

International HPC Summer School 2021: Performance analysis and optimization

Analysis report examination with Cube

Ilya Zhukov
Jülich Supercomputing Centre

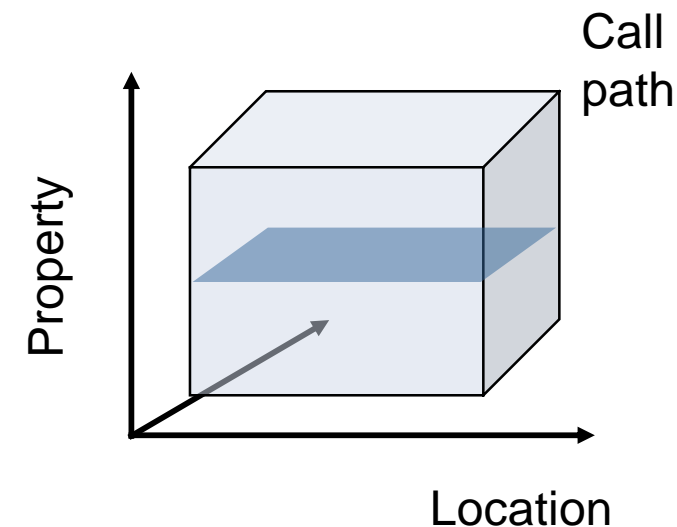


Cube

- 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 \geq 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 4.6 (April 2021)

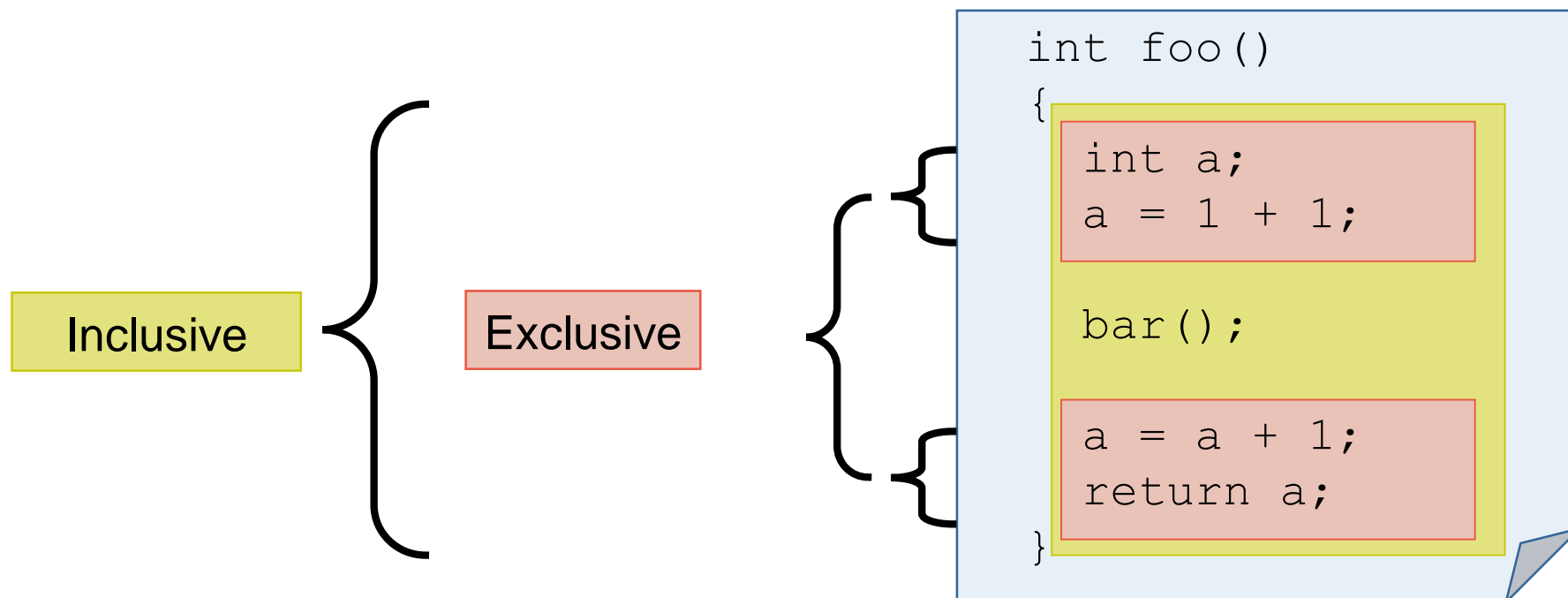
Analysis presentation and exploration

- Representation of values (severity matrix) on three hierarchical axes
 - Performance property (metric)
 - Call path (program location)
 - System location (process/thread)
- Three coupled tree browsers
- Cube displays severities
 - As value: for precise comparison
 - As color: for easy identification of hotspots
 - Inclusive value when closed & exclusive value when expanded
 - Customizable via display modes

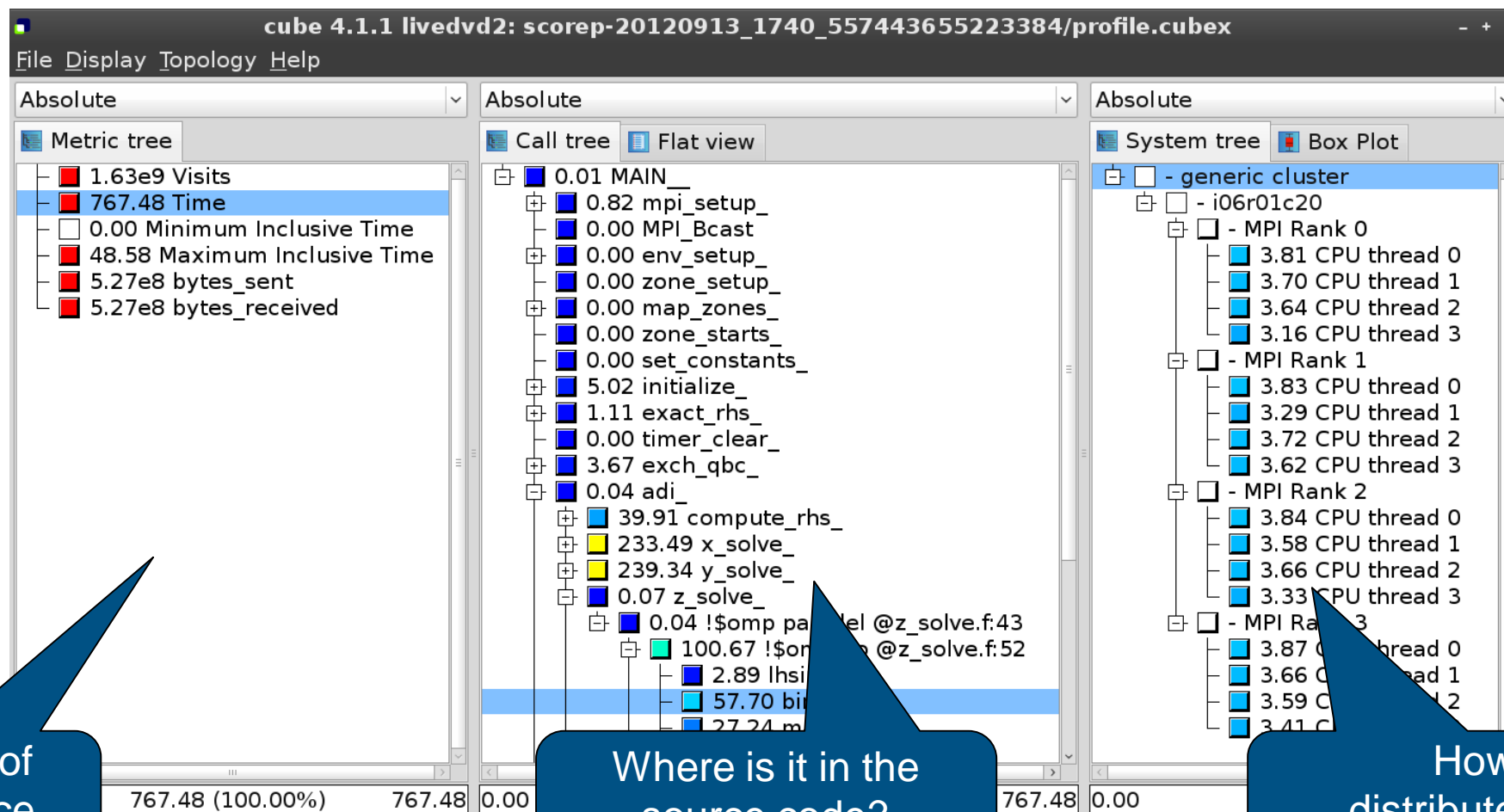


Inclusive vs. exclusive values

- Inclusive
 - Information of all sub-elements aggregated into single value
- Exclusive
 - Information cannot be subdivided further



Analysis presentation

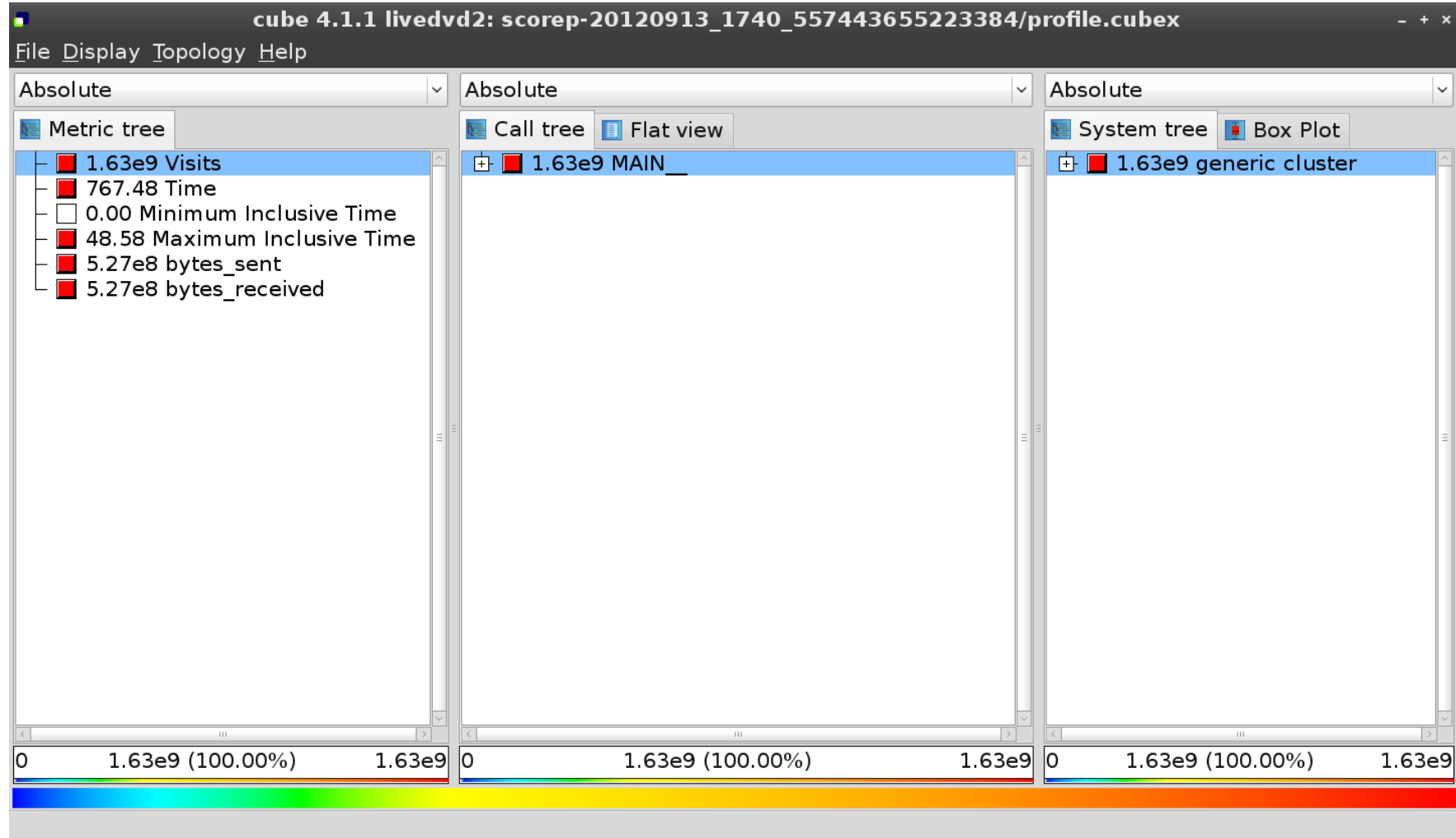


What kind of performance metric?

