HIGH-PERFORMANCE COMPUTING

Pierre Jolivet, IRIT–CNRS
pierre.jolivet@enseeiht.fr

CSMA Juniors
May 15

INTRODUCTION

HARDWARE ARCHITECTURES FOR HPC

- platforms are ranked according to various metrics
- Floating Point Operations per second is critical
- many other important characteristics
- frequency-scaling has ended since circa 2005

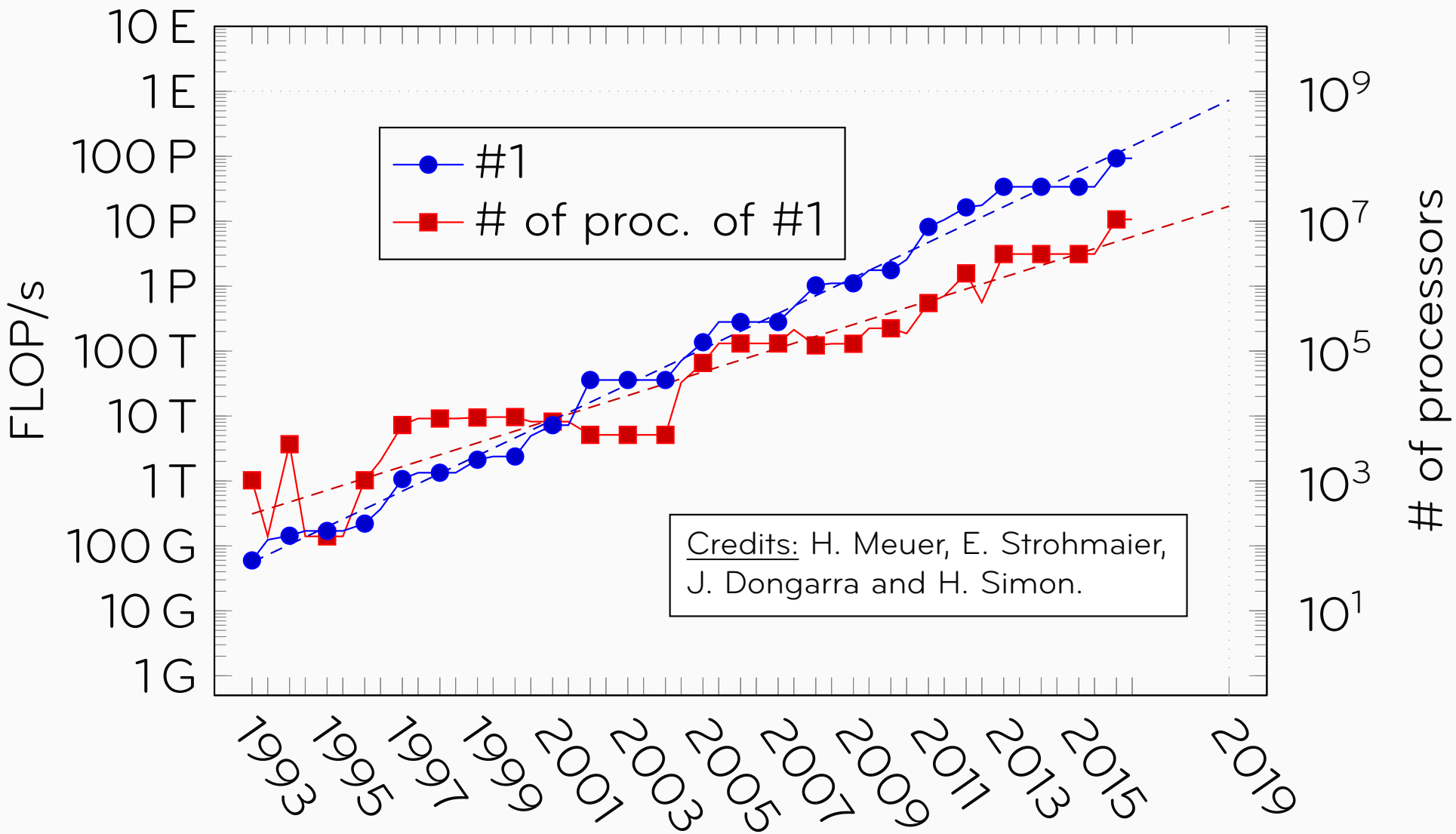
HARDWARE ARCHITECTURES FOR HPC

- platforms are ranked according to various metrics
- Floating Point Operations per second is critical
- many other important characteristics
- frequency-scaling has ended since circa 2005

