An overview of the application of HPC in CEA codes for simulating nuclear reactors

Ansar Calloo
CEA
Direction des EnergieS (DES)
Institut des Sciences Appliquées et de la Simulation aux énergies bas carbone (ISAS)
Département de Modélisation des Systèmes et Structures (DM2S)

Table of contents

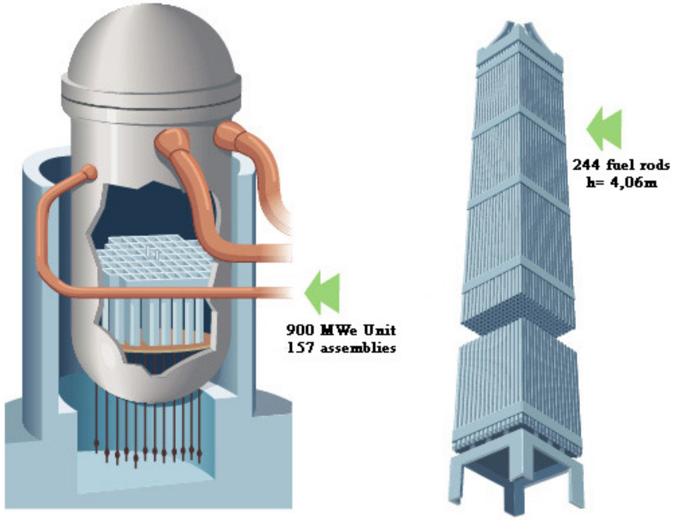
1. **Context**
2. **Overview of some codes**
   - TRUST
   - Neutronics: Monte Carlo and deterministic
   - Mechanics
   - Code coupling strategy
3. **Performance portability with Kokkos**
   - CExA
   - Kokkos programming model, HPSF
   - NumPEx
4. **Concluding remarks**

1. Context

Context

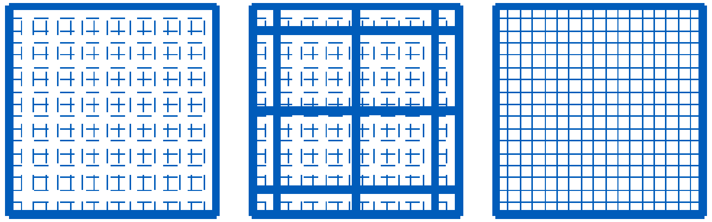
- Reactor physics is a highly multiphysics field involving nuclear physics, neutronics, thermalhydraulics, structural mechanics, material science, ….

- Industries rely on highly verified and validated code systems to perform routine computations based on low-order approximations for optimal precision to computing time ratio.
  - Parametric computations to sweep a very large domain for design and operational studies
  - ex: two-scale (assembly and core) approach in neutronics, 1D simplified thermalhydraulics

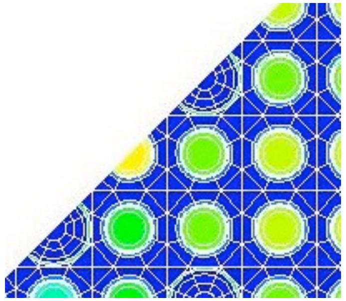


REACTOR PRESSURE VESSEL FUEL ASSEMBLY

- High-fidelity simulations are required to provide sufficient validation on low-order model validity to safety authorities for usual Pressurised Water Reactor from the nuclear fleet or to design new reactors such as Small Modular Reactors, Advanced Modular Reactors, …etc.
  - CFD for mixing grid studies or transport computation in neutronics

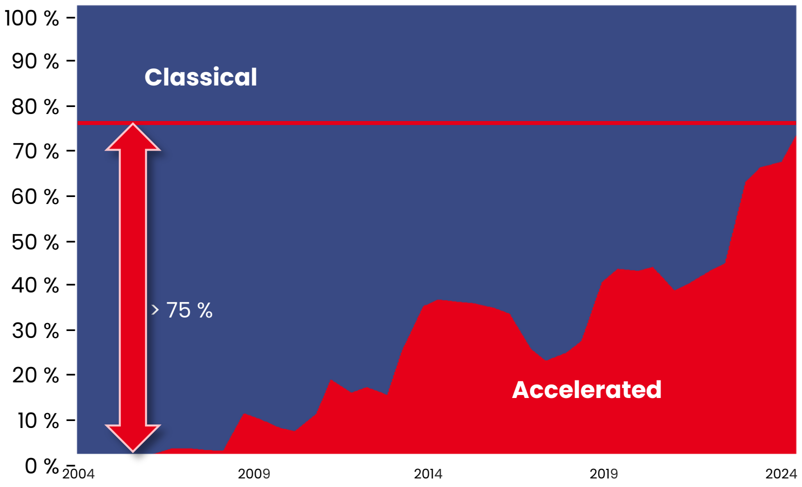


Homogeneous Domain Homogeneous Pin-by-pin

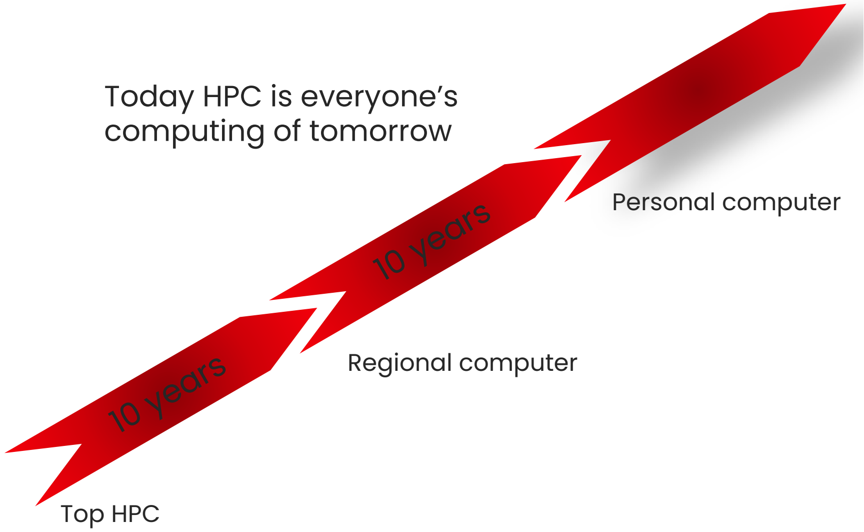


Context

- HPC is a tool in a wide range of domains, **source of competitivity**
  - At CEA, we host machines
  - To take part in the French & European HPC ecosystem
- We just entered the **Exascale** era, that means **GPU**
  - European pre-Exascale systems: Mix of AMD & Nvidia
  - First Exascale machines planned in Europe for 2024/2025
    - Jupiter machine at Jülich (Germany) => Nvidia & Rhea
    - Alice Recoque machine at CEA/TGCC (open call)
  - Need to adapt or re-develop applications with **Performance portability**
- GPU programming models: software catalysts
  - France and Europe had great research but no production tool
- A need for a **long-term sustainable solution**
  - Adapted to **our hardware and software specificities**
  - **Trust** in the roadmap

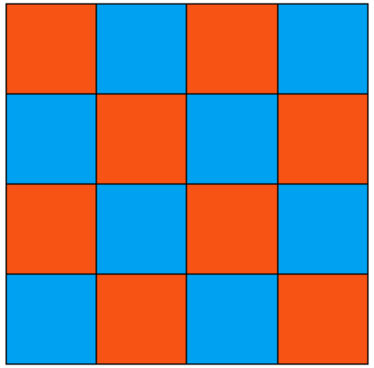


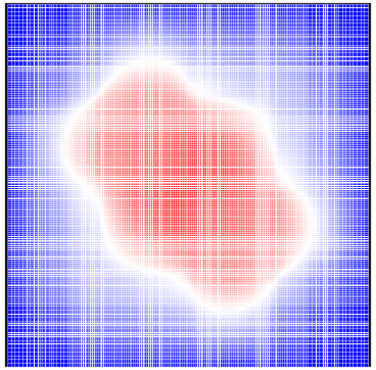
Computing power of the 500 top supercomputers from june 2004 to june 2024 (source Top500)

