Energy-aware CFD simulations: Sharing Experience from the CEEC project

Kajol Kulkarni, Samuel Kemmler, Anna Schwarz, Gulcin Gedik,
Yanxiang Chen, Dimitrios Papageorgiou, Ioannis Kavroulakis,
Roman Iakymchuk (Umeå University, Sweden)

CEEC
Centre of Excellence in Exascale CFD

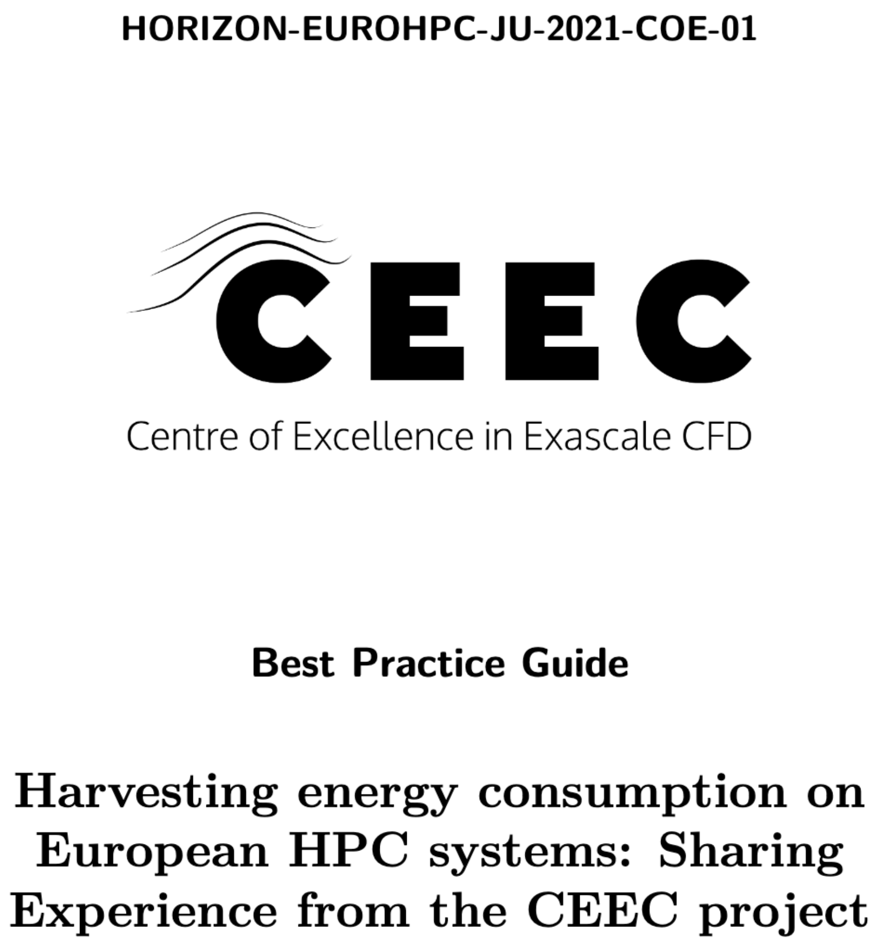
Motivation

- **Why energy efficiency matters in HPC ?**
  - Exponential growth in computational demand and system scale
  - Exascale energy consumption at peak operation
    - JUPITER Booster (Germany): 15 MWh
    - Fugaku (Japan): 29 MWh
    - El Captain (USA): 29 MWh

- **Why is it difficult to measure Energy ?**
  - Energy profiling is intrinsically complex and fragmented
  - Heterogeneous architectures lead to heterogeneous counters
  - Measurement challenges, such as vendor-specific APIs, different sampling granularity, restricted user access on systems etc.

How to measure energy consumption?

- **How to measure**:
    - Job scheduler: Slurm
    - Tools: LIKWID, EAR, MERIC
    - Hardware interfaces: RAPL
- **CEEC Best Practice Guide**:
    - Facilitate energy measurements on EU clusters
    - Promote experience with the community
    - Provide examples with easy-to-use guide



HORIZON-EUROHPC-JU-2021-COE-01

Centre of Excellence in Exascale CFD

Best Practice Guide

Harvesting energy consumption on European HPC systems: Sharing Experience from the CEEC project

Energy metrics

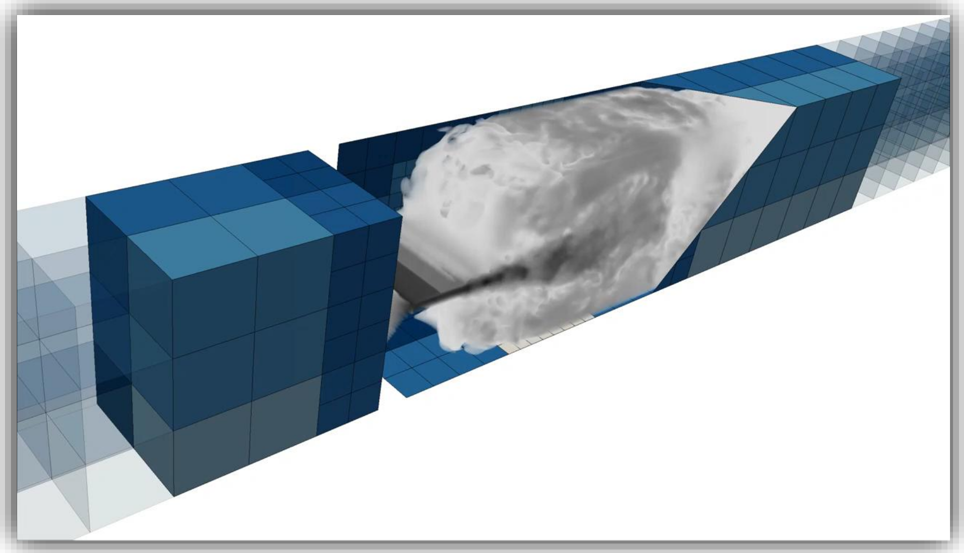
- Energy Consumption per CPU socket
  - SLURM workload manager (in Joules)
- Energy Ratio
  - Fraction of energy used by the parallel simulation compared to the energy used by the baseline simulation running with the minimum number of ranks: energy_p / energy_p0
- Energy normalized performance index (EPID)

$$\mathrm{EPID} = \frac{\mathrm{walltime} \times \mathrm{power}}{\#\mathrm{RK\text{-}stages} \times \#\mathrm{DoF}} = \frac{\mathrm{power}}{\underbrace{\#\mathrm{ranks}}_{P_{\mathrm{rank}}}} \times \mathrm{PID}$$

