ISC’18 Tutorial: Hands-on Practical Hybrid Parallel Application Performance Engineering

Christian Feld
Jülich Supercomputing Centre

Markus Geimer
Jülich Supercomputing Centre

Sameer Shende
University of Oregon

Ronny Tschüter
TU Dresden

Agenda (morning)

Time	Topic	Presenter
09:00	Introduction to VI-HPS & parallel performance engineering	Geimer/Shende
09:45	Setup for hands-on exercises with Live-ISO/OVA & Stampede2	all
10:00	Instrumentation & measurement of applications with ***Score-P***	Feld/Tschüter
10:30	Exploration & visualization of call-path profiles with ***CUBE***	Geimer
11:00	*Coffee break*	
11:30	Configuration & customization of ***Score-P*** measurements	Feld/Tschüter
12:00	Examination & visualization of profiles with ***TAU***	Shende
12:45	Specialized ***Score-P*** measurements and analyses	Feld
13:00	*Lunch break*	

Agenda (afternoon)

Time	Topic	Presenter
14:00	Automated analysis of traces for inefficiencies with ***Scalasca***	Geimer
14:45	Interactive visualization and time-interval statistics with ***Vampir***	Tschüter
15:30	Specialized ***Score-P*** measurements and analyses	Feld
16:00	*Coffee break*	
16:30	Performance data management with *TAU* ***PerfExplorer***	Shende
16:45	Parallel application performance analysis case studies	all
17:45	Review & conclusion	Geimer
18:00	*Adjourn*	

Virtual Institute – High Productivity Supercomputing

- **Goal**: Improve the quality and accelerate the development process of complex simulation codes running on highly-parallel computer systems
- Start-up funding (2006–2011)
  by Helmholtz Association of German Research Centres
- Activities
  - Development and integration of HPC programming tools
    - Correctness checking & performance analysis
  - Academic workshops
  - Training workshops
  - Service
    - Support email lists
    - Application engagement

HELMHOLTZ
RESEARCH FOR GRAND CHALLENGES

http://www.vi-hps.org

VI-HPS partners (founders)

Forschungszentrum Jülich
- Jülich Supercomputing Centre

RWTH Aachen University
- Centre for Computing & Communication

Technische Universität Dresden
- Centre for Information Services & HPC

University of Tennessee (Knoxville)
- Innovative Computing Laboratory

VI-HPS partners (cont.)

Allinea Software Ltd.

- Now part of ARM

Barcelona Supercomputing Center

- Centro Nacional de Supercomputación

Lawrence Livermore National Lab.

- Center for Applied Scientific Computing

Leibniz Supercomputing Centre

Technical University of Darmstadt

- Laboratory for Parallel Programming

VI-HPS partners (cont.)

Technical University of Munich
- Chair for Computer Architecture

TECHNISCHE UNIVERSITÄT MÜNCHEN

University of Oregon
- Performance Research Laboratory

UNIVERSITY OF OREGON

University of Stuttgart
- HPC Centre

Universität Stuttgart

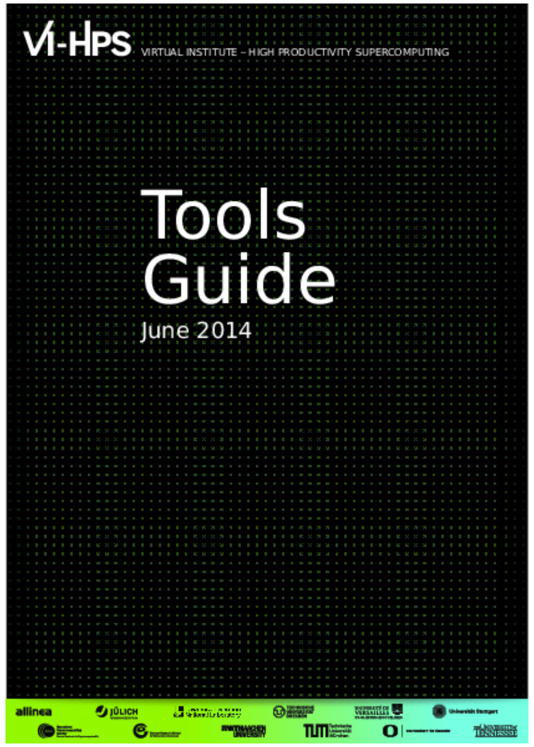
University of Versailles St-Quentin
- LRC ITACA

UNIVERSITÉ DE VERSAILLES ST-QUENTIN-EN-YVELINES

Productivity tools

- MUST / ARCHER
  - MPI & OpenMP usage correctness checking
- PAPI
  - Interfacing to hardware performance counters
- Periscope Tuning Framework
  - Automatic analysis and tuning
- **Scalasca**
  - Large-scale parallel performance analysis
- **TAU**
  - Integrated parallel performance system
- **Vampir**
  - Interactive graphical trace visualization & analysis
- **Score-P**
  - Community-developed instrumentation & measurement infrastructure

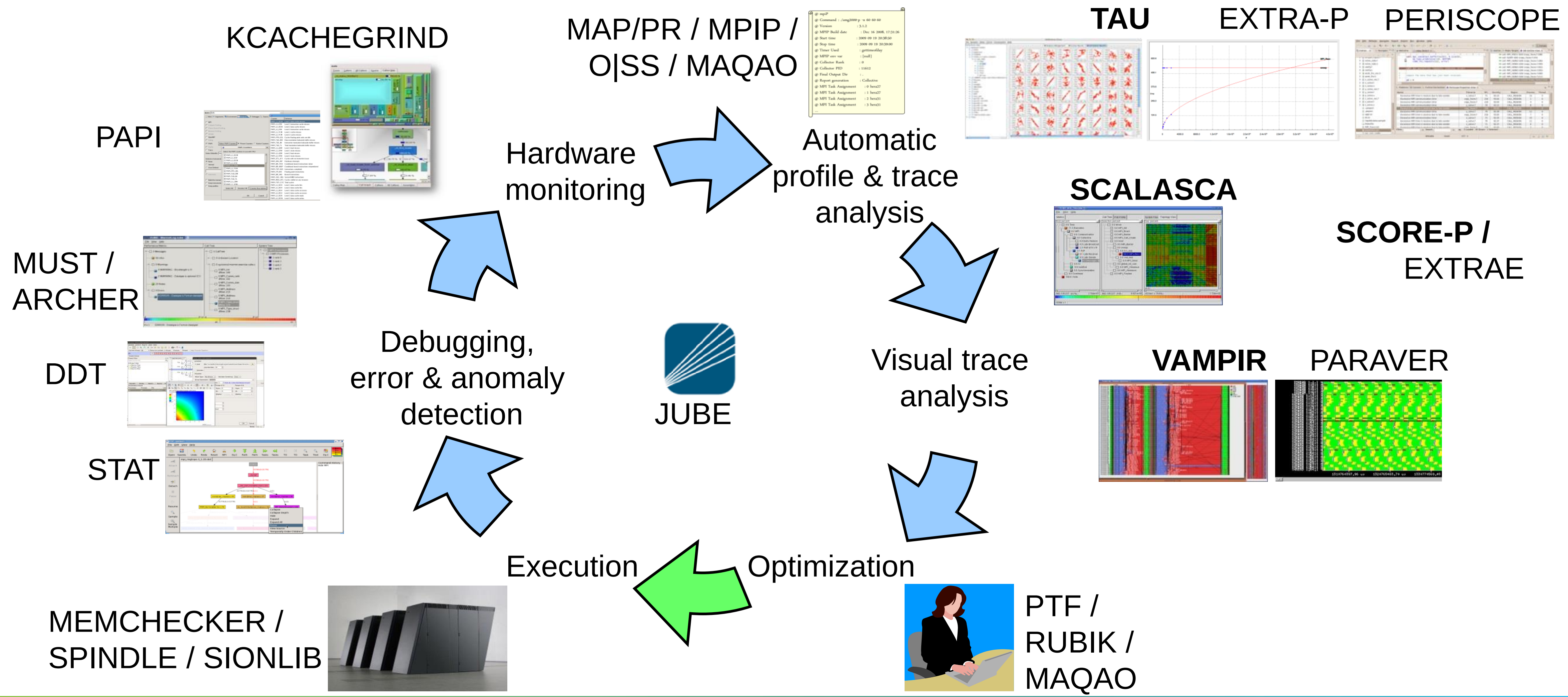
For a brief overview of tools consult the VI-HPS Tools Guide:



Productivity tools (cont.)

- DDT/MAP/PR: Parallel debugging, profiling & performance reports
- Extra-P: Automated performance modelling
- JUBE: Automatic workflow execution for benchmarking, testing & production
- Kcachegrind: Callgraph-based cache analysis [x86 only]
- MAQAO: Assembly instrumentation & optimization [x86-64 only]
- mpiP/mpiPview: MPI profiling tool and analysis viewer
- Open MPI: Integrated memory checking
- Open|SpeedShop: Integrated parallel performance analysis environment
- Paraver/Dimemas/Extrae: Event tracing and graphical trace visualization & analysis
- Rubik: Process mapping generation & optimization [BG only]
- SIONlib/Spindle: Optimized native parallel file I/O & shared library loading
- STAT: Stack trace analysis tools

Technologies and their integration

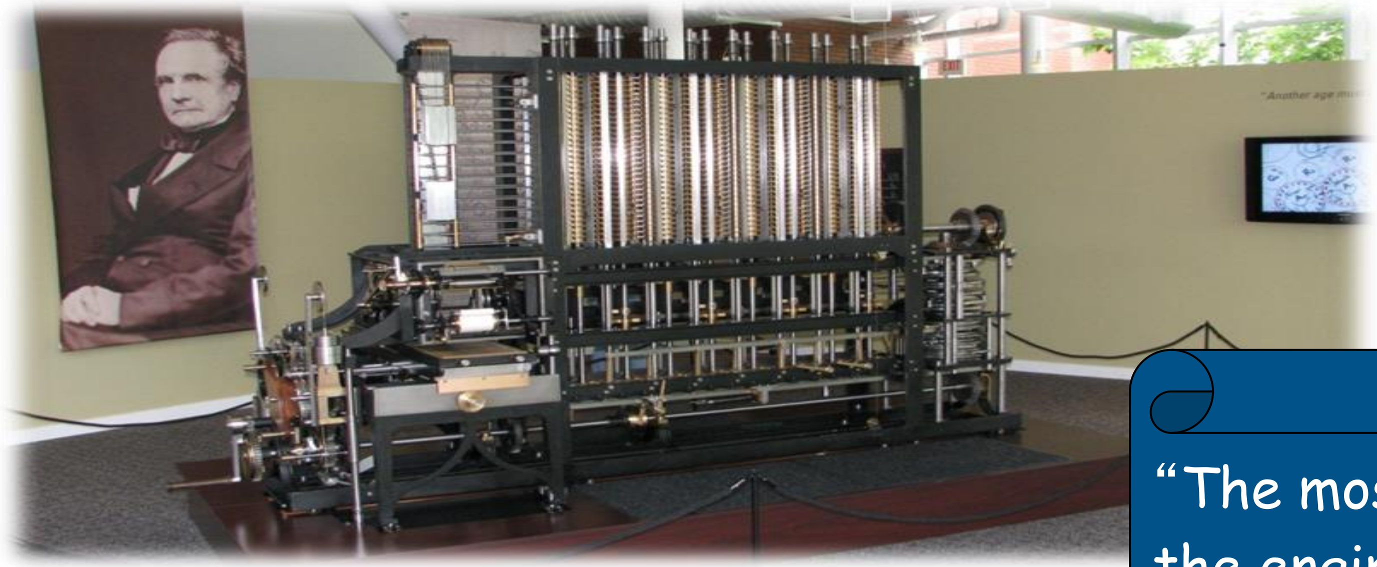


Introduction to Parallel Performance Engineering

Sameer Shende
University of Oregon

(with content used with permission from tutorials by Bernd Mohr/JSC and Luiz DeRose/Cray)

Performance: an old problem


Difference Engine

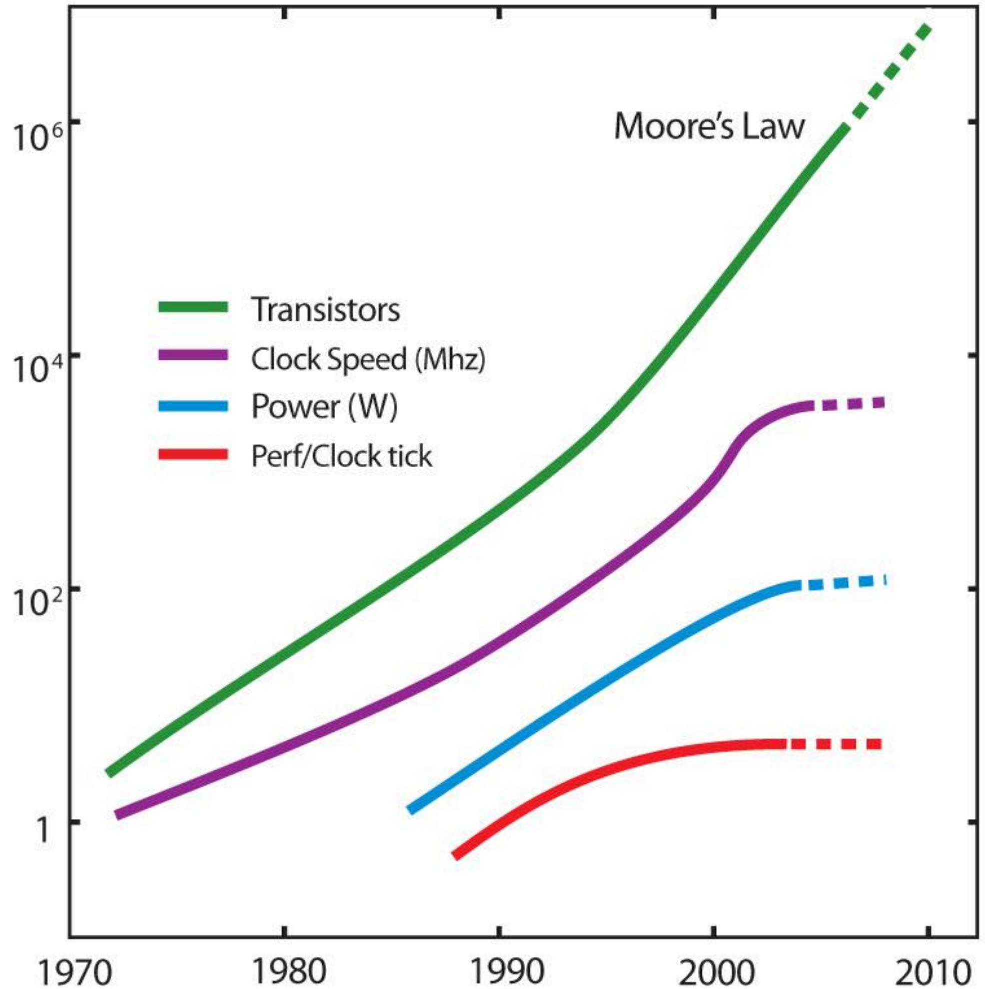
"The most constant difficulty in contriving the engine has arisen from the desire to reduce the time in which the calculations were executed to the shortest which is possible."

Charles Babbage
1791 – 1871

Today: the "free lunch" is over

- Moore's law is still in charge, but
  - Clock rates no longer increase
  - Performance gains only through increased parallelism
- Optimizations of applications more difficult
  - Increasing application complexity
    - Multi-physics
    - Multi-scale
  - Increasing machine complexity
    - Hierarchical networks / memory
    - More CPUs / multi-core

☞ Every doubling of scale reveals a new bottleneck!



Performance factors of parallel applications

- "Sequential" performance factors
  - Computation
    - ☞ Choose right algorithm, use optimizing compiler
  - Cache and memory
    - ☞ Tough! Only limited tool support, hope compiler gets it right
  - Input / output
    - ☞ Often not given enough attention

- "Parallel" performance factors
  - Partitioning / decomposition
  - Communication (i.e., message passing)
  - Multithreading
  - Synchronization / locking
    - ☞ More or less understood, good tool support

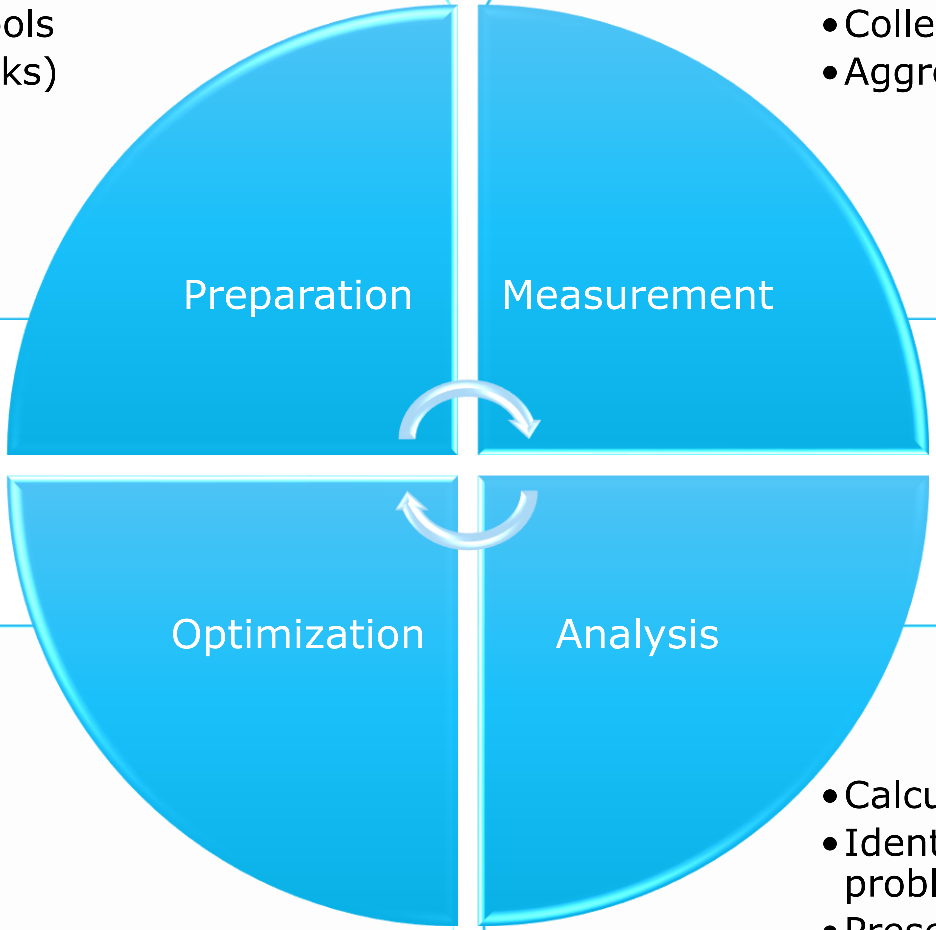
Tuning basics

- Successful engineering is a combination of
  - Careful setting of various tuning parameters
  - The right algorithms and libraries
  - Compiler flags and directives
  - …
  - Thinking !!!
- Measurement is better than guessing
  - To determine performance bottlenecks
  - To compare alternatives
  - To validate tuning decisions and optimizations
    - ☞ After each step!

Performance engineering workflow

- Prepare application with symbols
- Insert extra code (probes/hooks)

- Collection of performance data
- Aggregation of performance data



- Modifications intended to eliminate/reduce performance problem

- Calculation of metrics
- Identification of performance problems
- Presentation of results

The 80/20 rule

- Programs typically spend 80% of their time in 20% of the code
- Programmers typically spend 20% of their effort to get 80% of the total speedup possible for the application
  - ☞ *Know when to stop!*
- Don't optimize what does not matter
  - ☞ *Make the common case fast!*

"If you optimize everything, you will always be unhappy."

Donald E. Knuth

Metrics of performance

- What can be measured?
  - A **count** of how often an event occurs
    - E.g., the number of MPI point-to-point messages sent
  - The **duration** of some interval
    - E.g., the time spent these send calls
  - The **size** of some parameter
    - E.g., the number of bytes transmitted by these calls

- Derived metrics
  - E.g., rates / throughput
  - Needed for normalization

Example metrics

- Execution time
- Number of function calls
- CPI
  - CPU cycles per instruction
- FLOPS
  - Floating-point operations executed per second

"math" Operations?
HW Operations?
HW Instructions?
32-/64-bit? …

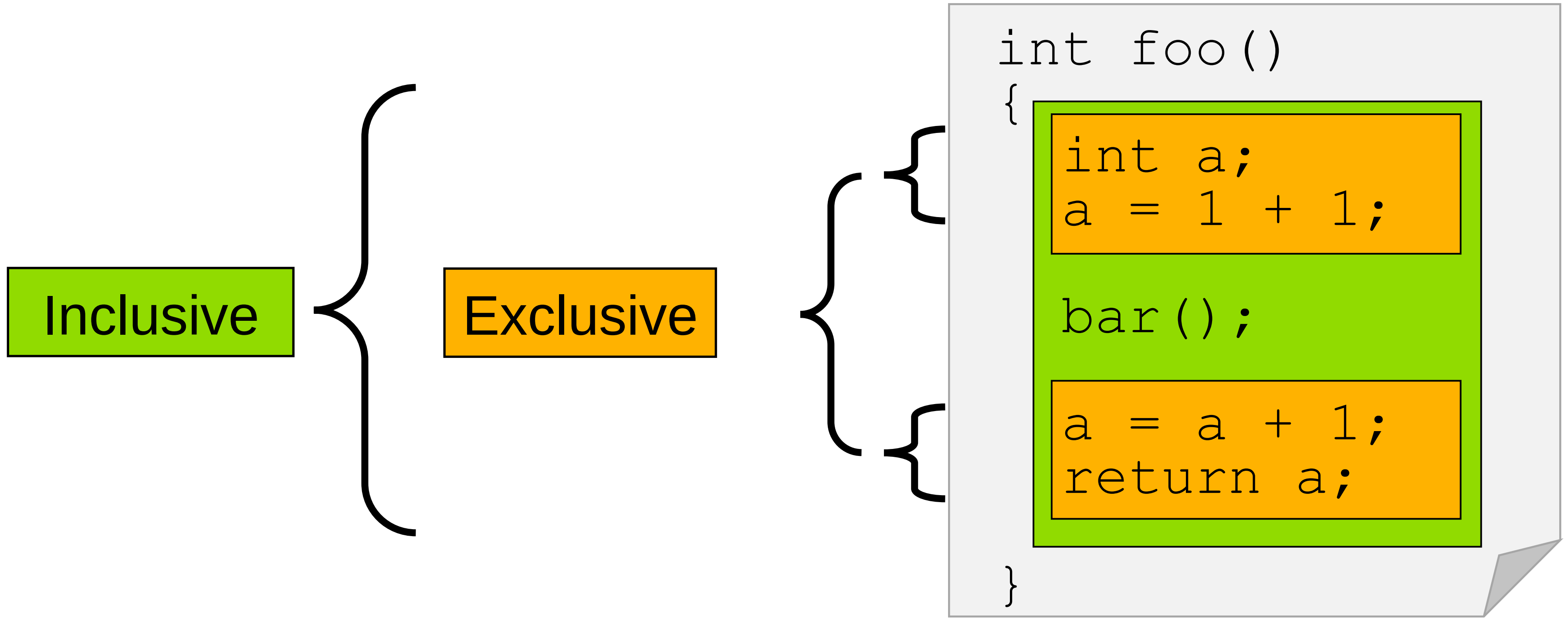
Execution time

- Wall-clock time
  - Includes waiting time: I/O, memory, other system activities
  - In time-sharing environments also the time consumed by other applications
- CPU time
  - Time spent by the CPU to execute the application
  - Does not include time the program was context-switched out
    - Problem: Does not include inherent waiting time (e.g., I/O)
    - Problem: Portability? What is user, what is system time?

- Problem: Execution time is non-deterministic
  - Use mean or minimum of several runs

Inclusive vs. Exclusive values

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



```
int foo()
{
  int a;
  a = 1 + 1;

  bar();

  a = a + 1;
  return a;
}
```

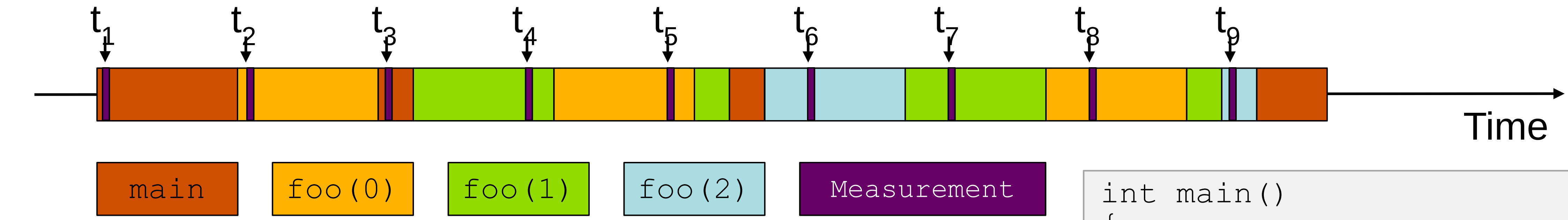
Classification of measurement techniques

- **How are performance measurements triggered?**
  - **Sampling**
  - **Code instrumentation**

- How is performance data recorded?
  - Profiling / Runtime summarization
  - Tracing

- How is performance data analyzed?
  - Online
  - Post mortem

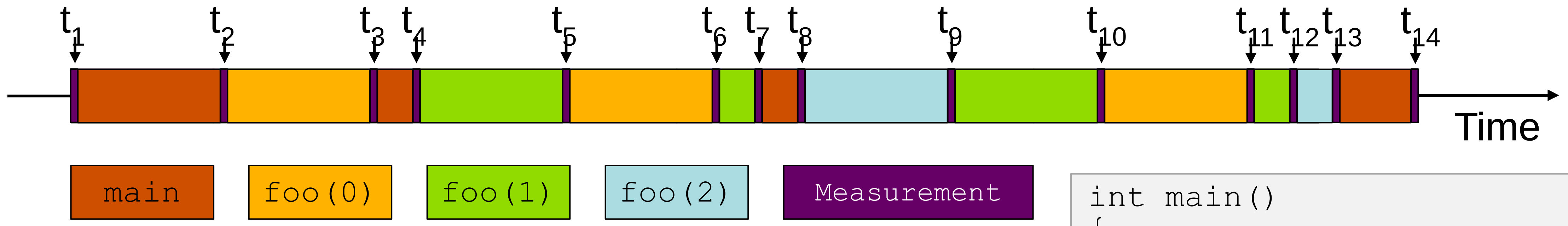
Sampling



- Running program is periodically interrupted to take measurement
  - Timer interrupt, OS signal, or HWC overflow
  - Service routine examines return-address stack
  - Addresses are mapped to routines using symbol table information
- Statistical inference of program behavior
  - Not very detailed information on highly volatile metrics
  - Requires long-running applications
- Works with unmodified executables

```
int main()
{
  int i;

  for (i=0; i < 3; i++)
    foo(i);

  return 0;
}

void foo(int i)
{

  if (i > 0)
    foo(i – 1);

}
```

Instrumentation



- Measurement code is inserted such that every event of interest is captured directly
  - Can be done in various ways
- Advantage:
  - Much more detailed information
- Disadvantage:
  - Processing of source-code / executable necessary
  - Large relative overheads for small functions

```
int main()
{
  int i;
  Enter(“main”);
  for (i=0; i < 3; i++)
    foo(i);
  Leave(“main”);
  return 0;
}

void foo(int i)
{
  Enter(“foo”);
  if (i > 0)
    foo(i – 1);
  Leave(“foo”);
}
```

Instrumentation techniques

- Static instrumentation
  - Program is instrumented prior to execution
- Dynamic instrumentation
  - Program is instrumented at runtime

- Code is inserted
  - Manually
  - Automatically
    - By a preprocessor / source-to-source translation tool
    - By a compiler
    - By linking against a pre-instrumented library / runtime system
    - By binary-rewrite / dynamic instrumentation tool

Critical issues

- Accuracy
  - Intrusion overhead
    - Measurement itself needs time and thus lowers performance
  - Perturbation
    - Measurement alters program behaviour
    - E.g., memory access pattern
  - Accuracy of timers & counters
- Granularity
  - How many measurements?
  - How much information / processing during each measurement?

☞ *Tradeoff: Accuracy vs. Expressiveness of data*

Classification of measurement techniques

- How are performance measurements triggered?
  - Sampling
  - Code instrumentation
- **How is performance data recorded?**
  - **Profiling / Runtime summarization**
  - **Tracing**
- How is performance data analyzed?
  - Online
  - Post mortem

Profiling / Runtime summarization

- Recording of aggregated information
  - Total, maximum, minimum, …
- For measurements
  - Time
  - Counts
    - Function calls
    - Bytes transferred
    - Hardware counters
- Over program and system entities
  - Functions, call sites, basic blocks, loops, …
  - Processes, threads

☞ *Profile = summarization of events over execution interval*

Types of profiles

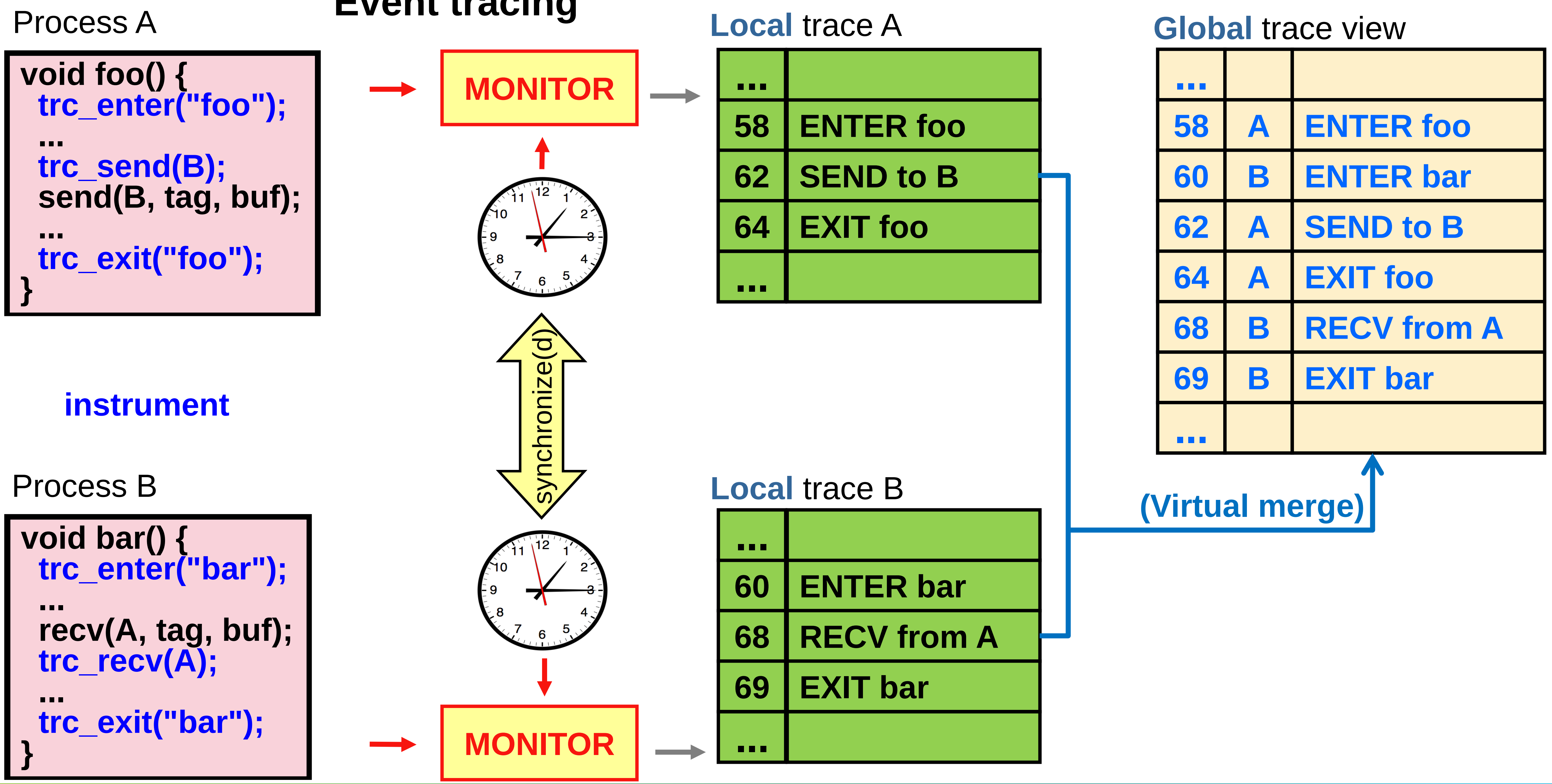
- Flat profile
    - Shows distribution of metrics per routine / instrumented region
    - Calling context is not taken into account
- Call-path profile
    - Shows distribution of metrics per executed call path
    - Sometimes only distinguished by partial calling context (e.g., two levels)
- Special-purpose profiles
    - Focus on specific aspects, e.g., MPI calls or OpenMP constructs
    - Comparing processes/threads

Tracing

- Recording detailed information about significant points (events) during execution of the program
  - Enter / leave of a region (function, loop, …)
  - Send / receive a message, …
- Save information in event record
  - Timestamp, location, event type
  - Plus event-specific information (e.g., communicator, sender / receiver, …)
- Abstract execution model on level of defined events

☞ *Event trace = Chronologically ordered sequence of event records*

Event tracing



Process A

```
void foo() {
  trc_enter("foo");
  ...
  trc_send(B);
  send(B, tag, buf);
  ...
  trc_exit("foo");
}
```

Process B

```
void bar() {
  trc_enter("bar");
  ...
  recv(A, tag, buf);
  trc_recv(A);
  ...
  trc_exit("bar");
}
```

Local trace A

...	
58	ENTER foo
62	SEND to B
64	EXIT foo
...	

Local trace B

...	
60	ENTER bar
68	RECV from A
69	EXIT bar
...	

Global trace view

...		
58	A	ENTER foo
60	B	ENTER bar
62	A	SEND to B
64	A	EXIT foo
68	B	RECV from A
69	B	EXIT bar
...		

Tracing Pros & Cons

- Tracing advantages
  - Event traces preserve the **temporal** and **spatial** relationships among individual events (☞ context)
  - Allows reconstruction of **dynamic** application behaviour on any required level of abstraction
  - Most general measurement technique
    - Profile data can be reconstructed from event traces
- Disadvantages
  - Traces can very quickly become extremely large
  - Writing events to file at runtime may causes perturbation

Classification of measurement techniques

- How are performance measurements triggered?
  - Sampling
  - Code instrumentation
- How is performance data recorded?
  - Profiling / Runtime summarization
  - Tracing
- **How is performance data analyzed?**
  - **Online**
  - **Post mortem**

Online analysis

- Performance data is processed during measurement run
  - Process-local profile aggregation
  - Requires formalized knowledge about performance bottlenecks
  - More sophisticated inter-process analysis using
    - “Piggyback” messages
    - Hierarchical network of analysis agents
- Online analysis often involves application steering to interrupt and re-configure the measurement

Post-mortem analysis

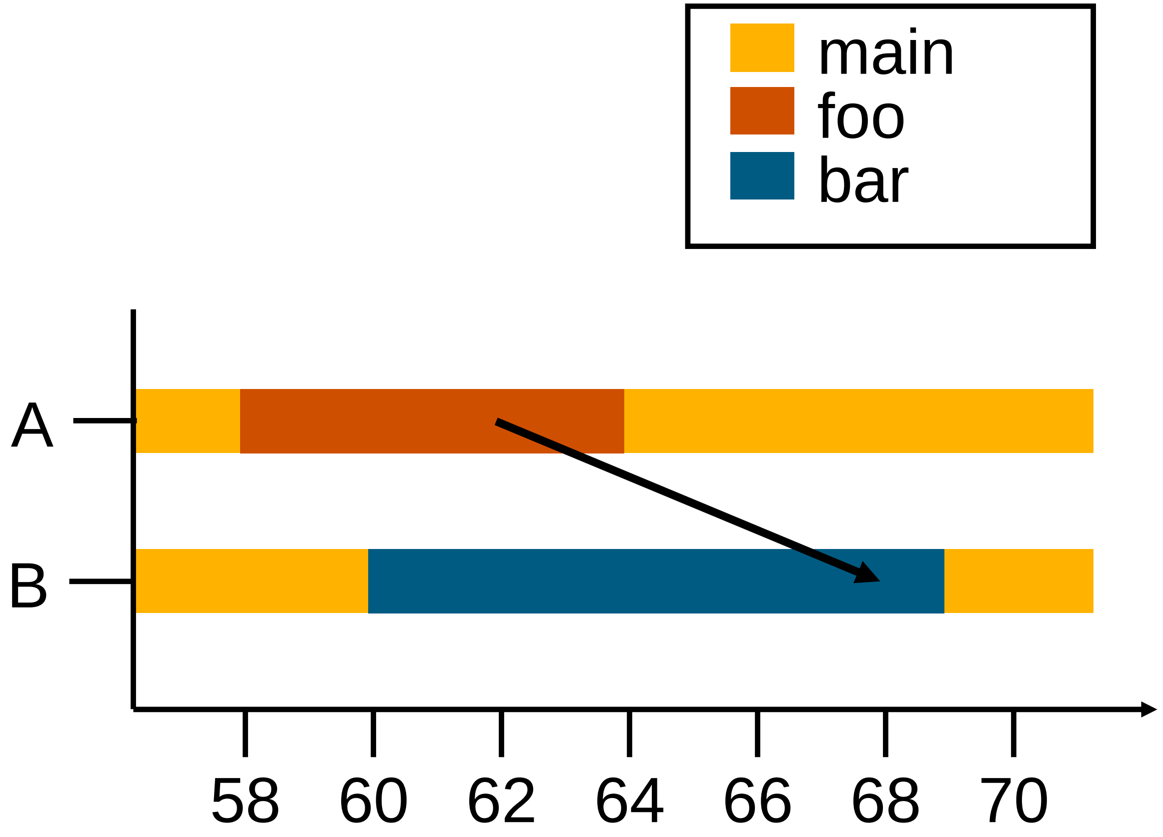
- Performance data is stored at end of measurement run
- Data analysis is performed afterwards
  - Automatic search for bottlenecks
  - Visual trace analysis
  - Calculation of statistics

Example: Time-line visualization

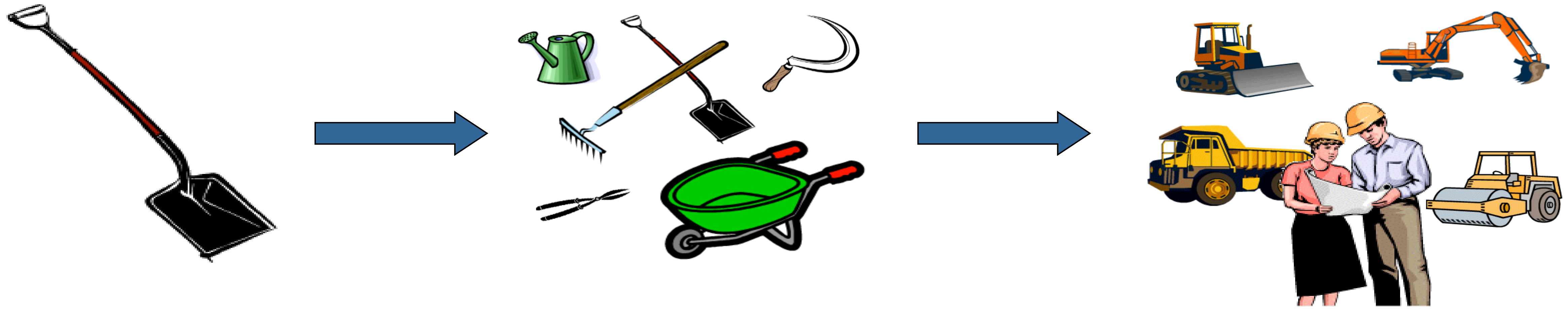
Global trace view

...		
58	A	ENTER foo
60	B	ENTER bar
62	A	SEND to B
64	A	EXIT foo
68	B	RECV from A
69	B	EXIT bar
...		

Post-Mortem
$\longrightarrow$
Analysis



No single solution is sufficient!



A combination of different methods, tools and techniques is typically needed!

- Analysis
  - Statistics, visualization, automatic analysis, data mining, ...
- Measurement
  - Sampling / instrumentation, profiling / tracing, ...
- Instrumentation
  - Source code / binary, manual / automatic, ...

Typical performance analysis procedure

- Do I have a performance problem at all?
  - Time / speedup / scalability measurements
- What is the key bottleneck (computation / communication)?
  - MPI / OpenMP / flat profiling
- Where is the key bottleneck?
  - Call-path profiling, detailed basic block profiling
- Why is it there?
  - Hardware counter analysis, trace selected parts to keep trace size manageable
- Does the code have scalability problems?
  - Load imbalance analysis, compare profiles at various sizes function-by-function

Hands-on: NPB-MZ-MPI / BT

VI-HPS Team

Tutorial exercise objectives

- Familiarize with usage of VI-HPS tools
  - Complementary tools' capabilities & interoperability
- Prepare to apply tools productively to *your* application(s)
- Exercise is based on a small portable benchmark code
  - Unlikely to have significant optimization opportunities

- Optional (recommended) exercise extensions
  - Analyze performance of alternative configurations
  - Investigate effectiveness of system-specific compiler/MPI optimizations and/or placement/binding/affinity capabilities
  - Investigate scalability and analyze scalability limiters
  - Compare performance on different HPC platforms
  - …

Access to Stampede2

```
# Connect to a Stampede2 login node
% ssh -Y userid@stampede2.tacc.utexas.edu
```

- Logging in to Stampede2

```
$HOME
$WORK
$SCRATCH

/home1/03529/tg828282/Tutorial
(shortcut: ~tg828282/Tutorial)
```

Tutorial materials

- File systems & directories
  - Use $SCRATCH for the tutorial
    - Fast Lustre file system, ~30 PB
    - No backup
    - Files may be automatically purged 10 days after last modification

- More extensive documentation:
  - https://portal.tacc.utexas.edu/user-guides/stampede2

Compiling & job submission

- Development environment: Intel compiler with Intel MPI
  - Use Intel's MPI compiler wrappers
    - `mpiicc`
    - `mpiicpc`
    - `mpiifort`

- Stampede2 uses the SLURM batch system
  - Jobs submitted from tutorial accounts with provided job scripts will automatically be run in a reservation

```
% sbatch jobscript.sbatch      ← Submit job
% squeue -u $USER              ← View job queue
% scancel <jobid>              ← Cancel job
```

Local installation

- VI-HPS tools not yet installed system-wide
  - Source provided shell code snippet to add local tool installations to $PATH
  - Required for each shell session

```
% source ~tg828282/Tutorial/vihps.sh
```

- Copy tutorial sources to your working directory, ideally on a parallel file system (recommended: $SCRATCH)

```
% cd $SCRATCH
% tar zxvf ~tg828282/Tutorial/NPB3.3-MZ-MPI.tar.gz
% cd NPB3.3-MZ-MPI
```

NPB-MZ-MPI suite

- The NAS Parallel Benchmark suite (MPI+OpenMP version)
  - Available from http://www.nas.nasa.gov/Software/NPB
  - 3 benchmarks in Fortran77
  - Configurable for various sizes & classes
- Move into the NPB3.3-MZ-MPI root directory

```
% ls
bin/     common/  jobscript/  Makefile  README.install   SP-MZ/
BT-MZ/   config/  LU-MZ/      README    README.tutorial  sys/
```

- Subdirectories contain source code for each benchmark
  - Plus additional configuration and common code
- The provided distribution has already been configured for the tutorial, such that it is ready to "make" one or more of the benchmarks and install them into a (tool-specific) "bin" subdirectory

Building an NPB-MZ-MPI benchmark

```
% make
  ===========================================
  =      NAS PARALLEL BENCHMARKS 3.3        =
  =      MPI+OpenMP Multi-Zone Versions     =
  =      F77                                =
  ===========================================

  To make a NAS multi-zone benchmark type

         make <benchmark-name> CLASS=<class> NPROCS=<nprocs>

  where <benchmark-name> is "bt-mz", "lu-mz", or "sp-mz"
        <class>          is "S", "W", "A" through "F"
        <nprocs>         is number of processes

  [...]

 ****************************************************************
 * Custom build configuration is specified in config/make.def   *
 * Suggested tutorial exercise configuration for Stampede2:     *
 *       make bt-mz CLASS=C NPROCS=32                           *
 ****************************************************************
```

- Type "make" for instructions

Building an NPB-MZ-MPI benchmark

```
% make bt-mz CLASS=C NPROCS=32
make[1]: Entering directory `BT-MZ'
make[2]: Entering directory `sys'
icc  -o setparams setparams.c -lm
make[2]: Leaving directory `sys'
../sys/setparams bt-mz 32 C
make[2]: Entering directory `../BT-MZ'
mpiifort -c  -g -O3 -qopenmp       bt.f
                                      […]
mpiifort -c  -g -O3 -qopenmp       mpi_setup.f
cd ../common;  mpiifort -c  -g -O3 -qopenmp        print_results.f
cd ../common;  mpiifort -c  -g -O3 -qopenmp        timers.f
mpiifort -g -O3 -qopenmp   -o ../bin/bt-mz_C.32 bt.o
 initialize.o exact_solution.o exact_rhs.o set_constants.o adi.o
 rhs.o zone_setup.o x_solve.o y_solve.o  exch_qbc.o solve_subs.o
 z_solve.o add.o error.o verify.o mpi_setup.o ../common/print_results.o
 ../common/timers.o
make[2]: Leaving directory `BT-MZ'
Built executable ../bin/bt-mz_C.32
make[1]: Leaving directory `BT-MZ'
```

- Specify the benchmark configuration
  - benchmark name: **bt-mz**, lu-mz, sp-mz
  - the number of MPI processes: NPROCS=**32**
  - the benchmark class (S, W, A, B, C, D, E): CLASS=**C**

Shortcut: `% make suite`

NPB-MZ-MPI / BT (Block Tridiagonal Solver)

- What does it do?
  - Solves a discretized version of the unsteady, compressible Navier-Stokes equations in three spatial dimensions
  - Performs 200 time-steps on a regular 3-dimensional grid
- Implemented in 20 or so Fortran77 source modules

- Uses MPI & OpenMP in combination
  - Proposed hands-on setup on Stampede2:
    - 2 compute nodes with 1 Intel Xeon Phi 7250 CPU (Knights Landing, KNL) each
    - 32 MPI processes with 4 OpenMP threads each
  - bt-mz_C.32 should run in less than 30 seconds

NPB-MZ-MPI / BT reference execution

```
% cd bin
% cp ../jobscript/stampede2/reference.sbatch .
% less reference.sbatch
% sbatch reference.sbatch
% less mzmpibt.o<job_id>
 NAS Parallel Benchmarks (NPB3.3-MZ-MPI) - BT-MZ MPI+OpenMP Benchmark
 Number of zones:  16 x  16
 Iterations:  200    dt:   0.000100
 Number of active processes:     32
 Total number of threads:      128  (  4.0 threads/process)

 Time step    1
 Time step   20
  [...]
 Time step  180
 Time step  200
 Verification Successful

 BT-MZ Benchmark Completed.
 Time in seconds = 22.34
```

- Copy jobscript and launch as a hybrid MPI+OpenMP application

Hint: save the benchmark output (or note the run time) to be able to refer to it later

Tutorial exercise steps

- Edit config/make.def to adjust build configuration
  - Modify specification of compiler/linker: MPIF77
  - See next slide for details
- Make clean and build new tool-specific executable

```
% make clean
% make bt-mz CLASS=C NPROCS=32
Built executable ../bin.$(TOOL)/bt-mz_C.32
```

- Change to the directory containing the new executable before running it with the desired tool configuration

```
% cd bin.$(TOOL)
% cp ../jobscript/stampede2/$(TOOL).sbatch .
% sbatch $(TOOL).sbatch
```

NPB-MZ-MPI / BT: config/make.def

```
#                 SITE- AND/OR PLATFORM-SPECIFIC DEFINITIONS.
#
#---------------------------------------------------------------------------

#---------------------------------------------------------------------------
# Configured for generic MPI with INTEL compiler
#---------------------------------------------------------------------------
#OPENMP  = -fopenmp      # GCC compiler
OPENMP = -qopenmp        # Intel compiler

...
#---------------------------------------------------------------------
# The Fortran compiler used for MPI programs
#---------------------------------------------------------------------
MPIF77 = mpiifort

# Alternative variant to perform instrumentation
#MPIF77 = scorep --user mpiifort

# PREP is a generic preposition macro for instrumentation preparation
#MPIF77 = $(PREP) mpiifort
...
```

Default (no instrumentation)

Hint: uncomment a compiler wrapper to do instrumentation

Score-P – A Joint Performance Measurement Run-Time Infrastructure for Periscope, Scalasca, TAU, and Vampir

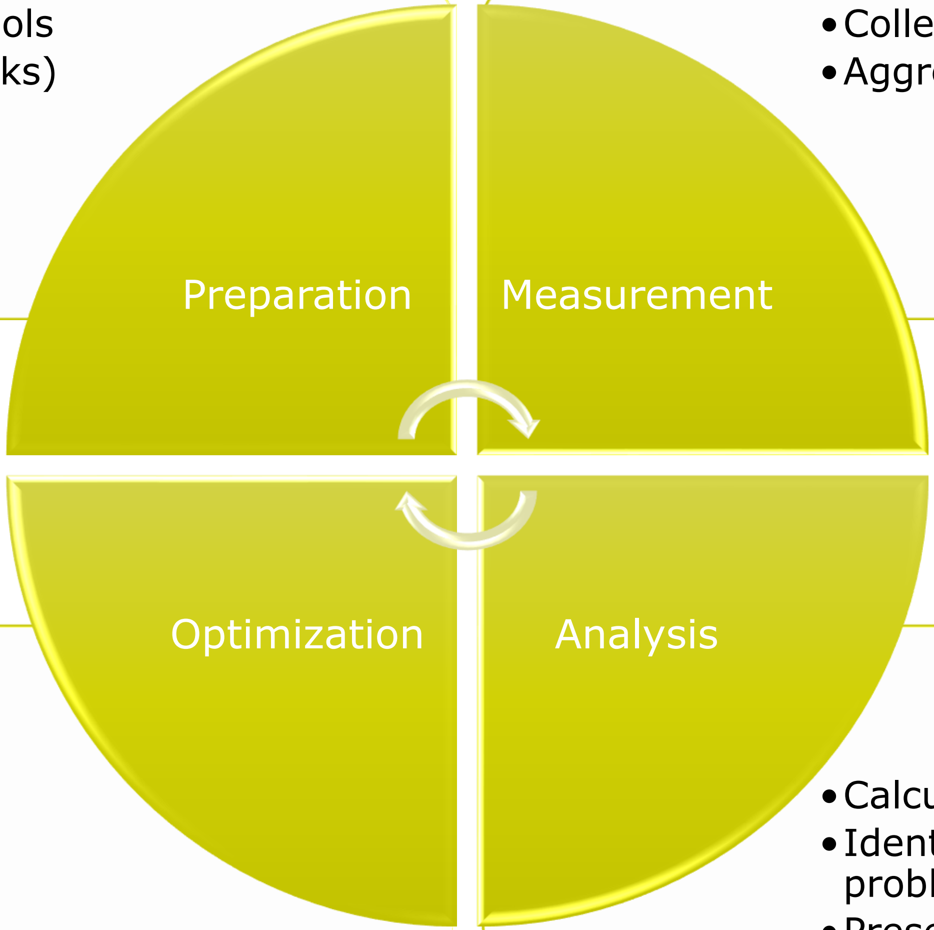
VI-HPS Team

Score-P
Scalable performance measurement infrastructure for parallel codes

Performance engineering workflow

- Prepare application with symbols
- Insert extra code (probes/hooks)

- Collection of performance data
- Aggregation of performance data

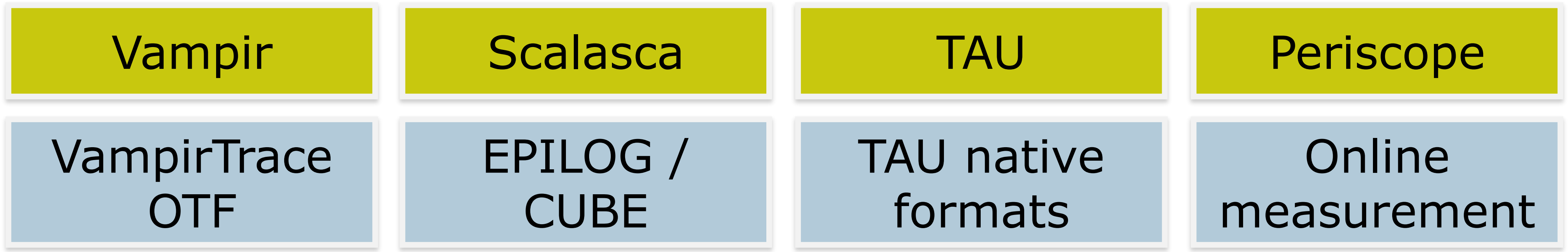


- Modifications intended to eliminate/reduce performance problem

- Calculation of metrics
- Identification of performance problems
- Presentation of results

Fragmentation of tools landscape

- Several performance tools co-exist
  - Separate measurement systems and output formats
- Complementary features and overlapping functionality
- Redundant effort for development and maintenance
  - Limited or expensive interoperability
- Complications for user experience, support, training



Score-P project idea

- Start a community effort for a common infrastructure
  - Score-P instrumentation and measurement system
  - Common data formats OTF2 and CUBE4
- Developer perspective:
  - Save manpower by sharing development resources
  - Invest in new analysis functionality and scalability
  - Save efforts for maintenance, testing, porting, support, training
- User perspective:
  - Single learning curve
  - Single installation, fewer version updates
  - Interoperability and data exchange
- Project funded by BMBF
- Close collaboration PRIMA project funded by DOE

GEFÖRDERT VOM

Bundesministerium für Bildung und Forschung

Partners

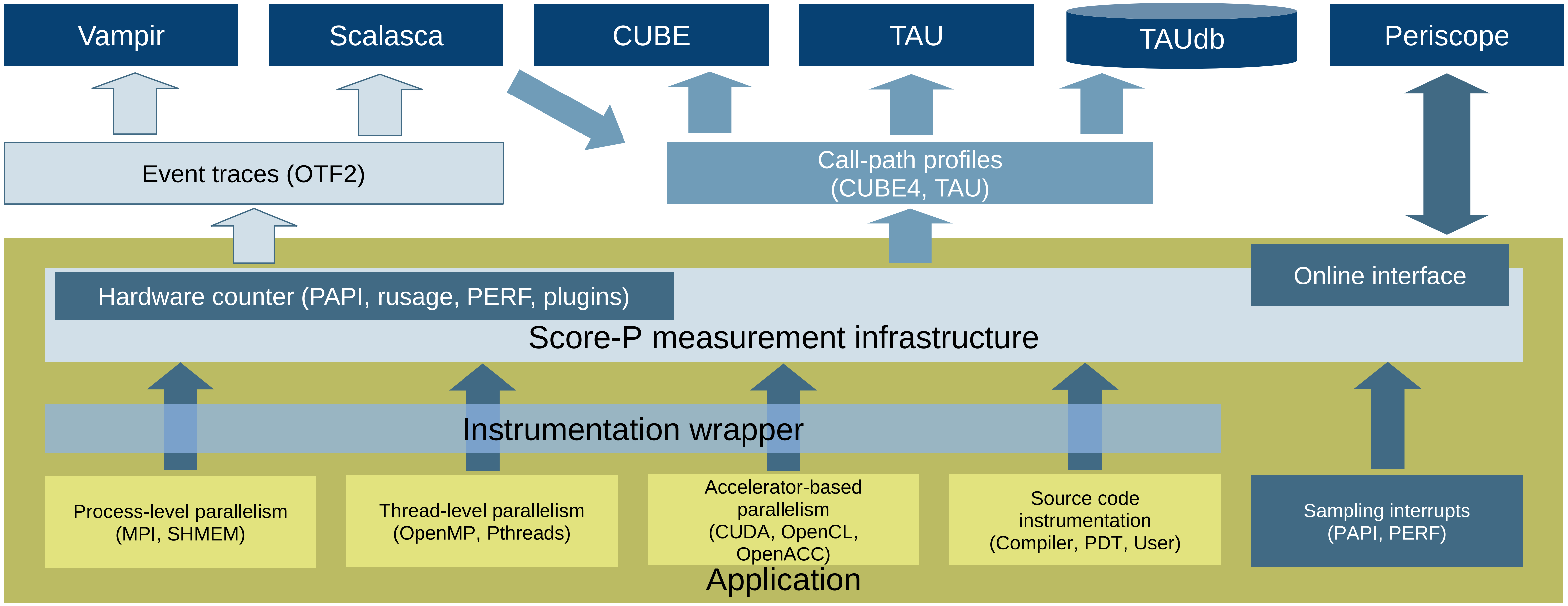
- Forschungszentrum Jülich, Germany
- Gesellschaft für numerische Simulation mbH Braunschweig, Germany
- RWTH Aachen, Germany
- Technische Universität Darmstadt, Germany
- Technische Universität Dresden, Germany
- Technische Universität München, Germany
- University of Oregon, Eugene, USA

Design goals

- Functional requirements
  - Generation of call-path profiles and event traces
  - Using direct instrumentation and sampling
  - Flexible measurement without re-compilation
  - Recording time, visits, communication data, hardware counters
  - Access and reconfiguration also at runtime
  - Support for MPI, SHMEM, OpenMP, Pthreads, CUDA, OpenCL, OpenACC and their valid combinations
  - Highly scalable I/O

- Non-functional requirements
  - Portability: all major HPC platforms
  - Scalability: petascale
  - Low measurement overhead
  - Robustness
  - Open Source: 3-clause BSD license

Score-P overview


Vampir
Scalasca
CUBE
TAU
TAUdb
Periscope
Event traces (OTF2)
Call-path profiles (CUBE4, TAU)
Hardware counter (PAPI, rusage, PERF, plugins)
Online interface
Score-P measurement infrastructure
Instrumentation wrapper
Process-level parallelism (MPI, SHMEM)
Thread-level parallelism (OpenMP, Pthreads)
Accelerator-based parallelism (CUDA, OpenCL, OpenACC)
Source code instrumentation (Compiler, PDT, User)
Sampling interrupts (PAPI, PERF)
Application

Future features and management

- Scalability to maximum available CPU core count
- Support for binary instrumentation
- Support for new programming models, e.g., PGAS
- Support for new architectures

- Ensure a single official release version at all times which will always work with the tools
- Allow experimental versions for new features or research

- Commitment to joint long-term cooperation
  - Development based on meritocratic governance model
  - Open for contributions and new partners

Hands-on:
NPB-MZ-MPI / BT

Performance analysis steps

- 0.0 Reference preparation for validation

- 1.0 Program instrumentation
- 1.1 Summary measurement collection
- 1.2 Summary analysis report examination

- 2.0 Summary experiment scoring
- 2.1 Summary measurement collection with filtering
- 2.2 Filtered summary analysis report examination

- 3.0 Event trace collection
- 3.1 Event trace examination & analysis

Recap: Local installation

- VI-HPS tools not yet installed system-wide
  - Source provided shell code snippet to add local tool installations to $PATH
  - Required for each shell session

```
% source ~tg828282/Tutorial/vihps.sh
```

- Copy tutorial sources to your working directory, ideally on a parallel file system (recommended: $SCRATCH)

```
% cd $SCRATCH
% tar zxvf ~tg828282/Tutorial/NPB3.3-MZ-MPI.tar.gz
% cd NPB3.3-MZ-MPI
```

NPB-MZ-MPI / BT instrumentation

```
#---------------------------------------------------------------
# The Fortran compiler used for MPI programs
#---------------------------------------------------------------
#MPIF77 = mpiifort

# Alternative variants to perform instrumentation
...
MPIF77 = scorep --user mpiifort

# This links MPI Fortran programs; usually the same as ${MPIF77}
FLINK   = $(MPIF77)
...
```

- Edit config/make.def to adjust build configuration
  - Modify specification of compiler/linker: MPIF77

Uncomment the Score-P compiler wrapper specification

NPB-MZ-MPI / BT instrumented build

```
% make clean

% make bt-mz CLASS=C NPROCS=32
cd BT-MZ; make CLASS=C NPROCS=32 VERSION=
make: Entering directory 'BT-MZ'
cd ../sys; icc  -o setparams setparams.c -lm
../sys/setparams bt-mz 32 C
scorep --user mpiifort -c  -g -O3 -qopenmp bt.f
 [...]
cd ../common;  scorep --user mpiifort -c  -g -O3 -qopenmp timers.f
 [...]
scorep --user mpiifort -g -O3 -qopenmp -o ../bin.scorep/bt-mz_C.32 \
bt.o initialize.o exact_solution.o exact_rhs.o set_constants.o \
adi.o rhs.o zone_setup.o x_solve.o y_solve.o exch_qbc.o \
solve_subs.o z_solve.o add.o error.o verify.o mpi_setup.o \
../common/print_results.o ../common/timers.o
Built executable ../bin.scorep/bt-mz_C.32
make: Leaving directory 'BT-MZ'
```

- Return to root directory and clean-up
- Re-build executable using Score-P compiler wrapper

