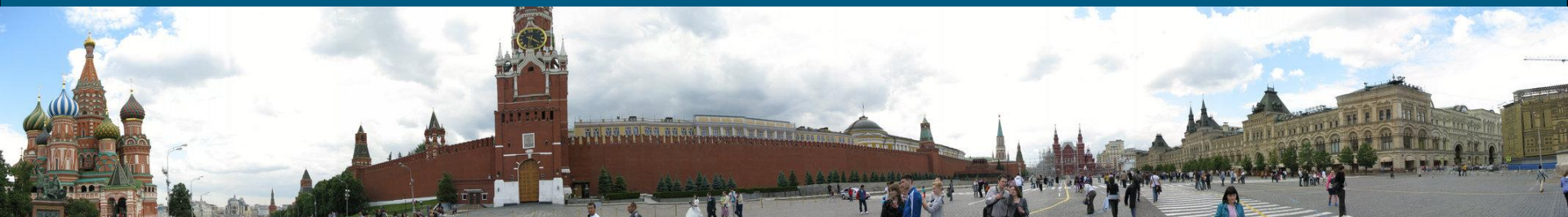


# Automatic Trace Analysis with Scalasca

Bernd Mohr, Jülich Supercomputing Centre

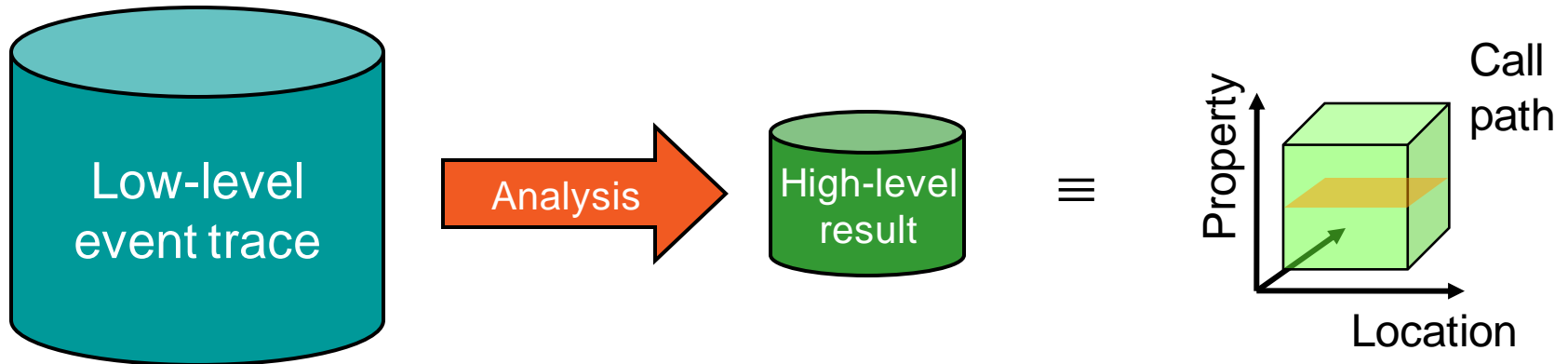


# Automatic Trace Analysis



- Idea

- Automatic search for patterns of inefficient behavior
- Classification of behavior & quantification of significance



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

# The Scalasca Project: Overview



- 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



# The Scalasca Project: Objective



- Development of a **scalable** performance analysis toolset
- Specifically targeting **large-scale** parallel applications
  - such as those running on IBM BlueGene or Cray XT with 10,000s to 100,000s of processes
- Latest release in July 2012: Scalasca v1.4.2
- Here: Scalasca v2.0α with Score-P support  
(no release date yet, available on request)