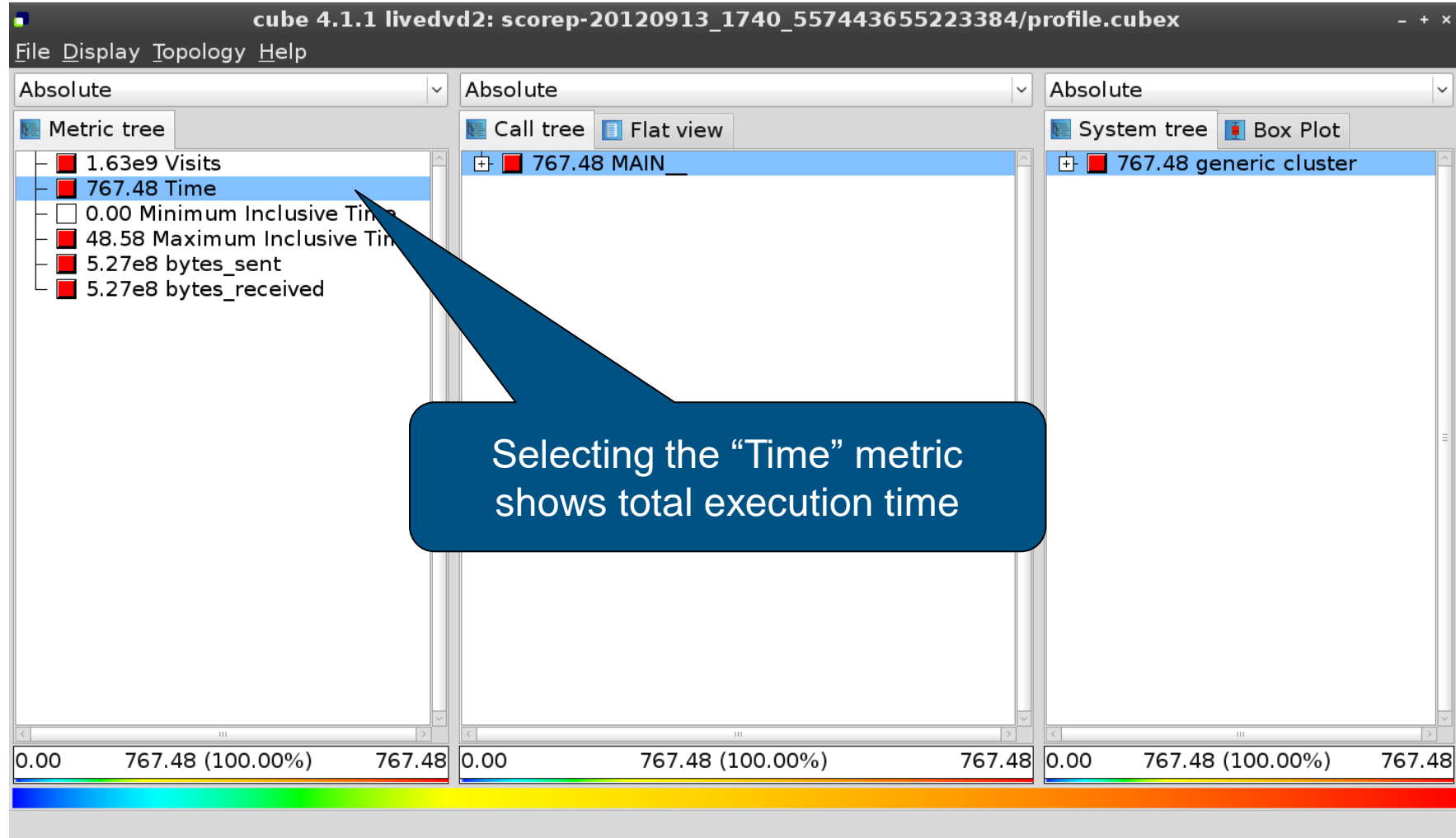
Where is it in the source code?
In what context?

How is it distributed across the processes/threads?

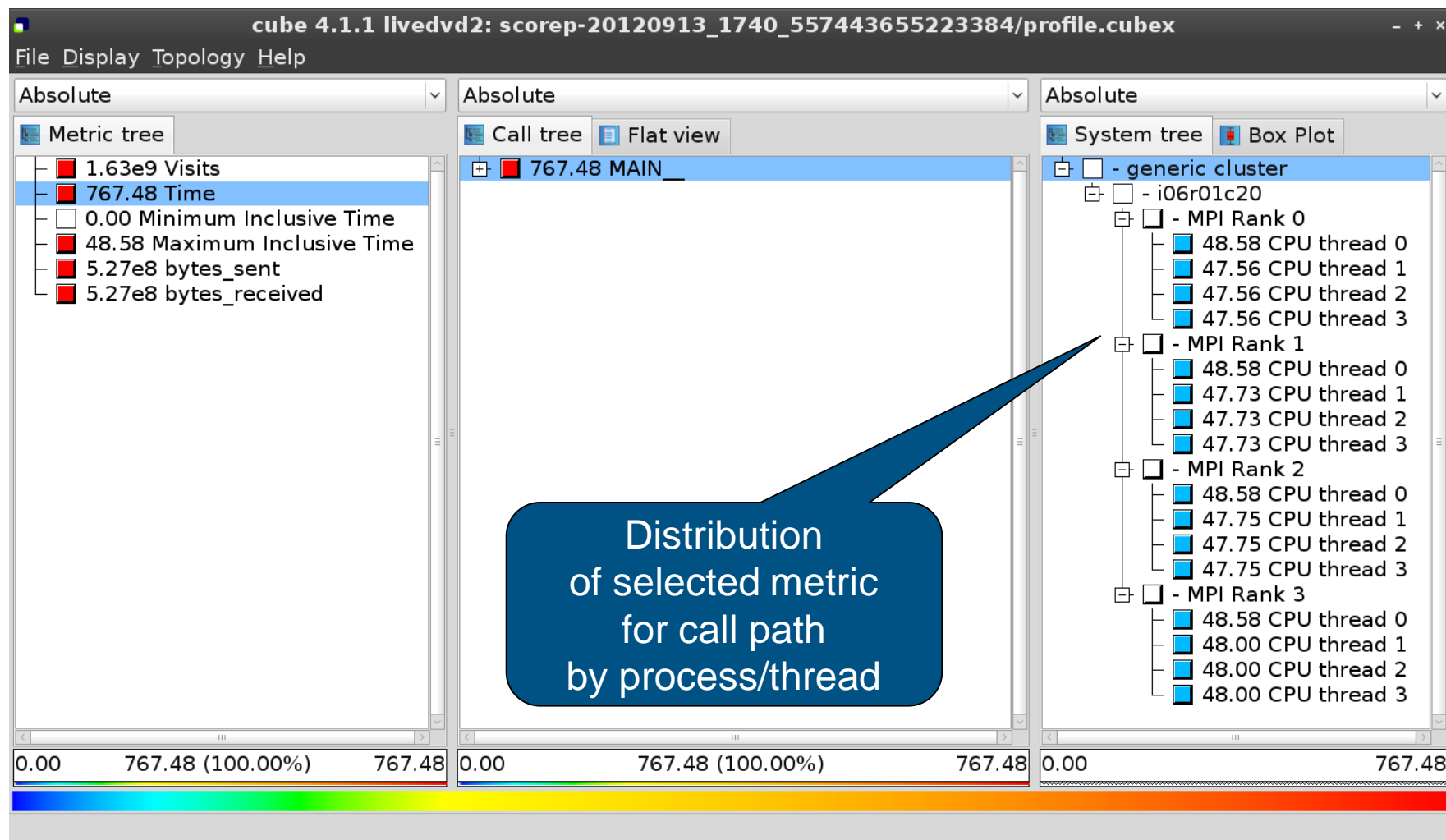
Score-P analysis report exploration (opening view)



