Automatic trace analysis with the Scalasca Trace Tools

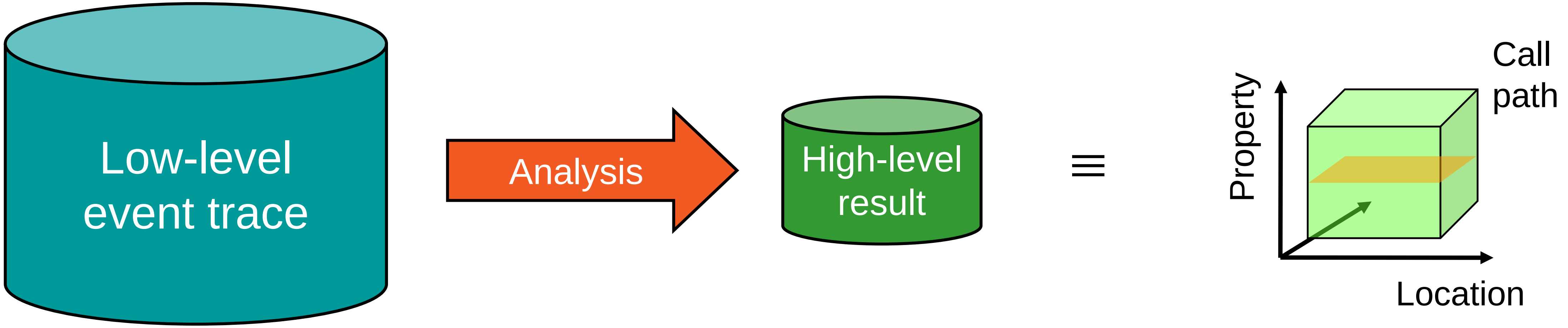
The Scalasca Team
Jülich Supercomputing Centre

trace tools
scalasca

arm
FAU Friedrich-Alexander Universität Erlangen-Nürnberg
Lawrence Livermore National Laboratory
RWTH Aachen University
Technische Universität Dresden
Université de Versailles St-Quentin-en-Yvelines
Universität Stuttgart
BSC Barcelona Supercomputing Center Centro Nacional de Supercomputación
Jülich Forschungszentrum
lrz Leibniz Supercomputing Centre of the Bavarian Academy of Sciences and Humanities
Technische Universität Darmstadt
TUM Technische Universität München
University of Oregon
The University of Tennessee

Automatic trace analysis

- Idea
  - Automatic search for patterns of inefficient behaviour
  - Classification of behaviour & quantification of significance
  - Identification of delays as root causes of inefficiencies



  - Guaranteed to cover the entire event trace
  - Quicker than manual/visual trace analysis
  - Parallel replay analysis exploits available memory & processors to deliver scalability

Scalasca Trace Tools: Objective

- Development of a **scalable trace-based** performance analysis toolset for the most popular parallel programming paradigms
  - Current focus: MPI, OpenMP, and (to a limited extend) POSIX threads

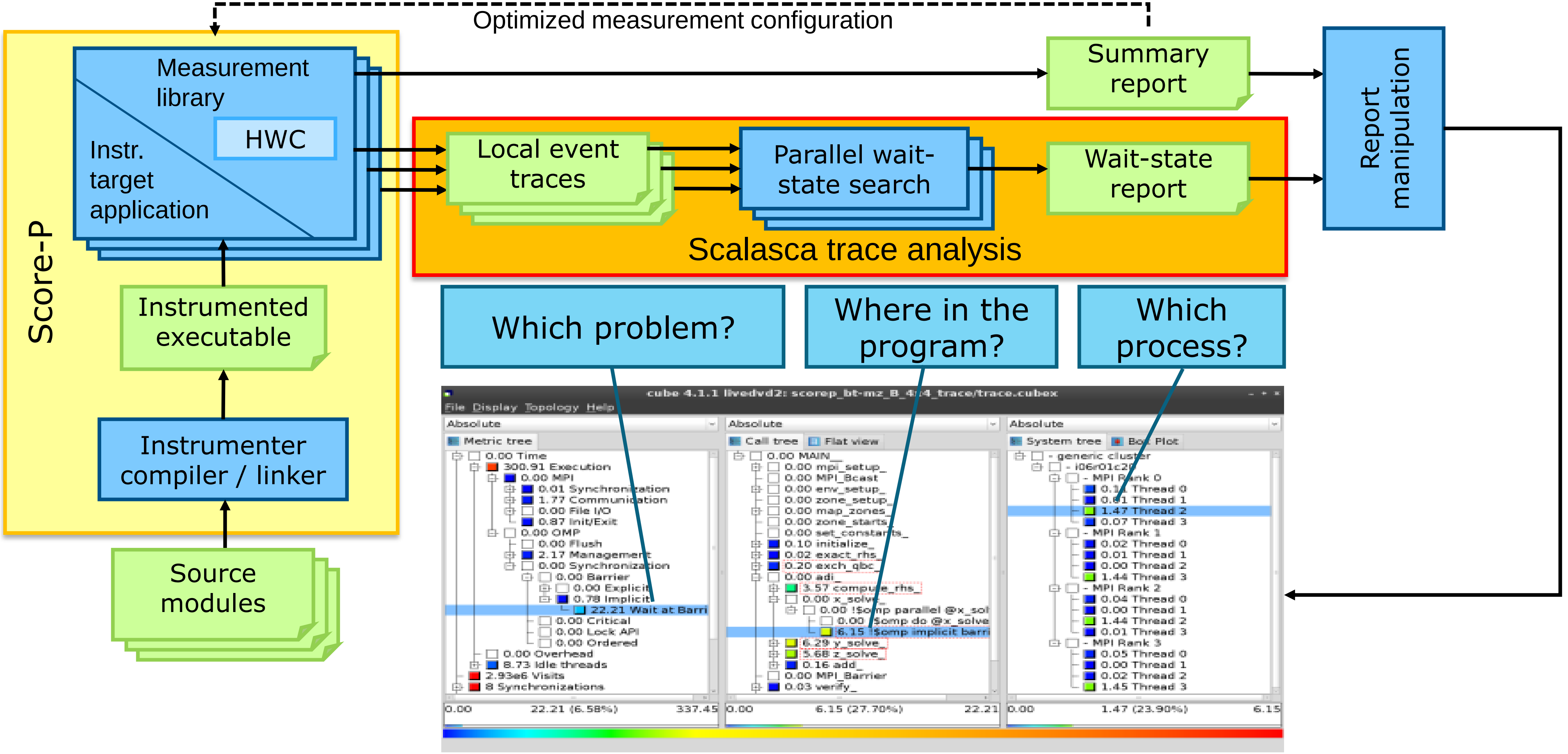
- Specifically targeting large-scale parallel applications
  - Demonstrated scalability up to 1.8 million parallel threads
  - Of course also works at small/medium scale

- Latest release:
  - Scalasca v2.6.1 (December 2022) coordinated with Score-P v8.1

Scalasca Trace Tools: Features

- Open source, 3-clause BSD license
- Fairly portable
  - HPC/Cray XT/XE/XK/XC/EX, IBM Blue Gene, SGI Altix, Fujitsu FX systems, Linux clusters (x86, Power, ARM), Intel Xeon Phi, ...
- Uses Score-P instrumenter & measurement libraries
  - Scalasca v2 core package focuses on trace-based analyses
  - Supports common data formats
    - Reads event traces in OTF2 format
    - Writes analysis reports in CUBE4 format
- Current limitations:
  - Unable to handle traces
    - with MPI thread level exceeding MPI_THREAD_FUNNELED
    - containing Memory events, CUDA/OpenCL device events (kernel, memcpy), SHMEM, or OpenMP nested parallelism
  - PAPI/rusage metrics for trace events are ignored

Scalasca workflow

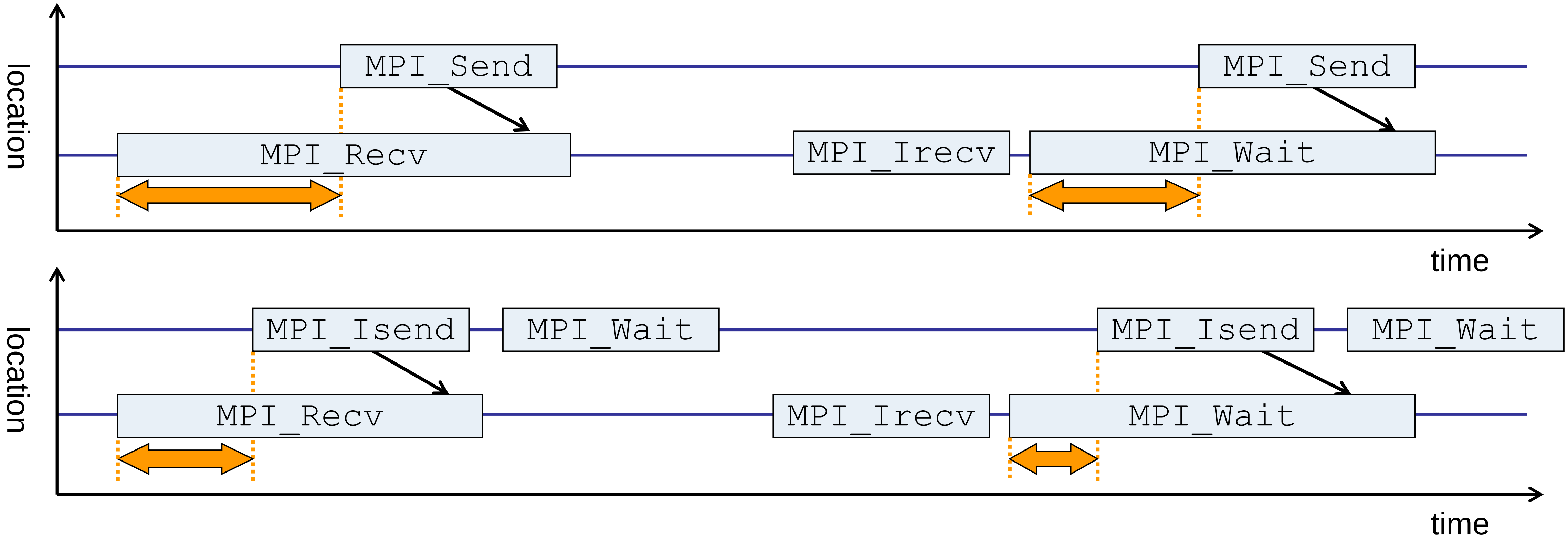


```
0.00 Time
  300.91 Execution
    0.00 MPI
      0.01 Synchronization
      1.77 Communication
      0.00 File I/O
      0.87 Init/Exit
    0.00 OMP
      0.00 Flush
      2.17 Management
      0.00 Synchronization
        0.00 Barrier
          0.00 Explicit
          0.78 Implicit
            22.21 Wait at Barri
        0.00 Critical
        0.00 Lock API
        0.00 Ordered
    0.00 Overhead
    8.73 Idle threads
2.93e6 Visits
8 Synchronizations
0.00          22.21 (6.58%)          337.45
```

```
0.00 MAIN__
  0.00 mpi_setup_
  0.00 MPI_Bcast
  0.00 env_setup_
  0.00 zone_setup_
  0.00 map_zones_
  0.00 zone_starts_
  0.00 set_constants_
  0.10 initialize_
  0.02 exact_rhs_
  0.20 exch_qbc_
  0.00 adi_
    3.57 compute_rhs_
    0.00 x_solve_
      0.00 !$omp parallel @x_sol
        0.00 !$omp do @x_solve
        6.15 !$omp implicit barri
    6.29 y_solve_
    5.68 z_solve_
    0.16 add_
  0.00 MPI_Barrier
  0.03 verify_
0.00          6.15 (27.70%)          22.21
```

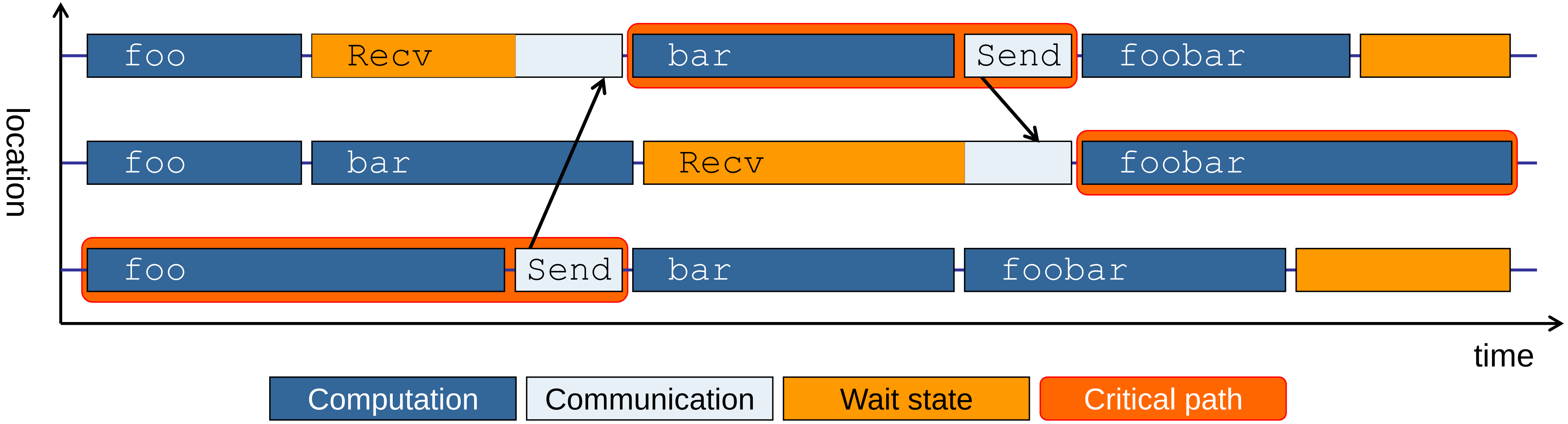
```
- generic cluster
  - i06r01c20
    - MPI Rank 0
      0.11 Thread 0
      0.01 Thread 1
      1.47 Thread 2
      0.07 Thread 3
    - MPI Rank 1
      0.02 Thread 0
      0.01 Thread 1
      0.00 Thread 2
      1.44 Thread 3
    - MPI Rank 2
      0.04 Thread 0
      0.00 Thread 1
      1.44 Thread 2
      0.01 Thread 3
    - MPI Rank 3
      0.05 Thread 0
      0.00 Thread 1
      0.02 Thread 2
      1.45 Thread 3
0.00          1.47 (23.90%)          6.15
```

Example: "*Late Sender*" wait state



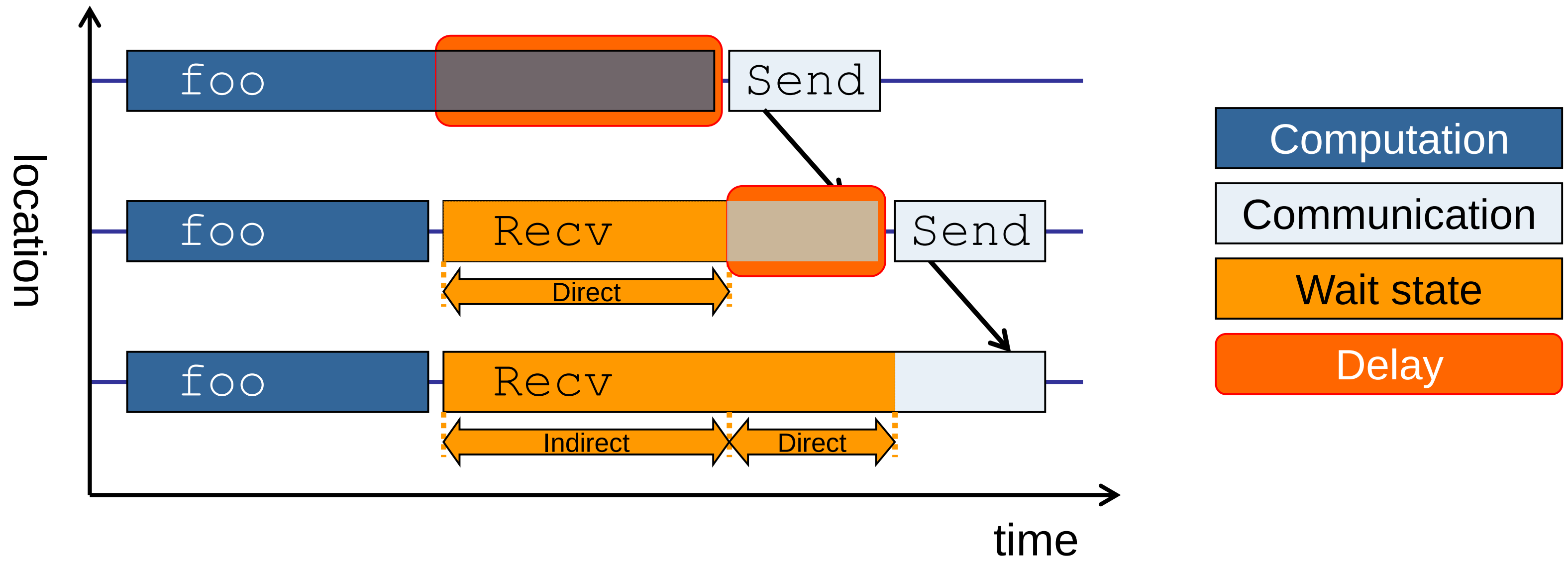
- Waiting time caused by a blocking receive operation posted earlier than the corresponding send
- Applies to blocking as well as non-blocking communication

Example: Critical path



- Shows call paths and processes/threads that are responsible for the program’s wall-clock runtime
- Identifies good optimization candidates and parallelization bottlenecks

Example: Root-cause analysis



- Classifies wait states into direct and indirect (i.e., caused by other wait states)
- Identifies *delays* (excess computation/communication) as root causes of wait states
- Attributes wait states as *delay costs*

Hands-on:
NPB-MZ-MPI / BT

trace tools
scalasca

Scalasca command – One command for (almost) everything

```
% scalasca
Scalasca 2.6
Toolset for scalable performance analysis of large-scale parallel applications
usage: scalasca [OPTION]... ACTION <argument>...
    1. prepare application objects and executable for measurement:
       scalasca -instrument <compile-or-link-command> # skin (using scorep)
    2. run application under control of measurement system:
       scalasca -analyze <application-launch-command> # scan
    3. interactively explore measurement analysis report:
       scalasca -examine <experiment-archive|report>  # square

Options:
   -c, --show-config     show configuration summary and exit
   -h, --help            show this help and exit
   -n, --dry-run         show actions without taking them
       --quickref        show quick reference guide and exit
       --remap-specfile  show path to remapper specification file and exit
   -v, --verbose         enable verbose commentary
   -V, --version         show version information and exit
```

- The '`scalasca -instrument`' command is deprecated and only provided for backwards compatibility with Scalasca 1.x., recommended: use Score-P instrumenter directly

