Dynamic Mesh Adaptation of massive unstructured grids for the simulation of flame fronts and gas/liquid interfaces

V. Moureau, **P. Bénard, K. Bioche, J. Carmona, L. Voivenel, Y. Béchane**
CORIA, CNRS UMR6614, Normandie Univ, UNIROUEN, INSA Rouen, France

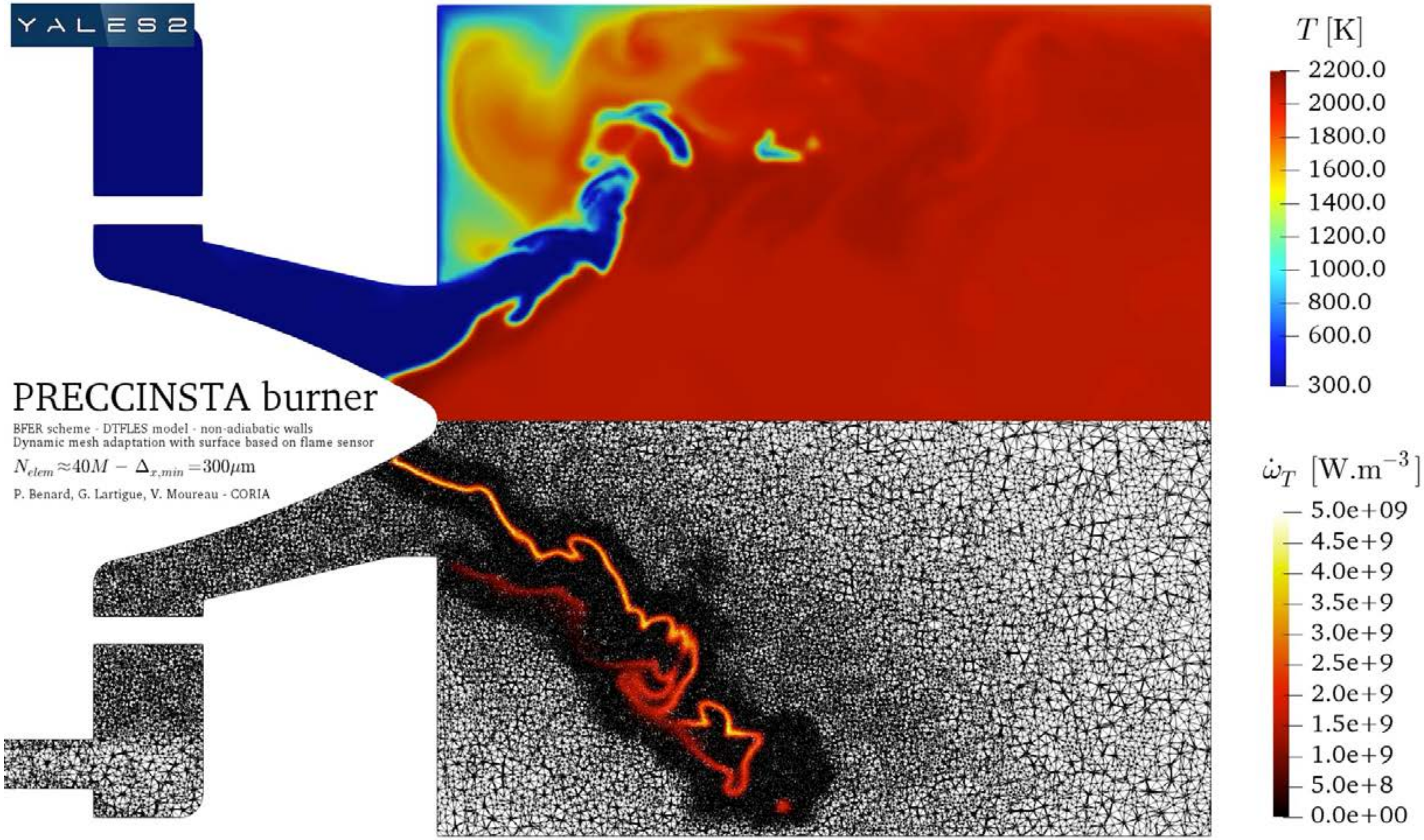
P. Begou, G. Balarac, **G. Ghigliotti,**– LEGI, Grenoble, France

A. Froehly, C. Dobrzynski – LMB/INRIA Bordeaux, France

R. Mercier, M. Cailler, J. Leparoux, R. Letournel
SAFRAN Tech, Magny-les-H., France

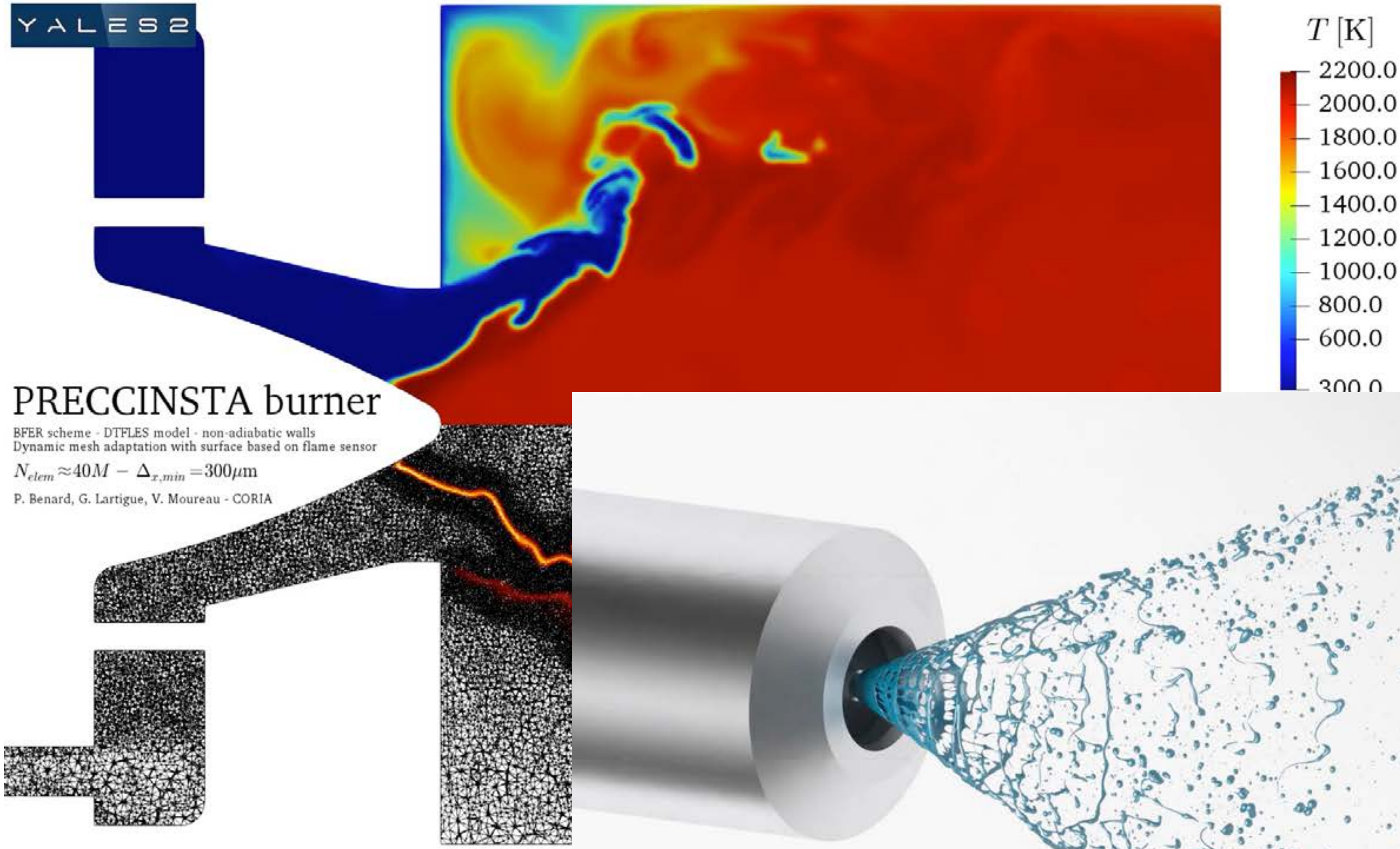
NUMPEX workshop, 6-7 février 2024

Motivation



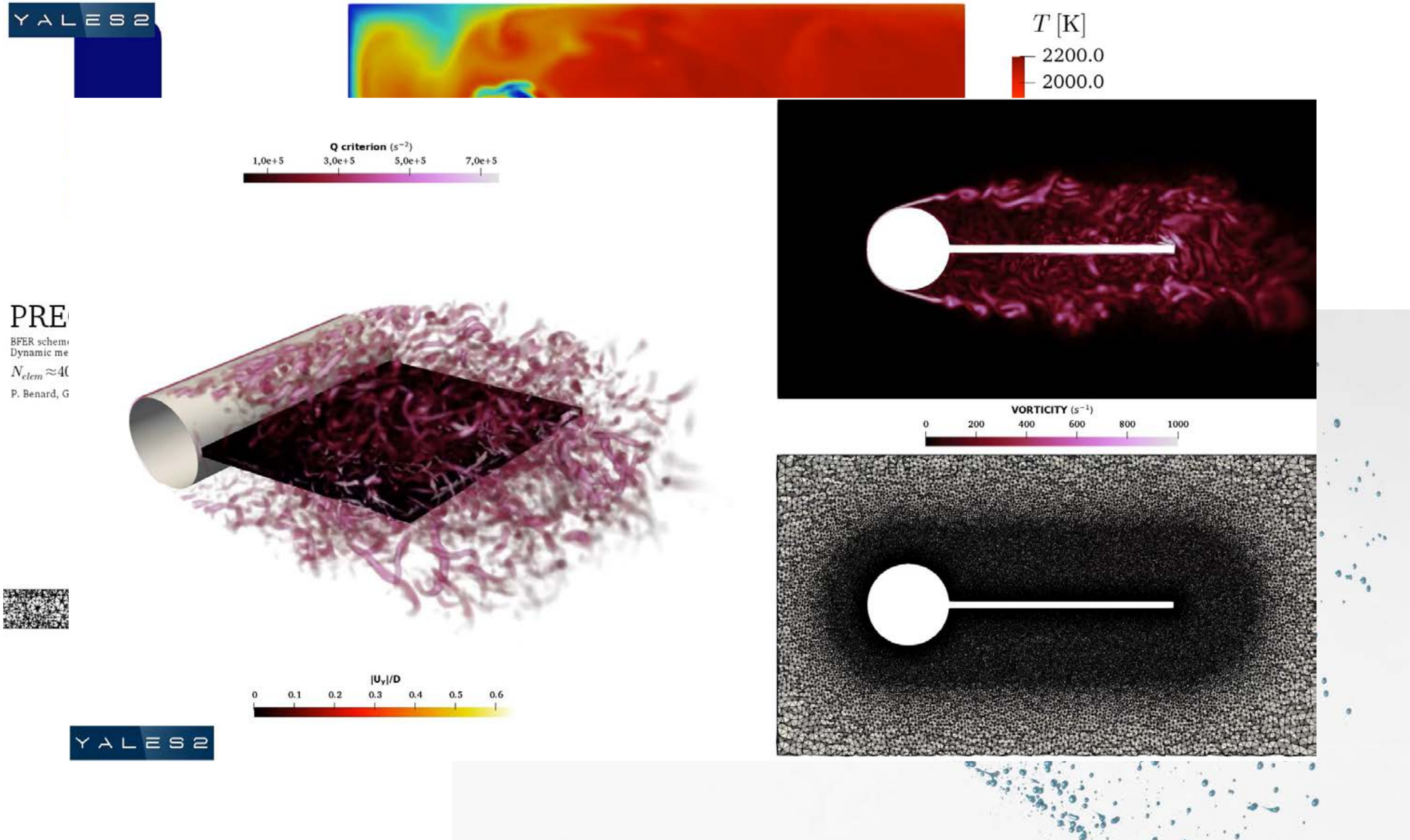
YALES2
PRECCINSTA burner
BFER scheme - DTFLES model - non-adiabatic walls
Dynamic mesh adaptation with surface based on flame sensor
$N_{elem} \approx 40M - \Delta_{x,min} = 300\mu m$
P. Benard, G. Lartigue, V. Moureau - CORIA
T [K]
2200.0
2000.0
1800.0
1600.0
1400.0
1200.0
1000.0
800.0
600.0
300.0
$\dot{\omega}_T$ [W.m^{-3}]
5.0e+09
4.5e+9
4.0e+9
3.5e+9
3.0e+9
2.5e+9
2.0e+9
1.5e+9
1.0e+9
5.0e+8
0.0e+00

