Sample performance assessment extract (MPI+OpenMP+CUDA)

Brian Wylie, Jülich Supercomputing Centre, June 2026

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

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

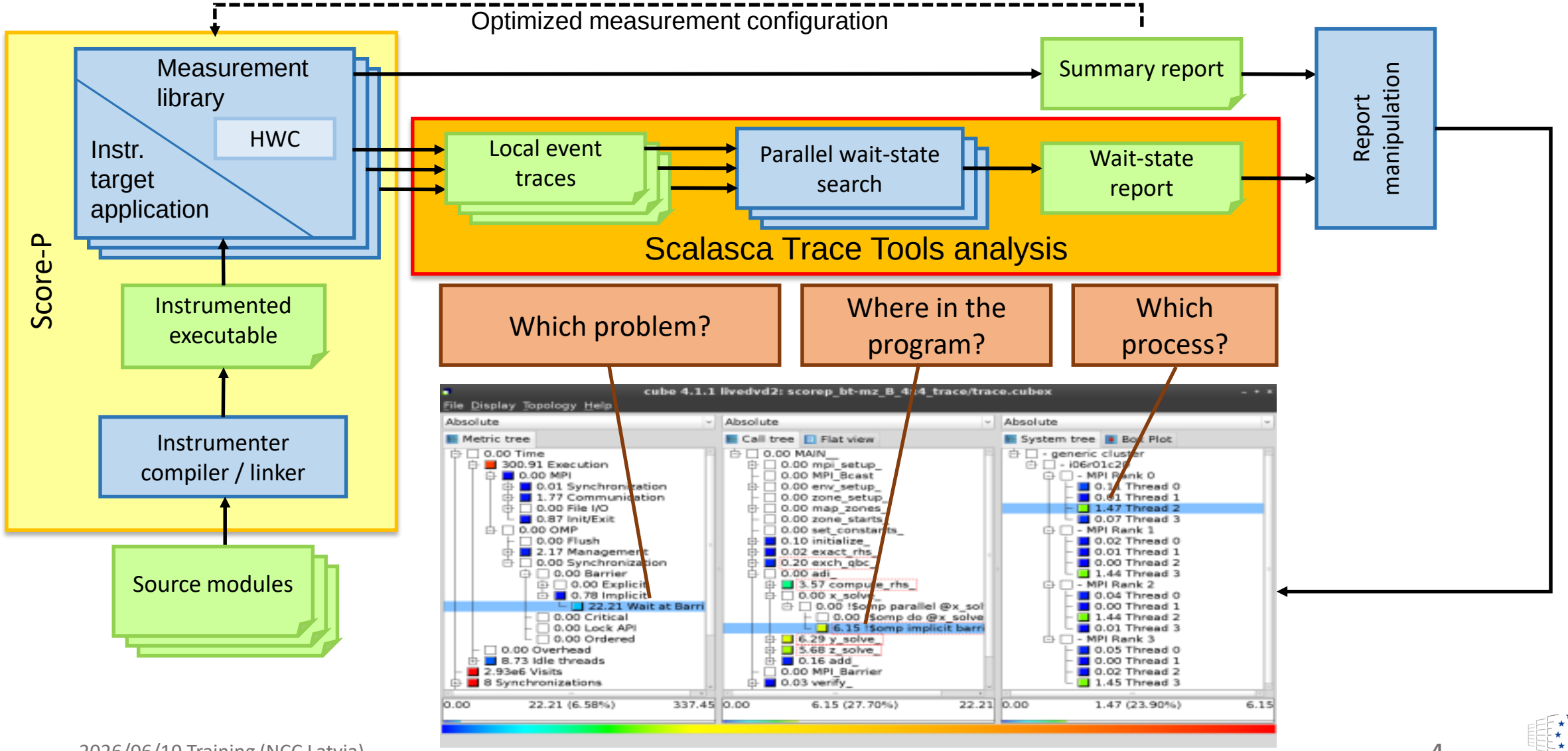
- Supported parallel paradigms:
  - Multi-process: MPI, SHMEM
  - Thread-parallel: OpenMP, Pthreads
  - Accelerator-based: CUDA, HIP, OpenCL, OpenACC, Kokkos

- 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

EuroHPC
Joint Undertaking

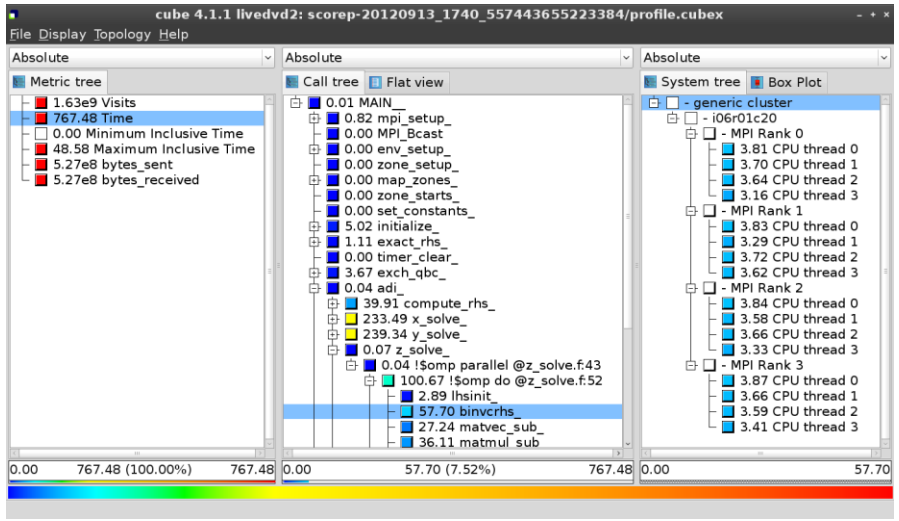
Scalasca workflow



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
- Contact:
  - mailto: scalasca@fz-juelich.de




JUPITER
THE FIRST EXASCALE SYSTEM IN EUROPE
JUPITER
JOINING FORCES
GCS
JÜLICH
ParTec
EVIDEN
NVIDIA
SiPEARL
IBM
jupiter.fz-juelich.de
EuroHPC
Joint Undertaking
JÜLICH
Forschungszentrum

