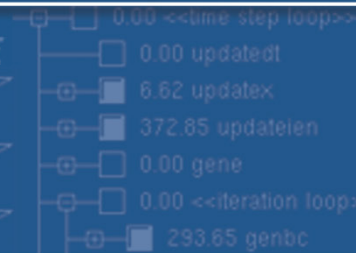


VI-HPS

SOFTWARE



FAST SOLUTIONS

- ☒ PAPI_L1_DCM
- ☒ PAPI_L1_ICM
- ☐ PAPI_L2_DCM
- ☒ PAPI_L2_ICM
- ☒ PAPI_L2_TCM
- ☐ PAPI_L2_TCM

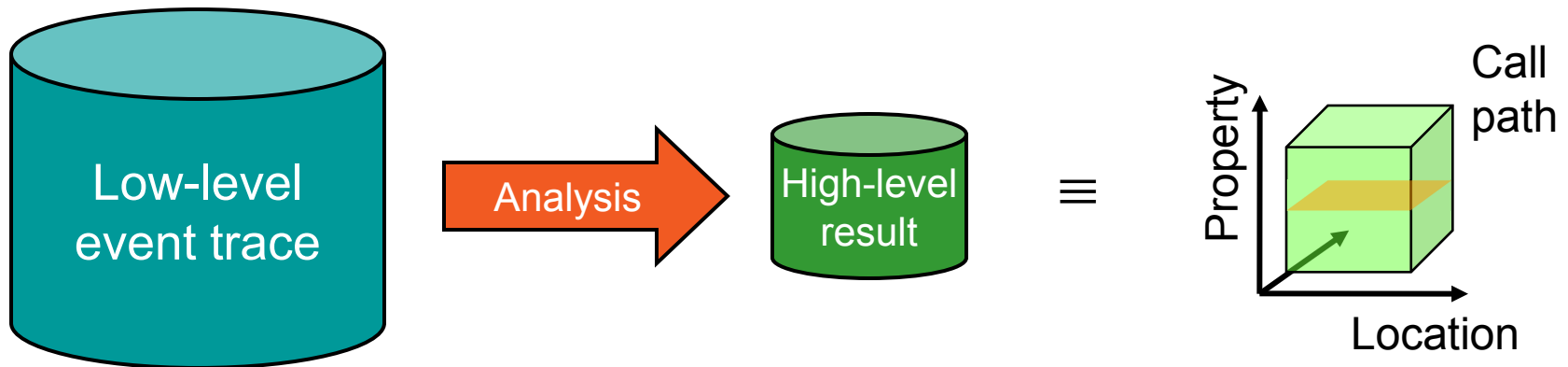
PRODUCTIVITY

Automatic trace analysis with Scalasca

Markus Geimer, Brian Wylie
Jülich Supercomputing Centre

scalasca

- Idea
 - Automatic search for patterns of inefficient behaviour
 - Classification of behaviour & quantification of significance



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

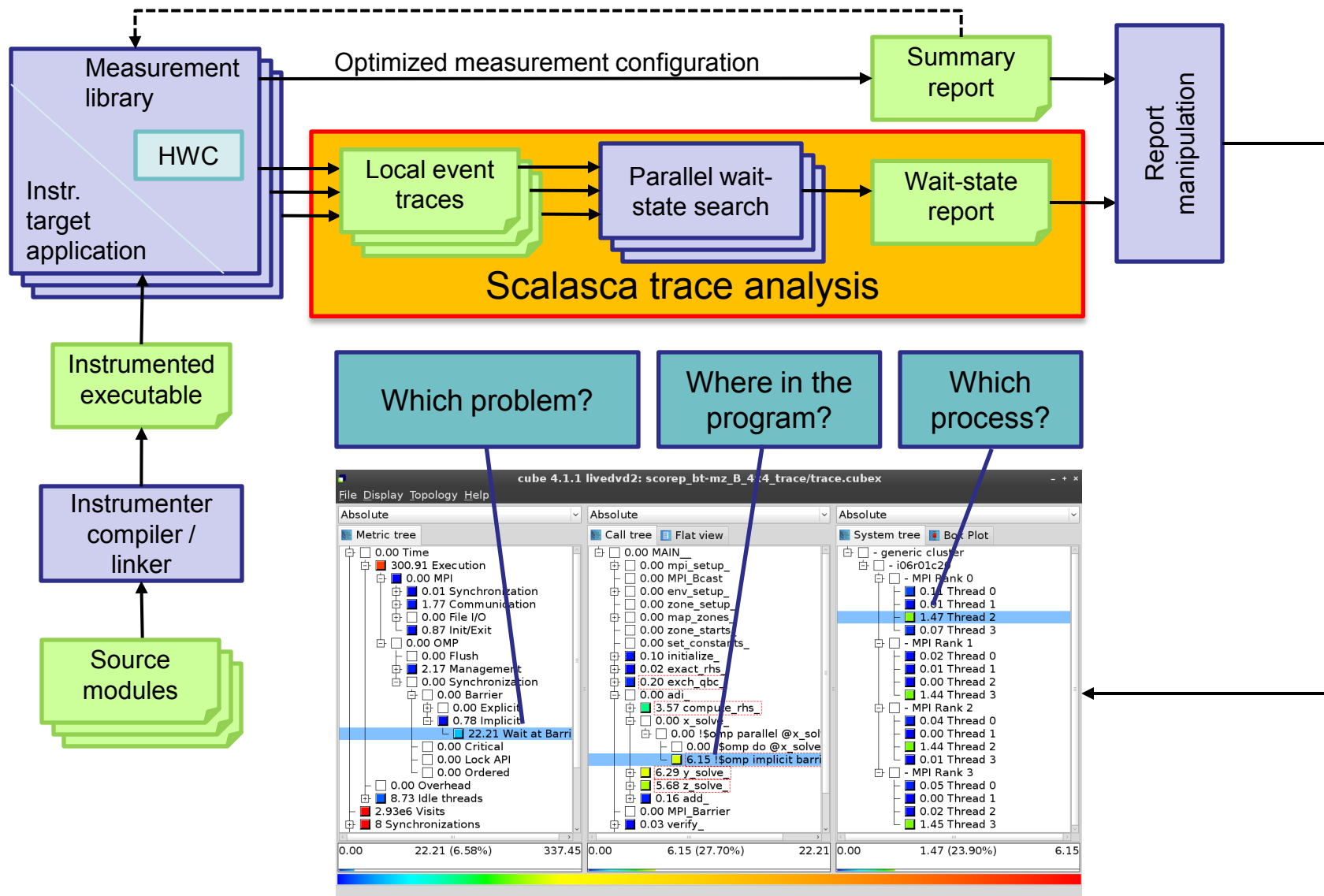
- Project started in 2006
 - Initial funding by Helmholtz Initiative & Networking Fund
 - Many follow-up projects
- Follow-up to pioneering KOJAK project (started 1998)
 - Automatic pattern-based trace analysis
- Now joint development of
 - Jülich Supercomputing Centre
 - German Research School for Simulation Sciences

