# Selected performance assessments using Scalasca/Score-P/CUBE

Brian Wylie (Jülich Supercomputing Centre)

HORIZON-EUROHPC-JU-2023-COE

EuroHPC Joint Undertaking

1 January 2024– 31 December 2026

Grant Agreement No 101143931

# Scalasca

DOI 10.5281/zenodo.7440854

- Collection of trace-based performance tools
  - Specifically designed for large-scale systems
  - Features automatic trace analyzer providing wait-state, critical-path & delay analysis
  - Supports MPI, OpenMP, POSIX threads, and hybrid MPI+OpenMP/Pthreads
  - Uses Score-P instrumentation & measurement infrastructure and CUBE analysis report infrastructure
- Available under 3-clause BSD open-source license

- Documentation & sources:
  - https://www.scalasca.org
- Contact:
  - mailto: scalasca@fz-juelich.de

trace tools scalasca



EuroHPC Joint Undertaking

# Score-P

DOI 10.5281/zenodo.10822140

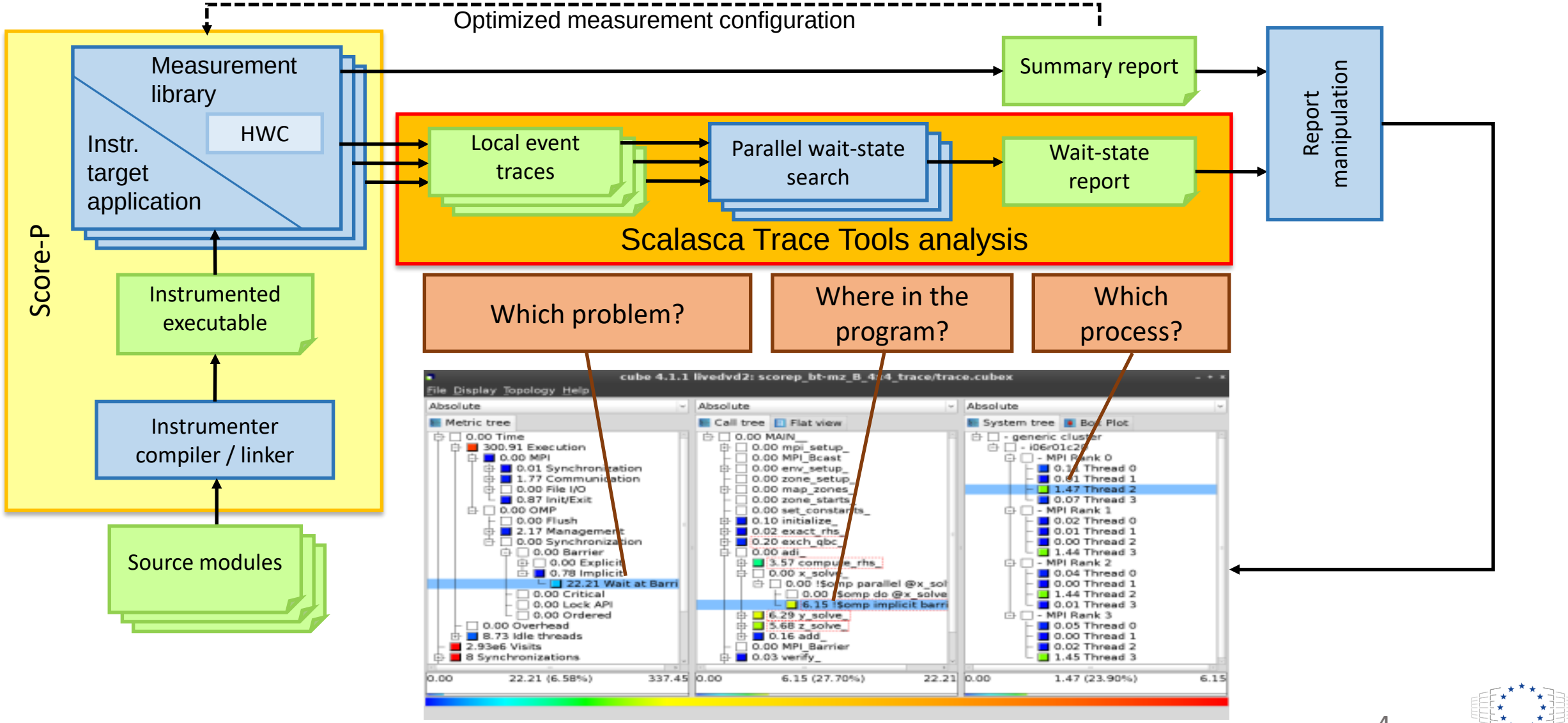
- Infrastructure for instrumentation and performance measurements
- Instrumented application can be used to produce several results:
  - Call-path profiling: CUBE4 data format used for data exchange
  - Event-based tracing: OTF2 data format used for data exchange

- Supported parallel paradigms:
  - Multi-process: MPI, SHMEM
  - Thread-parallel: OpenMP, Pthreads
  - Accelerator-based: CUDA, HIP, OpenCL, OpenACC, Kokkos

- Open Source; portable and scalable to all major HPC systems
- Initial project funded by BMBF
- Further developed in multiple third-party funded projects

- Documentation & sources: https://www.score-p.org

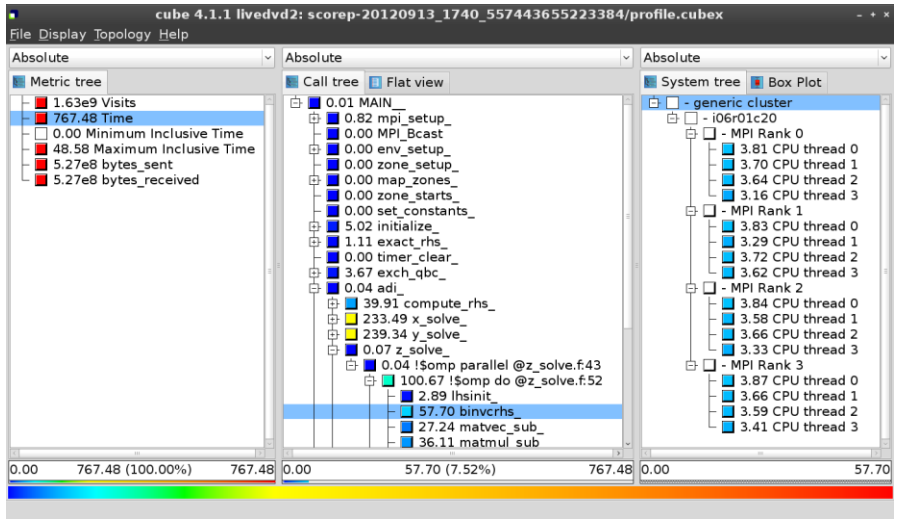
# Scalasca workflow



# CUBE

DOI 10.5281/zenodo.8345207

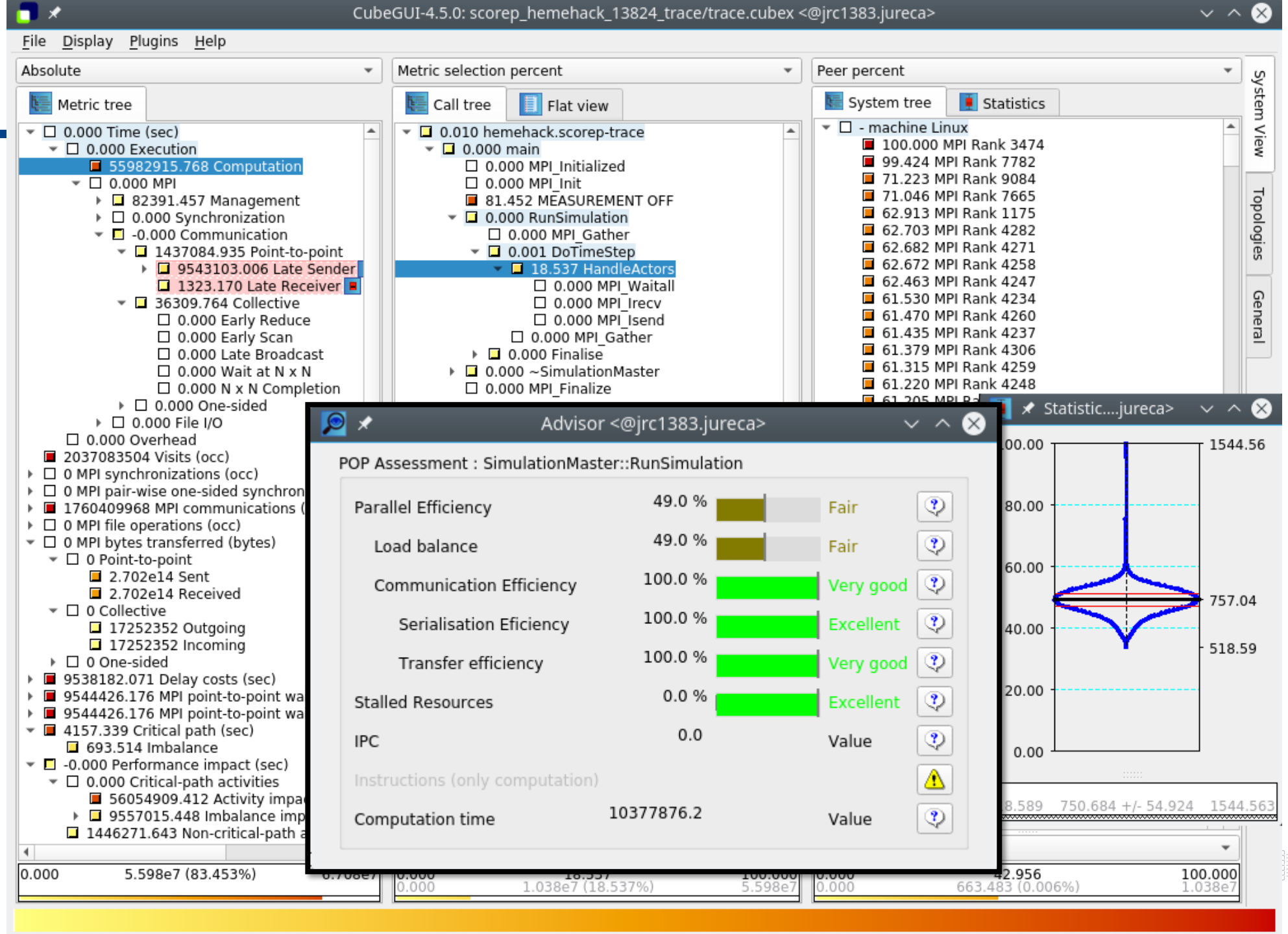
- Parallel program analysis report exploration tools
  - Libraries for Cube report reading & writing
  - Algebra utilities for report processing
  - GUI for interactive analysis exploration
- Available under 3-clause BSD open-source license
- Documentation & sources:
  - http://www.scalasca.org
- User guide also part of installation:
  - <prefix>/share/doc/CubeGuide.pdf
- Contact:
  - mailto: scalasca@fz-juelich.de



