

Analysis report examination with Cube

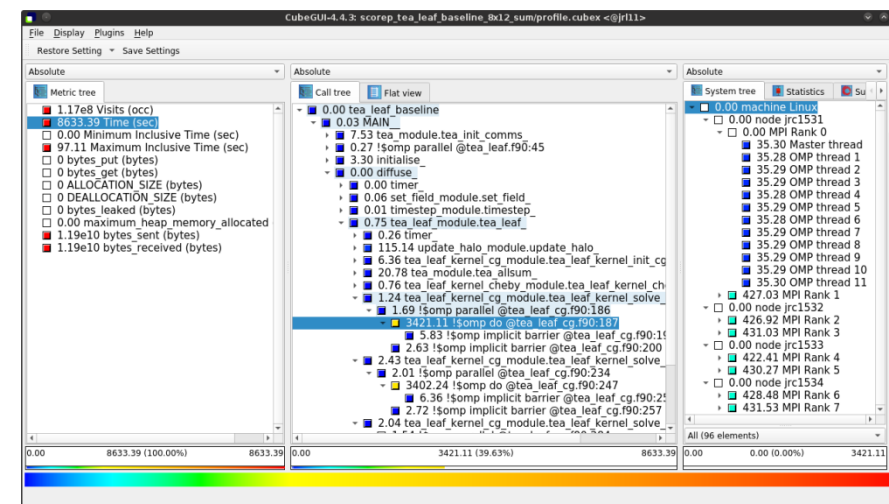
Brian Wylie
Jülich Supercomputing Centre



Cube

CubeLib DOI 10.5281/zenodo.7737408
 CubeGUI DOI 10.5281/zenodo.7737411

- Parallel program analysis report exploration tools
 - Libraries for XML+binary report reading & writing
 - Algebra utilities for report processing
 - GUI for interactive analysis exploration
 - Requires Qt \geq 5
- Originally developed as part of the Scalasca toolset
- Now available as a separate components
 - Can be installed independently of Score-P, e.g., on laptop or desktop
 - Latest release: Cube v4.8.2 (Sept 2023)



Note: source distribution tarballs for Linux, as well as binary packages provided for Windows & MacOS, from www.scalasca.org website in software/Cube-4x

Cube GUI

mailto: scalasca@fz-juelich.de



- Run **remote** (e.g. **Jupyter-JSC**)
 - start Jupyter-JSC and then start Xpra desktop
 - load CubeGUI module and start cube

```
[turpanlogin~]$ module load CubeGUI  
[turpanlogin~]$ cube ./scorep-*/profile.cubex
```

- Run **remote** (**ssh**)
 - start X server (e.g., Xming) locally
 - connect to system with X forwarding enabled
 - load cube module and start cube remotely

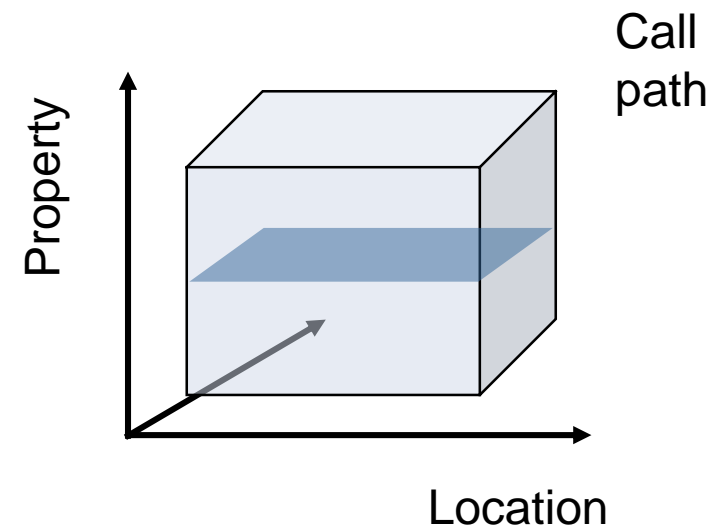
```
desk$ ssh -X <yourid>@turpan  
Welcome to Turpan ...  
[turpanlogin~]$ source /tmpdir/vi-hps/opt/setup.sh  
[turpanlogin~]$ module load cube  
[turpanlogin~]$ cube ./scorep-*/profile.cubex
```

- Install & run **local**
 - install Cube GUI locally on desktop
 - binary packages available for MacOS & Windows and externally provided by OpenHPC and various Linux distributions
 - source package available for Linux, requires Qt
 - configure/build/install manually or use your favourite framework (e.g. Spack or EasyBuild)
 - copy .cubex file (or entire scorep directory) to desktop from remote system
OR locally mount remote filesystem
 - start cube locally

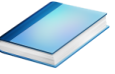
```
desk$ mkdir $HOME/mnt  
desk$ sshfs [user@]remote.sys:[dir] $HOME/mnt  
desk$ cd $HOME/mnt  
desk$ cube ./scorep-*/profile.cubex
```

Analysis presentation and exploration

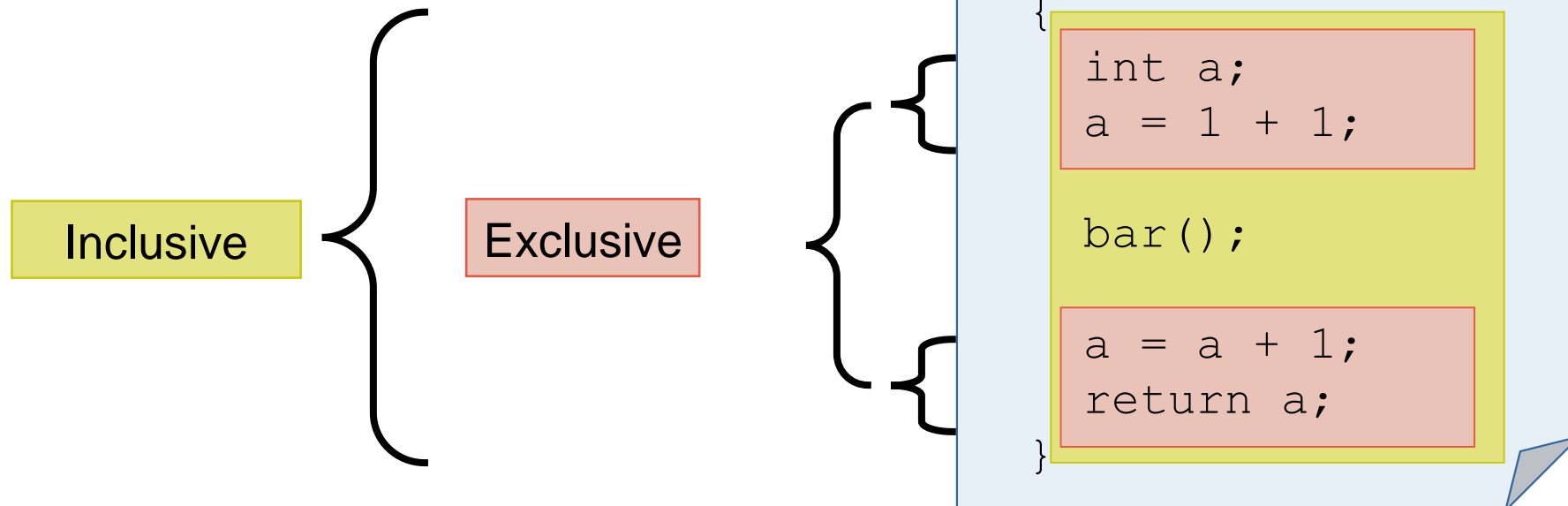
- Representation of values (severity matrix) on three hierarchical axes
 - Performance property (metric)
 - Call path (program location)
 - System location (process/thread)
- Three coupled tree browsers
- Cube displays severities
 - *As value*: for precise comparison
 - *As colour*: for easy identification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

