Resolution of PDE using domain decomposition methods on TeraFLOPic architectures

Pierre Jolivet

Laboratoire Jacques-Louis Lions*
Laboratoire Jean Kuntzmann‡

December 6, 2011
Third workshop on `FreeFem++`

Work supervised by F. Nataf*, C. Prud'Homme‡, F. Hecht*

Outline

Context

We want to solve large systems arising from the finite element method.

Context

We want to solve large systems arising from the finite element method.

What are the different alternatives ?

- parallel direct solvers (`MUMPS`, `SuperLU` ..),
- parallel iterative solvers (`Hypre` ..),
- domain decomposition methods,
- ...

Context

We want to solve large systems arising from the finite element method.

What are the different alternatives ?

- parallel direct solvers (`MUMPS`, `SuperLU` ..),
- parallel iterative solvers (`Hypre` ..),
- domain decomposition methods,
- ...

$\Longrightarrow$ high-performance algorithms on massively parallel distributed memory multiprocessor architectures.

Within FreeFem++

Since version 1.16, bundled with the `Message Parsing Interface`.

Within FreeFem++

Since version 1.16, bundled with the Message Parsing Interface.

C++ plus load function within FreeFem++:
$\Longrightarrow$ OpenMP (shared memory architectures), C for CUDA (GPGPU).

Outline

`FreeFem++` is working on the following parallel architectures (amongst others):

	N° of cores	Memory	Peak performance	Compilers
hpc1@LJLL	64@2.00 Ghz	252 Go	$<$ 1 TFLOP/s	`Intel`
titane@CEA	12192*@2.93 Ghz	37 To	140 TFLOP/s	`Intel`
babel@IDRIS	40960@850 Mhz	20 To	139 TFLOP/s	`IBM+GNU`

* + 46080 CUDA cores

`http://www-ccrt.cea.fr`, Bruyères-le-Châtel, France.
`http://www.idris.fr`, Orsay, France.

Outline

The original Schwarz method for a 2-way decomposition

Consider the following BVP in $\mathbb{R}^d$ ($d = 2, 3$):

$$\begin{aligned} \nabla \cdot \kappa \nabla u &= F(u) \quad \text{in } \Omega \\ B(u) &= 0 \qquad \text{on } \partial\Omega \end{aligned}$$

The original Schwarz method for a 2-way decomposition

Consider the following BVP in $\mathbb{R}^d$ ($d = 2, 3$):

$$\begin{aligned} \nabla \cdot \kappa \nabla u &= F(u) && \text{in } \Omega \\ B(u) &= 0 && \text{on } \partial\Omega \end{aligned}$$

Then, solve in parallel:

$$\begin{aligned} \nabla \cdot \kappa \nabla u_1^{n+1} &= F(u_1^{n+1}) && \text{in } \Omega_1 \\ B(u_1^{n+1}) &= 0 && \text{on } \partial\Omega_1 \cap \partial\Omega \\ u_1^{n+1} &= u_2^n && \text{on } \partial\Omega_1 \cap \overline{\Omega_2} \end{aligned}
\qquad
\begin{aligned} \nabla \cdot \kappa \nabla u_2^{n+1} &= F(u_2^{n+1}) && \text{in } \Omega_2 \\ B(u_2^{n+1}) &= 0 && \text{on } \partial\Omega_2 \cap \partial\Omega \\ u_2^{n+1} &= u_1^n && \text{on } \partial\Omega_2 \cap \overline{\Omega_1} \end{aligned}$$

The original Schwarz method for a 2-way decomposition

Consider the following BVP in $\mathbb{R}^d$ ($d = 2, 3$):

$$\begin{aligned} \nabla \cdot \kappa \nabla u &= F(u) && \text{in } \Omega \\ B(u) &= 0 && \text{on } \partial\Omega \end{aligned}$$

Then, solve in parallel:

$$\begin{aligned} \nabla \cdot \kappa \nabla u_1^{n+1} &= F(u_1^{n+1}) && \text{in } \Omega_1 \\ B(u_1^{n+1}) &= 0 && \text{on } \partial\Omega_1 \cap \partial\Omega \\ u_1^{n+1} &= u_2^n && \text{on } \partial\Omega_1 \cap \overline{\Omega_2} \end{aligned}$$

$$\begin{aligned} \nabla \cdot \kappa \nabla u_2^{n+1} &= F(u_2^{n+1}) && \text{in } \Omega_2 \\ B(u_2^{n+1}) &= 0 && \text{on } \partial\Omega_2 \cap \partial\Omega \\ u_2^{n+1} &= u_1^n && \text{on } \partial\Omega_2 \cap \overline{\Omega_1} \end{aligned}$$

Limitations

1. Decomposition of Ω into N overlapping domains $(\Omega_i)_{1 \leqslant i \leqslant N}$,

Limitations

1. Decomposition of Ω into N overlapping domains $(\Omega_i)_{1 \leqslant i \leqslant N}$,
$\implies$ the original Schwarz method is not a *scalable* algorithm.