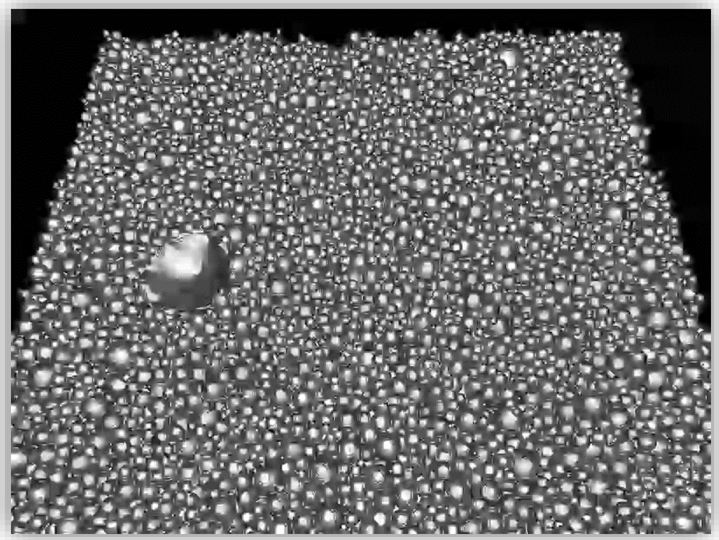
Software
waLBerla framework

- Written in C++, Python interface, code generation
- Main applications: CFD with the lattice Boltzmann method (LBM), rigid body dynamics using the Discrete Element Method (DEM), particulate flows, free-surface flows, phase fields
- Open source: www.walberla.net

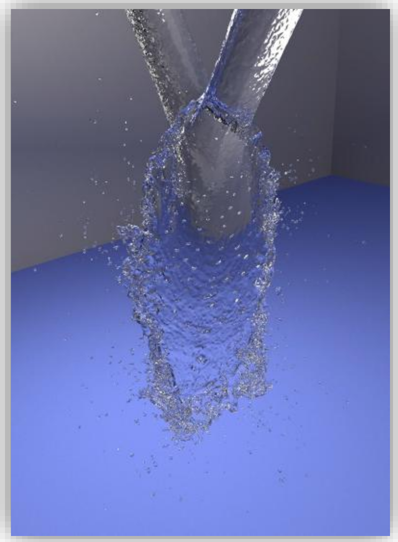
Source: Lehrstuhl für Systemsimulation, FAU



Adaptivity and dynamic load balancing



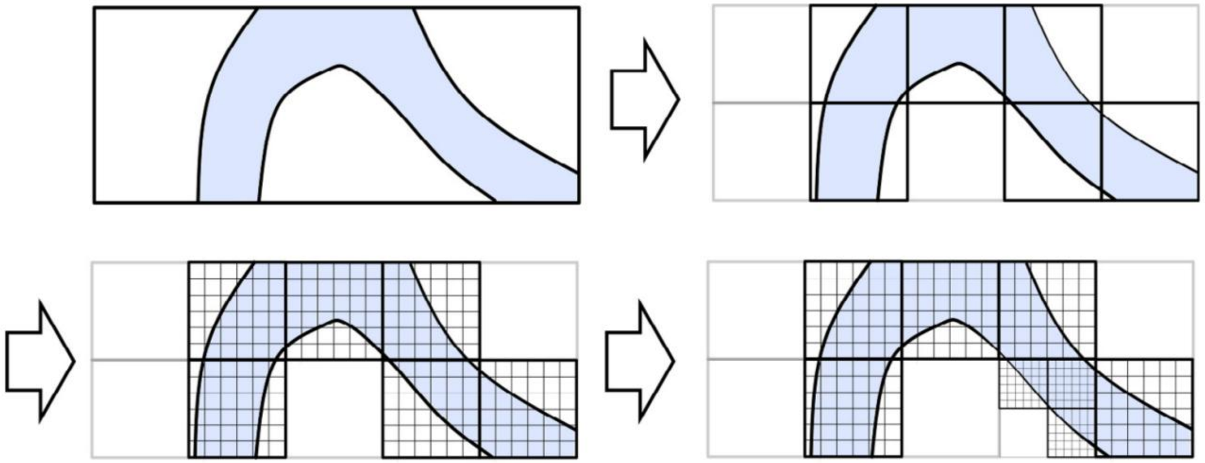
Additive manufacturing



Free surface flows

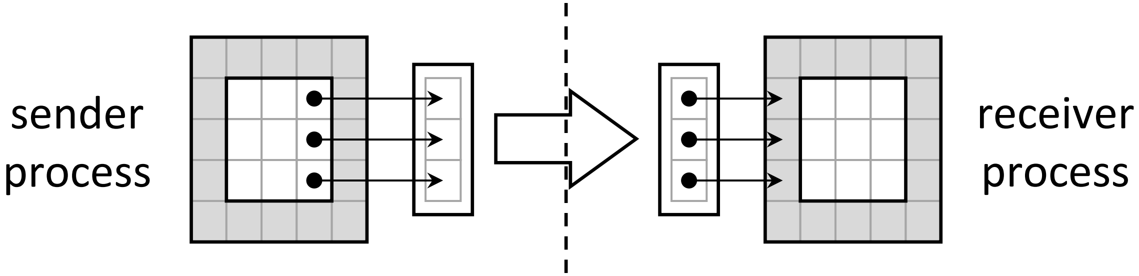
Software

Block-structured domain partitioning



The concept of the block forest

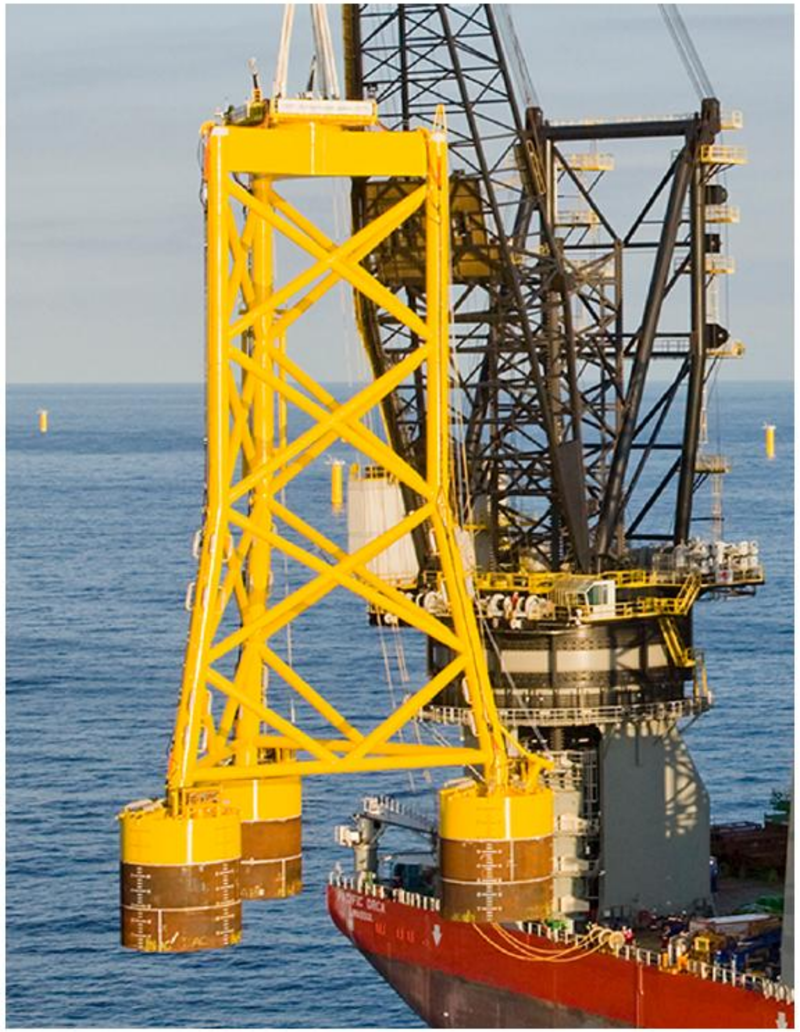
Communication: MPI-based halo exchange



M. Bauer, et al. *waLBerla: A block-structured high-performance framework for multiphysics simulations.* Computers & Mathematics with Applications, 81, 478-501, 2021.