Application use case

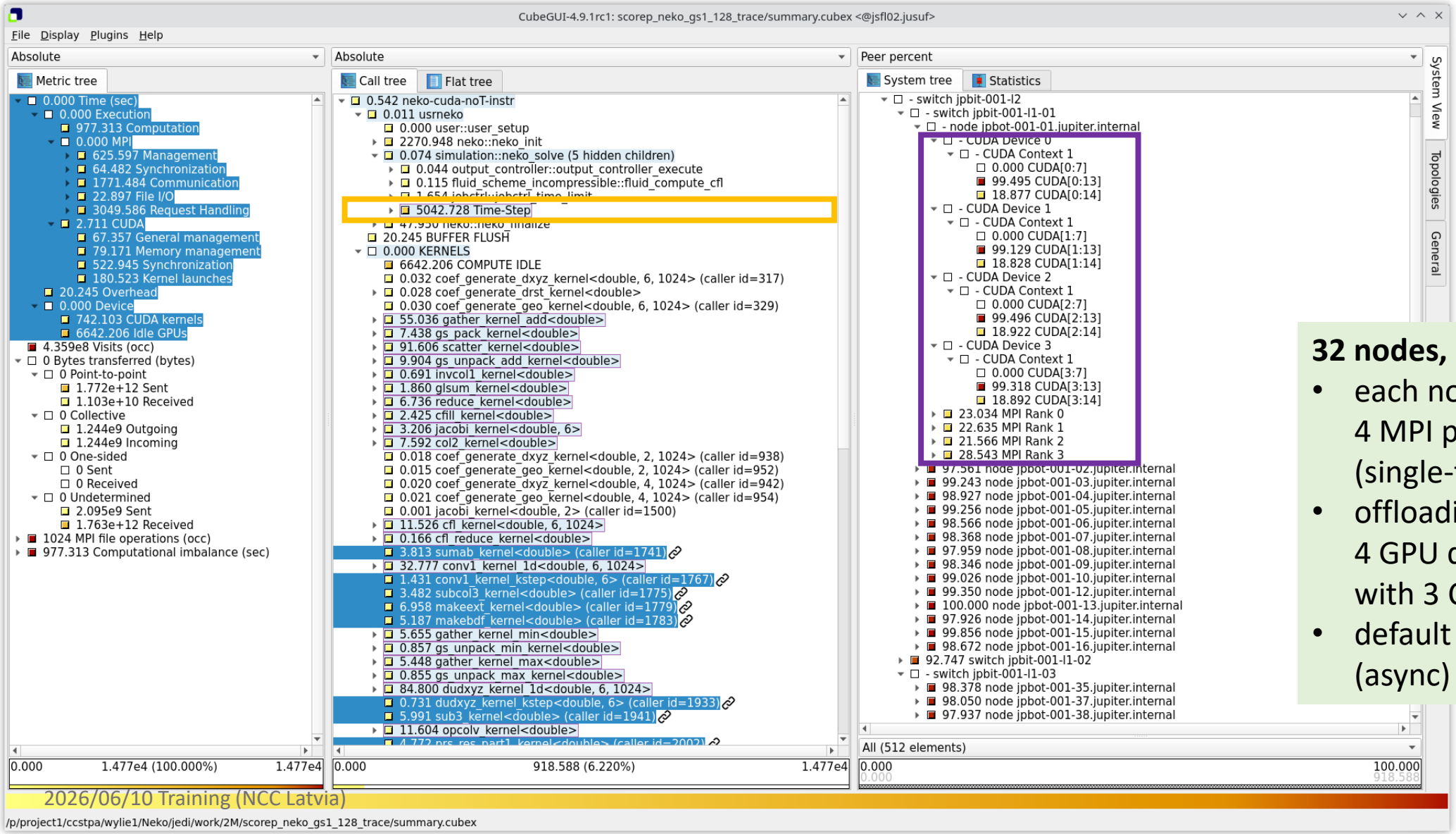
- Neko
  - Portable framework for high-order spectral-element flow simulations
    - roots in Nek5000
  - Assessment requested by Niclas Jansson (KTH/S) for CEEC CoE
  - Version 0.9.99
- Fortran parallelized with MPI (+OpenMP) & CUDA: one MPI process per GPU
  - **GCC** 13.3.0, **OpenMPI** 5.0.5 (GPU Aware), OpenBLAS 0.3.27, CUDA 12.6.0
- Testcase: Taylor-Green Vortex (TGV) ExaCB benchmark
  - 301 solver time-steps, 3D meshes with 32768, 262144, **2M** & 16M elements
- Executed on *JEDI* Eviden BullSequana XH3000 (JSC)
  - test & development module for *JUPITER*
  - 48 compute nodes with quad Nvidia GH200 superchips:
    72-core Nvidia Grace (Arm Neoverse-V2) CPUs [120GB] @ 3.1 GHz and Nvidia H100 ('Hopper') GPUs [96GB]
  - Nvidia NVLink-C2C interconnection network
- Measurements with Scalasca/2.6.2 using Score-P/9.0, analysis with Vampir/10.5 & CUBE/4.9β
- Focus of analysis (FOA): neko-solve/Time-Step

Instrumentation & measurement

- Score-P instrumenter captured NVTX region markers of application
  - Initial measurement filter subsequently applied when instrumenting

- Score-P measurement dilation 7-40% (for 2M TGV testcase)
  - run-to-run variability typically 10-20% for *solve_time* (both 1 or 2 threads)

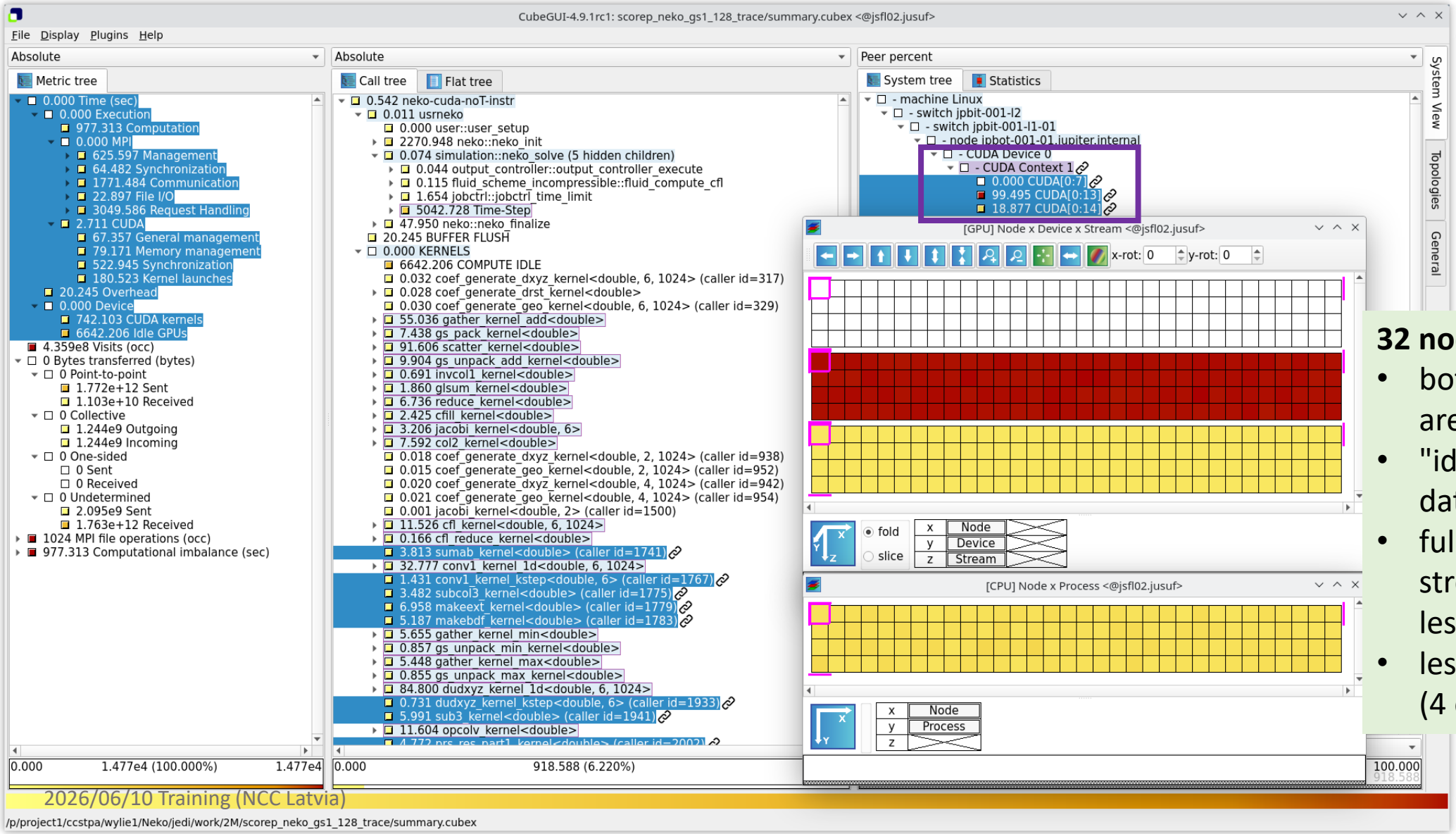
CUBE GUI: Time-Step [2M]



32 nodes, 128 GPUs

- each node running 4 MPI processes (single-threaded)
- offloading kernels to 4 GPU devices each with 3 CUDA streams
- default stream 7 for (async) data transfers