# Scalasca 1.4 Features



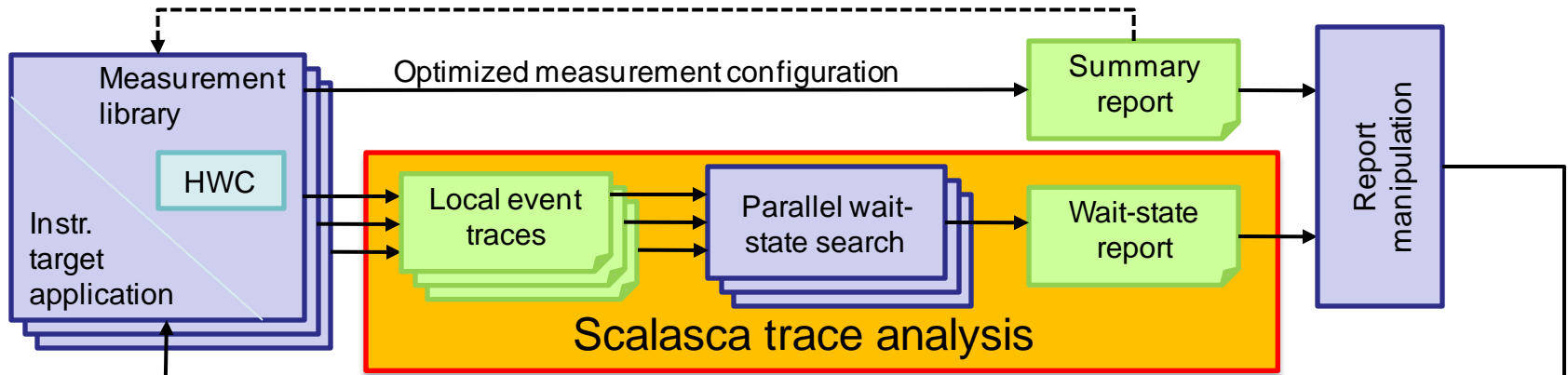
- Open source, New BSD license
- Portable
  - BG/Q, BG/P, BG/L, IBM SP & blade clusters, Cray XT + XK, NEC SX, Solaris & Linux clusters, ...
- Supports parallel programming paradigms & languages
  - MPI, OpenMP & hybrid OpenMP/MPI
  - Fortran, C, C++
- Integrated measurement & analysis toolset
  - Runtime summarization (aka profiling)
  - Automatic event trace analysis
- Uses Scalasca internal EPILOG trace and CUBE3 formats

# Scalasca 2.0 $\alpha$ Features

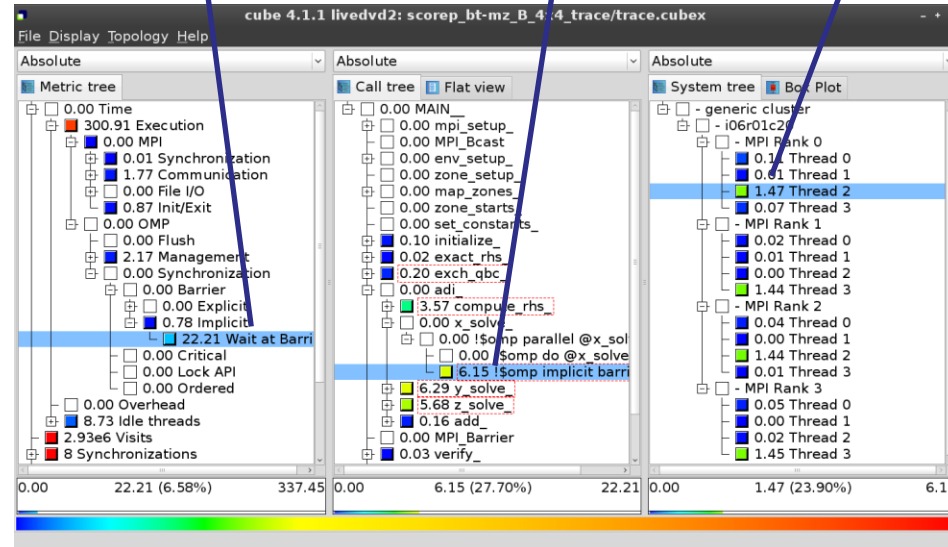


- Open source, New BSD license
- Still aims to be portable
  - But not widely tested yet
- Scalasca 1.4 measurement system superseded by Score-P
  - Scalasca 2.0 focuses on trace-based analyses only
- Supports common data formats
  - Reads event traces in OTF2 format
  - Writes analysis results in CUBE4 format