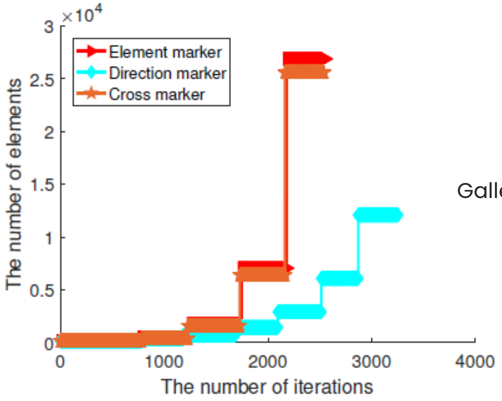


High-fidelity simulations

- Distinguish between particle-based *vs.* mesh-based approaches
  - e.g. Monte Carlo and deterministic methods for neutronics
- For instance, starting point for Monte Carlo is a CAD-like geometry model. This model is meshed for the mesh-class family of methods
- Pre-processing: meshes
- Online stage:
  - Numerical schemes set up for nuclear reactor simulation
    - High-order methods
    - Adaptive methods (AMR)
    - ….
  - Numerical strategies for solving the problem
- Post-processing: Error estimates
- Computer science: algorithms and parallel programming models





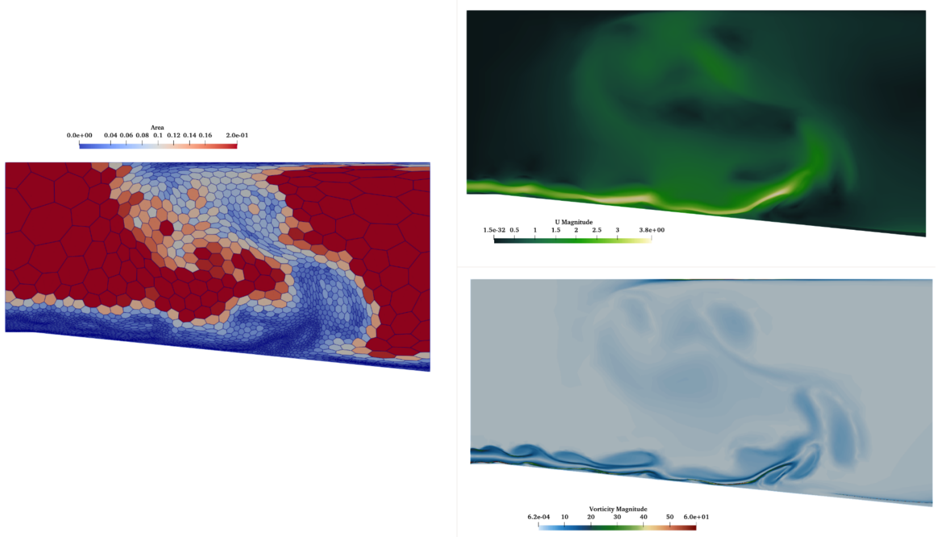


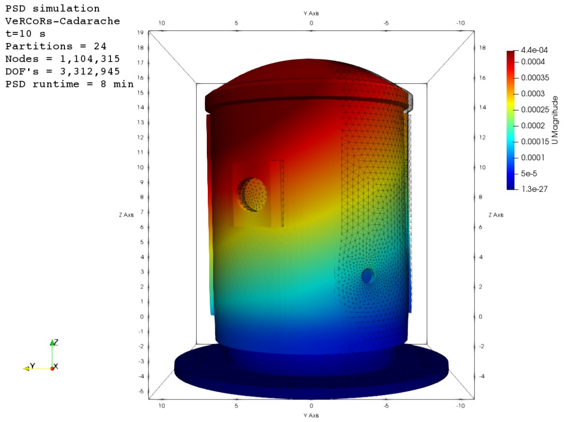
Gallery courtesy: F. Madiot

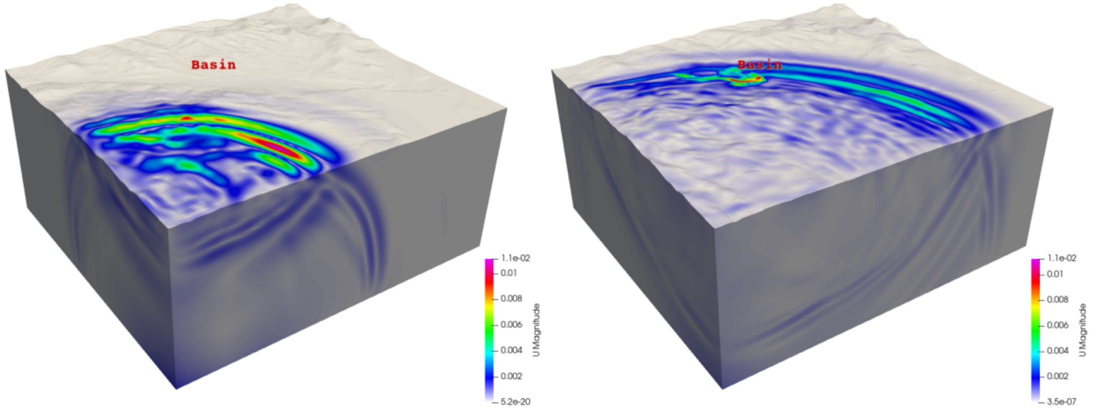
2. Overview of some codes

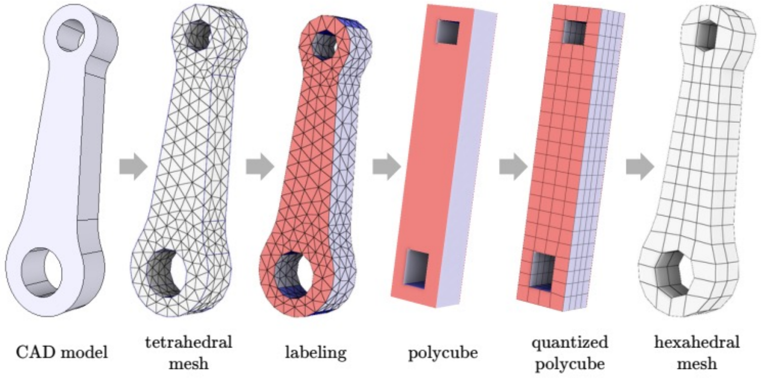
Meshes: towards more complex elements

- Meshes: first part of the model
  (except particle-based methods, *cf*. previous slide)
- Mesh tool within the SALOME environment
  - Co-developed with EDF
  - SMESH: usual simplices
- Polygonal/polyhedral meshes
  - New R&D endeavours
  - Improve accuracy *vs* computational cost ratio
- On-going R&D: Hexahedral meshes using AI algorithms to convert to polycubes
- Recent work: generation of a 6 B-cell mesh with parallel IO management (SNA + MC 2024)