Motivation



Courtesy J. Vernier, J. Leparoux, R. Mercier, SAFRAN TECH

Motivation



Courtesy J. Vernier, J. Leparoux, R. Mercier, SAFRAN TECH

Objectives

- To design a **dynamic mesh adaptation** method under constraints:
  - Based on **unstructured grids** for complex geometries
  - Tailored for **distributed-memory** system (10,000+ cores)
  - **Efficient** enough to be called every 10 fluid iterations
  - Compatible with the modeling approach (**finite-volume + LES**)
  - No remeshing at material interfaces to avoid interpolation errors

- Some choices
  - **Isotropic** mesh adaptation (for now)
  - Parallelism handled by the flow solver (interpolation, data transfer)

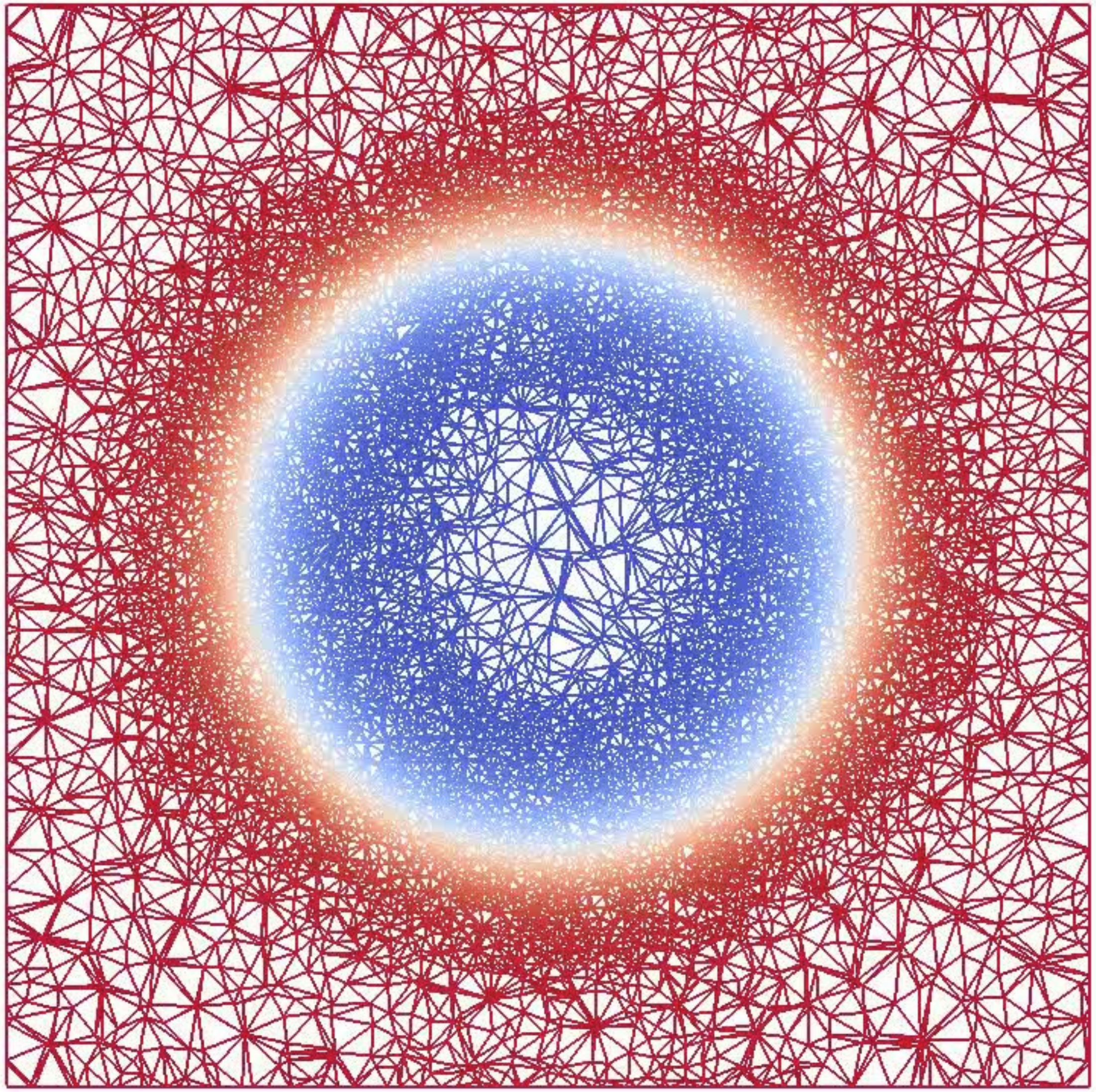
- A first check if these objectives are reachable for tets
  - Reduced computation cost of fluid solver: 30 to 500 μs/iter/node
  - Adaptation of a distributed mesh with **MMG**: order of 100 μs/node

mmgtools.org

Upgrade your meshes

A first attempt

- January 2015: first use of MMG directly in YALES2

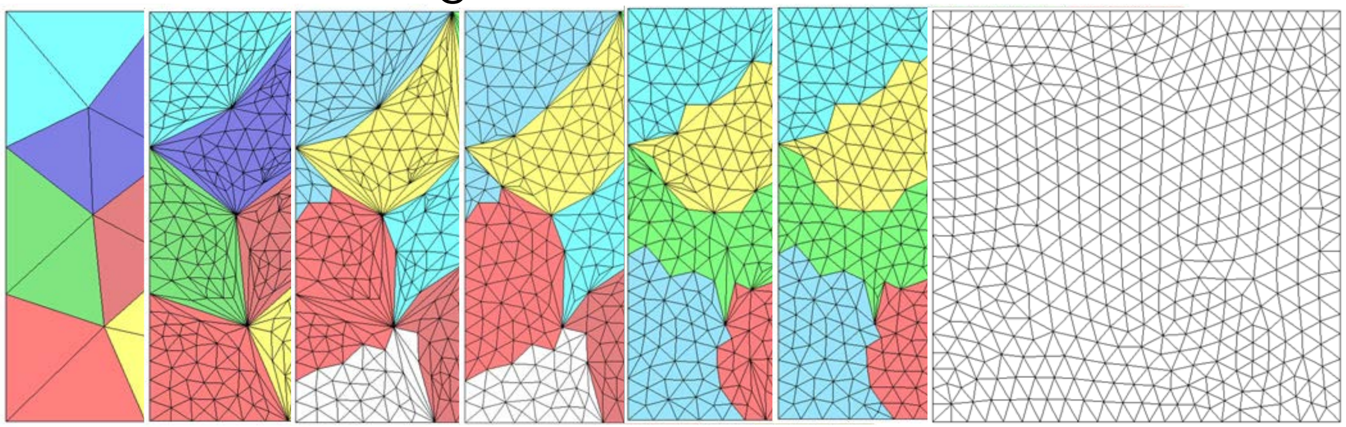


High-performance dynamic mesh adaptation

- A combination of several kernels

Moving interface method [1,2]