Motivation CEEC LHC4
Suction-aided foundations

- Increasingly popular method to install offshore structures into the seabed
- **Faster, more cost-efficient, reversible, and environmentally friendly**, e.g., very low noise emissions, when compared to conventional methods like dynamic blow-driving of monopiles
- Installed using self-weight and hydraulic under-pressure
- Problem: Piping erosion (granular fluidization) may cause a failure of the bucket installation process
- The conditions leading to piping erosion are not sufficiently understood yet
- Goal: better understanding of piping erosion using large-scale fully-resolved simulations



Source: NGI/Ørsted

Results
Numerical setup

- Fluid: lattice Boltzmann method
- Particles: discrete element method
- Dominating computational cost:
  - Fluid: simulates the fluid dynamics (stencil-based kernel)
  - Mapping: maps particle information onto the fluid grid
  - Reduction: aggregates hydrodynamic forces acting from the fluid on the particles (global reduction operation)

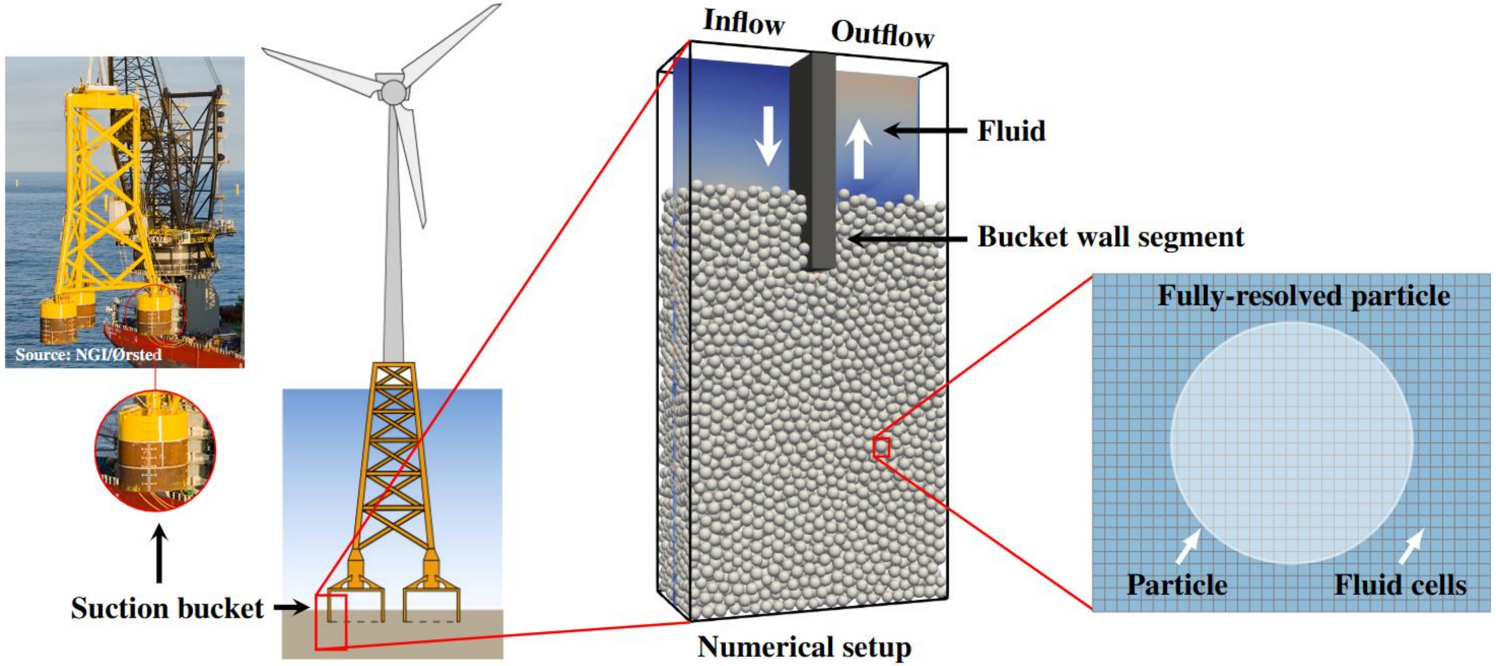


Table 1: Domain partitioning on LUMI.

Partition	Cores/GCD per node	MPI processes			Fluid cells per process		
		x	y	z	x	y	z
LUMI-C	128	4	2	16	112	112	56
LUMI-G	8	2	1	4	224	224	224

Results

Time and energy to solution

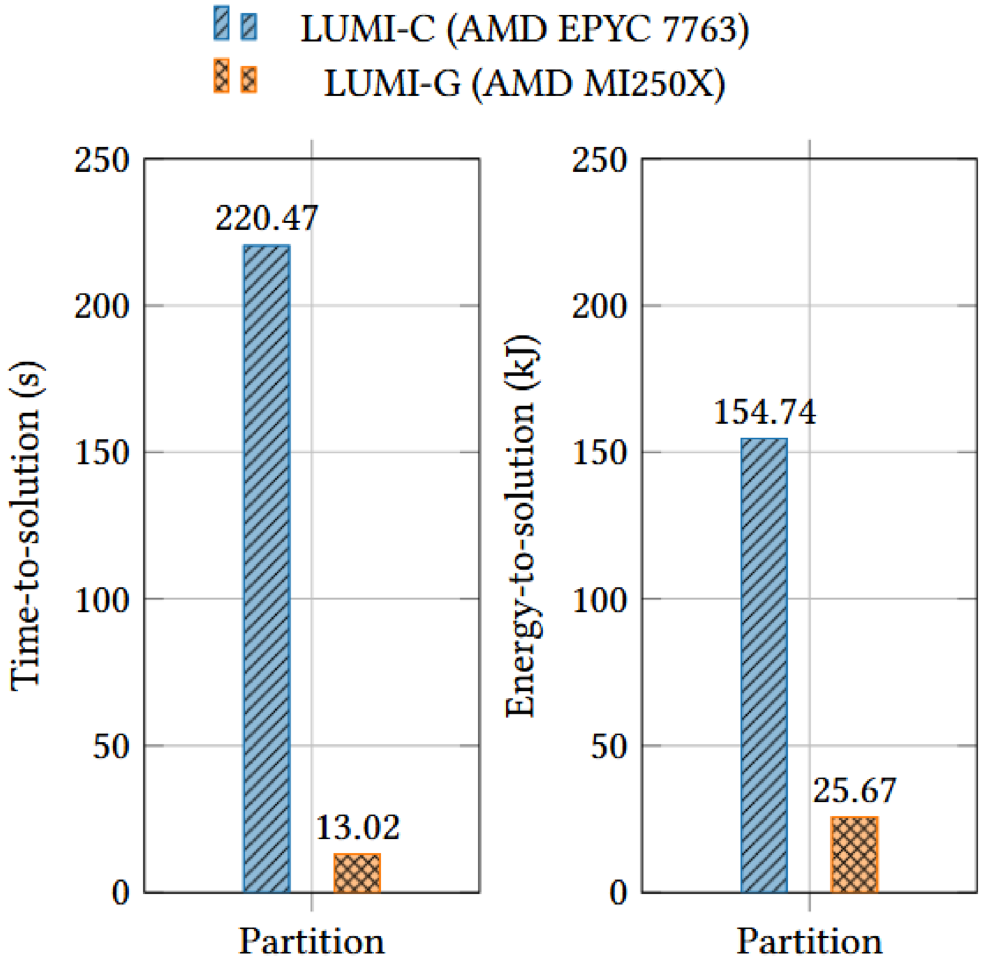


Figure 3: Time-to-solution (left) and energy-to-solution (right) on a single node run of LHC4, waLBerla.