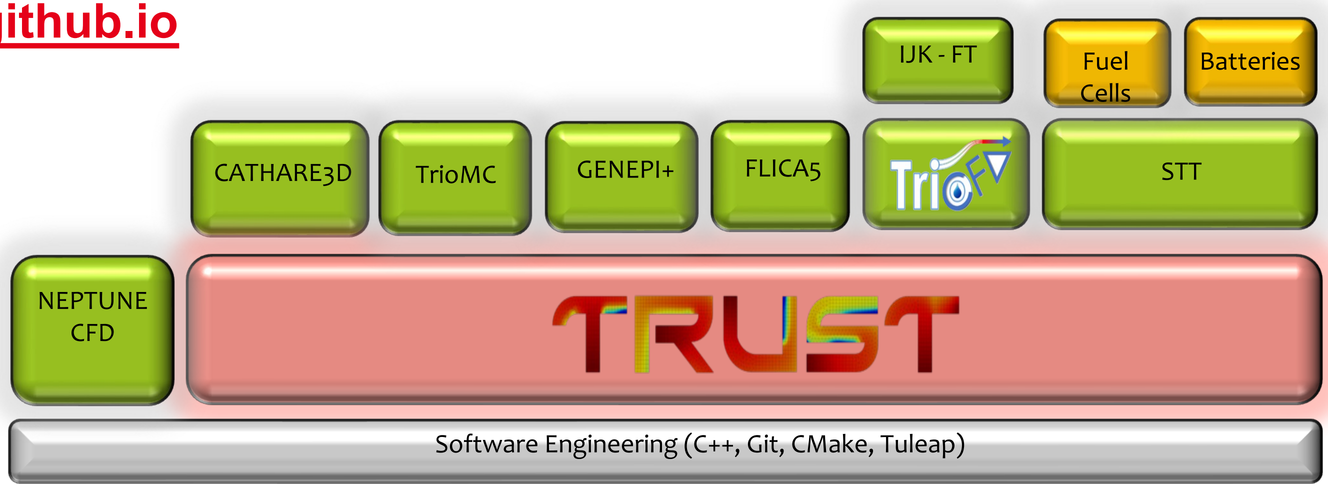




Gallery courtesy: M. A. Badri

TRUST: a brief overview

- TRUST: HPC thermalhydraulic platform for multiphase problems
  - More than 20 years of development
  - Born out of the splitting of Trio_U into TRUST and TrioCFD
  - **Opensource**: https://cea-trust-platform.github.io
  - Initially, incompressible Navier-Stokes
  - Now multiphase and compressible
  - Laminar/Turbulent flow
  - Spatial discretisations: VDF, VEF, MAC, PolyMAC



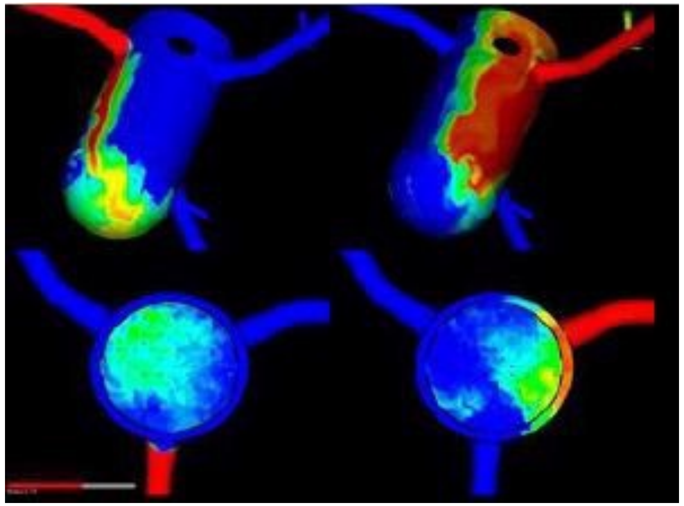
- Several *applications* are developed on the TRUST platform
  - Derived physics application using TRUST as a core building block

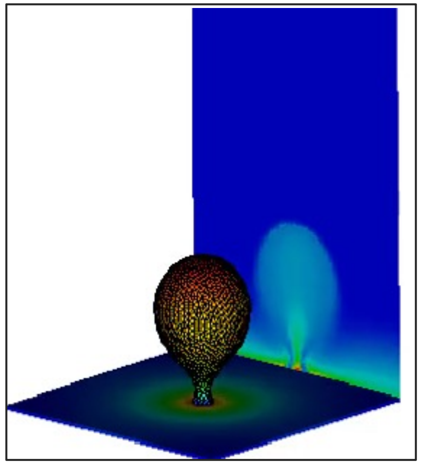
Numerical and physical models of TRUST

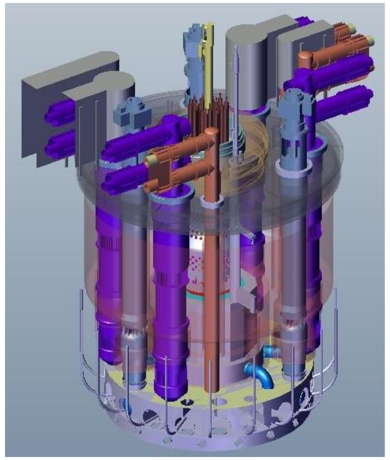
- Front-tracking model:
  - Eulerian mesh where incompressible Navier-Stokes equations are solved
  - Lagrangian moving mesh for the interface location
  - Coalescence or breakup models for bubbles and drops

- Multiphase flow simulation
  - Arbitrary number of phases
  - Source terms and operators for phase coupling
  - Dedicated numerical schemes for resolution

- Examples of performed simulations
  - **Academic case studies**
    - Plane channel with conduction coupling at the wall
    - Flow around an obstacle
    - Pipe flow
    - Isotropic turbulence…
  - **Industrial case studies**
    - Various studies about the core of a reactor
    - Thermal stress in a T-shaped mixing pipe
    - Natural convection in a storage room of waste
    - Atmospheric dispersion (pollution or radio-nuclides) …



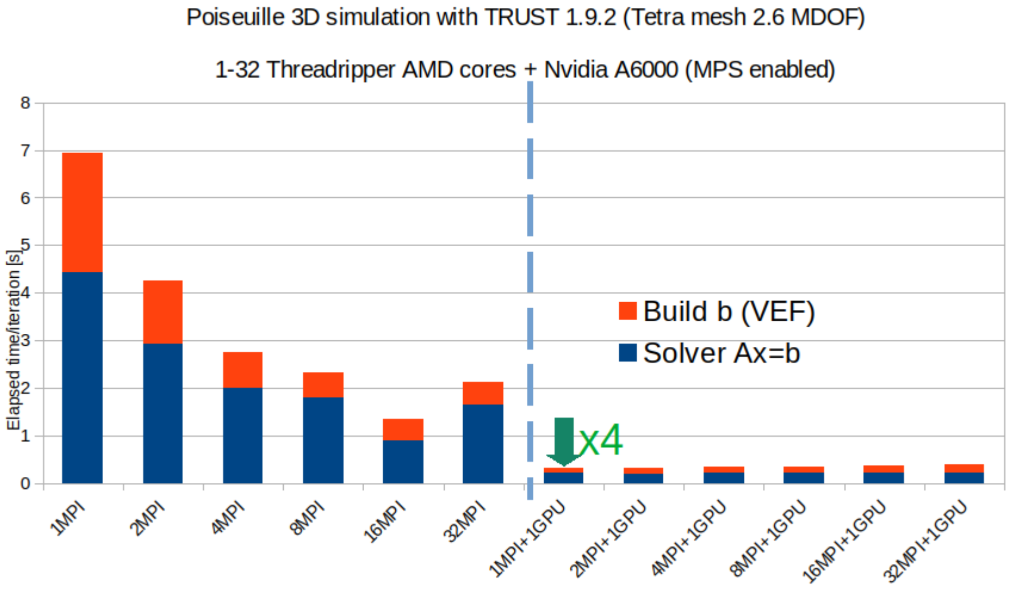




Gallery courtesy: TRUST team

Porting TRUST to GPU

- Computational efforts for models are levelled by HPC cutting-edge technologies (hardware: ex Adastra) and techniques (algorithms, adapted numerical schemes)
    - On-going GPU-porting with OpenMP and Kokkos programming models
    - Promising performance gains from workstation to supercomputer
    - Running a first module (incompressible laminar flow) on a mesh with 1 billion cells, accelerated by 512 Nvidia V100 (JeanZay, IDRIS) or 1024 AMD MI100 (Adastra, CINES)



