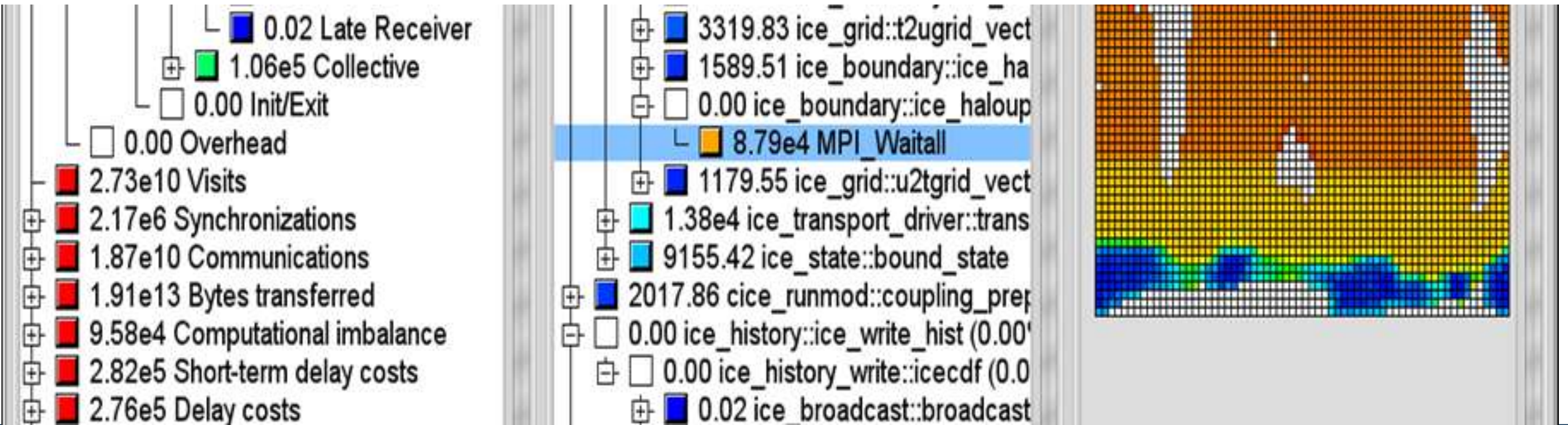




COMPUTING THE POP METRICS WITH SCORE-P, SCALASCA, AND CUBE

AIXCELERATE 2022 | DEC. 05, 2022 | MICHAEL KNOBLOCH | M.KNOBLOCH@FZ-JUELICH.DE



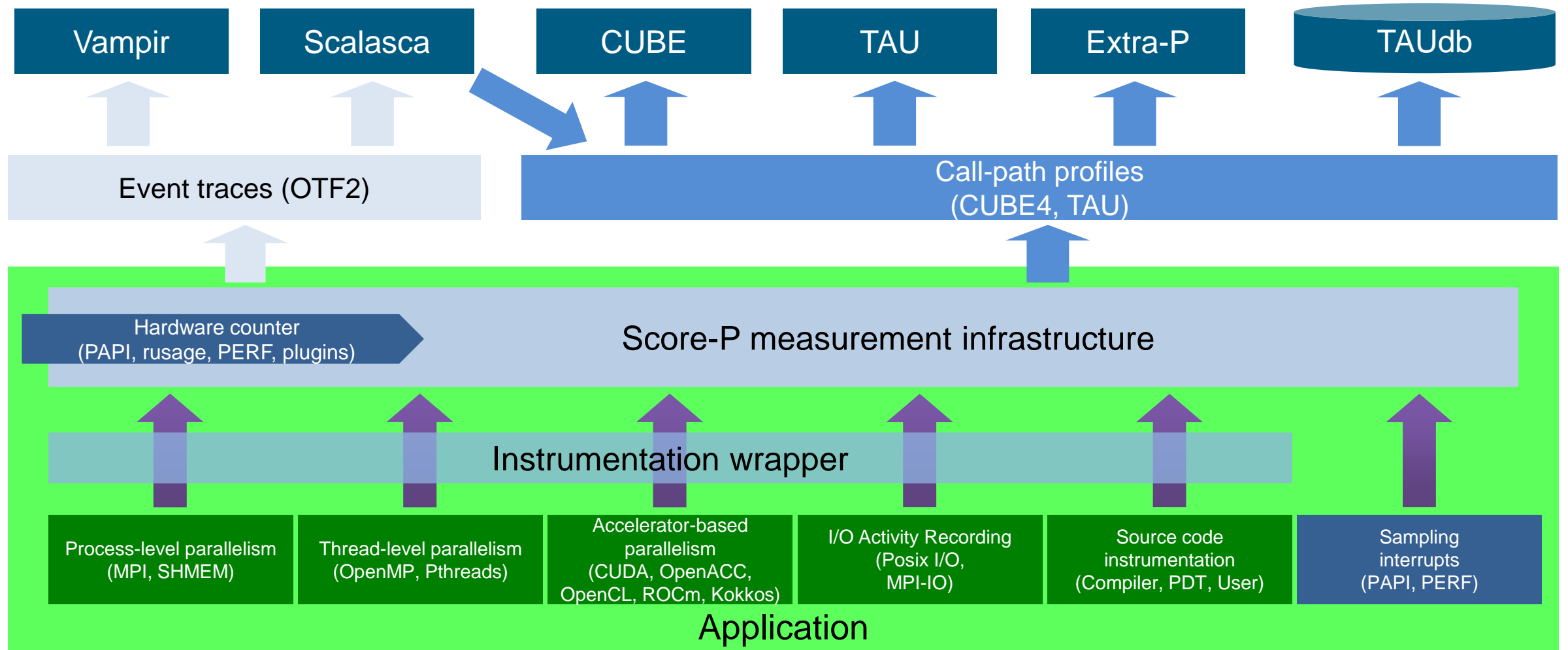
SCORE-P AND SCALASCA

SCORE-P

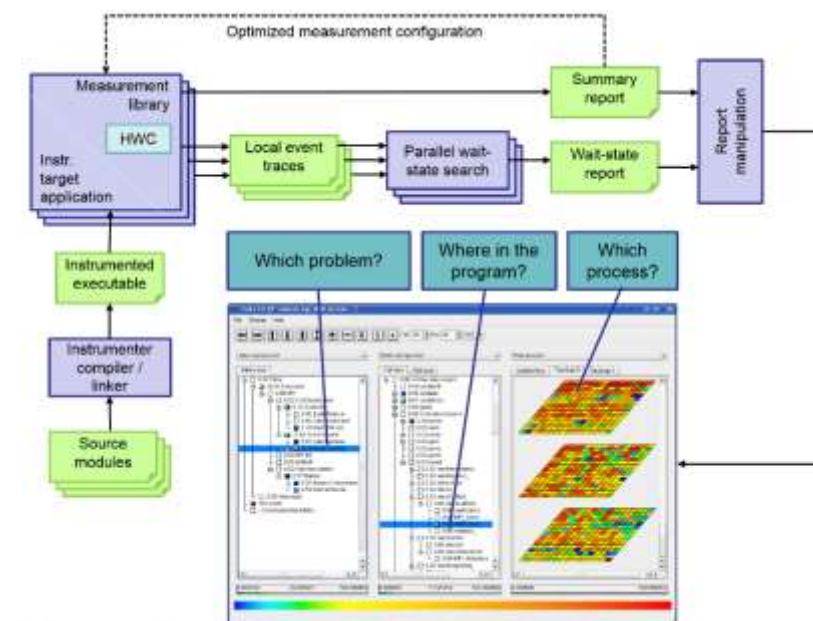
- Infrastructure for instrumentation and performance measurements
- Instrumented application can be used to produce several results:
 - Call-path profiling: CUBE4 data format used for data exchange
 - Event-based tracing: OTF2 data format used for data exchange
- Supported parallel paradigms:
 - Multi-process: MPI, SHMEM
 - Thread-parallel: OpenMP, Pthreads
 - Accelerator-based: CUDA, OpenCL, OpenACC, ROCm, Kokkos
- Open Source; portable and scalable to all major HPC systems
- Initial project funded by BMBF
- Further developed in multiple 3rd-party funded projects



SCORE-P OVERVIEW

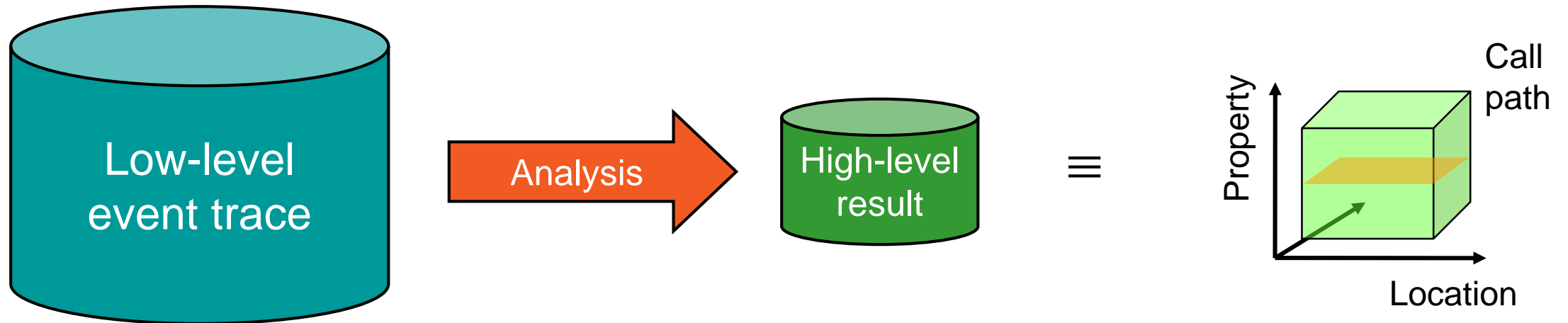


- Scalable Analysis of Large Scale Applications
- Approach
 - Instrument C, C++, and Fortran parallel applications (with Score-P)
 - Option 1: scalable call-path profiling
 - Option 2: scalable event trace analysis
 - Collect event traces
 - Process trace in parallel
 - Wait-state analysis
 - Delay and root-cause analysis
 - Critical path analysis
 - Categorize and rank results



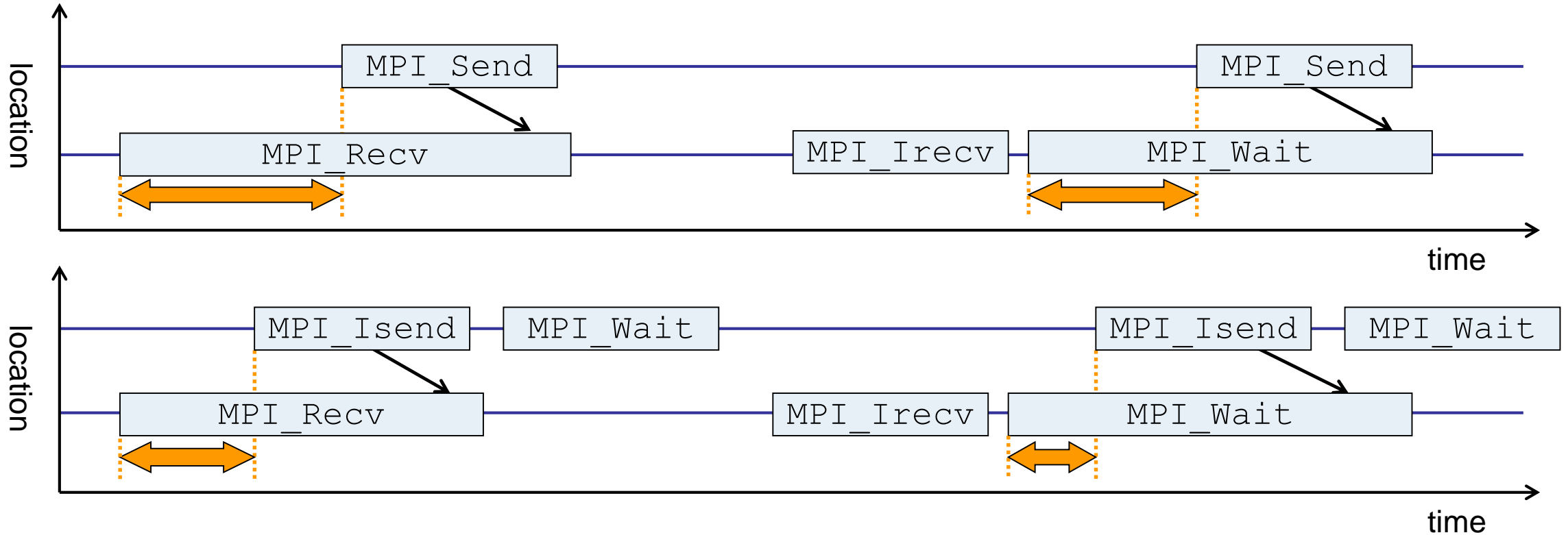
AUTOMATIC TRACE ANALYSIS

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



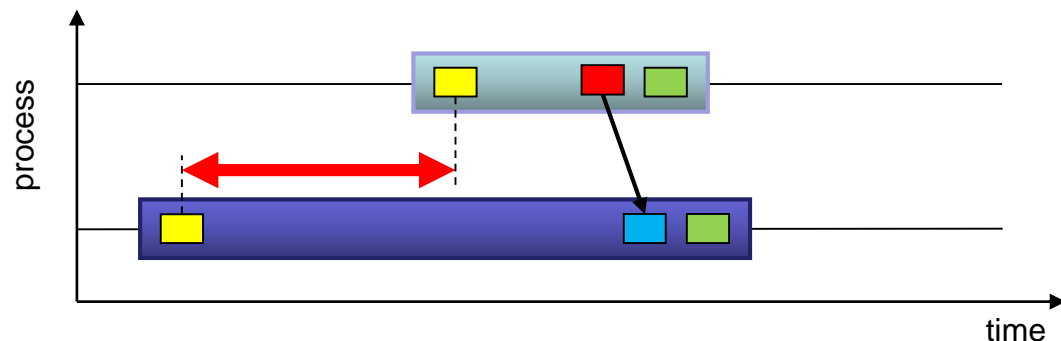
- Guaranteed to cover the entire event trace
- Quicker than manual/visual trace analysis
- Parallel replay analysis exploits available memory & processors to deliver scalability

