A portable algebraic implementation for reliable industrial LES

M. Mosqueda-Otero[1], À. Alsalti-Baldellou[1,2], J. Plana-Riu[1], X. Álvarez-Farré[3], G. Colomer[1], F.X. Trias[1], A. Gorobets[4], A. Oliva[1]

[1] Heat and Mass Transfer Technological Center, Universitat Politècnica de Catalunya, Spain
[2] Department of Information Engineering, University of Padova, Italy
[3] High-Performance Computing Team, SURF, The Netherlands
[4] Keldysh Institute of Applied Mathematics, Russia

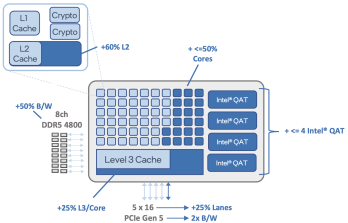
International ERCOFTAC Symposium on Engineering Turbulence Modelling and Measurements (ETMM-15)
Sep 22^{nd} - 24^{th}, 2025
Dubrovnik, Croatia

ETMM
Ercoftac Symposium on Engineering Turbulence Modelling and Measurements

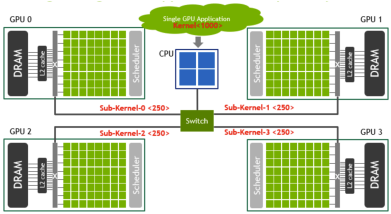
Motivation

The continuous evolution of hardware, coupled with the widespread adoption of accelerators across various tech domains, has driven the development of modern hybrid HPC architectures.

Intel Xeon 4th gen CPU architecture[1]



GPU-CPU hybrid architecture[2]



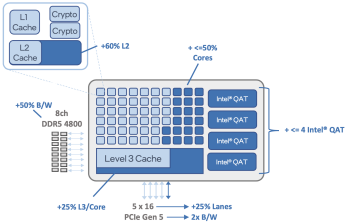
[1] D. Coyle et al. *Maximizing vCMTS Data Plane Performance with 4th Gen Intel® Xeon® Scalable Processor Architecture*. July 2023

[2] U. Milic et al. "Beyond the socket: NUMA-aware GPUs". In: *Proceedings of the 50th Annual IEEE/ACM International Symposium on Microarchitecture*. 2017. DOI: 10.1145/3123939.3124534

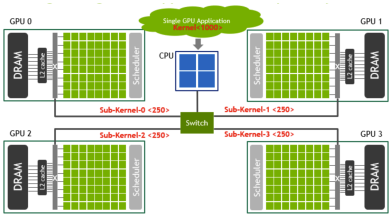
Motivation

The continuous evolution of hardware, coupled with the widespread adoption of accelerators across various tech domains, has driven the development of modern hybrid HPC architectures.

Intel Xeon 4th gen CPU architecture[1]



GPU-CPU hybrid architecture[2]



- How can we achieve portable CFD codes for different architectures and hardware vendors?

[1] Coyle et al., *Maximizing vCMTS Data Plane Performance with 4th Gen Intel® Xeon® Scalable Processor Architecture*

[2] Milic et al., "Beyond the socket: NUMA-aware GPUs"

TFA+HPC^2

TFA+HPC^2 presents thoroughly conservative discretization methods[3] on unstructured grids, build on a set of algebra-dominant kernels[4], easily portable to modern HPC architectures

Main HPC^2 kernels

Kernels	Operation
`axpy`	Linear combination of vectors
`axty`	Element-wise product of vectors
`dot`	dot product of vectors
`SpMV`	Sparse matrix-vector product

[3] F. X. Trias et al. "Symmetry-preserving discretization of Navier-Stokes equations on collocated unstructured meshes". In: *Journal of Computational Physics* 258 (2014), pp. 246–267

[4] X. Álvarez-Farré et al. "HPC^2 – A fully portable algebra-dominant framework for heterogeneous computing. Application to CFD". In: *Computers & Fluids* 173 (2018), pp. 285–292

TFA+HPC^2

TFA+HPC^2 presents thoroughly conservative discretization methods[3] on unstructured grids, build on a set of algebra-dominant kernels[4], easily portable to modern HPC architectures

Main HPC^2 kernels

Kernels	Operation
`axpy`	Linear combination of vectors
`axty`	Element-wise product of vectors
`dot`	dot product of vectors
`SpMV`	Sparse matrix-vector product

- How do they handle demands for larger-scale problems?