Measurement configuration: scorep-info

```
% scorep-info config-vars --full
SCOREP_ENABLE_PROFILING
  Description: Enable profiling
 [...]
SCOREP_ENABLE_TRACING
  Description: Enable tracing
 [...]
SCOREP_TOTAL_MEMORY
  Description: Total memory in bytes for the measurement system
 [...]
SCOREP_EXPERIMENT_DIRECTORY
  Description: Name of the experiment directory
 [...]
SCOREP_FILTERING_FILE
  Description: A file name which contain the filter rules
 [...]
SCOREP_METRIC_PAPI
  Description: PAPI metric names to measure
 [...]
SCOREP_METRIC_RUSAGE
  Description: Resource usage metric names to measure
 [... More configuration variables ...]
```

- Score-P measurements are configured via environmental variables

Summary measurement collection

```
% cd bin.scorep
% cp ../jobscript/stampede2/scorep.sbatch .
% vim scorep.sbatch

# Score-P measurement configuration
export SCOREP_EXPERIMENT_DIRECTORY=scorep_bt-mz_sum
#export SCOREP_FILTERING_FILE=../config/scorep.filt
#export SCOREP_TOTAL_MEMORY=50M
#export SCOREP_METRIC_PAPI=PAPI_TOT_INS,PAPI_TOT_CYC
#export SCOREP_ENABLE_TRACING=true

# Run the application
ibrun ./bt-mz_${CLASS}.${PROCS}

% sbatch ./scorep.sbatch
```

- Change to the directory containing the new executable before running it with the desired configuration
- Check settings

Leave these lines commented out for the moment

- Submit job

Summary measurement collection

```
% less mzmpibt.o<job_id>

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

 Number of zones:  16 x  16
 Iterations: 200    dt:   0.000100
 Number of active processes:    32

 Use the default load factors with threads
 Total number of threads:    128  (  4.0 threads/process)

 Calculated speedup =    125.90

 Time step    1

 [... More application output ...]
```

- Check the output of the application run

BT-MZ summary analysis report examination

```
% ls
bt-mz_C.32  mzmpibt.e<job_id>  mzmpibt.o<job_id>
scorep_bt-mz_sum
% ls scorep_bt-mz_sum
profile.cubex  scorep.cfg







% cube scorep_bt-mz_sum/profile.cubex

       [CUBE GUI showing summary analysis report]
```

- Creates experiment directory including
  - A record of the measurement configuration (scorep.cfg)
  - The analysis report that was collated after measurement (profile.cubex)
- Interactive exploration with Cube

Hint:
Copy ‘profile.cubex’ to Live-DVD environment using ‘scp’ to improve responsiveness of GUI

Further information

- Community instrumentation & measurement infrastructure
  - Instrumentation (various methods)
  - Basic and advanced profile generation
  - Event trace recording
  - Online access to profiling data
- Available under 3-clause BSD open-source license
- Documentation & Sources:
  - http://www.score-p.org
- User guide also part of installation:
  - `<prefix>/share/doc/scorep/{pdf,html}/`
- Support and feedback: support@score-p.org
- Subscribe to news@score-p.org, to be up to date

VI-HPS

VIRTUAL INSTITUTE – HIGH PRODUCTIVITY SUPERCOMPUTING

Analysis report examination with Cube

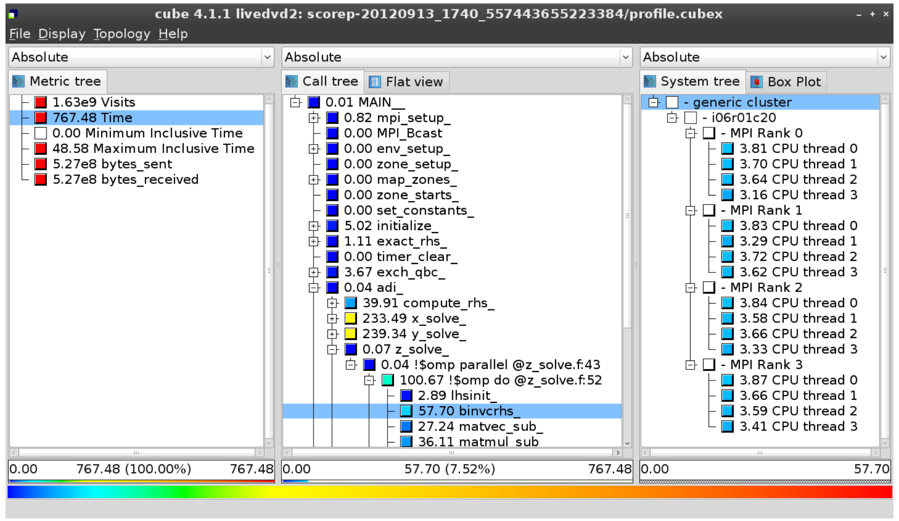
Markus Geimer
Jülich Supercomputing Centre

cube scalasca

allinea · JÜLICH Forschungszentrum · lrz Leibniz Supercomputing Centre · TECHNISCHE UNIVERSITÄT DARMSTADT · TUM Technische Universität München · UNIVERSITY OF OREGON · THE UNIVERSITY of TENNESSEE · BSC Barcelona Supercomputing Center Centro Nacional de Supercomputación · Lawrence Livermore National Laboratory · RWTH AACHEN UNIVERSITY · TECHNISCHE UNIVERSITÄT DRESDEN · UNIVERSITÉ DE VERSAILLES ST-QUENTIN-EN-YVELINES · Universität Stuttgart

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 ≥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 v4.4 (May 2018)


cube 4.1.1 livedvd2: scorep-20120913_1740_557443655223384/profile.cubex
File Display Topology Help
Absolute
Metric tree
Absolute
Call tree
Flat view
Absolute
System tree
Box Plot

```
1.63e9 Visits
767.48 Time
0.00 Minimum Inclusive Time
48.58 Maximum Inclusive Time
5.27e8 bytes_sent
5.27e8 bytes_received
```

```
0.01 MAIN__
  0.82 mpi_setup_
  0.00 MPI_Bcast
  0.00 env_setup_
  0.00 zone_setup_
  0.00 map_zones_
  0.00 zone_starts_
  0.00 set_constants_
  5.02 initialize_
  1.11 exact_rhs_
  0.00 timer_clear_
  3.67 exch_qbc_
  0.04 adi_
    39.91 compute_rhs_
    233.49 x_solve_
    239.34 y_solve_
    0.07 z_solve_
      0.04 !$omp parallel @z_solve.f:43
        100.67 !$omp do @z_solve.f:52
          2.89 lhsinit_
          57.70 binvcrhs_
          27.24 matvec_sub_
          36.11 matmul_sub
```

```
- generic cluster
  - i06r01c20
    - MPI Rank 0
      3.81 CPU thread 0
      3.70 CPU thread 1
      3.64 CPU thread 2
      3.16 CPU thread 3
    - MPI Rank 1
      3.83 CPU thread 0
      3.29 CPU thread 1
      3.72 CPU thread 2
      3.62 CPU thread 3
    - MPI Rank 2
      3.84 CPU thread 0
      3.58 CPU thread 1
      3.66 CPU thread 2
      3.33 CPU thread 3
    - MPI Rank 3
      3.87 CPU thread 0
      3.66 CPU thread 1
      3.59 CPU thread 2
      3.41 CPU thread 3
```

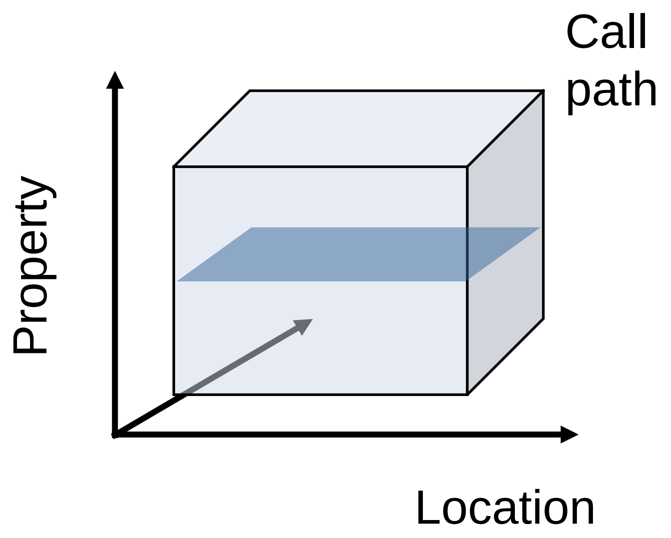
```
0.00   767.48 (100.00%)   767.48 | 0.00   57.70 (7.52%)   767.48 | 0.00   57.70
```

Analysis presentation and exploration

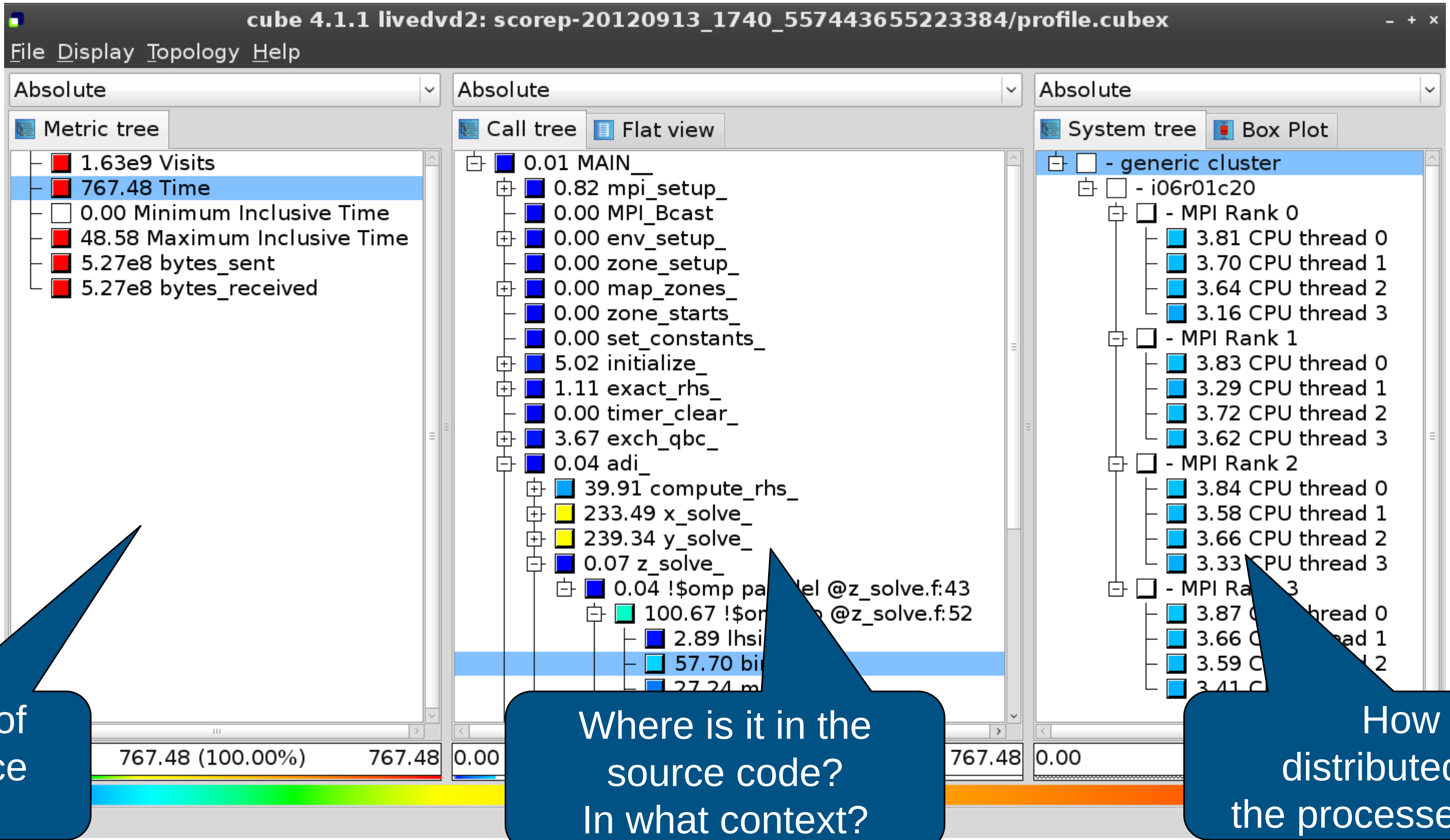
- Representation of values (severity matrix) on three hierarchical axes
  - Performance property (metric)
  - Call path (program location)
  - System location (process/thread)

- 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



Analysis presentation


cube 4.1.1 livedvd2: scorep-20120913_1740_557443655223384/profile.cubex
File Display Topology Help
Absolute
Absolute
Absolute
Metric tree
Call tree
Flat view
System tree
Box Plot

```
1.63e9 Visits
767.48 Time
0.00 Minimum Inclusive Time
48.58 Maximum Inclusive Time
5.27e8 bytes_sent
5.27e8 bytes_received
```

```
0.01 MAIN__
  0.82 mpi_setup_
  0.00 MPI_Bcast
  0.00 env_setup_
  0.00 zone_setup_
  0.00 map_zones_
  0.00 zone_starts_
  0.00 set_constants_
  5.02 initialize_
  1.11 exact_rhs_
  0.00 timer_clear_
  3.67 exch_qbc_
  0.04 adi_
    39.91 compute_rhs_
    233.49 x_solve_
    239.34 y_solve_
    0.07 z_solve_
      0.04 !$omp pa    lel @z_solve.f:43
        100.67 !$on      @z_solve.f:52
          2.89 lhsi
          57.70 bi
          27.24 m
```

```
- generic cluster
  - i06r01c20
    - MPI Rank 0
      3.81 CPU thread 0
      3.70 CPU thread 1
      3.64 CPU thread 2
      3.16 CPU thread 3
    - MPI Rank 1
      3.83 CPU thread 0
      3.29 CPU thread 1
      3.72 CPU thread 2
      3.62 CPU thread 3
    - MPI Rank 2
      3.84 CPU thread 0
      3.58 CPU thread 1
      3.66 CPU thread 2
      3.33 CPU thread 3
    - MPI Ra   3
      3.87 C    hread 0
      3.66 C      ad 1
      3.59 C       2
      3.41 C
```

767.48 (100.00%) 767.48 | 0.00 | 767.48 | 0.00

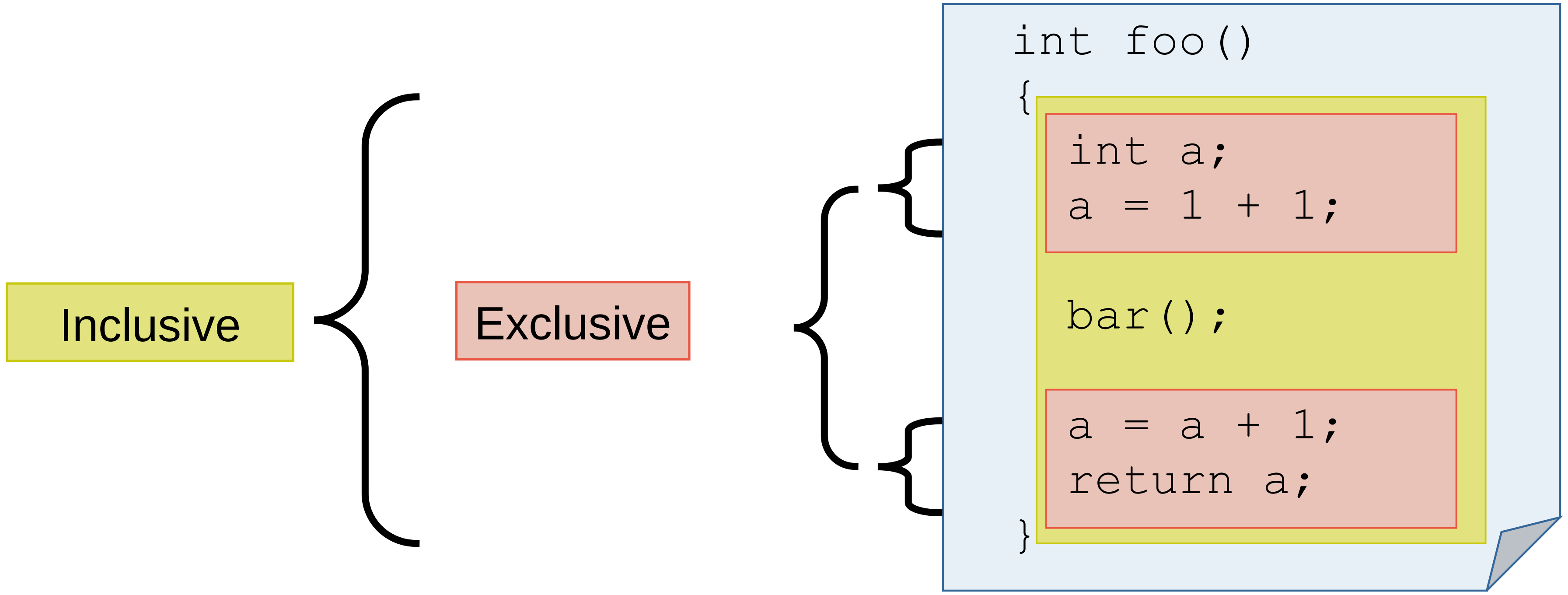
What kind of performance metric?

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

How is it distributed across the processes/threads?

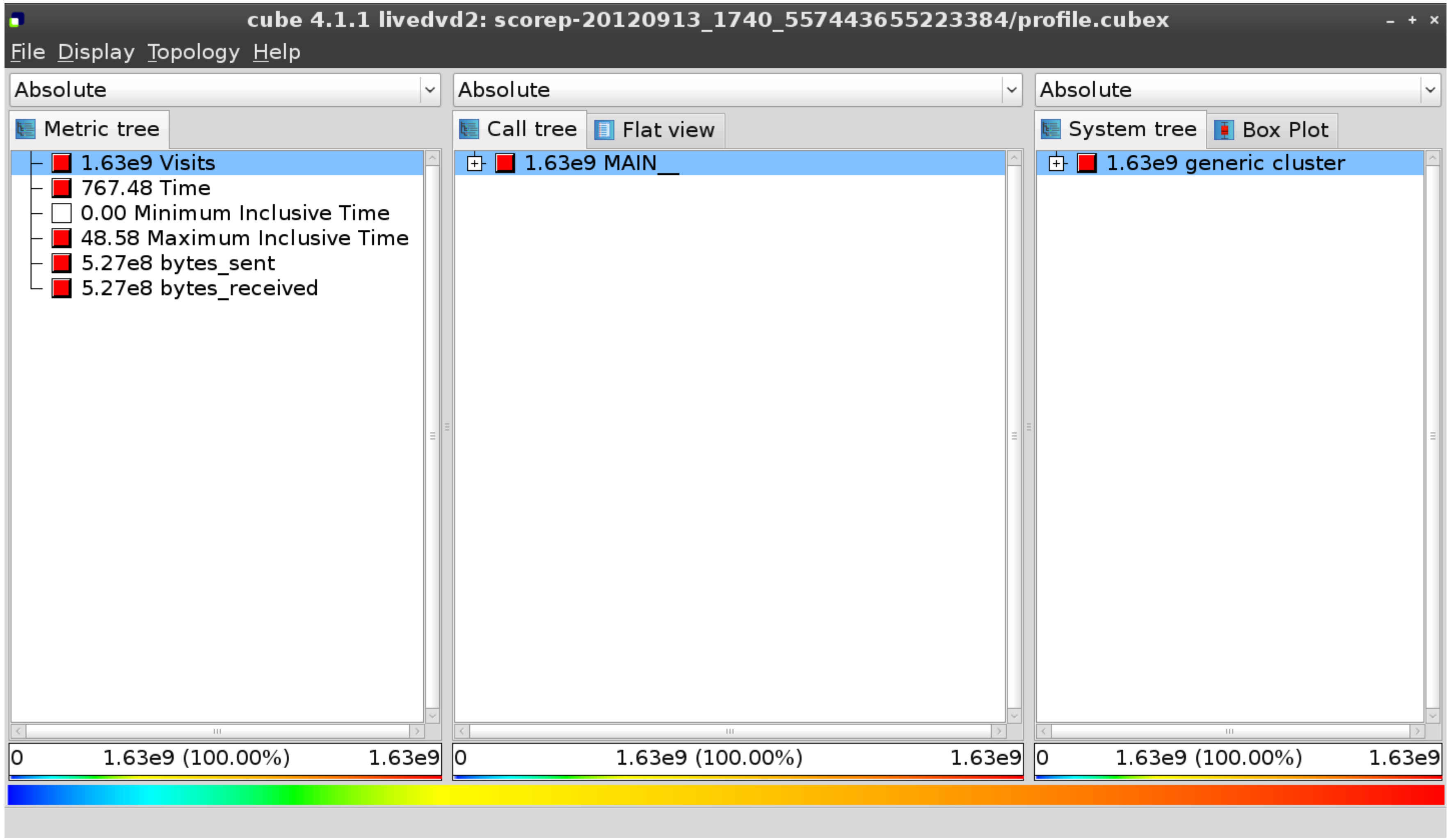
Inclusive vs. exclusive values

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



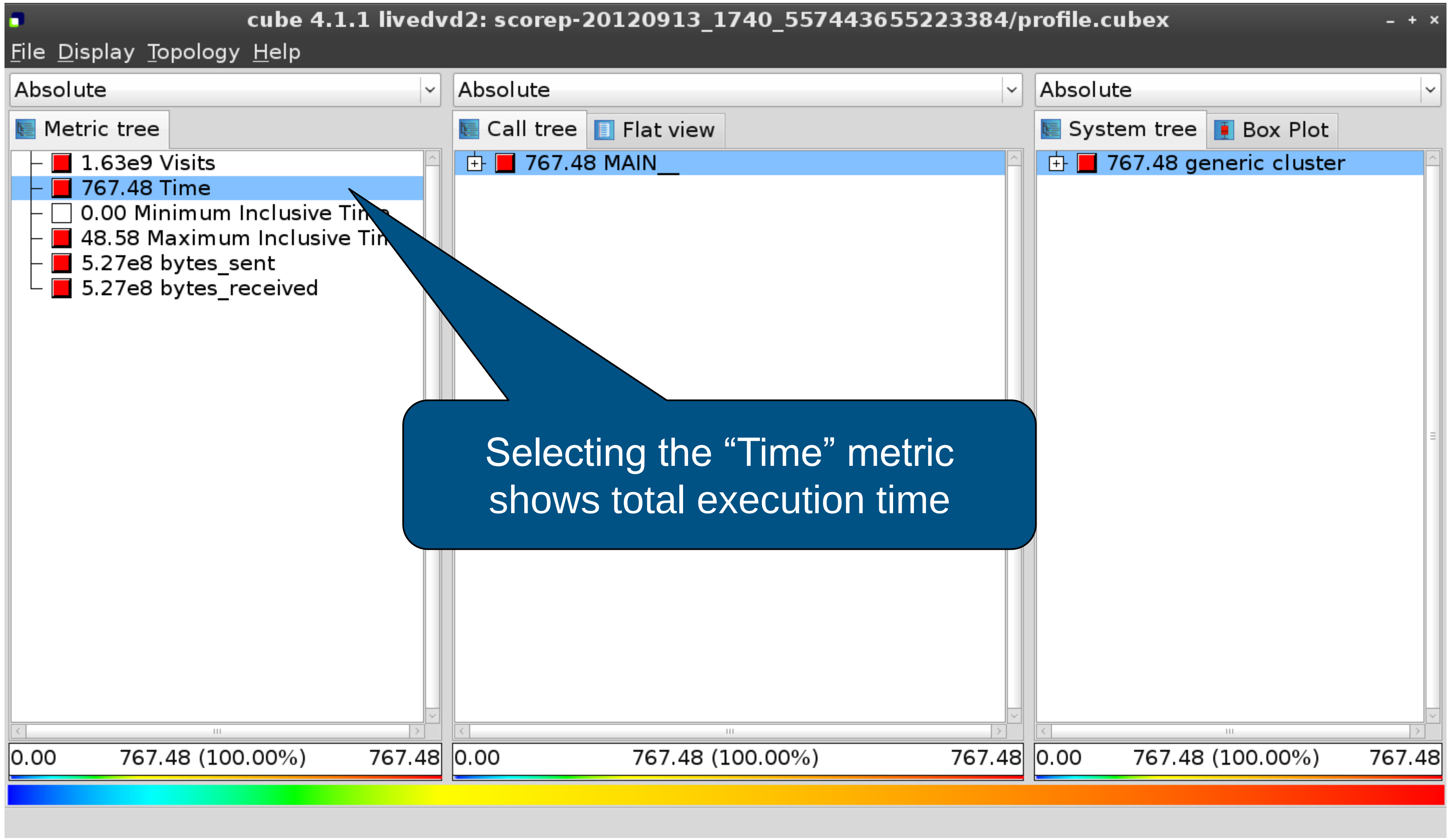
```
int foo()
{
  int a;
  a = 1 + 1;

  bar();

  a = a + 1;
  return a;
}
```

Score-P analysis report exploration (opening view)


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

```
Metric tree                         Call tree             System tree
1.63e9 Visits                       1.63e9 MAIN__         1.63e9 generic cluster
767.48 Time
0.00 Minimum Inclusive Time
48.58 Maximum Inclusive Time
5.27e8 bytes_sent
5.27e8 bytes_received

0   1.63e9 (100.00%)   1.63e9       0   1.63e9 (100.00%)   1.63e9       0   1.63e9 (100.00%)   1.63e9
```

Metric selection


cube 4.1.1 livedvd2: scorep-20120913_1740_557443655223384/profile.cubex
File Display Topology Help
Absolute
Absolute
Absolute
Metric tree
Call tree
Flat view
System tree
Box Plot

```
1.63e9 Visits
767.48 Time
0.00 Minimum Inclusive Time
48.58 Maximum Inclusive Time
5.27e8 bytes_sent
5.27e8 bytes_received
```

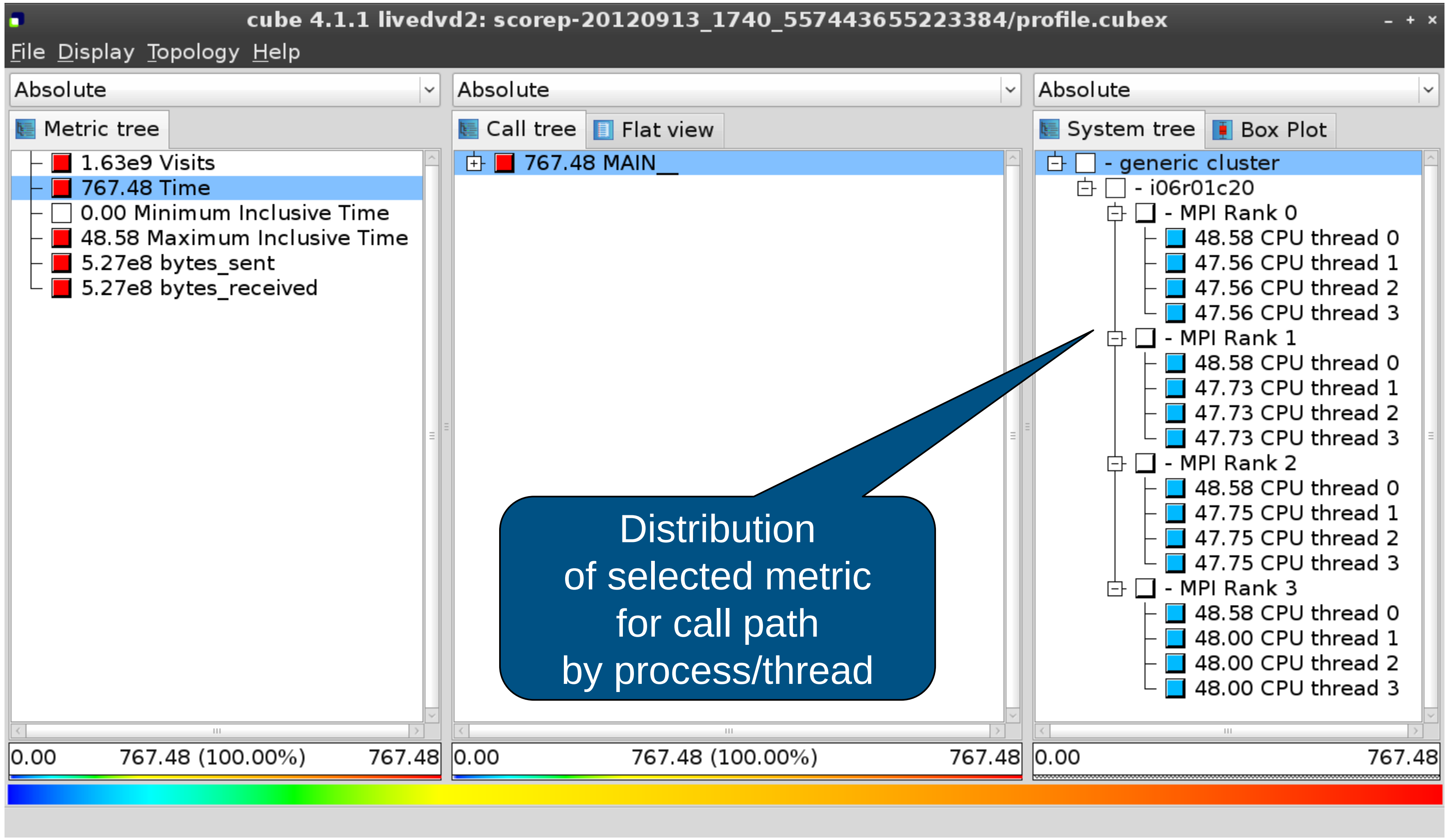
```
767.48 MAIN__
```

```
767.48 generic cluster
```

```
0.00    767.48 (100.00%)    767.48
```

Selecting the “Time” metric shows total execution time

Expanding the system tree


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

```
1.63e9 Visits
767.48 Time
0.00 Minimum Inclusive Time
48.58 Maximum Inclusive Time
5.27e8 bytes_sent
5.27e8 bytes_received
```

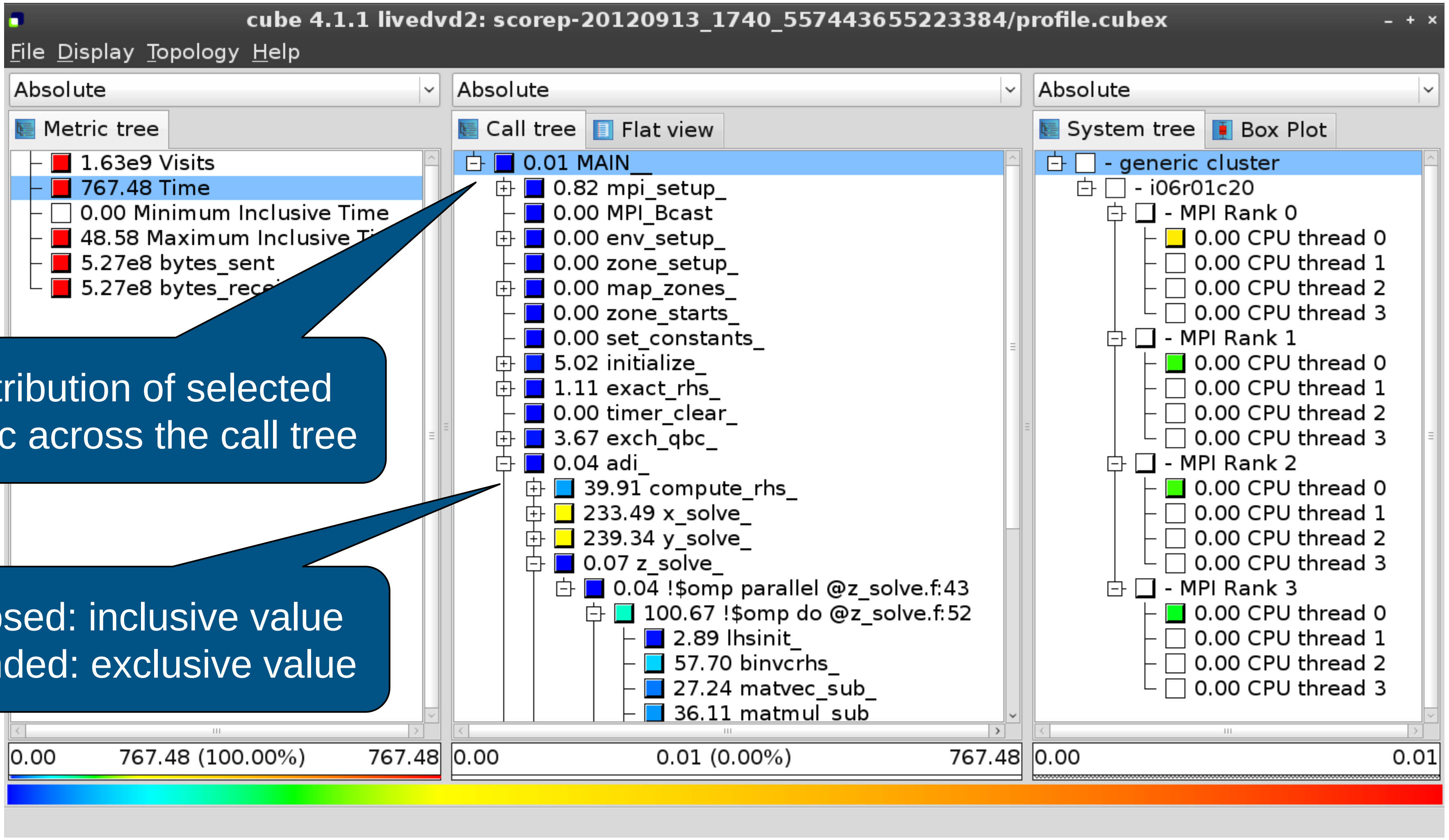
```
767.48 MAIN__
```

```
- generic cluster
  - i06r01c20
    - MPI Rank 0
      48.58 CPU thread 0
      47.56 CPU thread 1
      47.56 CPU thread 2
      47.56 CPU thread 3
    - MPI Rank 1
      48.58 CPU thread 0
      47.73 CPU thread 1
      47.73 CPU thread 2
      47.73 CPU thread 3
    - MPI Rank 2
      48.58 CPU thread 0
      47.75 CPU thread 1
      47.75 CPU thread 2
      47.75 CPU thread 3
    - MPI Rank 3
      48.58 CPU thread 0
      48.00 CPU thread 1
      48.00 CPU thread 2
      48.00 CPU thread 3
```

```
0.00   767.48 (100.00%)   767.48
0.00   767.48 (100.00%)   767.48
0.00                      767.48
```

Distribution of selected metric for call path by process/thread

Expanding the call tree



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

File Display Topology Help

Absolute | Absolute | Absolute

Metric tree:
```
1.63e9 Visits
767.48 Time
0.00 Minimum Inclusive Time
48.58 Maximum Inclusive Ti
5.27e8 bytes_sent
5.27e8 bytes_rece
```

Call tree | Flat view:
```
0.01 MAIN__
  0.82 mpi_setup_
  0.00 MPI_Bcast
  0.00 env_setup_
  0.00 zone_setup_
  0.00 map_zones_
  0.00 zone_starts_
  0.00 set_constants_
  5.02 initialize_
  1.11 exact_rhs_
  0.00 timer_clear_
  3.67 exch_qbc_
  0.04 adi_
    39.91 compute_rhs_
    233.49 x_solve_
    239.34 y_solve_
    0.07 z_solve_
      0.04 !$omp parallel @z_solve.f:43
        100.67 !$omp do @z_solve.f:52
          2.89 lhsinit_
          57.70 binvcrhs_
          27.24 matvec_sub_
          36.11 matmul_sub_
```

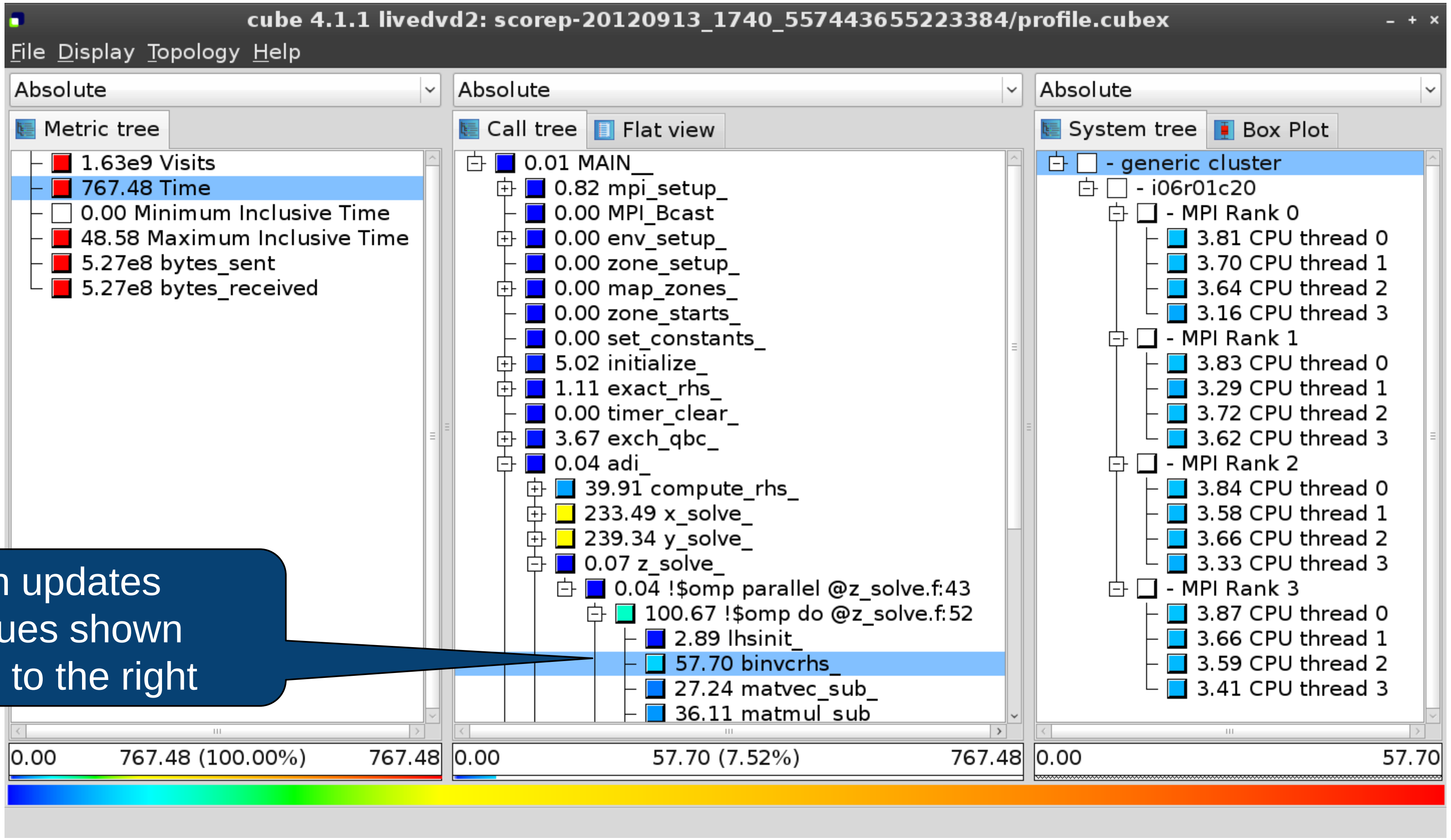
System tree | Box Plot:
```
- generic cluster
  - i06r01c20
    - MPI Rank 0
      0.00 CPU thread 0
      0.00 CPU thread 1
      0.00 CPU thread 2
      0.00 CPU thread 3
    - MPI Rank 1
      0.00 CPU thread 0
      0.00 CPU thread 1
      0.00 CPU thread 2
      0.00 CPU thread 3
    - MPI Rank 2
      0.00 CPU thread 0
      0.00 CPU thread 1
      0.00 CPU thread 2
      0.00 CPU thread 3
    - MPI Rank 3
      0.00 CPU thread 0
      0.00 CPU thread 1
      0.00 CPU thread 2
      0.00 CPU thread 3
```

0.00 767.48 (100.00%) 767.48 | 0.00 0.01 (0.00%) 767.48 | 0.00 0.01

Distribution of selected metric across the call tree

Collapsed: inclusive value
Expanded: exclusive value

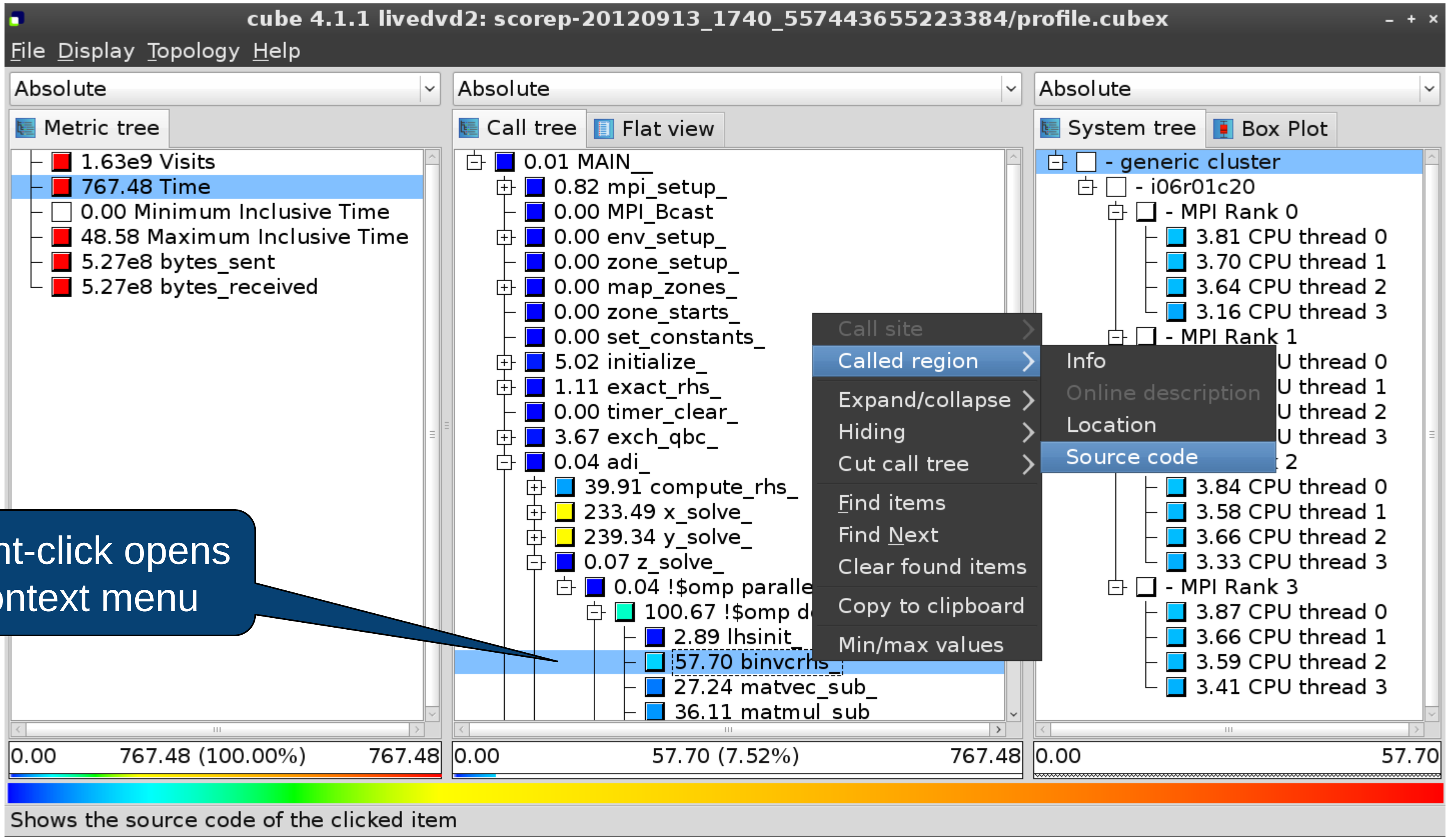
Selecting a call path



```
cube 4.1.1 livedvd2: scorep-20120913_1740_557443655223384/profile.cubex
File  Display  Topology  Help

Absolute                        Absolute                                   Absolute
Metric tree                     Call tree | Flat view                      System tree | Box Plot

1.63e9 Visits                   0.01 MAIN__                                - generic cluster
767.48 Time                       0.82 mpi_setup_                            - i06r01c20
0.00 Minimum Inclusive Time       0.00 MPI_Bcast                               - MPI Rank 0
48.58 Maximum Inclusive Time      0.00 env_setup_                                3.81 CPU thread 0
5.27e8 bytes_sent                 0.00 zone_setup_                               3.70 CPU thread 1
5.27e8 bytes_received             0.00 map_zones_                                3.64 CPU thread 2
                                  0.00 zone_starts_                              3.16 CPU thread 3
                                  0.00 set_constants_                          - MPI Rank 1
                                  5.02 initialize_                               3.83 CPU thread 0
                                  1.11 exact_rhs_                                3.29 CPU thread 1
                                  0.00 timer_clear_                              3.72 CPU thread 2
                                  3.67 exch_qbc_                                 3.62 CPU thread 3
                                  0.04 adi_                                    - MPI Rank 2
                                    39.91 compute_rhs_                           3.84 CPU thread 0
                                    233.49 x_solve_                              3.58 CPU thread 1
                                    239.34 y_solve_                              3.66 CPU thread 2
                                    0.07 z_solve_                                3.33 CPU thread 3
                                      0.04 !$omp parallel @z_solve.f:43        - MPI Rank 3
                                        100.67 !$omp do @z_solve.f:52            3.87 CPU thread 0
                                          2.89 lhsinit_                          3.66 CPU thread 1
                                          57.70 binvcrhs_                        3.59 CPU thread 2
                                          27.24 matvec_sub_                      3.41 CPU thread 3
                                          36.11 matmul_sub_

0.00   767.48 (100.00%)   767.48  0.00   57.70 (7.52%)   767.48      0.00            57.70
```

Selection updates metric values shown in columns to the right

Source-code view via context menu



```
1.63e9 Visits
767.48 Time
0.00 Minimum Inclusive Time
48.58 Maximum Inclusive Time
5.27e8 bytes_sent
5.27e8 bytes_received
```

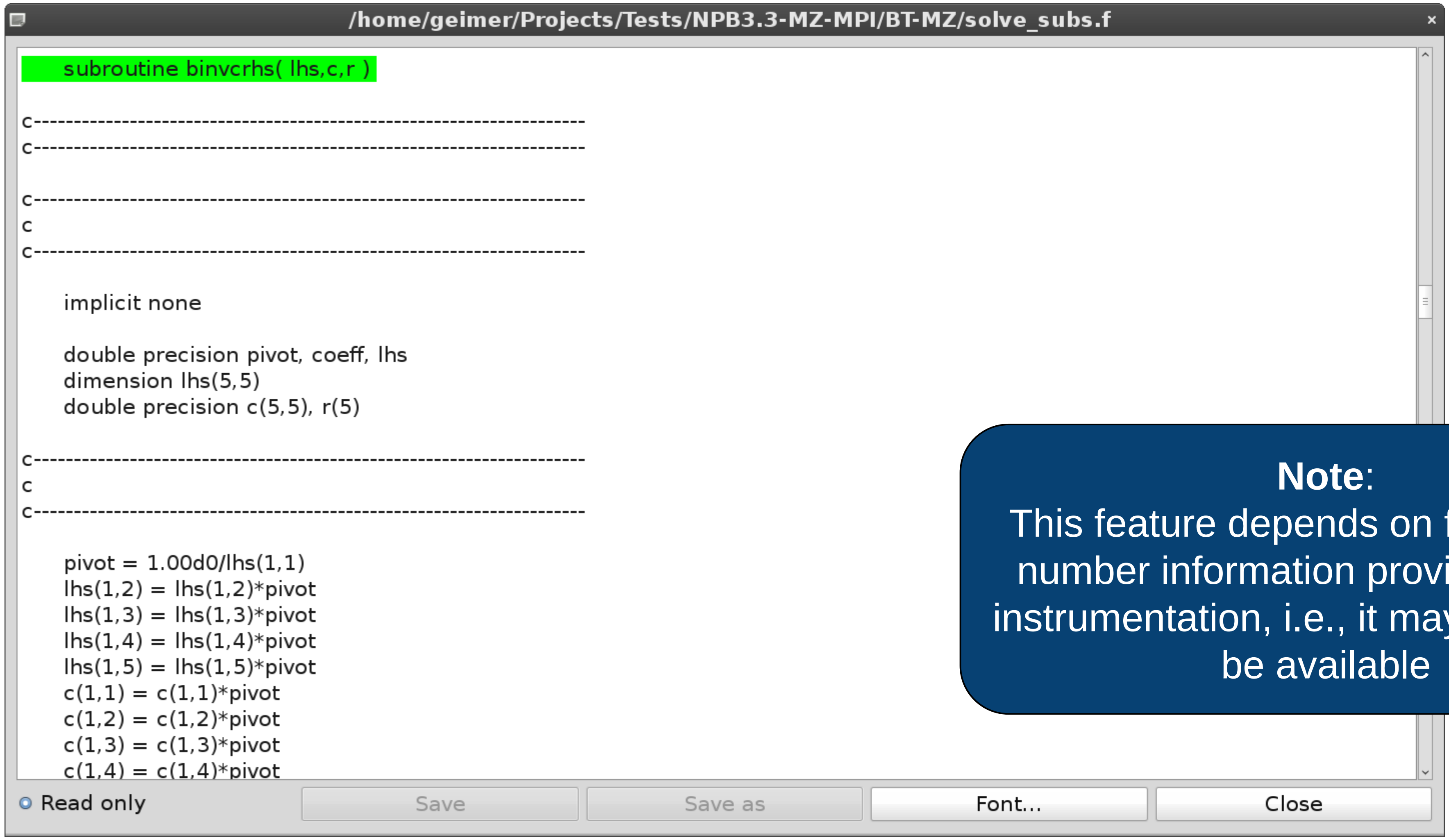
```
0.01 MAIN__
  0.82 mpi_setup_
  0.00 MPI_Bcast
  0.00 env_setup_
  0.00 zone_setup_
  0.00 map_zones_
  0.00 zone_starts_
  0.00 set_constants_
  5.02 initialize_
  1.11 exact_rhs_
  0.00 timer_clear_
  3.67 exch_qbc_
  0.04 adi_
    39.91 compute_rhs_
    233.49 x_solve_
    239.34 y_solve_
    0.07 z_solve_
      0.04 !$omp paralle
        100.67 !$omp d
          2.89 lhsinit_
          57.70 binvcrhs_
          27.24 matvec_sub_
          36.11 matmul_sub
```

```
- generic cluster
  - i06r01c20
    - MPI Rank 0
      3.81 CPU thread 0
      3.70 CPU thread 1
      3.64 CPU thread 2
      3.16 CPU thread 3
    - MPI Rank 1
      U thread 0
      U thread 1
      U thread 2
      U thread 3
    2
      3.84 CPU thread 0
      3.58 CPU thread 1
      3.66 CPU thread 2
      3.33 CPU thread 3
    - MPI Rank 3
      3.87 CPU thread 0
      3.66 CPU thread 1
      3.59 CPU thread 2
      3.41 CPU thread 3
```

```
0.00    767.48 (100.00%)    767.48
0.00    57.70 (7.52%)    767.48
0.00    57.70
Shows the source code of the clicked item
```

Right-click opens context menu

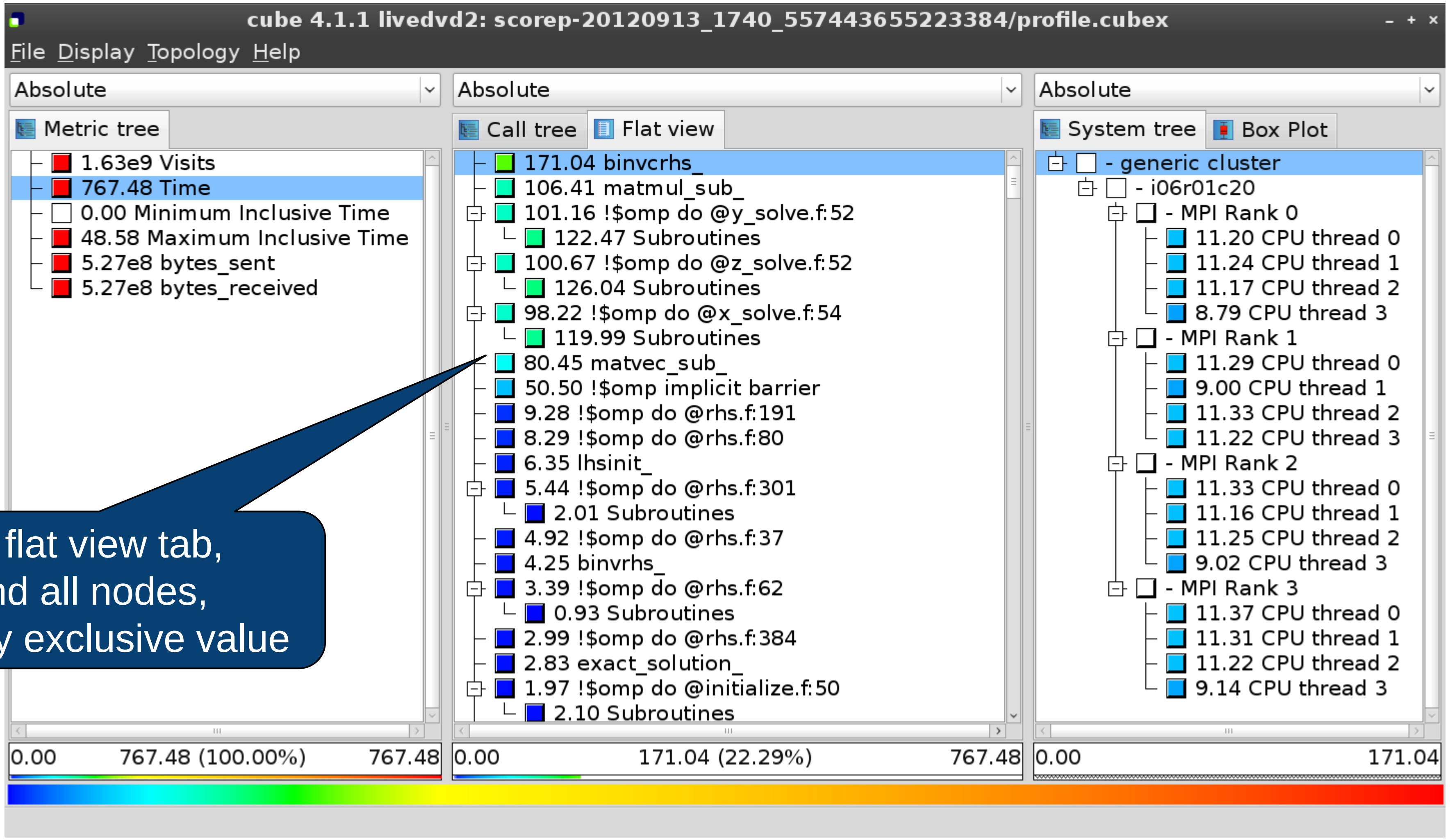
Source-code view


/home/geimer/Projects/Tests/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f
Read only
Save
Save as
Font...
Close

```
      subroutine binvcrhs( 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
```

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


cube 4.1.1 livedvd2: scorep-20120913_1740_557443655223384/profile.cubex
File Display Topology Help
Absolute
Absolute
Absolute
Metric tree
Call tree
Flat view
System tree
Box Plot

```
1.63e9 Visits
767.48 Time
0.00 Minimum Inclusive Time
48.58 Maximum Inclusive Time
5.27e8 bytes_sent
5.27e8 bytes_received
```

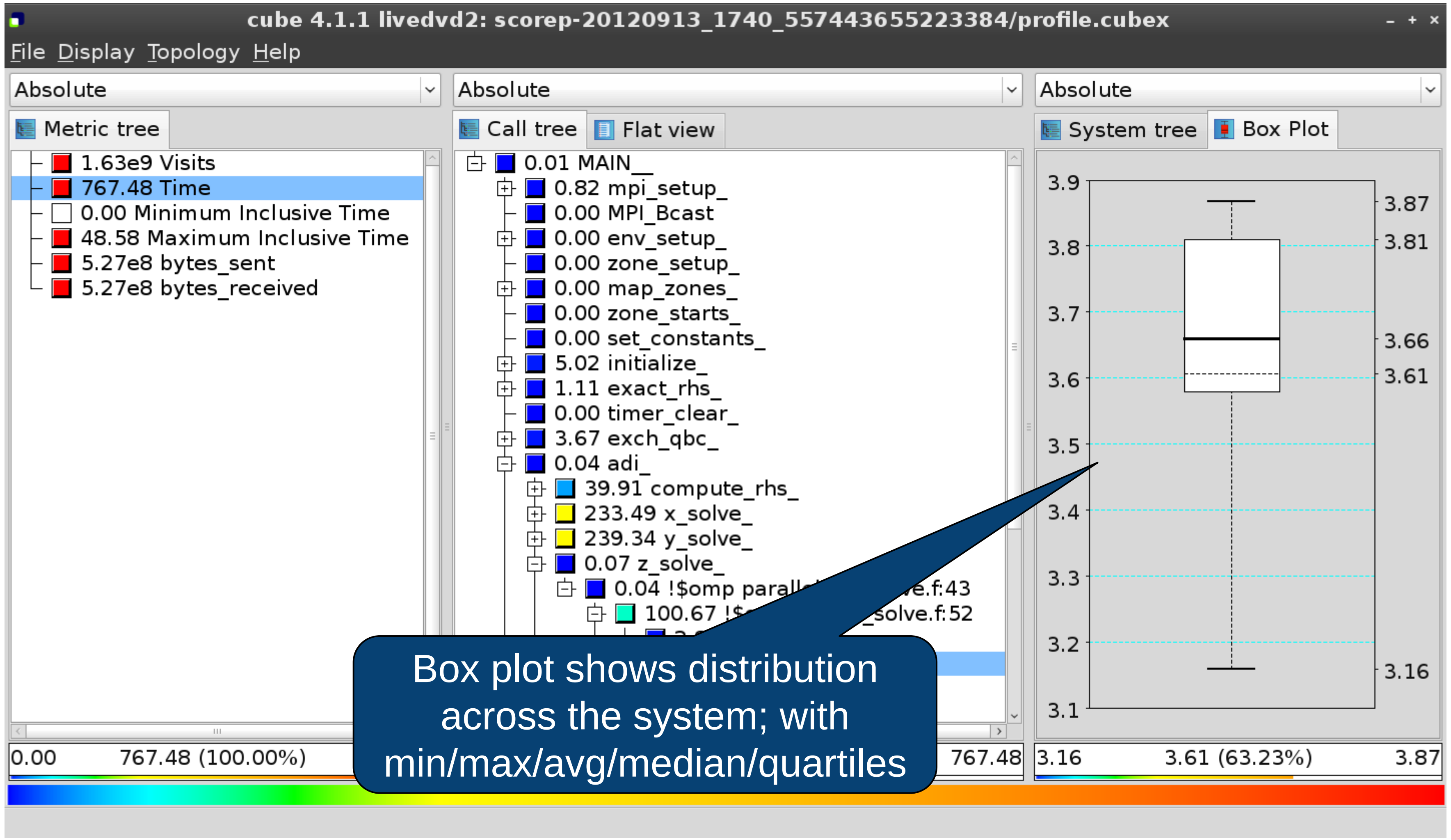
```
171.04 binvcrhs_
106.41 matmul_sub_
101.16 !$omp do @y_solve.f:52
  122.47 Subroutines
100.67 !$omp do @z_solve.f:52
  126.04 Subroutines
98.22 !$omp do @x_solve.f:54
  119.99 Subroutines
80.45 matvec_sub_
50.50 !$omp implicit barrier
9.28 !$omp do @rhs.f:191
8.29 !$omp do @rhs.f:80
6.35 lhsinit_
5.44 !$omp do @rhs.f:301
  2.01 Subroutines
4.92 !$omp do @rhs.f:37
4.25 binvrhs_
3.39 !$omp do @rhs.f:62
  0.93 Subroutines
2.99 !$omp do @rhs.f:384
2.83 exact_solution_
1.97 !$omp do @initialize.f:50
  2.10 Subroutines
```

```
- generic cluster
  - i06r01c20
    - MPI Rank 0
      11.20 CPU thread 0
      11.24 CPU thread 1
      11.17 CPU thread 2
      8.79 CPU thread 3
    - MPI Rank 1
      11.29 CPU thread 0
      9.00 CPU thread 1
      11.33 CPU thread 2
      11.22 CPU thread 3
    - MPI Rank 2
      11.33 CPU thread 0
      11.16 CPU thread 1
      11.25 CPU thread 2
      9.02 CPU thread 3
    - MPI Rank 3
      11.37 CPU thread 0
      11.31 CPU thread 1
      11.22 CPU thread 2
      9.14 CPU thread 3
```

```
0.00   767.48 (100.00%)   767.48
0.00   171.04 (22.29%)   767.48
0.00   171.04
```