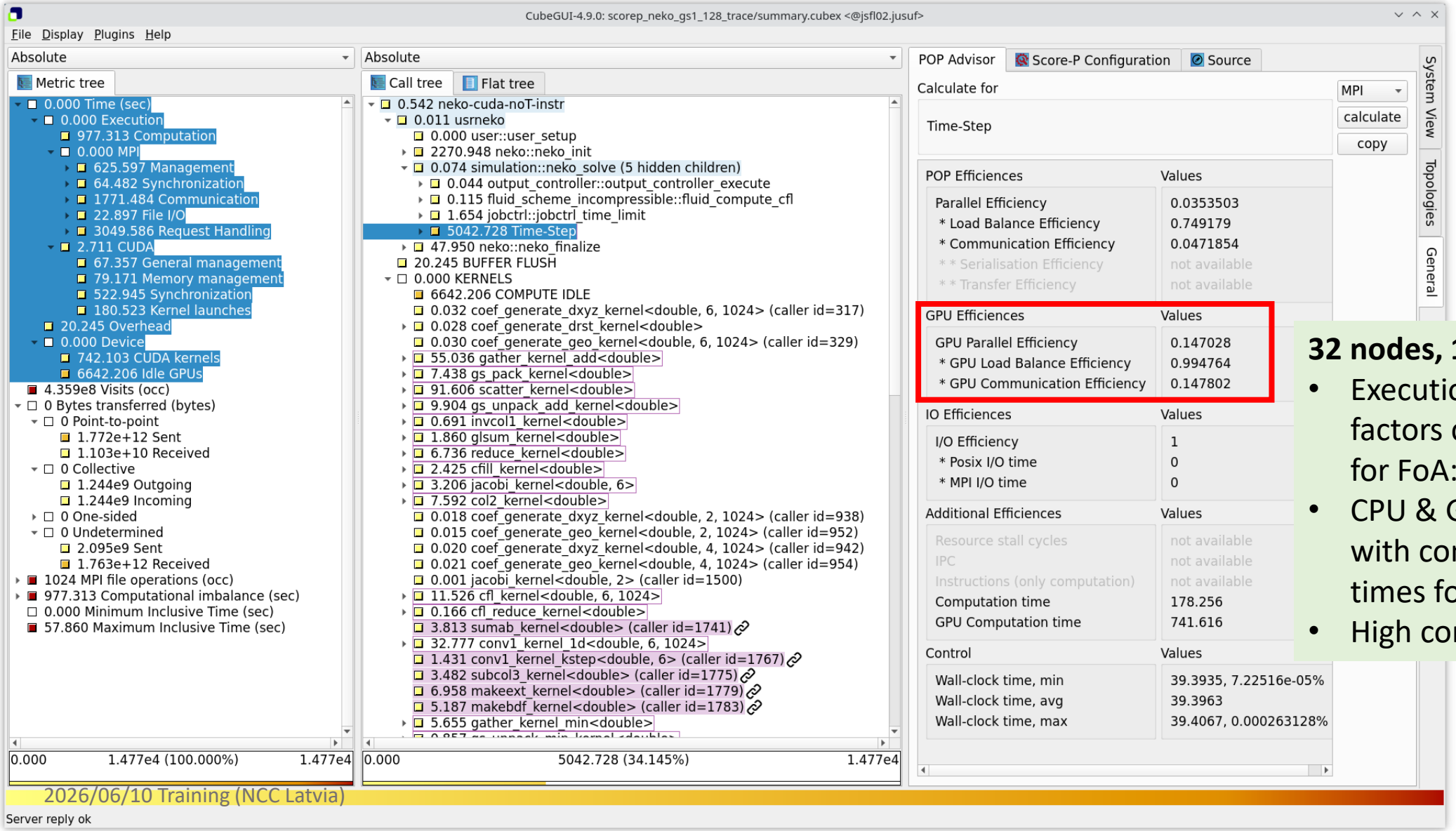
CUBE CPU & GPU topologies



32 nodes, 128 GPUs

- both GPUs & CPUs are well balanced
- "idle" stream 7 for data transfers
- fully-utilized GPU stream 13, and less used stream 14
- less loaded CPUs (4 of 72 available)

CUBE POP Advisor: Time-Step [2M]

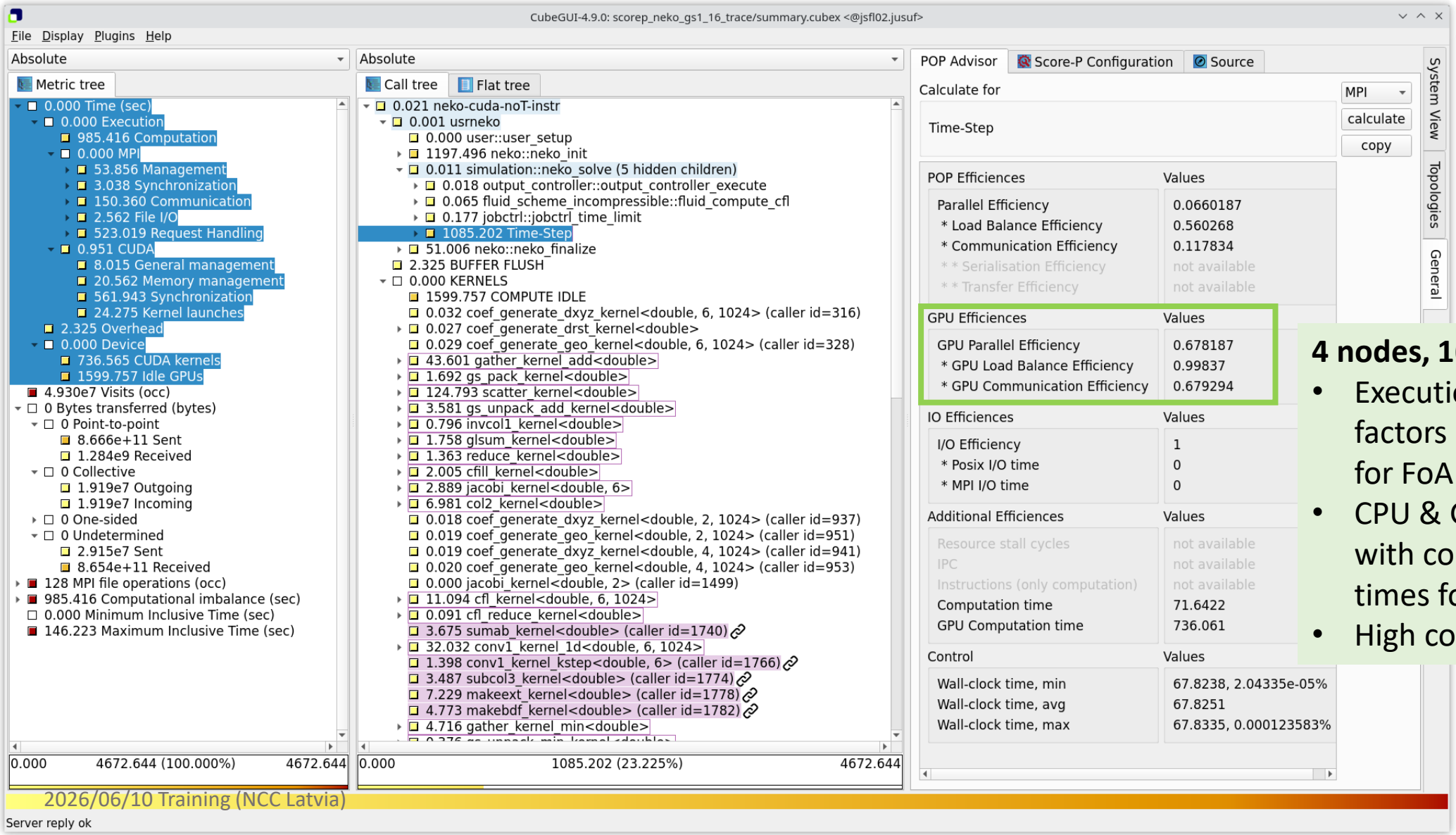


32 nodes, 128 GPUs

- Execution efficiency factors calculated for FoA: *Time-Step*
- CPU & GPU variants with computation times for each
- High confidence FoA

