain mesh size 25^3 15 Kdof	30^3 27 Kdof	35^3 43 Kdof	40^3 64 Kdof	45^3 91 Kdof	50^3 125 Kdof	55^3 166 Kdof
Exact: 100% in U	254	551	1058	1861	3091	4760	7108
Appro: 21% in U	55	114	216	383	654	998	1506

Exact vs. approximate Schur: computing time (sec)

kept entries in factor	sub-domain grid size 25^3 15 Kdof	30^3 27 Kdof	35^3 43 Kdof	40^3 64 Kdof	45^3 91 Kdof	50^3 125 Kdof	55^3 166 Kdof
Exact: 100% in U	4.1	12.1	35.4	67.6	137	245	581
Appro: 21% in U	6.1	15.1	31.2	60.8	128	208	351
Appro: 10% in U	2.9	7.5	16.5	29.8	64	100	169

Approximate Schur aproach: motivations joint work with Y. Saad

Exact vs. approximate Schur: memory saving (MB)

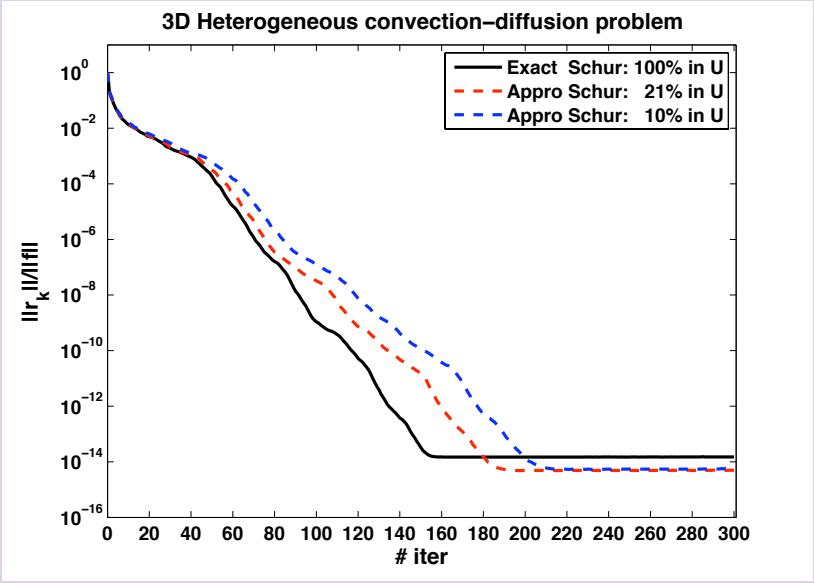
kept entries in factor	sub-domain mesh size 25^3 15 Kdof	30^3 27 Kdof	35^3 43 Kdof	40^3 64 Kdof	45^3 91 Kdof	50^3 125 Kdof	55^3 166 Kdof
Exact: 100% in U	254	551	1058	1861	3091	4760	7108
Appro: 21% in U	55	114	216	383	654	998	1506

Exact vs. approximate Schur: computing time (sec)

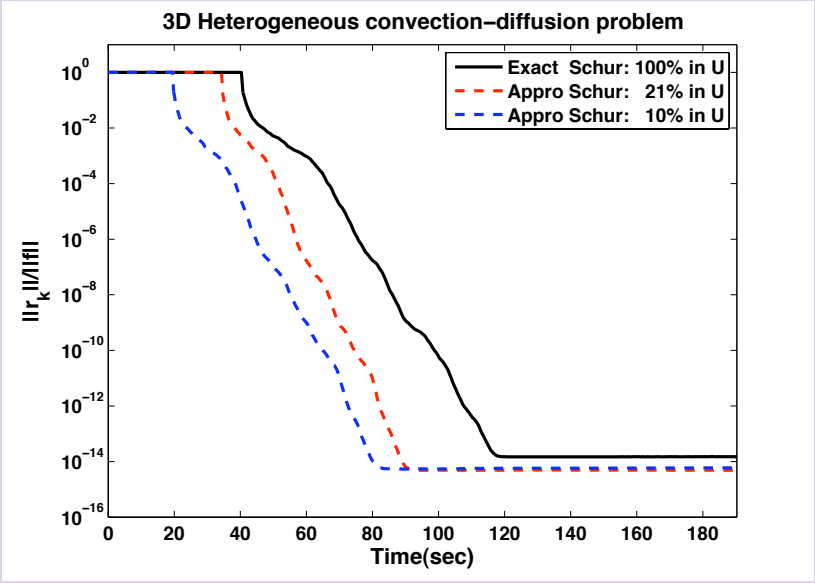
kept entries in factor	sub-domain grid size 25^3 15 Kdof	30^3 27 Kdof	35^3 43 Kdof	40^3 64 Kdof	45^3 91 Kdof	50^3 125 Kdof	55^3 166 Kdof
Exact: 100% in U	4.1	12.1	35.4	67.6	137	245	581
Appro: 21% in U	6.1	15.1	31.2	60.8	128	208	351
Appro: 10% in U	2.9	7.5	16.5	29.8	64	100	169