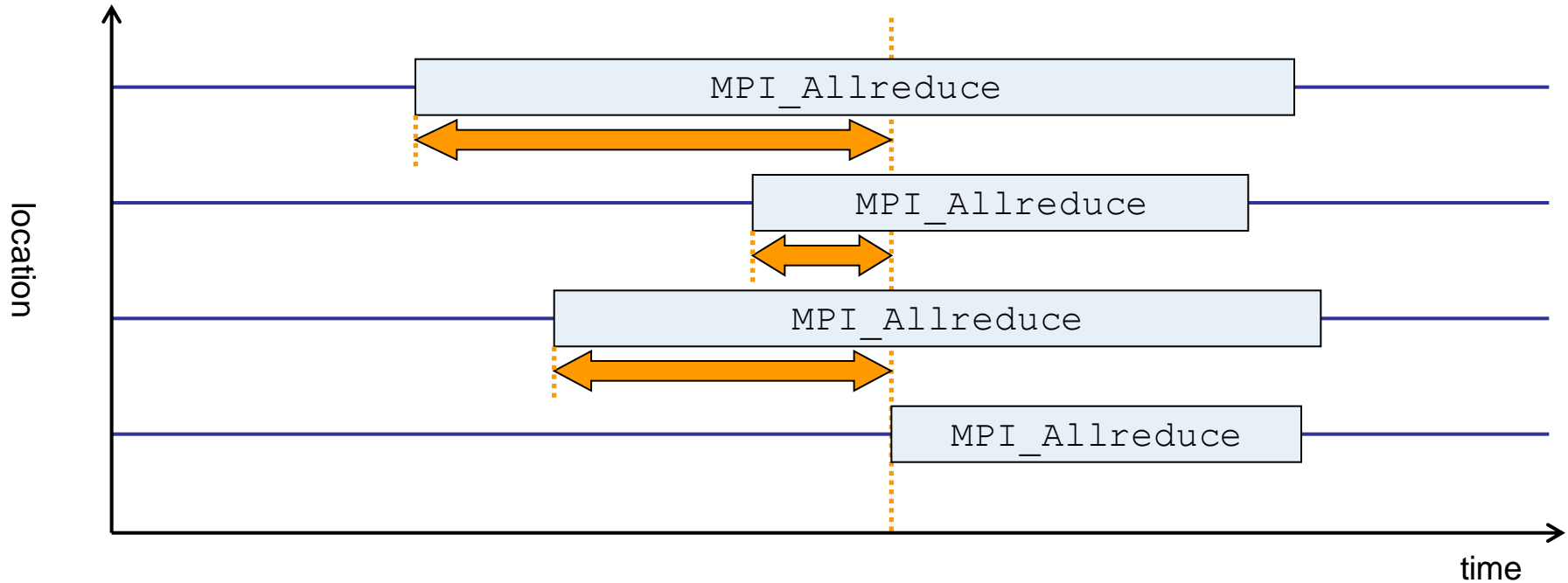
# Scalasca Workflow



Which problem?      Where in the program?      Which process?



