

Analysis report examination with Cube

Marc-André Hermanns
Jülich Supercomputing Centre

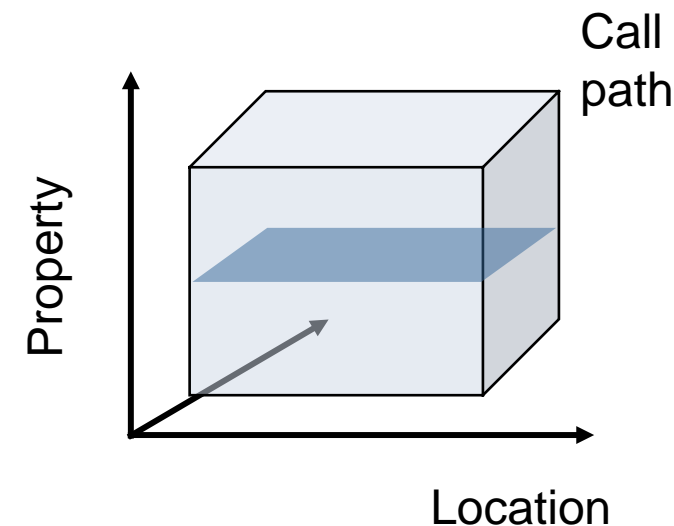


Cube

- 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: Cube 4.3.4 (April 2016)