EuroHPC Joint Undertaking

CUBE POP Advisor: Time-Step [2M]



4 nodes, 16 GPUs

- Execution efficiency factors calculated for FoA: *Time-Step*
- CPU & GPU variants with computation times for each
- High confidence FoA

EuroHPC Joint Undertaking

Efficiency model

2M		gs1_16	gs1_32	gs1_64	gs1_128
	Nodes	4	8	16	32
	GPUs	16	32	64	128
	Wall [s]	146.07	79.15	59.09	57.86
	Time-Step [s]	**67.83**	**43.32**	**34.15**	**39.41**
GPU	Global scaling efficiency	0.678	0.531	0.336	0.146
	- Computation scaling	1.000	0.973	0.998	0.993
	- Parallel efficiency	0.678	0.546	0.337	0.147
	- - Load balance efficiency	0.998	0.997	0.997	0.995
	- - Orchestration efficiency	0.679	0.547	0.339	0.148
CPU	Global scaling efficiency	0.066	0.052	0.033	0.014
	- Computation scaling	1.000	1.077	0.713	0.403
	- Parallel efficiency	0.066	0.048	0.046	0.035
	- - Load balance efficiency	0.561	0.794	0.801	0.747
	- - Orchestration efficiency	0.118	0.06	0.057	0.047

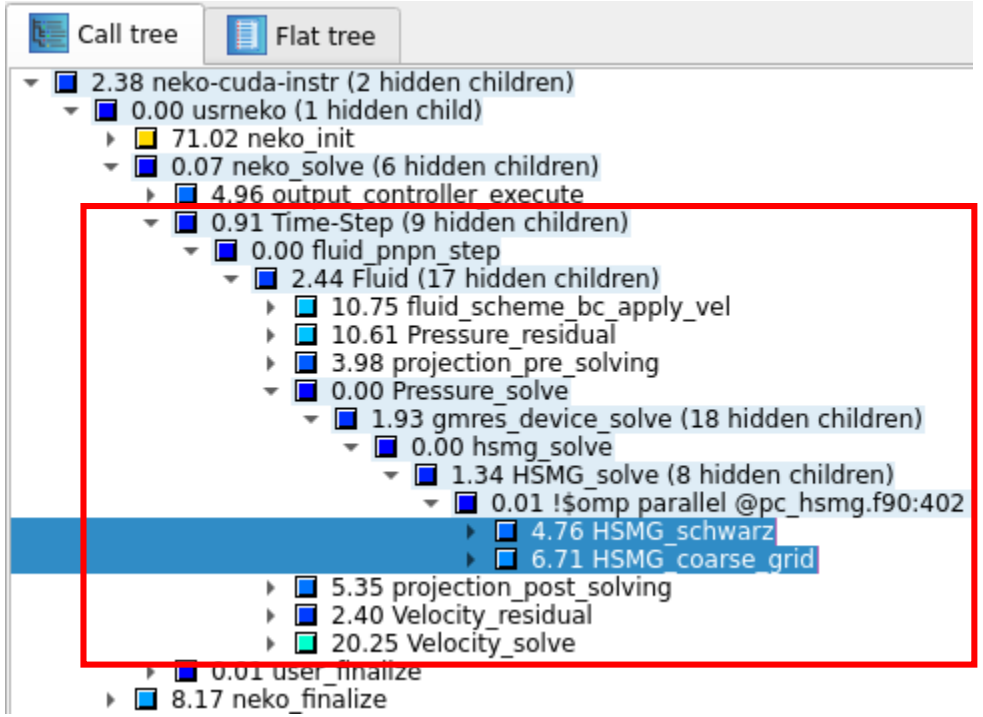
Orchestration efficiency

- MPI communication & GPU management
- aka Communication efficiency

- Excellent GPU strong scaling load balance efficiency & computation scaling
- Poor parallel & orchestration efficiencies, deteriorating with increasing scale
- Very poor CPU efficiencies

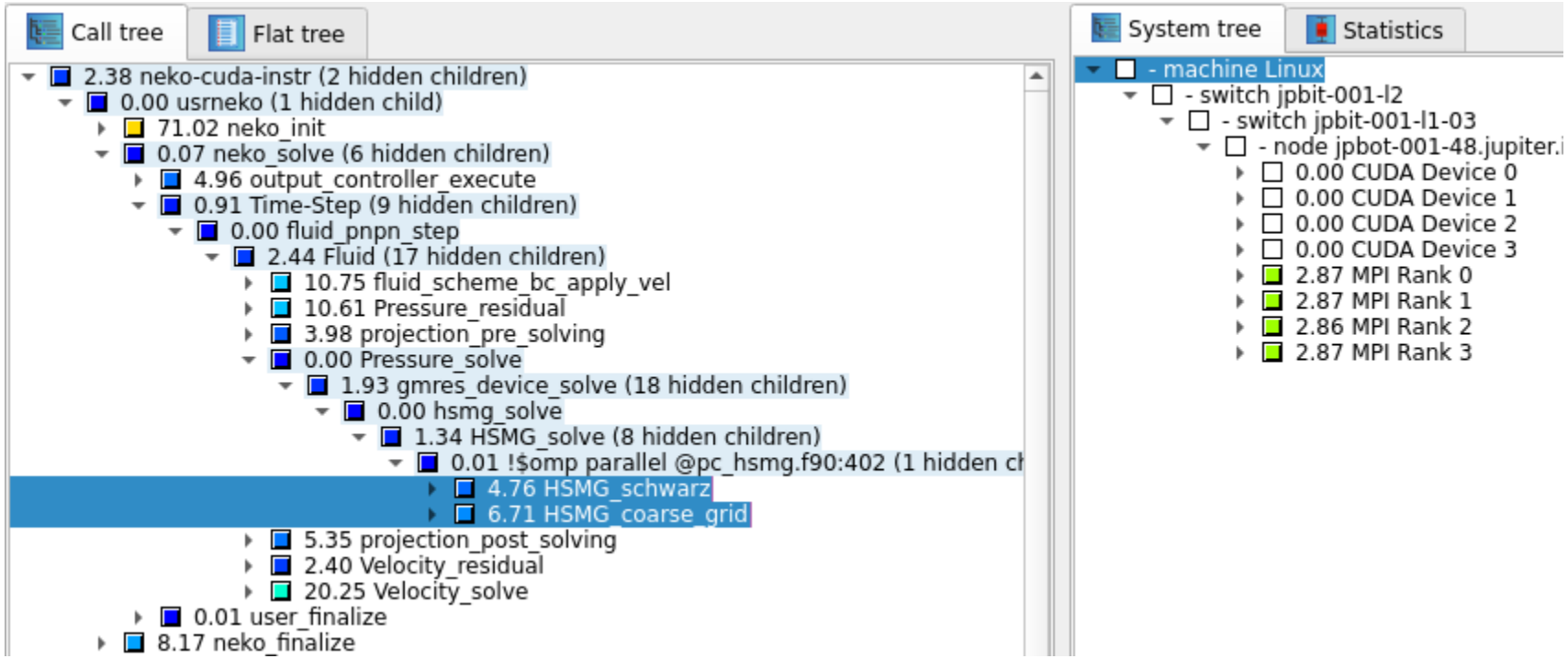
EuroHPC
Joint Undertaking

Execution call-tree & Focus of Analysis



- Expensive initialization (neko_init) for small testcases
- Output disabled for TGV benchmark
- Time-Step within neko_solve used as Focus of Analysis for assessment
  - time also reported directly by application
  - 301 steps for TGV benchmark testcase