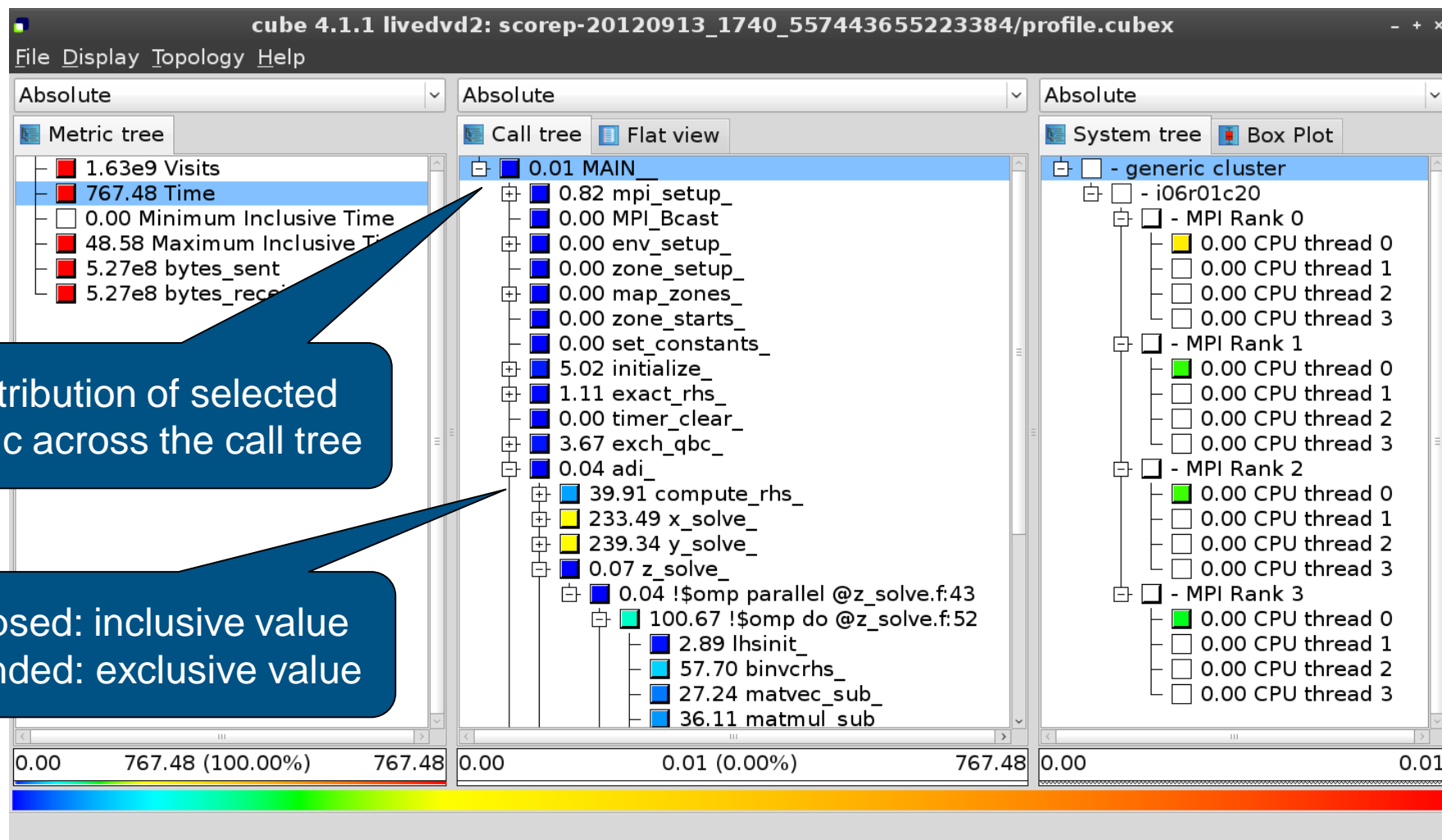
Metric selection



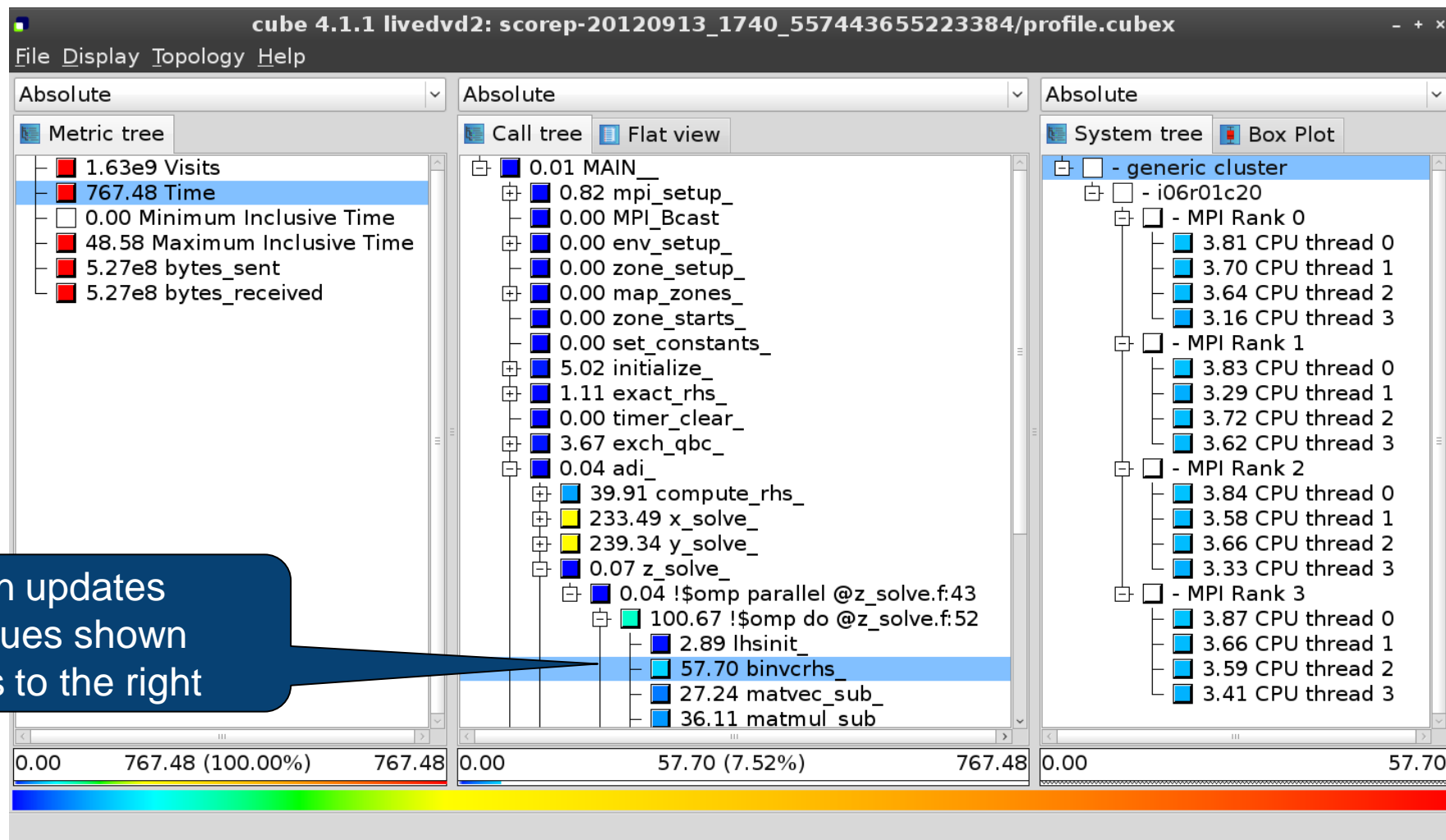
Expanding the system tree



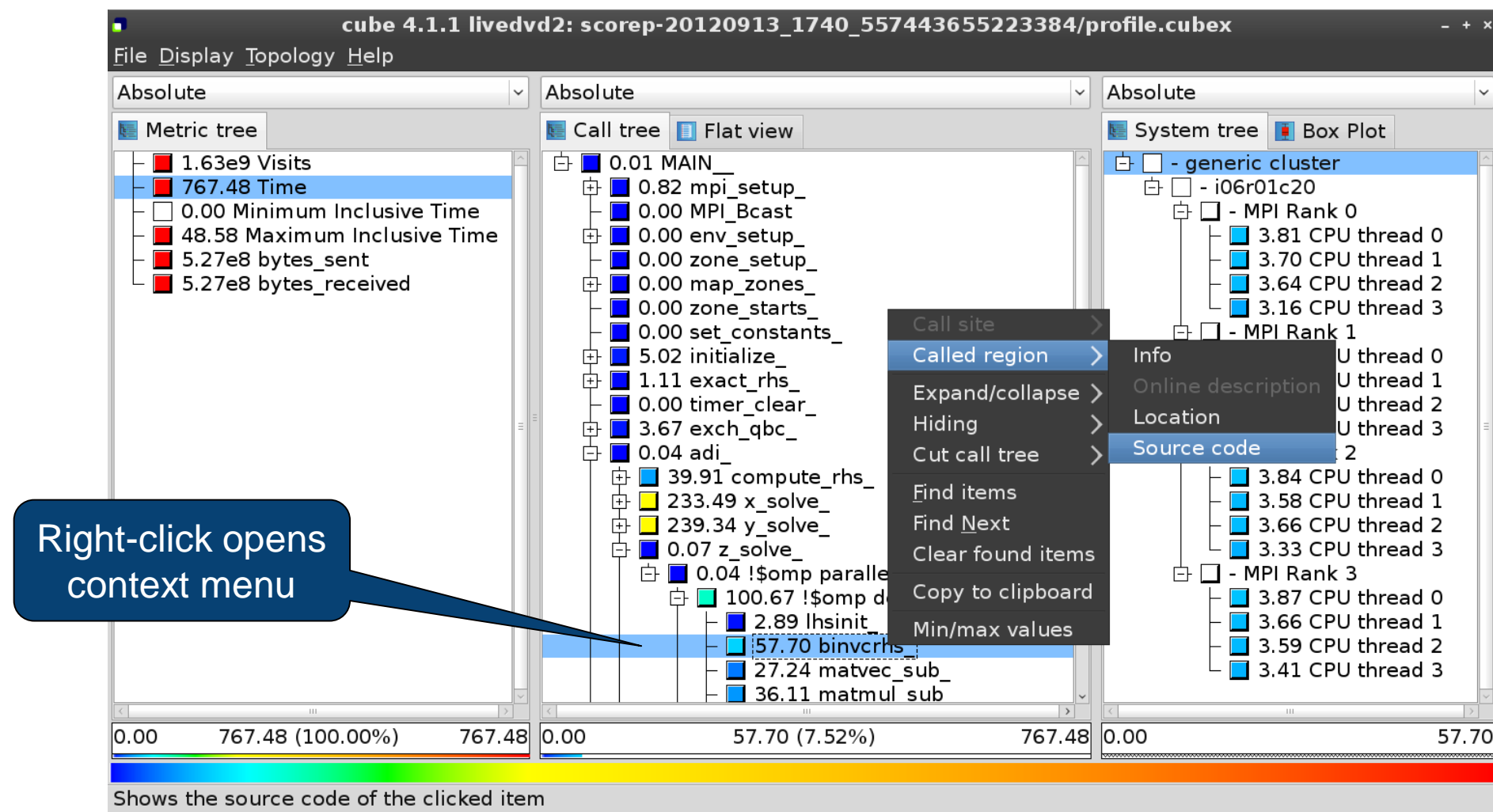
Expanding the call tree



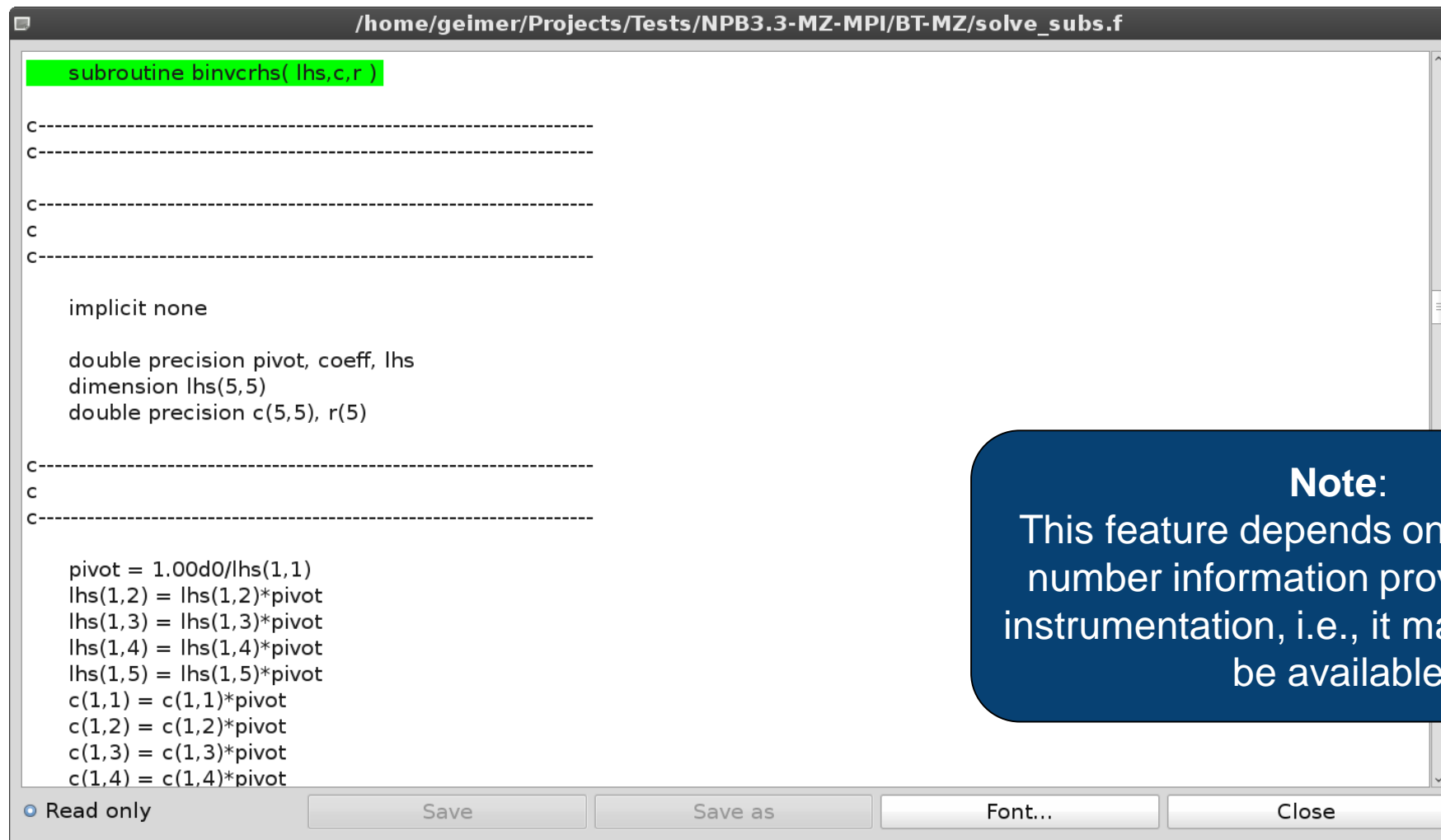
Selecting a call path



Source-code view via context menu



Source-code view



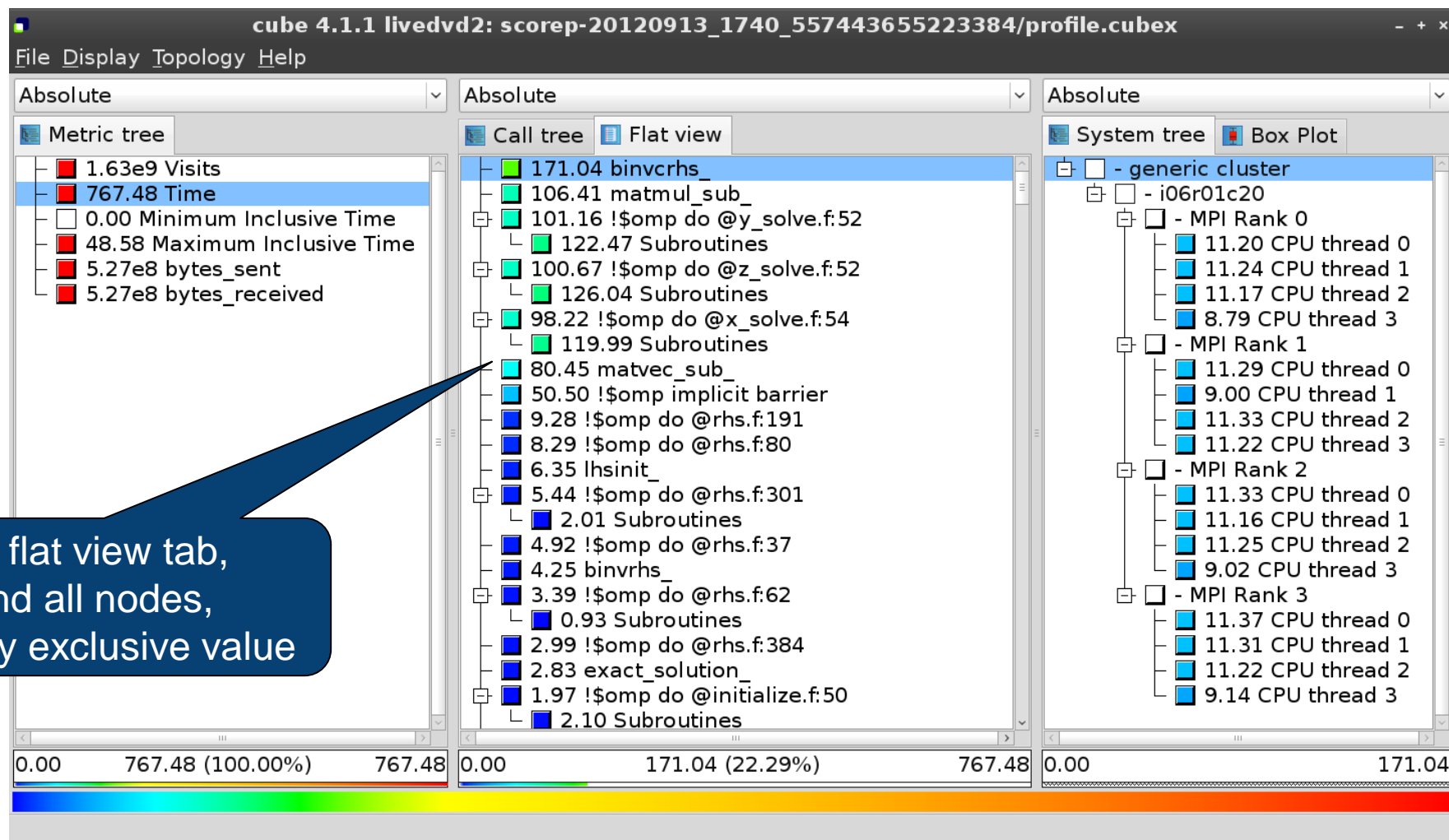
```
subroutine binvcrhs( lhs,c,r )  
  
C-----  
C-----  
  
C-----  
C  
C-----  
  
implicit none  
  
double precision pivot, coeff, lhs  
dimension lhs(5,5)  