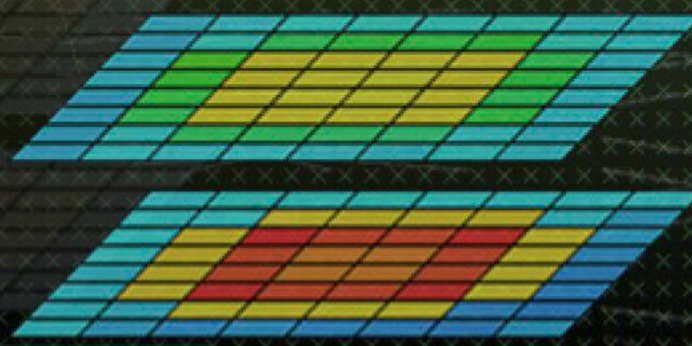


Inclusive vs. exclusive values



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





Demo: TeaLeaf case study



Case study: TeaLeaf

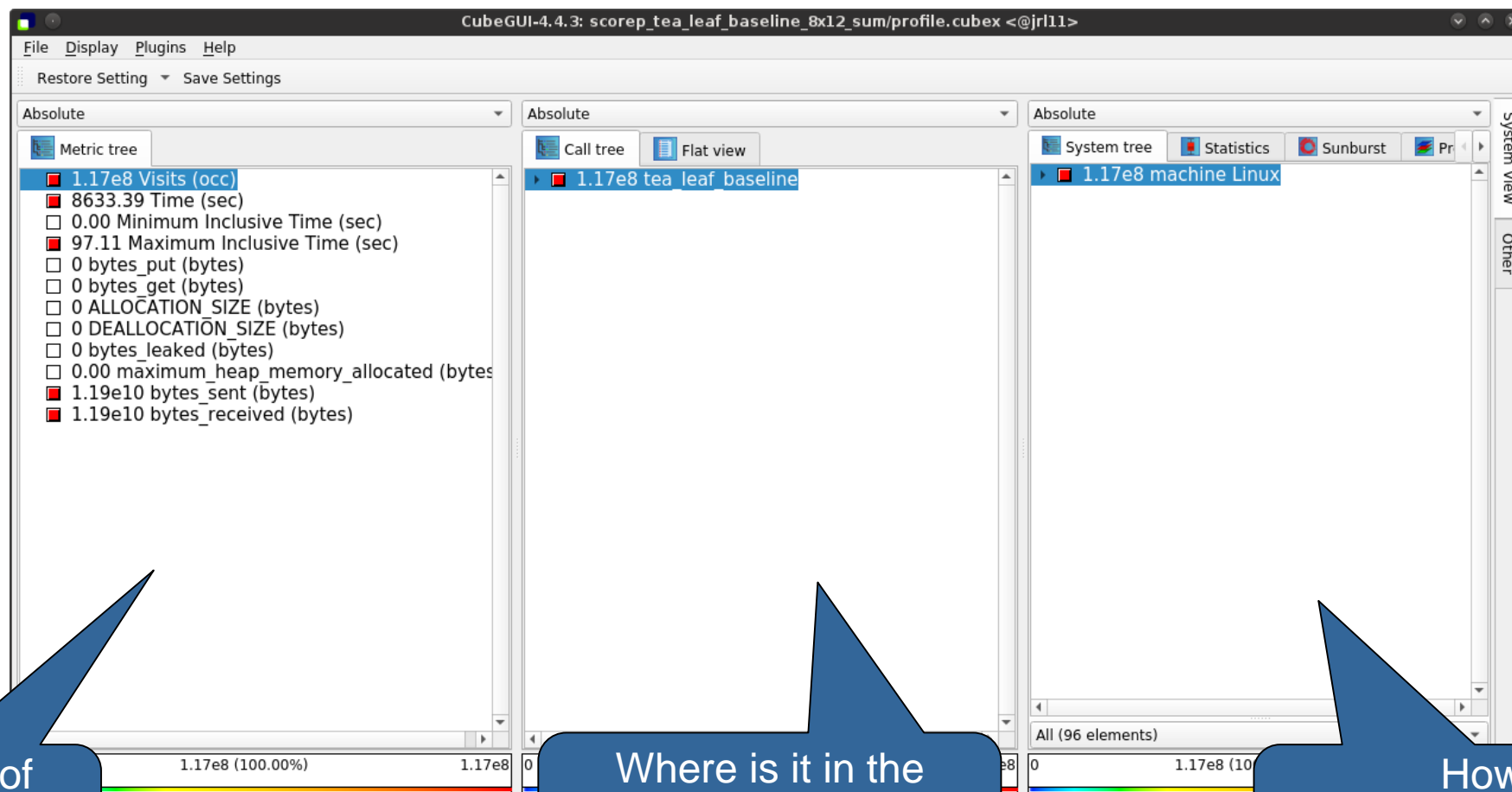
- HPC mini-app developed by the UK Mini-App Consortium
 - Solves the linear 2D heat conduction equation on a spatially decomposed regular grid using a 5 point stencil with implicit solvers
 - Part of the Mantevo 3.0 suite
 - Available on GitHub: <http://uk-mac.github.io/TeaLeaf/>

- Measurements of TeaLeaf reference v1.0 taken on Jureca cluster @ JSC
 - Using Intel 19.0.3 compilers, Intel MPI 2019.3, and Score-P 5.0
 - Run configuration
 - 8 MPI ranks with 12 OpenMP threads each



```
% cd ~/workshop-vihps/Experiments
% cube scorep_tea_leaf_baseline_8x12_sum/profile.cubex
[GUI showing summary analysis report]
```

Score-P analysis report exploration (opening view)



What kind of performance metric?

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

How is it distributed across the processes/threads?

Metric selection

CubeGUI-4.4.3: scorep_tea_leaf_baseline_8x12_sum/profile.cubex <@jrl11>

File Display Plugins Help

Restore Setting Save Settings

Absolute

Metric tree

- 1.17e8 Visits (occ)
- 8633.39 Time (sec)
- 0.00 Minimum Inclusive Time (sec)
- 97.11 Maximum Inclusive Time (sec)
- 0 bytes_put (bytes)
- 0 bytes_get (bytes)
- 0 ALLOCATION_SIZE (bytes)
- 0 DEALLOCATION_SIZE (bytes)
- 0 bytes_leaked (bytes)
- 0.00 maximum_heap_memory_allocated (bytes)
- 1.19e10 bytes_sent (bytes)
- 1.19e10 bytes_received (bytes)

Absolute

Call tree Flat view

8633.39 tea leaf baseline

Absolute

System tree Statistics Sunburst Pr

8633.39 machine Linux

System View Other

0.00 8633.39 (100.00%) 8633.39

0.00 8633.39 (100.00%) 8633.39

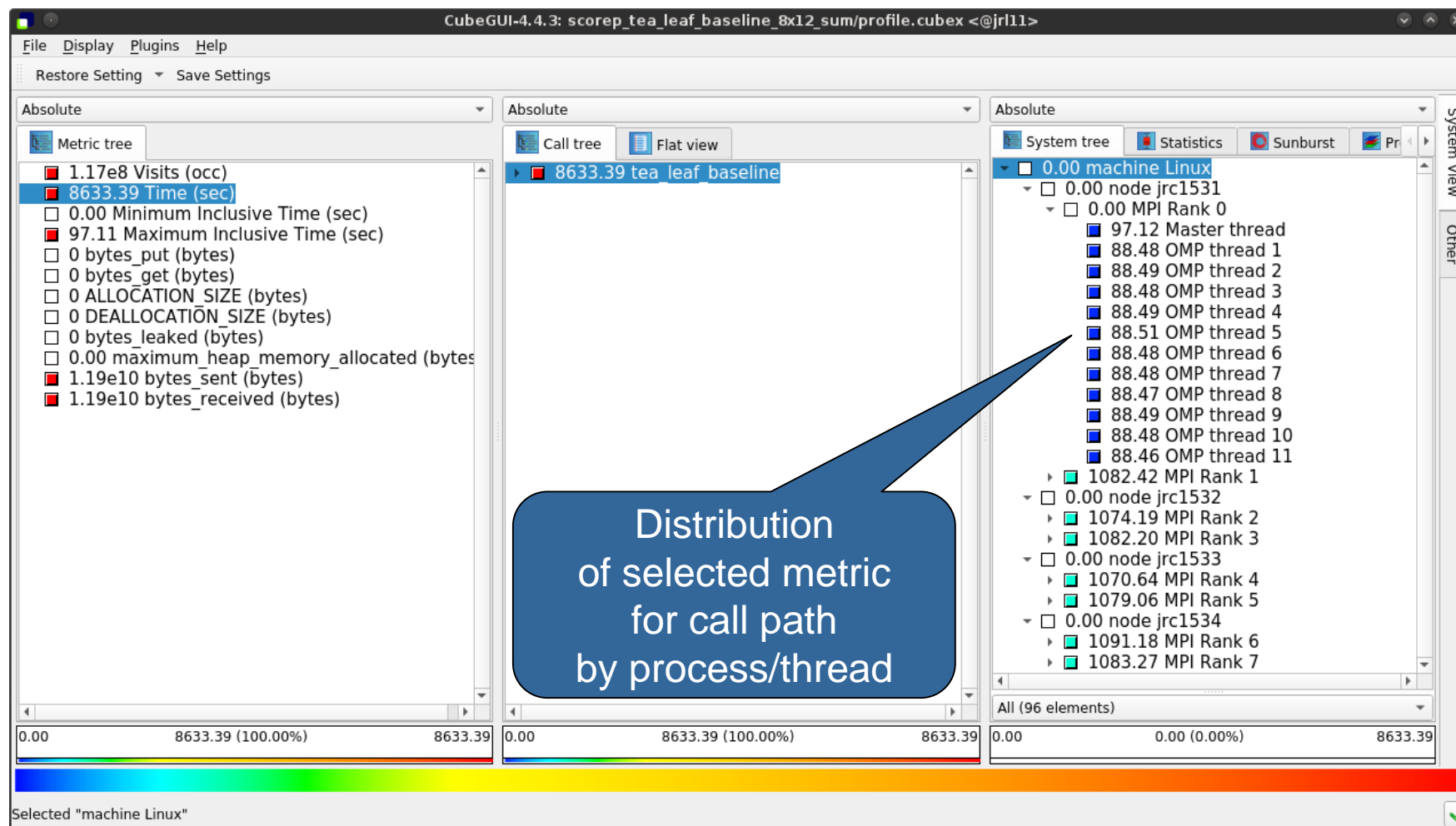
0.00 8633.39 (100.00%) 8633.39

All (96 elements)