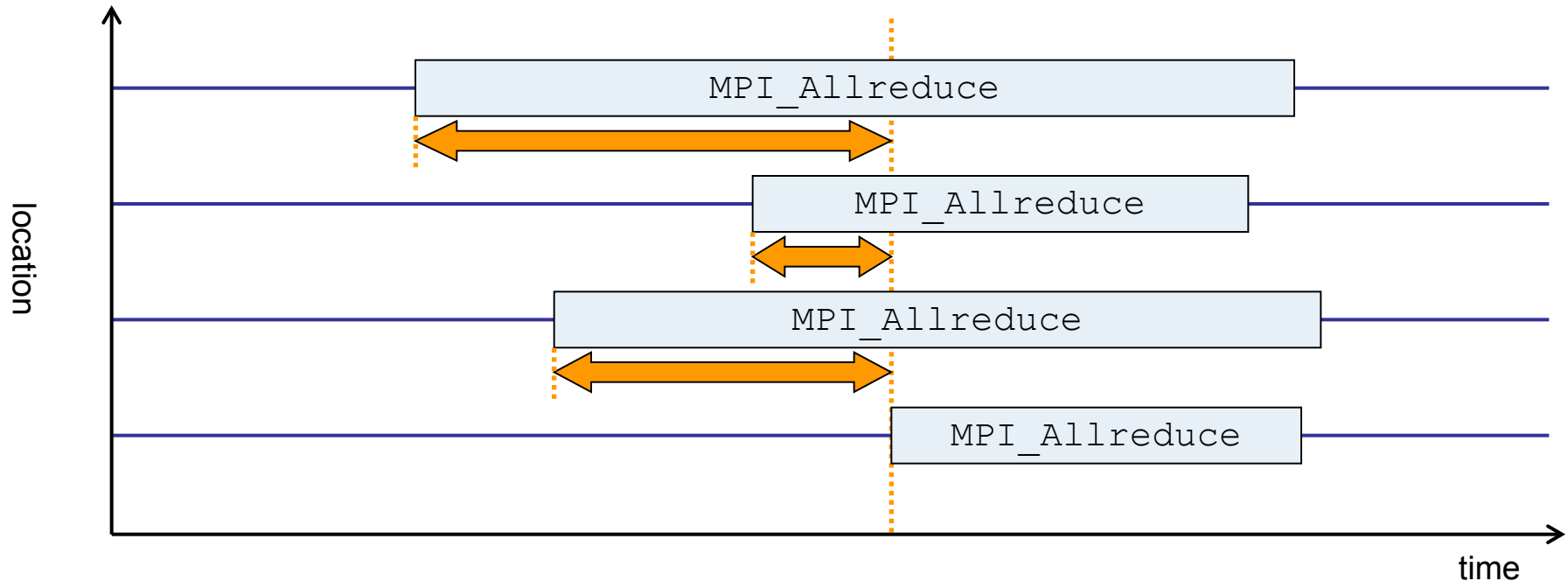


- Development of a **scalable** performance analysis toolset for most popular parallel programming paradigms
- Specifically targeting **large-scale** parallel applications
 - such as those running on IBM BlueGene or Cray XT systems with one million or more processes/threads
- Latest release in March 2013: Scalasca v1.4.3
- Here: Scalasca v2.0 β with Score-P support
 - release targeted with Score-P v1.2 in summer