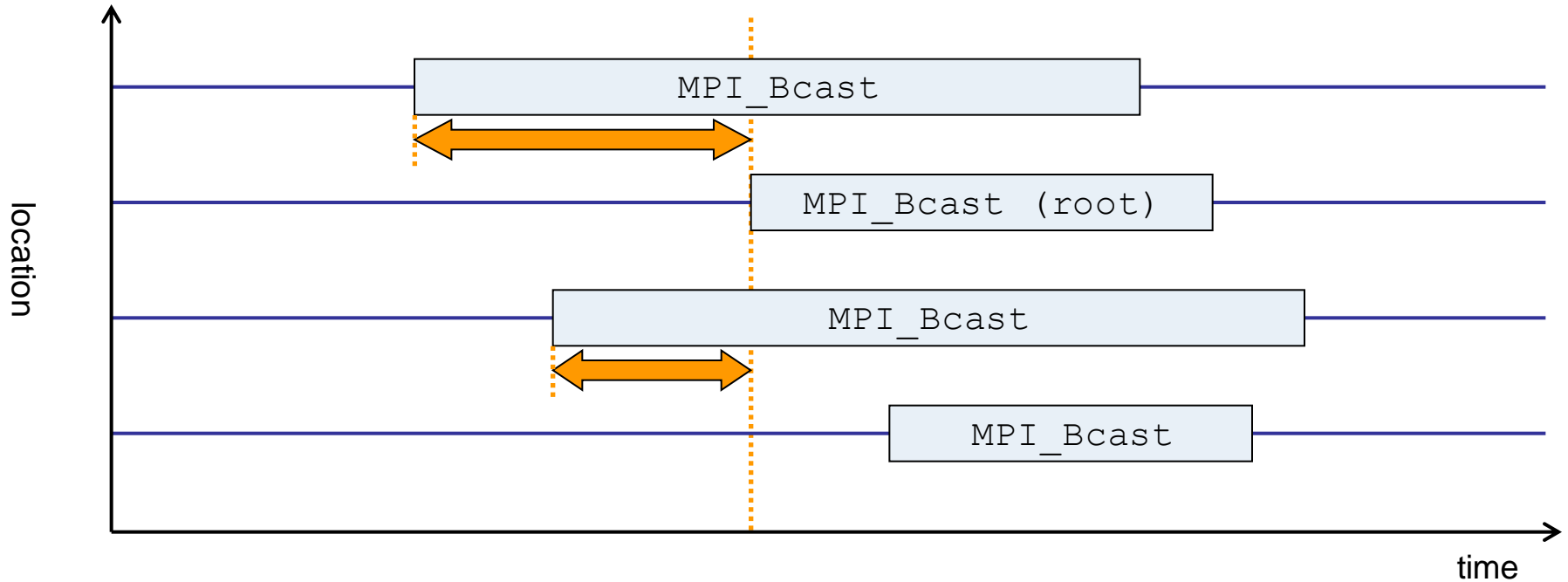
# Example: Wait at NxN



- 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

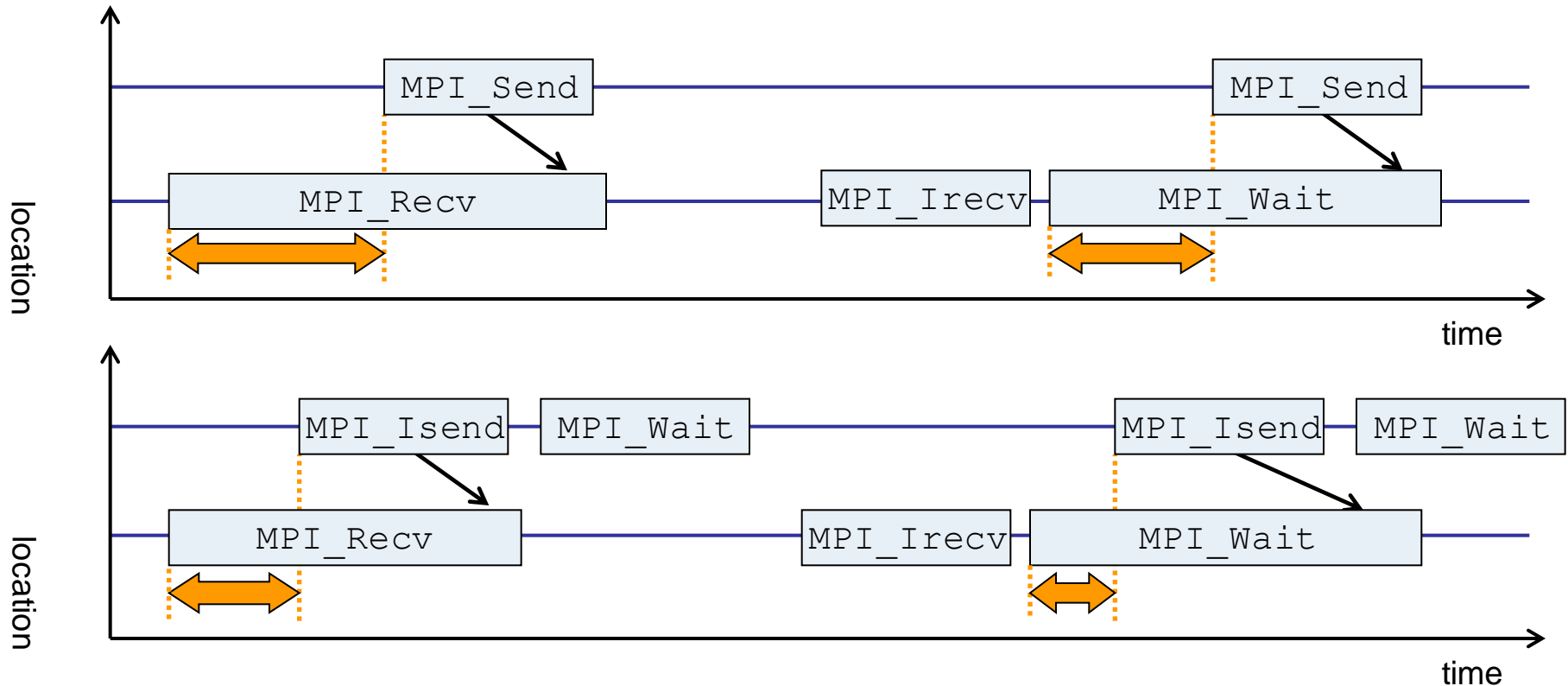


# Example: Late Broadcast



- 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



# Hands-on: NPB-MZ-MPI / BT

scalasca 

# Scalasca Helper Commands



- 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.
- m metrics : Metric specification for measurement.

# Scalasca Helper Commands (cont.)



- 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
```

# Before Collecting a New Measurement...



- **scan** configures Score-P by setting some environment variables automatically
  - Precedence order:
    - Command-line arguments
    - Environment variables already set
    - Automatically determines values
- Also, **scan** prevents overwriting experiment directories

# BT-MZ Summary Measurement



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

```
% cd bin.scorep
% export OMP_NUM_THREADS=4
% scan -f ../config/scorep.filt mpiexec -n 4 ./bt-mz_B.4
S=C=A=N: Scalasca 2.0 runtime summarization
S=C=A=N: ./scorep_bt-mz_B_4x4_sum experiment archive
S=C=A=N: Thu Sep 13 18:05:17 2012: Collect start
mpiexec -n 4 ./bt-mz_B.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_B_4x4_sum complete.
```

- Creates experiment directory `./scorep_bt-mz_B_4x4_sum`

# BT-MZ Summary Analysis Report Examination



- Score summary analysis report