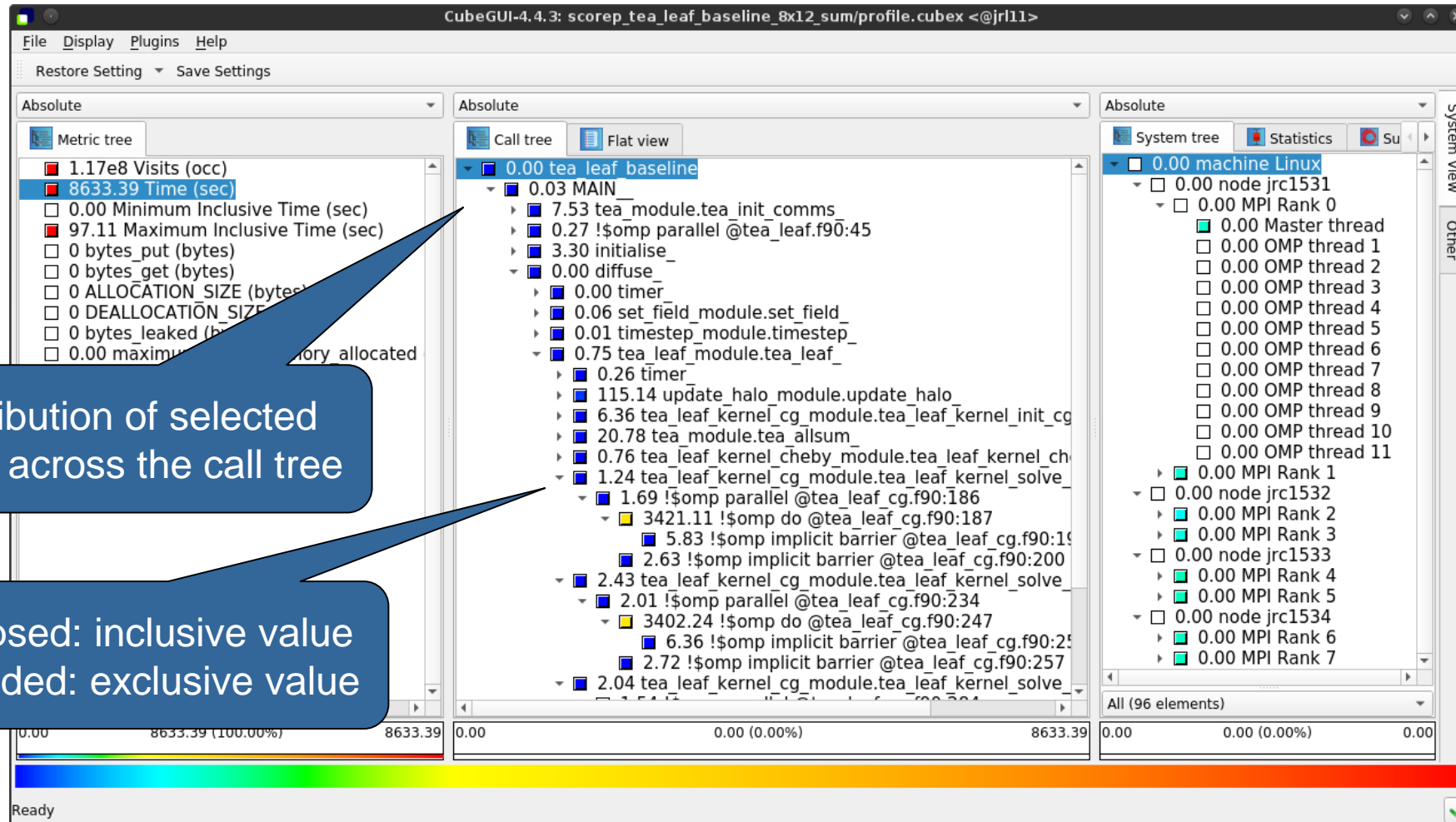
Selected "Time"

Selecting the "Time" metric shows total execution time

Expanding the system tree



Expanding the call tree



Selecting a call path

The screenshot displays the CubeGUI-4.4.3 interface for a profile named 'scorep_tea_leaf_baseline_8x12_sum/profile.cubex'. The interface is divided into three main panels:

- Metric tree (Left):** Shows various performance metrics. The '8633.39 Time (sec)' metric is highlighted in blue.
- Call tree (Middle):** Shows a hierarchical view of the call paths. The path '0.75 tea_leaf_module.tea_leaf' is expanded, and the sub-path '3421.11 !\$omp do @tea_leaf_cg.f90:187' is highlighted in blue.
- System tree (Right):** Shows the system tree structure. The path '0.00 machine Linux' is expanded, and the sub-path '427.03 MPI Rank 1' is highlighted in blue.

At the bottom of the interface, there are three progress bars corresponding to the selected paths in the three panels. The first bar shows '0.00' and '8633.39 (100.00%)'. The second bar shows '0.00' and '3421.11 (39.63%)'. The third bar shows '0.00' and '0.00 (0.00%)'. A color scale bar is located at the bottom of the interface.

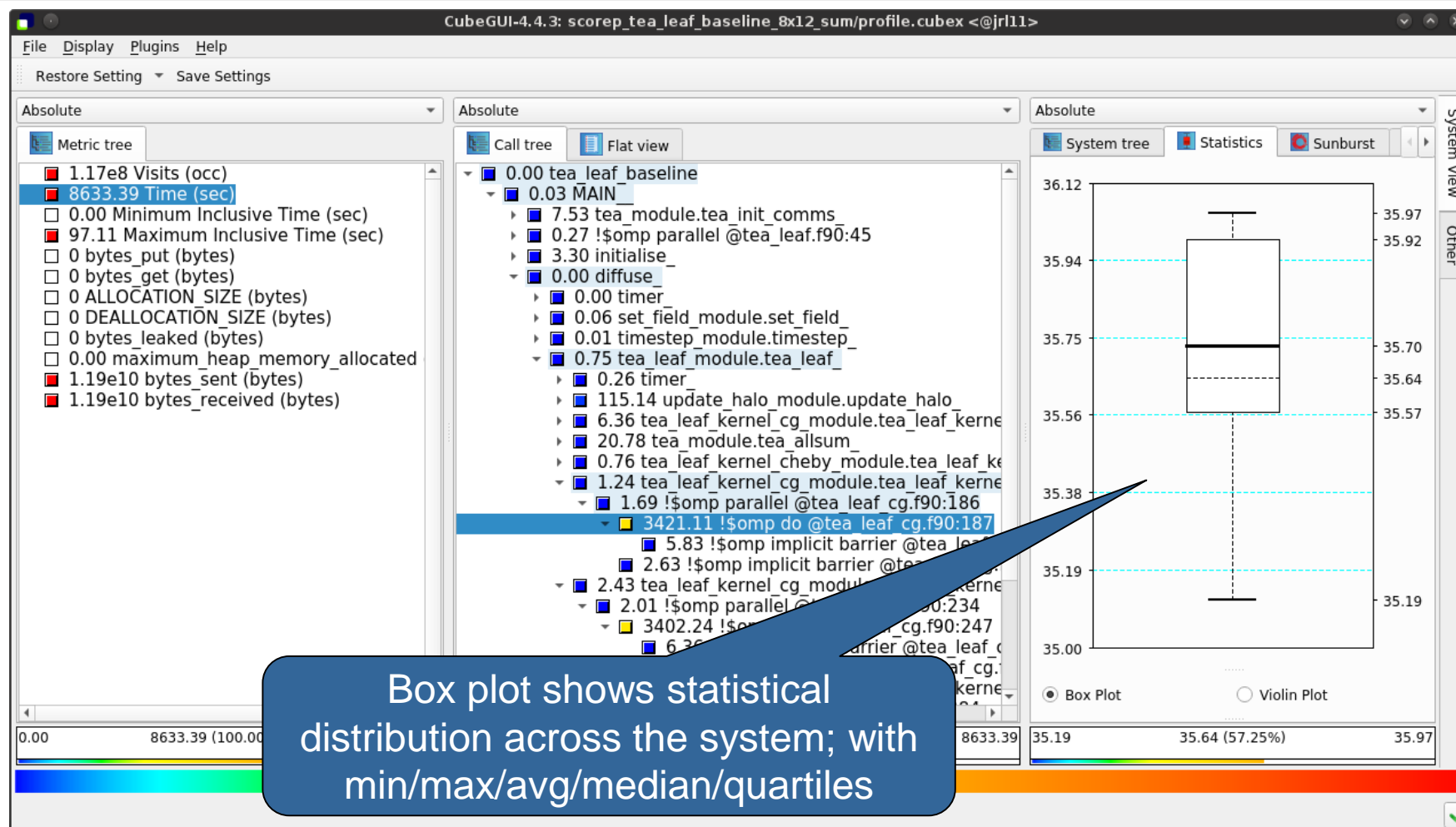
Selection updates metric values shown in columns to the right