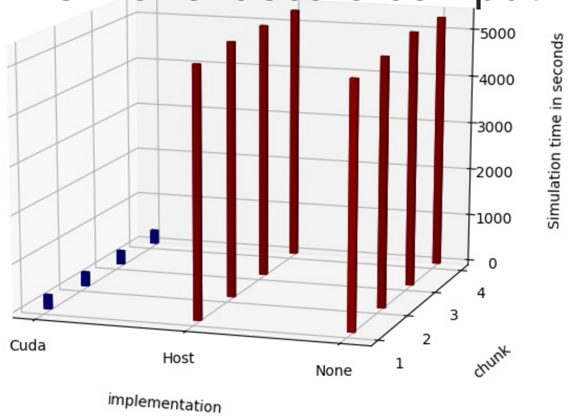
Gallery courtesy: P. Ledac

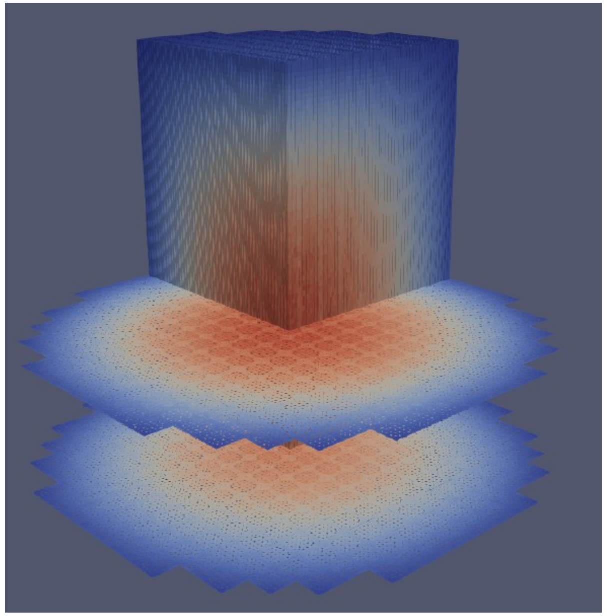
Advanced neutron transport (Monte Carlo)

- Monte Carlo: embarrasingly parallel
  - TRIPOLI-4 developed at CEA

- PATMOS
  - Prototype for testing new programming paradigms for Monte Carlo in particle
  - Developed at SERMA
  - Led to the seeding of the Tripoli-5 project supported by EDF and co-developed with IRSN
  - Very efficient scaling on the Hoogenboom-Martin (PWR) benchmark
  - On-the-fly Doppler broadening (to account for feedback with thermalhydraulics)

- On-going GPU-porting of Monte Carlo algorithm in view of exascale computing
  - Testing of several programming models: Cuda, OpenACC, OpenMP offloading, Kokkos





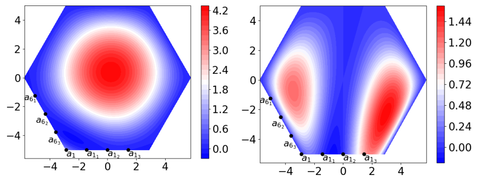
Coupling of PATMOS and THEDI (1D thermalhydraulics): power distribution

Gallery courtesy: TRIPOLI-5 team

Advanced neutron transport (deterministic)

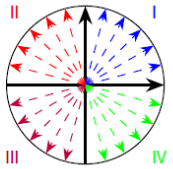
- Deterministic neutron transport: APOLLO3® code for industrial schemes

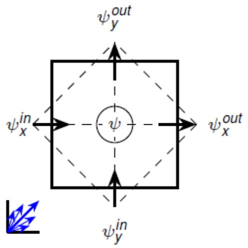
- HPC
  - Vectorisation of sweep kernel with Kokkos
  - Porting to GPU: not just simple rewriting but re-conceiving the sweep algorithm

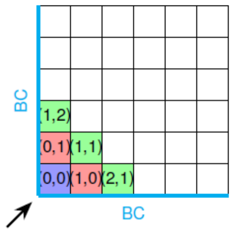


- Numerical schemes
  - High-order DGFEM* on structured meshes (improvement of *dofs* vs time)
  - Towards one-step: PhD works on dynamic homogenisation and coupled IDT-TH computation

DGFEM*: Discontinuous Galerkin Finite Element Method

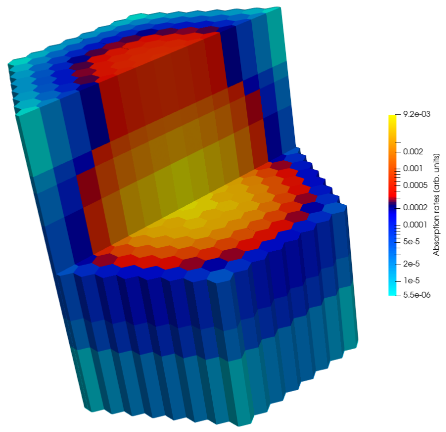






Sweep algorithm for source iteration

Power distribution Takeda 4 benchmark



Performance tests and results: AVX

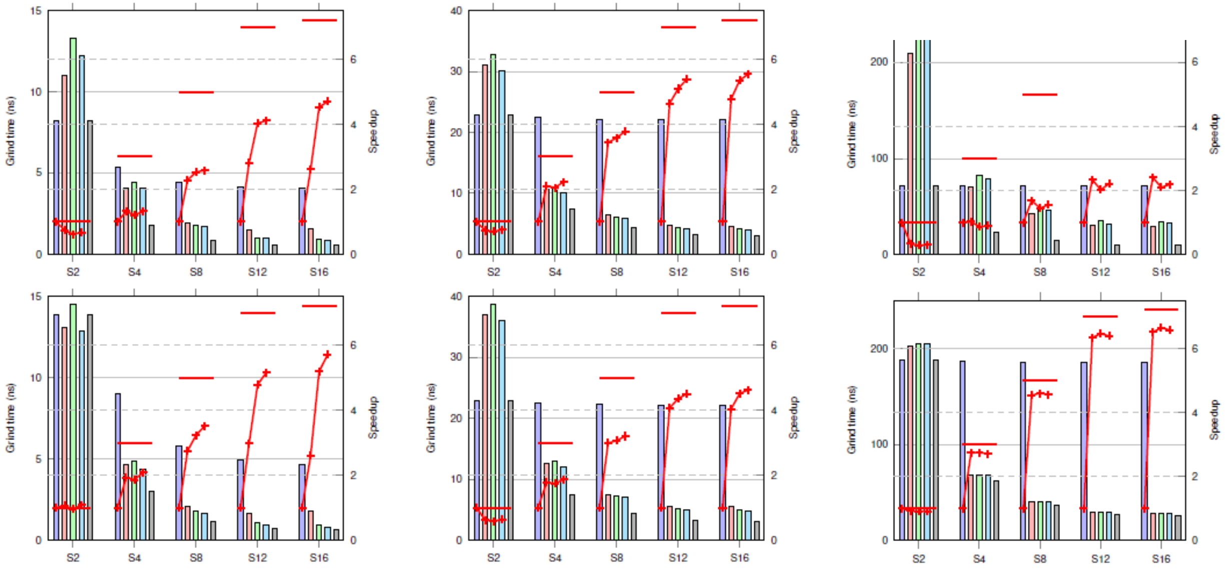


Figure 3 – GCC (up) and Clang (down) using AVX. Left to right: $\mathcal{DD}0, \mathcal{DD}1, \mathcal{DD}2$. Bars: grind-time (left to right: Scalar, A, B, C, Ideal). "+": measured speedup. "–": ideal speedup.