cube scalasca

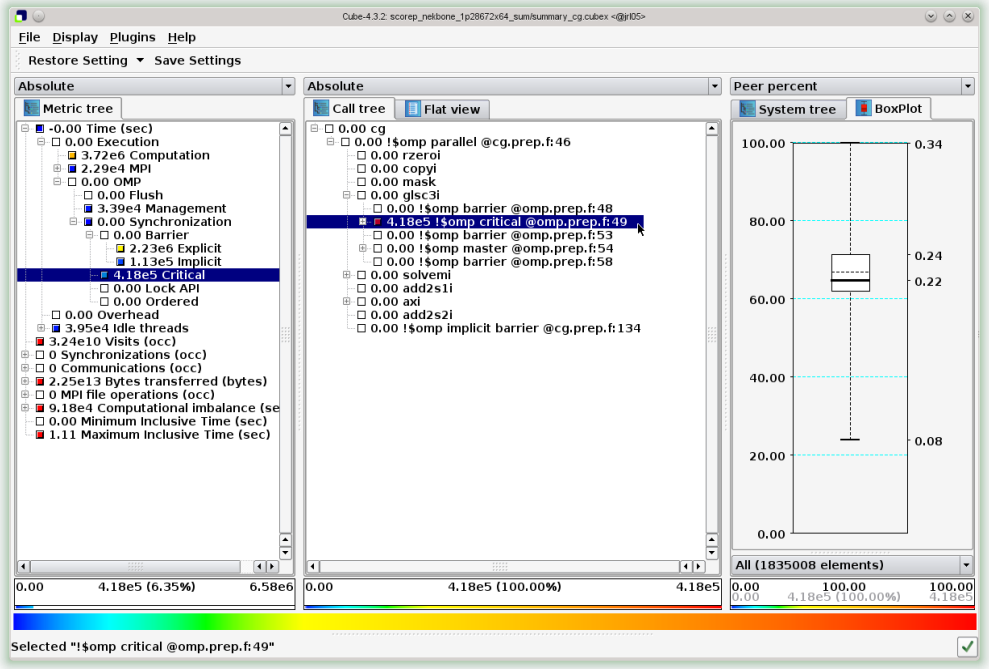
## Advisor assessment

Assessment of execution efficiency factors using POP model



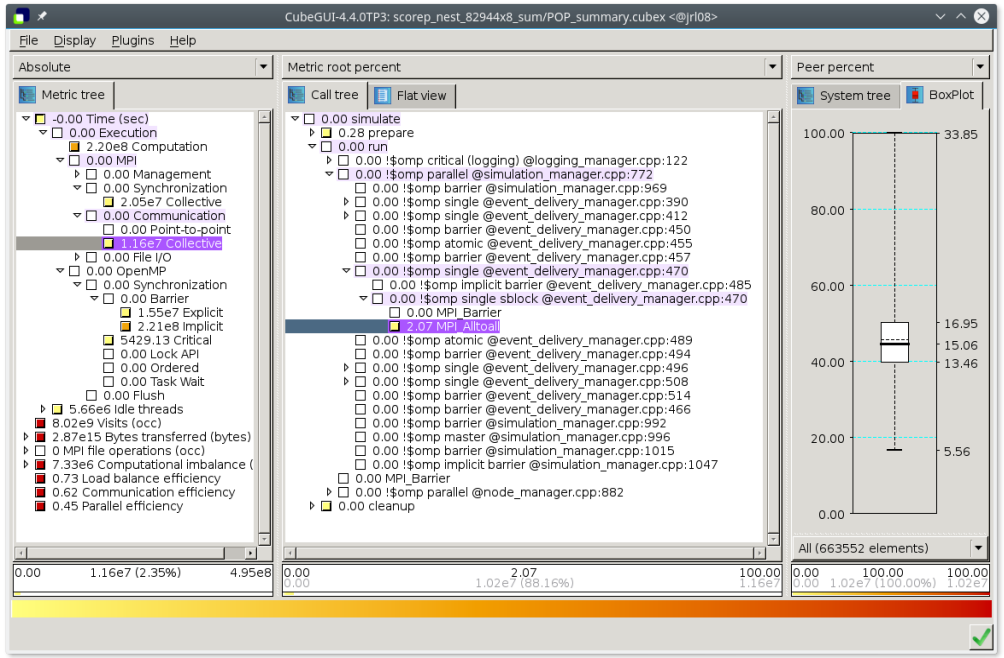
# Scalasca exa-scale readiness

- Largest experiment
  - Application: NekBone
  - System: *JUQUEEN* IBM BG/Q
  - 28,672 x 64 = 1,835,008 threads




https://co-design.pop-coe.eu/reports/POP-AR-112-Nekbone.html

- Largest experiment by user
  - Application: NEST
  - System: *K* computer
  - 82,944 x 8 = 663,552 threads




https://co-design.pop-coe.eu/reports/POP-AR-111-Nest_5g.html



EuroHPC Joint Undertaking

# Example performance assessments


- HemeLB (MPI) on *SuperMUC-NG*
  - also previously assessed on *ARCHER* Cray XC30 & *Blue Waters* Cray XE6
  - accelerated prototype (MPI+CUDA) assessed on *JUWELS-Volta* V100 GPUs

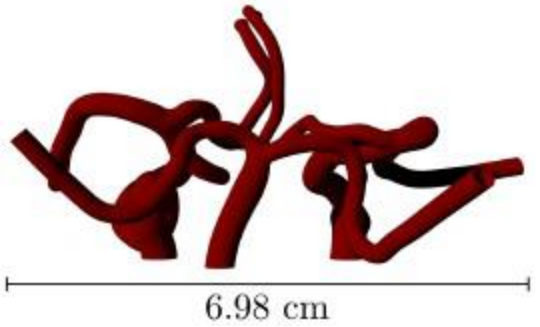
- SPECFEM3D (MPI+CUDA) on *Leonardo-B*
  - MPI version previously assessed on *Joloit-Curie*

# HemeLB (SuperMUC-NG)

https://co-design.pop-coe.eu/reports/POP2-AR-041-HemeLB.html

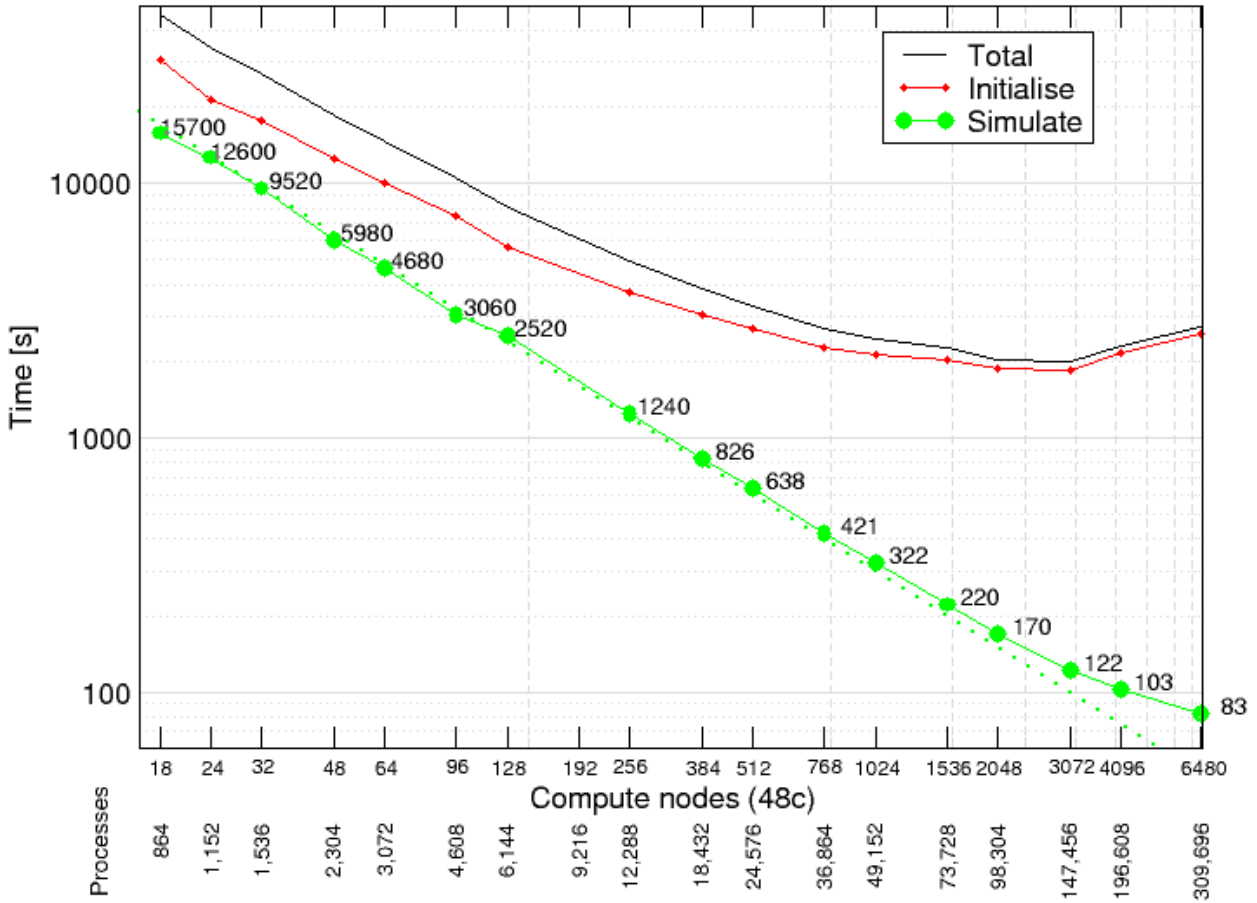
DOI 10.5281/zenodo.4105743
DOI 10.5281/zenodo.4275232

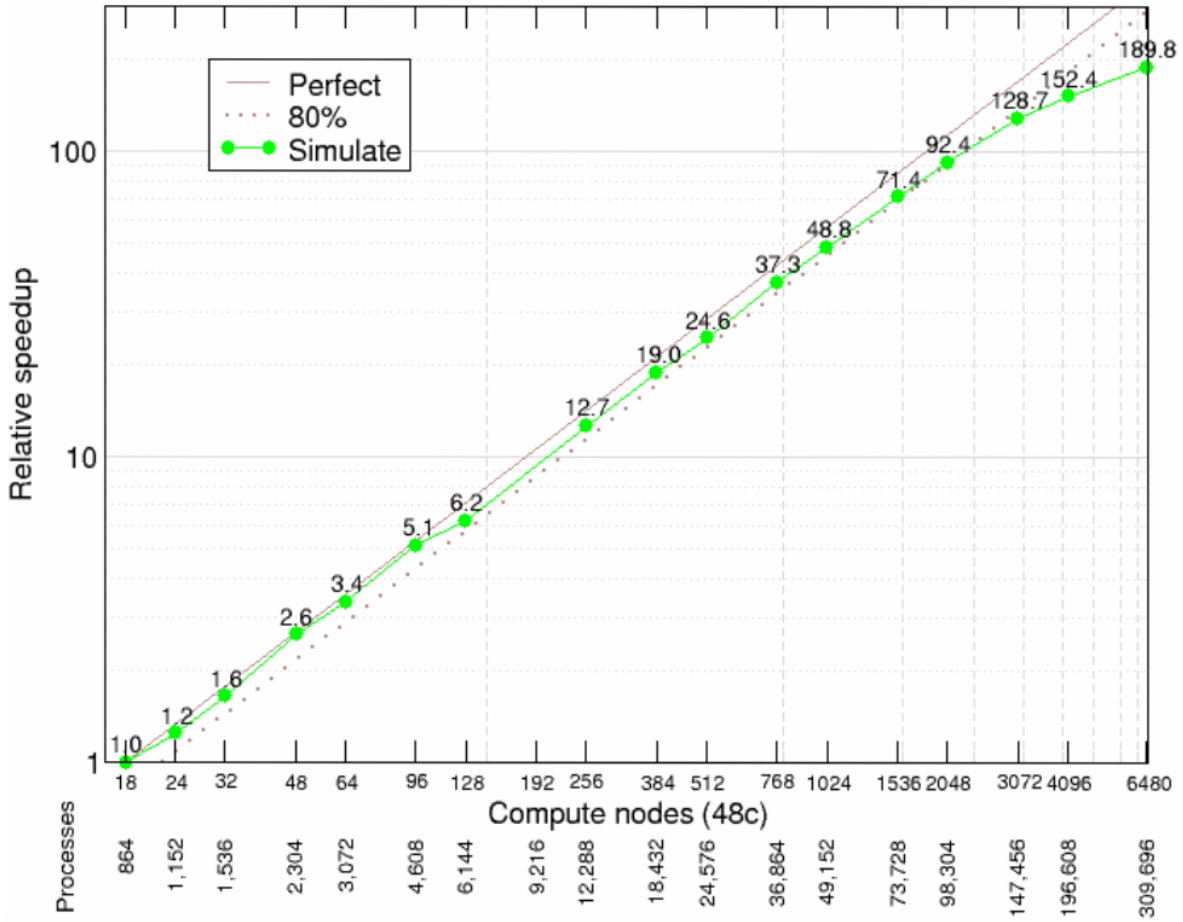
- 3D macroscopic blood flow in human arterial system developed by UC London (UK)
  - lattice-Boltzmann method tracking fluid particles on a lattice grid with complex boundary conditions
  - exascale flagship application of EU H2020 HPC Centre of Excellence for Computational Biomedicine (CompBioMed)
- HemeLB open-source code and test case: www.hemelb.org
  - C++ parallelized with MPI
    - Intel Studio 2019u4 compiler and MPI library (v19.0.4.243)
    - configured with 2 ‘reader’ processes (intermediate MPI file writing disabled)
    - MPI-3 shared-memory model employed within compute nodes to reduce memory requirements when distributing lattice blocks from reader processes
  - Focus of analysis 5,000 time-step (500µs) simulation of cerebrovascular “circle of Willis” geometry
    - 6.4µm lattice resolution (21.15 GiB): 10,154,448,502 lattice sites
- Executed on *SuperMUC-NG* Lenovo ThinkSystem SD650 (LRZ):
  - 2x 24-core Intel Xeon Platinum 8174 (‘Skylake’) @ 3.1GHz
  - 48 MPI processes/node, 6452 (of 6480) compute nodes: 309,696 MPI processes
  - 190x speed-up from 864 cores: 80% scaling efficiency to over 100,000 cores




***⇒ Identification & quantification of impact of load balance and its variation***

# HemeLB@SNG strong scaling




[Execution of 9,216 processes on 192 compute nodes not possible due to insufficient compute nodes with adequate memory in 'fat' partition (768 GiB vs. regular 96 GiB node memory]

