International HPC Summer School 2018: Performance analysis and optimization

Case Studies

VI-HPS Team
Ilya Zhukov – Jülich Supercomputing Centre

PENNANT

- **PENNANT** is an unstructured mesh physics mini-app designed for advanced architecture research
- contains mesh data structures and a few physics algorithms
- C++ application, supports MPI and OpenMP
- Developed by Los Alamos National Laboratory
- https://github.com/lanl/PENNANT

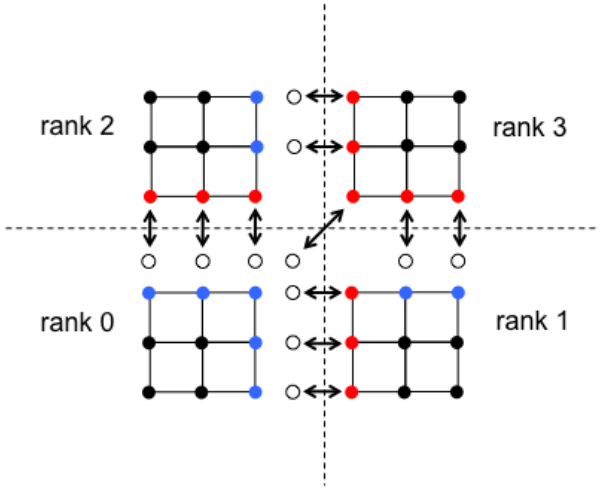
- **Nohpoly** testcase executed on Bridges (8 MPI x 4 OpenMP)
- Copy **pennant_8x4_bridges.cubex** to your laptop

```
https://fz-juelich.sciebo.de/s/JRFcwYZcaTRQ2sD
```

- Examine measurement with CUBE

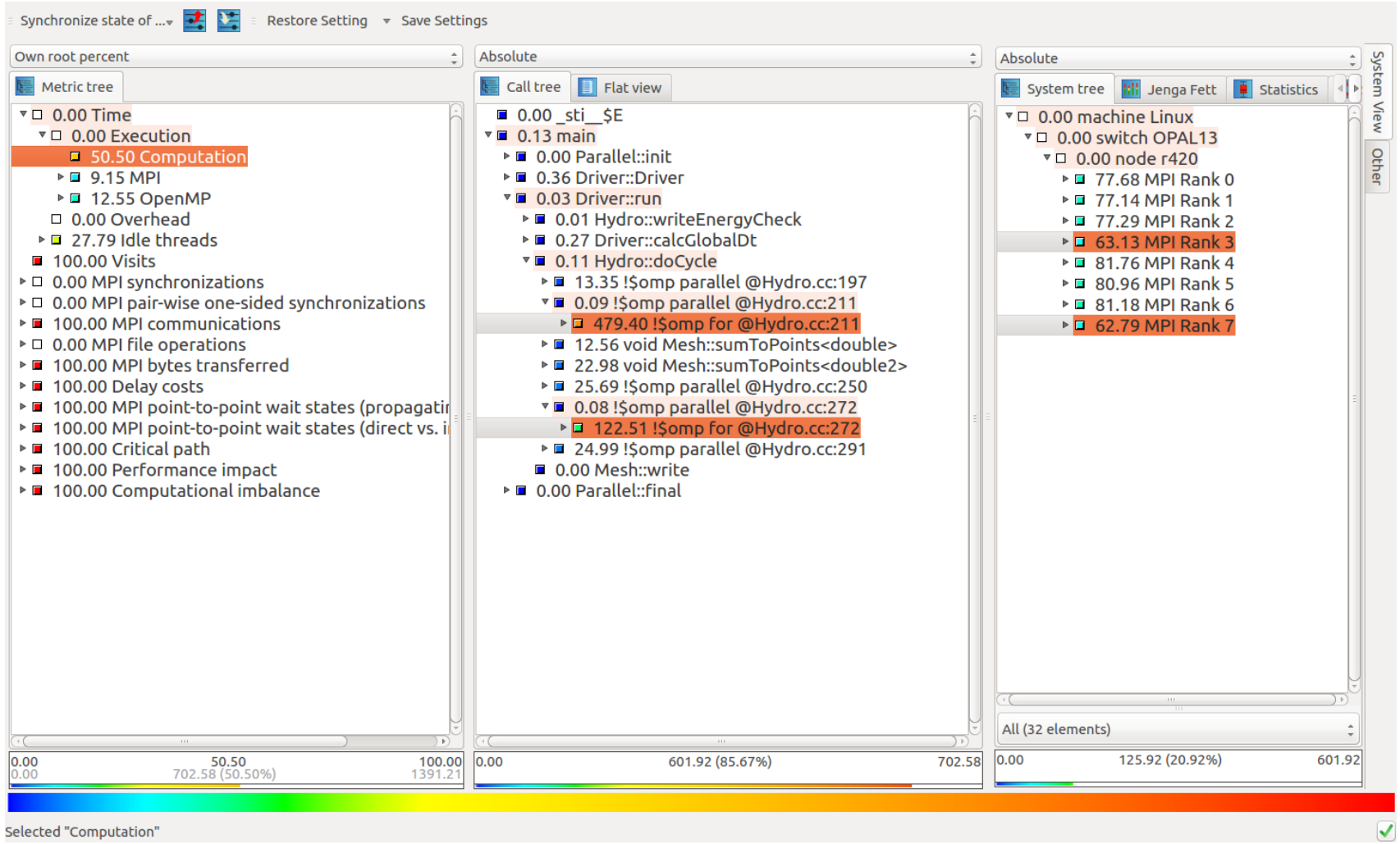
```
laptop> cube pennant_8x4_bridges.cubex
```