Numerical behaviour of approximate preconditioners

Convergence history of GMRES



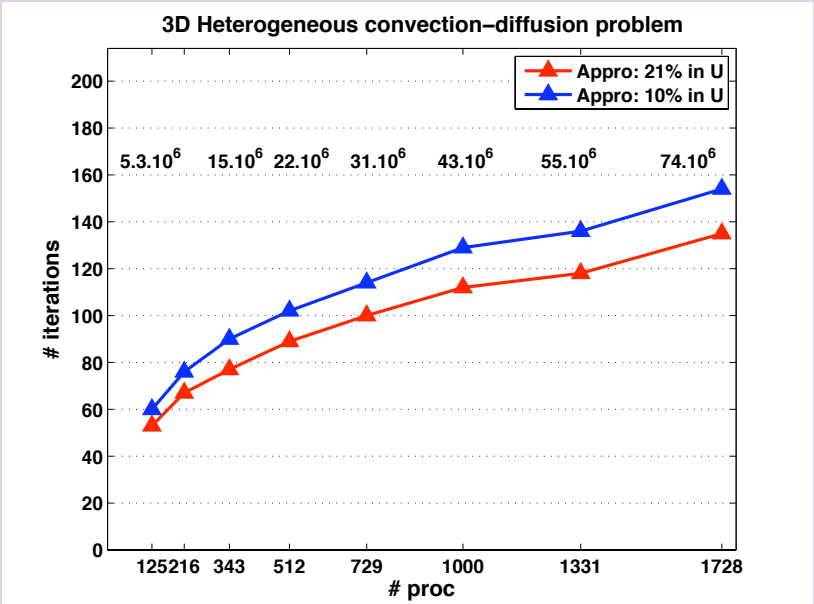
Time history of GMRES



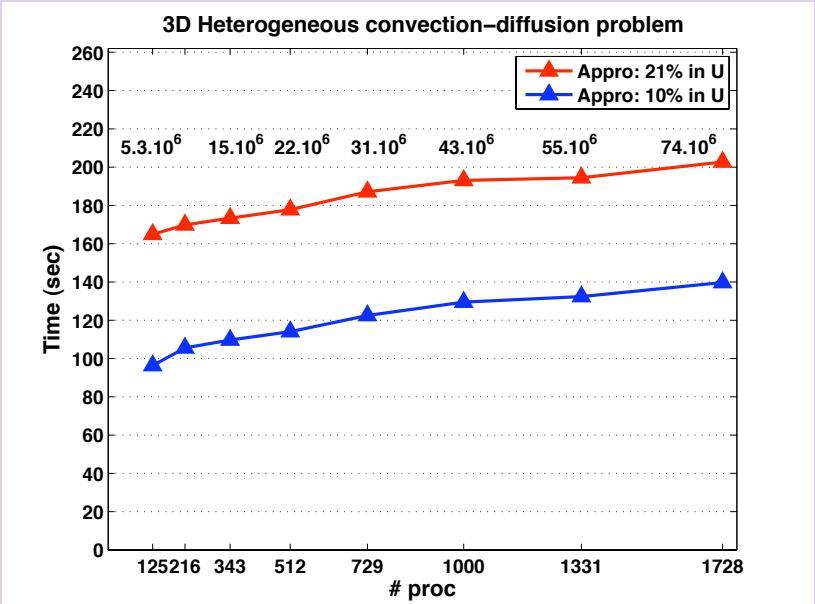
- 3*D* heterogeneous convection-diffusion problem of 74 Mdof mapped on 1728 processors
- the convergence is marginally affected while the memory saving is significant
- Even though they require more iterations, the approximate variant converge faster as the time per iteration is smaller and the setup of the preconditioner is cheaper