[3] F. X. Trias et al., "Symmetry-preserving discretization of Navier-Stokes equations on collocated unstructured meshes"

[4] X. Álvarez-Farré et al., "HPC^2 – A fully portable algebra-dominant framework for heterogeneous computing. Application to CFD"

TFA+HPC^2

TFA+HPC^2 presents thoroughly conservative discretization methods[3] on unstructured grids, build on a set of algebra-dominant kernels[4], easily portable to modern HPC architectures

Main HPC^2 kernels

Kernels	Operation
`axpy`	Linear combination of vectors
`axty`	Element-wise product of vectors
`dot`	dot product of vectors
`SpMV`	Sparse matrix-vector product

- How do they handle demands for larger-scale problems?
- Do they provide efficient, portable solutions?

[3] F. X. Trias et al., "Symmetry-preserving discretization of Navier-Stokes equations on collocated unstructured meshes"

[4] X. Álvarez-Farré et al., "HPC^2 – A fully portable algebra-dominant framework for heterogeneous computing. Application to CFD"

Numerical test

Base case

- Turbulent channel flow
- Conjugate Gradient with a Jacobi preconditioner
- Explicit time integration scheme
- Solving 10 time steps with 800 iterations per step

CPU system

- Strong and Weak scalability
- Performed in Marenostrum 5 GPP at BSC
  - CPU: Intel Xeon Platinum 8480+ (2×)
- Focus on MPI-Only vs. MPI+OpenMP

GPU system

- Strong and Weak scalability
- Performed in Marenostrum 5 ACC at BSC
  - CPU: Intel Xeon Platinum 8460Y (2×)
  - GPU: NVIDIA H100-64 GiB HBM3 (4×)
- Focus on OpenCL

[CPU] Strong Scalability

Considerations

- 1-node baseline
- Node configuration:
  - MPI-Only: 112 MPI processes
- Base workload of 525k CVs per CPU-core

  Mesh size:
  - $350 \times 480 \times 350$ - 58.8M CVs

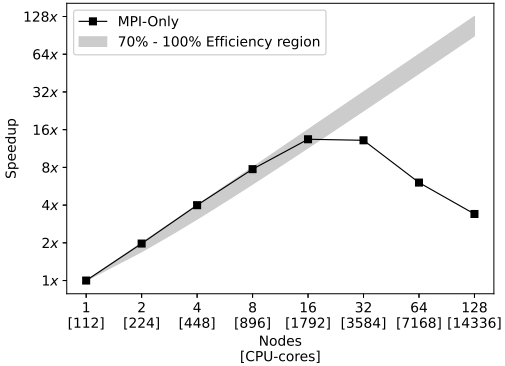


Figure: MPI-Only strong scalability on Marenostrum 5 GPP

[CPU] Strong Scalability

Considerations

- 1-node baseline
- Node configuration:
  - MPI-Only: 112 MPI processes
  - MPI+OpenMP: 2 MPI processes with 56 computational threads per process
- Base workload of 525k CVs per CPU-core

  Mesh size:
  - $350 \times 480 \times 350$ - 58.8M CVs

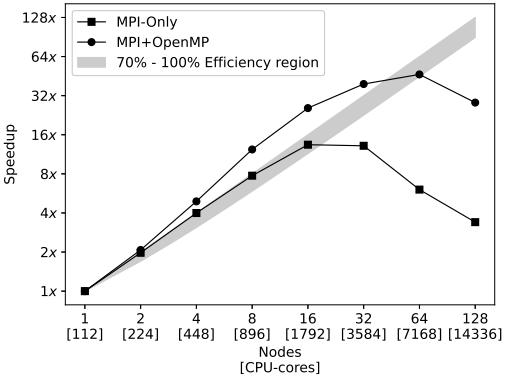


Figure: MPI-Only vs. MPI+OpenMP strong scalability on Marenostrum 5 GPP

[CPU] Weak Scalability

Considerations

- Node configuration:
  - MPI+OpenMP: 2 MPI processes with 56 computational threads per process
- Starting with 1 node (112 CPU-cores) up to 128 nodes (14336 CPU-cores)
- Base workload of 525k CVs per CPU-core

  Mesh sizes:
  - 1 node: $350 \times 480 \times 350$ - 58.8M
  - 16 nodes: $800 \times 1470 \times 800$ - 940.8M
  - 128 nodes: $2000 \times 2352 \times 1600$ - 7.52B