Scalasca convenience command: scan / scalasca -analyze

```
% scan
Scalasca 2.6: measurement collection & analysis nexus
usage: scan {options} [launchcmd [launchargs]] target [targetargs]
      where {options} may include:
  -h    Help      : show this brief usage message and exit.
  -v    Verbose   : increase verbosity.
  -n    Preview   : show command(s) to be launched but don't execute.
  -q    Quiescent : execution with neither summarization nor tracing.
  -s    Summary   : enable runtime summarization. [Default]
  -t    Tracing   : enable trace collection and analysis.
  -a    Analyze   : skip measurement to (re-)analyze an existing trace.
  -e exptdir      : Experiment archive to generate and/or analyze.
                    (overrides default experiment archive title)
  -f filtfile     : File specifying measurement filter.
  -l lockfile     : File that blocks start of measurement.
  -R #runs        : Specify the number of measurement runs per config.
  -M cfgfile      : Specify a config file for a multi-run measurement.
```

- Scalasca measurement collection & analysis nexus

Automatic measurement configuration

- scan configures Score-P measurement by automatically setting some environment variables and exporting them
  - E.g., experiment title, profiling/tracing mode, filter file, …
  - Precedence order:
    - Command-line arguments
    - Environment variables already set
    - Automatically determined values
- Also, scan includes consistency checks and prevents corrupting existing experiment directories
- For tracing experiments, after trace collection completes then automatic parallel trace analysis is initiated
  - Uses identical launch configuration to that used for measurement (i.e., the same allocated compute resources)

Scalasca convenience command: square / scalasca -examine

```
% square
Scalasca 2.6: analysis report explorer
usage: square [OPTIONS] <experiment archive | cube file>
   -c <none | quick | full> : Level of sanity checks for newly created reports
   -F                       : Force remapping of already existing reports
   -f filtfile              : Use specified filter file when doing scoring (-s)
   -s                       : Skip display and output textual score report
   -v                       : Enable verbose mode
   -n                       : Do not include idle thread metric
   -S <mean | merge>        : Aggregation method for summarization results of
                              each configuration (default: merge)
   -T <mean | merge>        : Aggregation method for trace analysis results of
                              each configuration (default: merge)
   -A                       : Post-process every step of a multi-run experiment
```

- Scalasca analysis report explorer (Cube)

Recap: Local installation (Archer2)

- Setup access to Scalasca and associated tools, accessible via “other-software”
  - Required for each shell session
  - Score-P and Scalasca installations are toolchain specific (GCC11 default)

```
% module swap PrgEnv-cray PrgEnv-gnu
% module load load-epcc-module other-software
% module load scalasca/2.6.1-gcc11
```

PrgEnv-cray
PrgEnv-aocc
PrgEnv-gnu

  - Check `module avail scalasca` for alternate Score-P/Scalasca modules available

- Copy tutorial sources to your personal workspace (if not already done)

```
% cd /work/ta115/ta115/$USER
% tar zxvf /work/y23/shared/tutorial/NPB3.4-MZ-MPI.tar.gz
% cd NPB3.4-MZ-MPI
```

BT-MZ summary measurement collection...

```
% cd bin.scorep
% cp ../jobscript/archer2/scalasca.sbatch .
% cat scalasca.sbatch

# Scalasca nexus configuration for profiling / summarization
#NEXUS="scalasca -analyze -s"
# Scalasca nexus configuration for trace collection & analysis
#NEXUS="scalasca -analyze -t"

# Score-P measurement configuration
export SCOREP_FILTERING_FILE=../config/scorep.filt
#export SCOREP_TOTAL_MEMORY=100M


# run the application
scalasca -analyze  srun ./bt-mz_C.x
```

```
% sbatch  scalasca.sbatch
```

- Change to directory holding the Score-P instrumented executable and edit the job script

Hint:
`scan` = `scalasca -analyze`
`-s` = profile/summary (def)

- Submit the job

BT-MZ summary measurement

```
S=C=A=N: Scalasca 2.6.1 runtime summarization
S=C=A=N: ./scorep_bt-mz_C_8x6_sum experiment archive
S=C=A=N: Thu Jun  10 11:48:50 2021: Collect start
srun  ./bt-mz_C.x

 NAS Parallel Benchmarks (NPB3.4-MZ MPI+OpenMP) -
    BT-MZ Benchmark

 Number of zones:  8 x  8
 Iterations: 200    dt:   0.000100
 Number of active processes:     8

 [... More application output ...]

S=C=A=N: Thu Jun  10 11:49:02 2021: Collect done (status=0) 12s
S=C=A=N: ./scorep_bt-mz_C_8x6_sum complete.
```

- Run the application using the Scalasca measurement collection & analysis nexus prefixed to launch command
- Creates experiment directory: scorep_bt-mz_C_8x6_sum

BT-MZ summary analysis report examination

- Score summary analysis report

```
% square -s  scorep_bt-mz_C_8x6_sum
INFO: Post-processing runtime summarization report (profile.cubex)...
INFO: Score report written to ./scorep_bt-mz_C_8x6_sum/scorep.score
```

- Post-processing and interactive exploration with Cube

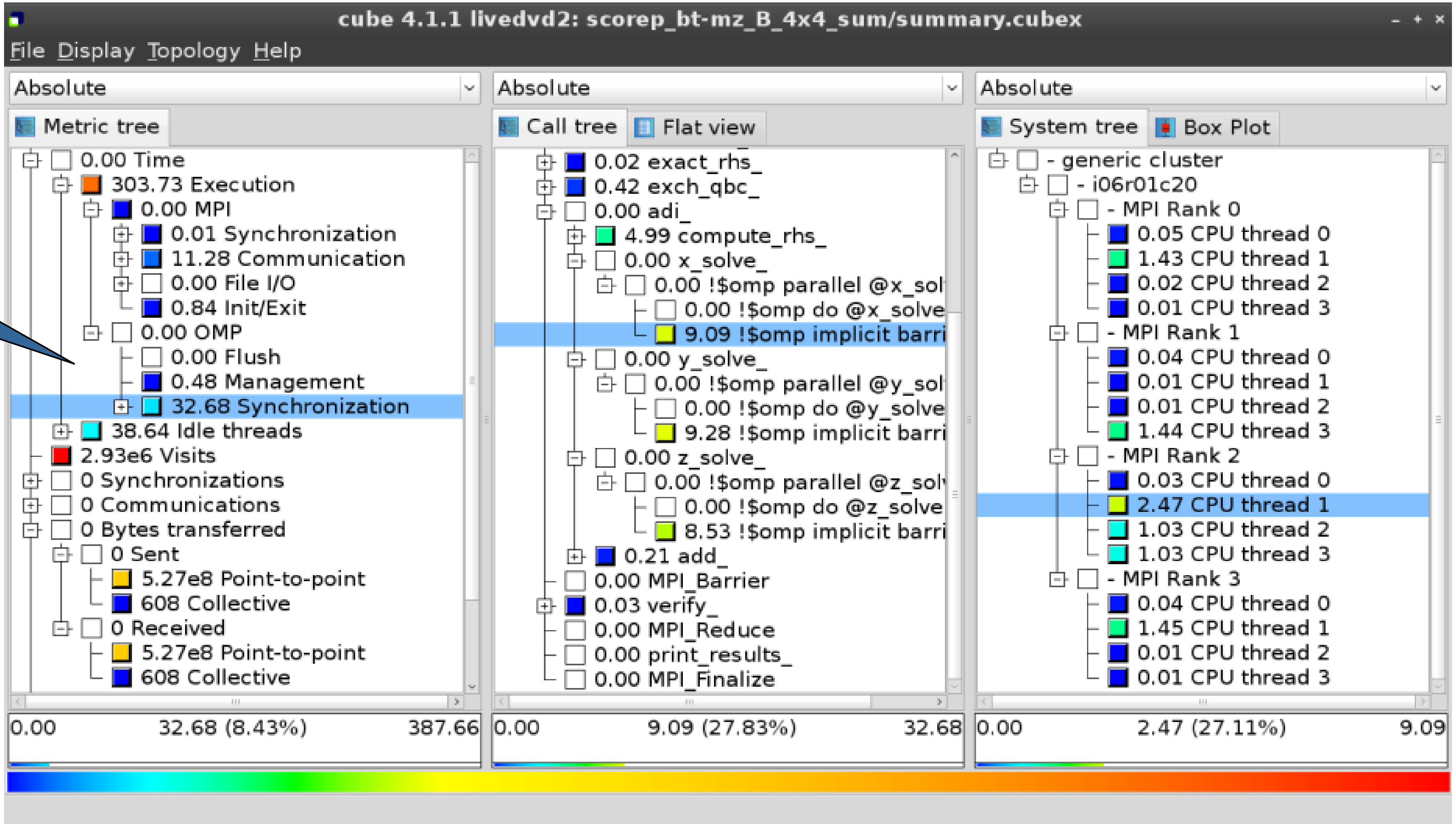
```
% square  scorep_bt-mz_C_8x6_sum
INFO: Displaying ./scorep_bt-mz_C_8x6_sum/summary.cubex...

                    [GUI showing summary analysis report]
```

Hint:
Copy ‘summary.cubex’ to local system (laptop) using ‘scp’ to improve responsiveness of GUI

- The post-processing derives additional metrics from the basic ones and generates a structured metric hierarchy

Post-processed summary analysis report

Split base metrics into more specific metrics



cube 4.1.1 livedvd2: scorep_bt-mz_B_4x4_sum/summary.cubex

```
0.00 Time
  303.73 Execution
    0.00 MPI
      0.01 Synchronization
      11.28 Communication
      0.00 File I/O
      0.84 Init/Exit
    0.00 OMP
      0.00 Flush
      0.48 Management
      32.68 Synchronization
  38.64 Idle threads
2.93e6 Visits
0 Synchronizations
0 Communications
0 Bytes transferred
  0 Sent
    5.27e8 Point-to-point
    608 Collective
  0 Received
    5.27e8 Point-to-point
    608 Collective
```

```
0.02 exact_rhs_
0.42 exch_qbc_
0.00 adi_
  4.99 compute_rhs_
  0.00 x_solve_
    0.00 !$omp parallel @x_sol
      0.00 !$omp do @x_solve
      9.09 !$omp implicit barri
  0.00 y_solve_
    0.00 !$omp parallel @y_sol
      0.00 !$omp do @y_solve
      9.28 !$omp implicit barri
  0.00 z_solve_
    0.00 !$omp parallel @z_sol
      0.00 !$omp do @z_solve
      8.53 !$omp implicit barri
  0.21 add_
0.00 MPI_Barrier
0.03 verify_
0.00 MPI_Reduce
0.00 print_results_
0.00 MPI_Finalize
```

```
- generic cluster
  - i06r01c20
    - MPI Rank 0
      0.05 CPU thread 0
      1.43 CPU thread 1
      0.02 CPU thread 2
      0.01 CPU thread 3
    - MPI Rank 1
      0.04 CPU thread 0
      0.01 CPU thread 1
      0.01 CPU thread 2
      1.44 CPU thread 3
    - MPI Rank 2
      0.03 CPU thread 0
      2.47 CPU thread 1
      1.03 CPU thread 2
      1.03 CPU thread 3
    - MPI Rank 3
      0.04 CPU thread 0
      1.45 CPU thread 1
      0.01 CPU thread 2
      0.01 CPU thread 3
```

```
0.00      32.68 (8.43%)      387.66
0.00       9.09 (27.83%)      32.68
0.00       2.47 (27.11%)       9.09
```

Performance analysis steps

- 0.0 Reference preparation for validation

- 1.0 Program instrumentation
- 1.1 Summary measurement collection
- 1.2 Summary analysis report examination

- 2.0 Summary experiment scoring
- 2.1 Summary measurement collection with filtering
- 2.2 Filtered summary analysis report examination

- 3.0 Event trace collection
- 3.1 Event trace examination & analysis

BT-MZ trace measurement collection...

```
% cd bin.scorep
% cp ../jobscript/archer2/scalasca.sbatch .
% vim scalasca.sbatch

# Scalasca nexus configuration for profiling / summarization
#NEXUS="scalasca -analyze -s"
# Scalasca nexus configuration for trace collection & analysis
#NEXUS="scalasca -analyze -t"

# Score-P measurement configuration
export SCOREP_FILTERING_FILE=../config/scorep.filt
export SCOREP_TOTAL_MEMORY=100M

# run the application
scalasca -analyze -t  srun ./bt-mz_C.8
```

```
% sbatch  scalasca.sbatch
```

- Change to directory with the Score-P instrumented executable and edit the job script

- Add “-t” to the `scan` command
- Submit the job

BT-MZ trace measurement ... collection

```
S=C=A=N: Scalasca 2.6.1 trace collection and analysis
S=C=A=N: ./scorep_bt-mz_C_8x6_trace experiment archive
S=C=A=N: Thu Jun  10 12:05:30 2021: Collect start
srun  ./bt-mz_C.x

 NAS Parallel Benchmarks (NPB3.4-MZ MPI+OMP) - BT-MZ Benchmark

 Number of zones:  16 x  16
 Iterations: 200    dt:   0.000100
 Number of active processes:     8

 [... More application output ...]

S=C=A=N: Thu Jun  10 12:05:44 2021: Collect done (status=0) 14s
```

- Using new experiment directory

  scorep_bt-mz_C_8x6_trace

- Starts measurement with collection of trace files …

BT-MZ trace measurement ... analysis

```
…
S=C=A=N: Thu Jun  10 12:05:44 2021: Analyze start
 srun  scout.hyb --time-correct \
>    ./scorep_bt-mz_C_8x6_trace/traces.otf2

SCOUT   (Scalasca 2.6.1)

Analyzing experiment archive ./scorep_bt-mz_C_8x6_trace/traces.otf2

Opening experiment archive ... done (0.002s).
Reading definition data    ... done (0.004s).
Reading event trace data   ... done (0.113s).
Preprocessing              ... done (0.179s).
Timestamp correction       ... done (0.431s).
Analyzing trace data       ... done (5.174s).
Writing analysis report    ... done (0.175s).

Max. memory usage         : 422.312MB

        # passes        : 1
        # violated      : 0

Total processing time     : 6.140s
S=C=A=N: Thu Jun  10 12:05:51 2021: Analyze done (status=0) 7s
```

- Continues with automatic (parallel) analysis of trace files

BT-MZ trace analysis report exploration

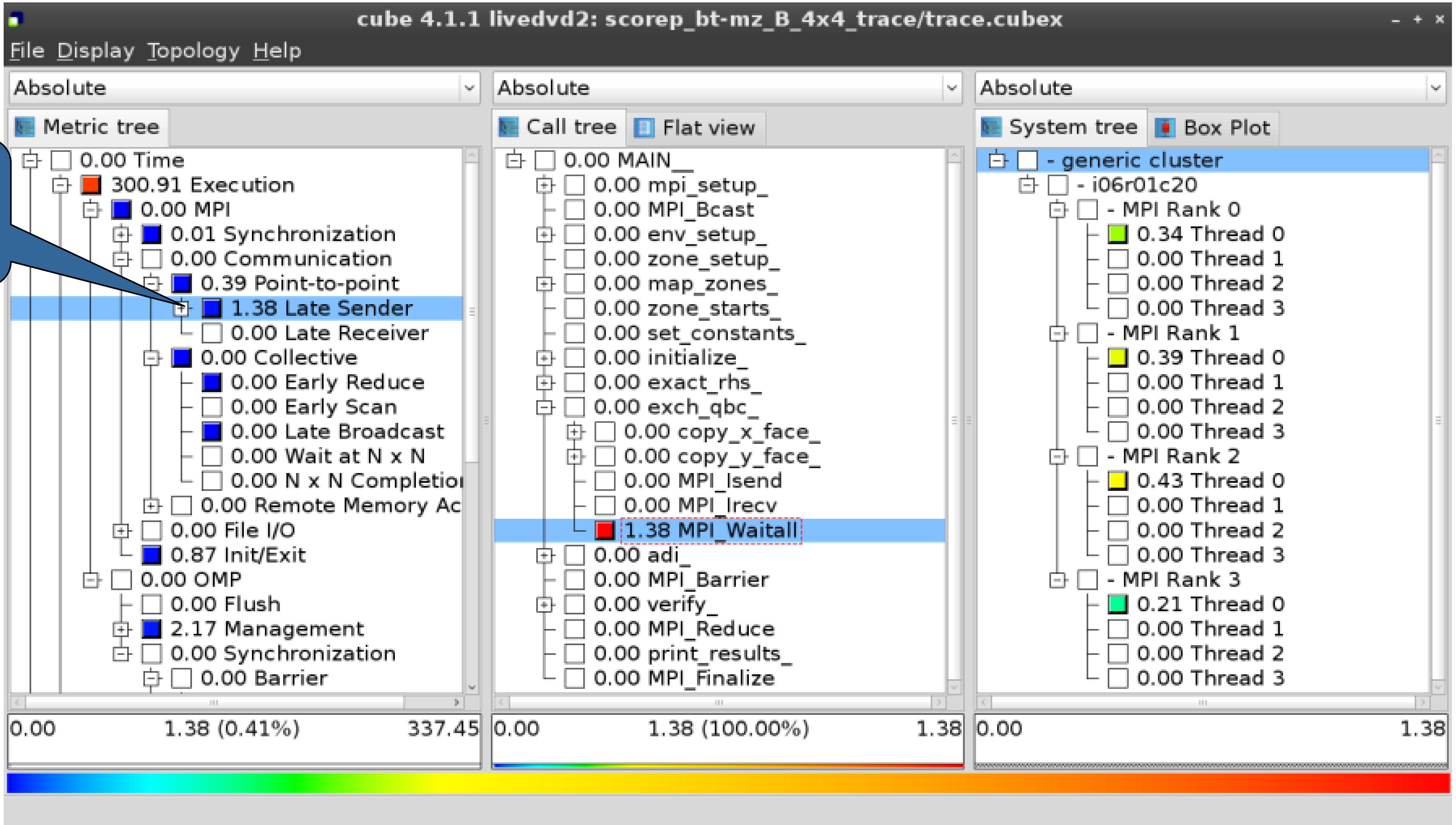
- Produces trace analysis report in the experiment directory containing trace-based wait-state metrics

```
% square  scorep_bt-mz_C_8x6_trace
INFO: Post-processing runtime summarization report (profile.cubex)...
INFO: Post-processing trace analysis report (scout.cubex)...
INFO: Displaying ./scorep_bt-mz_C_8x6_trace/trace.cubex...

                [GUI showing trace analysis report]
```

Hint:
Run ‘square -s’ first and then copy ‘trace.cubex’ to local system (laptop) using ‘scp’ to improve responsiveness of GUI

Post-processed trace analysis report

Additional trace-based metrics in metric hierarchy



cube 4.1.1 livedvd2: scorep_bt-mz_B_4x4_trace/trace.cubex

```
0.00 Time
  300.91 Execution
    0.00 MPI
      0.01 Synchronization
      0.00 Communication
        0.39 Point-to-point
          1.38 Late Sender
          0.00 Late Receiver
        0.00 Collective
          0.00 Early Reduce
          0.00 Early Scan
          0.00 Late Broadcast
          0.00 Wait at N x N
          0.00 N x N Completio
        0.00 Remote Memory Ac
      0.00 File I/O
      0.87 Init/Exit
    0.00 OMP
      0.00 Flush
      2.17 Management
      0.00 Synchronization
        0.00 Barrier
```

