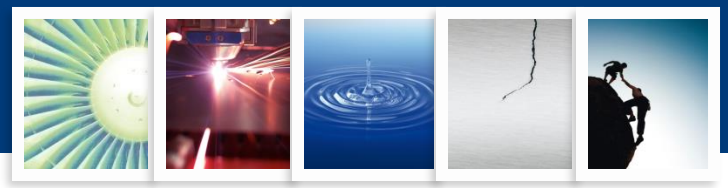


Cenaero

Tiering up:

Scaling scientific workflow from local to ... exascale resources

Lumi-BE users day

Brussels, 17 December 2025

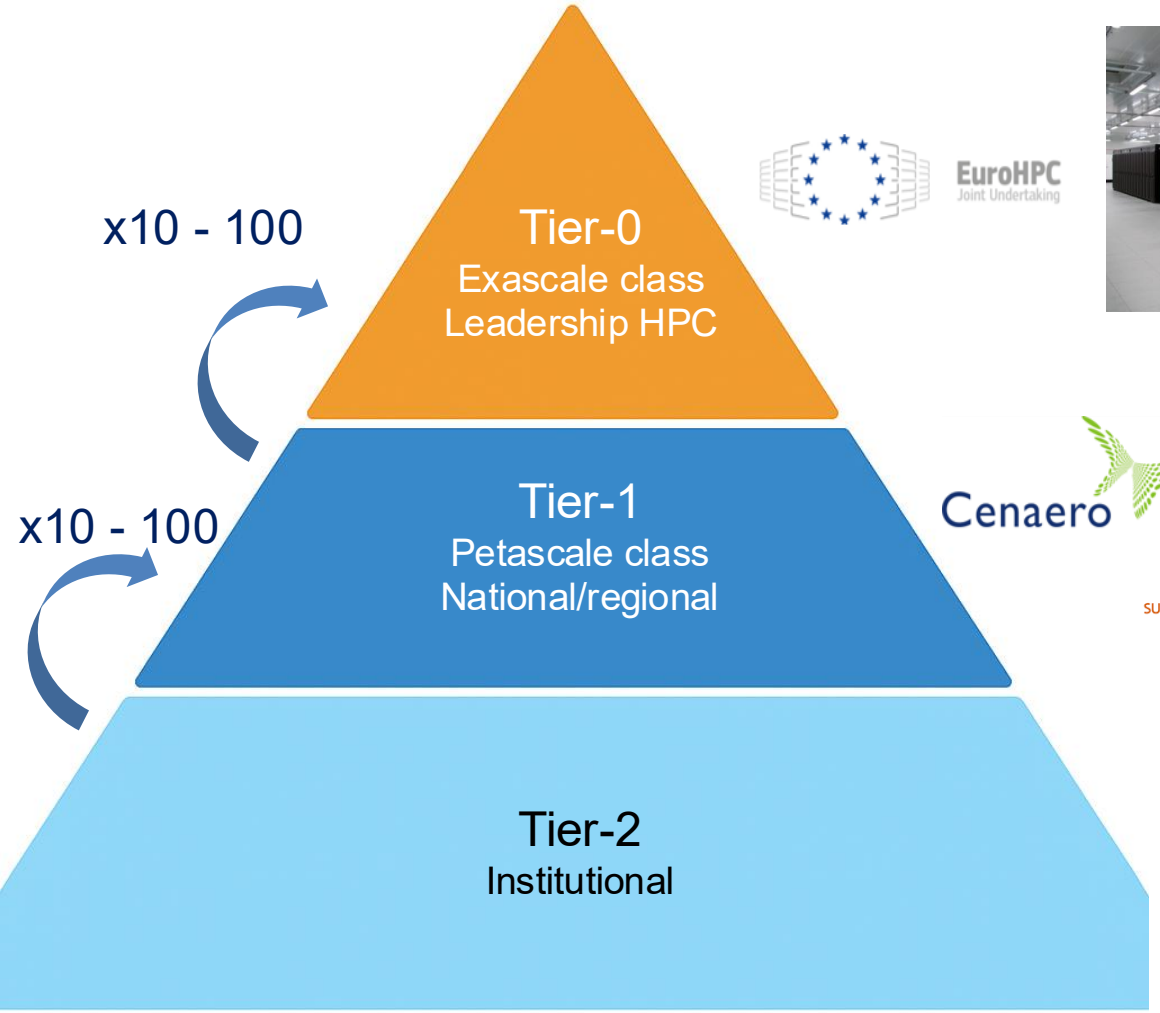
Michel Rasquin, Nicolas Goffart, Manuel Diaz, Giove De Cosmo, Koen Hillewaert, Thomas Toulorge

Cenaero, Gosselies, Belgium

Contact: michel.rasquin@cenaero.be

Doc. ref.: GT2025-151633

Tier concept



EuroHPC Joint Undertaking

- **Exascale system (1):** Jupiter
- **Pre-exascale systems (3):** Lumi, Leonardo, Marenostrum 5
- **Mid-range petascale systems (7):** Meluxina, Karolina, Vega, Discoverer, Deucalion, Daedalus, Arrhenius

Cenaero

CÉCI

VLAAMS SUPERCOMPUTER CENTRUM Vlaanderen is supercomputing

Lucia

Sofia

CÉCI

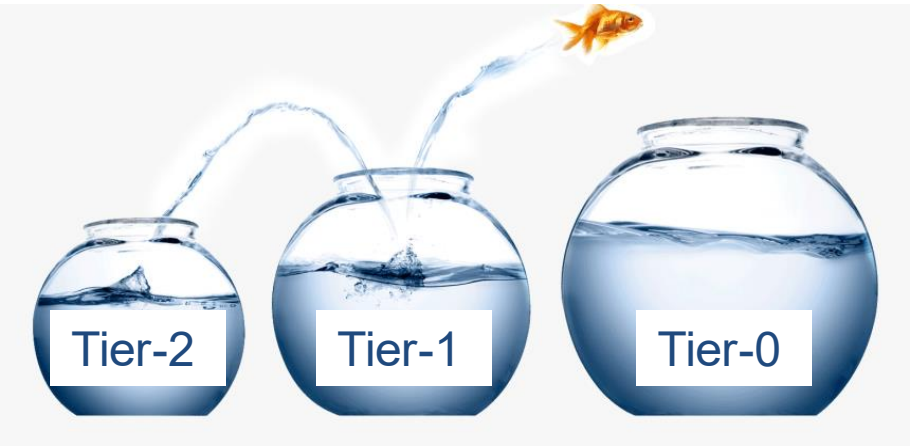
VLAAMS SUPERCOMPUTER CENTRUM Vlaanderen is supercomputing

All universities

Cenaero

Access

- **When is it time to request an allocation on a bigger Tier?**
  - **When your computation takes too much time**
  - **When your problem does not fit in memory**
  - **Preparatory access to larger systems is essential**
- **Project based access**
  - **Tier-1 Lucia:**
    - **Academia:** https://www.ceci-hpc.be/projetstier1.html
    - **Research centers and industries**: https://www.cenaero.be/en/hpc
  - **Tier-0**
    - **EuroHPC regular calls:** https://access.eurohpc-ju.europa.eu/
    - **Lumi-BE: Belgium is the second contributor of the Lumi consortium** (Finland, Belgium Belgium, Czech Republic, Denmark, Estonia, Iceland, Norway, Poland, Sweden, and Switzerland)
      - 200 million €, 380 PFlops,
      - ~12,000 AMD Mi250x GPUs on Lumi-G and ~262,000 AMD CPU cores on Lumi-C
      - 50% of the compute time distributed through EuroHPC regular calls
      - The other 50% is distributed among the participating countries according to their contribution
        → 7.4% of the total compute time on Lumi allocated to Lumi-BE
      - https://www.enccb.be/GettingAccess



