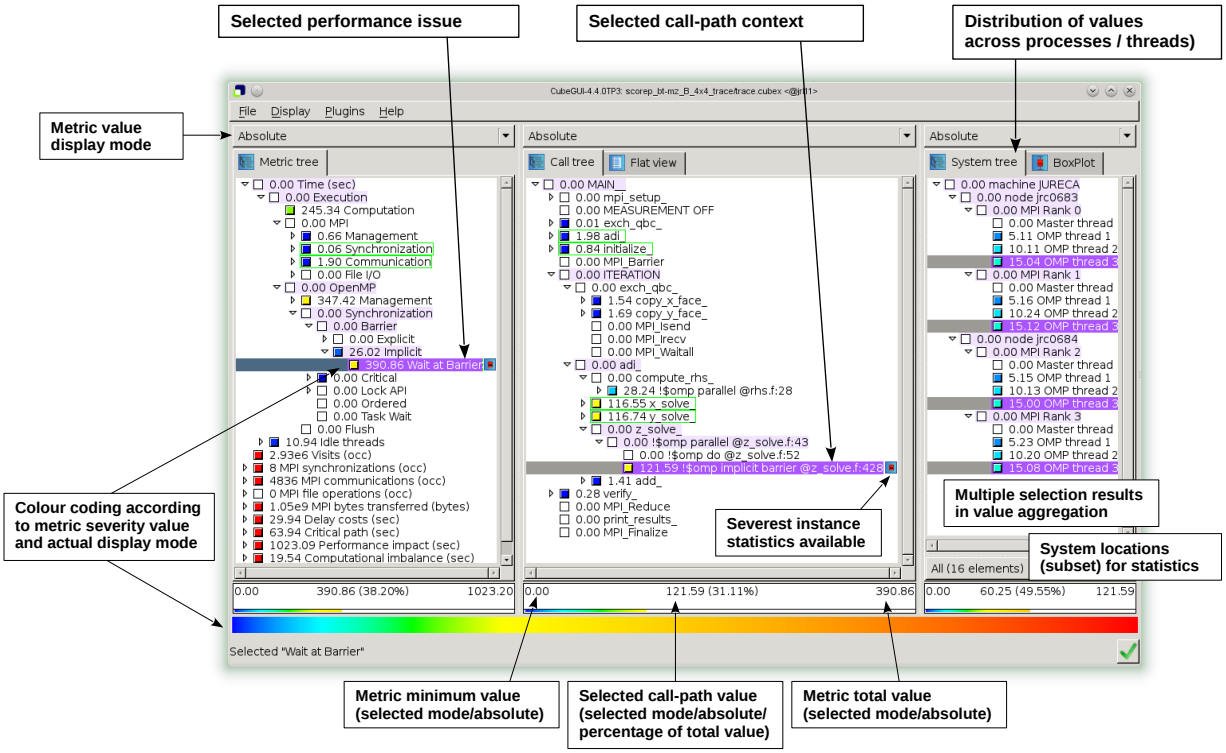
CUBE v4.4 Display Quick Reference

General

- CUBE is an open-source toolset for creating, processing and examining parallel program execution performance analysis reports. Originally developed as part of the SCALASCA toolset, and now used by the community-developed measurement system SCORE-P. Used with both runtime summarization profiles and augmented automated trace analyses, however, designed to be flexible to handle values in three orthogonal dimensions.
- Provides a variety of command-line utilities for combining, differencing and other report processing, e.g., extraction of interesting call-trees, as well as a highly customisable GUI for interactive analysis report exploration.