$\mathcal{M}_{\text{init}}$

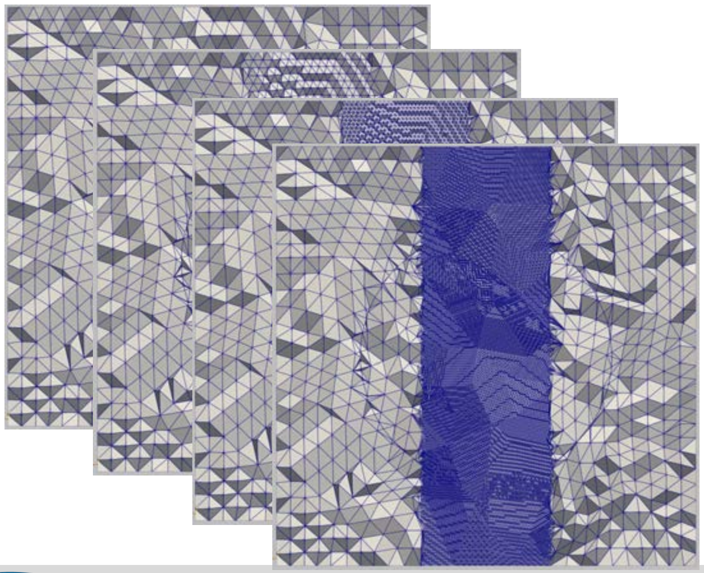


$\mathcal{M}_{\text{target}}$

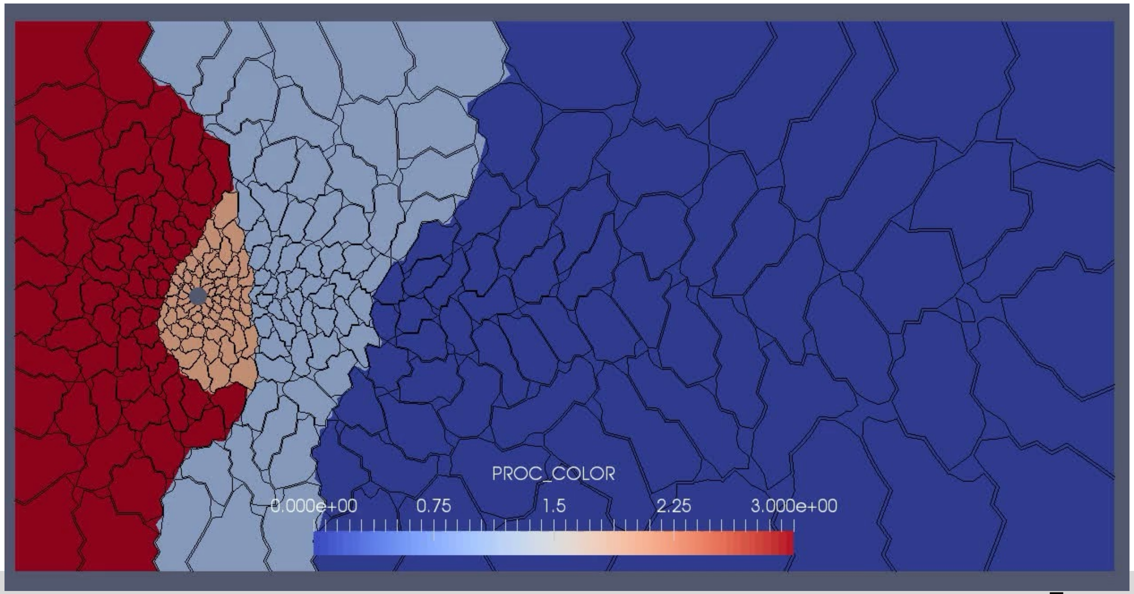
upgrade your meshes

mmgtools.org

Parallel edge-cutting



Parallel load-balancing



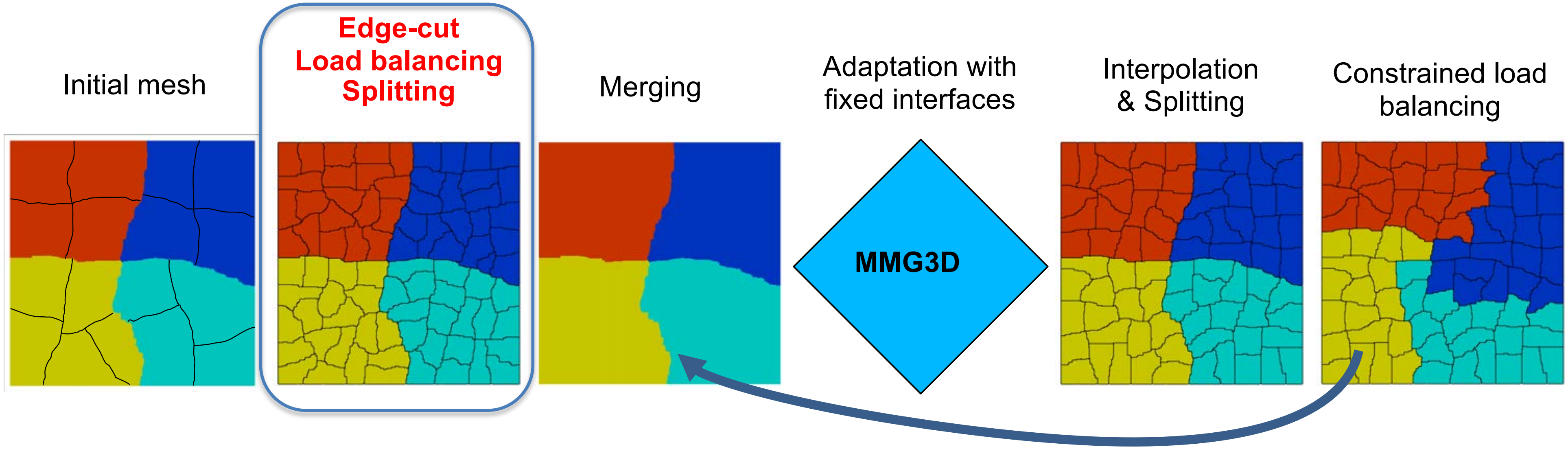
[1] Digonnet et al., IJHPCA, 2017
[2] Balarac et al., Maths in Action, 2022

High-performance dynamic mesh adaptation

www.mmgtools.org

Upgrade your meshes

- Full adaptation workflow
  - Example for 4 processors

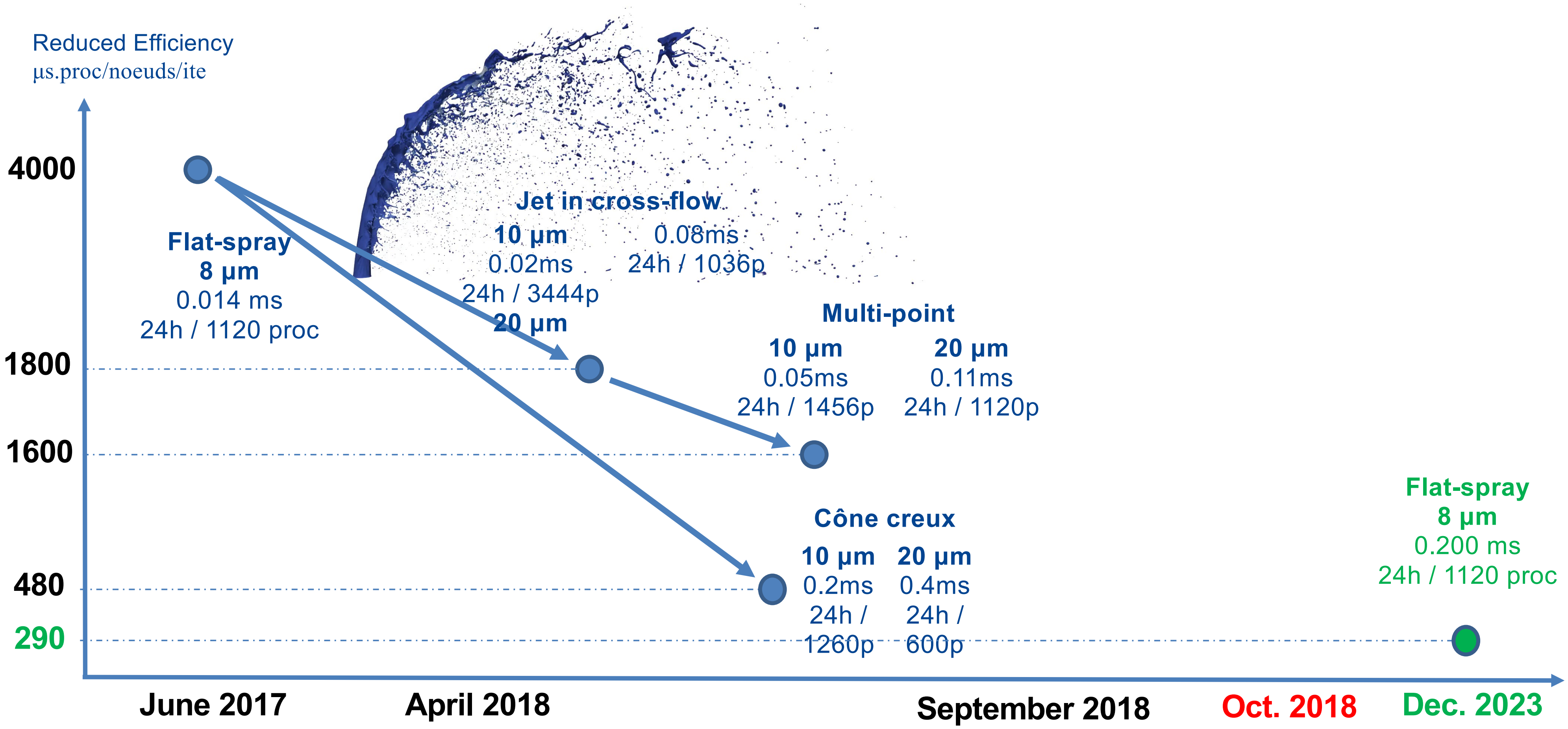


if $Sk > Sk_{max}$ or $\frac{|\mathcal{M} - \mathcal{M}_{\text{target}}|}{\min(\mathcal{M}, \mathcal{M}_{\text{target}})} > \mathcal{E}_{\mathcal{M}}$

- Skewness is the main measure of the mesh quality

[1] Bénard et al., IJNMF, 2015 [2] Dapogny et al., JCP 2014

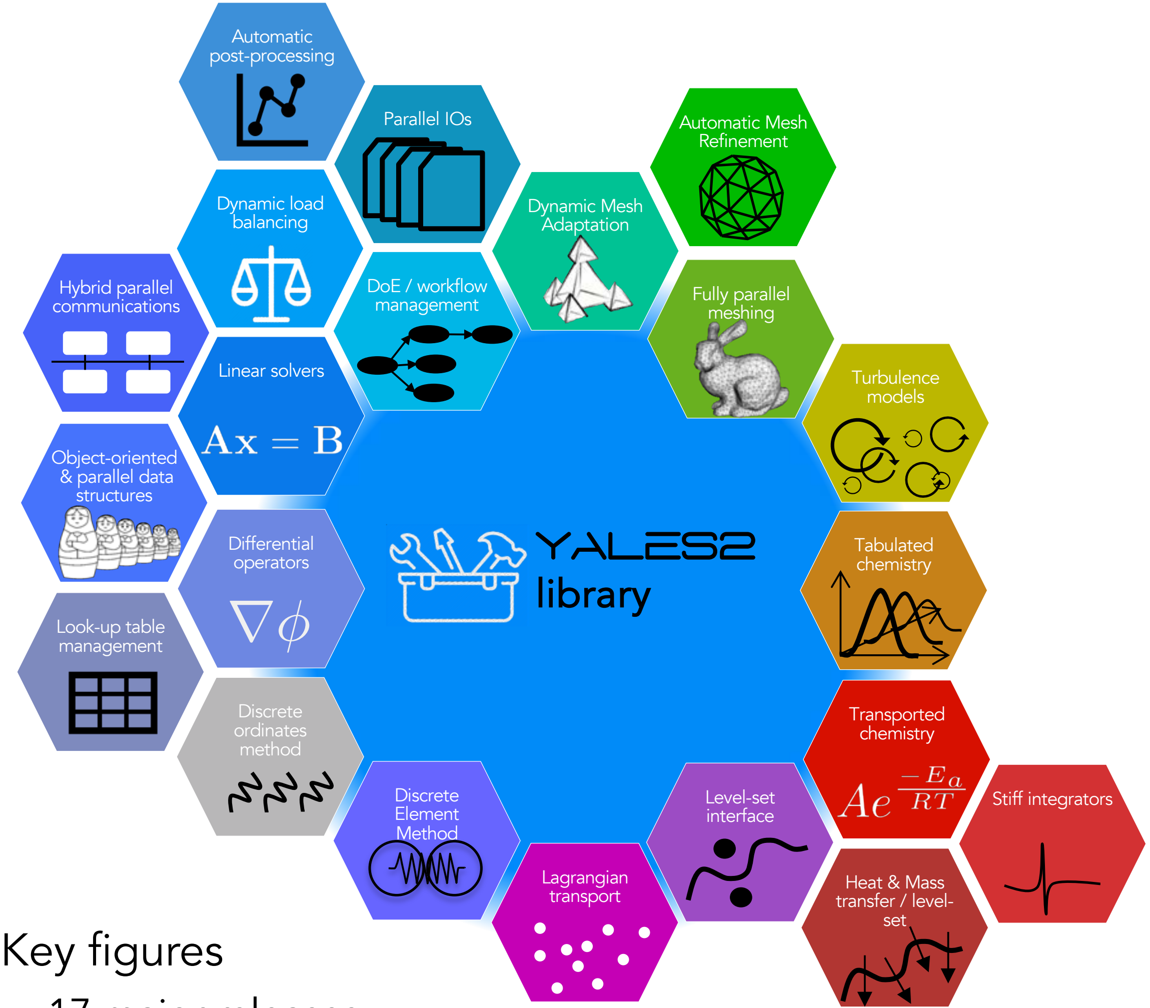
Performance evolution for atomization problems



Courtesy R. Mercier, SAFRAN TECH

CFD platform

The CFD platform



- Key figures
  - 17 major releases
  - 930 000 object-oriented F2008 lines
  - 20 000+ commits
  - 10 to 20 commits/day
  - 300+ active branches
  - 1 200+ merge requests
  - 120+ contributors

YALES2 solvers

ICS Incompressible at constant density

VDS Incompressible at variable density

SPS Spray with level-set and ghost-fluid method

ALE Arbitrary Lagrangian Eulerian

GFS Granular flow with Discrete Element Method

HTS Heat transfer

MHD Magneto-hydro-dyn

SMS+FS Structural mech.