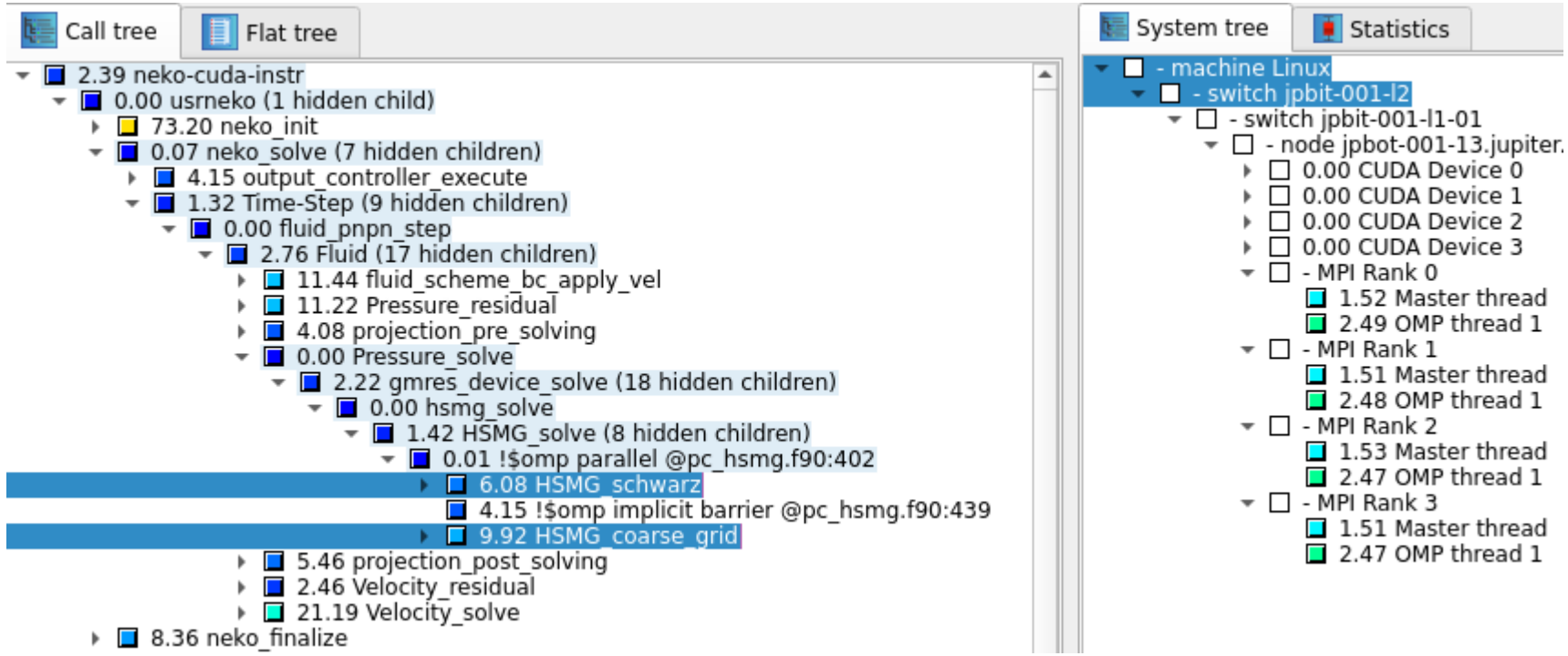
Time-Step FOA structure



NVTX marked regions augment function profile

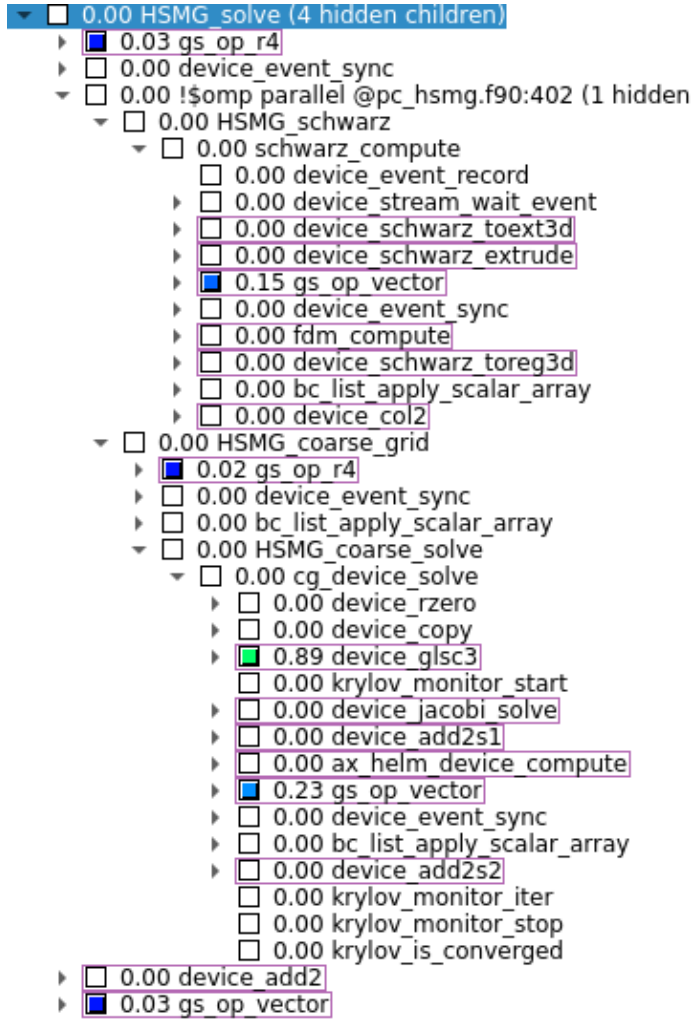
- Time-Step within neko_solve contains Fluid simulation
- Pressure_solve/HSMG_solve contains OpenMP parallel region for two threads
- HSMG_Schwarz & HSMG_coarse_grid (potentially) executed by different threads

Time-Step FOA with OpenMP threads



- Time-Step within neko_solve contains Fluid simulation
- Pressure_solve/HSMG_solve contains OpenMP parallel region for two threads
- HSMG_Schwarz & HSMG_coarse_grid (potentially) executed by different threads
  - concurrent execution is slightly faster
  - however, each thread's computation is somewhat slower
  - and the other serial computation is also slower!