Cenaero

Tier-0 access

- **LUMI-BE Preparatory**
  - **Objective**: prepare access to Tier-0
  - ≤ 500 CPU.kH or 25 GPU.kH
  - Intended for:
    - code porting / optimization (AMD / LUMI)
    - benchmarking & scalability studies (Tier-1 → LUMI)
    - preparation of EuroHPC proposals
  - No production science
  - Duration: 4 months → up to 1 year (with justification)

- **Lumi-BE Regular**
  - **3 calls per year (02/03/26, 01/06/26 and 05/10/26)**
  - **Objective:** large-scale scientific production
  - **≤ 10 core.MH on Lumi-C** or **500 GPU.kH on Lumi-G**
  - **Fixed duration:** 1 year, non-extendable
  - Code must already be ready and scalable

- **EuroHPC Regular**
  - **2 calls per year (March and September)**
  - **Objective:** large-scale scientific production
  - **Last call: ≥ 7.7M core.MH on Lumi-C** or **≥ 160 GPU.kH on Lumi-G**
  - **Fixed duration:** 1 year, non-extendable
  - Code must already be ready and scalable

SYSTEM*	SITE (COUNTRY)	PARTITION	PROCESSOR	ACCELERATOR	TOTAL RESOURCES**	MIN. (MAX.) REQUEST***
JUPITER	Jülich (DE)	**JUPITER Booster**	NVIDIA Grace	NVIDIA Hopper	527 040	25 000 (Max. 220 000)
LUMI	CSC (FI)	**LUMI-C**	AMD Epyc	-	1 275 760	60 000 (Max. 120 000)
		LUMI-G		AMD Instinct	309 181	20 000 (Max. 150 000)
MN5 MARENOSTRUM	BSC (ES)	**MN5 GPP**	Intel Sapphire Rapids	-	3 321 907	60 000 (Max. 230 000)
		MN5 ACC		Nvidia Hopper	100 000	20 000 (Max. 100 000)

Regular access

- **Lumi-BE**
  - **Description of the research project**
  - **Technical readiness**
    - Real benchmarks
    - Scalability on comparable architectures (Tier-1 / LUMI)
    - Efficiency
      - Use of full nodes (especially GPU nodes)
    - **Compatibility**
      - AMD CPUs and GPUs (no CUDA)
      - Cray programming environment / ROCm
  - **Coherence of the requested resources**
    - CPU/GPU, memory, storage billed in **TB·hours**

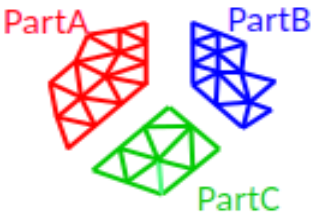
- **EuroHPC regular**
  - Same as Lumi-BE + strong scientific review

- **Scalable solver**
  - **Hide P2P communications with computations**
  - **Profile your code to identify performance bottlenecks**
- **Load balancing**
  - **Pre-processing**
  - **Partitioning for domain decomposition approaches (e.g. DGM, FEM, FV)**
- **Efficient IO**
- **Post-processing / in-situ analysis**
- **GPU**
- **Portability**
- **Robust hardware and system software**

Load balancing

- Essential to reduce waiting time in P2P and global communication
- Target applications: CFD codes based on domain decomposition approaches (DGM, FEM)
- METIS usually good at balancing elements ... but not necessarily other mesh entities

- Fidelio developed at Cenaero
  - Explicit DGM flow solver on GPU
  - Communications based on ghost elements
  - Cost of volume integrals proportional to the number of elements
  - Cost of the flux evaluation linked to the number of faces

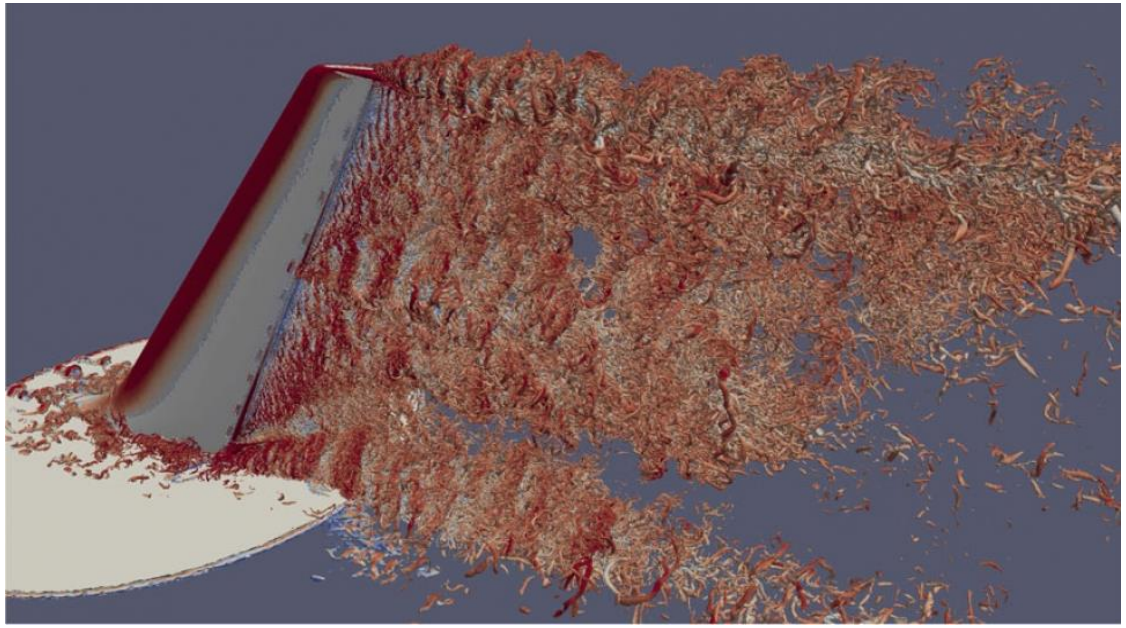


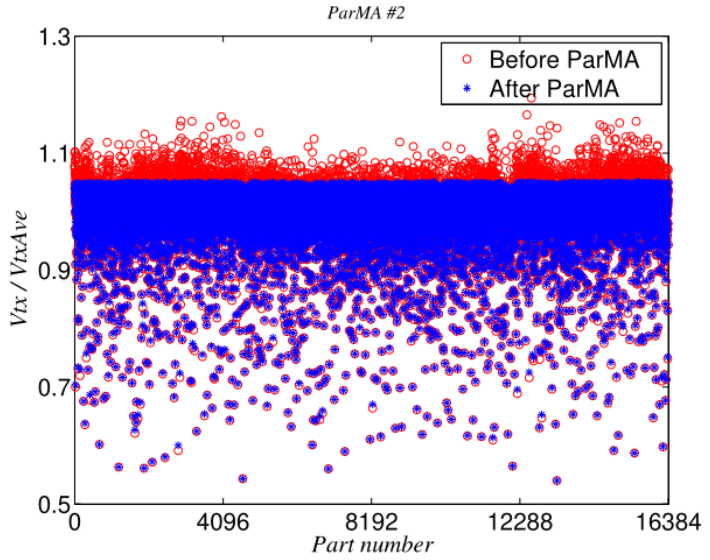
- PHASTA developed at U. Colorado Boulder (Prof. Kenneth Jansen)
  - Implicit massively parallel FEM flow solver
  - Communication based on duplicate vertices on part boundary (no ghost cells)
  - Cost of the linear system assembly proportional to the number of elements
  - Cost of the linear system solve proportional to the number of vertices

Cenaero

Load balancing

- **Flow control application over the vertical tail of a commercial aircraft with PHASTA**
- **1.2B cell anisotropic mesh**
- **Local ParMETIS applied to 8K partitions to create 16K, 32K, ...1024K partitions on BGQ Mira@ALCF (2018)**
- **ParMA (Partitioning using Mesh Adjacencies) library developed at Renselaer Polytechnic Institute used to improve load balancing of other mesh entities:** elements, edges, faces, vertices





Vertex imbalance spikes.