Multiple selection

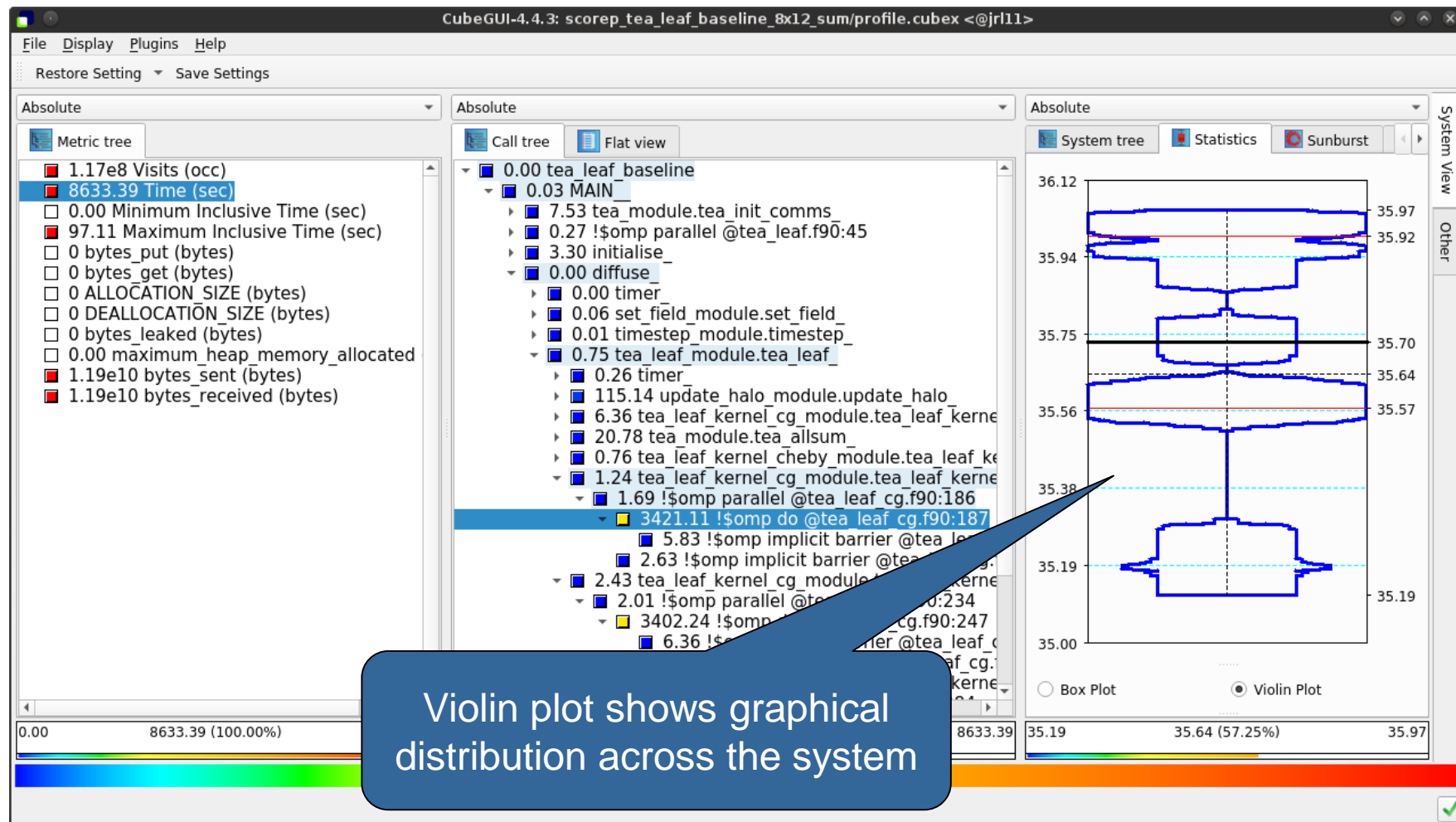
The screenshot displays the CubeGUI-4.4.3 interface for a performance profile. The main window is titled "CubeGUI-4.4.3: scorep_tea_leaf_baseline_8x12_sum/profile.cubex <@jrl11>". It features three main panels: "Metric tree" on the left, "Call tree" in the center, and "System tree" on the right. The "Call tree" panel is set to "Flat view" and shows a hierarchical structure of function calls. Several nodes are highlighted with blue selection bars, including "0.75 tea_leaf_module.tea_leaf_" and its sub-nodes, "1.69 !\$omp parallel @tea leaf cg.f90:186", "3421.11 !\$omp do @tea leaf cg.f90:187", "2.63 !\$omp implicit barrier @tea leaf cg.f90:200", "2.43 tea_leaf_kernel_cg_module.tea_leaf_kernel_solve", "2.01 !\$omp parallel @tea leaf cg.f90:234", "3402.24 !\$omp do @tea leaf cg.f90:247", "2.72 !\$omp implicit barrier @tea leaf cg.f90:257", "2.04 tea_leaf_kernel_cg_module.tea_leaf_kernel_solve", "1.54 !\$omp parallel @tea leaf cg.f90:284", "1580.11 !\$omp do @tea leaf cg.f90:294", "40.82 !\$omp implicit barrier @tea leaf cg.f90:302", "3.24 !\$omp implicit barrier @tea leaf cg.f90:302", and "1.37 tea_leaf_kernel_module.tea_leaf_kernel_finalise_". A blue callout box with a white border and a pointer to the selected nodes contains the text "Select multiple nodes with Ctrl-click". The bottom of the interface shows a progress bar with three segments: "0.00 8633.39 (100.00%) 8633.39", "0.00 8403.46 (97.34%) 8633.39", and "0.00 0.00 (0.00%) 8403.46".