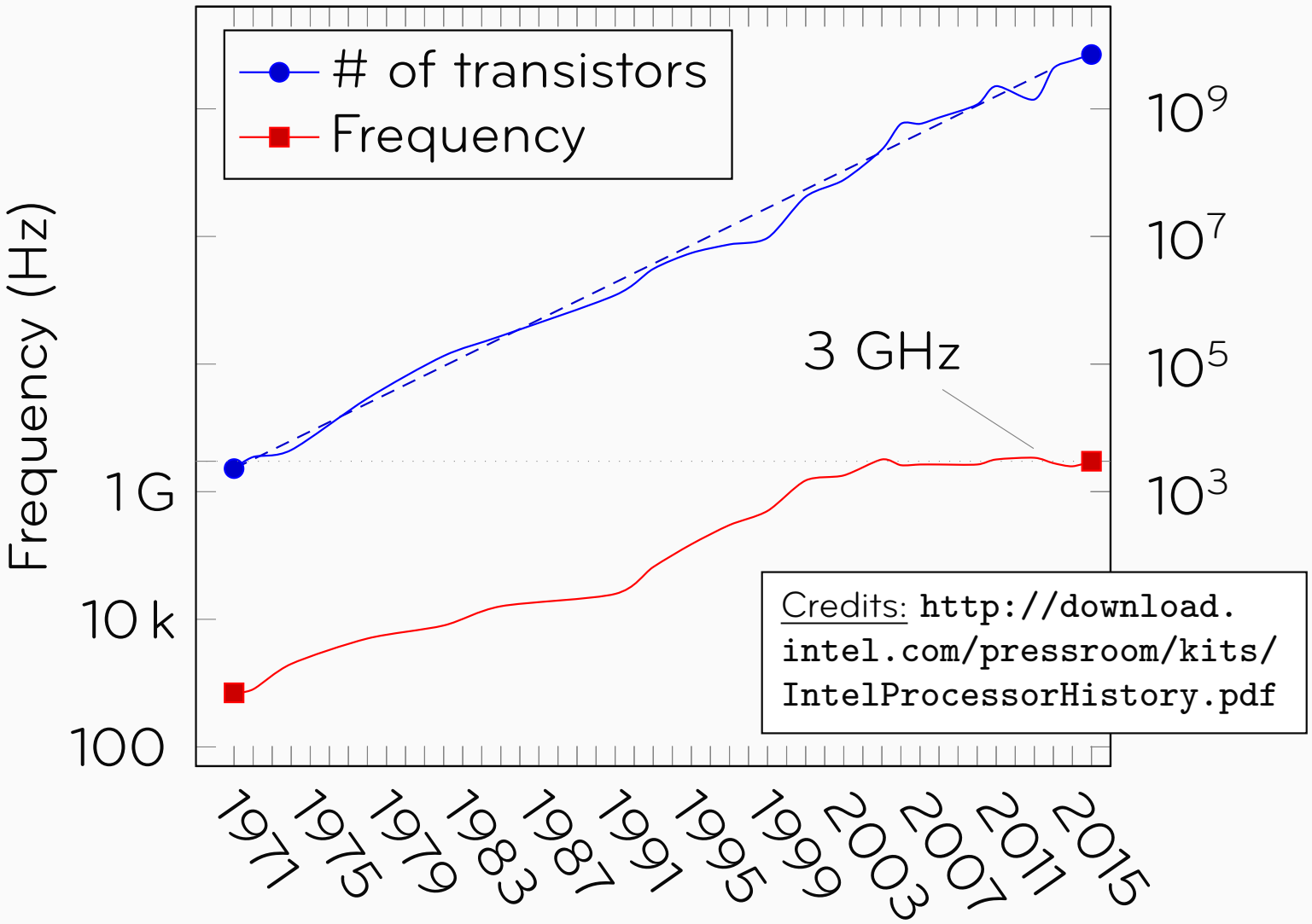
Most used paradigms/interfaces

- Message Passing Interface (`http://mpi-forum.org`)
- OpenMP (`http://www.openmp.org`)