Domain decomposition



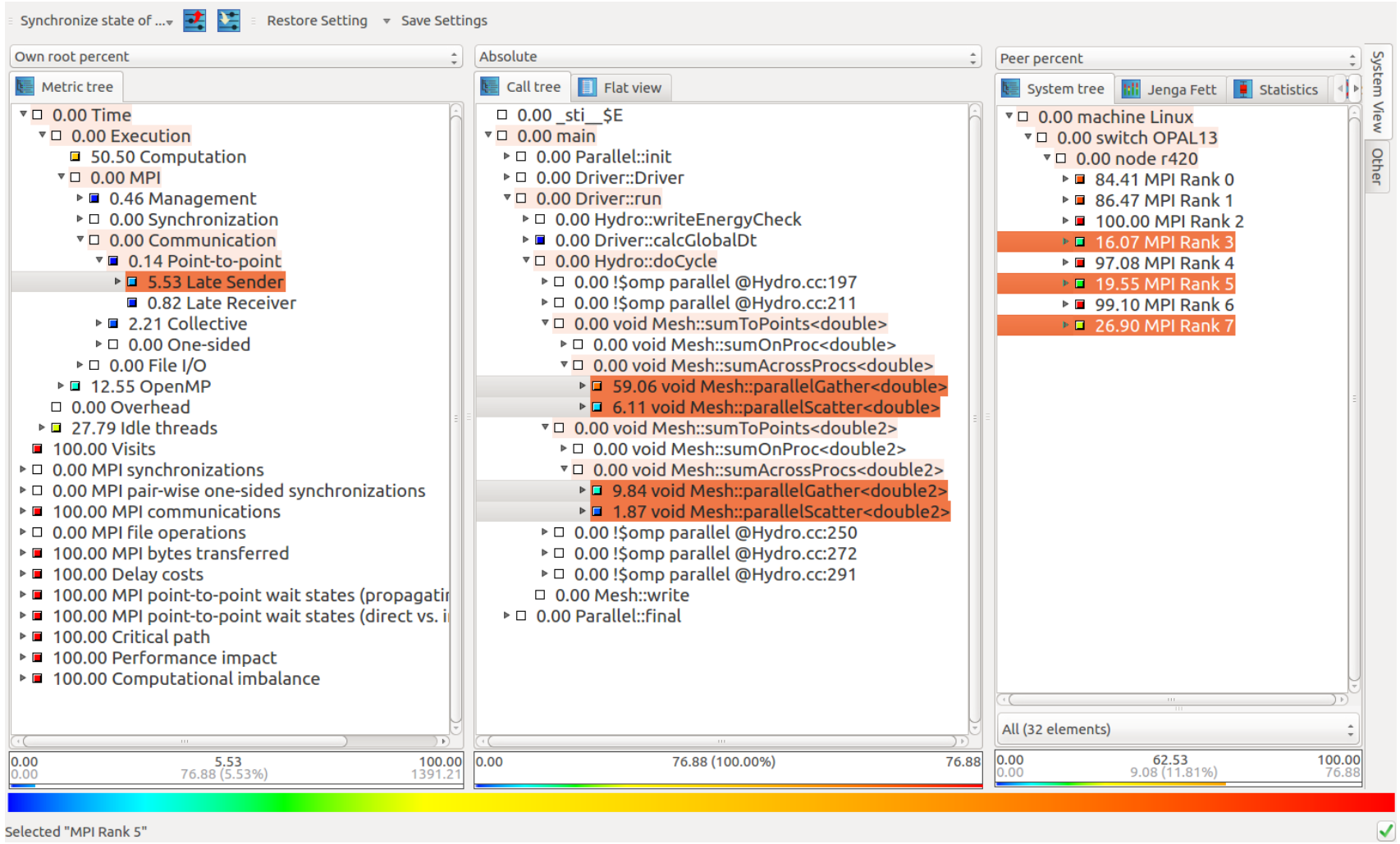
- In *Mesh::parallelGather*, slave (red) point values are assembled into messages, and sent to corresponding proxy points (white) on the same rank as their masters(blue).
- In *Mesh::parallelSum*, master points sum their own values and all proxy values, and store sum at master and all proxies.
- In *Mesh::parallelScatter*, the updated proxy point values are assembled into messages and sent back to their corresponding slave points.

PENNANT: execution breakdown



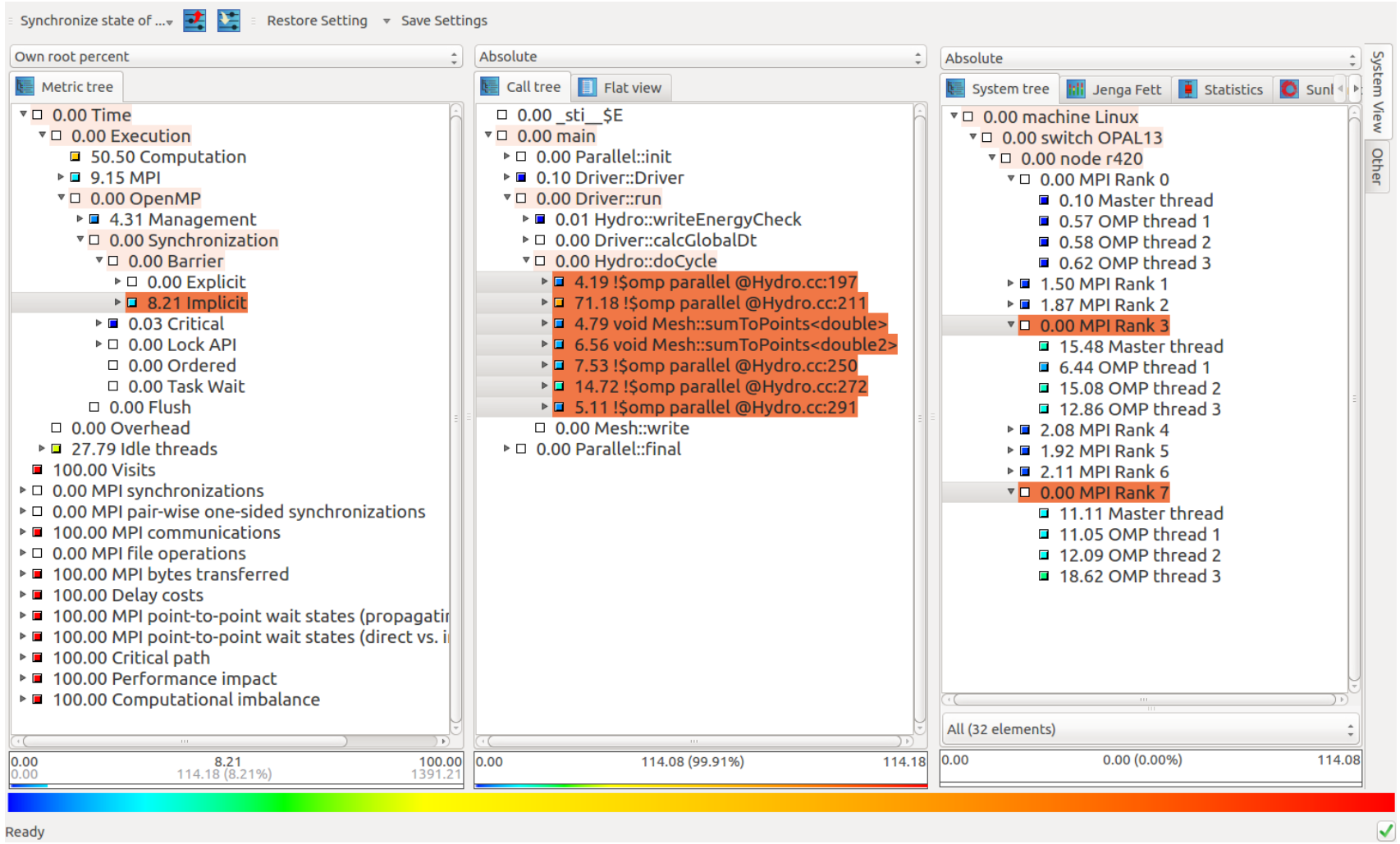
- Computation ~51%
- Communication ~9%
- OpenMP sync ~13%
- ~86% of computation spent in two OpenMP parallel regions
- Workload is not equally distributed