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

Box plot view



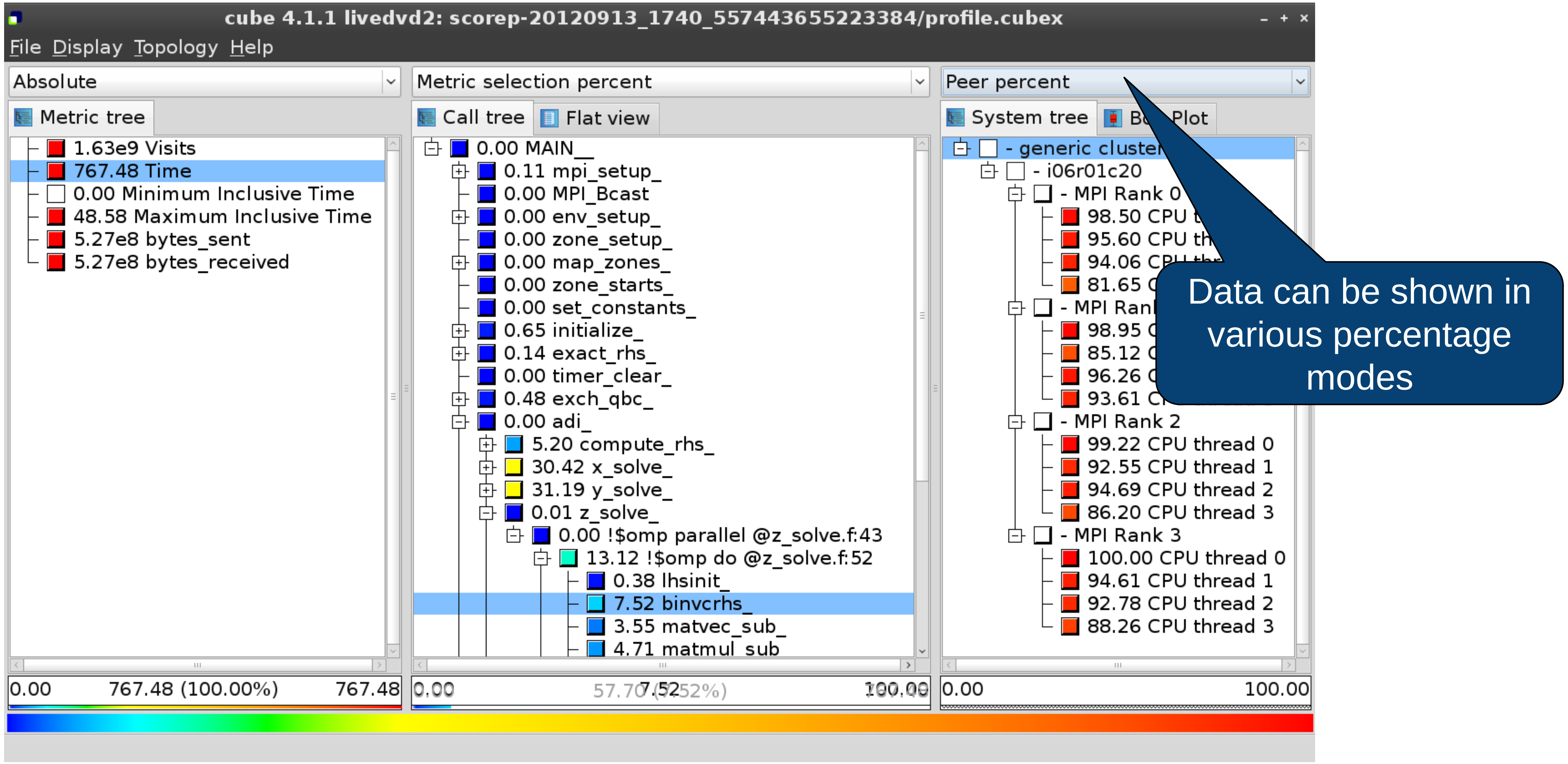
```
1.63e9 Visits
767.48 Time
0.00 Minimum Inclusive Time
48.58 Maximum Inclusive Time
5.27e8 bytes_sent
5.27e8 bytes_received
```

```
0.01 MAIN__
  0.82 mpi_setup_
  0.00 MPI_Bcast
  0.00 env_setup_
  0.00 zone_setup_
  0.00 map_zones_
  0.00 zone_starts_
  0.00 set_constants_
  5.02 initialize_
  1.11 exact_rhs_
  0.00 timer_clear_
  3.67 exch_qbc_
  0.04 adi_
    39.91 compute_rhs_
    233.49 x_solve_
    239.34 y_solve_
    0.07 z_solve_
      0.04 !$omp parall      e.f:43
        100.67 !$          solve.f:52
```

```
0.00        767.48 (100.00%)                767.48   3.16        3.61 (63.23%)        3.87
```

Box plot shows distribution across the system; with min/max/avg/median/quartiles

Alternative display modes


cube 4.1.1 livedvd2: scorep-20120913_1740_557443655223384/profile.cubex
File Display Topology Help
Absolute
Metric selection percent
Peer percent
Metric tree
Call tree
Flat view
System tree
Box Plot
Data can be shown in various percentage modes

```
1.63e9 Visits
767.48 Time
0.00 Minimum Inclusive Time
48.58 Maximum Inclusive Time
5.27e8 bytes_sent
5.27e8 bytes_received
```

```
0.00 MAIN__
  0.11 mpi_setup_
  0.00 MPI_Bcast
  0.00 env_setup_
  0.00 zone_setup_
  0.00 map_zones_
  0.00 zone_starts_
  0.00 set_constants_
  0.65 initialize_
  0.14 exact_rhs_
  0.00 timer_clear_
  0.48 exch_qbc_
  0.00 adi_
    5.20 compute_rhs_
    30.42 x_solve_
    31.19 y_solve_
    0.01 z_solve_
      0.00 !$omp parallel @z_solve.f:43
        13.12 !$omp do @z_solve.f:52
          0.38 lhsinit_
          7.52 binvcrhs_
          3.55 matvec_sub_
          4.71 matmul_sub
```

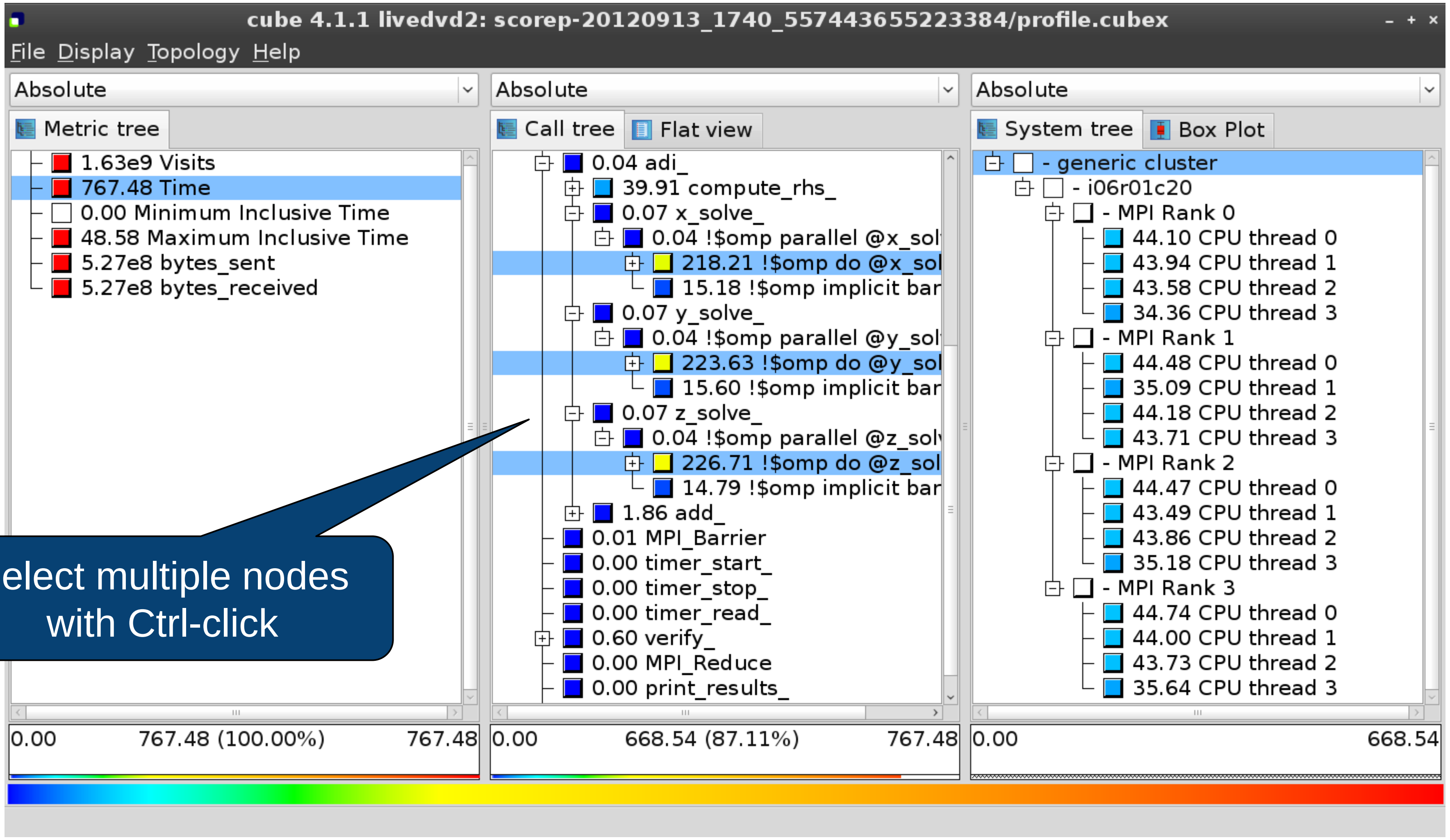
```
- generic cluster
  - i06r01c20
    - MPI Rank 0
      98.50 CPU t
      95.60 CPU th
      94.06 CPU thr
      81.65 C
    - MPI Rank
      98.95 C
      85.12 C
      96.26 C
      93.61 C
    - MPI Rank 2
      99.22 CPU thread 0
      92.55 CPU thread 1
      94.69 CPU thread 2
      86.20 CPU thread 3
    - MPI Rank 3
      100.00 CPU thread 0
      94.61 CPU thread 1
      92.78 CPU thread 2
      88.26 CPU thread 3
```

0.00 767.48 (100.00%) 767.48 | 0.00 57.70 7.52 (7.52%) 100.00 | 0.00 100.00

Important display modes

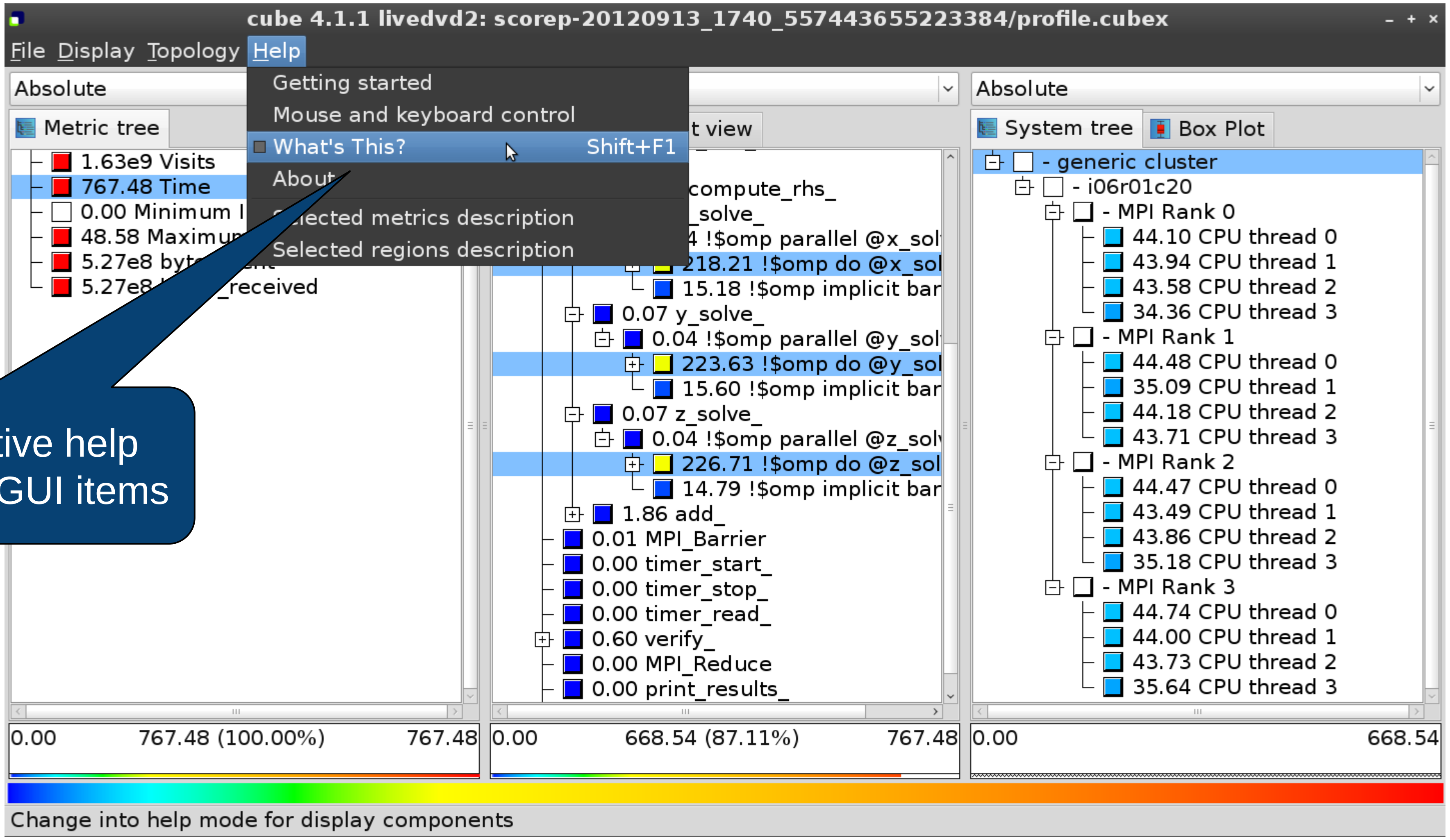
- Absolute
  - Absolute value shown in seconds/bytes/counts
- Selection percent
  - Value shown as percentage w.r.t. the selected node "on the left" (metric/call path)
- Peer percent (system tree only)
  - Value shown as percentage relative to the maximum peer value

Multiple selection


cube 4.1.1 livedvd2: scorep-20120913_1740_557443655223384/profile.cubex
File Display Topology Help
Absolute
Metric tree
Call tree
Flat view
System tree
Box Plot
Select multiple nodes with Ctrl-click

```
1.63e9 Visits
767.48 Time
0.00 Minimum Inclusive Time
48.58 Maximum Inclusive Time
5.27e8 bytes_sent
5.27e8 bytes_received

0.04 adi_
  39.91 compute_rhs_
  0.07 x_solve_
    0.04 !$omp parallel @x_sol
      218.21 !$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 !$omp parallel @z_solv
      226.71 !$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_

- 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
```

Context-sensitive help


cube 4.1.1 livedvd2: scorep-20120913_1740_557443655223384/profile.cubex
File Display Topology Help
Getting started
Mouse and keyboard control
What's This? Shift+F1
About
Selected metrics description
Selected regions description
Absolute
Metric tree
System tree
Box Plot

```
1.63e9 Visits
767.48 Time
0.00 Minimum I
48.58 Maximum
5.27e8 byt
5.27e8 received
```

```
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_
```

```
- 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

Context-sensitive help available for all GUI items

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

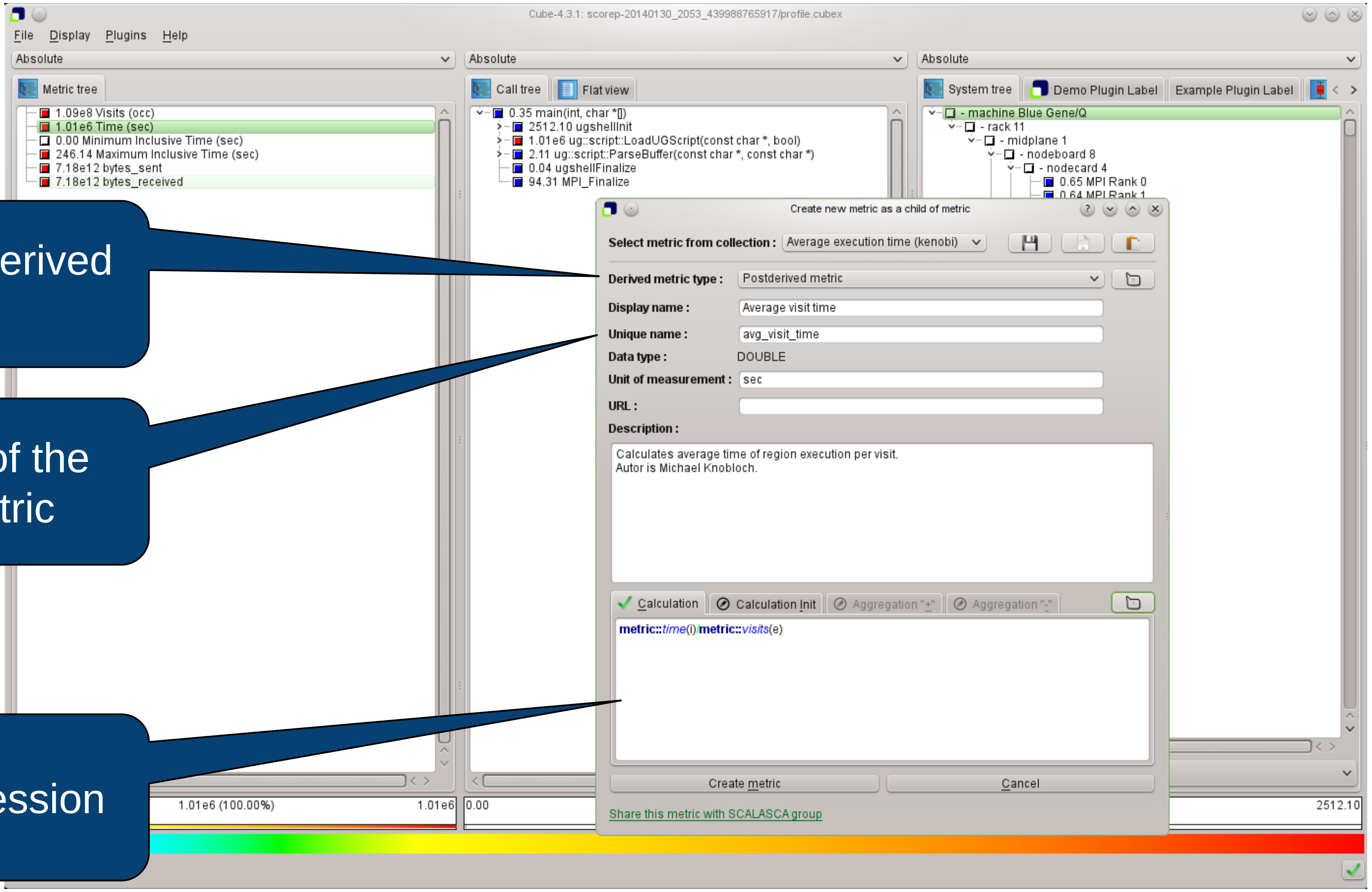
- Examples:
  - "Average execution time": Postderived metric with expression

    metric::time(i)/metric::visits(e)

  - "Number of FLOP per second": Postderived metric with expression

    metric::FLOP()/metric::time()

Derived metrics in Cube GUI



```
1.09e8 Visits (occ)
1.01e6 Time (sec)
0.00 Minimum Inclusive Time (sec)
246.14 Maximum Inclusive Time (sec)
7.18e12 bytes_sent
7.18e12 bytes_received
```

```
0.35 main(int, char *[])
  2512.10 ugshellInit
  1.01e6 ug::script::LoadUGScript(const char *, bool)
  2.11 ug::script::ParseBuffer(const char *, const char *)
  0.04 ugshellFinalize
  94.31 MPI_Finalize
```

```
- machine Blue Gene/Q
  - rack 11
    - midplane 1
      - nodeboard 8
        - nodecard 4
          0.65 MPI Rank 0
          0.64 MPI Rank 1
```

Select metric from collection : Average execution time (kenobi)

Derived metric type : Postderived metric

Display name : Average visit time

Unique name : avg_visit_time

Data type : DOUBLE

Unit of measurement : sec

URL :

Description :

Calculates average time of region execution per visit.
Autor is Michael Knobloch.

```
metric::time(i)/metric::visits(e)
```

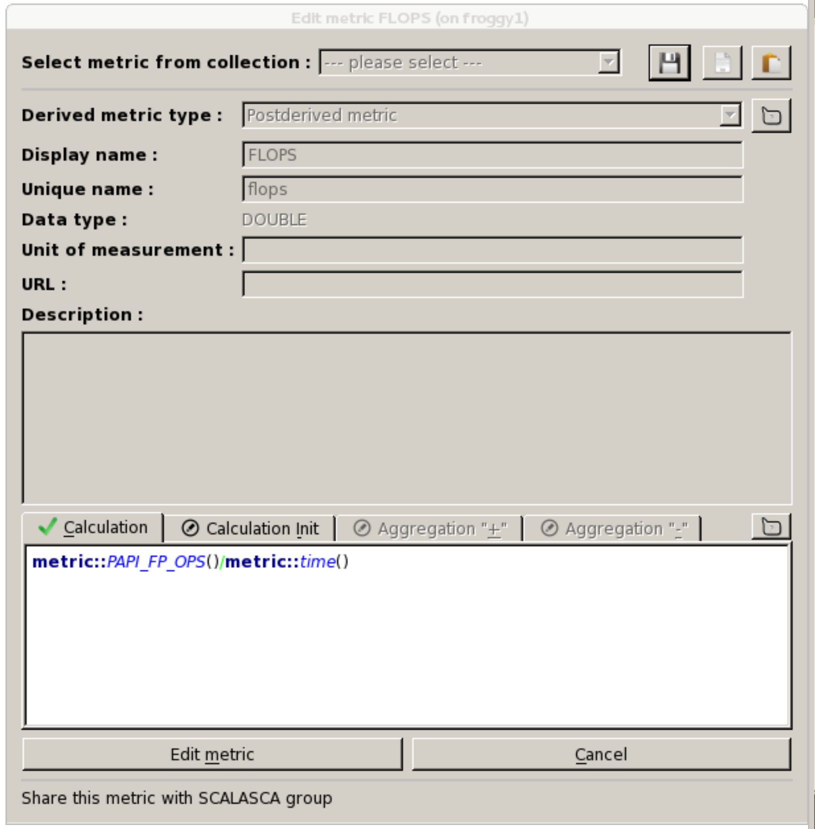
1.01e6 (100.00%) 1.01e6 0.00 2512.10

Collection of derived metrics

Parameters of the derived metric

CubePL expression

Example: FLOPS based on PAPI_FP_OPS and time



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

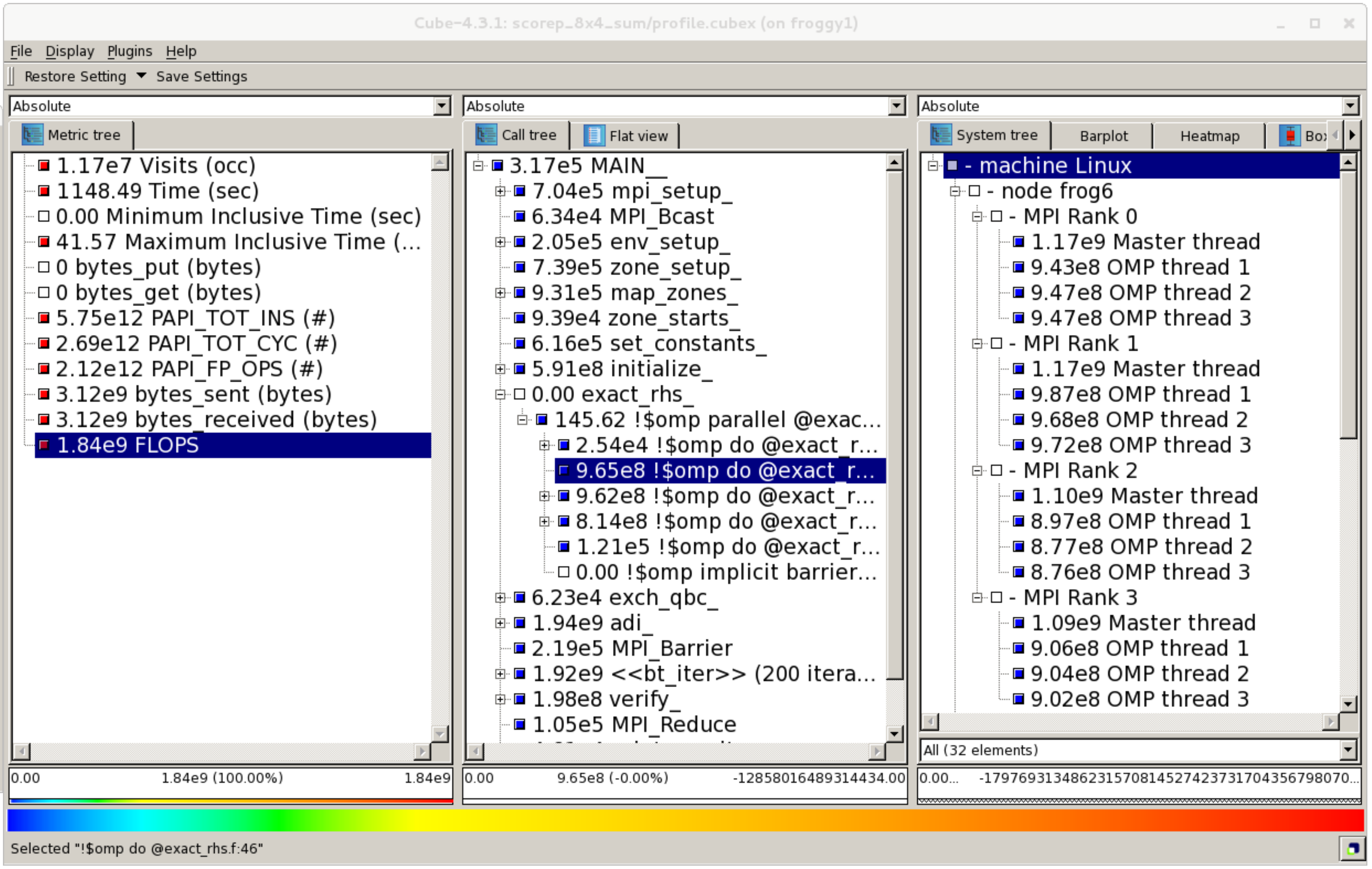
Unit of measurement :

URL :

Description :

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

Share this metric with SCALASCA group



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

```
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
```

```
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
```

```
- 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

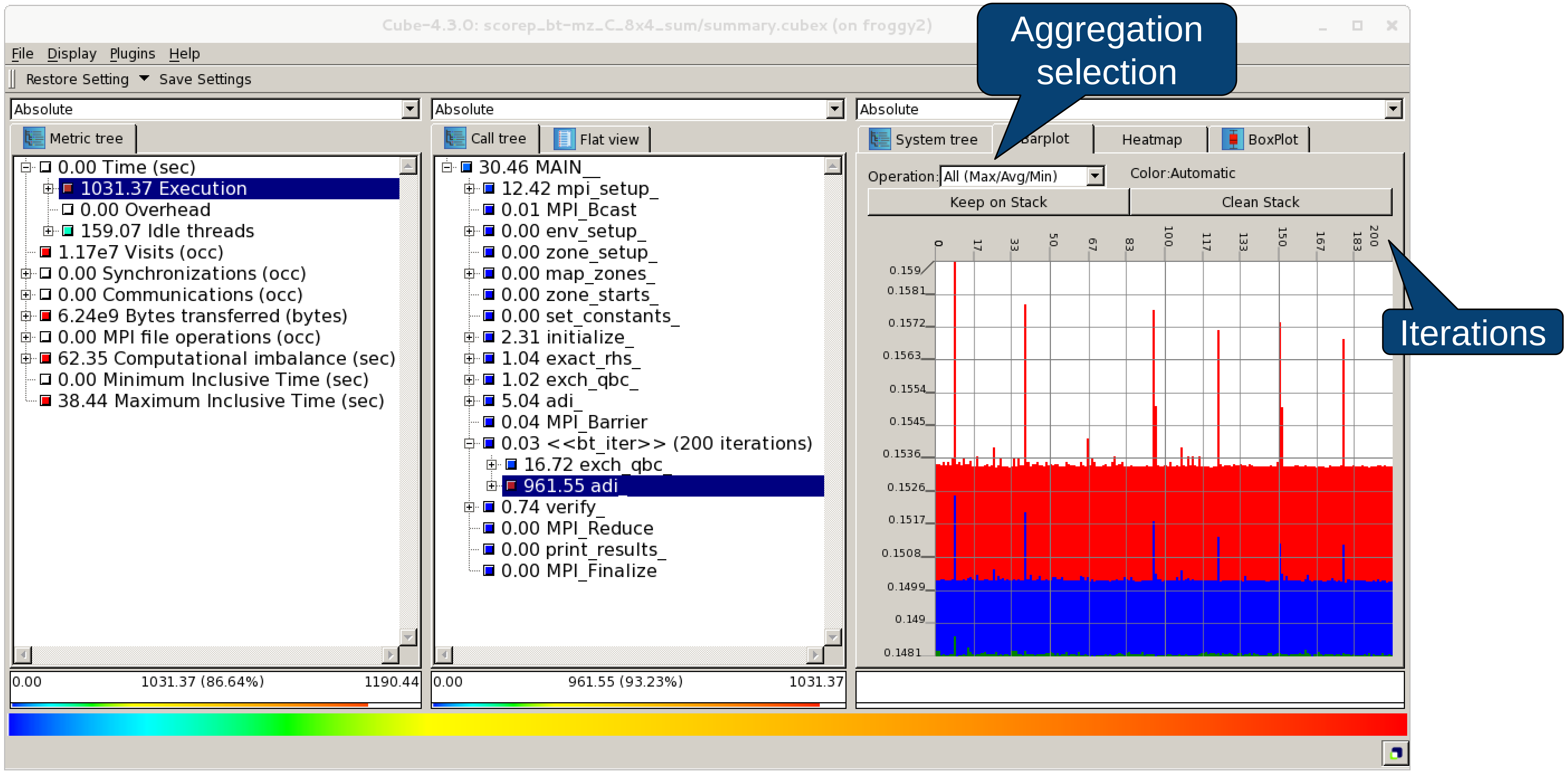
  ➢Useful for checking results for specific iterations

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

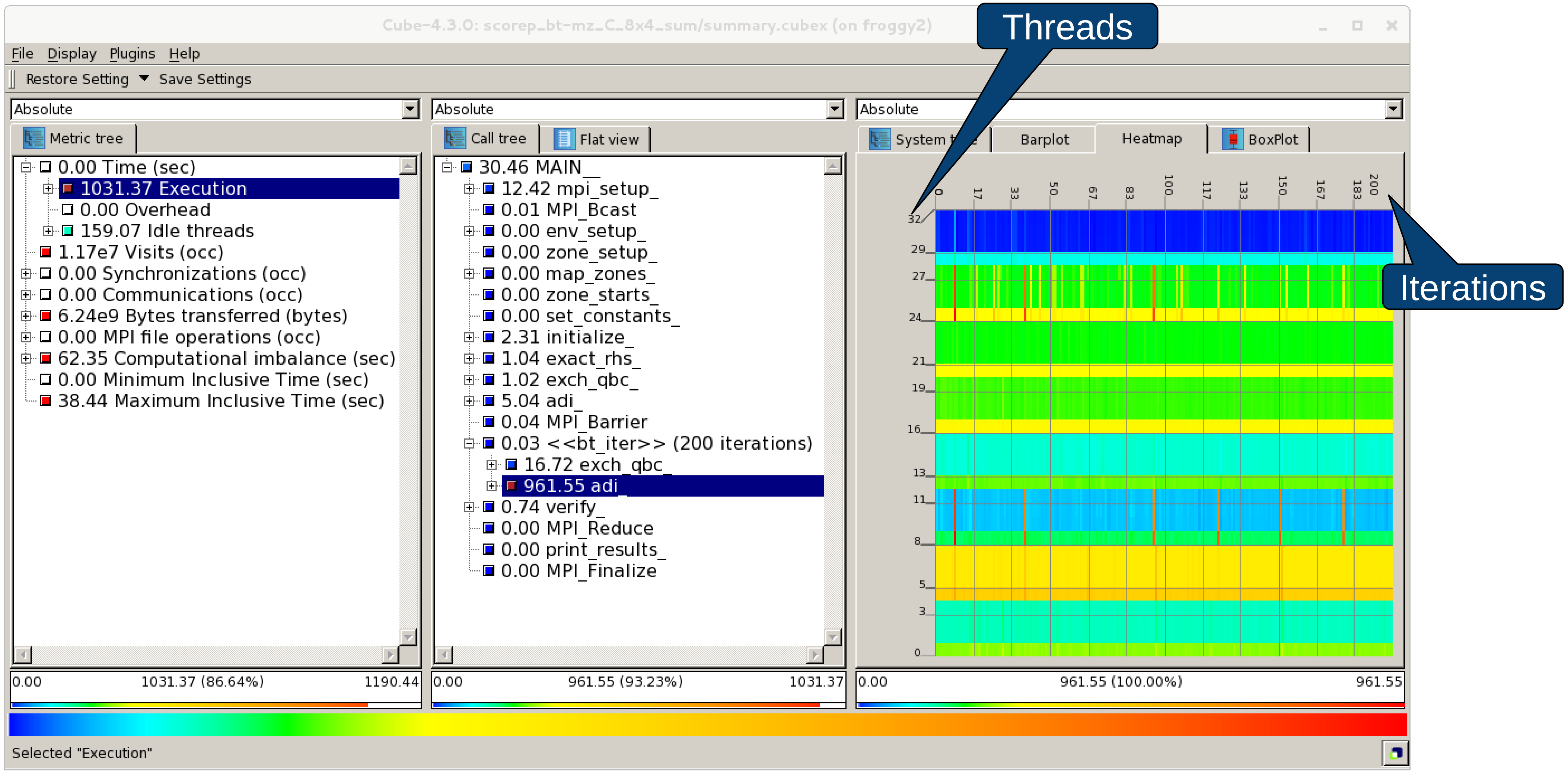

Cube-4.3.0: scorep_bt-mz_C_8x4_sum/summary.cubex (on froggy2)
File Display Plugins Help
Restore Setting Save Settings
Absolute
Metric tree
Call tree
Flat view
System tree
Barplot
Heatmap
BoxPlot
Operation: All (Max/Avg/Min)
Color:Automatic
Keep on Stack
Clean Stack
Aggregation selection
Iterations

```
0.00 Time (sec)
  1031.37 Execution
  0.00 Overhead
  159.07 Idle threads
1.17e7 Visits (occ)
0.00 Synchronizations (occ)
0.00 Communications (occ)
6.24e9 Bytes transferred (bytes)
0.00 MPI file operations (occ)
62.35 Computational imbalance (sec)
0.00 Minimum Inclusive Time (sec)
38.44 Maximum Inclusive Time (sec)
```

```
30.46 MAIN_
  12.42 mpi_setup_
  0.01 MPI_Bcast
  0.00 env_setup_
  0.00 zone_setup_
  0.00 map_zones_
  0.00 zone_starts_
  0.00 set_constants_
  2.31 initialize_
  1.04 exact_rhs_
  1.02 exch_qbc_
  5.04 adi_
  0.04 MPI_Barrier
  0.03 <<bt_iter>> (200 iterations)
    16.72 exch_qbc_
    961.55 adi_
  0.74 verify_
  0.00 MPI_Reduce
  0.00 print_results_
  0.00 MPI_Finalize
```

```
0.00    1031.37 (86.64%)    1190.44
0.00    961.55 (93.23%)    1031.37
```

Iteration profiling: Heatmap



```
0.00 Time (sec)
  1031.37 Execution
  0.00 Overhead
  159.07 Idle threads
1.17e7 Visits (occ)
0.00 Synchronizations (occ)
0.00 Communications (occ)
6.24e9 Bytes transferred (bytes)
0.00 MPI file operations (occ)
62.35 Computational imbalance (sec)
0.00 Minimum Inclusive Time (sec)
38.44 Maximum Inclusive Time (sec)
```

```
30.46 MAIN__
  12.42 mpi_setup_
  0.01 MPI_Bcast
  0.00 env_setup_
  0.00 zone_setup_
  0.00 map_zones_
  0.00 zone_starts_
  0.00 set_constants_
  2.31 initialize_
  1.04 exact_rhs_
  1.02 exch_qbc_
  5.04 adi_
  0.04 MPI_Barrier
  0.03 <<bt_iter>> (200 iterations)
    16.72 exch_qbc_
    961.55 adi_
  0.74 verify_
  0.00 MPI_Reduce
  0.00 print_results_
  0.00 MPI_Finalize
```

```
0.00    1031.37 (86.64%)    1190.44
0.00    961.55 (93.23%)     1031.37
0.00    961.55 (100.00%)    961.55
```

Selected "Execution"

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:
  - http://www.scalasca.org
- User guide also part of installation:
  - \`cube-config --cube-dir\`/share/doc/CubeGuide.pdf
- Contact:
  - mailto: scalasca@fz-juelich.de

Score-P – A Joint Performance Measurement Run-Time Infrastructure for Periscope, Scalasca, TAU, and Vampir

VI-HPS Team

Score-P

Scalable performance measurement infrastructure for parallel codes

Congratulations!?

- If you made it this far, you successfully used Score-P to
  - instrument the application
  - analyze its execution with a summary measurement, and
  - examine it with one the interactive analysis report explorer GUIs
- ... revealing the call-path profile annotated with
  - the “Time” metric
  - Visit counts
  - MPI message statistics (bytes sent/received)
- ... but how ***good*** was the measurement?
  - The measured execution produced the desired valid result
  - however, the execution took rather longer than expected!
    - even when ignoring measurement start-up/completion, therefore
    - it was probably dilated by instrumentation/measurement overhead

Performance analysis steps

- 0.0 Reference preparation for validation

- 1.0 Program instrumentation
- 1.1 Summary measurement collection
- 1.2 Summary analysis report examination

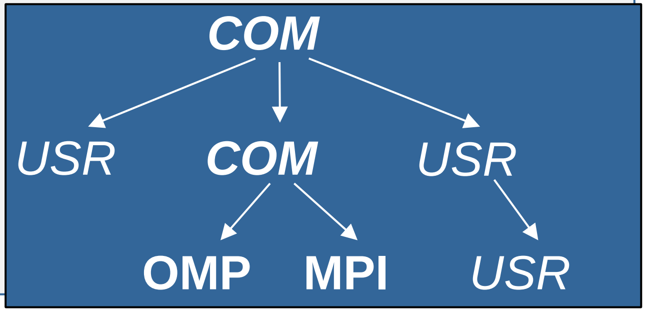
- 2.0 Summary experiment scoring
- 2.1 Summary measurement collection with filtering
- 2.2 Filtered summary analysis report examination

- 3.0 Event trace collection
- 3.1 Event trace examination & analysis

BT-MZ summary analysis result scoring

```
% scorep-score scorep_bt-mz_sum/profile.cubex

Estimated aggregate size of event trace:                    160 GB
Estimated requirements for largest trace buffer (max_buf):    6 GB
Estimated memory requirements (SCOREP_TOTAL_MEMORY):          6 GB
(warning: The memory requirements cannot be satisfied by Score-P to avoid
 intermediate flushes when tracing. Set SCOREP_TOTAL_MEMORY=4G to get the
 maximum supported memory or reduce requirements using USR regions filters.)

flt type      max_buf[B]        visits  time[s] time[%] time/visit[us]  region
     ALL   5,421,104,056 6,586,922,497  8162.56   100.0           1.24  ALL
     USR   5,407,570,350 6,574,832,225  3960.99    48.5           0.60  USR
     OMP      15,783,372    10,975,232  4085.92    50.1         372.29  OMP
     MPI         944,200       386,560    92.05     1.1         238.13  MPI
     COM         665,210       728,480    23.60     0.3          32.40  COM
```

160 GB total memory
6 GB per rank!

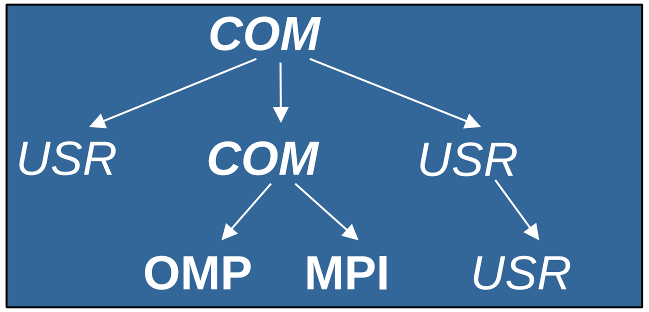


- Report scoring as textual output

- Region/callpath classification
  - **MPI** pure MPI functions
  - **OMP** pure OpenMP regions
  - **USR** user-level computation
  - **COM** “combined” USR+OpenMP/MPI
  - **ANY/ALL** aggregate of all region types

BT-MZ summary analysis report breakdown

```
% scorep-score -r scorep_bt-mz_sum/profile.cubex
  [...]
  [...]
flt type      max_buf[B]         visits  time[s] time[%] time/visit[us]  region
     ALL   5,421,104,056 6,586,922,497  8162.56   100.0           1.24  ALL
     USR   5,407,570,350 6,574,832,225  3960.99    48.5           0.60  USR
     OMP      15,783,372    10,975,232  4085.92    50.1         372.29  OMP
     MPI         944,200       386,560    92.05     1.1         238.13  MPI
     COM         665,210       728,480    23.60     0.3          32.40  COM

     USR   1,741,005,318 2,110,313,472  1204.11    14.8           0.57  matmul_sub_
     USR   1,741,005,318 2,110,313,472   851.97    10.4           0.40  matvec_sub_
     USR   1,741,005,318 2,110,313,472  1754.58    21.5           0.83  binvcrhs_
     USR      76,367,538    87,475,200    65.93     0.8           0.75  lhsinit_
     USR      76,367,538    87,475,200    59.43     0.7           0.68  binvrhs_
     USR      56,913,688    68,892,672    24.62     0.3           0.36  exact_solution_
```



More than
5 GB just for these
6 regions

BT-MZ summary analysis score

- Summary measurement analysis score reveals
  - Total size of event trace would be ~160 GB
  - Maximum trace buffer size would be ~6 GB per rank
    - smaller buffer would require flushes to disk during measurement resulting in substantial perturbation
  - 99.7% of the trace requirements are for USR regions
    - purely computational routines never found on COM call-paths common to communication routines or OpenMP parallel regions
  - These USR regions contribute around 49% of total time
    - however, much of that is very likely to be measurement overhead for frequently-executed small routines
- Advisable to tune measurement configuration
  - Specify an adequate trace buffer size
  - Specify a filter file listing (USR) regions not to be measured

BT-MZ summary analysis report filtering

```
% cat ../config/scorep.filt
SCOREP_REGION_NAMES_BEGIN
  EXCLUDE
    binvcrhs*
    matmul_sub*
    matvec_sub*
    exact_solution*
    binvrhs*
    lhs*init*
    timer_*
SCOREP_REGION_NAMES_END

% scorep-score -f ../config/scorep.filt -c 2 \
      scorep_bt-mz_sum/profile.cubex

Estimated aggregate size of event trace:                   1156 MB
Estimated requirements for largest trace buffer (max_buf):   41 MB
Estimated memory requirements (SCOREP_TOTAL_MEMORY):         49 MB
(hint: When tracing set SCOREP_TOTAL_MEMORY=49MB to avoid \
>intermediate flushes
 or reduce requirements using USR regions filters.)
```

- Report scoring with prospective filter listing 6 USR regions

1,1 GB of memory in total, 49 MB per rank!

(Including 2 metric values)

BT-MZ summary analysis report filtering

```
% scorep-score -r -f ../config/scorep.filt \
      scorep_bt-mz_sum/profile.cubex
flt type     max_buf[B]         visits  time[s] time[%] time/       region
                                                        visit[us]
 -   ALL 5,421,104,056 6,586,922,497  8162.56   100.0        1.24  ALL
 -   USR 5,407,570,350 6,574,832,225  3960.99    48.5        0.60  USR
 -   OMP    15,783,372    10,975,232  4085.92    50.1      372.29  OMP
 -   MPI       944,200       386,560    92.05     1.1      238.13  MPI
 -   COM       665,210       728,480    23.60     0.3       32.40  COM

 *   ALL    17,390,726    12,138,209  4201.91    51.5      346.17  ALL-FLT
 +   FLT 5,407,531,376 6,574,784,288  3960.65    48.5        0.60  FLT
 -   OMP    15,783,372    10,975,232  4085.92    50.1      372.29  OMP-FLT
 -   MPI       944,200       386,560    92.05     1.1      238.13  MPI-FLT
 *   COM       665,210       728,480    23.60     0.3       32.40  COM-FLT
 *   USR        38,974        47,937     0.34     0.0        7.14  USR-FLT

 +   USR 1,741,005,318 2,110,313,472  1204.11    14.8        0.57  matmul_sub_
 +   USR 1,741,005,318 2,110,313,472   851.97    10.4        0.40  matvec_sub_
 +   USR 1,741,005,318 2,110,313,472  1754.58    21.5        0.83  binvcrhs_
 +   USR    76,367,538    87,475,200    65.93     0.8        0.75  lhsinit_
 +   USR    76,367,538    87,475,200    59.43     0.7        0.68  binvrhs_
 +   USR    56,913,688    68,892,672    24.62     0.3        0.36  exact_solution_
```

- Score report breakdown by region

Filtered routines marked with '+'

BT-MZ filtered summary measurement

```
% cd bin.scorep
% cp ../jobscript/stampede2/scorep.sbatch .
% vim scorep.sbatch

# Score-P measurement configuration
export SCOREP_EXPERIMENT_DIRECTORY=scorep_bt-mz_sum_filter
export SCOREP_FILTERING_FILE=../config/scorep.filt
#export SCOREP_TOTAL_MEMORY=50M
#export SCOREP_METRIC_PAPI=PAPI_TOT_INS,PAPI_TOT_CYC
#export SCOREP_ENABLE_TRACING=true

# Run the application
ibrun ./bt-mz_${CLASS}.${PROCS}

% sbatch ./scorep.sbatch
```

- Set new experiment directory and re-run measurement with new filter configuration

- Submit job

Score-P filtering

```
% cat ../config/scorep.filt
SCOREP_REGION_NAMES_BEGIN
  EXCLUDE
    binvcrhs*
    matmul_sub*
    matvec_sub*
    exact_solution*
    binvrhs*
    lhs*init*
    timer_*
SCOREP_REGION_NAMES_END

% export SCOREP_FILTERING_FILE=\
../config/scorep.filt
```

Region name filter block using wildcards

Apply filter

- Filtering by source file name
  - All regions in files that are excluded by the filter are ignored
- Filtering by region name
  - All regions that are excluded by the filter are ignored
  - Overruled by source file filter for excluded files
- Apply filter by
  - exporting **`SCOREP_FILTERING_FILE`** environment variable
- Apply filter at
  - Run-time
  - Compile-time (GCC-plugin only)
    - Add cmd-line option **`--instrument-filter`**
    - No overhead for filtered regions but recompilation

Source file name filter block

- Keywords
  - Case-sensitive
  - `SCOREP_FILE_NAMES_BEGIN`, `SCOREP_FILE_NAMES_END`
    - Define the source file name filter block
    - Block contains `EXCLUDE`, `INCLUDE` rules
  - `EXCLUDE`, `INCLUDE` rules
    - Followed by one or multiple white-space separated source file names
    - Names can contain bash-like wildcards `*`, `?`, `[]`
    - Unlike bash, `*` may match a string that contains slashes
- `EXCLUDE`, `INCLUDE` rules are applied in sequential order
- Regions in source files that are excluded after all rules are evaluated, get filtered

```
# This is a comment
SCOREP_FILE_NAMES_BEGIN
  # by default, everything is included
  EXCLUDE */foo/bar*
  INCLUDE */filter_test.c
SCOREP_FILE_NAMES_END
```

Region name filter block

- Keywords
  - Case-sensitive
  - `SCOREP_REGION_NAMES_BEGIN`, `SCOREP_REGION_NAMES_END`
    - Define the region name filter block
    - Block contains `EXCLUDE`, `INCLUDE` rules
  - `EXCLUDE`, `INCLUDE` rules
    - Followed by one or multiple white-space separated region names
    - Names can contain bash-like wildcards `*`, `?`, `[]`
- `EXCLUDE`, `INCLUDE` rules are applied in sequential order
- Regions that are excluded after all rules are evaluated, get filtered

```
# This is a comment
SCOREP_REGION_NAMES_BEGIN
  # by default, everything is included
  EXCLUDE *
  INCLUDE bar foo
          baz
          main
SCOREP_REGION_NAMES_END
```

Region name filter block, mangling

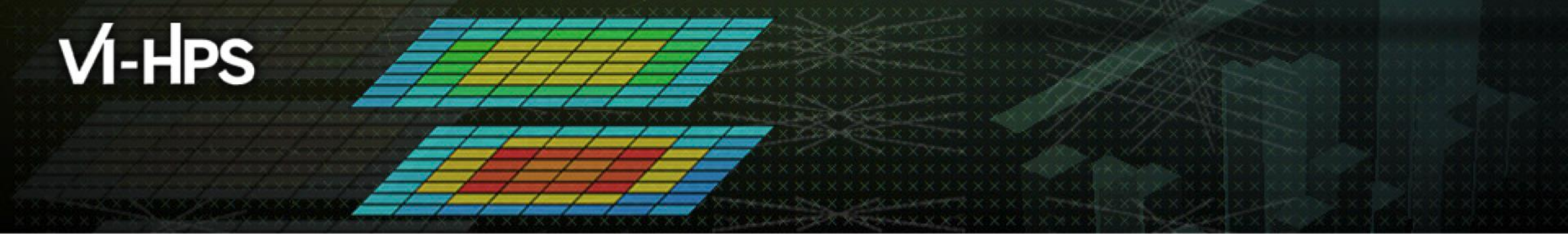
- Name mangling
  - Filtering based on names seen by the measurement system
    - Dependent on compiler
    - Actual name may be mangled
- `scorep-score` names as starting point (e.g. `matvec_sub_`)
    - Use `*` for Fortran trailing underscore(s) for portability
    - Use `?` and `*` as needed for full signatures or overloading

```
void bar(int* a) {
    *a++;
}
int main() {
    int i = 42;
    bar(&i);
    return 0;
}
```

```
# filter bar:
# for gcc-plugin, scorep-score
# displays 'void bar(int*)',
# other compilers may differ

SCOREP_REGION_NAMES_BEGIN
  EXCLUDE void?bar(int?)
SCOREP_REGION_NAMES_END
```

Further information

- Community instrumentation & measurement infrastructure
  - Instrumentation (various methods)
  - Basic and advanced profile generation
  - Event trace recording
  - Online access to profiling data
- Available under 3-clause BSD open-source license
- Documentation & Sources:
  - http://www.score-p.org
- User guide also part of installation:
  - `<prefix>/share/doc/scorep/{pdf,html}/`
- Support and feedback: support@score-p.org
- Subscribe to news@score-p.org, to be up to date



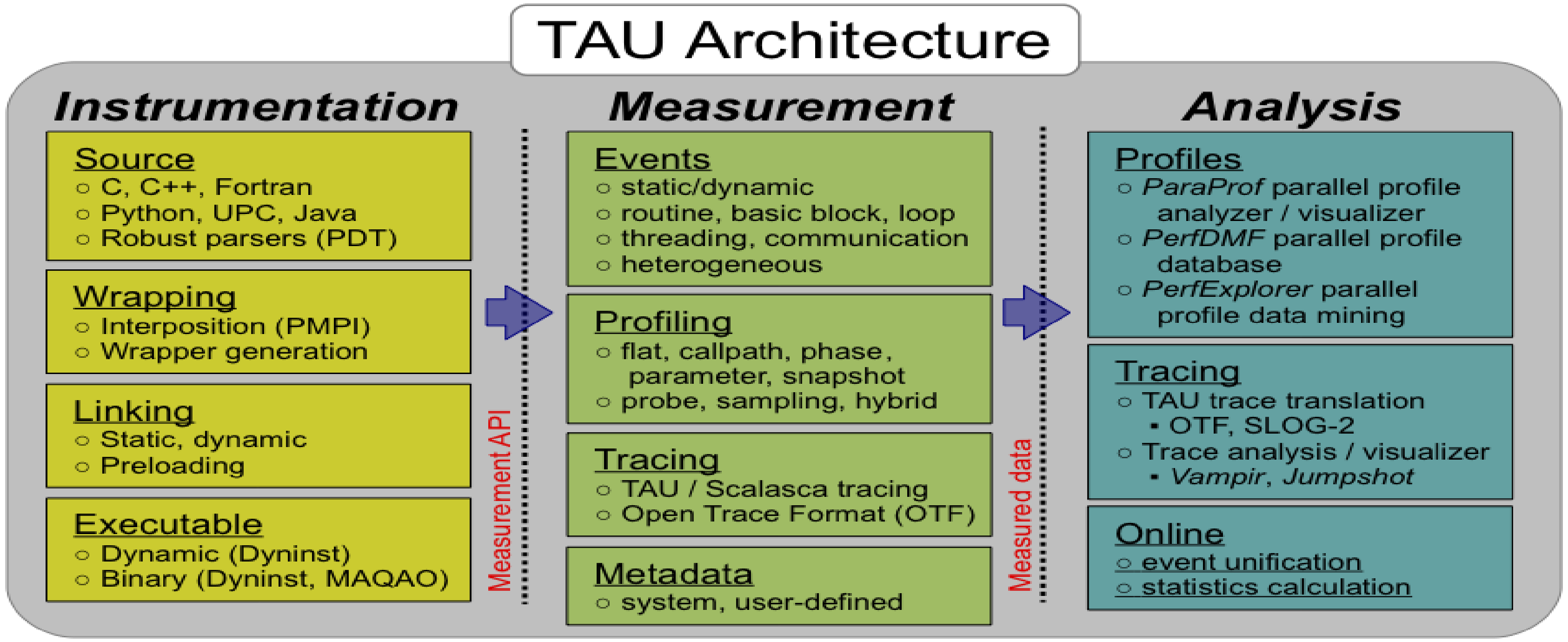
Examination and Visualization of profiles with TAU

Tuning and Analysis Utilities

Sameer Shende
sameer@cs.uoregon.edu
University of Oregon
http://tau.uoregon.edu

TAU Performance System®

- Parallel performance framework and toolkit
  - Supports all HPC platforms, compilers, runtime system
  - Provides portable instrumentation, measurement, analysis



TAU Performance System

- Instrumentation
  - Fortran, C++, C, UPC, Java, Python, Chapel
  - Automatic instrumentation
- Measurement and analysis support
  - MPI, OpenSHMEM, ARMCI, PGAS, DMAPP
  - pthreads, OpenMP, OMPT interface, hybrid, other thread models
  - GPU, CUDA, OpenCL, OpenACC
  - Parallel profiling and tracing
  - Use of Score-P for native OTF2 and CUBEX generation
  - Efficient callpath proflles and trace generation using Score-P
- Analysis
  - Parallel profile analysis (ParaProf), data mining (PerfExplorer)
  - Performance database technology (TAUdb)
  - 3D profile browser

TAU Performance System

- TAU supports both sampling and direct instrumentation
- Memory debugging as well as I/O performance evaluation
- Profiling as well as tracing
- Interfaces with Score-P for more efficient measurements
- TAU's instrumentation covers:
  - Runtime library interposition (tau_exec)
  - Compiler-based instrumentation
  - Native generation of OTF2 traces (TAU_TRACE=1, TAU_TRACE_FORMAT=otf2)
  - Callsite instrumentation with profiles and traces (TAU_CALLSITE=1)
  - PDT based Source level instrumentation: routine & loop
  - Event based sampling (TAU_SAMPLING=1 or tau_exec -ebs)
  - Callstack unwinding with sampling (TAU_EBS_UNWIND=1)
  - OpenMP Tools Interface (OMPT, tau_exec –T ompt)
  - CUDA CUPTI, OpenCL (tau_exec -T cupti -cupti)

Application Performance Engineering using TAU

- How much time is spent in each application routine and outer *loops*? Within loops, what is the contribution of each *statement*? What is the time spent in OpenMP loops?
- How many instructions are executed in these code regions? Floating point, Level 1 and 2 *data cache misses*, hits, branches taken? What is the extent of vectorization for loops on Intel MIC?
- What is the memory usage of the code? When and where is memory allocated/de-allocated? Are there any memory leaks? What is the memory footprint of the application? What is the memory high water mark?
- How much energy does the application use in Joules? What is the peak power usage?
- What are the I/O characteristics of the code? What is the peak read and write *bandwidth* of individual calls, total volume?
- What is the contribution of each *phase* of the program? What is the time wasted/spent waiting for collectives, and I/O operations in Initialization, Computation, I/O phases?
- How does the application *scale*? What is the efficiency, runtime breakdown of performance across different core counts?

Using TAU

- TAU supports several compilers, measurement, and thread options
  - Intel compilers, profiling with hardware counters using PAPI, MPI library, CUDA…
  - Each measurement configuration of TAU corresponds to a unique stub makefile (configuration file) and library that is generated when you configure it
- To instrument source code automatically using PDT
  - Choose an appropriate TAU stub makefile in <arch>/lib:

```
% module load tau
% export TAU_MAKEFILE=$TAU/Makefile.tau-icpc-papi-ompt-mpi-pdt-openmp
% export TAU_OPTIONS='-optVerbose …' (see tau_compiler.sh )
```

  - Use tau_f90.sh, tau_cxx.sh, tau_upc.sh, or tau_cc.sh as F90, C++, UPC, or C compilers respectively:

```
% mpif90 foo.f90        changes to
% tau_f90.sh foo.f90
```

- Set runtime environment variables, execute application and analyze performance data:

```
% pprof   (for text based profile display)
% paraprof  (for GUI)
```

Installing and Configuring TAU

- Installing PDT:
  - wget http://tau.uoregon.edu/pdt_lite.tgz
  - ./configure –prefix=<dir>; make ; make install
- Installing TAU:
  - wget http://tau.uoregon.edu/tau.tgz
  - ./configure –scorep=download –arch=x86_64 -bfd=download -pdt=<dir> -papi=<dir> ...
  - For MIC (KNC):
  - ./configure -scorep=download –arch=mic_linux –pdt=<dir> -pdt_c++=g++ -papi=dir …
  - make install
- Using TAU:
  - export TAU_MAKEFILE=<taudir>/x86_64/lib/Makefile.tau-<TAGS>
  - make CC=tau_cc.sh   CXX=tau_cxx.sh   F90=tau_f90.sh

Different Makefiles for TAU Compiler and Runtime Options

```
%  . ~tg828282/Tutorial/vihps.sh
% ls $TAU/Makefile.*
Makefile.tau-icpc-papi-mpi-pdt
Makefile.tau-icpc-papi-mpi-pdt-openmp-opari
Makefile.tau-icpc-papi-ompt-mpi-pdt-openmp
Makefile.tau-icpc-papi-mpi-pdt-openmp-opari-scorep
Makefile.tau-icpc-papi-mpi-pdt-scorep
Makefile.tau-icpc-papi-ompt-mpi-pdt-openmp
```

- For an MPI+OpenMP+F90 application with Intel MPI, you may choose

  Makefile.tau-icpc-papi-mpi-pdt
  - Supports MPI instrumentation & PDT for automatic source instrumentation

```
% export TAU_MAKEFILE=$TAU/Makefile.tau-icpc-papi-ompt-mpi-pdt-openmp

% tau_f90.sh matmult.f90 -o matmult
% mpirun -np 256 ./matmult
% paraprof
```

Compile-Time Options

Optional parameters for the TAU_OPTIONS environment variable:

% tau_compiler.sh

-optVerbose	Turn on verbose debugging messages
-optCompInst	Use compiler based instrumentation
-optNoCompInst	Do not revert to compiler instrumentation if source instrumentation fails.
-optTrackIO	Wrap POSIX I/O call and calculates vol/bw of I/O operations (configure TAU with –*iowrapper*)
-optTrackGOMP	Enable tracking GNU OpenMP runtime layer (used without –opari)
-optMemDbg	Enable runtime bounds checking (see TAU_MEMDBG_* env vars)
-optKeepFiles	Does not remove intermediate .pdb and .inst.* files
-optPreProcess	Preprocess sources (OpenMP, Fortran) before instrumentation
-optTauSelectFile=”<file>"	Specify selective instrumentation file for *tau_instrumentor*
-optTauWrapFile=”<file>"	Specify path to *link_options.tau* generated by *tau_gen_wrapper*
-optHeaderInst	Enable Instrumentation of headers
-optTrackUPCR	Track UPC runtime layer routines (used with tau_upc.sh)
-optLinking=""	Options passed to the linker. Typically $(TAU_MPI_FLIBS) $(TAU_LIBS) $(TAU_CXXLIBS)
-optCompile=""	Options passed to the compiler. Typically $(TAU_MPI_INCLUDE) $(TAU_INCLUDE) $(TAU_DEFS)
-optPdtF95Opts=""	Add options for Fortran parser in PDT (f95parse/gfparse) …

Compile-Time Options (contd.)