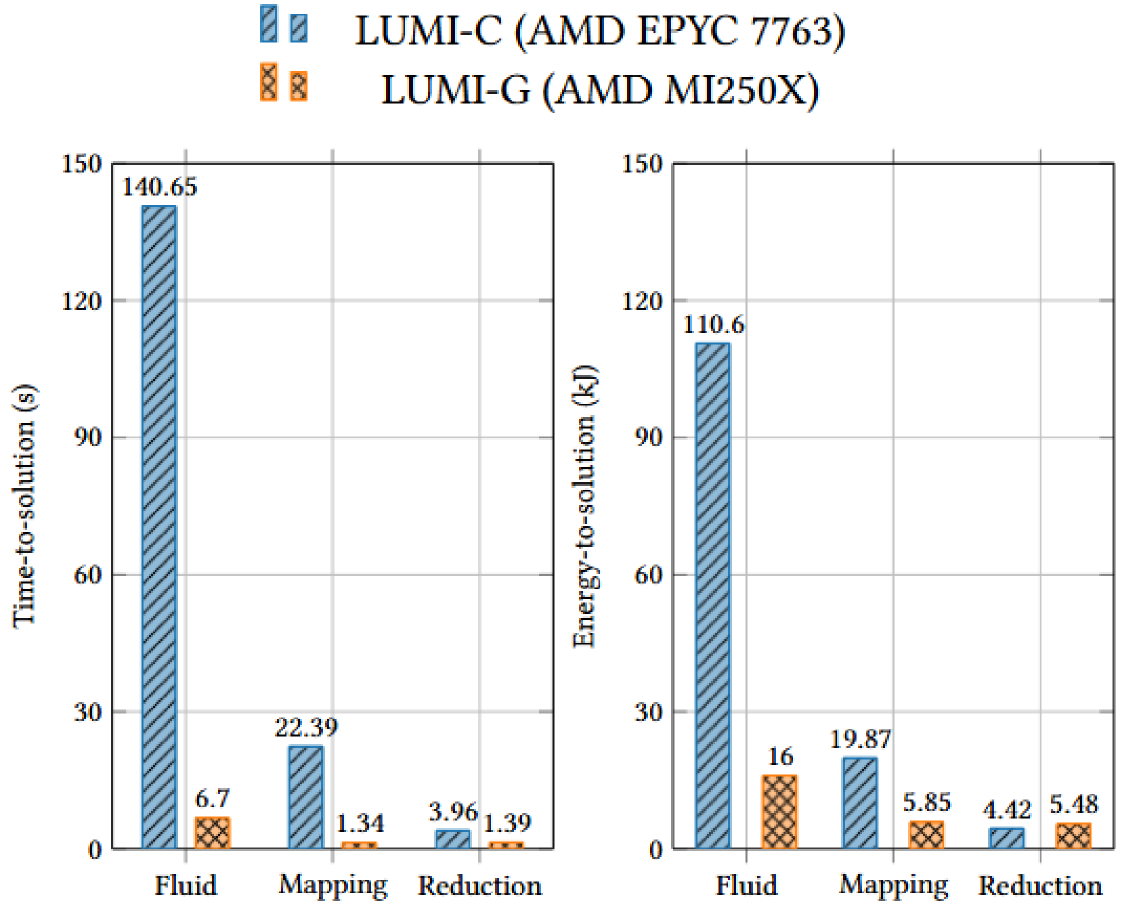
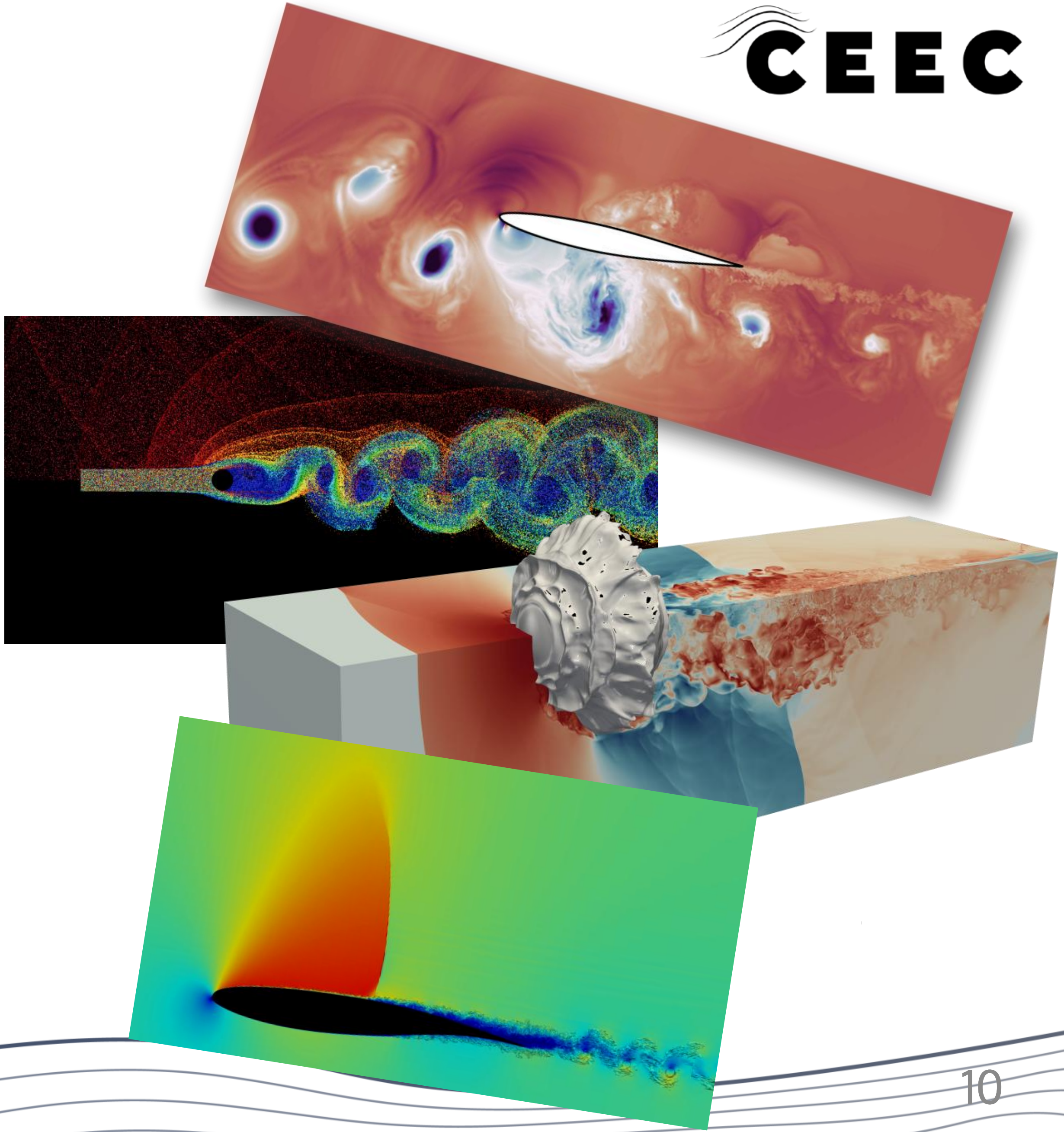


