Calcul parallèle et éléments finis

Frédéric Nataf

Laboratory J.L. Lions (LJLL), CNRS, Equipe Alpines (Inria and Sorbonne University)

with Xavier Claeys, Pierre Jolivet, Frédéric Hecht, Pierre Marchand and P.H. Tournier

Café Groupe Calcul – March 2021

Outline

Outline

Benefits of DSL for Finite Element

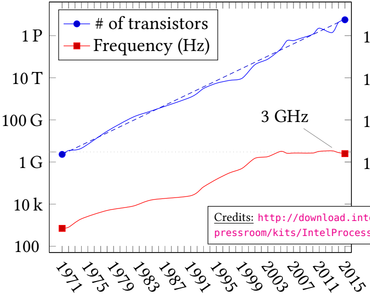
- Short and expressive scripts
- Closer to the math
- Multigrid $\Rightarrow$ faster algorithm
- Domain decomposition methods $\Rightarrow$ faster algorithm

Outline

Stagnation in hardware ⇒ Go parallel

Since year 2004:

- CPU frequency stalls at 2-3 GHz due to the heat dissipation wall.



Credits: http://download.intel.com/pressroom/kits/IntelProcessorHistory.pdf

All fields of computer science are impacted.

Some hardware facts

Energy

- a 32-bit floating-point operation requires 3.1 pJ
- whereas the same DRAM read requires 640 pJ.

Speed

- Since year 2004, CPU frequency stalls at 3 GHz due to the heat wall.
- Infiniband latency 1μ sec., 3,000 operations at 3GHz
- Minimum latency for an internode distance of 3 meters: 0.01μ sec. 30 operations at 3GHz

All fields of computer science are impacted.

Need for parallel linear solvers

A simplified view of modern architectures

- Unlimited number of fast cores
- Distributed data
- Limited amount of slow and energy intensive communication

Coarse Grain algorithm

- Maximize local computations
- Minimize communications (saves time and energy altogether)
- Minimize sequential task
- Redundant computations are welcome if they decrease communication

$A\,u = f$? Panorama of parallel linear solvers

Parallel Direct Solvers

MUMPS (J.Y. L'Excellent), SuperLU (Demmel, ...), PastiX, UMFPACK, PARDISO (O. Schenk),

Iterative Methods

- Fixed point iteration: Jacobi, Gauss-Seidel, SSOR
- Krylov type methods: Conjuguate Gradient (Stiefel-Hestenes), GMRES (Y. Saad), QMR (R. Freund), MinRes, BiCGSTAB (van der Vorst)

"Hybrid Methods"

- Multigrid (A. Brandt, Ruge-Stüben, Falgout, McCormick, A. Ruhe, U. Rüde, Y. Notay, ...) Frequency decomposition methods
- Domain decomposition methods (O. Widlund, C. Farhat, J. Mandel, P.L. Lions,) are a naturally parallel compromise

Outline

Why use a DS(E)L (FreeFem++, Feel++, Dune, Fenics or Firedrake) instead of `C/C++/Fortran/..` ?

- performances close to low-level language implementation,
- hard to beat something as simple as:

```
varf a(u, v) = int3d(mesh)([dx(u), dy(u), dz(u)]' * [dx(v), dy(v), dz(v)])
             - int3d(mesh)(f * v) + on(boundary_mesh)(u = 0) ,
```

- access to the variational formulation is then natural and that's what we need.

A few facts

- 1987: First version by O. Pironneau written in Pascal on Macintosh
- Since 1992: the main developer is Frédéric Hecht

Some FreeFem features

- Integrates many state of the art libraries.
- Automatic Mesh refinement native in 2d and via the plugin ”Mmg” (Frey at al.) in 3D
- Interpolate between different finite element spaces defined on different meshes, clouds of points to mesh
- Extensible via dynamic plugins
- Interface to MPI
- parallel version runs on Linux, Windows, Mac since 2017
- Docker on Qarnot and Rescale cloud computing platform
- Web browser (Javascript port thanks to A. Le Hyaric (LJLL))