EXAMPLE: “*LATE SENDER*” WAIT STATE

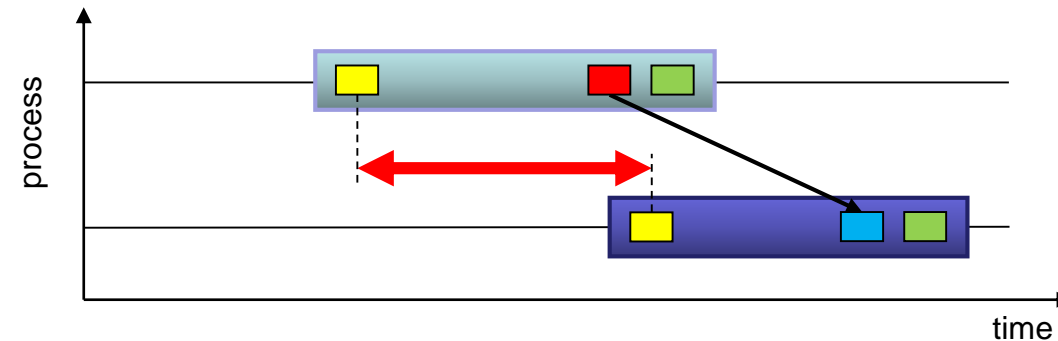


- Waiting time caused by a blocking receive operation posted earlier than the corresponding send
- Applies to blocking as well as non-blocking communication

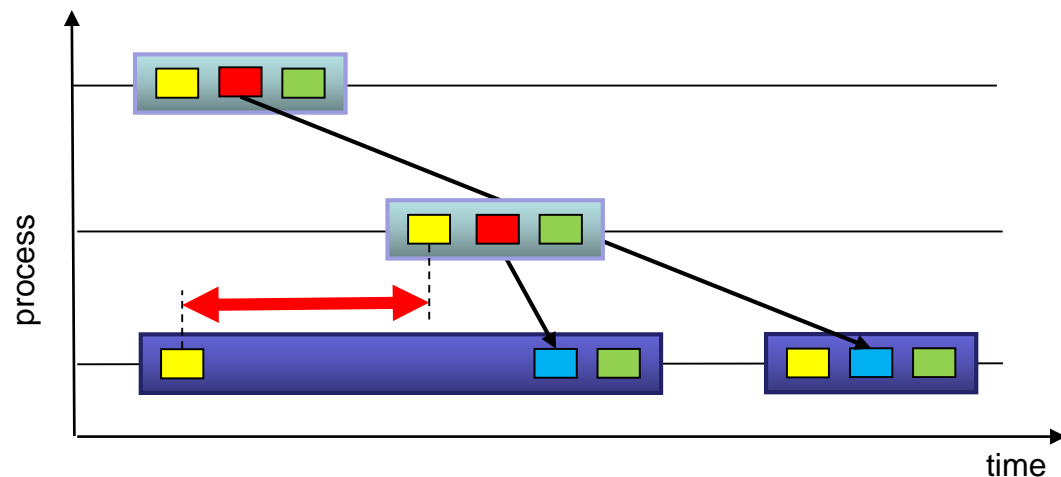
EXAMPLE MPI WAIT STATES



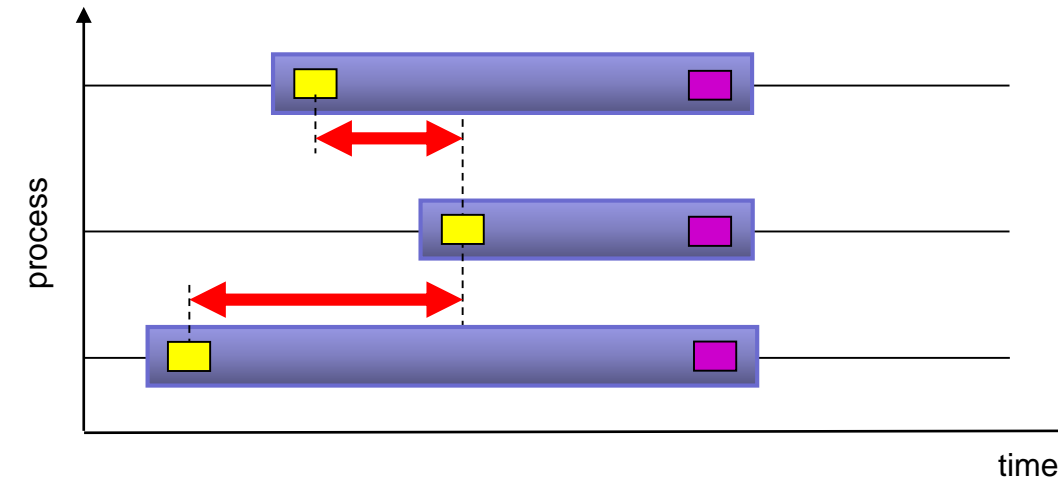
(a) Late Sender



(b) Late Receiver



(c) Late Sender / Wrong Order



(d) Wait at N x N

ENTER EXIT SEND RECV COLLEXIT

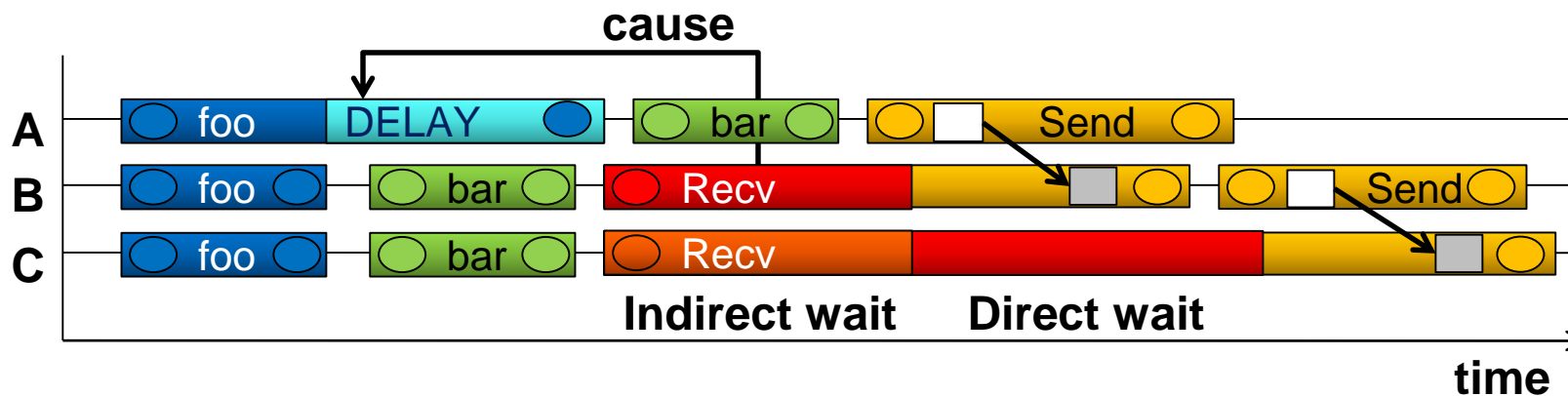
EXAMPLE: ROOT CAUSE ANALYSIS

- **Root-cause analysis**

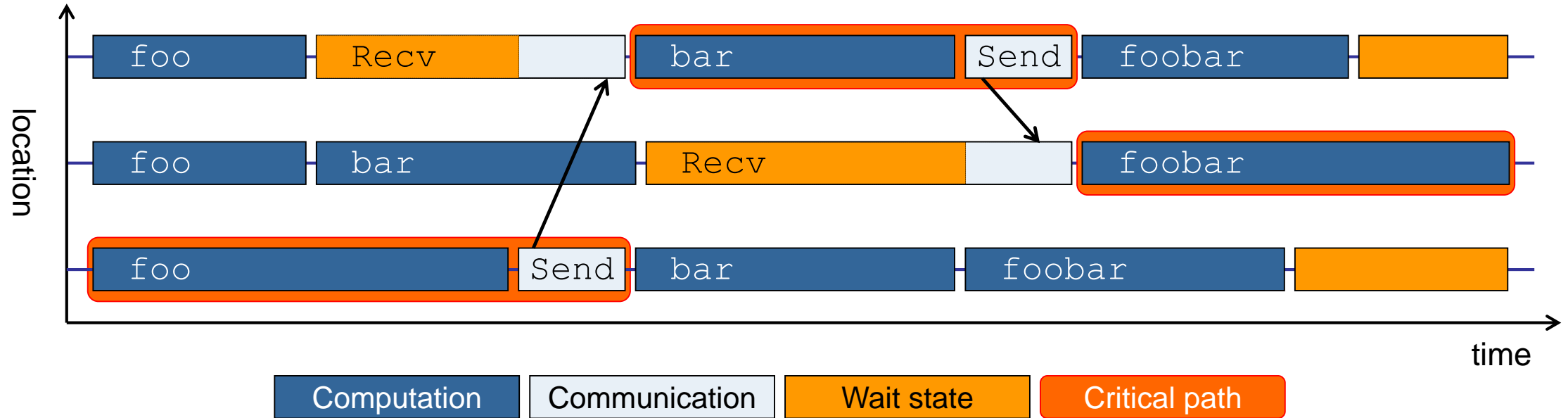
- Wait states typically caused by load or communication imbalances earlier in the program
- Waiting time can also propagate (e.g., indirect waiting time)
- Enhanced performance analysis to find the root cause of wait states

- **Approach**

- Distinguish between direct and indirect waiting time
- Identify call path/process combinations delaying other processes and causing first order waiting time
- Identify original **delay**



EXAMPLE: CRITICAL PATH



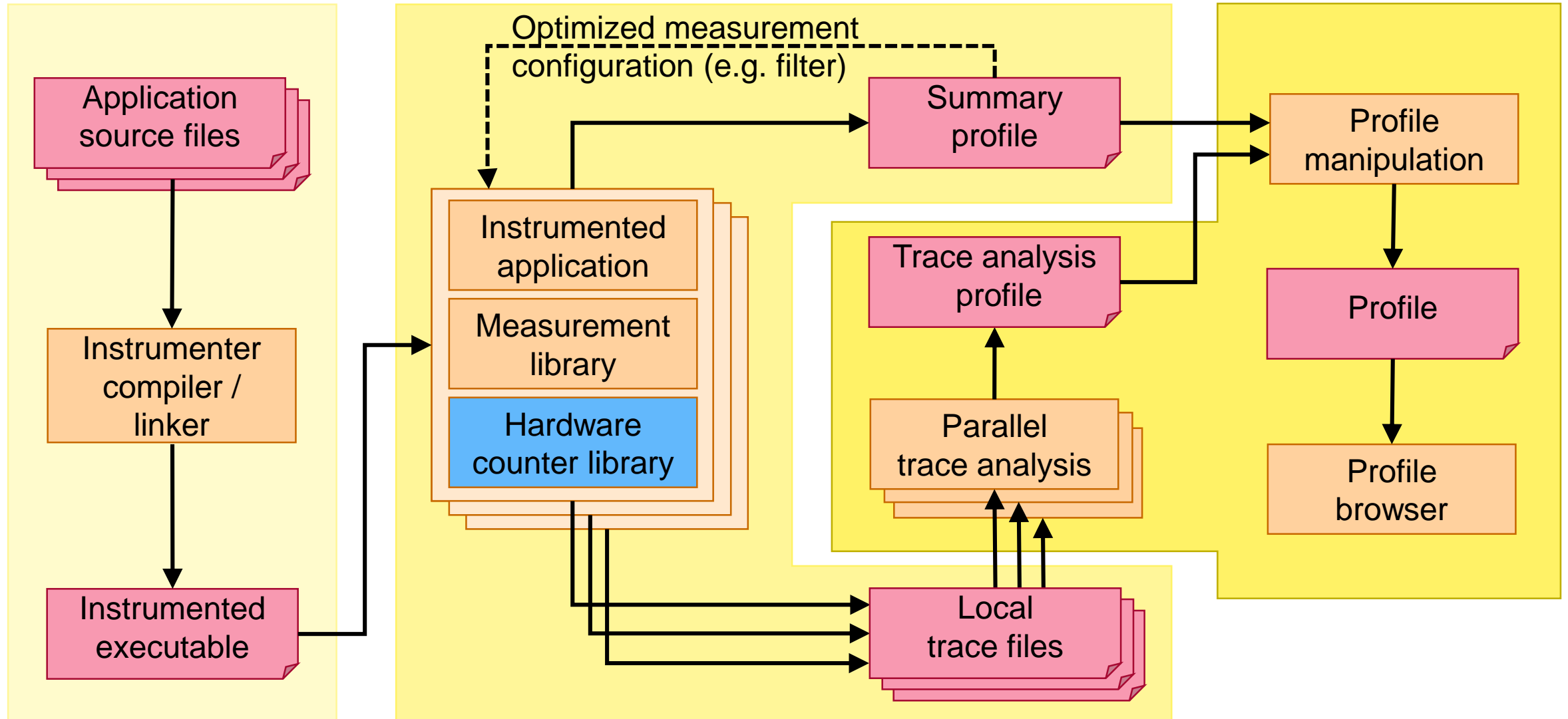
- Shows call paths and processes/threads that are responsible for the program's wall-clock runtime
- Identifies good optimization candidates and parallelization bottlenecks