HSMG_solve structure detail



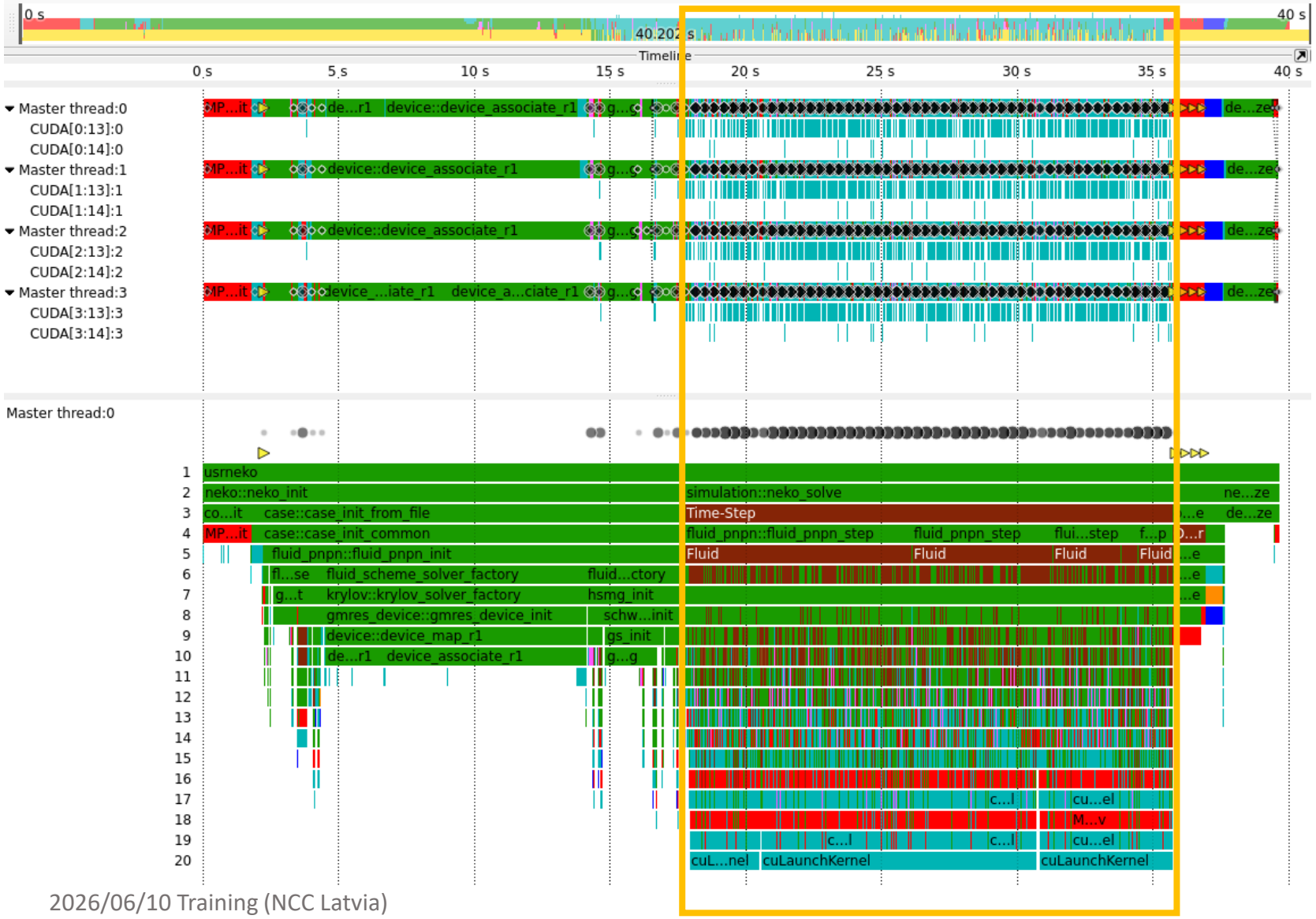
MPI

- device_glsc3 includes MPI_Allreduce
- gs_op routines include GatherScatter using MPI P2P
- both include CUDA kernels (possibly on additional streams)
  - gather_kernel_add, scatter_kernel, gs_pack_kernel, gs_unpack_kernel

HSMG_schwarz & HSMG_coarse_grid

- both do MPI communication
- both launch CUDA kernels (to different streams)
  - HSMG_Schwarz uses stream 14 (etc)
    - only used here
  - HSMG_coarse_grid uses stream 13 (etc)
    - common stream used elsewhere

Execution timeline



Score-P trace shown by Vampir

Single node execution
- 4 MPI processes each using a dedicated GPU device via 2 CUDA streams

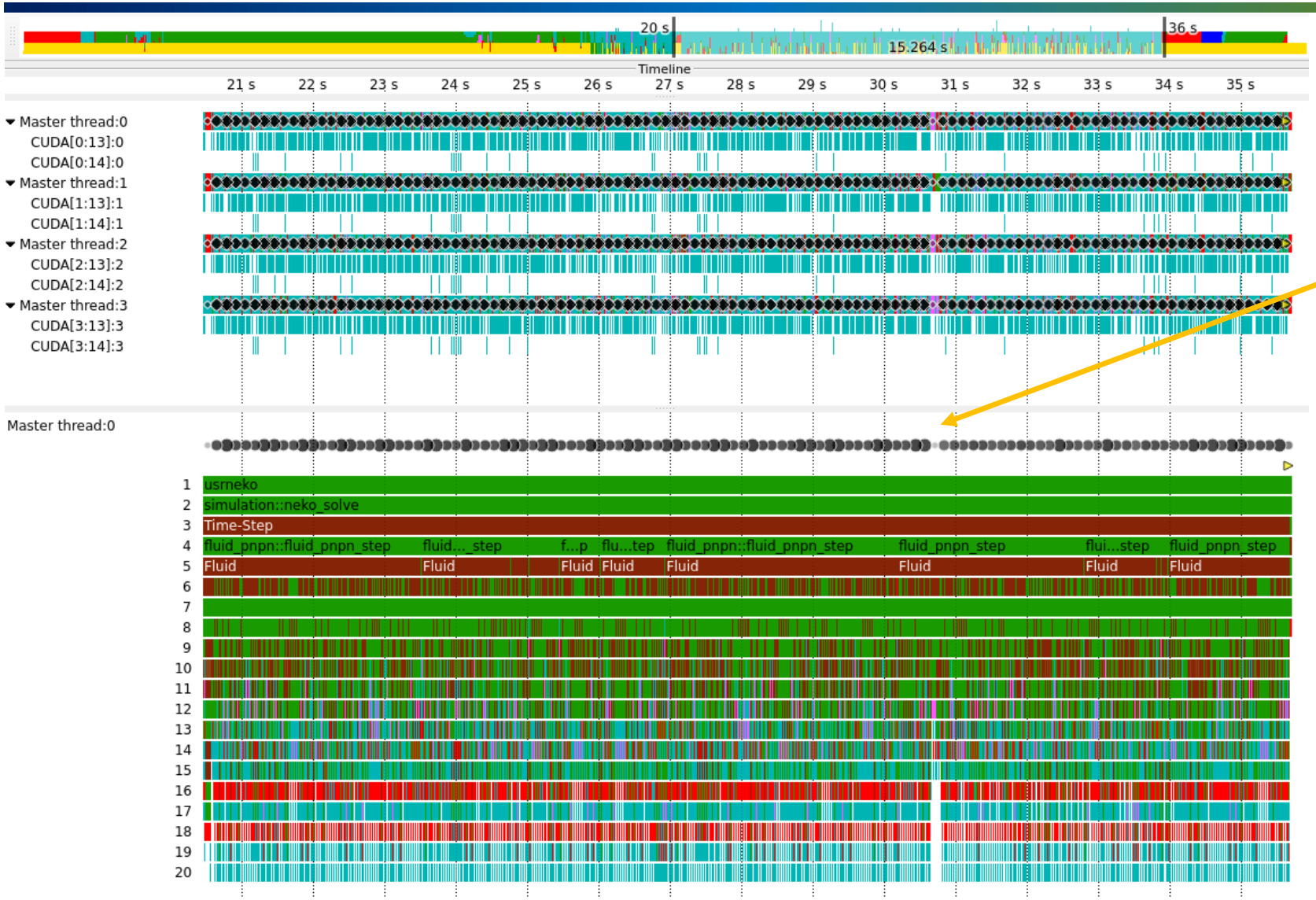
Substantial initialization
- dominated by some device_associate calls

Solve = Focus of Analysis
- multiple Time-Steps
- heavy GPU utilization

Finalization
- (optional) file output

EuroHPC
Joint Undertaking

Execution timeline: solve (FOA)