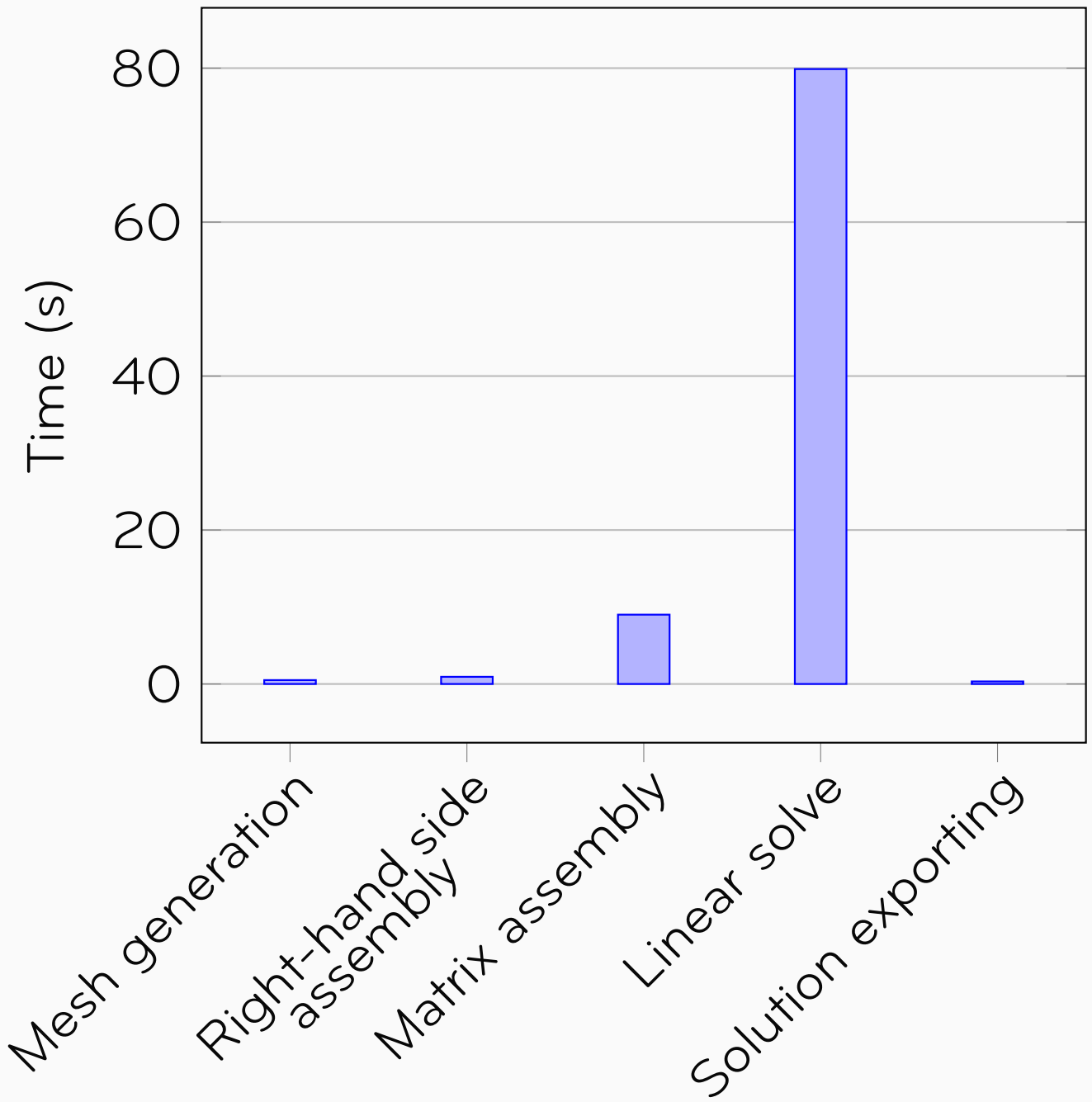
TOP 500


10 E
1 E
100 P
10 P
1 P
100 T
10 T
1 T
100 G
10 G
1 G
FLOP/s
#1
# of proc. of #1
Credits: H. Meuer, E. Strohmaier, J. Dongarra and H. Simon.
10^9
10^7
10^5
10^3
10^1
# of processors
1993
1995
1997
1999
2001
2003
2005
2007
2009
2011
2013
2015
2019

f-SCALING


# of transistors
Frequency
Frequency (Hz)
1 G
10 k
100
10^9
10^7
10^5
10^3
3 GHz
1971
1975
1979
1983
1987
1991
1995
1999
2003
2007
2011
2015
Credits: http://download.intel.com/pressroom/kits/IntelProcessorHistory.pdf