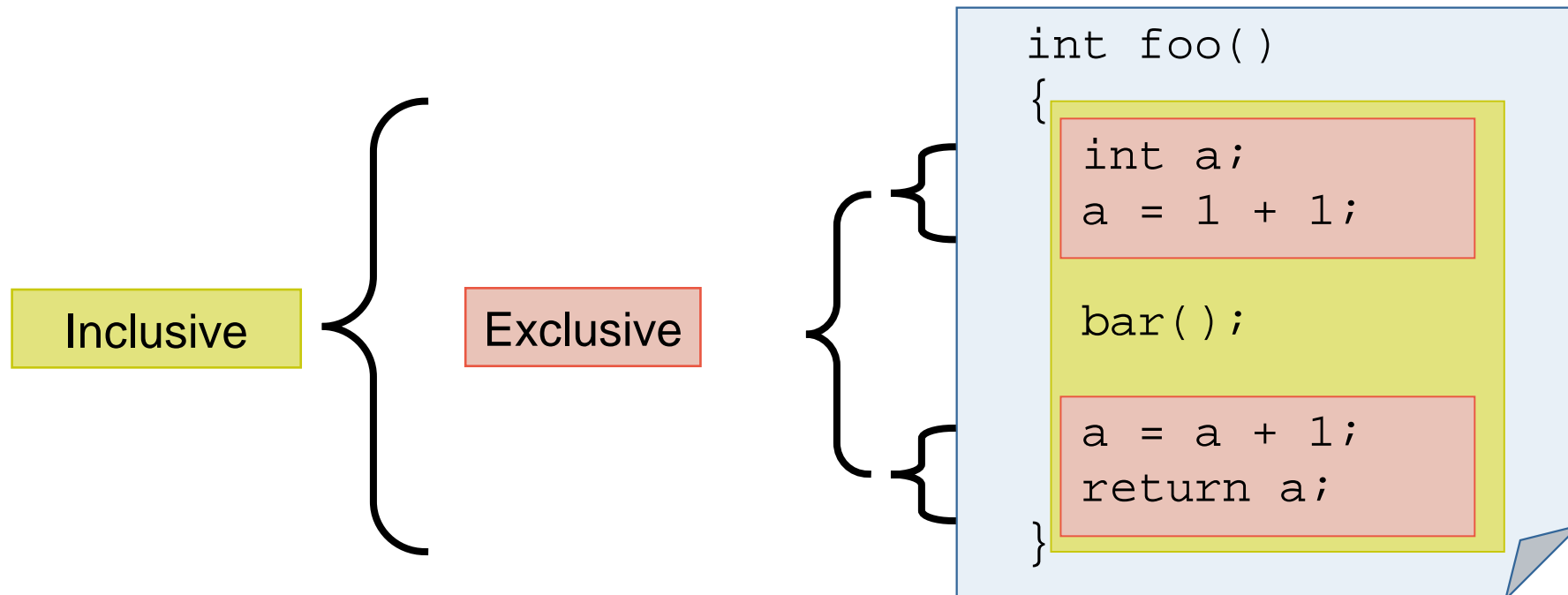
Analysis presentation and exploration

- 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

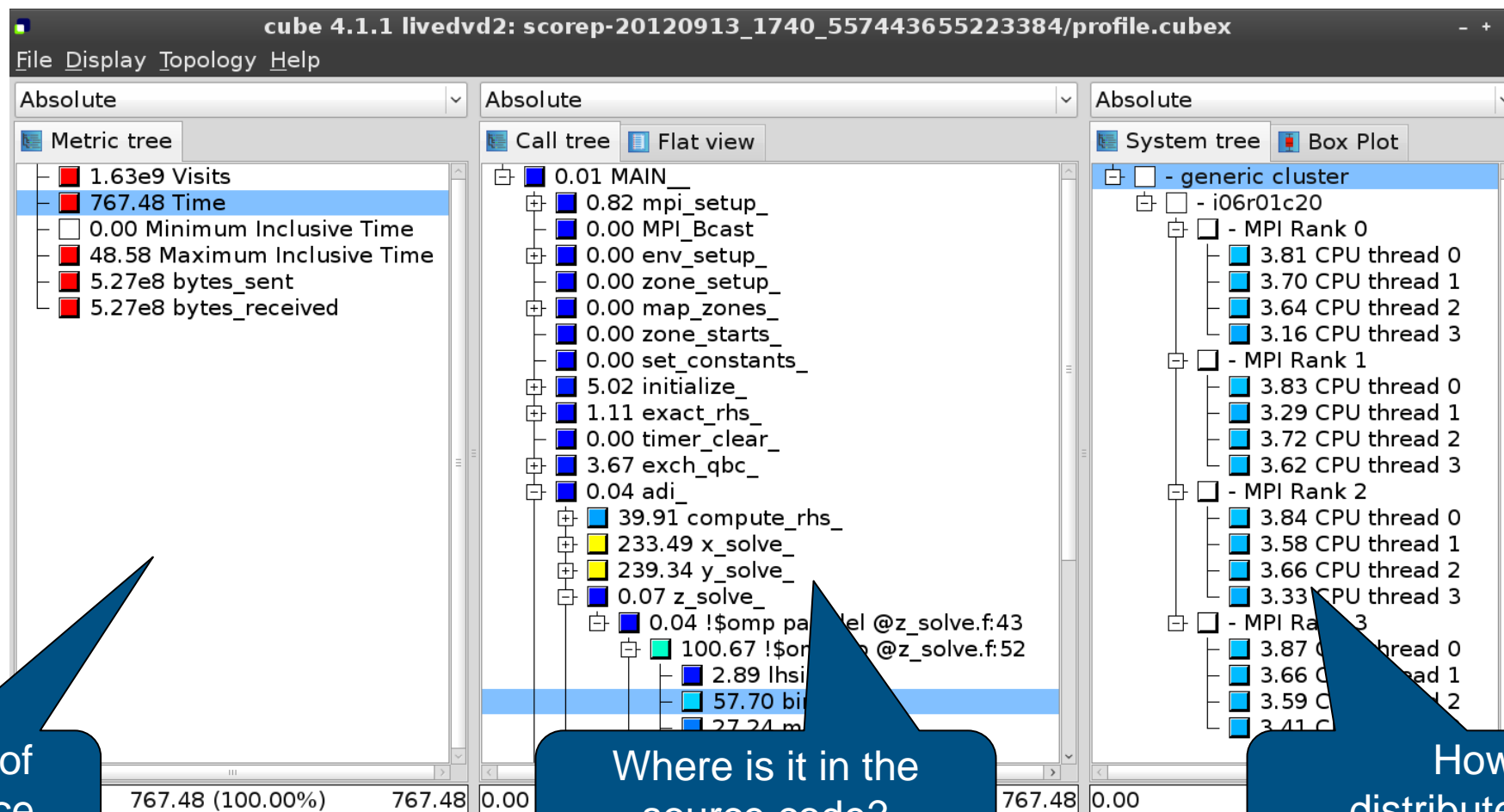


Inclusive vs. exclusive values

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



Analysis presentation

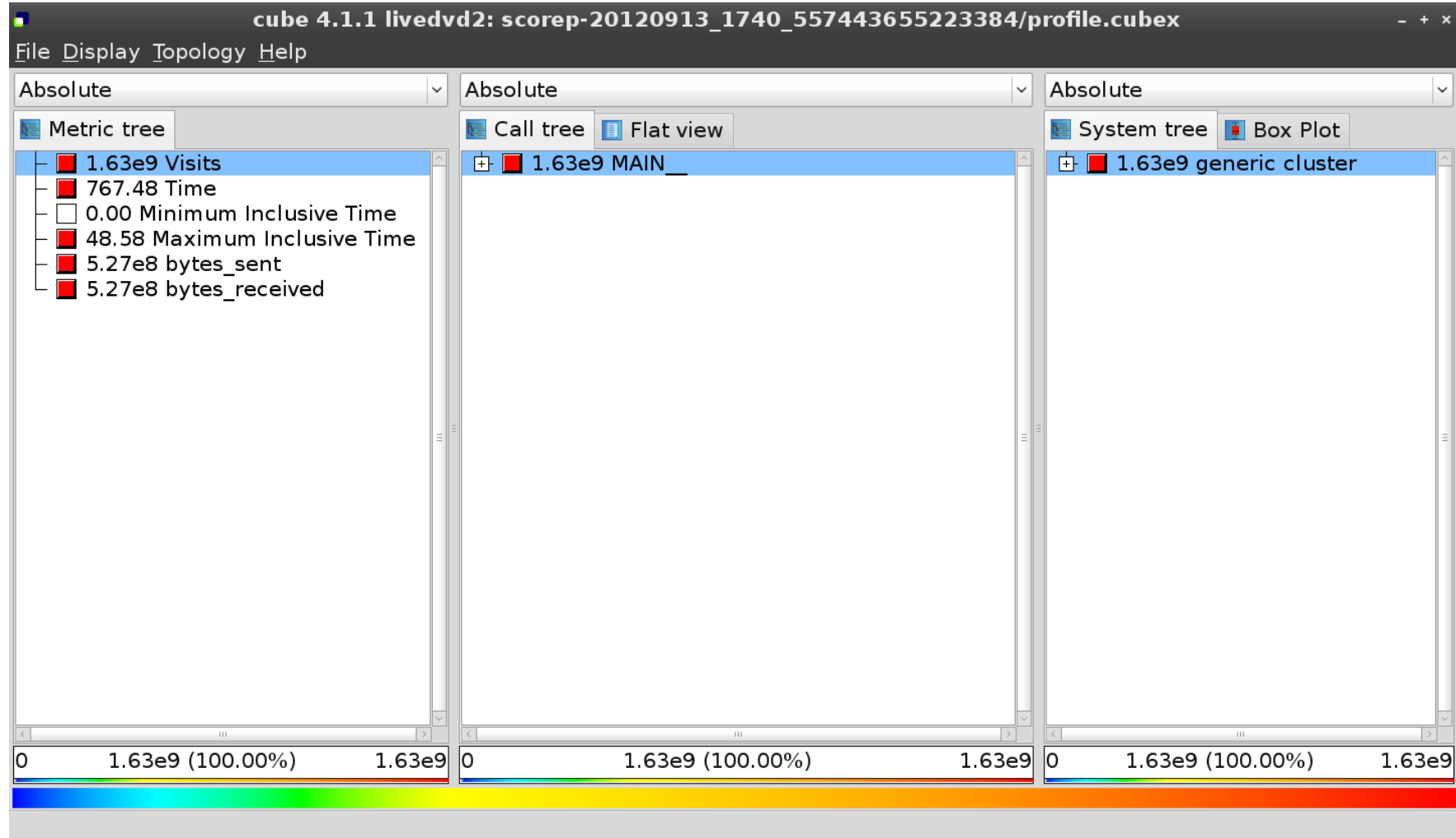


What kind of performance metric?

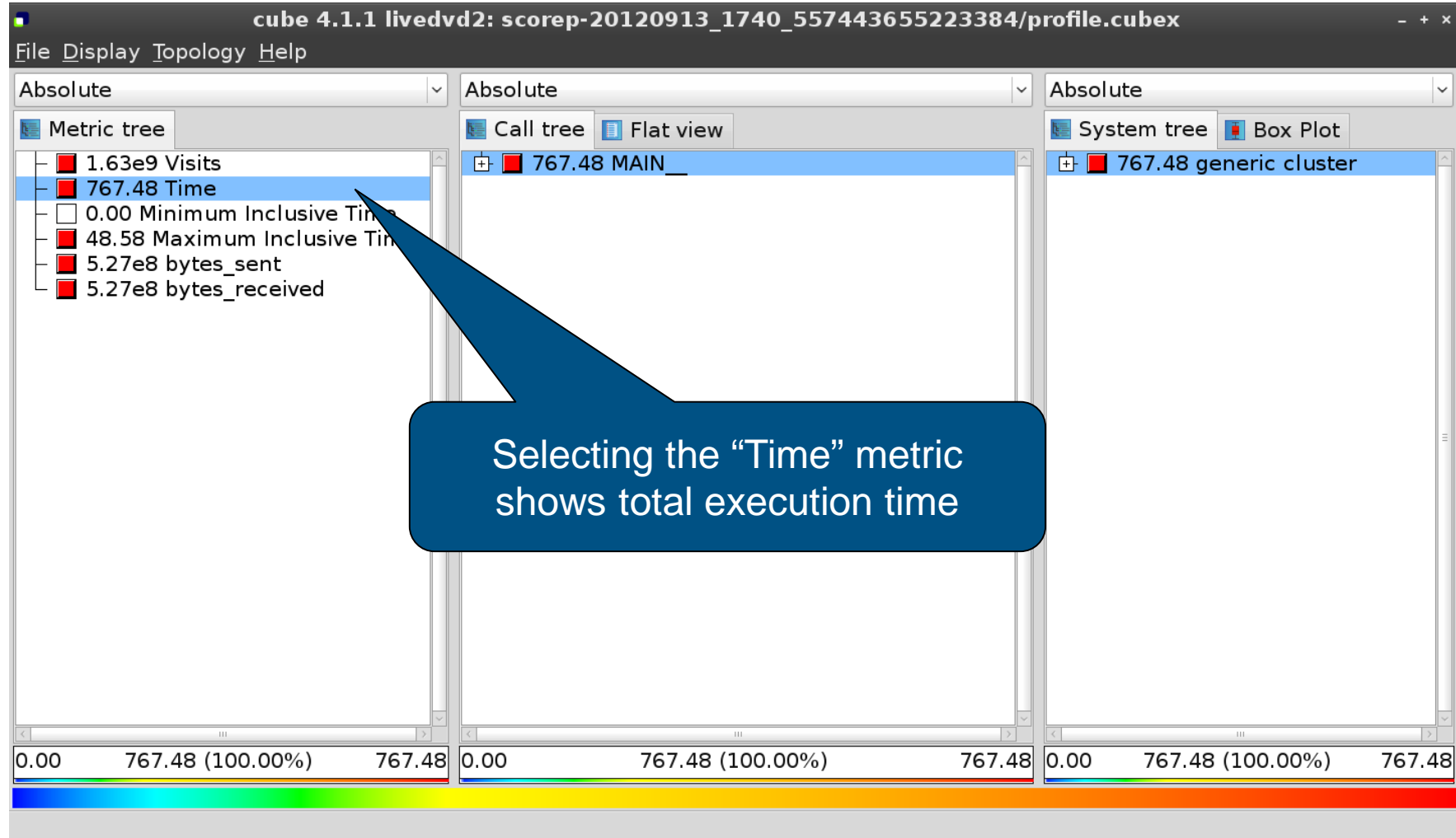
Where is it in the source code?
In what context?