Weak scalability on massively parallel platforms

Numerical scalability



Parallel performance



- The solved problem size varies from 2.7 up to 74 Mdof
- The computing time increases slightly when increasing # sub-domains
- Even if the number of iterations to converge increases as the number of subdomains increases, the parallel scalability of the preconditioners remains acceptable

Summary on the model problems

[L.Giraud, A.Haidar, L.T.Watson - 08 ; L.Giraud, A.Haidar, Y.Saad - 10]

Sparse preconditioner

- For reasonable choice of the dropping parameter ξ the convergence is marginally affected
- The sparse preconditioner outperforms the dense one in time and memory

Mixed preconditioner

- Mixed arithmetic and 64-bit both attained an accuracy at the level of 64-bit machine precision
- Mixed preconditioner does not delay too much the convergence

Approximate preconditioner

- The convergence is marginally affected while the memory saving is significant
- The approximate variant converge faster as the time per iteration is smaller and the setup of the preconditioner is cheaper
- This preconditioner require some tuning for very hard problem (structural mechanics...)

On the weak scalability

- Although these preconditioners are local, possibly not numerically scalable, they exhibit a fairly good parallel time scalability (possible fix for elliptic problems)
- The trends that have been observed on this choice of model problem have been observed on many other problems

The Solstice framework

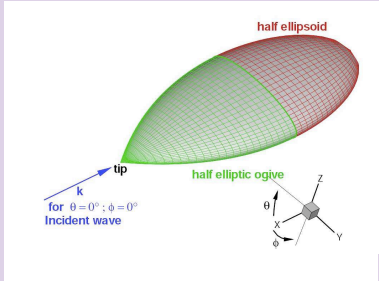
From meshes to adjacency graphs

- Extend the ideas from meshes to graph of matrices including unsymmetric matrices
- Experiments on end-users test problems
  1. Indefinite linear systems from EDF: structural mechanics
  2. Symmetric non-Hermitian linear system from CEA-CESTA: electromagnetism
- Towards a parallel package

Black-box algebraic domain decomposition solver: problem characteristics

Amande (Almond) problem

- Electromagnetism problem
- 6,994,683 dof
- 58,477,383 nnz



Haltere problem

- Electromagnetism problem
- 1,288,825 dof
- 10,476,775 nnz

"10 Millions"

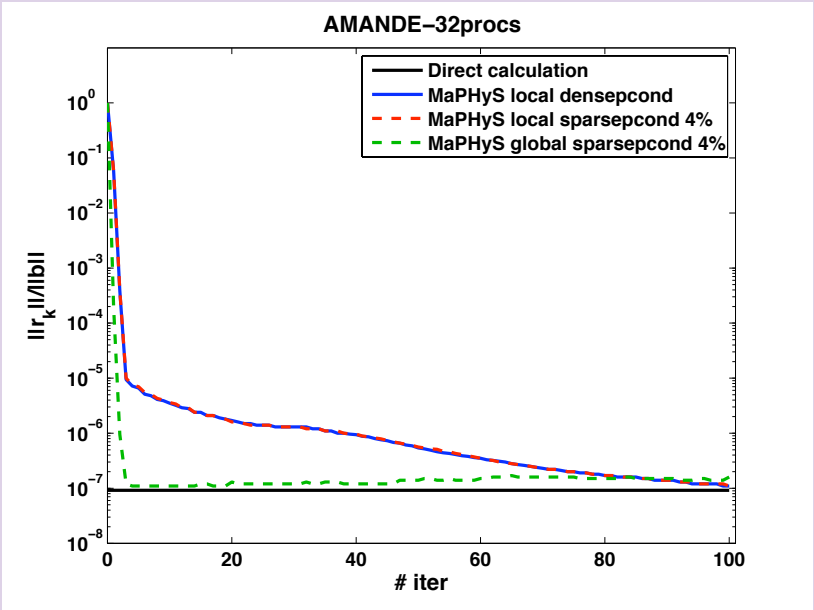
- Electromagnetism problem
- 10,423,737 dof
- 89,072,871 nnz