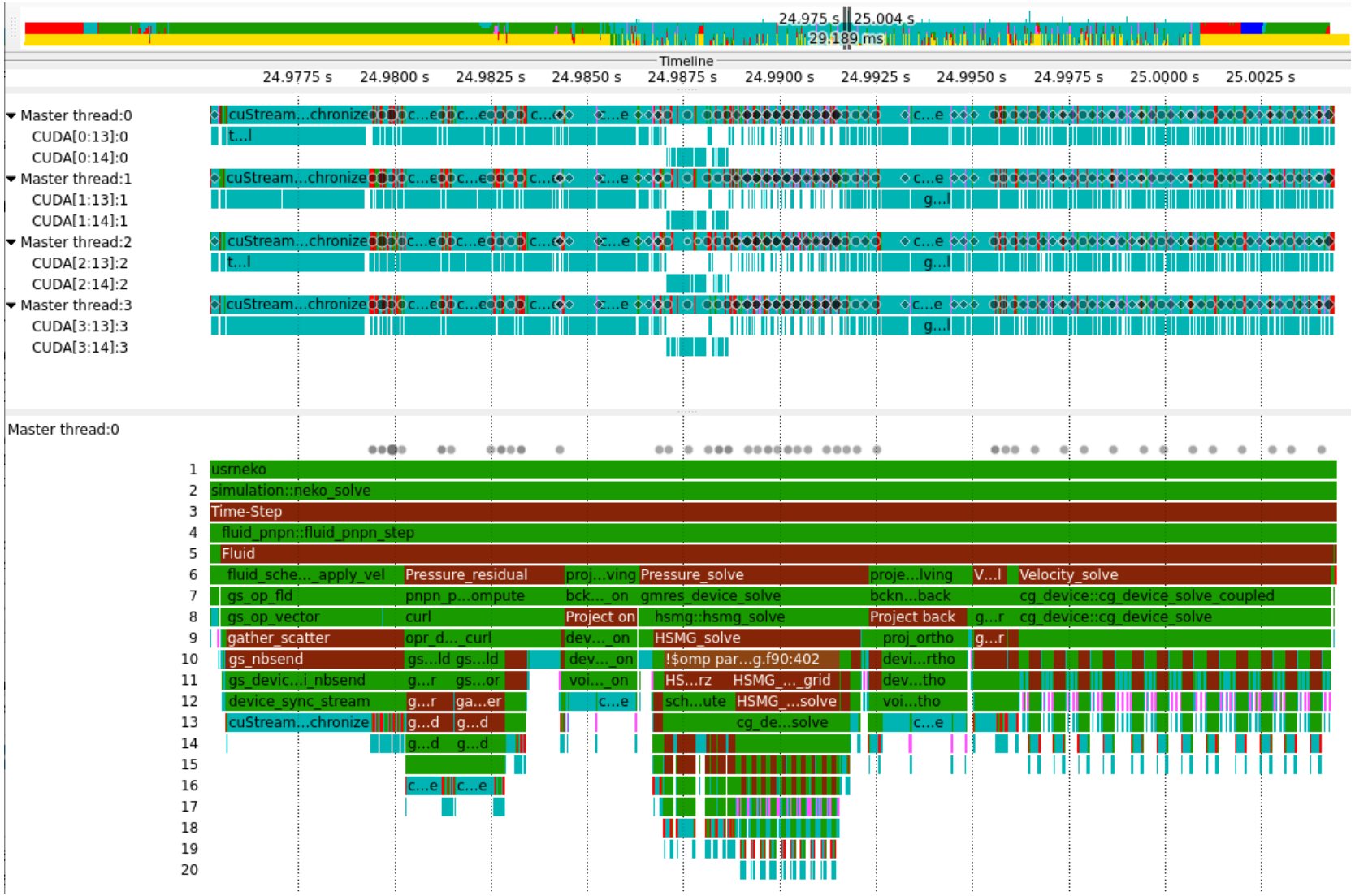
Solve

- multiple Time-Steps
  - 301 for TGV bench case
- first rather longer than others
- occasional disruption

Heavy GPU utilization

- mostly stream 13
- occasionally stream 14
- (rarely other streams for GS)

Execution timeline: solve (single step)



One of 301 simulation time-steps

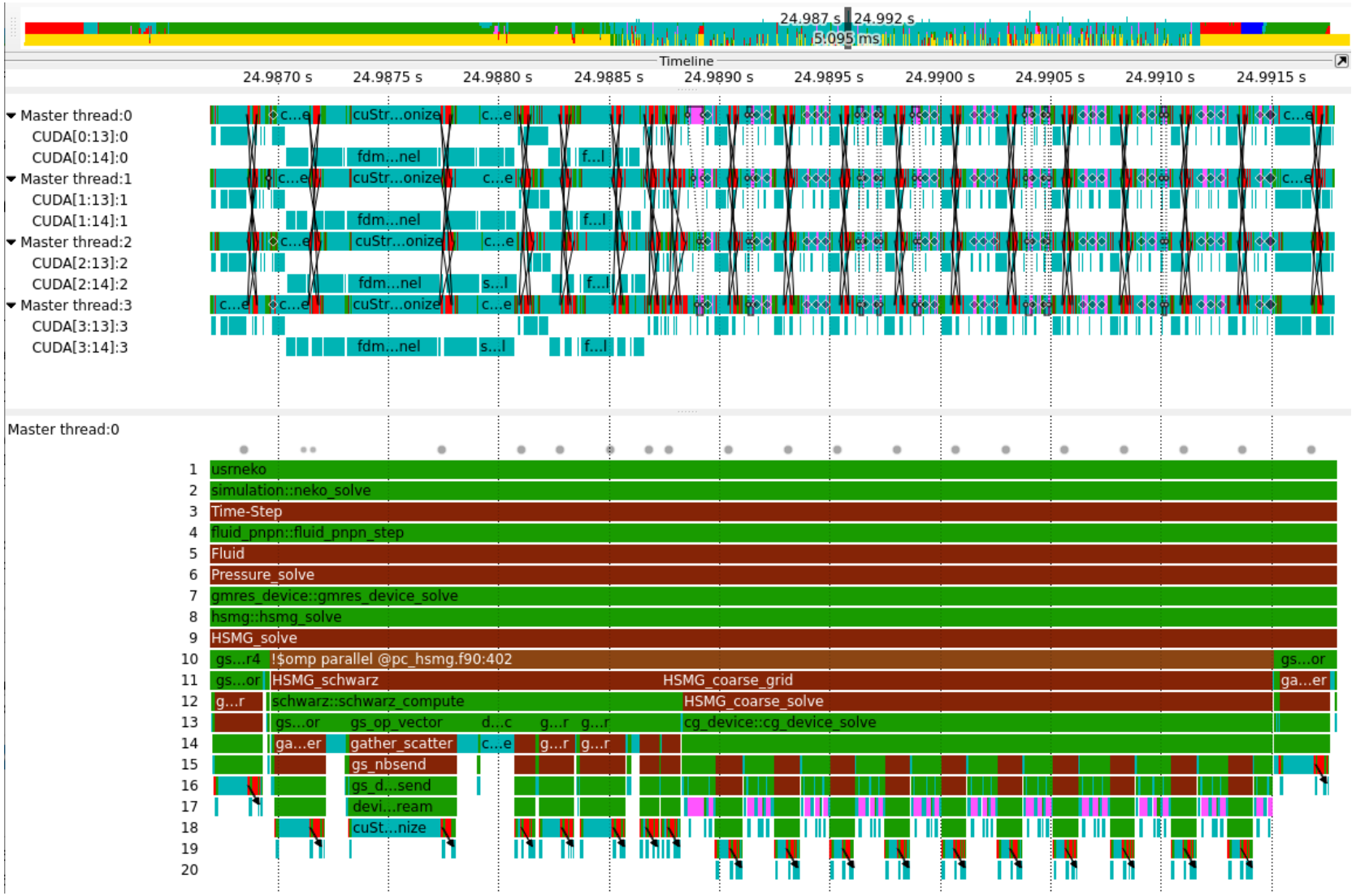
Time-Step (Fluid)

- fluid_scheme_bc_apply_vel
- Pressure_residual
- Project_on
- **Pressure_solve***
- Project_back
- Velocity_residual
- Velocity_solve

Heavy GPU utilization

- mostly stream 13
- partially stream 14

Execution timeline: Pressure solve



HSMG_solve

- HSMG_schwarz
  - (CUDA stream 14)
- HSMG_coarse_grid
  - (CUDA stream 13)