# HemeLB@SNG strong scaling efficiency

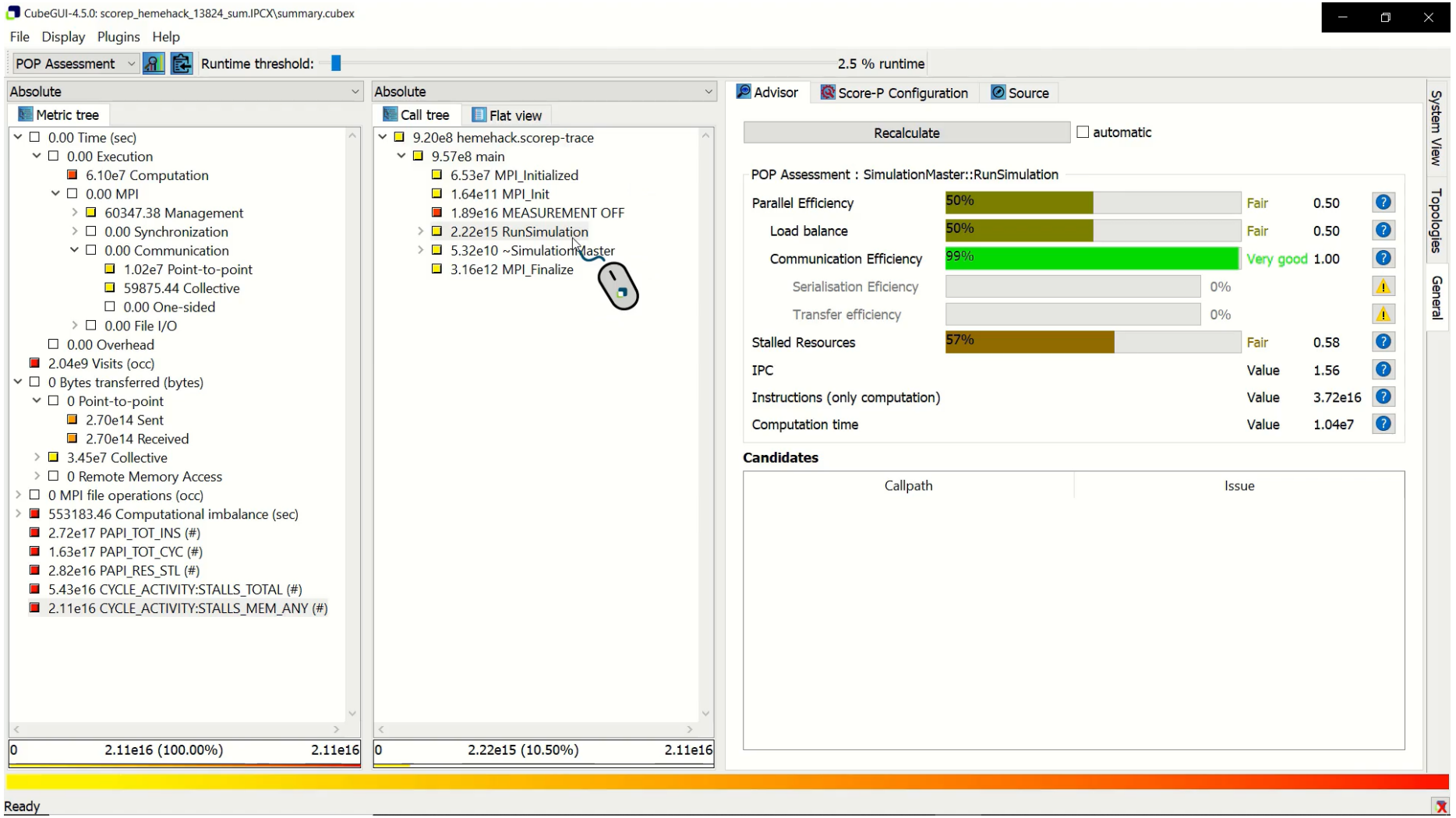
| Compute nodes | 24 | 32 | 48 | 64 | 96 | 128 | 192 | 256 | 384 | 512 | 768 | 1024 | 1536 | 2048 | 3072 | 4096 | 6452 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Processes | **1152** | **1536** | **2304** | **3072** | **4608** | **6144** | **9216** | **12288** | **18432** | **24576** | **36864** | **49152** | **73728** | **98304** | **147456** | **196608** | **309696** |
| **Global scaling efficiency** | **0.79** | **0.79** | **0.84** | **0.80** | **0.82** | **0.75** | | **0.73** | **0.72** | **0.73** | **0.74** | **0.68** | **0.68** | **0.65** | **0.62** | **0.57** | **0.45** |
| - Parallel efficiency | 0.79 | 0.80 | 0.87 | 0.83 | 0.86 | 0.80 | | 0.75 | 0.74 | 0.74 | 0.77 | 0.71 | 0.72 | 0.70 | 0.72 | 0.70 | 0.73 |
| - - Load balance efficiency | 0.79 | 0.80 | 0.88 | 0.84 | 0.86 | 0.80 | | 0.75 | 0.74 | 0.75 | 0.78 | 0.72 | 0.74 | 0.72 | 0.74 | 0.73 | 0.80 |
| - - Communication efficiency | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | | 1.00 | 1.00 | 0.99 | 0.99 | 0.99 | 0.98 | 0.98 | 0.97 | 0.96 | 0.92 |
| - Computation scaling | *1.00* | 0.99 | 0.96 | 0.96 | 0.95 | 0.93 | | 0.98 | 0.98 | 0.98 | 0.96 | 0.96 | 0.94 | 0.93 | 0.87 | 0.81 | 0.61 |
| - - Instructions scaling | *1.00* | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | | 1.00 | 1.00 | 1.00 | 0.99 | 0.97 | 0.94 | 0.89 | 0.79 | 0.67 | 0.45 |
| - - IPC scaling | *1.00* | 0.99 | 0.96 | 0.96 | 0.95 | 0.93 | | 0.98 | 0.98 | 0.99 | 0.98 | 0.99 | 1.00 | 1.04 | 1.11 | 1.21 | 1.36 |
| IPC | 1.411 | 1.395 | 1.353 | 1.355 | 1.342 | 1.316 | | 1.377 | 1.387 | 1.396 | 1.383 | 1.390 | 1.417 | 1.473 | 1.566 | 1.704 | 1.919 |
| | | | | | | | | | | | Key: | <0.65 | <0.75 | <0.85 | <0.95 | <1.00 | >1.00 |

Global scaling efficiency fairly good around 80%, before degrading at larger scales

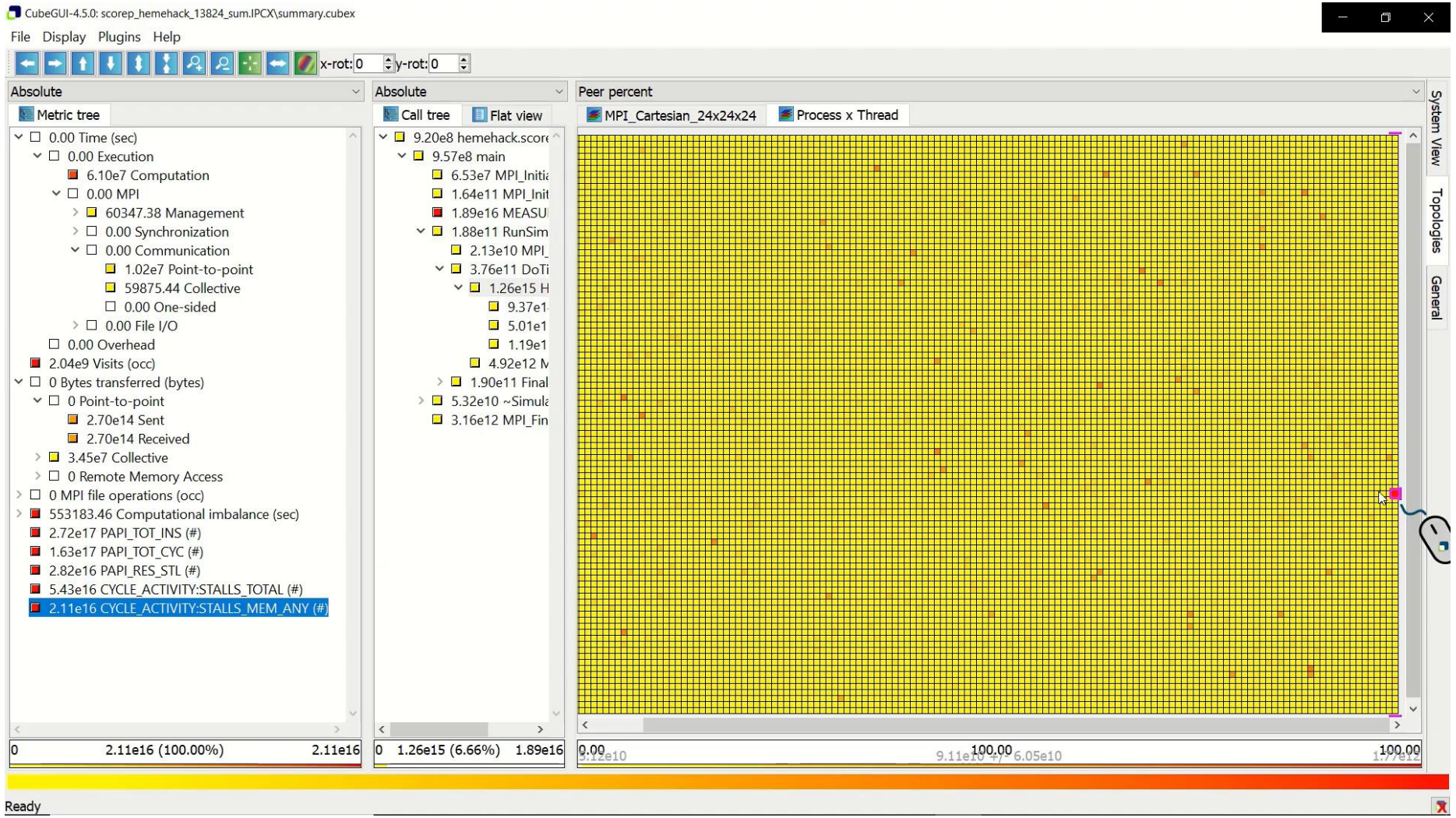
- Parallel efficiency deteriorating following Load balance efficiency
  - Communication efficiency excellent throughout
- Computation scaling (relative to 1152 processes) very good except at largest scale
  - Degradation of Instructions scaling partially compensated by improving IPC scaling

[POP CoE scaling efficiency model: www.pop-coe.eu]

# Advisor: POP efficiency assessment



# Topological presentation



# HemeLB_GPU (JUWELS-Volta)

https://co-design.pop-coe.eu/reports/POP2-AR-065-HemeLB_GPU.html

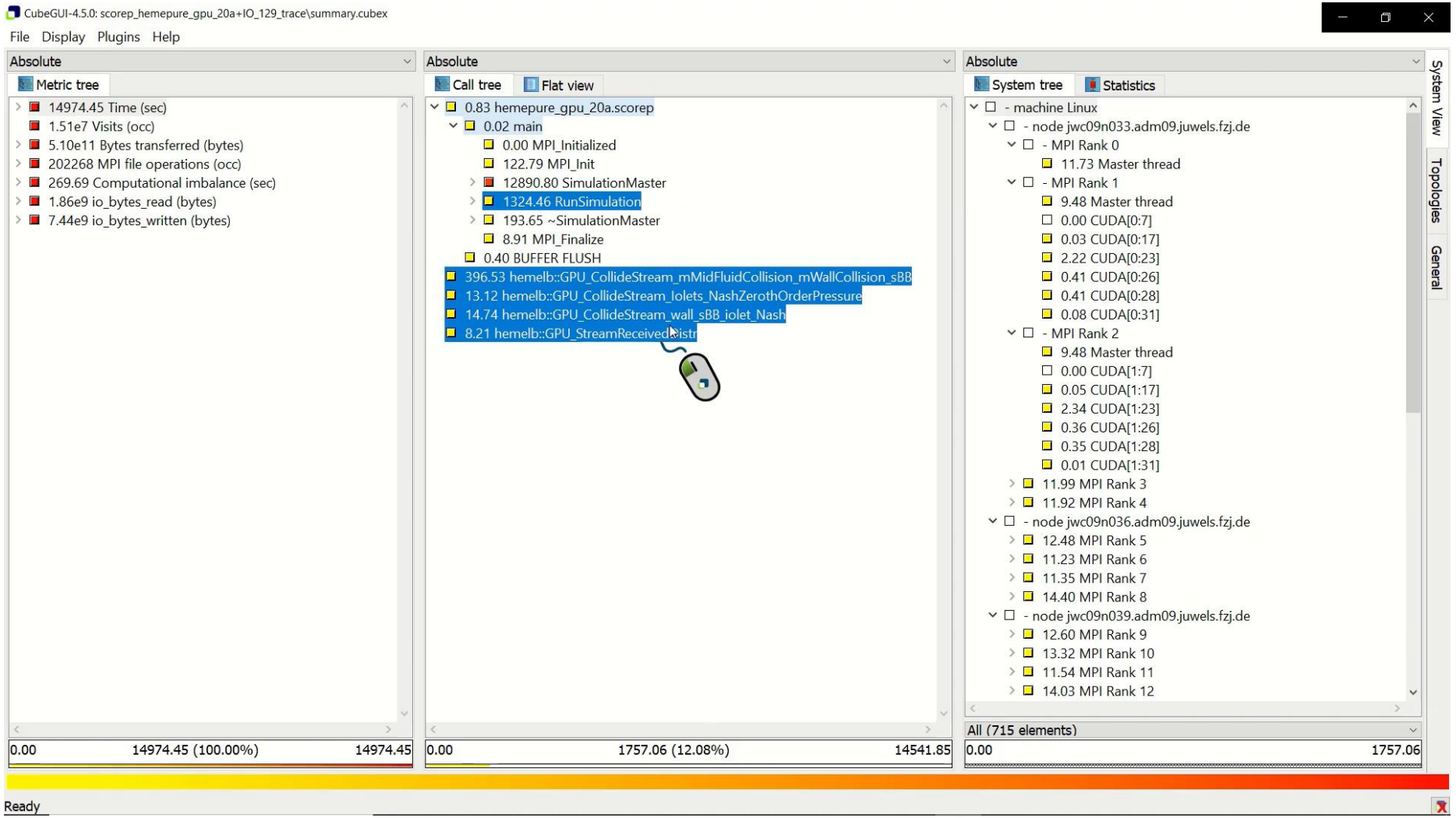
DOI 10.5281/zenodo.4117942
DOI 10.5281/zenodo.4081080