How is it distributed across the processes/threads?

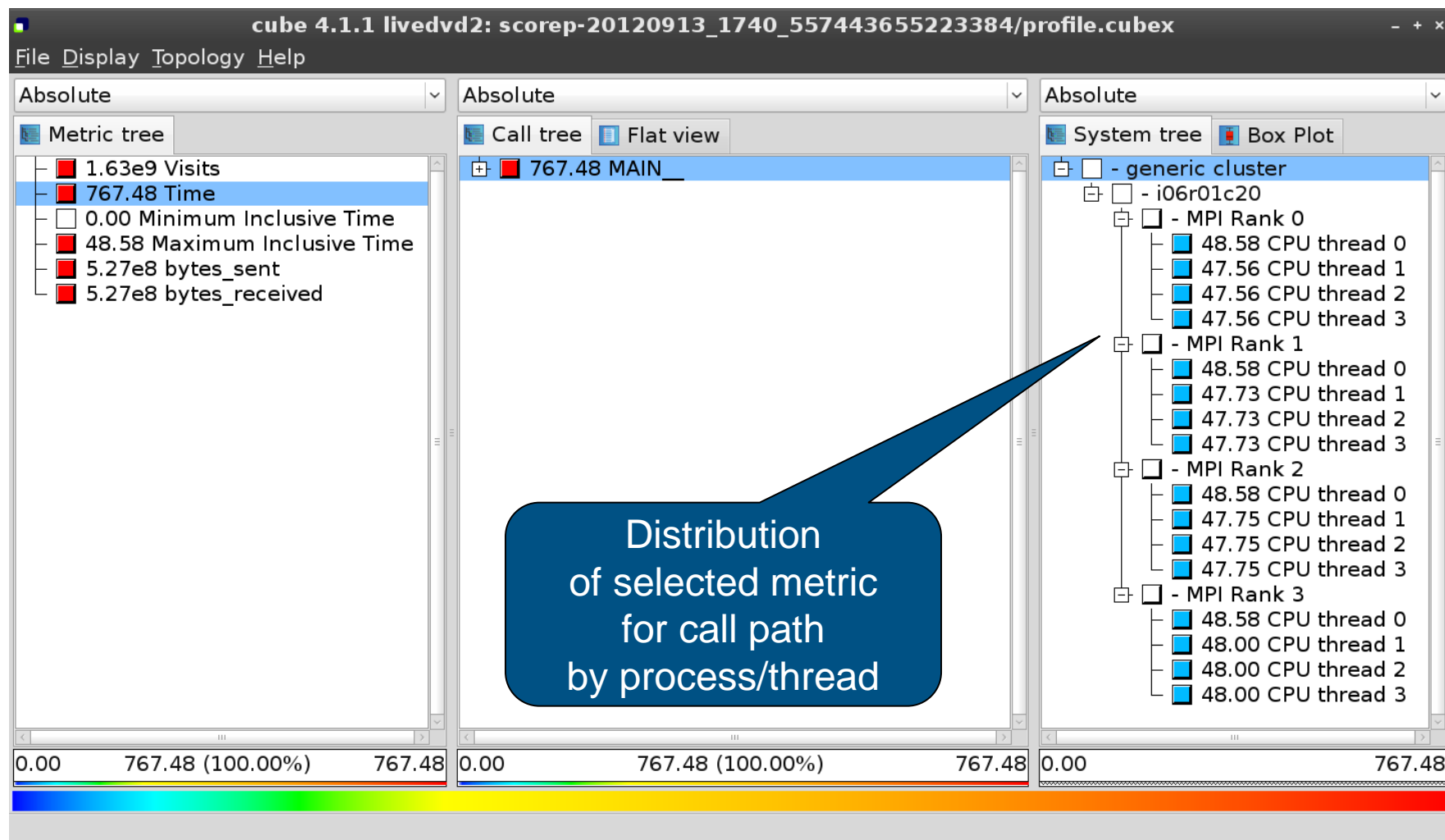
Score-P analysis report exploration (opening view)