PERFORMANCE ANALYSIS WORKFLOW

1. Instrumentation

2. Measurement

3. Analysis

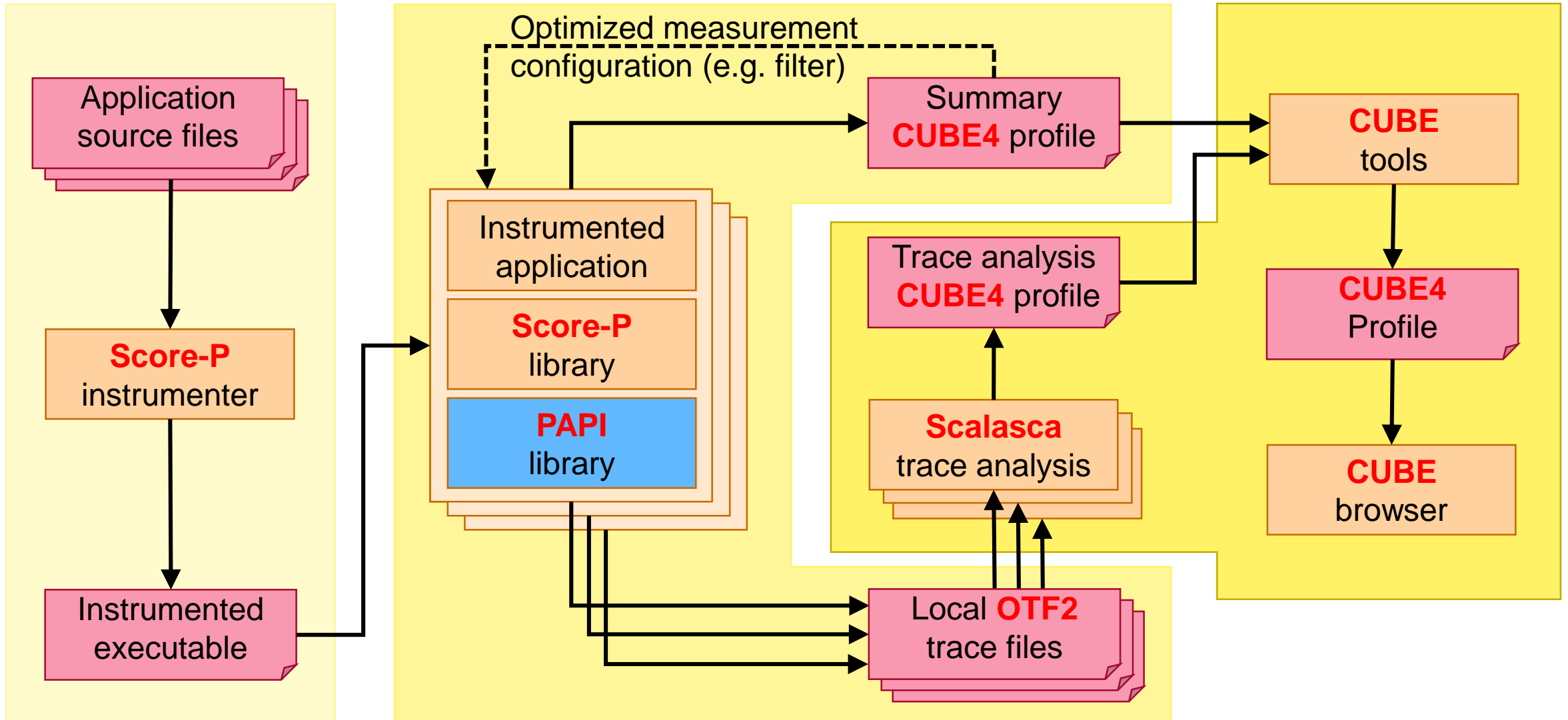


PERFORMANCE ANALYSIS WORKFLOW

1. Instrumentation

2. Measurement

3. Analysis

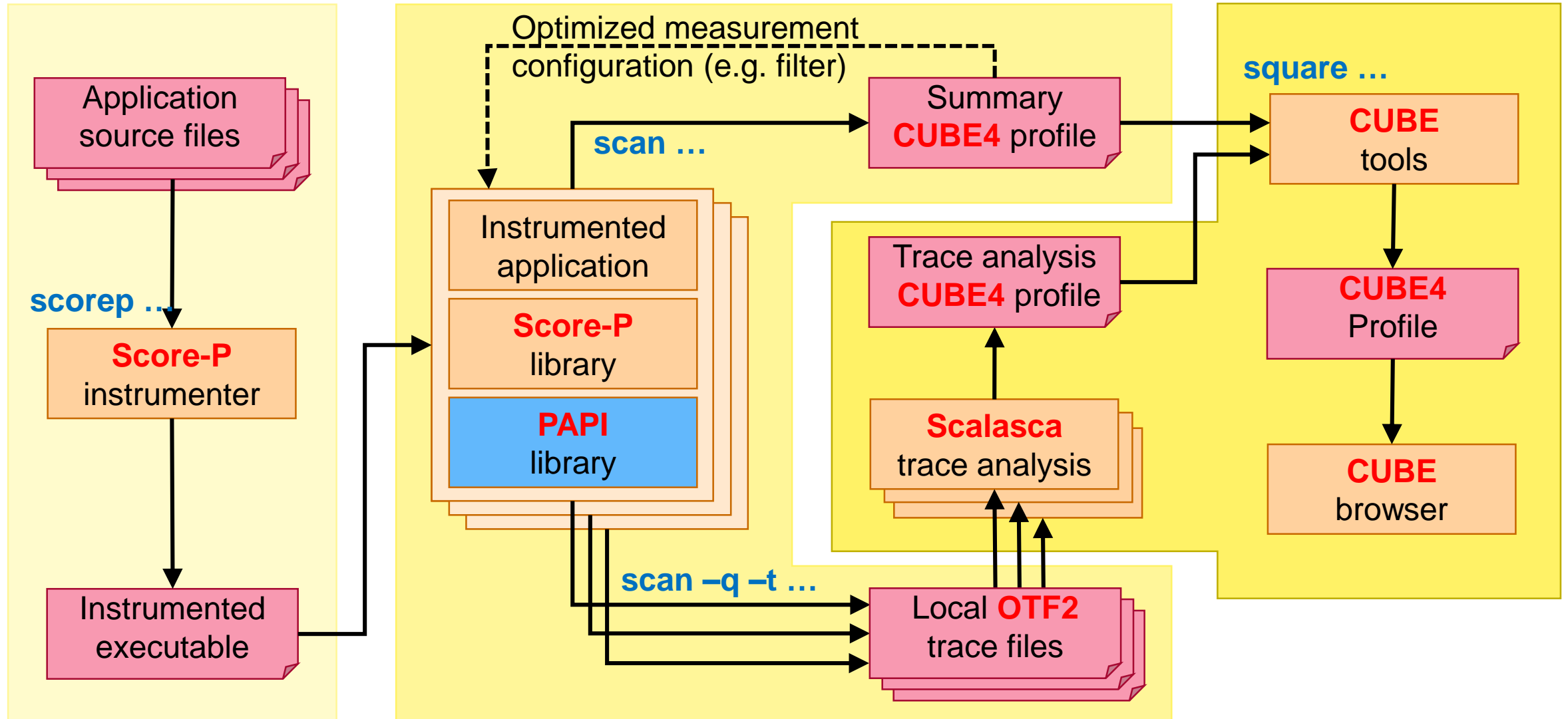


PERFORMANCE ANALYSIS WORKFLOW

1. Instrumentation

2. Measurement

3. Analysis



DEMO

- Measurement of simple Jacobi solver
 - Solves Poisson equation on rectangular grid assuming
 - Uniform discretization in each direction
 - Dirichlet boundary conditions
- Available in multiple variants (shipped with Score-P)
 - C, C++ or Fortran source code
 - MPI, OpenMP, or hybrid (MPI+OpenMP)

DEMO: BASE RUN OF APPLICATION

Notes

- Compile application
- Execute application with 2 threads on 2 processes
- Write down execution time for later comparison

```
openSUSE-Leap-15-1
zam310:~/jacobi/hybrid/C [1] make CC=mpicc CFLAGS=-fopenmp
mpicc -fopenmp -c jacobi.c
mpicc -fopenmp -c main.c
mpicc -fopenmp -o jacobi jacobi.o main.o -lm
zam310:~/jacobi/hybrid/C [2] export OMP_NUM_THREADS=2
zam310:~/jacobi/hybrid/C [3] mpiexec -np 2 ./jacobi
Jacobi 2 MPI-3.1#1 process(es) with 2 OpenMP-201511 thread(s)/process

-> matrix size: 2000x2000
-> alpha: 0.800000
-> relax: 1.000000
-> tolerance: 0.000000
-> iterations: 100

Number of iterations : 100
Residual             : 5.955111e-10
Solution Error       : 0.000266483315
Elapsed Time         : 3.2385783
MFlops               : 1602.433142
zam310:~/jacobi/hybrid/C [4] |
```

DEMO: INSTRUMENT + PROFILE

Notes

- Make sure tools are in \$PATH
- Instrument: prepend scorep
- Measure profile: prepend scan
- Compare execution time to check overhead

```
openSUSE-Leap-15-1
zam310:~/jacobi/hybrid/C [4] make clean
rm -f jacobi jacobi.o main.o
zam310:~/jacobi/hybrid/C [5] export PATH=/opt/local/ScoreP-6.0/bin:/opt/local/Scalasca-2.5/bin:/opt/local/Cube-4.5/bin:${PATH}
zam310:~/jacobi/hybrid/C [6] make CC="scorep mpicc" CFLAGS=-fopenmp
scorep mpicc -fopenmp -c jacobi.c
scorep mpicc -fopenmp -c main.c
scorep mpicc -fopenmp -o jacobi jacobi.o main.o -lm
zam310:~/jacobi/hybrid/C [7] scan mpiexec -np 2 ./jacobi
S=C=A=N: Scalasca 2.5 runtime summarization
S=C=A=N: ./scorep_jacobi_2x2_sum experiment archive
S=C=A=N: Mon May 25 16:28:55 2020: Collect start
/opt/local/easybuild-4.1.1/software/OpenMPI/3.1.4-GCC-system-2.31/bin/mpiexec -np 2 ./jacobi
Jacobi 2 MPI-3.1#1 process(es) with 2 OpenMP-201511 thread(s)/process

-> matrix size: 2000x2000
-> alpha: 0.800000
-> relax: 1.000000
-> tolerance: 0.000000
-> iterations: 100

Number of iterations : 100
Residual             : 5.955111e-10
Solution Error       : 0.000266483315
Elapsed Time         : 3.3544007
MFlops               : 1547.103541
S=C=A=N: Mon May 25 16:28:59 2020: Collect done (status=0) 4s
S=C=A=N: ./scorep_jacobi_2x2_sum complete.
zam310:~/jacobi/hybrid/C [8]
```

DEMO: OPTIMIZE MEASUREMENT CONFIGURATION (SCORE)

```
zam310:~/jacobi/hybrid/C [8] square -s ./scorep_jacobi_2x2_sum/
INFO: Post-processing runtime summarization report (profile.cubex)...
/opt/local/ScoreP-6.0/bin/scorep-score -r ./scorep_jacobi_2x2_sum/profile.cubex > ./scorep_jacobi_2x2_sum/scorep.score
INFO: Score report written to ./scorep_jacobi_2x2_sum/scorep.score
zam310:~/jacobi/hybrid/C [9] head -25 ./scorep_jacobi_2x2_sum/scorep.score
```

```
Estimated aggregate size of event trace:          179kB
Estimated requirements for largest trace buffer (max_buf): 90kB
Estimated memory requirements (SCOREP_TOTAL_MEMORY): 7MB
(hint: When tracing set SCOREP_TOTAL_MEMORY=7MB to avoid intermediate flushes
or reduce requirements using USR regions filters.)
```

flt	type	max_buf[B]	visits	time[s]	time[%]	time/visit[us]	region
	ALL	91,341	3,840	13.40	100.0	3488.63	ALL
	OMP	61,078	2,812	12.76	95.2	4536.48	OMP
	MPI	27,440	812	0.55	4.1	683.27	MPI
	COM	2,756	212	0.08	0.6	400.31	COM
	SCOREP	41	2	0.00	0.0	18.35	SCOREP
	USR	26	2	0.00	0.0	17.05	USR
	OMP	17,400	400	0.00	0.0	2.09	!\$omp parallel @jacobi.c:61
	OMP	17,400	400	0.00	0.0	1.55	!\$omp parallel @jacobi.c:148
	MPI	8,900	200	0.00	0.0	7.43	MPI_Irecv
	MPI	8,900	200	0.00	0.0	4.51	MPI_Isend
	MPI	6,800	200	0.33	2.5	1663.23	MPI_Allreduce
	OMP	5,200	400	0.96	7.1	2389.23	!\$omp implicit barrier @jacobi.c:79
	OMP	5,200	400	2.09	15.6	5223.62	!\$omp for @jacobi.c:148
	OMP	5,200	400	0.22	1.6	538.58	!\$omp implicit barrier @jacobi.c:155
	OMP	5,200	400	9.31	69.5	23269.84	!\$omp for @jacobi.c:64
	OMP	5,200	400	0.00	0.0	1.73	!\$omp implicit barrier @jacobi.c:80

```
zam310:~/jacobi/hybrid/C [10]
```

Notes

- Optimize measurement config: scoring with square -s
- Also does post-processing
- Potential need for filtering
→ see user guides
- Set SCOREP_TOTAL_MEMORY

DEMO: TRACE + ANALYZE

Notes

- Measure trace:
prepend scan
 - -q:
profile off
 - -t:
trace on
- After trace measurement, Scalasca trace analyzer runs automatically

```
openSUSE-Leap-15-1
zam310:~/jacobi/hybrid/C [10] export SCOREP_TOTAL_MEMORY=10MB
zam310:~/jacobi/hybrid/C [11] scan -q -t mpiexec -np 2 ./jacobi
S=C=A=N: Scalasca 2.5 trace collection and analysis
S=C=A=N: ./scorep_jacobi_2x2_trace experiment archive
S=C=A=N: Mon May 25 16:32:12 2020: Collect start
/opt/local/easybuild-4.1.1/software/OpenMPI/3.1.4-GCC-system-2.31/bin/mpiexec -np 2 ./jacobi
Jacobi 2 MPI-3.1#1 process(es) with 2 OpenMP-201511 thread(s)/process

-> matrix size: 2000x2000
-> alpha: 0.800000
-> relax: 1.000000
-> tolerance: 0.000000
-> iterations: 100

Number of iterations : 100
Residual              : 5.955111e-10
Solution Error        : 0.000266483315
Elapsed Time          : 3.2049717
MFlops                : 1619.235889
S=C=A=N: Mon May 25 16:32:17 2020: Collect done (status=0) 5s
S=C=A=N: Mon May 25 16:32:17 2020: Analyze start
/opt/local/easybuild-4.1.1/software/OpenMPI/3.1.4-GCC-system-2.31/bin/mpiexec -np 2 /opt/local/Scalasca-2.5/bin/scout
.hyb ./scorep_jacobi_2x2_trace/traces.otf2
SCOUT (Scalasca 2.5)
Copyright (c) 1998-2019 Forschungszentrum Juelich GmbH
Copyright (c) 2009-2014 German Research School for Simulation Sciences GmbH

Analyzing experiment archive ./scorep_jacobi_2x2_trace/traces.otf2

Opening experiment archive ... done (0.000s).
Reading definition data    ... done (0.001s).
Reading event trace data  ... done (0.015s).
Preprocessing             ... done (0.001s).
Analyzing trace data      ... done (0.026s).
```

FURTHER USEFUL INFORMATION

Extended and more detailed example based on NAS Parallel Benchmark (NPB) BT

- Scalasca documentation
 - A full workflow example
- Score-P documentation
 - Performance Analysis Workflow Using Score-P
- Slides from 33rd VI-HPS Tuning Workshop
 - Score-P instrumentation & measurement toolset
 - Score-P analysis scoring & measurement filtering
 - Score-P specialized instrumentation and measurement (Advanced)
 - Scalasca automated trace analysis