- 3D macroscopic blood flow in human arterial system developed by UC London (UK)
  - lattice-Boltzmann method tracking fluid particles on a lattice grid with complex boundary conditions
  - exascale flagship application of EU H2020 HPC Centre of Excellence for Computational Biomedicine
- HemeLB open-source code and test case: www.hemelb.org
  - C++ parallelized with MPI + CUDA (in development)
    - GCC/8.3.0 compiler, CUDA/10.1.105 and ParaStationMPI/5.4 library
    - configured with 2 ‘reader’ processes and intermediate MPI file writing
    - rank 0 ‘monitor’ process doesn’t participate in simulation

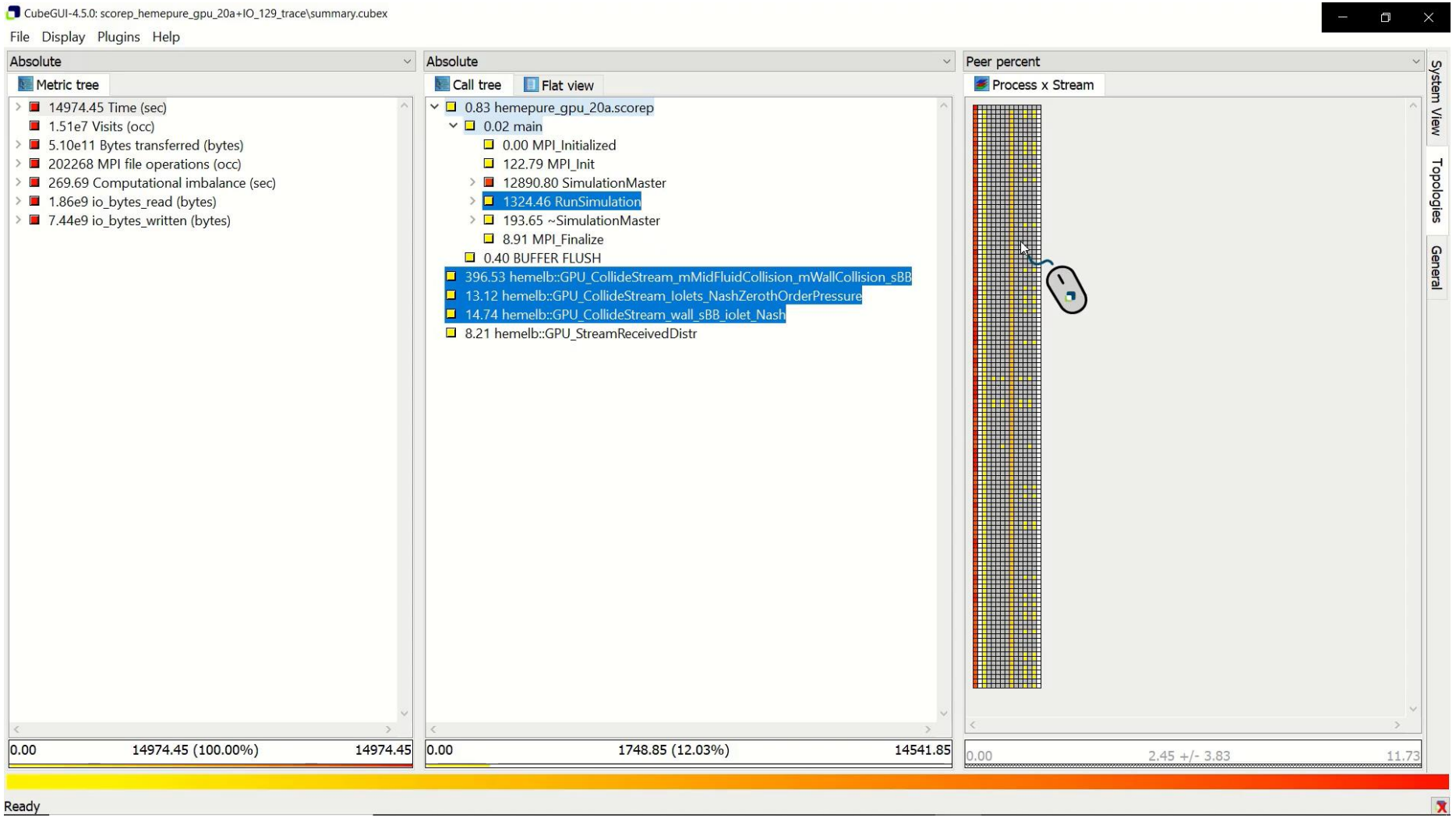
  - Focus of analysis 2,000 time-step (each 100µs) simulation of CBM2019_Arteries_patched geometry
    - 1.78 GiB: 66,401,494 lattice sites, 1+38 iolets
- Executed on *JUWELS-Volta* (@JSC):
  - 2x 20-core Intel Xeon Platinum 8168 (‘Skylake’) CPUs + 4 Nvidia V100 ‘Volta’ GPUs
  - 4* MPI processes/node (one per GPU), 32 (of 56) compute nodes: 129 MPI processes