Select multiple nodes with Ctrl-click

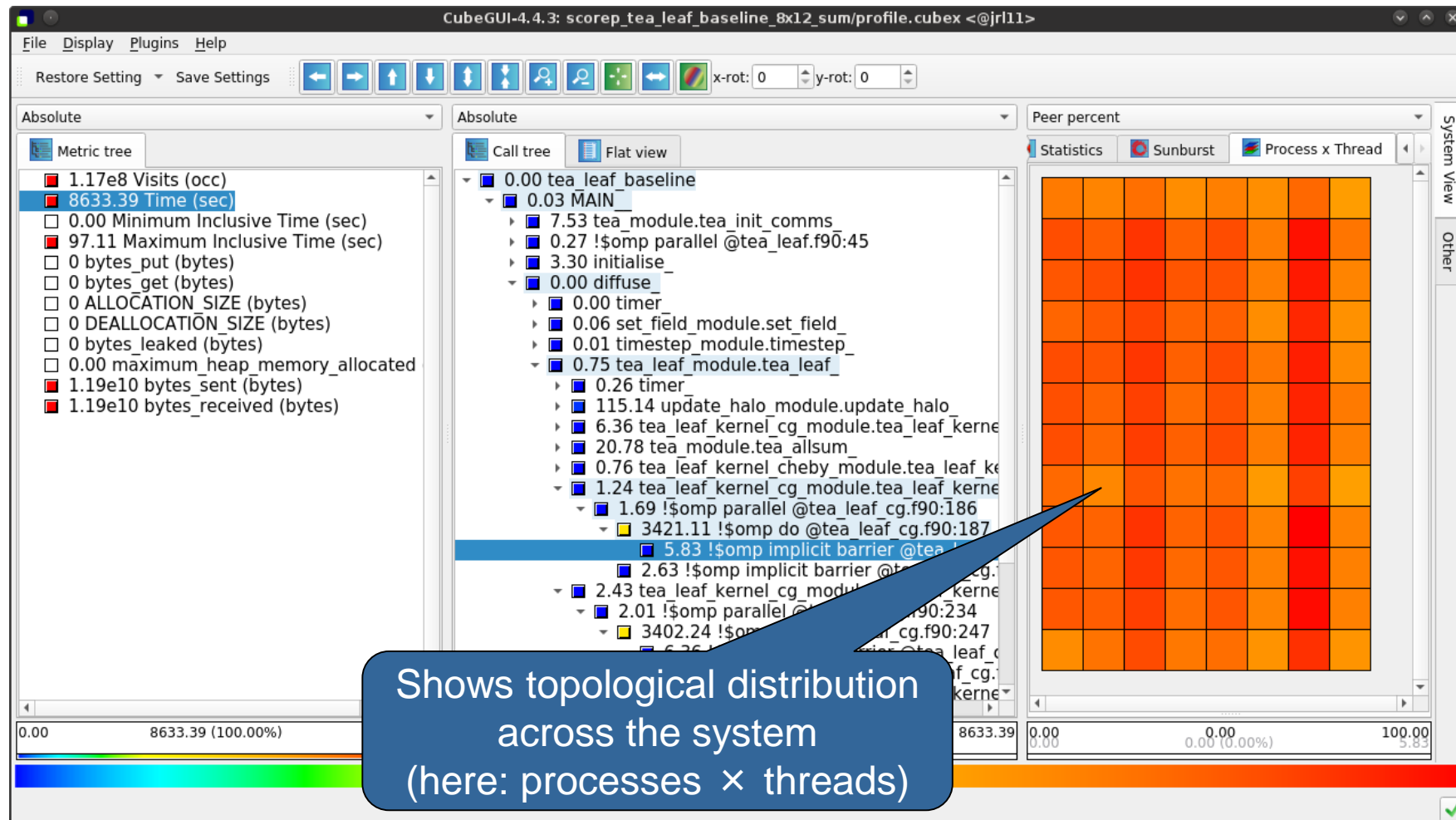
Box plot view



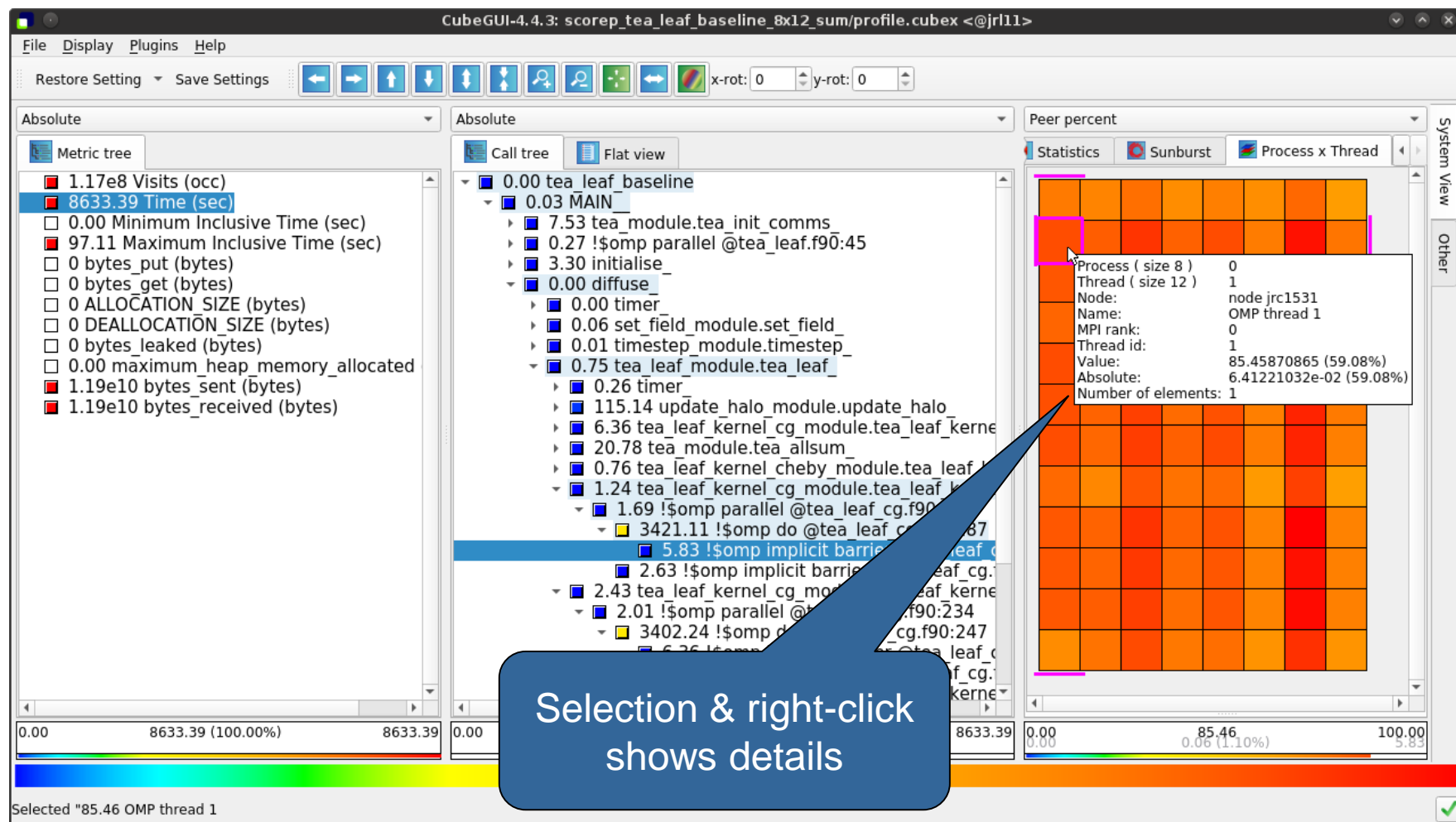
Violin plot view



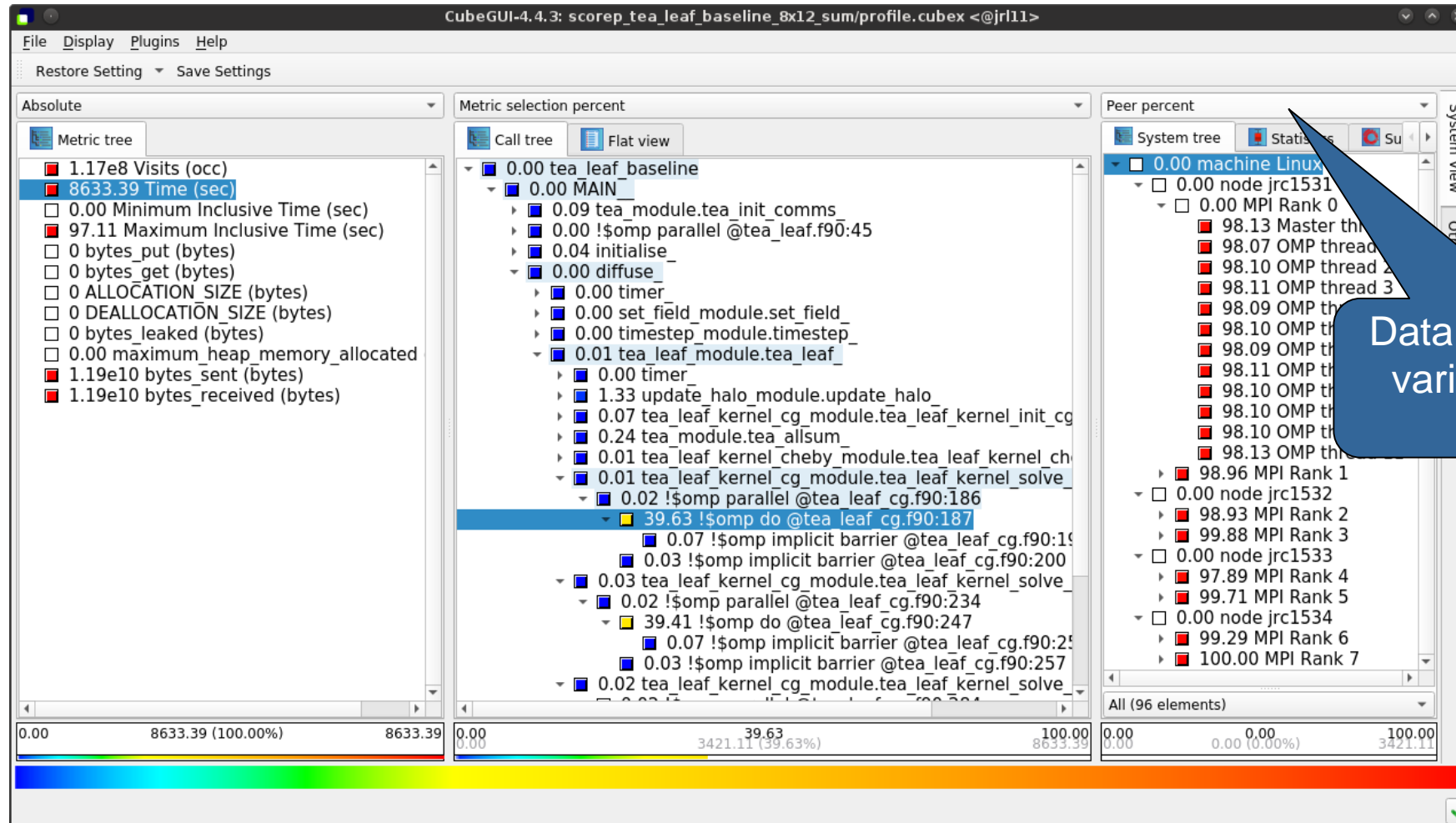
Topology view



Topology view (cont.)