Perf001a

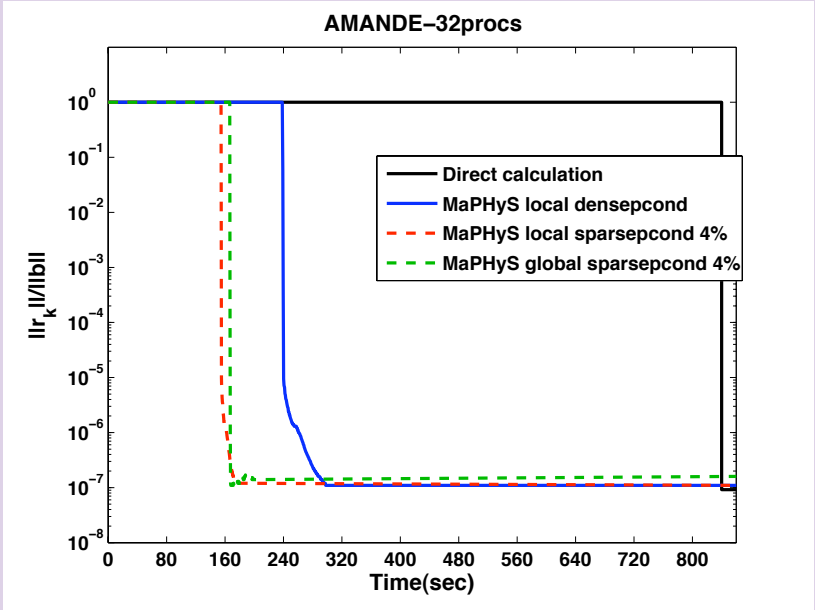
- Structural engineering
- 504,012 dof
- 17,262,024 nnz