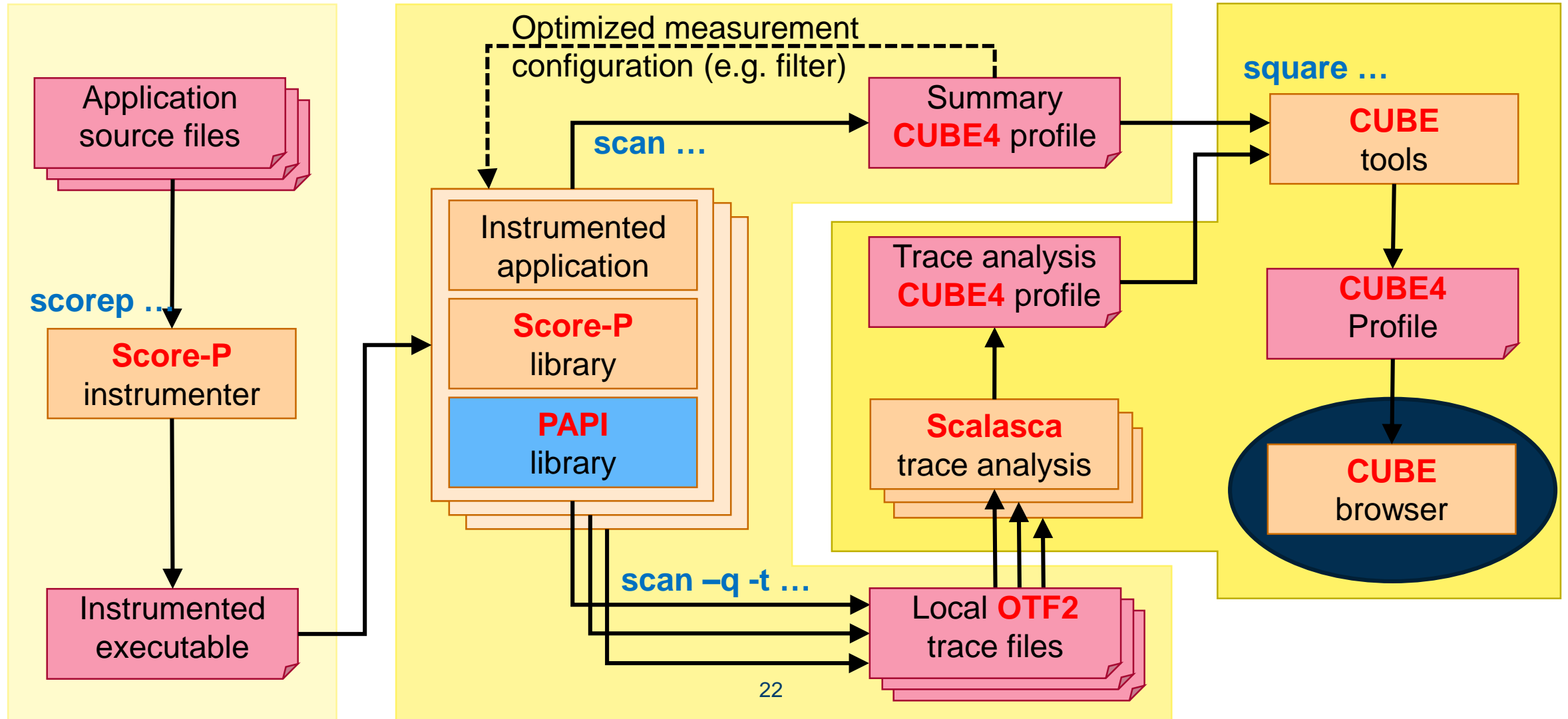
CUBE

PERFORMANCE ANALYSIS WORKFLOW

1. Instrumentation

2. Measurement

3. Analysis

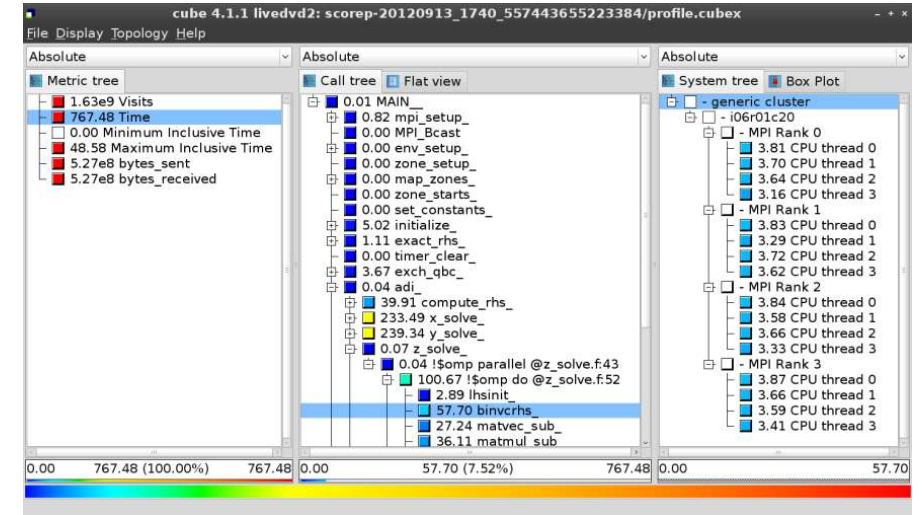


SCORE-P / SCALASCA MEASUREMENT DATA

- All measured data and meta information stored in [experiment directory](#)
 - `scorep_<executable>_<size>_[sum|trace]`
 - Example: `scorep_tea_leaf_2p8x12_trace`
- Contents
 - `profile.cubex` Score-P profile measurement
 - [summary.cubex](#) Post-processed profile measurement
 - `scout.cubex` Scalasca trace analyzer result
 - [trace.cubex](#) Post-processed trace analyzer result
- Post-processed Cube files include
 - Additional derived metrics
 - Enhanced metrics hierarchy

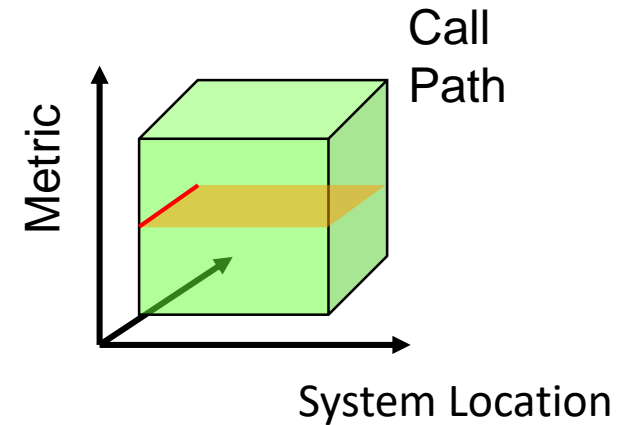
CUBE

- Parallel program analysis report exploration tools
 - Libraries for XML+binary report reading & writing
 - Algebra utilities for report processing
 - GUI for interactive analysis exploration
 - Requires Qt4 ≥ 4.6 or Qt 5
- Originally developed as part of the Scalasca toolset
- Now available as a separate component
 - Can be installed independently of Score-P, e.g., on laptop or desktop
 - Latest release: Cube v4.7



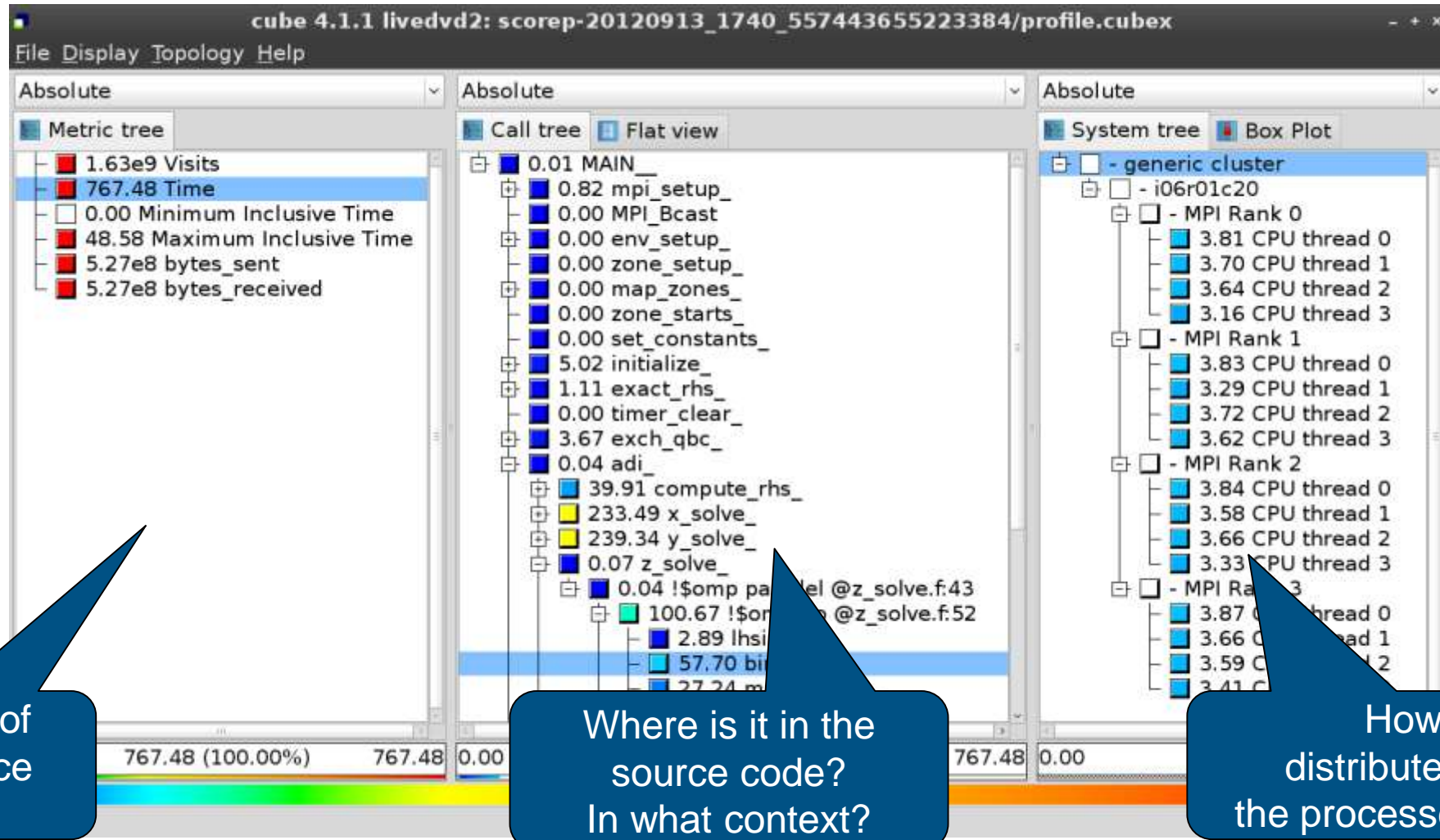
CUBE DATA

- Measured values organized in 3D block (“Cube”) along **three hierarchical axes**
 - Metrics (general → specific)
 - Call path (program location)
 - System location (machine → node → process → thread)
- Displayed as **three coupled tree browsers**
 - Each node displays metric value
 - As color: for easy identification of bottlenecks
 - As number: for precise comparison
 - Displayed metric value depends on state
 - Collapsed (inclusive value)
 - Expanded (exclusive value)



v	■	10 main
>	■	30 foo
>	■	60 bar

ANALYSIS PRESENTATION



CUBE BASIC COMMANDS

- **Expand / Collapse tree nodes**
 - Chooses level of granularity
 - Use context menu to expand / collapse whole (sub)trees
- **Select tree nodes**
 - Shows distribution of metric value in tree to the right
 - Use Ctrl+Click to select multiple nodes

STARTING CUBE GUI

Local scenario:

```
% square <experiment_directory>
```

- Performs post-processing if necessary
- Executes Cube GUI

Remote scenario:

```
% square -s <experiment_directory>
```

- Performs post-processing if necessary
- Copy desired Cube file to local system
- (or remotely mount file system)

```
% cube <cube_file>
```



SCALASCA CASE STUDY – TEA LEAF

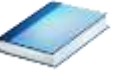
CASE STUDY: TEALEAF

- 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

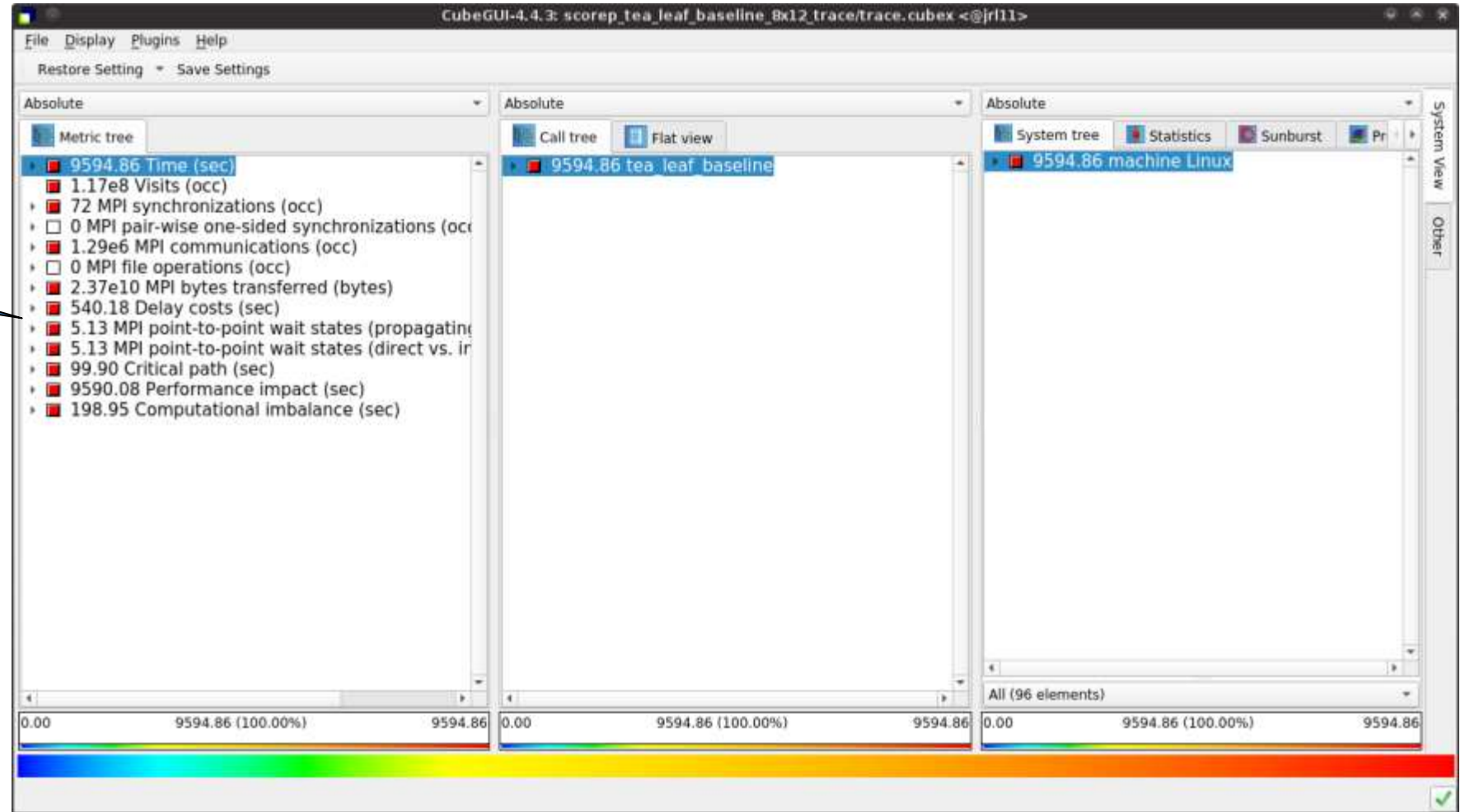


SCALASCA ANALYSIS REPORT

EXPLORATION (OPENING VIEW)