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

Source-code view via context menu

The screenshot displays the CubeGUI-4.4.3 interface with three main panels: Metric tree, Call tree, and System tree. The Call tree panel is active, showing a hierarchical view of the execution profile. A node representing a parallel loop is selected, and a context menu is open over it, listing various actions such as 'Info', 'Documentation', 'Set as loop', and 'Copy to clipboard'. A blue callout box with a speech bubble points to the context menu, containing the text 'Right-click opens context menu'. The bottom of the interface features a color-coded progress bar and a status line indicating the current node's time and percentage of the total execution.

Right-click opens context menu

Source-code view

The screenshot shows the CubeGUI-4.4.3 interface with the following components:

- Metric tree (Left):** A list of performance metrics including '8633.39 Time (sec)', '97.11 Maximum Inclusive Time (sec)', and '1.19e10 bytes_sent (bytes)'.
- Call tree (Center):** A hierarchical view of the program's execution flow, showing subroutines like 'tea_leaf_baseline', 'MAIN_', and 'tea_leaf_kernel_solve_cg_fortran_calc_ur(x, m)'. The 'tea_leaf_kernel_solve_cg_fortran_calc_ur(x, m)' routine is highlighted with a blue bar.
- Source view (Right):** A code editor displaying Fortran source code for the selected routine. The code includes OpenMP directives and a nested loop structure. A blue callout box points to the 'Source' tab in the top right corner.

Note: This feature depends on the availability of the source code, as well as file and line number information provided by the instrumentation, i.e., it may not always be available

Context-sensitive help

The screenshot displays the CubeGUI-4.4.3 interface with the 'Help' menu open. The 'What's This?' option is selected, and a blue callout box points to it with the text: 'Context-sensitive help available for all GUI items'. The main window shows a hierarchical tree of metrics and system components. The selected metric is '39.63 !\$omp do @tea_leaf_cg.f90:187'. The interface includes a 'Metric tree' on the left, a 'System tree' on the right, and a 'Statistics' panel at the bottom. A color bar at the bottom indicates the range of values for the selected metric.

Change into help mode for display components

Scalasca report post-processing

- Scalasca's report post-processing derives additional metrics and generates a structured metric hierarchy
- Automatically run (if needed) when using the **square** convenience command:

```
% square scorep_tea_leaf_baseline_8x12_sum  
INFO: Post-processing runtime summarization report (profile.cubex)...  
INFO: Displaying ./scorep_tea_leaf_baseline_8x12_sum/summary.cubex...
```

```
[GUI showing post-processed summary analysis report]
```

Post-processed summary analysis report

Split base metrics into more specific metrics, e.g. computation vs parallelization costs

The screenshot displays the CubeGUI-4.4.3 interface for analyzing the 'scorep_tea_leaf_baseline_8x12_sum/summary.cubex' file. It features three main panels:

- Metric tree (Absolute):** Shows a hierarchical breakdown of metrics. The total time is 9289.50 seconds. The largest component is Computation at 8478.33 seconds (91.27%), followed by Idle threads at 656.11 seconds (7.06%).
- Call tree (Absolute):** Provides a detailed view of function calls. The total time is 8478.33 seconds. The dominant call is '\$omp do @tea_leaf_cg' at 3421.11 seconds (40.35%).
- System tree (Absolute):** Displays system-level details, including MPI Ranks and threads. MPI Rank 0 has a total time of 35.30 seconds, while MPI Rank 1 has a total time of 35.59 seconds.

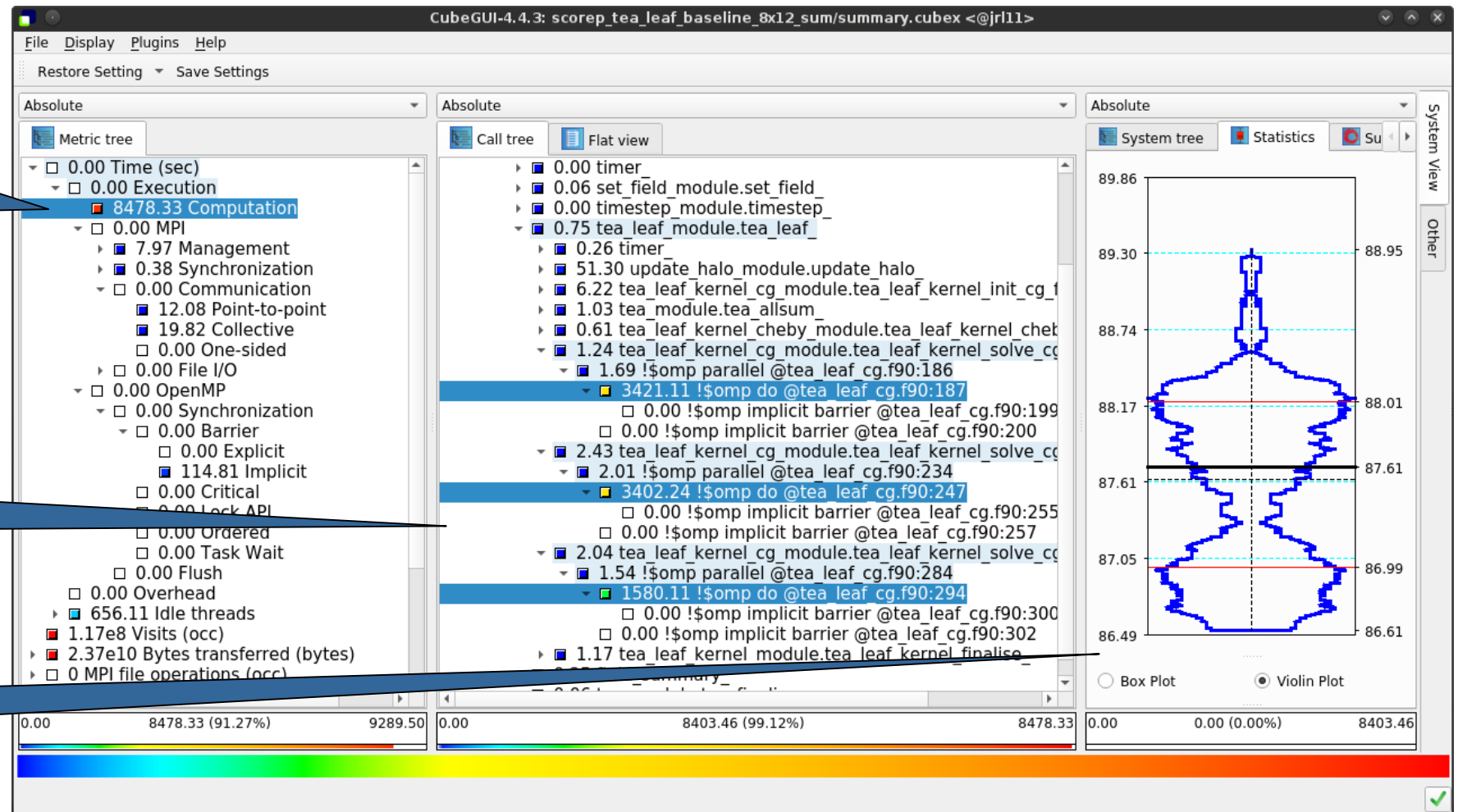
A color-coded progress bar at the bottom of the window indicates the relative contribution of various metrics to the total execution time.

TeaLeaf summary report analysis (I)

91% of the execution time is computation...

...almost entirely spent in 3 OpenMP do loops...