MaPHyS: Almond problem

Convergence history



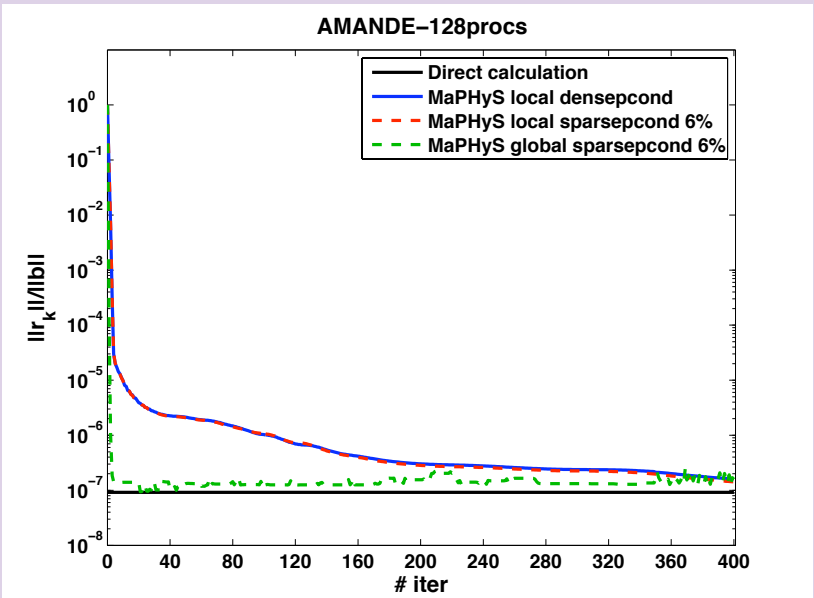
Time history



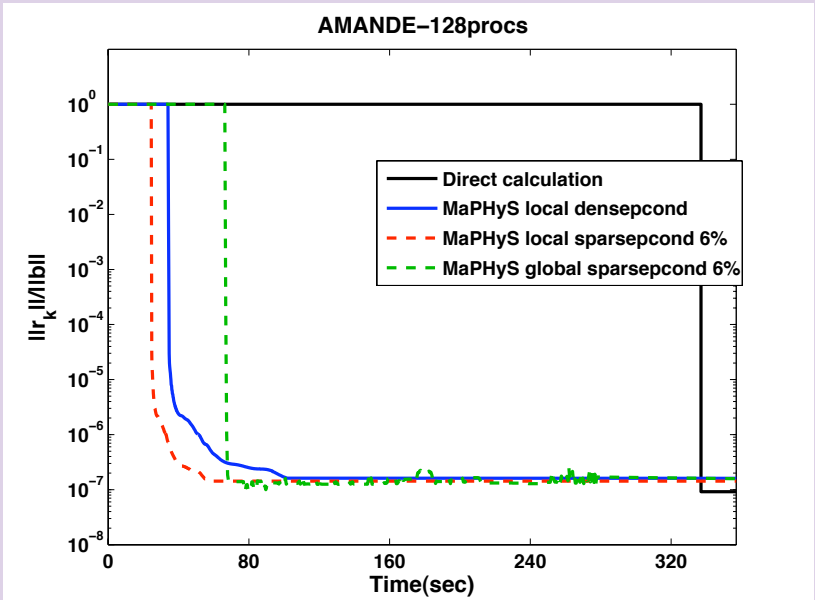
- Almond problem of 6.99 Mdof mapped on 32 processors
- In term of computing time, the sparse algorithm is about twice faster
- The global sparse preconditioner perform very well on this number of processors
- The attainable accuracy of the hybrid solver is comparable to the one computed with the direct solver