Figure 4: Time-to-solution (left) and energy-to-solution (right) on a single node run for the dominating simulation modules of LHC4, waLBerla.

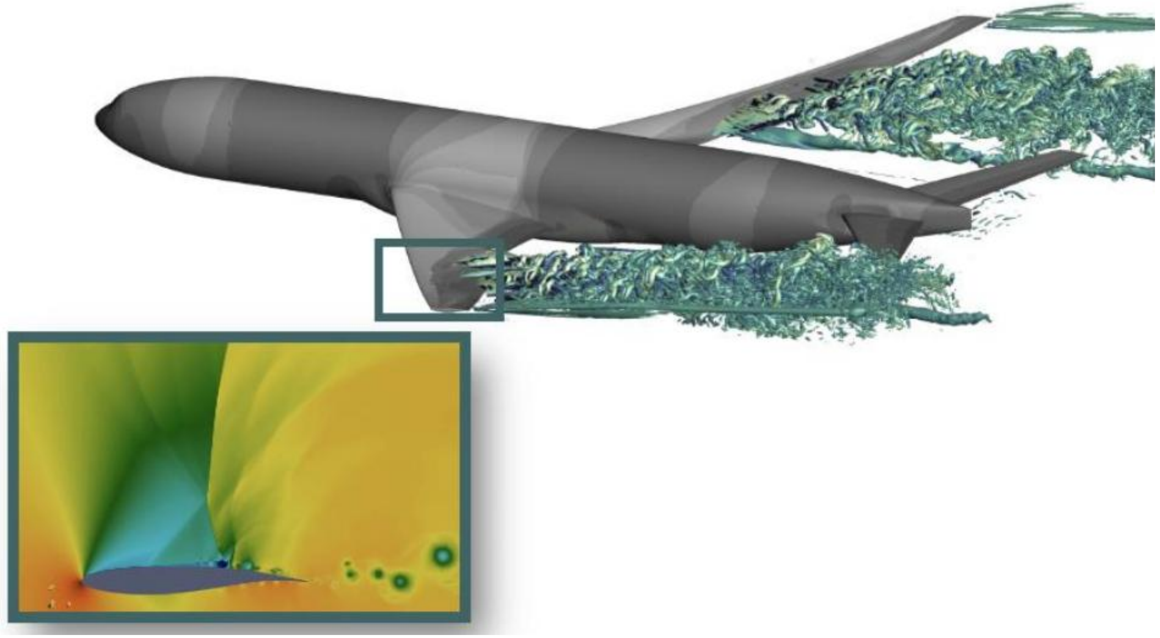
Software
FLEXI

- **High-order accurate** open-source solver with strong excellent scaling behavior
- Based on the discontinuous Galerkin spectral element method along with a high-order explicit Runge-Kutta time integration
- Written in modern Fortran and based on the MPI paradigm for parallelization
- Optimized for heterogeneous architectures including CPU and GPU based HPC systems
- Support of curved hexahedrons, tetrahedrons, prism, and pyramids
- **Main applications:** LES/DNS of multiscale, multi-physics, and multiphase problems focusing on the compressible regime



Motivation CEEC-LHC1
Transonic Buffet

- Transport aircraft cruise at transonic flight conditions
- Localized supersonic flow can occur which is terminated by a normal shock wave
- At the edge of the flight envelope the shock interacts with the flow separation which leads to a low frequency, high amplitude periodic oscillation
- Buffet causes:
  - A loss of lift and maneuverability
  - High dynamic loads and vibrations
  - Structural fatigue
- **Goal:** understand currently undiscovered mechanisms leading to shock buffet



Results

Time and energy to solution

- **Setup:** The simulation domain is a Cartesian box composed of hexahedral element using all code features required to simulate LHC1 such as finite volume for shock capturing
- A constant number of 2.81×10^7 DoFs (CPU) and 1.31×10^8 DoFs (GPU) is considered

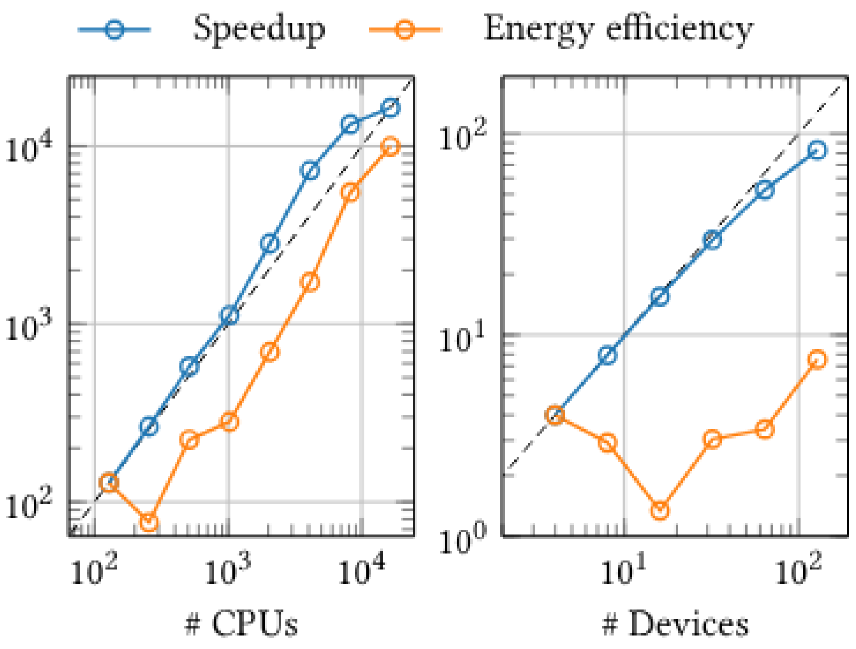


Figure 6: Normalized energy efficiency (energy-to-solution) and speedup (time-to-solution) of FLEXI/GALÆXI on the CPU (left) and GPU (right) partition of MeluXina.