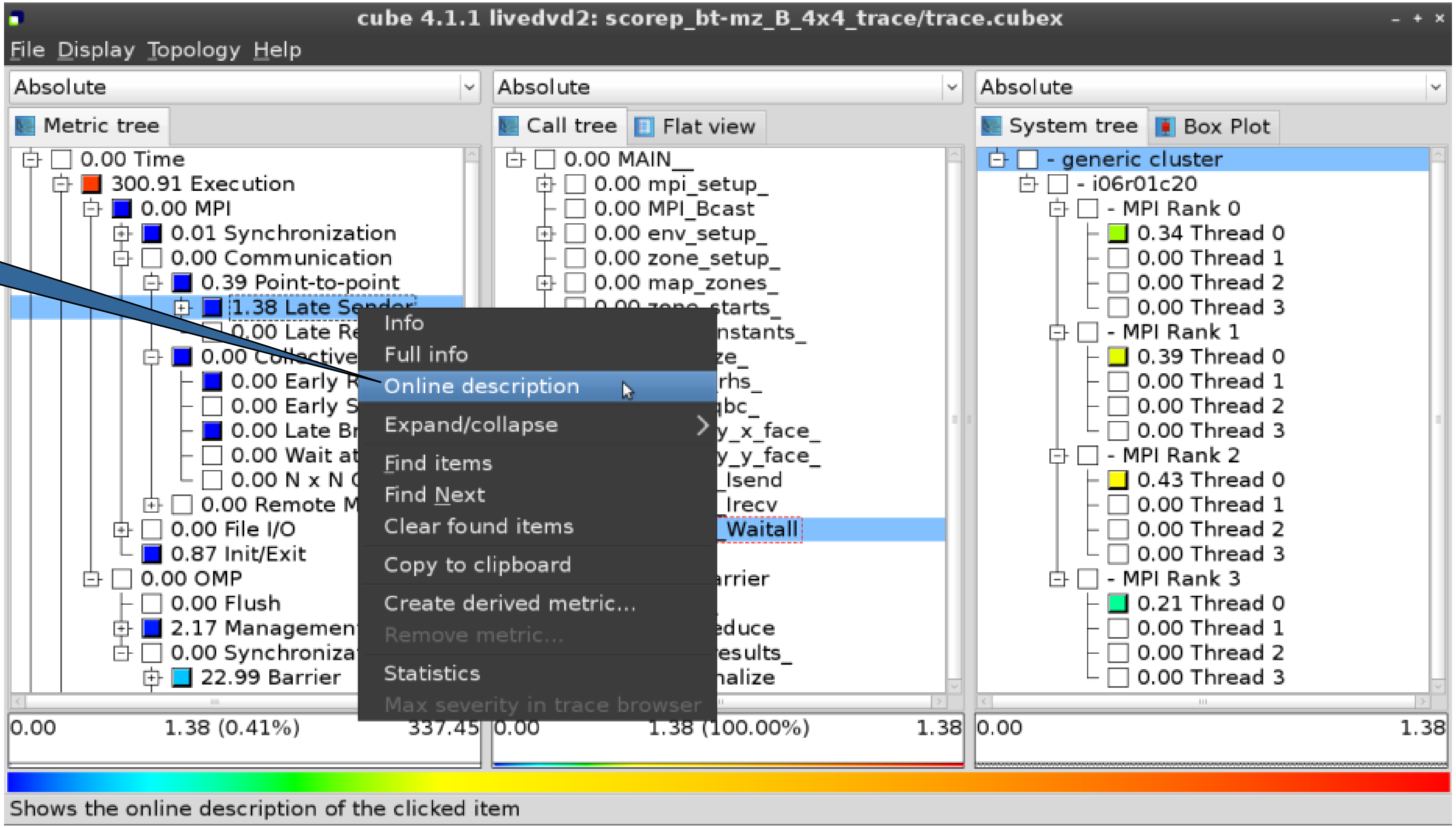
```
0.00 MAIN__
  0.00 mpi_setup_
  0.00 MPI_Bcast
  0.00 env_setup_
  0.00 zone_setup_
  0.00 map_zones_
  0.00 zone_starts_
  0.00 set_constants_
  0.00 initialize_
  0.00 exact_rhs_
  0.00 exch_qbc_
    0.00 copy_x_face_
    0.00 copy_y_face_
    0.00 MPI_Isend
    0.00 MPI_Irecv
    1.38 MPI_Waitall
  0.00 adi_
  0.00 MPI_Barrier
  0.00 verify_
  0.00 MPI_Reduce
  0.00 print_results_
  0.00 MPI_Finalize
```

```
- generic cluster
  - i06r01c20
    - MPI Rank 0
      0.34 Thread 0
      0.00 Thread 1
      0.00 Thread 2
      0.00 Thread 3
    - MPI Rank 1
      0.39 Thread 0
      0.00 Thread 1
      0.00 Thread 2
      0.00 Thread 3
    - MPI Rank 2
      0.43 Thread 0
      0.00 Thread 1
      0.00 Thread 2
      0.00 Thread 3
    - MPI Rank 3
      0.21 Thread 0
      0.00 Thread 1
      0.00 Thread 2
      0.00 Thread 3
```

```
0.00        1.38 (0.41%)        337.45
0.00        1.38 (100.00%)      1.38
0.00                            1.38
```

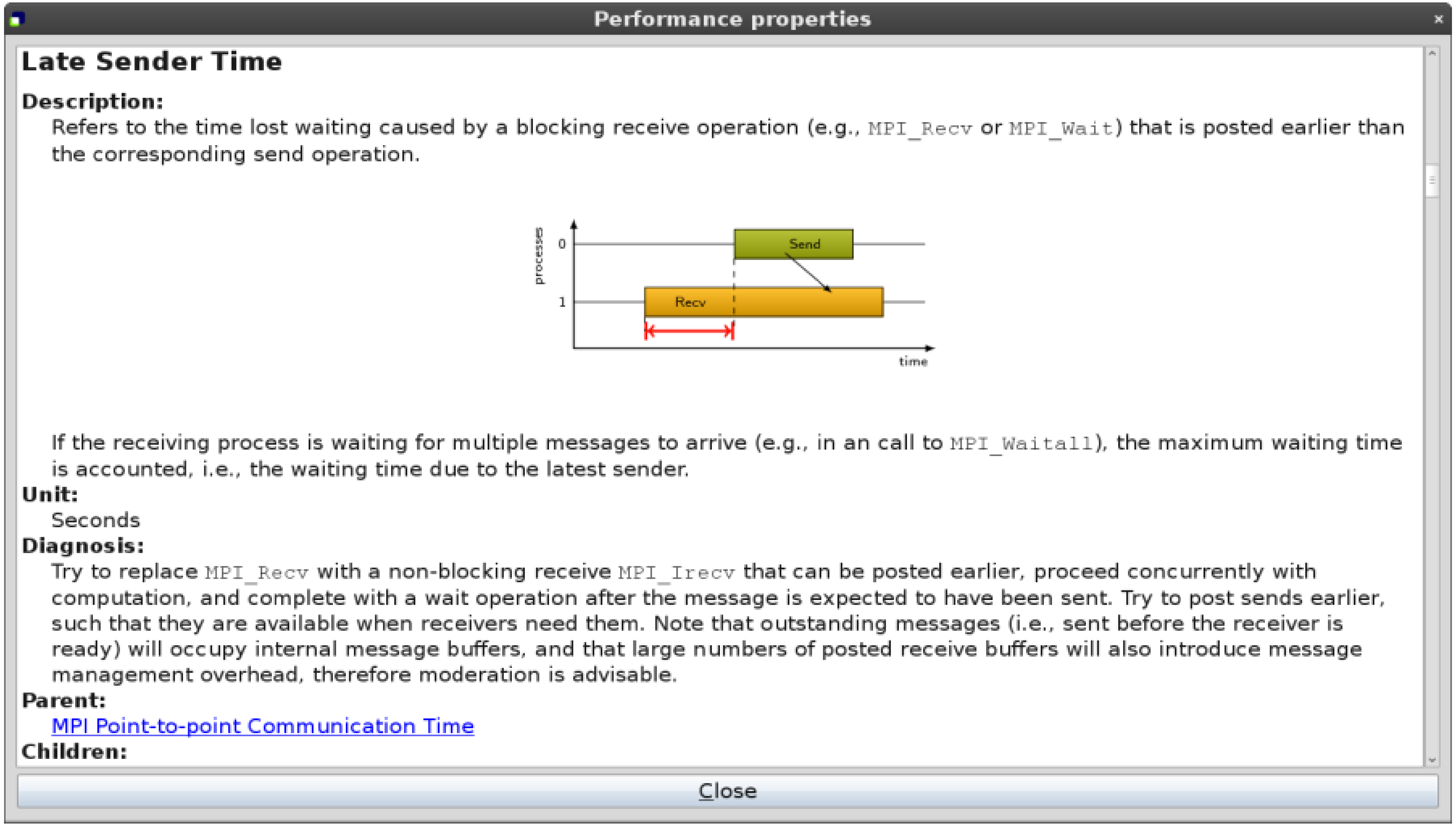
Online metric description

Access online metric description via context menu



Shows the online description of the clicked item

Online metric description



Performance properties

Late Sender Time

Description:

Refers to the time lost waiting caused by a blocking receive operation (e.g., `MPI_Recv` or `MPI_Wait`) that is posted earlier than the corresponding send operation.

If the receiving process is waiting for multiple messages to arrive (e.g., in an call to `MPI_Waitall`), the maximum waiting time is accounted, i.e., the waiting time due to the latest sender.

Unit:

Seconds

Diagnosis:

Try to replace `MPI_Recv` with a non-blocking receive `MPI_Irecv` that can be posted earlier, proceed concurrently with computation, and complete with a wait operation after the message is expected to have been sent. Try to post sends earlier, such that they are available when receivers need them. Note that outstanding messages (i.e., sent before the receiver is ready) will occupy internal message buffers, and that large numbers of posted receive buffers will also introduce message management overhead, therefore moderation is advisable.

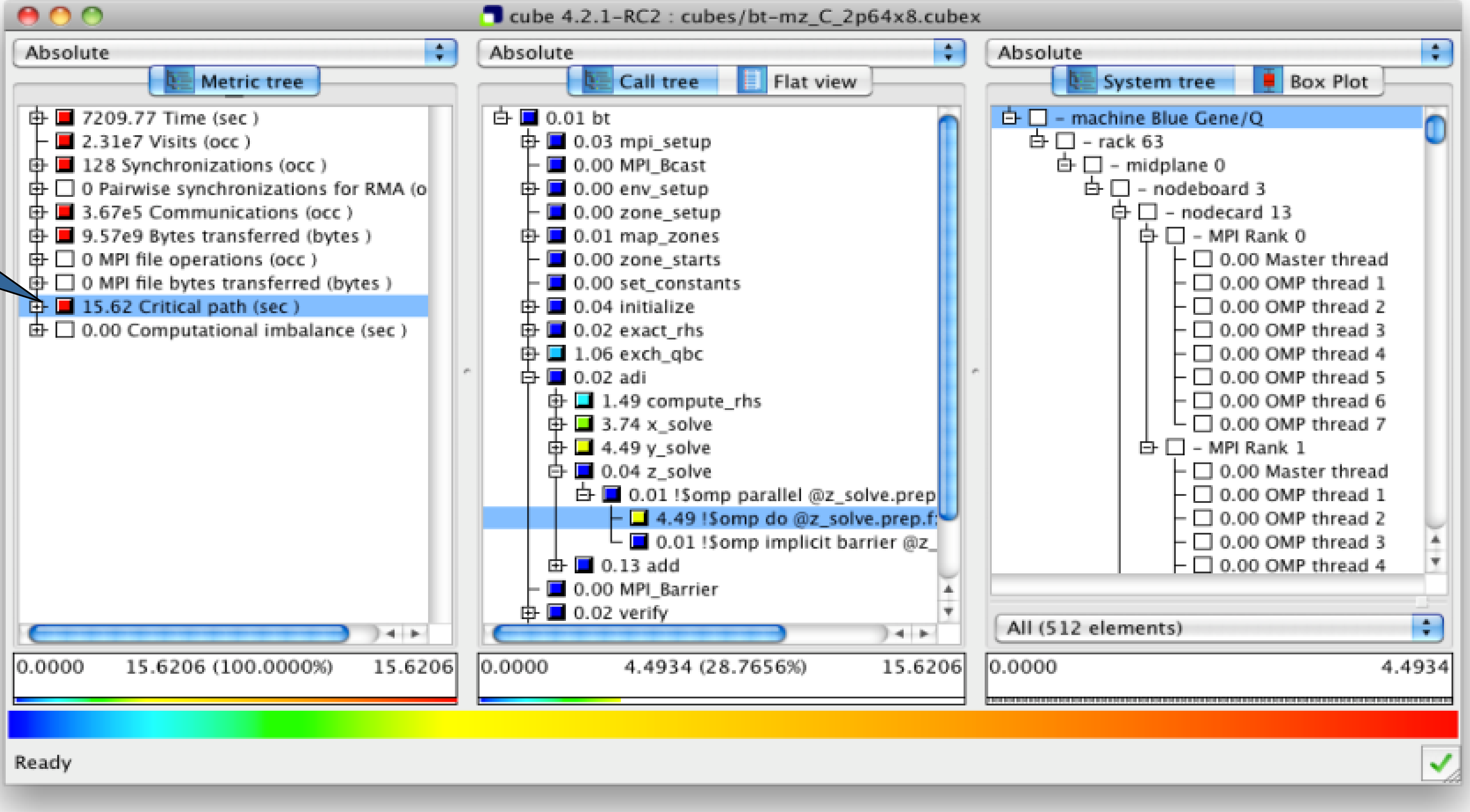
Parent:

MPI Point-to-point Communication Time

Children:

Critical-path analysis

Critical-path profile shows wall-clock time impact



cube 4.2.1-RC2 : cubes/bt-mz_C_2p64x8.cubex

Absolute — Metric tree

```
7209.77 Time (sec )
2.31e7 Visits (occ )
128 Synchronizations (occ )
0 Pairwise synchronizations for RMA (o
3.67e5 Communications (occ )
9.57e9 Bytes transferred (bytes )
0 MPI file operations (occ )
0 MPI file bytes transferred (bytes )
15.62 Critical path (sec )
0.00 Computational imbalance (sec )
```

0.0000 15.6206 (100.0000%) 15.6206

Absolute — Call tree | Flat view

```
0.01 bt
  0.03 mpi_setup
  0.00 MPI_Bcast
  0.00 env_setup
  0.00 zone_setup
  0.01 map_zones
  0.00 zone_starts
  0.00 set_constants
  0.04 initialize
  0.02 exact_rhs
  1.06 exch_qbc
  0.02 adi
    1.49 compute_rhs
    3.74 x_solve
    4.49 y_solve
    0.04 z_solve
      0.01 !$omp parallel @z_solve.prep
        4.49 !$omp do @z_solve.prep.f:
        0.01 !$omp implicit barrier @z_
    0.13 add
  0.00 MPI_Barrier
  0.02 verify
```

0.0000 4.4934 (28.7656%) 15.6206

Absolute — System tree | Box Plot

```
- machine Blue Gene/Q
  - rack 63
    - midplane 0
      - nodeboard 3
        - nodecard 13
          - MPI Rank 0
            0.00 Master thread
            0.00 OMP thread 1
            0.00 OMP thread 2
            0.00 OMP thread 3
            0.00 OMP thread 4
            0.00 OMP thread 5
            0.00 OMP thread 6
            0.00 OMP thread 7
          - MPI Rank 1
            0.00 Master thread
            0.00 OMP thread 1
            0.00 OMP thread 2
            0.00 OMP thread 3
            0.00 OMP thread 4
```

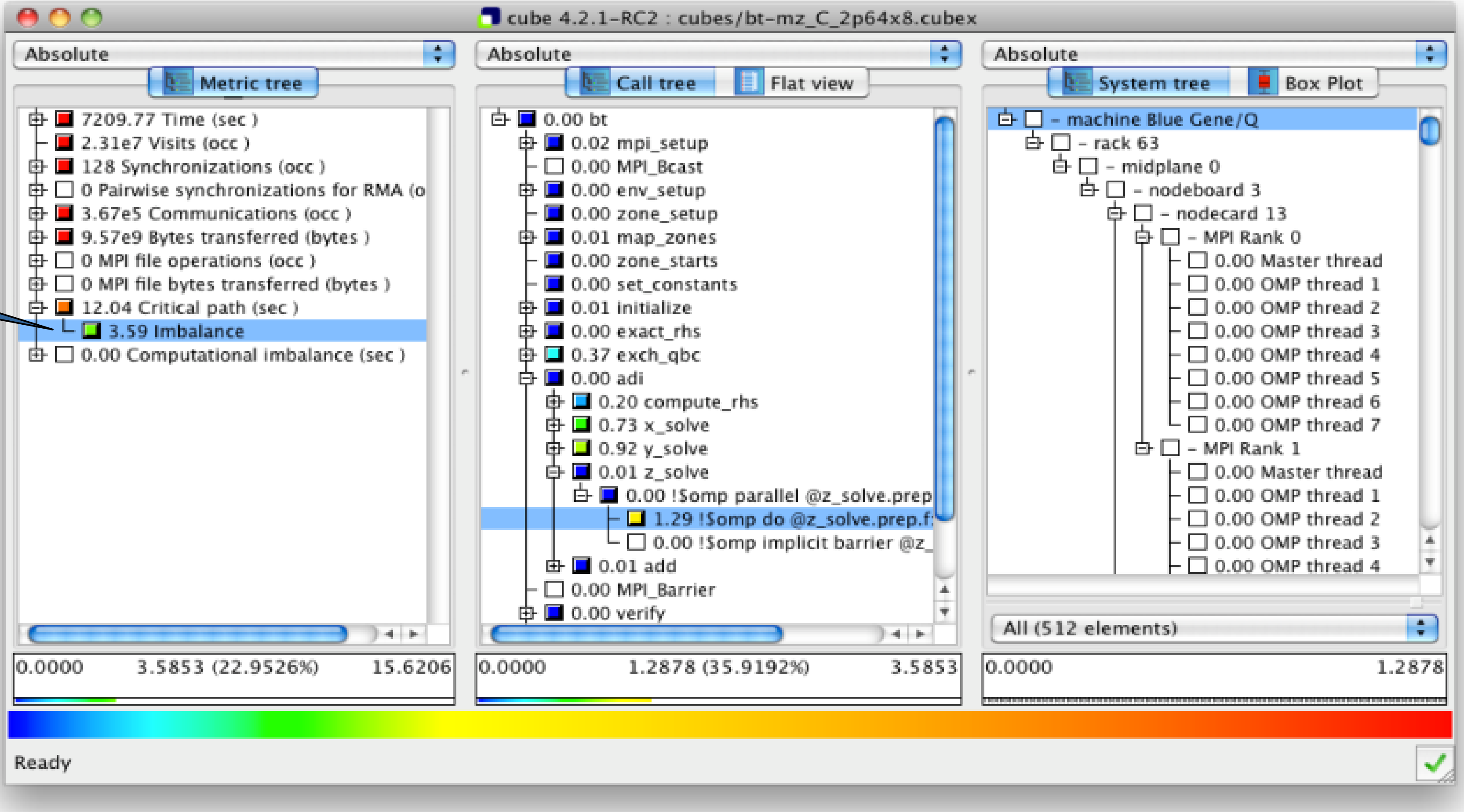
All (512 elements)