Imbalance is often limited to a small number of heavily loaded parts, referred to as spikes, which limit application scalability

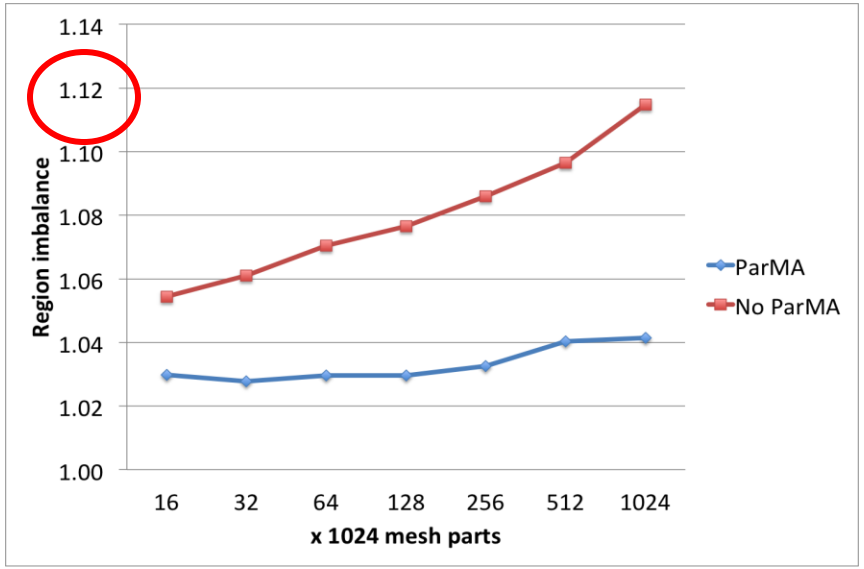
U.S. DEPARTMENT OF ENERGY INCITE LEADERSHIP COMPUTING

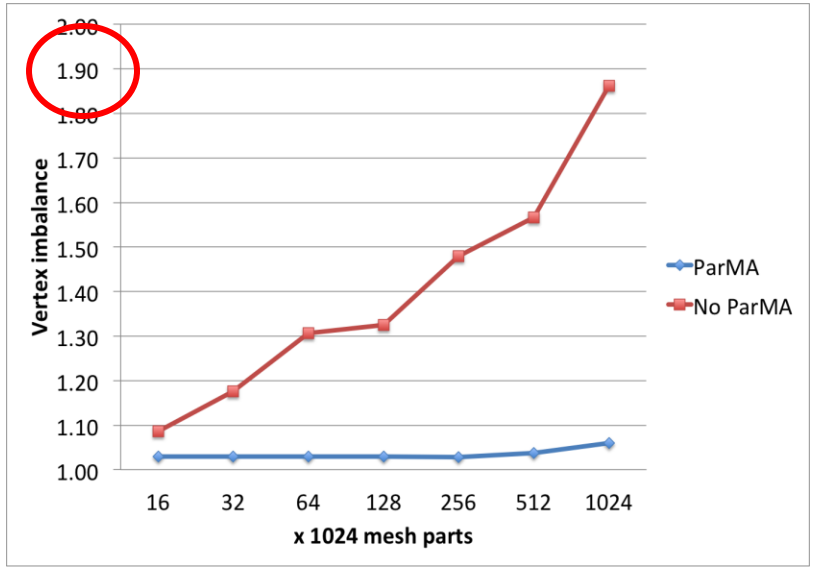
Argonne NATIONAL LABORATORY

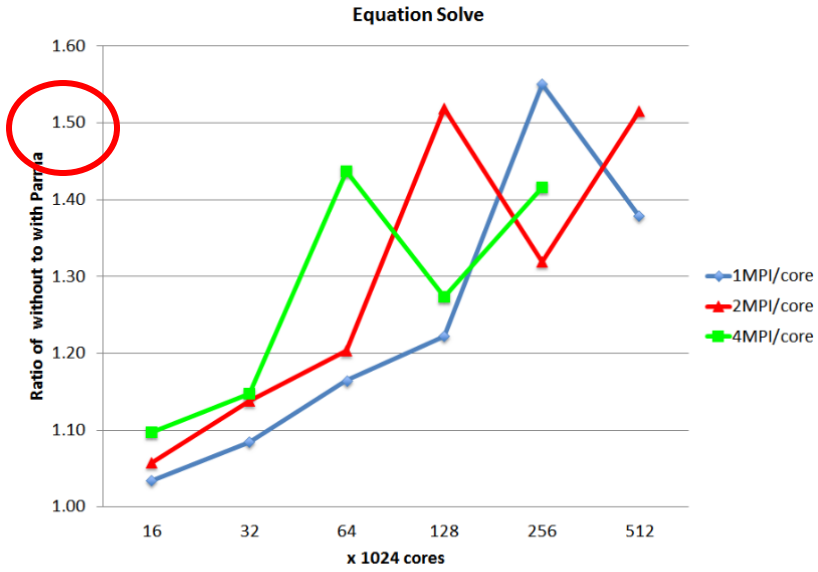
Cenaero

Load balancing

- **Load balancing improvement with ParMA and impact on the solver**







Improvement of PHASTA performance with ParMA.

- **The advent of GPU supercomputers releases the pressure on mesh partitioning imbalance...**
  … but still keep an eye on low order entities

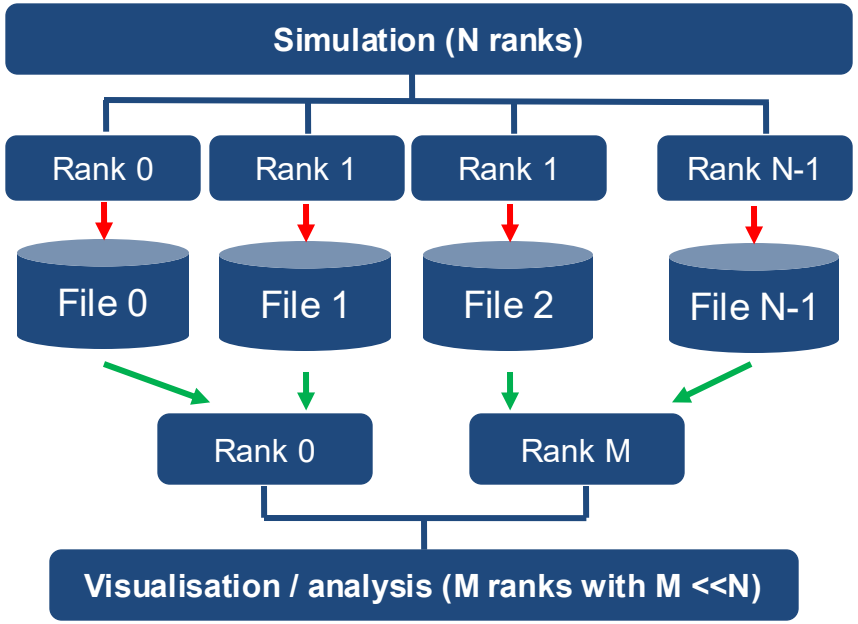
Cenaero

Input/Output

- **I/O is needed for checkpointing and post-processing / data analysis in all programs...**
- **… but it is often overlooked**
- **How to transfer data efficiently from thousands of nodes to physical disks?**
- **How to reload them for post-processing / data analysis later ?**
- **Most parallel filesystems have performance problems for large number of (small) files**
- **Without parallelization, I/O becomes a scalability bottleneck for every application**
  - Collective I/O with MPI-IO, high-level libraries (HDF5, netcdf, etc)
- **Who will consume the data and how? Visualisation / data analysis?**