From Europlexus/CAST3M to MANTA

- Europlexus: *Explicit* dynamics for structures and compressible fluids
- Fluid / structure interactions
- Industrial applications
- Finite-elements, finite-volumes, sph, discrete element method
- ~40 years of development

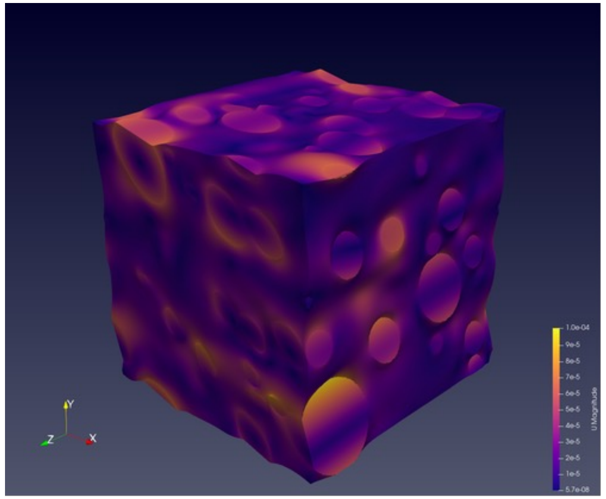
- CAST3M: Generic tool for “*implicit* problems”
- Mainly geared for (non-linear) mechanics
- … but also applied to incompressible fluids, electromagnetism, metallurgy, …
- Industrial applications
- Finite-elements
- ~40 years of development

- **M**echanical **A**nalysis **N**umerical **T**oolbox for advanced **A**pplications
- Next gen., **HPC** oriented
- *Both implicit and explicit problems*
- Structure / compressible fluids / … , interactions
- Industrial applications
- Every mesh-based method (FE, FV, HDG, …)
- C++
- “automatic parallelism”
- Easy to maintain and evolve on the long term
- Open-source

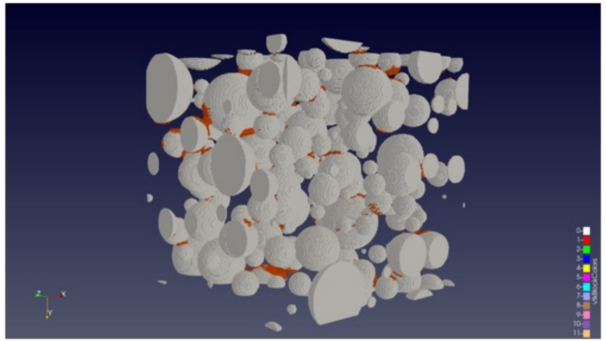
MANTA: current status

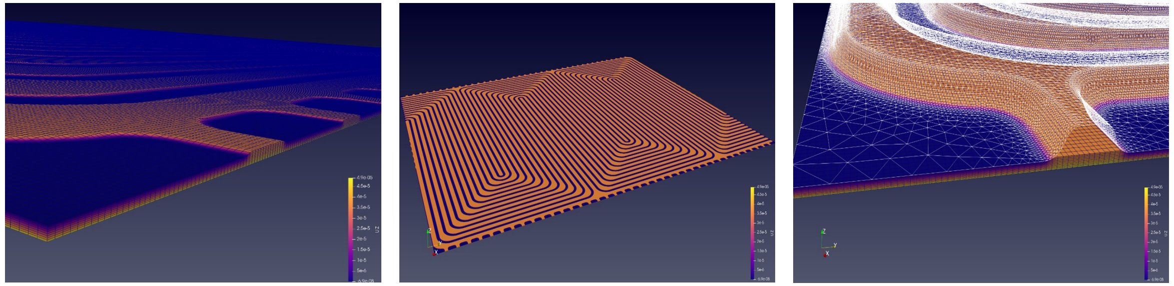
- **On-going development at CEA with contribution from EDF**

- **Collaboration between software and mechanical research engineers**
  - Support: I/O med, mesh partitioning, parallel solvers, …
  - Management for checkpoint/restart
  - Performance portability: hybrid parallelisation techniques (MPI+X) CPU, then GPU



- REV simulation
- 2-phase: elastic inclusions and elasto-plastic matrix
- Contrast: 100
- Periodic boundary conditions
- 400x10e6 dofs
- 16384 MPI subdomains
- Implicit calculation



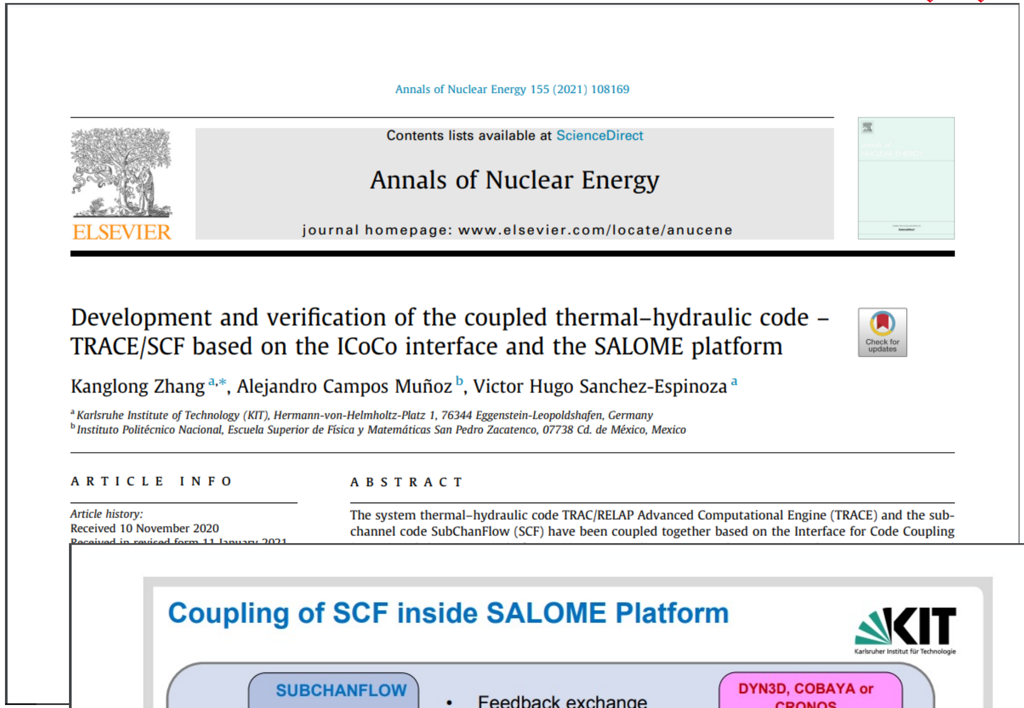


Shaping test for a fuel cell plate

Gallery courtesy: O. Jamond

One to bind them all

- Physical systems require multiphysics modelling
- Partitioned coupling approach
  - Few monolithic code with intricate coupling of all physics (*e.g.* SIMMER)
  - Simpler maintainance
- Code coupling: ICoCo + MEDCoupling (SALOME)
  - ICoCo = Interface for Code Coupling
  - MEDCoupling = Library with remapping options for quantities defined as fields (scalars/vectors on a mesh)
  - C3PO open-source python library (https://github.com/code-coupling/c3po)
    - Provides algorithms and accelerations for coupling
- NumPEx (ExaMA WP3) project on code coupling for exascale