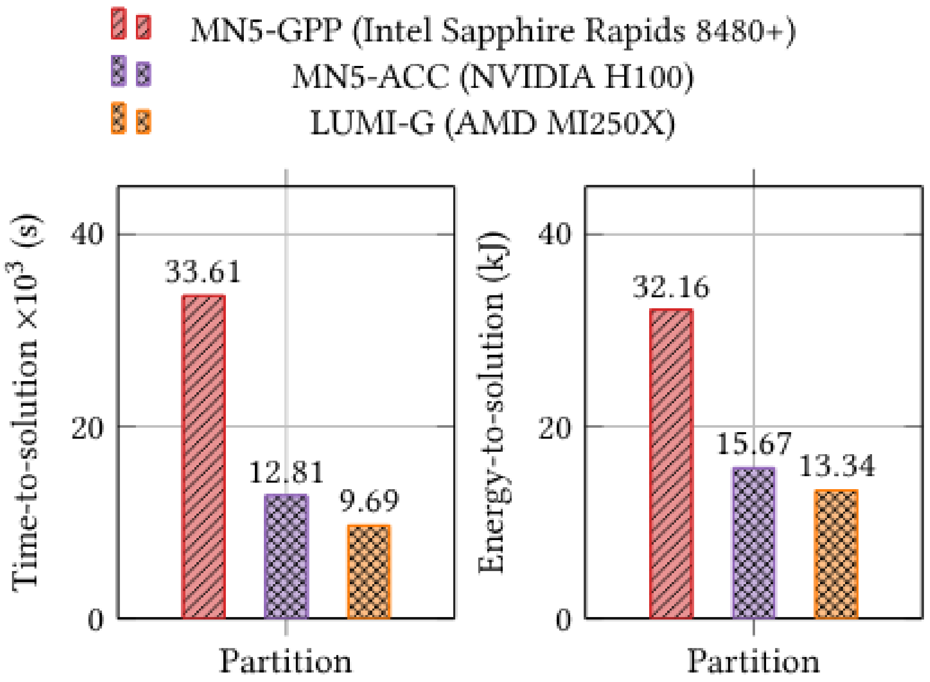


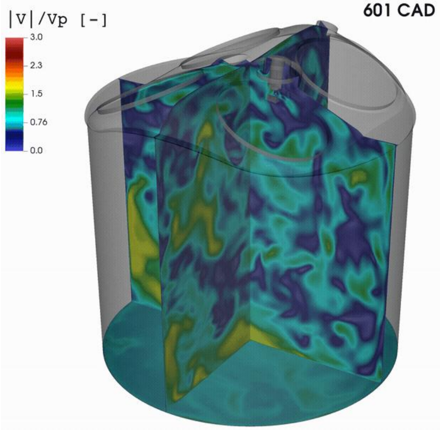
Figure 7: Time-to-solution (left) and energy-to-solution (right) on a single node for FLEXI/GALÆXI.

Software **NekRS**

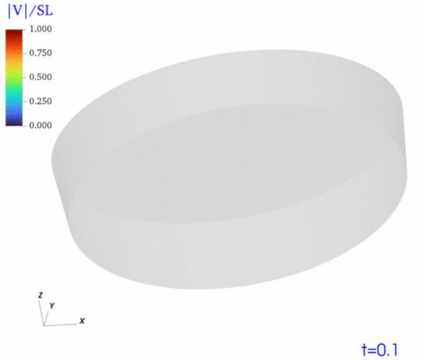
- Highly-efficient and scalable open source GPU/CPU incompressible and low Mach flow solver. Successor of its precursor CPU-based solver Nek5000
- Supports high-order curvilinear conformal Hex spectral elements in space
- Variable time step 2nd/3rd order semi-implicit time integration (BDF/EXT). Also supports operator-integration-factor-splitting (OIFS) scheme for incompressible flows (BDF/OIFS) overcoming CFL restrictions imposed by standard schemes
- MPI + OCCA supporting CUDA, HIP, DPC++, SERIAL (C++) backends
- LES and RANS turbulence models
- Arbitrary-Lagrangian-Eulerian moving mesh & overlapping mesh with multirate timestepping support
- Lagrangian phase model & conjugate fluid-solid heat transfer & various boundary conditions support
- VisIt & Paraview for data analysis and visualization including in-situ support through Ascent
- **Main applications:** reactive flows, multiphase flows, multimodel physics and moving/overlapping domains

APPLICARIONS

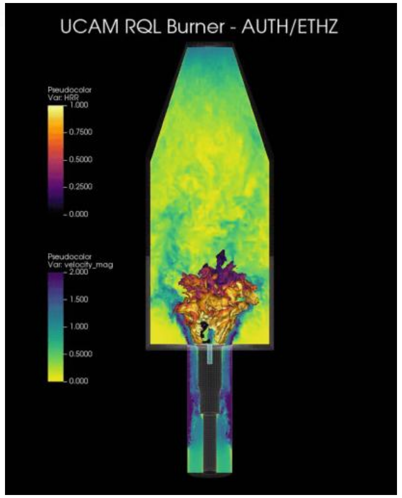
IC Engine



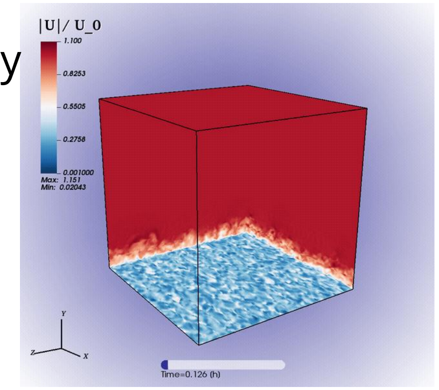
Flame Kernel



Soot Burner



ABL Flow



Motivation CEEC-LHC5

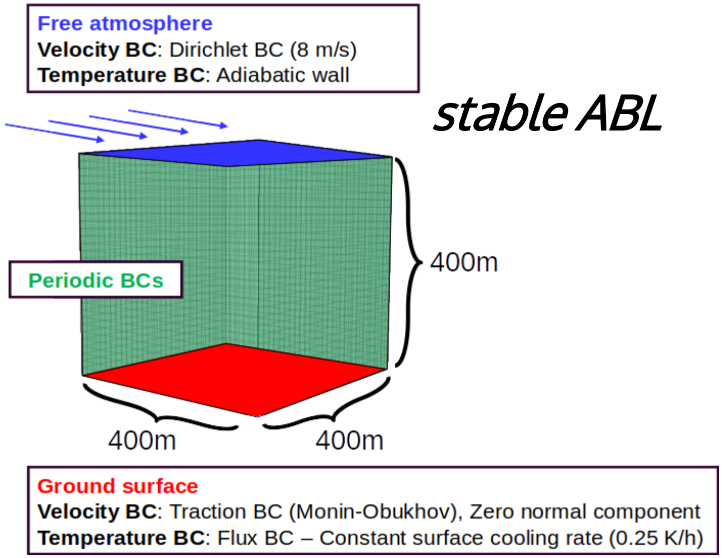
Atmospheric Boundary Layer Flows

Objective: Simulation of stable and unstable Atmospheric Boundary Layer (ABL) flows

- High turbulent and stratified flows with Re ~ O(10^8)
- Weather prediction models, climate simulations and environmental impact assessments require precise ABL representation

Motivation:

- Precise ABL -> SGS-WMLES models + high grid resolution -> GPU-accelerated HPC
- GPU-based large-scale production simulations with **NekRS**
- Utilization of various **subgrid-scale (SGS) and wall models**
- Investigation of numerical convergence as a function of Reynolds and grid resolution.
- Generation of ultra-high resolution and high-fidelity datasets for cross-comparison and input in MPTRAC and ICON, two macroscale earth system modeling codes