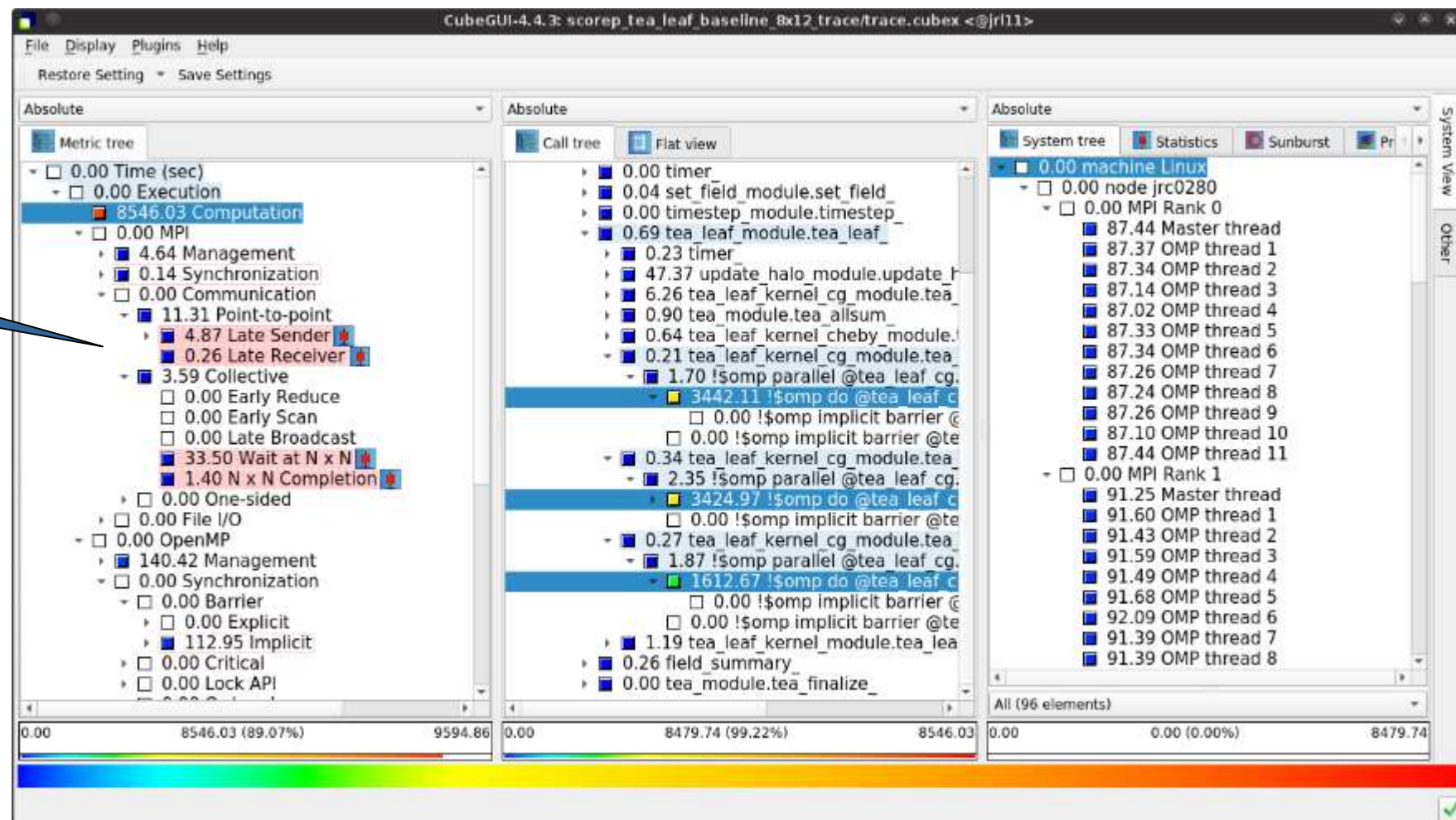
Additional top-level metrics produced by the trace analysis...



SCALASCA WAIT-STATE METRICS



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

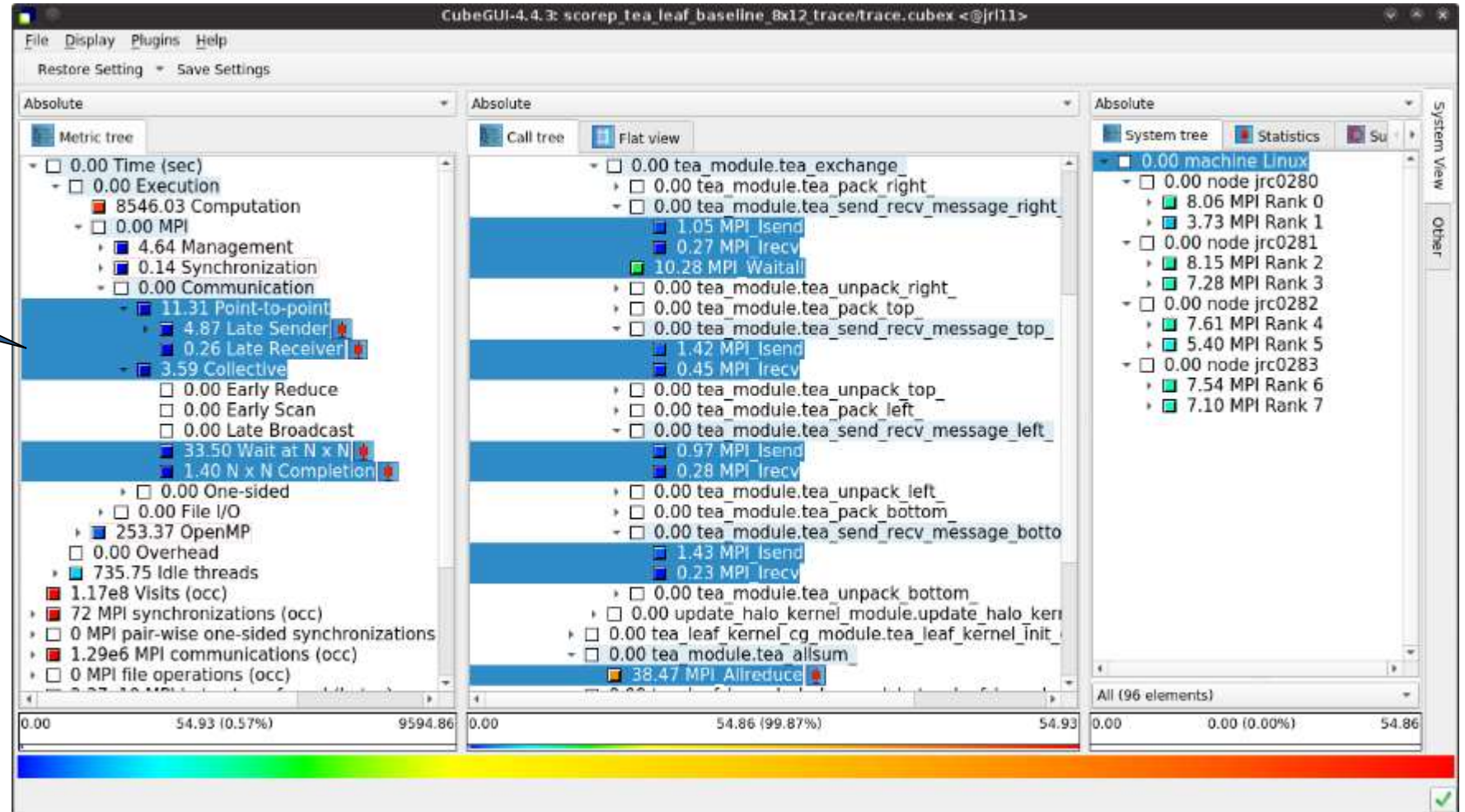


TEALEAF SCALASCA REPORT ANALYSIS



(I)

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

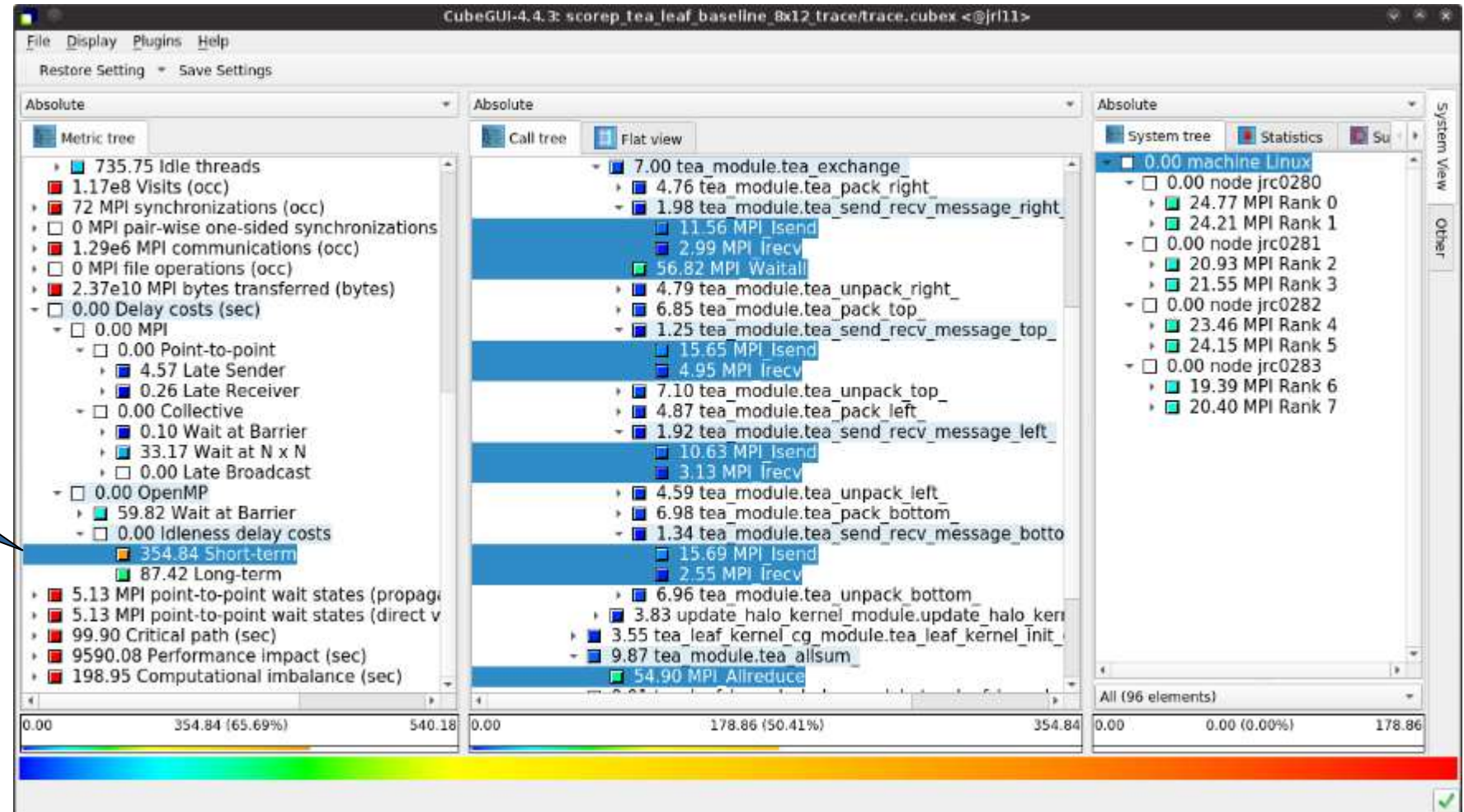


TEALEAF SCALASCA REPORT ANALYSIS

(II)



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

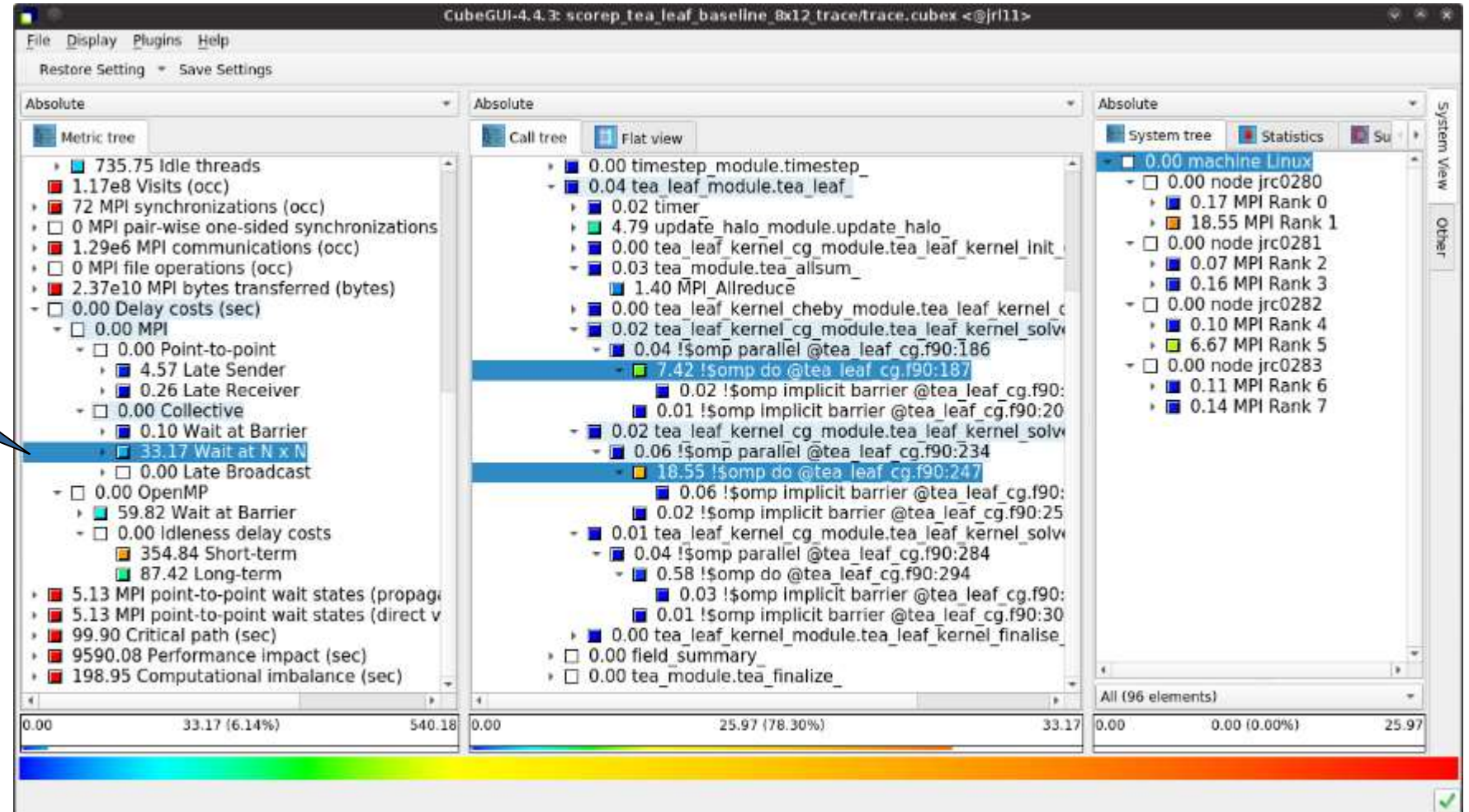


TEALEAF SCALASCA REPORT ANALYSIS



(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.)...