double precision c(5,5), r(5)  
  
C-----  
C  
C-----  
  
pivot = 1.00d0/lhs(1,1)  
lhs(1,2) = lhs(1,2)*pivot  
lhs(1,3) = lhs(1,3)*pivot  
lhs(1,4) = lhs(1,4)*pivot  
lhs(1,5) = lhs(1,5)*pivot  
c(1,1) = c(1,1)*pivot  
c(1,2) = c(1,2)*pivot  
c(1,3) = c(1,3)*pivot  
c(1,4) = c(1,4)*pivot
```

☒ Read only Save Save as Font... Close

Note:

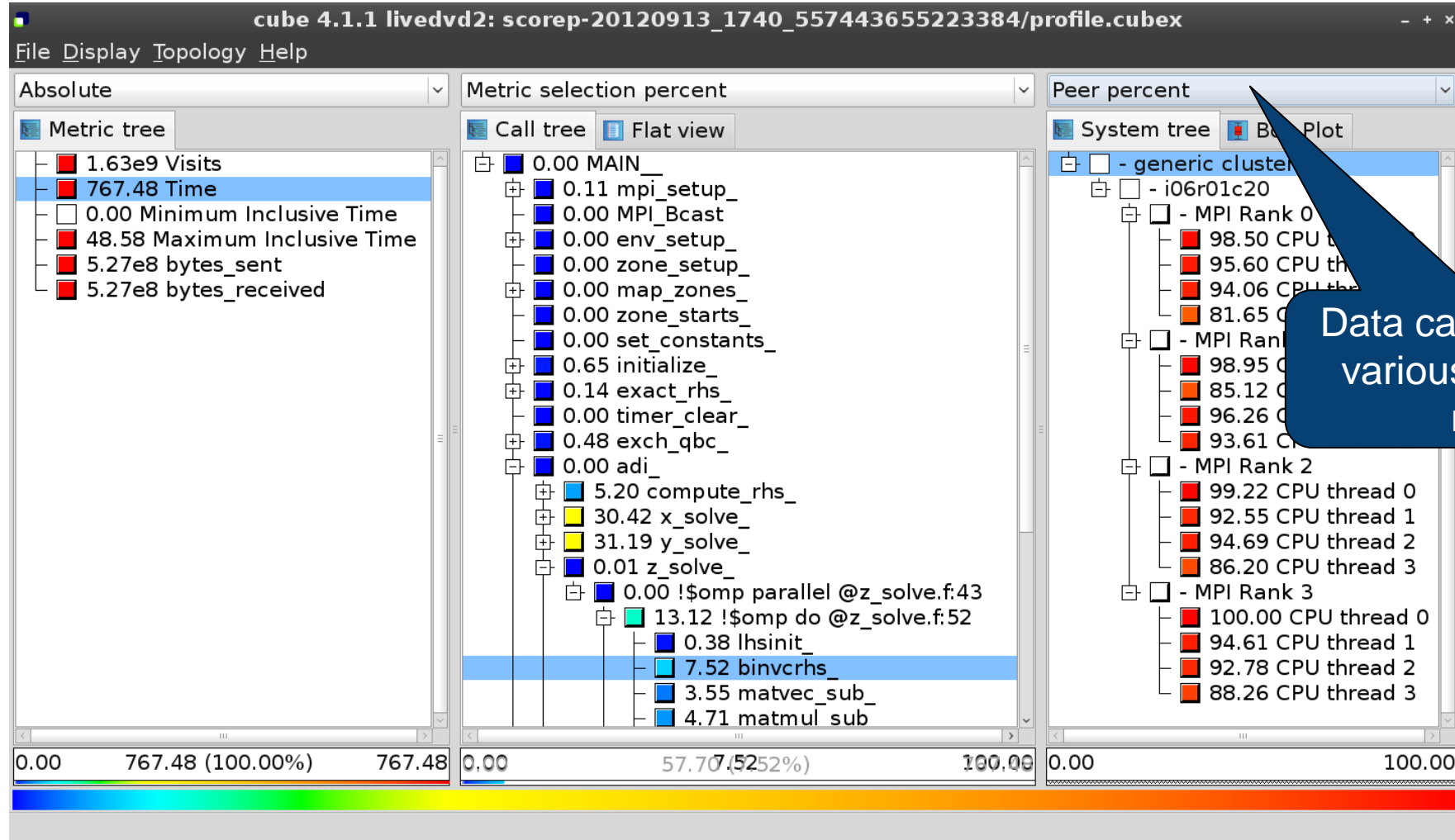
This feature depends on file and line number information provided by the instrumentation, i.e., it may not always be available