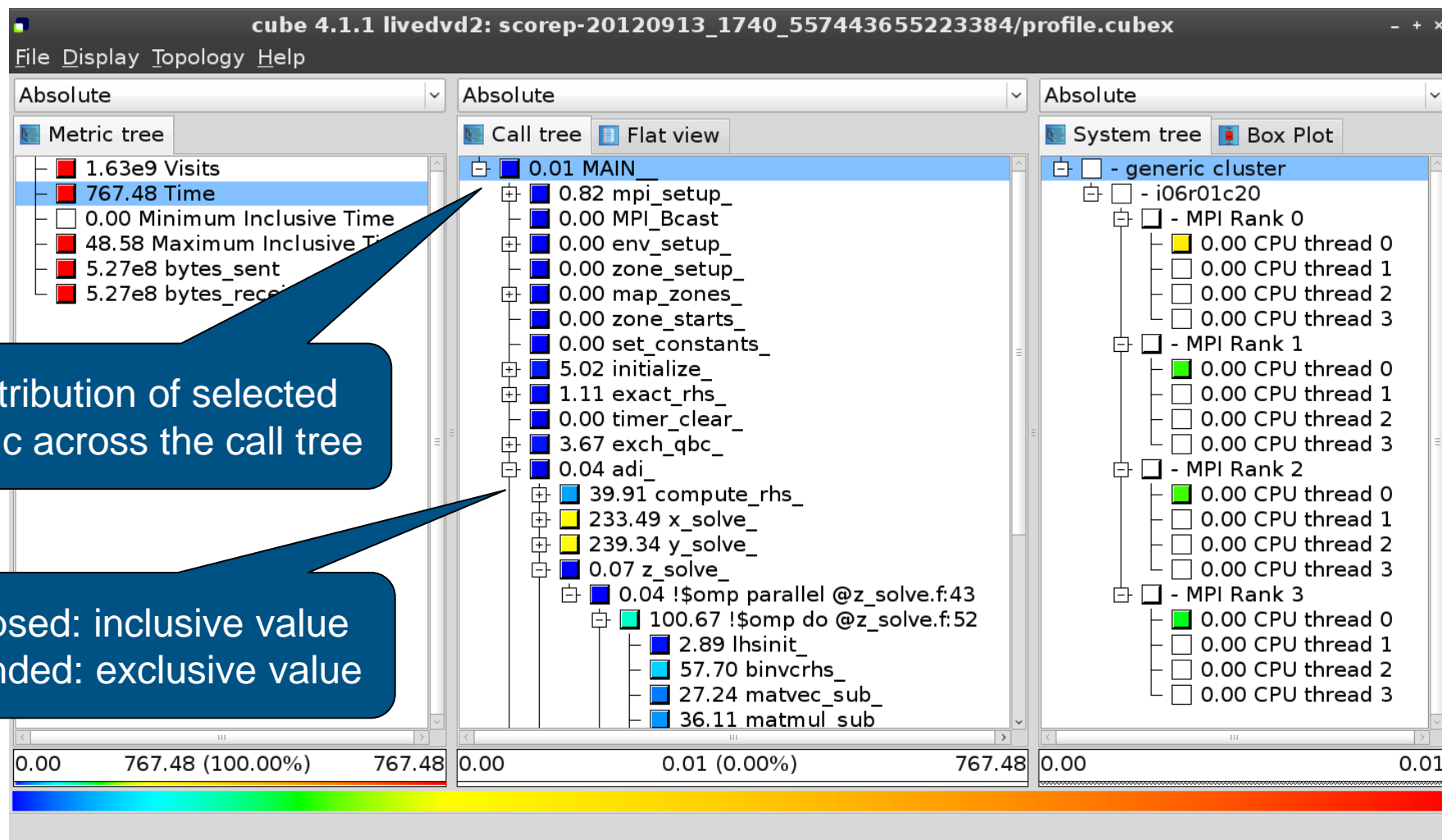
Metric selection



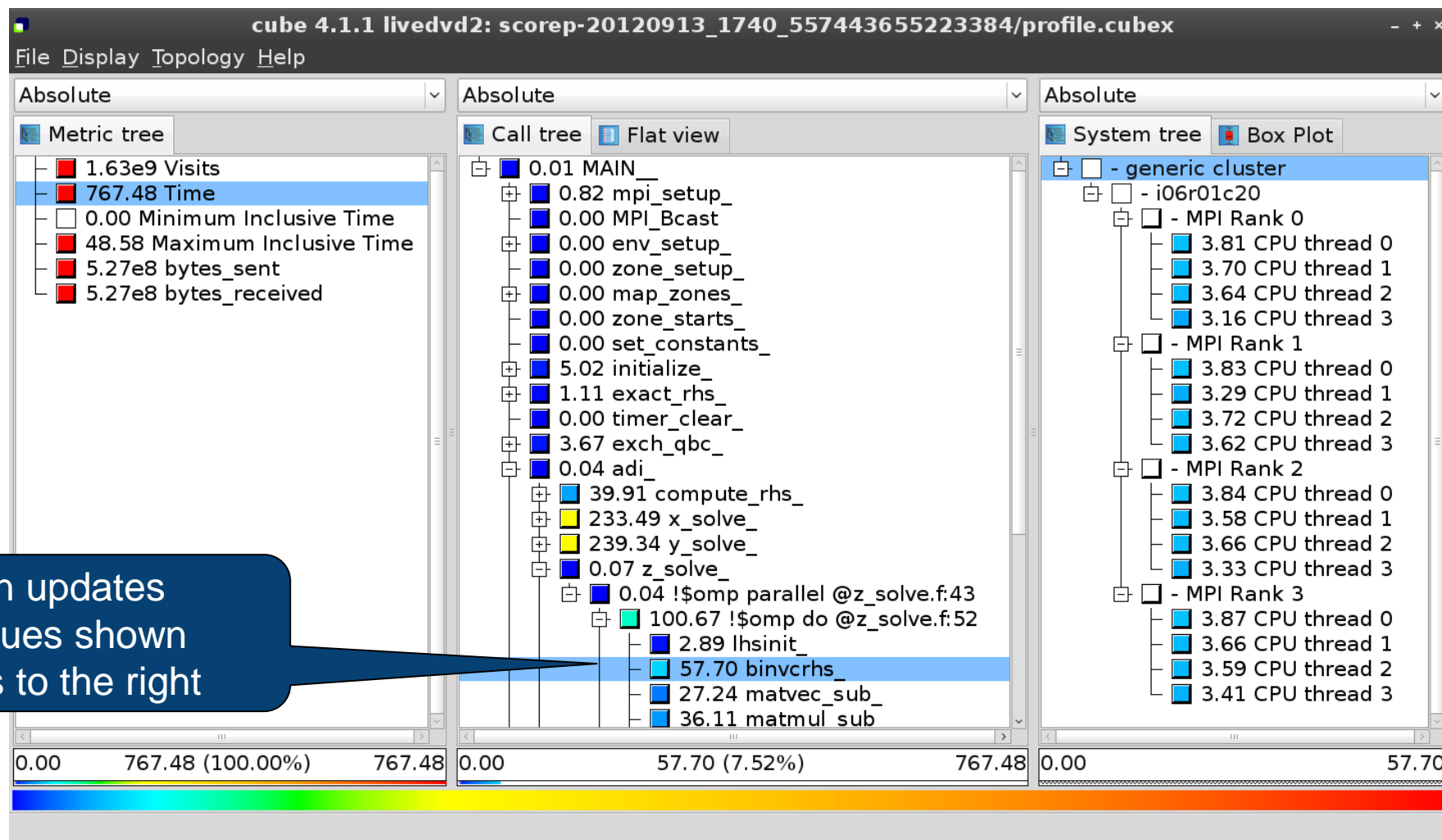
Expanding the system tree

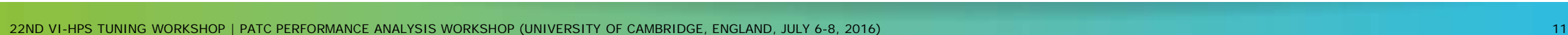


Expanding the call tree

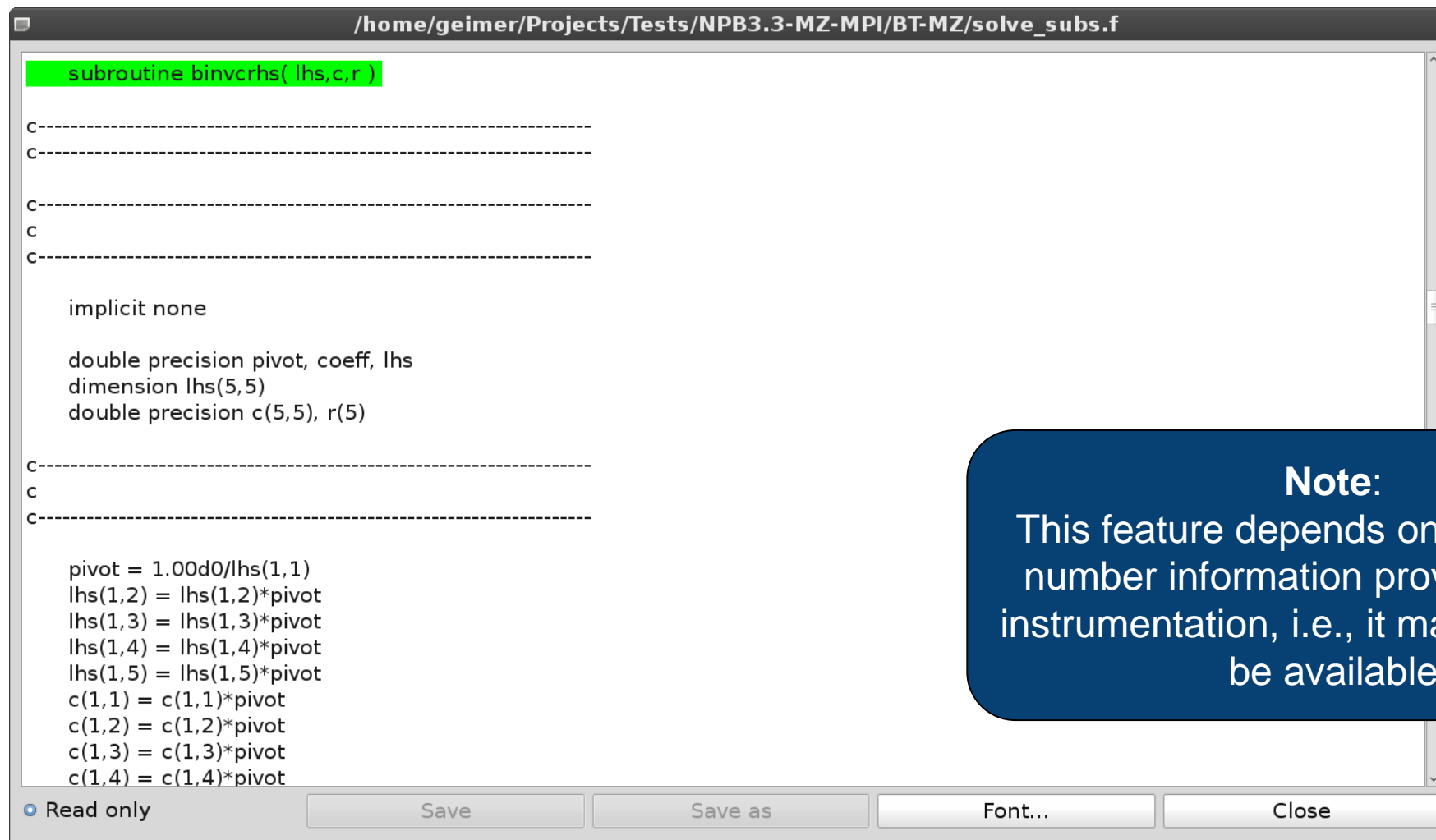


Selecting a call path





Source-code view



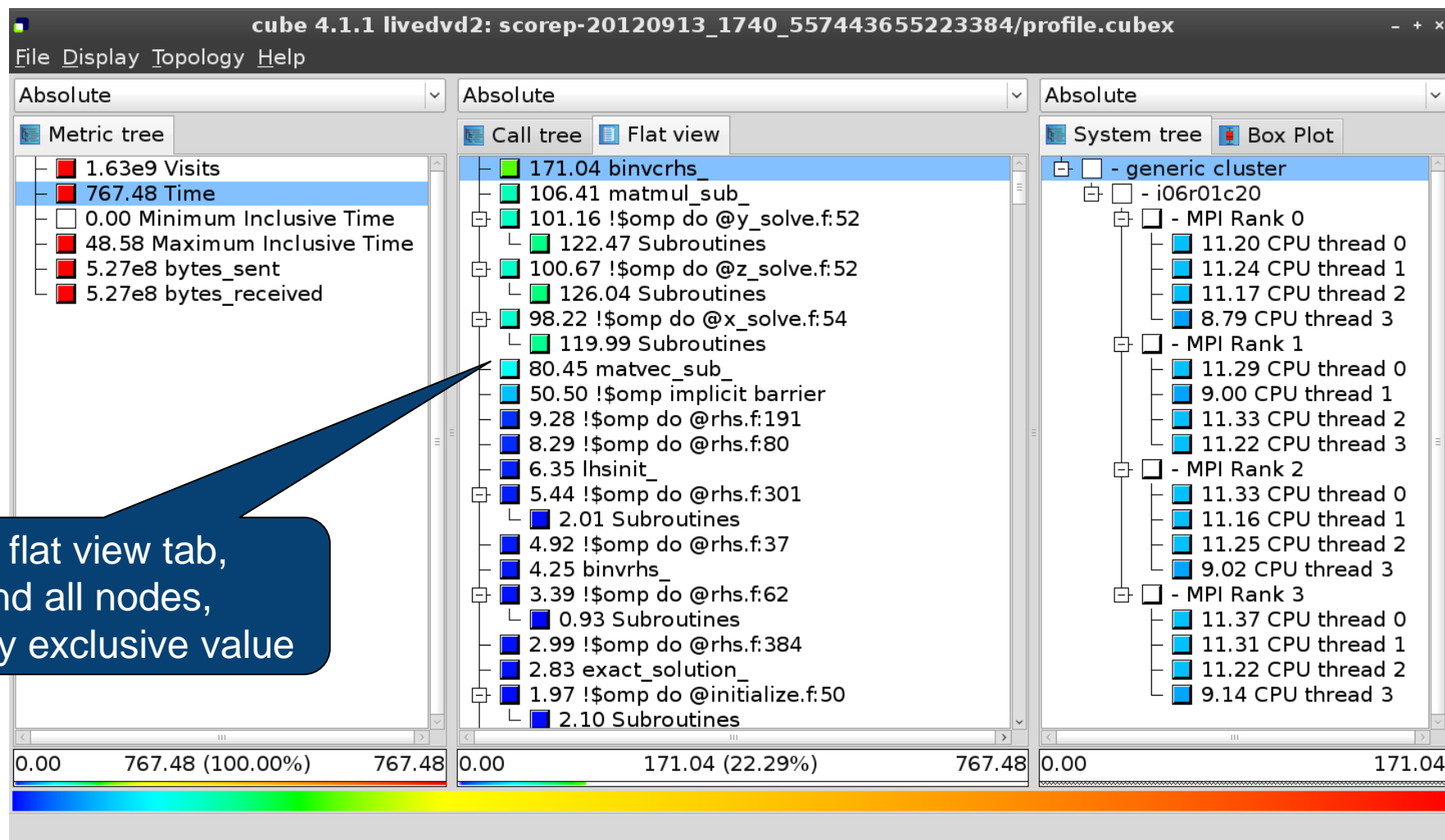
```
subroutine binvrhs( lhs,c,r )  
  
C-----  
C-----  
  
C-----  
C  
C-----  
  
implicit none  
  
double precision pivot, coeff, lhs  
dimension lhs(5,5)  
double precision c(5,5), r(5)  
  
C-----  
C  
C-----  
  
pivot = 1.00d0/lhs(1,1)  
lhs(1,2) = lhs(1,2)*pivot  
lhs(1,3) = lhs(1,3)*pivot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

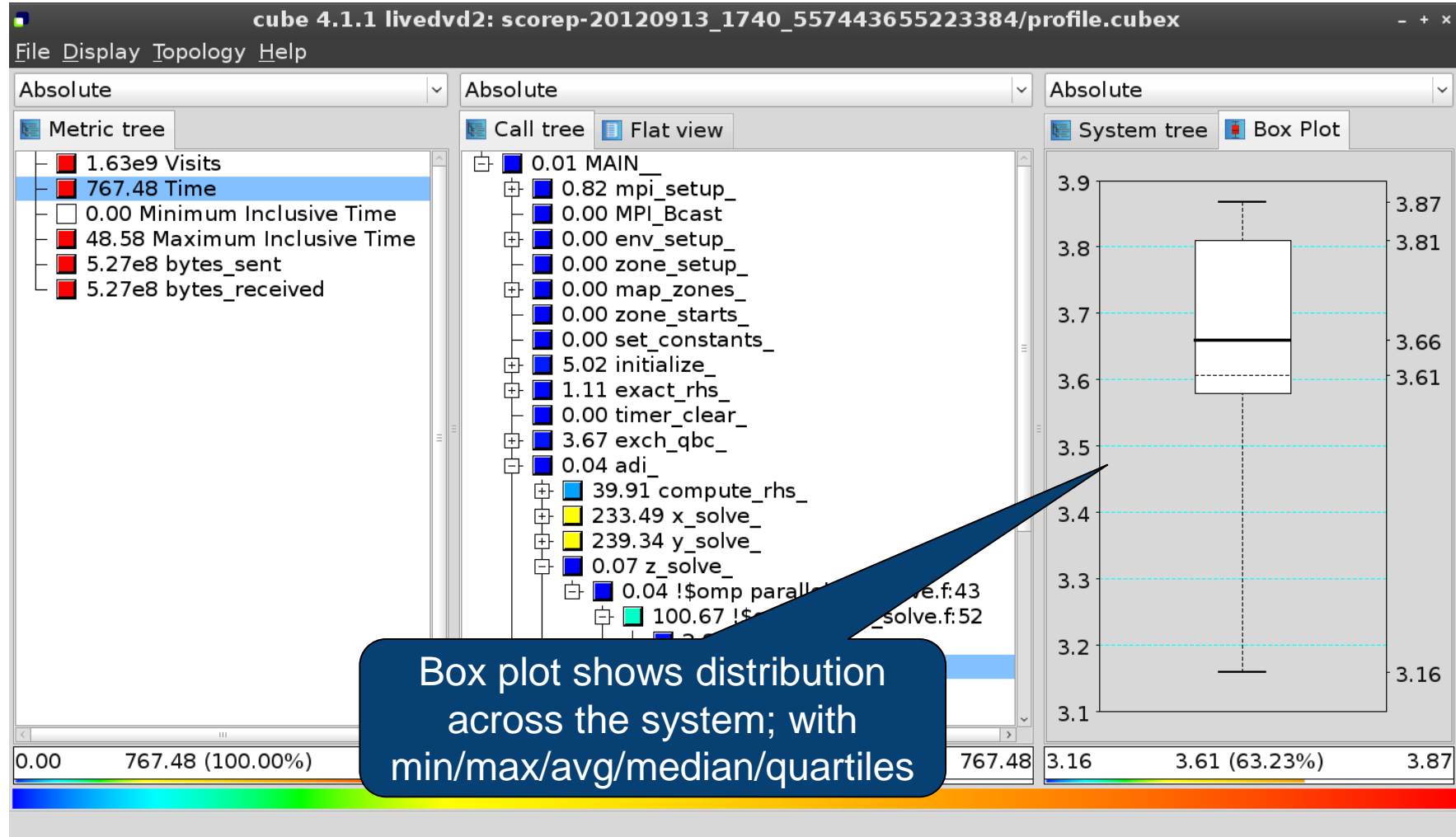
Note:

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

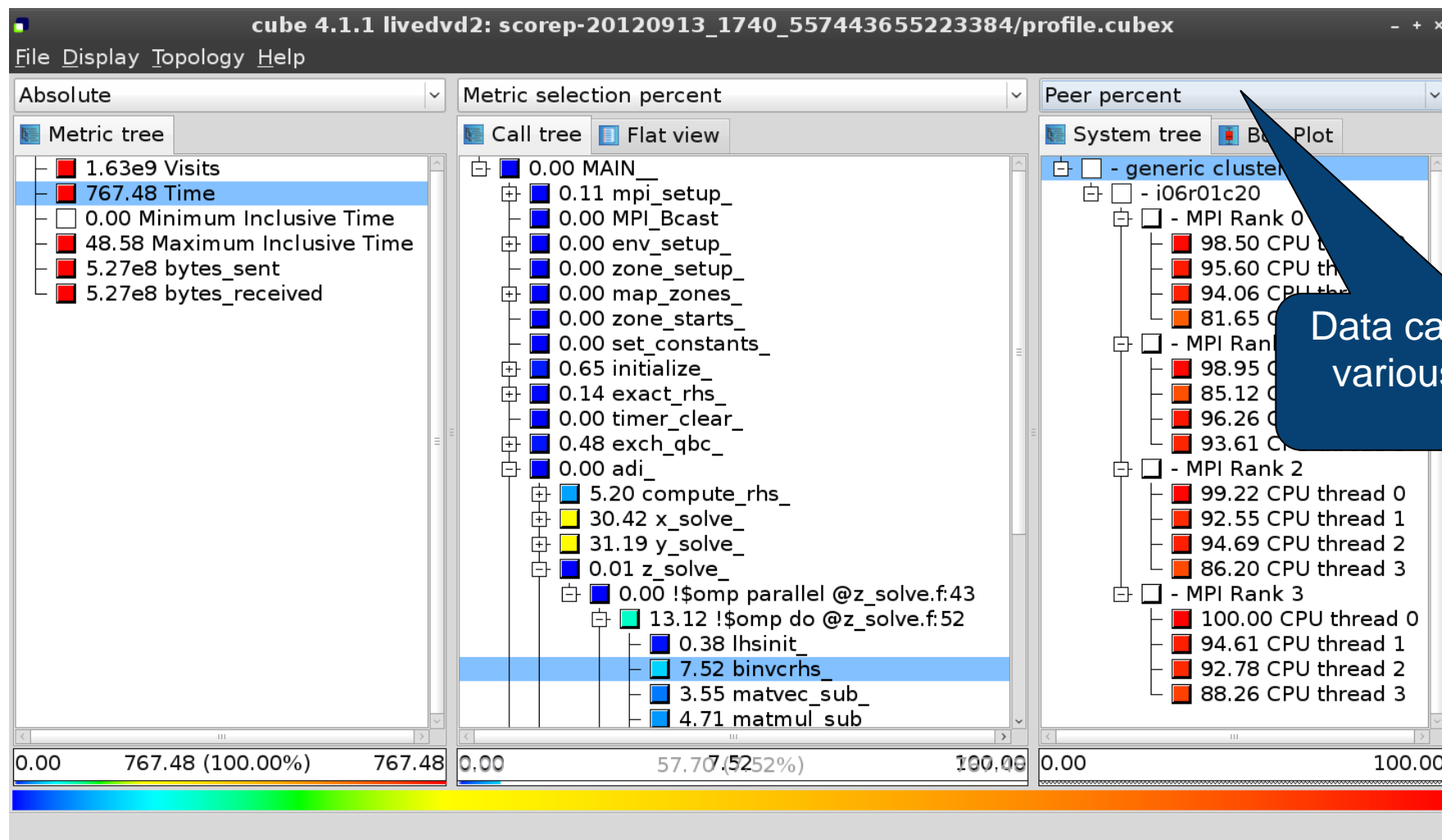
Flat profile view



Box plot view



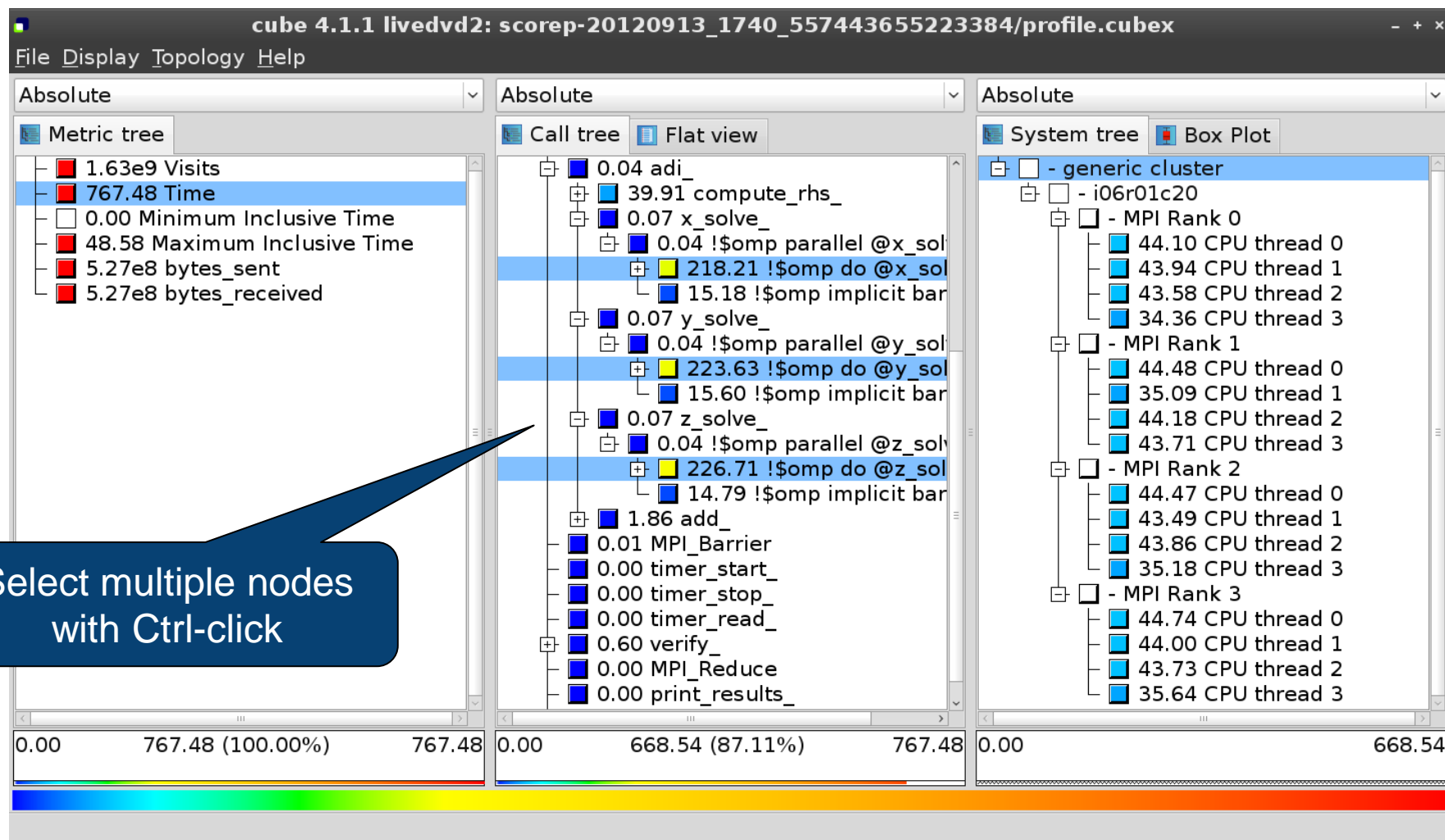
Alternative display modes