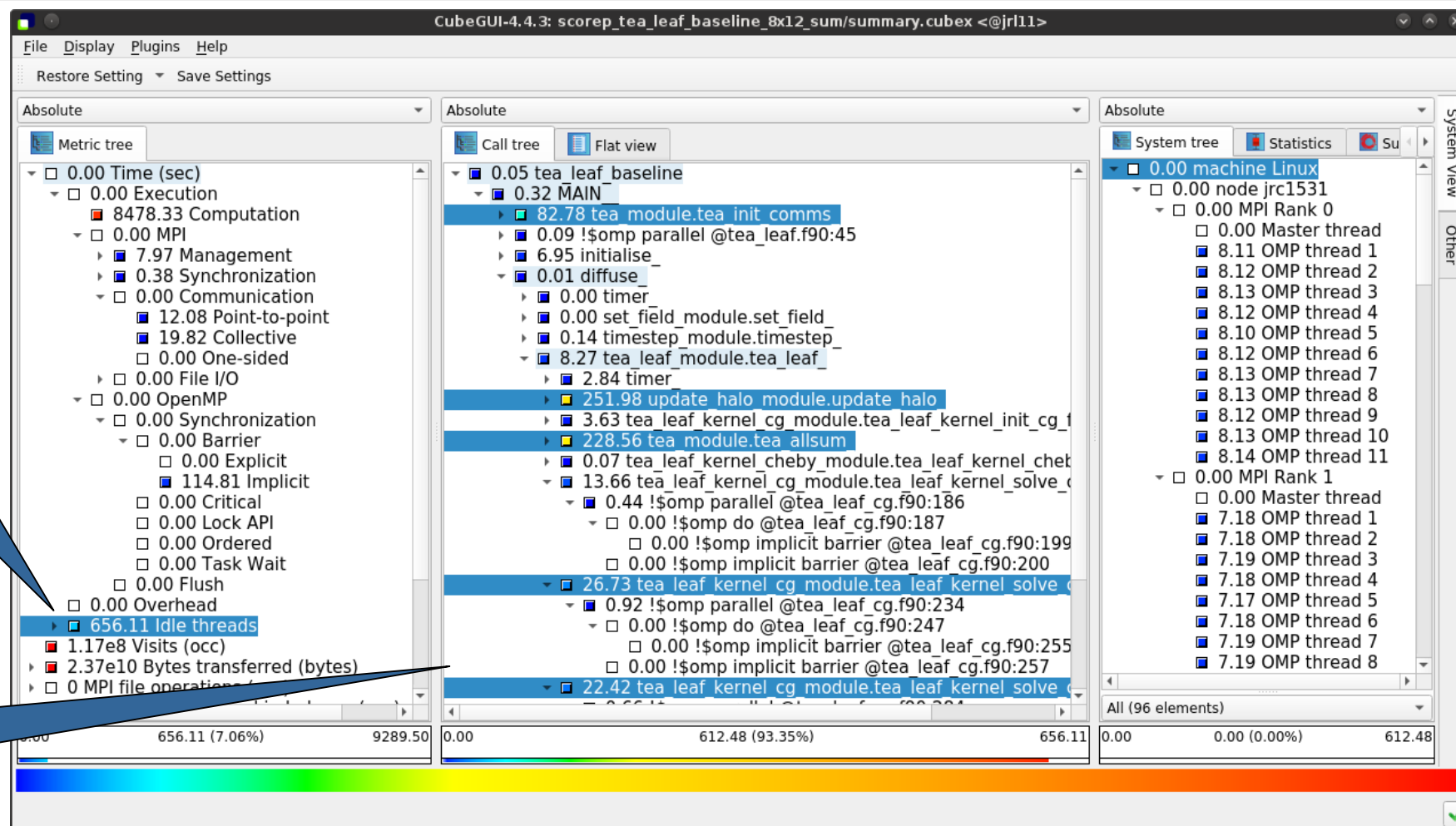
...with a slight imbalance across ranks & threads



TeaLeaf summary report analysis (II)

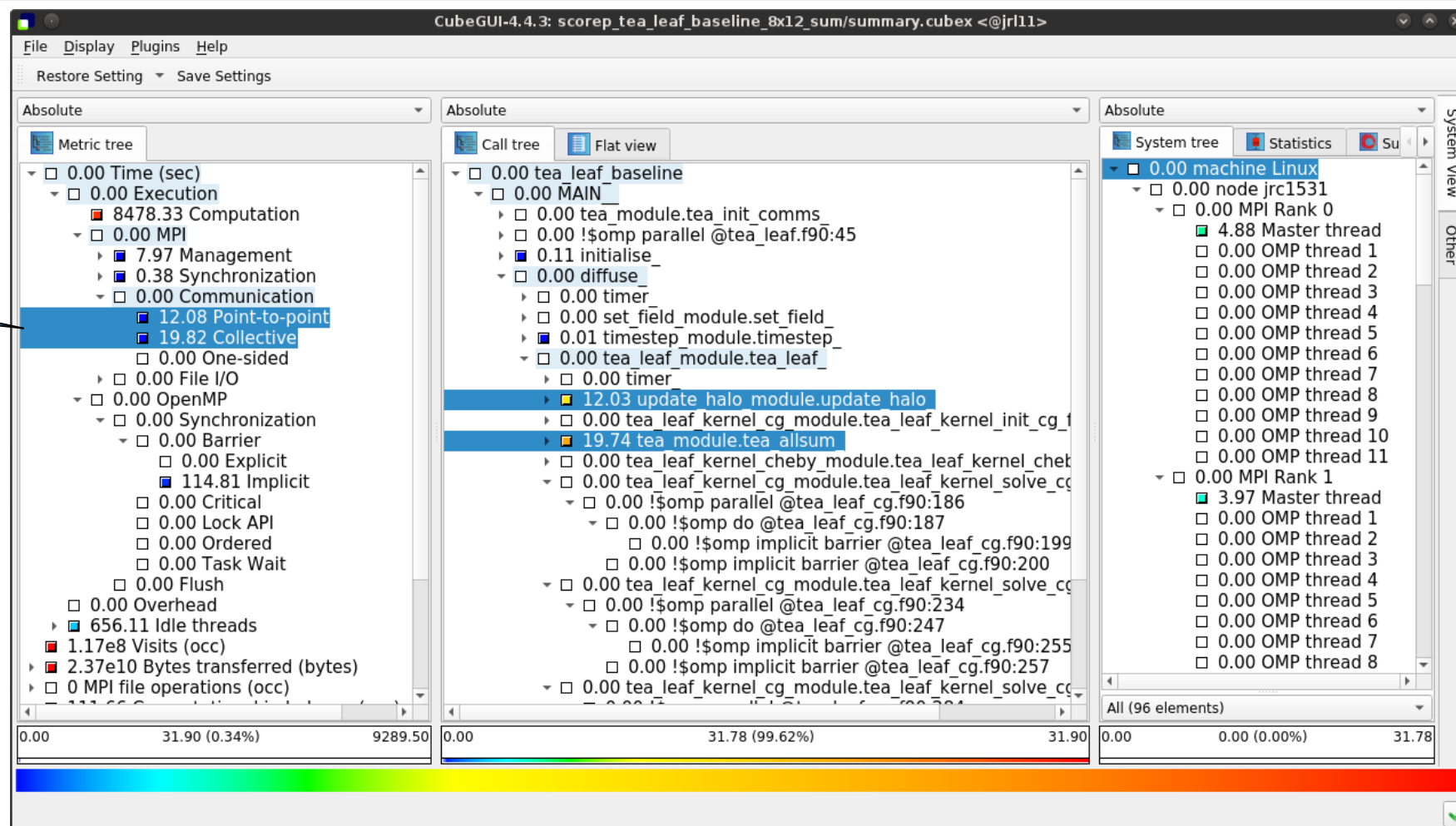
7% of the total CPU execution time is due due to "idle threads" ...

... when not within OpenMP-parallelized code regions



TeaLeaf summary report analysis (III)

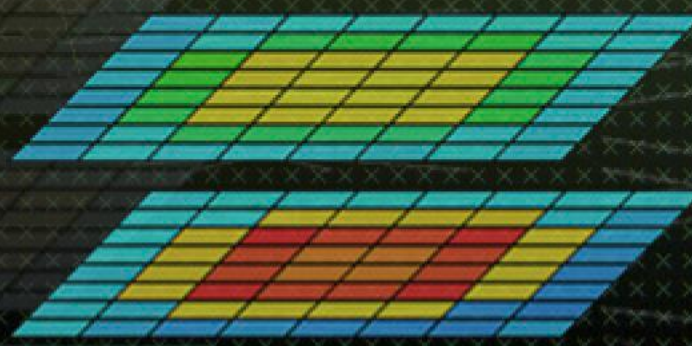
MPI communication time is negligible (0.34%); but communication is only on the master threads (MPI_THREAD_FUNNELED)