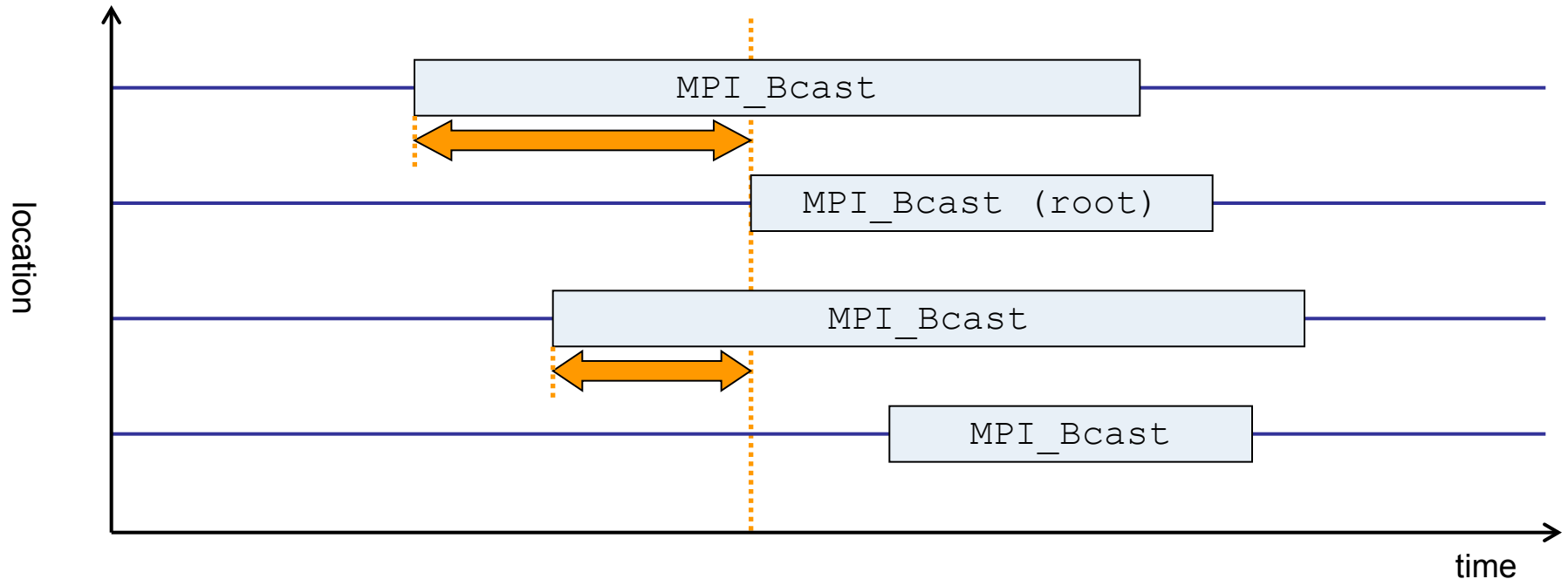
- Open source, New BSD license
- Portable
 - IBM BlueGene, IBM SP & blade clusters, Cray XT, SGI Altix, Fujitsu FX10 & K computer, NEC SX, Intel Xeon Phi, Solaris & Linux clusters, ...
- Supports parallel programming paradigms & languages
 - MPI, OpenMP & hybrid MPI+OpenMP
 - Fortran, C, C++
- Integrated instrumentation, measurement & analysis toolset
 - Runtime summarization (callpath profiling)
 - Automatic event trace analysis

- Open source, New BSD license
- Uses Score-P instrumenter & measurement libraries
 - Scalasca 2.0 core package focuses on trace-based analyses
 - Generally same usage as Scalasca 1.4
- Supports common data formats
 - Reads event traces in OTF2 format
 - Writes analysis reports in CUBE4 format
- Still aims to be portable
 - But not widely tested yet
 - Known issues
 - Unable to handle OTF2 traces containing CUDA events
 - Trace flush & pause event regions not handled correctly
 - OTF2 traces created with SIONlib not handled correctly