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

Multiple selection



Context-sensitive help

The screenshot displays the 'cube 4.1.1' application window with the title bar 'cube 4.1.1 livedvd2: scorep-20120913_1740_557443655223384/profile.cubex'. The 'Help' menu is open, showing options: 'Getting started', 'Mouse and keyboard control', 'What's This?' (highlighted with a mouse cursor and 'Shift+F1'), and 'About'. The 'What's This?' option is context-sensitive, displaying a list of selected metrics and regions. The 'Metric tree' on the left shows a list of metrics, with '767.48 Time' selected. The 'System tree' on the right shows a hierarchical view of the system, with 'generic cluster' selected. The bottom status bar shows three progress bars: '0.00 767.48 (100.00%) 767.48', '0.00 668.54 (87.11%) 767.48', and '0.00 668.54'. A blue callout box points to the 'What's This?' menu item with the text 'Context-sensitive help available for all GUI items'. A status bar at the bottom reads 'Change into help mode for display components'.

cube 4.1.1 livedvd2: scorep-20120913_1740_557443655223384/profile.cubex

File Display Topology Help

Absolute

Metric tree

- 1.63e9 Visits
- 767.48 Time
- 0.00 Minimum I
- 48.58 Maximum
- 5.27e8 byt
- 5.27e8
- _received

Getting started

Mouse and keyboard control

What's This? Shift+F1

About

Selected metrics description

Selected regions description

compute_rhs_
solve
4 !\$omp parallel @x_sol
218.21 !\$omp do @x_sol
15.18 !\$omp implicit bar
0.07 y_solve_
0.04 !\$omp parallel @y_sol
223.63 !\$omp do @y_sol
15.60 !\$omp implicit bar
0.07 z_solve_
0.04 !\$omp parallel @z_sol
226.71 !\$omp do @z_sol
14.79 !\$omp implicit bar
1.86 add_
0.01 MPI_Barrier
0.00 timer_start_
0.00 timer_stop_
0.00 timer_read_
0.60 verify_
0.00 MPI_Reduce
0.00 print_results_

System tree

Box Plot

- generic cluster
 - i06r01c20