▪Optional parameters for the TAU_OPTIONS environment variable:

% tau_compiler.sh

-optMICOffload	Links code for Intel MIC offloading, requires both host and MIC TAU libraries
-optShared	Use TAU's shared library (libTAU.so) instead of static library (default)
-optPdtCxxOpts=“”	Options for C++ parser in PDT (cxxparse).
-optPdtF90Parser=“”	Specify a different Fortran parser
-optPdtCleanscapeParser	Specify the Cleanscape Fortran parser instead of GNU gfparser
-optTau=“”	Specify options to the tau_instrumentor
-optTrackDMAPP	Enable instrumentation of low-level DMAPP API calls on Cray
-optTrackPthread	Enable instrumentation of pthread calls

See tau_compiler.sh for a full list of TAU_OPTIONS.

…

Compiling Fortran Codes with TAU

- If your Fortran code uses free format in .f files (fixed is default for .f), you may use:

```
% export TAU_OPTIONS='-optPdtF95Opts="-R free" -optVerbose '
```

- To use the compiler based instrumentation instead of PDT (source-based):

```
% export TAU_OPTIONS='-optCompInst -optVerbose'
```

- If your Fortran code uses C preprocessor directives (#include, #ifdef, #endif):

```
% export TAU_OPTIONS='-optPreProcess -optVerbose -optDetectMemoryLeaks'
```

- To use an instrumentation specification file:

```
% export TAU_OPTIONS='-optTauSelectFile=select.tau -optVerbose -optPreProcess'
% cat select.tau
BEGIN_INSTRUMENT_SECTION
loops  routine="#"
# this statement instruments all outer loops in all routines. # is wildcard as well as comment in first column.
END_INSTRUMENT_SECTION
```

Runtime Environment Variables

Environment Variable	Default	Description
TAU_TRACE	0	Setting to 1 turns on tracing
TAU_TRACE_FORMAT	default	Setting to “otf2” generates OTF2 traces natively.
TAU_CALLPATH	0	Setting to 1 turns on callpath profiling
TAU_CALLSITE	0	Setting to 1 generates callsite information in events. May be used with tracing.
TAU_TRACK_MEMORY_FOOTPRINT	0	Setting to 1 turns on tracking memory usage by tracking the resident set size and high water mark of memory usage
TAU_TRACK_LOAD	0	Setting to 1 tracks system load periodically.
TAU_CALLPATH_DEPTH	2	Specifies depth of callpath. Setting to 0 generates no callpath or routine information, setting to 1 generates flat profile and context events have just parent information (e.g., Heap Entry: foo)
TAU_SAMPLING	1	Setting to 1 enables event-based sampling.
TAU_TRACK_SIGNALS	0	Setting to 1 generate debugging callstack info when a program crashes
TAU_COMM_MATRIX	0	Setting to 1 generates communication matrix display using context events
TAU_THROTTLE	1	Setting to 0 turns off throttling. Throttles instrumentation in lightweight routines that are called frequently
TAU_THROTTLE_NUMCALLS	100000	Specifies the number of calls before testing for throttling
TAU_THROTTLE_PERCALL	10	Specifies value in microseconds. A routine is throttled if it takes less than 10 microseconds per call (and called > 10000 times).
TAU_COMPENSATE	0	Setting to 1 enables runtime compensation of instrumentation overhead
TAU_PROFILE_FORMAT	Profile	Setting to “merged” generates a single file. “snapshot” generates xml format
TAU_METRICS	TIME	Setting to a comma separated list generates other metrics. (e.g., ENERGYTIME,P_VIRTUAL_TIME,PAPI_FP_INS,PAPI_NATIVE_<event>:<subevent>)

Runtime Environment Variables (contd.)

Environment Variable	Default	Description
TAU_TRACK_MEMORY_LEAKS	0	Tracks allocates that were not de-allocated (needs –optMemDbg or tau_exec –memory)
TAU_EBS_SOURCE	TIME	Allows using PAPI hardware counters for periodic interrupts for EBS (e.g., TAU_EBS_SOURCE=PAPI_TOT_INS when TAU_SAMPLING=1)
TAU_EBS_PERIOD	100000	Specifies the overflow count for interrupts
TAU_MEMDBG_ALLOC_MIN/MAX	0	Byte size minimum and maximum subject to bounds checking (used with TAU_MEMDBG_PROTECT_*)
TAU_MEMDBG_OVERHEAD	0	Specifies the number of bytes for TAU's memory overhead for memory debugging.
TAU_MEMDBG_PROTECT_BELOW/ABOVE	0	Setting to 1 enables tracking runtime bounds checking below or above the array bounds (requires –optMemDbg while building or tau_exec –memory)
TAU_MEMDBG_ZERO_MALLOC	0	Setting to 1 enables tracking zero byte allocations as invalid memory allocations.
TAU_MEMDBG_PROTECT_FREE	0	Setting to 1 detects invalid accesses to deallocated memory that should not be referenced until it is reallocated (requires –optMemDbg or tau_exec –memory)
TAU_MEMDBG_ATTEMPT_CONTINUE	0	Setting to 1 allows TAU to record and continue execution when a memory error occurs at runtime.
TAU_MEMDBG_FILL_GAP	Undefined	Initial value for gap bytes
TAU_MEMDBG_ALINGMENT	Sizeof(int)	Byte alignment for memory allocations
TAU_EVENT_THRESHOLD	0.5	Define a threshold value (e.g., .25 is 25%) to trigger marker events for min/max

Simplifying TAU's usage (tau_exec)

- Uninstrumented execution
  - % mpirun -np 4 ./a.out
- Track MPI performance
  - % mpirun -np 4   tau_exec ./a.out
- Track POSIX I/O and MPI performance (MPI enabled by default)
  - % mpirun -np 4 tau_exec –T mpi,pdt   –io ./a.out
- Track memory operations
  - % export TAU_TRACK_MEMORY_LEAKS=1
  - % mpirun –np 8 tau_exec –memory_debug ./a.out (bounds check)
- Use event based sampling (compile with –g)
  - % mpirun –np 8 tau_exec –ebs ./a.out
  - Also –ebs_source=<PAPI_COUNTER> -ebs_period=<overflow_count>
- Load wrapper interposition library
  - % mpirun –np 8 tau_exec –loadlib=<path/libwrapper.so> ./a.out
- Track GPGPU operations
  - % mpirun –np 8 tau_exec –cupti ./a.out
  - % mpirun –np 8 tau_exec –opencl ./a.out
  - % mpirun –np 8 tau_exec –openacc ./a.out

Binary Rewriting Instrumentation

- Support for both static and dynamic executables
- Specify a list of routines to instrument
- Specify the TAU measurement library to be injected
- MAQAO [UVSQ, Intel Exascale Labs]:

```
% tau_rewrite –T [tags] a.out –o a.inst
```

- DyninstAPI [U. Maryland and U. Wisconsin, Madison]:

```
% tau_run –T [tags] a.out –o a.inst
```

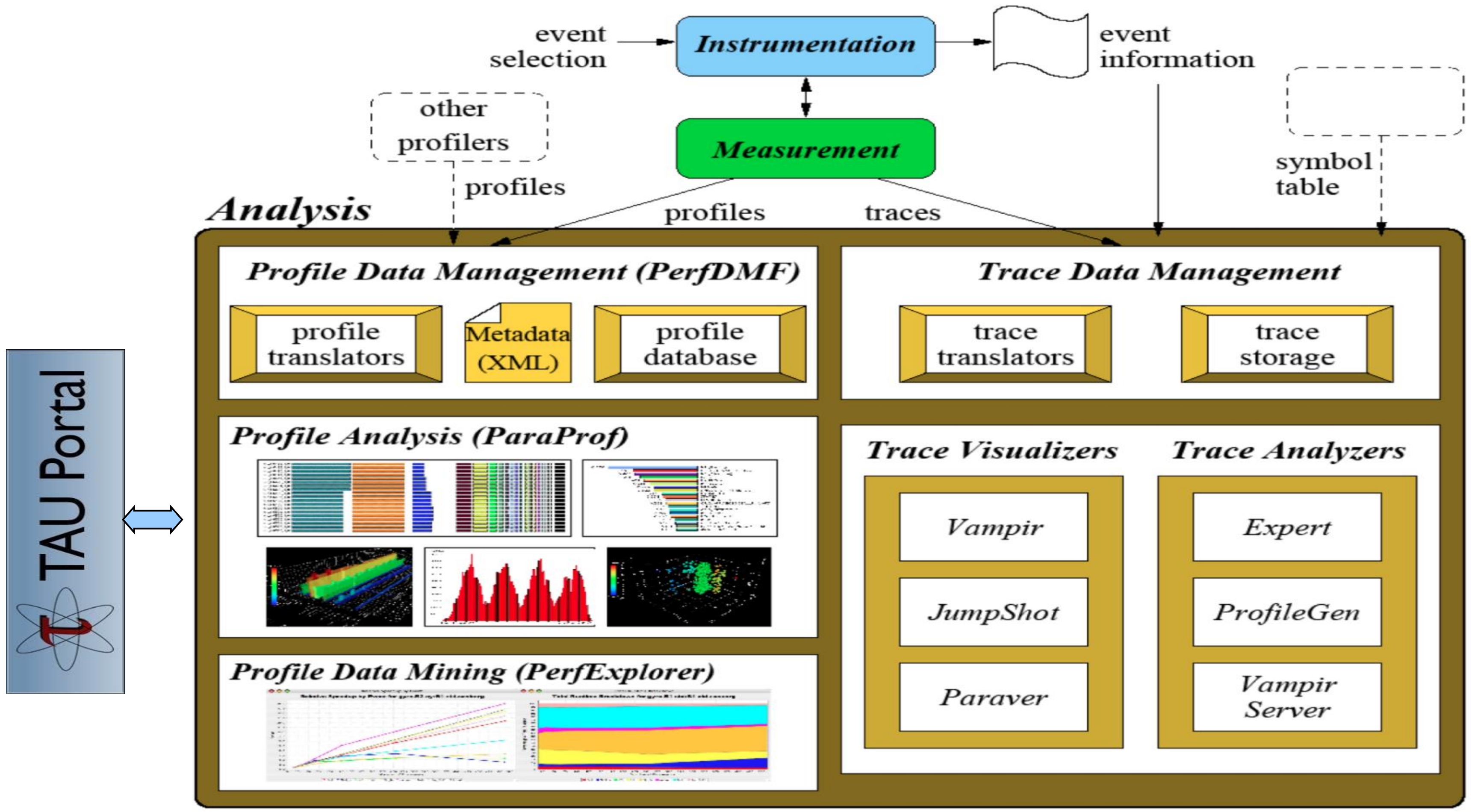
- Pebil [UC San Diego]:

```
% tau_pebil_rewrite –T [tags] a.out –o a.inst
```

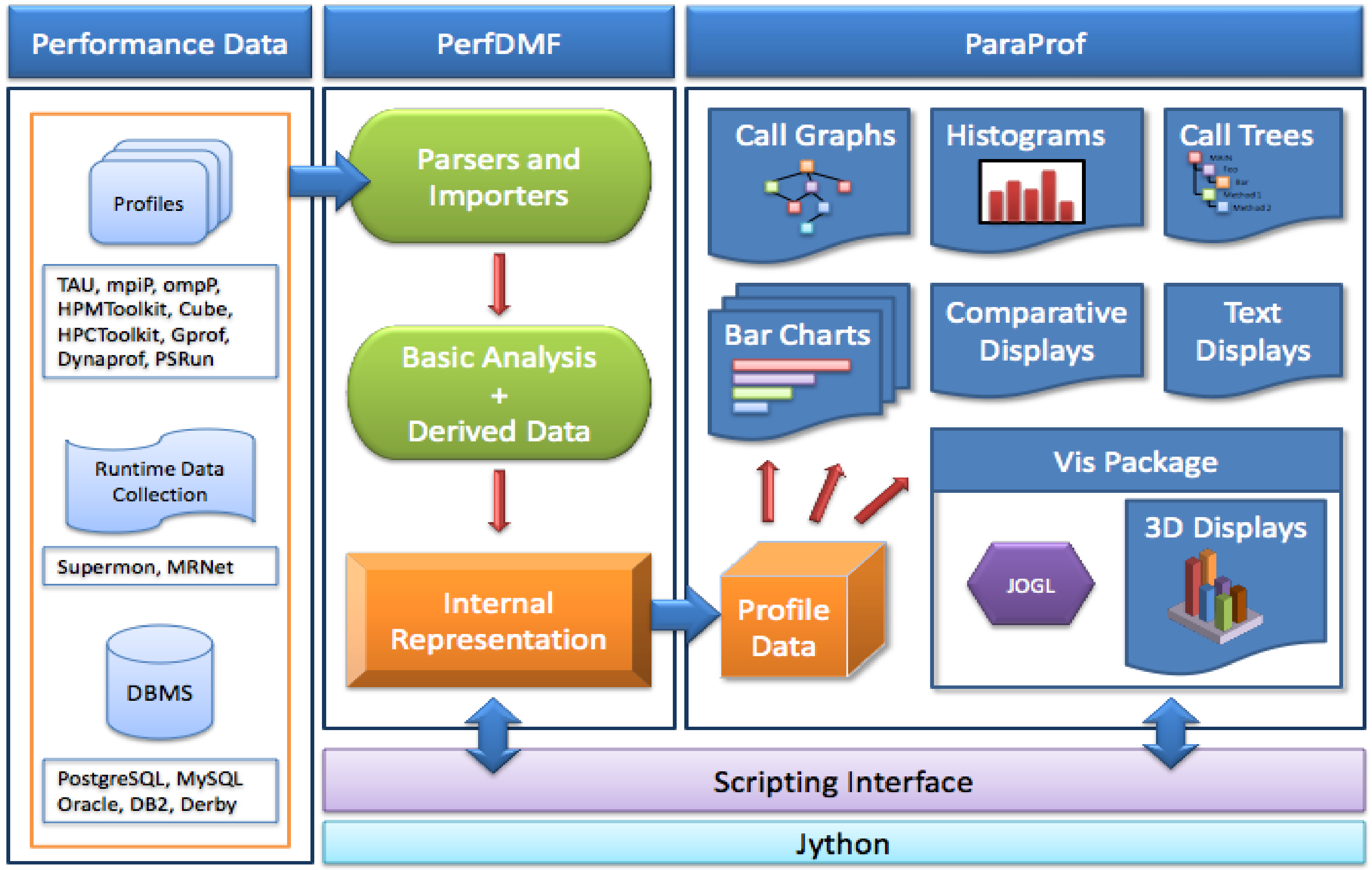
- Execute the application to get measurement data:

```
% mpirun –np 256 ./a.inst
```

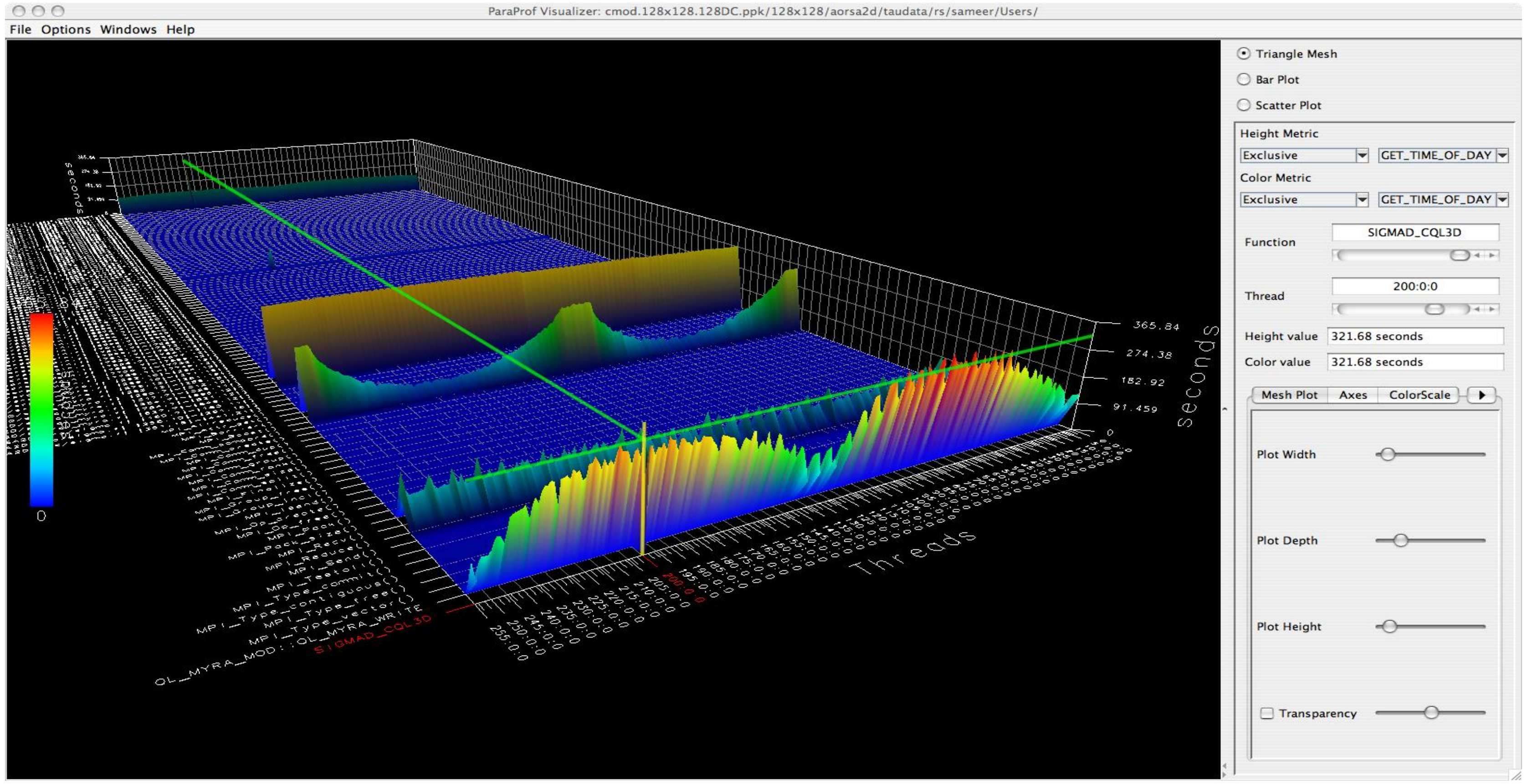
TAU Analysis



ParaProf Profile Analysis Framework



Parallel Profile Visualization: ParaProf



ParaProf Visualizer: cmod.128x128.128DC.ppk/128x128/aorsa2d/taudata/rs/sameer/Users/

File Options Windows Help

[✓ Triangle Mesh] Bar Plot Scatter Plot

Height Metric: Exclusive GET_TIME_OF_DAY

Color Metric: Exclusive GET_TIME_OF_DAY

Function: SIGMAD_CQL3D

Thread: 200:0:0

Height value: 321.68 seconds

Color value: 321.68 seconds

Mesh Plot | Axes | ColorScale

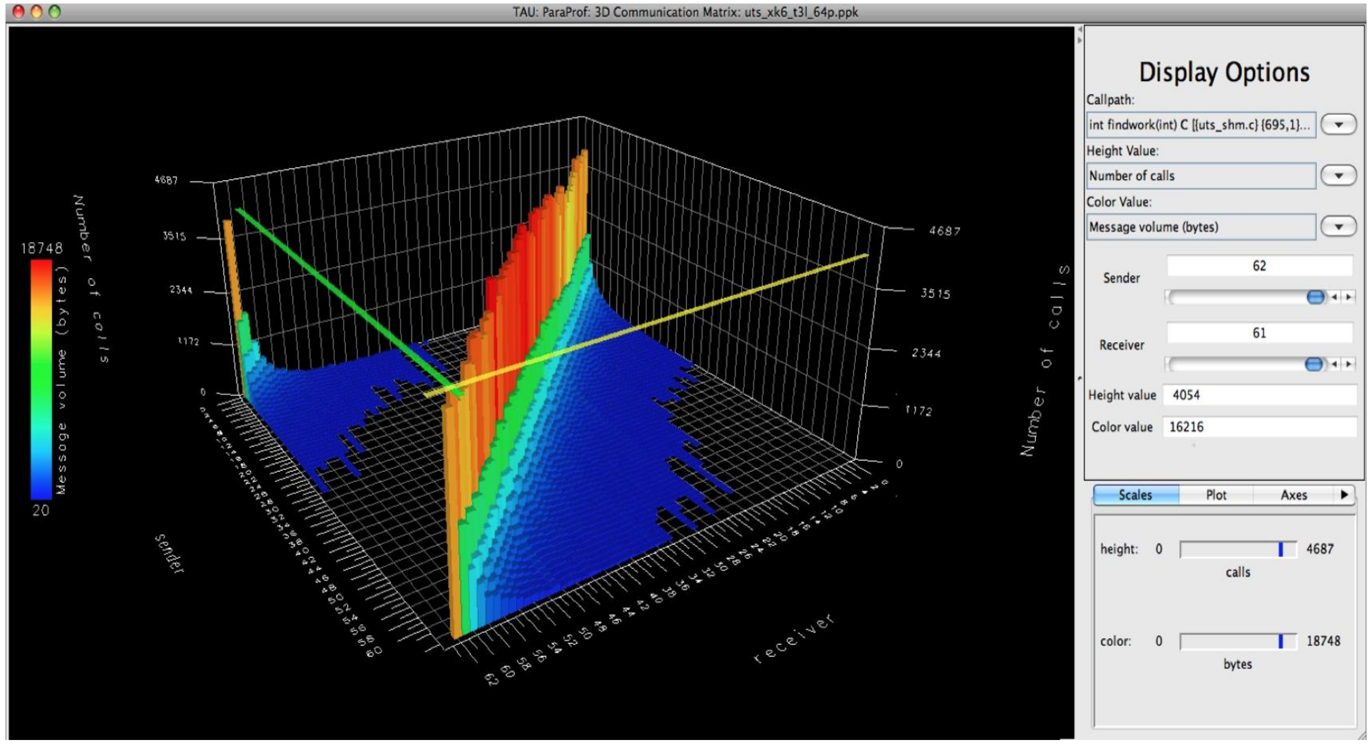
Plot Width

Plot Depth

Plot Height

Transparency

ParaProf 3D Communication Matrix



Display Options

Callpath: int findwork(int) C [{uts_shm.c} {695,1}...

Height Value: Number of calls

Color Value: Message volume (bytes)

Sender: 62

Receiver: 61

Height value: 4054

Color value: 16216

height: 0 4687 calls

color: 0 18748 bytes

```
% export TAU_COMM_MATRIX=1
```

TAU tutorial exercise objectives

- Familiarise with usage of TAU tools
  - complementary tools' capabilities & interoperability
- Prepare to apply tools productively to *your* applications(s)
- Exercise is based on a small portable benchmark code
  - unlikely to have significant optimisation opportunities

- Optional (recommended) exercise extensions
  - analyse performance of alternative configurations
  - investigate effectiveness of system-specific compiler/MPI optimisations and/or placement/binding/affinity capabilities
  - investigate scalability and analyse scalability limiters
  - compare performance on different HPC platforms
  - …

Local Installation (*Stampede, TACC*)

- Setup preferred program environment compilers
  - Default set Intel Compilers with Intel MPI
  - Generate profile files using Score-P

```
% . /home1/03529/tg828282/Tutorial/vihps.sh
% paraprof profile.cubex &
```

For PerfExplorer:

```
% wget http://tau.uoregon.edu/data.tgz; tar zxf data.tgz; cd data
% cat README
And follow the steps
```

NPB-MZ-MPI Suite

- The NAS Parallel Benchmark suite (MPI+OpenMP version)
  - Available from:

    http://www.nas.nasa.gov/Software/NPB
  - 3 benchmarks in Fortran77
  - Configurable for various sizes & classes
- Move into the NPB3.3-MZ-MPI root directory

```
% ls
bin/     common/   jobscript/   Makefile   README.install    SP-MZ/
BT-MZ/   config/   LU-MZ/       README     README.tutorial   sys/
```

- Subdirectories contain source code for each benchmark
  - plus additional configuration and common code
- The provided distribution has already been configured for the tutorial, such that it's ready to “make” one or more of the benchmarks and install them into a (tool-specific) “bin” subdirectory

NPB-MZ-MPI / BT: config/make.def

```
#                 SITE- AND/OR PLATFORM-SPECIFIC DEFINITIONS.
#
#---------------------------------------------------------------------------

#---------------------------------------------------------------------------
# Configured for generic MPI with GCC compiler
#---------------------------------------------------------------------------
#OPENMP  = -fopenmp      # GCC compiler
OPENMP = -openmp         # Intel compiler

...
#---------------------------------------------------------------------
# The Fortran compiler used for MPI programs
#---------------------------------------------------------------------
# MPIF77 = mpiifort # Intel compiler

# Alternative variant to perform instrumentation
MPIF77 = tau_f90.sh -tau_makefile=<path>/Makefile.tau-[options]

# PREP is a generic preposition macro for instrumentation preparation
#MPIF77 = $(PREP) mpif77 -f77=ifort
#MPIF77 = scorep …
...
```

Default (no instrumentation)

Uncomment TAU's compiler wrapper to do source instrumentation with TAU
Comment out Score-P wrapper

Building an NPB-MZ-MPI Benchmark

```
% make
  ===========================================
  =      NAS PARALLEL BENCHMARKS 3.3        =
  =      MPI+OpenMP Multi-Zone Versions     =
  =      F77                                =
  ===========================================

  To make a NAS multi-zone benchmark type

         make <benchmark-name> CLASS=<class> NPROCS=<nprocs>

  where <benchmark-name> is "bt-mz", "lu-mz", or "sp-mz"
        <class>          is "S", "W", "A" through "F"
        <nprocs>         is number of processes

  [...]

 ****************************************************************
 * Custom build configuration is specified in config/make.def   *
 * Suggested tutorial exercise configuration for HPC systems:   *
 *       make bt-mz CLASS=C NPROCS=32                            *
 ****************************************************************
```

- Type "make" for instructions

Building an NPB-MZ-MPI Benchmark

```
% make suite
make[1]: Entering directory `BT-MZ'
make[2]: Entering directory `sys'
cc  -o setparams setparams.c -lm
make[2]: Leaving directory `sys'
../sys/setparams bt-mz 32 C
make[2]: Entering directory `../BT-MZ'
tau_f90.sh -c  -O3 -g -openmp       bt.f
                                      […]
tau_f90.sh -c  -O3 -g -openmp       mpi_setup.f
cd ../common;  mpiifort -c  -O3 -g -openmp        print_results.f
cd ../common;  mpiifort -c  -O3 -g -openmp        timers.f
tau_f90.sh -O3 -g -openmp   -o ../bin.tau/bt-mz_C.8 bt.o
 initialize.o exact_solution.o exact_rhs.o set_constants.o adi.o
 rhs.o zone_setup.o x_solve.o y_solve.o  exch_qbc.o solve_subs.o
 z_solve.o add.o error.o verify.o mpi_setup.o ../common/print_results.o
 ../common/timers.o
make[2]: Leaving directory `BT-MZ'
Built executable ../bin.tau/bt-mz_C.32
make[1]: Leaving directory `BT-MZ'
```

- Specify the benchmark configuration
  - benchmark name: **bt-mz**, lu-mz, sp-mz
  - the number of MPI processes: NPROCS=**3C**
  - the benchmark class (S, W, A, B, C, D, E): CLASS=**C**

Shortcut: `% make suite`

NPB-MZ-MPI / BT (Block Tridiagonal Solver)

- What does it do?
  - Solves a discretized version of the unsteady, compressible Navier-Stokes equations in three spatial dimensions
  - Performs 200 time-steps on a regular 3-dimensional grid
- Implemented in 20 or so Fortran77 source modules

- Uses MPI & OpenMP in combination
  - 2 compute nodes with 1 Intel Xeon Phi 7250 CPU (Knights Landing, KNL) each
  - 32 processes each with 4 OpenMP threads should be reasonable

  - bt-mz_C.32 should take around 30 seconds

TAU Source Instrumentation

- Edit config/make.def to adjust build configuration
  - Uncomment specification of compiler/linker: MPIF77 = tau_f77.sh
- Make clean and build new tool-specific executable

```
% make clean
% make bt-mz CLASS=C NPROCS=32
Built executable ../bin.tau/bt-mz_C.32
```

- Change to the directory containing the new executable before running it with the desired tool configuration

```
% cd bin.tau
% cp ../jobscript/stampede2/tau.sbatch .
% sbatch tau.sbatch
```

NPB-MZ-MPI / BT with TAU

```
% cd bin
% cp ../jobscript/stampede2/tau.sbatch .
% sbatch tau.sbatch
% cat mzmpibt.o<job_id>
 NAS Parallel Benchmarks (NPB3.3-MZ-MPI) - BT-MZ MPI+OpenMP Benchmark
 Number of zones:   16 x 16
 Iterations:  200    dt:   0.000300
 Number of active processes:     32
 Total number of threads:       128 (  4.0 threads/process)

 Time step    1
 Time step   20
  [...]
 Time step  180
 Time step  200
 Verification Successful

 BT-MZ Benchmark Completed.
 Time in seconds = 22.34
% paraprof &
% paraprof --pack bt.ppk
<Copy file over to desktop using scp>
% paraprof bt.ppk  &
```

- Copy jobscript and launch as a hybrid MPI+OpenMP application

Hint: save the benchmark output (or note the run time) to be able to refer to it later

tau_exec

```
$ tau_exec

Usage: tau_exec [options] [--] <exe> <exe options>

Options:
        -v             Verbose mode
        -s             Show what will be done but don't actually do anything (dryrun)
        -qsub          Use qsub mode (BG/P only, see below)
        -io            Track I/O
        -memory        Track memory allocation/deallocation
        -memory_debug Enable memory debugger
        -cuda          Track GPU events via CUDA
        -cupti         Track GPU events via CUPTI (Also see env. variable TAU_CUPTI_API)
        -opencl        Track GPU events via OpenCL
        -openacc       Track GPU events via OpenACC (currently PGI only)
        -ompt          Track OpenMP events via OMPT interface
        -armci         Track ARMCI events via PARMCI
        -ebs           Enable event-based sampling
        -ebs_period=<count> Sampling period (default 1000)
        -ebs_source=<counter> Counter (default itimer)
        -um           Enable Unified Memory events via CUPTI
        -T <DISABLE,GNU,ICPC,MPI,OMPT,OPENMP,PAPI,PDT,PROFILE,PTHREAD,SCOREP,SERIAL> : Specify TAU tags
        -loadlib=<file.so>   : Specify additional load library
        -XrunTAUsh-<options> : Specify TAU library directly
        -gdb           Run program in the gdb debugger

Notes:
           Defaults if unspecified: -T MPI
           MPI is assumed unless SERIAL is specified
```

- Tau_exec preloads the TAU wrapper libraries and performs measurements.

No need to recompile the application!

tau_exec Example (continued)

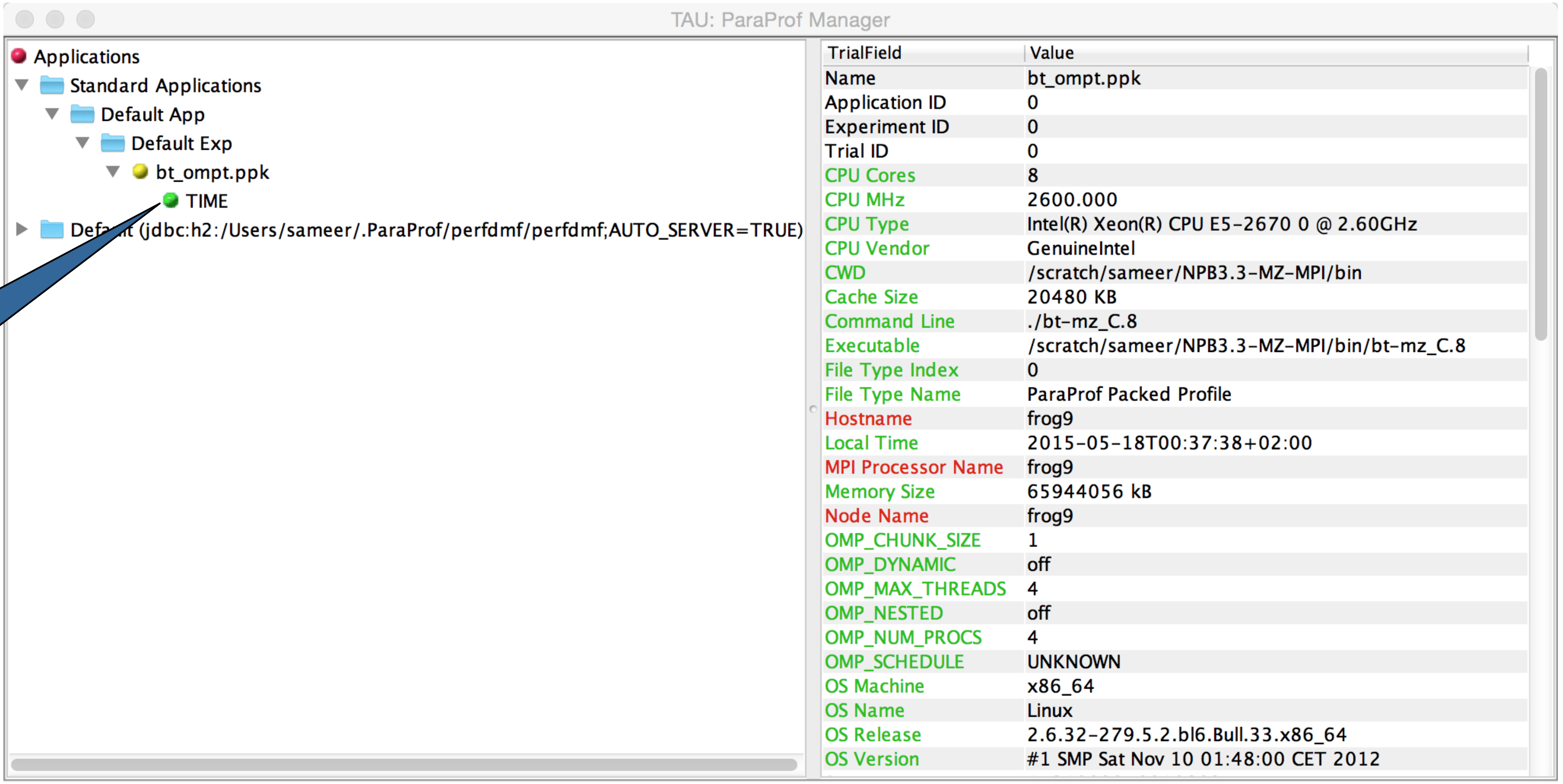
```
Example:
    mpirun -np 2 tau_exec -T icpc,ompt,mpi  -ompt ./a.out
    mpirun -np 2 tau_exec -io ./a.out
Example - event-based sampling with samples taken every 1,000,000 FP instructions
    mpirun -np 8 tau_exec -ebs -ebs_period=1000000 -ebs_source=PAPI_FP_INS ./ring
Examples - GPU:
    tau_exec -T serial,cupti -cupti ./matmult (Preferred for CUDA 4.1 or later)
    tau_exec -openacc ./a.out
    tau_exec -T serial -opencl ./a.out (OPENCL)
    mpirun -np 2 tau_exec -T mpi,cupti,papi -cupti -um ./a.out (Unified Virtual Memory in CUDA 6.0+)

qsub mode (IBM BG/Q only):
    Original:
      qsub -n 1 --mode smp -t 10 ./a.out
    With TAU:
      tau_exec -qsub -io -memory -- qsub -n 1 … -t 10 ./a.out

Memory Debugging:
    -memory option:
      Tracks heap allocation/deallocation and memory leaks.
    -memory_debug option:
      Detects memory leaks, checks for invalid alignment, and checks for
      array overflow.  This is exactly like setting TAU_TRACK_MEMORY_LEAKS=1
      and TAU_MEMDBG_PROTECT_ABOVE=1 and running with -memory
```

- tau_exec can enable event based sampling while launching the executable using the –ebs flag!
- On stampede, you need to put perl-mic/bin in your path
- ibrun.symm –m test.sh
- Within test.sh call tau_exec –T ompt

TAU Analysis Tools: paraprof

- Launch paraprof

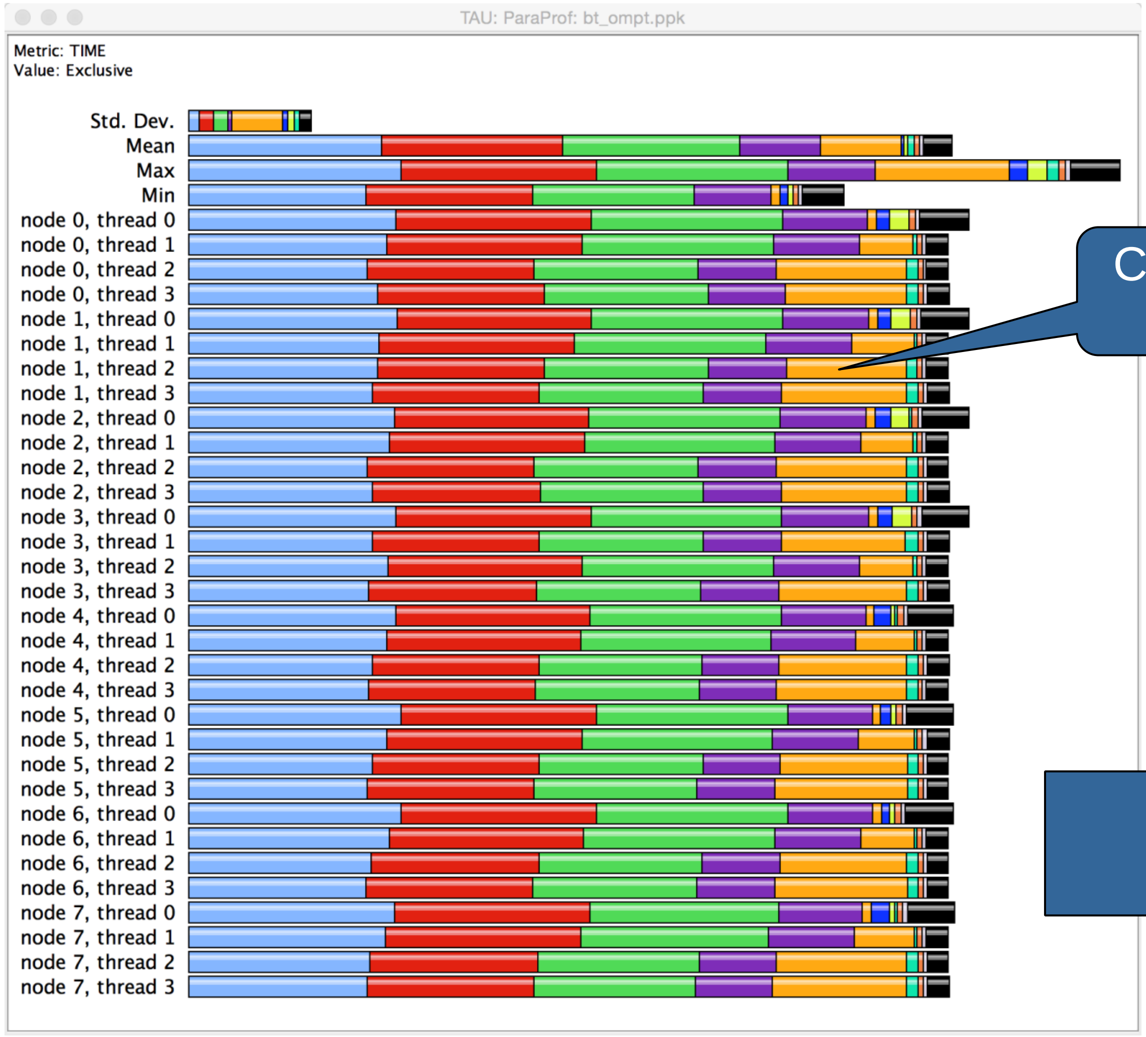
```
% paraprof
```



```
● Applications
▼ Standard Applications
  ▼ Default App
    ▼ Default Exp
      ▼ bt_ompt.ppk
          TIME
▶ Default (jdbc:h2:/Users/sameer/.ParaProf/perfdmf/perfdmf;AUTO_SERVER=TRUE)
```

TrialField	Value
Name	bt_ompt.ppk
Application ID	0
Experiment ID	0
Trial ID	0
CPU Cores	8
CPU MHz	2600.000
CPU Type	Intel(R) Xeon(R) CPU E5-2670 0 @ 2.60GHz
CPU Vendor	GenuineIntel
CWD	/scratch/sameer/NPB3.3-MZ-MPI/bin
Cache Size	20480 KB
Command Line	./bt-mz_C.8
Executable	/scratch/sameer/NPB3.3-MZ-MPI/bin/bt-mz_C.8
File Type Index	0
File Type Name	ParaProf Packed Profile
Hostname	frog9
Local Time	2015-05-18T00:37:38+02:00
MPI Processor Name	frog9
Memory Size	65944056 kB
Node Name	frog9
OMP_CHUNK_SIZE	1
OMP_DYNAMIC	off
OMP_MAX_THREADS	4
OMP_NESTED	off
OMP_NUM_PROCS	4
OMP_SCHEDULE	UNKNOWN
OS Machine	x86_64
OS Name	Linux
OS Release	2.6.32-279.5.2.bl6.Bull.33.x86_64
OS Version	#1 SMP Sat Nov 10 01:48:00 CET 2012

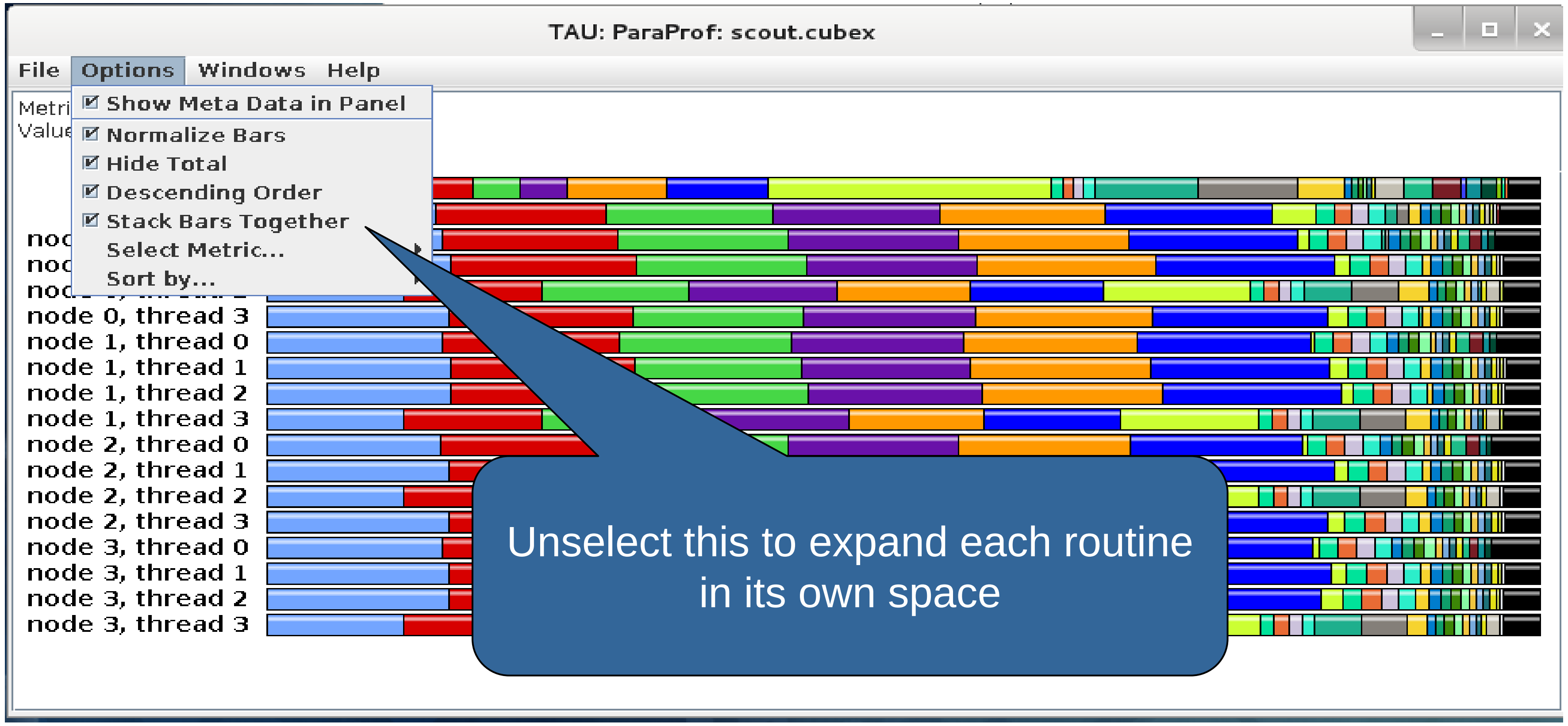
Paraprof main window



Colors represent code regions

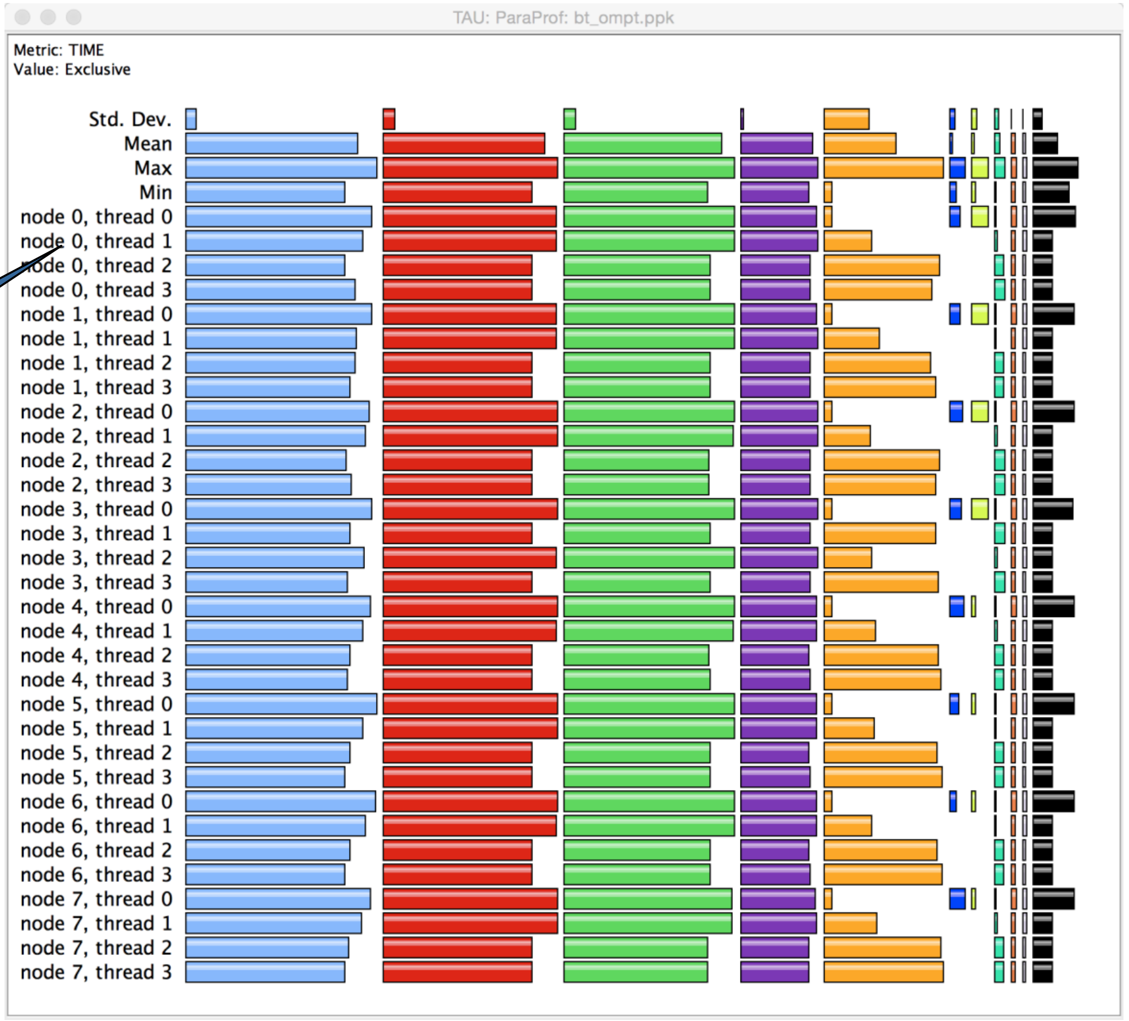
Options -> uncheck Stack Bars Together

Paraprof main window



Unselect this to expand each routine in its own space

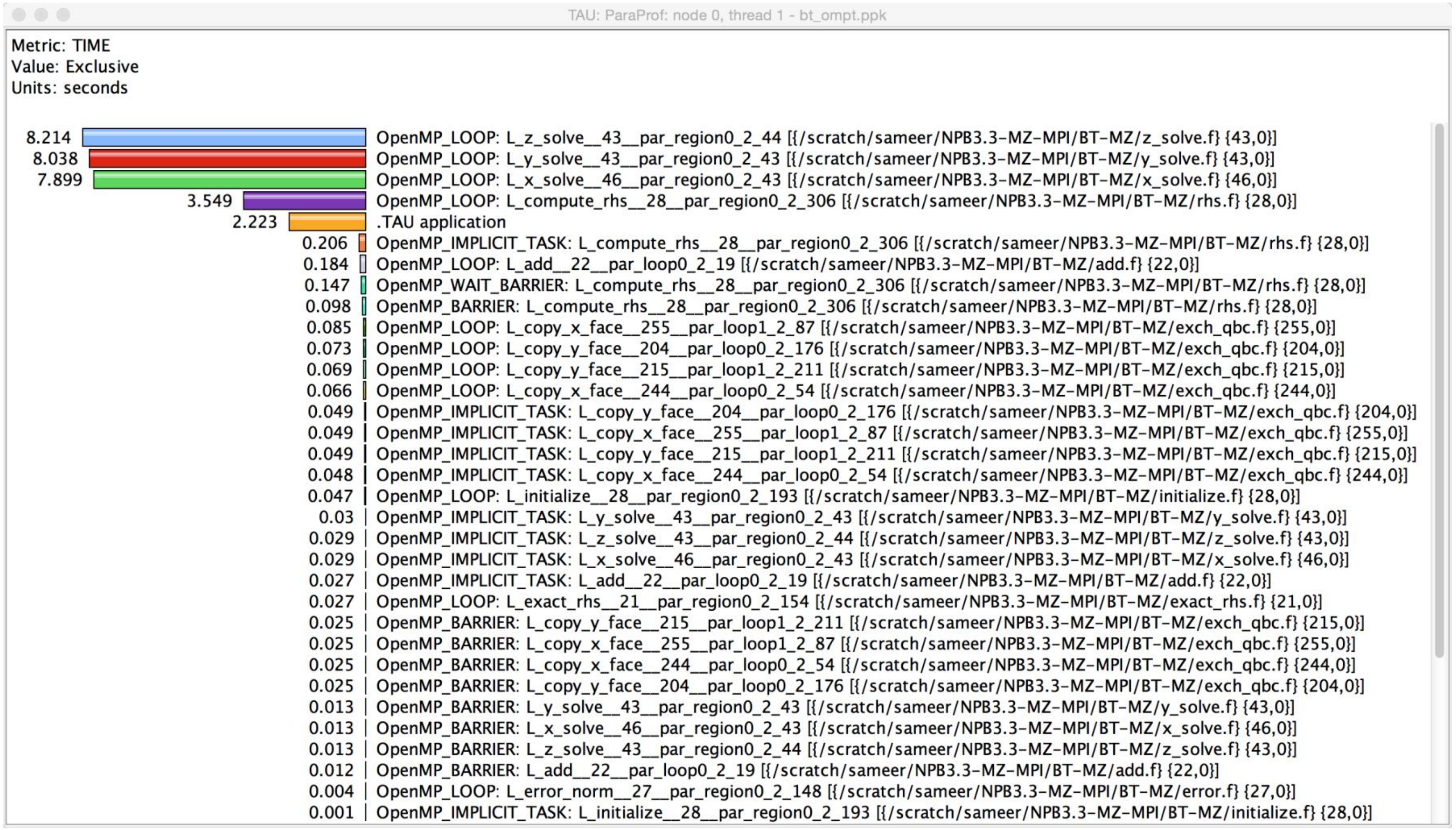
Paraprof main window



Left/right click here

Each routine occupies its own space. Can see the extent of imbalance across all threads.