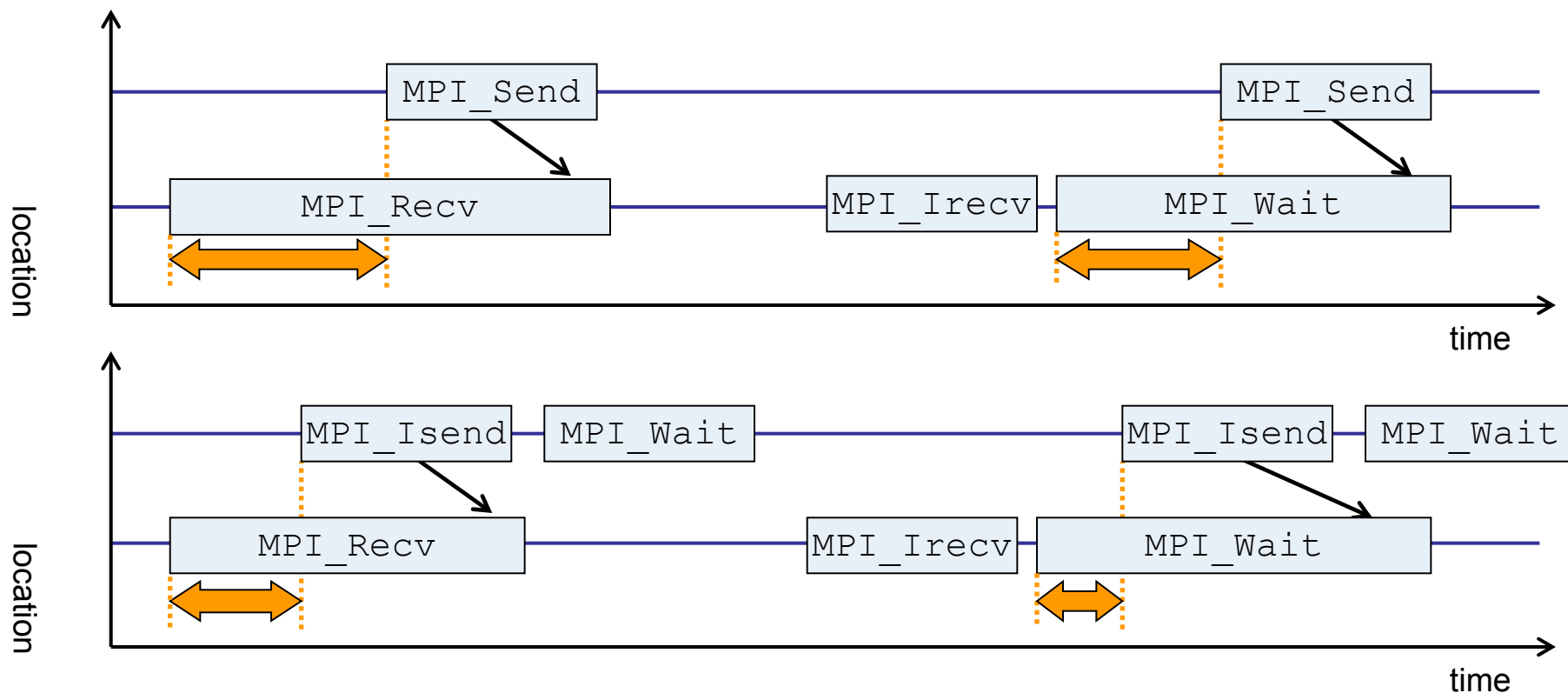


- Time spent waiting in front of synchronizing collective operation until the last process reaches the operation
- Applies to: MPI_Allgather, MPI_Allgatherv, MPI_Alltoall, MPI_Reduce_scatter, MPI_Reduce_scatter_block, MPI_Allreduce



- Waiting times if the destination processes of a collective 1-to-N operation enter the operation earlier than the source process (root)
- Applies to: MPI_Bcast, MPI_Scatter, MPI_Scatterv

Example: Late Sender



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

VI-HPS



Hands-on: NPB-MZ-MPI / BT

scalasca

- Scalasca application instrumenter

```
% skin  
Scalasca 2.0: application instrumenter using scorep  
usage: skin [-v] [-comp] [-pdt] [-pomp] [-user] <compile-or-link-cmd>  
  -comp={all|none|...}: routines to be instrumented by compiler  
                        (... custom instrumentation specification for compiler)  
  -pdt:  process source files with PDT instrumenter  
  -pomp: process source files for POMP directives  
  -user: enable EPIK user instrumentation API macros in source code  
  -v:    enable verbose commentary when instrumenting  
  
  --*:   options to pass to Score-P instrumenter
```

- Provides compatibility with Scalasca 1.X
- Generally use Score-P instrumenter directly

- Scalasca measurement collection & analysis nexus

```
% scan
Scalasca 2.0: measurement collection & analysis nexus
usage: scan {options} [launchcmd [launchargs]] target [targetargs]
      where {options} may include:
  -h      Help: show this brief usage message and exit.
  -v      Verbose: increase verbosity.
  -n      Preview: show command(s) to be launched but don't execute.
  -q      Quiescent: execution with neither summarization nor tracing.
  -s      Summary: enable runtime summarization. [Default]
  -t      Tracing: enable trace collection and analysis.
  -a      Analyze: skip measurement to (re-)analyze an existing trace.
  -e exptdir   : Experiment archive to generate and/or analyze.
                  (overrides default experiment archive title)
  -f filtdir   : File specifying measurement filter.
  -l lockfile  : File that blocks start of measurement.
```

- Scalasca analysis report explorer

```
% square
Scalasca 2.0: analysis report explorer
usage: square [-v] [-s] [-f filtfiler] [-F] <experiment archive
              | cube file>
    -F          : Force remapping of already existing reports
    -f filtfiler : Use specified filter file when doing scoring
    -s          : Skip display and output textual score report
    -v          : Enable verbose mode
```

- **scan** configures Score-P measurement by setting some environment variables automatically
 - e.g., experiment title, profiling/tracing mode, filter file, ...
 - Precedence order:
 - Command-line arguments
 - Environment variables already set
 - Automatically determined values
- Also, **scan** includes consistency checks and prevents corrupting existing experiment directories
- For tracing experiments, after trace collection completes then automatic parallel trace analysis is initiated
 - uses identical launch configuration to that used for measurement (i.e., the same allocated compute resources)