ACS Acoustics

BOI Boiling

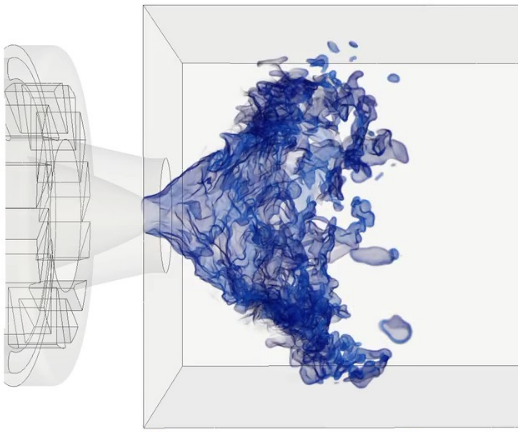
ECS Compressible flow

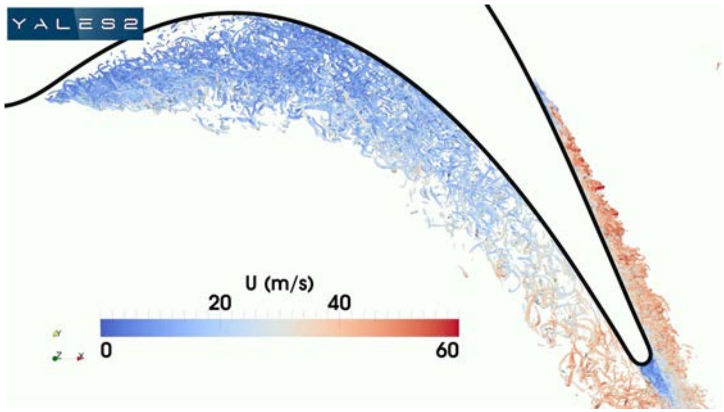
YALES2 features

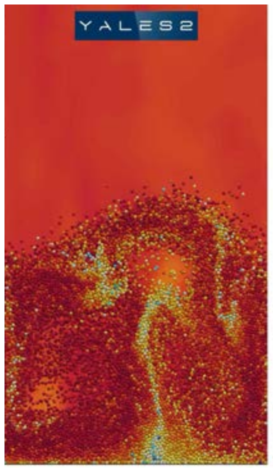
- High fidelity
- Multiphysics
- High performance

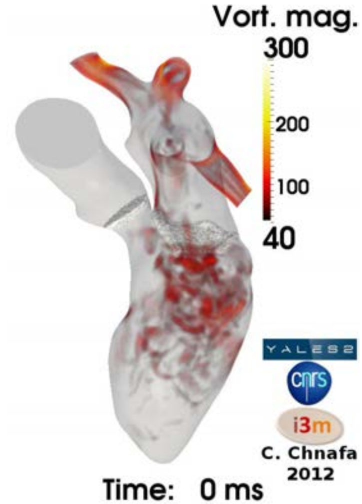
CFD platform: YALES2

- Features
  - **Unstructured meshes** and **adaptive grid refinement**
  - Low-Mach number Navier-Stokes equations (incompressible and variable density)
  - **4-level domain decomposition** [3] and hybrid OpenMP/MPI communications
  - Highly efficient solvers for linear system inversion (PCG, DPCG) [4]
  - 4th-order central **finite-volume method** and 4th-order time integration
  - Two-phase flows (Lagrangian particles), **spray and atomization** (Levelset)
  - Combustion modeling (Tabulated or **finite-rate chemistry**, NOx model, …)
  - Suited for massively parallel computing (>32 000 procs)









[1] YALES2 web site, http://www.coria-cfd.fr
[2] SUCCESS web site, http://success.coria-cfd.fr
[3] Moureau et al., CR Mecanique, 2011
[4] Malandain et al., JCP, 2013

The YALES2 network

- Developed by CORIA, the French Combustion Community and others
  - 600+ researchers/engineers trained at CORIA since 2009
  - 200+ articles (Google Scholar)
- A unique network to ease collaboration and disseminate numerics, algorithms and models to the community

Academic partners

CORIA, IMAG, LEGI, EM2C, IMFT
CERFACS, ONERA, IFP-EN, LMA, LJK
LOMC, PPRIME, LMB/INRIA, ICARE

ULB, UMONS, UCL, VUB, TU DELFT
CORNELL U., VERMONT U.
SHERBROOKE U.

HPC centers

CRIANN, IDRIS, CINES, TGCC
ROMEO, GENCI, PRACE

Industrial partners

SAFRAN
ARIANE GROUP
GE HYDRO
SIEMENS/GAMESA
AIR LIQUIDE
TOTAL ENERGIES
…

YALES2

www.coria-cfd.fr

HPC experts

INTEL/CEA/GENCI/UVSQ, HPE
ATOS, IBM, AMD, NVIDIA

SMEs

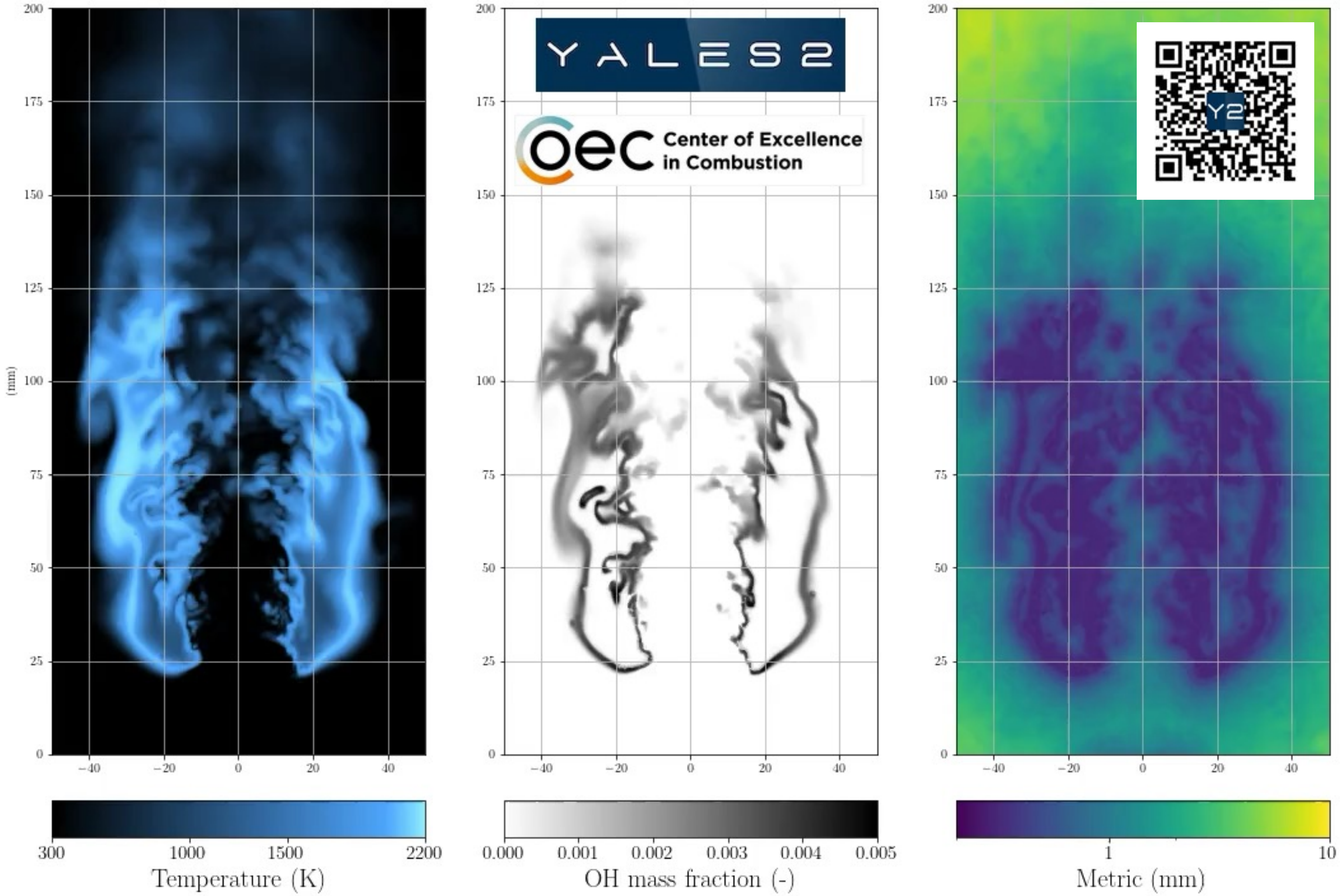
GDTech
YPSO FACTO

The YALES2 network

A few applications

DMA of spray combustion

- LES simulations with finite-rate chemistry using dynamic adaptive mesh refinement.
  - Chemistry: ARC for n-heptane (CERFACS), 25 species, 210 reactions
  - Spray: Lagrangian Particle with dynamic load balancing (Stock et al., IJNMF 2023)



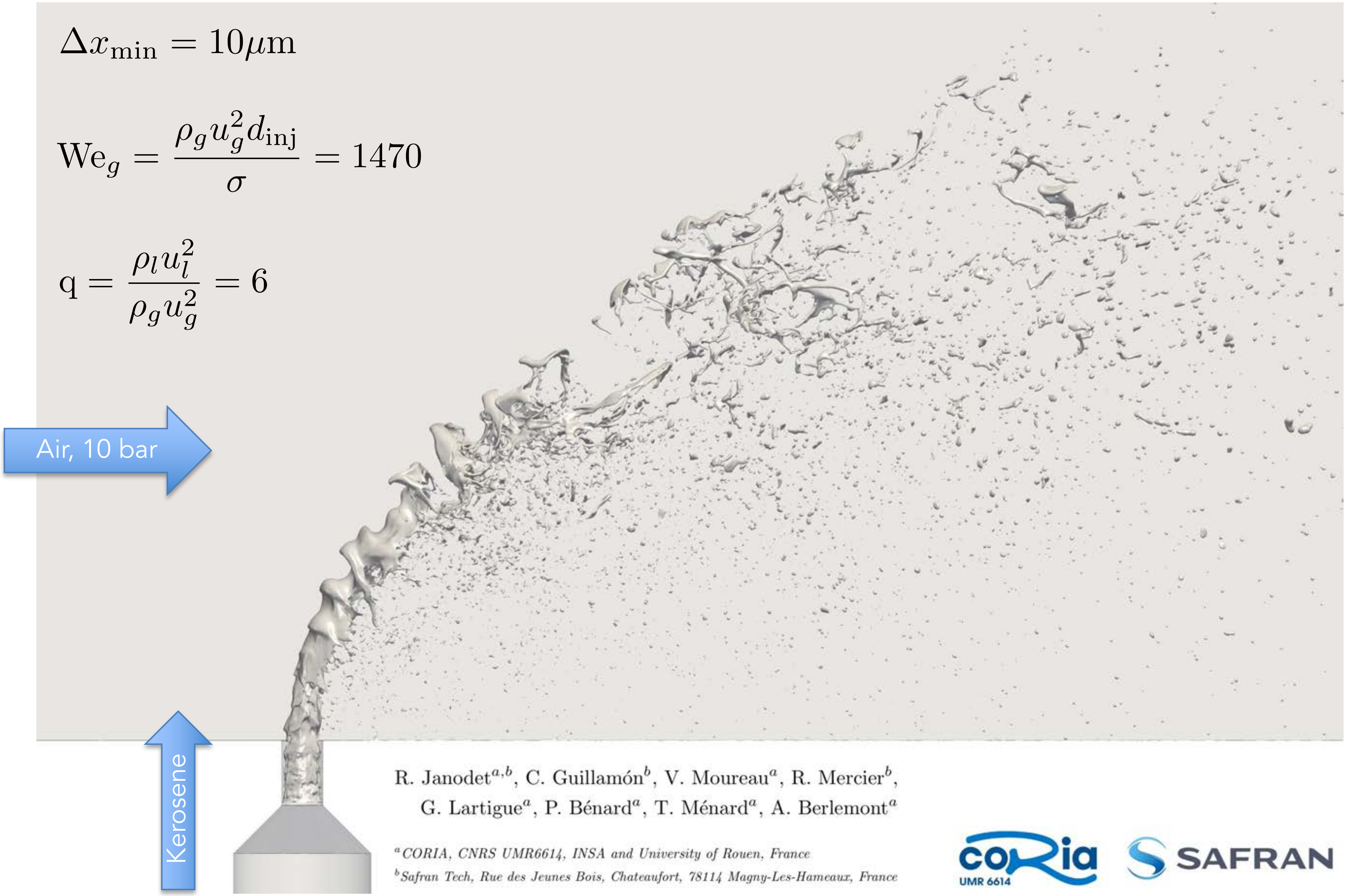
$\Delta x_{min} = 200\ \mu m$,
300M cells
10,000 cores
Jean-Zay @ IDRIS

DLR jet-in-cross-flow [1]

$$\Delta x_{\min} = 10\mu\mathrm{m}$$

$$\mathrm{We}_g = \frac{\rho_g u_g^2 d_{\mathrm{inj}}}{\sigma} = 1470$$

$$\mathrm{q} = \frac{\rho_l u_l^2}{\rho_g u_g^2} = 6$$



R. Janodet[a,b], C. Guillamón[b], V. Moureau[a], R. Mercier[b], G. Lartigue[a], P. Bénard[a], T. Ménard[a], A. Berlemont[a]