0.0000 4.4934

Ready

Critical-path analysis

Critical-path imbalance highlights inefficient parallelism



cube 4.2.1-RC2 : cubes/bt-mz_C_2p64x8.cubex

Absolute | Metric tree

- 7209.77 Time (sec)
- 2.31e7 Visits (occ)
- 128 Synchronizations (occ)
- 0 Pairwise synchronizations for RMA (o
- 3.67e5 Communications (occ)
- 9.57e9 Bytes transferred (bytes)
- 0 MPI file operations (occ)
- 0 MPI file bytes transferred (bytes)
- 12.04 Critical path (sec)
  - 3.59 Imbalance
- 0.00 Computational imbalance (sec)

0.0000 3.5853 (22.9526%) 15.6206

Absolute | Call tree | Flat view

- 0.00 bt
  - 0.02 mpi_setup
  - 0.00 MPI_Bcast
  - 0.00 env_setup
  - 0.00 zone_setup
  - 0.01 map_zones
  - 0.00 zone_starts
  - 0.00 set_constants
  - 0.01 initialize
  - 0.00 exact_rhs
  - 0.37 exch_qbc
  - 0.00 adi
    - 0.20 compute_rhs
    - 0.73 x_solve
    - 0.92 y_solve
    - 0.01 z_solve
      - 0.00 !$omp parallel @z_solve.prep
        - 1.29 !$omp do @z_solve.prep.f
        - 0.00 !$omp implicit barrier @z_
    - 0.01 add
  - 0.00 MPI_Barrier
  - 0.00 verify

0.0000 1.2878 (35.9192%) 3.5853

Absolute | System tree | Box Plot

- machine Blue Gene/Q
  - rack 63
    - midplane 0
      - nodeboard 3
        - nodecard 13
          - MPI Rank 0
            - 0.00 Master thread
            - 0.00 OMP thread 1
            - 0.00 OMP thread 2
            - 0.00 OMP thread 3
            - 0.00 OMP thread 4
            - 0.00 OMP thread 5
            - 0.00 OMP thread 6
            - 0.00 OMP thread 7
          - MPI Rank 1
            - 0.00 Master thread
            - 0.00 OMP thread 1
            - 0.00 OMP thread 2
            - 0.00 OMP thread 3
            - 0.00 OMP thread 4

All (512 elements)

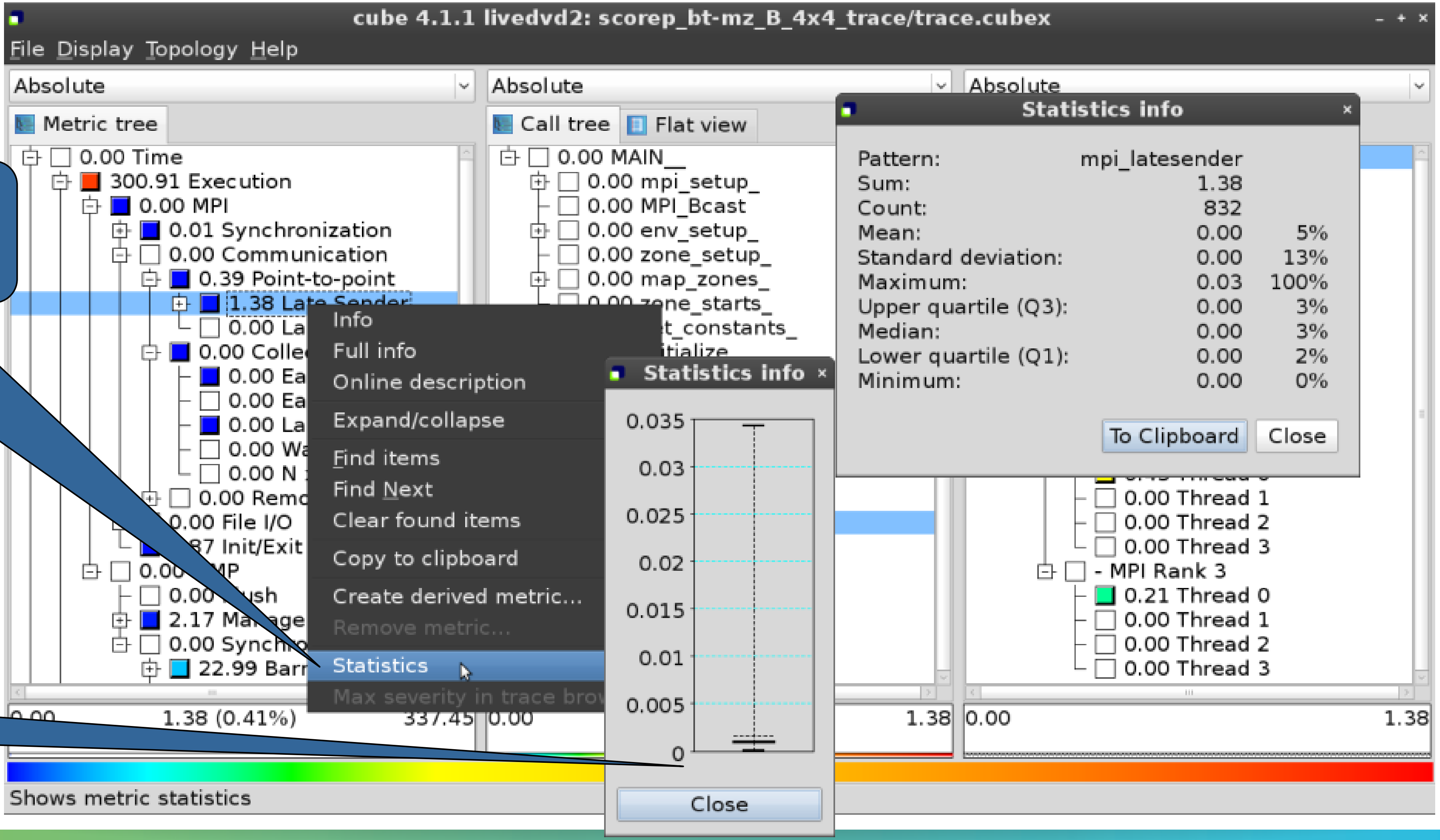
0.0000 1.2878

Ready

Pattern instance statistics

Access pattern instance statistics via context menu

Click to get statistics details



Statistics info

Pattern:	mpi_latesender	
Sum:	1.38	
Count:	832	
Mean:	0.00	5%
Standard deviation:	0.00	13%
Maximum:	0.03	100%
Upper quartile (Q3):	0.00	3%
Median:	0.00	3%
Lower quartile (Q1):	0.00	2%
Minimum:	0.00	0%

To Clipboard Close

Demo:
TeaLeaf case study

trace tools
scalasca

arm
FAU FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG
Lawrence Livermore National Laboratory
RWTH AACHEN UNIVERSITY
TECHNISCHE UNIVERSITÄT DRESDEN
UNIVERSITÉ DE VERSAILLES ST-QUENTIN-EN-YVELINES
Universität Stuttgart
BSC Barcelona Supercomputing Center Centro Nacional de Supercomputación
JÜLICH Forschungszentrum
lrz Leibniz Supercomputing Centre of the Bavarian Academy of Sciences and Humanities
TECHNISCHE UNIVERSITÄT DARMSTADT
TUM Technische Universität München
UNIVERSITY OF OREGON
THE UNIVERSITY of TENNESSEE

Case study: TeaLeaf

- HPC mini-app developed by the UK Mini-App Consortium
  - Solves the linear 2D heat conduction equation on a spatially decomposed regular grid using a 5 point stencil with implicit solvers
  - Part of the Mantevo 3.0 suite
  - Available on GitHub: https://uk-mac.github.io/TeaLeaf/

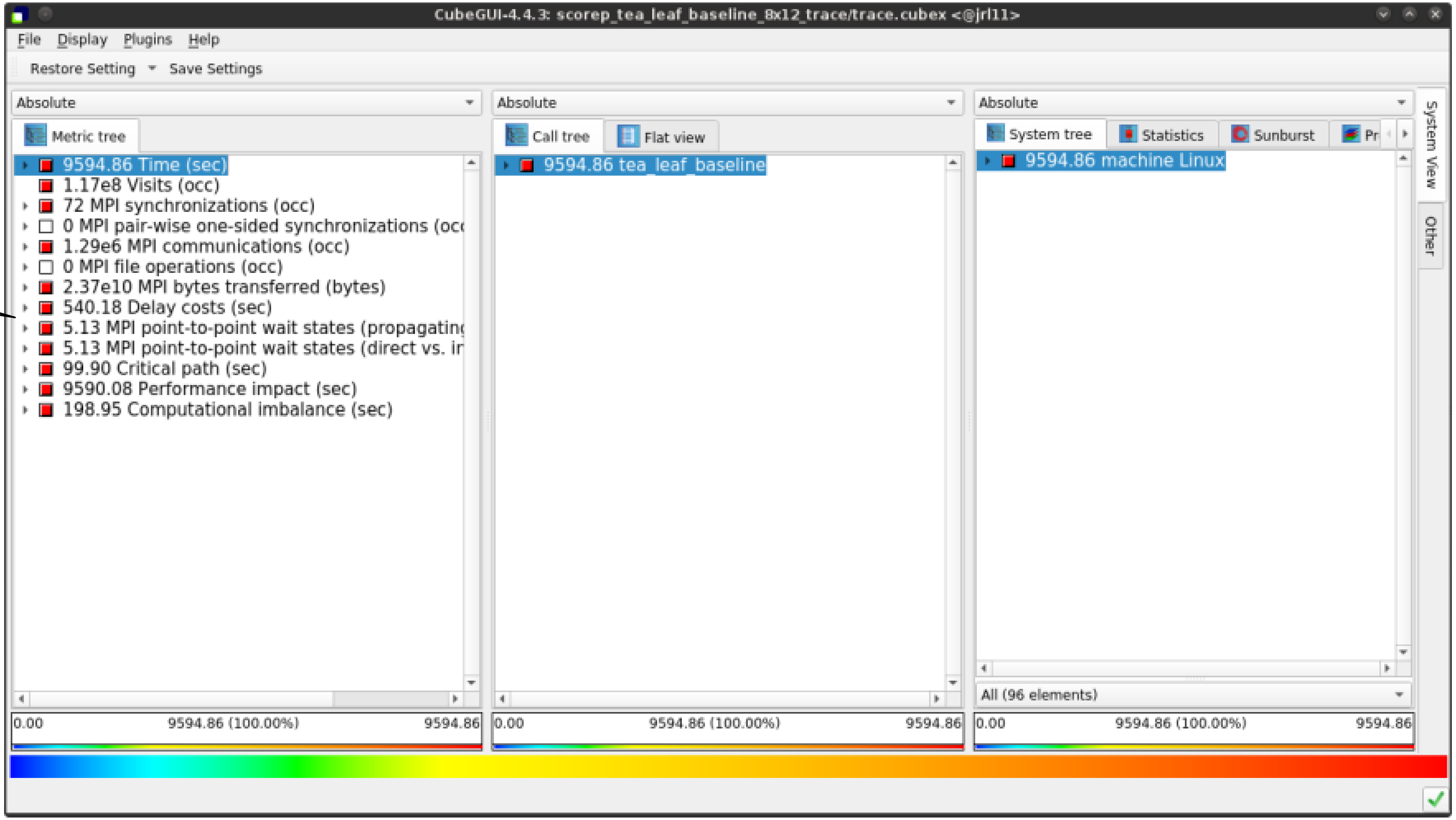
- Measurements of TeaLeaf reference v1.0 taken on Jureca cluster @ JSC
  - Using Intel 19.0.3 compilers, Intel MPI 2019.3, Score-P 5.0, and Scalasca 2.5
  - Run configuration
    - 8 MPI ranks with 12 OpenMP threads each
    - Distributed across 4 compute nodes (2 ranks per node)
    - Test problem "5": 4000 × 4000 cells, CG solver

```
% cd /work/y23/shared/tutorial/samples
% cube scorep_tea_leaf_baseline_8x12_trace/trace.cubex
                         [GUI showing post-processed trace analysis report]
```

Hint:
Copy 'trace.cubex' to local system (laptop) using 'scp' to improve responsiveness of GUI

Scalasca analysis report exploration (opening view)

Additional top-level metrics produced by the trace analysis…



CubeGUI-4.4.3: scorep_tea_leaf_baseline_8x12_trace/trace.cubex <@jrl11>

Metric tree:

- 9594.86 Time (sec)
- 1.17e8 Visits (occ)
- 72 MPI synchronizations (occ)
- 0 MPI pair-wise one-sided synchronizations (occ
- 1.29e6 MPI communications (occ)
- 0 MPI file operations (occ)
- 2.37e10 MPI bytes transferred (bytes)
- 540.18 Delay costs (sec)
- 5.13 MPI point-to-point wait states (propagating
- 5.13 MPI point-to-point wait states (direct vs. ir
- 99.90 Critical path (sec)
- 9590.08 Performance impact (sec)
- 198.95 Computational imbalance (sec)

Call tree: 9594.86 tea_leaf_baseline

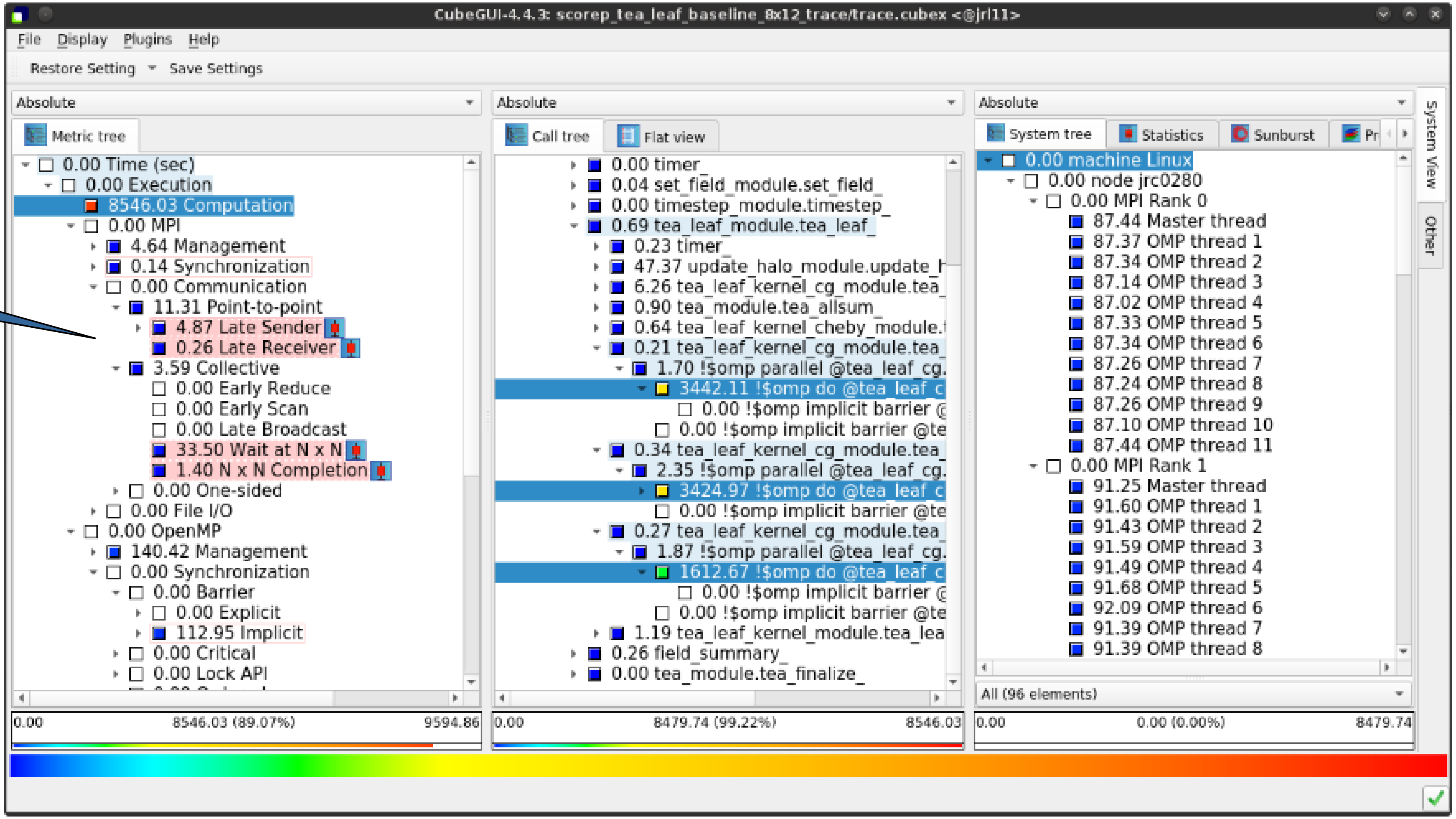
System tree: 9594.86 machine Linux

All (96 elements)