demo avec chaleurEllipse.edp

Outline

Domain Decomposition via Metis or Scotch interface

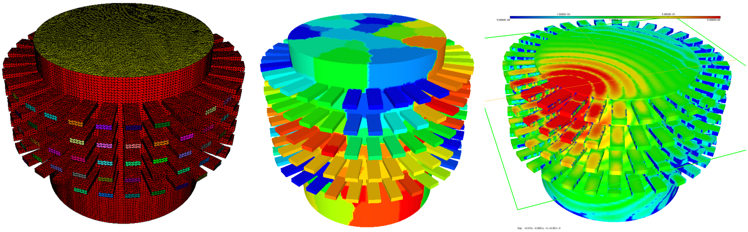


Figure: Electromagnetic chamber

Overlap is done by FreeFem based on the mesh connectivity

Strong Scalability test for 3D Maxwell

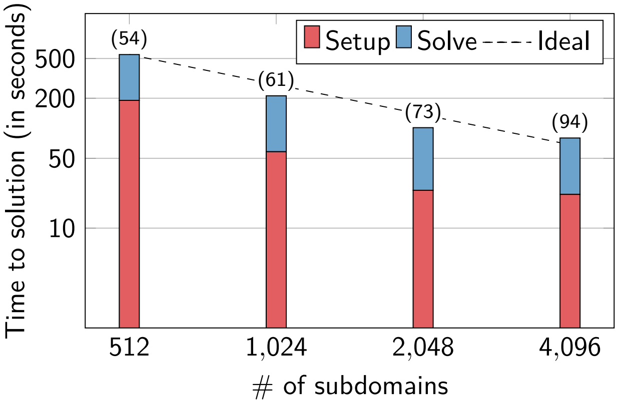


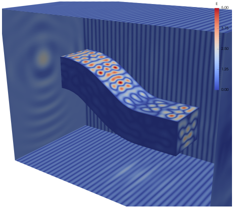
Figure: Maxwell 3D with edge elements of degree 2 - 119M d.o.f.

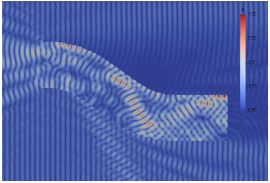
Bonazzoli, Dolean, Graham, Spence, Tournier, 2018.

order 2 edge elements (Nedelec), 10 pts per wavelength

$f = 10\ GHz$: $n \approx 1.07 \times 10^8$ $f = 16\ GHz$: $n \approx 1.98 \times 10^8$

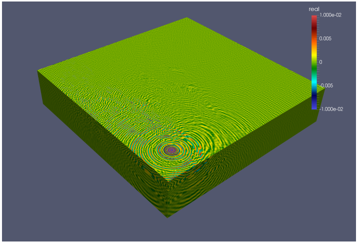
f	N_{sub}	# it	inner it	Total	Setup	GMRES	inner
10GHz	1536	32	1527	515.8	383.2	132.6	61.8
10GHz	3072	33	2083	285.0	201.6	83.4	40.6
16GHz	3072	43	3610	549.2	336.8	212.4	118.6
16GHz	6144	46	4744	363.0	210.5	152.5	96.8





5 points per wave length, P2 FE Simulations réalisées sur Occigen (CINES) noeuds Haswell

		cartesian mesh			adaptive mesh		
f	# cores	# dofs	# it	sec.	# dofs	# it	sec.
5	384	22 M	167	58	11 M	125	25
10	3072	176 M	340	121	85 M	253	59
20	12288	—	—	—	678 M	438	218



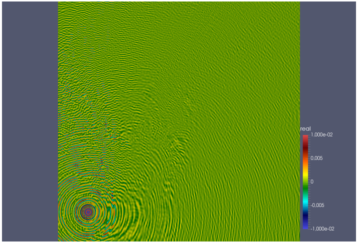


Figure: Simulations with FreeFem++ (P.H. Tournier)

Radiative transfer problem

$$(\mathbf{s} \cdot \nabla + \beta(\mathbf{x}))I(\mathbf{x}, \mathbf{s}) - \sigma_s(\mathbf{x}) \oint_{\mathcal{S}} I(\mathbf{x}, \mathbf{s}')\Phi(\mathbf{s}, \mathbf{s}') \, \mathrm{d}\mathbf{s}' \\ - \kappa(\mathbf{x})I_b(\mathbf{x}) = 0 \qquad \forall \mathbf{x} \in \Omega, \mathbf{s} \in \mathcal{S}$$