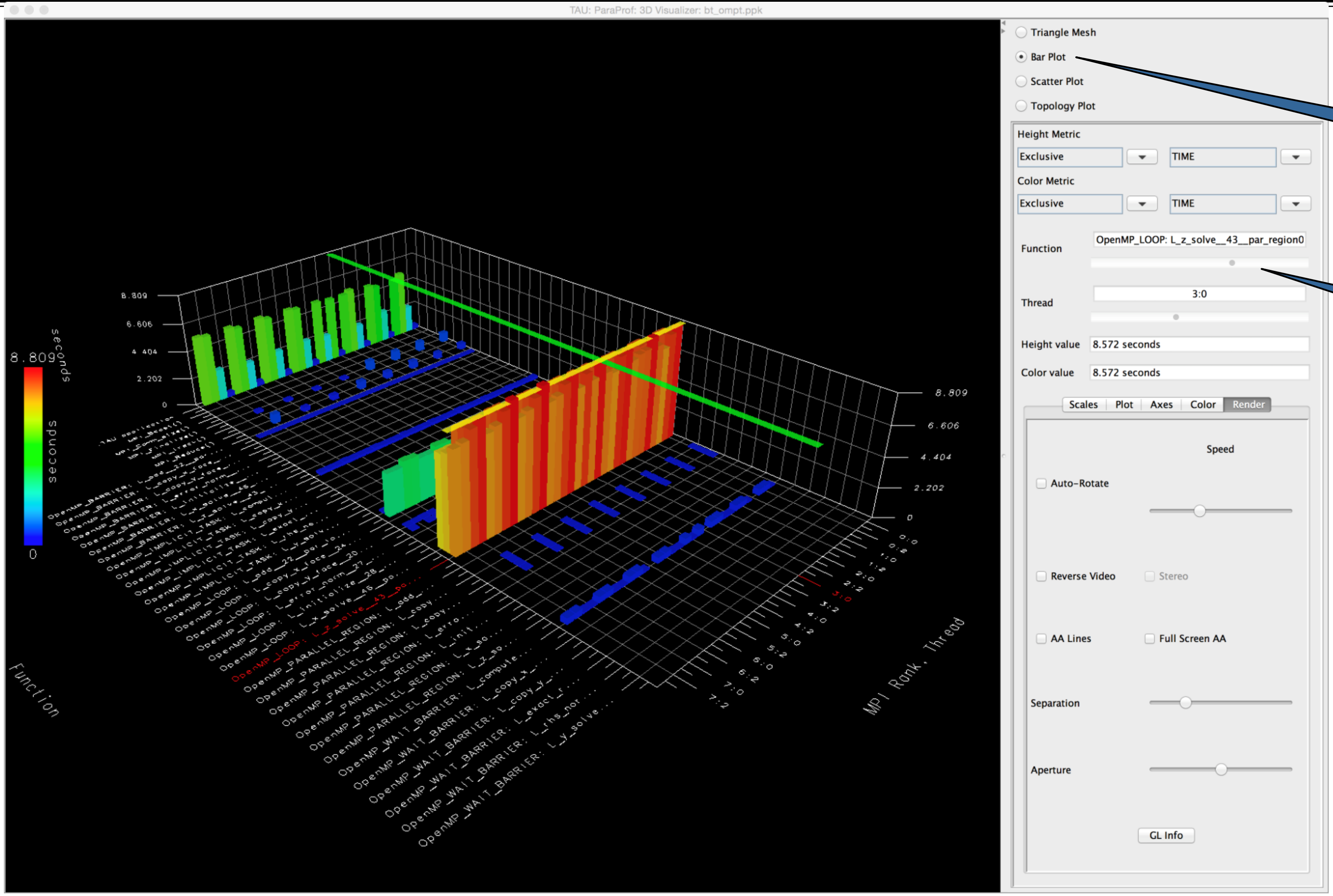
Paraprof node window (function barchart window)



```
Metric: TIME
Value: Exclusive
Units: seconds

8.214  OpenMP_LOOP: L_z_solve__43__par_region0_2_44 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/z_solve.f} {43,0}]
8.038  OpenMP_LOOP: L_y_solve__43__par_region0_2_43 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/y_solve.f} {43,0}]
7.899  OpenMP_LOOP: L_x_solve__46__par_region0_2_43 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/x_solve.f} {46,0}]
3.549  OpenMP_LOOP: L_compute_rhs__28__par_region0_2_306 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/rhs.f} {28,0}]
2.223  .TAU application
0.206  OpenMP_IMPLICIT_TASK: L_compute_rhs__28__par_region0_2_306 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/rhs.f} {28,0}]
0.184  OpenMP_LOOP: L_add__22__par_loop0_2_19 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/add.f} {22,0}]
0.147  OpenMP_WAIT_BARRIER: L_compute_rhs__28__par_region0_2_306 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/rhs.f} {28,0}]
0.098  OpenMP_BARRIER: L_compute_rhs__28__par_region0_2_306 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/rhs.f} {28,0}]
0.085  OpenMP_LOOP: L_copy_x_face__255__par_loop1_2_87 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/exch_qbc.f} {255,0}]
0.073  OpenMP_LOOP: L_copy_y_face__204__par_loop0_2_176 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/exch_qbc.f} {204,0}]
0.069  OpenMP_LOOP: L_copy_y_face__215__par_loop1_2_211 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/exch_qbc.f} {215,0}]
0.066  OpenMP_LOOP: L_copy_x_face__244__par_loop0_2_54 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/exch_qbc.f} {244,0}]
0.049  OpenMP_IMPLICIT_TASK: L_copy_y_face__204__par_loop0_2_176 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/exch_qbc.f} {204,0}]
0.049  OpenMP_IMPLICIT_TASK: L_copy_x_face__255__par_loop1_2_87 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/exch_qbc.f} {255,0}]
0.049  OpenMP_IMPLICIT_TASK: L_copy_y_face__215__par_loop1_2_211 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/exch_qbc.f} {215,0}]
0.048  OpenMP_IMPLICIT_TASK: L_copy_x_face__244__par_loop0_2_54 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/exch_qbc.f} {244,0}]
0.047  OpenMP_LOOP: L_initialize__28__par_region0_2_193 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/initialize.f} {28,0}]
0.03   OpenMP_IMPLICIT_TASK: L_y_solve__43__par_region0_2_43 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/y_solve.f} {43,0}]
0.029  OpenMP_IMPLICIT_TASK: L_z_solve__43__par_region0_2_44 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/z_solve.f} {43,0}]
0.029  OpenMP_IMPLICIT_TASK: L_x_solve__46__par_region0_2_43 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/x_solve.f} {46,0}]
0.027  OpenMP_IMPLICIT_TASK: L_add__22__par_loop0_2_19 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/add.f} {22,0}]
0.027  OpenMP_LOOP: L_exact_rhs__21__par_region0_2_154 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/exact_rhs.f} {21,0}]
0.025  OpenMP_BARRIER: L_copy_y_face__215__par_loop1_2_211 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/exch_qbc.f} {215,0}]
0.025  OpenMP_BARRIER: L_copy_x_face__255__par_loop1_2_87 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/exch_qbc.f} {255,0}]
0.025  OpenMP_BARRIER: L_copy_x_face__244__par_loop0_2_54 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/exch_qbc.f} {244,0}]
0.025  OpenMP_BARRIER: L_copy_y_face__204__par_loop0_2_176 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/exch_qbc.f} {204,0}]
0.013  OpenMP_BARRIER: L_y_solve__43__par_region0_2_43 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/y_solve.f} {43,0}]
0.013  OpenMP_BARRIER: L_x_solve__46__par_region0_2_43 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/x_solve.f} {46,0}]
0.013  OpenMP_BARRIER: L_z_solve__43__par_region0_2_44 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/z_solve.f} {43,0}]
0.012  OpenMP_BARRIER: L_add__22__par_loop0_2_19 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/add.f} {22,0}]
0.004  OpenMP_LOOP: L_error_norm__27__par_region0_2_148 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/error.f} {27,0}]
0.001  OpenMP_IMPLICIT_TASK: L_initialize__28__par_region0_2_193 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/initialize.f} {28,0}]
```

Exclusive time spent in each code region (OpenMP loop) is shown here for MPI rank 0 thread 1

Paraprof 3D visualization window



Triangle Mesh
[✓ Bar Plot]
Scatter Plot
Topology Plot

Height Metric: Exclusive, TIME
Color Metric: Exclusive, TIME
Function: OpenMP_LOOP: L_z_solve__43__par_region0
Thread: 3:0
Height value: 8.572 seconds
Color value: 8.572 seconds

Scales | Plot | Axes | Color | Render

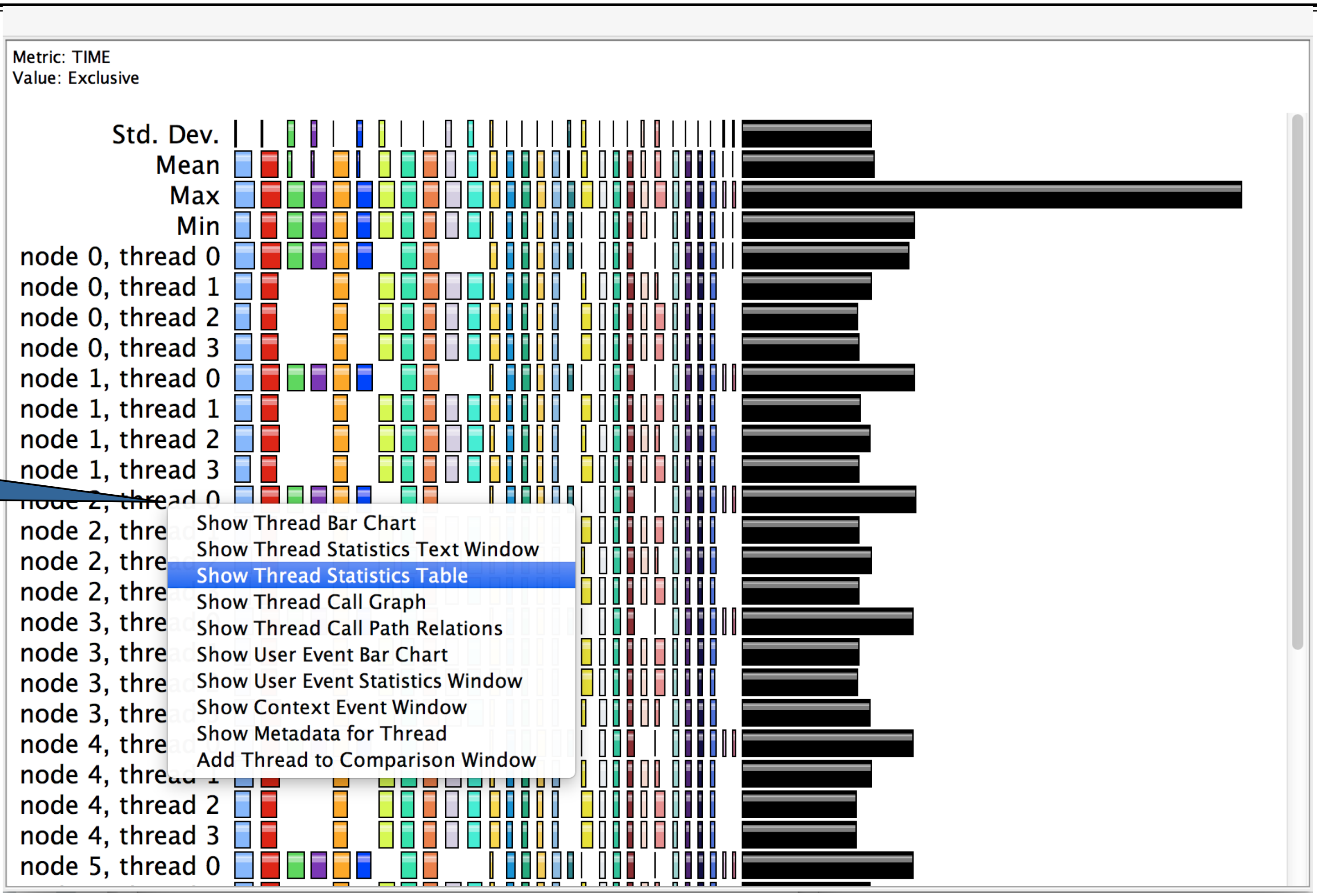
Speed
Auto-Rotate
Reverse Video
Stereo
AA Lines
Full Screen AA
Separation
Aperture
GL Info

Click Bar Plot

Move Function and Thread Sliders

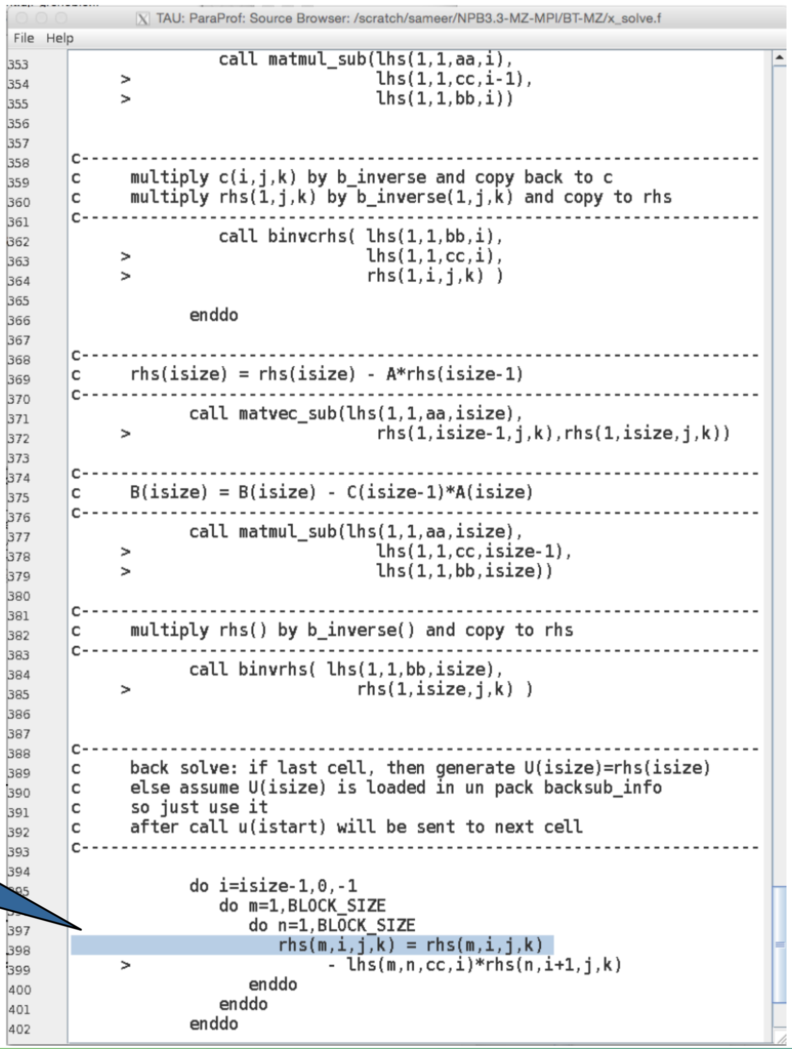
Windows -> 3D visualization

Paraprof Thread Statistics Table with TAU_SAMPLING=1



Right click here

Statement Level Profiling with TAU

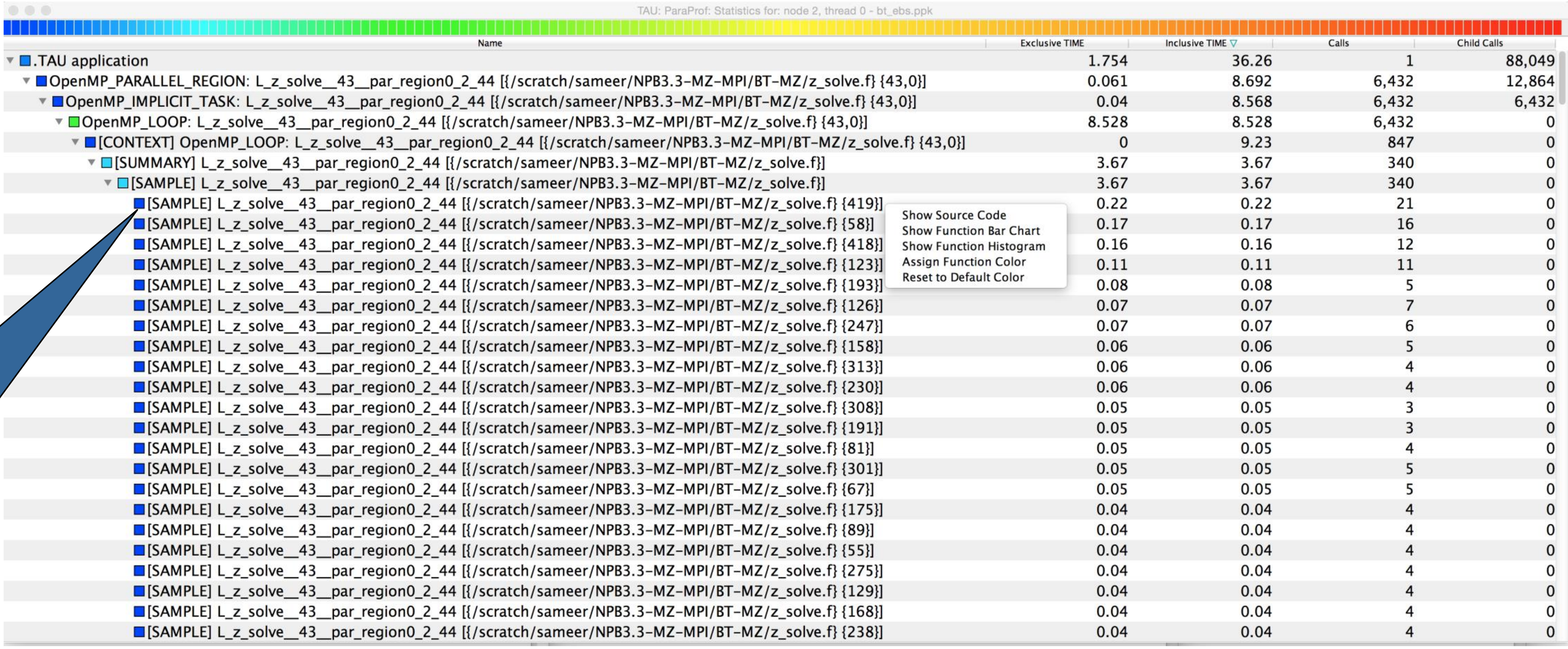

TAU: ParaProf: Source Browser: /scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/x_solve.f

File Help

```
               call matmul_sub(lhs(1,1,aa,i),
     >                         lhs(1,1,cc,i-1),
     >                         lhs(1,1,bb,i))


c----------------------------------------------------------------------
c     multiply c(i,j,k) by b_inverse and copy back to c
c     multiply rhs(1,j,k) by b_inverse(1,j,k) and copy to rhs
c----------------------------------------------------------------------
               call binvcrhs( lhs(1,1,bb,i),
     >                        lhs(1,1,cc,i),
     >                        rhs(1,i,j,k) )

            enddo

c----------------------------------------------------------------------
c     rhs(isize) = rhs(isize) - A*rhs(isize-1)
c----------------------------------------------------------------------
            call matvec_sub(lhs(1,1,aa,isize),
     >                         rhs(1,isize-1,j,k),rhs(1,isize,j,k))

c----------------------------------------------------------------------
c     B(isize) = B(isize) - C(isize-1)*A(isize)
c----------------------------------------------------------------------
            call matmul_sub(lhs(1,1,aa,isize),
     >                         lhs(1,1,cc,isize-1),
     >                         lhs(1,1,bb,isize))

c----------------------------------------------------------------------
c     multiply rhs() by b_inverse() and copy to rhs
c----------------------------------------------------------------------
            call binvrhs( lhs(1,1,bb,isize),
     >                       rhs(1,isize,j,k) )


c----------------------------------------------------------------------
c     back solve: if last cell, then generate U(isize)=rhs(isize)
c     else assume U(isize) is loaded in un pack backsub_info
c     so just use it
c     after call u(istart) will be sent to next cell
c----------------------------------------------------------------------

            do i=isize-1,0,-1
               do m=1,BLOCK_SIZE
                  do n=1,BLOCK_SIZE
                     rhs(m,i,j,k) = rhs(m,i,j,k)
     >                    - lhs(m,n,cc,i)*rhs(n,i+1,j,k)
                  enddo
               enddo
            enddo
```

Source location where samples are taken. Compute intensive region.

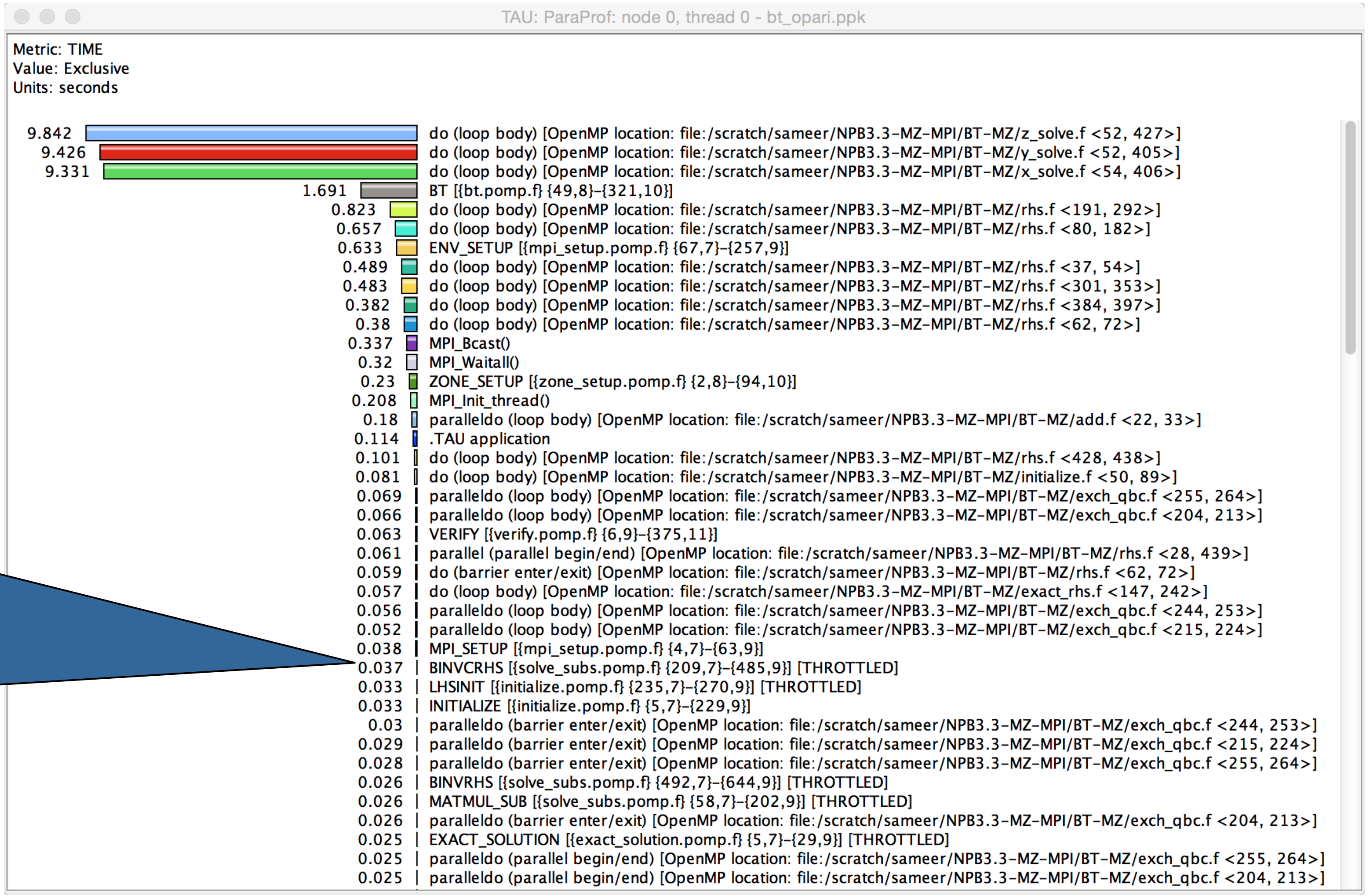
Paraprof Thread Statistics Table



Name	Exclusive TIME	Inclusive TIME ▽	Calls	Child Calls
.TAU application	1.754	36.26	1	88,049
OpenMP_PARALLEL_REGION: L_z_solve__43__par_region0_2_44 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/z_solve.f} {43,0}]	0.061	8.692	6,432	12,864
OpenMP_IMPLICIT_TASK: L_z_solve__43__par_region0_2_44 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/z_solve.f} {43,0}]	0.04	8.568	6,432	6,432
OpenMP_LOOP: L_z_solve__43__par_region0_2_44 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/z_solve.f} {43,0}]	8.528	8.528	6,432	0
[CONTEXT] OpenMP_LOOP: L_z_solve__43__par_region0_2_44 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/z_solve.f} {43,0}]	0	9.23	847	0
[SUMMARY] L_z_solve__43__par_region0_2_44 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/z_solve.f}]	3.67	3.67	340	0
[SAMPLE] L_z_solve__43__par_region0_2_44 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/z_solve.f}]	3.67	3.67	340	0
[SAMPLE] L_z_solve__43__par_region0_2_44 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/z_solve.f} {419}]	0.22	0.22	21	0
[SAMPLE] L_z_solve__43__par_region0_2_44 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/z_solve.f} {58}]	0.17	0.17	16	0
[SAMPLE] L_z_solve__43__par_region0_2_44 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/z_solve.f} {418}]	0.16	0.16	12	0
[SAMPLE] L_z_solve__43__par_region0_2_44 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/z_solve.f} {123}]	0.11	0.11	11	0
[SAMPLE] L_z_solve__43__par_region0_2_44 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/z_solve.f} {193}]	0.08	0.08	5	0
[SAMPLE] L_z_solve__43__par_region0_2_44 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/z_solve.f} {126}]	0.07	0.07	7	0
[SAMPLE] L_z_solve__43__par_region0_2_44 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/z_solve.f} {247}]	0.07	0.07	6	0
[SAMPLE] L_z_solve__43__par_region0_2_44 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/z_solve.f} {158}]	0.06	0.06	5	0
[SAMPLE] L_z_solve__43__par_region0_2_44 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/z_solve.f} {313}]	0.06	0.06	4	0
[SAMPLE] L_z_solve__43__par_region0_2_44 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/z_solve.f} {230}]	0.06	0.06	4	0
[SAMPLE] L_z_solve__43__par_region0_2_44 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/z_solve.f} {308}]	0.05	0.05	3	0
[SAMPLE] L_z_solve__43__par_region0_2_44 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/z_solve.f} {191}]	0.05	0.05	3	0
[SAMPLE] L_z_solve__43__par_region0_2_44 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/z_solve.f} {81}]	0.05	0.05	4	0
[SAMPLE] L_z_solve__43__par_region0_2_44 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/z_solve.f} {301}]	0.05	0.05	5	0
[SAMPLE] L_z_solve__43__par_region0_2_44 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/z_solve.f} {67}]	0.05	0.05	5	0
[SAMPLE] L_z_solve__43__par_region0_2_44 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/z_solve.f} {175}]	0.04	0.04	4	0
[SAMPLE] L_z_solve__43__par_region0_2_44 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/z_solve.f} {89}]	0.04	0.04	4	0
[SAMPLE] L_z_solve__43__par_region0_2_44 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/z_solve.f} {55}]	0.04	0.04	4	0
[SAMPLE] L_z_solve__43__par_region0_2_44 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/z_solve.f} {275}]	0.04	0.04	4	0
[SAMPLE] L_z_solve__43__par_region0_2_44 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/z_solve.f} {129}]	0.04	0.04	4	0
[SAMPLE] L_z_solve__43__par_region0_2_44 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/z_solve.f} {168}]	0.04	0.04	4	0
[SAMPLE] L_z_solve__43__par_region0_2_44 [{/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/z_solve.f} {238}]	0.04	0.04	4	0

Right click here and choose “Show Source Code” for a sample

Instrumenting Source Code with PDT and Opari

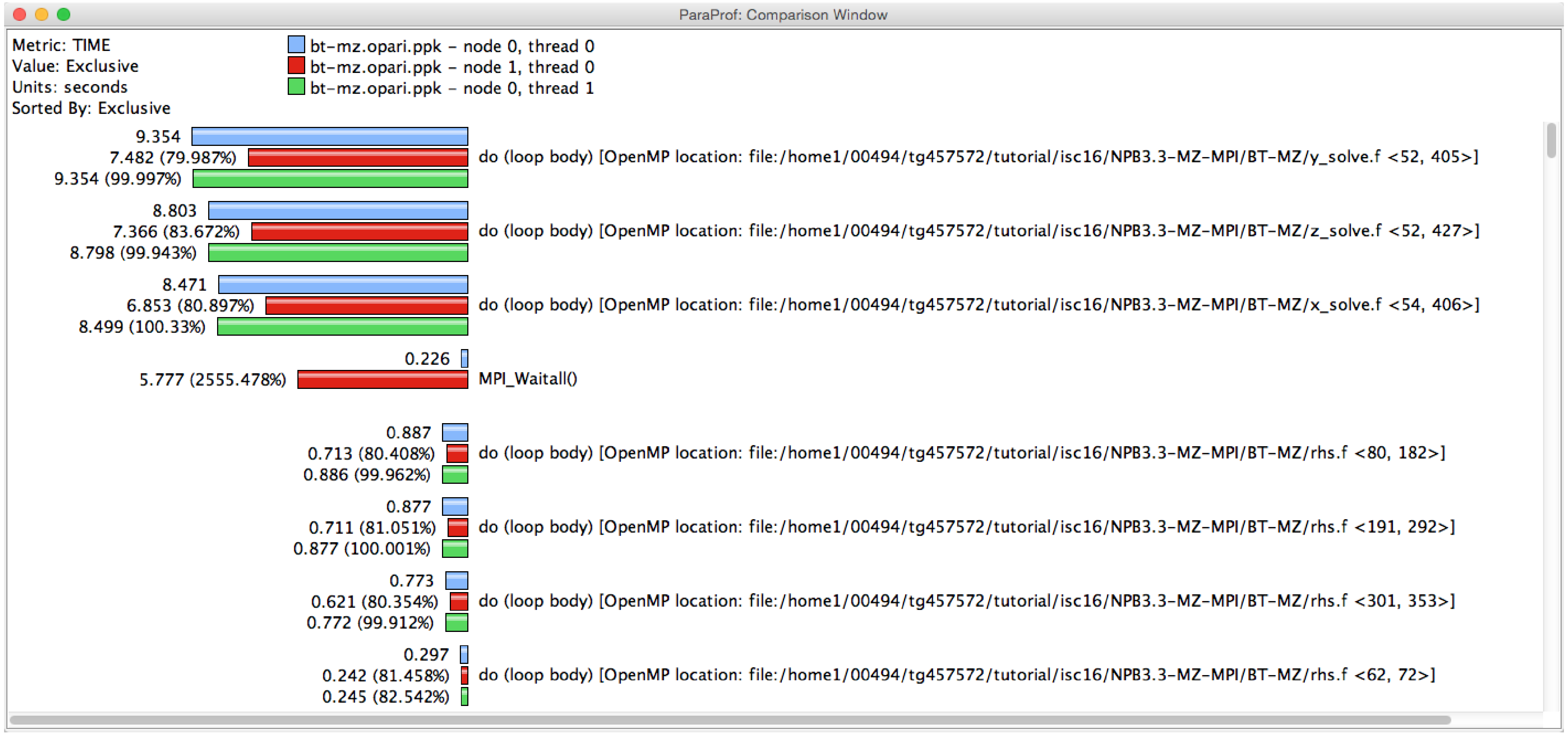


TAU: ParaProf: node 0, thread 0 - bt_opari.ppk

```
Metric: TIME
Value: Exclusive
Units: seconds

9.842  do (loop body) [OpenMP location: file:/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/z_solve.f <52, 427>]
9.426  do (loop body) [OpenMP location: file:/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/y_solve.f <52, 405>]
9.331  do (loop body) [OpenMP location: file:/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/x_solve.f <54, 406>]
1.691  BT [{bt.pomp.f} {49,8}-{321,10}]
0.823  do (loop body) [OpenMP location: file:/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/rhs.f <191, 292>]
0.657  do (loop body) [OpenMP location: file:/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/rhs.f <80, 182>]
0.633  ENV_SETUP [{mpi_setup.pomp.f} {67,7}-{257,9}]
0.489  do (loop body) [OpenMP location: file:/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/rhs.f <37, 54>]
0.483  do (loop body) [OpenMP location: file:/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/rhs.f <301, 353>]
0.382  do (loop body) [OpenMP location: file:/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/rhs.f <384, 397>]
0.38   do (loop body) [OpenMP location: file:/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/rhs.f <62, 72>]
0.337  MPI_Bcast()
0.32   MPI_Waitall()
0.23   ZONE_SETUP [{zone_setup.pomp.f} {2,8}-{94,10}]
0.208  MPI_Init_thread()
0.18   paralleldo (loop body) [OpenMP location: file:/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/add.f <22, 33>]
0.114  .TAU application
0.101  do (loop body) [OpenMP location: file:/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/rhs.f <428, 438>]
0.081  do (loop body) [OpenMP location: file:/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/initialize.f <50, 89>]
0.069  paralleldo (loop body) [OpenMP location: file:/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/exch_qbc.f <255, 264>]
0.066  paralleldo (loop body) [OpenMP location: file:/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/exch_qbc.f <204, 213>]
0.063  VERIFY [{verify.pomp.f} {6,9}-{375,11}]
0.061  parallel (parallel begin/end) [OpenMP location: file:/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/rhs.f <28, 439>]
0.059  do (barrier enter/exit) [OpenMP location: file:/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/rhs.f <62, 72>]
0.057  do (loop body) [OpenMP location: file:/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/exact_rhs.f <147, 242>]
0.056  paralleldo (loop body) [OpenMP location: file:/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/exch_qbc.f <244, 253>]
0.052  paralleldo (loop body) [OpenMP location: file:/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/exch_qbc.f <215, 224>]
0.038  MPI_SETUP [{mpi_setup.pomp.f} {4,7}-{63,9}]
0.037  BINVCRHS [{solve_subs.pomp.f} {209,7}-{485,9}] [THROTTLED]
0.033  LHSINIT [{initialize.pomp.f} {235,7}-{270,9}] [THROTTLED]
0.033  INITIALIZE [{initialize.pomp.f} {5,7}-{229,9}]
0.03   paralleldo (barrier enter/exit) [OpenMP location: file:/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/exch_qbc.f <244, 253>]
0.029  paralleldo (barrier enter/exit) [OpenMP location: file:/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/exch_qbc.f <215, 224>]
0.028  paralleldo (barrier enter/exit) [OpenMP location: file:/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/exch_qbc.f <255, 264>]
0.026  BINVRHS [{solve_subs.pomp.f} {492,7}-{644,9}] [THROTTLED]
0.026  MATMUL_SUB [{solve_subs.pomp.f} {58,7}-{202,9}] [THROTTLED]
0.026  paralleldo (barrier enter/exit) [OpenMP location: file:/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/exch_qbc.f <204, 213>]
0.025  EXACT_SOLUTION [{exact_solution.pomp.f} {5,7}-{29,9}] [THROTTLED]
0.025  paralleldo (parallel begin/end) [OpenMP location: file:/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/exch_qbc.f <255, 264>]
0.025  paralleldo (parallel begin/end) [OpenMP location: file:/scratch/sameer/NPB3.3-MZ-MPI/BT-MZ/exch_qbc.f <204, 213>]
```

Frequently executing lightweight routines are automatically throttled at runtime. Reduces runtime dilation.

ParaProf Comparison Window

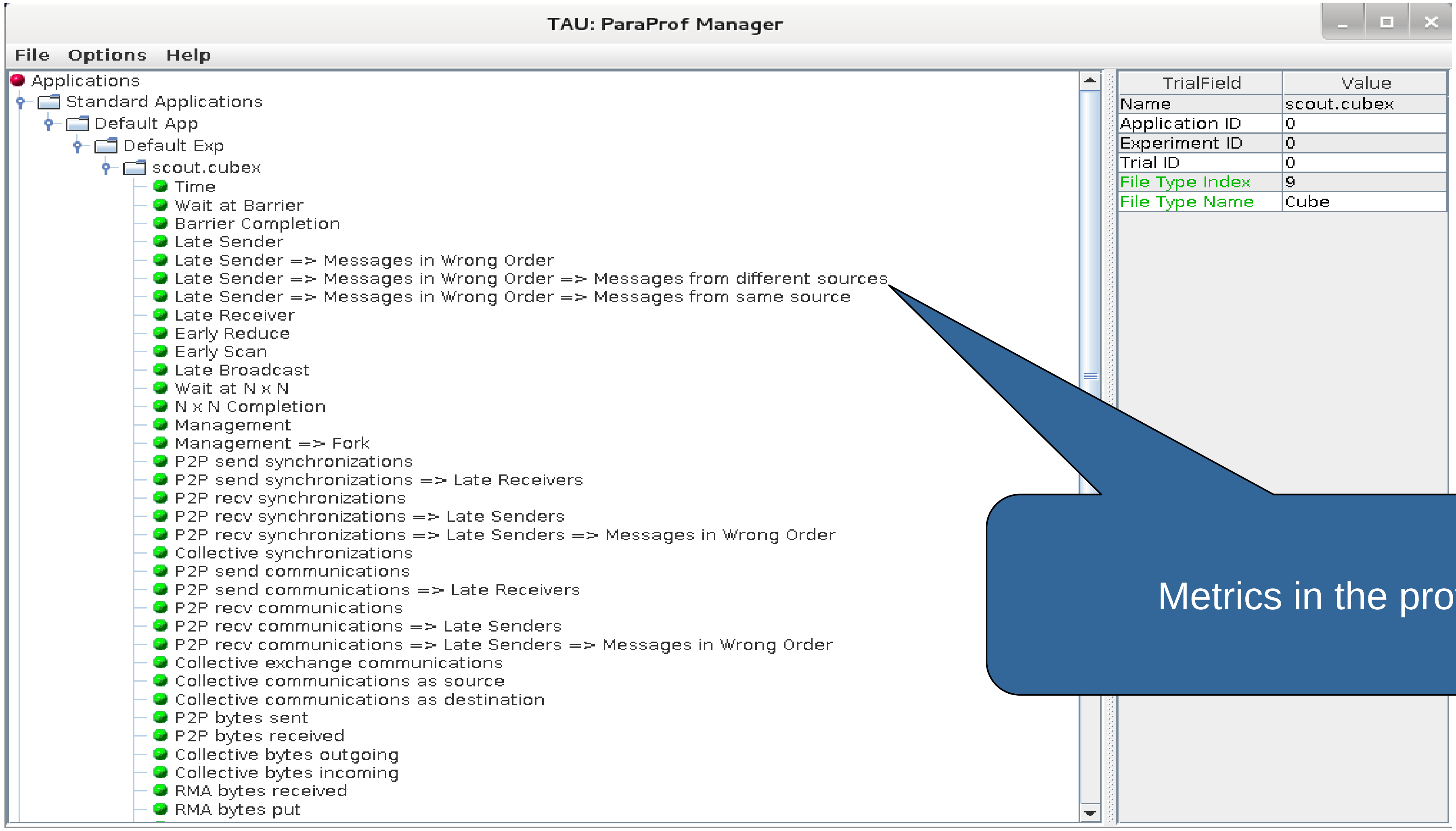

ParaProf: Comparison Window

Metric: TIME
Value: Exclusive
Units: seconds
Sorted By: Exclusive

🟦 bt-mz.opari.ppk – node 0, thread 0
🟥 bt-mz.opari.ppk – node 1, thread 0
🟩 bt-mz.opari.ppk – node 0, thread 1

```
9.354
7.482 (79.987%)     do (loop body) [OpenMP location: file:/home1/00494/tg457572/tutorial/isc16/NPB3.3-MZ-MPI/BT-MZ/y_solve.f <52, 405>]
9.354 (99.997%)

8.803
7.366 (83.672%)     do (loop body) [OpenMP location: file:/home1/00494/tg457572/tutorial/isc16/NPB3.3-MZ-MPI/BT-MZ/z_solve.f <52, 427>]
8.798 (99.943%)

8.471
6.853 (80.897%)     do (loop body) [OpenMP location: file:/home1/00494/tg457572/tutorial/isc16/NPB3.3-MZ-MPI/BT-MZ/x_solve.f <54, 406>]
8.499 (100.33%)

0.226
5.777 (2555.478%)   MPI_Waitall()

0.887
0.713 (80.408%)     do (loop body) [OpenMP location: file:/home1/00494/tg457572/tutorial/isc16/NPB3.3-MZ-MPI/BT-MZ/rhs.f <80, 182>]
0.886 (99.962%)

0.877
0.711 (81.051%)     do (loop body) [OpenMP location: file:/home1/00494/tg457572/tutorial/isc16/NPB3.3-MZ-MPI/BT-MZ/rhs.f <191, 292>]
0.877 (100.001%)

0.773
0.621 (80.354%)     do (loop body) [OpenMP location: file:/home1/00494/tg457572/tutorial/isc16/NPB3.3-MZ-MPI/BT-MZ/rhs.f <301, 353>]
0.772 (99.912%)

0.297
0.242 (81.458%)     do (loop body) [OpenMP location: file:/home1/00494/tg457572/tutorial/isc16/NPB3.3-MZ-MPI/BT-MZ/rhs.f <62, 72>]
0.245 (82.542%)
```

ParaProf Manager Widow: scout.cubex

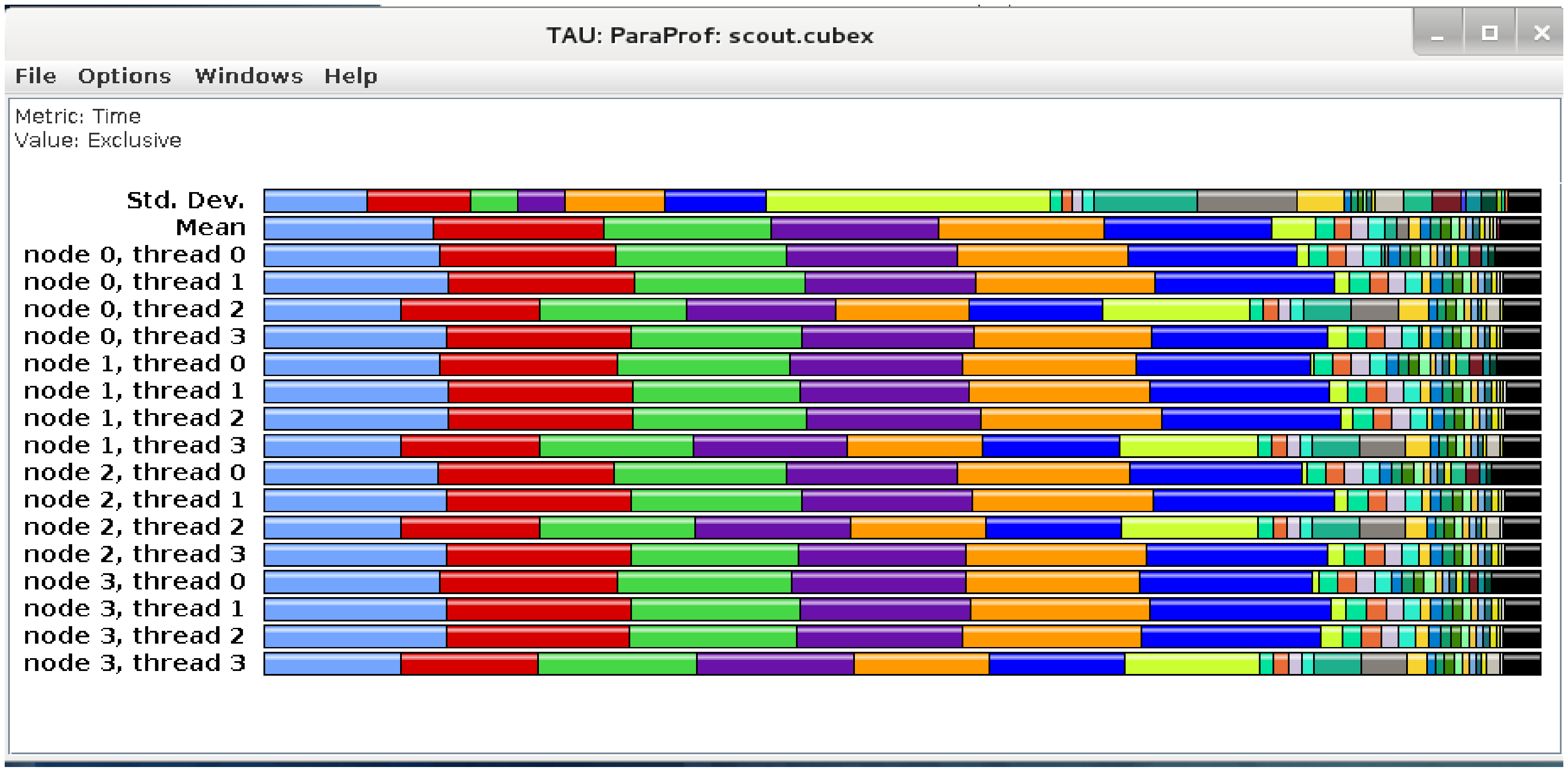


- Applications
  - Standard Applications
    - Default App
      - Default Exp
        - scout.cubex
          - Time
          - Wait at Barrier
          - Barrier Completion
          - Late Sender
          - Late Sender => Messages in Wrong Order
          - Late Sender => Messages in Wrong Order => Messages from different sources
          - Late Sender => Messages in Wrong Order => Messages from same source
          - Late Receiver
          - Early Reduce
          - Early Scan
          - Late Broadcast
          - Wait at N x N
          - N x N Completion
          - Management
          - Management => Fork
          - P2P send synchronizations
          - P2P send synchronizations => Late Receivers
          - P2P recv synchronizations
          - P2P recv synchronizations => Late Senders
          - P2P recv synchronizations => Late Senders => Messages in Wrong Order
          - Collective synchronizations
          - P2P send communications
          - P2P send communications => Late Receivers
          - P2P recv communications
          - P2P recv communications => Late Senders
          - P2P recv communications => Late Senders => Messages in Wrong Order
          - Collective exchange communications
          - Collective communications as source
          - Collective communications as destination
          - P2P bytes sent
          - P2P bytes received
          - Collective bytes outgoing
          - Collective bytes incoming
          - RMA bytes received
          - RMA bytes put

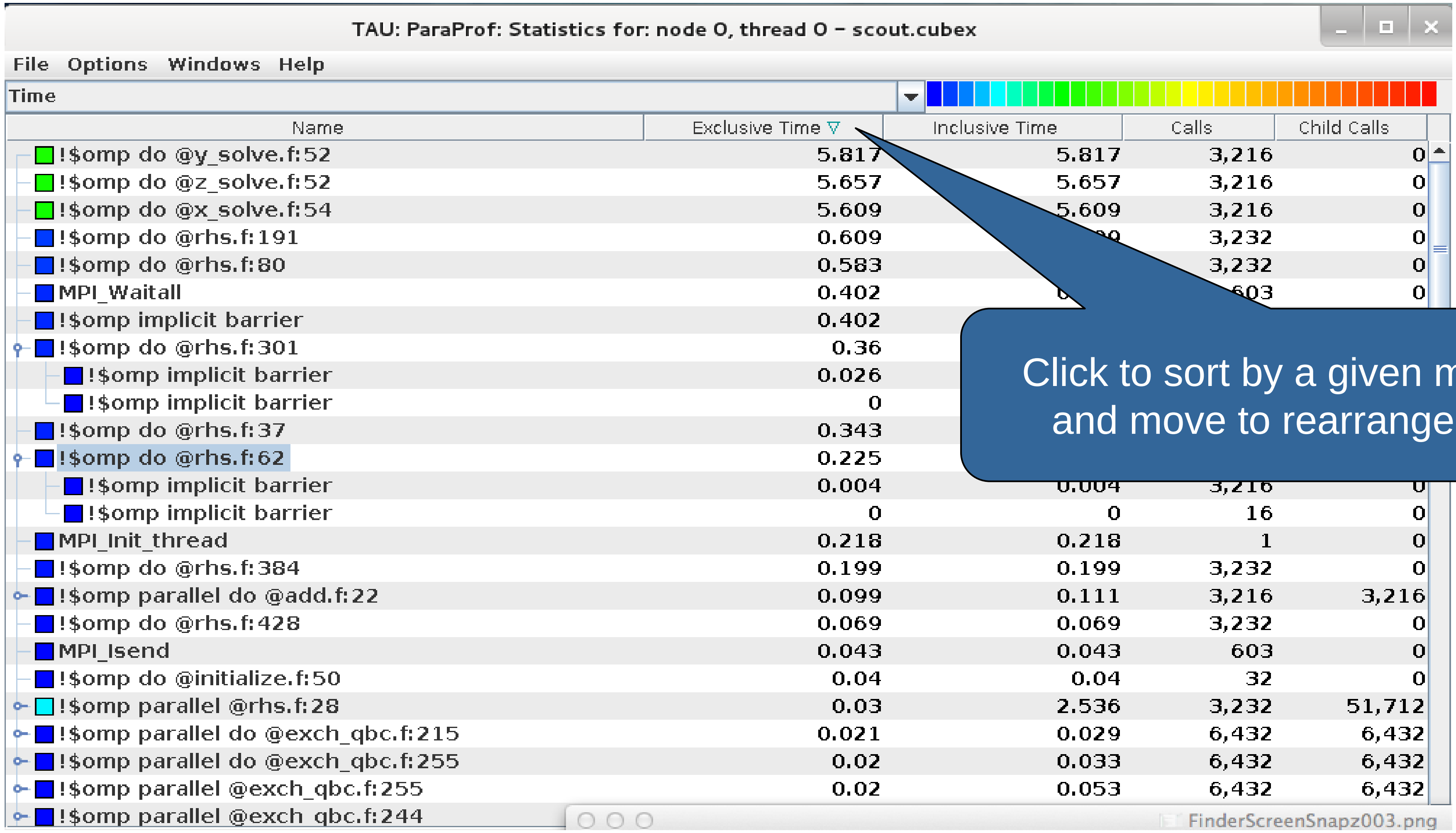
TrialField	Value
Name	scout.cubex
Application ID	0
Experiment ID	0
Trial ID	0
File Type Index	9
File Type Name	Cube

Metrics in the profile

ParaProf: Main Window



ParaProf: Thread Statistics Table



TAU: ParaProf: Statistics for: node 0, thread 0 – scout.cubex

File Options Windows Help

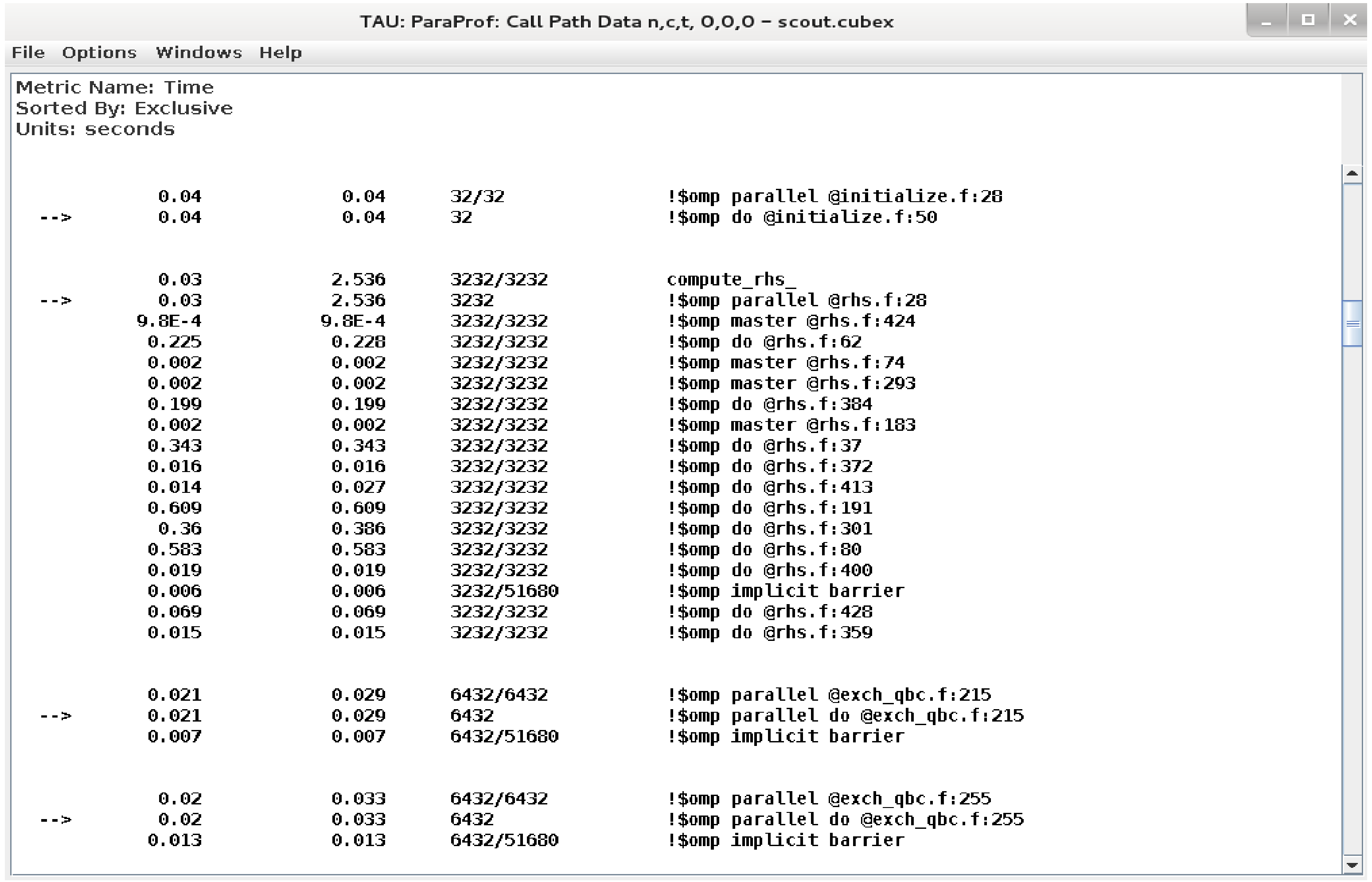
Time

Name	Exclusive Time ▽	Inclusive Time	Calls	Child Calls
🟩 !$omp do @y_solve.f:52	5.817	5.817	3,216	0
🟩 !$omp do @z_solve.f:52	5.657	5.657	3,216	0
🟩 !$omp do @x_solve.f:54	5.609	5.609	3,216	0
🟦 !$omp do @rhs.f:191	0.609	[illegible]	3,232	0
🟦 !$omp do @rhs.f:80	0.583	[illegible]	3,232	0
🟦 MPI_Waitall	0.402	[illegible]	603	0
🟦 !$omp implicit barrier	0.402	[illegible]	[illegible]	[illegible]
🟦 !$omp do @rhs.f:301	0.36	[illegible]	[illegible]	[illegible]
🟦 !$omp implicit barrier	0.026	[illegible]	[illegible]	[illegible]
🟦 !$omp implicit barrier	0	[illegible]	[illegible]	[illegible]
🟦 !$omp do @rhs.f:37	0.343	[illegible]	[illegible]	[illegible]
🟦 !$omp do @rhs.f:62	0.225	[illegible]	[illegible]	[illegible]
🟦 !$omp implicit barrier	0.004	0.004	3,216	0
🟦 !$omp implicit barrier	0	0	16	0
🟦 MPI_Init_thread	0.218	0.218	1	0
🟦 !$omp do @rhs.f:384	0.199	0.199	3,232	0
🟦 !$omp parallel do @add.f:22	0.099	0.111	3,216	3,216
🟦 !$omp do @rhs.f:428	0.069	0.069	3,232	0
🟦 MPI_Isend	0.043	0.043	603	0
🟦 !$omp do @initialize.f:50	0.04	0.04	32	0
🟦 !$omp parallel @rhs.f:28	0.03	2.536	3,232	51,712
🟦 !$omp parallel do @exch_qbc.f:215	0.021	0.029	6,432	6,432
🟦 !$omp parallel do @exch_qbc.f:255	0.02	0.033	6,432	6,432
🟦 !$omp parallel @exch_qbc.f:255	0.02	0.053	6,432	6,432
🟦 !$omp parallel @exch_qbc.f:244				

FinderScreenSnapz003.png

Click to sort by a given metric, drag and move to rearrange columns

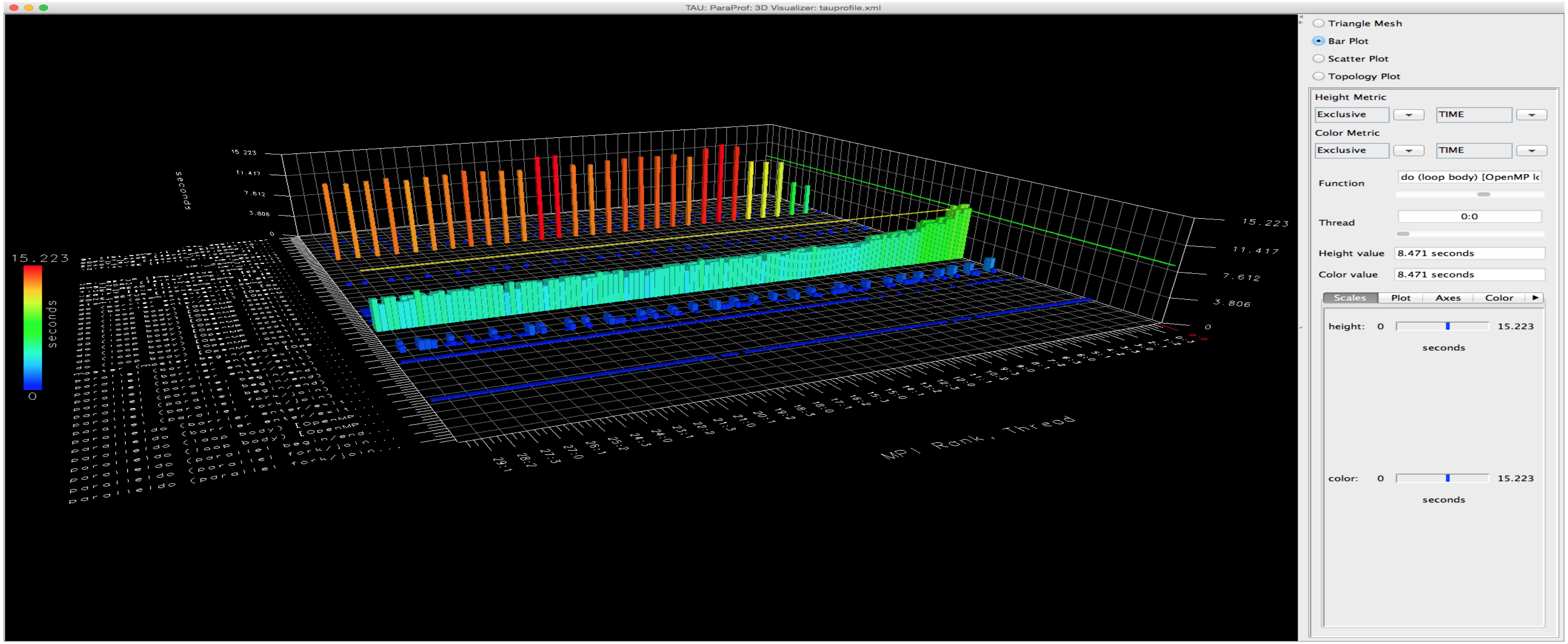
ParaProf: Callpath Thread Relations Window


TAU: ParaProf: Call Path Data n,c,t, 0,0,0 - scout.cubex

File Options Windows Help

```
Metric Name: Time
Sorted By: Exclusive
Units: seconds


          0.04            0.04      32/32               !$omp parallel @initialize.f:28
  -->     0.04            0.04      32                  !$omp do @initialize.f:50


          0.03           2.536      3232/3232           compute_rhs_
  -->     0.03           2.536      3232                !$omp parallel @rhs.f:28
        9.8E-4          9.8E-4      3232/3232           !$omp master @rhs.f:424
         0.225           0.228      3232/3232           !$omp do @rhs.f:62
         0.002           0.002      3232/3232           !$omp master @rhs.f:74
         0.002           0.002      3232/3232           !$omp master @rhs.f:293
         0.199           0.199      3232/3232           !$omp do @rhs.f:384
         0.002           0.002      3232/3232           !$omp master @rhs.f:183
         0.343           0.343      3232/3232           !$omp do @rhs.f:37
         0.016           0.016      3232/3232           !$omp do @rhs.f:372
         0.014           0.027      3232/3232           !$omp do @rhs.f:413
         0.609           0.609      3232/3232           !$omp do @rhs.f:191
          0.36           0.386      3232/3232           !$omp do @rhs.f:301
         0.583           0.583      3232/3232           !$omp do @rhs.f:80
         0.019           0.019      3232/3232           !$omp do @rhs.f:400
         0.006           0.006      3232/51680          !$omp implicit barrier
         0.069           0.069      3232/3232           !$omp do @rhs.f:428
         0.015           0.015      3232/3232           !$omp do @rhs.f:359


         0.021           0.029      6432/6432           !$omp parallel @exch_qbc.f:215
  -->    0.021           0.029      6432                !$omp parallel do @exch_qbc.f:215
         0.007           0.007      6432/51680          !$omp implicit barrier


          0.02           0.033      6432/6432           !$omp parallel @exch_qbc.f:255
  -->     0.02           0.033      6432                !$omp parallel do @exch_qbc.f:255
         0.013           0.013      6432/51680          !$omp implicit barrier
```

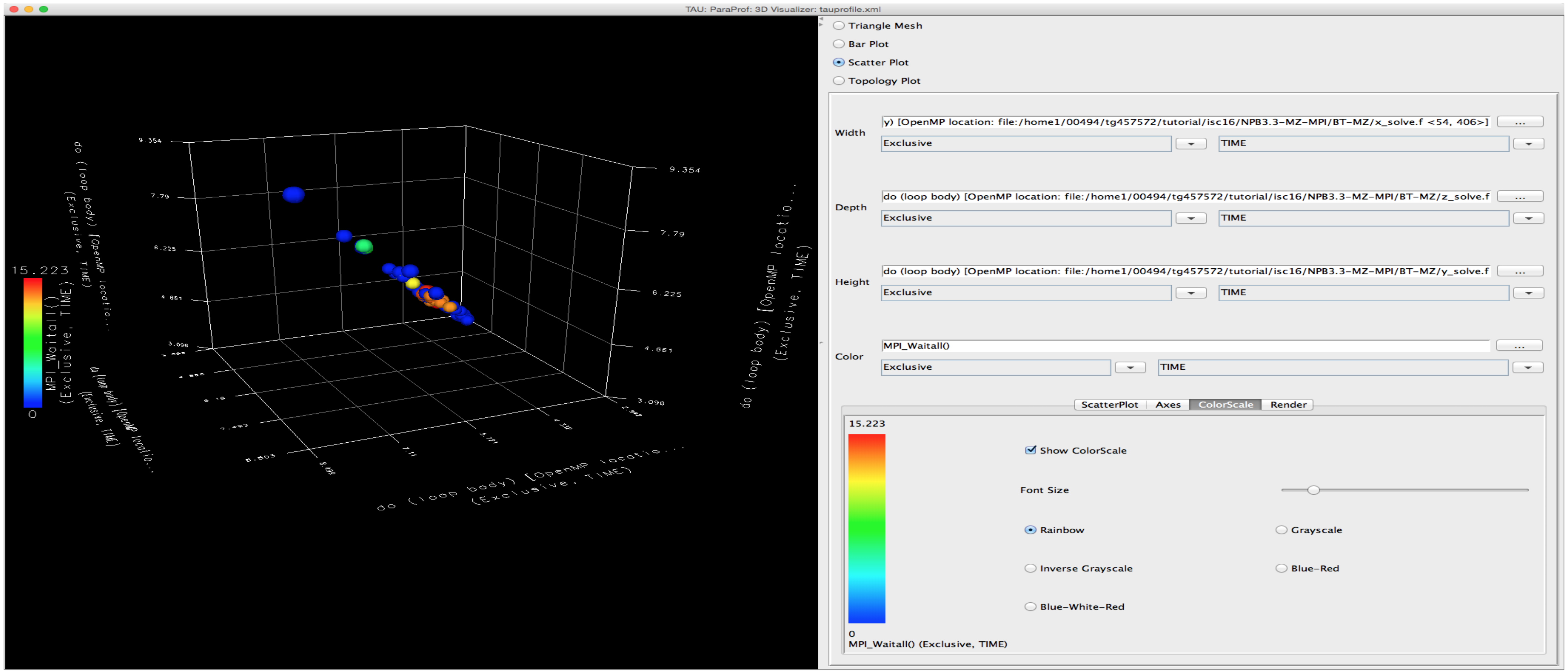
ParaProf: 3D Visualization Window Showing Entire Profile



○ Triangle Mesh
● Bar Plot
○ Scatter Plot
○ Topology Plot

Height Metric: Exclusive, TIME
Color Metric: Exclusive, TIME
Function: do (loop body) [OpenMP lc
Thread: 0:0
Height value: 8.471 seconds
Color value: 8.471 seconds
height: 0 – 15.223 seconds
color: 0 – 15.223 seconds

ParaProf: 3D Scatter Plot


TAU: ParaProf: 3D Visualizer: tauprofile.xml
15.223
0
MPI_Waitall() (Exclusive, TIME)
do (loop body) [OpenMP locatio...
(Exclusive, TIME)
9.354
7.79
6.225
4.661
3.098

Triangle Mesh
Bar Plot
[✓ Scatter Plot]
Topology Plot

Width: y) [OpenMP location: file:/home1/00494/tg457572/tutorial/isc16/NPB3.3-MZ-MPI/BT-MZ/x_solve.f <54, 406>] — Exclusive — TIME

Depth: do (loop body) [OpenMP location: file:/home1/00494/tg457572/tutorial/isc16/NPB3.3-MZ-MPI/BT-MZ/z_solve.f — Exclusive — TIME

Height: do (loop body) [OpenMP location: file:/home1/00494/tg457572/tutorial/isc16/NPB3.3-MZ-MPI/BT-MZ/y_solve.f — Exclusive — TIME