Limitations

1. Decomposition of Ω into N overlapping domains $(\Omega_i)_{1 \leqslant i \leqslant N}$,
   $\implies$ the original Schwarz method is not a *scalable* algorithm.

2. Highly heterogeneous coefficient κ in the BVP,

Limitations

1. Decomposition of Ω into N overlapping domains $(\Omega_i)_{1 \leqslant i \leqslant N}$,
   $\Longrightarrow$ the original Schwarz method is not a *scalable* algorithm.

2. Highly heterogeneous coefficient κ in the BVP,
   $\Longrightarrow$ long plateaux in the convergence of the algorithm.

How to deal with those problems

We introduce a *second* level (a coarse operator), denoted E.

How to deal with those problems

We introduce a *second* level (a coarse operator), denoted E.

Construction of E

1. build a coarse mesh $\mathcal{T}_{\text{coarse}}$, then a new `fespace` and a new `varf` on $\mathcal{T}_{\text{coarse}}$ (available under `examples++-mpi/MPIGMRES*D.edp`).
2. use the low-frequency modes of the *Dirichlet-to-Neumann* operator of the BVP at the interface of each neighboring domains (Nataf *et al.* , 2011).

How to deal with those problems

We introduce a *second* level (a coarse operator), denoted E.

Construction of E

1. build a coarse mesh $\mathcal{T}_{\text{coarse}}$, then a new `fespace` and a new `varf` on $\mathcal{T}_{\text{coarse}}$ (available under `examples++-mpi/MPIGMRES*D.edp`).
2. use the low-frequency modes of the *Dirichlet-to-Neumann* operator of the BVP at the interface of each neighboring domains (Nataf *et al.* , 2011).

Two-level methods

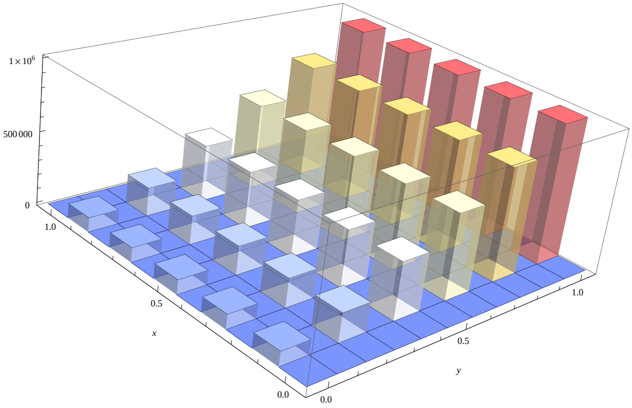
1. solve the global coarse problem $Ex = b$ on one node,
2. use the solution of the coarse problem on each local fine problem.

This is used to precondition a Krylov method (CG, GMRES) (Tang *et al.* , 2009).

Test case in $\mathbb{R}^2$

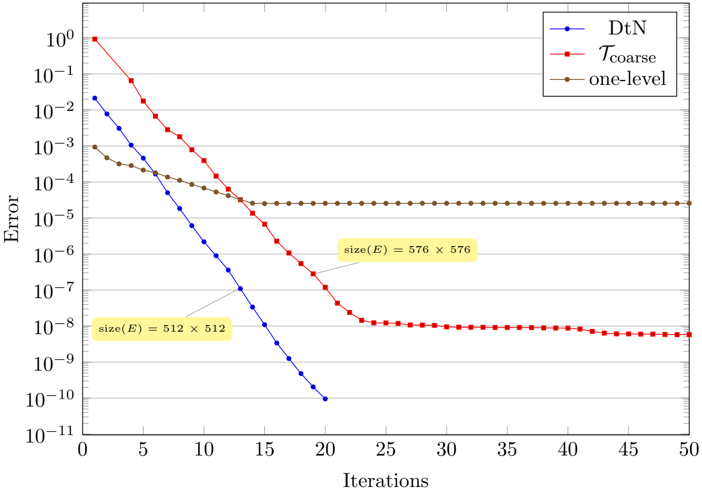
$$\begin{aligned} \nabla \cdot \kappa \nabla u &= 1 \quad \text{in } \Omega \\ u &= 0 \quad \text{on } \partial\Omega \end{aligned}$$

with a *skyscrapper* viscosity $\kappa(x, y)$:



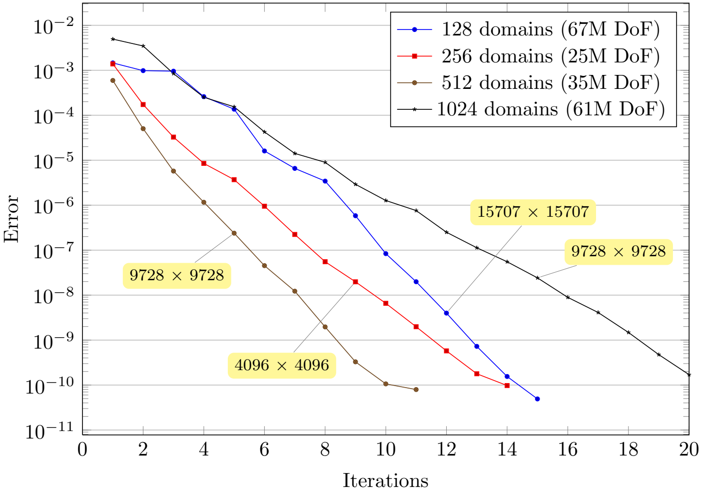
Convergence curves

2D test case running on 64 processors using $\mathbb{P}_2$ FE ($\varepsilon = 10^{-10}$, 450k DoF)



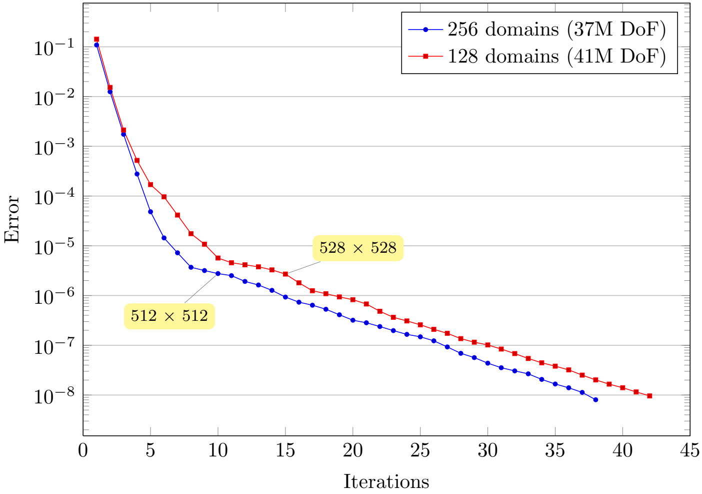
Some more 2D results

2D test case running on various number of processors using $\mathbb{P}_1$ FE ($\varepsilon = 10^{-10}$)



And some 3D results

3D test case running on various number of processors using $\mathbb{P}_2$ FE ($\varepsilon = 10^{-8}$)



Outline

Final words

Concerning `FreeFem++`:

1. efficient parallel algorithms (memory and CPU overhead with the `FreeFem++` - `MPI` bindings),
2. not very user-friendly interface at the moment (for DDM),
3. nice research tools (coarse operators, preconditionners ..).

Final words

Concerning `FreeFem++`:

1. efficient parallel algorithms (memory and CPU overhead with the `FreeFem++` - `MPI` bindings),
2. not very user-friendly interface at the moment (for DDM),
3. nice research tools (coarse operators, preconditionners ..).

Concerning domain decomposition methods:

1. add another level,
2. do parallel computations on more complex systems (e.g. elasticity).

Thanks for your attention.

NATAF, F., XIANG, H., DOLEAN, V., & SPILLANE, N. 2011.
A coarse space construction based on local Dirichlet to Neumann maps, to appear.
SIAM Journal on Scientific Computing.

TANG, J.M., NABBEN, R., VUIK, C., & ERLANGGA, Y.A. 2009.
Comparison of two-level preconditioners derived from deflation, domain decomposition and multigrid methods.
Journal of Scientific Computing, **39**(3), 340–370.