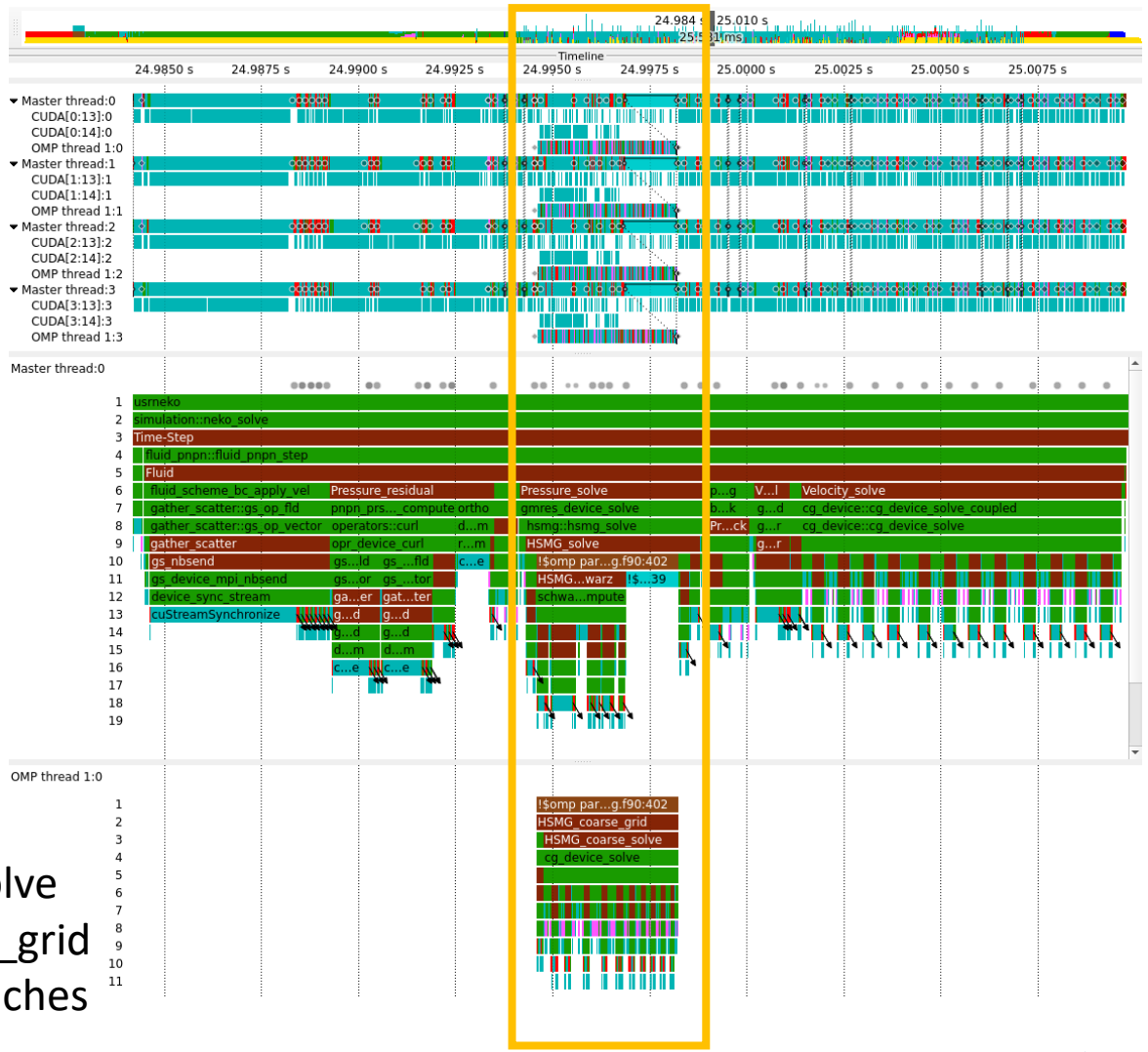
Executed serially

- OMP_NUM_THREADS=1

MPI communication

- gs_op_r4 (GS P2P)
- gs_op_vector (GS P2P)
- Allreduce

Single step comparison (1v2 threads)

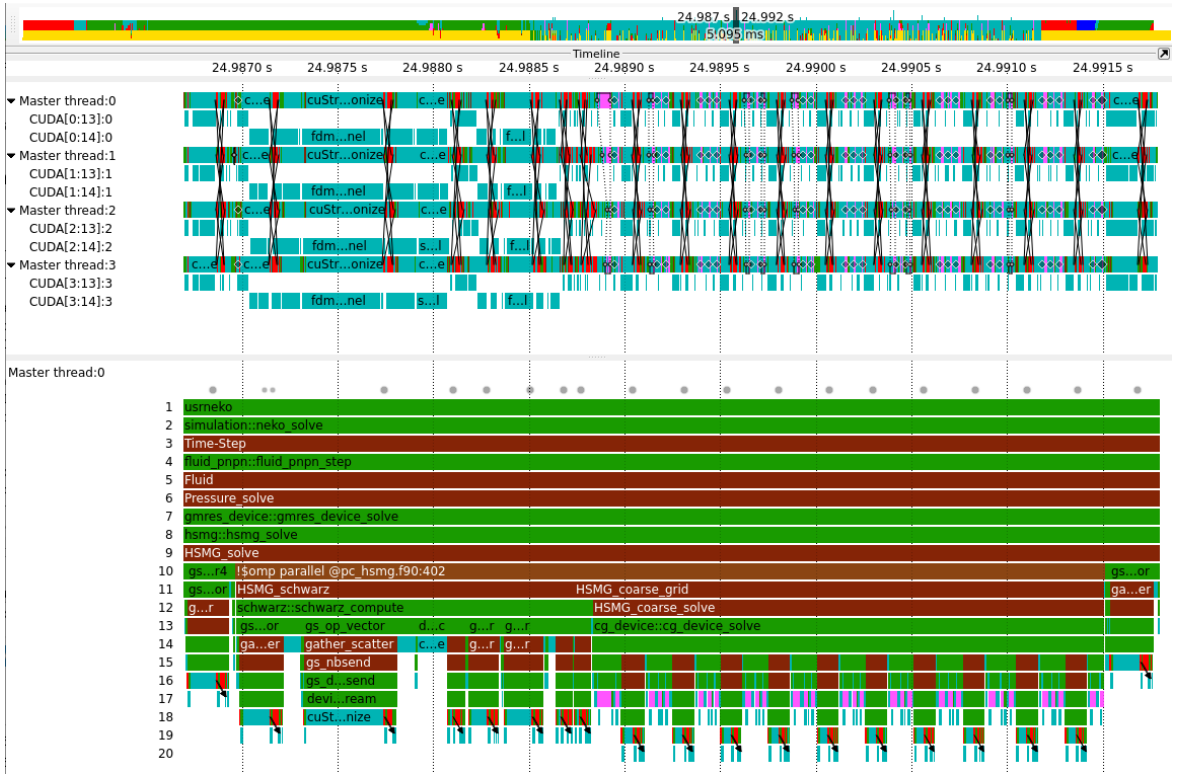


Execution with additional OpenMP thread only for HSMG_solve

- concurrent execution of HSMG_Schwarz & HSMG_coarse_grid
- both with MPI (GS) communication and CUDA kernel launches

EuroHPC Joint Undertaking

Single *psolve* comparison (1v2 thrds)



Execution with additional OpenMP thread only for HSMG_solve

- concurrent execution of HSMG_Schwarz & HSMG_coarse_grid
- each is slower, but together slightly faster

Acknowledgements

Score-P development supported by Joint Laboratory for Extreme-Scale Computing (JLESC) and Performance Optimisation and Productivity (POP) Centre of Excellence.

Neko test case provided by Niclas Jansson (KTH), Centre of Excellence in Exascale CFD (CEEC).

This project received access to *JEDI* (JUPITER Exascale Development Instrument), a preparation system for the *JUPITER* supercomputer, through the JUPITER Research & Early Access Programme (JUREAP).

The acquisition and operation of the *JUPITER* supercomputer is funded jointly by the European High-Performance Computing (EuroHPC) Joint Undertaking through the European Union's Digital Europe programme, the German Bundesministerium für Bildung und Forschung (BMFTR, Federal Ministry of Research, Technology & Space) and Ministerium für Kultur und Wissenschaft des Landes Nordrhein-Westfalen (MKW-NRW, Ministry of Culture & Science of North Rhine-Westphalia) via the Gauss Centre for Supercomputing (GCS).

Performance Optimisation & Productivity

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 & Czechia.