Color: MPI_Waitall() — Exclusive — TIME

ScatterPlot | Axes | ColorScale | Render

15.223

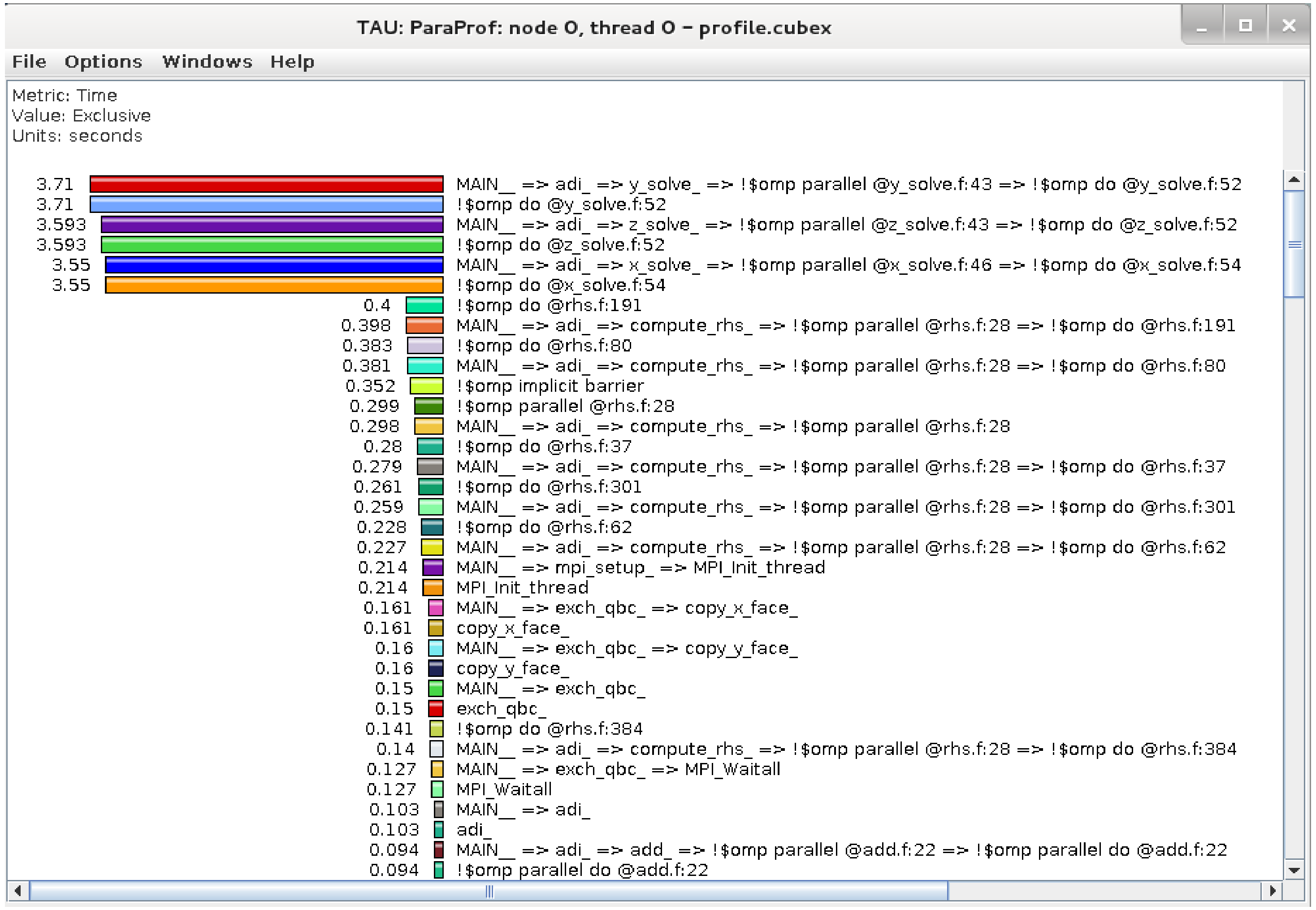
[✓] Show ColorScale

Font Size

[✓ Rainbow] Grayscale
Inverse Grayscale Blue–Red
Blue–White–Red

0
MPI_Waitall() (Exclusive, TIME)

ParaProf: Node View

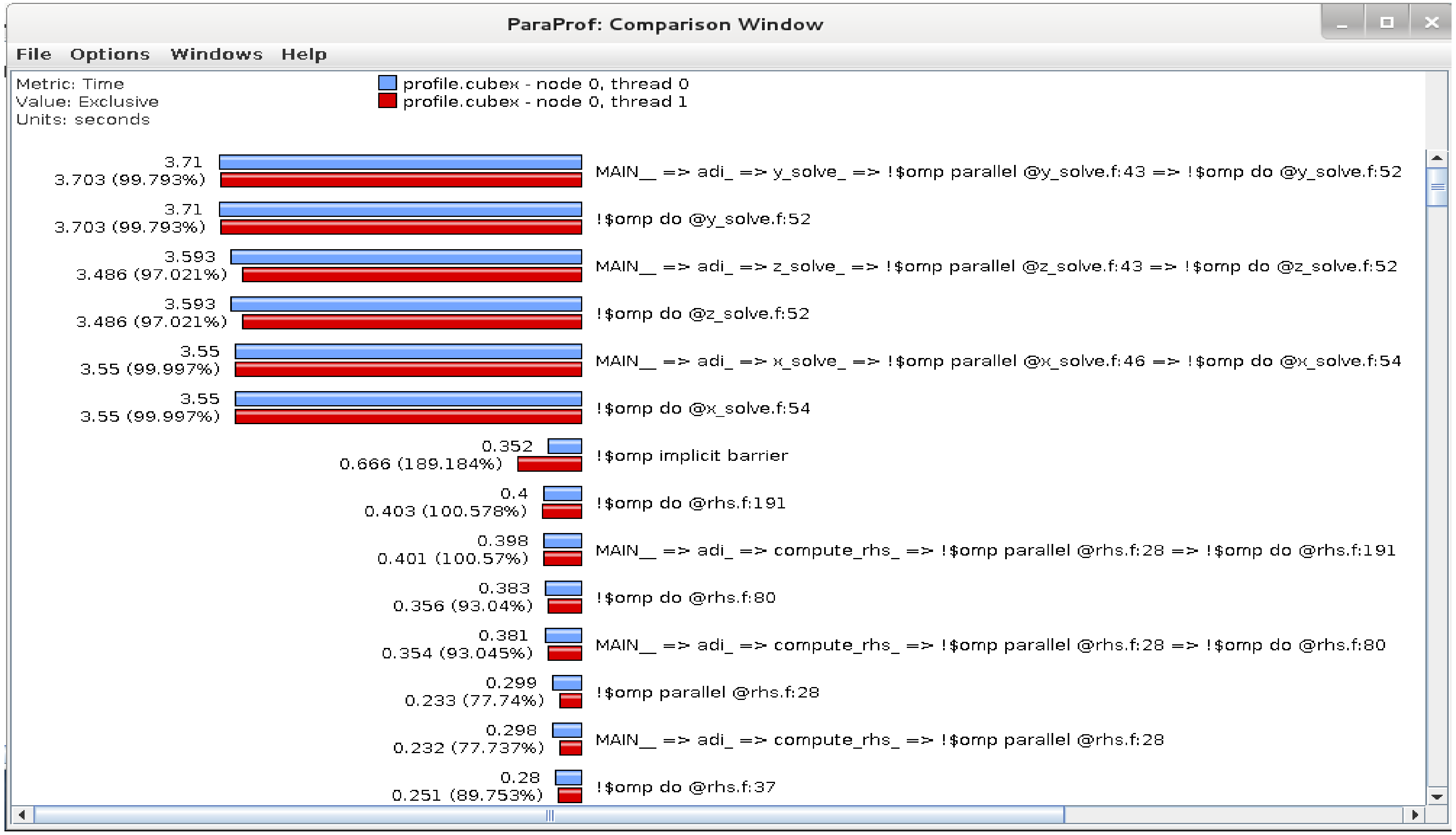


TAU: ParaProf: node 0, thread 0 - profile.cubex

File Options Windows Help

```
Metric: Time
Value: Exclusive
Units: seconds

3.71   MAIN__ => adi_ => y_solve_ => !$omp parallel @y_solve.f:43 => !$omp do @y_solve.f:52
3.71   !$omp do @y_solve.f:52
3.593  MAIN__ => adi_ => z_solve_ => !$omp parallel @z_solve.f:43 => !$omp do @z_solve.f:52
3.593  !$omp do @z_solve.f:52
3.55   MAIN__ => adi_ => x_solve_ => !$omp parallel @x_solve.f:46 => !$omp do @x_solve.f:54
3.55   !$omp do @x_solve.f:54
0.4    !$omp do @rhs.f:191
0.398  MAIN__ => adi_ => compute_rhs_ => !$omp parallel @rhs.f:28 => !$omp do @rhs.f:191
0.383  !$omp do @rhs.f:80
0.381  MAIN__ => adi_ => compute_rhs_ => !$omp parallel @rhs.f:28 => !$omp do @rhs.f:80
0.352  !$omp implicit barrier
0.299  !$omp parallel @rhs.f:28
0.298  MAIN__ => adi_ => compute_rhs_ => !$omp parallel @rhs.f:28
0.28   !$omp do @rhs.f:37
0.279  MAIN__ => adi_ => compute_rhs_ => !$omp parallel @rhs.f:28 => !$omp do @rhs.f:37
0.261  !$omp do @rhs.f:301
0.259  MAIN__ => adi_ => compute_rhs_ => !$omp parallel @rhs.f:28 => !$omp do @rhs.f:301
0.228  !$omp do @rhs.f:62
0.227  MAIN__ => adi_ => compute_rhs_ => !$omp parallel @rhs.f:28 => !$omp do @rhs.f:62
0.214  MAIN__ => mpi_setup_ => MPI_Init_thread
0.214  MPI_Init_thread
0.161  MAIN__ => exch_qbc_ => copy_x_face_
0.161  copy_x_face_
0.16   MAIN__ => exch_qbc_ => copy_y_face_
0.16   copy_y_face_
0.15   MAIN__ => exch_qbc_
0.15   exch_qbc_
0.141  !$omp do @rhs.f:384
0.14   MAIN__ => adi_ => compute_rhs_ => !$omp parallel @rhs.f:28 => !$omp do @rhs.f:384
0.127  MAIN__ => exch_qbc_ => MPI_Waitall
0.127  MPI_Waitall
0.103  MAIN__ => adi_
0.103  adi_
0.094  MAIN__ => adi_ => add_ => !$omp parallel @add.f:22 => !$omp parallel do @add.f:22
0.094  !$omp parallel do @add.f:22
```

ParaProf: Add thread to comparison window

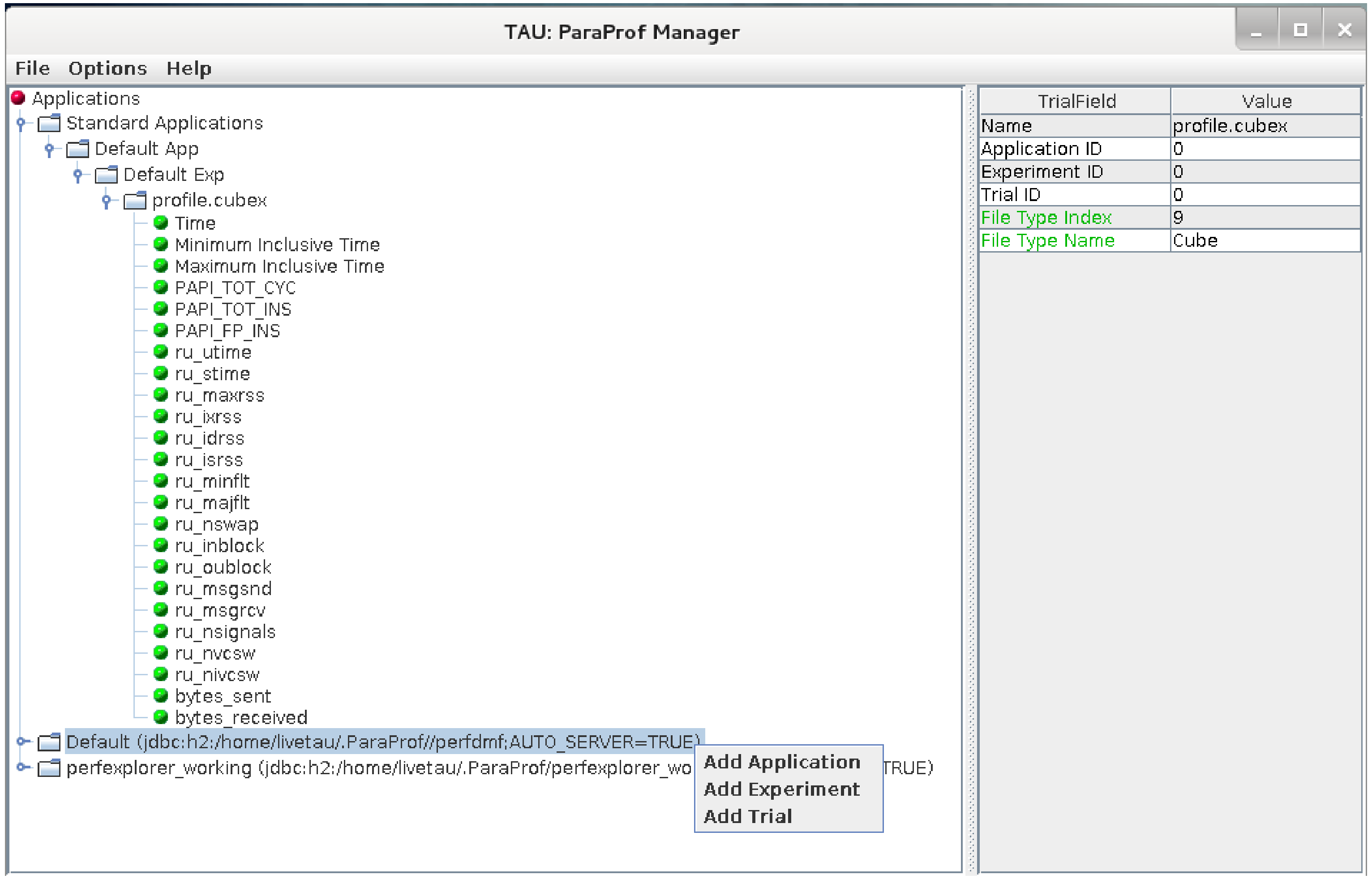


Metric: Time
Value: Exclusive
Units: seconds

🟦 profile.cubex - node 0, thread 0
🟥 profile.cubex - node 0, thread 1

```
3.71
3.703 (99.793%)     MAIN__ => adi_ => y_solve_ => !$omp parallel @y_solve.f:43 => !$omp do @y_solve.f:52
3.71
3.703 (99.793%)     !$omp do @y_solve.f:52
3.593
3.486 (97.021%)     MAIN__ => adi_ => z_solve_ => !$omp parallel @z_solve.f:43 => !$omp do @z_solve.f:52
3.593
3.486 (97.021%)     !$omp do @z_solve.f:52
3.55
3.55 (99.997%)      MAIN__ => adi_ => x_solve_ => !$omp parallel @x_solve.f:46 => !$omp do @x_solve.f:54
3.55
3.55 (99.997%)      !$omp do @x_solve.f:54
0.352
0.666 (189.184%)    !$omp implicit barrier
0.4
0.403 (100.578%)    !$omp do @rhs.f:191
0.398
0.401 (100.57%)     MAIN__ => adi_ => compute_rhs_ => !$omp parallel @rhs.f:28 => !$omp do @rhs.f:191
0.383
0.356 (93.04%)      !$omp do @rhs.f:80
0.381
0.354 (93.045%)     MAIN__ => adi_ => compute_rhs_ => !$omp parallel @rhs.f:28 => !$omp do @rhs.f:80
0.299
0.233 (77.74%)      !$omp parallel @rhs.f:28
0.298
0.232 (77.737%)     MAIN__ => adi_ => compute_rhs_ => !$omp parallel @rhs.f:28
0.28
0.251 (89.753%)     !$omp do @rhs.f:37
```

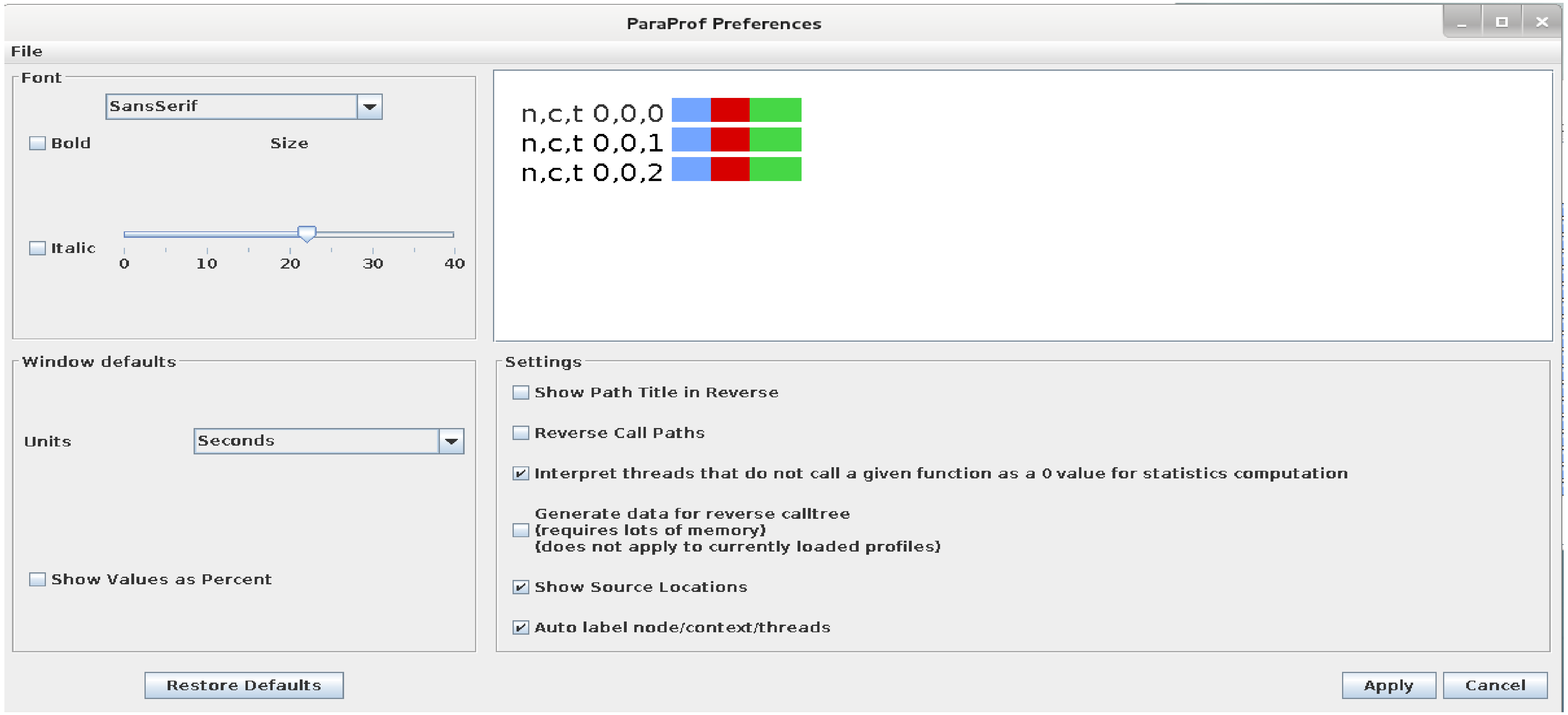
ParaProf: Score-P Profile Files, Database



- Applications
  - Standard Applications
    - Default App
      - Default Exp
        - profile.cubex
          - Time
          - Minimum Inclusive Time
          - Maximum Inclusive Time
          - PAPI_TOT_CYC
          - PAPI_TOT_INS
          - PAPI_FP_INS
          - ru_utime
          - ru_stime
          - ru_maxrss
          - ru_ixrss
          - ru_idrss
          - ru_isrss
          - ru_minflt
          - ru_majflt
          - ru_nswap
          - ru_inblock
          - ru_oublock
          - ru_msgsnd
          - ru_msgrcv
          - ru_nsignals
          - ru_nvcsw
          - ru_nivcsw
          - bytes_sent
          - bytes_received
  - Default (jdbc:h2:/home/livetau/.ParaProf//perfdmf;AUTO_SERVER=TRUE)
  - perfexplorer_working (jdbc:h2:/home/livetau/.ParaProf/perfexplorer_wo TRUE)

TrialField	Value
Name	profile.cubex
Application ID	0
Experiment ID	0
Trial ID	0
File Type Index	9
File Type Name	Cube

ParaProf: File Preferences Window


ParaProf Preferences

File

Font
SansSerif
☐ Bold
Size
☐ Italic
0 10 20 30 40

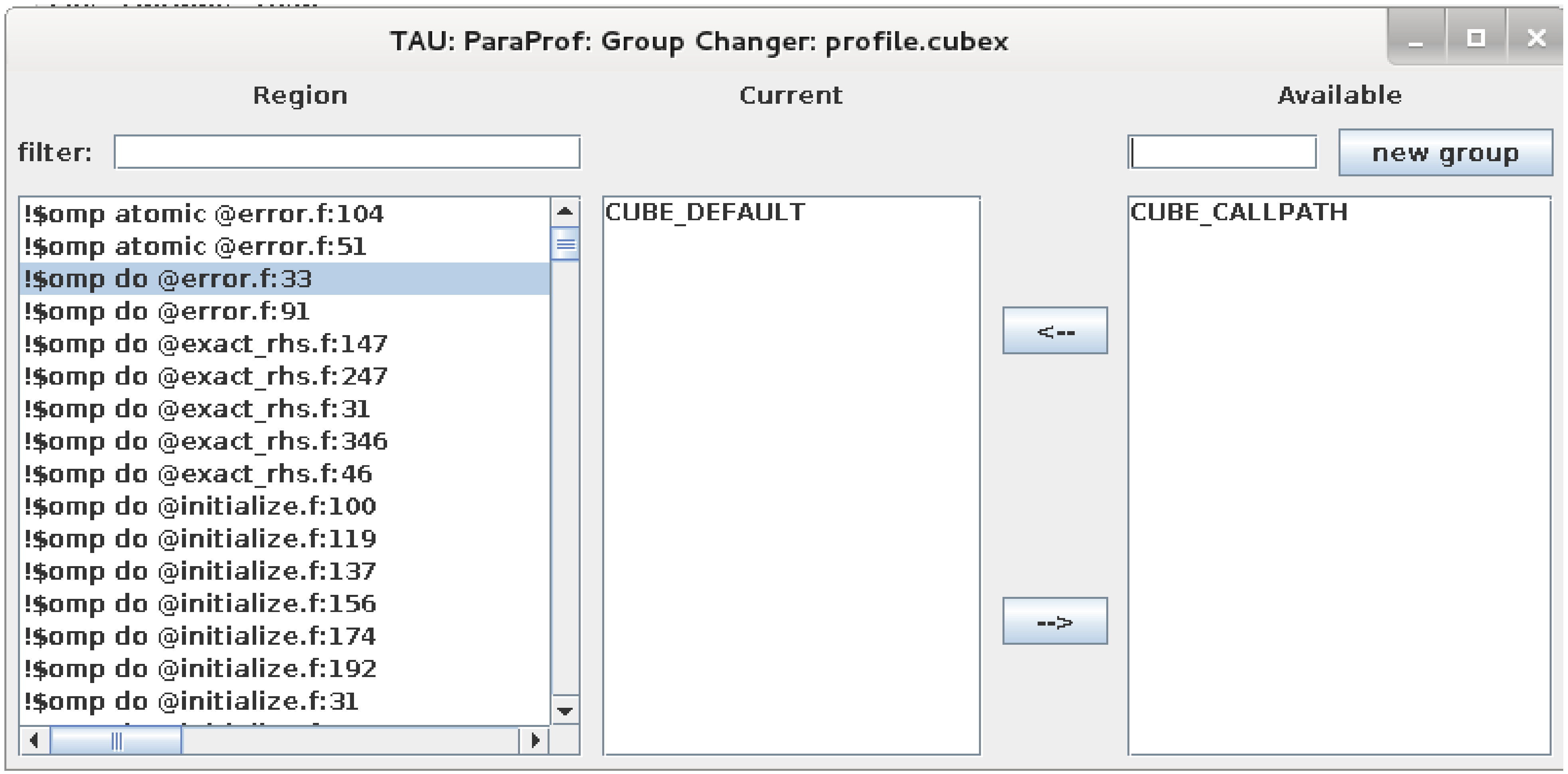
n,c,t 0,0,0
n,c,t 0,0,1
n,c,t 0,0,2

Window defaults
Units: Seconds
☐ Show Values as Percent

Settings
☐ Show Path Title in Reverse
☐ Reverse Call Paths
☑ Interpret threads that do not call a given function as a 0 value for statistics computation
☐ Generate data for reverse calltree
(requires lots of memory)
(does not apply to currently loaded profiles)
☑ Show Source Locations
☑ Auto label node/context/threads

Restore Defaults
Apply
Cancel

ParaProf: Group Changer Window


TAU: ParaProf: Group Changer: profile.cubex

Region

filter:

```
!$omp atomic @error.f:104
!$omp atomic @error.f:51
!$omp do @error.f:33
!$omp do @error.f:91
!$omp do @exact_rhs.f:147
!$omp do @exact_rhs.f:247
!$omp do @exact_rhs.f:31
!$omp do @exact_rhs.f:346
!$omp do @exact_rhs.f:46
!$omp do @initialize.f:100
!$omp do @initialize.f:119
!$omp do @initialize.f:137
!$omp do @initialize.f:156
!$omp do @initialize.f:174
!$omp do @initialize.f:192
!$omp do @initialize.f:31
```

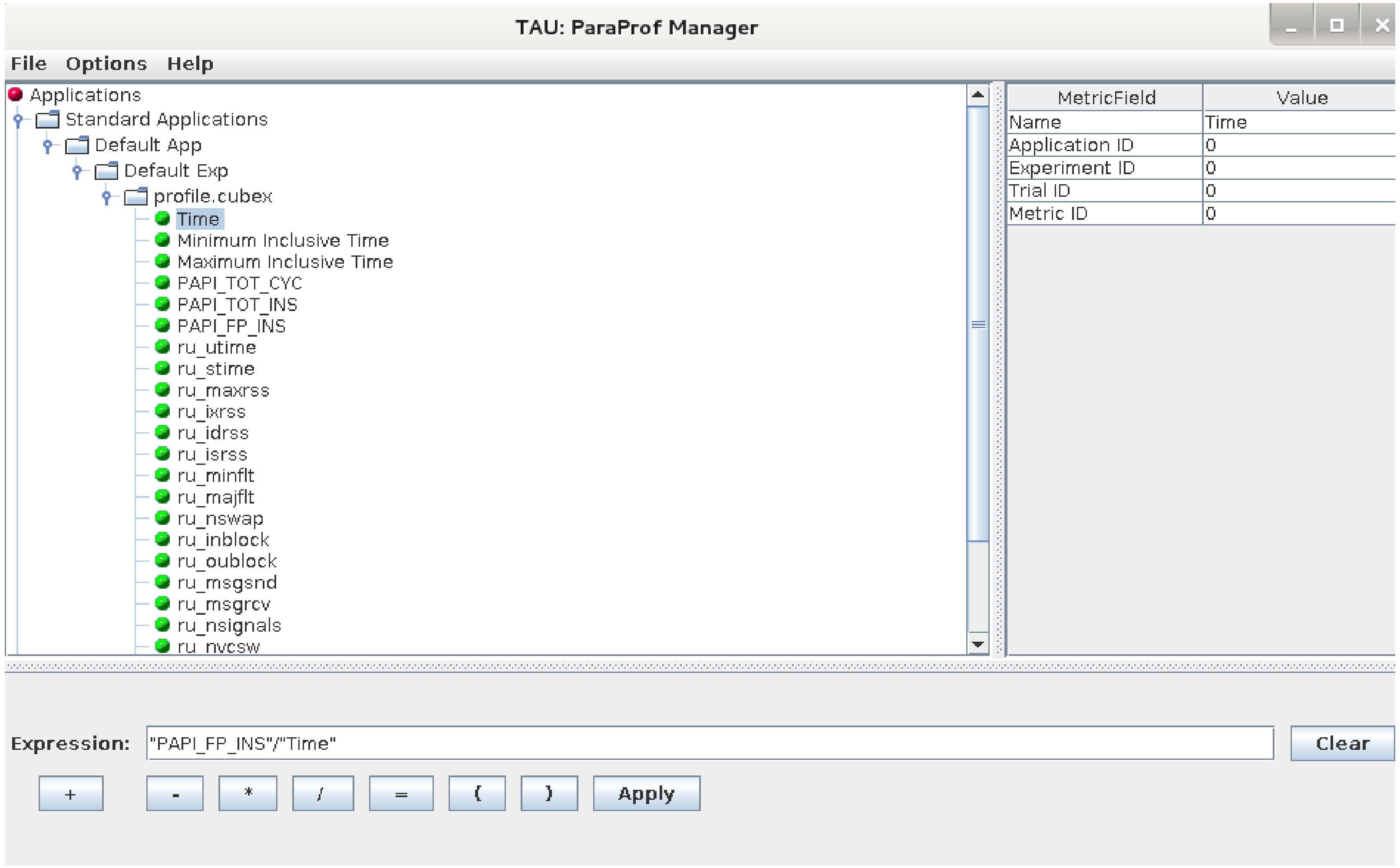
Current

CUBE_DEFAULT

Available

CUBE_CALLPATH

ParaProf: Derived Metric Panel in Manager Window


TAU: ParaProf Manager

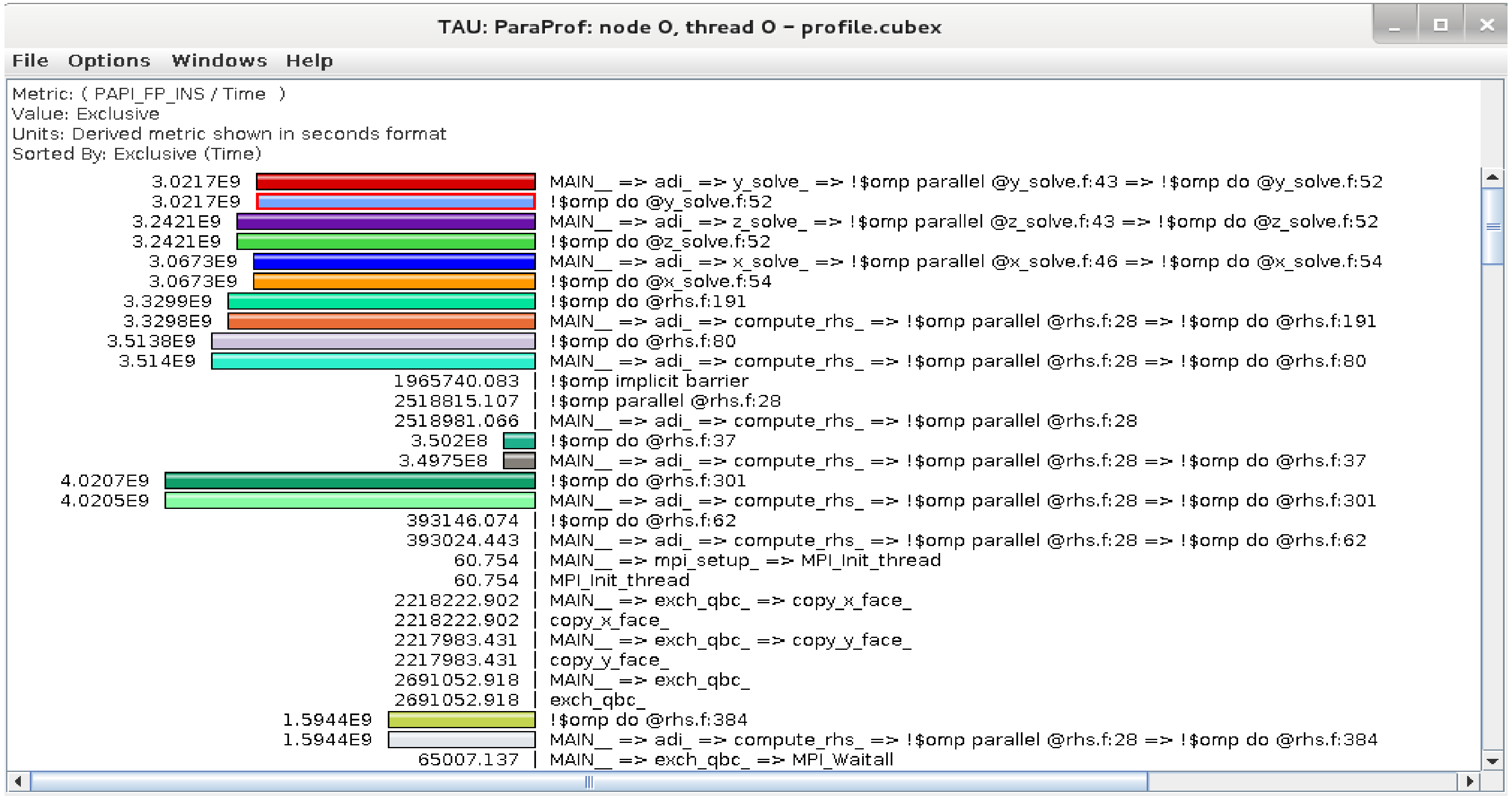
File Options Help

- Applications
  - Standard Applications
    - Default App
      - Default Exp
        - profile.cubex
          - Time
          - Minimum Inclusive Time
          - Maximum Inclusive Time
          - PAPI_TOT_CYC
          - PAPI_TOT_INS
          - PAPI_FP_INS
          - ru_utime
          - ru_stime
          - ru_maxrss
          - ru_ixrss
          - ru_idrss
          - ru_isrss
          - ru_minflt
          - ru_majflt
          - ru_nswap
          - ru_inblock
          - ru_oublock
          - ru_msgsnd
          - ru_msgrcv
          - ru_nsignals
          - ru_nvcsw

MetricField	Value
Name	Time
Application ID	0
Experiment ID	0
Trial ID	0
Metric ID	0

Expression: "PAPI_FP_INS"/"Time"

Sorting Derived FLOPS metric by Exclusive Time


TAU: ParaProf: node 0, thread 0 - profile.cubex
File Options Windows Help

```
Metric: ( PAPI_FP_INS / Time )
Value: Exclusive
Units: Derived metric shown in seconds format
Sorted By: Exclusive (Time)

   3.0217E9 | MAIN__ => adi_ => y_solve_ => !$omp parallel @y_solve.f:43 => !$omp do @y_solve.f:52
   3.0217E9 | !$omp do @y_solve.f:52
   3.2421E9 | MAIN__ => adi_ => z_solve_ => !$omp parallel @z_solve.f:43 => !$omp do @z_solve.f:52
   3.2421E9 | !$omp do @z_solve.f:52
   3.0673E9 | MAIN__ => adi_ => x_solve_ => !$omp parallel @x_solve.f:46 => !$omp do @x_solve.f:54
   3.0673E9 | !$omp do @x_solve.f:54
   3.3299E9 | !$omp do @rhs.f:191
   3.3298E9 | MAIN__ => adi_ => compute_rhs_ => !$omp parallel @rhs.f:28 => !$omp do @rhs.f:191
   3.5138E9 | !$omp do @rhs.f:80
    3.514E9 | MAIN__ => adi_ => compute_rhs_ => !$omp parallel @rhs.f:28 => !$omp do @rhs.f:80
1965740.083 | !$omp implicit barrier
2518815.107 | !$omp parallel @rhs.f:28
2518981.066 | MAIN__ => adi_ => compute_rhs_ => !$omp parallel @rhs.f:28
    3.502E8 | !$omp do @rhs.f:37
   3.4975E8 | MAIN__ => adi_ => compute_rhs_ => !$omp parallel @rhs.f:28 => !$omp do @rhs.f:37
   4.0207E9 | !$omp do @rhs.f:301
   4.0205E9 | MAIN__ => adi_ => compute_rhs_ => !$omp parallel @rhs.f:28 => !$omp do @rhs.f:301
 393146.074 | !$omp do @rhs.f:62
 393024.443 | MAIN__ => adi_ => compute_rhs_ => !$omp parallel @rhs.f:28 => !$omp do @rhs.f:62
     60.754 | MAIN__ => mpi_setup_ => MPI_Init_thread
     60.754 | MPI_Init_thread
2218222.902 | MAIN__ => exch_qbc_ => copy_x_face_
2218222.902 | copy_x_face_
2217983.431 | MAIN__ => exch_qbc_ => copy_y_face_
2217983.431 | copy_y_face_
2691052.918 | MAIN__ => exch_qbc_
2691052.918 | exch_qbc_
   1.5944E9 | !$omp do @rhs.f:384
   1.5944E9 | MAIN__ => adi_ => compute_rhs_ => !$omp parallel @rhs.f:28 => !$omp do @rhs.f:384
  65007.137 | MAIN__ => exch_qbc_ => MPI_Waitall
```

Download TAU from U. Oregon

http://tau.uoregon.edu

http://www.hpclinux.com [LiveDVD, OVA]

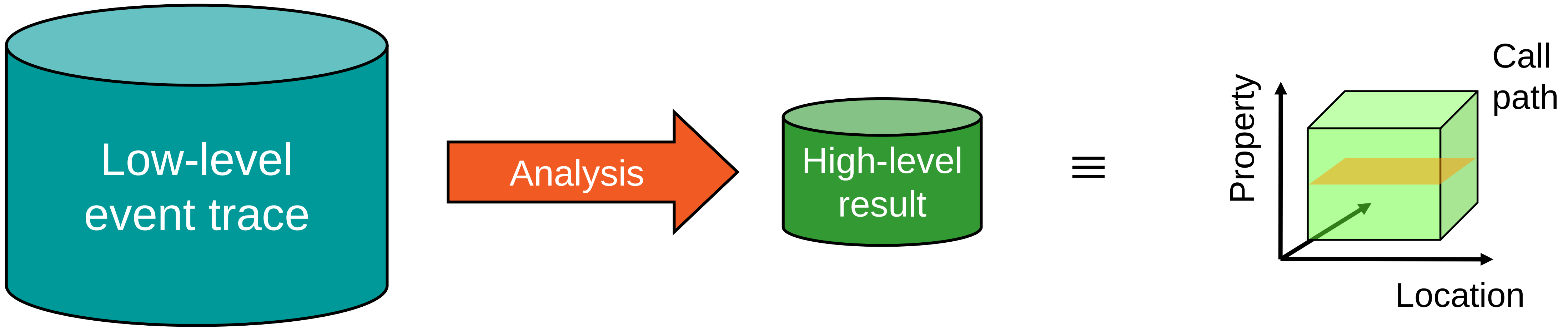
Free download, open source, BSD license

Automatic trace analysis with the Scalasca Trace Tools

Markus Geimer
Jülich Supercomputing Centre

Automatic trace analysis

- Idea
  - Automatic search for patterns of inefficient behavior
  - Classification of behavior & quantification of significance
  - Identification of delays as root causes of inefficiencies



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

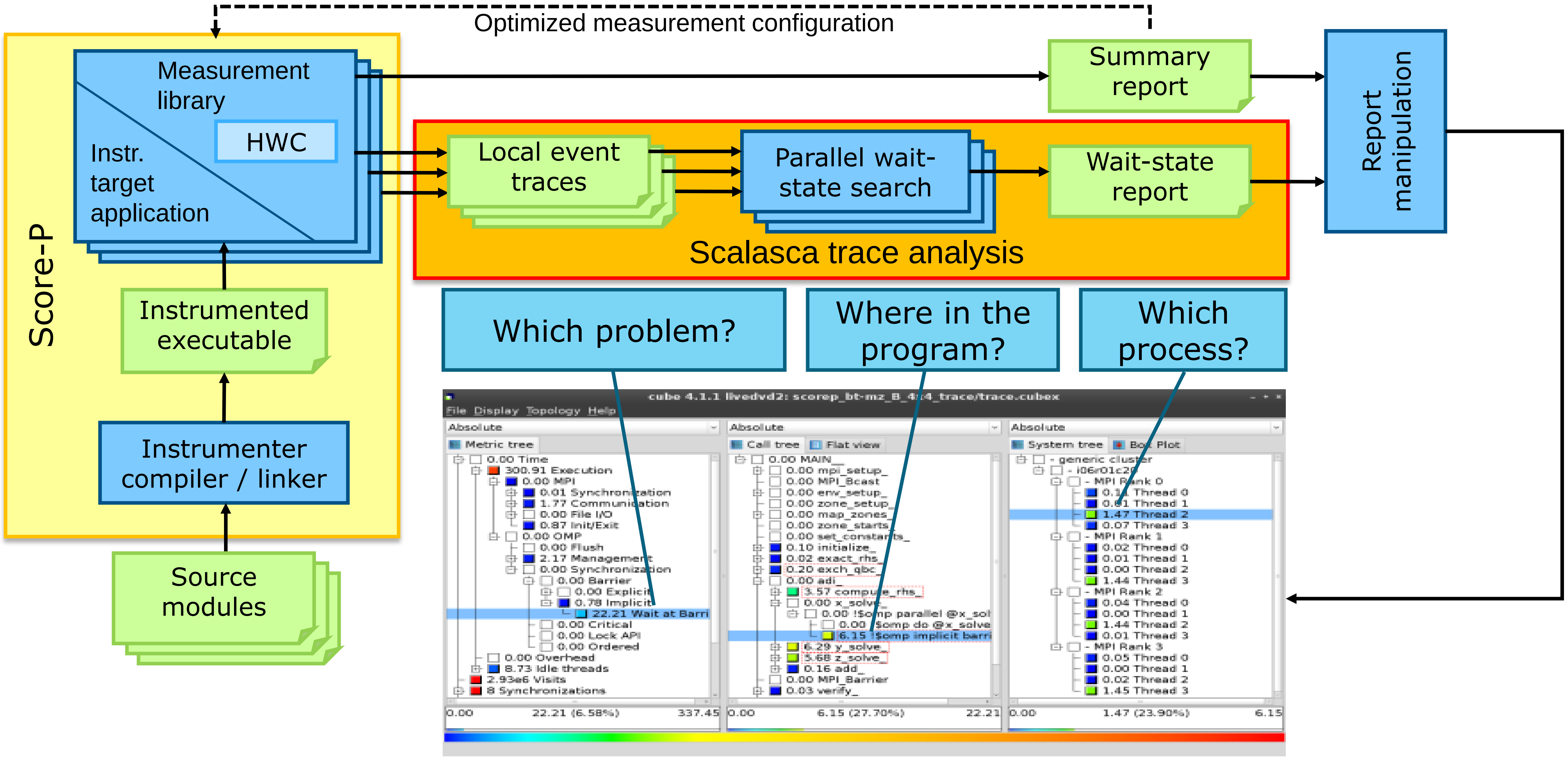
Scalasca Trace Tools: Objective

- Development of a **scalable trace-based** performance analysis toolset for the most popular parallel programming paradigms
  - Current focus: MPI, OpenMP, and POSIX threads
- Specifically targeting large-scale parallel applications
  - Such as those running on IBM Blue Gene or Cray systems with one million or more processes/threads
- Latest release:
  - Scalasca v2.4 coordinated with Score-P v4.0 (May 2018)

Scalasca Trace Tools features

- Open source, 3-clause BSD license
- Fairly portable
  - IBM Blue Gene, Cray XT/XE/XK/XC, SGI Altix, Fujitsu FX10/100 & K computer, Linux clusters (x86, Power, ARM), Intel Xeon Phi, ...
- Uses Score-P instrumenter & measurement libraries
  - Scalasca v2 core package focuses on trace-based analyses
  - Supports common data formats
    - Reads event traces in OTF2 format
    - Writes analysis reports in CUBE4 format
- Current limitations:
  - Unable to handle traces
    - With MPI thread level exceeding MPI_THREAD_FUNNELED
    - Containing CUDA or SHMEM events, or OpenMP nested parallelism
  - PAPI/rusage metrics for trace events are ignored

Scalasca workflow



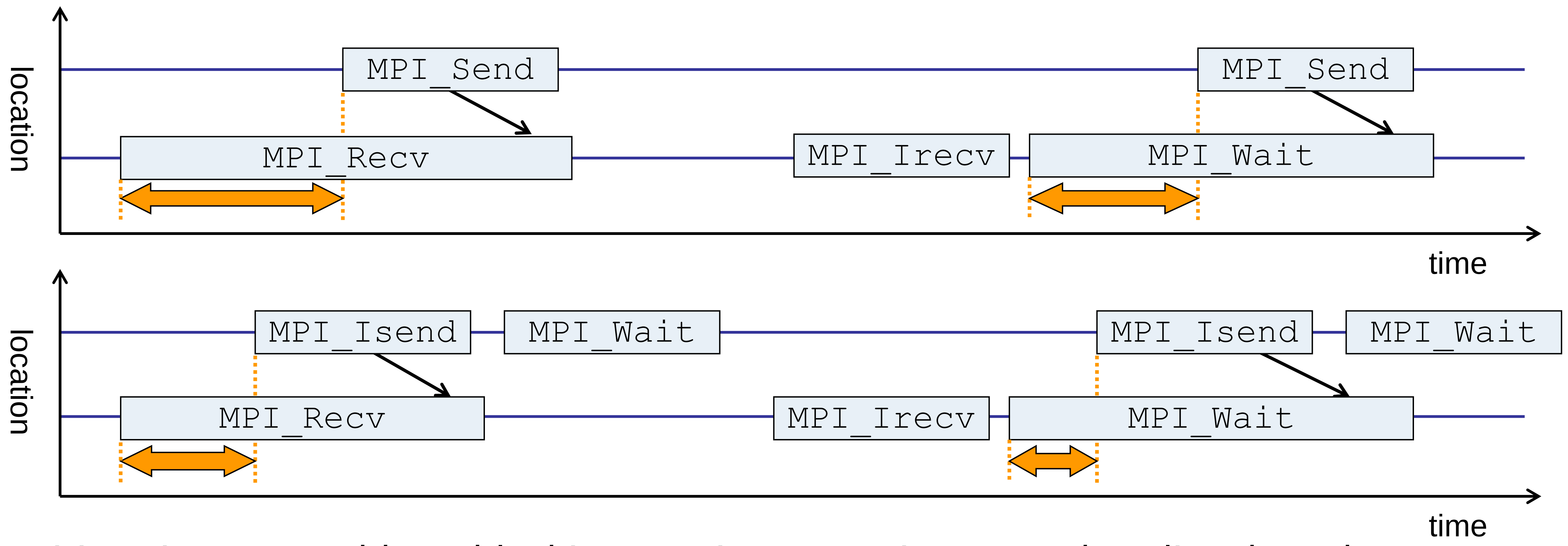
```
cube 4.1.1 livedvd2: scorep_bt-mz_B_4x4_trace/trace.cubex
```

```
0.00 Time
  300.91 Execution
    0.00 MPI
      0.01 Synchronization
      1.77 Communication
      0.00 File I/O
      0.87 Init/Exit
    0.00 OMP
      0.00 Flush
      2.17 Management
      0.00 Synchronization
        0.00 Barrier
          0.00 Explicit
          0.78 Implicit
            22.21 Wait at Barri
        0.00 Critical
        0.00 Lock API
        0.00 Ordered
    0.00 Overhead
    8.73 Idle threads
  2.93e6 Visits
  8 Synchronizations
0.00          22.21 (6.58%)          337.45
```

```
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 set_constants_
  0.10 initialize_
  0.02 exact_rhs_
  0.20 exch_qbc_
  0.00 adi_
    3.57 compute_rhs_
    0.00 x_solve_
      0.00 !$omp parallel @x_sol
        0.00 !$omp do @x_solve
        6.15 !$omp implicit barri
    6.29 y_solve_
    5.68 z_solve_
    0.16 add_
  0.00 MPI_Barrier
  0.03 verify_
0.00          6.15 (27.70%)          22.21
```

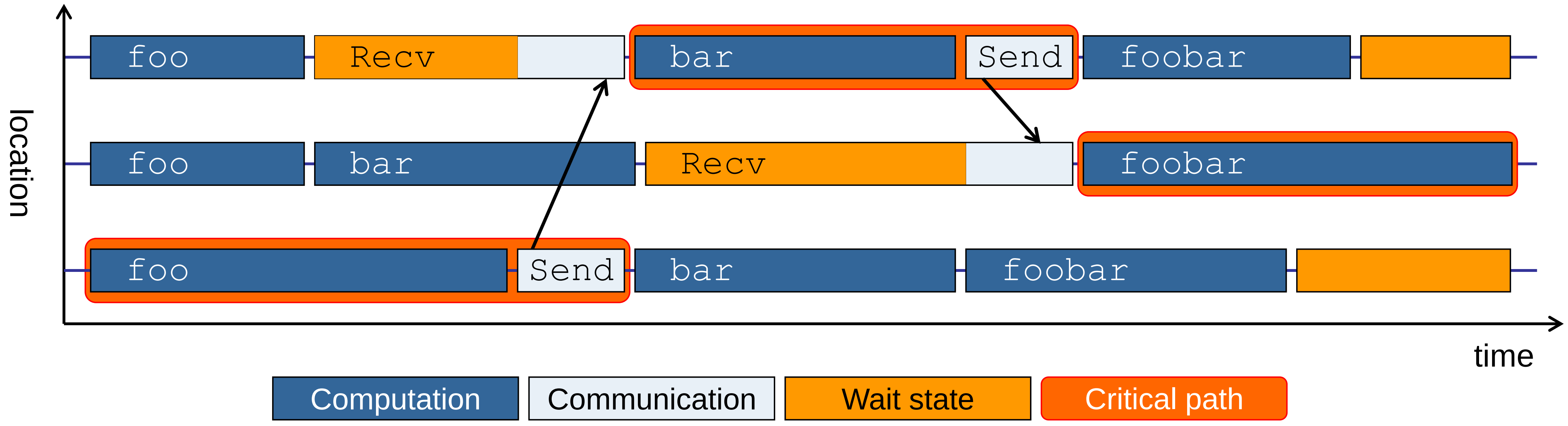
```
- generic cluster
  - i06r01c20
    - MPI Rank 0
      0.11 Thread 0
      0.01 Thread 1
      1.47 Thread 2
      0.07 Thread 3
    - MPI Rank 1
      0.02 Thread 0
      0.01 Thread 1
      0.00 Thread 2
      1.44 Thread 3
    - MPI Rank 2
      0.04 Thread 0
      0.00 Thread 1
      1.44 Thread 2
      0.01 Thread 3
    - MPI Rank 3
      0.05 Thread 0
      0.00 Thread 1
      0.02 Thread 2
      1.45 Thread 3
0.00          1.47 (23.90%)          6.15
```

Example: "*Late Sender*" wait state



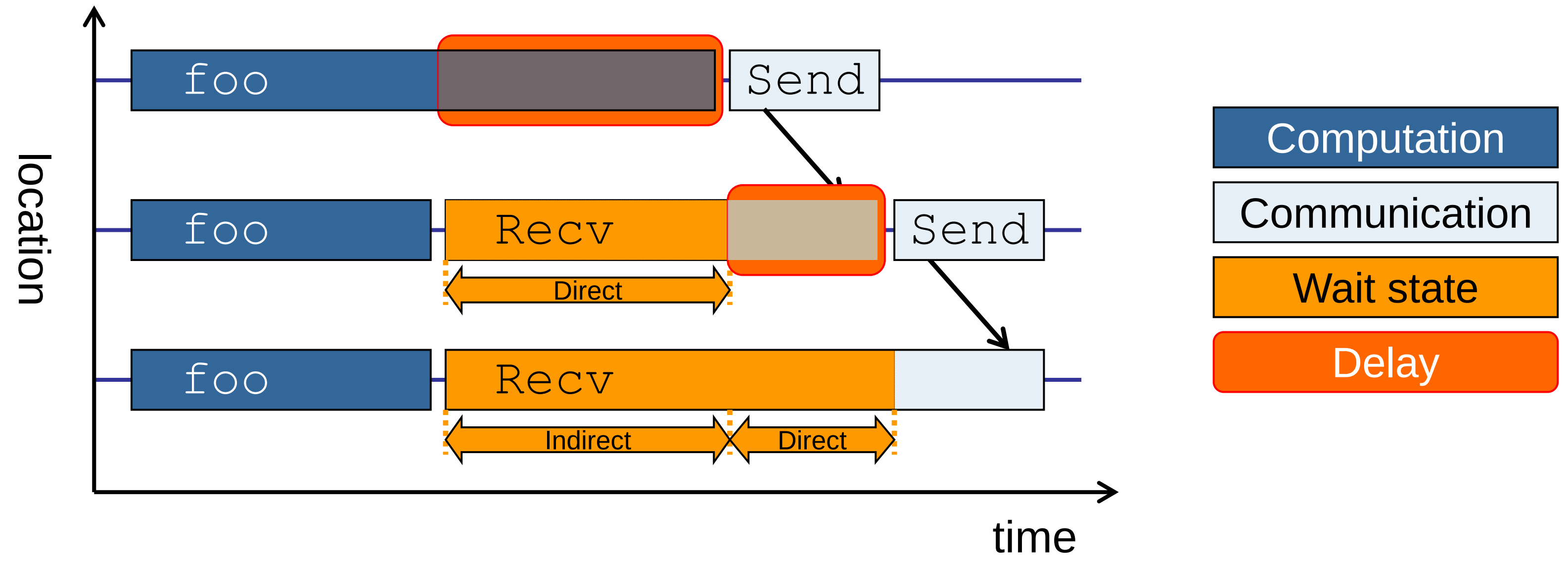
- Waiting time caused by a blocking receive operation posted earlier than the corresponding send
- Applies to blocking as well as non-blocking communication

Example: Critical path



- Shows call paths and processes/threads that are responsible for the program’s wall-clock runtime
- Identifies good optimization candidates and parallelization bottlenecks

Example: Root-cause analysis



- Classifies wait states into direct and indirect (i.e., caused by other wait states)
- Identifies *delays* (excess computation/communication) as root causes of wait states
- Attributes wait states as *delay costs*

Hands-on:
NPB-MZ-MPI / BT

trace tools
scalasca

Performance analysis steps

- 0.0 Reference preparation for validation

- 1.0 Program instrumentation
- 1.1 Summary measurement collection
- 1.2 Summary analysis report examination

- 2.0 Summary experiment scoring
- 2.1 Summary measurement collection with filtering
- 2.2 Filtered summary analysis report examination

- 3.0 Event trace collection
- 3.1 Event trace examination & analysis

Scalasca command – One command for (almost) everything

```
% scalasca
Scalasca 2.4
Toolset for scalable performance analysis of large-scale parallel applications
usage: scalasca [OPTION]... ACTION <argument>...
    1. prepare application objects and executable for measurement:
       scalasca -instrument <compile-or-link-command> # skin (using scorep)
    2. run application under control of measurement system:
       scalasca -analyze <application-launch-command> # scan
    3. interactively explore measurement analysis report:
       scalasca -examine <experiment-archive|report>  # square

Options:
   -c, --show-config     show configuration summary and exit
   -h, --help            show this help and exit
   -n, --dry-run         show actions without taking them
       --quickref        show quick reference guide and exit
       --remap-specfile  show path to remapper specification file and exit
   -v, --verbose         enable verbose commentary
   -V, --version         show version information and exit
```

- The ‘`scalasca -instrument`’ command is deprecated and only provided for backwards compatibility with Scalasca 1.x., recommended: use Score-P instrumenter directly

Scalasca compatibility command: skin / scalasca -instrument

```
% skin
Scalasca 2.4: application instrumenter (using Score-P instrumenter)
usage: skin [-v] [-comp] [-pdt] [-pomp] [-user] [--*] <compile-or-link-command>
  -comp={all|none|...}: routines to be instrumented by compiler [default: all]
                 (... custom instrumentation specification depends on compiler)
  -pdt:  process source files with PDT/TAU 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
```

- Scalasca application instrumenter
  - Provides compatibility with Scalasca 1.x
  - **Deprecated! Use Score-P instrumenter directly.**

Scalasca convenience command: scan / scalasca -analyze

```
% scan
Scalasca 2.4: 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 filtfile  : File specifying measurement filter.
  -l lockfile  : File that blocks start of measurement.
  -m metrics   : Metric specification for measurement.
```

- Scalasca measurement collection & analysis nexus

Scalasca advanced command:
scout - Scalasca automatic trace analyzer

```
% scout.hyb --help
SCOUT   Copyright (c) 1998-2018 Forschungszentrum Juelich GmbH
        Copyright (c) 2009-2014 German Research School for Simulation
                                Sciences GmbH

Usage: <launchcmd> scout.hyb [OPTION]... <ANCHORFILE | EPIK DIRECTORY>
Options:
  --statistics       Enables instance tracking and statistics [default]
  --no-statistics    Disables instance tracking and statistics
  --critical-path    Enables critical-path analysis [default]
  --no-critical-path Disables critical-path analysis
  --rootcause        Enables root-cause analysis [default]
  --no-rootcause     Disables root-cause analysis
  --single-pass      Single-pass forward analysis only
  --time-correct     Enables enhanced timestamp correction
  --no-time-correct  Disables enhanced timestamp correction [default]
  --verbose, -v      Increase verbosity
  --help             Display this information and exit
```

- Provided in serial (.ser), OpenMP (.omp), MPI (.mpi) and MPI+OpenMP (.hyb) variants

Scalasca advanced command: clc_synchronize

- Scalasca trace event timestamp consistency correction

```
Usage: <launchcmd> clc_synchronize.hyb <ANCHORFILE | EPIK_DIRECTORY>
```

- Provided in MPI (.mpi) and MPI+OpenMP (.hyb) variants
- Takes as input a trace experiment archive where the events may have timestamp inconsistencies
  - E.g., multi-node measurements on systems without adequately synchronized clocks on each compute node
- Generates a new experiment archive (always called ./clc_sync) containing a trace with event timestamp inconsistencies resolved
  - E.g., suitable for detailed examination with a time-line visualizer

Scalasca convenience command: square / scalasca -examine

```
% square
Scalasca 2.4: analysis report explorer
usage: square [-v] [-s] [-f filtfile] [-F] <experiment archive | cube file>
   -c <none | quick | full> : Level of sanity checks for newly created reports
   -F                       : Force remapping of already existing reports
   -f filtfile              : Use specified filter file when doing scoring
   -s                       : Skip display and output textual score report
   -v                       : Enable verbose mode
   -n                       : Do not include idle thread metric
```

- Scalasca analysis report explorer (Cube)

Automatic measurement configuration

- scan configures Score-P measurement by automatically setting some environment variables and exporting them
  - 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)

Setup environment

- Remember to source provided shell code snippet to add local tool installations to $PATH

```
% source ~tg828282/Tutorial/vihps.sh
```

- Change to directory containing NPB3.3-MZ-MPI sources
- Existing instrumented executable in bin.scorep/ directory can be reused

```
% cd $SCRATCH/NPB3.3-MZ-MPI
```

BT-MZ summary measurement collection...

```
% cd bin.scorep
% cp ../jobscript/stampede2/scalasca.sbatch .
% vi scalasca.sbatch

# Score-P measurement configuration
export SCOREP_FILTERING_FILE=../config/scorep.filt
#export SCOREP_TOTAL_MEMORY=50M
#export SCOREP_METRIC_PAPI=PAPI_TOT_INS,PAPI_TOT_CYC

# Scalasca configuration
export SCAN_ANALYZE_OPTS="--time-correct"

# Run the application using Scalasca nexus
scalasca -analyze ibrun ./bt-mz_${CLASS}.${PROCS}
```

- Change to directory with the executable and edit the job script

```
% sbatch scalasca.sbatch
```

- Submit the job

BT-MZ summary measurement

```
S=C=A=N: Scalasca 2.4 runtime summarization
S=C=A=N: ./scorep_bt-mz_C_32x4_sum experiment archive
S=C=A=N: Mon Aug 21 07:52:03 2017: Collect start
ibrun ./bt-mz_C.32

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

 Number of zones:  16 x  16
 Iterations: 200    dt:   0.000100
 Number of active processes:    32

 [... More application output ...]

S=C=A=N: Mon Aug 21 07:52:36 2017: Collect done (status=0) 33s
S=C=A=N: ./scorep_bt-mz_C_32x4_sum complete.
```

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

- Creates experiment directory: scorep_bt-mz_C_32x4_sum

BT-MZ summary analysis report examination

- Score summary analysis report

```
% square -s  scorep_bt-mz_C_32x4_sum
INFO: Post-processing runtime summarization result...
INFO: Score report written to ./scorep_bt-mz_C_32x4_sum/scorep.score
```

Hint:
Copy ‘summary.cubex’ to Live-DVD environment using ‘scp’ to improve responsiveness of GUI

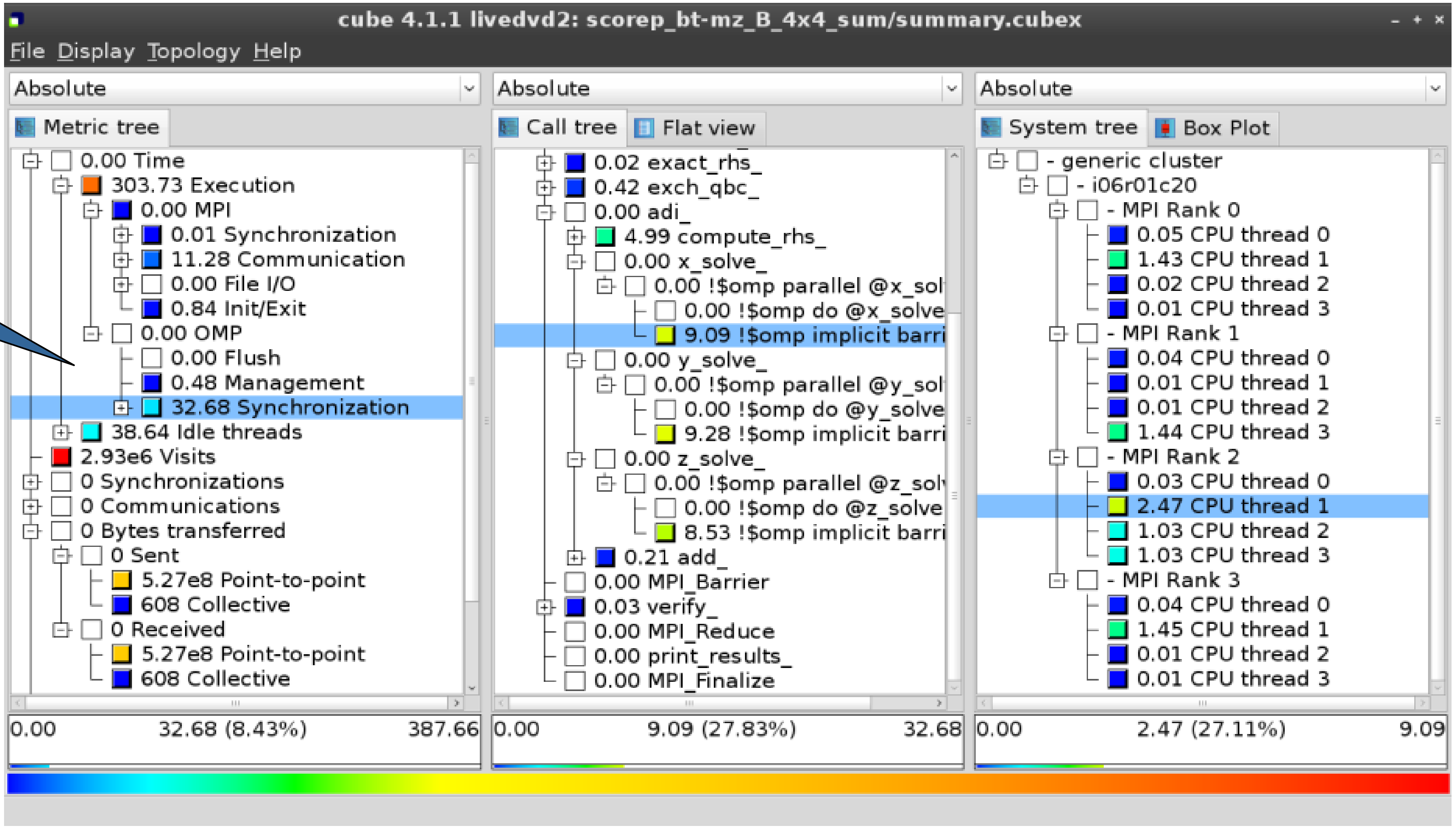
- Post-processing and interactive exploration with Cube

```
% square  scorep_bt-mz_C_32x4_sum
INFO: Displaying ./scorep_bt-mz_C_32x4_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

Split base metrics into more specific metrics



cube 4.1.1 livedvd2: scorep_bt-mz_B_4x4_sum/summary.cubex

File Display Topology Help

```
Absolute
Metric tree
0.00 Time
  303.73 Execution
    0.00 MPI
      0.01 Synchronization
      11.28 Communication
      0.00 File I/O
      0.84 Init/Exit
    0.00 OMP
      0.00 Flush
      0.48 Management
      32.68 Synchronization
  38.64 Idle threads
2.93e6 Visits
0 Synchronizations
0 Communications
0 Bytes transferred
  0 Sent
    5.27e8 Point-to-point
    608 Collective
  0 Received
    5.27e8 Point-to-point
    608 Collective
0.00          32.68 (8.43%)          387.66
```

```
Absolute
Call tree | Flat view
0.02 exact_rhs_
0.42 exch_qbc_
0.00 adi_
  4.99 compute_rhs_
  0.00 x_solve_
    0.00 !$omp parallel @x_sol
      0.00 !$omp do @x_solve
      9.09 !$omp implicit barri
  0.00 y_solve_
    0.00 !$omp parallel @y_sol
      0.00 !$omp do @y_solve
      9.28 !$omp implicit barri
  0.00 z_solve_
    0.00 !$omp parallel @z_sol
      0.00 !$omp do @z_solve
      8.53 !$omp implicit barri
  0.21 add_
0.00 MPI_Barrier
0.03 verify_
0.00 MPI_Reduce
0.00 print_results_
0.00 MPI_Finalize
0.00          9.09 (27.83%)          32.68
```

```
Absolute
System tree | Box Plot
- generic cluster
  - i06r01c20
    - MPI Rank 0
      0.05 CPU thread 0
      1.43 CPU thread 1
      0.02 CPU thread 2
      0.01 CPU thread 3
    - MPI Rank 1
      0.04 CPU thread 0
      0.01 CPU thread 1
      0.01 CPU thread 2
      1.44 CPU thread 3
    - MPI Rank 2
      0.03 CPU thread 0
      2.47 CPU thread 1
      1.03 CPU thread 2
      1.03 CPU thread 3
    - MPI Rank 3
      0.04 CPU thread 0
      1.45 CPU thread 1
      0.01 CPU thread 2
      0.01 CPU thread 3
0.00          2.47 (27.11%)          9.09
```

Performance analysis steps

- 0.0 Reference preparation for validation

- 1.0 Program instrumentation
- 1.1 Summary measurement collection
- 1.2 Summary analysis report examination

- 2.0 Summary experiment scoring
- 2.1 Summary measurement collection with filtering
- 2.2 Filtered summary analysis report examination

- 3.0 Event trace collection
- 3.1 Event trace examination & analysis

BT-MZ trace measurement collection...