Flat profile view



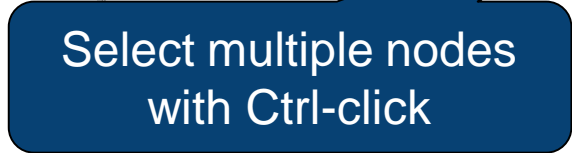


Alternative display modes

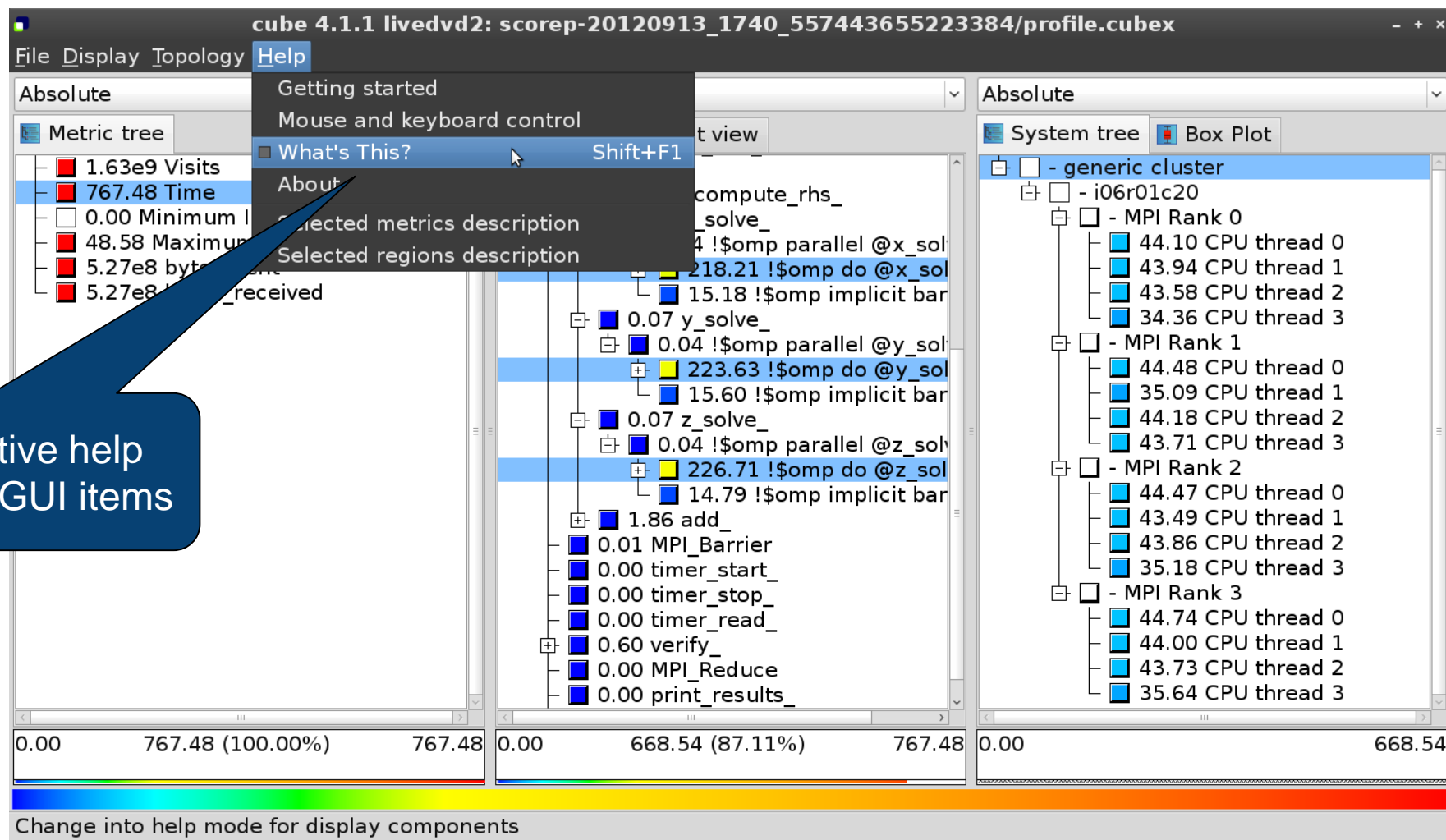


Important display modes

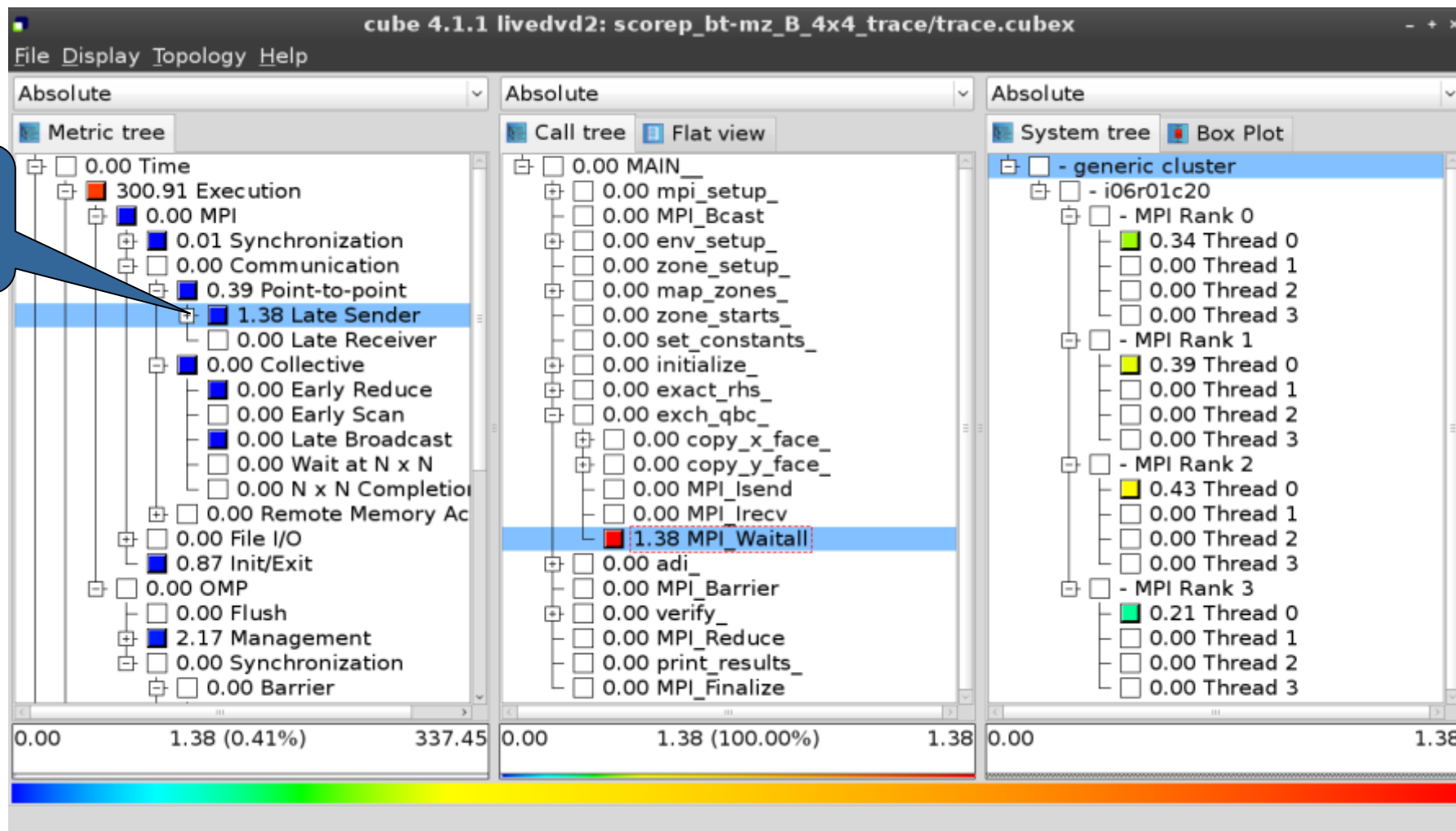
- Absolute
 - Absolute value shown in seconds/bytes/counts
- Selection percent
 - Value shown as percentage w.r.t. the selected node
“on the left” (metric/call path)
- Peer percent (system tree only)
 - Value shown as percentage relative to the maximum peer value



Context-sensitive help



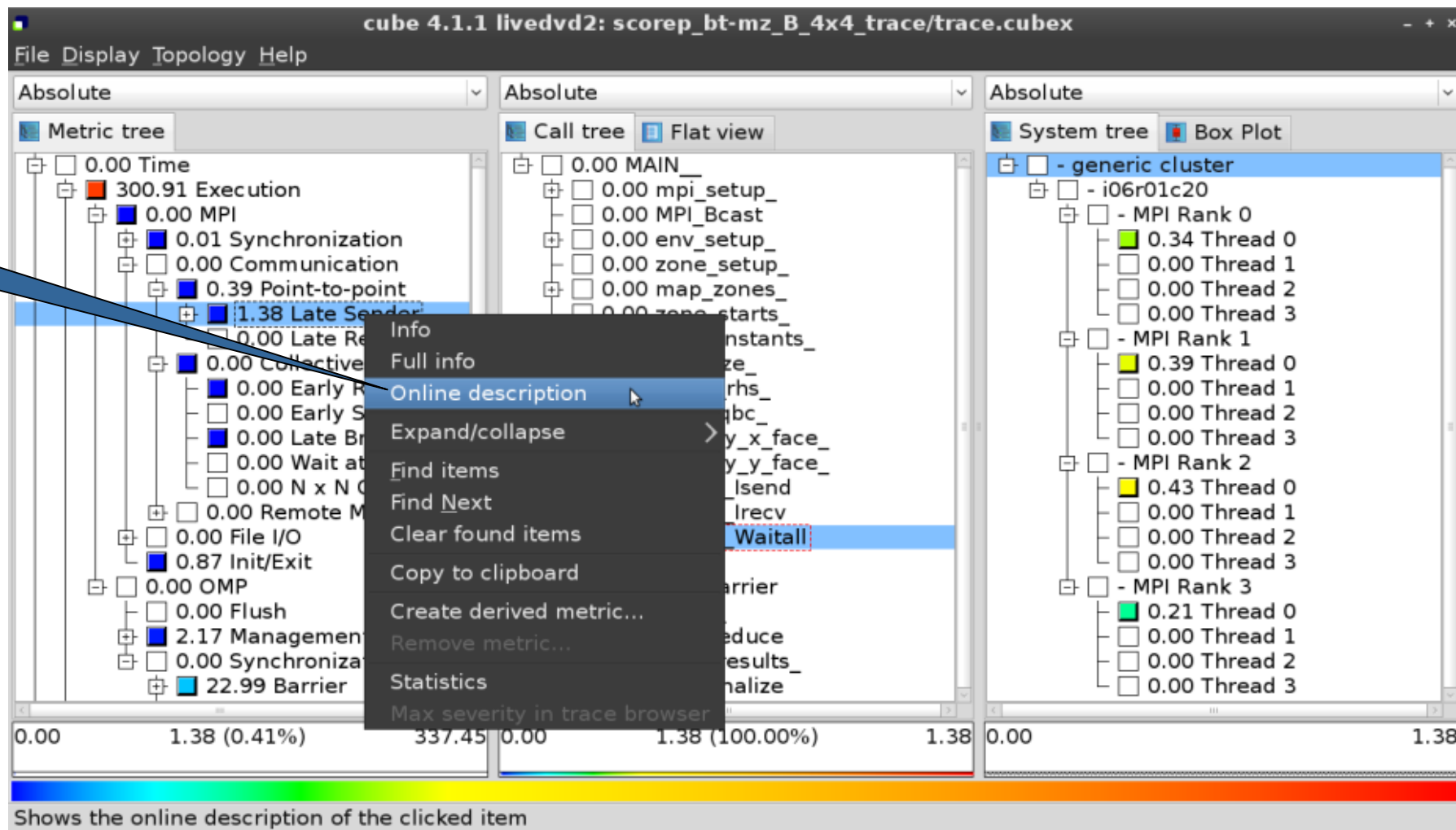
Post-processed trace analysis report



Additional trace-based metrics in metric hierarchy

Online metric description

Access online metric description via context menu

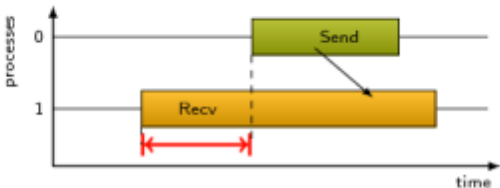


Online metric description

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.

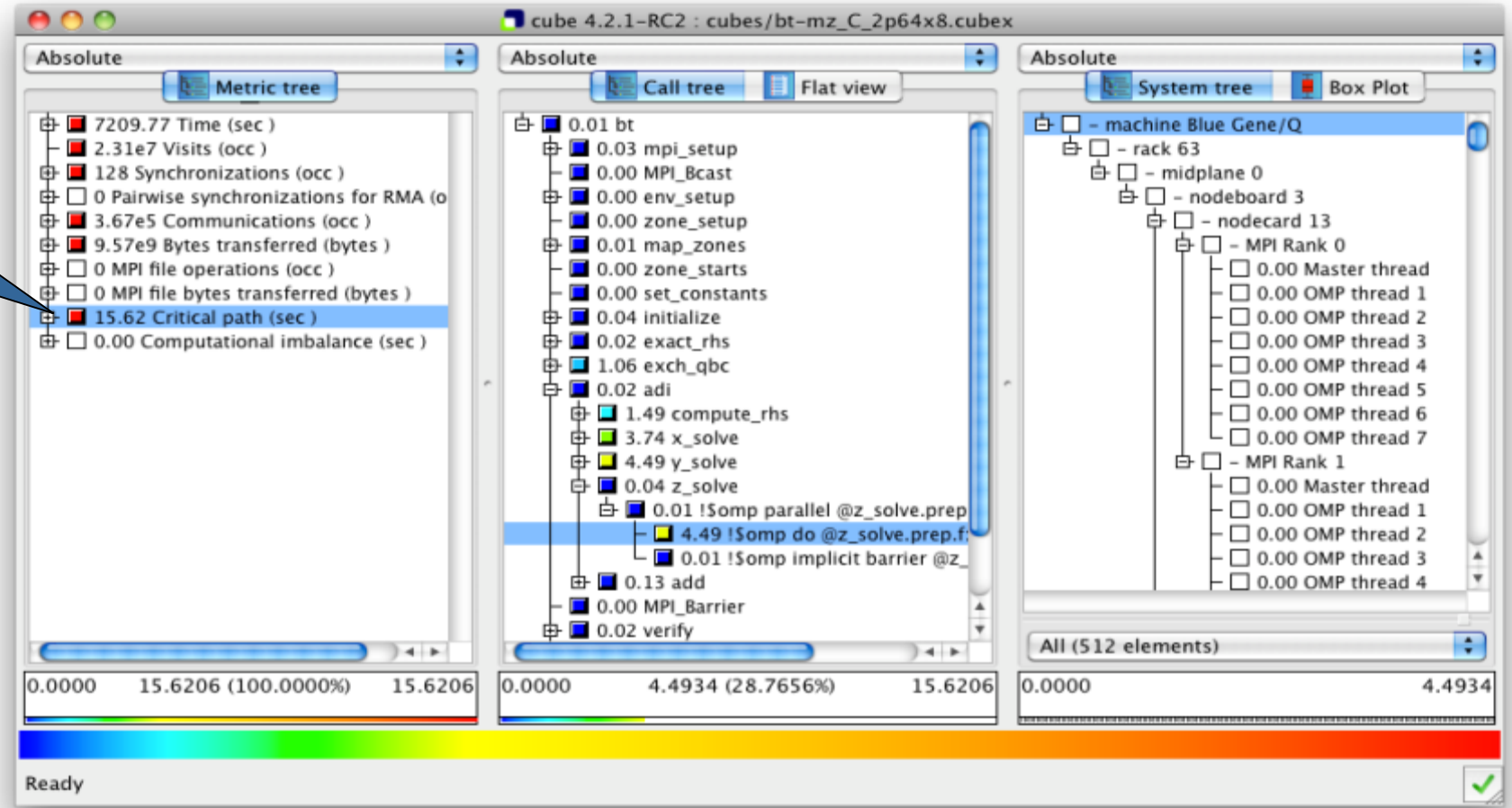
Parent:
[MPI Point-to-point Communication Time](#)