TEALEAF SCALASCA REPORT ANALYSIS



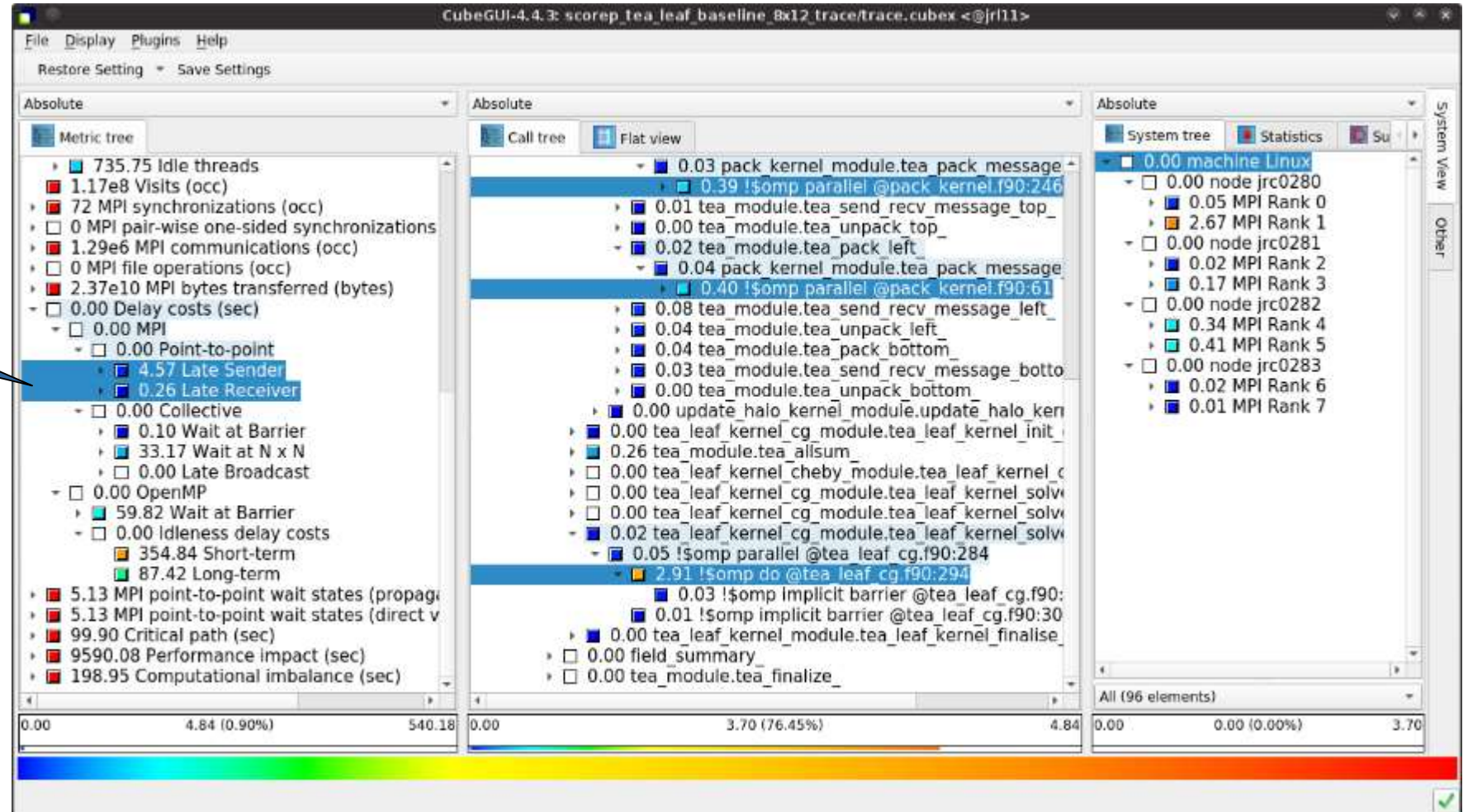
JÜLICH
Forschungszentrum

JÜLICH
SUPERCOMPUTING
CENTRE



(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



TEALEAF SCALASCA REPORT ANALYSIS



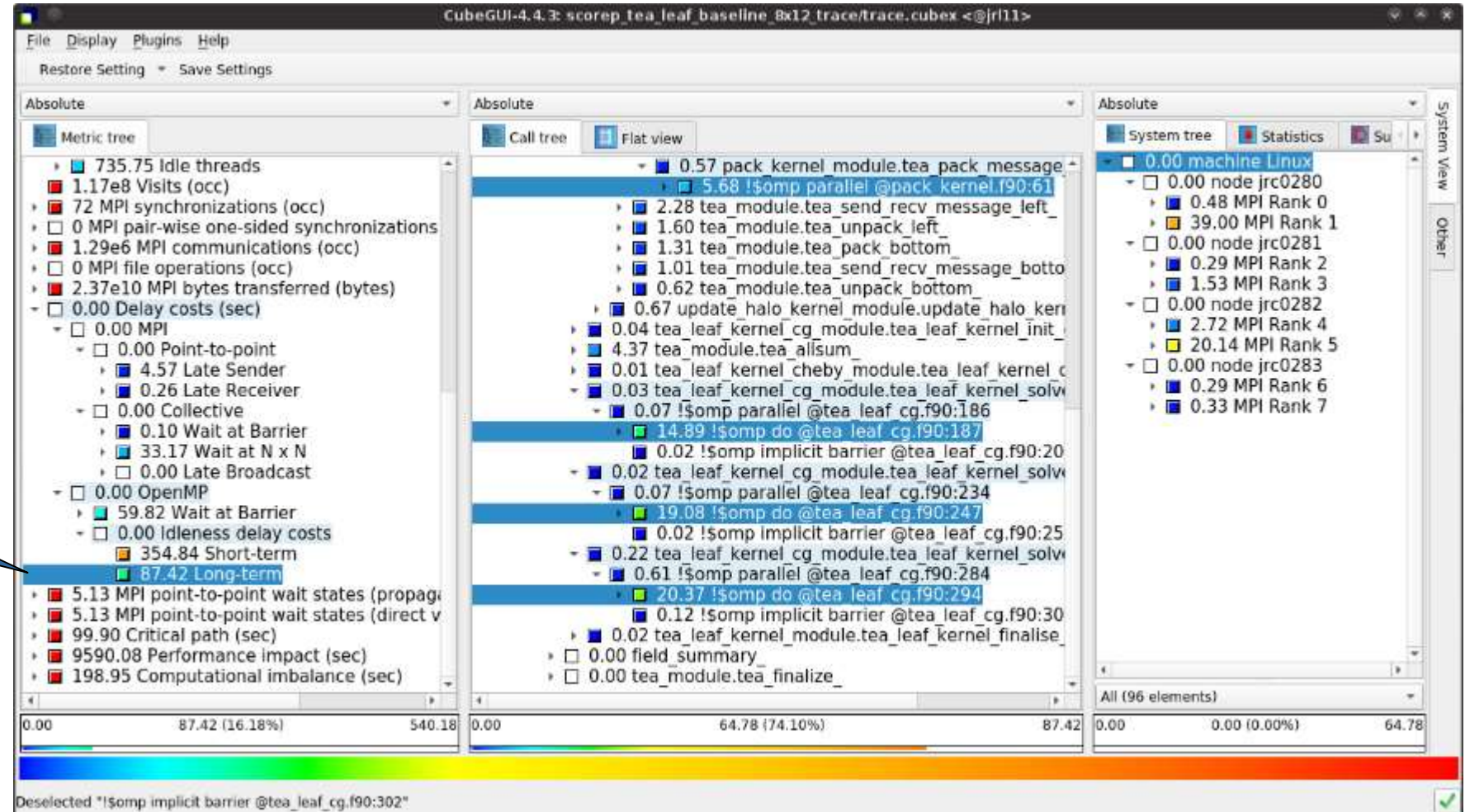
JÜLICH
Forschungszentrum

JÜLICH
SUPERCOMPUTING
CENTRE



(V)

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

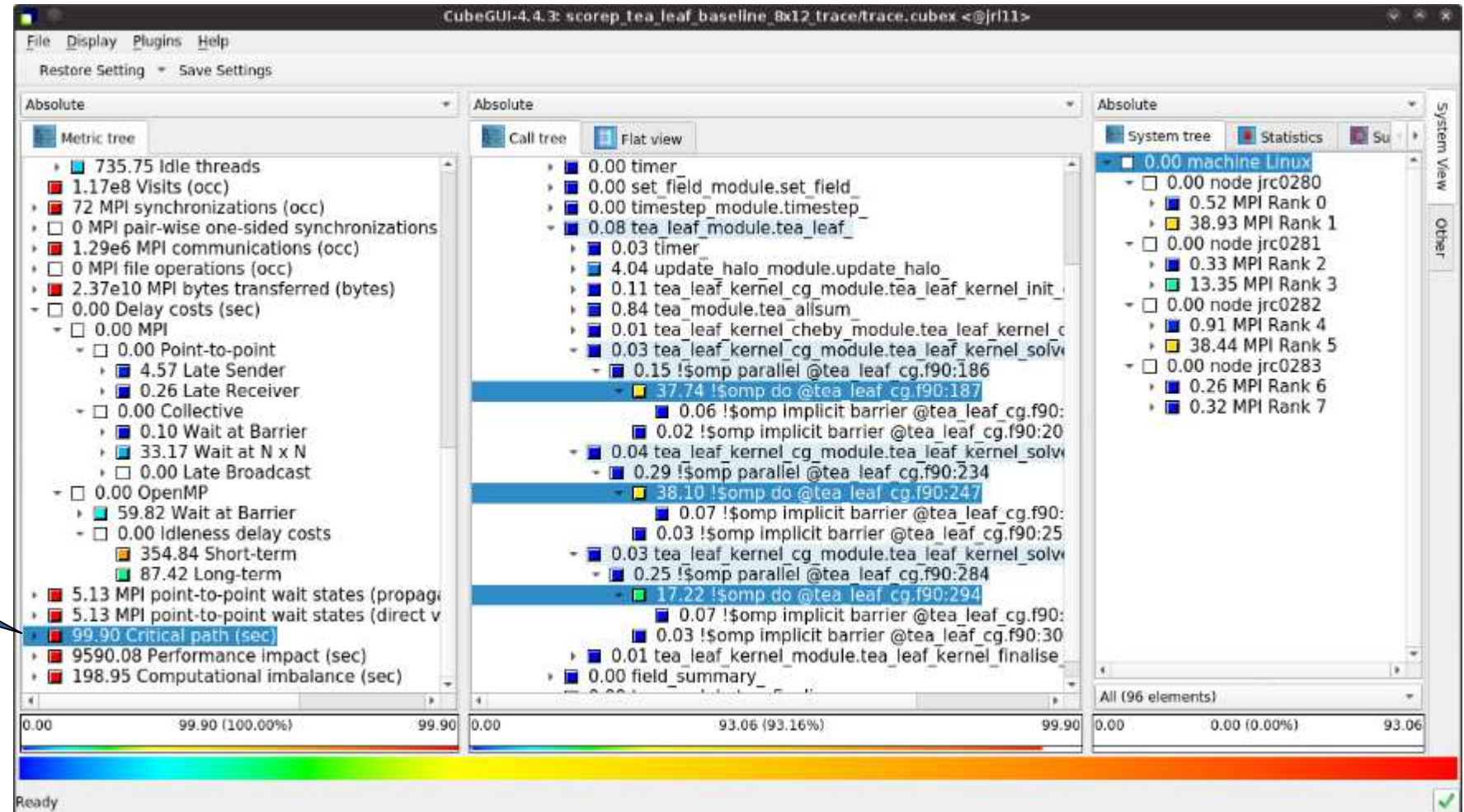


TEALEAF SCALASCA REPORT ANALYSIS

(VI)