PENNANT: communication breakdown



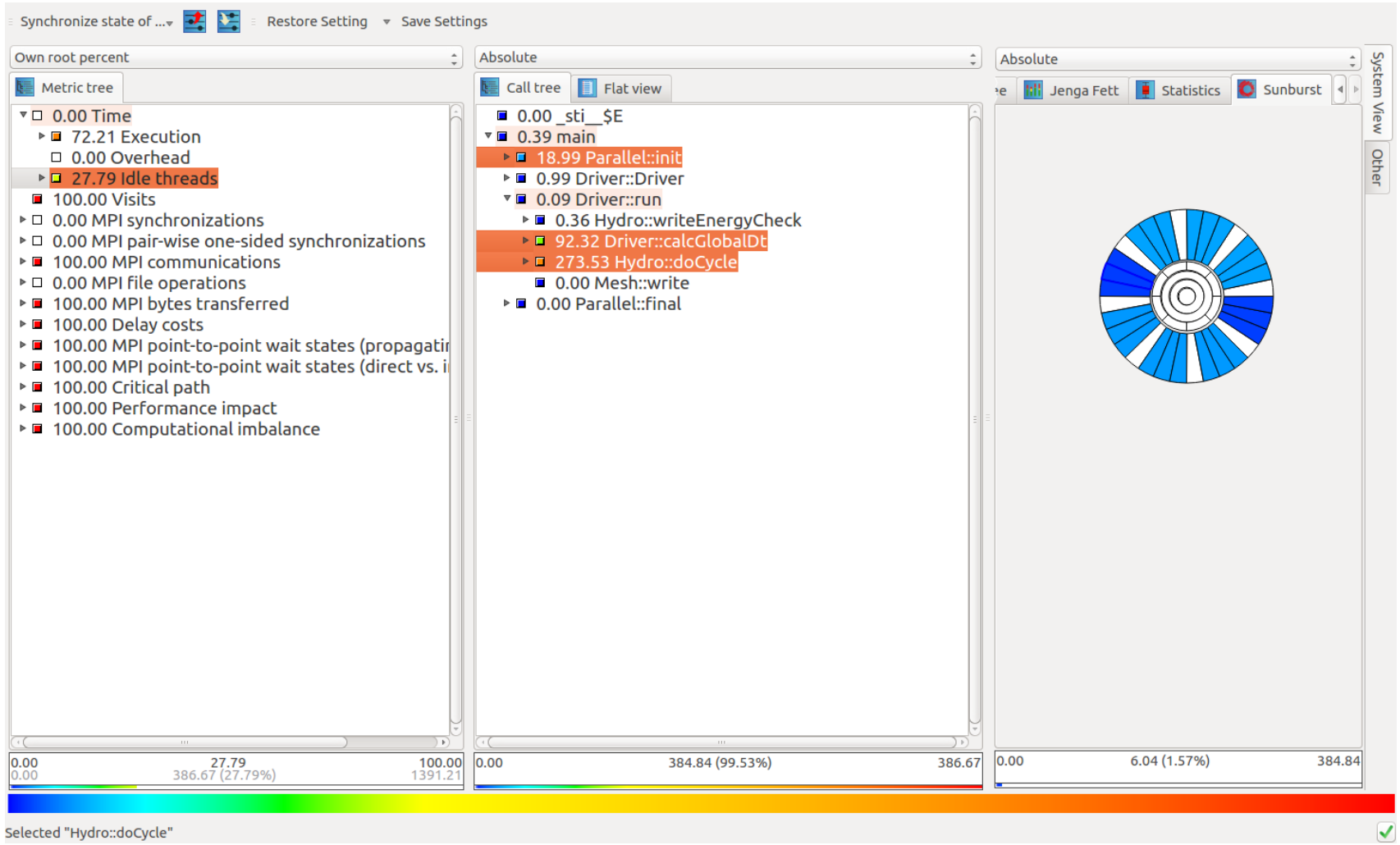
- Late Sender ~6% in two routines *parallelGather* and *parallelScatter*
- Ranks 3, 5 and 7 have significantly smaller waiting time than others

PENNANT: OpenMP sync



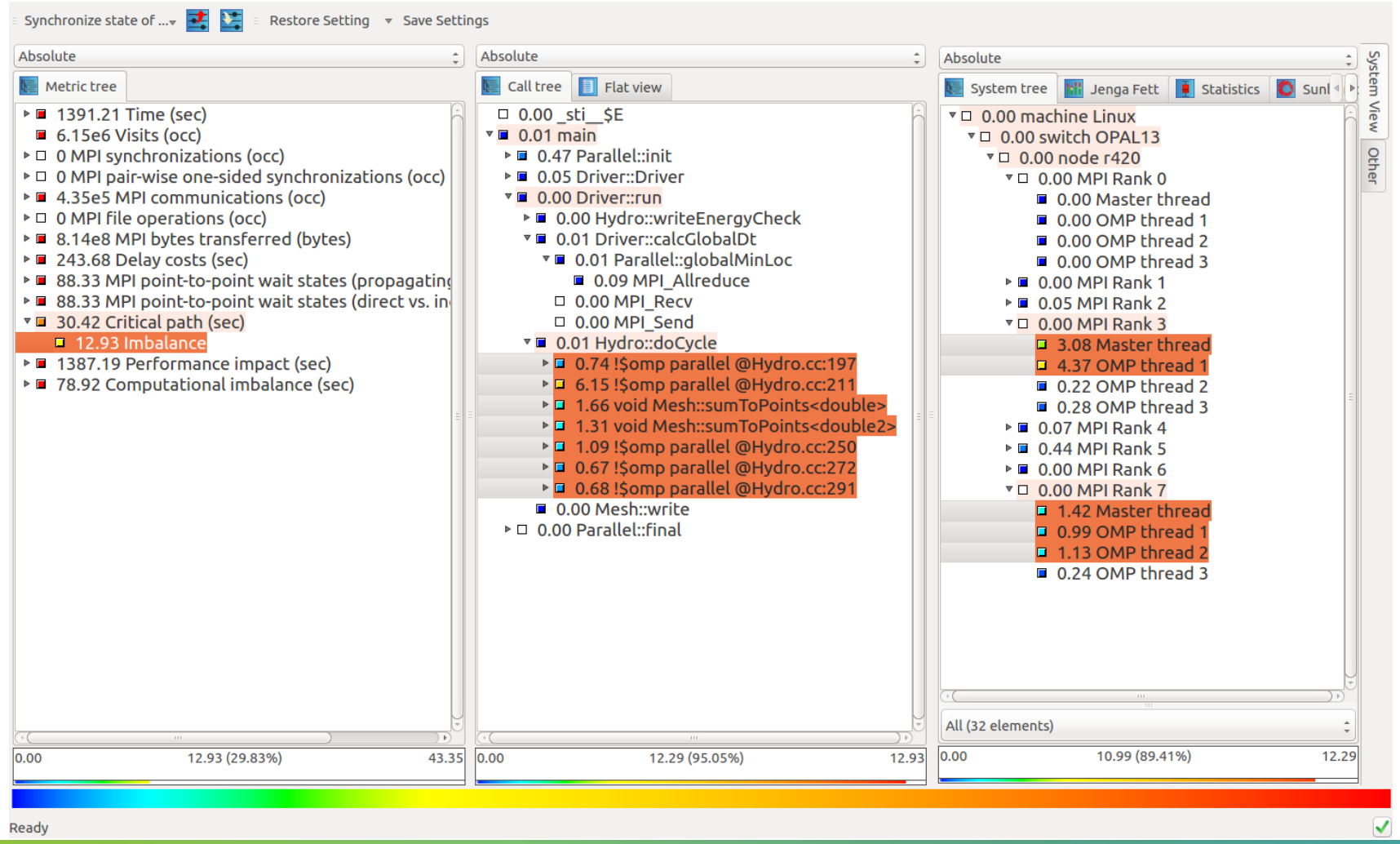
- OpenMP sync is spent in OpenMP implicit barriers
- Significant time on OpenMP barriers spent on rank 3 and 7
- ~86% of computation spent in two OpenMP parallel regions
- Workload is not equally distributed