MaPHyS: Almond problem

Convergence history



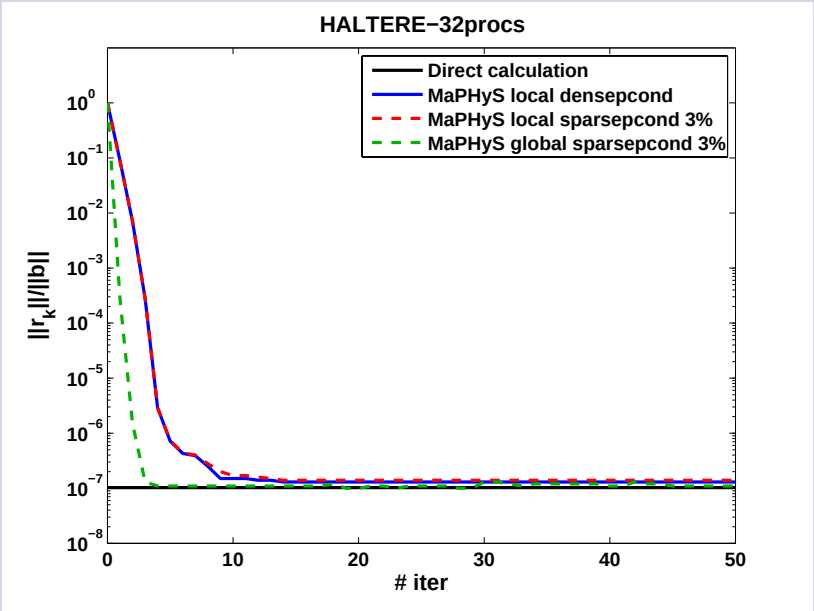
Time history



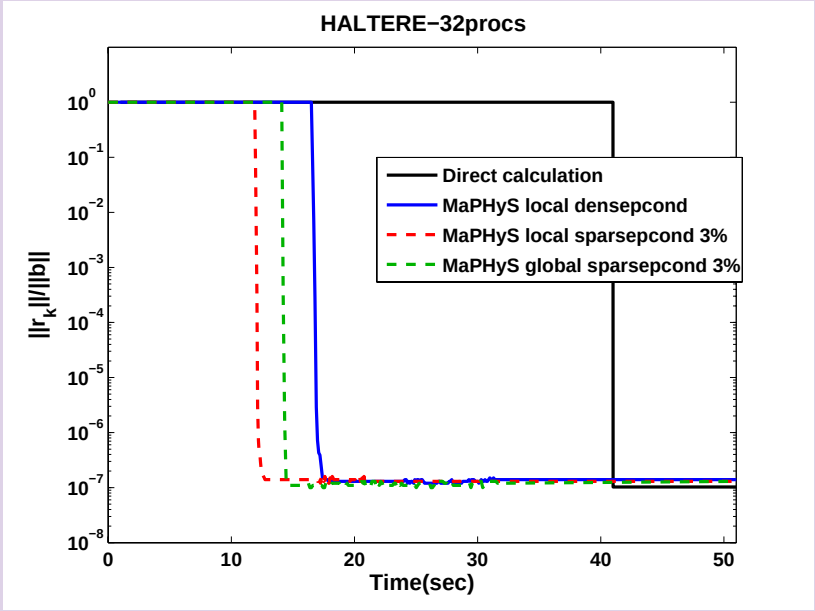
- Amende problem of 6.99 Mdof mapped on 128 processors
- The local sparse algorithm perform as well as the dense
- The local Schur complement are of small size thus the dense preconditioner perform well
- The global sparse preconditioner perform well numerically but slower in computing time

MaPHyS: Haltere problem

Convergence history



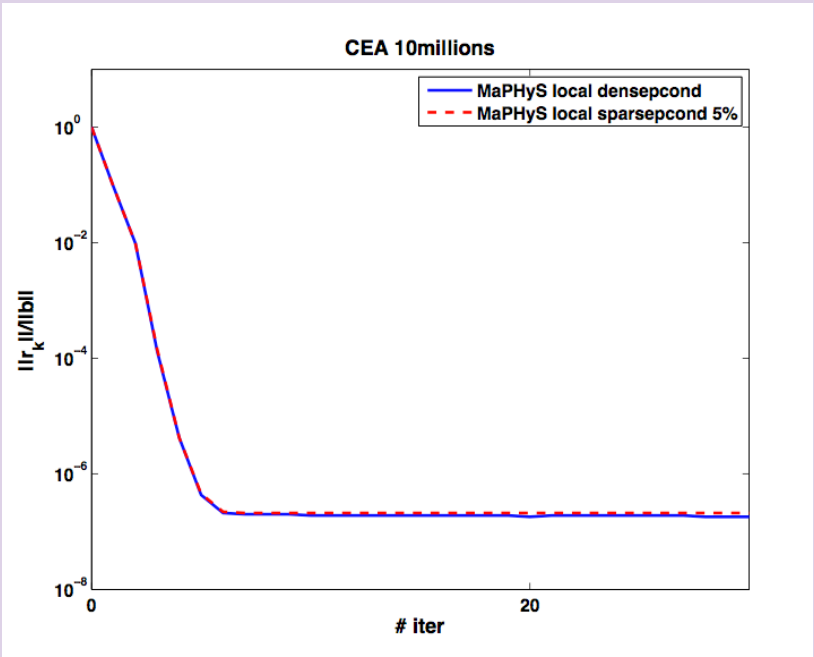
Time history



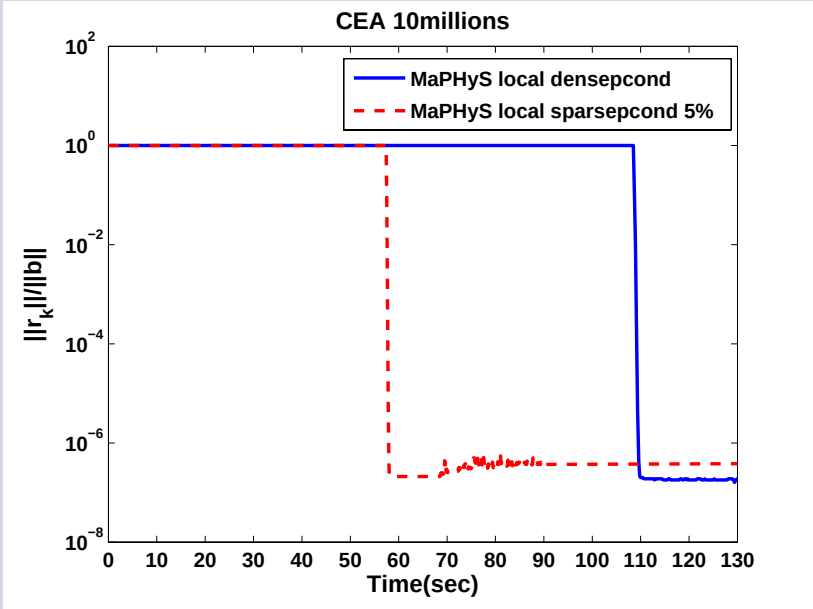
- Haltere problem of 1.3 Mdof mapped on 32 processors
- The local sparse algorithm perform as well as the dense
- The global sparse preconditioner perform very well on this number of processors
- The attainable accuracy of the hybrid solver is comparable to the one computed with the direct solver