 - MPI Rank 0
 - 44.10 CPU thread 0
 - 43.94 CPU thread 1
 - 43.58 CPU thread 2
 - 34.36 CPU thread 3
 - MPI Rank 1
 - 44.48 CPU thread 0
 - 35.09 CPU thread 1
 - 44.18 CPU thread 2
 - 43.71 CPU thread 3
 - MPI Rank 2
 - 44.47 CPU thread 0
 - 43.49 CPU thread 1
 - 43.86 CPU thread 2
 - 35.18 CPU thread 3
 - MPI Rank 3
 - 44.74 CPU thread 0
 - 44.00 CPU thread 1
 - 43.73 CPU thread 2
 - 35.64 CPU thread 3

0.00 767.48 (100.00%) 767.48

0.00 668.54 (87.11%) 767.48

0.00 668.54

Change into help mode for display components

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

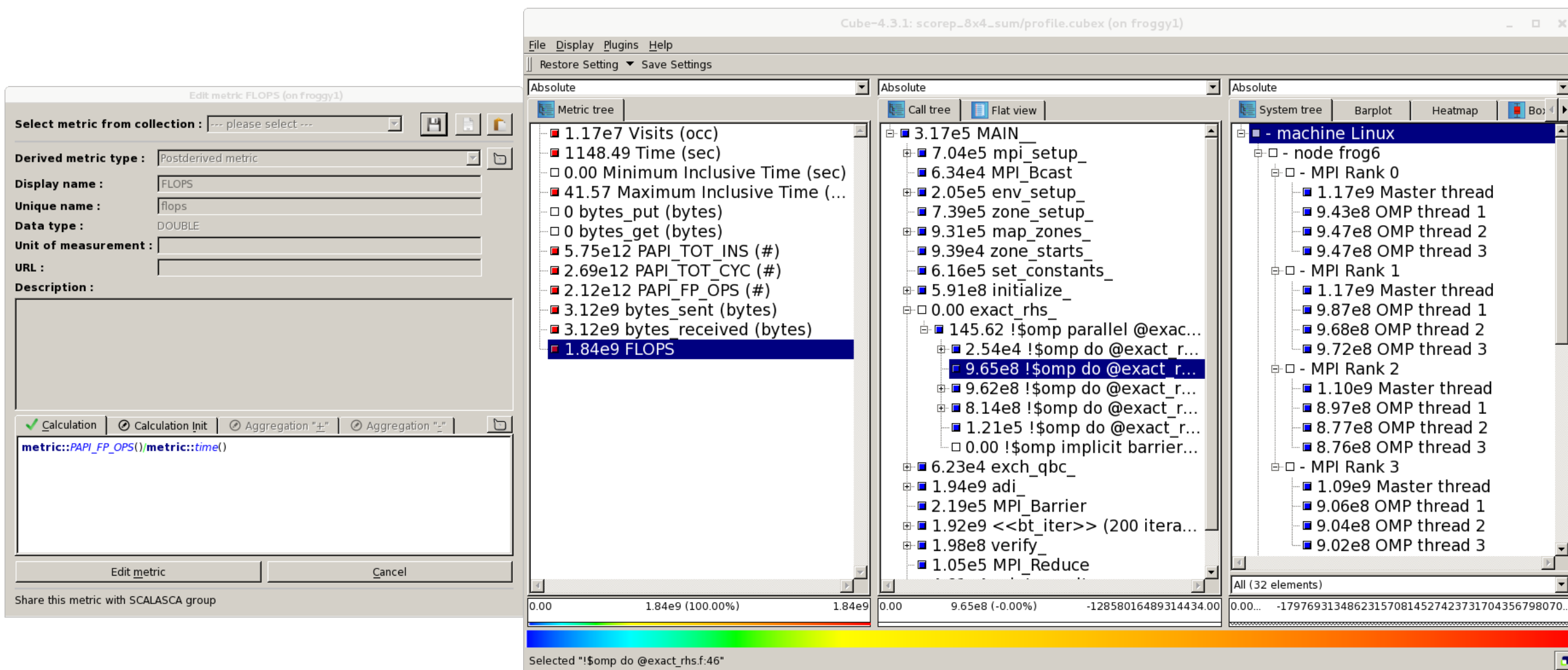
Parameters of the derived metric

CubePL expression

1.01e6 (100.00%) 1.01e6 0.00

2512.10

Example: FLOPS based on PAPI_FP_OPS and time



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:

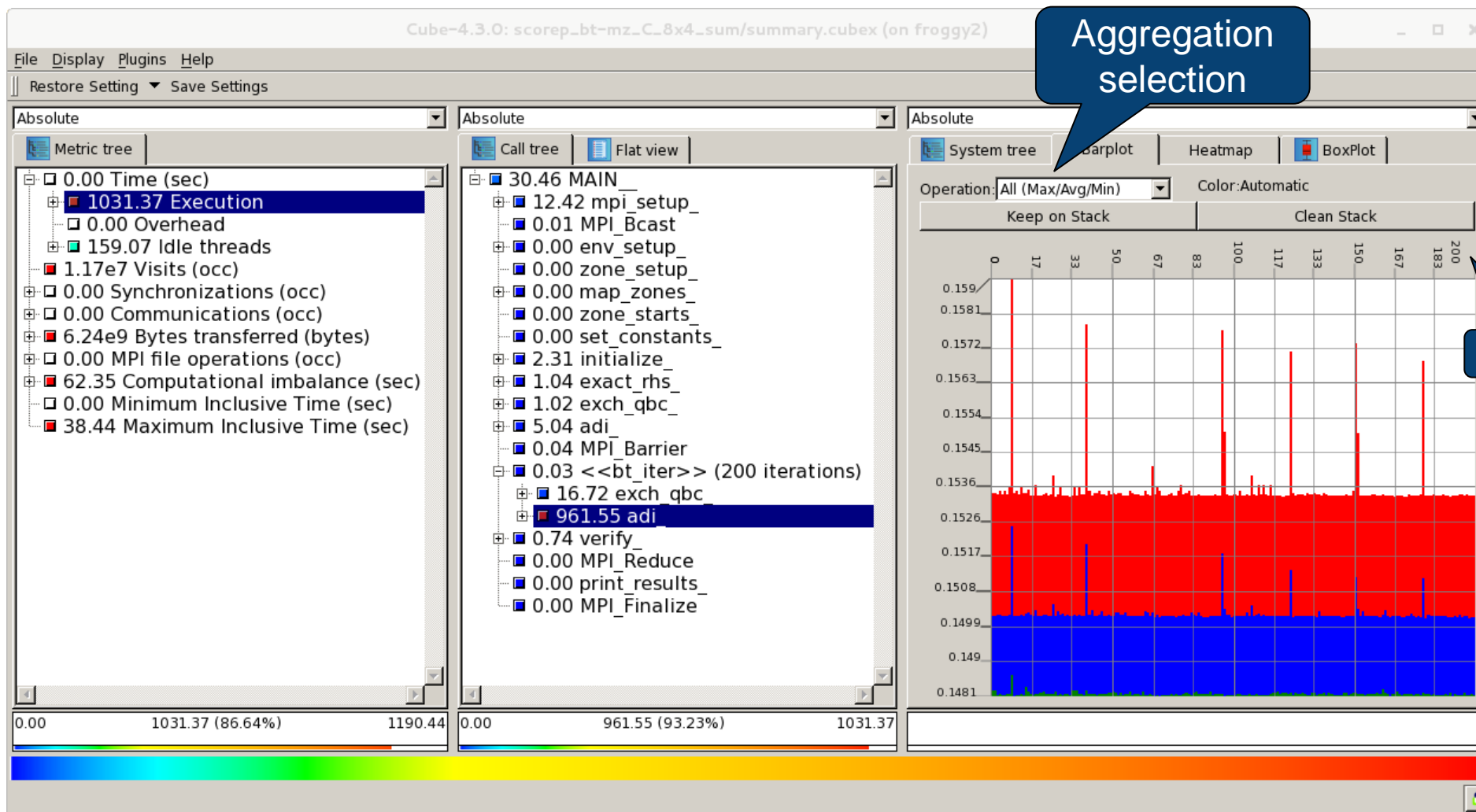
- Mark loop body by using Score-P instrumentation API in your source code