Annals of Nuclear Energy 155 (2021) 108169

Contents lists available at ScienceDirect

Annals of Nuclear Energy

journal homepage: www.elsevier.com/locate/anucene

ELSEVIER

Development and verification of the coupled thermal–hydraulic code – TRACE/SCF based on the ICoCo interface and the SALOME platform

Kanglong Zhang [a,*], Alejandro Campos Muñoz [b], Victor Hugo Sanchez-Espinoza [a]

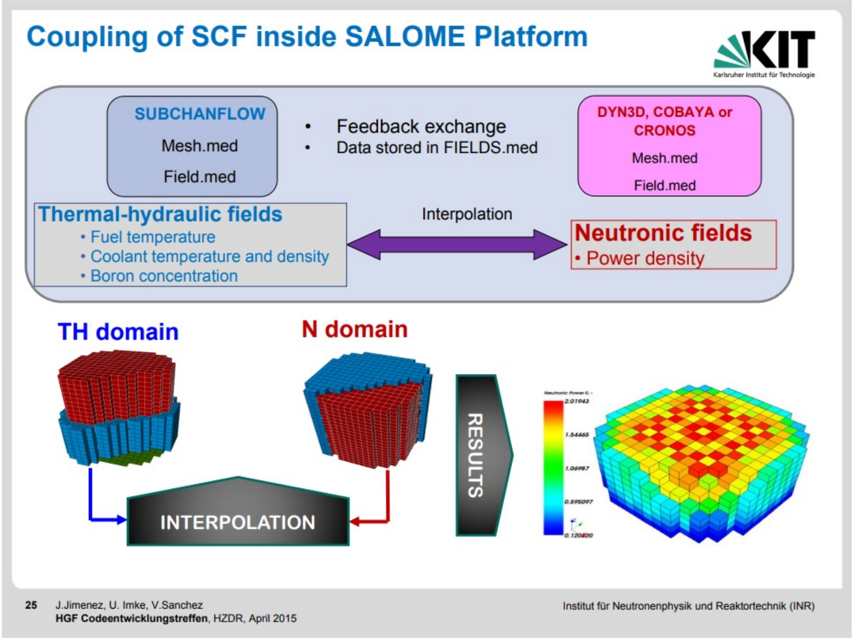
[a] Karlsruhe Institute of Technology (KIT), Hermann-von-Helmholtz-Platz 1, 76344 Eggenstein-Leopoldshafen, Germany
[b] Instituto Politécnico Nacional, Escuela Superior de Física y Matemáticas San Pedro Zacatenco, 07738 Cd. de México, Mexico

ARTICLE INFO

Article history:
Received 10 November 2020

ABSTRACT

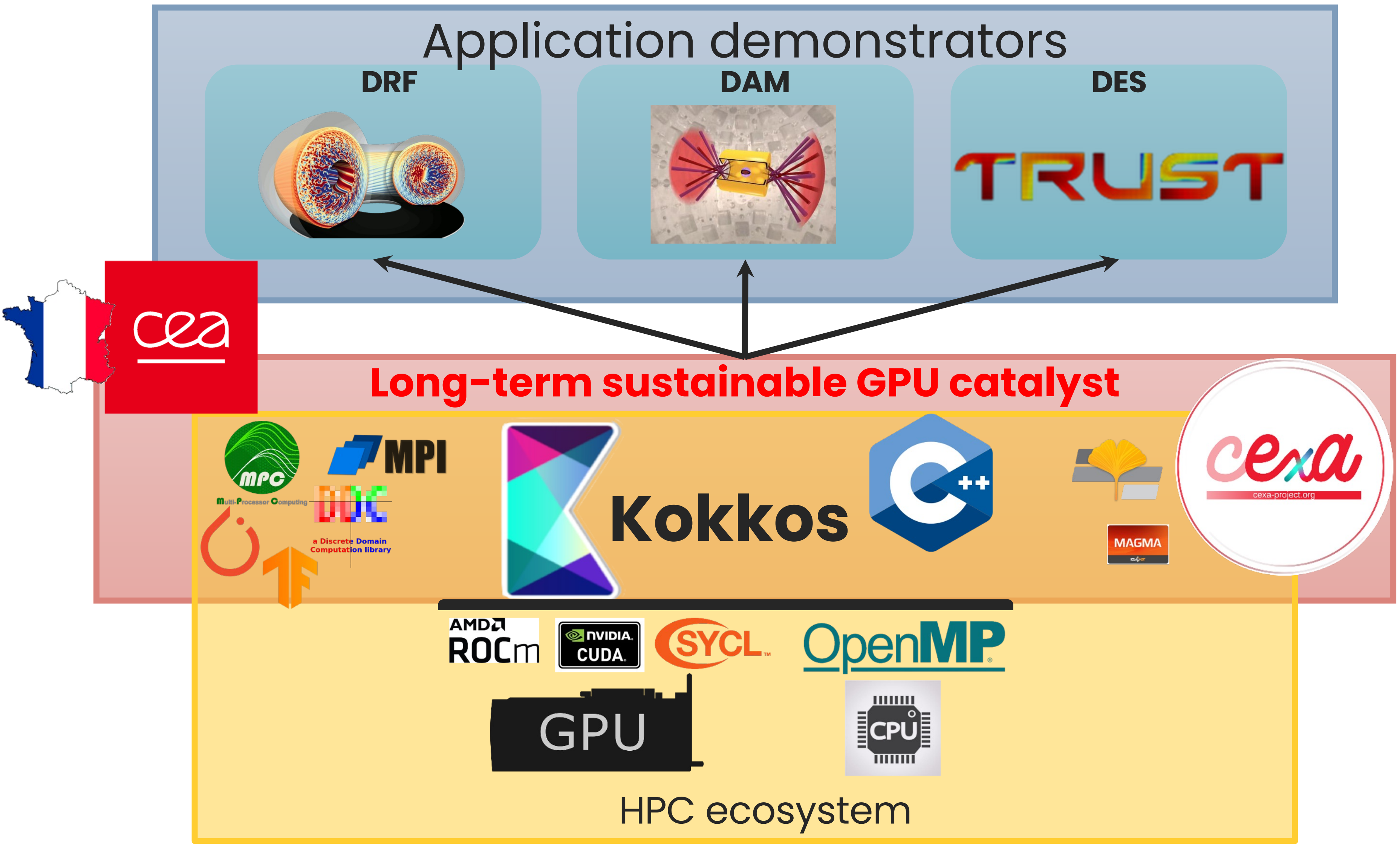
The system thermal–hydraulic code TRAC/RELAP Advanced Computational Engine (TRACE) and the sub-channel code SubChanFlow (SCF) have been coupled together based on the Interface for Code Coupling



https://publikationen.bibliothek.kit.edu/1000077003/5110456

3. Performance portability

The CExA project



Disseminate and offer training in CEA and at large

Adapt application demonstrators

Provide a long-term sustainable software catalyst for GPU computing

Available solutions

- Cuda
- HIP
- **Kokkos**
- OpenACC
- **OpenMP (target)**
- Raja
- SYCL
  - OneAPI/DPC++
  - AdaptiveC++/OpenSYCL/hipSYCL

- Production grade, with public support
- Vendor neutral
- Annotations
  - Works best with imperative languages: C, Fortran, …
  - Compiler integration: potential for additional optimizations
  - Requires to re-design applications for GPU
- Library
  - Suited to language with deep encapsulation: C++, …
  - On top of vendor backends: easier to port to new hardware
  - Requires to re-write applications for GPU