One billion unknowns in 60 seconds with 8192 MPI processes

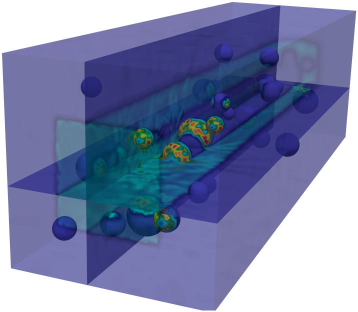


Figure: Badria, Jolivet, Rousseau, Le Corre, Digonnet and Favennec, 2018 – FreeFem++

Parallel Solvers in FreeFem

Two approaches:

- PETSc Interface (P. Jolivet (IRIT) with some inputs from S. Zampini (KAUST))
- FFDDM (P.H. Tournier (LJLL) with some inputs from F.N.)

PETSc Interface

If existing PETSc solvers (Direct solvers e.g. MUMPS, Multigrid e.g. GAMG, Domain Decomposition methods (hpddm)) do the job = Good solution

Pros :

- Huge library with not only solvers but also time schemes, optimizers, . . .

Cons :

- cannot handle Real and Complex value problems together
- Customising is tricky (PETSc library in C + MPI)

Near kernel, multigrid and Domain Specific Language

Efficiency in multigrid depends on the access to the near kernel of the matrix:

- Infer it from some assumption on the problem at hand (e.g. Graph Laplacian or Darcy) (Notay-Napov 2016). Limited applicability
- user provided near kernel of the matrix (GAMG with PETSc) $\Rightarrow$ DSL is a real bonus (any FE, any PDE)

demo via the examples of the FreeFem distribution

Pros :

- Handles both Real and Complex value problems at the same time
- Written in FreeFem language via prefixed MACROS
- Very good for experimenting with DD methods
- Parallel direct solver MUMPS is also provided

Cons :

- Multigrid is accessible only via the PETSc interface.
- No GPU access (but with the new ARM SoC (cf. A64FX from Fujitsu or m1 from Apple) this is maybe only a temporary issue)

Note that a FreeFem script can mix the PETSc interface with ffddm

DDM, Neumann sub matrix and Domain Specific Language

Scalable DDM needs for each subdomain j to have access to the local Neumann matrix A_j^{Neu}.

- Ask the developer of the simulation code to provide it (HPDDM, Jolivet & N.)
- Infer it from some assumption on the problem at hand (e.g. Graph Laplacian)
- Use Domain Specific Language (DSL) for finite element (or volume) method

DSL also useful for Optimized interface conditions
Voir exemples avec ffddm

Outline

https://freefem.org/

- Online documentation
- Sources on Gihub
- Liste de discussion
- FreeFem days
  - Introduction to parallel FreeFem by P. Jolivet and P.H. Tournier on Youtube https://www.youtube.com/watch?v=-Aw2O46V2bo&feature=youtu.be
  - Youtube channel FreeFem

Recap and Not mentioned here

Recap: Bonuses from DSL

- Short and expressive scripts
- Closer to the math
- Multigrid $\Rightarrow$ faster algorithm
- Domain decomposition methods $\Rightarrow$ faster algorithm

Not mentioned here

- Boundary Element Method and Domain Decomposition for BEM (Xavier Claeys, Pierre Marchand, P.H Tournier) along with matrix compression
- ParMmg parallel Mmg
- Paraview

Acknowledgements

These developments have been developed, benchmarked and used
on national HPC resources of TGCC at IDRIS and of OCCIGEN at CINES
under the allocations 20XX-067730 made by Grand Equipement National de Calcul Intensif (GENCI).

THANK YOU
FOR YOUR ATTENTION!

Acknowledgements

These developments have been developed, benchmarked and used
on national HPC resources of TGCC at IDRIS and of OCCIGEN at CINES
under the allocations 20XX-067730 made by Grand Equipement National de Calcul Intensif (GENCI).

THANK YOU
FOR YOUR ATTENTION!