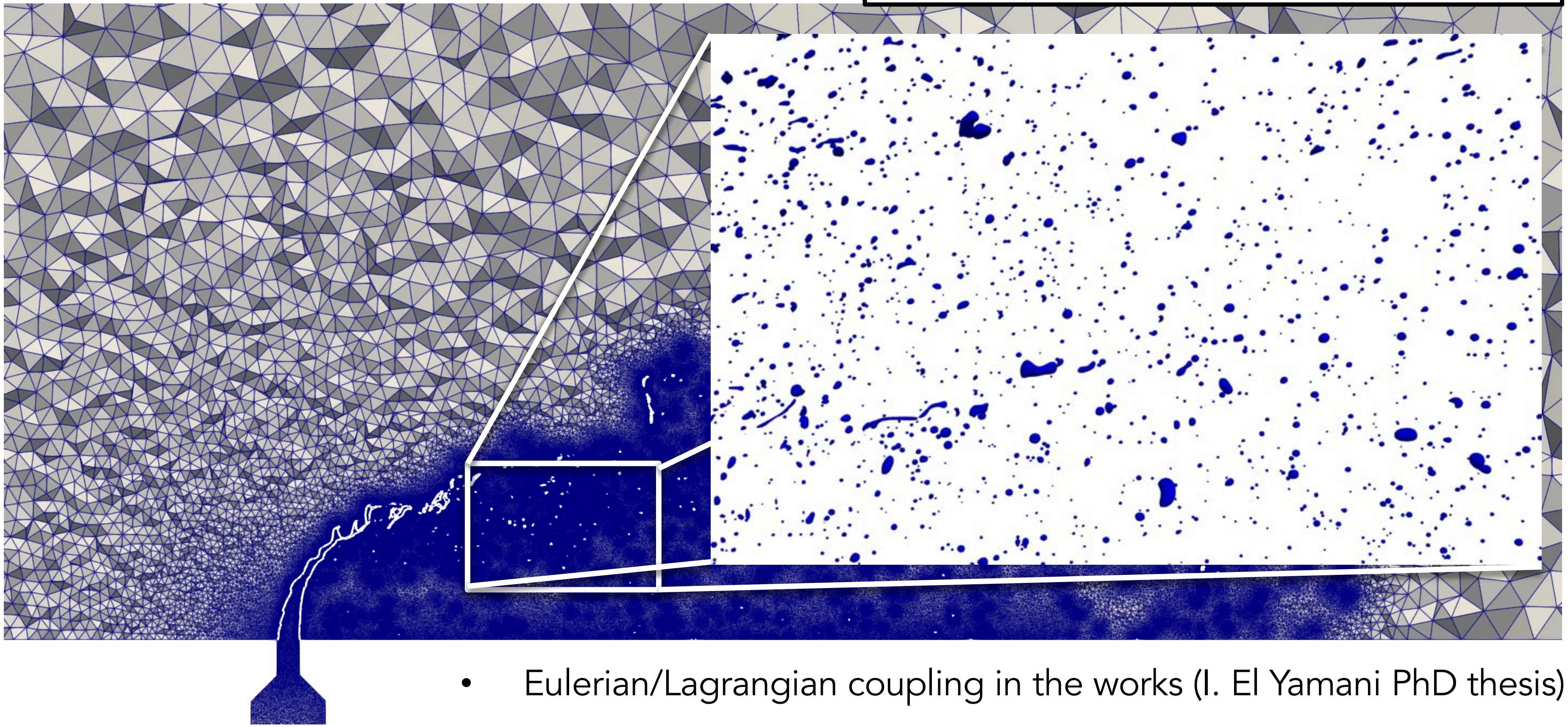
[a] CORIA, CNRS UMR6614, INSA and University of Rouen, France

[b] Safran Tech, Rue des Jeunes Bois, Chateaufort, 78114 Magny-Les-Hameaux, France

DLR jet-in-cross-flow [1]

- Flow topology analysis

$\Delta x = 10\ \mu m$, 1B cells, 15 600 liquid structures
10 000 cores of Irene AMD, TGCC, CEA



- Eulerian/Lagrangian coupling in the works (I. El Yamani PhD thesis)

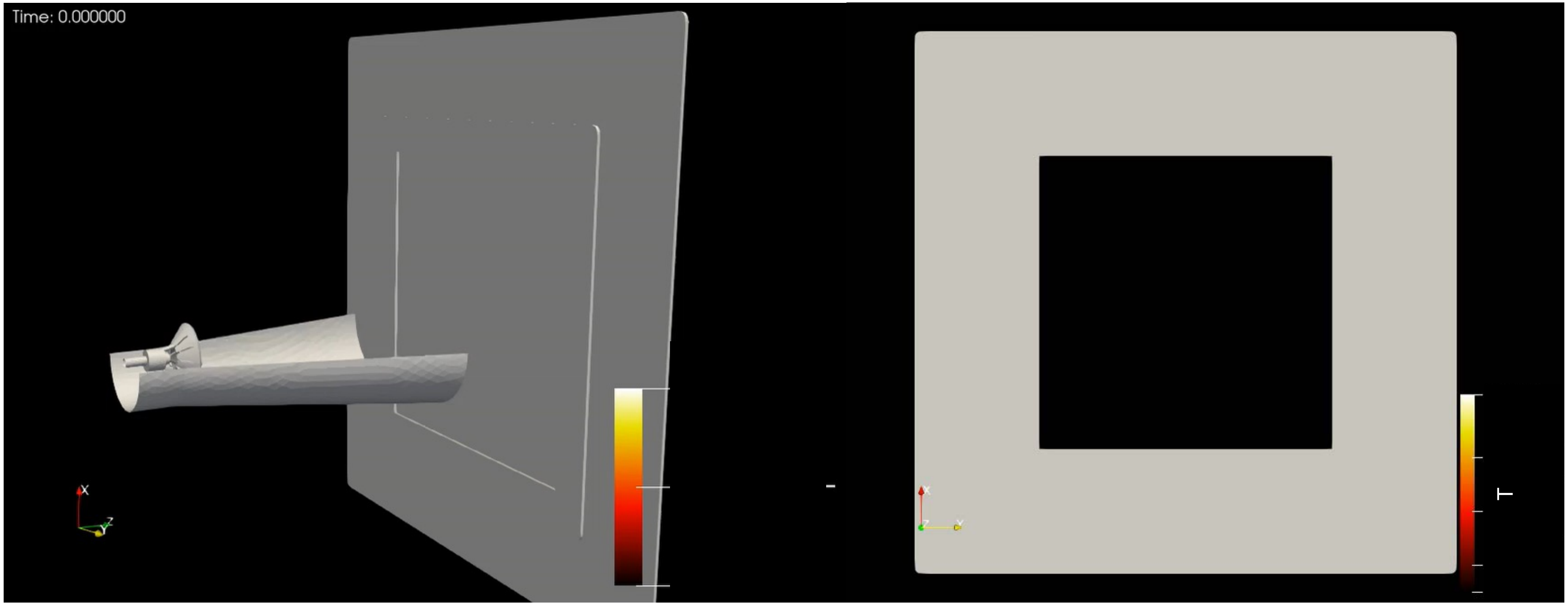
[1] J. Becker, C. Hassa, Atomization and Sprays 11 (2002) 49-67.

Modeling of fire resistance tests for composite materials

- Kerosene spray burner
  - BFER mechanism for kerosene, 6 species, 2 reactions
- Coupling of 3 solvers
  - combustion, conduction, radiative heat transfer (Discrete Ordinate Method)