Generic presentation

- Three-dimensional severity values are displayed by default using three coupled tree browsers showing
  - Metrics (i.e., performance properties/problems)
  - Context call-tree or flat region list profile
  - System locations (compute nodes, processes & threads)
- Values are aggregated from rightmost to leftmost tree pane, with presentation in centre and right panes based on selections from the left. Metric values in the left pane are aggregates of every callpath or region in the central pane from every process or thread in the right pane. The metric selected in the left panel has its value shown for every callpath or region in the central pane, and the values shown for every process or thread are those for the callpath/region selected there.
- Each tree-browser panel can be re-arranged, resized (or collapsed to be hidden), or non-tree presentations may also be offered (such as the boxplot statistics for system locations, or topology-based representations).
- Analyses are presented in trees, where collapsed nodes represent *inclusive* values (consisting of the value of the node itself and all of its child nodes), which can be selectively expanded to reveal *exclusive* values (i.e., the node 'self' value) and child nodes. Selective expansion of nodes with significant values, guided by the colour scale, can be used to refine and hone in on performance problems.

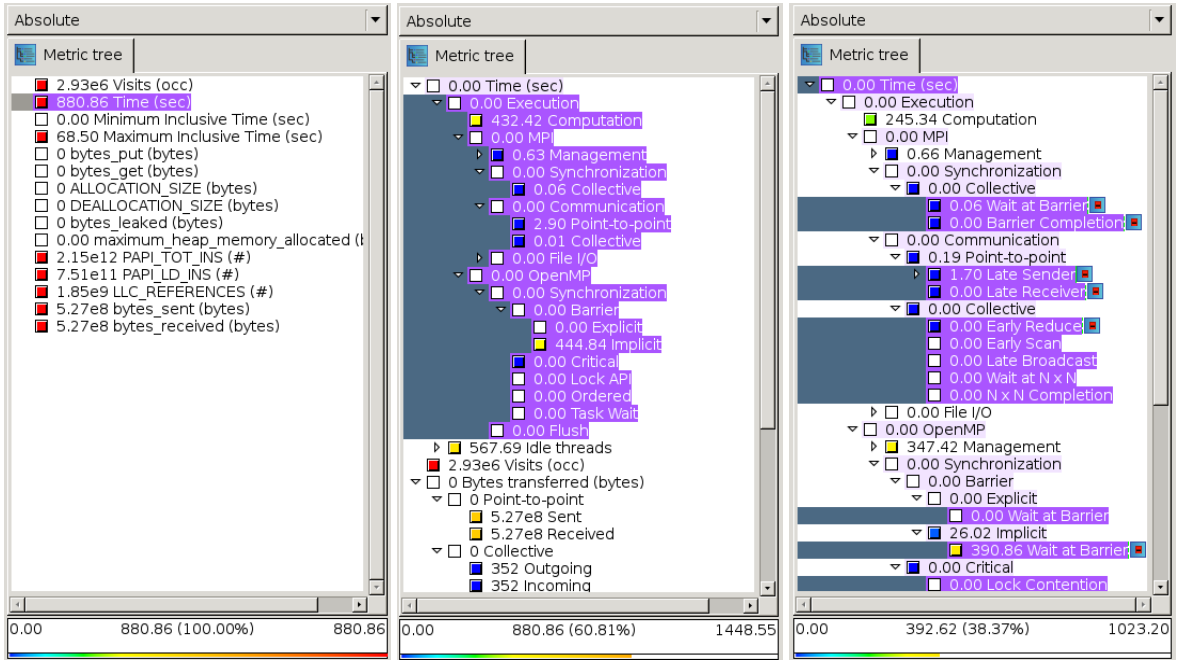


- When a node is selected from any tree, its *severity* value (and percentage) are shown in the data panel below it. When multiple nodes are selected, their aggregate value is used. The context of intermediate tree nodes up to the root of the current selection(s) is distinguished with shading.
- The colour scale for coding metric severity values is at the bottom of the window. Alternate and customised colour scales can be defined as appropriate. Zero (or the minimum metric value) appears on the left, with the total (or maximum metric value) on the right. Each panel typically has its own value scale, shown in the data panel below it.