- Run the application using the Scalasca measurement collection & analysis nexus prefixed to launch command

```
% cd bin.scorep
% OMP_NUM_THREADS=4 scan -f scorep.filt mpiexec -np 4 ./bt-mz_W.4
S=C=A=N: Scalasca 2.0 runtime summarization
S=C=A=N: ./scorep_bt-mz_W_4x4_sum experiment archive
S=C=A=N: Thu Sep 13 18:05:17 2012: Collect start
mpiexec -np 4 ./bt-mz_W.4

NAS Parallel Benchmarks (NPB3.3-MZ-MPI) - BT-MZ MPI+OpenMP Benchmark

Number of zones:      8 x      8
Iterations: 200      dt:    0.000300
Number of active processes:      4

[... More application output ...]

S=C=A=N: Thu Sep 13 18:05:39 2012: Collect done (status=0) 22s
S=C=A=N: ./scorep_bt-mz_W_4x4_sum complete.
```

copy / edit / lsubmit
../jobscripts/mds/run_scan.ll

- Creates experiment directory `./scorep_bt-mz_W_4x4_sum`

- Score summary analysis report

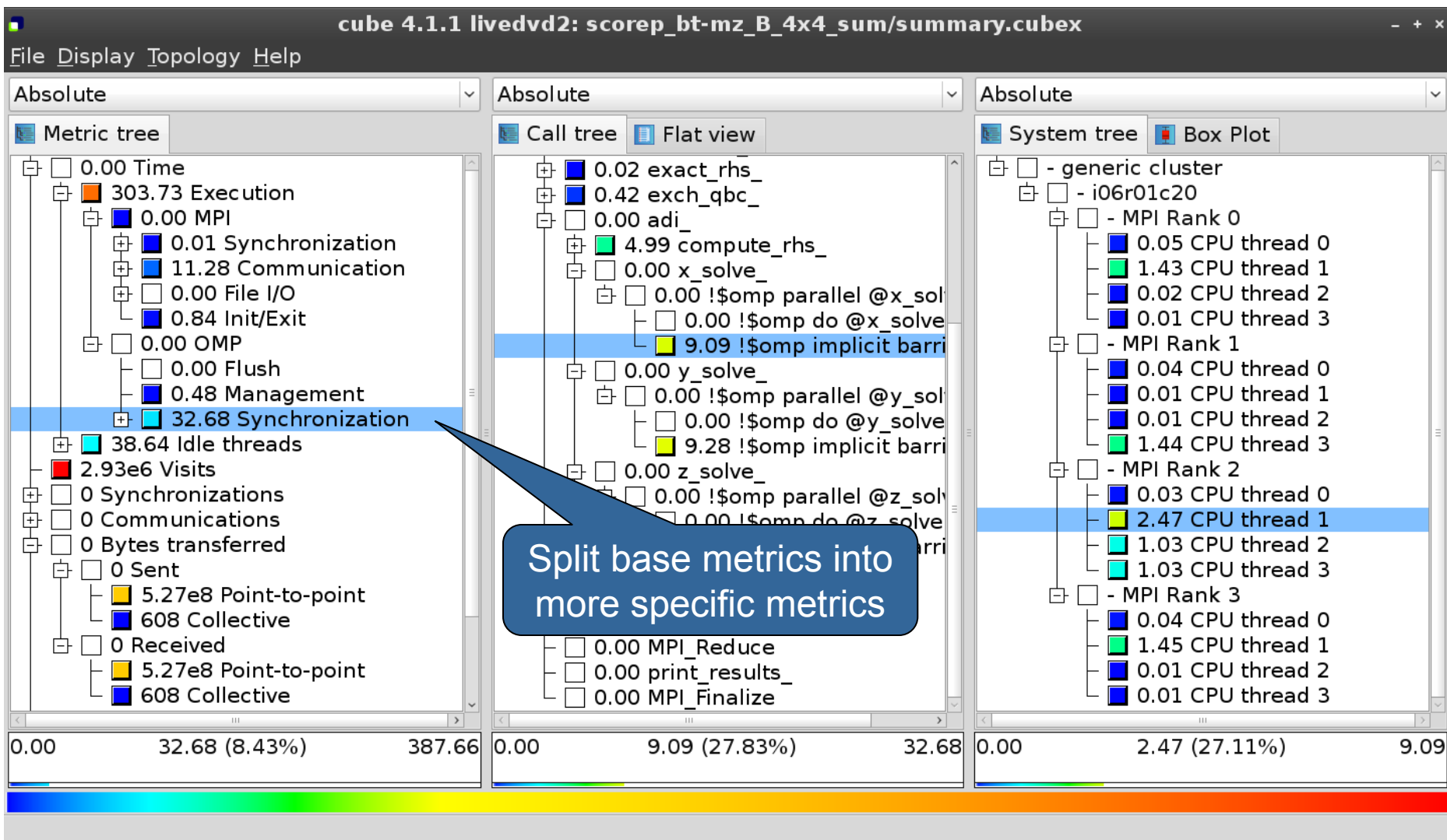
```
% square -s scorep_bt-mz_W_4x4_sum  
INFO: Post-processing runtime summarization result...  
INFO: Score report written to ./scorep_bt-mz_W_4x4_sum/scorep.score
```