CWIPI

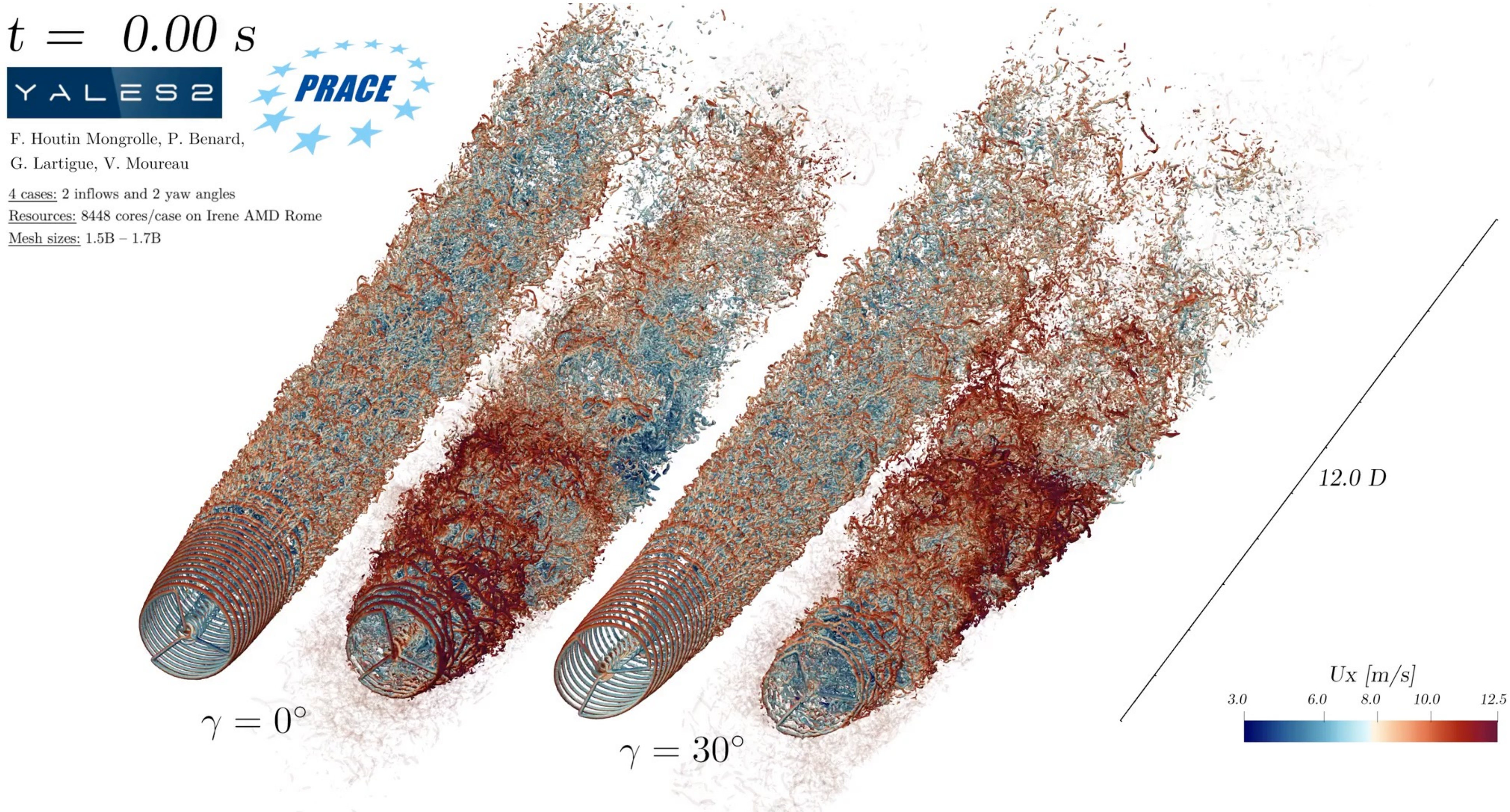
ONERA THE FRENCH AEROSPACE LAB



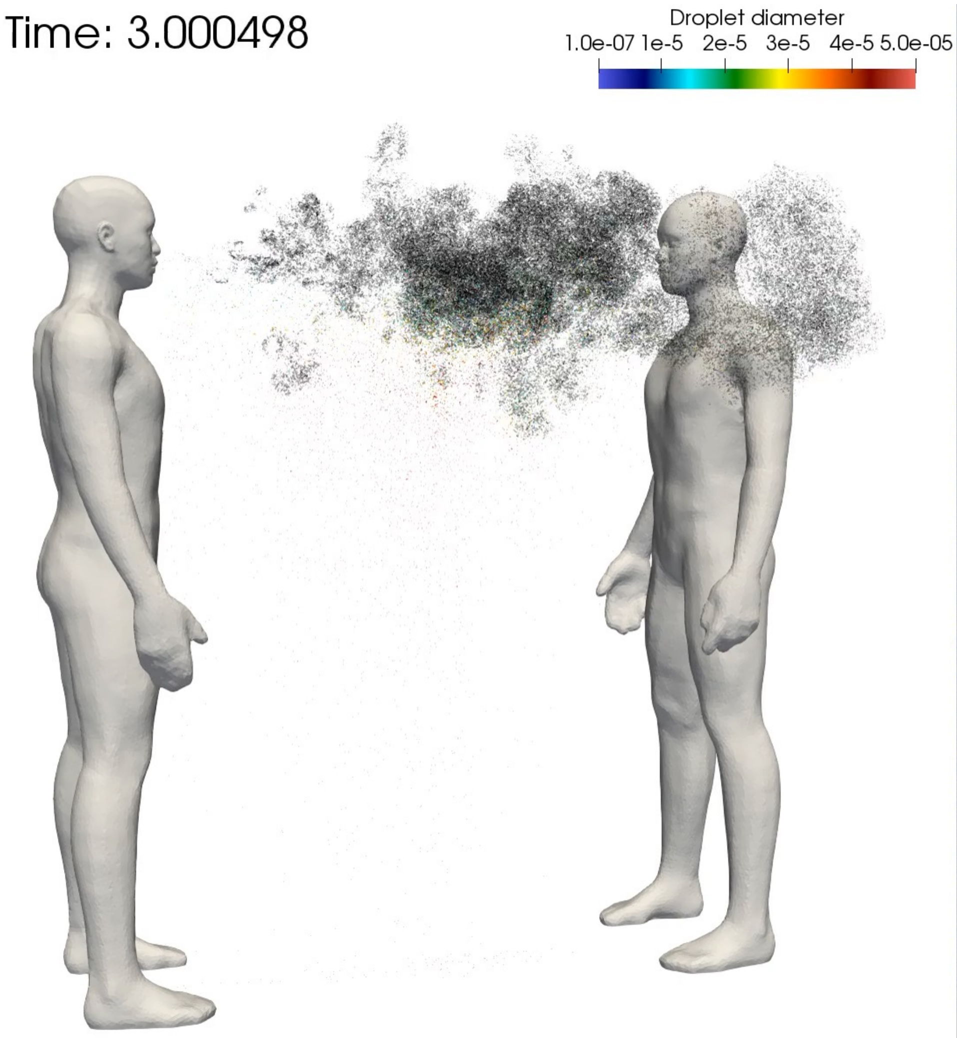
Courtesy R. Letournel, SAFRAN TECH

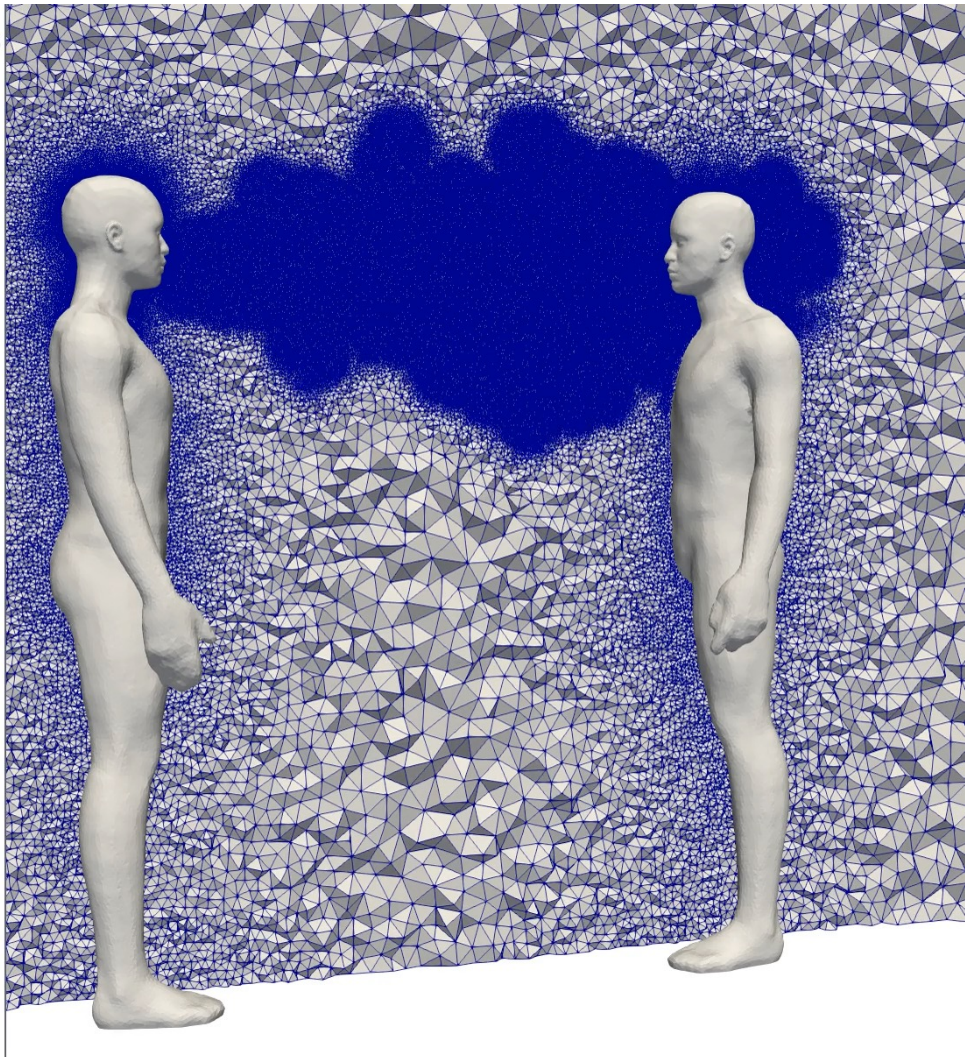
Wind turbines

- Impact of yaw on wake develpoment behind offshore wind turbines
- Collaboration with SIEMENS/GAMESA Renewable Energies



Simulations in the framework of the COVID pandemic

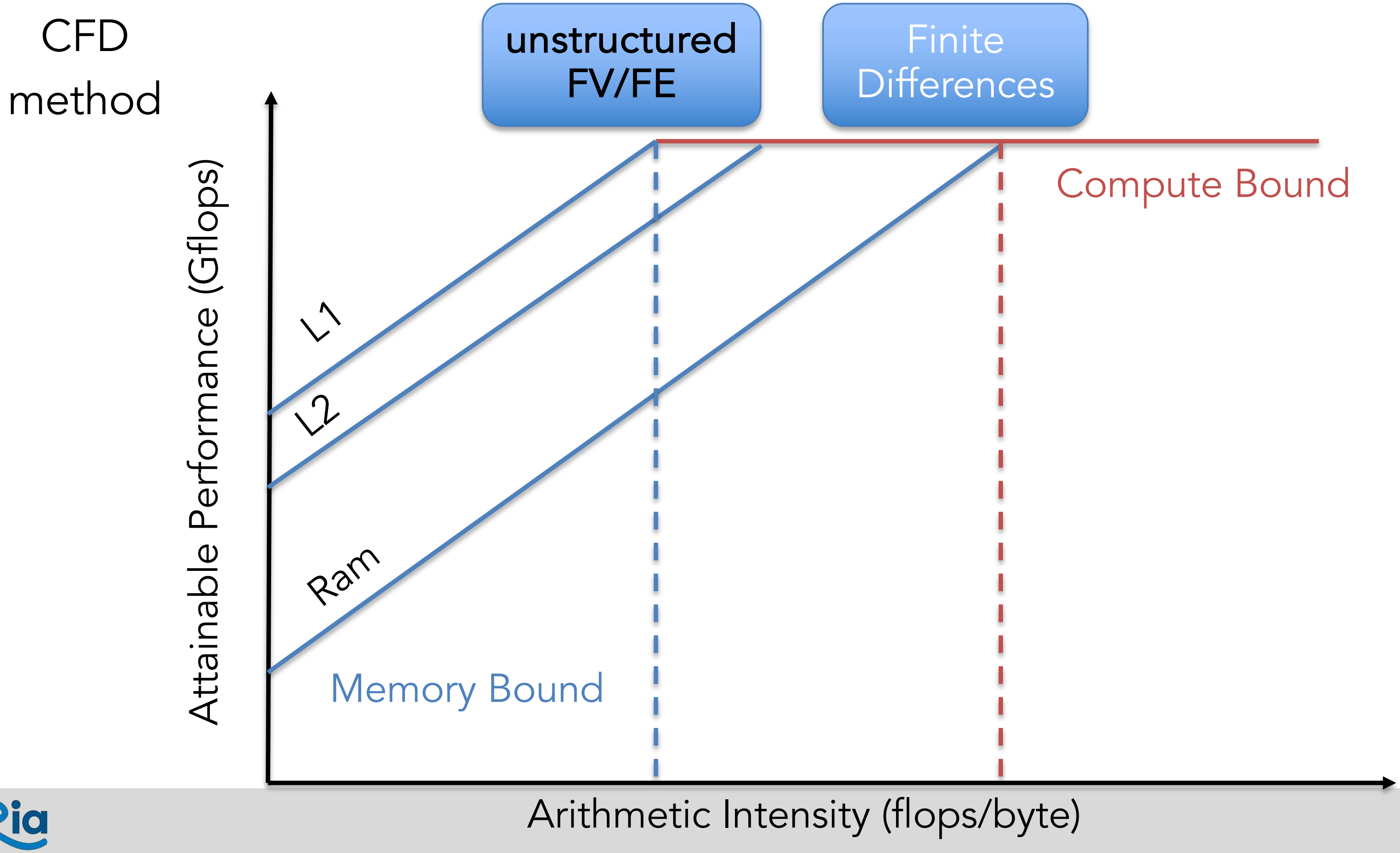




Performance optimization

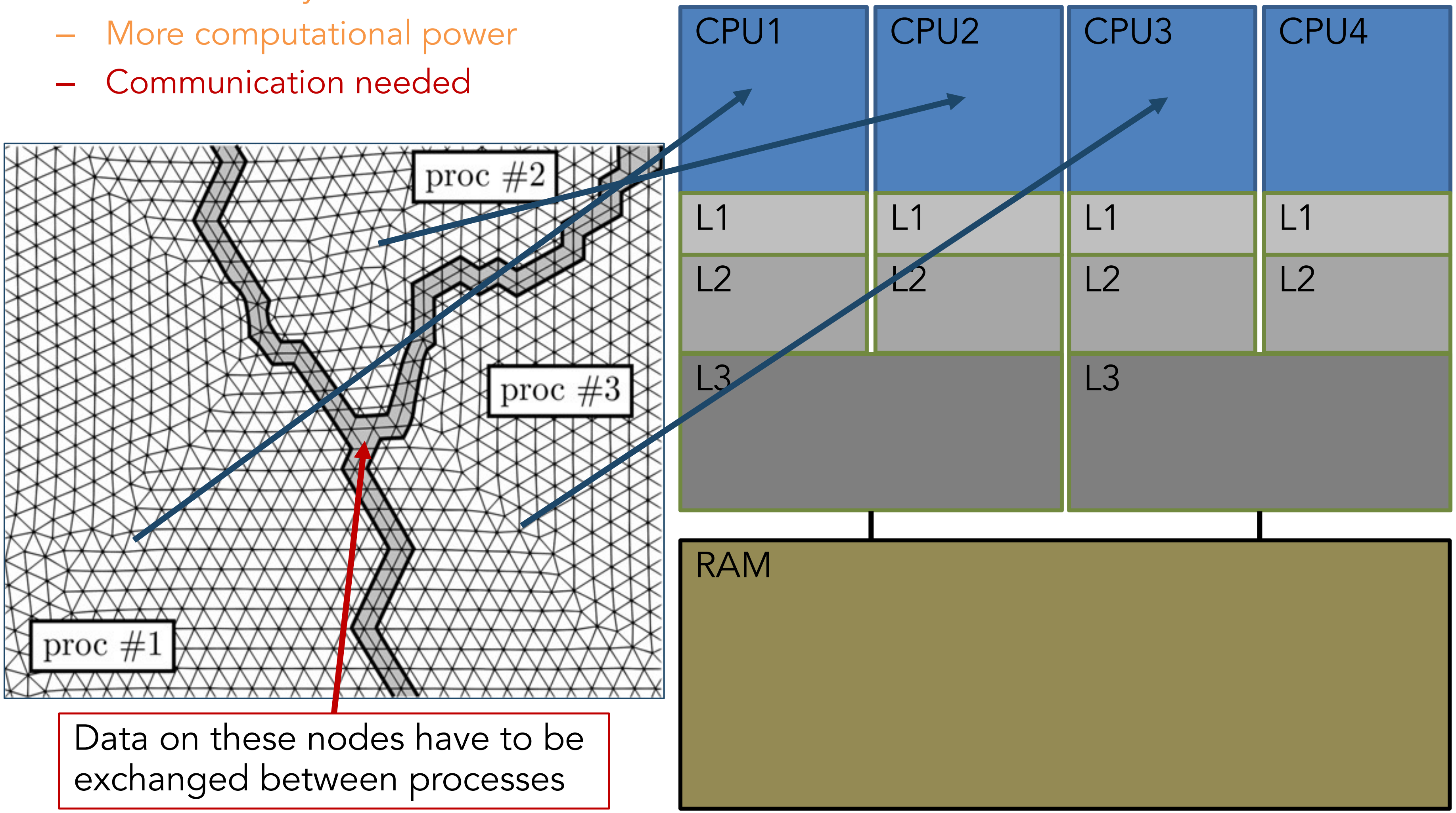
Importance of memory accesses in unstructured FV codes