- Metric severity values can be displayed in various modes:

Mode	Description
Absolute	Absolute value in the corresponding unit of measurement.
Root percent	Percentage relative to the inclusive value of the root node of the corresponding hierarchy.
Selection percent	Percentage relative to the value of selected node in corresponding tree browser to the left.
Peer percent	Percentage relative to the maximum of all peer values (all values of the current leaf level).
Peer distribution	Percentage relative to the maximum and non-zero minimum of all peer values.
External percent	Similar to "Root percent," but reference values are taken from another experiment.

Metrics pane



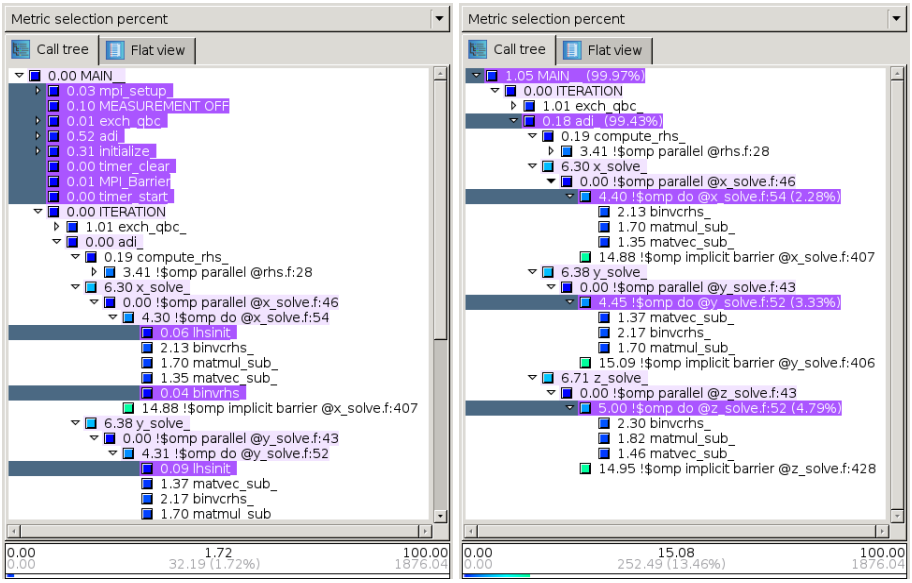
(a) SCORE-P profile (b) SCALASCA summary (c) SCALASCA trace analysis

Figure 1: List of basic metrics available in a SCORE-P profile (including optional PAPI and native hardware counter metrics), post-processed by SCALASCA to derive additional metrics and hierarchies, and compared to augmented version offering wait-state characterisation metrics (amongst others) only available from SCALASCA automated trace analysis.

Metric tree

- SCORE-P profile and SCALASCA scout trace analysis reports provide a number of basic metrics in a (flat) list, such as *Time* and *Visits*. Optional metrics recorded, such as hardware counters and OS resource usage, appear appended to the default list of metrics. Additional metrics can also be derived on demand.
- SCALASCA analysis report post-processing (known as 'remapping') derives hierarchies of metrics from the basic metrics, distinguishing *Computation* time used for local calculation from time in MPI, OpenMP and other categories associated with parallel execution runtime costs.
- In a hierarchical metric tree, concise node names may be ambiguous unless qualified by the intermediate nodes up to the root (where the unit of measurement is shown), e.g., Time/ Execution/ OpenMP/ Synchronization is therefore "OpenMP synchronization time" in seconds.
- SCALASCA automated trace analysis determines a variety of additional metrics, including a variety of blocking/waiting times for communication and synchronisation, the critical path of execution, and others. Statistics for the number and duration of the severest instances of blocking/waiting events are also available (when indicated).
- Descriptions of these metrics, how they are defined and can be employed in diagnosis, are available within the GUI itself or can be browsed on the SCALASCA website:
  `https://apps.fz-juelich.de/scalasca/releases/scalasca/current/help/scalasca_patterns.html`