IMPLICIT CODE

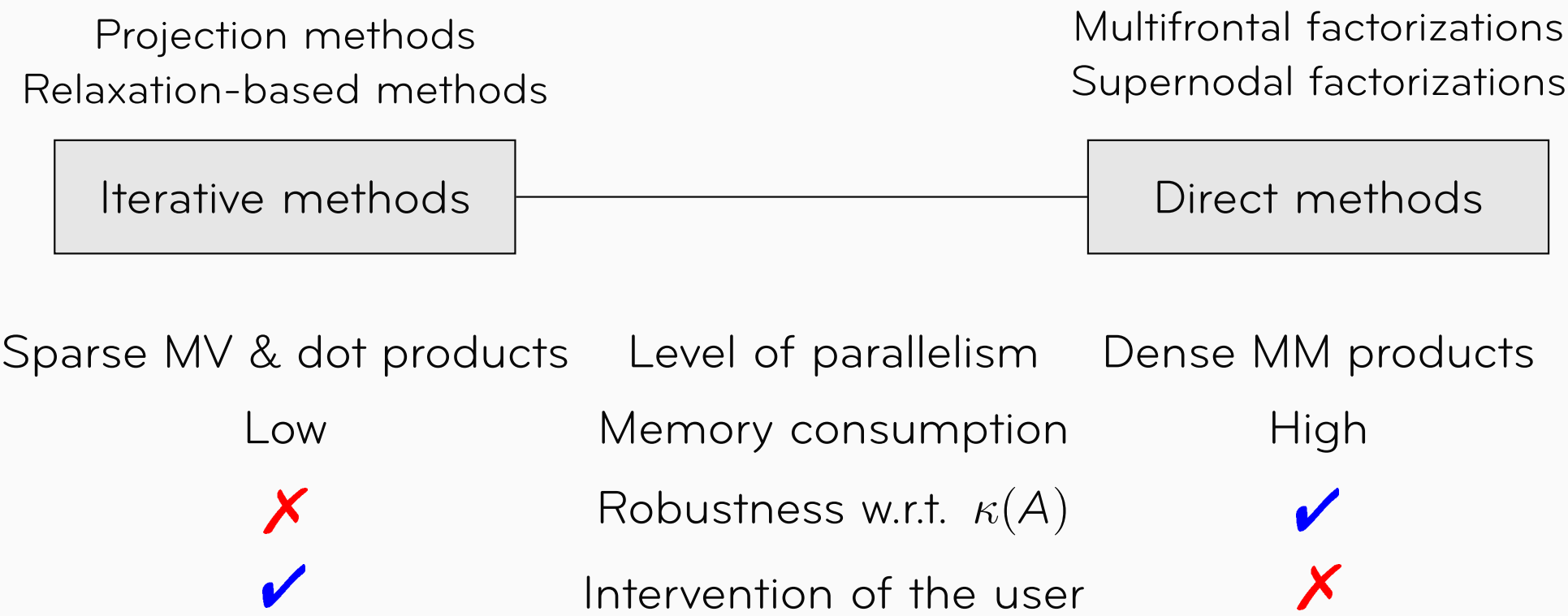


METHODOLOGY AND AVAILABLE LIBRAIRIES

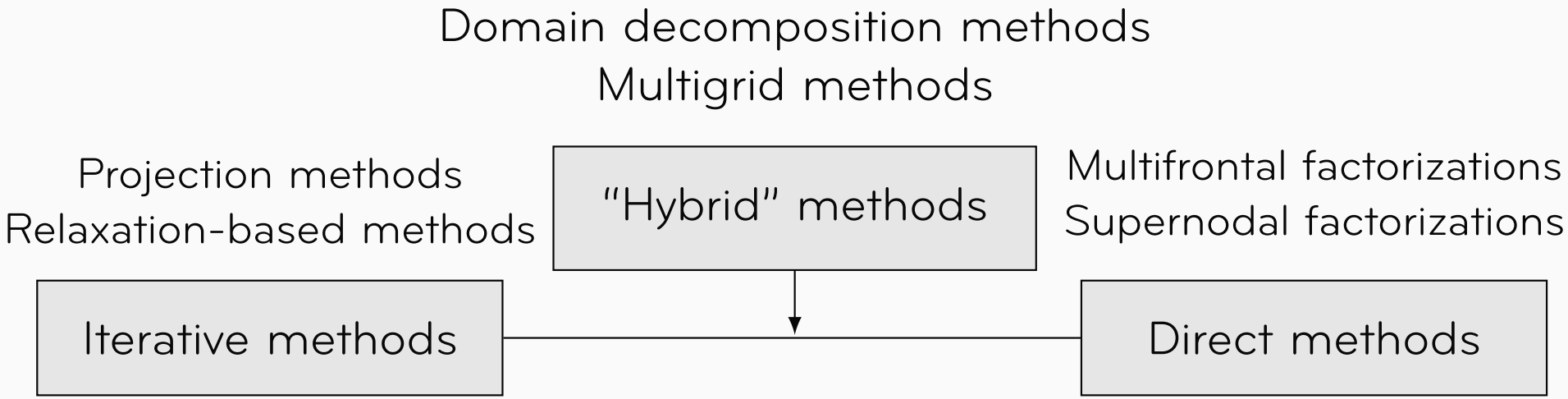
FOCUS OF THIS MINI-COURSE

- efficient strategies for solving linear systems $Ax = b$
- appropriate use of modern architectures
- introduction to most commonly used librairies for HPC

STRATEGIES FOR SOLVING LINEAR SYSTEMS



STRATEGIES FOR SOLVING LINEAR SYSTEMS



Sparse MV & dot products	Level of parallelism	Dense MM products
Low	Memory consumption	High
✗	Robustness w.r.t. $\kappa(A)$	✓
✓	Intervention of the user	✗