- Code performance on a CPU can be limited by:
  - Processor speed (compute bound)
  - **Memory access speed** (memory bound)
  - The roofline model



CPU: Parallel computation and domain decomposition

- Large problems can not be computed by a single process
- Domain decomposition to divide the problem amongst many processes
  - More memory available
  - More computational power
  - Communication needed

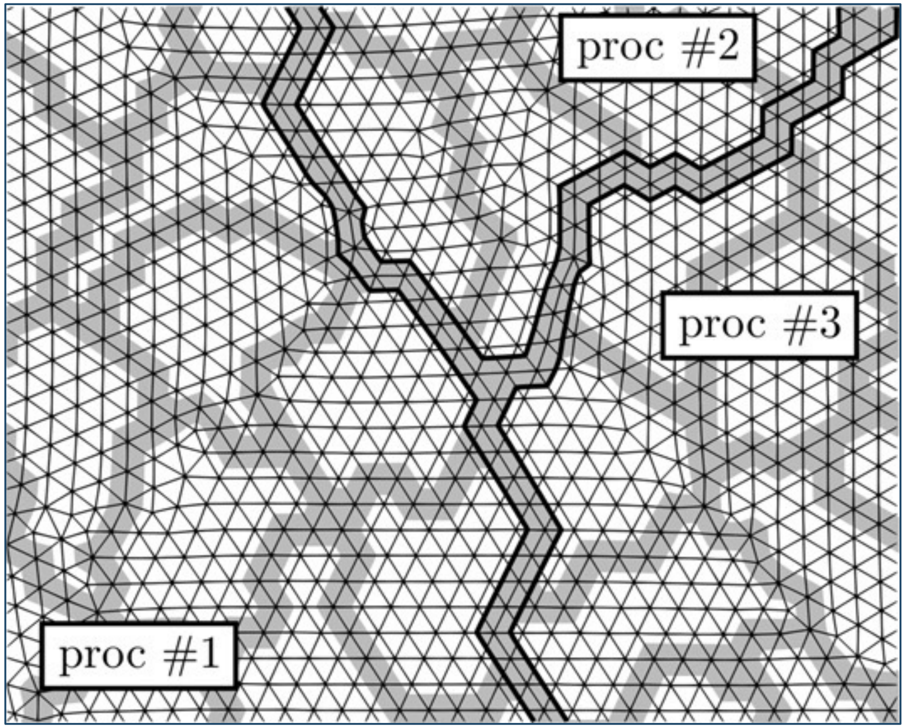


Data on these nodes have to be exchanged between processes

MPI/OpenMP + cache-blocking: 3-level domain decomposition

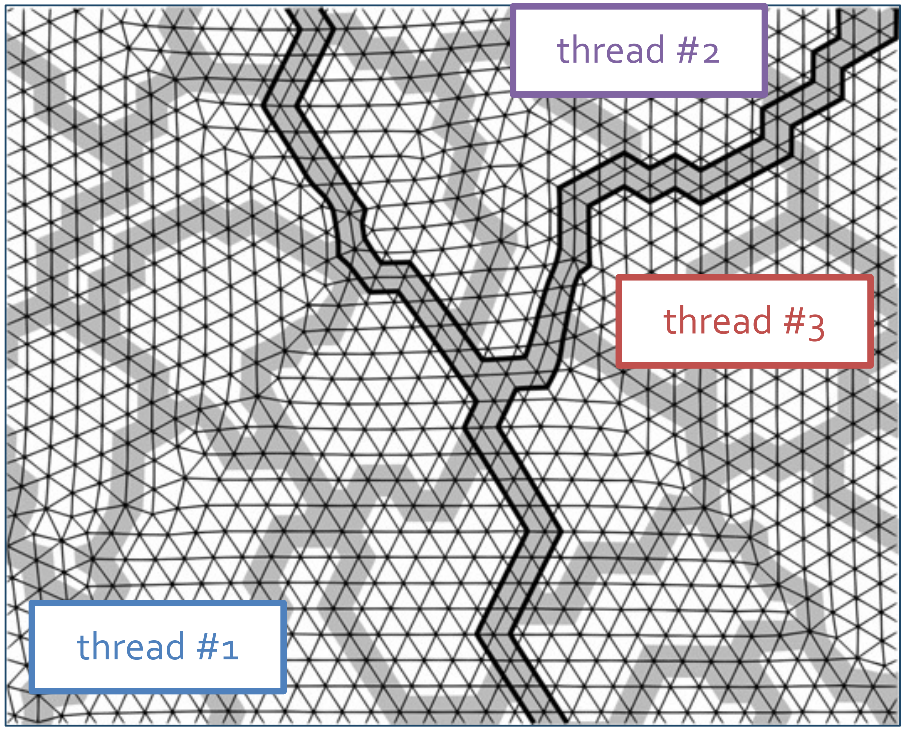
MPI + OpenMP + in-thread domain decomposition

Full MPI [1]



- Cell groups for cache-blocking
- Cell group size ~ 2000
- **Overhead: node duplication**

Coarse-grain Hybrid OpenMP + MPI

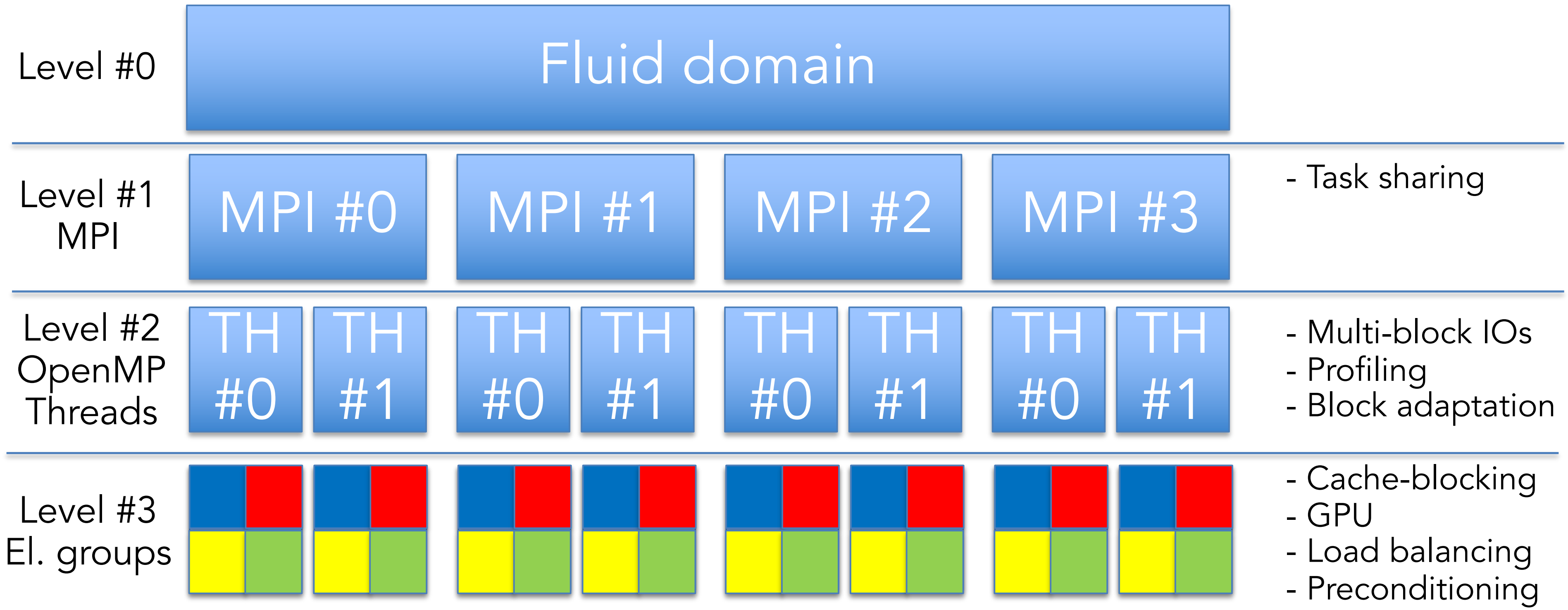


- Threads substitute MPI ranks
- Fewer MPI ranks but thread-safety
- Overhead:
  - Threads must communicate
  - Node duplication

[1] Moureau et al., CR Mecanique, 2011

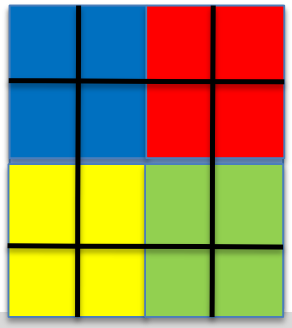
Hierarchical domain decomposition in YALES2

- Explicit partitioning



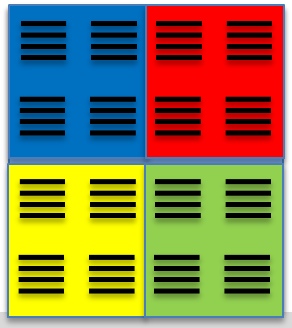
- Implicit partitioning

Level #4
Sub. el.
groups



- Interpolation
- Part. localization

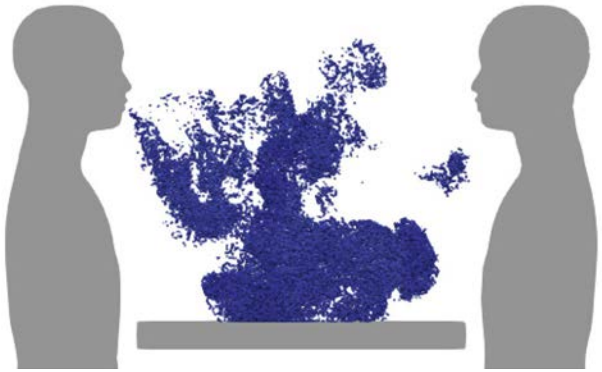
Level #5
Pair/node/
particle
groups

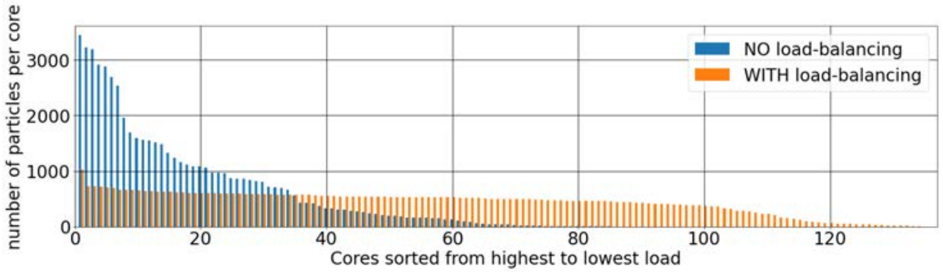


- Vectorization
- Graph operators

Dynamic load balancing for Euler/Lagrange models

- A difficult task which requires: mesh coloring, data movement, …
- Balancing cells and particles requires 2-constraints optimization [1]