OpenMP & Kokkos : the simplest GPU loop

```
for (int j = 0 ; j < Nj ; ++j) {
        // [...]
}
```

Sequential

```
#pragma omp teams distribute parallel for
for (int j = 0 ; j < Nj ; ++j) {
        // [...]
}
```

OpenMP Target

```
parallel_for(Nj, KOKKOS_LAMBDA(int j) {
        // [...]
});
```

Kokkos

Execute in parallel, on a separate GPU thread each,

the same workload *[...]*

identified by a unique identifier `j`

`Nj` times between `0` and `Nj-1`

OpenMP & Kokkos : memory transfer

```
double* x = malloc(Ni*sizeof(double));
double* y = malloc(Nj*sizeof(double));
double* A = omp_target_alloc(
     Ni*Nj*sizeof(double),
     omp_get_initial_device());

#pragma omp target data \
     map(to: x[0:Ni]) \
     map(from: y[0:Nj])
{
#pragma omp teams distribute parallel for
for (int j = 0 ; j < Nj ; ++j) {
          for (int i = 0 ; i < Ni ; ++i) {
                    y[j] += x[i] * A[j*Ni+i];
          }
}
```

OpenMP Target

```
View<double*, Kokkos::HostSpace> x(Ni);
View<double*, Kokkos::HostSpace> y(Nj);
View<double*> A(Nj, Ni);

{
auto dx = create_mirror_view_and_copy(dev, x);
auto dy = create_mirror_view(dev, y);
parallel_for(Nj, KOKKOS_LAMBDA(int j) {
          for (int i = 0 ; i < Ni ; ++i) {
          dy(j) += dx(i) * A(j,i);
          }
});
deep_copy(y, dy);
}
```

Kokkos

Copy x to GPU from device before kernel
and y from GPU to device after kernel
Keep A on the device

What's in Kokkos

- Multi-dimensional arrays
  - Layout auto change for performance
- Other containers
  - Key-value maps, …
- Automatic ref-counted Host/Device memory allocation & management
- Host/device memory transfers
- Support of "dual" arrays with one version on each side
  - Up-to-date tracking & automatic transfers when required
- Scratch memory
  - Using "core-local" fast memory on the device

- Parallel patterns w. asynchronous support
  - Independent interactions, Reductions, Scans
- Iteration strategies
  - Tiled, Hierarchical, …
- Algorithms
  - Sorting
  - Random number generation
  - Most of STL parallel algorithms
  - …
- QoL features: portable printf, etc.
- Portable atomic operations
- SIMD
- Coarse & fine-grain tasks
- And much more…

Kokkos an anteroom for standard C++

C++ is (at last) standardizing base tools for HPC

- **Parallel programming** is slowly entering the **ISO C++ language**
  - Parallel algorithms, sender/receivers, etc.
- **The Kokkos team** leads the **standardization** of many required features
  - Multi-D arrays (`std::mdspan`)
  - Vectorization (`std::simd`)
  - Linear algebra (`std::linalg`)
  - And much more to come (mixed precision, etc.)

Kokkos offers a **stable API** today for the features of the **C++ of tomorrow**

- Standardization is slow (9 years for mdspan)
  - Consensus with all communities
- Kokkos offers the **features today**
  - And will **keep maintaining the API** on top of standardized ISO C++
    - With added interoperability layers (Cf. `kokkos::view` / `std::mdspan`)
  - And in a **GPU-compatible** implementation (Cf. `kokkos::array`)

HIGH PERFORMANCE SOFTWARE FOUNDATION

HPSF Goals

- Provide neutral home for key HPC projects to enable collaboration between government, industry and academia
- Promote use of HPSF projects
- Ensure that HPC software is accessible and reliable by providing CI and turn-key builds
- Ensure that HPC software is secure and ready for cloud through collaborations with CNCF and OpenSSF
- Sponsor events and training to grow a diverse, skilled workforce for software in the HPSF ecosystem.

Members

Premier

aws
Hewlett Packard Enterprise
Lawrence Livermore National Laboratory
Sandia National Laboratories

General

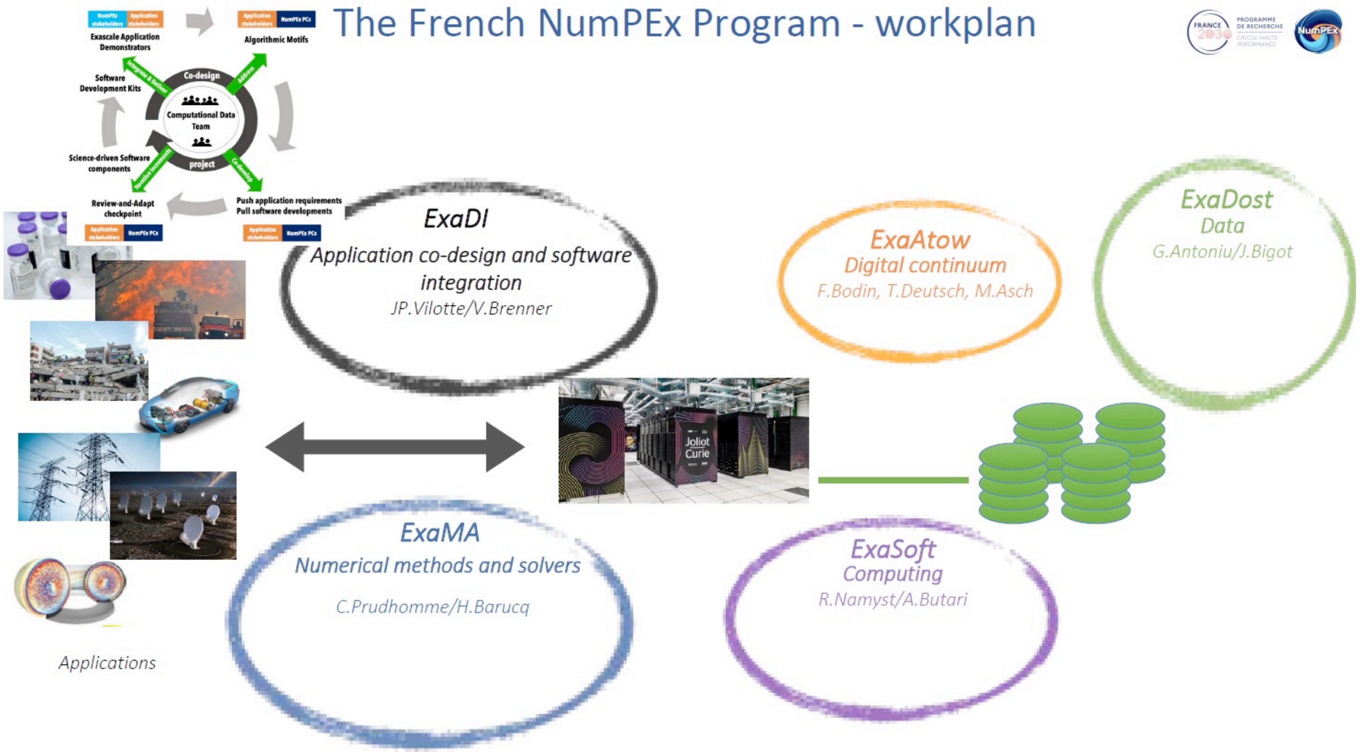
AMD
intel
NVIDIA
Argonne National Laboratory
Oak Ridge National Laboratory
Los Alamos National Laboratory
kitware
cea