Context pane

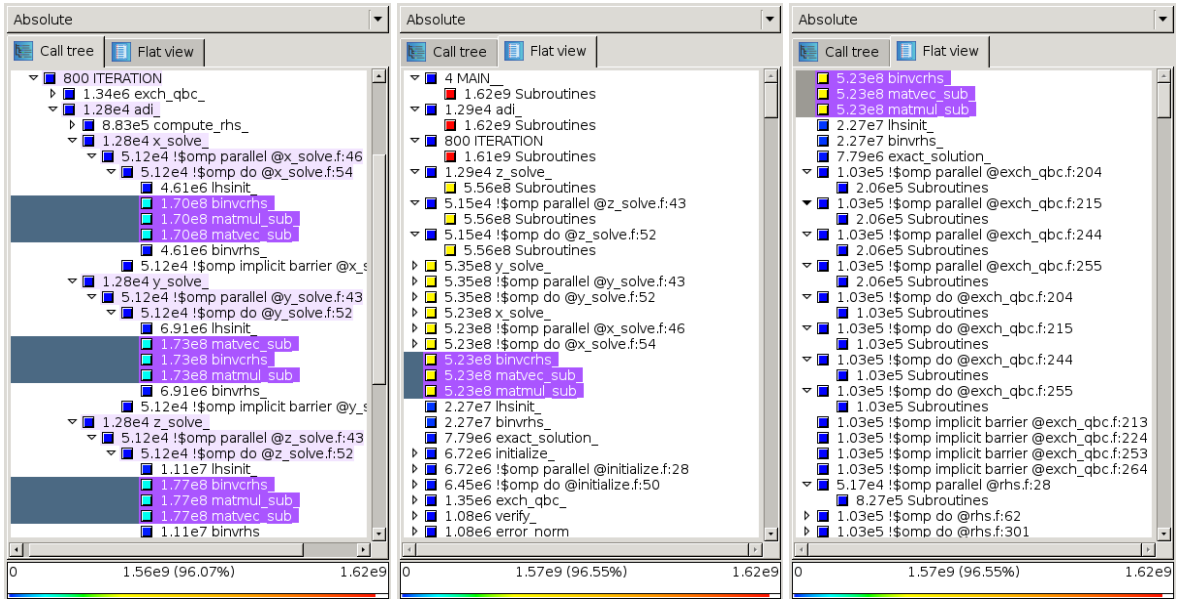


(a) Selected values below 1%. (b) Hidden values below 1%.

Figure 2: Context calltree with callpaths marked and hidden that have inclusive values of *Time* metric less than 1%. Values for hidden callpaths are attributed to their parents' values (with hidden proportion in parenthesis).

Call tree

- Presents execution call-paths in a tree (or multiple trees), typically with `main`, `MAIN_` or a program name as root. Regions that are executed from a call-path (including functions and subroutines called) appear as child nodes, shown by default in the order of first encounter.
- Call-tree nodes are shown as named by the compiler ('demangled' if possible), OpenMP instrumenter, or explicitly specified user-region annotation. Routine names can be shortened by not displaying module/namespace/class/etc.
- Call-tree nodes can be 'hidden', either individually or when their value is below a specified threshold percentage: values for 'hidden' nodes are not lost but get attributed to their parent, which is annotated with an indicator of the percentage of its new (exclusive) value that corresponds to hidden children.