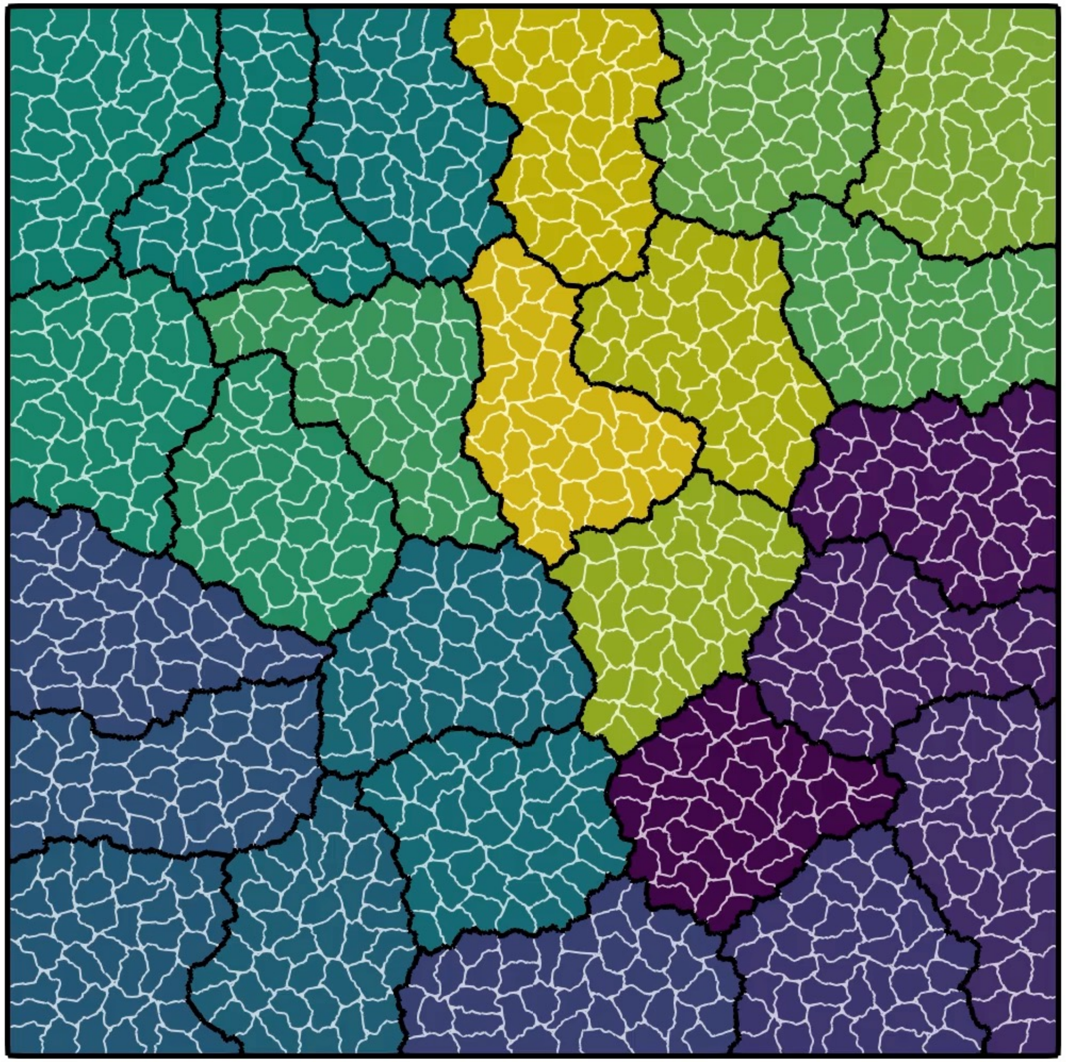




Center of Excellence in Combustion

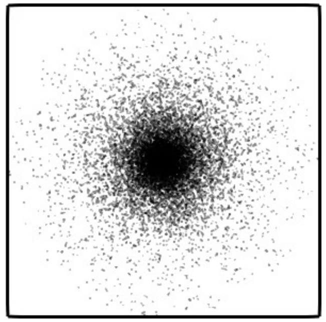
[1] Diffusive orthogonal load balancing for Euler–Lagrange simulations, A. Stock, G. Lartigue, V. Moureau, IJNMF 2023

Dynamic load balancing for Euler/Lagrange models

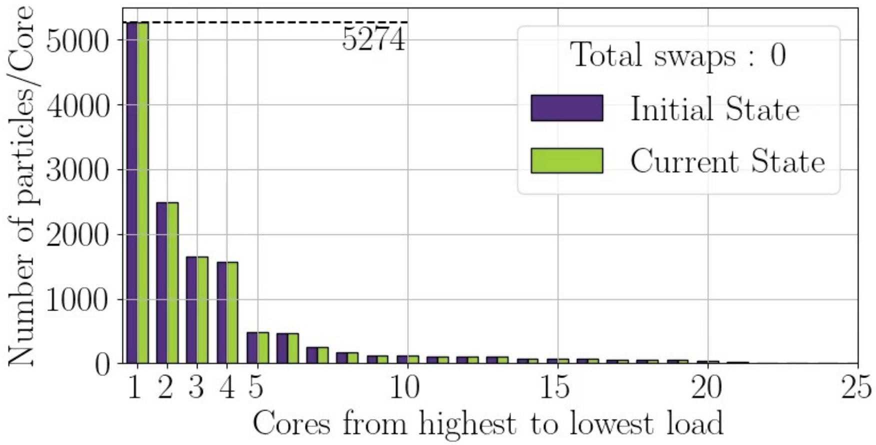


A. Stock, G. Lartigue,
V. Moureau

A core and
its swappable subparts



Lagrangian particle
distribution



GPUs & APUs

- Working on OpenACC port since 2017
  - Benefits from array objects (r1_t, r2_t, …, i1_t, i2_t, …)
  - GPU memory management in the data structures

```
!============================================================
!# <STRUCTURE NAME=r2_t>
!# <DESCRIPTION>Pointer for two-dimension real variables.</DESCRIPTION>
type r2_t
  real(WP), allocatable :: val(:,:) !# <VAR NAME=val TYPE=real(:,:)>Values</VAR>
  character(len = LEN_MAX) :: name  !# <VAR NAME=name TYPE=character>Name of the pointer</VAR>
  integer :: name_len               !# <VAR NAME=name_len TYPE=integer>Name length</VAR>
  integer :: hash                   !# <VAR NAME=hash TYPE=integer>Hash</VAR>
  integer :: flag                   !# <VAR NAME=flag TYPE=integer>Flag</VAR>
  logical :: used                   !# <VAR NAME=used TYPE=logical>In use flag</VAR>
  integer :: dim1                   !# <VAR NAME=dim1 TYPE=integer>First dimension</VAR>
  integer :: dim2                   !# <VAR NAME=dim2 TYPE=integer>Second dimension</VAR>
  integer :: allocdim1              !# <VAR NAME=allocdim1 TYPE=integer>First dimension allocation</VAR>
  integer :: allocdim2              !# <VAR NAME=allocdim2 TYPE=integer>Second dimension allocation</VAR>
  integer :: exchange               !# <VAR NAME=exchange TYPE=integer>To exchange or not</VAR>
  integer :: ptr_index              !# <VAR NAME=ptr_index TYPE=integer>Pointer index at creation</VAR>
  type(r2_t), pointer :: next       !# <VAR NAME=next TYPE=r2_t>Next pointer</VAR>
end type r2_t
!# </STRUCTURE>
!============================================================
```

```
  if (acc_is_present(r2%val)) then
     !$acc update device(r2)
     !$acc enter data create(r2%val)
     !$acc parallel loop collapse(2) present(r2%val,r2_tmp) copyin(new_allocdim1,new_allocdim2)
     do j=1,new_allocdim2
        do i=1,new_allocdim1
           r2%val(i,j) = r2_tmp(i,j)
        end do
     end do
     !$acc exit data delete(r2_tmp) finalize
   end if
```

GPUs & APUs

- Status of OpenACC port in YALES2_2024.04
- One solver is ported (SCS), two main solvers on-going (ICS, ECS)
- Supported on
  - NVIDIA, Jean-Zay with gfortran and nvfortran
- Support to come
  - Adastra AMD MI250: SCS port is done with cce17 but results are still wrong
  - AMD MI300A (APU): CPU port done during ECFD7 Jan. 2024

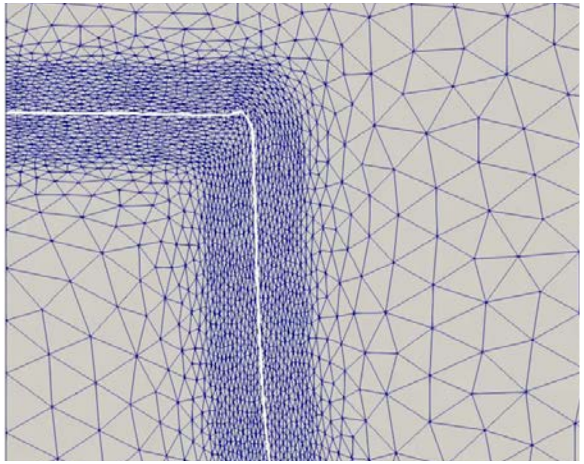
Conclusions

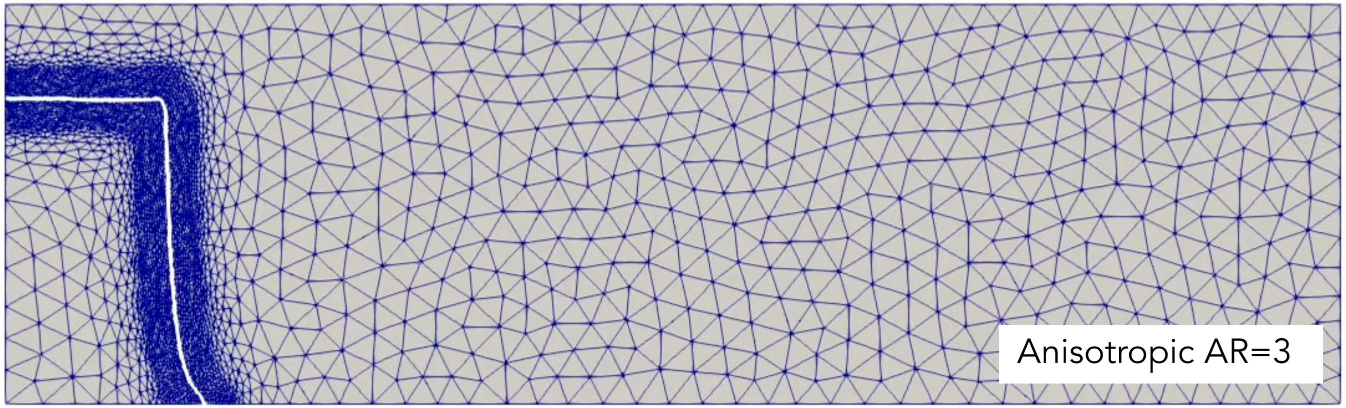
Conclusions

- Data partitioning is the key of efficient unstructured adaptive mesh refinement.

- Many algorithms still need improvement
  - Multi-level contiguous parallel graph coloring
  - Collapse/face swap parallel kernels

- On-going work: anisotropic AMR





Mesh generation

- Arbitrary STL mesh adaptation