```
% cd bin.scorep
% cp ../jobscript/stampede2/scalasca.sbatch .
% vi scalasca.sbatch

# Score-P measurement configuration
export SCOREP_FILTERING_FILE=../config/scorep.filt
export SCOREP_TOTAL_MEMORY=50M
export SCOREP_METRIC_PAPI=PAPI_TOT_INS,PAPI_TOT_CYC

# Scalasca configuration
export SCAN_ANALYZE_OPTS="--time-correct"

# Run the application using Scalasca nexus
scalasca -analyze -t ibrun ./bt-mz_${CLASS}.${PROCS}
```

- Change to directory with the executable and edit the job script
- Add "-t" to the scalasca -analyze command

```
% sbatch scalasca.sbatch
```

- Submit the job

BT-MZ trace measurement ... collection

```
S=C=A=N: Scalasca 2.4 trace collection and analysis
S=C=A=N: Mon Aug 21 07:58:54 2017: Collect start
ibrun ./bt-mz_C.32

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

 Number of zones:  16 x  16
 Iterations: 200    dt:   0.000100
 Number of active processes:    32

 [... More application output ...]

S=C=A=N: Mon Aug 21 07:59:29 2017: Collect done (status=0) 35s
```

- Starts measurement with collection of trace files …

BT-MZ trace measurement ... analysis

```
S=C=A=N: Mon Aug 21 07:59:30 2017: Analyze start
ibrun scout.hyb ./scorep_bt-mz_C_32x4_trace/traces.otf2

Analyzing experiment archive ./scorep_bt-mz_C_32x4_trace/traces.otf2

Opening experiment archive ... done (0.040s).
Reading definition data    ... done (0.127s).
Reading event trace data   ... done (0.726s).
Preprocessing              ... done (0.311s).
Timestamp correction       ... done (0.556s).
Analyzing trace data       ... done (15.144s).
Writing analysis report    ... done (0.738s).

Total processing time     : 17.754s
S=C=A=N: Mon Aug 21 07:59:50 2017: Analyze done (status=0) 20s
```

- Continues with automatic (parallel) analysis of trace files

BT-MZ trace analysis report exploration

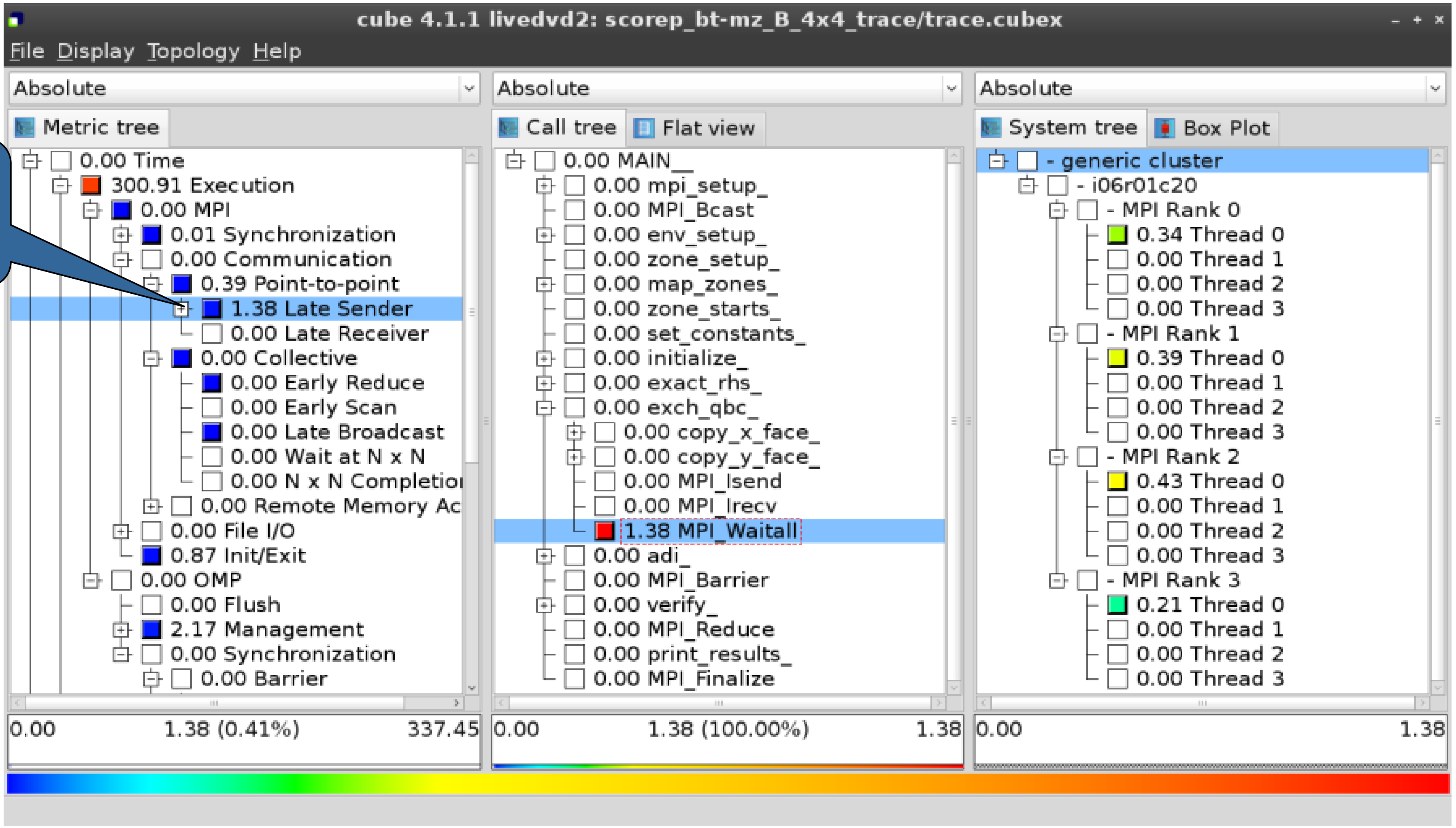
- Produces trace analysis report in the experiment directory containing trace-based wait-state metrics

```
% square  scorep_bt-mz_C_32x4_trace
INFO: Post-processing runtime summarization result...
INFO: Post-processing trace analysis report...
INFO: Displaying ./scorep_bt-mz_C_32x4_trace/trace.cubex...

              [GUI showing trace analysis report]
```

Hint:
Run ‘square -s’ first and then copy ‘trace.cubex’ to Live-DVD environment using ‘scp’ to improve responsiveness of GUI

Post-processed trace analysis report

Additional trace-based metrics in metric hierarchy


cube 4.1.1 livedvd2: scorep_bt-mz_B_4x4_trace/trace.cubex

File Display Topology Help

Absolute | Absolute | Absolute

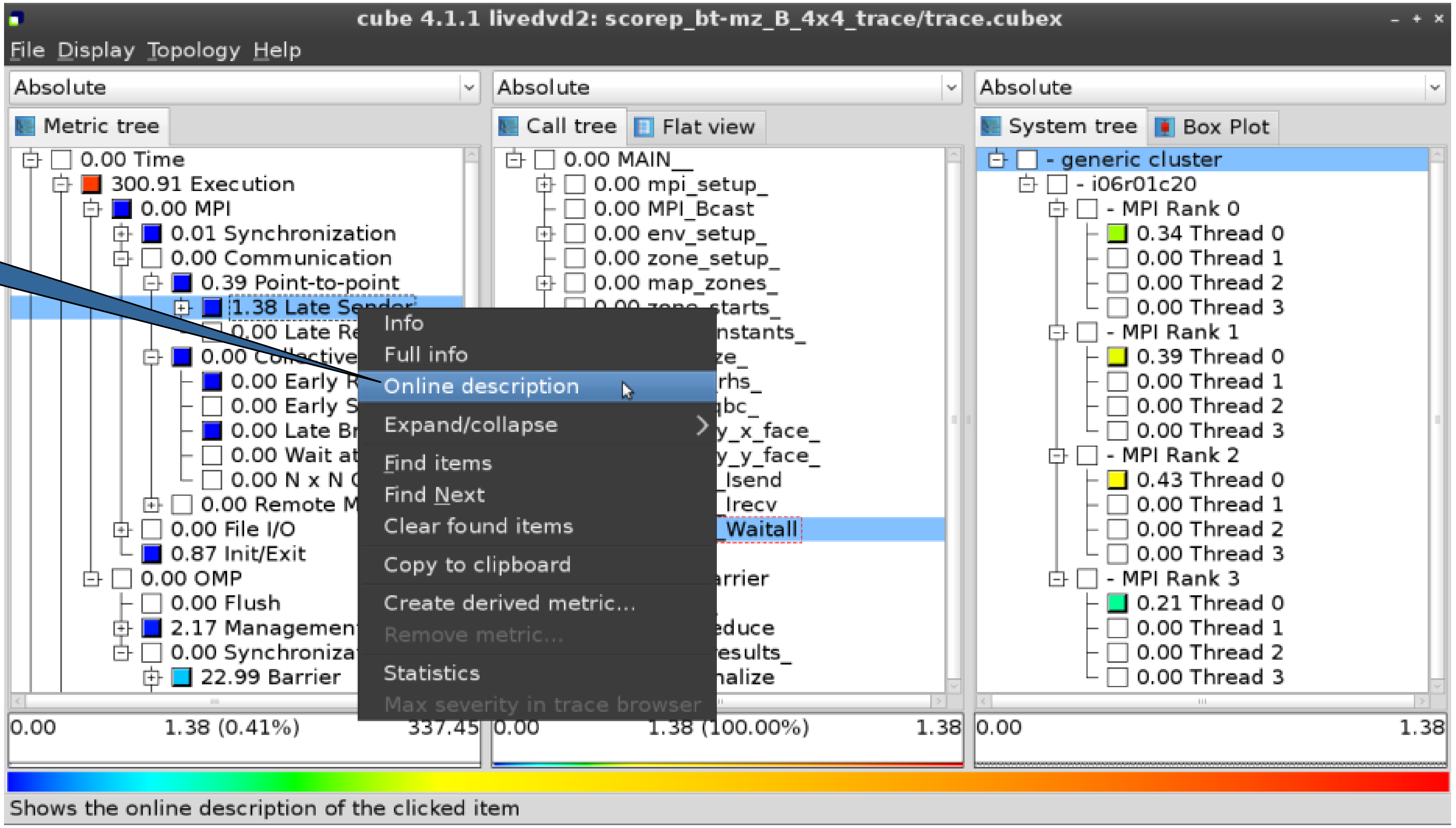
Metric tree:
```
0.00 Time
  300.91 Execution
    0.00 MPI
      0.01 Synchronization
      0.00 Communication
        0.39 Point-to-point
          1.38 Late Sender
          0.00 Late Receiver
        0.00 Collective
          0.00 Early Reduce
          0.00 Early Scan
          0.00 Late Broadcast
          0.00 Wait at N x N
          0.00 N x N Completio
        0.00 Remote Memory Ac
      0.00 File I/O
      0.87 Init/Exit
    0.00 OMP
      0.00 Flush
      2.17 Management
      0.00 Synchronization
        0.00 Barrier
```
0.00   1.38 (0.41%)   337.45

Call tree / Flat view:
```
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 set_constants_
  0.00 initialize_
  0.00 exact_rhs_
  0.00 exch_qbc_
    0.00 copy_x_face_
    0.00 copy_y_face_
    0.00 MPI_Isend
    0.00 MPI_Irecv
    1.38 MPI_Waitall
  0.00 adi_
  0.00 MPI_Barrier
  0.00 verify_
  0.00 MPI_Reduce
  0.00 print_results_
  0.00 MPI_Finalize
```
0.00   1.38 (100.00%)   1.38

System tree / Box Plot:
```
- generic cluster
  - i06r01c20
    - MPI Rank 0
      0.34 Thread 0
      0.00 Thread 1
      0.00 Thread 2
      0.00 Thread 3
    - MPI Rank 1
      0.39 Thread 0
      0.00 Thread 1
      0.00 Thread 2
      0.00 Thread 3
    - MPI Rank 2
      0.43 Thread 0
      0.00 Thread 1
      0.00 Thread 2
      0.00 Thread 3
    - MPI Rank 3
      0.21 Thread 0
      0.00 Thread 1
      0.00 Thread 2
      0.00 Thread 3
```
0.00   1.38

Online metric description

Access online metric description via context menu


cube 4.1.1 livedvd2: scorep_bt-mz_B_4x4_trace/trace.cubex

```
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 Re
        0.00 Collective
          0.00 Early R
          0.00 Early S
          0.00 Late Br
          0.00 Wait at
          0.00 N x N C
        0.00 Remote M
      0.00 File I/O
      0.87 Init/Exit
    0.00 OMP
      0.00 Flush
      2.17 Managemen
      0.00 Synchroniza
        22.99 Barrier
```

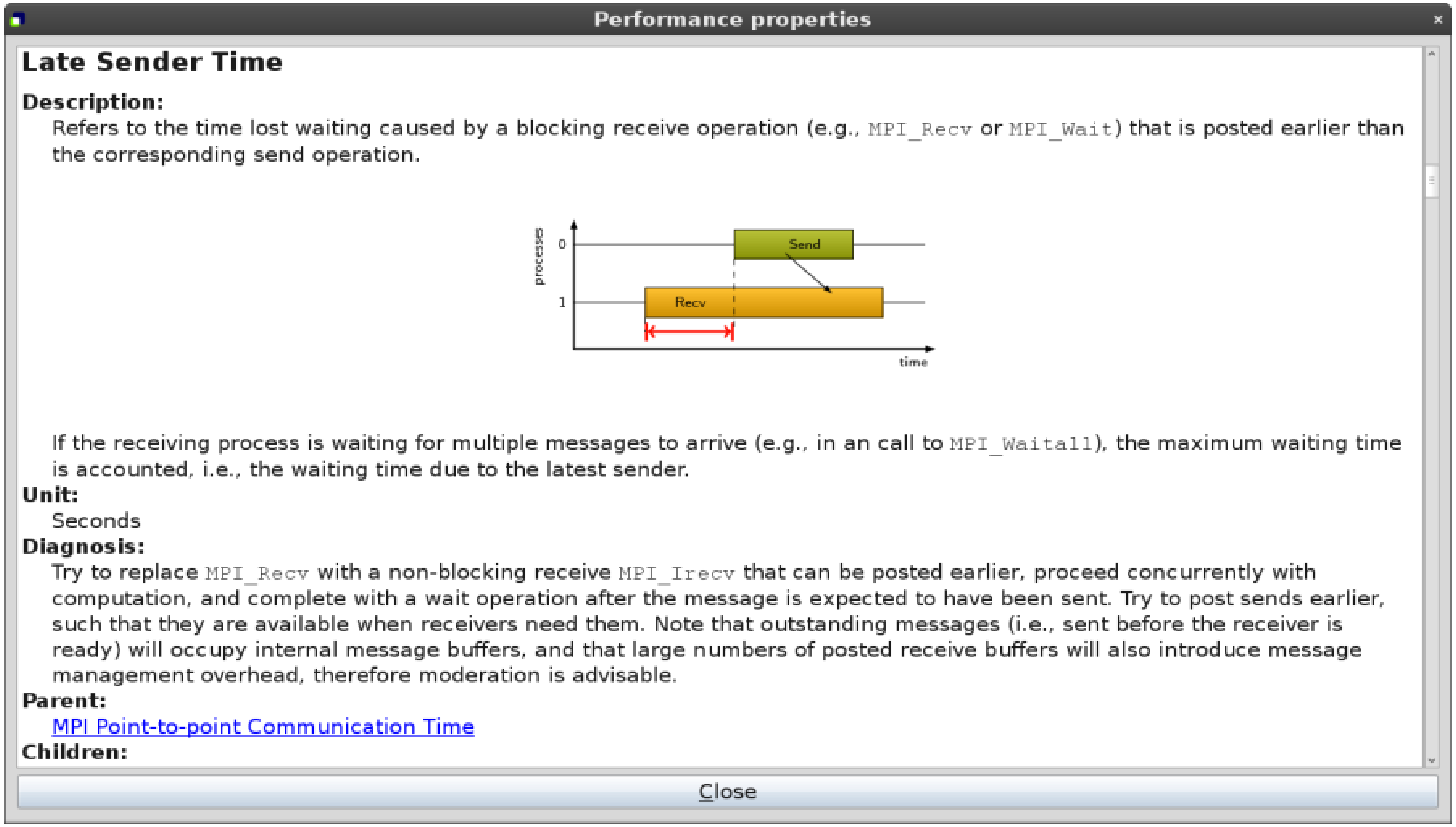
```
0.00 MAIN__
  0.00 mpi_setup_
  0.00 MPI_Bcast
  0.00 env_setup_
  0.00 zone_setup_
  0.00 map_zones_
  0.00 zone_starts
```

```
- generic cluster
  - i06r01c20
    - MPI Rank 0
      0.34 Thread 0
      0.00 Thread 1
      0.00 Thread 2
      0.00 Thread 3
    - MPI Rank 1
      0.39 Thread 0
      0.00 Thread 1
      0.00 Thread 2
      0.00 Thread 3
    - MPI Rank 2
      0.43 Thread 0
      0.00 Thread 1
      0.00 Thread 2
      0.00 Thread 3
    - 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 (100.00%)    1.38   0.00    1.38
```

Shows the online description of the clicked item

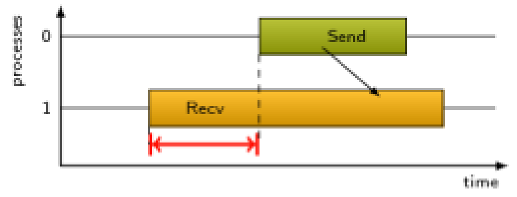
Online metric description


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.



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.

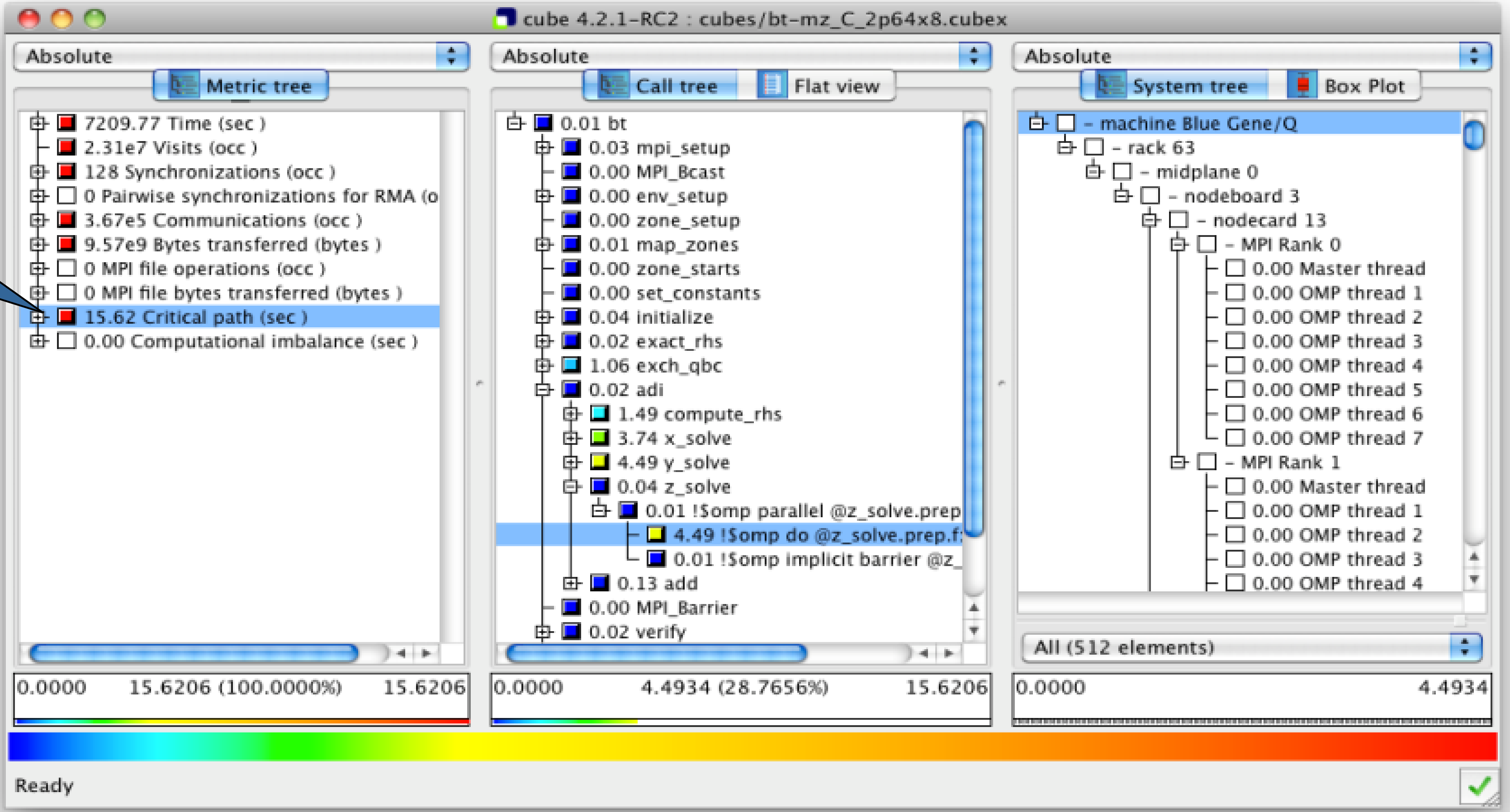
Parent:

MPI Point-to-point Communication Time

Children:

Critical-path analysis

Critical-path profile shows wall-clock time impact



cube 4.2.1-RC2 : cubes/bt-mz_C_2p64x8.cubex

Absolute | Absolute | Absolute

Metric tree | Call tree | Flat view | System tree | Box Plot

```
7209.77 Time (sec )
2.31e7 Visits (occ )
128 Synchronizations (occ )
0 Pairwise synchronizations for RMA (o
3.67e5 Communications (occ )
9.57e9 Bytes transferred (bytes )
0 MPI file operations (occ )
0 MPI file bytes transferred (bytes )
15.62 Critical path (sec )
0.00 Computational imbalance (sec )
```

```
0.01 bt
  0.03 mpi_setup
  0.00 MPI_Bcast
  0.00 env_setup
  0.00 zone_setup
  0.01 map_zones
  0.00 zone_starts
  0.00 set_constants
  0.04 initialize
  0.02 exact_rhs
  1.06 exch_qbc
  0.02 adi
    1.49 compute_rhs
    3.74 x_solve
    4.49 y_solve
    0.04 z_solve
      0.01 !$omp parallel @z_solve.prep
        4.49 !$omp do @z_solve.prep.f
        0.01 !$omp implicit barrier @z_
    0.13 add
  0.00 MPI_Barrier
  0.02 verify
```

```
- machine Blue Gene/Q
  - rack 63
    - midplane 0
      - nodeboard 3
        - nodecard 13
          - MPI Rank 0
            0.00 Master thread
            0.00 OMP thread 1
            0.00 OMP thread 2
            0.00 OMP thread 3
            0.00 OMP thread 4
            0.00 OMP thread 5
            0.00 OMP thread 6
            0.00 OMP thread 7
          - MPI Rank 1
            0.00 Master thread
            0.00 OMP thread 1
            0.00 OMP thread 2
            0.00 OMP thread 3
            0.00 OMP thread 4
```

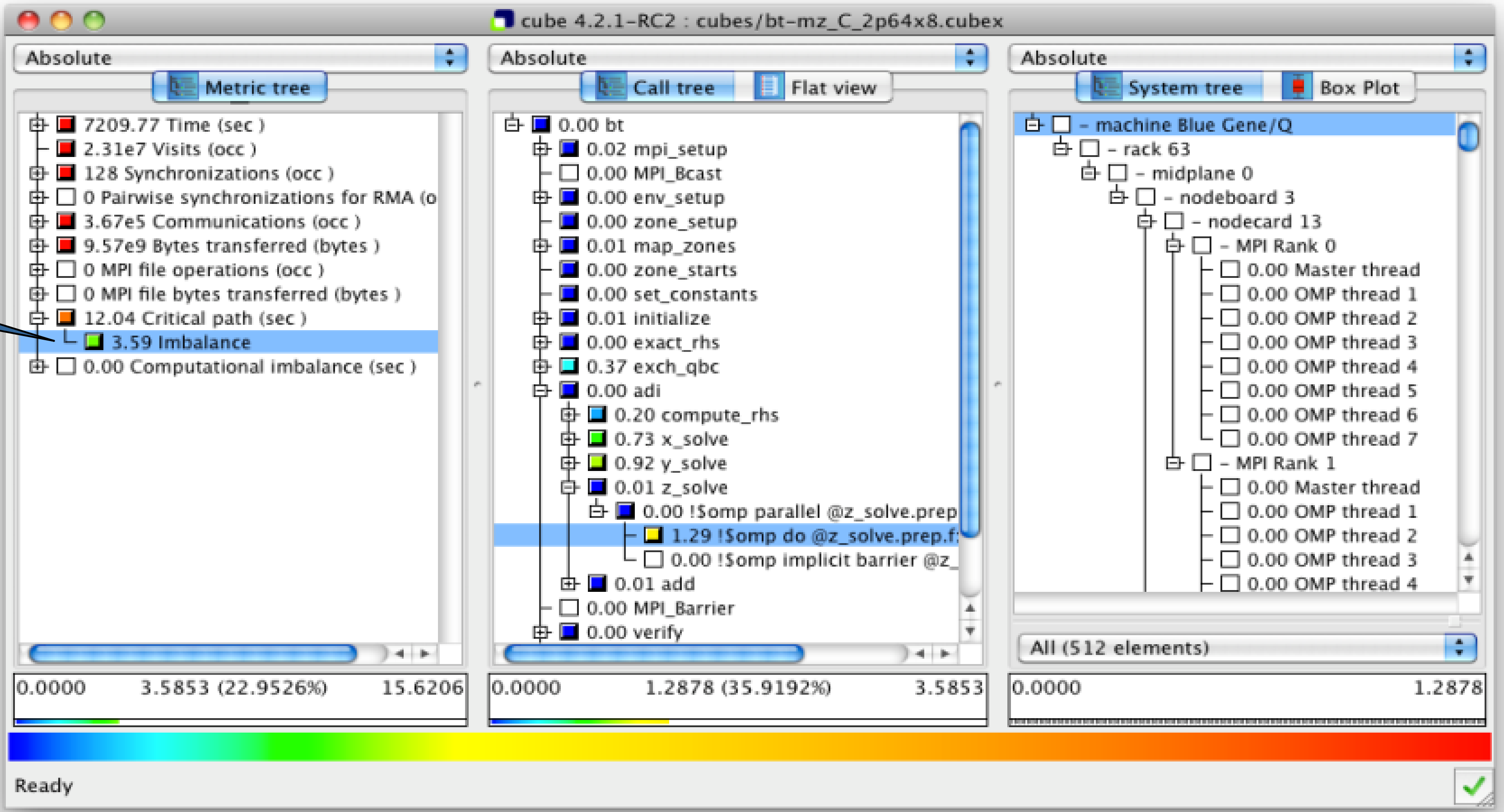
All (512 elements)

```
0.0000    15.6206 (100.0000%)    15.6206
0.0000    4.4934 (28.7656%)    15.6206
0.0000    4.4934
```

Ready

Critical-path analysis

Critical-path imbalance highlights inefficient parallelism


cube 4.2.1-RC2 : cubes/bt-mz_C_2p64x8.cubex

Absolute | Metric tree

```
7209.77 Time (sec )
2.31e7 Visits (occ )
128 Synchronizations (occ )
0 Pairwise synchronizations for RMA (o
3.67e5 Communications (occ )
9.57e9 Bytes transferred (bytes )
0 MPI file operations (occ )
0 MPI file bytes transferred (bytes )
12.04 Critical path (sec )
  3.59 Imbalance
0.00 Computational imbalance (sec )
```

0.0000   3.5853 (22.9526%)   15.6206

Absolute | Call tree | Flat view

```
0.00 bt
  0.02 mpi_setup
  0.00 MPI_Bcast
  0.00 env_setup
  0.00 zone_setup
  0.01 map_zones
  0.00 zone_starts
  0.00 set_constants
  0.01 initialize
  0.00 exact_rhs
  0.37 exch_qbc
  0.00 adi
    0.20 compute_rhs
    0.73 x_solve
    0.92 y_solve
    0.01 z_solve
      0.00 !$omp parallel @z_solve.prep
        1.29 !$omp do @z_solve.prep.f
        0.00 !$omp implicit barrier @z_
    0.01 add
  0.00 MPI_Barrier
  0.00 verify
```

0.0000   1.2878 (35.9192%)   3.5853

Absolute | System tree | Box Plot

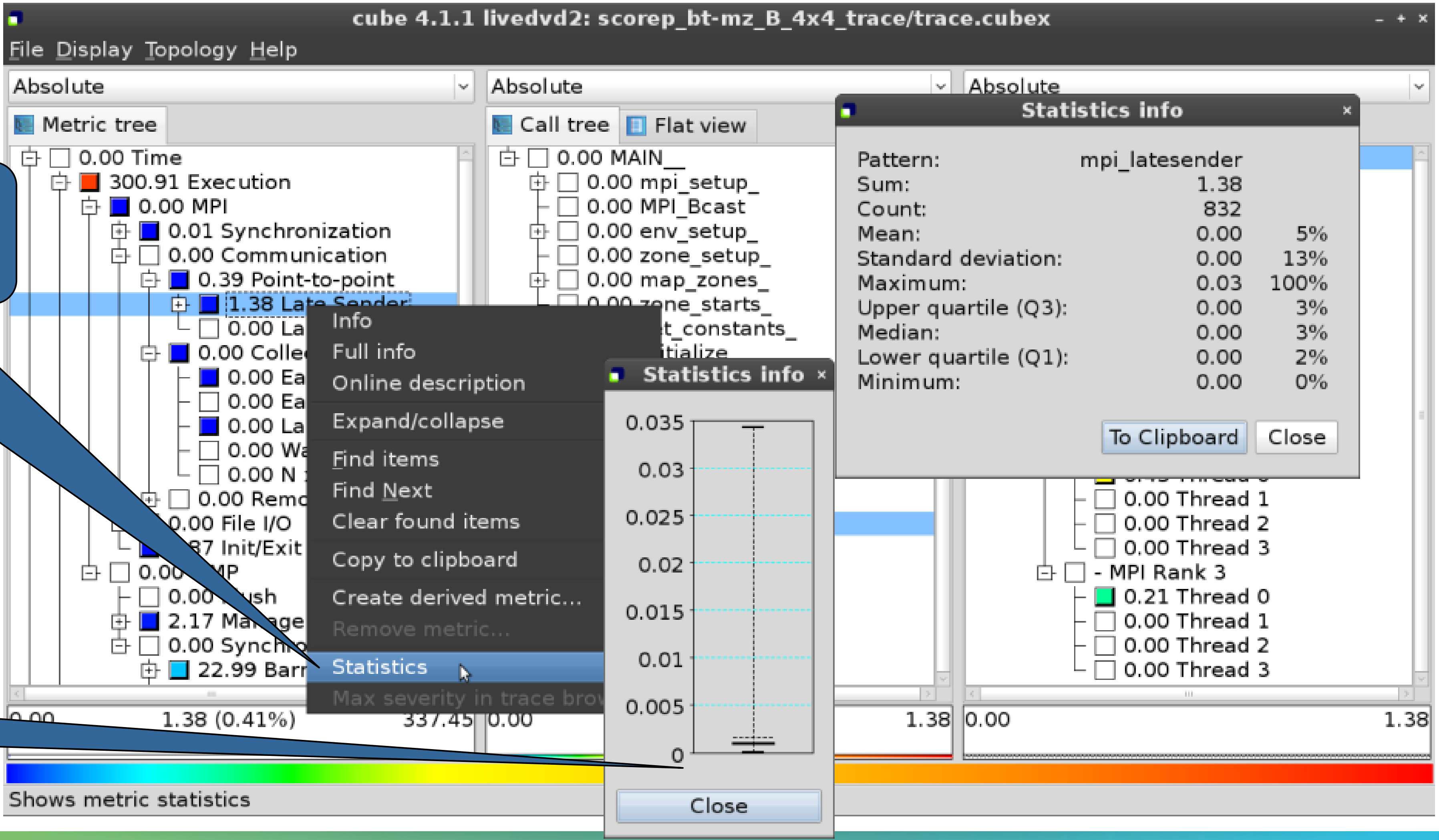
```
- machine Blue Gene/Q
  - rack 63
    - midplane 0
      - nodeboard 3
        - nodecard 13
          - MPI Rank 0
            0.00 Master thread
            0.00 OMP thread 1
            0.00 OMP thread 2
            0.00 OMP thread 3
            0.00 OMP thread 4
            0.00 OMP thread 5
            0.00 OMP thread 6
            0.00 OMP thread 7
          - MPI Rank 1
            0.00 Master thread
            0.00 OMP thread 1
            0.00 OMP thread 2
            0.00 OMP thread 3
            0.00 OMP thread 4
```

All (512 elements)

0.0000   1.2878

Ready

Pattern instance statistics



Statistics info

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%

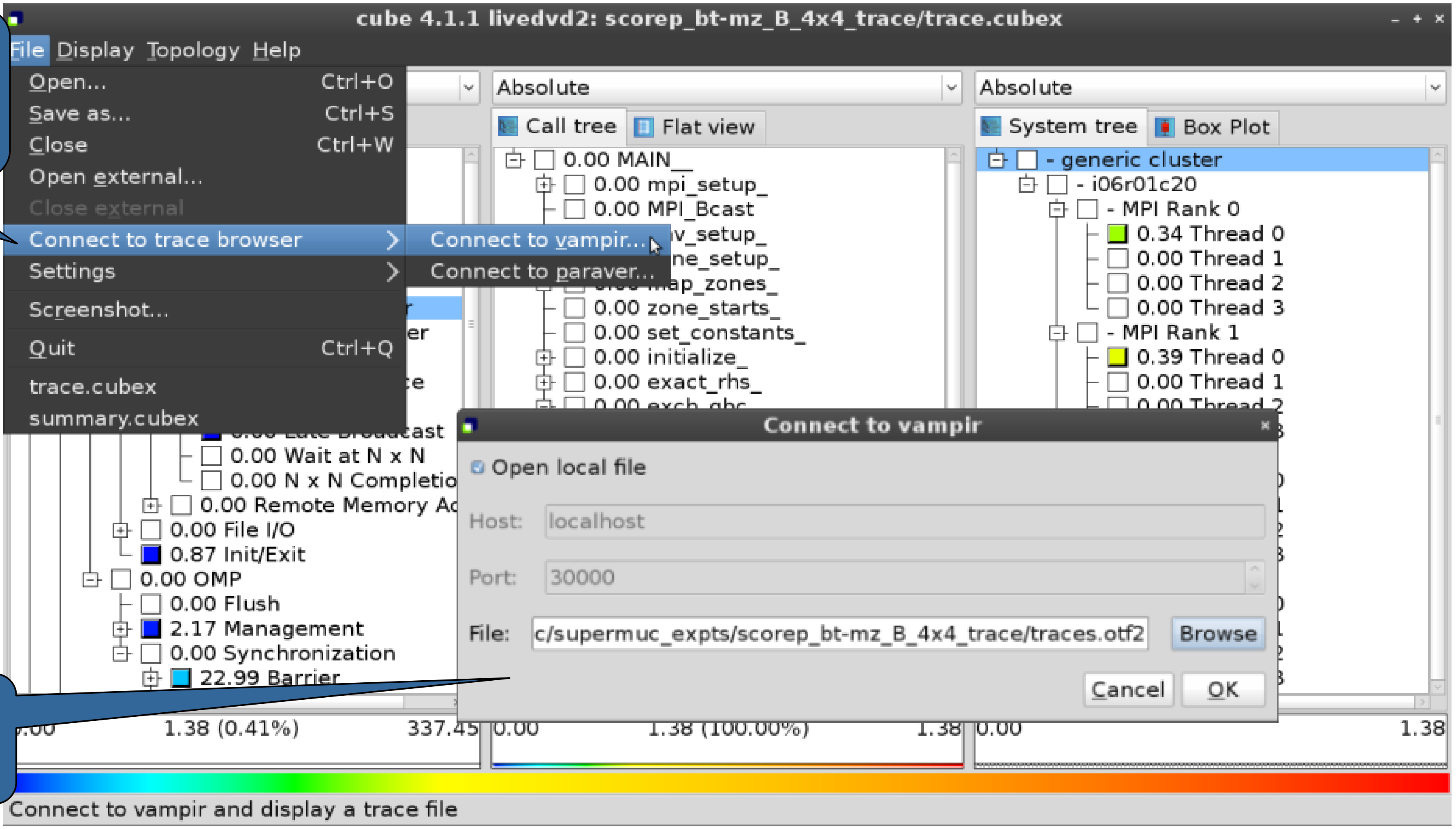
Access pattern instance statistics via context menu

Click to get statistics details

Connect to Vampir trace browser

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

…and select trace file from the experiment directory



cube 4.1.1 livedvd2: scorep_bt-mz_B_4x4_trace/trace.cubex

File Display Topology Help

Open...	Ctrl+O
Save as...	Ctrl+S
Close	Ctrl+W
Open external...	
Close external	
Connect to trace browser	
Settings	
Screenshot...	
Quit	Ctrl+Q
trace.cubex	
summary.cubex	

Connect to vampir...
Connect to paraver...

Connect to vampir

Open local file

Host: localhost

Port: 30000

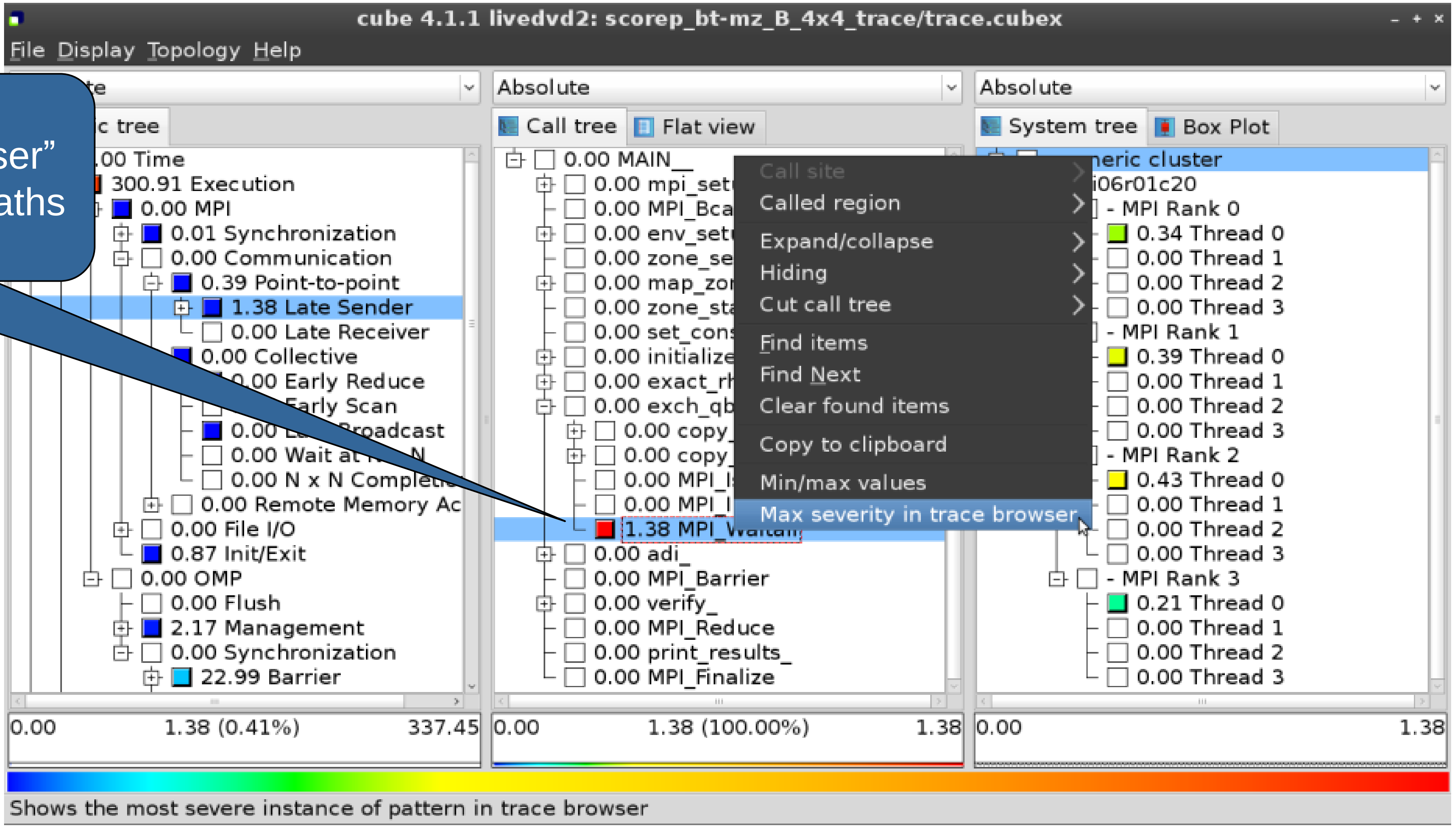
File: c/supermuc_expts/scorep_bt-mz_B_4x4_trace/traces.otf2

Browse | Cancel | OK

Connect to vampir and display a trace file

Show most severe pattern instances

Select
“Max severity in trace browser”
from context menu of call paths
marked with a red frame

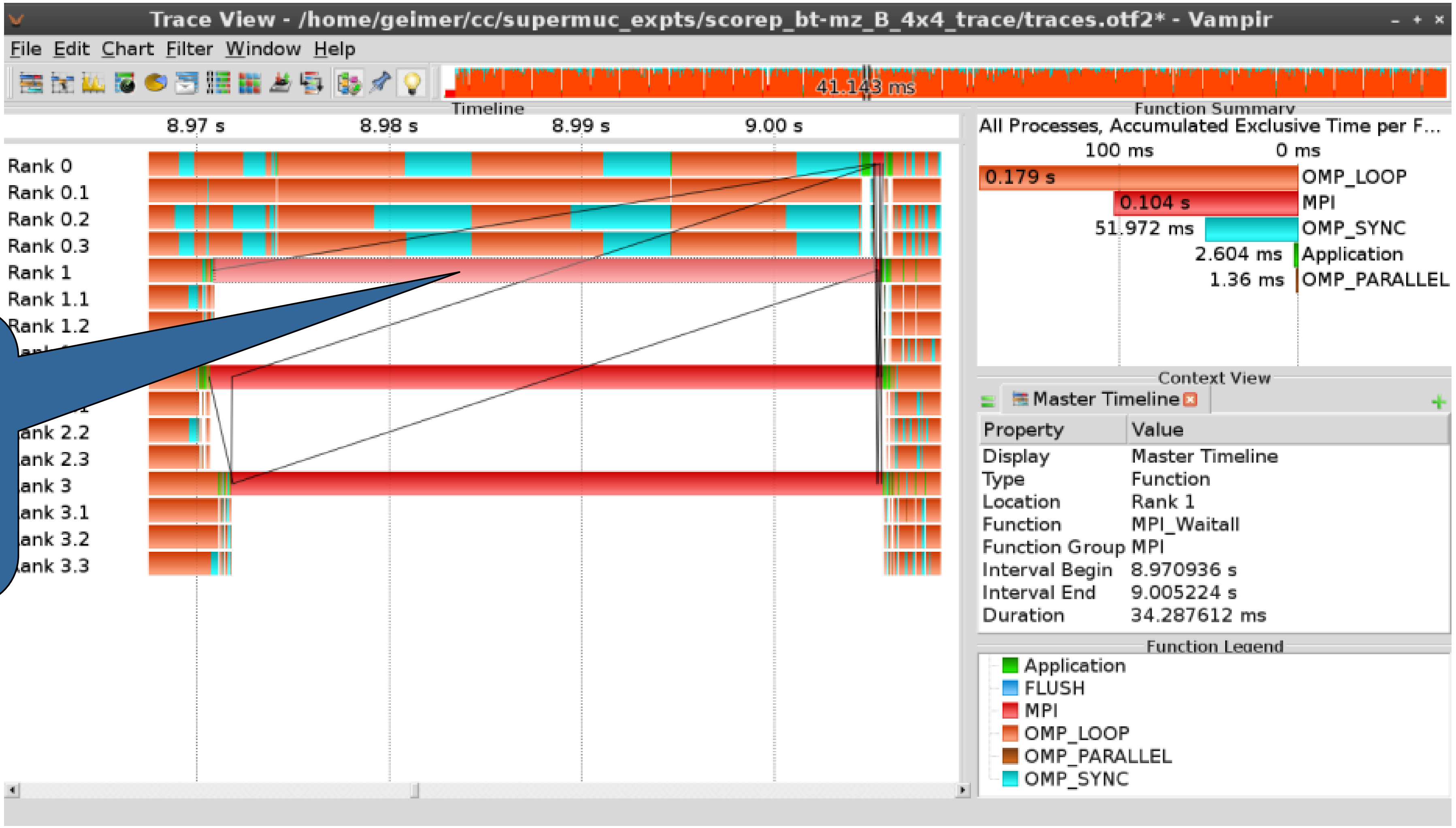


```
cube 4.1.1 livedvd2: scorep_bt-mz_B_4x4_trace/trace.cubex
File  Display  Topology  Help
```

```
...te                              Absolute                          Absolute
...c tree                          Call tree | Flat view             System tree | Box Plot
.00 Time                           0.00 MAIN__                       ...neric cluster
300.91 Execution                     0.00 mpi_setu                     ...i06r01c20
  0.00 MPI                           0.00 MPI_Bca                        - MPI Rank 0
    0.01 Synchronization             0.00 env_setu                         0.34 Thread 0
    0.00 Communication               0.00 zone_se                          0.00 Thread 1
      0.39 Point-to-point            0.00 map_zor                          0.00 Thread 2
        1.38 Late Sender             0.00 zone_sta                         0.00 Thread 3
        0.00 Late Receiver           0.00 set_cons                       - MPI Rank 1
      0.00 Collective                0.00 initialize                       0.39 Thread 0
        0.00 Early Reduce            0.00 exact_rh                         0.00 Thread 1
        ...arly Scan                 0.00 exch_qb                          0.00 Thread 2
        0.00 ... Broadcast             0.00 copy                           0.00 Thread 3
        0.00 Wait at ... N             0.00 copy                         - MPI Rank 2
        0.00 N x N Complet...          0.00 MPI_I                          0.43 Thread 0
      0.00 Remote Memory Ac            0.00 MPI_I                          0.00 Thread 1
    0.00 File I/O                      1.38 MPI_Waitall                    0.00 Thread 2
    0.87 Init/Exit                   0.00 adi_                             0.00 Thread 3
  0.00 OMP                           0.00 MPI_Barrier                    - MPI Rank 3
    0.00 Flush                       0.00 verify_                          0.21 Thread 0
    2.17 Management                  0.00 MPI_Reduce                       0.00 Thread 1
    0.00 Synchronization             0.00 print_results_                   0.00 Thread 2
      22.99 Barrier                  0.00 MPI_Finalize                     0.00 Thread 3

0.00   1.38 (0.41%)   337.45       0.00   1.38 (100.00%)   1.38        0.00              1.38
```

Shows the most severe instance of pattern in trace browser

Investigate most severe instance in Vampir



Property	Value
Display	Master Timeline
Type	Function
Location	Rank 1
Function	MPI_Waitall
Function Group	MPI
Interval Begin	8.970936 s
Interval End	9.005224 s
Duration	34.287612 ms

Function Legend:
- 🟩 Application
- 🟦 FLUSH
- 🟥 MPI
- 🟧 OMP_LOOP
- 🟫 OMP_PARALLEL
- 🟦 OMP_SYNC

Vampir will automatically zoom to the worst instance in multiple steps (i.e., undo zoom provides more context)

Scalasca Trace Tools: Further information

- Collection of trace-based performance tools
  - Specifically designed for large-scale systems
  - Features an automatic trace analyzer providing wait-state, critical-path, and delay analysis
  - Supports MPI, OpenMP, POSIX threads, and hybrid MPI+OpenMP/Pthreads
- Available under 3-clause BSD open-source license
- Documentation & sources:
  - http://www.scalasca.org
- Contact:
  - mailto: scalasca@fz-juelich.de

Performance Analysis with Vampir

Ronny Tschüter, Bert Wesarg, Matthias Weber
Technische Universität Dresden

allinea · JÜLICH Forschungszentrum · lrz Leibniz Supercomputing Centre · TECHNISCHE UNIVERSITÄT DARMSTADT · TUM Technische Universität München · UNIVERSITY OF OREGON · THE UNIVERSITY of TENNESSEE · BSC Barcelona Supercomputing Center · Lawrence Livermore National Laboratory · RWTH AACHEN UNIVERSITY · TECHNISCHE UNIVERSITÄT DRESDEN · UNIVERSITÉ DE VERSAILLES ST-QUENTIN-EN-YVELINES · Universität Stuttgart

Outline

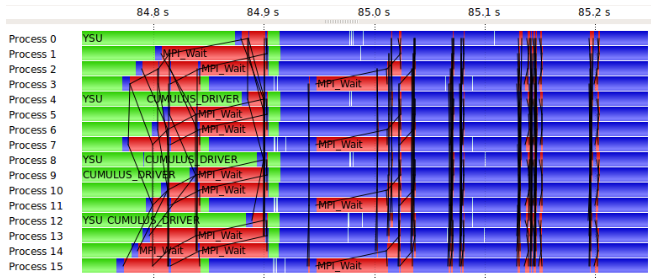
- **Part I: Welcome to the Vampir Tool Suite**
  - Mission
  - Event Trace Visualization
  - Vampir & VampirServer
  - The Vampir Displays
- **Part II: Vampir Hands-On**
  - Visualizing and analyzing NPB-MZ-MPI / BT

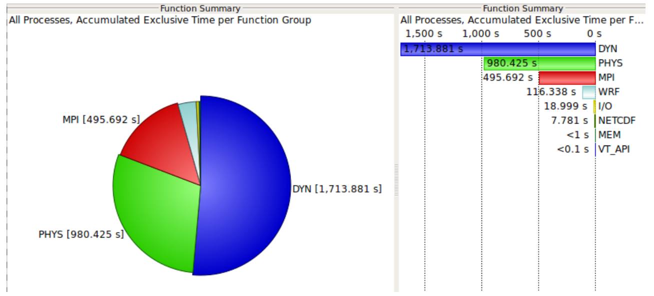
Event Trace Visualization with Vampir

- Alternative and supplement to automatic analysis
- Show dynamic run-time behavior graphically at any level of detail
- Provide statistics and performance metrics

- **Timeline charts**
  - Show application activities and communication along a time axis

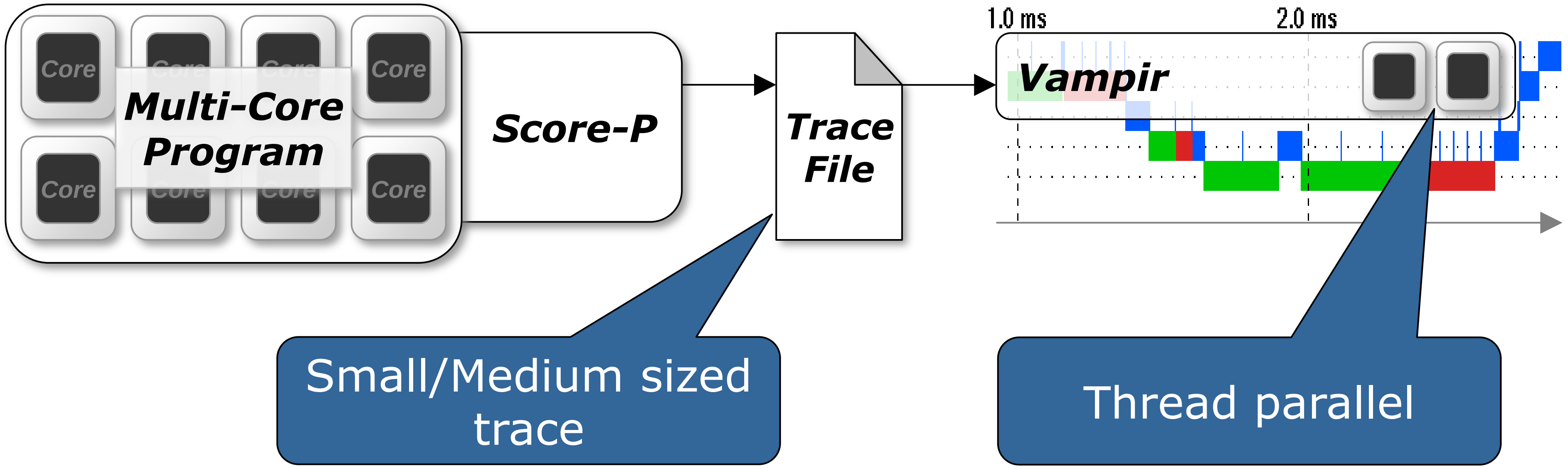
- **Summary charts**
  - Provide quantitative results for the currently selected time interval





Visualization Modes (1)
Directly on front end or local machine

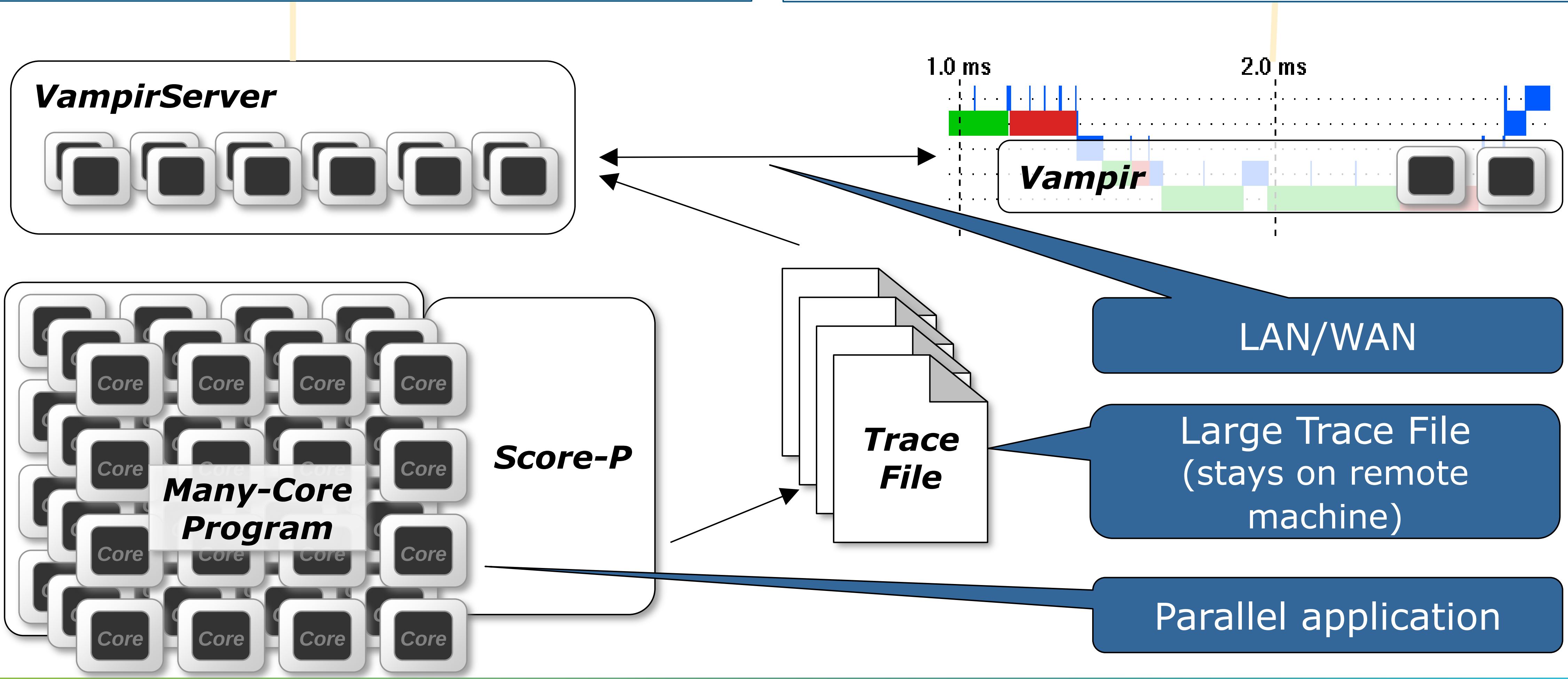
```
% vampir
```



Visualization Modes (2)
On local machine with remote VampirServer

```
% vampirserver start
```

```
% vampir
```



The main displays of Vampir

- Timeline Charts:
  - Master Timeline
  - Process Timeline
  - Counter Data Timeline
  - Performance Radar
- Summary Charts:
  - Function Summary
  - Message Summary
  - Process Summary
  - Communication Matrix View

Hands-on:
Visualizing and analyzing NPB-MZ-MPI / BT

Help! Where is my trace file?

```
% ls $SCRATCH/NPB3.3-MZ-MPI/bin.scorep/\
> scorep_bt-mz_C_32x4_trace
profile.cubex  scorep.cfg   traces/   traces.def  traces.otf2
```

- If you followed the Score-P hands-on up to the trace experiment

```
% ls ~tg828282/Tutorial/Experiments/scorep_bt-mz_C_32x4_trace
profile.cubex  scorep.cfg   traces/   traces.def  traces.otf2
```

- If you did not follow to that point, take a prepared trace

Starting VampirServer on Stampede

```
% vampirserver start
Launching VampirServer...
Submitting batch job (this might take a while)...
```

- Start VampirServer on Stampede2

Starting VampirServer on Stampede

```
% vampirserver start
Launching VampirServer...
Submitting batch job (this might take a while)...