MaPHyS: “10 Million” problem

Convergence history



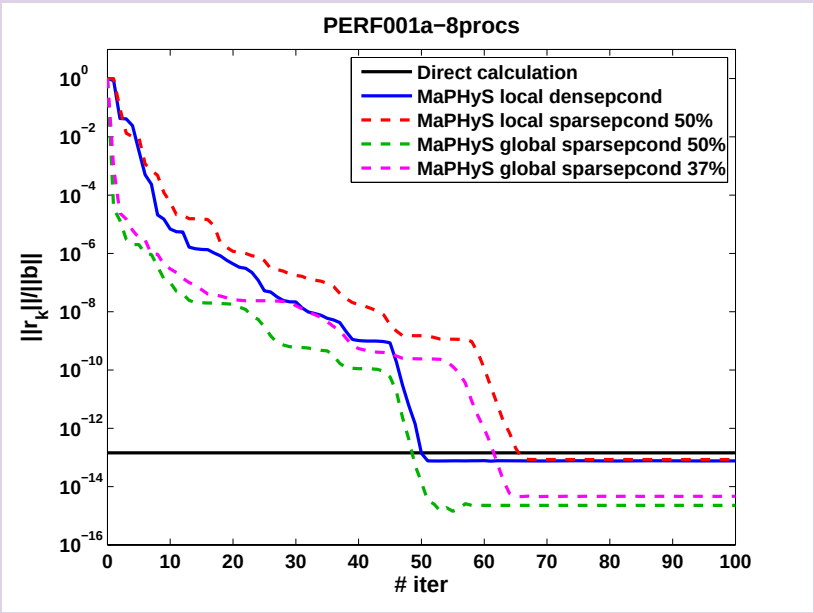
Time history



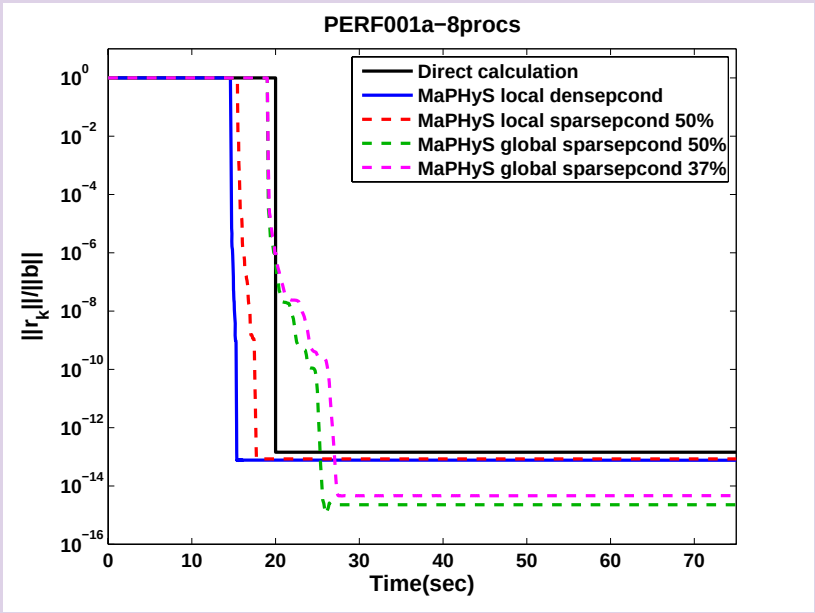
- “10 Millions” problem mapped on 64 processors
- The local sparse algorithm perform as well as the dense