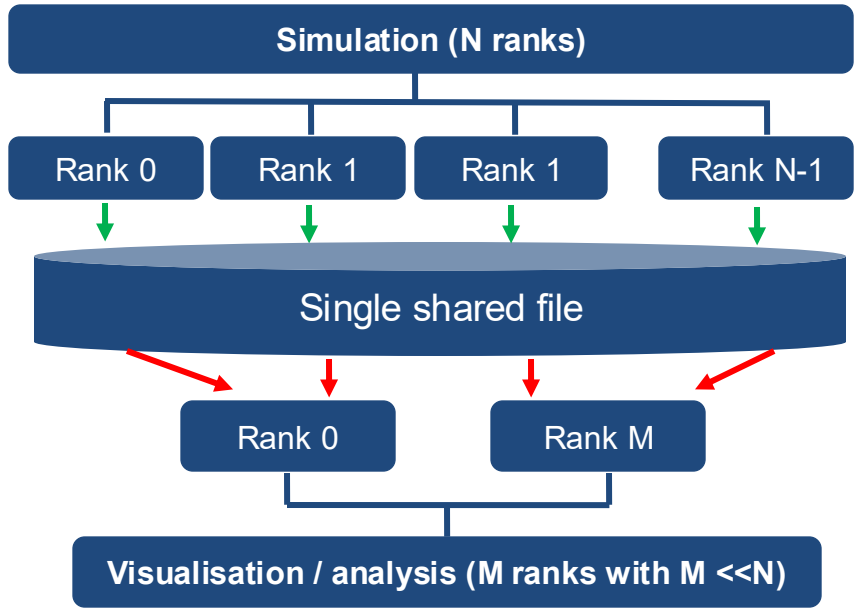
  → Plan the whole data workflow

One file per MPI rank



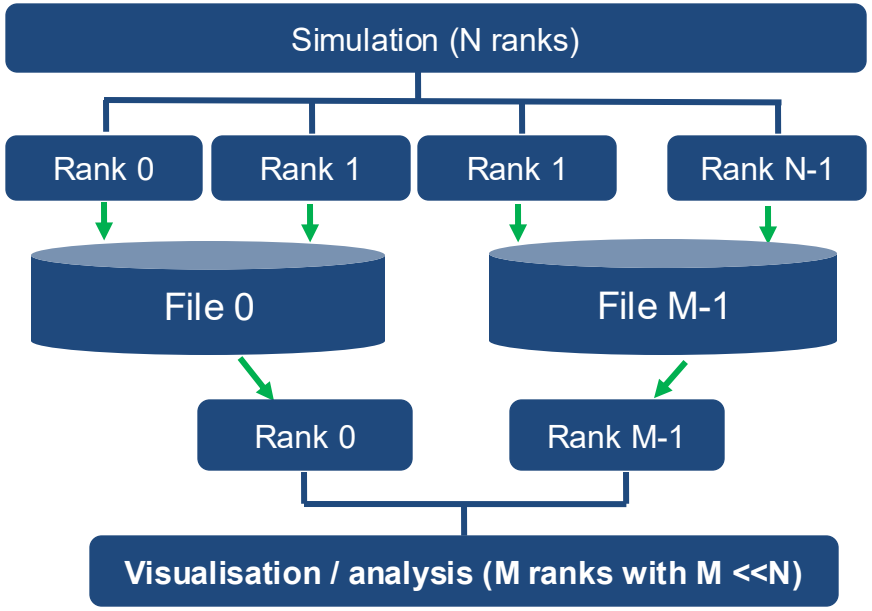
Simulation	• 1000s of small files • Meta data bottleneck • Difficult to manage (transfer, etc)
Visualisation	• Usually no collective IO • Reading 1000s of small files is slow but usually works

MPI-IO with a single shared file



Simulation	• Collective IO • High performance • Easy to handle
Visualisation	• Usually no collective IO • IO contention when many ranks access a single giant file independently

MPI-IO with M files shared between N processes (M<<N)



Simulation

- **Collective IO**
- **High performance**
- **Easy to handle**

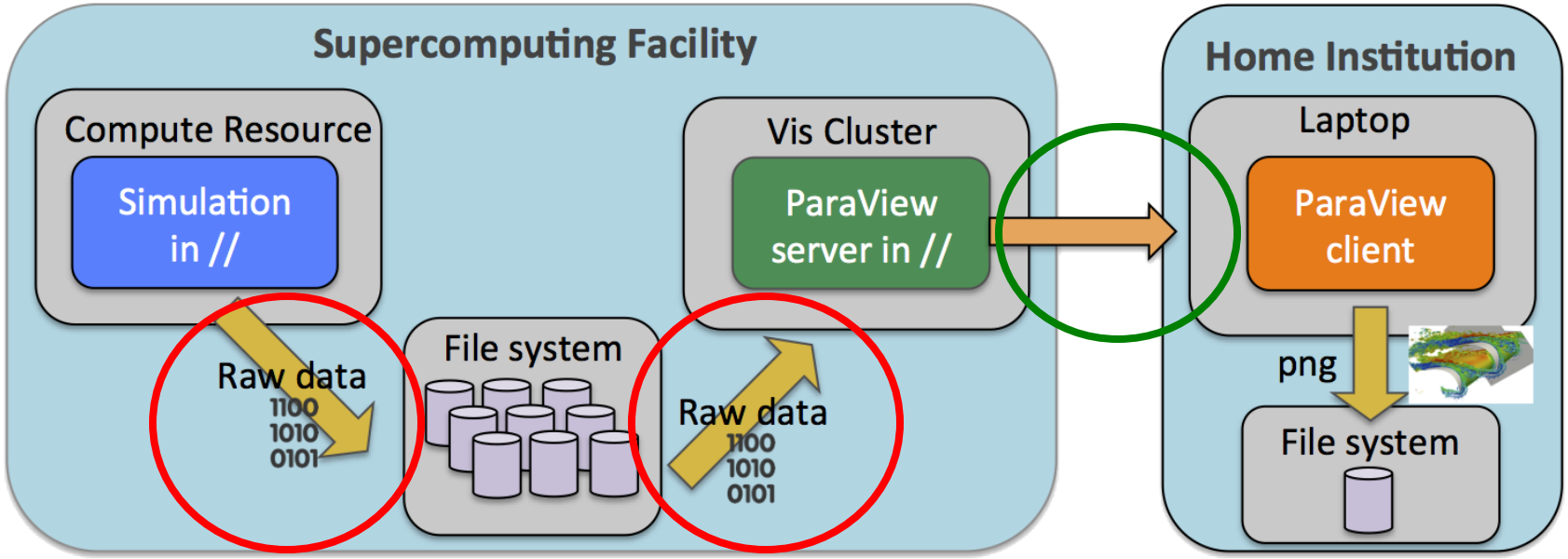
Visualisation

- **Usually no collective IO**
- **IO contention avoided with only a few files accessed independently**

→ Alternative: develop a reader based on collective IO for visualisation purposes