PENNANT: idle threads



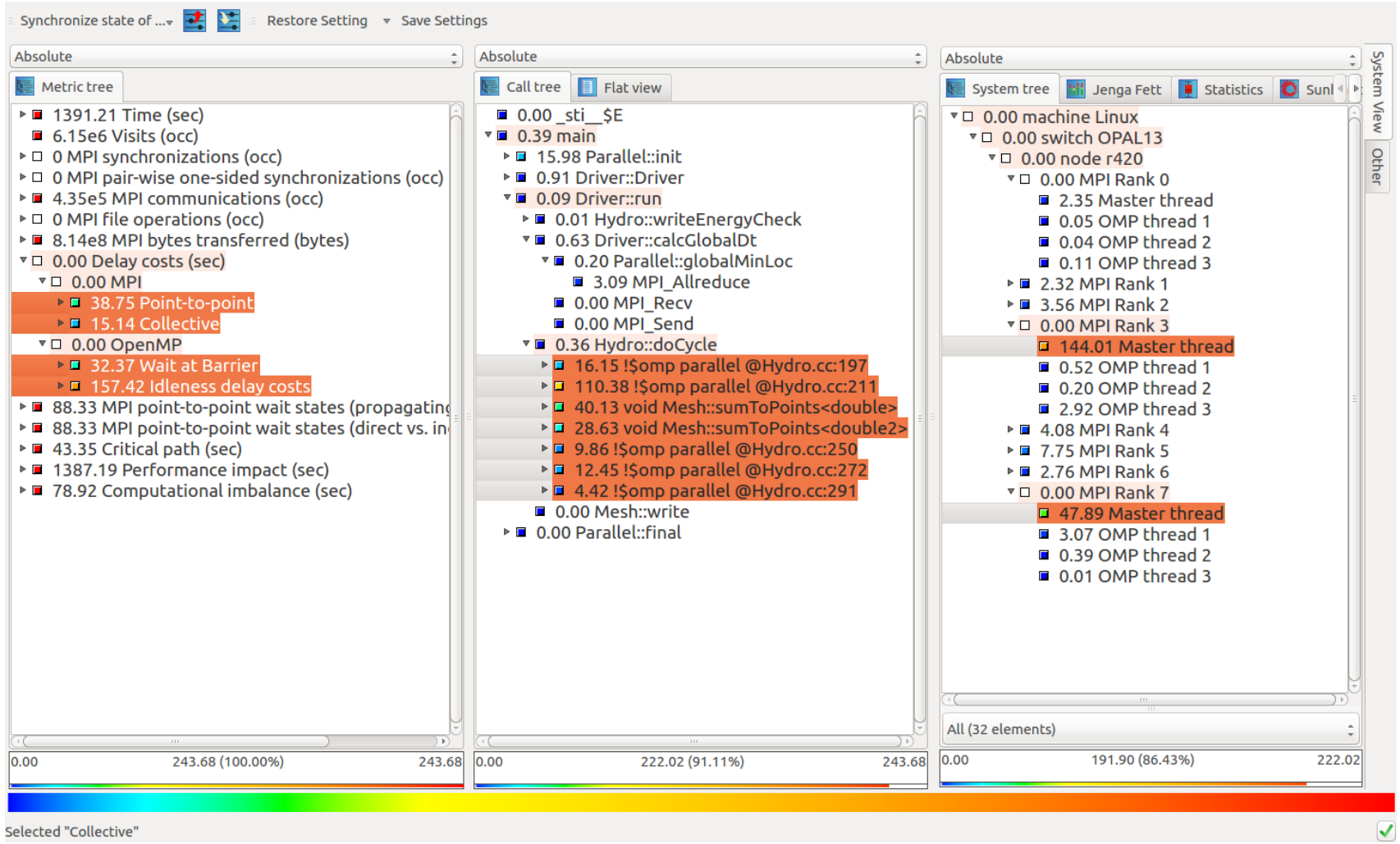
- ~28% of total runtime OpenMP threads are idling in three routines, i.e. *Parallel:init*, *Driver::calcGlobalDt*, *Hydro::doCycle*

PENNANT: critical path



- Selected routines of MPI ranks 3 and 5 show significant impact on critical path -> potential candidates for optimization

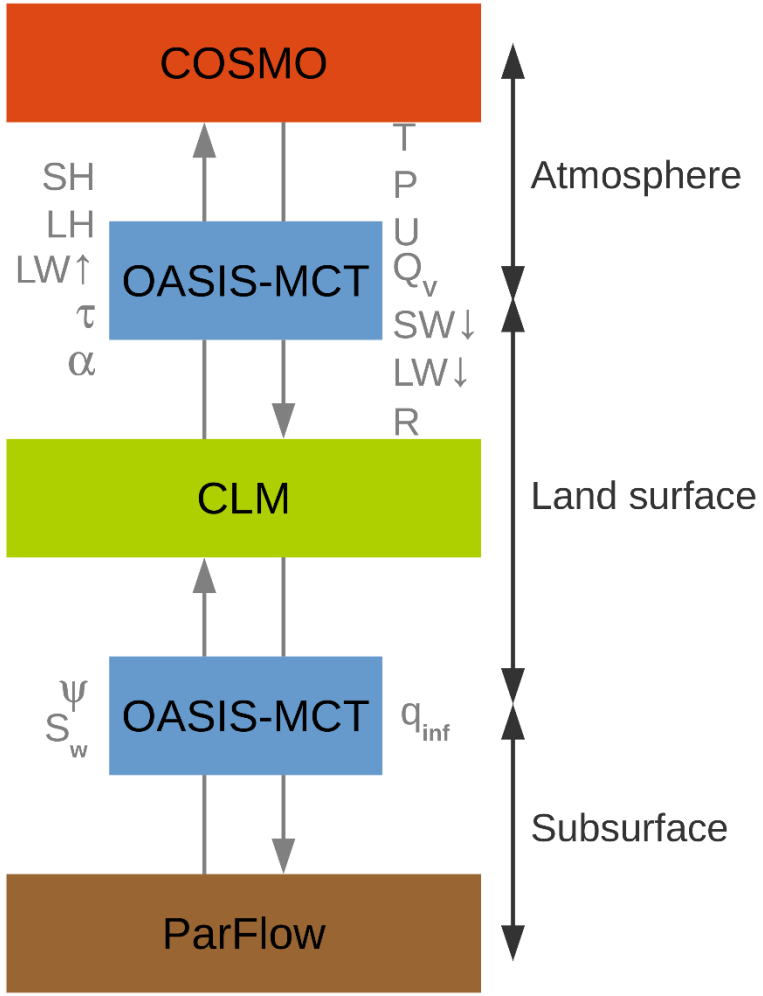
PENNANT: delay analysis



- Most of the delay caused by imbalanced ranks 3 and 5
- Consider decomposition scheme where load of ranks is balanced

Terrestrial System Modeling Platform (TerrSysMP)

- **TerrSysMP** simulates the interaction between lateral flow processes in river basins with the lower atmospheric boundary layer
- **MPMD:** Multiple Program Multiple Data Execution Model
- Consists of three model components: **COSMO**, **CLM** , **ParFlow** and an external MPI-based coupler **OASIS3** that drives the system
- Developed by Transregional Collaborative Research Center 32
- http://www.terrsysmp.org