```
% square -s scorep_bt-mz_B_4x4_sum  
INFO: Post-processing runtime summarization result...  
INFO: Score report written to ./scorep_bt-mz_B_4x4_sum/scorep.score
```

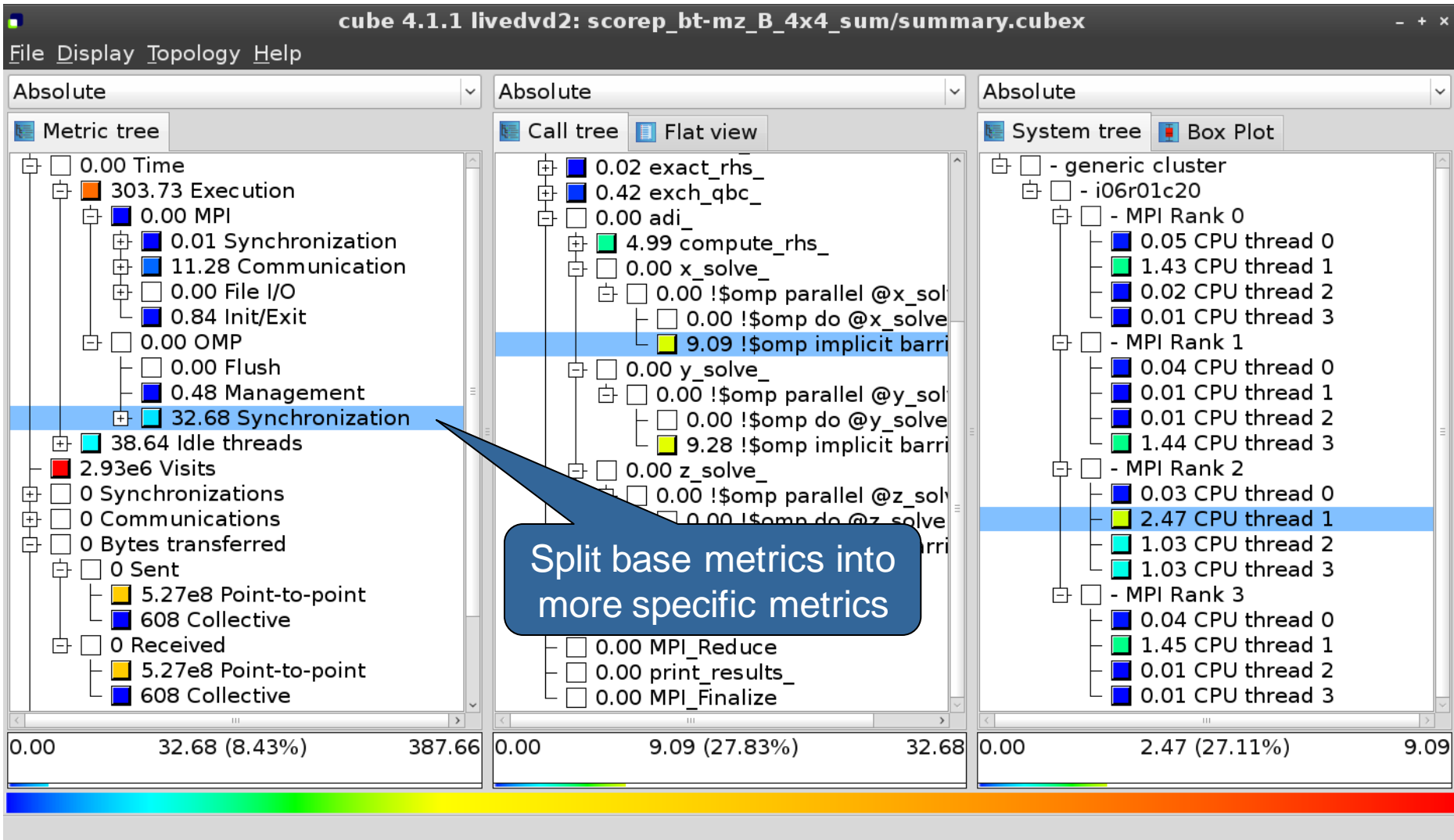
- Post-processing and interactive exploration with CUBE

```
% square scorep_bt-mz_B_4x4_sum  
INFO: Displaying ./scorep_bt-mz_B_4x4_sum/summary.cubex...  
  
[GUI showing summary analysis report]
```

- The post-processing generates a metric hierarchy, splitting some base metrics into more specific metrics



# Post-Processed Summary Analysis Report





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

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

- Re-run the application using Scalasca nexus with “-t” flag

```
% export OMP_NUM_THREADS=4
% scan -f ../config/scorep.filt -t mpiexec -n 4 ./bt-mz_B.4
S=C=A=N: Scalasca 2.0 trace collection and analysis
S=C=A=N: ./scorep_bt-mz_B_4x4_trace experiment archive
S=C=A=N: Thu Sep 13 18:05:39 2012: Collect start
mpiexec -n 4 ./bt-mz_B.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 ...]
```

# ... and Analysis



- Continues with automatic (parallel) analysis of trace files