MaPHyS: PerfOO1a

Convergence history



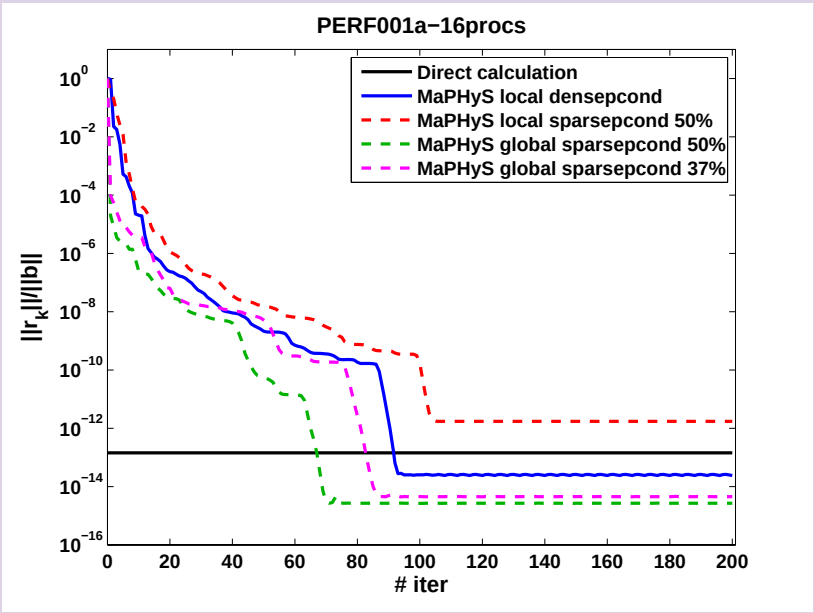
Time history



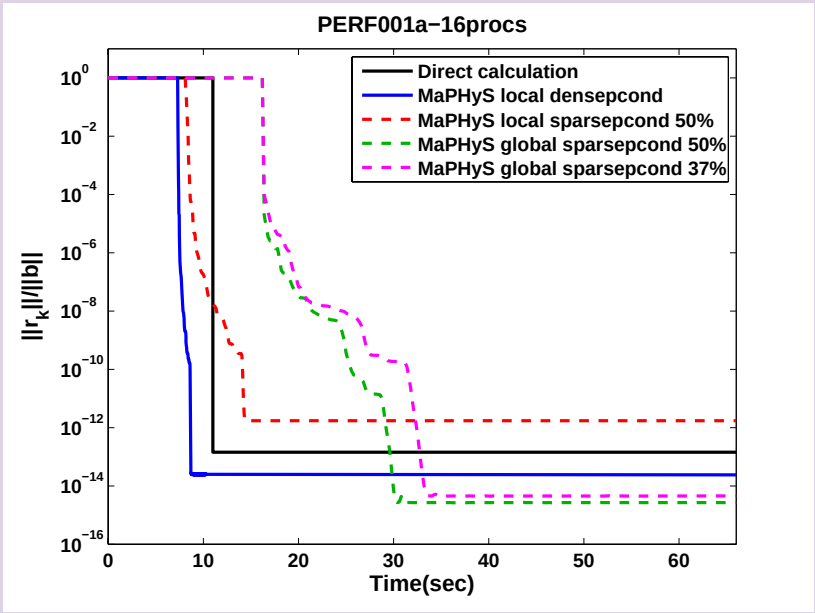
- Perf001a mapped on 8 processors

MaPHyS: PerfOO1a

Convergence history



Time history



- Perf001a mapped on 16 processors

Ongoing and future activities

- Integration of other direct solvers (multithreaded PaSTiX, SuperLU) and partitioners (Scotch/PT-Scotch) - ADT INRIA funding
- Improve the solver capability for symmetric indefinite et fully unsymmetric
- Complete the complexity analysis to study the computational scalability

`http://www.inria.fr/recherche/equipes/hiepacs.fr.html`

Acknowledgments

Credit to recent co-workers

- S. Pralet (SAMTECH now Bull)
- Y. Saad (Univ. Minnesota - PhyLeas INRIA associated team funding)
- L. T. Watson (Virginia Polytechnic Institute)
- MUMPS & PaStiX developers

Merci pour votre attention

Questions ?