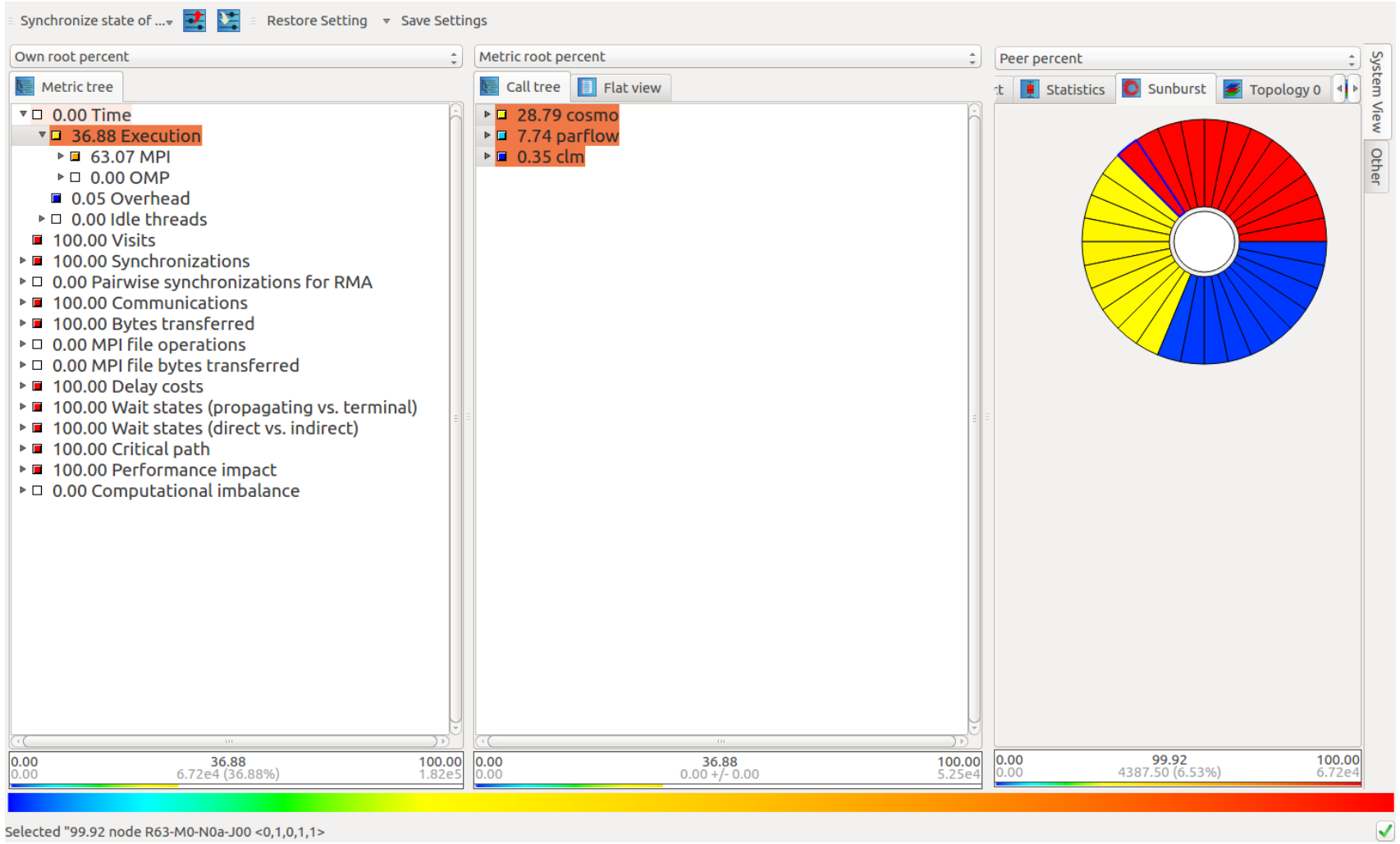
- Testcase executed on JUQUEEN (512 MPI ranks)
- Copy **terrsysmp_mpmd_juqueen.cubex** to your laptop

```
https://fz-juelich.sciebo.de/s/JRFcwYZcaTRQ2sD
```

- Examine measurement with CUBE

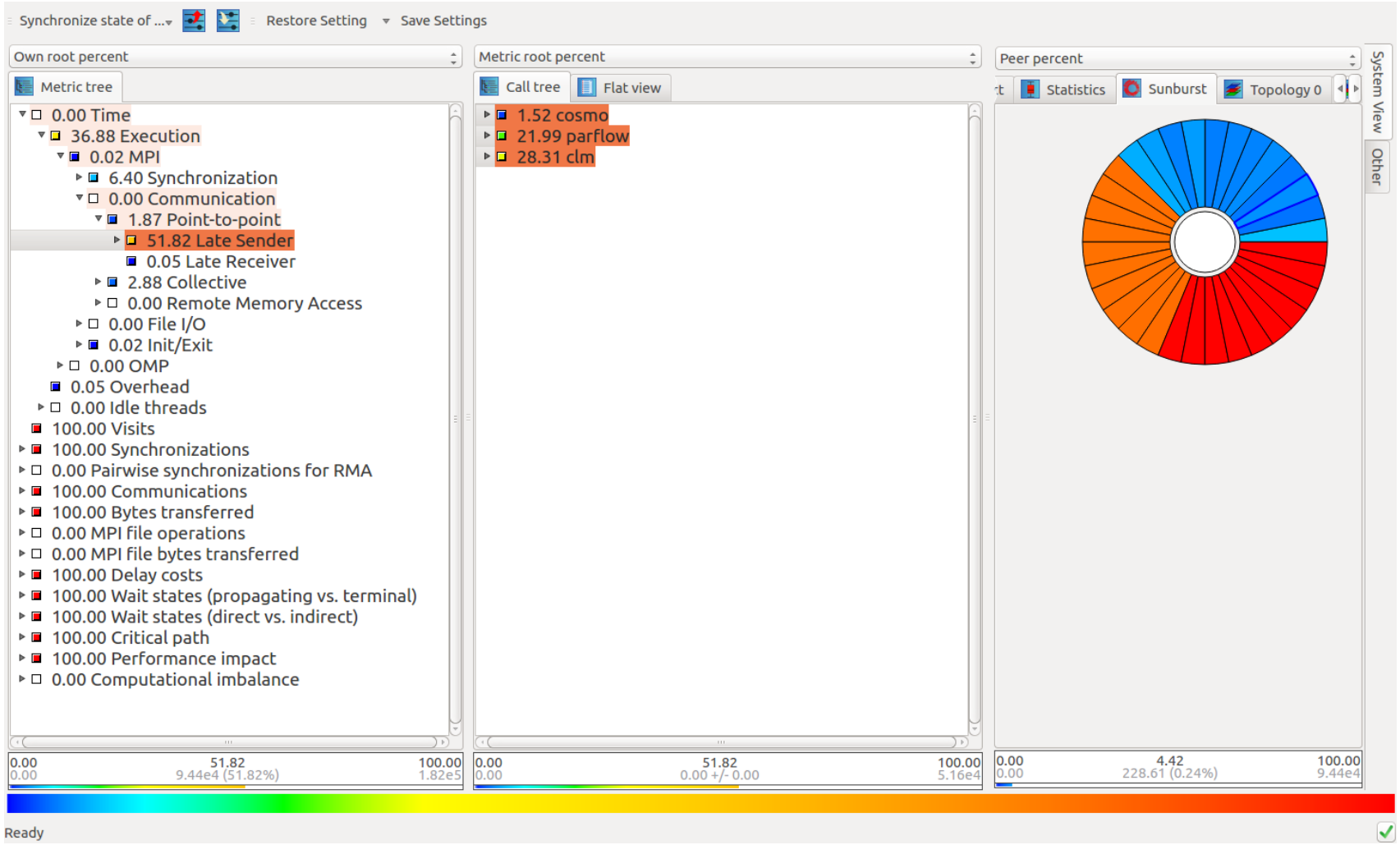
```
laptop> cube terrsysmp_mpmd_juqueen.cubex
```