ABSTRACT SETTING FOR USING PRECONDITIONERS

Left preconditioning

$$M^{-1}Ax = M^{-1}b$$

Right preconditioning

$$AM^{-1}u = b$$

with $x = M^{-1}u$

ABSTRACT SETTING FOR USING PRECONDITIONERS

Left preconditioning

$$M^{-1}Ax = M^{-1}b$$

Right preconditioning

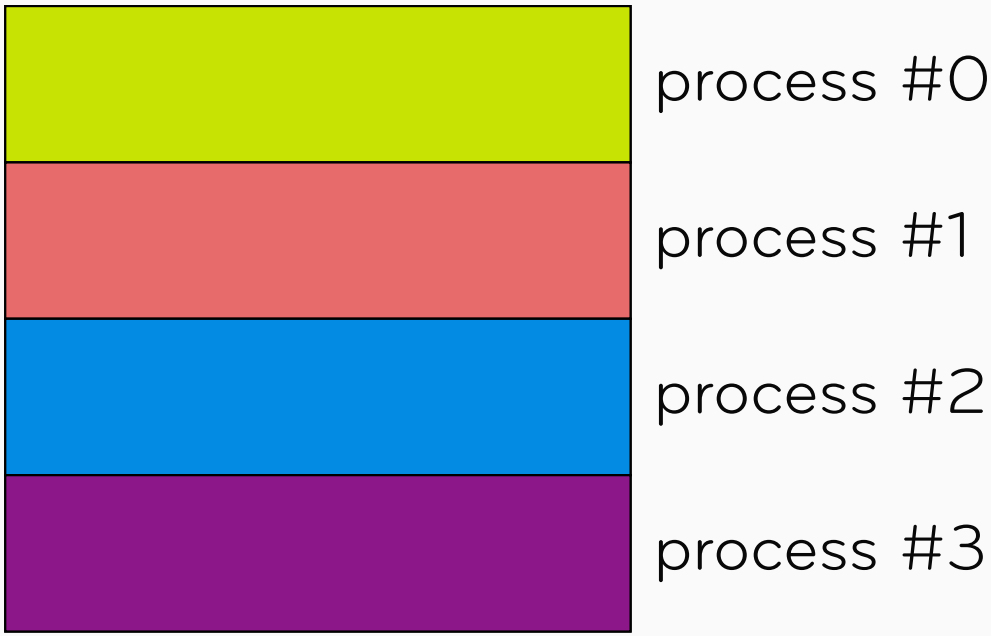
$$AM^{-1}u = b$$

with $x = M^{-1}u$

$\Longrightarrow$ M^{-1} must be cheap to apply!