***⇒ Identification & quantification of impact of CPU/GPU load balance and its variation***

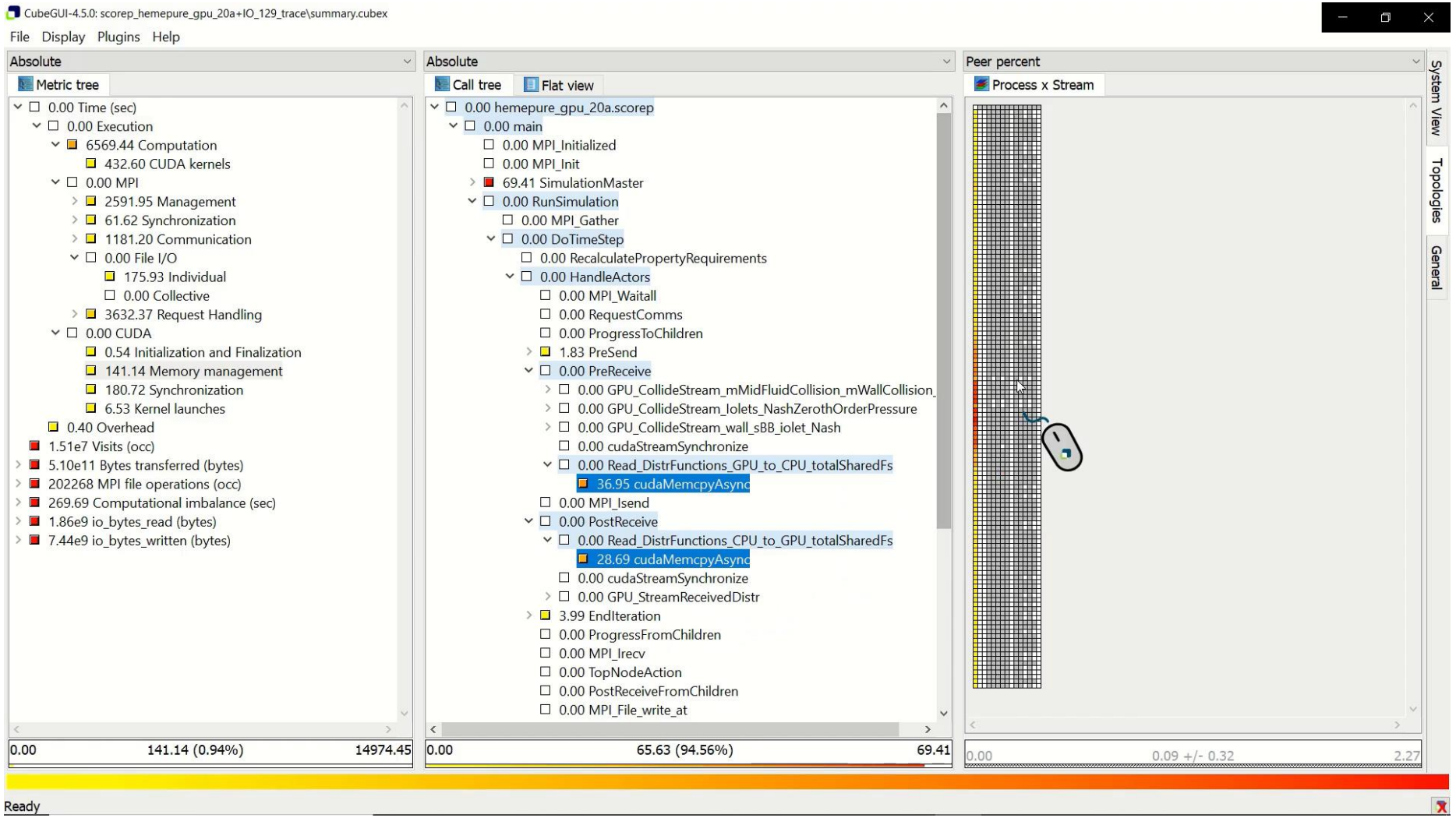
# Time for asynch. CUDA kernels



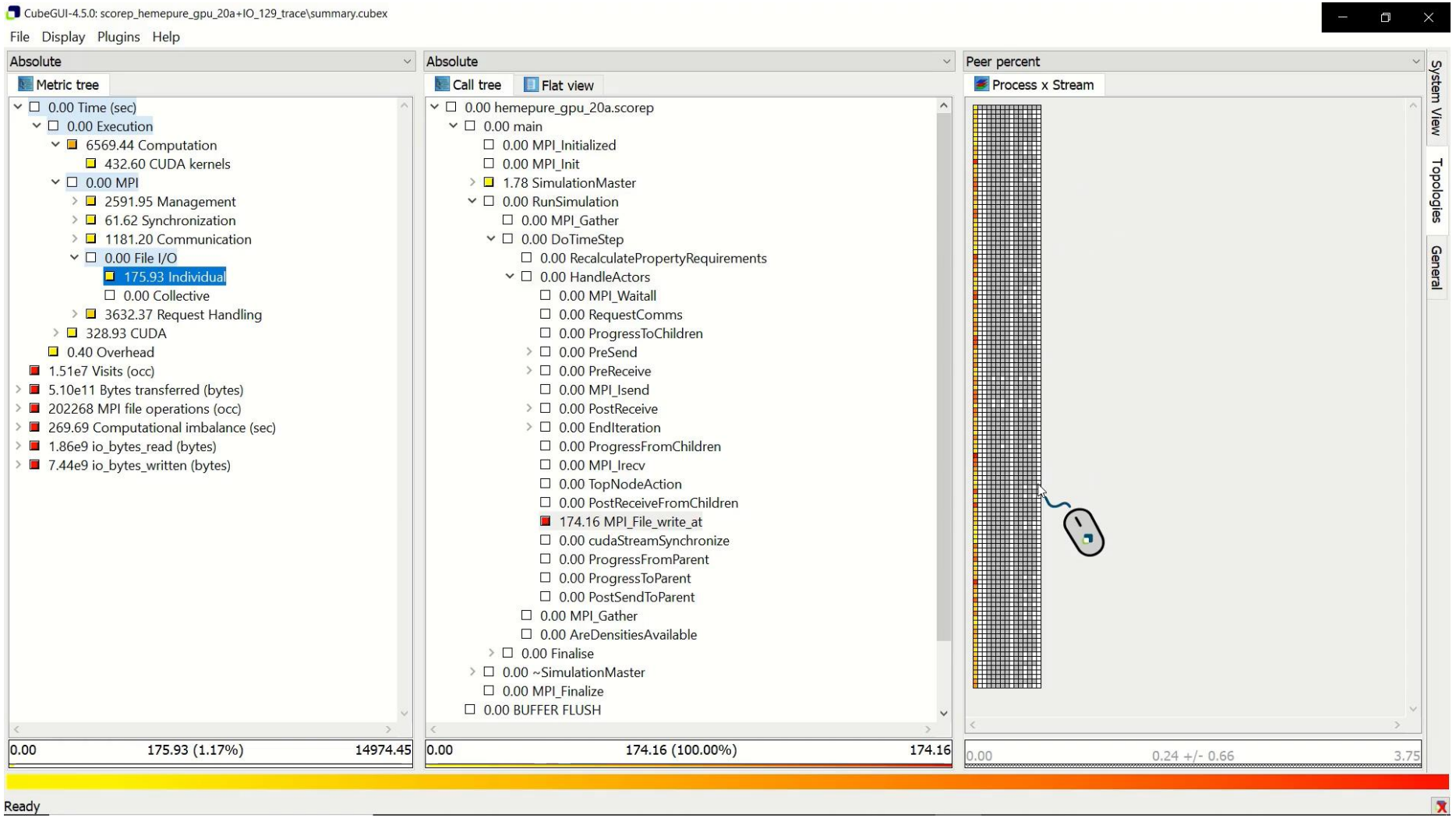
# Time for asynch. CUDA kernels



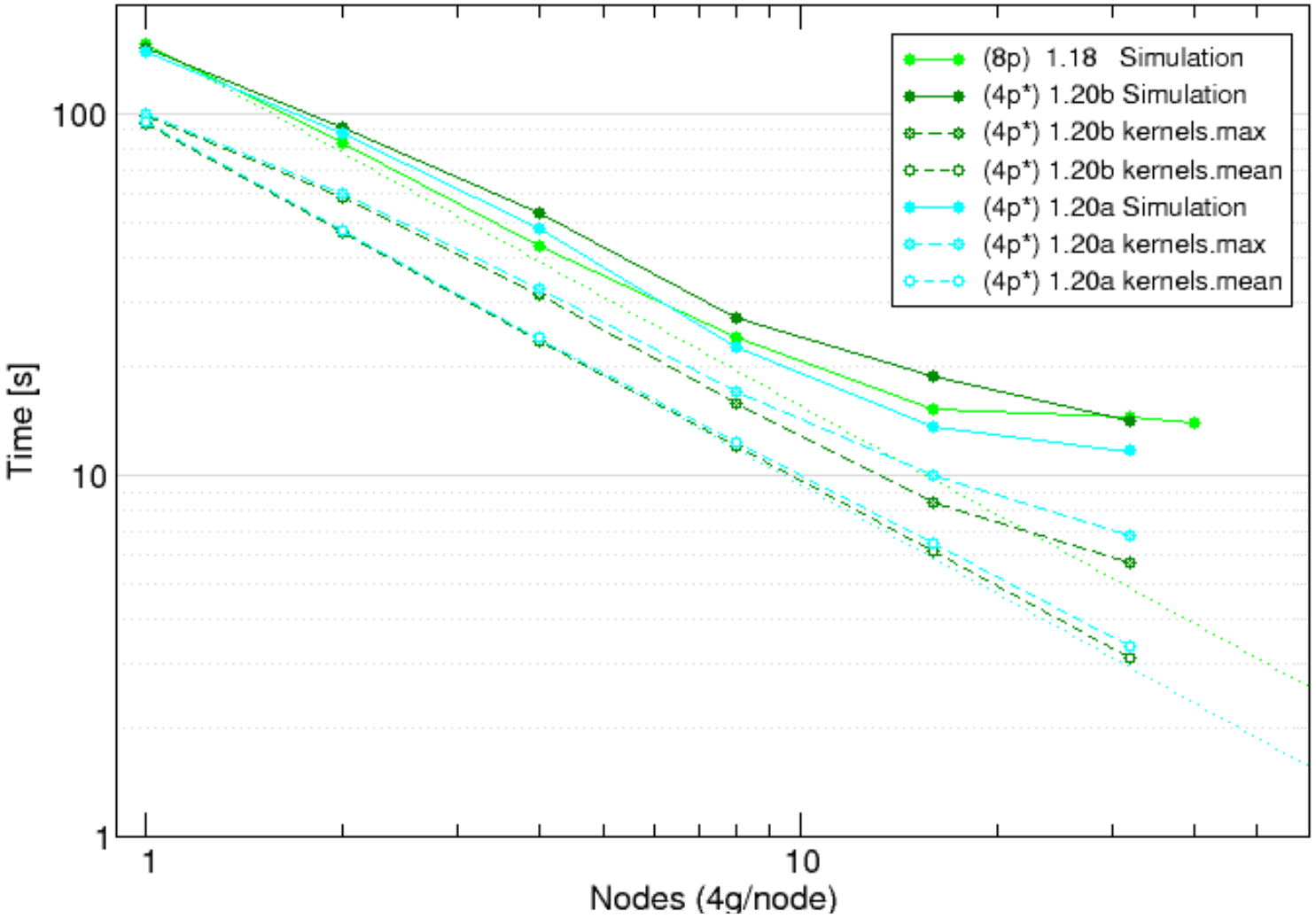
# Time for CUDA asynch. memory copies



# Time for MPI file writing on CPU



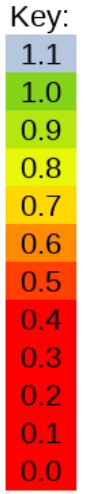
# HemeLB@JUWELS-Volta strong scaling




- Reference execution with 8ppn
  - multiple processes offloading GPU kernels generally unproductive
- Comparison of versions (4ppn)
  - v1.20a generally better
- Synchronous MPI file writing is the primary bottleneck
- CUDA kernels on GPUs
  - less than half of Simulation time (therefore GPUs mostly idle)
  - total kernel time scales very well (0.93 scaling efficiency)
  - load balance deteriorates (0.95 for single node, 0.50 for 32 nodes)

# HemeLB@JUWELS-V scaling efficiency

| | 1n 5p | 2n 9p | 4n 17p | 8n 33p | 16n 65p | 32n 129p |
|---|---|---|---|---|---|---|
| Simulation time [s] | 147.87 | 88.38 | 48.13 | 22.66 | 13.68 | 11.67 |
| Global scaling efficiency | 0.64 | 0.53 | 0.49 | 0.52 | 0.43 | 0.25 |
| – Parallel efficiency | 0.64 | 0.53 | 0.50 | 0.54 | 0.47 | 0.29 |
| – – Load balance efficiency (GPU) | 0.95 | 0.78 | 0.73 | 0.73 | 0.65 | 0.50 |
| – – Communication efficiency (GPU) | 0.67 | 0.68 | 0.68 | 0.75 | 0.73 | 0.58 |
| – Computation scaling (GPU) | 1.00 | 1.00 | 0.99 | 0.96 | 0.92 | 0.87 |

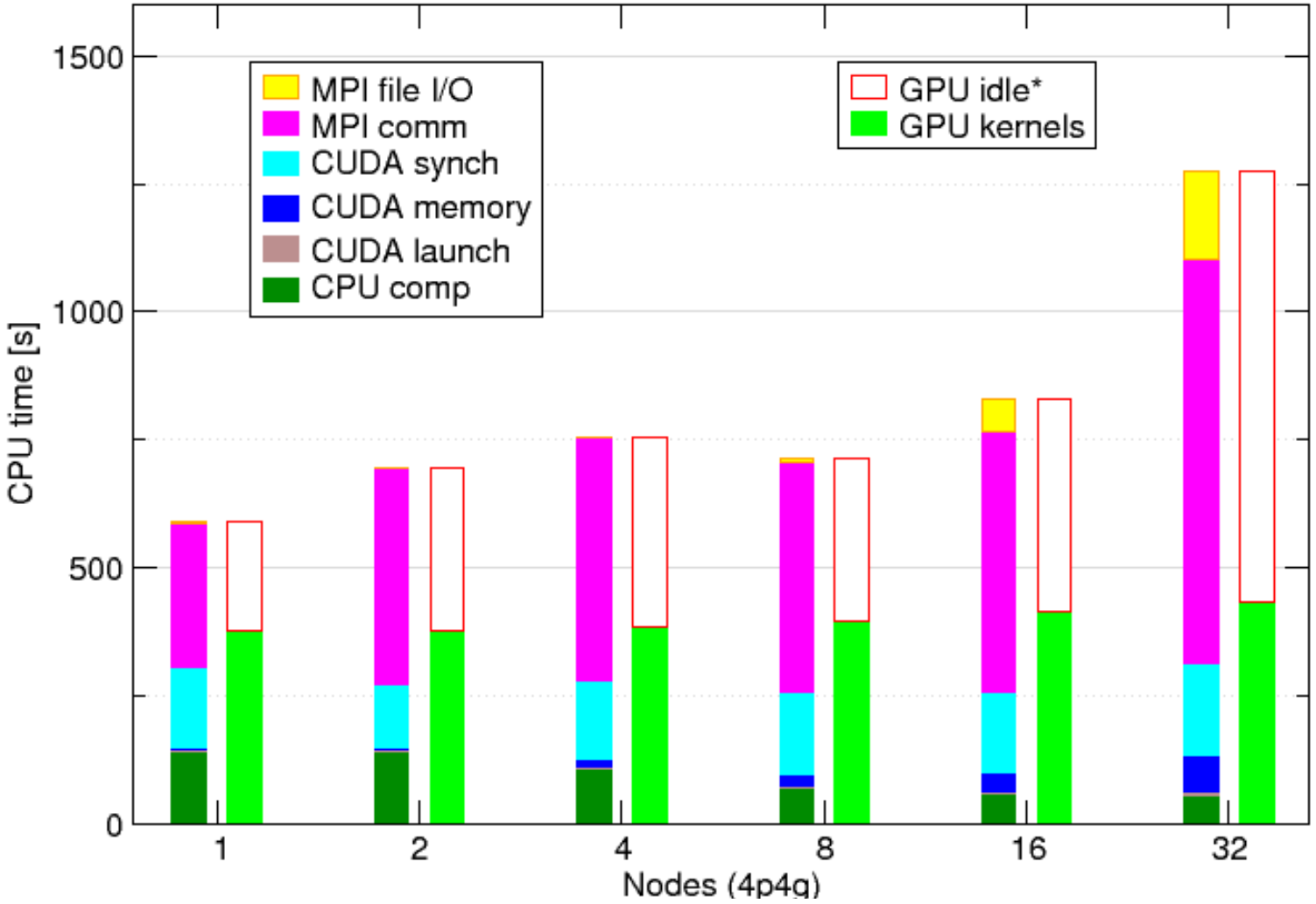



Only considering GPUs (ignoring all CPU cores, 90% of which are completely unused)

- Single (quad-GPU) node already suffers significant communication inefficiency
  - includes MPI file writing, but doesn't degrade much as additional nodes are included
- Load balance of GPUs deteriorates progressively
- GPU computation scaling remains reasonably good

[POP CoE scaling efficiency model: www.pop-coe.eu]

# HemeLB@JUWELS-V strong scaling




- CPU+GPU time breakdown
- CUDA kernels on GPUs
  - less than half of Simulation time (therefore GPUs mostly idle)
  - total kernel time scales very well (0.87 scaling efficiency)
- MPI processes on CPUs
  - computation time decreases
  - CUDA synchronization time fairly constant, but time for memory management increases somewhat
  - MPI communication time dominates, with much more time for file writing with 16+ nodes

# Example performance assessments


- HemeLB on *SuperMUC-NG* (MPI)
  - also previously assessed on *ARCHER* Cray XC30 & *Blue Waters* Cray XE6
  - accelerated prototype (MPI+CUDA) assessed on *JUWELS-Volta* V100 GPUs

- SPECFEM3D on *Leonardo-B* (MPI+CUDA)
  - also previously assessed on *Joloit-Curie*