TerrSysMP: execution breakdown



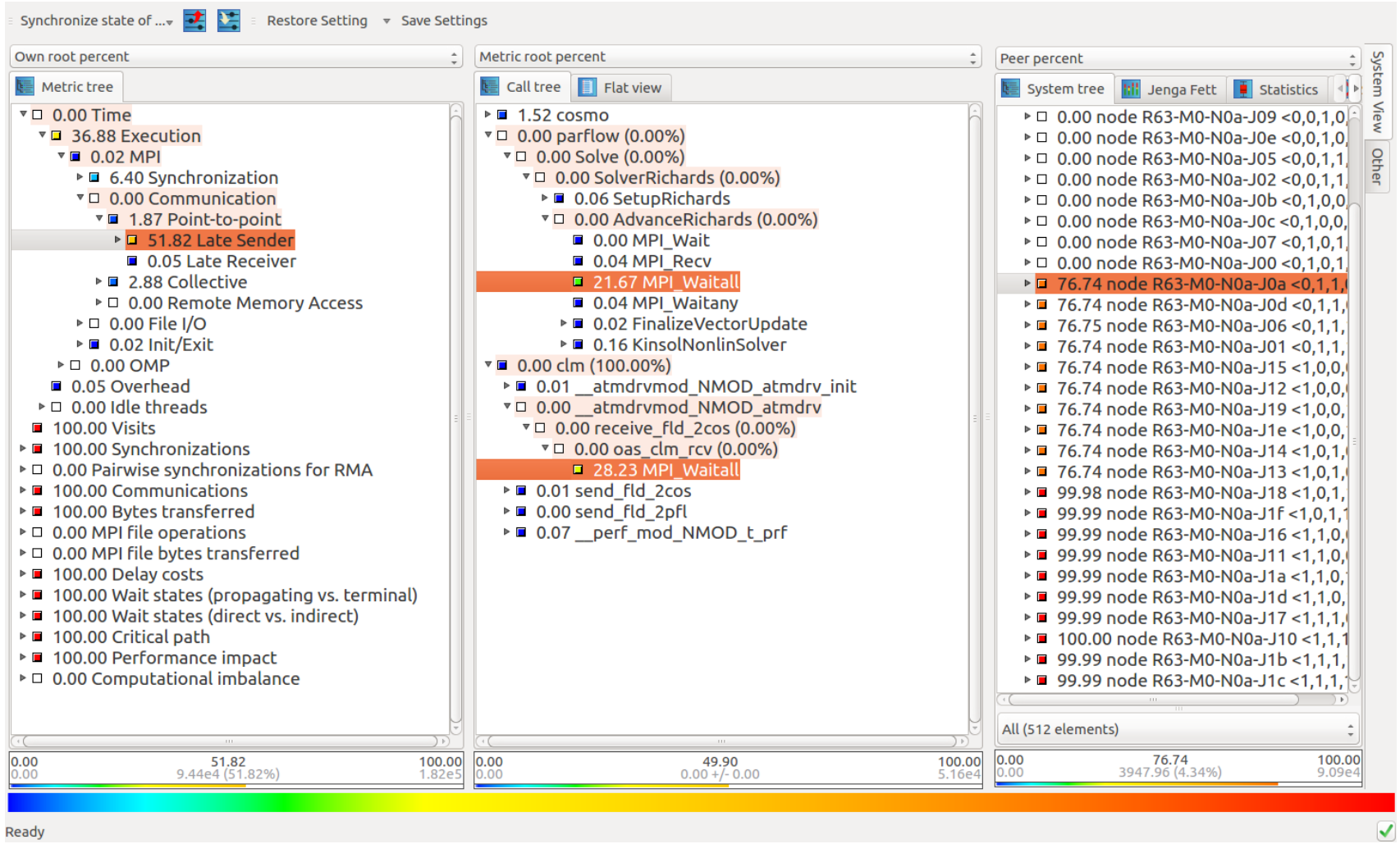
- COSMO 12 nodes (red)
- CLM 10 nodes (blue)
- ParFlow 10 nodes (yellow)
- ~37% in computation and ~63% in MPI

TerrSysMP: communication breakdown



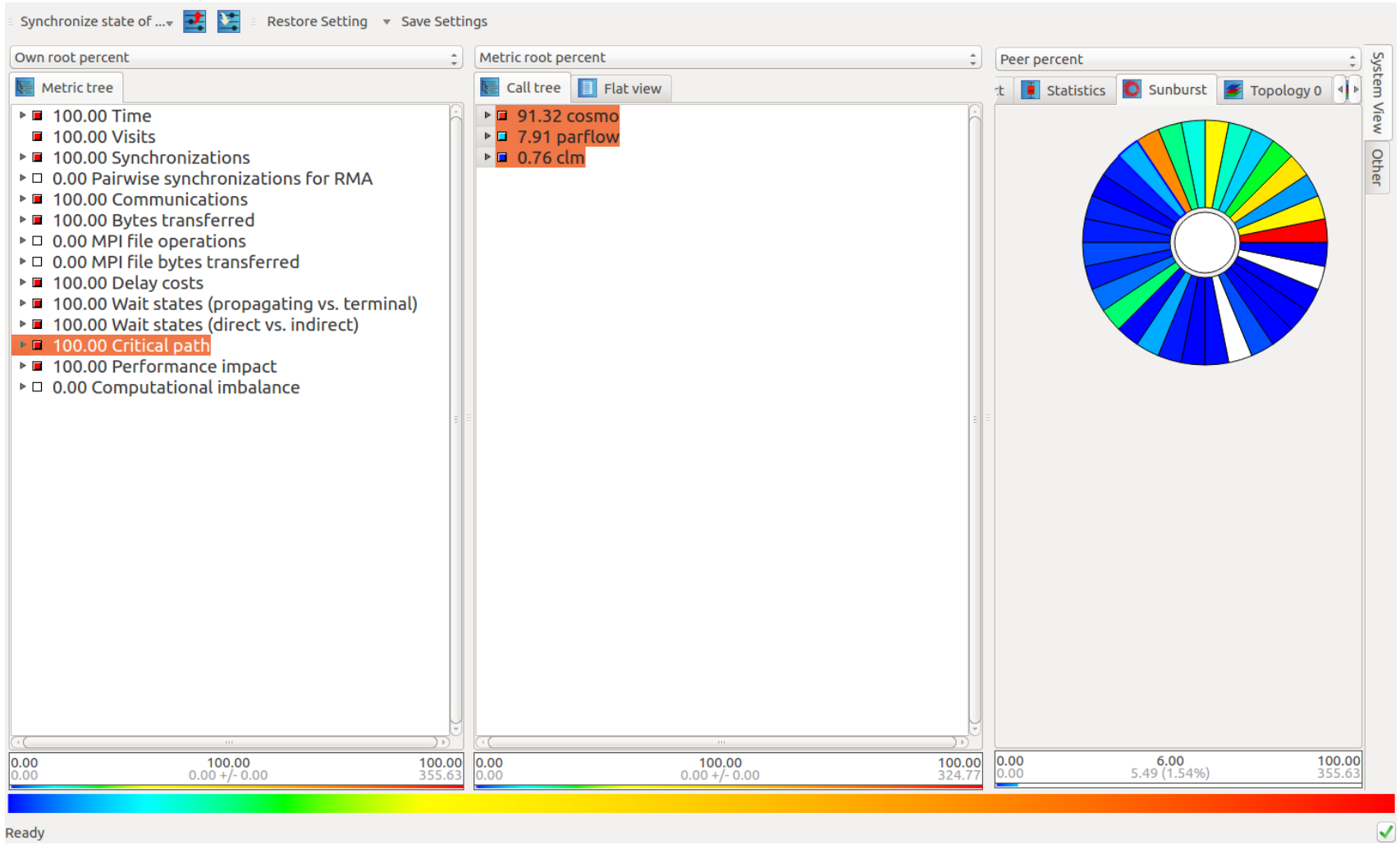
- Late Sender:
  - COSMO ~2%
  - CLM ~28%
  - ParFow ~22%

TerrSysMP: communication breakdown (cont)