0.00 9594.86 (100.00%) 9594.86

Scalasca wait-state metrics

…plus additional wait-state metrics as part of the “Time” hierarchy



CubeGUI-4.4.3: scorep_tea_leaf_baseline_8x12_trace/trace.cubex <@jrl11>

```
▾ □ 0.00 Time (sec)
  ▾ □ 0.00 Execution
      ■ 8546.03 Computation
    ▾ □ 0.00 MPI
      ▸ ■ 4.64 Management
      ▸ ■ 0.14 Synchronization
      ▾ □ 0.00 Communication
        ▾ ■ 11.31 Point-to-point
          ▸ ■ 4.87 Late Sender
            ■ 0.26 Late Receiver
        ▾ ■ 3.59 Collective
            □ 0.00 Early Reduce
            □ 0.00 Early Scan
            □ 0.00 Late Broadcast
            ■ 33.50 Wait at N x N
            ■ 1.40 N x N Completion
        ▸ □ 0.00 One-sided
      ▸ □ 0.00 File I/O
    ▾ □ 0.00 OpenMP
      ▸ ■ 140.42 Management
      ▾ □ 0.00 Synchronization
        ▾ □ 0.00 Barrier
          ▸ □ 0.00 Explicit
          ▸ ■ 112.95 Implicit
        ▸ □ 0.00 Critical
        ▸ □ 0.00 Lock API
```

```
0.00          8546.03 (89.07%)          9594.86
```

```
▸ ■ 0.00 timer_
▸ ■ 0.04 set_field_module.set_field_
▸ ■ 0.00 timestep_module.timestep_
▾ ■ 0.69 tea_leaf_module.tea_leaf_
  ▸ ■ 0.23 timer_
  ▸ ■ 47.37 update_halo_module.update_h
  ▸ ■ 6.26 tea_leaf_kernel_cg_module.tea_
  ▸ ■ 0.90 tea_module.tea_allsum_
  ▸ ■ 0.64 tea_leaf_kernel_cheby_module.
  ▾ ■ 0.21 tea_leaf_kernel_cg_module.tea_
    ▾ ■ 1.70 !$omp parallel @tea_leaf_cg.
      ▾ ■ 3442.11 !$omp do @tea_leaf_c
          □ 0.00 !$omp implicit barrier @
        □ 0.00 !$omp implicit barrier @te
  ▾ ■ 0.34 tea_leaf_kernel_cg_module.tea_
    ▾ ■ 2.35 !$omp parallel @tea_leaf_cg.
      ▸ ■ 3424.97 !$omp do @tea_leaf_c
        □ 0.00 !$omp implicit barrier @te
  ▾ ■ 0.27 tea_leaf_kernel_cg_module.tea_
    ▾ ■ 1.87 !$omp parallel @tea_leaf_cg.
      ▾ ■ 1612.67 !$omp do @tea_leaf_c
          □ 0.00 !$omp implicit barrier @
        □ 0.00 !$omp implicit barrier @te
  ▸ ■ 1.19 tea_leaf_kernel_module.tea_lea
▸ ■ 0.26 field_summary_
▸ ■ 0.00 tea_module.tea_finalize_
```

```
0.00          8479.74 (99.22%)          8546.03
```

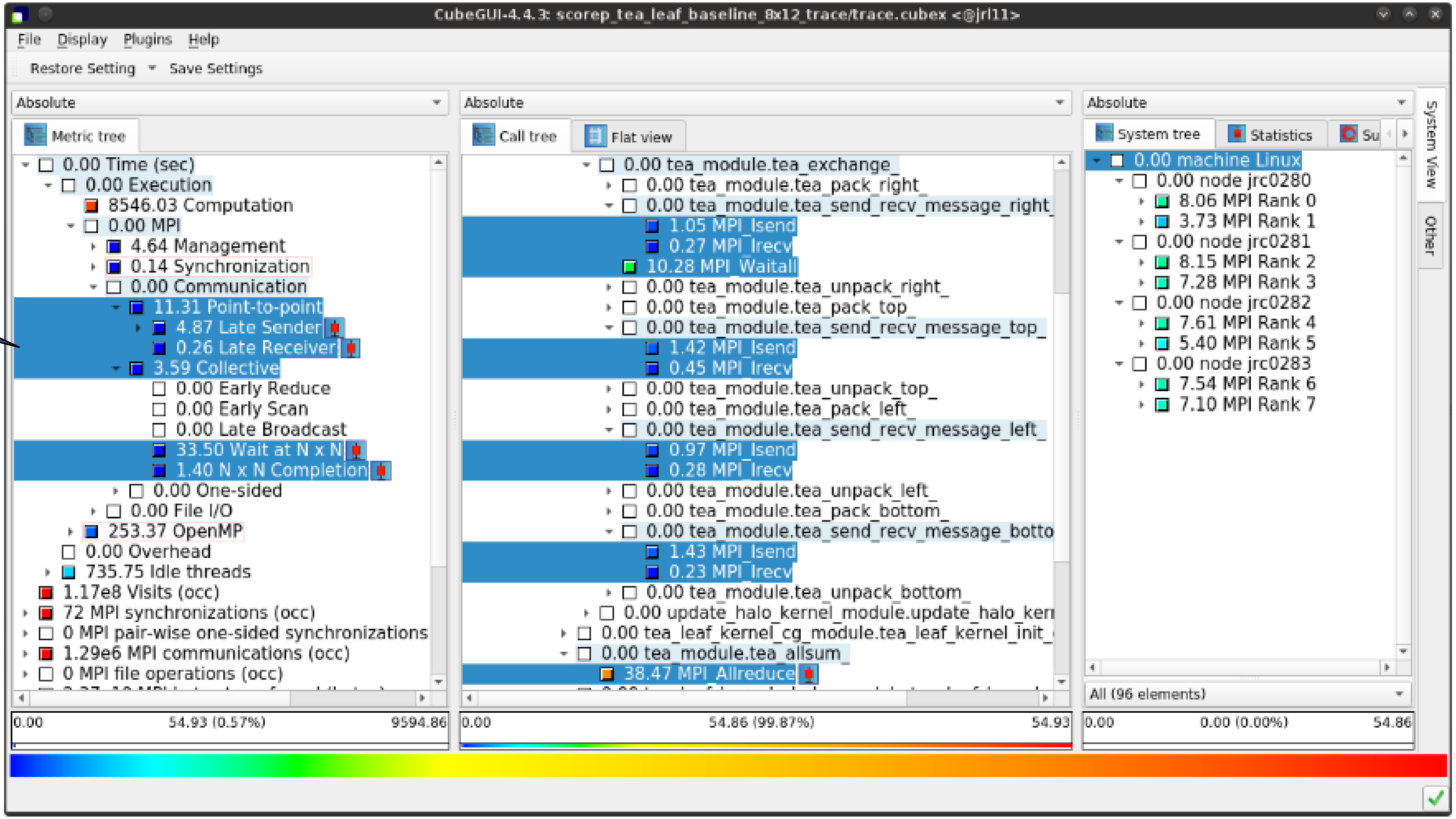
```
▾ □ 0.00 machine Linux
  ▾ □ 0.00 node jrc0280
    ▾ □ 0.00 MPI Rank 0
        ■ 87.44 Master thread
        ■ 87.37 OMP thread 1
        ■ 87.34 OMP thread 2
        ■ 87.14 OMP thread 3
        ■ 87.02 OMP thread 4
        ■ 87.33 OMP thread 5
        ■ 87.34 OMP thread 6
        ■ 87.26 OMP thread 7
        ■ 87.24 OMP thread 8
        ■ 87.26 OMP thread 9
        ■ 87.10 OMP thread 10
        ■ 87.44 OMP thread 11
    ▾ □ 0.00 MPI Rank 1
        ■ 91.25 Master thread
        ■ 91.60 OMP thread 1
        ■ 91.43 OMP thread 2
        ■ 91.59 OMP thread 3
        ■ 91.49 OMP thread 4
        ■ 91.68 OMP thread 5
        ■ 92.09 OMP thread 6
        ■ 91.39 OMP thread 7
        ■ 91.39 OMP thread 8
```

All (96 elements)

```
0.00          0.00 (0.00%)          8479.74
```

TeaLeaf Scalasca report analysis (I)

While MPI communication time and wait states are small (~0.6% of the total execution time)…



CubeGUI-4.4.3: scorep_tea_leaf_baseline_8x12_trace/trace.cubex <@jrl11>
File Display Plugins Help
Restore Setting Save Settings
Absolute
Metric tree
Call tree
Flat view
System tree
Statistics
Su
System View
Other

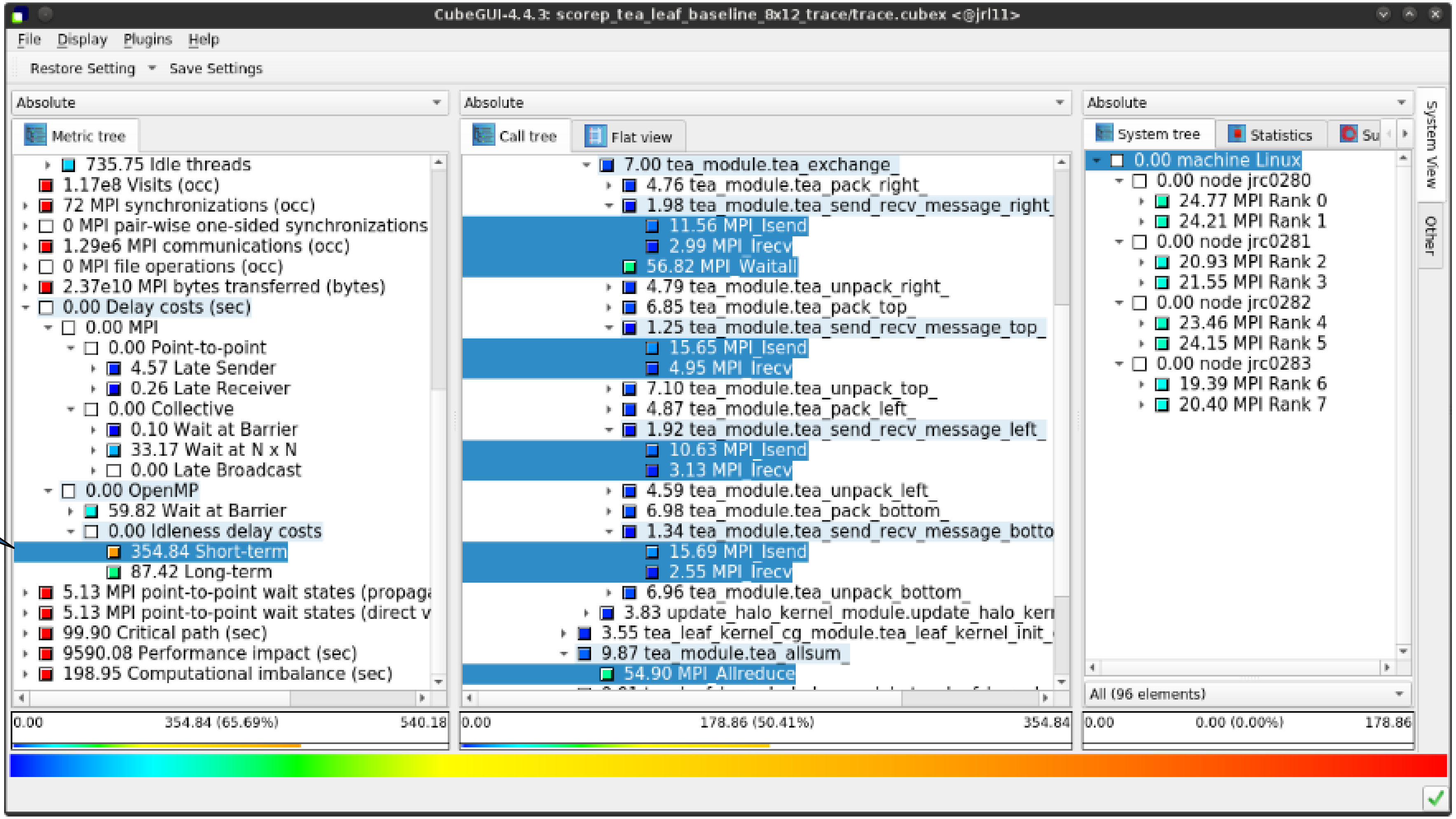
```
0.00 Time (sec)
  0.00 Execution
    8546.03 Computation
    0.00 MPI
      4.64 Management
      0.14 Synchronization
      0.00 Communication
        11.31 Point-to-point
          4.87 Late Sender
          0.26 Late Receiver
        3.59 Collective
          0.00 Early Reduce
          0.00 Early Scan
          0.00 Late Broadcast
          33.50 Wait at N x N
          1.40 N x N Completion
        0.00 One-sided
      0.00 File I/O
    253.37 OpenMP
  0.00 Overhead
  735.75 Idle threads
1.17e8 Visits (occ)
72 MPI synchronizations (occ)
0 MPI pair-wise one-sided synchronizations
1.29e6 MPI communications (occ)
0 MPI file operations (occ)

0.00          54.93 (0.57%)          9594.86
```

```
0.00 tea_module.tea_exchange_
  0.00 tea_module.tea_pack_right_
  0.00 tea_module.tea_send_recv_message_right_
    1.05 MPI_Isend
    0.27 MPI_Irecv
  10.28 MPI_Waitall
  0.00 tea_module.tea_unpack_right_
  0.00 tea_module.tea_pack_top_
  0.00 tea_module.tea_send_recv_message_top_
    1.42 MPI_Isend
    0.45 MPI_Irecv
  0.00 tea_module.tea_unpack_top_
  0.00 tea_module.tea_pack_left_
  0.00 tea_module.tea_send_recv_message_left_
    0.97 MPI_Isend
    0.28 MPI_Irecv
  0.00 tea_module.tea_unpack_left_
  0.00 tea_module.tea_pack_bottom_
  0.00 tea_module.tea_send_recv_message_botto
    1.43 MPI_Isend
    0.23 MPI_Irecv
  0.00 tea_module.tea_unpack_bottom_
0.00 update_halo_kernel_module.update_halo_ker
0.00 tea_leaf_kernel_cg_module.tea_leaf_kernel_init_
0.00 tea_module.tea_allsum_
  38.47 MPI_Allreduce

0.00          54.86 (99.87%)          54.93
```

```
0.00 machine Linux
  0.00 node jrc0280
    8.06 MPI Rank 0
    3.73 MPI Rank 1
  0.00 node jrc0281
    8.15 MPI Rank 2
    7.28 MPI Rank 3
  0.00 node jrc0282
    7.61 MPI Rank 4
    5.40 MPI Rank 5
  0.00 node jrc0283
    7.54 MPI Rank 6
    7.10 MPI Rank 7

All (96 elements)
0.00          0.00 (0.00%)          54.86
```

TeaLeaf Scalasca report analysis (II)

…they directly cause a significant amount of the OpenMP thread idleness



CubeGUI-4.4.3: scorep_tea_leaf_baseline_8x12_trace/trace.cubex <@jrl11>

```
735.75 Idle threads
1.17e8 Visits (occ)
72 MPI synchronizations (occ)
0 MPI pair-wise one-sided synchronizations
1.29e6 MPI communications (occ)
0 MPI file operations (occ)
2.37e10 MPI bytes transferred (bytes)
0.00 Delay costs (sec)
  0.00 MPI
    0.00 Point-to-point
      4.57 Late Sender
      0.26 Late Receiver
    0.00 Collective
      0.10 Wait at Barrier
      33.17 Wait at N x N
      0.00 Late Broadcast
  0.00 OpenMP
    59.82 Wait at Barrier
    0.00 Idleness delay costs
      354.84 Short-term
      87.42 Long-term
5.13 MPI point-to-point wait states (propaga
5.13 MPI point-to-point wait states (direct v
99.90 Critical path (sec)
9590.08 Performance impact (sec)
198.95 Computational imbalance (sec)
```

```
7.00 tea_module.tea_exchange_
  4.76 tea_module.tea_pack_right_
  1.98 tea_module.tea_send_recv_message_right_
    11.56 MPI_Isend
    2.99 MPI_Irecv
  56.82 MPI_Waitall
  4.79 tea_module.tea_unpack_right_
  6.85 tea_module.tea_pack_top_
  1.25 tea_module.tea_send_recv_message_top_
    15.65 MPI_Isend
    4.95 MPI_Irecv
  7.10 tea_module.tea_unpack_top_
  4.87 tea_module.tea_pack_left_
  1.92 tea_module.tea_send_recv_message_left_
    10.63 MPI_Isend
    3.13 MPI_Irecv
  4.59 tea_module.tea_unpack_left_
  6.98 tea_module.tea_pack_bottom_
  1.34 tea_module.tea_send_recv_message_botto
    15.69 MPI_Isend
    2.55 MPI_Irecv
  6.96 tea_module.tea_unpack_bottom_
3.83 update_halo_kernel_module.update_halo_kern
3.55 tea_leaf_kernel_cg_module.tea_leaf_kernel_init_
9.87 tea_module.tea_allsum_
  54.90 MPI_Allreduce
```

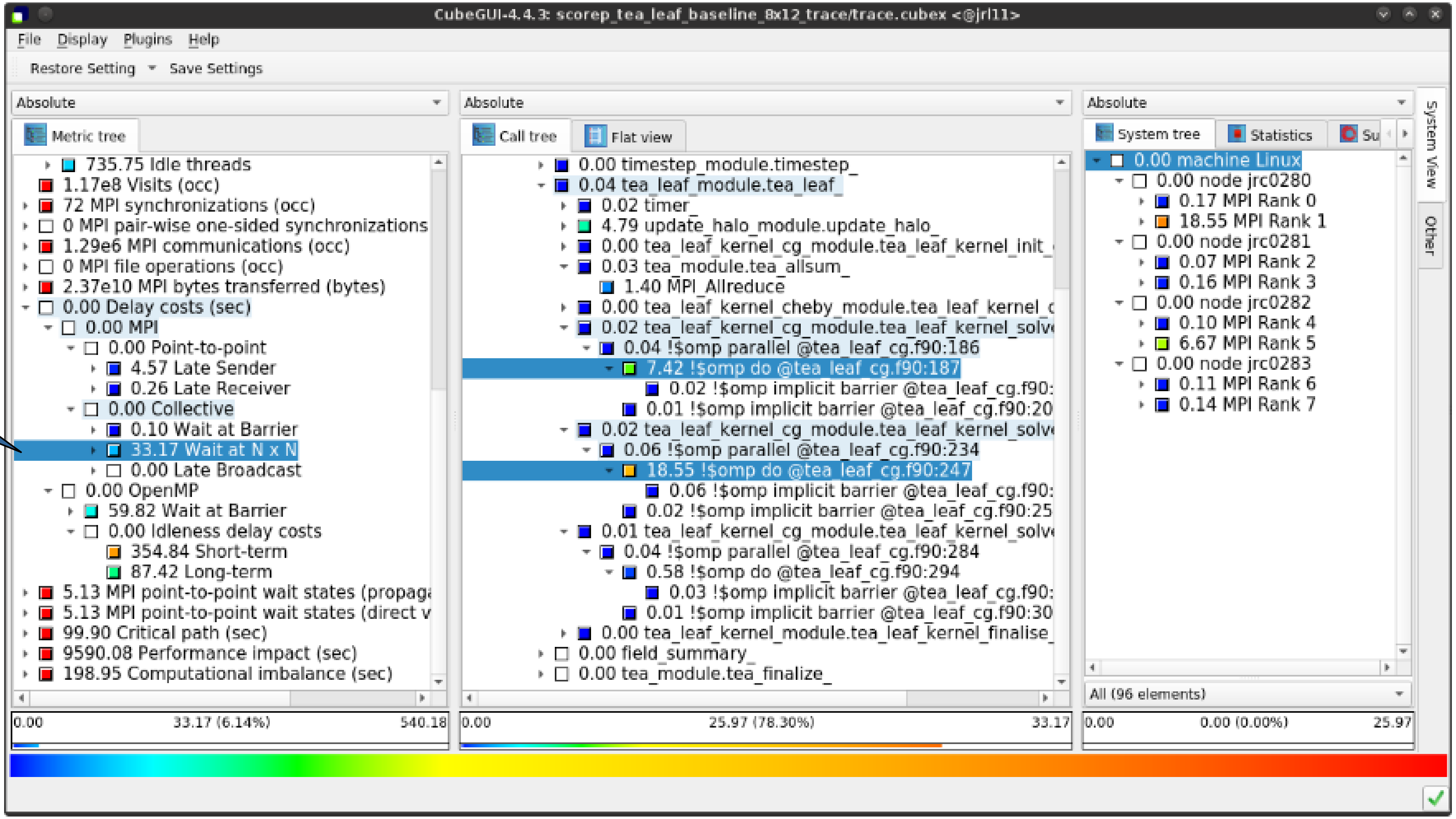
```
0.00 machine Linux
  0.00 node jrc0280
    24.77 MPI Rank 0
    24.21 MPI Rank 1
  0.00 node jrc0281
    20.93 MPI Rank 2
    21.55 MPI Rank 3
  0.00 node jrc0282
    23.46 MPI Rank 4
    24.15 MPI Rank 5
  0.00 node jrc0283
    19.39 MPI Rank 6
    20.40 MPI Rank 7
```