- Post-processing and interactive exploration with CUBE

```
% square scorep_bt-mz_W_4x4_sum  
INFO: Displaying ./scorep_bt-mz_W_4x4_sum/summary.cubex...  
  
[GUI showing summary analysis report]
```

- The post-processing derives additional metrics and generates a structured metric hierarchy

Post-processed summary analysis report



- To enable additional statistics and pattern instance tracking, set `SCAN_ANALYZE_OPTS="-i s"`

```
% export SCAN_ANALYZE_OPTS="-i -s"
```

- Re-run the application using Scalasca nexus with `"-t"` flag

```
% OMP_NUM_THREADS=4 scan -f scorep.filt -t mpiexec -np 4 ./bt-mz_W.4
S=C=A=N: Scalasca 2.0 trace collection and analysis
S=C=A=N: ./scorep_bt-mz_W_4x4_trace experiment archive
S=C=A=N: Thu Sep 13 18:05:39 2012: Collect start
mpiexec -np 4 ./bt-mz_W.4
  NAS Parallel Benchmarks (NPB3.3-MZ-MPI) - BT-MZ MPI+OpenMP Benchmark

Number of zones:    8 x    8
Iterations: 200      dt:    0.000300
Number of active processes:    4

[... More application output ...]

S=C=A=N: Thu Sep 13 18:05:58 2012: Collect done (status=0) 19s
[... continued ...]
```

- Continues with automatic (parallel) analysis of trace files

```
S=C=A=N: Thu Sep 13 18:05:58 2012: Analyze start
mpiexec -np 4 scout.hyb -i -s ./scorep_bt-mz_W_4x4_trace/traces.otf2
SCOUT   Copyright (c) 1998-2012 Forschungszentrum Juelich GmbH
        Copyright (c) 2009-2012 German Research School for Simulation
        Sciences GmbH

Analyzing experiment archive ./scorep_bt-mz_W_4x4_trace/traces.otf2

Opening experiment archive ... done (0.002s).
Reading definition data     ... done (0.004s).
Reading event trace data   ... done (0.669s).
Preprocessing               ... done (0.975s).
Analyzing trace data       ... done (0.675s).
Writing analysis report    ... done (0.112s).

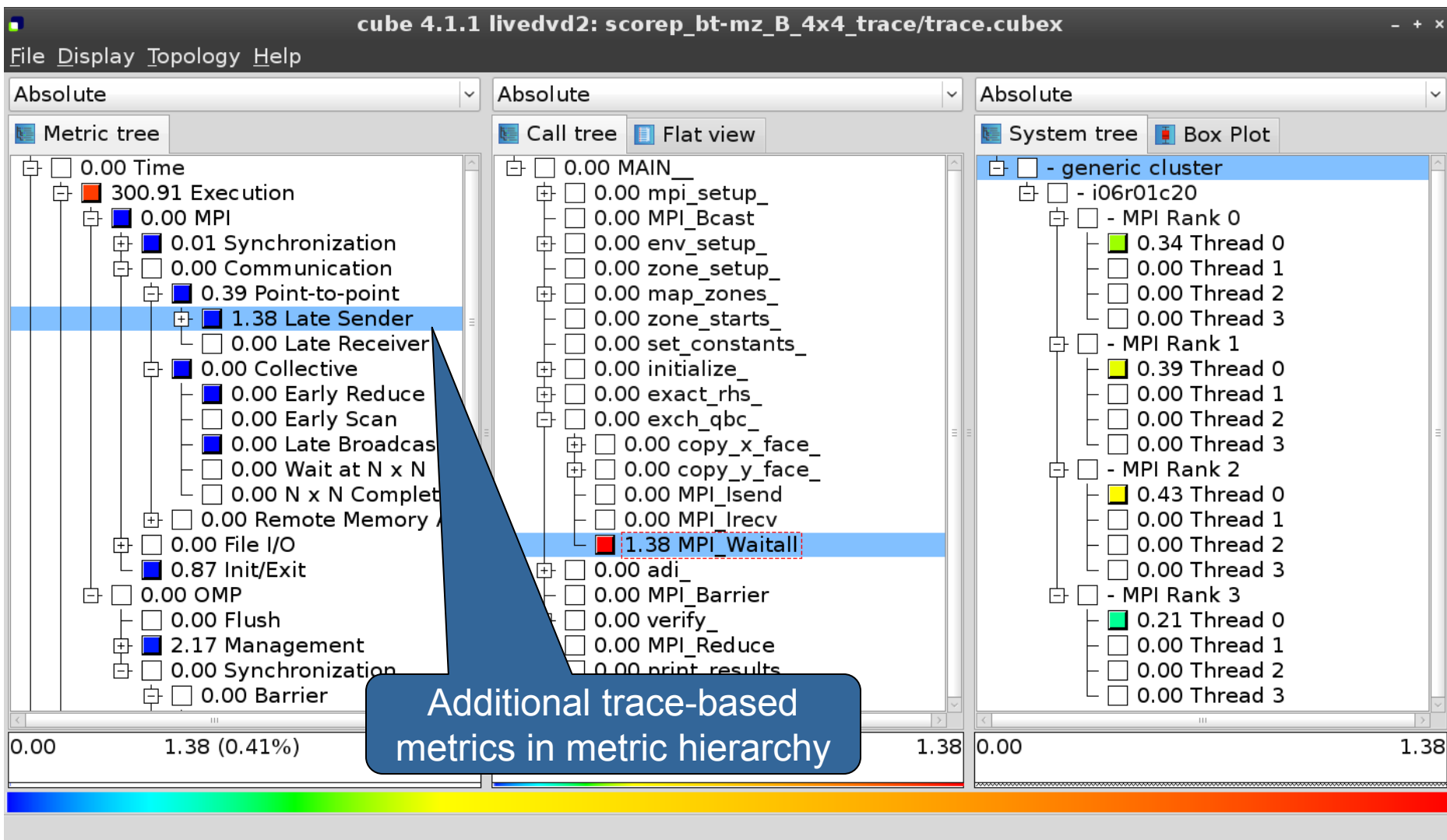
Max. memory usage          : 145.078MB

Total processing time      : 2.785s
S=C=A=N: Thu Sep 13 18:06:02 2012: Analyze done (status=0) 4s
```