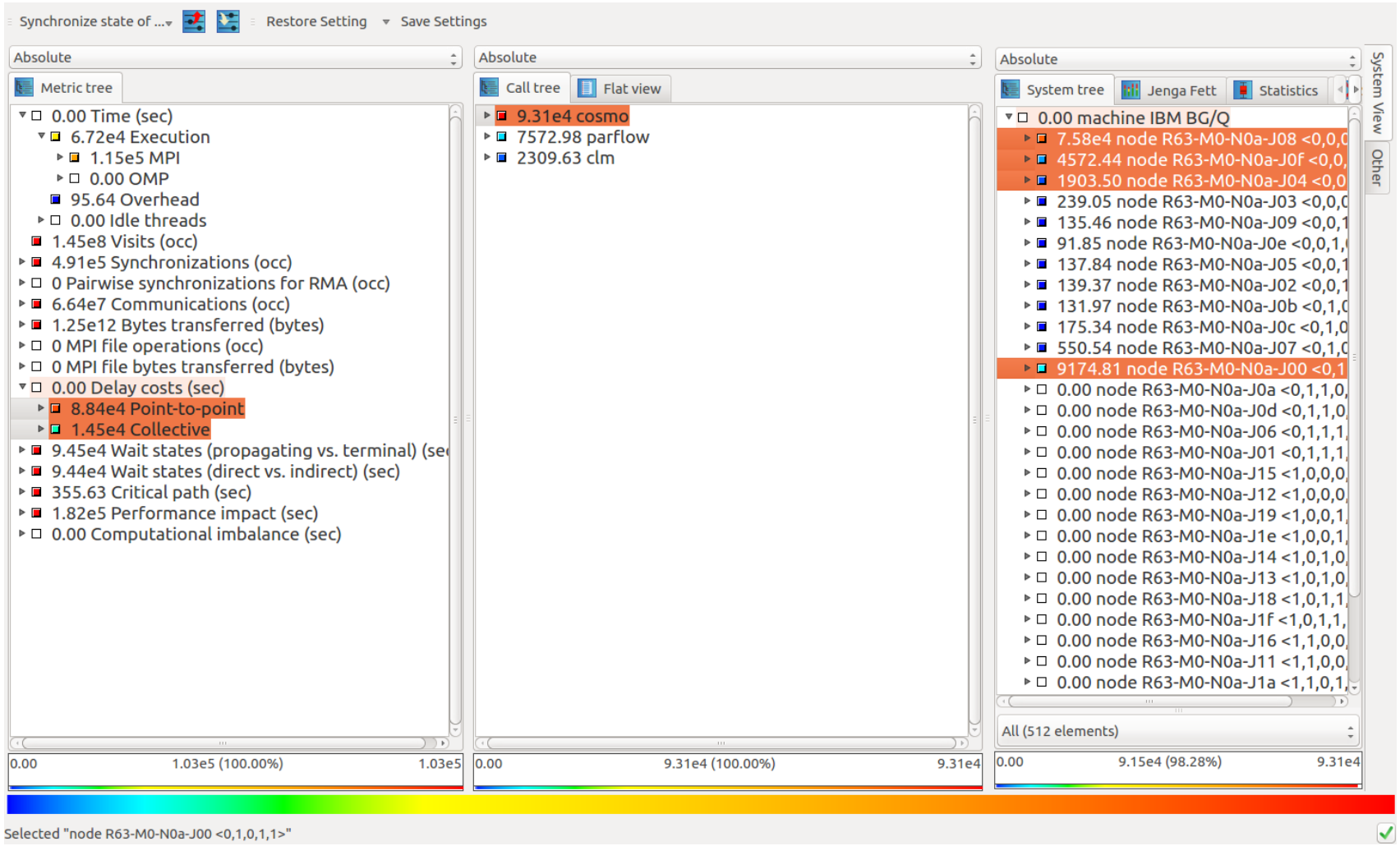
- Late Sender:
  - ~50% in MPI_Waitall

TerrSysMP: critical path



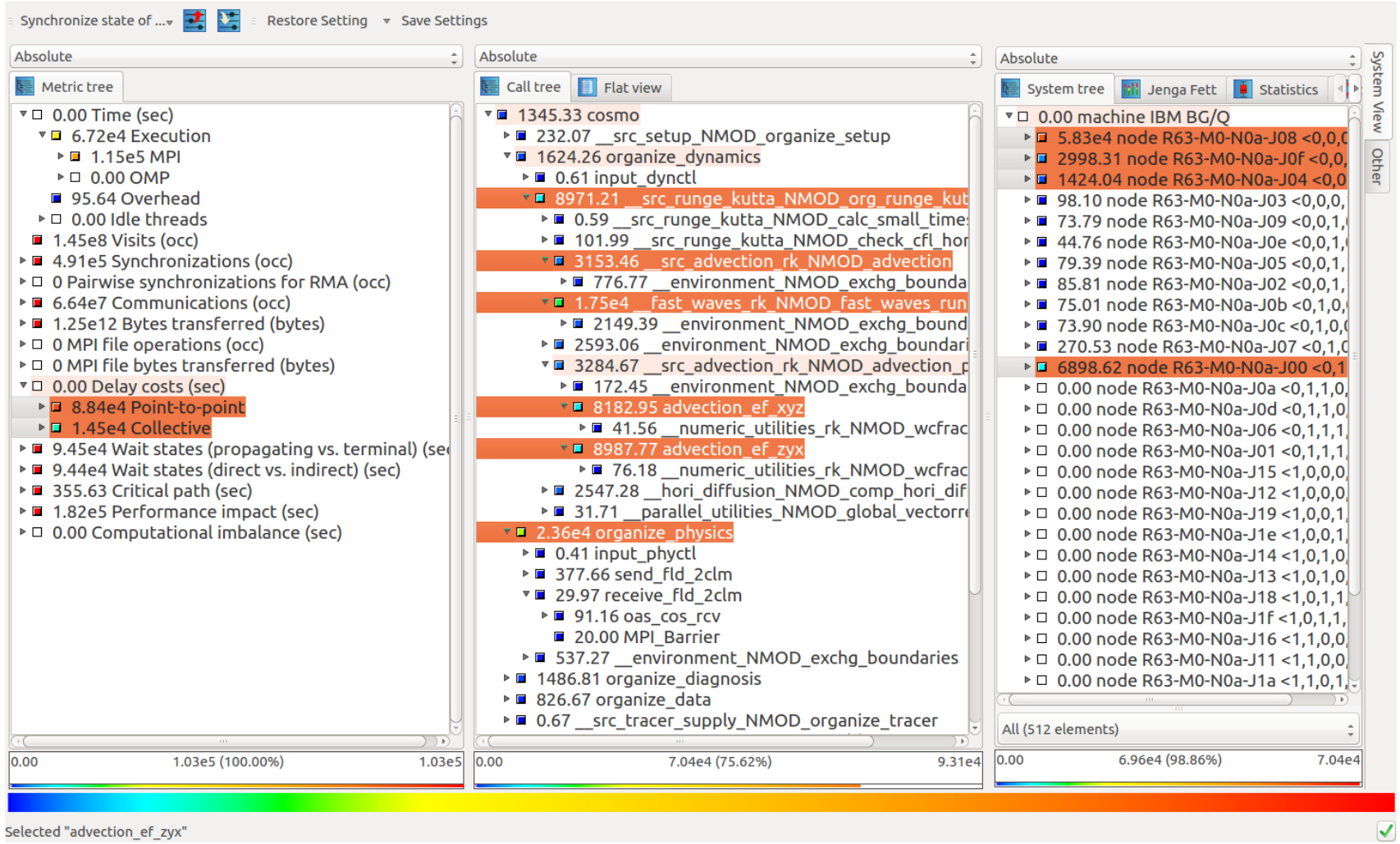
- The lion’s share of the critical path is in COSMO