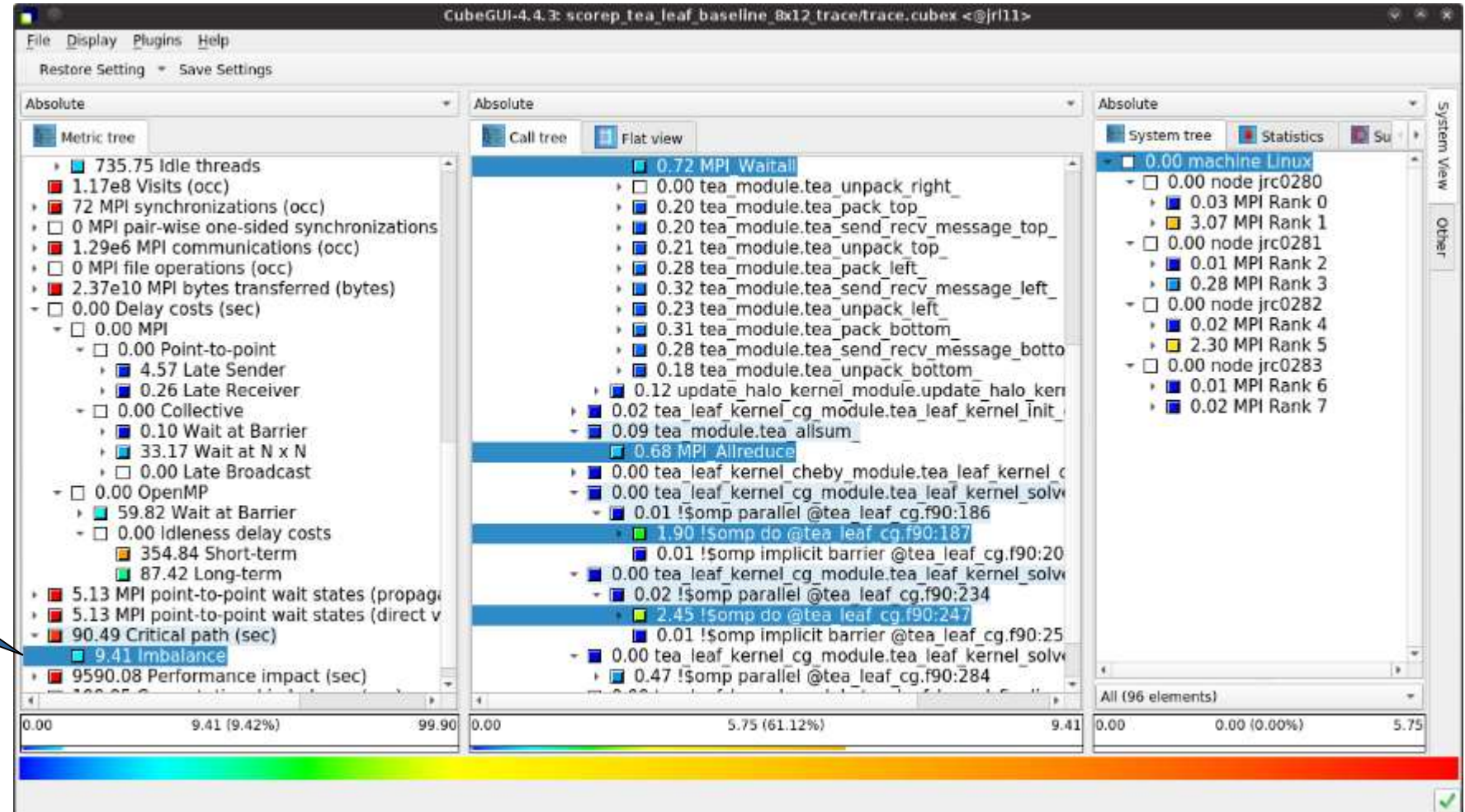
The Critical Path also
highlights the three
solver loops...



TEALEAF SCALASCA REPORT ANALYSIS (VII)



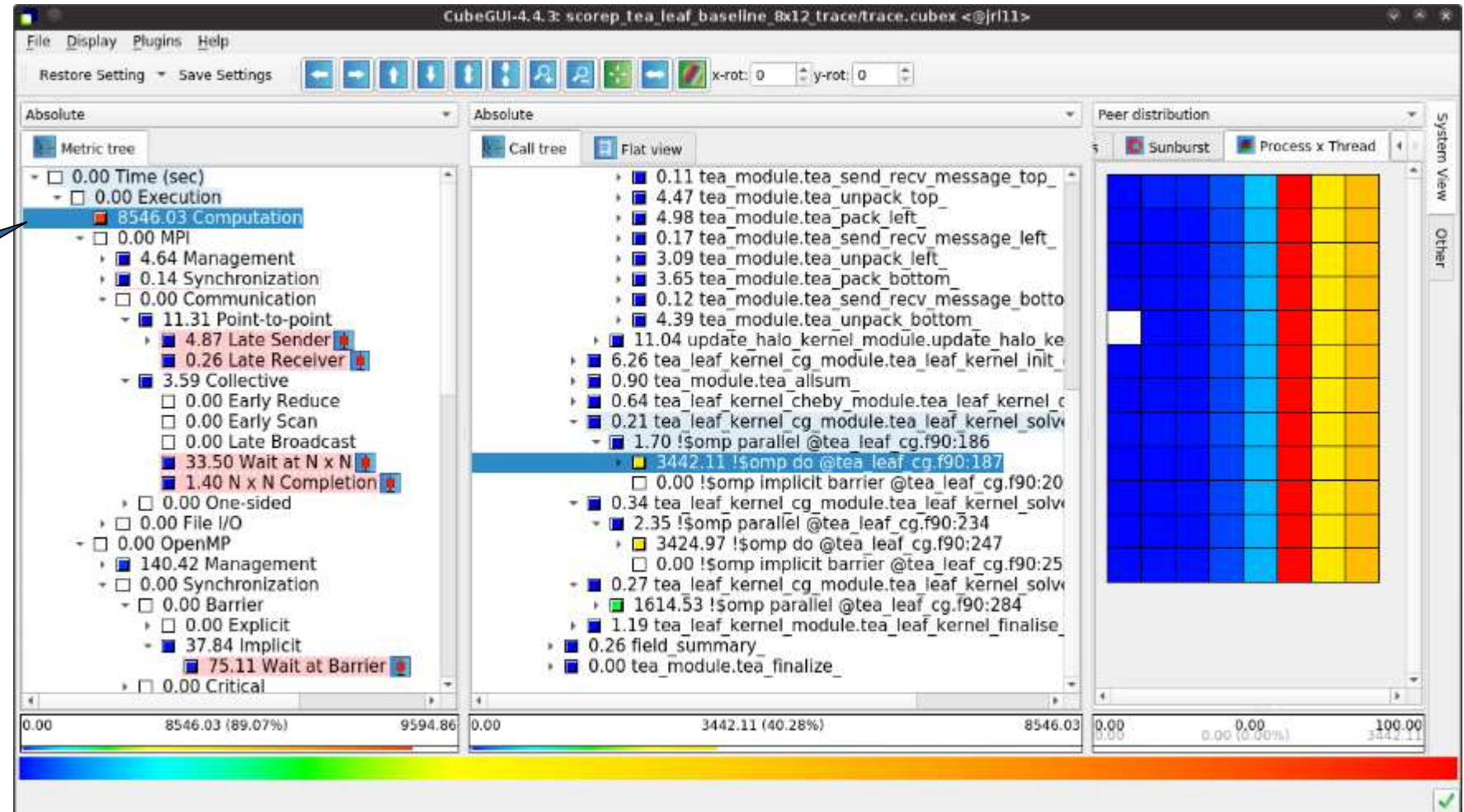
...with imbalance (time on critical path above average) mostly in the first two loops and MPI communication



TEALEAF SCALASCA REPORT ANALYSIS (VIII)



Computation time of
1st...

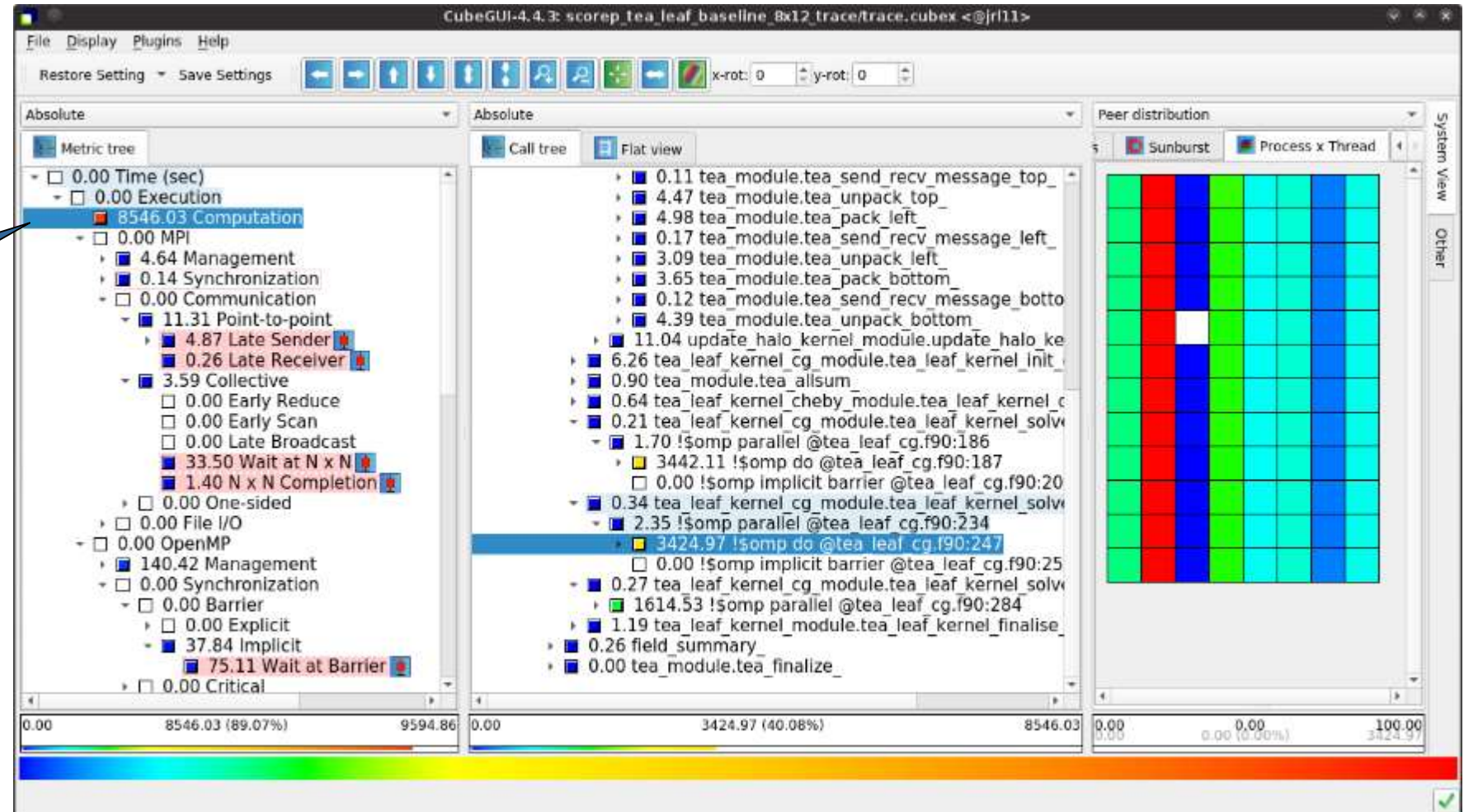


TEALEAF SCALASCA REPORT ANALYSIS

(IX)



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

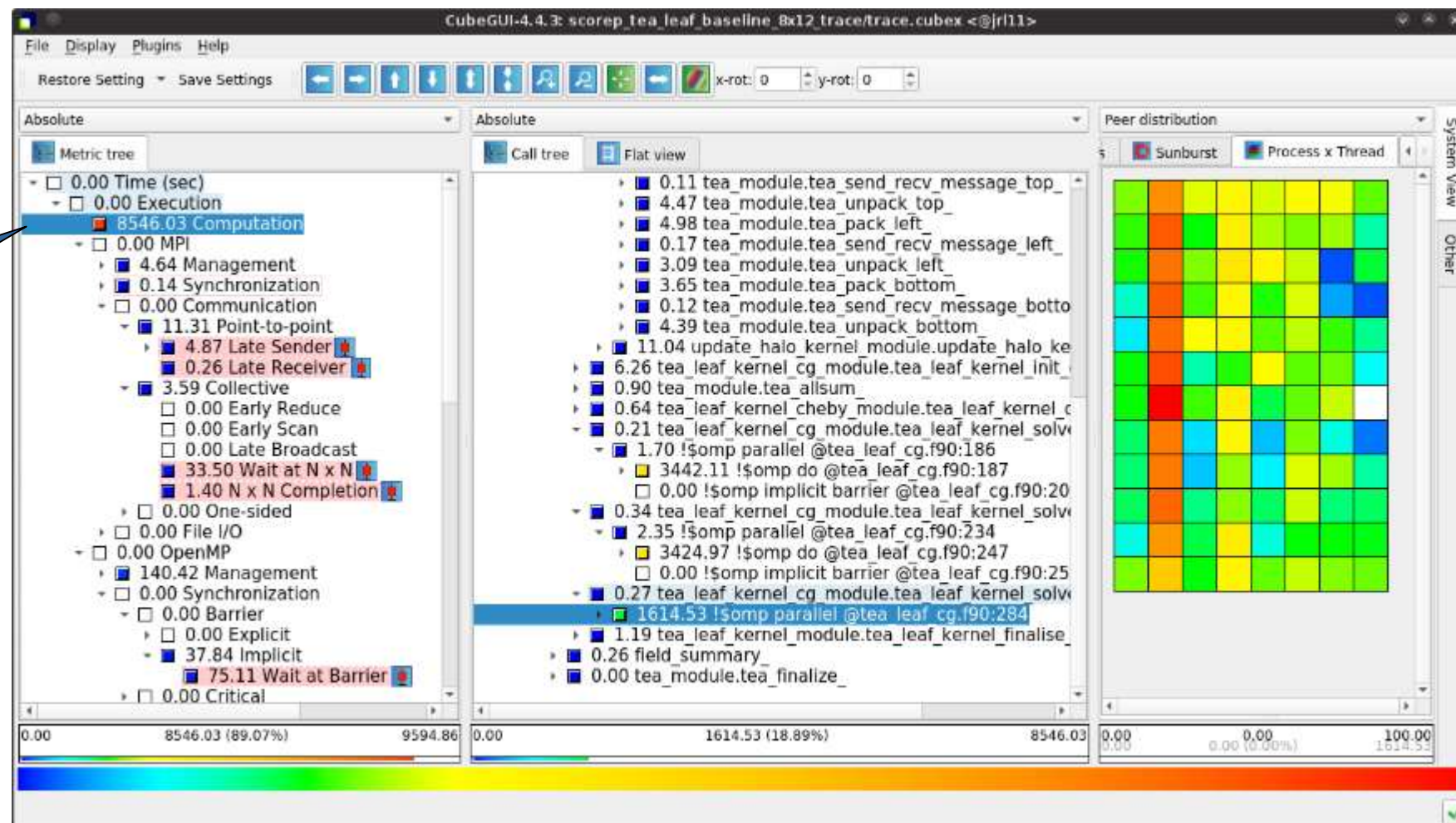


TEALEAF SCALASCA REPORT ANALYSIS

(X)