All (96 elements)

0.00	354.84 (65.69%)	540.18
0.00	178.86 (50.41%)	354.84
0.00	0.00 (0.00%)	178.86

TeaLeaf Scalasca report analysis (III)

The “Wait at NxN” collective wait states are mostly caused by the first 2 OpenMP `do` loops of the solver (on ranks 5 & 1, resp.)…



CubeGUI-4.4.3: scorep_tea_leaf_baseline_8x12_trace/trace.cubex <@jrl11>

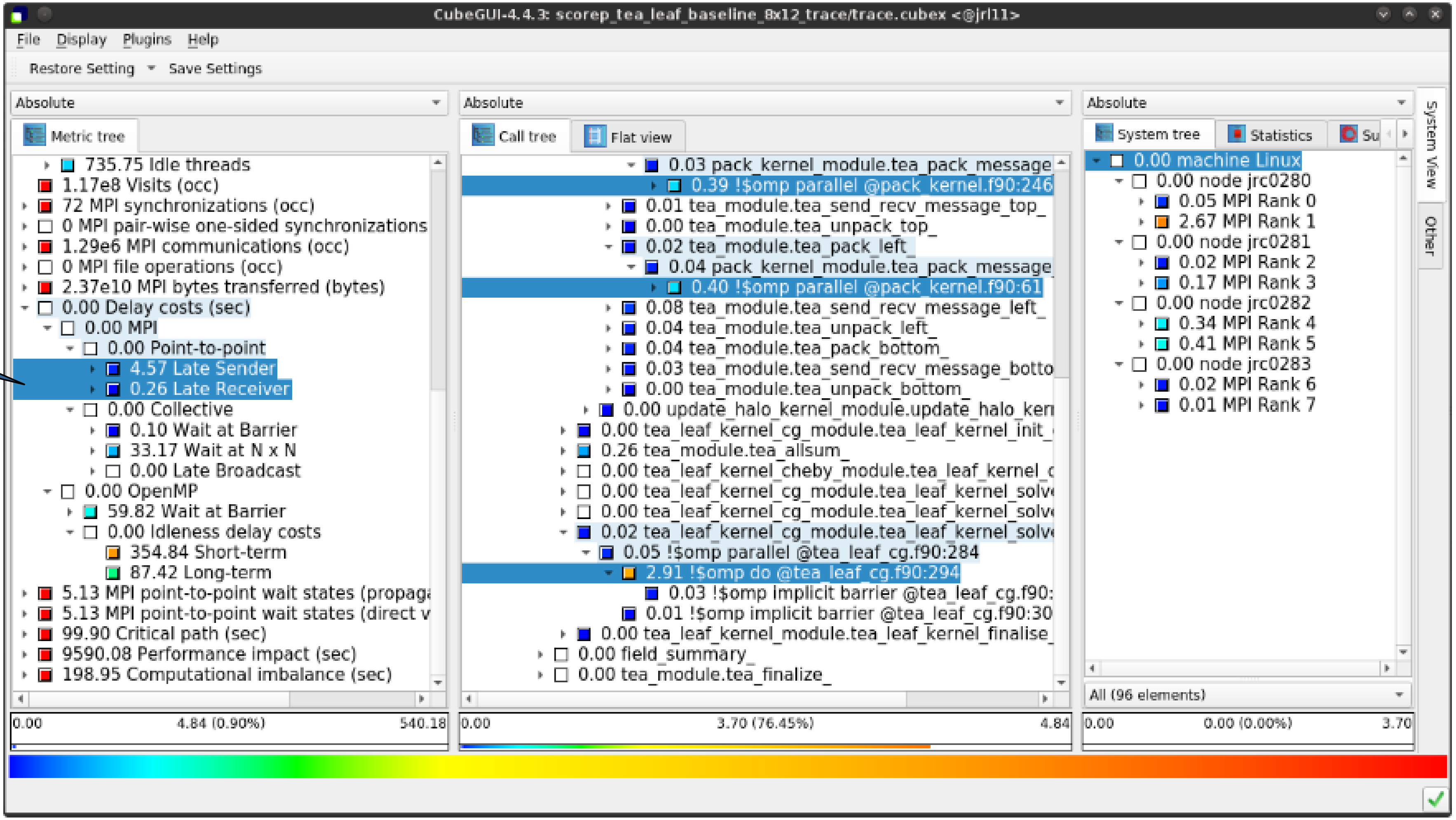
```
  735.75 Idle threads
  1.17e8 Visits (occ)
  72 MPI synchronizations (occ)
  0 MPI pair-wise one-sided synchronizations
  1.29e6 MPI communications (occ)
  0 MPI file operations (occ)
  2.37e10 MPI bytes transferred (bytes)
  0.00 Delay costs (sec)
    0.00 MPI
      0.00 Point-to-point
        4.57 Late Sender
        0.26 Late Receiver
      0.00 Collective
        0.10 Wait at Barrier
        33.17 Wait at N x N
        0.00 Late Broadcast
    0.00 OpenMP
      59.82 Wait at Barrier
      0.00 Idleness delay costs
        354.84 Short-term
        87.42 Long-term
  5.13 MPI point-to-point wait states (propaga
  5.13 MPI point-to-point wait states (direct v
  99.90 Critical path (sec)
  9590.08 Performance impact (sec)
  198.95 Computational imbalance (sec)

0.00            33.17 (6.14%)            540.18
```

```
0.00 timestep_module.timestep_
0.04 tea_leaf_module.tea_leaf_
  0.02 timer
  4.79 update_halo_module.update_halo_
  0.00 tea_leaf_kernel_cg_module.tea_leaf_kernel_init_
  0.03 tea_module.tea_allsum_
    1.40 MPI_Allreduce
  0.00 tea_leaf_kernel_cheby_module.tea_leaf_kernel_c
  0.02 tea_leaf_kernel_cg_module.tea_leaf_kernel_solve
    0.04 !$omp parallel @tea_leaf_cg.f90:186
      7.42 !$omp do @tea_leaf_cg.f90:187
        0.02 !$omp implicit barrier @tea_leaf_cg.f90:
      0.01 !$omp implicit barrier @tea_leaf_cg.f90:20
  0.02 tea_leaf_kernel_cg_module.tea_leaf_kernel_solve
    0.06 !$omp parallel @tea_leaf_cg.f90:234
      18.55 !$omp do @tea_leaf_cg.f90:247
        0.06 !$omp implicit barrier @tea_leaf_cg.f90:
      0.02 !$omp implicit barrier @tea_leaf_cg.f90:25
  0.01 tea_leaf_kernel_cg_module.tea_leaf_kernel_solve
    0.04 !$omp parallel @tea_leaf_cg.f90:284
      0.58 !$omp do @tea_leaf_cg.f90:294
        0.03 !$omp implicit barrier @tea_leaf_cg.f90:
      0.01 !$omp implicit barrier @tea_leaf_cg.f90:30
  0.00 tea_leaf_kernel_module.tea_leaf_kernel_finalise_
0.00 field_summary_
0.00 tea_module.tea_finalize_

0.00            25.97 (78.30%)            33.17
```

```
0.00 machine Linux
  0.00 node jrc0280
    0.17 MPI Rank 0
    18.55 MPI Rank 1
  0.00 node jrc0281
    0.07 MPI Rank 2
    0.16 MPI Rank 3
  0.00 node jrc0282
    0.10 MPI Rank 4
    6.67 MPI Rank 5
  0.00 node jrc0283
    0.11 MPI Rank 6
    0.14 MPI Rank 7

All (96 elements)
0.00            0.00 (0.00%)            25.97
```

TeaLeaf Scalasca report analysis (IV)

…while the MPI point-to-point wait states are caused by the 3rd solver `do` loop (on rank 1) and two loops in the halo exchange



CubeGUI-4.4.3: scorep_tea_leaf_baseline_8x12_trace/trace.cubex <@jrl11>

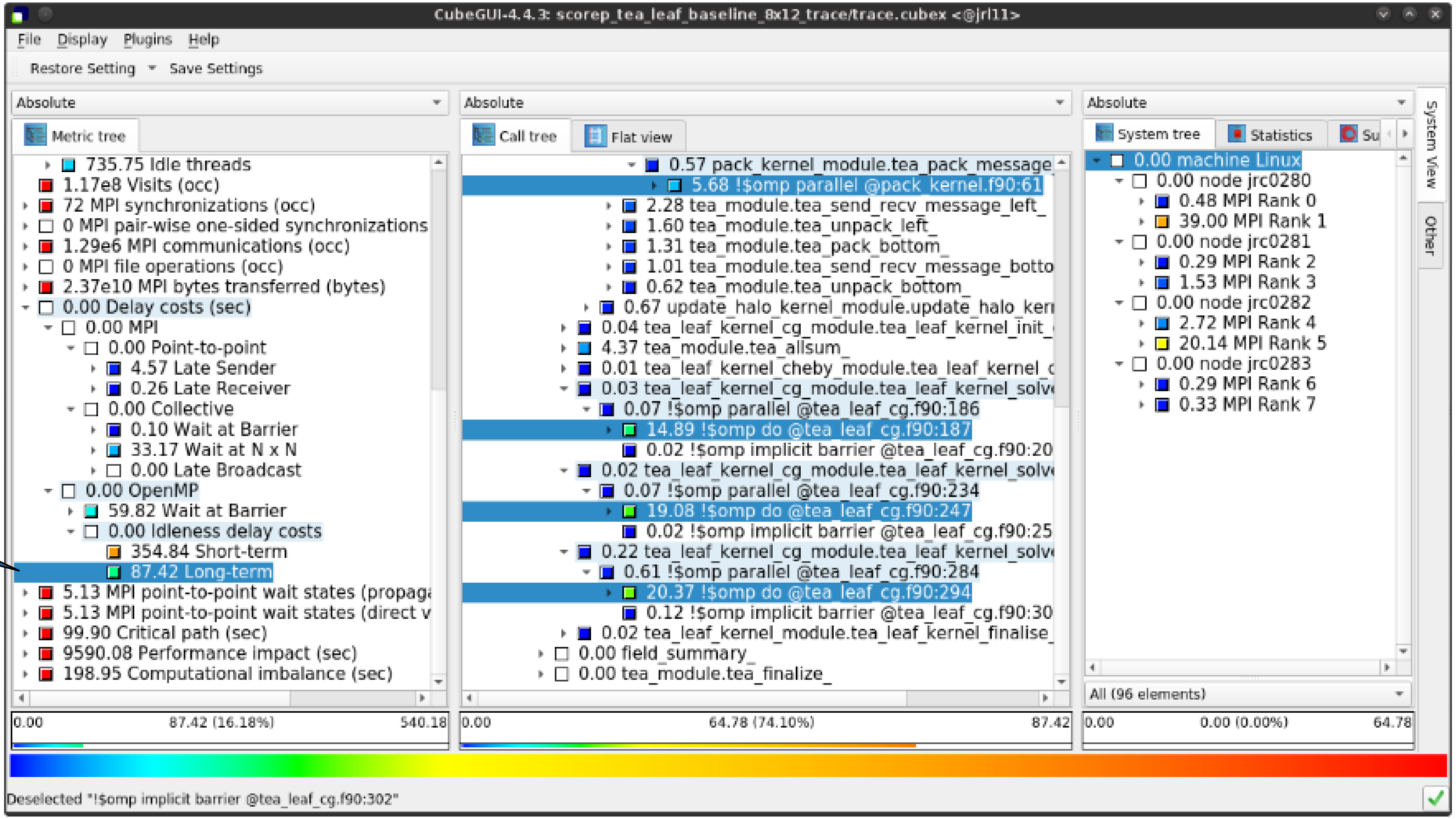
```
Metric tree:
 ▸ 735.75 Idle threads
   1.17e8 Visits (occ)
 ▸ 72 MPI synchronizations (occ)
 ▸ 0 MPI pair-wise one-sided synchronizations
 ▸ 1.29e6 MPI communications (occ)
 ▸ 0 MPI file operations (occ)
 ▸ 2.37e10 MPI bytes transferred (bytes)
 ▾ 0.00 Delay costs (sec)
   ▾ 0.00 MPI
     ▾ 0.00 Point-to-point
       ▸ 4.57 Late Sender
       ▸ 0.26 Late Receiver
     ▾ 0.00 Collective
       ▸ 0.10 Wait at Barrier
       ▸ 33.17 Wait at N x N
       ▸ 0.00 Late Broadcast
   ▾ 0.00 OpenMP
     ▸ 59.82 Wait at Barrier
     ▾ 0.00 Idleness delay costs
         354.84 Short-term
         87.42 Long-term
 ▸ 5.13 MPI point-to-point wait states (propag
 ▸ 5.13 MPI point-to-point wait states (direct v
 ▸ 99.90 Critical path (sec)
 ▸ 9590.08 Performance impact (sec)
 ▸ 198.95 Computational imbalance (sec)

0.00          4.84 (0.90%)          540.18
```

```
Call tree:
       ▾ 0.03 pack_kernel_module.tea_pack_message
         ▸ 0.39 !$omp parallel @pack_kernel.f90:246
     ▸ 0.01 tea_module.tea_send_recv_message_top_
     ▸ 0.00 tea_module.tea_unpack_top_
     ▾ 0.02 tea_module.tea_pack_left_
       ▾ 0.04 pack_kernel_module.tea_pack_message
         ▸ 0.40 !$omp parallel @pack_kernel.f90:61
     ▸ 0.08 tea_module.tea_send_recv_message_left_
     ▸ 0.04 tea_module.tea_unpack_left_
     ▸ 0.04 tea_module.tea_pack_bottom_
     ▸ 0.03 tea_module.tea_send_recv_message_botto
     ▸ 0.00 tea_module.tea_unpack_bottom_
   ▸ 0.00 update_halo_kernel_module.update_halo_kerı
 ▸ 0.00 tea_leaf_kernel_cg_module.tea_leaf_kernel_init_
 ▸ 0.26 tea_module.tea_allsum_
 ▸ 0.00 tea_leaf_kernel_cheby_module.tea_leaf_kernel_c
 ▸ 0.00 tea_leaf_kernel_cg_module.tea_leaf_kernel_solve
 ▸ 0.00 tea_leaf_kernel_cg_module.tea_leaf_kernel_solve
 ▾ 0.02 tea_leaf_kernel_cg_module.tea_leaf_kernel_solve
   ▾ 0.05 !$omp parallel @tea_leaf_cg.f90:284
     ▾ 2.91 !$omp do @tea_leaf_cg.f90:294
         0.03 !$omp implicit barrier @tea_leaf_cg.f90:
       0.01 !$omp implicit barrier @tea_leaf_cg.f90:30
 ▸ 0.00 tea_leaf_kernel_module.tea_leaf_kernel_finalise_
▸ 0.00 field_summary_
▸ 0.00 tea_module.tea_finalize_

0.00          3.70 (76.45%)          4.84
```

```
System tree:
▾ 0.00 machine Linux
  ▾ 0.00 node jrc0280
    ▸ 0.05 MPI Rank 0
    ▸ 2.67 MPI Rank 1
  ▾ 0.00 node jrc0281
    ▸ 0.02 MPI Rank 2
    ▸ 0.17 MPI Rank 3
  ▾ 0.00 node jrc0282
    ▸ 0.34 MPI Rank 4
    ▸ 0.41 MPI Rank 5
  ▾ 0.00 node jrc0283
    ▸ 0.02 MPI Rank 6
    ▸ 0.01 MPI Rank 7

All (96 elements)
0.00          0.00 (0.00%)          3.70
```

TeaLeaf Scalasca report analysis (V)

Various OpenMP do loops (incl. the solver loops) also cause OpenMP thread idleness on other ranks via propagation



CubeGUI-4.4.3: scorep_tea_leaf_baseline_8x12_trace/trace.cubex <@jrl11>

File Display Plugins Help
Restore Setting Save Settings
Absolute
Metric tree
Absolute
Call tree Flat view
Absolute
System tree Statistics Su
System View
Other

```
  735.75 Idle threads
1.17e8 Visits (occ)
72 MPI synchronizations (occ)
0 MPI pair-wise one-sided synchronizations
1.29e6 MPI communications (occ)
0 MPI file operations (occ)
2.37e10 MPI bytes transferred (bytes)
0.00 Delay costs (sec)
  0.00 MPI
    0.00 Point-to-point
      4.57 Late Sender
      0.26 Late Receiver
    0.00 Collective
      0.10 Wait at Barrier
      33.17 Wait at N x N
      0.00 Late Broadcast
  0.00 OpenMP
    59.82 Wait at Barrier
    0.00 Idleness delay costs
      354.84 Short-term
      87.42 Long-term
5.13 MPI point-to-point wait states (propaga
5.13 MPI point-to-point wait states (direct v
99.90 Critical path (sec)
9590.08 Performance impact (sec)
198.95 Computational imbalance (sec)
```

```
        0.57 pack_kernel_module.tea_pack_message
          5.68 !$omp parallel @pack_kernel.f90:61
      2.28 tea_module.tea_send_recv_message_left_
      1.60 tea_module.tea_unpack_left_
      1.31 tea_module.tea_pack_bottom_
      1.01 tea_module.tea_send_recv_message_botto
      0.62 tea_module.tea_unpack_bottom_
    0.67 update_halo_kernel_module.update_halo_kerr
  0.04 tea_leaf_kernel_cg_module.tea_leaf_kernel_init_
  4.37 tea_module.tea_allsum_
  0.01 tea_leaf_kernel_cheby_module.tea_leaf_kernel_c
  0.03 tea_leaf_kernel_cg_module.tea_leaf_kernel_solve
    0.07 !$omp parallel @tea_leaf_cg.f90:186
      14.89 !$omp do @tea_leaf_cg.f90:187
      0.02 !$omp implicit barrier @tea_leaf_cg.f90:20
  0.02 tea_leaf_kernel_cg_module.tea_leaf_kernel_solve
    0.07 !$omp parallel @tea_leaf_cg.f90:234
      19.08 !$omp do @tea_leaf_cg.f90:247
      0.02 !$omp implicit barrier @tea_leaf_cg.f90:25
  0.22 tea_leaf_kernel_cg_module.tea_leaf_kernel_solve
    0.61 !$omp parallel @tea_leaf_cg.f90:284
      20.37 !$omp do @tea_leaf_cg.f90:294
      0.12 !$omp implicit barrier @tea_leaf_cg.f90:30
  0.02 tea_leaf_kernel_module.tea_leaf_kernel_finalise_
0.00 field_summary_
0.00 tea_module.tea_finalize_
```

```
0.00 machine Linux
  0.00 node jrc0280
    0.48 MPI Rank 0
    39.00 MPI Rank 1
  0.00 node jrc0281
    0.29 MPI Rank 2
    1.53 MPI Rank 3
  0.00 node jrc0282
    2.72 MPI Rank 4
    20.14 MPI Rank 5
  0.00 node jrc0283
    0.29 MPI Rank 6
    0.33 MPI Rank 7
```