- Produces trace analysis report in experiment directory containing trace-based wait-state metrics

```
% square scorep_bt-mz_W_4x4_trace  
INFO: Post-processing runtime summarization result...  
INFO: Post-processing trace analysis report...  
INFO: Displaying ./scorep_bt-mz_W_4x4_sum/trace.cubex...  
  
[GUI showing trace analysis report]
```

Post-processed trace analysis report



The screenshot displays the 'cube 4.1.1 livedvd2: score_bt-mz_B_4x4_trace/trace.cubex' application window. It features three main panels: 'Metric tree', 'Call tree', and 'System tree'. The 'Metric tree' panel on the left shows a hierarchical view of metrics, with 'Late Sender' (1.38) selected. A context menu is open over this item, listing options such as 'Info', 'Full info', 'Online description', 'Expand/collapse', 'Find items', 'Find Next', 'Clear found items', 'Copy to clipboard', 'Create derived metric...', 'Remove metric...', 'Statistics', and 'Max severity in trace browser'. The 'Online description' option is highlighted. The 'Call tree' panel in the middle shows a call stack, and the 'System tree' panel on the right shows a system tree. A blue callout box with a pointer indicates that the 'Online description' option is used to access the online metric description.

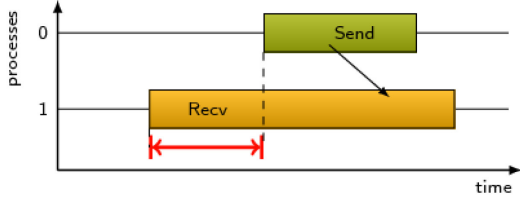
Access online metric description via context menu



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:

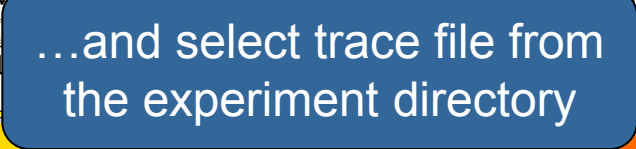
Close

The screenshot displays the VI-HPS interface with the file `cube 4.1.1 livedvd2: scorep_bt-mz_B_4x4_trace/trace.cubex` open. The 'Metric tree' on the left shows a hierarchy of metrics, with '1.38 Late Sender' selected. A context menu is open over this metric, listing options such as 'Info', 'Full info', 'Online description', 'Expand/collapse', 'Find items', 'Find Next', 'Clear found items', 'Copy to clipboard', 'Create derived metric...', 'Remove metric...', 'Statistics', and 'Max severity in trace browser'. The 'Statistics' option is highlighted. A callout bubble points to the 'Statistics' option with the text 'Access pattern instance statistics via context menu'. Another callout bubble points to the 'Statistics info' dialog box with the text 'Click to get statistics details'. The 'Statistics info' dialog box shows the following data for the pattern 'mpi_latesender':

Metric	Value	Percentage
Sum:	1.38	
Count:	832	
Mean:	0.00	5%
Standard deviation:	0.00	13%
Maximum:	0.03	100%
Upper quartile (Q3):	0.00	3%
Median:	0.00	3%
Lower quartile (Q1):	0.00	2%
Minimum:	0.00	0%

The dialog box also includes 'To Clipboard' and 'Close' buttons. The background interface shows a 'Call tree' and a 'Metric tree' with various metrics and their values. A status bar at the bottom indicates 'Shows metric statistics'.





Show most severe pattern instances

The screenshot displays the cube 4.1.1 livedvd2: scorep_bt-mz_B_4x4_trace/trace.cubex application interface. It features three main panels: Metric tree, Call tree, and System tree, all set to 'Absolute' scale.