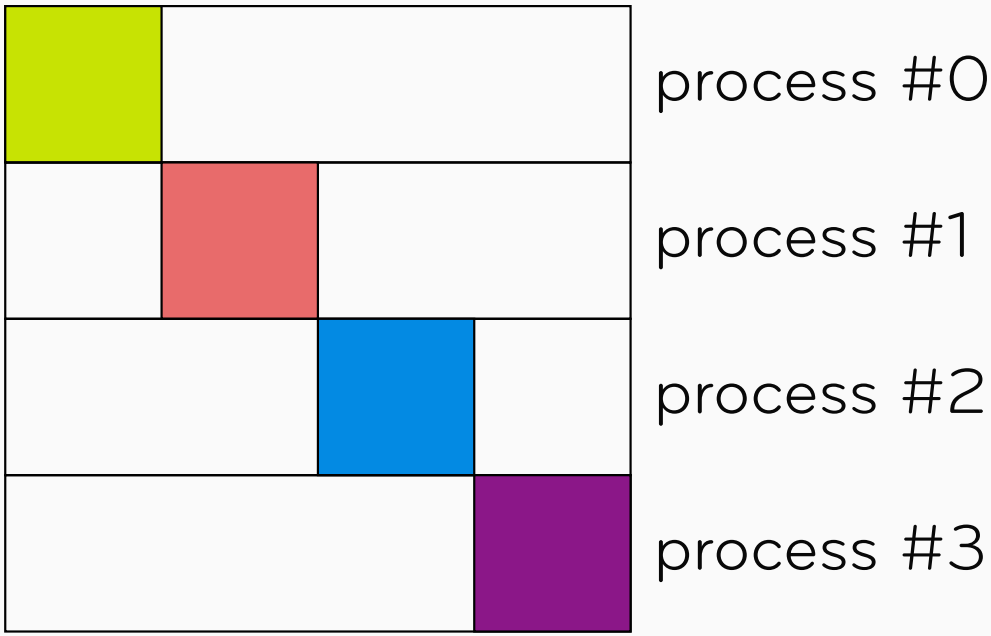
A SIMPLE EXAMPLE

Instead of using the inverse of the distributed matrix



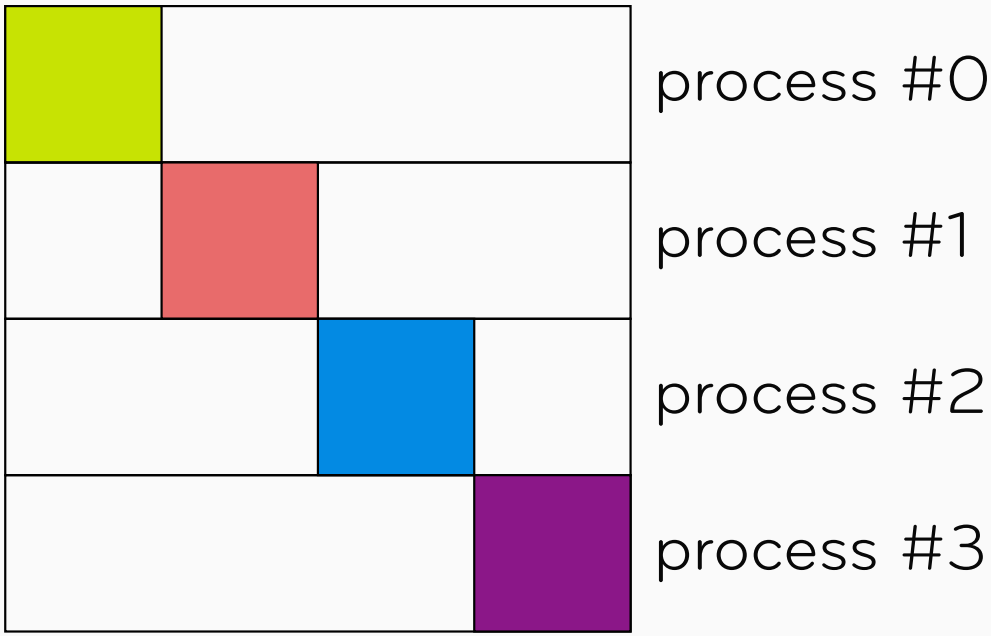
A SIMPLE EXAMPLE

"Invert" only the diagonal blocks to define M^{-1}



A SIMPLE EXAMPLE

"Invert" only the diagonal blocks to define M^{-1}



Known as the block Jacobi method (the most basic DDM)

GENERAL GUIDELINES

- if you want performance
- if you don't want to waste time

$\Longrightarrow$ do not reinvent the wheel

GENERAL GUIDELINES

- if you want performance
- if you don't want to waste time

$$\Longrightarrow \text{ do not reinvent the wheel}$$

Two kinds of libraries in the HPC ecosystem:

- black-box solvers (direct methods, algebraic multrigrid methods, iterative methods)
- general-purpose backends (collections of linear/nonlinear solvers, time-stepping methods)

MESH PARTITIONING

Coarse-grained parallelism

- less synchronization
- low amount of communication

$$\Longrightarrow \text{ domain decomposition over MPI}$$

MESH PARTITIONING

Coarse-grained parallelism

- less synchronization
- low amount of communication