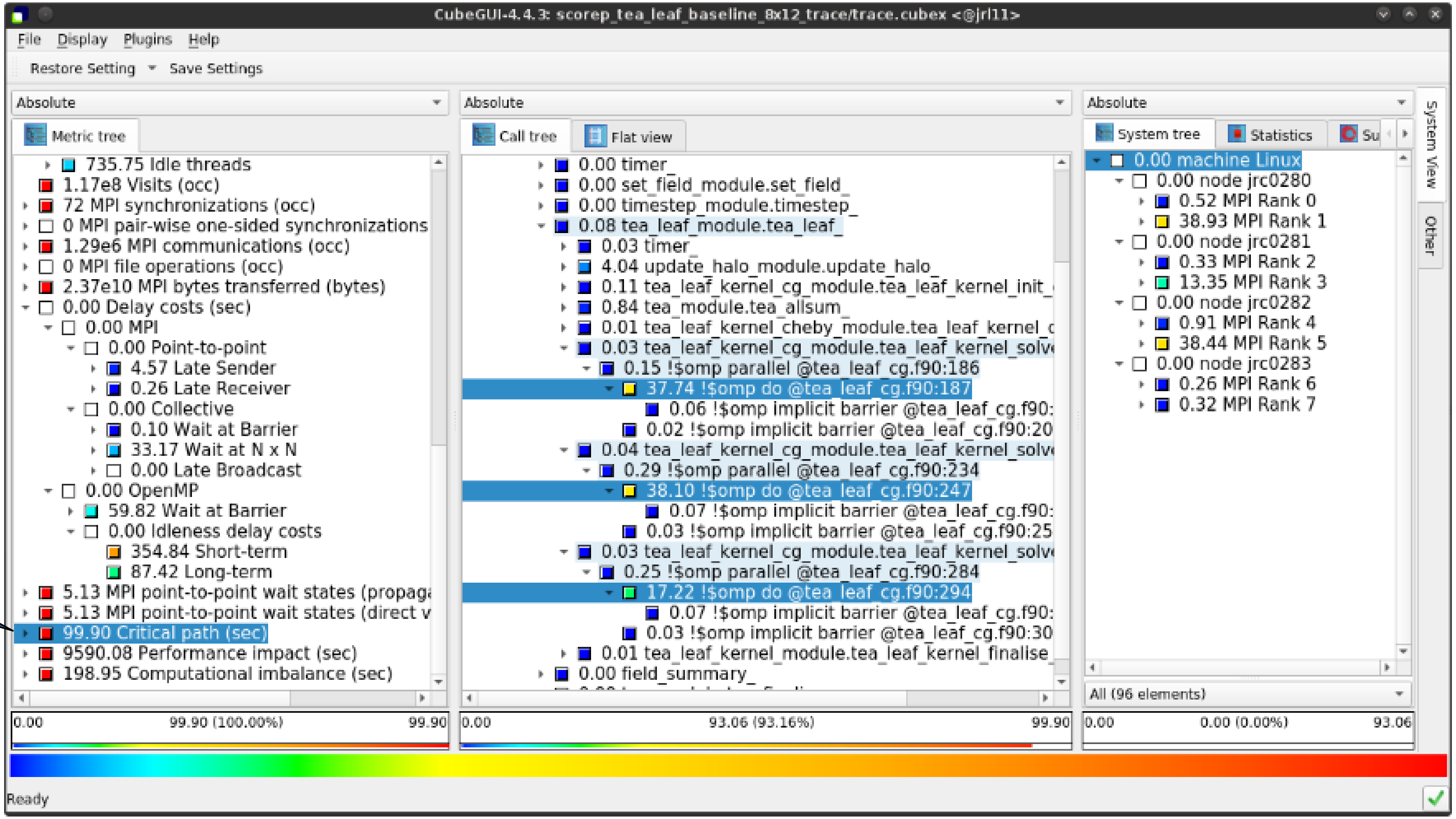
All (96 elements)

```
0.00      87.42 (16.18%)      540.18
0.00      64.78 (74.10%)      87.42
0.00      0.00 (0.00%)        64.78
```

Deselected "!$omp implicit barrier @tea_leaf_cg.f90:302"

TeaLeaf Scalasca report analysis (VI)

The Critical Path also highlights the three solver loops…



CubeGUI-4.4.3: scorep_tea_leaf_baseline_8x12_trace/trace.cubex <@jrl11>

```
  735.75 Idle threads
  1.17e8 Visits (occ)
  72 MPI synchronizations (occ)
  0 MPI pair-wise one-sided synchronizations
  1.29e6 MPI communications (occ)
  0 MPI file operations (occ)
  2.37e10 MPI bytes transferred (bytes)
  0.00 Delay costs (sec)
    0.00 MPI
      0.00 Point-to-point
        4.57 Late Sender
        0.26 Late Receiver
      0.00 Collective
        0.10 Wait at Barrier
        33.17 Wait at N x N
        0.00 Late Broadcast
    0.00 OpenMP
      59.82 Wait at Barrier
      0.00 Idleness delay costs
        354.84 Short-term
        87.42 Long-term
  5.13 MPI point-to-point wait states (propaga
  5.13 MPI point-to-point wait states (direct v
  99.90 Critical path (sec)
  9590.08 Performance impact (sec)
  198.95 Computational imbalance (sec)
```

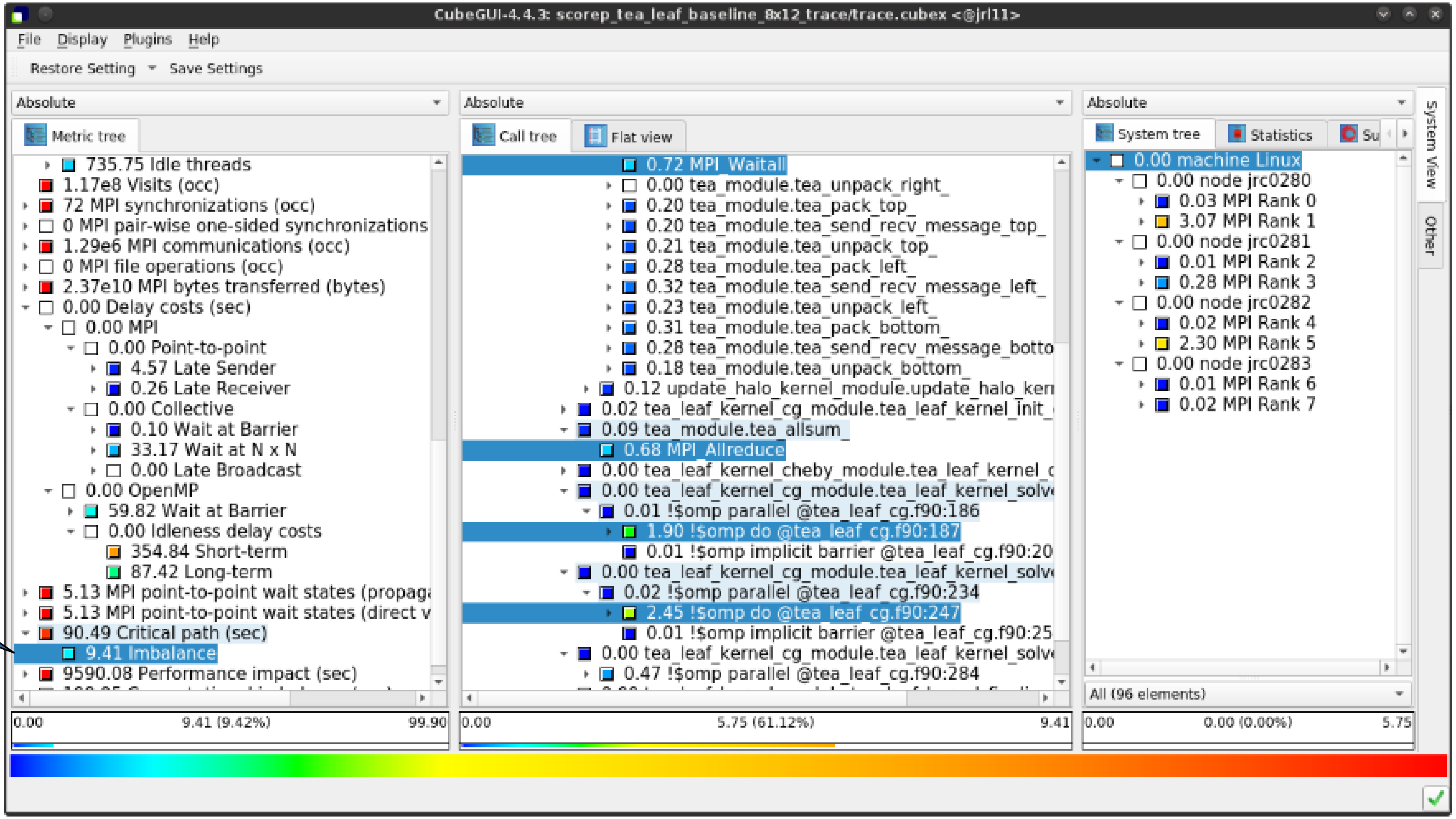
```
0.00 timer_
0.00 set_field_module.set_field_
0.00 timestep_module.timestep_
0.08 tea_leaf_module.tea_leaf_
  0.03 timer_
  4.04 update_halo_module.update_halo_
  0.11 tea_leaf_kernel_cg_module.tea_leaf_kernel_init_
  0.84 tea_module.tea_allsum_
  0.01 tea_leaf_kernel_cheby_module.tea_leaf_kernel_c
  0.03 tea_leaf_kernel_cg_module.tea_leaf_kernel_solve
    0.15 !$omp parallel @tea_leaf_cg.f90:186
      37.74 !$omp do @tea_leaf_cg.f90:187
        0.06 !$omp implicit barrier @tea_leaf_cg.f90:
      0.02 !$omp implicit barrier @tea_leaf_cg.f90:20
  0.04 tea_leaf_kernel_cg_module.tea_leaf_kernel_solve
    0.29 !$omp parallel @tea_leaf_cg.f90:234
      38.10 !$omp do @tea_leaf_cg.f90:247
        0.07 !$omp implicit barrier @tea_leaf_cg.f90:
      0.03 !$omp implicit barrier @tea_leaf_cg.f90:25
  0.03 tea_leaf_kernel_cg_module.tea_leaf_kernel_solve
    0.25 !$omp parallel @tea_leaf_cg.f90:284
      17.22 !$omp do @tea_leaf_cg.f90:294
        0.07 !$omp implicit barrier @tea_leaf_cg.f90:
      0.03 !$omp implicit barrier @tea_leaf_cg.f90:30
  0.01 tea_leaf_kernel_module.tea_leaf_kernel_finalise_
0.00 field_summary_
```

```
0.00 machine Linux
  0.00 node jrc0280
    0.52 MPI Rank 0
    38.93 MPI Rank 1
  0.00 node jrc0281
    0.33 MPI Rank 2
    13.35 MPI Rank 3
  0.00 node jrc0282
    0.91 MPI Rank 4
    38.44 MPI Rank 5
  0.00 node jrc0283
    0.26 MPI Rank 6
    0.32 MPI Rank 7
All (96 elements)
```

```
0.00    99.90 (100.00%)    99.90
0.00    93.06 (93.16%)     99.90
0.00    0.00 (0.00%)       93.06
Ready
```

TeaLeaf Scalasca report analysis (VII)

…with imbalance (time on critical path above average) mostly in the first two loops and MPI communication



CubeGUI-4.4.3: scorep_tea_leaf_baseline_8x12_trace/trace.cubex <@jrl11>

File Display Plugins Help
Restore Setting Save Settings
Absolute
Metric tree
Absolute
Call tree Flat view
Absolute
System tree Statistics Su
System View
Other

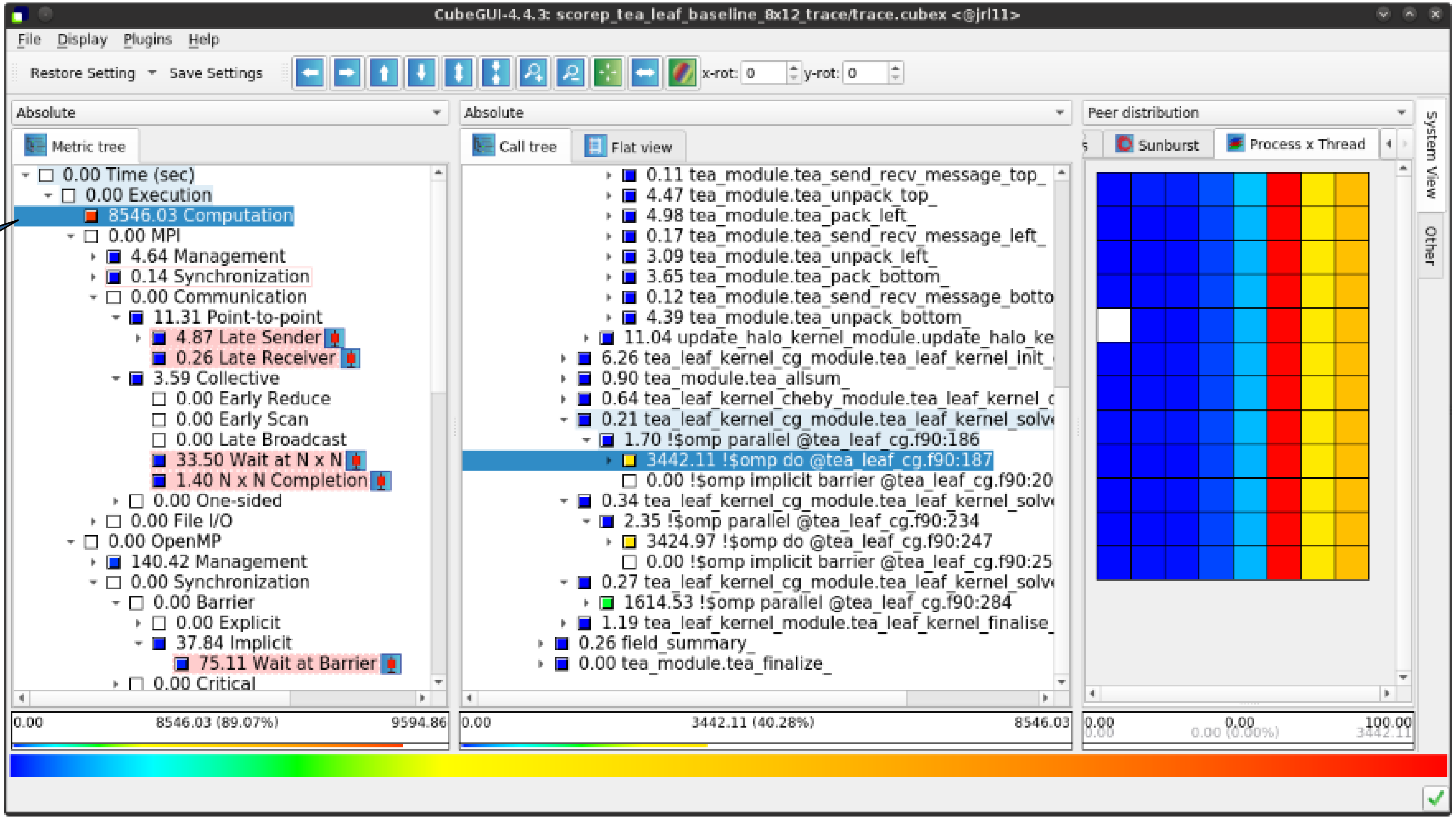
```
735.75 Idle threads
1.17e8 Visits (occ)
72 MPI synchronizations (occ)
0 MPI pair-wise one-sided synchronizations
1.29e6 MPI communications (occ)
0 MPI file operations (occ)
2.37e10 MPI bytes transferred (bytes)
0.00 Delay costs (sec)
  0.00 MPI
    0.00 Point-to-point
      4.57 Late Sender
      0.26 Late Receiver
    0.00 Collective
      0.10 Wait at Barrier
      33.17 Wait at N x N
      0.00 Late Broadcast
  0.00 OpenMP
    59.82 Wait at Barrier
    0.00 Idleness delay costs
      354.84 Short-term
      87.42 Long-term
5.13 MPI point-to-point wait states (propaga
5.13 MPI point-to-point wait states (direct v
90.49 Critical path (sec)
  9.41 Imbalance
9590.08 Performance impact (sec)

0.00          9.41 (9.42%)          99.90
```

```
    0.72 MPI_Waitall
    0.00 tea_module.tea_unpack_right_
    0.20 tea_module.tea_pack_top_
    0.20 tea_module.tea_send_recv_message_top_
    0.21 tea_module.tea_unpack_top_
    0.28 tea_module.tea_pack_left_
    0.32 tea_module.tea_send_recv_message_left_
    0.23 tea_module.tea_unpack_left_
    0.31 tea_module.tea_pack_bottom_
    0.28 tea_module.tea_send_recv_message_botto
    0.18 tea_module.tea_unpack_bottom_
  0.12 update_halo_kernel_module.update_halo_kerı
0.02 tea_leaf_kernel_cg_module.tea_leaf_kernel_init_
0.09 tea_module.tea_allsum_
  0.68 MPI_Allreduce
0.00 tea_leaf_kernel_cheby_module.tea_leaf_kernel_c
0.00 tea_leaf_kernel_cg_module.tea_leaf_kernel_solve
  0.01 !$omp parallel @tea_leaf_cg.f90:186
    1.90 !$omp do @tea_leaf_cg.f90:187
    0.01 !$omp implicit barrier @tea_leaf_cg.f90:20
0.00 tea_leaf_kernel_cg_module.tea_leaf_kernel_solve
  0.02 !$omp parallel @tea_leaf_cg.f90:234
    2.45 !$omp do @tea_leaf_cg.f90:247
    0.01 !$omp implicit barrier @tea_leaf_cg.f90:25
0.00 tea_leaf_kernel_cg_module.tea_leaf_kernel_solve
  0.47 !$omp parallel @tea_leaf_cg.f90:284

0.00          5.75 (61.12%)          9.41
```

```
0.00 machine Linux
  0.00 node jrc0280
    0.03 MPI Rank 0
    3.07 MPI Rank 1
  0.00 node jrc0281
    0.01 MPI Rank 2
    0.28 MPI Rank 3
  0.00 node jrc0282
    0.02 MPI Rank 4
    2.30 MPI Rank 5
  0.00 node jrc0283
    0.01 MPI Rank 6
    0.02 MPI Rank 7

All (96 elements)
0.00          0.00 (0.00%)          5.75
```