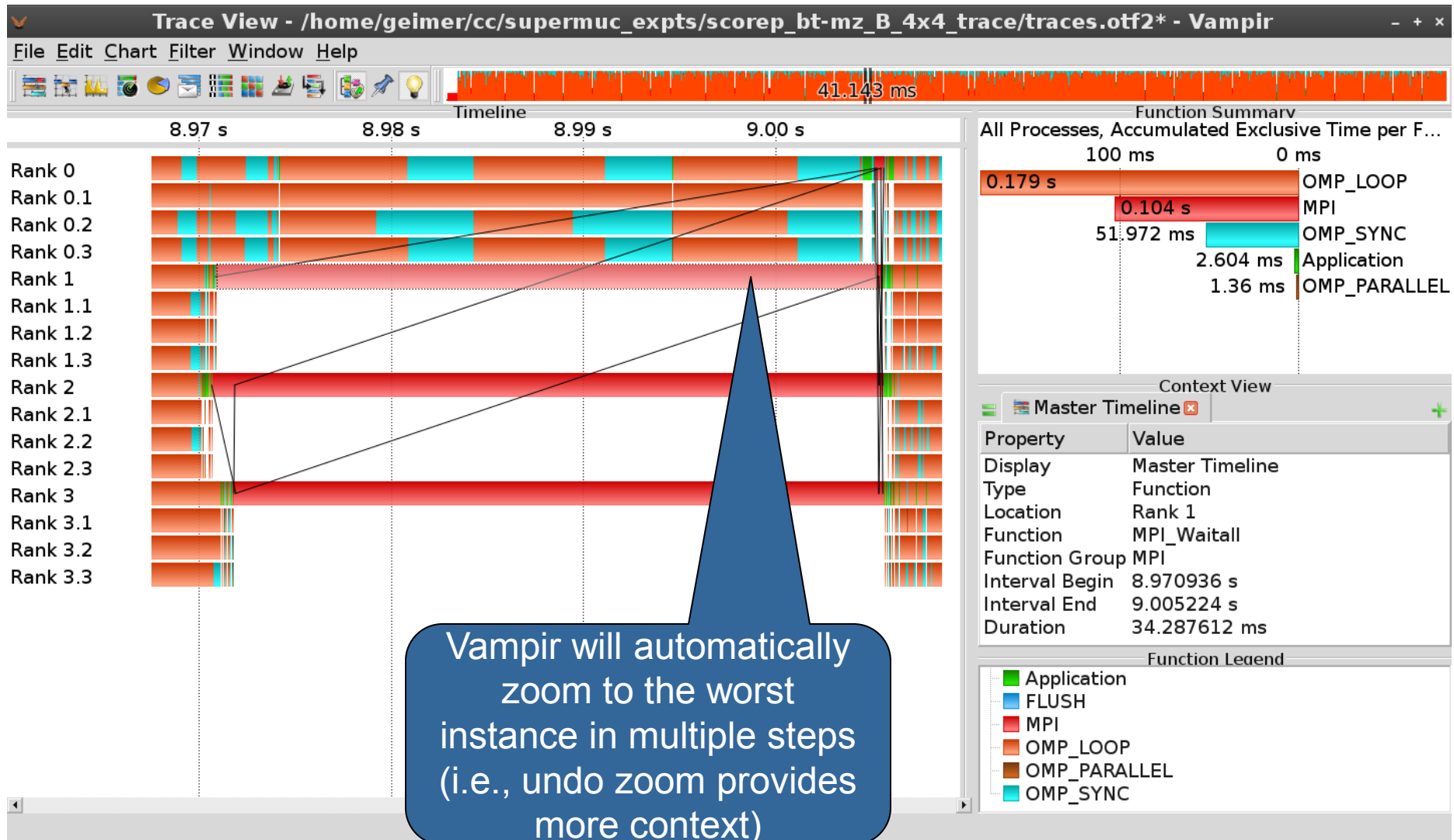
- Metric tree:** Shows a hierarchy of metrics. The 'Late Sender' metric is highlighted with a blue background and a red border, indicating it is the most severe instance of the 'Point-to-point' pattern.
- Call tree:** Displays a hierarchy of MPI calls. The '1.38 MPI_Wait' call is highlighted with a blue background and a red border, indicating it is the most severe instance of the 'MPI_Wait' pattern.
- System tree:** Shows a hierarchy of system components. The 'MPI_Wait' component is highlighted with a blue background and a red border, indicating it is the most severe instance of the 'MPI_Wait' pattern.

A context menu is open over the '1.38 MPI_Wait' item in the Call tree. The menu options are:

- Call site
- Called region
- Expand/collapse
- Hiding
- Cut call tree
- Find items
- Find Next
- Clear found items
- Copy to clipboard
- Min/max values
- Max severity in trace browser

A blue callout bubble points to the 'Max severity in trace browser' option with the text: "Select 'Max severity in trace browser' from context menu of call paths marked with a red frame".

Investigate most severe instance in Vampir



Scalable performance analysis of large-scale parallel applications

- toolset for scalable performance measurement & analysis of MPI, OpenMP & hybrid parallel applications
- supporting most popular HPC computer systems
- available under New BSD open-source license
- sources, documentation & publications:
 - <http://www.scalasca.org>
 - mailto: scalasca@fz-juelich.de