Cenaero

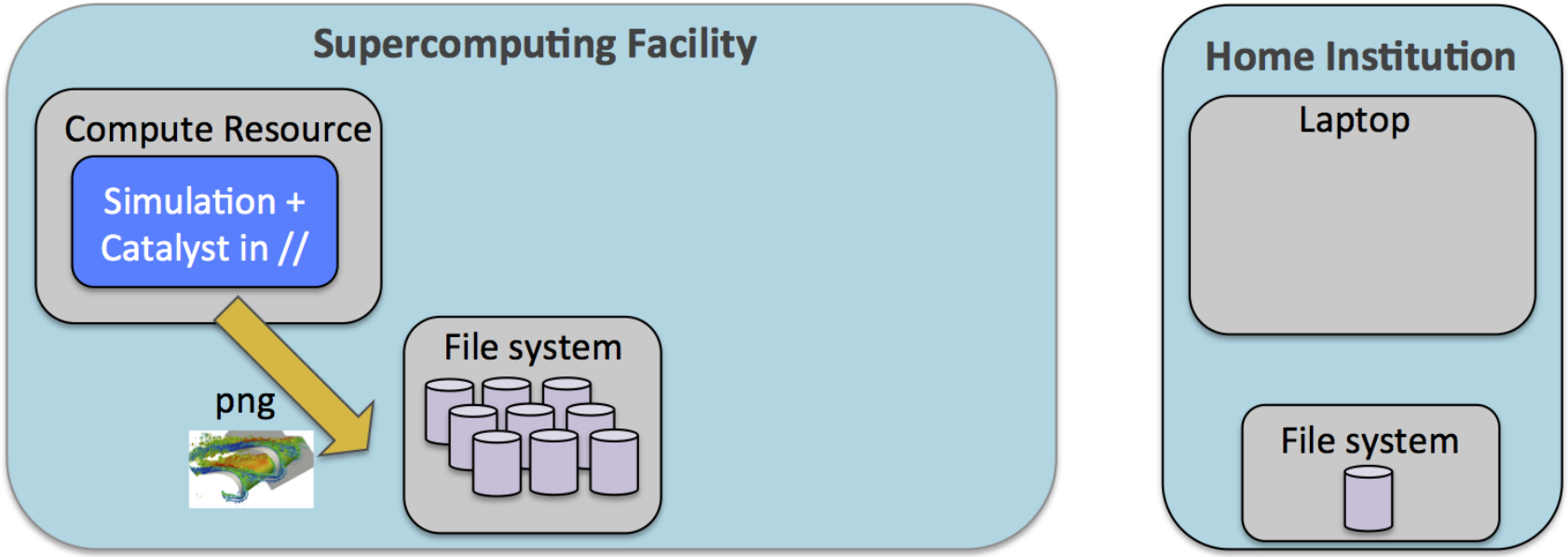
- **Client/server post-processing mode**



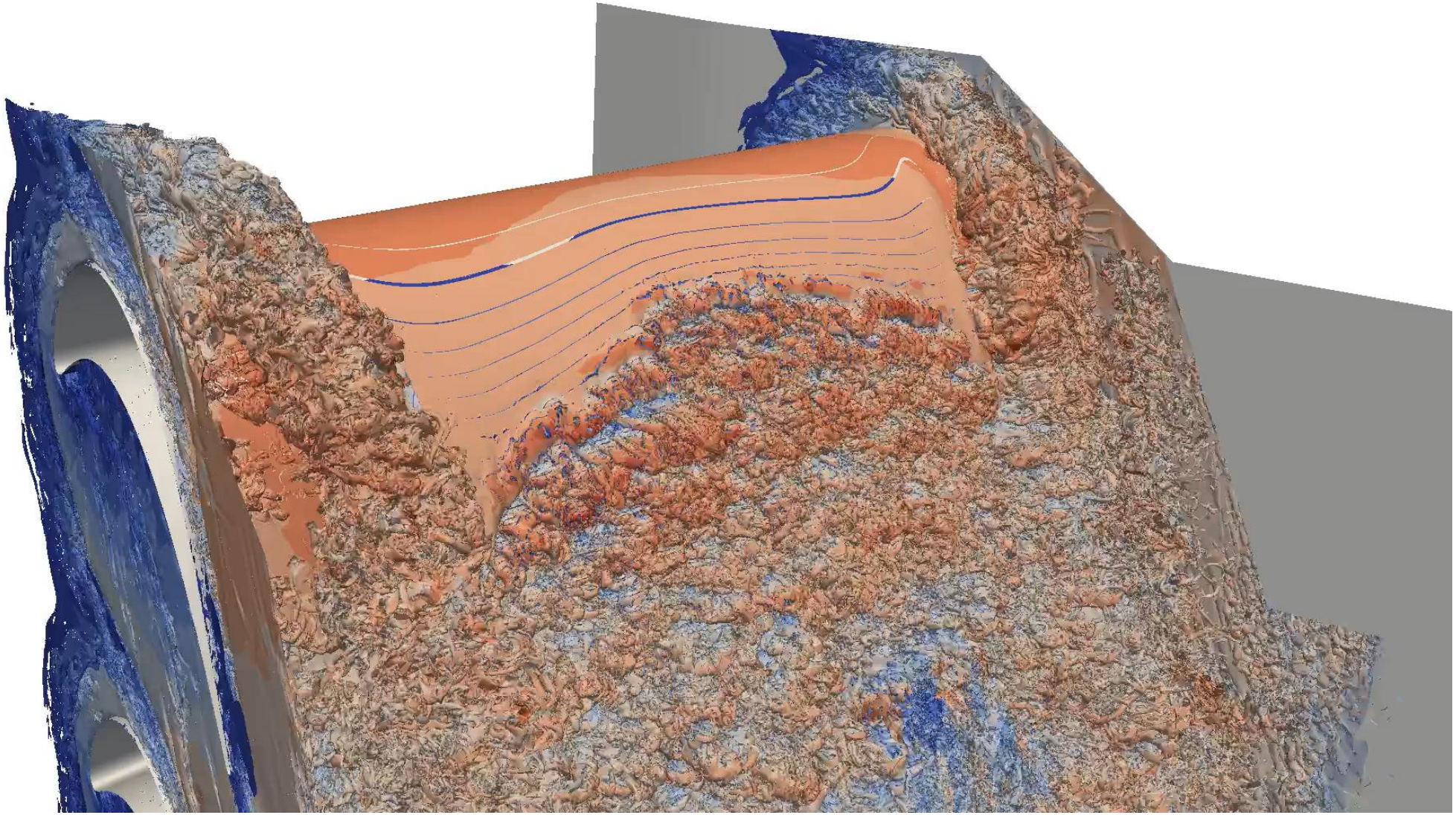
  - **Reduced data transfer (pixels)**
  - **IO performance is not keeping pace with increasing compute power**
  - **Vis cluster usually 2-3 orders of magnitude smaller than compute resource**
  - **Still slow and inadequate for time-accurate simulation of large data sets**

In situ visualisation/co-processing
How?