```
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 )
```

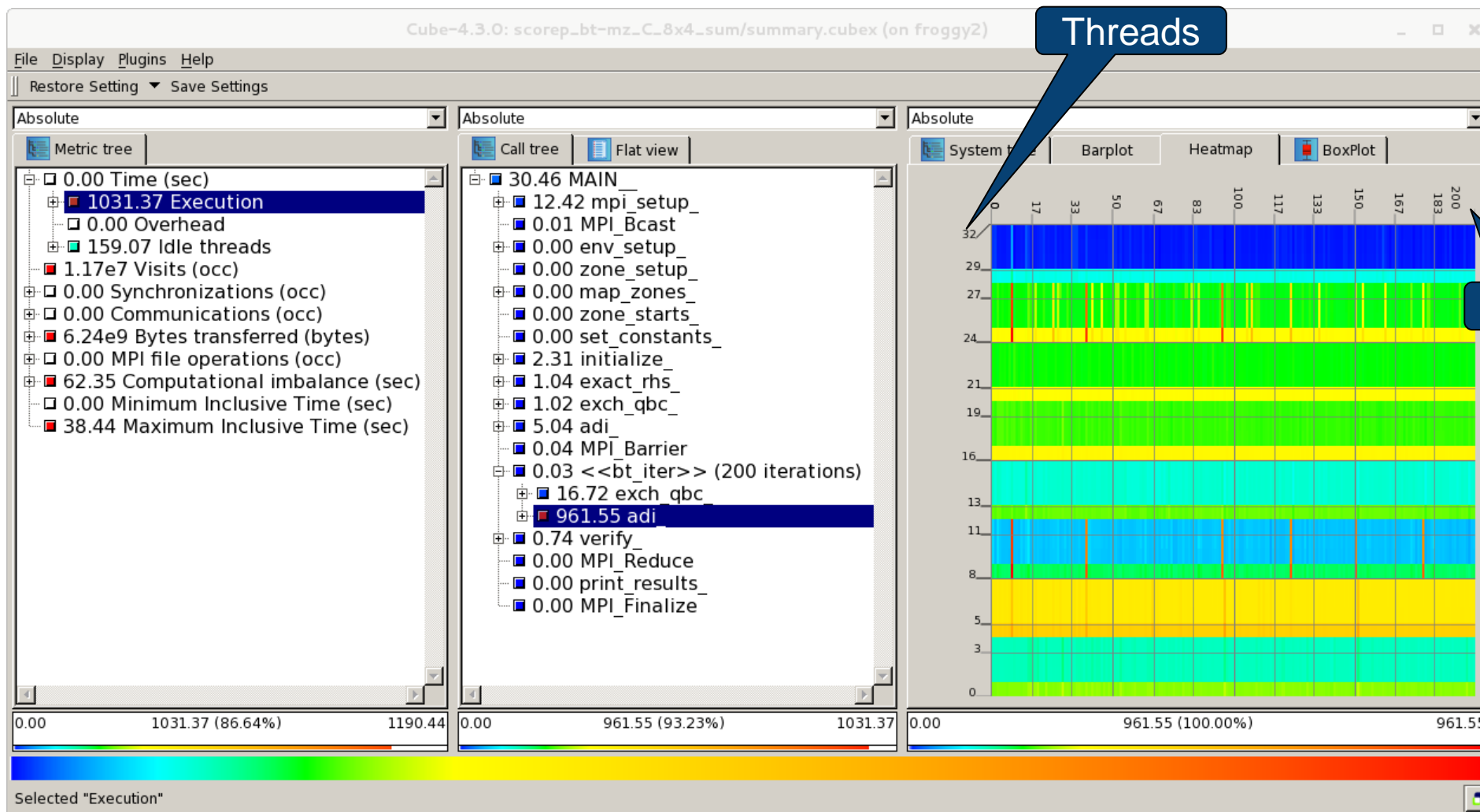
- Result in the Cube profile:

- 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



Iteration profiling: Heatmap



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](mailto:scalasca@fz-juelich.de)