```
S=C=A=N: Thu Sep 13 18:05:58 2012: Analyze start
mpiexec -n 4 scout.hyb -i ./scorep_bt-mz_B_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_B_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
```

## ... on Graphit



- Run the application using the tracing mode of Score-P

```
% vim run.sh
% export SCOREP_EXPERIMENT_DIRECTORY=scorep_bt-mz_B_4x4_trace
% export SCOREP_FILTERING_FILE=../config/scorep.filt
% export SCOREP_ENABLE_TRACING=true
% export SCOREP_ENABLE_PROFILING=false
% export SCOREP_TOTAL_MEMORY=100M

% cleo-submit -np 4 ./run.sh

NAS Parallel Benchmarks (NPB3.3-MZ-MPI) - BT-MZ MPI+OpenMP Benchmark
[... More application output ...]
```

- Manually execute the scalasca trace analyzer:
  - `scout.mpi` or `scout.hby`

```
% cleo-submit -np 4 `which scout.hyb` -i -s -v \  
scorep_bt-mz_B_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_B_4x4_trace/traces.otf2
...
```

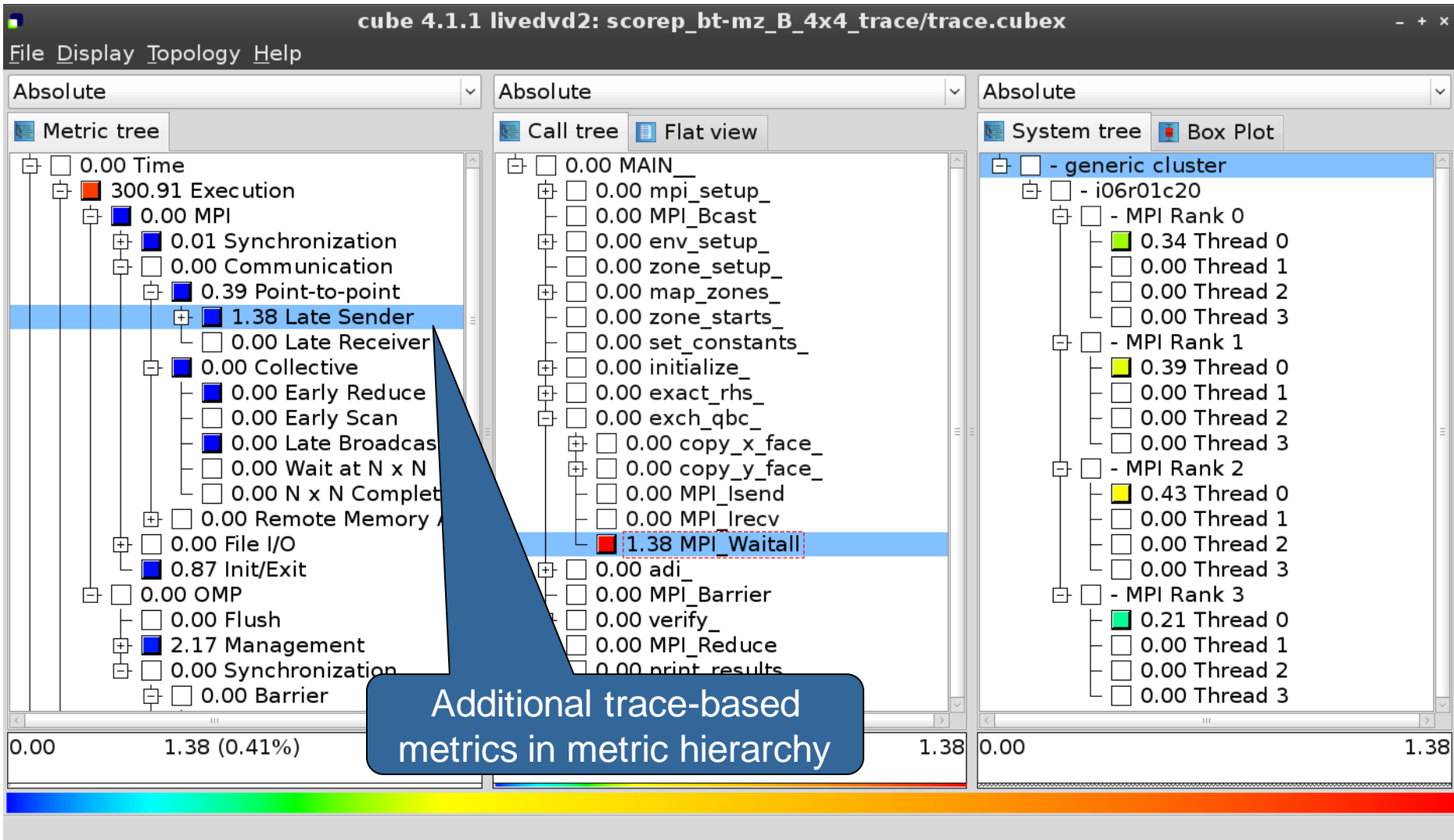
# BT-MZ Trace Analysis Report Exploration



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

```
% square scorep_bt-mz_B_4x4_trace  
INFO: Post-processing runtime summarization result...  
INFO: Post-processing trace analysis report...  
INFO: Displaying ./scorep_bt-mz_B_4x4_sum/trace.cubex...  
  