# SPECFEM3D (Leonardo-B)

https://co-design.pop-coe.eu/reports/POP3-AR-002-SPECFEM3D.html

DOI 10.5281/zenodo.13643996

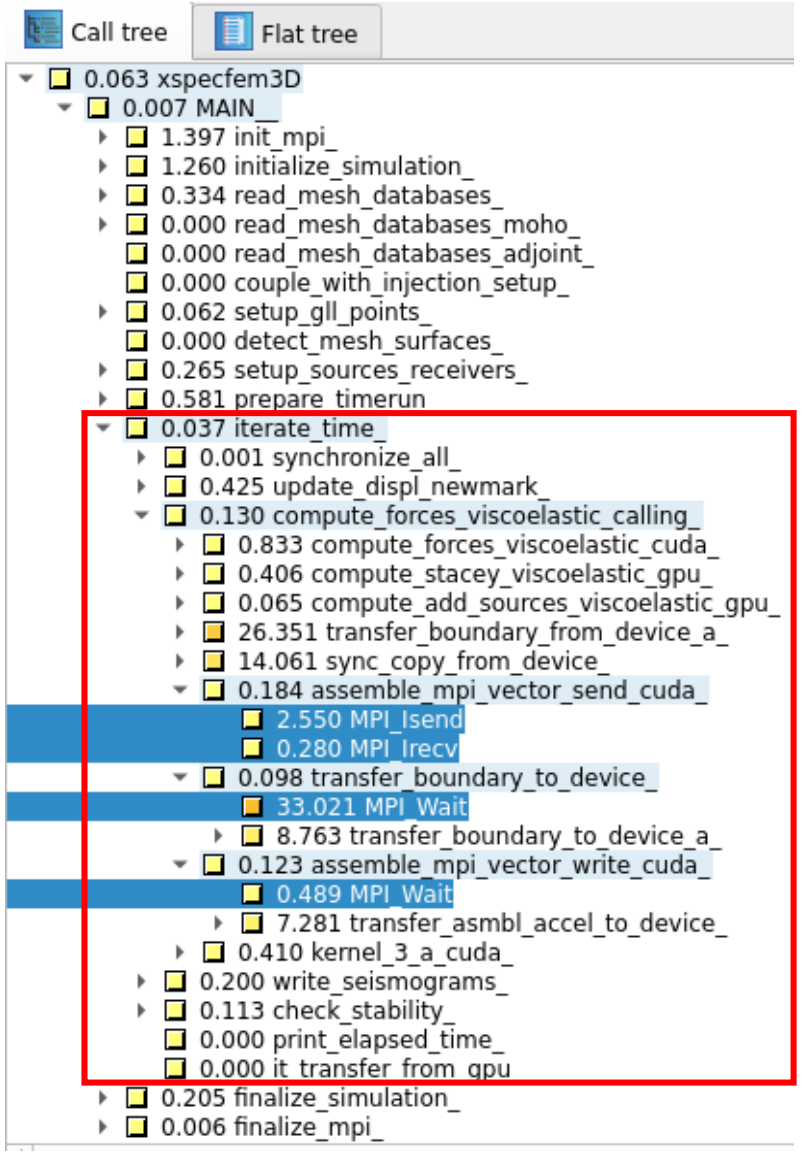
- SPECFEM3D
  - Software package for simulation of seismic wave propagation based on the spectral-element method
  - Assessment for HPC CoE for Exascale in Solid Earth (ChEESE)
  - Version 4.0.0 (release)
- Fortran90 (and some C) parallelized with MPI & CUDA: one MPI process per GPU
  - Intel oneAPI 2023.0.0 compilers and 2021.7.1 MPI libraries (not GPU-Aware)
- Testcase: 1 source in elastic domain; 4 seismic receiver stations
  - 48000 solver timesteps with intermediate writing disabled
  - weak scaling (22x 128x128 = 360,448 elastic elements per rank)
  - strong scaling (22x 1024x1024 = 23,068,672 elastic elements total)
- Executed on *Leonardo-Booster* Atos Bull Sequana XH21355 (CINECA)
  - 2345 compute nodes with 32-core Intel Xeon Platinum 8358 ('IceLake') CPUs @ 2.6 GHz and quad Nvidia A100 ('Ampere') GPUs [64GB]
  - Nvidia Mellanox HDR DragonFly++ interconnection network
- Measurements with Scalasca/2.6.1 using Score-P/8.3
- Focus of analysis (FOA): xspecfem3D/*iterate_time*

ChEESE



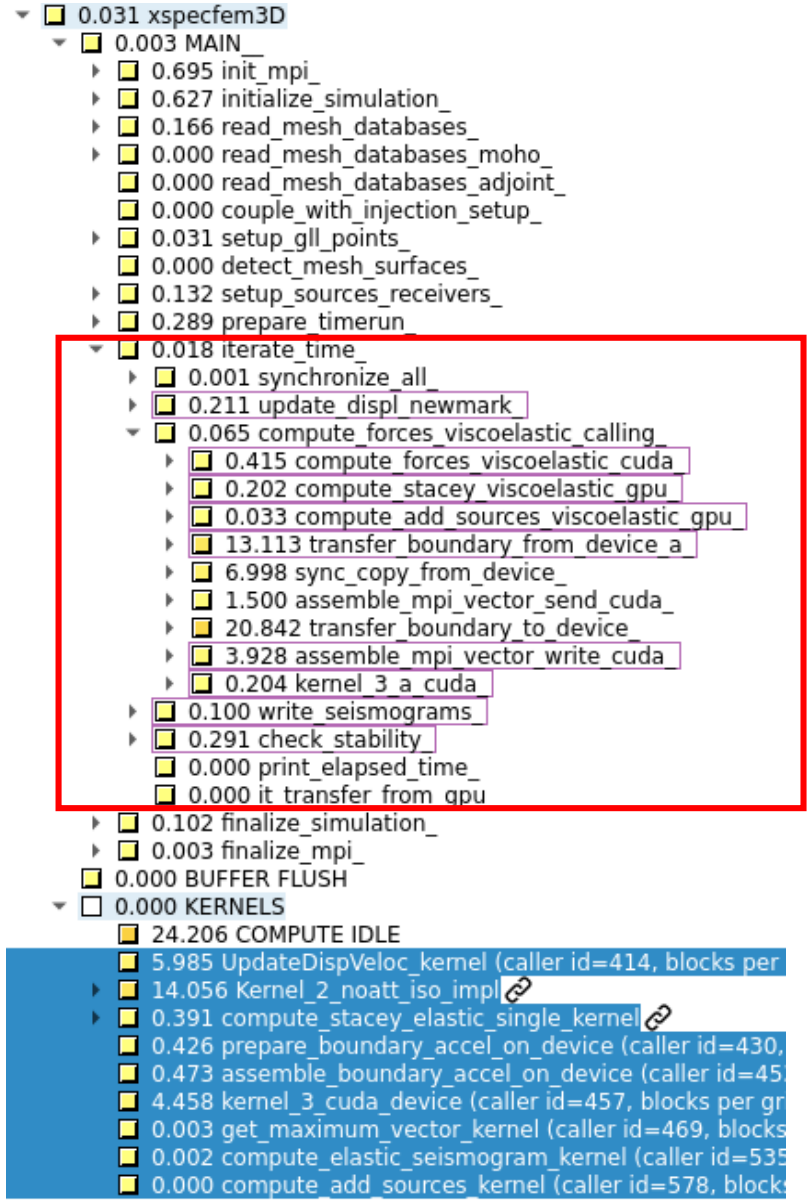
EuroHPC Joint Undertaking

# Execution call-tree & Focus of Analysis



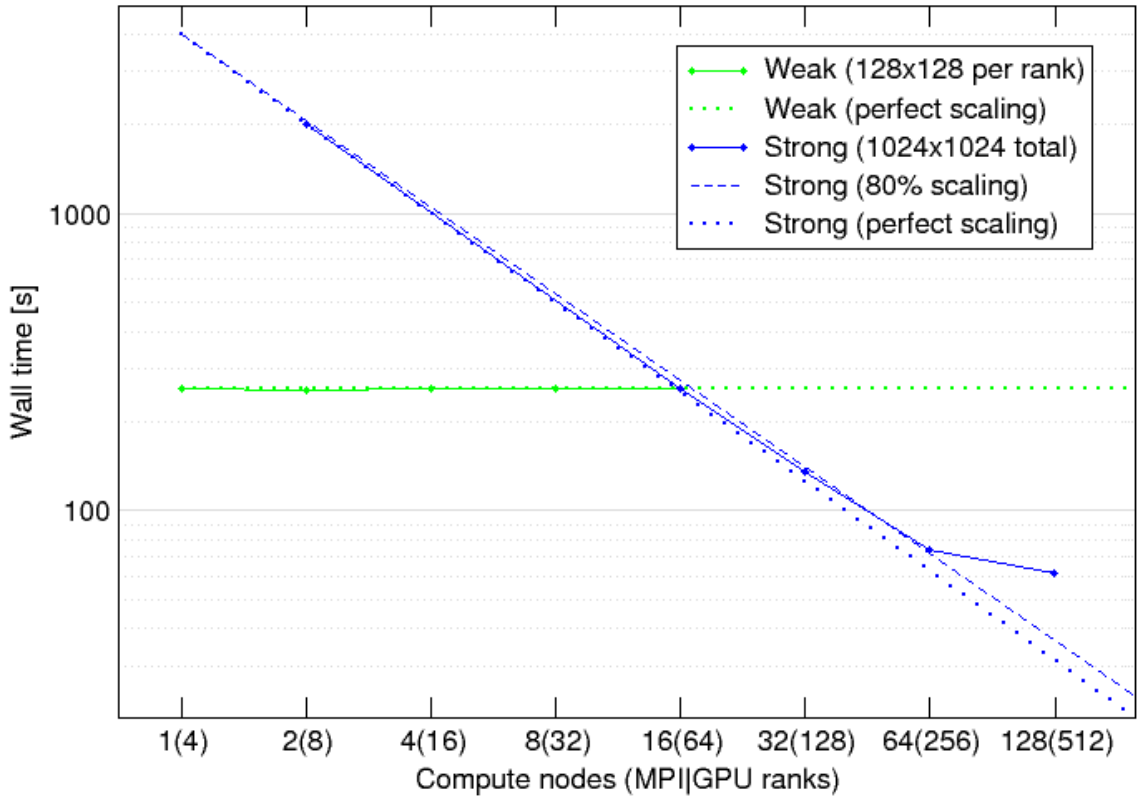

- Structure
  - setup/initialise (amortised in full run)
    - read (w/o MPI), MPI_Bcast, MPI_Reduce, etc
  - solver (*iterate_time*)
    - 48000 timesteps
      - non-blocking point-to-point communication for boundary exchange with 2D neighbours
      - data transfer to/from associated GPU device and corresponding stream synchronization
    - summary output every 500 steps
      - collective MPI_Reduce
  - *write_seismograms* executed only once
- Focus of Analysis selected for assessment: *iterate_time*

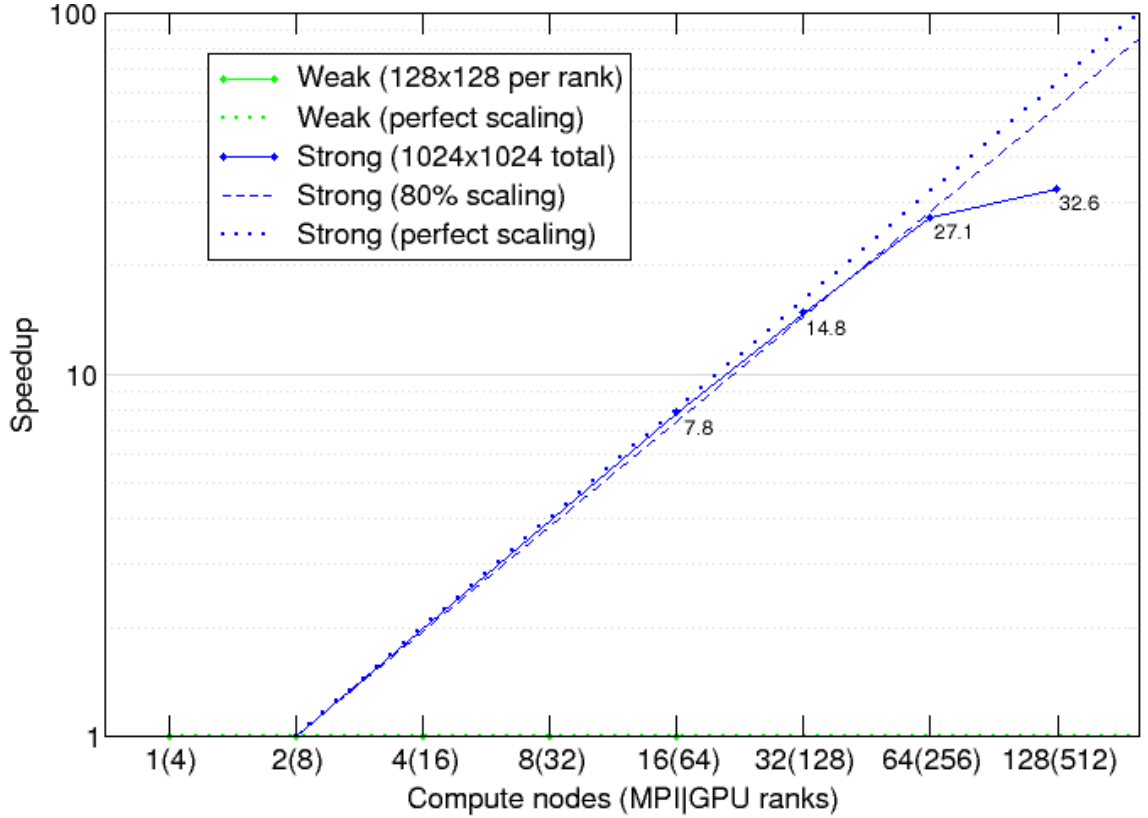
# Execution call-tree & kernels




- Structure
  - solver (*iterate_time*)
    - contains all of the CUDA kernel executions
      - GPU devices idle during init/finalize
    - 48000 timesteps
      - seven of nine kernels executed by all ranks
      - characteristics often determined by position in 2D grid
    - *compute_add_sources_kernel* only executed by a single GPU (rank 243 of 512)
    - *compute_elastic_seismogram* only by 4 nearby GPUs (ranks 241, 245, 273, 277 of 512)
- Focus of Analysis selected for assessment: *iterate_time*