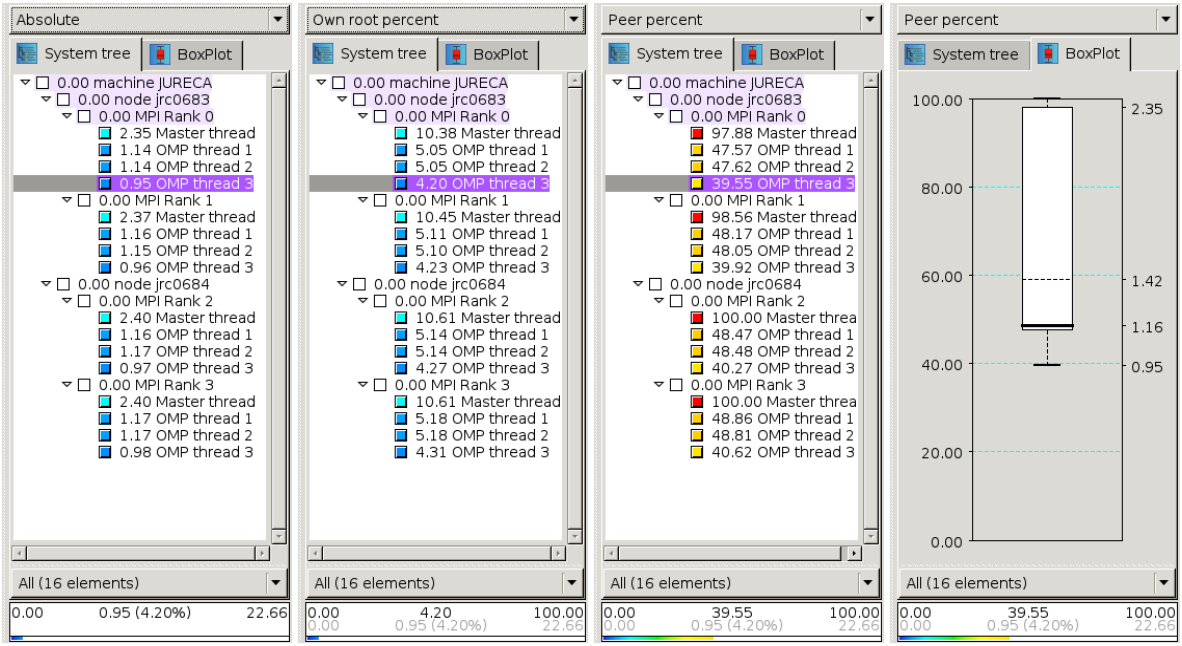
(a) Context calltree (b) Region list sorted by inclusive value (c) Region list sorted by exclusive value

Figure 3: Example of context calltree and flat list view of regions sorted by inclusive and exclusive value of *Visits* metric.

Region list flat view

- Regions and routines are aggregated from each call-path where they occur and listed, such that they may be conveniently sorted by name or the value of the currently selected metric.
- Children of each region are accumulated as its 'Subroutines' to be able to distinguish exclusive metric values.

System pane



(a) Absolute (b) Own root percent (c) Peer percent (d) Boxplot

Figure 4: Example of system tree and boxplot with various metric display modes (according to selection in menu at top).

System tree

- Tree of processes and associated threads for each compute node employed, sometimes with additional physical machine context levels (e.g., rack). When non-threaded, only the process is shown, otherwise initial thread is labeled 'Master'.
- Generally shown listed in rank order, but can alternatively be sorted by value.
- *Peer*-based metric modes use the maximum (and minimum non-zero) metric values of any single thread (or process).

System boxplot

- Summarizes system location value distribution statistics: whiskers show maximum/minimum values, with box covering range from lower (Q1) to upper (Q3) quartile containing bold line for median (Q2) and with mean shown by dashed line. Standard deviation and other statistics are provided in a detail popup dialog.
- The number and type of system locations is indicated by the mode menu below the boxplot, which by default is all threads but can be restricted to only threads which have visited the currently selected call-path(s) or other defined subsets of system locations.

Further information

- Scalasca/Cube project website `http://www.scalasca.org/` has a comprehensive Cube User Guide as well as the software for free download.
- Scalasca/Cube training is coordinated via the Virtual Institute – High Productivity Supercomputing (VI-HPS) and recent training material from tutorials and workshops is available from `http://www.vi-hps.org/training/`.