- Couple simulation code to an analysis or visualization library (e.g. Catalyst)
- Run the analysis concurrently with the simulation
- Batch in situ visualization chain
  - Writes png instead of raw data in an autonomous way
  - Works well for embarrassingly parallel filters (slices, contours)



In situ visualisation/co-processing
Example



- **MTU LP turbine blade T161**
- **Re=90k**
- **Full span, diverging end walls**

The advent of GPU

Rank	System	Cores	Rmax (PFlop/s)	Rpeak (PFlop/s)	Power (kW)
1	**El Capitan** - HPE Cray EX255a, AMD 4th Gen EPYC 24C 1.8GHz, AMD Instinct MI300A, Slingshot-11, TOSS, **HPE** DOE/NNSA/LLNL United States	11,340,000	1,809.00	2,821.10	29,685
2	**Frontier** - HPE Cray EX235a, AMD Optimized 3rd Generation EPYC 64C 2GHz, AMD Instinct MI250X, Slingshot-11, HPE Cray OS, **HPE** DOE/SC/Oak Ridge National Laboratory United States	9,066,176	1,353.00	2,055.72	24,607
3	**Aurora** - HPE Cray EX - Intel Exascale Compute Blade, Xeon CPU Max 9470 52C 2.4GHz, Intel Data Center GPU Max, Slingshot-11, **Intel** DOE/SC/Argonne National Laboratory United States	9,264,128	1,012.00	1,980.01	38,698
4	**JUPITER Booster** - BullSequana XH3000, GH Superchip 72C 3GHz, NVIDIA GH200 Superchip, Quad-Rail NVIDIA InfiniBand NDR200, RedHat Enterprise Linux, **EVIDEN** EuroHPC/FZJ Germany	4,801,344	1,000.00	1,226.28	15,794
5	**Eagle** - Microsoft NDv5, Xeon Platinum 8480C 48C 2GHz, NVIDIA H100, NVIDIA Infiniband NDR, **Microsoft Azure** Microsoft Azure United States	2,073,600	561.20	846.84	
6	**HPC6** - HPE Cray EX235a, AMD Optimized 3rd Generation EPYC 64C 2GHz, AMD Instinct MI250X, Slingshot-11, RHEL 8.9, **HPE** Eni S.p.A. Italy	3,143,520	477.90	606.97	8,461
7	**Supercomputer Fugaku** - Supercomputer Fugaku, A64FX 48C 2.2GHz, Tofu interconnect D, **Fujitsu** RIKEN Center for Computational Science Japan	7,630,848	442.01	537.21	29,899
8	**Alps** - HPE Cray EX254n, NVIDIA Grace 72C 3.1GHz, NVIDIA GH200 Superchip, Slingshot-11, HPE Cray OS, **HPE** Swiss National Supercomputing Centre (CSCS) Switzerland	2,121,600	434.90	574.84	7,124
9	**LUMI** - HPE Cray EX235a, AMD Optimized 3rd Generation EPYC 64C 2GHz, AMD Instinct MI250X, Slingshot-11, **HPE** EuroHPC/CSC Finland	2,752,704	379.70	531.51	7,107
10	**Leonardo** - BullSequana XH2000, Xeon Platinum 8358 32C 2.6GHz, NVIDIA A100 SXM4 64 GB, Quad-rail NVIDIA HDR100 Infiniband, **EVIDEN** EuroHPC/CINECA Italy	1,824,768	241.20	306.31	7,494

GPU AMD MI300A

GPU AMD MI250x

GPU Intel data center Max

GPU NV GH200

GPU NV H100

GPU AMD MI250x

CPU ARM A64FX

GPU NV GH200

GPU AMD MI250x

GPU NV A100

- **Top 20 supercomputers**
  - **GPU NVIDIA: 12**
  - **GPU AMD: 6**
  - **GPU Intel: 1**
  - **CPU ARM: 1 (only exception!)**
- **HPC: a niche market initially driven by gaming and now by AI**
- **GPU vs CPU: better Flop-per-Watt ratio**

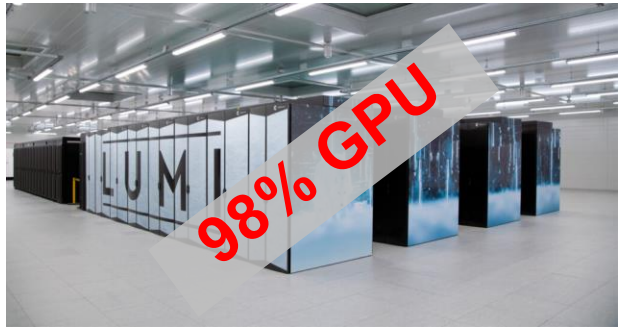
Cores	Rmax (PFlop/s)	Rpeak (PFlop/s)	Power (kW)
11,340,000	1,809.00	2,821.10	29,685
7,630,848	442.01	537.21	29,899

- **Green top 500 dominated by GH200, H100 and Mi300 with 60 - 70 GFlops/Watt**
- **Best ARM CPU: 16 GFlops/Watt**
- **Lumi-G: 53 GFlops/Watt, Lumi-C: 5 GFlops/Watt**

4x

10x

Lucia-G@Cenaero - #457 TOP500
Rmax Lucia-G: 2.78 PFlops/s
Rmax Lucia-C: 1.14 PFlops/s



LUMI-G@CSC - #9 TOP500
LUMI-C@CSC - #237 TOP500
Rmax Lumi-G: 379.70 PFlop/s
Rmax Lumi-C: 8.48 PFlops/s

Cenaero

Lucia
CPU vs GPU comparison

Lucia CPU					Lucia GPU
	Rpeak (DP)	1505 TFLOPS		Rpeak (DP)	3900 TFLOPS
	Nb. nodes	300		Nb. GPU's	200
	Rpeak/node	5.02 TFLOPS	x 3.9	Rpeak/GPU	19.5 TFLOPS
	Bandwidth/node	409.6 GB/s	x 3.8	Bandwidth/GPU	1555 GB/s
	Arith. intensity	12.25 FLOP/Byte	x 1	Arith. intensity	12.54 FLOP/Byte
	Mem./node	240 GB	x 0.17	Mem./GPU	40 GB
	Cost	128 BUh	x 1	Cost	128 BUh

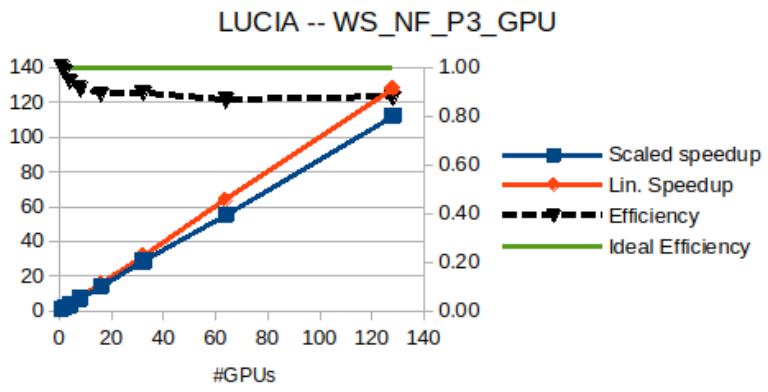
Cenaero

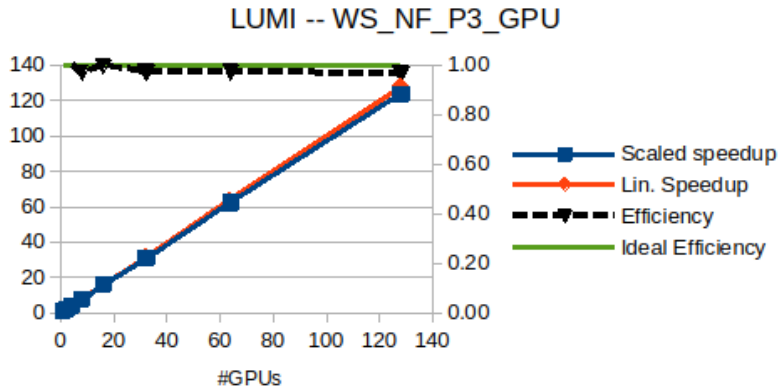
Some GPU porting initiatives
Virtual Lab FEDER

- **Uliège: implementation of a multi-GPU solver GmshDG for Maxell equations**
  - Lucia: Dof/s increased by 4x on GPU vs CPU, energy cost reduced by 6x
  - See Orian Louant's presentation