# Scaling & speed-up




- xspecfem3d FOA *iterate_time* on Leonardo-Booster
- Excellent weak scaling (expected to continue to higher node counts)
- Very good strong scaling (above 80% of perfect) to around 64 compute nodes (256 GPUs)

# Efficiency model (weak scaling)

| | Problem size | 256x256 | 512x256 | 512x512 | 1024x512 | 1024x1024 |
|---|---|---|---|---|---|---|
| | MPI\|GPU ranks | **4** | **8** | **16** | **32** | **64** |
| | Wall time [s] | **255.873** | **255.462** | **256.035** | **255.898** | **255.948** |
| **XPU** | Global scaling efficiency | 0.514 | 0.516 | 0.514 | 0.514 | 0.514 |
| | - Computation time scaling | *1.000* | 0.992 | 0.977 | 0.972 | 0.963 |
| | - Parallel efficiency | 0.514 | 0.520 | 0.526 | 0.529 | 0.534 |
| | - - Load balance efficiency | 0.522 | 0.526 | 0.533 | 0.536 | 0.541 |
| | - - Orchestration efficiency | 0.986 | 0.988 | 0.988 | 0.988 | 0.988 |
| **GPU** | Global scaling efficiency | 0.986 | 0.987 | 0.985 | 0.986 | 0.986 |
| | - Computation time scaling | *1.000* | 1.000 | 1.000 | 1.000 | 1.000 |
| | - Parallel efficiency | 0.986 | 0.987 | 0.985 | 0.986 | 0.986 |
| | - - Load balance efficiency | 1.000 | 0.999 | 0.997 | 0.998 | 0.998 |
| | - - Orchestration efficiency | 0.986 | 0.988 | 0.988 | 0.988 | 0.988 |

- Orchestration efficiency
  - MPI communication & CUDA management
  - aka Communication efficiency

Key:
1.10
1.00
0.90
0.80
0.70
0.60
0.50
0.40
0.30
0.20
0.10
0.00

- Excellent GPU weak scaling efficiency
- Very poor CPU efficiency?
- Moderate XPU (GPU+CPU) efficiency?

# Efficiency model (strong scaling)

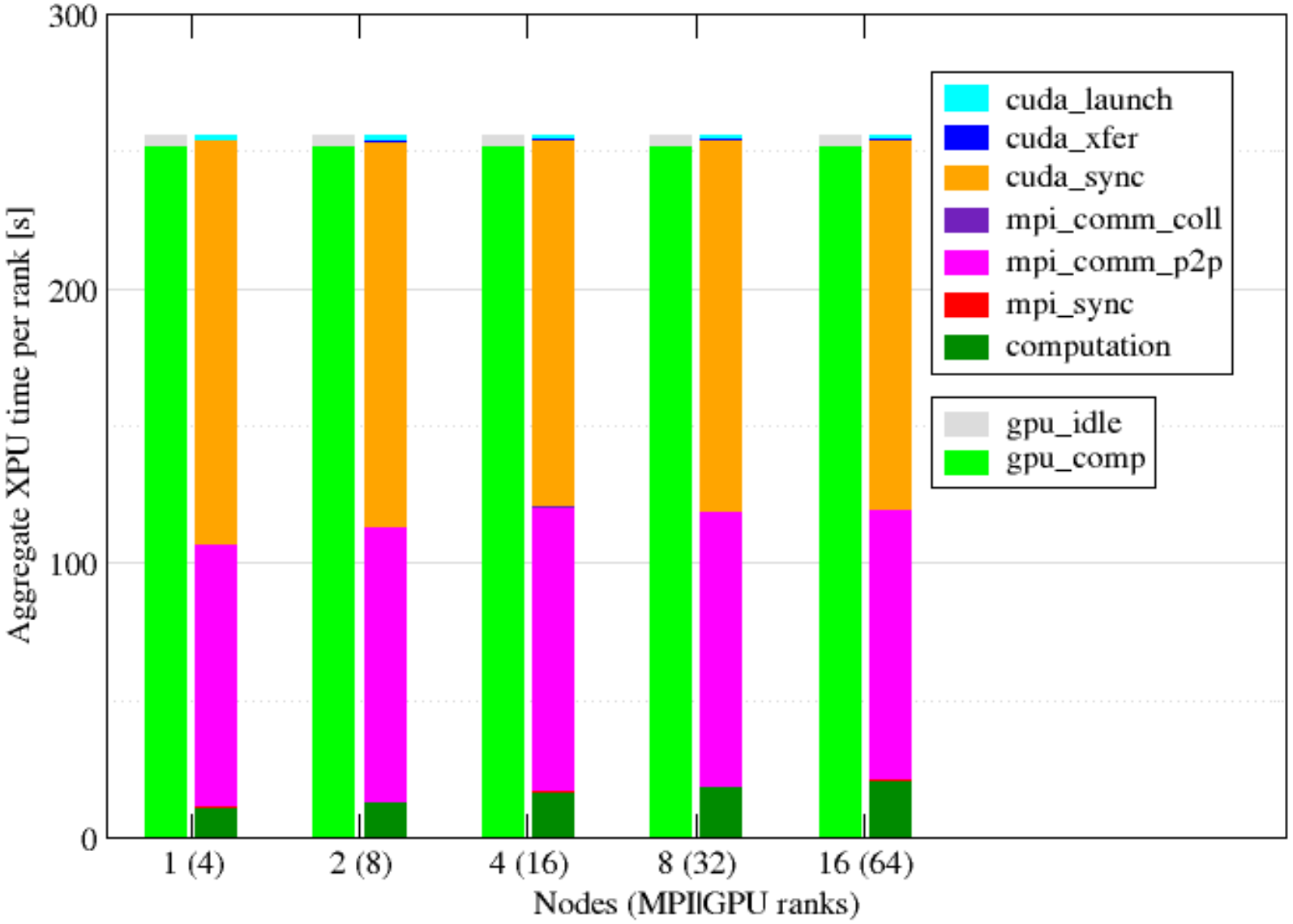
| | Problem size | 1024x1024 | 1024x1024 | 1024x1024 | 1024x1024 | 1024x1024 |
|---|---|---|---|---|---|---|
| | MPI\|GPU ranks | **8** | **64** | **128** | **256** | **512** |
| | Wall time [s] | **2001.806** | **255.948** | **135.478** | **73.846** | **61.400** |
| **XPU** | Global scaling efficiency | 0.515 | 0.504 | 0.476 | 0.437 | 0.263 |
| | - Computation time scaling | *1.000* | 0.943 | 0.861 | 0.835 | 0.715 |
| | - Parallel efficiency | 0.515 | 0.534 | 0.553 | 0.523 | 0.367 |
| | - - Load balance efficiency | 0.518 | 0.541 | 0.583 | 0.586 | 0.671 |
| | - - Orchestration efficiency | 0.995 | 0.988 | 0.948 | 0.892 | 0.547 |
| **GPU** | Global scaling efficiency | 0.995 | 0.973 | 0.919 | 0.843 | 0.507 |
| | - Computation time scaling | *1.000* | 0.987 | 0.972 | 0.950 | 0.937 |
| | - Parallel efficiency | 0.995 | 0.986 | 0.945 | 0.887 | 0.541 |
| | - - Load balance efficiency | 1.000 | 0.998 | 0.996 | 0.994 | 0.989 |
| | - - Orchestration efficiency | 0.995 | 0.988 | 0.948 | 0.892 | 0.547 |

- Orchestration efficiency
  - MPI communication & CUDA management
  - aka Communication efficiency

Key:
1.10
1.00
0.90
0.80
0.70
0.60
0.50
0.40
0.30
0.20
0.10
0.00

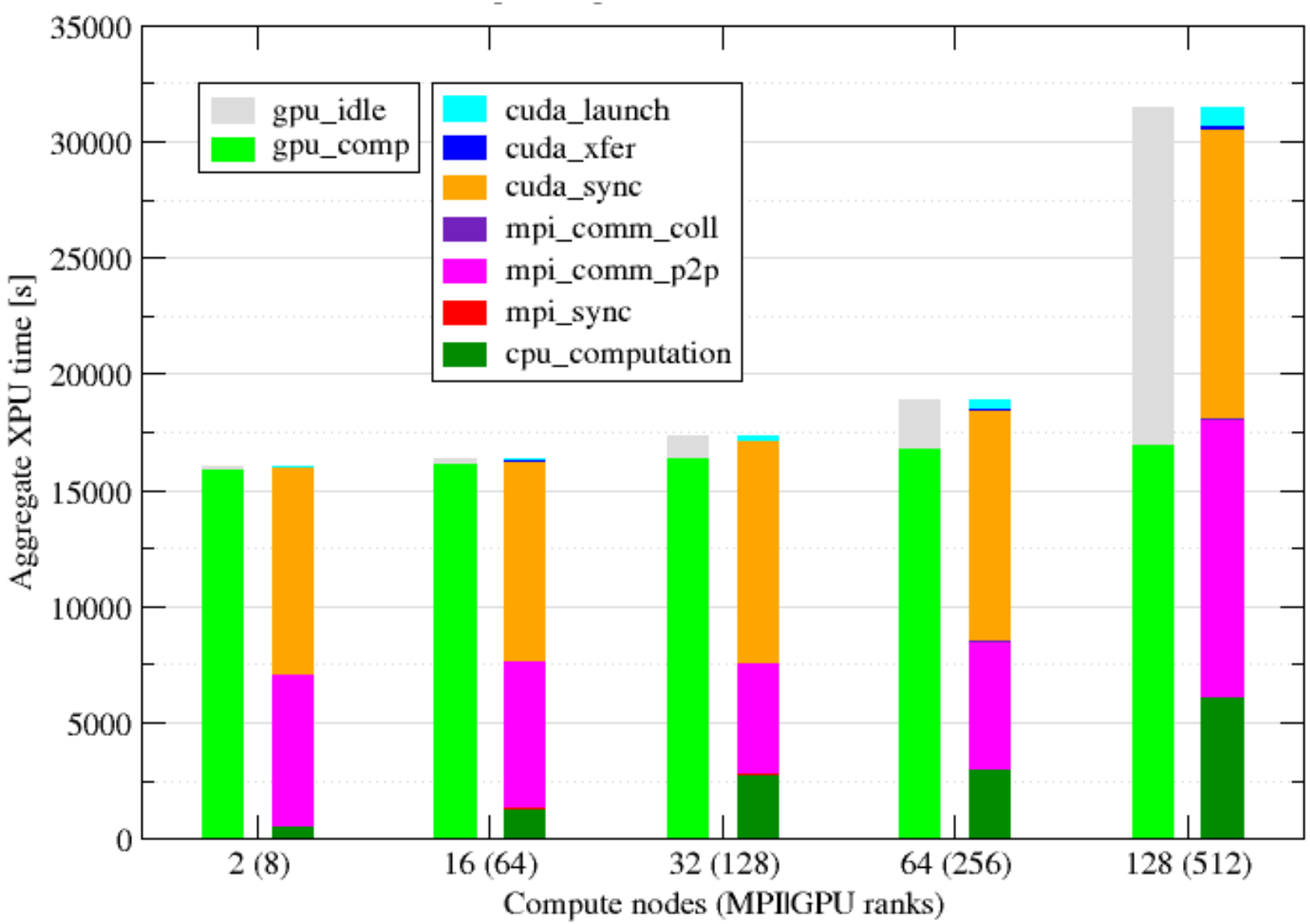
- Good GPU weak scaling efficiency to 128 GPUs (excellent load balance)
- Very poor CPU efficiency?
- Moderate XPU (GPU+CPU) efficiency?