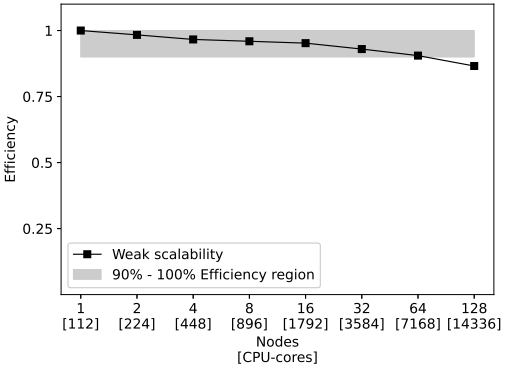


Figure: MPI+OpenMP weak scalability on Marenostrum 5 GPP

[GPU] Strong Scalability

Considerations

- 1-node baseline (4 GPUs)
- Node configuration:
  - 4 MPI processes - 1 MPI per GPU card
  - 54 computational threads
  - 2 communication threads
- Base workload of 42.5M CVs per GPU

  Mesh size:
  - 1-node: $500 \times 1000 \times 340$ - 170M CVs

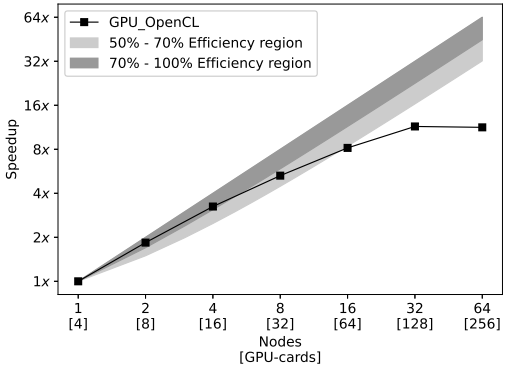


Figure: GPU strong scalability analysis on Marenostrum 5 ACC

[GPU] Weak Scalability

Considerations

- Node configuration:
  - 4 MPI processes - 1 MPI per GPU card
  - 54 computational threads
  - 2 communication threads
- Starting with 1 node (4 GPUs) up to 64 nodes (256 GPUs)
- Base workload of 42.5M CVs per GPU

  Mesh sizes:
  - 1 node: $500 \times 1000 \times 340$ - 170M
  - 8 nodes: $1000 \times 1360 \times 1000$ - 1.36B
  - 64 nodes: $2000 \times 2720 \times 2000$ - 10.88B

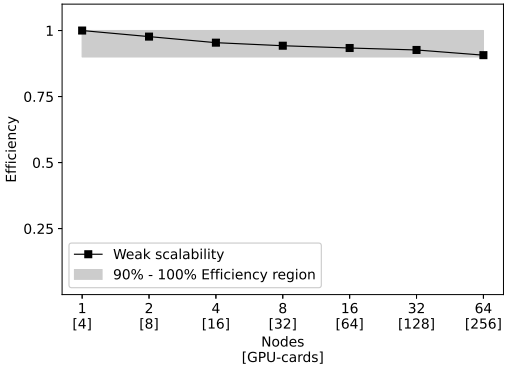


Figure: GPU weak scalability analysis on Marenostrum 5 ACC

Parameters

Equivalent Arithmetic Intensity (AI_{eq})

$$AI_{eq} = \frac{\sum_{k \in K} \alpha_k FLOPS_k}{\sum_{k \in K} \alpha_k BYTES_k}$$

Equivalent Performance (P_{eq})

$$P_{eq} = \sum_{k \in K} \alpha_k P_k$$

Data Throughput (DT_{eq})

$$DT_{eq} = \sum_{k \in K} \alpha_k DT_k$$

Where K refers to a set of HPC^2 kernels

Kernels	Operation
`axpy`	Linear combination of vectors
`axty`	Element-wise product of vectors
`dot`	dot product of vectors
`SpMV`	Sparse matrix-vector product

and

$$\alpha_k = \sum_{k \in K} \frac{N_k}{N_{total}}$$

CPU Performance

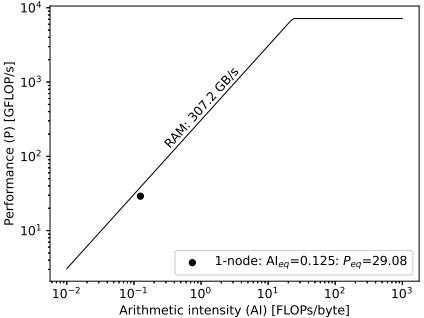


Figure: Roofline analysis on Marenostrum 5 GPP; showing 1 node (112 CPU-cores) with 58.8M CVs grid