[GUI showing trace analysis report]
```

# Post-Processed Trace Analysis Report



# Online Metric Description



cube 4.1.1 livedvd2: scorep\_bt-mz\_B\_4x4\_trace/trace.cubex

File Display Topology Help

Absolute Absolute Absolute

Metric tree Call tree Flat view System tree Box Plot

0.00 Time  
300.91 Execution  
0.00 MPI  
0.01 Synchronization  
0.00 Communication  
0.39 Point-to-point  
1.38 Late Sender  
0.00 Late Receiver  
0.00 Collective  
0.00 Early Receiver  
0.00 Early Sender  
0.00 Late Broadcast  
0.00 Wait at barrier  
0.00 N x N Communication  
0.00 Remote Memory Access  
0.00 File I/O  
0.87 Init/Exit  
0.00 OMP  
0.00 Flush  
2.17 Management  
0.00 Synchronization  
22.99 Barrier

0.00 MAIN\_  
0.00 mpi\_setup\_  
0.00 MPI\_Bcast  
0.00 env\_setup\_  
0.00 zone\_setup\_  
0.00 map\_zones\_  
0.00 zone\_starts\_  
0.00 MPI Rank 0  
0.34 Thread 0  
0.00 Thread 1  
0.00 Thread 2  
0.00 Thread 3  
0.00 MPI Rank 1  
0.39 Thread 0  
0.00 Thread 1  
0.00 Thread 2  
0.00 Thread 3  
0.00 MPI Rank 2  
0.43 Thread 0  
0.00 Thread 1  
0.00 Thread 2  
0.00 Thread 3  
0.00 MPI Rank 3  
0.21 Thread 0  
0.00 Thread 1  
0.00 Thread 2  
0.00 Thread 3

0.00 1.38 (0.41%) 337.45 0.00 1.38 (100%) 1.38

Shows the online description of the clicked item

Info  
Full info  
Online description  
Expand/collapse  
Find items  
Find Next  
Clear found items  
Copy to clipboard  
Create derived metric...  
Remove metric...  
Statistics  
Max severity in trace browser

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.

The diagram is a timeline with two horizontal axes representing processes, labeled '0' and '1'. Process 0 has a green box labeled 'Send' starting at a certain point in time. Process 1 has a yellow box labeled 'Recv' starting earlier than the 'Send' operation. A red double-headed arrow is drawn below the 'Recv' box, extending from its start to the start of the 'Send' box, indicating the period of blocking. A dashed vertical line connects the start of the 'Send' box to the 'Recv' box, showing that the message is not yet available when the receiver starts.

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



# Pattern Instance Statistics



The screenshot shows the 'cube 4.1.1 livedvd2: scorep\_bt-mz\_B\_4x4\_trace/trace.cubex' application. The 'Metric tree' on the left shows a hierarchy where '1.38 Late Sender' is selected. A context menu is open over this node, with 'Statistics' highlighted. A 'Statistics info' dialog is open, displaying a box plot and a table of statistics for the 'mpi\_latesender' pattern. A second 'Statistics info' dialog is also visible in the background.