TerrSysMP: delay analysis



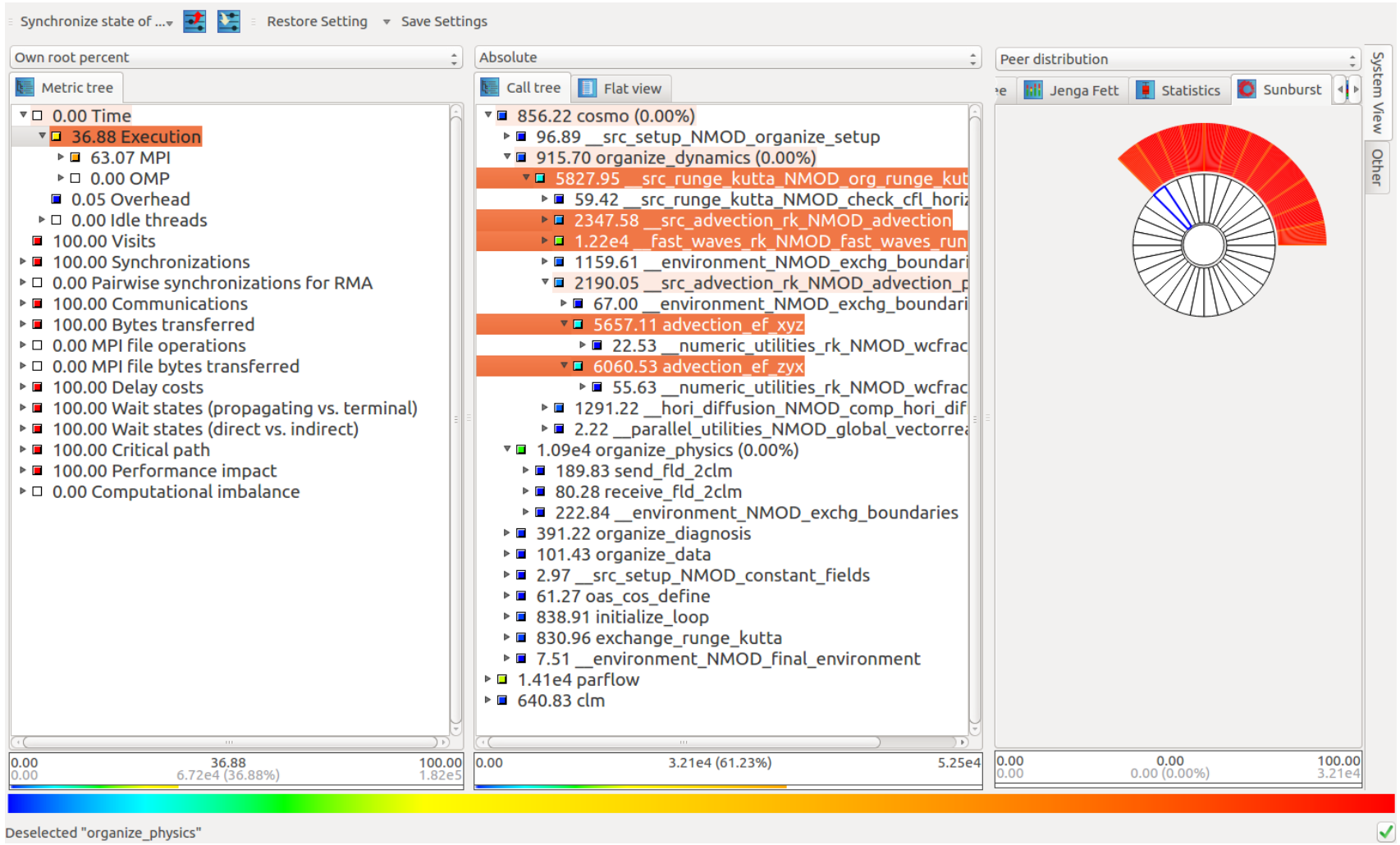
- Most of the delays are caused by 4 nodes in COSMO

TerrSysMP: delay analysis (cont)



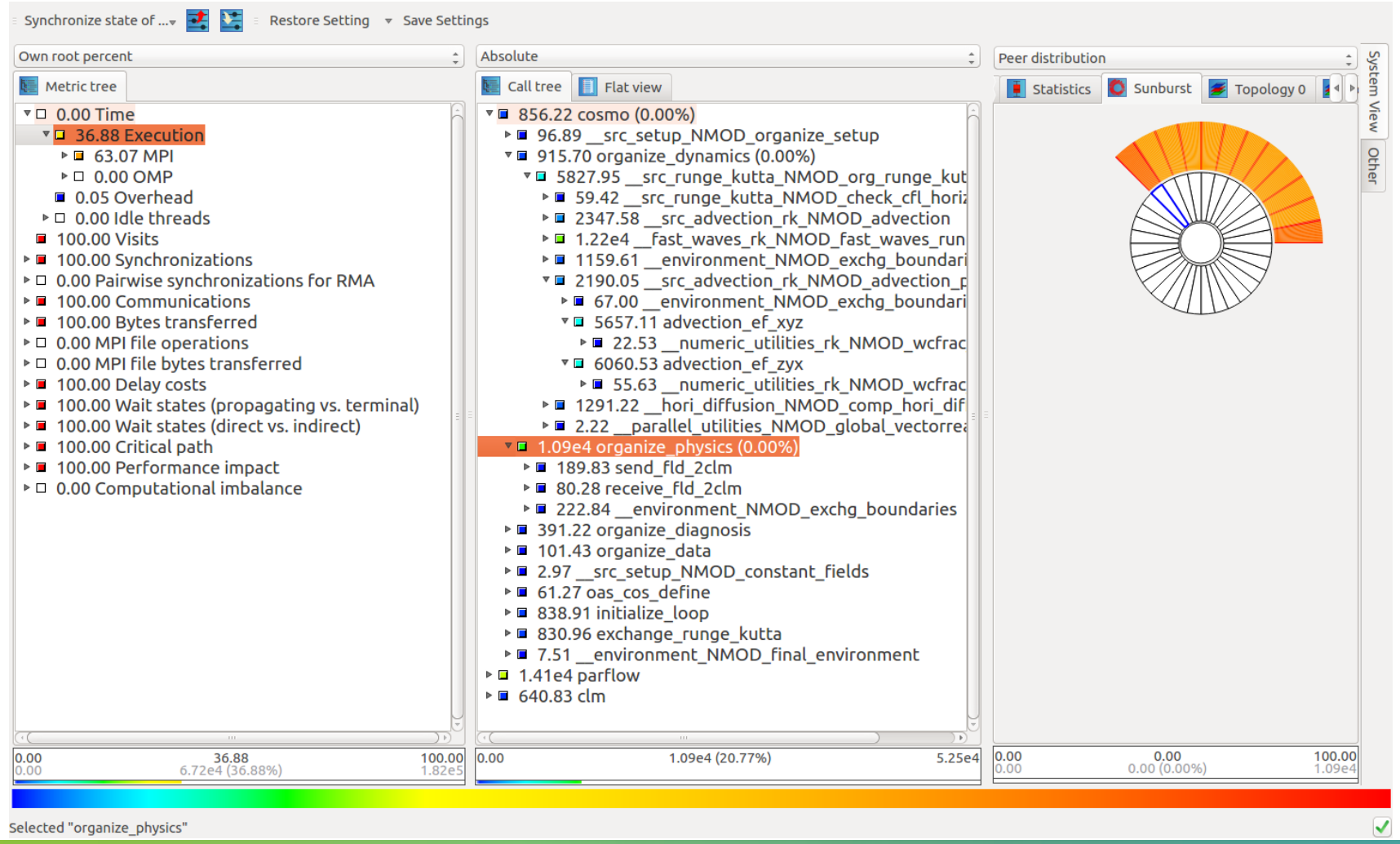
- Most of the delays are caused by 6 routines

COSMO: computational load balance



- The first and the last MPI ranks in COSMO partition have less work in selected routines

COSMO: computational load balance



- The first and the last MPI ranks and nodes in COSMO partition have more work in selected routine
- Better load balance of COSMO component can reduce waiting time and improve overall performance