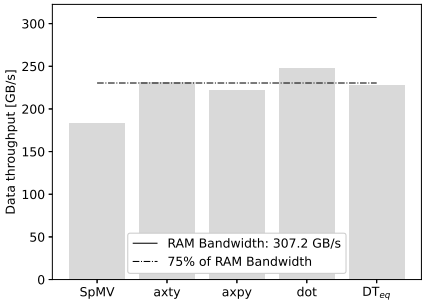


Figure: Data Throughput analysis on Marenostrum 5 GPP; showing 1 node (112 CPU-cores) with 58.8M CVs grid

CPU Performance

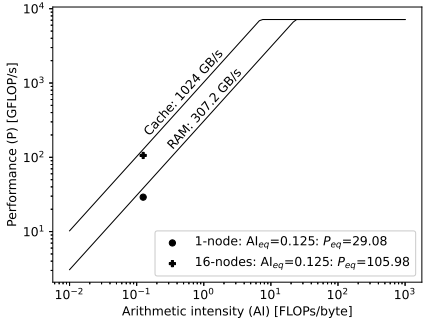


Figure: Hierarchical roofline analysis on Marenostrum 5 GPP; showing 1 node (112 CPU-cores) with 58.8M CVs grid, and 16 nodes (1792 CPU-cores) for MPI+openMP strong scalability results

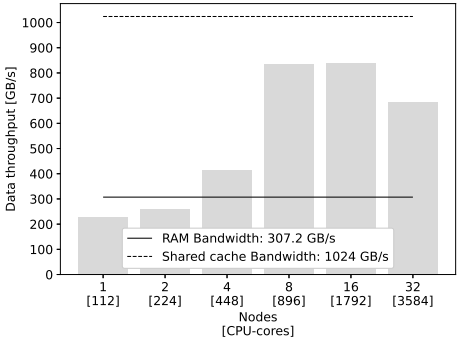


Figure: Equivalent Data Throughput results for strong scalability data on Marenostrum 5 GPP

GPU Performance

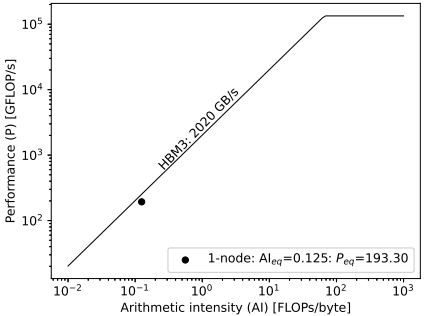


Figure: Roofline analysis on Marenostrum 5 ACC; showing 1 node (4 GPU-cards) with 170M CVs grid

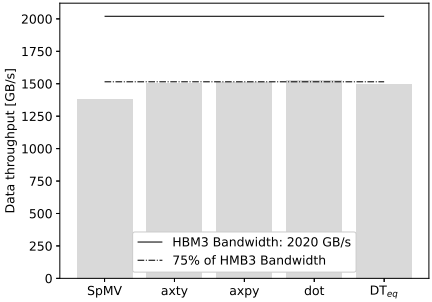


Figure: Data Throughput analysis on Marenostrum 5 ACC; kernels and equivalent data transfer for 1-node case

Conclusions

- TFA+HPC^2 design improves portability into different HPC architectures.
- Efforts to increase the arithmetic intensity are required to improve its memory-bound behavior.

Conclusions

- TFA+HPC^2 design improves portability into different HPC architectures.
- Efforts to increase the arithmetic intensity are required to improve its memory-bound behavior.
- CPU systems exhibit superior strong scalability, with the hybrid paradigm (MPI+OpenMP) delivering higher performance than MPI-only, primarily due to the benefits of cache utilization and reduced communication overhead.

Conclusions

- TFA+HPC^2 design improves portability into different HPC architectures.
- Efforts to increase the arithmetic intensity are required to improve its memory-bound behavior.
- CPU systems exhibit superior strong scalability, with the hybrid paradigm (MPI+OpenMP) delivering higher performance than MPI-only, primarily due to the benefits of cache utilization and reduced communication overhead.
- Finally, weak scaling analysis delivers great efficiency, showing the capability of this implementation to scale to demanding Industrial applications.

Future work

- To perform large-scale urban simulations leveraging spatial regularities[5]
- Continue exploring strategies to increase GPU computation.

[5] À. Alsalti-Baldellou et al. "Lighter and faster simulations on domains with symmetries". In: *Computers & Fluids* 275 (2024), p. 106247. ISSN: 0045-7930. DOI: https://doi.org/10.1016/j.compfluid.2024.106247