Pattern:	mpi_latesender
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%

Context menu options:

- Info
- Full info
- Online description
- Expand/collapse
- Find items
- Find Next
- Clear found items
- Copy to clipboard
- Create derived metric...
- Remove metric...
- Statistics
- Max severity in trace browser

Statistics info dialog content:

0.035  
0.03  
0.025  
0.02  
0.015  
0.01  
0.005  
0

Close

Statistics info dialog content:

Pattern: mpi\_latesender  
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%

To Clipboard Close

Click to get statistics details

Access pattern instance statistics via context menu

# Connect to Vampir Trace Browser



The screenshot shows the Vampir Trace Browser interface. The 'File' menu is open, with 'Connect to trace browser' selected. A dialog box titled 'Connect to vampir' is displayed, with the following fields: 'Open local file' (checked), 'Host: localhost', 'Port: 30000', and 'File: c:/supermuc\_expts/scorep\_bt-mz\_B\_4x4\_trace/traces.otf2'. The background shows a call tree and a system tree view.

**To investigate most severe pattern instances, connect to a trace browser...**

**...and select trace file from the experiment directory**

Connect to vampir and display a trace file

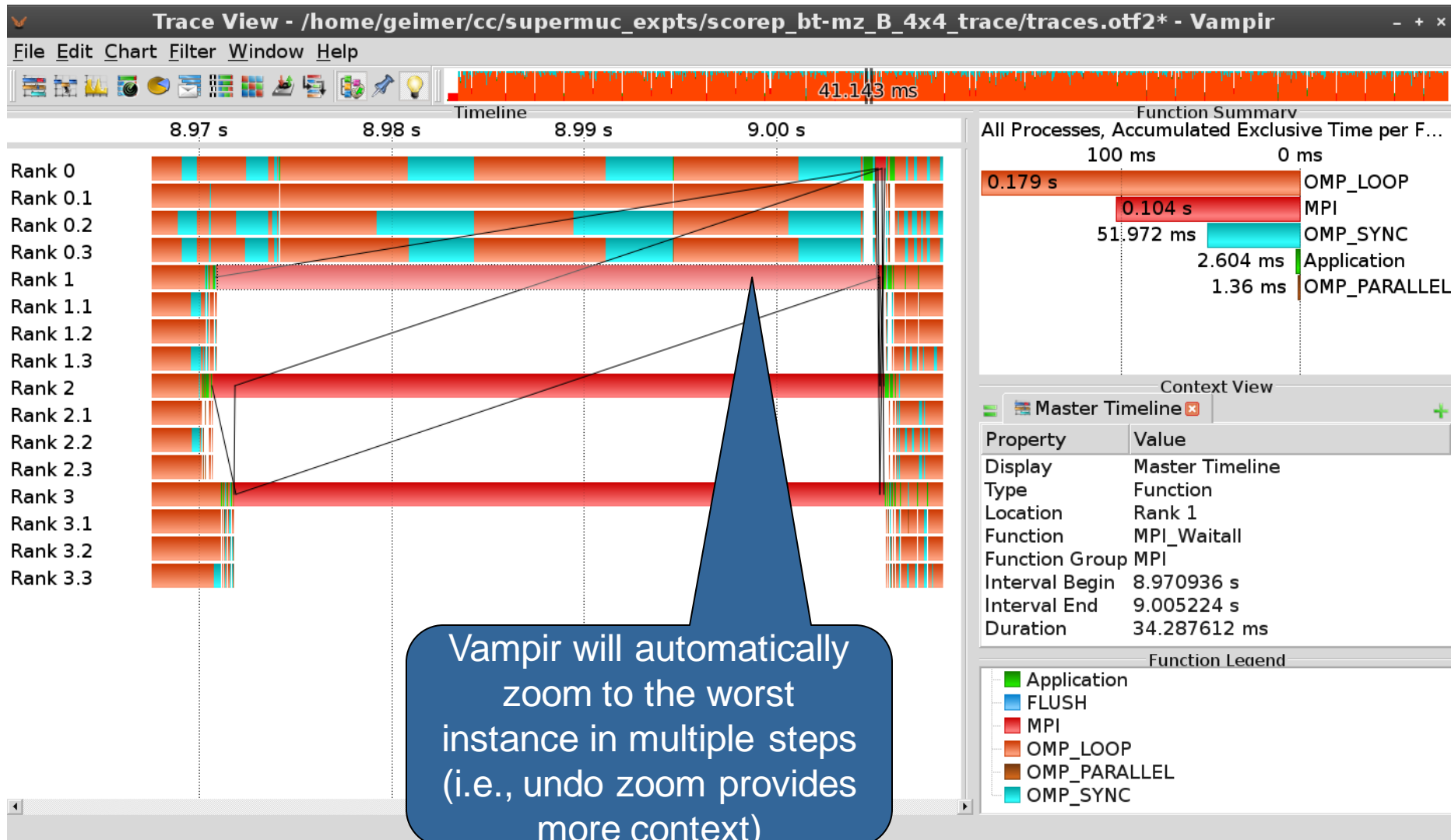
# Show Most Severe Pattern Instances



The screenshot displays the 'cube 4.1.1 livedvd2: scorep\_bt-mz\_B\_4x4\_trace/trace.cubex' application. It features three main panels: 'Metric tree' on the left, 'Call tree' in the center, and 'System tree' on the right. The 'Metric tree' shows a hierarchy of performance metrics, with '1.38 Late Sender' highlighted. The 'Call tree' shows a call path for '1.38 MPI\_Waitany' with a red border around it. A context menu is open over this call path, listing various actions such as 'Call site', 'Called region', 'Expand/collapse', and 'Max severity in trace browser'. The 'Max severity in trace browser' option is highlighted in blue. A blue callout box with white text points to this option, stating: 'Select "Max severity in trace browser" from context menu of call paths marked with a red frame'. The 'System tree' panel shows a 'Metric cluster' with threads for MPI Ranks 0, 1, 2, and 3.

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

**scalasca** 





# Questions?