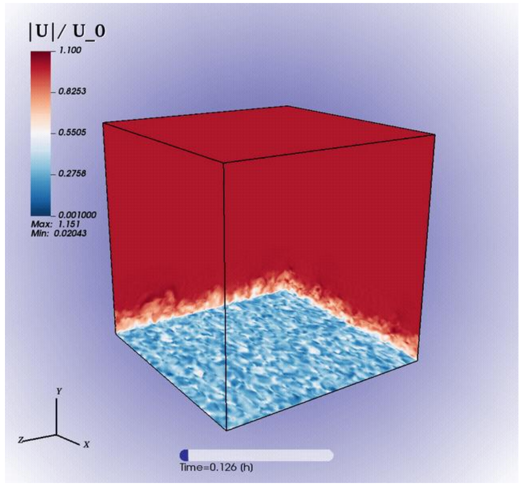


Free atmosphere
Velocity BC: Dirichlet BC (8 m/s)
Temperature BC: Adiabatic wall

stable ABL

Periodic BCs

Ground surface
Velocity BC: Traction BC (Monin-Obukhov), Zero normal component
Temperature BC: Flux BC – Constant surface cooling rate (0.25 K/h)



Results

Time and energy to solution

- **Linear energy scaling:** Energy consumption scales directly with the problem size; (e.g., 8x resolution -> 8x energy)
- **Lower parallel efficiency = higher energy cost:** As parallel efficiency drops (GPU underutilization), the energy required for the same simulation increases.
- **Scalability limiting factor:** The p-multigrid overlapping additive Schwarz method for the pressure preconditioner
- **Improve energy efficiency:** Efficient multigrid methods -> Increased scalability -> Enhanced GPU utilization

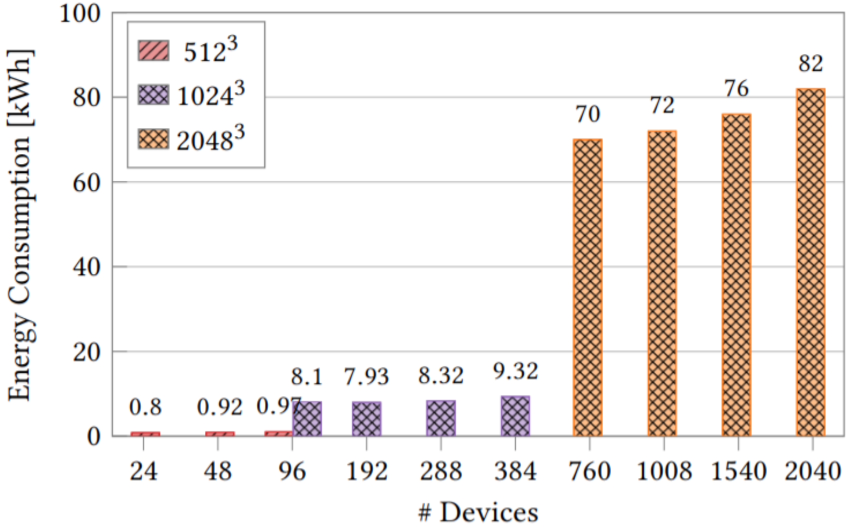


Figure 10: Energy-to-solution per 5,000 timesteps for the GABLS case, NekRS with resolutions of 512^3 , 1024^3 and 2048^3 on Nvidia A100 GPUs (JUWELS Booster).

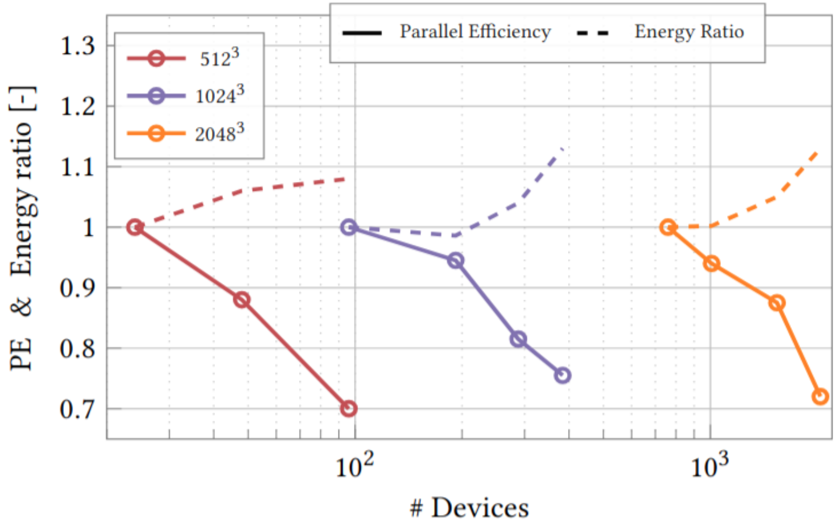


Figure 11: Parallel efficiency (PE) and energy ratio for the GABLS case, NekRS with resolutions of 512^3 , 1024^3 and 2048^3 on Nvidia A100 GPUs (JUWELS Booster).

$$\text{Energy Ratio} = \frac{\text{Energy}_P}{\text{Energy}_{P_0}},$$

- Energy_P is the energy required for a simulation with P ranks
- Energy_{P_0} is the energy required for a simulation with P_0 ranks

Software

Neko

Neko

- Portable simulation framework based on high-order, written in modern Fortran.
- Spectral Element Method (SEM) on hexahedral meshes, mainly focusing on incompressible flow simulations.
- Focus on single core/ single accelerator efficiency via tensor product operator evaluations.
- It is matrix-free, where one always works with the unassembled matrix on a per-element basis.
- More on https://neko.cfd/

Nekbone

- A mini-app that captures the fundamental design of Nek5000, a large-scale, high-order solver for incompressible Navier-Stokes equations based on SEM.
- It solves a standard Poisson equation by partitioning the computational domain into high-order quadrilateral elements and using the Conjugate Gradient (CG) method as its main computational kernel.
- The CG solver can optionally be compiled with a multigrid preconditioner.