Children:

Close

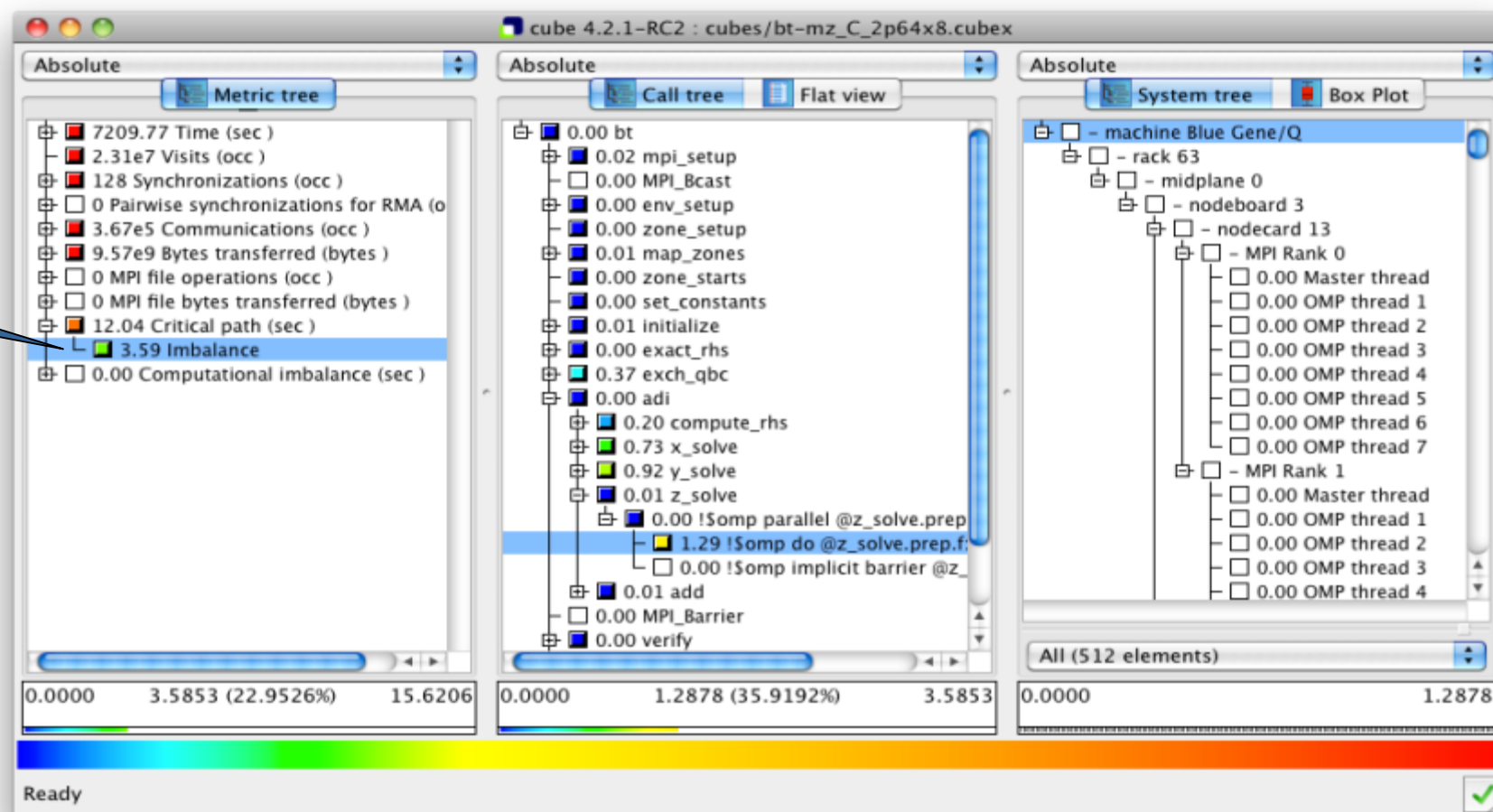
Critical-path analysis

Critical-path profile shows wall-clock time impact



Critical-path analysis

Critical-path imbalance highlights inefficient parallelism



Derived metrics

- Derived metrics are defined using CubePL expressions, e.g.:

`metric::time(i)/metric::visits(e)`

- Values of derived metrics are not stored, but calculated on-the-fly
- Types of derived metrics:
 - Prederived: evaluation of the CubePL expression is performed before aggregation
 - Postderived: evaluation of the CubePL expression is performed after aggregation

- Examples:

- “Average execution time”: Postderived metric with expression

`metric::time(i)/metric::visits(e)`

- “Number of FLOP per second”: Postderived metric with expression

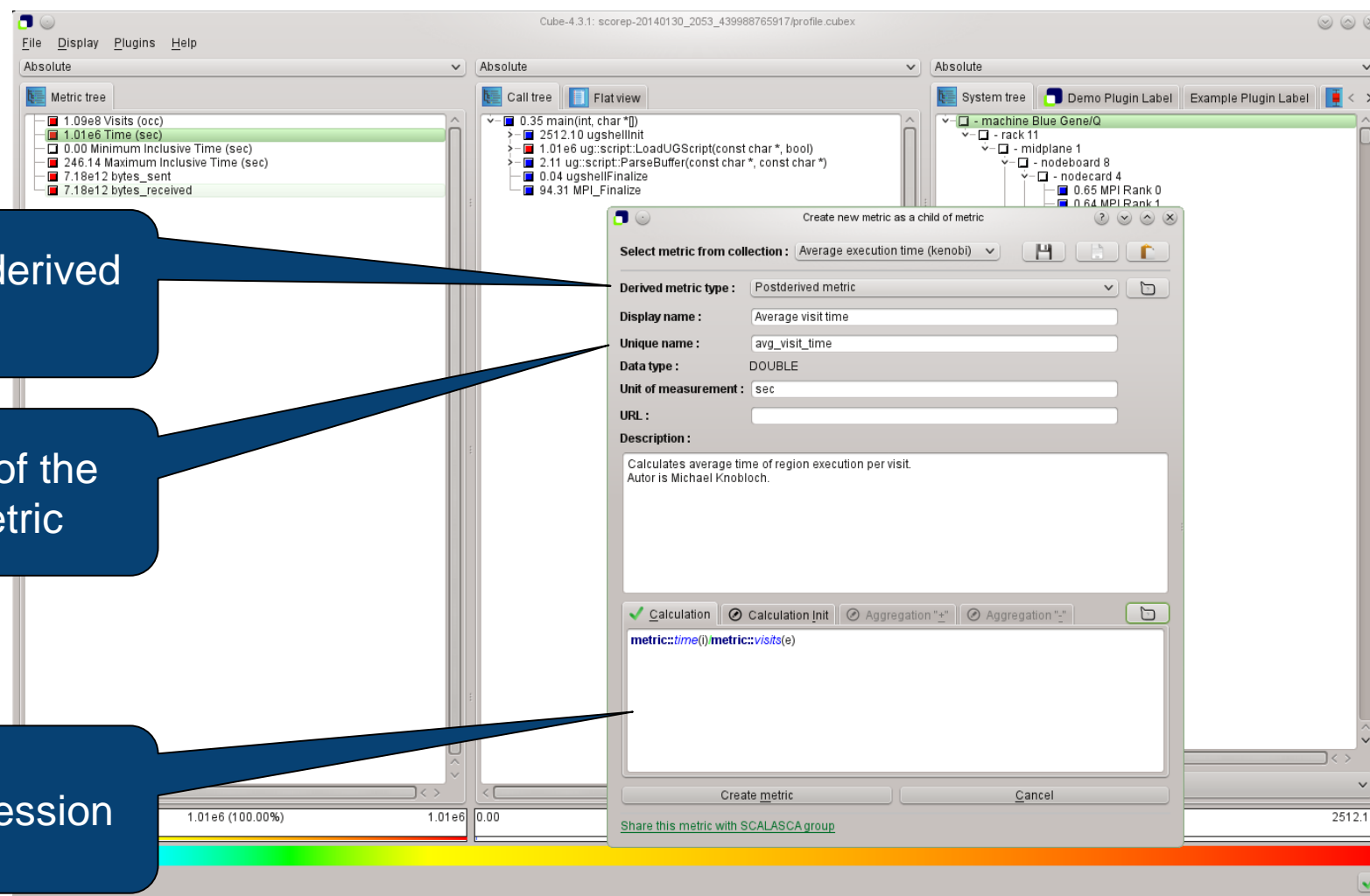
`metric::FLOP()/metric::time()`

Derived metrics in Cube GUI

Collection of derived metrics

Parameters of the derived metric

CubePL expression



Example: FLOPS based on PAPI_FP_OPS and time

The image displays the Cube-4.3.1 performance analysis tool interface, showing the configuration of a derived metric and its hierarchical breakdown.

Left Panel: Edit metric FLOPS (on froggy1)

- Select metric from collection: --- please select ---
- Derived metric type: Postderived metric
- Display name: FLOPS
- Unique name: flops
- Data type: DOUBLE
- Unit of measurement:
- URL:
- Description:
- Calculation: `metric::PAPI_FP_OPS()/metric::time()`
- Buttons: Edit metric, Cancel
- Share this metric with SCALASCA group

Right Panel: Cube-4.3.1: scorep_8x4_sum/profile.cubex (on froggy1)

The right panel shows the Metric tree, Call tree, and Flat view. The Metric tree is expanded, showing the following hierarchy:

- 1.17e7 Visits (occ)
- 1148.49 Time (sec)
- 0.00 Minimum Inclusive Time (sec)
- 41.57 Maximum Inclusive Time (sec)
- 0 bytes_put (bytes)
- 0 bytes_get (bytes)
- 5.75e12 PAPI_TOT_INS (#)
- 2.69e12 PAPI_TOT_CYC (#)
- 2.12e12 PAPI_FP_OPS (#)
- 3.12e9 bytes_sent (bytes)
- 3.12e9 bytes_received (bytes)
- 1.84e9 FLOPS

The Call tree shows the following hierarchy:

- 3.17e5 MAIN
- 7.04e5 mpi_setup
- 6.34e4 MPI_Bcast
- 2.05e5 env_setup
- 7.39e5 zone_setup
- 9.31e5 map_zones
- 9.39e4 zone_starts
- 6.16e5 set_constants
- 5.91e8 initialize
- 0.00 exact_rhs
- 145.62 !\$omp parallel @exact_r...
- 2.54e4 !\$omp do @exact_r...
- 9.65e8 !\$omp do @exact_r...
- 9.62e8 !\$omp do @exact_r...
- 8.14e8 !\$omp do @exact_r...
- 1.21e5 !\$omp do @exact_r...
- 0.00 !\$omp implicit barrier...
- 6.23e4 exch_qbc
- 1.94e9 adi
- 2.19e5 MPI_Barrier
- 1.92e9 <<bt_iter>> (200 itera...
- 1.98e8 verify
- 1.05e5 MPI_Reduce

The System tree shows the following hierarchy:

- machine Linux
- node frog6
- MPI Rank 0
- 1.17e9 Master thread
- 9.43e8 OMP thread 1
- 9.47e8 OMP thread 2
- 9.47e8 OMP thread 3
- MPI Rank 1
- 1.17e9 Master thread
- 9.87e8 OMP thread 1
- 9.68e8 OMP thread 2
- 9.72e8 OMP thread 3
- MPI Rank 2
- 1.10e9 Master thread
- 8.97e8 OMP thread 1
- 8.77e8 OMP thread 2
- 8.76e8 OMP thread 3
- MPI Rank 3
- 1.09e9 Master thread
- 9.06e8 OMP thread 1
- 9.04e8 OMP thread 2
- 9.02e8 OMP thread 3

The bottom status bar shows the selected metric: `Selected "$omp do @exact_rhs.f:46"`.