$$\Longrightarrow \text{ domain decomposition over MPI}$$

- METIS (`http://glaros.dtc.umn.edu/gkhome/metis/metis/overview`)
- SCOTCH (`http://www.labri.fr/perso/pelegrin/scotch`)

MESH PARTITIONING

Coarse-grained parallelism

- less synchronization
- low amount of communication

$$\Longrightarrow \text{ domain decomposition over MPI}$$

- METIS (`http://glaros.dtc.umn.edu/gkhome/metis/metis/overview`)
- SCOTCH (`http://www.labri.fr/perso/pelegrin/scotch`)

Fined-grained parallelism

- more synchronization
- better load-balance

$$\Longrightarrow \text{ graph coloring over threads}$$

Discretization kernel

- after domain decomposition/coloring
- depends on the physics
- yields a linear system

DIRECT SOLVERS

Discretization kernel

- after domain decomposition/coloring
- depends on the physics
- yields a linear system

- MUMPS (`http://mumps.enseeiht.fr`)
- PaStiX (`http://pastix.gforge.inria.fr`)
- UMFPACK (`http://faculty.cse.tamu.edu/davis/suitesparse.html`)
- CHOLMOD
- MKL PARDISO (`https://software.intel.com`)
- PARDISO (`http://www.pardiso-project.org`)
- SuperLU (`http://crd-legacy.lbl.gov/~xiaoye/SuperLU`)