- **UCLouvain: development of a multi-physics HPC platform Murphy for the resolution of PDES**
  - Fluid (wake prediction), fluid + MHD, fluid + icing
  - Lucia: cost of the Poisson solver reduced by 4x on GPU vs CPU

- **Cenaero: implementation of a new multi-GPU DGM flow solver Fidelio**
  - Billing unit cost reduced on Lucia by 5x – 8x on GPU vs historical CPU code





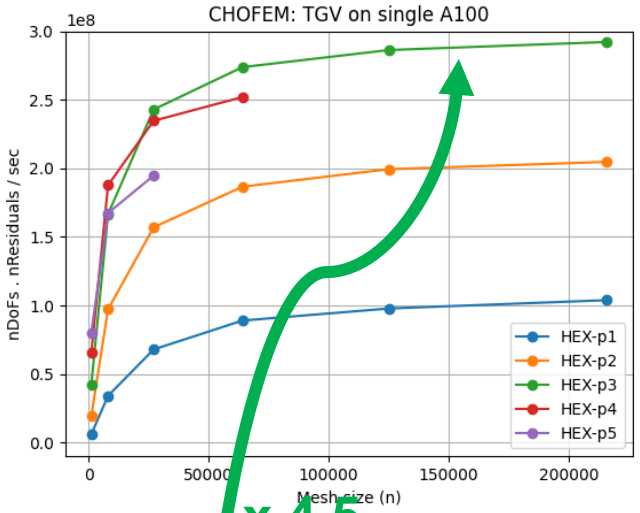
Fidelio weak scaling on GPU
- **DGM hexa P3**
- **Lucia (NVIDIA A100): 130M DoF/GPU**
- **Lumi (AMD Mi250x): 164M DoF/GCD**