...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%**



HOW-TO GET POP METRICS

RECAP: POP METRICS

<https://pop-coe.eu/further-information/learning-material>

- **Original (POP1) Metrics**

- Article explaining the POP Standard Metrics for Parallel Performance Analysis
- Presentation summarizing the POP Standard Metrics for Parallel Performance Analysis

- **New (POP2) Hybrid Metrics**

- ➡ • Introduction explaining the POP2 Standard Metrics for Performance Analysis of Hybrid Parallel Applications
- Cheat sheet for Additive Hybrid Metrics
- Cheat sheet for Multiplicative Hybrid Metrics
- In-depth explanation of the Additive Hybrid Metrics
- ➡ • Webinar Identifying Performance Bottlenecks in Hybrid MPI + OpenMP Software

POP METRICS + SCALASCA

1. Instrument application and setup measurement parameters (e.g. filtering)
 - `scorep <comp+link+cmds>`
 - `scan <exec+cmd> ...`
2. For parallel efficiency: perform trace measurement and analysis
3. For computational scaling: perform profile measurement with suitable HW counters
 - `scan -P pop <exec+cmd>`
4. Merge profile and trace measurement
5. Post-process measurement
6. Analyze POP metrics with [Cube Advisor](#)
 - `square <measurement+archive>`

Requires

- Scalasca >= V2.6
- Cube >= V4.6



MEASUREMENT DEMO

- Measurement of simple Jacobi solver
 - Solves Poisson equation on rectangular grid assuming
 - Uniform discretization in each direction
 - Dirichlet boundary conditions
- Available in multiple variants (Shipped with Score-P)
 - C, C++ or Fortran source code
 - MPI, OpenMP, or hybrid (MPI+OpenMP)

DEMO: POP PRESET MEASUREMENT

Notes

- -P pop selects POP metrics measurement
- Automatically executes necessary trace and profile measurements

```
zam310:~/jacobi/hybrid/C [1] scan -P pop mpiexec -np 2 ./jacobi
path:>/opt/local/Scalasca-2.6/share/scalasca/presets/pop.preset<
S=C=A=N: RUN: 0 REPETITION: 0 Archive=./scorep_jacobi_2x2_preset_pop_c2/scorep_jacobi_2x2_c1
S=C=A=N: RUN: Scalasca tracing run
S=C=A=N: Scalasca 2.6 trace collection and analysis
S=C=A=N: ./scorep_jacobi_2x2_preset_pop_c2/scorep_jacobi_2x2_c1 experiment archive
S=C=A=N: Tue May 25 16:35:02 2021: Collect start
/opt/local/easybuild-4.1.1/software/OpenMPI/3.1.4-GCC-system-2.31/bin/mpiexec -np 2 ./jacobi
Jacobi 2 MPI-3.1#1 process(es) with 2 OpenMP-201511 thread(s)/process

-> matrix size: 2000x2000
-> alpha: 0.800000
-> relax: 1.000000
-> tolerance: 0.000000
-> iterations: 100

Number of iterations : 100
Residual              : 5.955111e-10
Solution Error        : 0.000266483315
Elapsed Time          : 3.1198404
MFlops                : 1663.420090
S=C=A=N: Tue May 25 16:35:06 2021: Collect done (status=0) 4s
S=C=A=N: Tue May 25 16:35:06 2021: Analyze start
/opt/local/easybuild-4.1.1/software/OpenMPI/3.1.4-GCC-system-2.31/bin/mpiexec -np 2 /opt/local/Scalasca-2.6/bin/scout.hy
b ./scorep_jacobi_2x2_preset_pop_c2/scorep_jacobi_2x2_c1/traces.otf2
SCOUT (Scalasca 2.6)
Copyright (c) 1998-2021 Forschungszentrum Juelich GmbH
Copyright (c) 2014-2021 RWTH Aachen University
Copyright (c) 2009-2014 German Research School for Simulation Sciences GmbH

Analyzing experiment archive ./scorep_jacobi_2x2_preset_pop_c2/scorep_jacobi_2x2_c1/traces.otf2

Opening experiment archive ... done (0.000s).
Reading definition data    ... done (0.001s).
Reading event trace data   ... done (0.004s).
```

DEMO: POP PRESET MEASUREMENT

```
openSUSE-Leap-15-1
Max. memory usage      : 0.000MB

Total processing time   : 0.119s
S=C=A=N: Tue May 25 16:35:07 2021: Analyze done (status=0) 1s
Warning: 0.172MB of analyzed trace data retained in ./scorep_jacobi_2x2_preset_pop_c2/scorep_jacobi_2x2_c1/traces!
S=C=A=N: ./scorep_jacobi_2x2_preset_pop_c2/scorep_jacobi_2x2_c1 complete.

S=C=A=N: RUN: 1 REPETITION: 0 Archive=./scorep_jacobi_2x2_preset_pop_c2/scorep_jacobi_2x2_c2
S=C=A=N: RUN: Profiling run with PAPI counters
S=C=A=N: Scalasca 2.6 runtime summarization
S=C=A=N: ./scorep_jacobi_2x2_preset_pop_c2/scorep_jacobi_2x2_c2 experiment archive
S=C=A=N: Tue May 25 16:35:07 2021: Collect start
/opt/local/easybuild-4.1.1/software/OpenMPI/3.1.4-GCC-system-2.31/bin/mpixexec -np 2 ./jacobi
Jacobi 2 MPI-3.1#1 process(es) with 2 OpenMP-201511 thread(s)/process

-> matrix size: 2000x2000
-> alpha: 0.800000
-> relax: 1.000000
-> tolerance: 0.000000
-> iterations: 100

Number of iterations : 100
Residual              : 5.955111e-10
Solution Error        : 0.000266483315
Elapsed Time          : 3.0536247
MFlops                : 1699.490183
S=C=A=N: Tue May 25 16:35:10 2021: Collect done (status=0) 3s
S=C=A=N: ./scorep_jacobi_2x2_preset_pop_c2/scorep_jacobi_2x2_c2 complete.
zam310:~/jacobi/hybrid/C [2] square -s ./scorep_jacobi_2x2_preset_pop_c2
INFO: Merging aggregated runtime summary and trace analysis reports...
INFO: Post-processing combined summary and trace analysis report (scout+profile.cubex)...
/opt/local/ScoreP-7.0/bin/scorep-score -r ./scorep_jacobi_2x2_preset_pop_c2/scout+profile.cubex > ./scorep_jacobi_2x2_preset_pop_c2/scorep.score
INFO: Score report written to ./scorep_jacobi_2x2_preset_pop_c2/scorep.score
zam310:~/jacobi/hybrid/C [3]
```

Notes

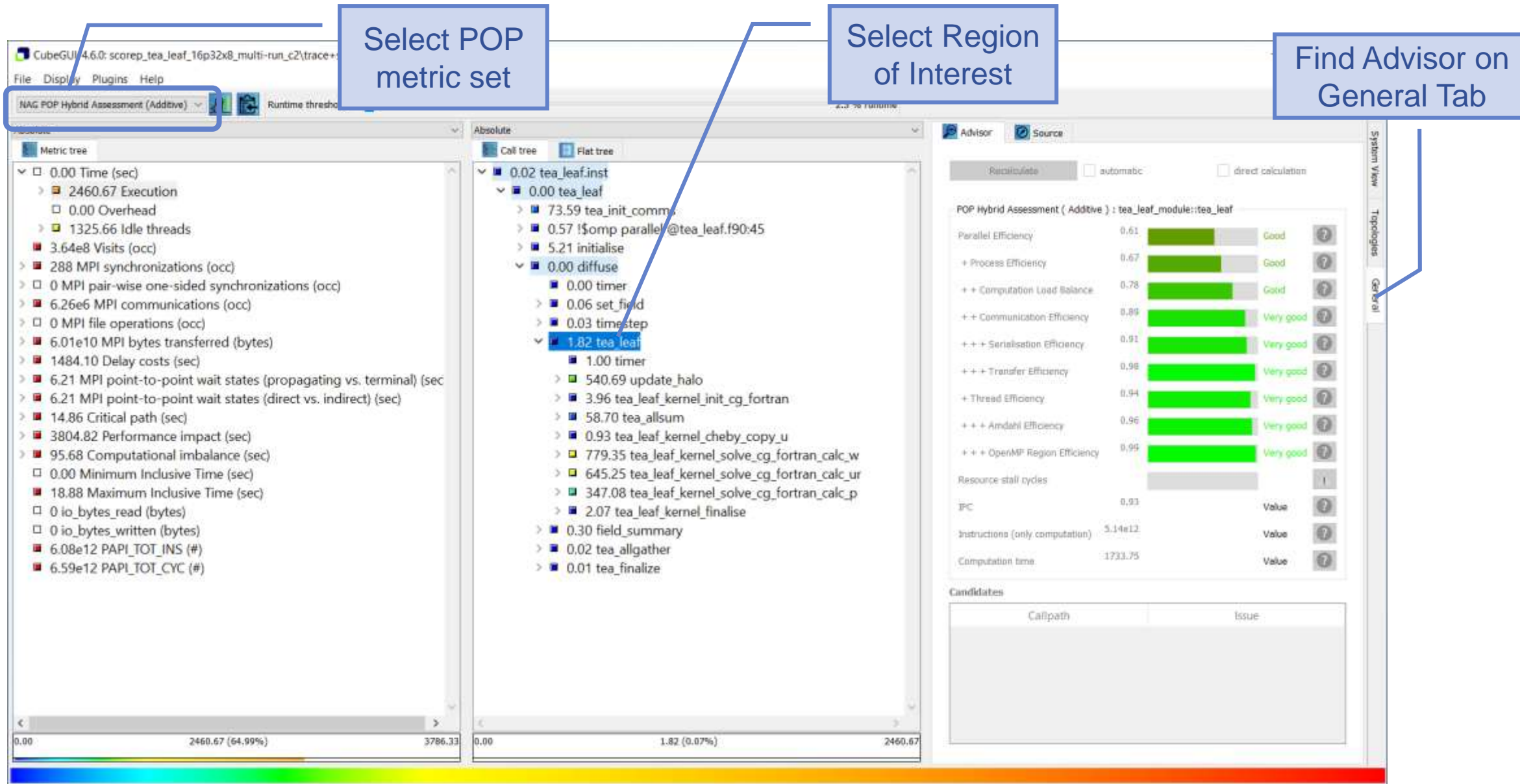
- Square recognizes POP metrics measurement
- Merges the two measurements before post-processing

ANALYSIS DEMO

- TeaLeaf Reference V1.0
- HPC mini-app developed by the UK Mini-App Consortium
 - Solves the linear 2D heat conduction equation on a spatially decomposed regular ζ using a 5 point stencil with implicit solvers
 - https://github.com/UK-MAC/TeaLeaf_ref/archive/v1.0.tar.gz
- Measurements performed on Jusuf cluster @ JSC
 - Run configuration
 - 32 MPI ranks with 8 OpenMP threads each
 - Distributed across 2 compute nodes (16 ranks per node)
 - Test problem “5”: 4000 × 4000 cells, CG solver



POP METRICS + CUBE ADVISOR



The screenshot displays the CubeAdvisor 4.6.0 interface. The left pane shows a 'Metric tree' with various performance metrics. The middle pane shows a 'Call tree' with a hierarchical view of the execution. The right pane shows the 'Advisor' tab with a table of performance metrics and their status.

Select POP metric set

Select Region of Interest

Find Advisor on General Tab

Advisor Tab Data:

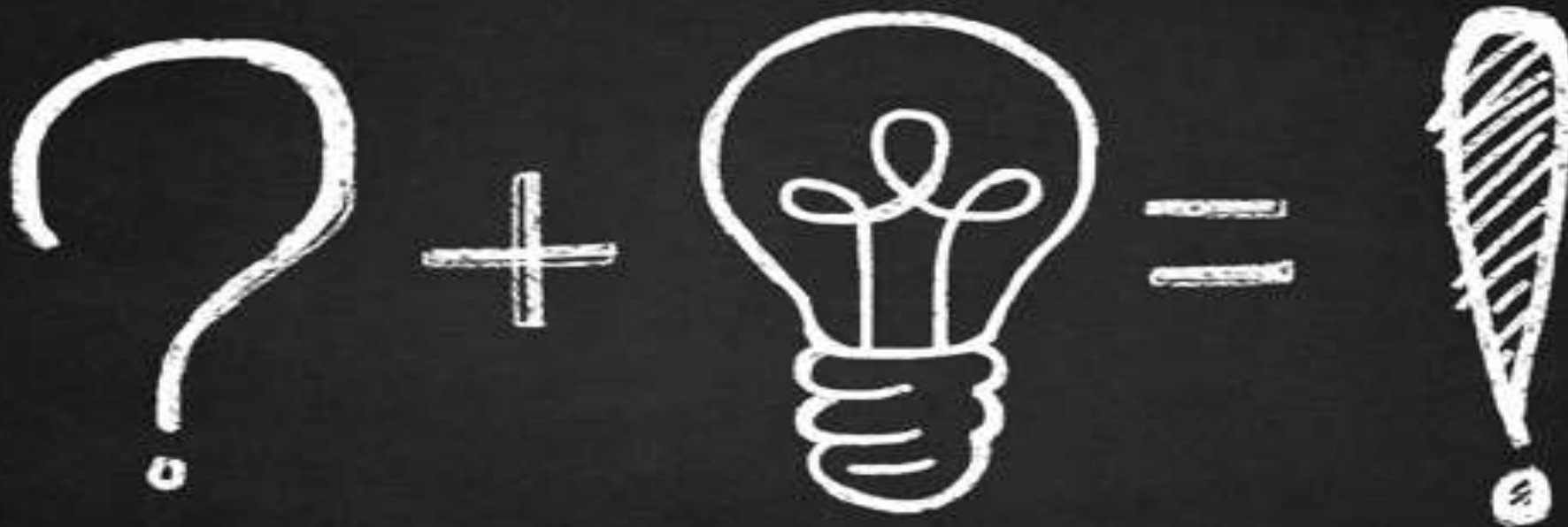
Metric	Value	Status
Parallel Efficiency	0.61	Good
+ Process Efficiency	0.67	Good
++ Computation Load Balance	0.78	Good
+++ Communication Efficiency	0.89	Very good
+++ Serialisation Efficiency	0.91	Very good
+++ Transfer Efficiency	0.98	Very good
+ Thread Efficiency	0.94	Very good
++ Amdahl Efficiency	0.96	Very good
+++ OpenMP Region Efficiency	0.99	Very good
Resource stall cycles		
IPC	0.93	Value
Instructions (only computation)	5.14e12	Value
Computation time	1733.75	Value

Candidates Table:

Callpath	Issue

WHERE TO GET HELP

Tool	Support Email
Score-P	support@score-p.org
Scalasca, Cube	scalasca@fz-juelich.de



QUESTIONS