Results

Time and energy to solution

- **Test case**
  - Possion equation on a cubic region
  - Neko shows strong scaling while Nekbone shows weak scaling
- **Mixed-precision advantage:**
  - Outperforms the double-precision baseline
  - It consistently achieves a lower time-to-solution and consumes less energy: **1.3x for Neko**
- **Impact of tolerance:**
  - Relaxing the solver tolerance leads to a general decrease in both time and energy-to-solution
  - Fewer iterations are required to reach convergence

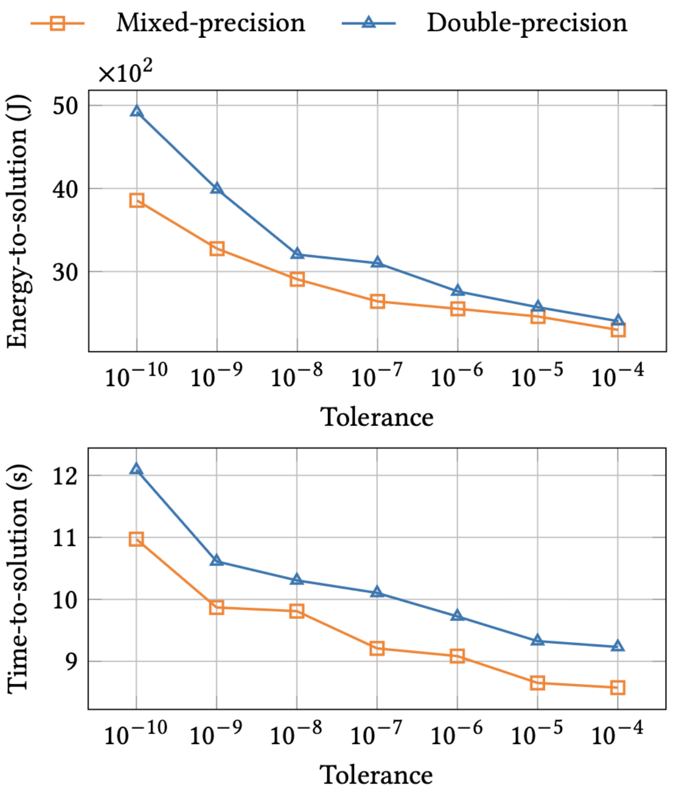


Figure 8: Energy-to-solution (top) and time-to-solution (bottom) vs. tolerance for mixed- and double-precision Nekbone (Poisson's equation) on a single node of MareNostrum5.

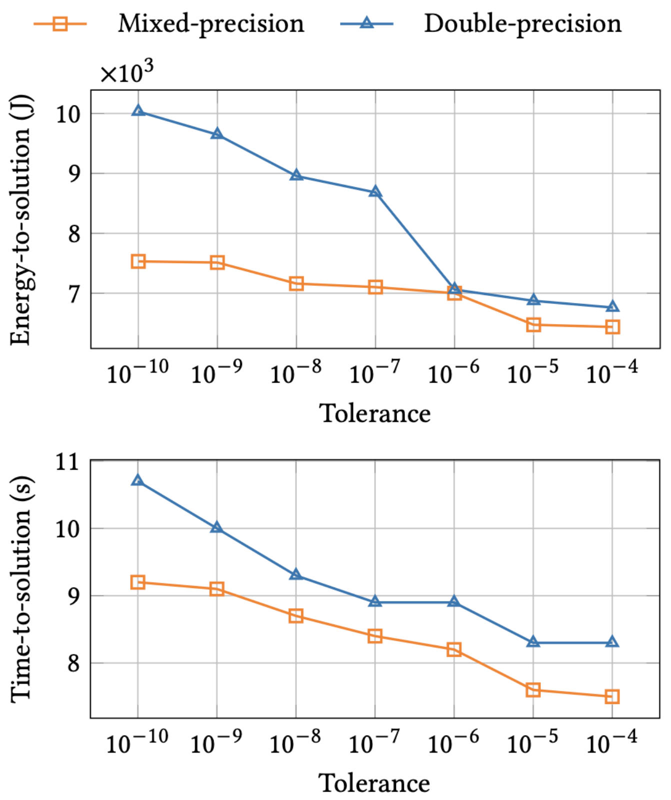
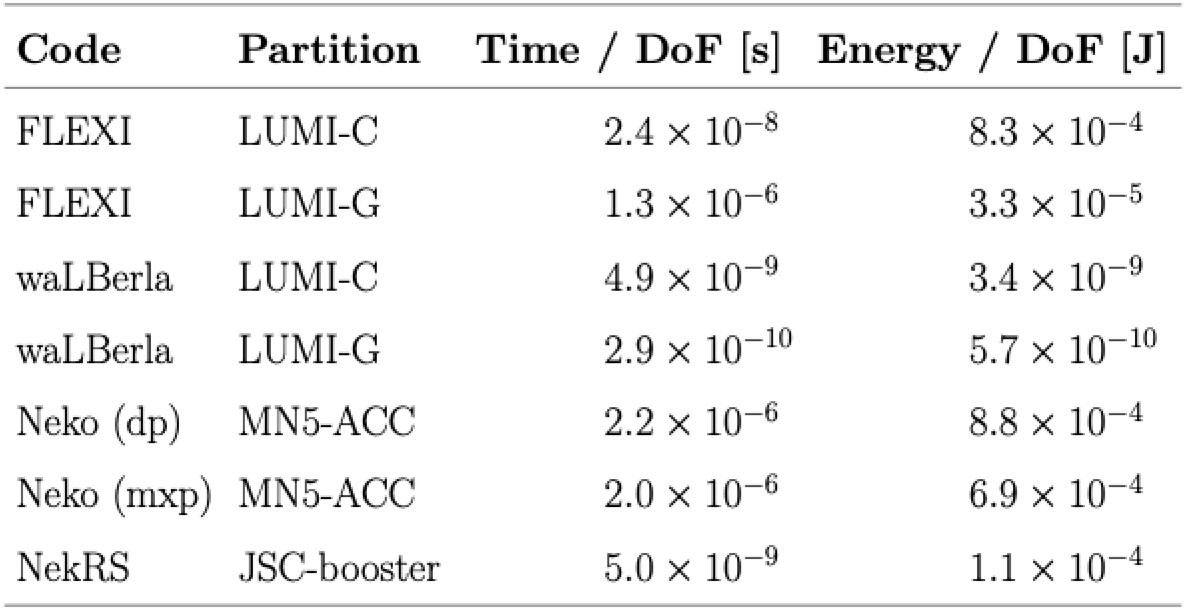


Figure 9: Energy-to-solution (top) and time-to-solution (bottom) vs. tolerance for mixed-precision and double-precision Neko (Poisson's equation) on a single node of MareNostrum5.

Summary

- Promote community-wide awareness and engagement in energy measurement
- **CEEC case studies shows energy-to-solution favors GPU and mixed precision**
- Underutilized resources reduce both performance and energy efficiency, even for optimized codes
- Sustainable exascale HPC requires joint optimization of performance, precision, and energy



Code	**Partition**	**Time / DoF [s]**	**Energy / DoF [J]**
FLEXI	LUMI-C	2.4×10^{-8}	8.3×10^{-4}
FLEXI	LUMI-G	1.3×10^{-6}	3.3×10^{-5}
waLBerla	LUMI-C	4.9×10^{-9}	3.4×10^{-9}
waLBerla	LUMI-G	2.9×10^{-10}	5.7×10^{-10}
Neko (dp)	MN5-ACC	2.2×10^{-6}	8.8×10^{-4}
Neko (mxp)	MN5-ACC	2.0×10^{-6}	6.9×10^{-4}
NekRS	JSC-booster	5.0×10^{-9}	1.1×10^{-4}

Time-to-solution and energy-to-solution required to advance one DoF for one time step on one node.

Funded by the European Union. This work has received funding from the European High Performance Computing Joint Undertaking (JU) and Sweden, Germany, Spain, Greece, and Denmark under grant agreement No 101093393.

Co-funded by
the European Union

EuroHPC
Joint Undertaking

CEEC
Centre of Excellence in Exascale CFD

Thank you
for your attention!

CEEC

Centre of Excellence in Exascale CFD