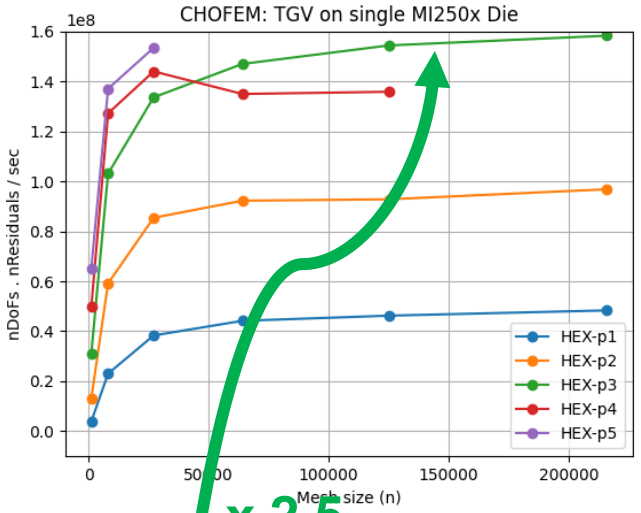
Single GPU vs CPU node performance

Lucia

Lumi

1 GPU

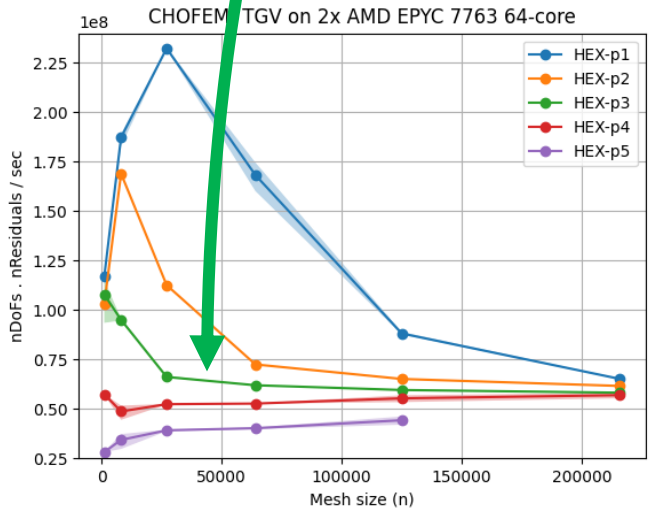


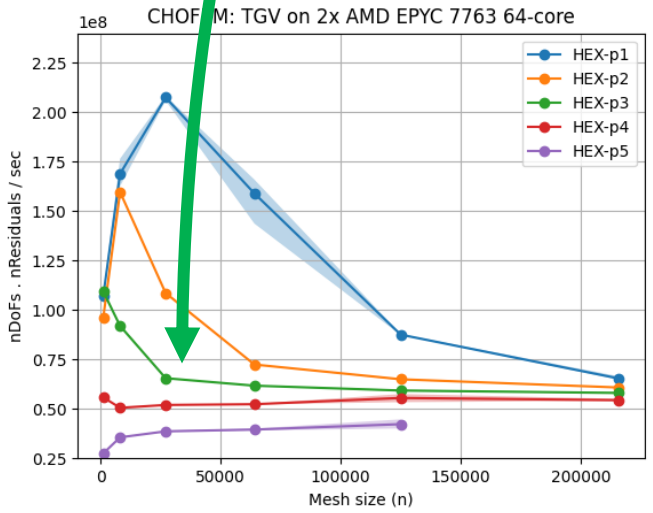


x 4.5

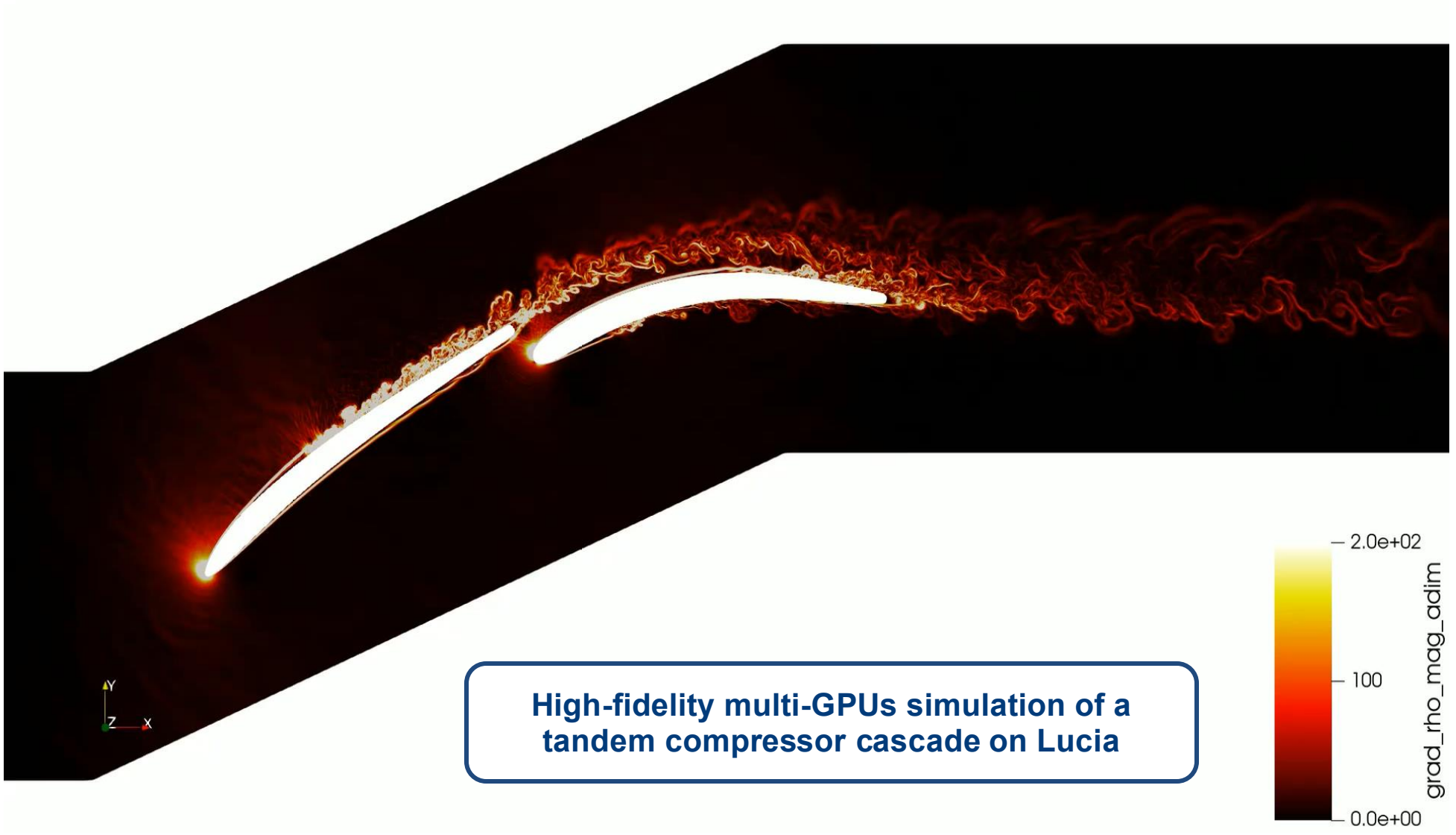
x 2.5

1 CPU node





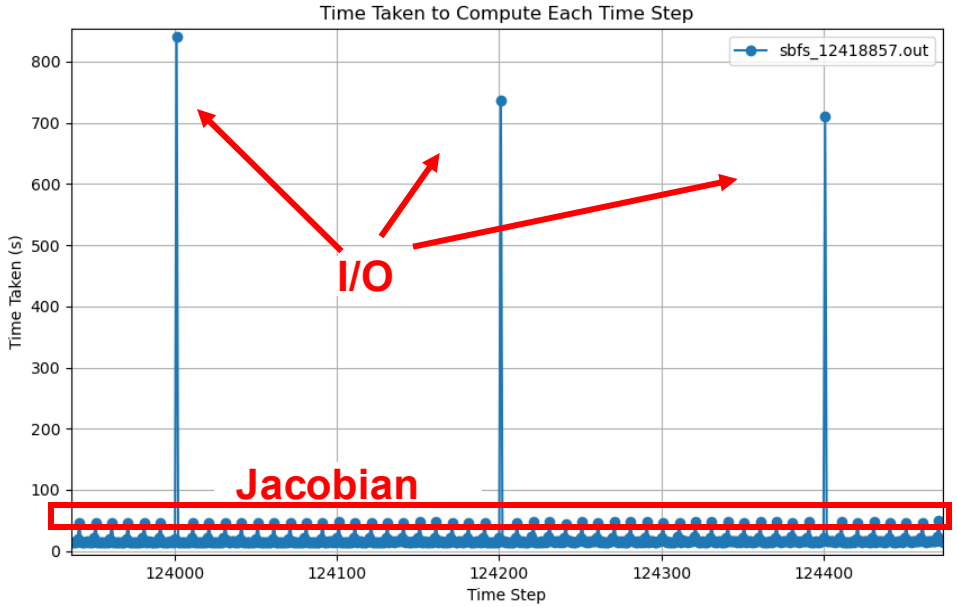
Fidelio
1st industrial application



High-fidelity multi-GPUs simulation of a tandem compressor cascade on Lucia

- Profile your code to identify bottlenecks
  - Wall-clock timers
    - Separate computation and communication
    - Use min, mean and max across ranks
  - Nsight, Rockprofiler, VTune, ...

Coarse grain - implicit solver



Fine grain - explicit solver

	#ranks	avg [s]	min [s]	max [s]	#rank_min	#rank_max
Initialization (before barrier)	128	3.744E+02	3.474E+02	4.333E+02	89	68
Initialization (after barrier)	128	4.333E+02	4.333E+02	4.333E+02	93	48
Iteration loop (before barrier)	128	1.113E+01	1.104E+01	1.127E+01	44	119
Iteration loop (after barrier)	128	1.127E+01	1.127E+01	1.127E+01	116	74
Non-overlapped comm	128	2.689E-01 (2.4%)	1.453E-02	7.071E-01	/	/
Volume interpolation	128	1.214E+00 (10.9%)	1.156E+00	1.255E+00	/	/
Volume cons. law	128	1.002E+00 (9.0%)	9.710E-01	1.027E+00	/	/
Volume projection	128	7.648E-01 (6.9%)	7.349E-01	7.858E-01	/	/
Volume source	128	0.000E+00 (0.0%)	0.000E+00	0.000E+00	/	/
Solution gradients	128	9.452E-01 (8.5%)	8.998E-01	1.008E+00	/	/
Face interpolation	128	3.457E+00 (31.1%)	3.340E+00	3.585E+00	/	/
Face cons. law	128	7.358E-01 (6.6%)	6.504E-01	8.293E-01	/	/
Face projection	128	2.657E-01 (2.4%)	2.541E-01	2.753E-01	/	/
Face to volume	128	1.128E+00 (10.1%)	1.034E+00	1.224E+00	/	/
Mass	128	1.097E+00 (9.9%)	1.061E+00	1.126E+00	/	/
Sol. update	128	1.816E-01 (1.6%)	1.761E-01	1.880E-01	/	/
Modal filtering	128	0.000E+00 (0.0%)	0.000E+00	0.000E+00	/	/
Not measured	128	7.171E-02 (0.6%)	5.607E-02	8.747E-02	/	/
Total time	128	1.113E+01 (100.0%)	1.104E+01	1.127E+01	/	/

Training opportunities

- **EuroCC Belgium** https://www.enccb.be/training#EuroCC
- **EuroCC** https://enccs.se/events/
- **CECI** https://www.ceci-hpc.be/training.html
- **VSC** https://www.vscentrum.be/vsctraining
- **LUMI** https://lumi-supercomputer.eu/events/
- **JSC** https://www.fz-juelich.de/en/jsc/education/training-courses
- **HLRS** https://www.hlrs.de/training/hpc-training
- **CINECA** https://www.hpc.cineca.it/training/
- **POP** (center of excellence in HPC) https://www.pop-coe.eu/
- **NVidia, AMD, Intel, ...**
- ...

Conclusions

- **Tier-0 systems deliver massive compute capability**
- **Performance is achieved only if the code and its workflow are scalable**
- **Moving from Tier-2 → Tier-1 → Tier-0 is not a simple scale-up**
  - I/O becomes a first-order bottleneck
  - Communication and load imbalance must be optimised
  - GPUs must be kept fed with data
  - Data movement must be carefully controlled
- **Successful tiering-up requires**
  - Scalable algorithms (compute, communication, I/O)
  - Early performance analysis and profiling
  - Appropriate programming models (MPI + accelerators)
  - Portability (NVidia – AMD)

Acknowledgments

- The development of the multi-GPU solvers GmshDG, Murphy and Fidelio has been supported by the European Regional Development Fund (ERDF/FEDER) and the Walloon Region of Belgium through respectively projects 1149 VirtualLab_ULiege, 1114 VirtualLab_UCL and 925 VirtualLab_Cenaero (programme 2021-2027), with additional complementary funding contributions.

- These developments benefited from computational resources made available on the Tier-1 supercomputer Lucia of the Fédération Wallonie-Bruxelles, infrastructure funded by the Walloon Region under the grant agreement n°1117545.

- We acknowledge EuroCC Belgium for awarding this project access to the LUMI supercomputer, owned by the EuroHPC Joint Undertaking, hosted by CSC (Finland) and the LUMI consortium through EuroCC Belgium.

Thank you for your attention