Associate

सी-डैक CDAC
University of Oregon
University of Maryland

NumPEx

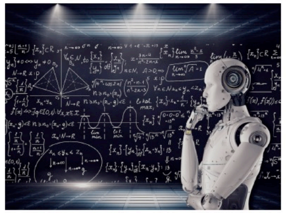
- French project to co-design the exascale software stack and prepare applications to exascale era
- Funded by PEPR (Programme et Equipements Prioritaires de Recherche)
  - Exascale Computing Project (USA): co-design centres, software stack
- Partners: CEA, CNRS, INRIA, Universities,…



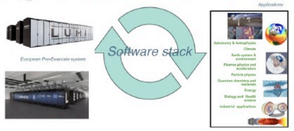
Some Exascale/post-Exascale challenges

From edge to HPC systems
The digital continuum

AI4Science – Science4AI



Software/application co-design



Software, the new frontier

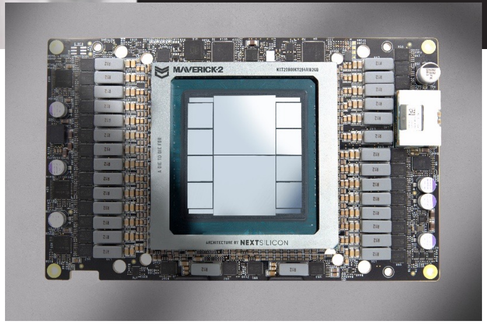
4. Concluding remarks

Concluding remarks

- Code development/maintainance is a tedious and everlasting process
  - Essential as numerical and physical models are safeguarded within code systems (decades)
  - Accounting for all constraints to deliver industrial-grade material (results, software and methods)

- GPU porting: still a challenge but necessary for exascale computation, requires the design and implementation of new algorithms

- Forthcoming short-term challenges
  - Floating-point arithmetics to explore intensive verification and numerical quality of codes and improve algorithms with mixed precision
  - More AI/ML to explore: promising to bridge the gaps on models



- How to adapt the codes to forthcoming hardware? Is it possible to win the rat race?
  - Quantum Computing: 2nd quantum revolution on the march, assess the relevance for our problems (linear algebra, iterative solvers), theoretical transition from the usual algorithms

Thanks for listening

Figures/Plots/Graphs – courtesy of DM2S and CExA team

CEA SACLAY
91191 Gif-sur-Yvette Cedex
France

Compilation

OpenMP Target

- Use an OpenMP compiler
  - Compatible with the target construct
  - Compatible with the hardware you target
- Each vendor provides its own OpenMP compiler
  - Usually based on LLVM infra
- Default Clang/LLVM & GCC also try to support this
  - For some hardware

Kokkos

- A C++ template library
  - No direct code generation, rely on vendors C++-like languages
- Multiple “backends”, selection at compile time
  - OpenMP, Cuda, OneAPI, HIP, …
- Maximum 3 backends enabled at once
  - Serial backend
  - 1 Host parallel backend (openmp)
  - 1 Device parallel backend (cuda, HIP, Sycl)

References

[1] "Numerical modeling of a moderate hydrogen leakage in a typical two-vented fuel cell configuration", E. Saikali, P. Ledac *et. al.*, International Conference on Hydrogen Safety, September 2021

[2] SALOME: https://www.salome-platform.org/?lang=en

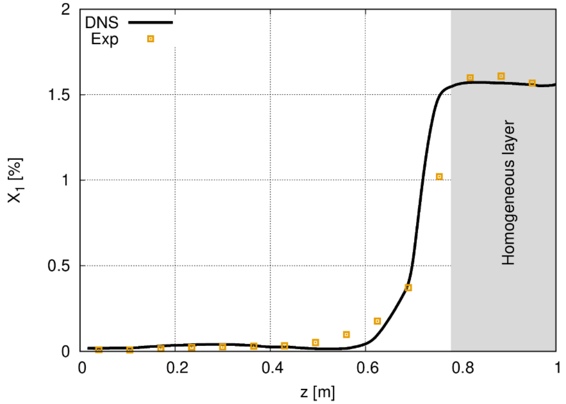
[3] TRUST: code https://github.com/cea-trust-platform; website https://cea-trust-platform.github.io/

[4] "PATMOS: A prototype Monte Carlo transport code to test high performance architectures", E. Brun, S. Chauveau, F. Malvagi , M&C 2017, April 2017

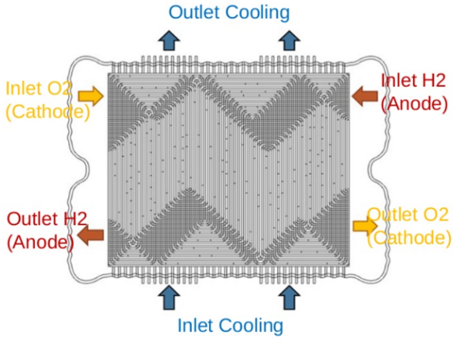
[5] "High-order Wachspress functions on convex polygons through computer algebra", D. Labeurthre, A. Calloo, R. Le Tellier, Journal of Computational Physics, 2022

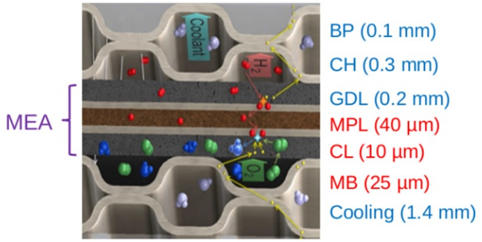
High-fidelity simulations for new systems

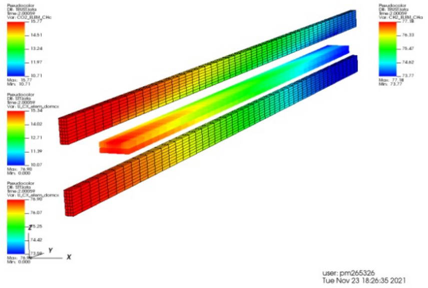
- GAMELAN (moderate hydrogen leakage) – 2 billion mesh cells, 50 000 cores [1]
- PEMFC and batteries – complex geometries



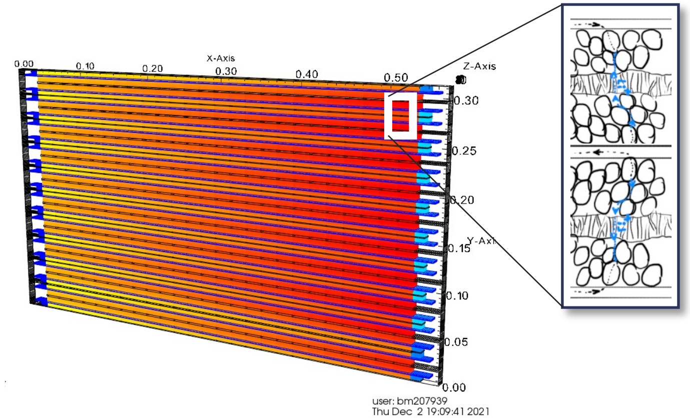
GAMELAN: reference results by CEA

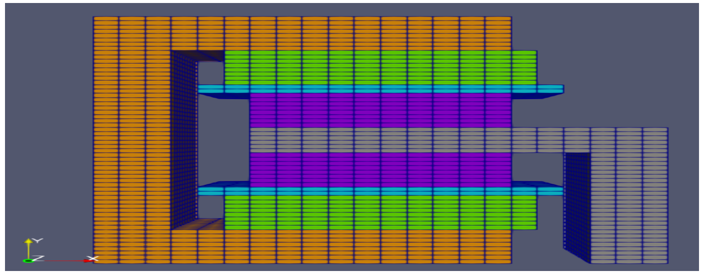






Battery simulation





MEDCoupling representation of PEMFC