TeaLeaf Scalasca report analysis (VIII)

Computation time of 1st…



CubeGUI-4.4.3: scorep_tea_leaf_baseline_8x12_trace/trace.cubex <@jrl11>
File Display Plugins Help
Restore Setting Save Settings
x-rot: 0 y-rot: 0
Absolute
Absolute
Peer distribution
Metric tree
Call tree
Flat view
Sunburst
Process x Thread
System View
Other

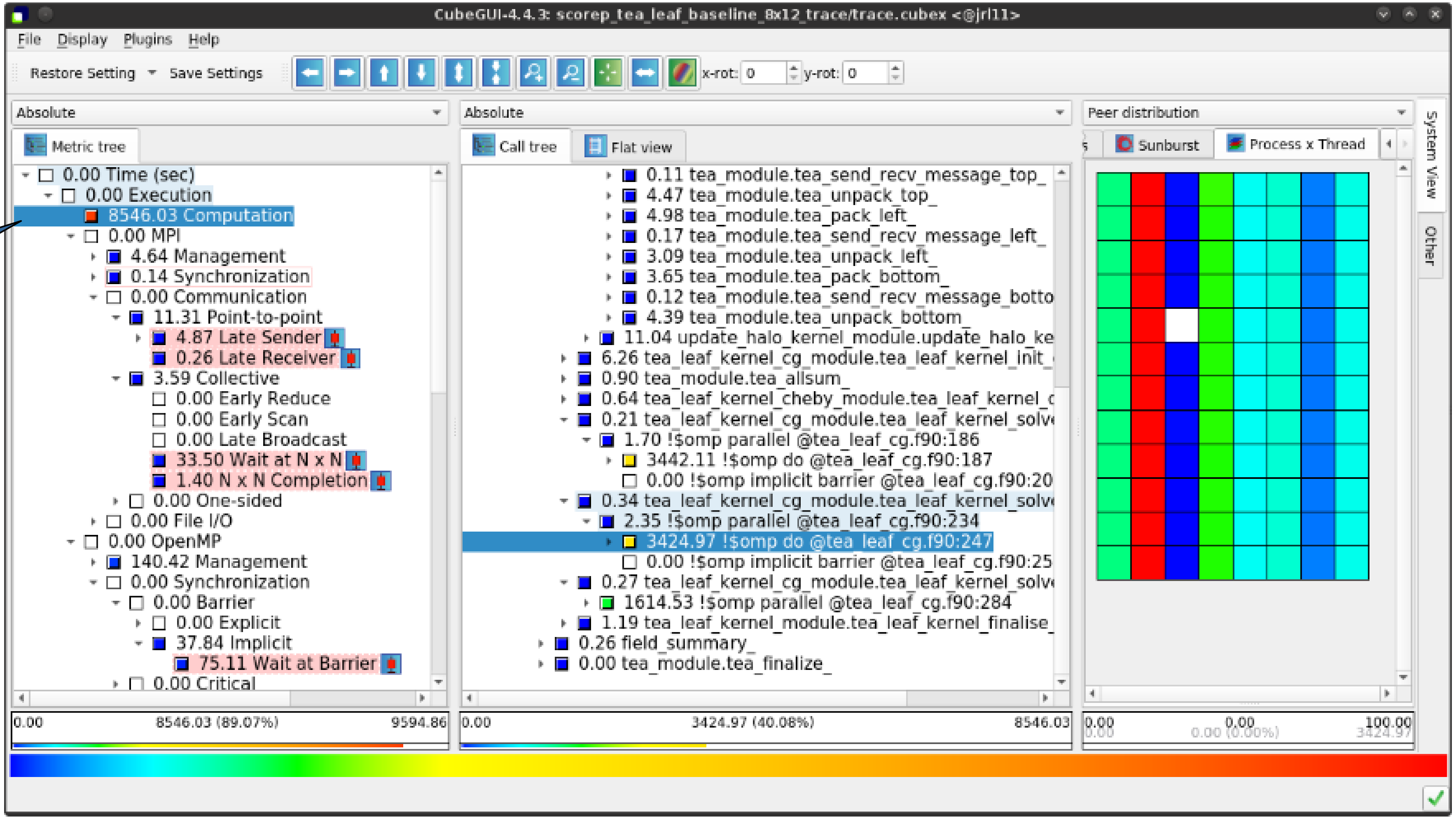
```
0.00 Time (sec)
  0.00 Execution
    8546.03 Computation
    0.00 MPI
      4.64 Management
      0.14 Synchronization
      0.00 Communication
        11.31 Point-to-point
          4.87 Late Sender
          0.26 Late Receiver
        3.59 Collective
          0.00 Early Reduce
          0.00 Early Scan
          0.00 Late Broadcast
          33.50 Wait at N x N
          1.40 N x N Completion
        0.00 One-sided
      0.00 File I/O
    0.00 OpenMP
      140.42 Management
      0.00 Synchronization
        0.00 Barrier
          0.00 Explicit
          37.84 Implicit
            75.11 Wait at Barrier
        0.00 Critical
```

```
      0.11 tea_module.tea_send_recv_message_top_
      4.47 tea_module.tea_unpack_top_
      4.98 tea_module.tea_pack_left_
      0.17 tea_module.tea_send_recv_message_left_
      3.09 tea_module.tea_unpack_left_
      3.65 tea_module.tea_pack_bottom_
      0.12 tea_module.tea_send_recv_message_botto
      4.39 tea_module.tea_unpack_bottom_
    11.04 update_halo_kernel_module.update_halo_ke
  6.26 tea_leaf_kernel_cg_module.tea_leaf_kernel_init_
  0.90 tea_module.tea_allsum_
  0.64 tea_leaf_kernel_cheby_module.tea_leaf_kernel_c
  0.21 tea_leaf_kernel_cg_module.tea_leaf_kernel_solve
    1.70 !$omp parallel @tea_leaf_cg.f90:186
      3442.11 !$omp do @tea_leaf_cg.f90:187
      0.00 !$omp implicit barrier @tea_leaf_cg.f90:20
  0.34 tea_leaf_kernel_cg_module.tea_leaf_kernel_solve
    2.35 !$omp parallel @tea_leaf_cg.f90:234
      3424.97 !$omp do @tea_leaf_cg.f90:247
      0.00 !$omp implicit barrier @tea_leaf_cg.f90:25
  0.27 tea_leaf_kernel_cg_module.tea_leaf_kernel_solve
    1614.53 !$omp parallel @tea_leaf_cg.f90:284
  1.19 tea_leaf_kernel_module.tea_leaf_kernel_finalise_
0.26 field_summary_
0.00 tea_module.tea_finalize_
```

```
0.00        8546.03 (89.07%)        9594.86
0.00        3442.11 (40.28%)        8546.03
0.00  0.00  0.00 (0.00%)  100.00  3442.11
```

TeaLeaf Scalasca report analysis (IX)

…and 2nd `do` loop mostly balanced within each rank, but vary considerably across ranks…

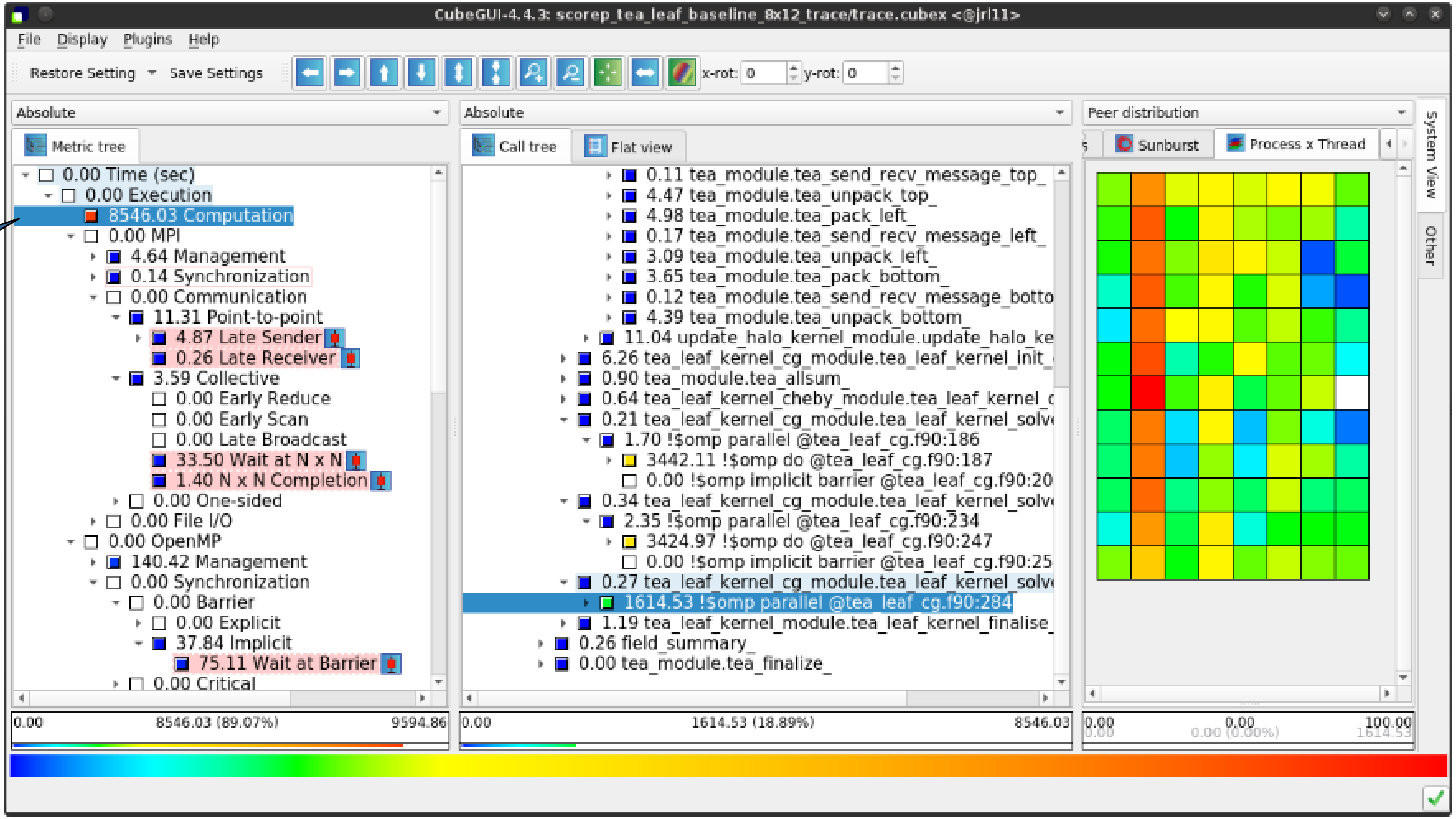


```
▾ □ 0.00 Time (sec)
  ▾ □ 0.00 Execution
      ■ 8546.03 Computation
    ▾ □ 0.00 MPI
      ▸ ■ 4.64 Management
      ▸ ■ 0.14 Synchronization
      ▾ □ 0.00 Communication
        ▾ ■ 11.31 Point-to-point
          ▸ ■ 4.87 Late Sender
            ■ 0.26 Late Receiver
        ▾ ■ 3.59 Collective
            □ 0.00 Early Reduce
            □ 0.00 Early Scan
            □ 0.00 Late Broadcast
            ■ 33.50 Wait at N x N
            ■ 1.40 N x N Completion
        ▸ □ 0.00 One-sided
      ▸ □ 0.00 File I/O
    ▾ □ 0.00 OpenMP
      ▸ ■ 140.42 Management
      ▾ □ 0.00 Synchronization
        ▾ □ 0.00 Barrier
          ▸ □ 0.00 Explicit
          ▾ ■ 37.84 Implicit
              ■ 75.11 Wait at Barrier
        ▸ □ 0.00 Critical
```

```
        ▸ ■ 0.11 tea_module.tea_send_recv_message_top_
        ▸ ■ 4.47 tea_module.tea_unpack_top_
        ▸ ■ 4.98 tea_module.tea_pack_left_
        ▸ ■ 0.17 tea_module.tea_send_recv_message_left_
        ▸ ■ 3.09 tea_module.tea_unpack_left_
        ▸ ■ 3.65 tea_module.tea_pack_bottom_
        ▸ ■ 0.12 tea_module.tea_send_recv_message_botto
        ▸ ■ 4.39 tea_module.tea_unpack_bottom_
      ▸ ■ 11.04 update_halo_kernel_module.update_halo_ke
    ▸ ■ 6.26 tea_leaf_kernel_cg_module.tea_leaf_kernel_init_
    ▸ ■ 0.90 tea_module.tea_allsum_
    ▸ ■ 0.64 tea_leaf_kernel_cheby_module.tea_leaf_kernel_c
    ▾ ■ 0.21 tea_leaf_kernel_cg_module.tea_leaf_kernel_solve
      ▾ ■ 1.70 !$omp parallel @tea_leaf_cg.f90:186
        ▸ ■ 3442.11 !$omp do @tea_leaf_cg.f90:187
          □ 0.00 !$omp implicit barrier @tea_leaf_cg.f90:20
    ▾ ■ 0.34 tea_leaf_kernel_cg_module.tea_leaf_kernel_solve
      ▾ ■ 2.35 !$omp parallel @tea_leaf_cg.f90:234
        ▸ ■ 3424.97 !$omp do @tea_leaf_cg.f90:247
          □ 0.00 !$omp implicit barrier @tea_leaf_cg.f90:25
    ▾ ■ 0.27 tea_leaf_kernel_cg_module.tea_leaf_kernel_solve
      ▸ ■ 1614.53 !$omp parallel @tea_leaf_cg.f90:284
    ▸ ■ 1.19 tea_leaf_kernel_module.tea_leaf_kernel_finalise_
  ▸ ■ 0.26 field_summary_
  ▸ ■ 0.00 tea_module.tea_finalize_
```

```
0.00    8546.03 (89.07%)    9594.86
0.00    3424.97 (40.08%)    8546.03
0.00 0.00    0.00 0.00 (0.00%)    100.00 3424.97
```

TeaLeaf Scalasca report analysis (X)



CubeGUI-4.4.3: scorep_tea_leaf_baseline_8x12_trace/trace.cubex <@jrl11>

```
□ 0.00 Time (sec)
  □ 0.00 Execution
    ■ 8546.03 Computation
    □ 0.00 MPI
      ■ 4.64 Management
      ■ 0.14 Synchronization
      □ 0.00 Communication
        ■ 11.31 Point-to-point
          ■ 4.87 Late Sender
          ■ 0.26 Late Receiver
        ■ 3.59 Collective
          □ 0.00 Early Reduce
          □ 0.00 Early Scan
          □ 0.00 Late Broadcast
          ■ 33.50 Wait at N x N
          ■ 1.40 N x N Completion
        □ 0.00 One-sided
      □ 0.00 File I/O
    □ 0.00 OpenMP
      ■ 140.42 Management
      □ 0.00 Synchronization
        □ 0.00 Barrier
          □ 0.00 Explicit
          ■ 37.84 Implicit
            ■ 75.11 Wait at Barrier
        □ 0.00 Critical
```

```
      ■ 0.11 tea_module.tea_send_recv_message_top_
      ■ 4.47 tea_module.tea_unpack_top_
      ■ 4.98 tea_module.tea_pack_left_
      ■ 0.17 tea_module.tea_send_recv_message_left_
      ■ 3.09 tea_module.tea_unpack_left_
      ■ 3.65 tea_module.tea_pack_bottom_
      ■ 0.12 tea_module.tea_send_recv_message_botto
      ■ 4.39 tea_module.tea_unpack_bottom_
    ■ 11.04 update_halo_kernel_module.update_halo_ke
  ■ 6.26 tea_leaf_kernel_cg_module.tea_leaf_kernel_init_
  ■ 0.90 tea_module.tea_allsum_
  ■ 0.64 tea_leaf_kernel_cheby_module.tea_leaf_kernel_c
  ■ 0.21 tea_leaf_kernel_cg_module.tea_leaf_kernel_solve
    ■ 1.70 !$omp parallel @tea_leaf_cg.f90:186
      ■ 3442.11 !$omp do @tea_leaf_cg.f90:187
      □ 0.00 !$omp implicit barrier @tea_leaf_cg.f90:20
  ■ 0.34 tea_leaf_kernel_cg_module.tea_leaf_kernel_solve
    ■ 2.35 !$omp parallel @tea_leaf_cg.f90:234
      ■ 3424.97 !$omp do @tea_leaf_cg.f90:247
      □ 0.00 !$omp implicit barrier @tea_leaf_cg.f90:25
  ■ 0.27 tea_leaf_kernel_cg_module.tea_leaf_kernel_solve
    ■ 1614.53 !$omp parallel @tea_leaf_cg.f90:284
  ■ 1.19 tea_leaf_kernel_module.tea_leaf_kernel_finalise_
■ 0.26 field_summary_
■ 0.00 tea_module.tea_finalize_
```

```
0.00        8546.03 (89.07%)        9594.86
0.00        1614.53 (18.89%)        8546.03
0.00        0.00                    100.00
0.00        0.00 (0.00%)            1614.53
```

…while the 3rd `do` loop also shows imbalance within each rank

TeaLeaf analysis summary

- The first two OpenMP do loops of the solver are well balanced within a rank, but are imbalanced across ranks
  - ➔ Requires a global load balancing strategy
- The third OpenMP do loop, however, is imbalanced within ranks,
  - causing direct "Wait at OpenMP Barrier" wait states,
  - which cause indirect MPI point-to-point wait states,
  - which in turn cause OpenMP thread idleness
  - ➔ Low-hanging fruit

- Adding a `SCHEDULE(guided)` clause reduced
  - the MPI point-to-point wait states by ~66%
  - the MPI collective wait states by ~50%
  - the OpenMP "Wait at Barrier" wait states by ~55%
  - the OpenMP thread idleness by ~11%
  - ➔ **Overall runtime (wall-clock) reduction by ~5%**

Scalasca Trace Tools: Further information

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

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

Exercises
(if you don’t have your own code)

trace tools

scalasca

Warm-up

- Build the BT-MZ example code for class (i.e., problem size) “D”
  - Perform a baseline measurement w/o instrumentation (should run in ~190s)
  - Re-build the executable with Score-P instrumentation

- Repeat the hands-on exercise with the new executable
  - Perform a summary measurement
  - Score the summary measurement result
  - Adjust the measurement configuration appropriately
  - Perform a trace measurement and analysis

Trace analysis report examination

- What is the poportion of computation time vs. parallelization overheads?

- Which code regions are mostly responsible for the overall execution time?

- Are there any load balancing issues?
- If so, in which routines?

- What are the most significant wait states/parallelization overheads?
- What are their root causes?

Optimization

- What are possible optimizations?
  - Hint: Take a look at the TeaLeaf case study

- Modify the source code to apply those and re-do the measurement
  - Don’t worry – it’s straightforward even if you don’t know the code ;-)
  - Remember: One step at a time!

- How did the performance change?
  - The `cube_diff` tool or the “Cube Diff” plugin of the GUI (see the File → Context-free plugin menu) may come in handy here