# Weak scaling (128x128 per rank)




- Excellent weak scaling
- Little GPU idle time
- MPI communication effectively overlapped with GPU kernel computation

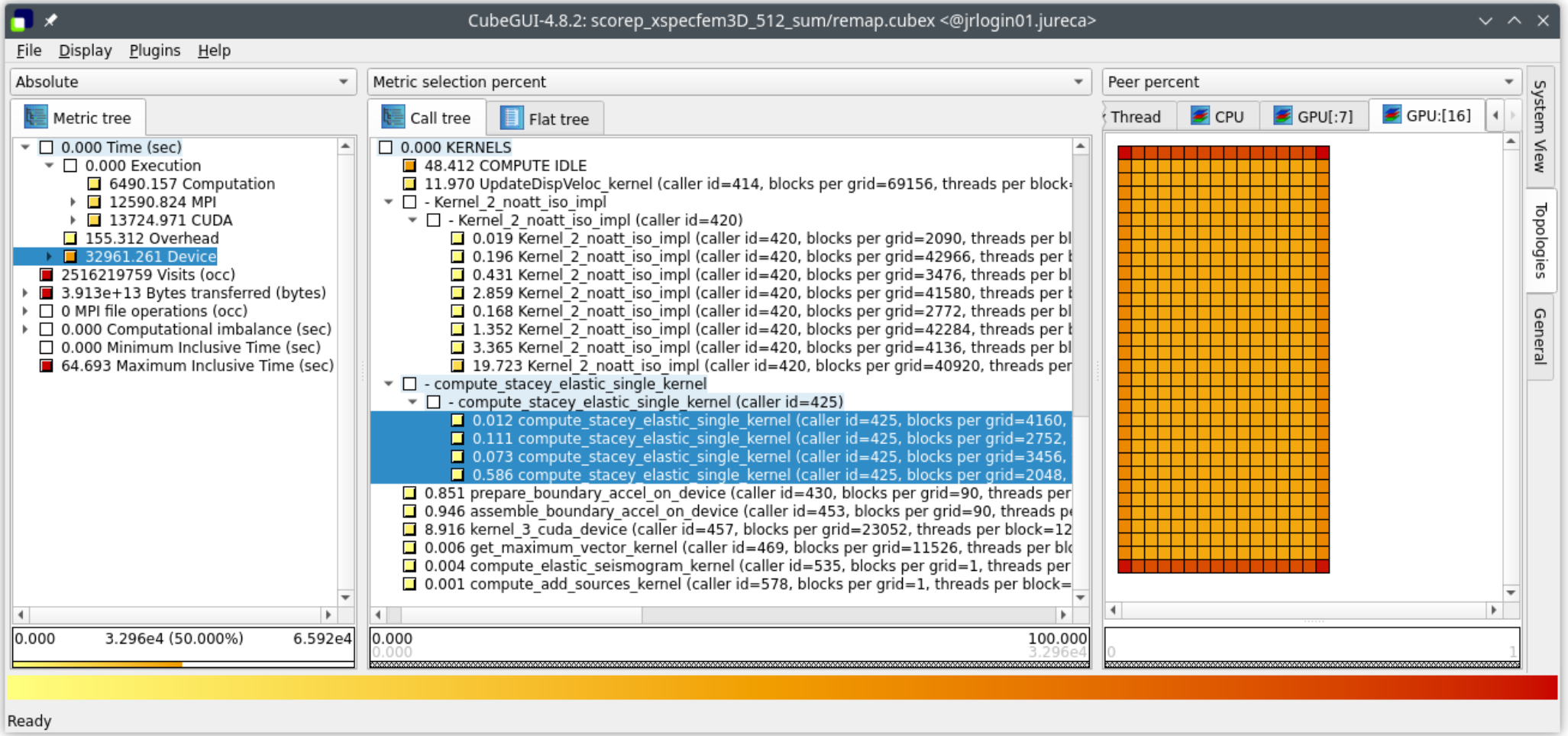
# Strong scaling (1024x1024 total)




- Good scaling to 256 GPUs (64 nodes)
- GPU computation time slowly grows progressively
- GPU idle time grows for 256 & particularly 512 GPUs
- CPU computation time grows substantially
  - sync_copy_from_device & transfer_boundary_to_device_a
- For 512 GPUs, growing MPI communication no longer fully overlapped with GPU kernel computation

# Topological inhomogeneities

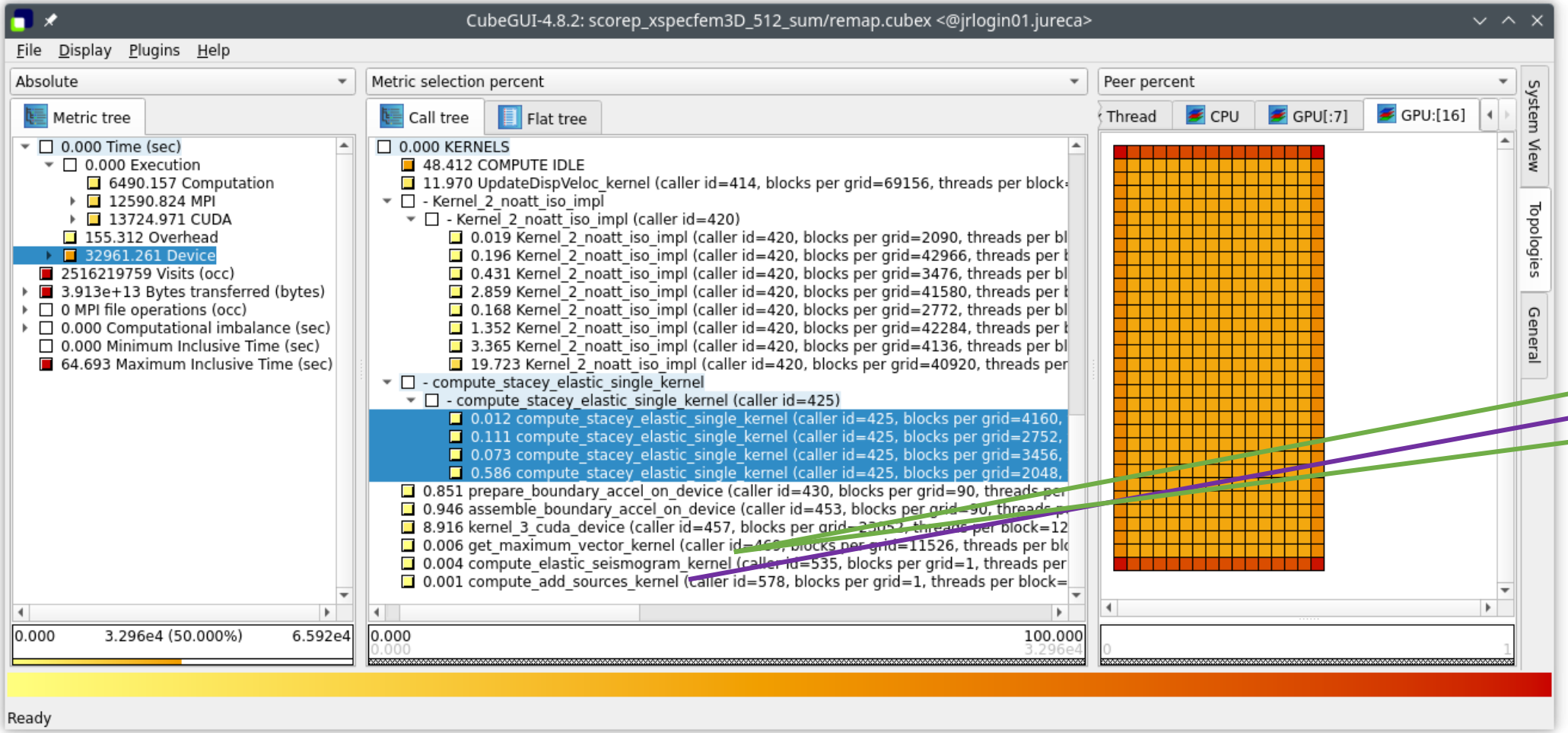
Kernel variant executed (characteristics and corresponding execution time) varies according to position in 2D grid: four corners, upper/lower & left/right edges, interior




16x32 grid

# Topological inhomogeneities

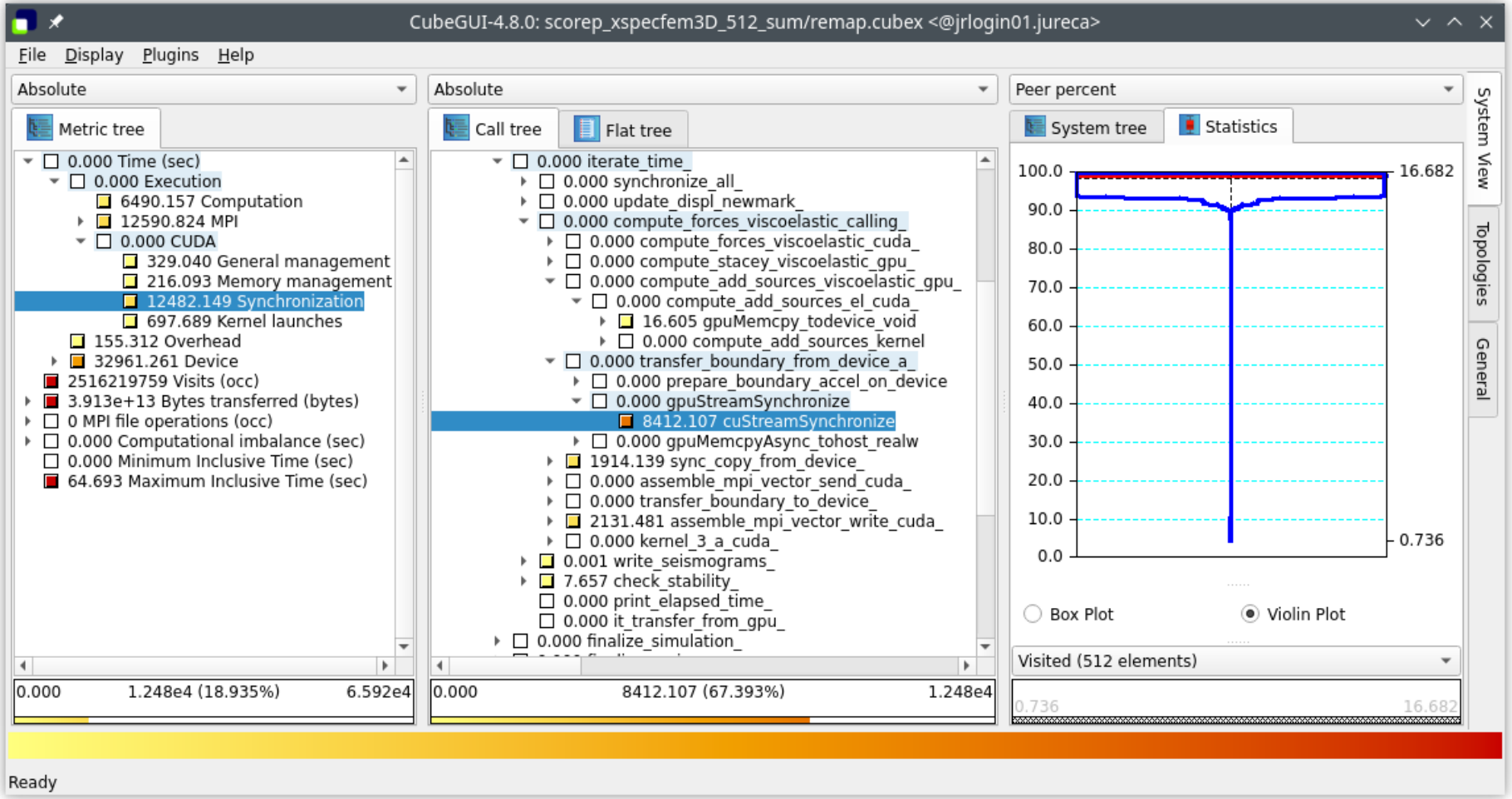
Kernel variant executed (characteristics and corresponding execution time) varies according to position in 2D grid: four corners, upper/lower & left/right edges, interior

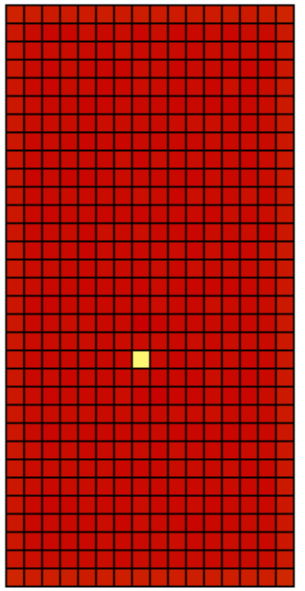



16x32 grid

# GPU computation imbalance

36% of CPU execution time is CUDA synchronization, 67% of which is 16s within *transfer_boundary_from_device_a* following *compute_add_sources_kernel* that's only executed by a single GPU (source rank 243 of 512)



# Summary of observations

- *iterate_time* (solver) chosen as focus of analysis
  - negligible time for initialization/finalization
- Excellent weak scaling up to 16 nodes (64 GPUs) and likely beyond
  - Computation very well balanced over GPUs; Excellent GPU utilization
  - MPI P2P communication time grows with scale, but effectively overlapped with GPU computation kernels
- Good strong scaling speedup up to 64 nodes (256 GPUs)
  - Computation remains very well balanced over GPUs
  - Orchestration efficiency progressively diminishes
    - *compute_add_sources_kernel* execution by a single GPU seems the main origin
  - MPI P2P communication time grows significantly, becomes no longer fully overlapped with GPU computation kernels

# Performance Optimisation and Productivity

A Centre of Excellence in HPC

## Contact:

- https://www.pop-coe.eu
- pop@bsc.es
- @POP_HPC
- youtube.com/POPHPC

This project has received funding from the European High-Performance Computing Joint Undertaking (JU) under grant agreement No 101143931. The JU receives support from the European Union's Horizon Europe research and innovation programme and Spain, Germany, France, Portugal and the Czech Republic.