Cube: Further information

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

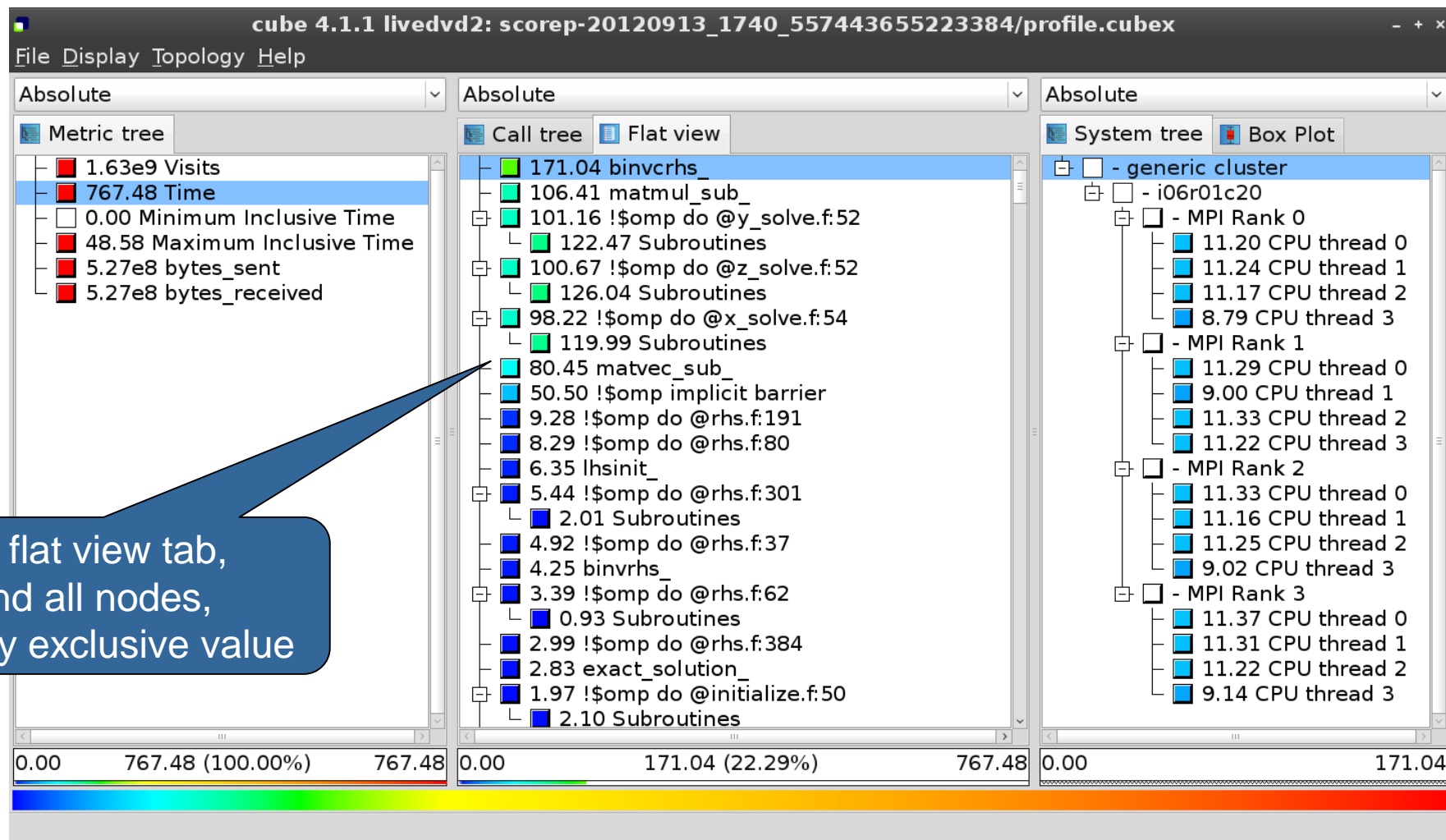




Reference material



Flat profile view



Select flat view tab,
expand all nodes,
and sort by exclusive value

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)`

Derived metrics in Cube GUI



Collection of derived metrics

Parameters of the derived metric

CubePL expression

The screenshot shows the Cube GUI interface with a dialog box for creating a new derived metric. The dialog is titled "Create new metric as a child of metric". It has several input fields and a description area. The "Select metric from collection" dropdown is set to "Average execution time (kenobi)". The "Derived metric type" is "Postderived metric". The "Display name" is "Average visit time", the "Unique name" is "avg_visit_time", the "Data type" is "DOUBLE", and the "Unit of measurement" is "sec". The "Description" field contains the text: "Calculates average time of region execution per visit. Autor is Michael Knobloch." At the bottom of the dialog, there is a "Calculation" section with a checked checkbox and a text area containing the CubePL expression: `metric::time()/metric::visits(e)`. There are "Create metric" and "Cancel" buttons at the bottom of the dialog. The background shows a performance metrics tree and a system tree.

Example: FLOPS based on PAPI_FP_OPS and time



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

File Display Plugins Help
Restore Setting Save Settings

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 Calculation Init Aggregation "+" Aggregation "-"

`metric::PAPI_FP_OPS()/metric::time()`

Edit metric Cancel

Share this metric with SCALASCA group

Absolute Metric tree

- 1.17e7 Visits (occ)
- 1148.49 Time (sec)
- 0.00 Minimum Inclusive Time (sec)
- 41.57 Maximum Inclusive Time (...)
- 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**

Absolute Call tree Flat view

- 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 @exac...
 - 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

Absolute System tree Barplot Heatmap

 - 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

All (32 elements)

0.00 1.84e9 (100.00%) 1.84e9 0.00 9.65e8 (-0.00%) -12858016489314434.00 0.00... -179769313486231570814527423731704356798070...

Selected "\$!omp do @exact_rhs.f:46"

CUBE algebra utilities



- Extracting solver sub-tree from analysis report

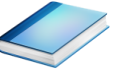
```
% cube_cut -r '<<ITERATION>>' scorep_bt-mz_C_32x4_sum/profile.cubex  
Writing cut.cubex... done.
```

- Calculating difference of two reports

```
% cube_diff scorep_bt-mz_C_32x4_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

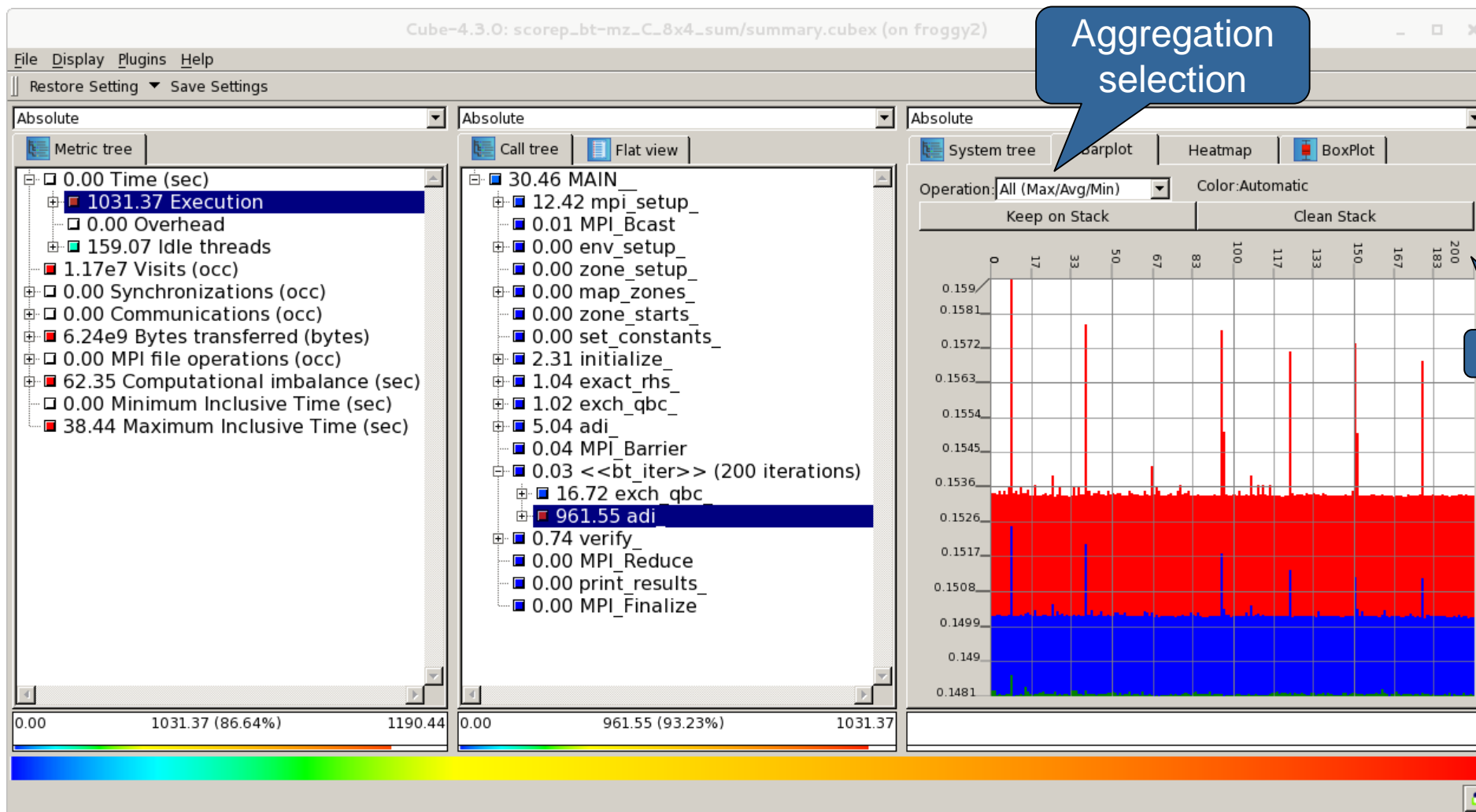


- Show time dependent behavior by “unrolling” iterations
- Preparations:
 - Mark loop body by using Score-P instrumentation API in your source code

```
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



Iteration profiling: Heatmap