CUBE algebra utilities

- Extracting solver sub-tree from analysis report

```
% cube_cut -r '<<ITERATION>>' scorep_bt-mz_B_mic15p30x4_sum/profile.cubex  
Writing cut.cubex... done.
```

- Calculating difference of two reports

```
% cube_diff scorep_bt-mz_B_mic15p30x4_sum/profile.cubex cut.cubex  
Writing diff.cubex... done.
```

- Additional utilities for merging, calculating mean, etc.
- Default output of `cube_utility` is a new report *utility.cubex*
- Further utilities for report scoring & statistics
- Run utility with `-h` (or no arguments) for brief usage info

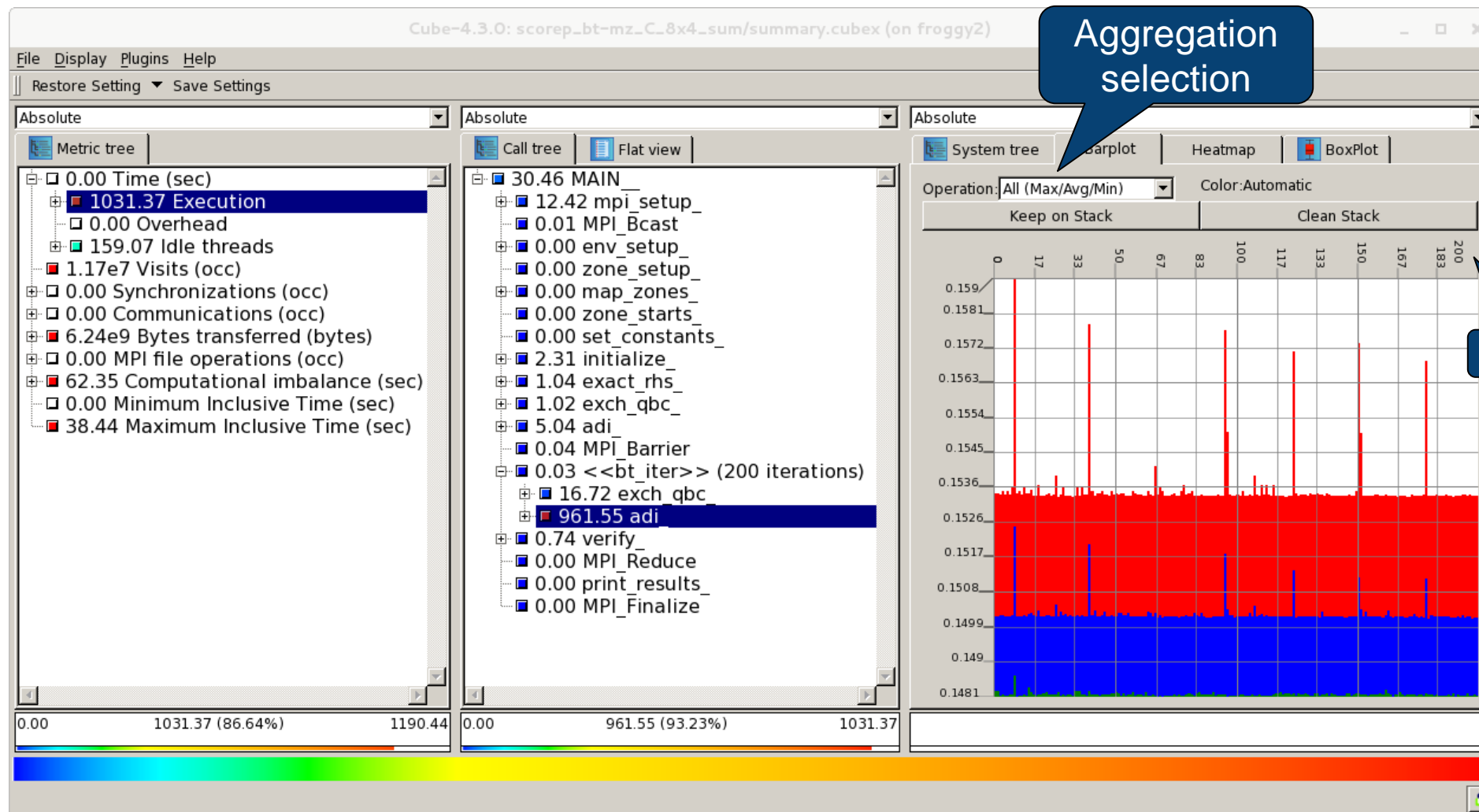
Iteration profiling

- Show time dependent behavior by “unrolling” iterations
- Preparations:
 - Mark loop body by using Score-P instrumentation API in your source code

```
SCOREP_USER_REGION_DEFINE( scorep_bt_loop )  
SCOREP_USER_REGION_BEGIN( scorep_bt_loop, "<<bt_iter>>", SCOREP_USER_REGION_TYPE_DYNAMIC )  
SCOREP_USER_REGION_END( scorep_bt_loop )
```

- Result in the Cube profile:
 - Iterations shown as separate call trees
 - Useful for checking results for specific iterations
 - or
 - Select your user-instrumented region and mark it as loop
 - Choose “Hide iterations”
 - View the Barplot statistics or the (thread x iterations) Heatmap

Iteration profiling: Barplot





Score-P/Intel compiler filter creation plugin

The screenshot displays the Score-P Configuration tool interface. The left pane shows a 'Metric tree' with various performance metrics. The middle pane shows a 'Call tree' with a selected node '0.96 Parallel::init'. The right pane shows the 'Score-P Configuration' tab with 'Measurement' and 'Filter rules' sections.

Metric tree (Left Pane):

- 1726.99 Time (sec)
- 1.11e7 Visits (occ)
- 16 MPI synchronizations (occ)
- 0 MPI pair-wise one-sided synchronizations (occ)
- 1.11e6 MPI communications (occ)
- 0 MPI file operations (occ)
- 1.92e9 MPI bytes transferred (bytes)
- 725.52 Delay costs (sec)
- 165.76 MPI point-to-point wait states (propaga)
- 165.76 MPI point-to-point wait states (direct vs)
- 53.97 Critical path (sec)
- 1726.95 Performance impact (sec)
- 39.37 Computational imbalance (sec)
- 1.11e9 Total size of full trace (bytes)
- 1.00e9 Total size of reduced trace (bytes)
- 11.15 Total measurement overhead (sec)
- 9.81 Reduced measurement overhead (sec)

Call tree (Middle Pane):

- 0.00 _sti_\$_E
- 5.49 main
 - 0.96 Parallel::init
 - 1.09 Driver::Driver
 - 0.36 Driver::run
 - 0.58 Hydro::writeEnergyCheck
 - 0.16 MPI_Barrier
 - 335.66 Driver::calcGlobalDt
 - 1382.69 Hydro::doCycle
 - 0.00 Mesh::write
 - 0.00 Parallel::final

Score-P Configuration (Right Pane):

Measurement

Full trace size	1.11e9
Reduced trace size	1.00e9
SCOREP_TOTAL_MEMORY	1.30e8
Measurement overhead	0.57 %