DIRECT SOLVERS

Discretization kernel

- after domain decomposition/coloring
- depends on the physics
- yields a linear system

- MUMPS (`http://mumps.enseeiht.fr`)
- PaStiX (`http://pastix.gforge.inria.fr`)
- UMFPACK (`http://faculty.cse.tamu.edu/davis/suitesparse.html`)
- CHOLMOD
- MKL PARDISO (`https://software.intel.com`)
- PARDISO (`http://www.pardiso-project.org`)
- SuperLU (`http://crd-legacy.lbl.gov/~xiaoye/SuperLU`)

GENERAL PURPOSE LINEAR ALGEBRA TOOLKITS

- PETSc (`http://www.mcs.anl.gov/petsc`)
- Trilinos (`https://trilinos.org`)

- *hypre* (`http://www.llnl.gov/casc/hypre`)
- DUNE (`https://www.dune-project.org`)
- PARALUTION (`http://www.paralution.com`)
- HPDDM (`https://github.com/hpddm/hpddm`)

GENERAL PURPOSE LINEAR ALGEBRA TOOLKITS

- PETSc (http://www.mcs.anl.gov/petsc)
- Trilinos (https://trilinos.org)

- *hypre* (http://www.llnl.gov/casc/hypre)
- DUNE (https://www.dune-project.org)
- PARALUTION (http://www.paralution.com)
- HPDDM (https://github.com/hpddm/hpddm)

COMPLETE DISTRIBUTED FINITE ELEMENT SOLVER

GOAL OF THE EXERCISE

- see the effect of mesh partitioning
- understand the consequences for a finite element kernel
- use a distributed linear algebra backend
- play with different solvers/preconditioners

GOAL OF THE EXERCISE

- see the effect of mesh partitioning
- understand the consequences for a finite element kernel
- use a distributed linear algebra backend
- play with different solvers/preconditioners

Dependencies

- Gmsh
- MPI
- FreeFem++
- PETSc

`http://jolivet.perso.enseeiht.fr/CSMA17_hpc.tar.gz`

STEPS RELATED TO THE MESH

1. mesh generation

STEPS RELATED TO THE MESH

1. mesh generation
2. mesh decomposition

STEPS RELATED TO THE MESH

1. mesh generation
2. mesh decomposition
3. local mesh coloring

STEPS RELATED TO THE DISCRETIZATION KERNEL

1. ghost elements

STEPS RELATED TO THE DISCRETIZATION KERNEL

1. ghost elements
2. local finite element numbering

STEPS RELATED TO THE DISCRETIZATION KERNEL

1. ghost elements
2. local finite element numbering
3. global numbering

SYSTEM OF LINEAR ELASTICITY IN VARIATIONAL FORM

- $-\text{div}\ \sigma = f$ in Ω, + suitable BCs
- $\varepsilon = \frac{1}{2}(\nabla u + \nabla u^T)$
- $\sigma = C : \varepsilon$

$$\int_\Omega \lambda \text{div}\ u \cdot \text{div}\ v + \int_\Omega 2\mu\epsilon(u) \cdot \epsilon(v) - \int_\Omega fv = 0$$

STEPS RELATED TO THE SOLUTION PHASE

1. global matrix assembly

STEPS RELATED TO THE SOLUTION PHASE

1. global matrix assembly
2. preconditioner definition

STEPS RELATED TO THE SOLUTION PHASE

1. global matrix assembly
2. preconditioner definition
3. solution of the linear system

STEPS RELATED TO THE SOLUTION PHASE

1. global matrix assembly
2. preconditioner definition
3. solution of the linear system
4. saving the solution to disk

CLOSING REMARKS

- parallelism is necessary for large-scale computations
- can be tricky on legacy codes, but there are tools
- solution are taylored for each application

CLOSING REMARKS

- parallelism is necessary for large-scale computations
- can be tricky on legacy codes, but there are tools
- solution are taylored for each application

Thank you!