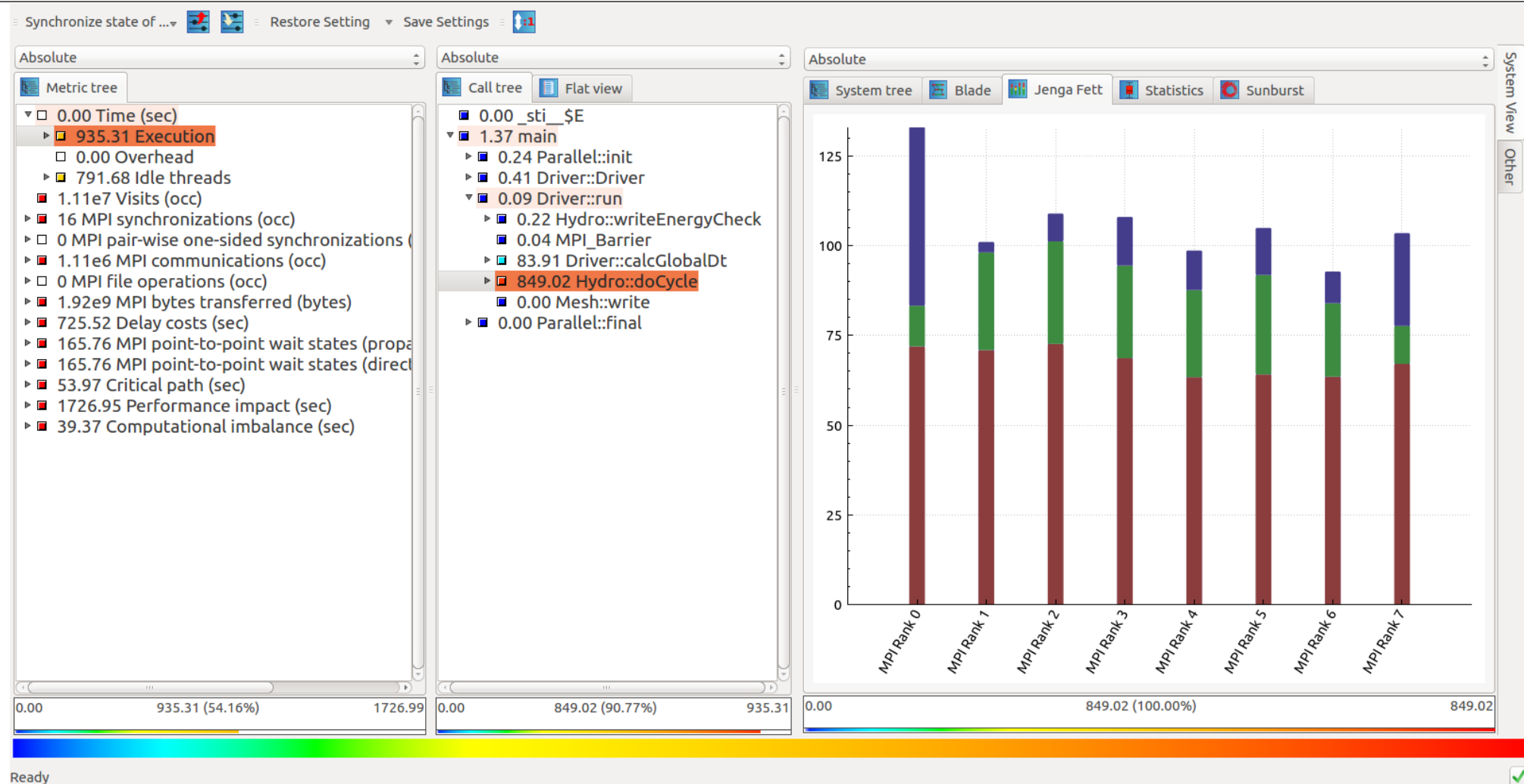
Filter rules

Exclude Region ☒ Parallel??init

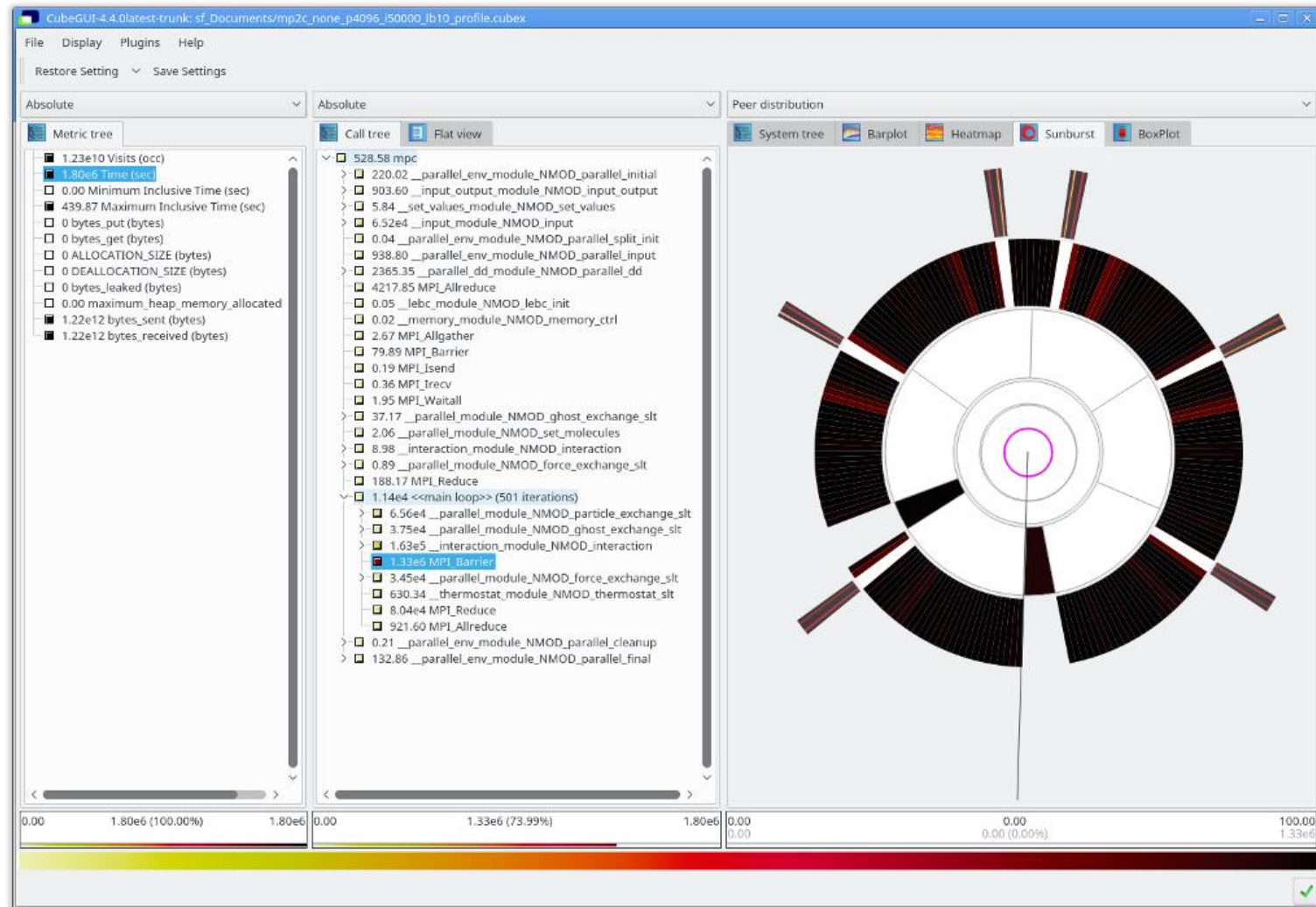
Exclude File ☒ /pylon5/ac560tp/zhukov/tests/PENNANT-pennant_v0.9/src/Hydro.cc

Selected "Parallel::init"

Metrics correlation explorer plugin

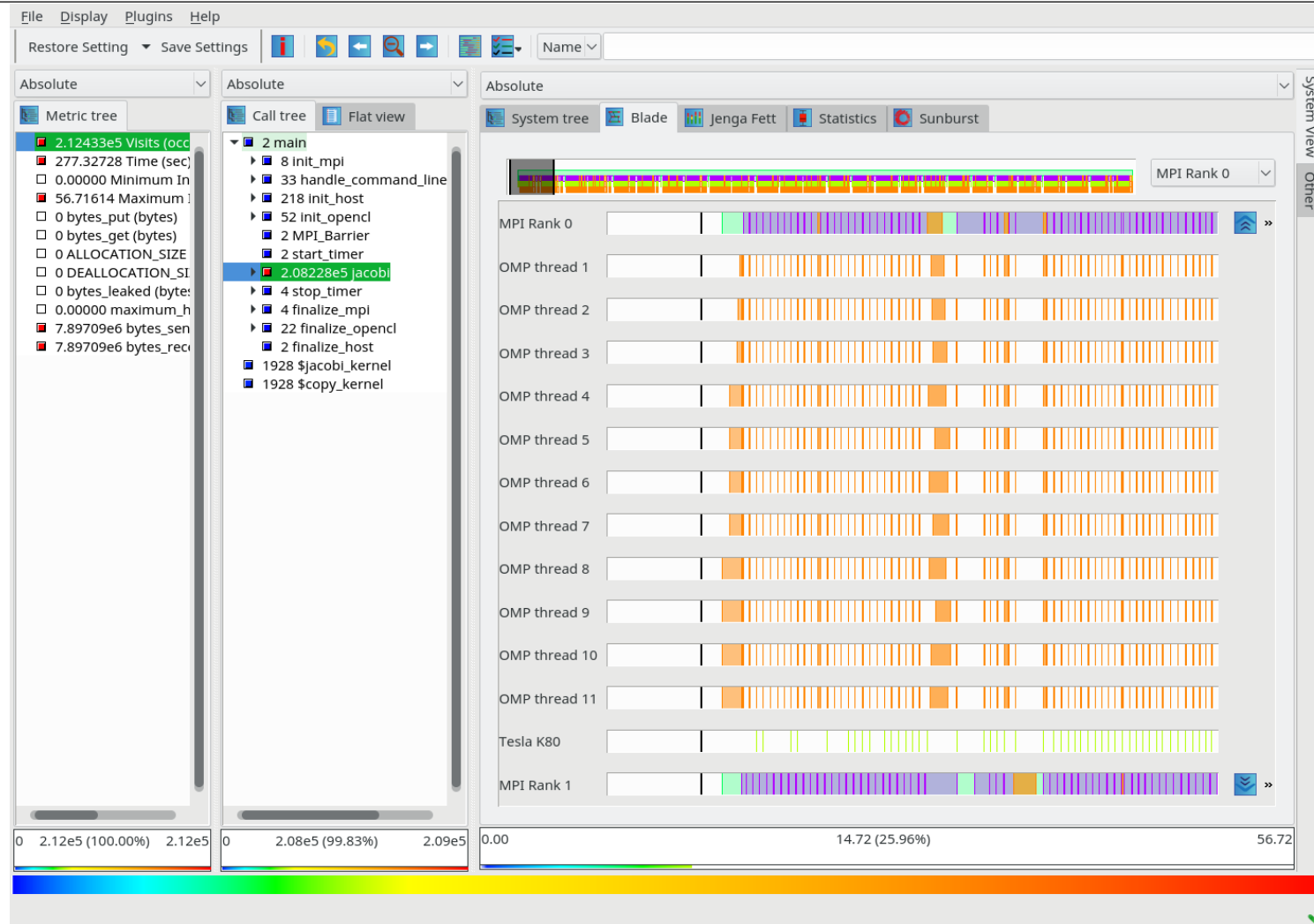


System sunburst plugin

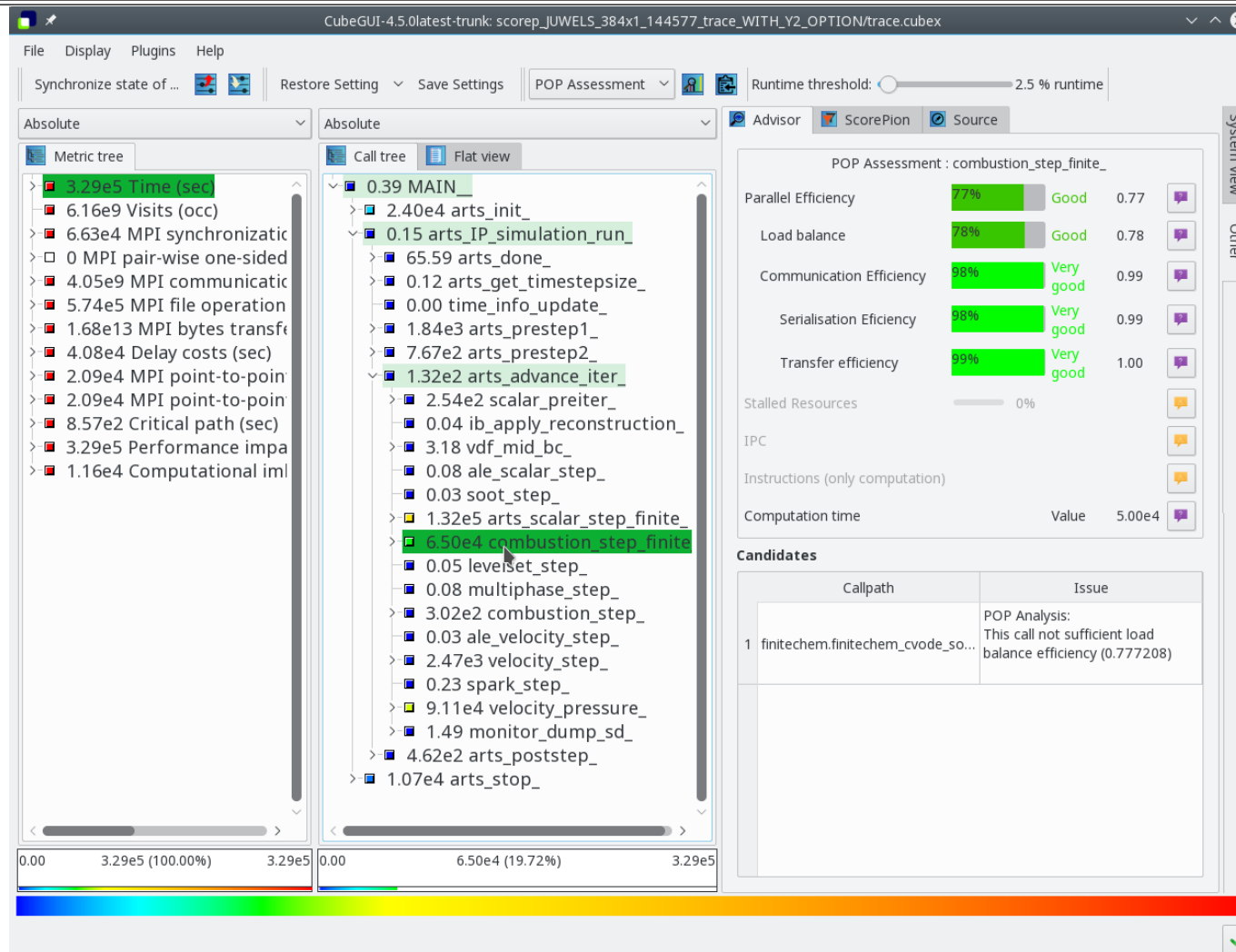




Simple trace explorer plugin (experimental)



Performance analysis plugin



Cube: Further information

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