VampirServer 9.2.0  (r10676)
Licensed to ZIH, TU Dresden (@ISC 2017)
Running 4 analysis processes... (abort with \
  vampirserver stop 28974)
VampirServer <28974> listens on: \
  c401-602.stampede2.tacc.utexas.edu:30019
```

- Start VampirServer on Stampede2

Copy host:port

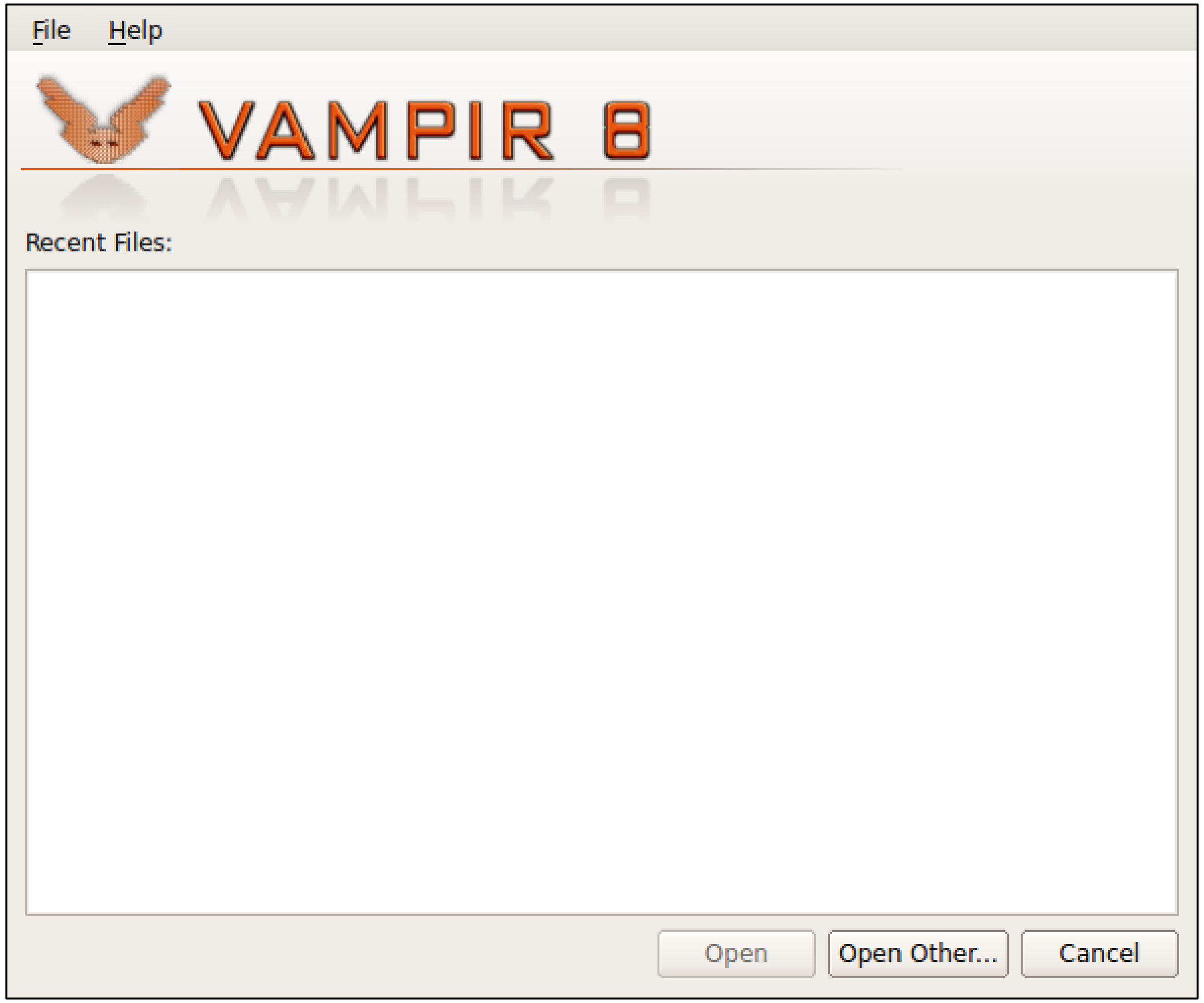
Start Vampir

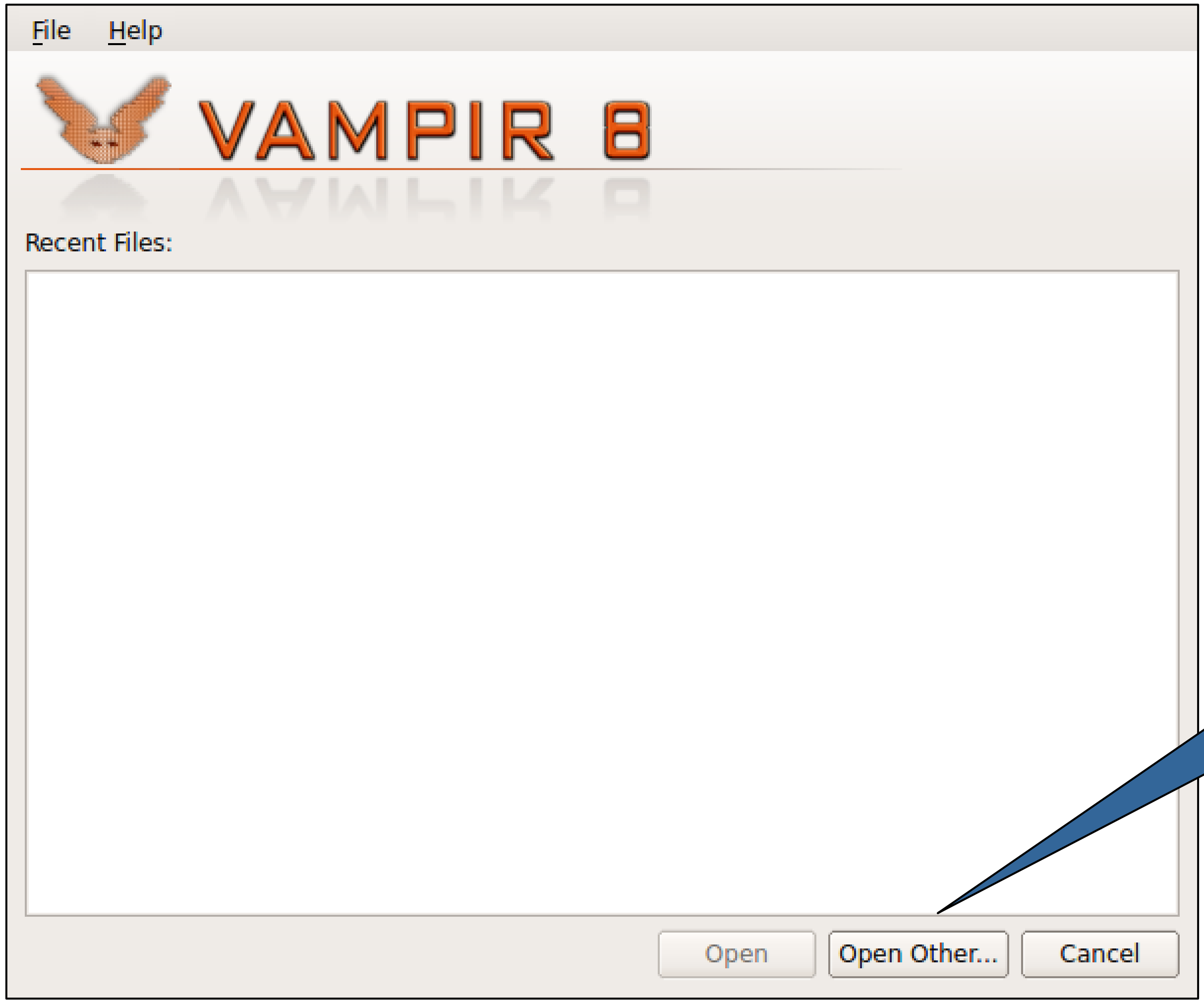
```
% ssh -N -L 30000:c401-602.stampede2.tacc.utexas.edu:30019 \
    stampede.tacc.utexas.edu
```

host:port from VampirServer output

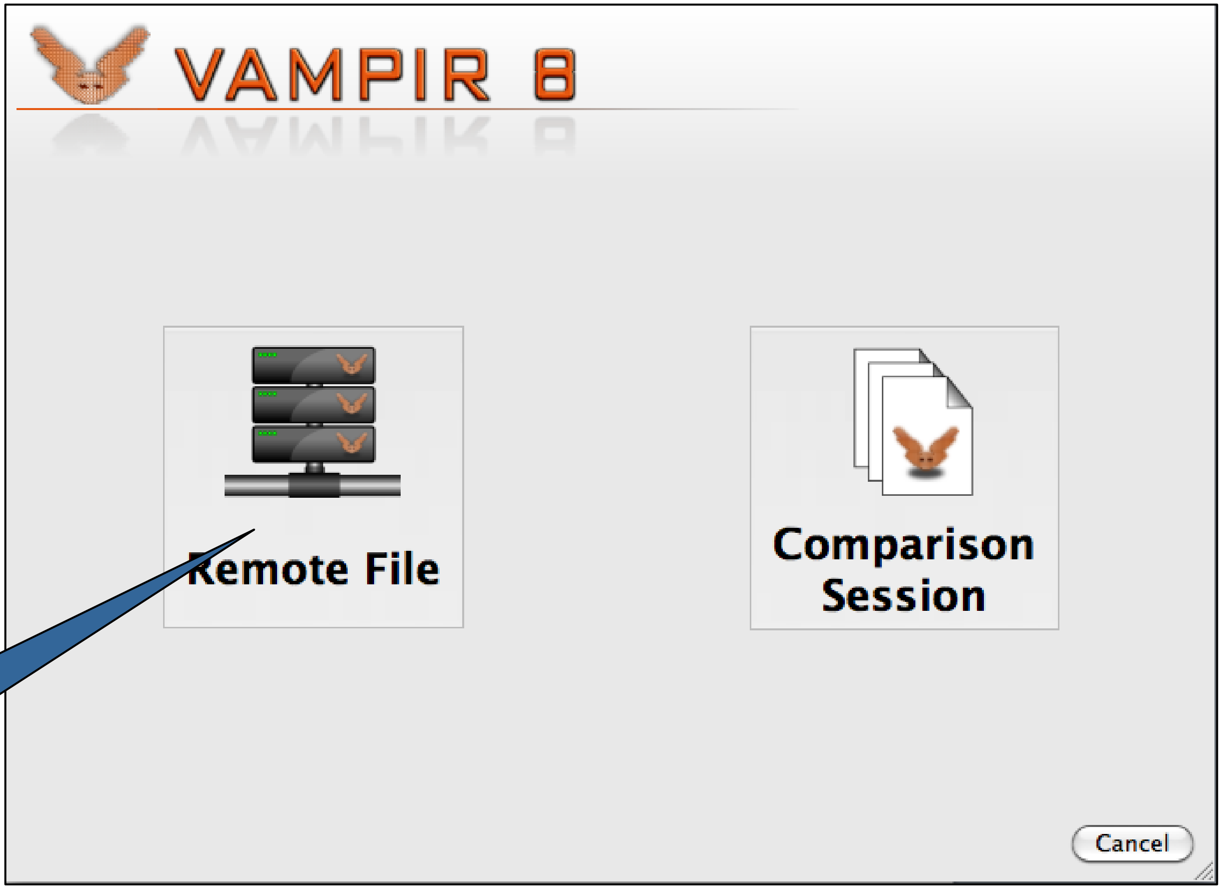
- Open a port forwarding to Stampede2 to be able to access the VampirServer

Start Vampir on local computer

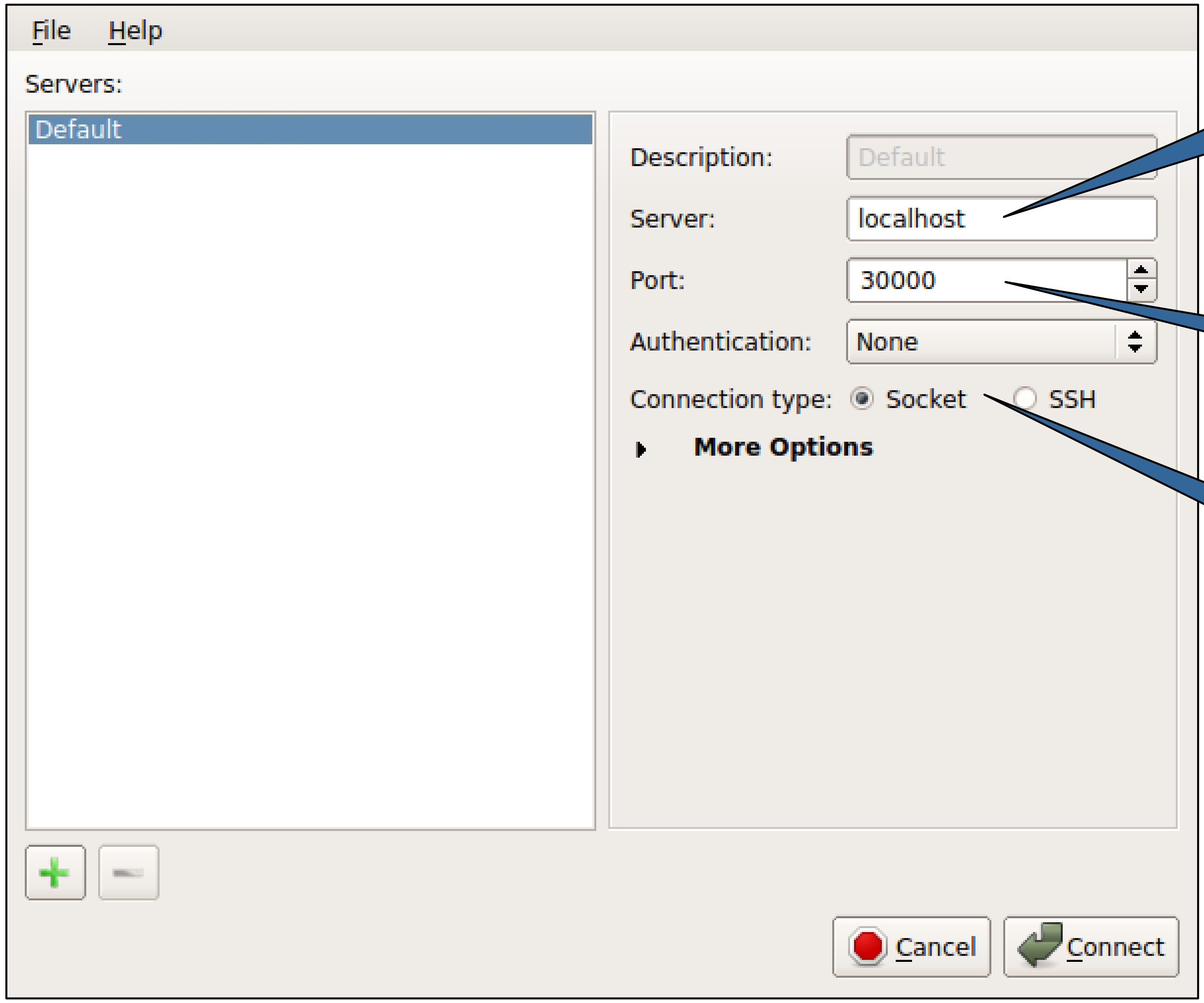




Use the "Open Other" option



Select “Remote File”



Servers: Default

Description: Default

Server: localhost

Port: 30000

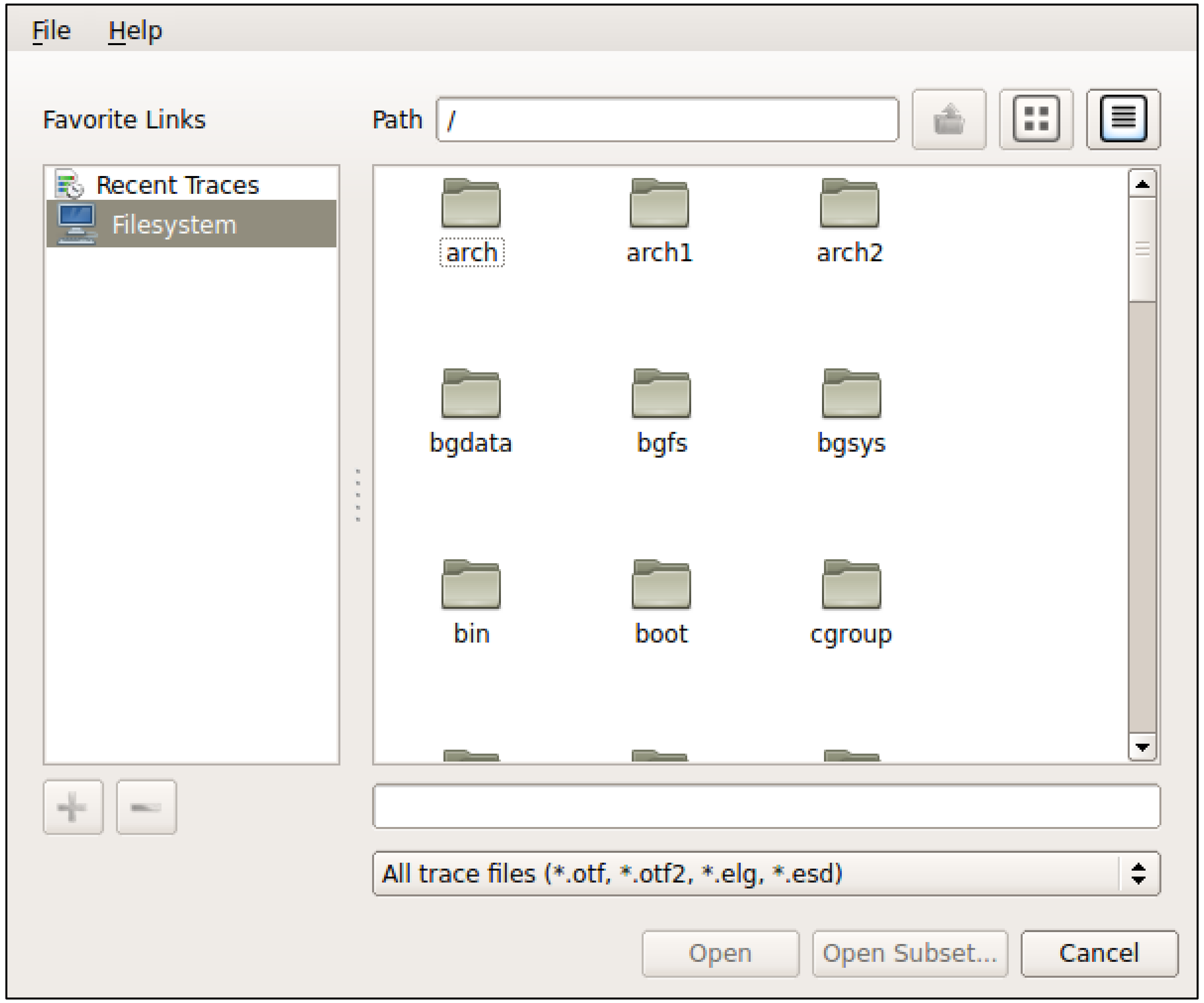
Authentication: None

Connection type: [● Socket] ○ SSH

Server is “localhost”

Port is “30000”

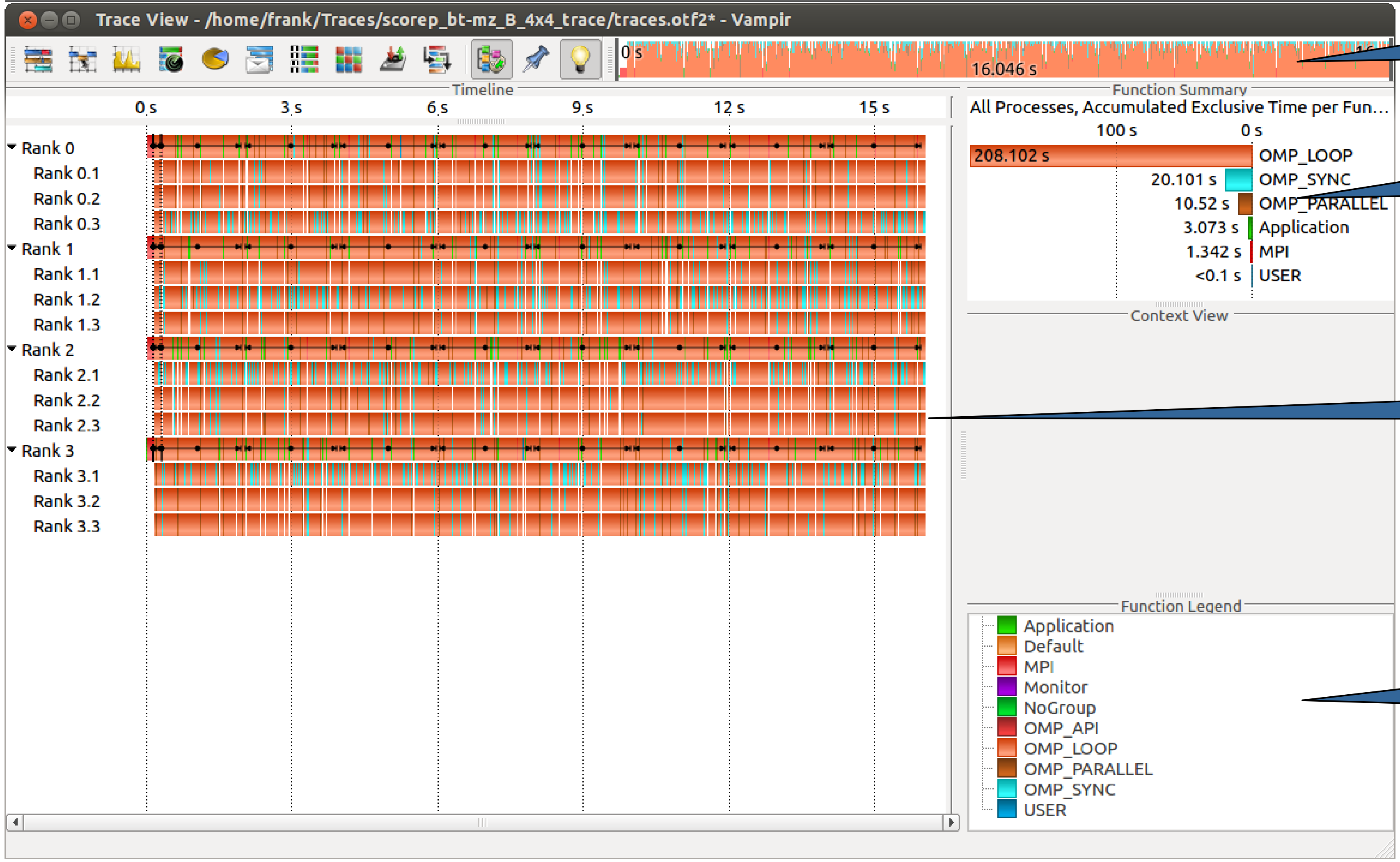
Connection type “Socket”



Path /

All trace files (*.otf, *.otf2, *.elg, *.esd)

Visualization of the NPB-MZ-MPI / BT trace



208.102 s	OMP_LOOP
20.101 s	OMP_SYNC
10.52 s	OMP_PARALLEL
3.073 s	Application
1.342 s	MPI
<0.1 s	USER

Function Legend:
- Application
- Default
- MPI
- Monitor
- NoGroup
- OMP_API
- OMP_LOOP
- OMP_PARALLEL
- OMP_SYNC
- USER

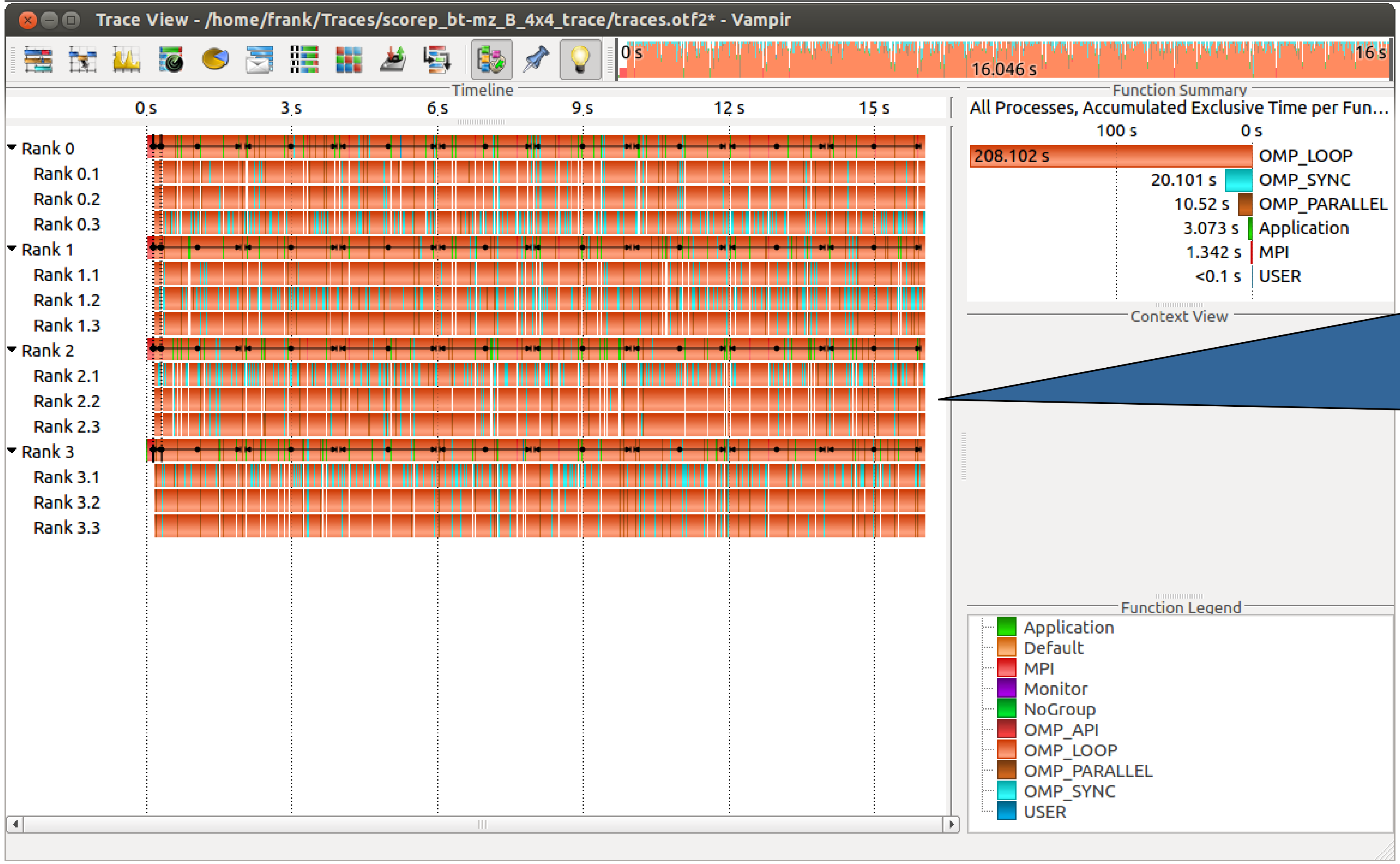
Navigation Toolbar

Function Summary

Master Timeline

Function Legend

Visualization of the NPB-MZ-MPI / BT trace
Master Timeline


Trace View - /home/frank/Traces/scorep_bt-mz_B_4x4_trace/traces.otf2* - Vampir

Function Summary

All Processes, Accumulated Exclusive Time per Fun…

Time	Function
208.102 s	OMP_LOOP
20.101 s	OMP_SYNC
10.52 s	OMP_PARALLEL
3.073 s	Application
1.342 s	MPI
<0.1 s	USER

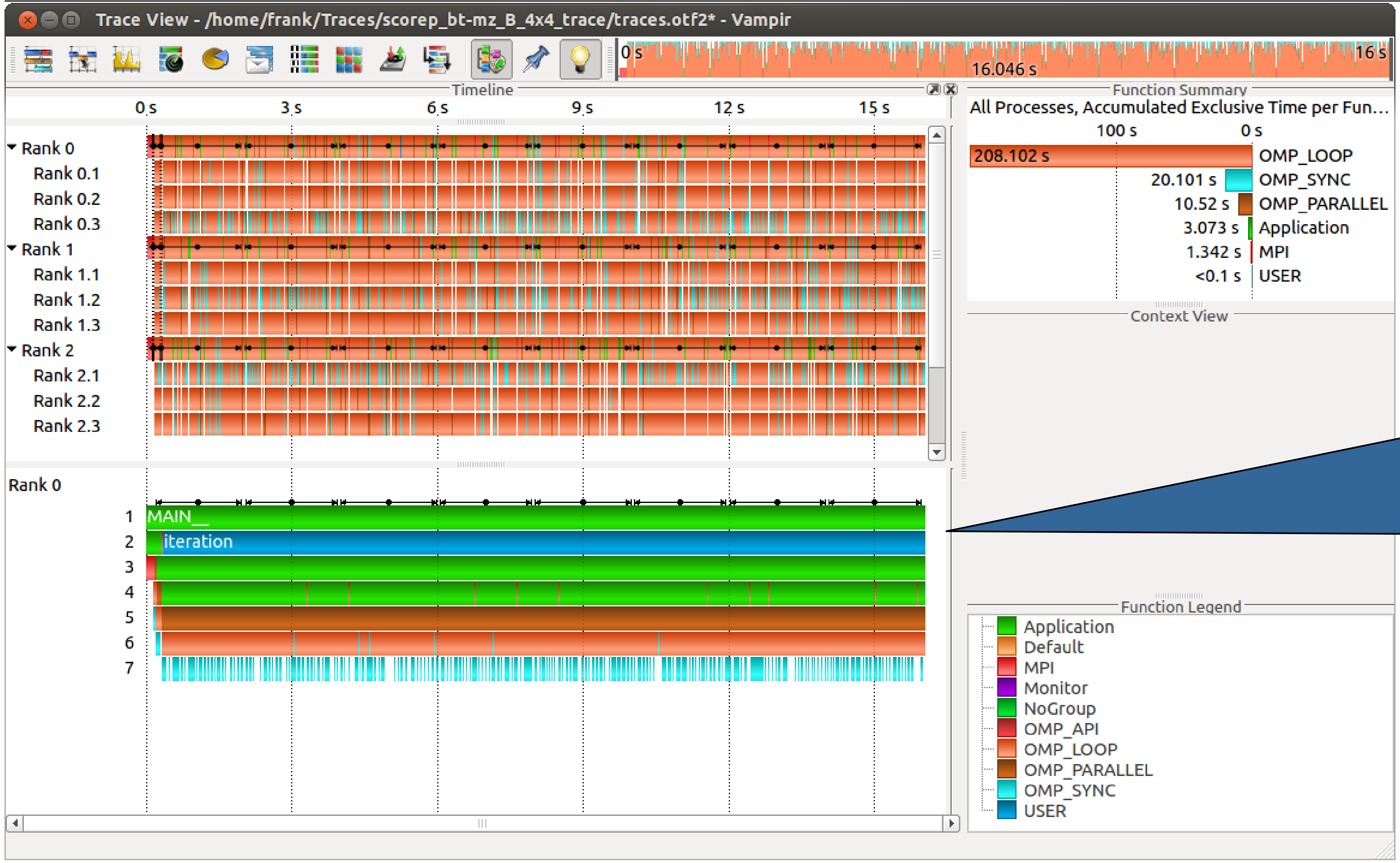
Function Legend

- 🟩 Application
- 🟧 Default
- 🟥 MPI
- 🟪 Monitor
- 🟩 NoGroup
- 🟥 OMP_API
- 🟧 OMP_LOOP
- 🟫 OMP_PARALLEL
- 🟦 OMP_SYNC
- 🟦 USER

Detailed information about functions, communication and synchronization events for collection of processes.

Visualization of the NPB-MZ-MPI / BT trace

Process Timeline


Trace View - /home/frank/Traces/scorep_bt-mz_B_4x4_trace/traces.otf2* - Vampir

Function Summary

All Processes, Accumulated Exclusive Time per Fun...

Time	Function
208.102 s	OMP_LOOP
20.101 s	OMP_SYNC
10.52 s	OMP_PARALLEL
3.073 s	Application
1.342 s	MPI
<0.1 s	USER

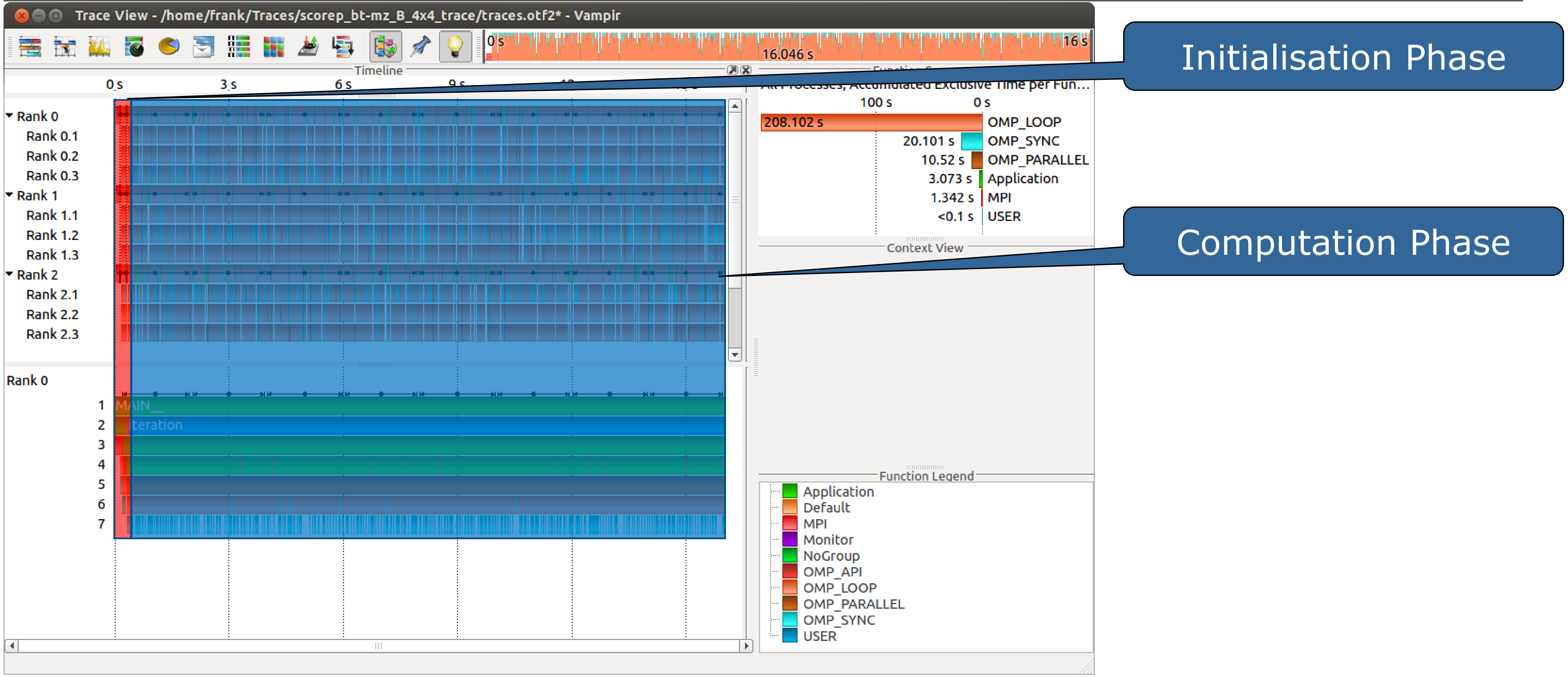
Function Legend

- 🟩 Application
- 🟧 Default
- 🟥 MPI
- 🟪 Monitor
- 🟩 NoGroup
- 🟥 OMP_API
- 🟧 OMP_LOOP
- 🟫 OMP_PARALLEL
- 🟦 OMP_SYNC
- 🟦 USER

Detailed information about different levels of function calls in a stacked bar chart for an individual process.

Visualization of the NPB-MZ-MPI / BT trace

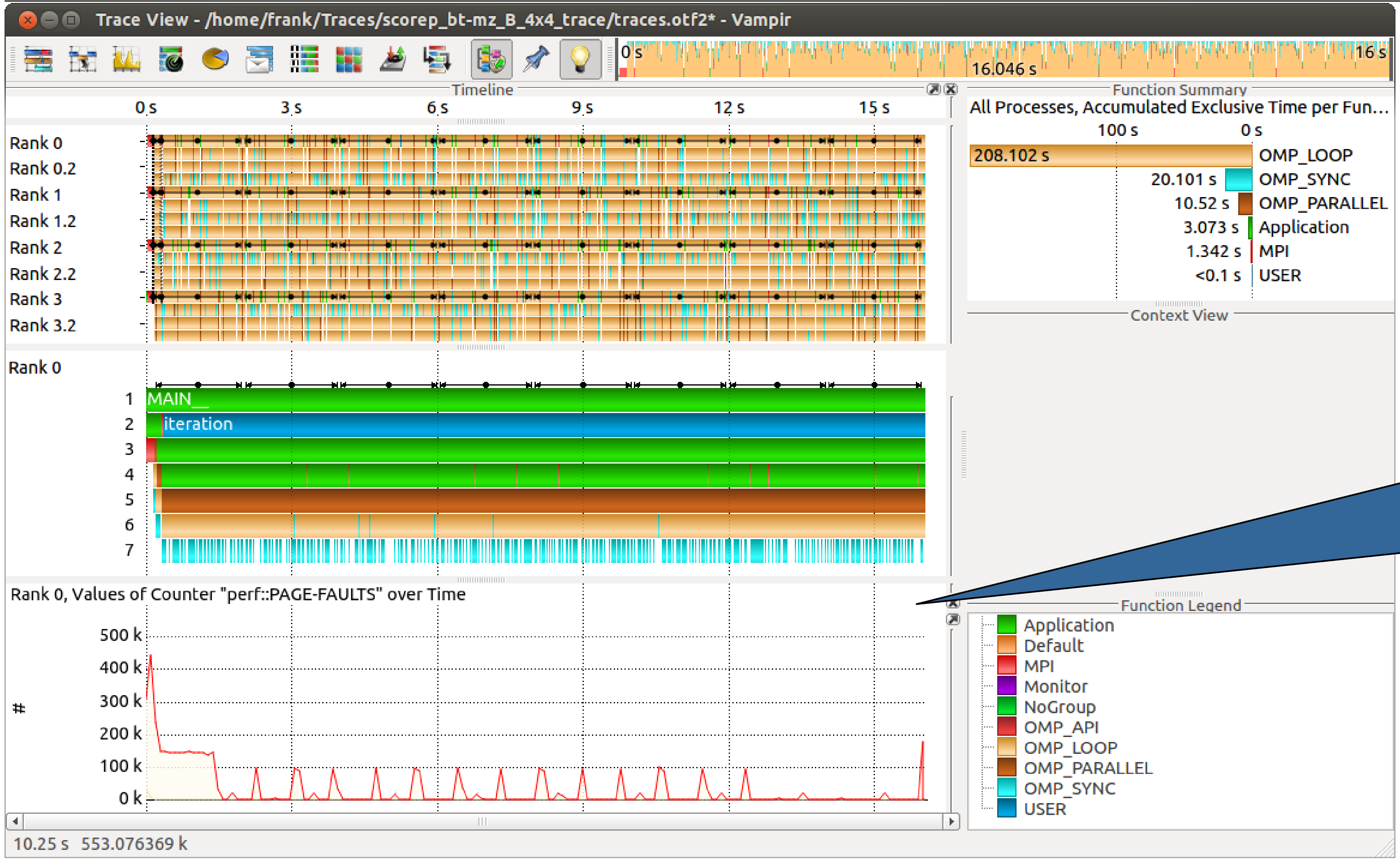
Typical program phases



208.102 s	OMP_LOOP
20.101 s	OMP_SYNC
10.52 s	OMP_PARALLEL
3.073 s	Application
1.342 s	MPI
<0.1 s	USER

- 🟩 Application
- 🟧 Default
- 🟥 MPI
- 🟪 Monitor
- 🟩 NoGroup
- 🟥 OMP_API
- 🟧 OMP_LOOP
- 🟫 OMP_PARALLEL
- 🟦 OMP_SYNC
- 🟦 USER

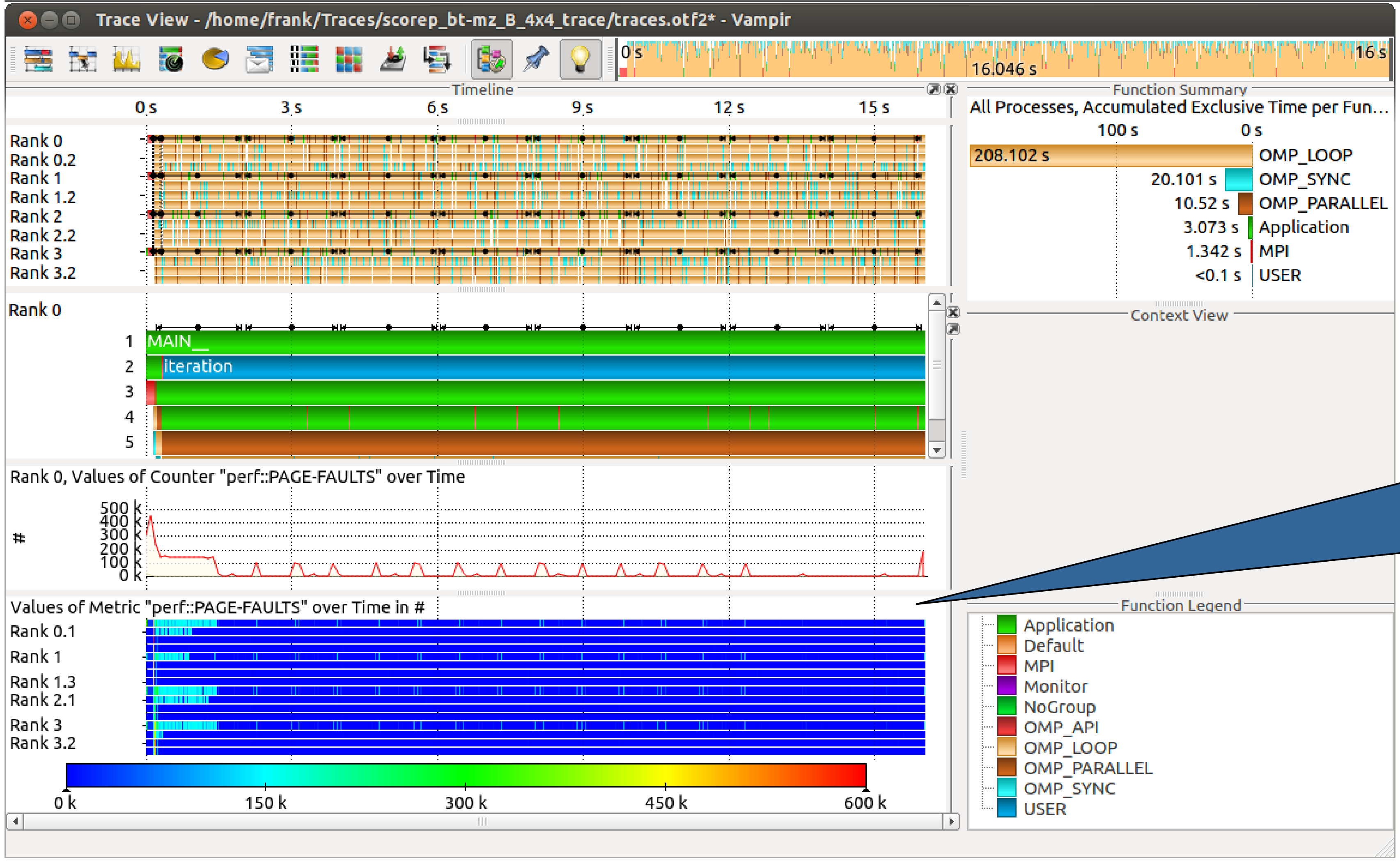
Visualization of the NPB-MZ-MPI / BT trace
Counter Data Timeline



Detailed counter information over time for an individual process.

Visualization of the NPB-MZ-MPI / BT trace

Performance Radar


Trace View - /home/frank/Traces/scorep_bt-mz_B_4x4_trace/traces.otf2* - Vampir

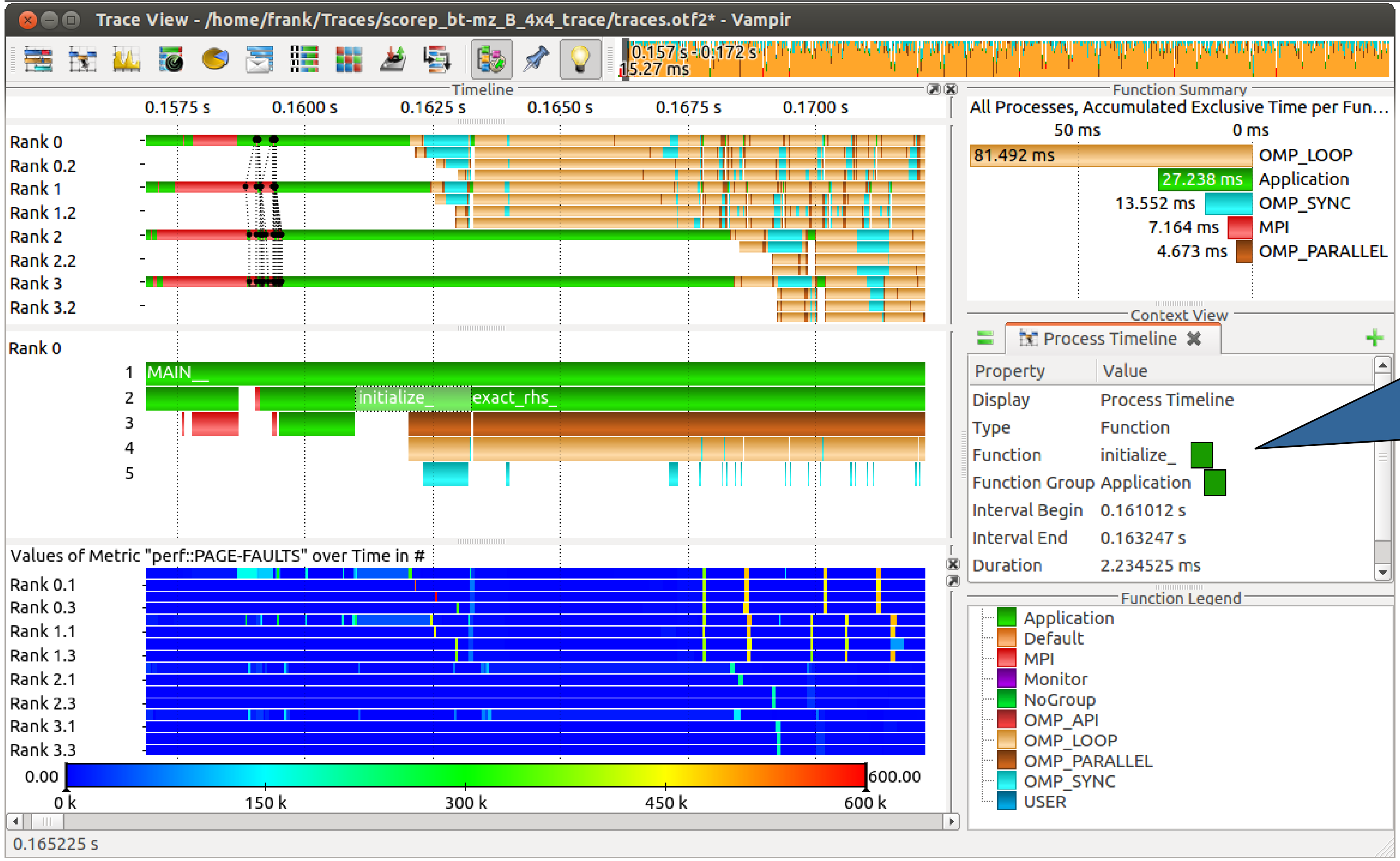
All Processes, Accumulated Exclusive Time per Fun…

Time	Function
208.102 s	OMP_LOOP
20.101 s	OMP_SYNC
10.52 s	OMP_PARALLEL
3.073 s	Application
1.342 s	MPI
<0.1 s	USER

Function Legend: 🟩 Application, 🟧 Default, 🟥 MPI, 🟪 Monitor, 🟩 NoGroup, 🟥 OMP_API, 🟧 OMP_LOOP, 🟫 OMP_PARALLEL, 🟦 OMP_SYNC, 🟦 USER

Detailed counter information over time for a collection of processes.

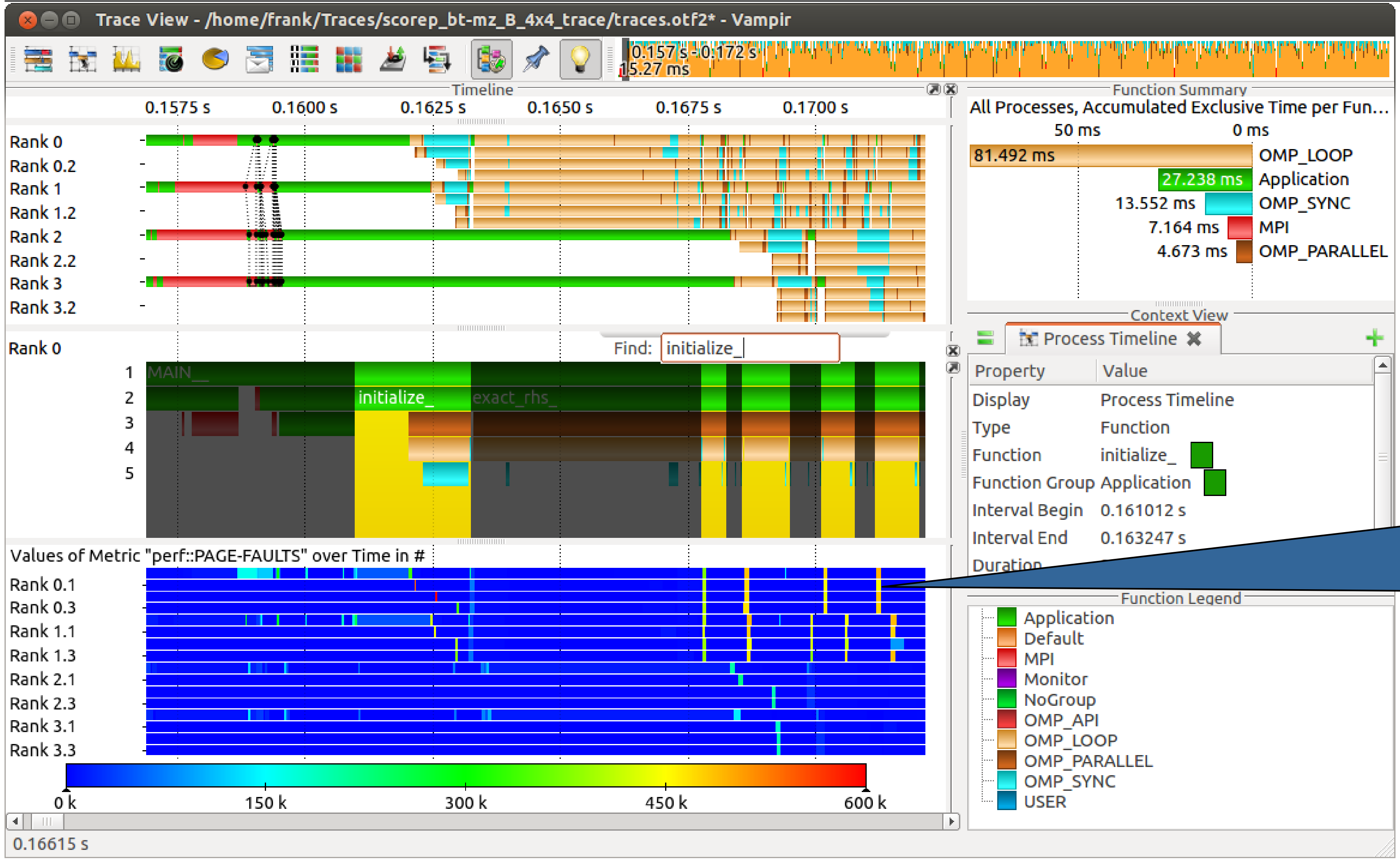
Visualization of the NPB-MZ-MPI / BT trace
Zoom in: Inititialisation Phase



Property	Value
Display	Process Timeline
Type	Function
Function	initialize_ 🟩
Function Group	Application 🟩
Interval Begin	0.161012 s
Interval End	0.163247 s
Duration	2.234525 ms

Context View: Detailed information about function "initialize_".

Visualization of the NPB-MZ-MPI / BT trace
Find Function


Trace View - /home/frank/Traces/scorep_bt-mz_B_4x4_trace/traces.otf2* - Vampir

Function Summary

All Processes, Accumulated Exclusive Time per Fun…

Time	Function
81.492 ms	OMP_LOOP
27.238 ms	Application
13.552 ms	OMP_SYNC
7.164 ms	MPI
4.673 ms	OMP_PARALLEL

Context View — Process Timeline

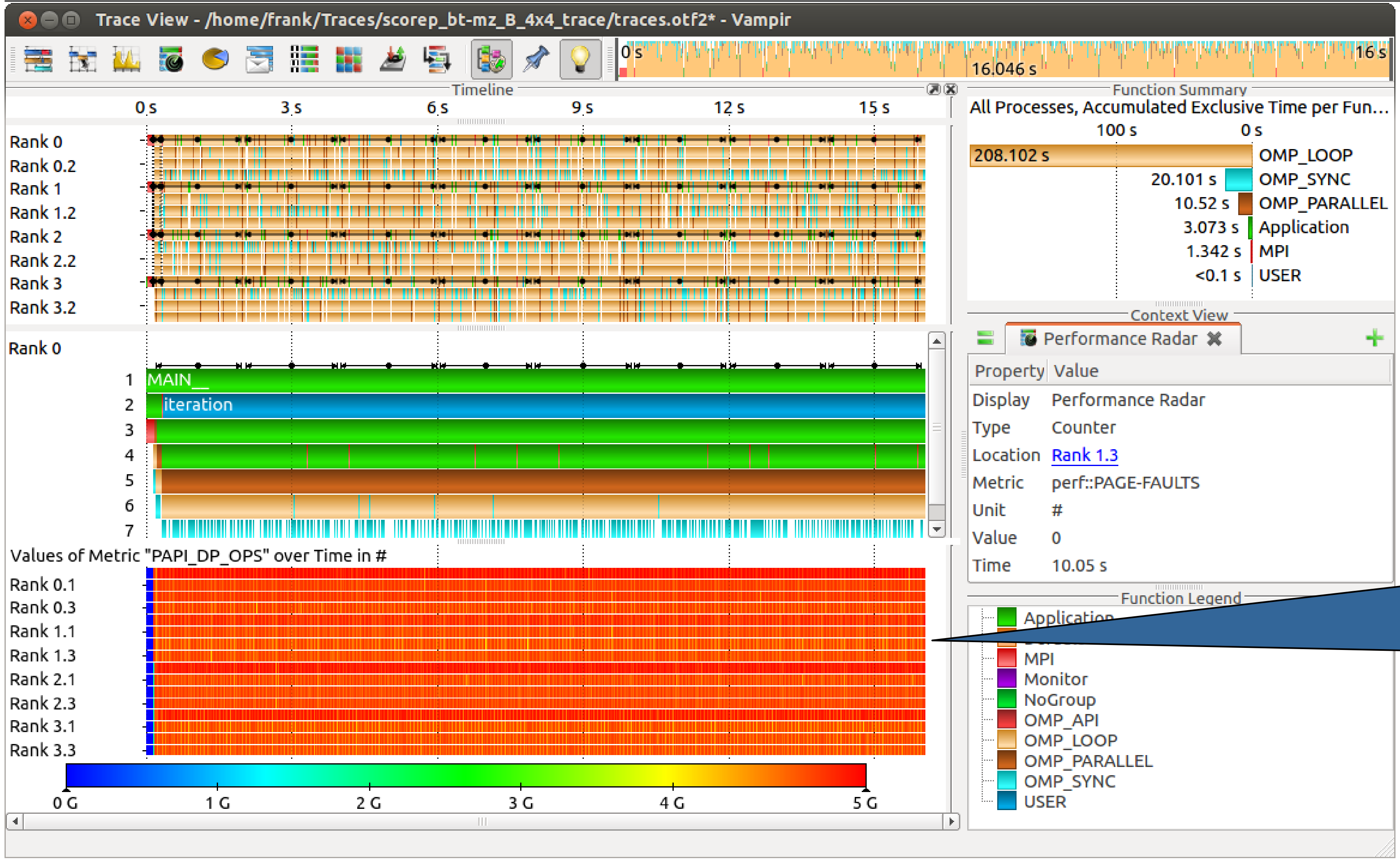
Property	Value
Display	Process Timeline
Type	Function
Function	initialize_ 🟩
Function Group	Application 🟩
Interval Begin	0.161012 s
Interval End	0.163247 s
Duration	

Function Legend

- 🟩 Application
- 🟧 Default
- 🟥 MPI
- 🟪 Monitor
- 🟩 NoGroup
- 🟥 OMP_API
- 🟧 OMP_LOOP
- 🟫 OMP_PARALLEL
- 🟦 OMP_SYNC
- 🟦 USER

Execution of function "initialize_" results in higher page fault rates.

Visualization of the NPB-MZ-MPI / BT trace
Computation Phase



Trace View - /home/frank/Traces/scorep_bt-mz_B_4x4_trace/traces.otf2* - Vampir

Function Summary

All Processes, Accumulated Exclusive Time per Fun...

Time	Function
208.102 s	OMP_LOOP
20.101 s	OMP_SYNC
10.52 s	OMP_PARALLEL
3.073 s	Application
1.342 s	MPI
<0.1 s	USER

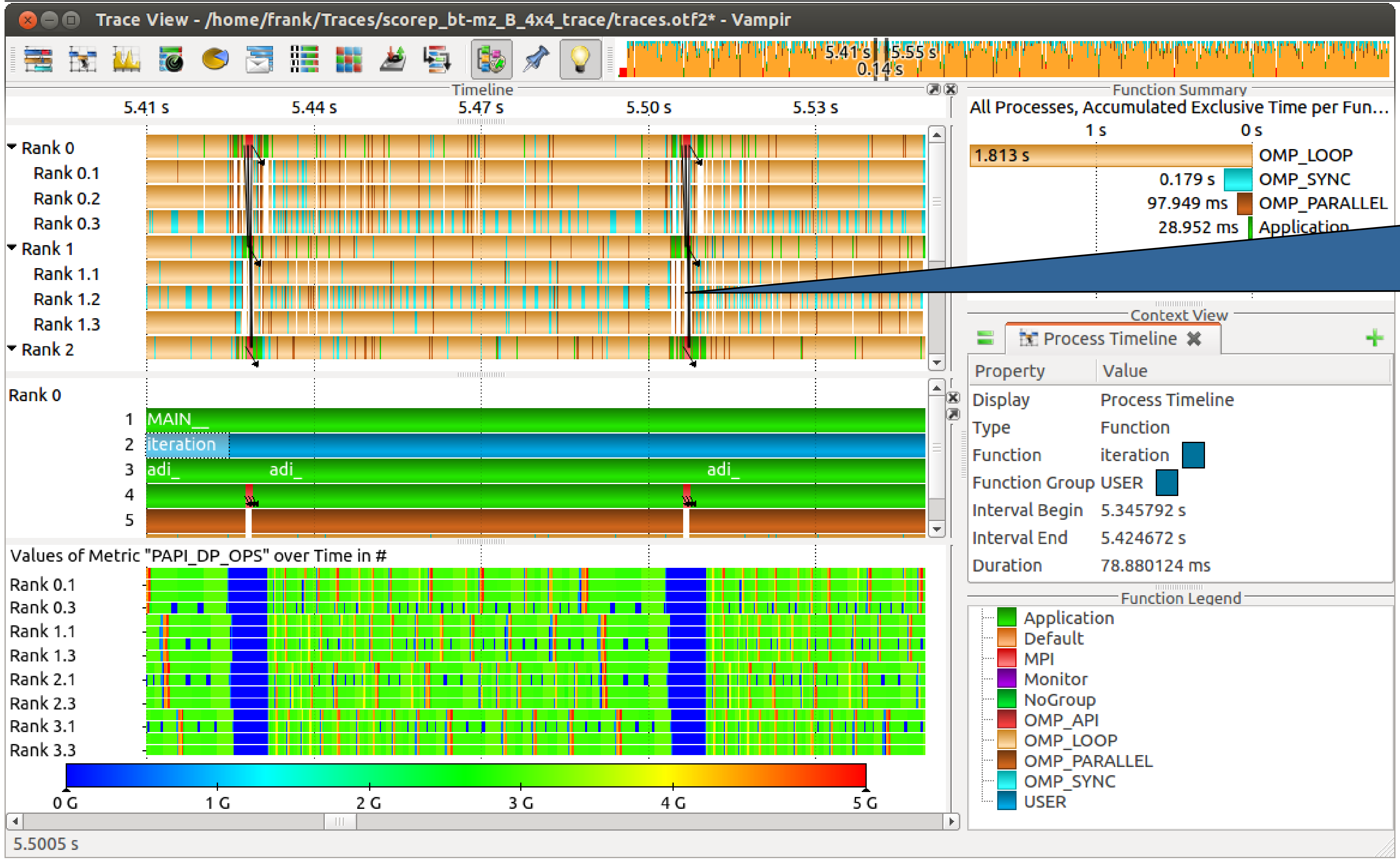
Property	Value
Display	Performance Radar
Type	Counter
Location	Rank 1.3
Metric	perf::PAGE-FAULTS
Unit	#
Value	0
Time	10.05 s

Function Legend
- 🟩 Application
- 🟥 MPI
- 🟪 Monitor
- 🟩 NoGroup
- 🟥 OMP_API
- 🟧 OMP_LOOP
- 🟫 OMP_PARALLEL
- 🟦 OMP_SYNC
- 🟦 USER

Computation phase results in higher floating point operations.

Visualization of the NPB-MZ-MPI / BT trace

Zoom in: Computation Phase


Trace View - /home/frank/Traces/scorep_bt-mz_B_4x4_trace/traces.otf2* - Vampir

Function Summary
All Processes, Accumulated Exclusive Time per Fun...

Time	Function
1.813 s	OMP_LOOP
0.179 s	OMP_SYNC
97.949 ms	OMP_PARALLEL
28.952 ms	Application

Context View — Process Timeline

Property	Value
Display	Process Timeline
Type	Function
Function	iteration
Function Group	USER
Interval Begin	5.345792 s
Interval End	5.424672 s
Duration	78.880124 ms

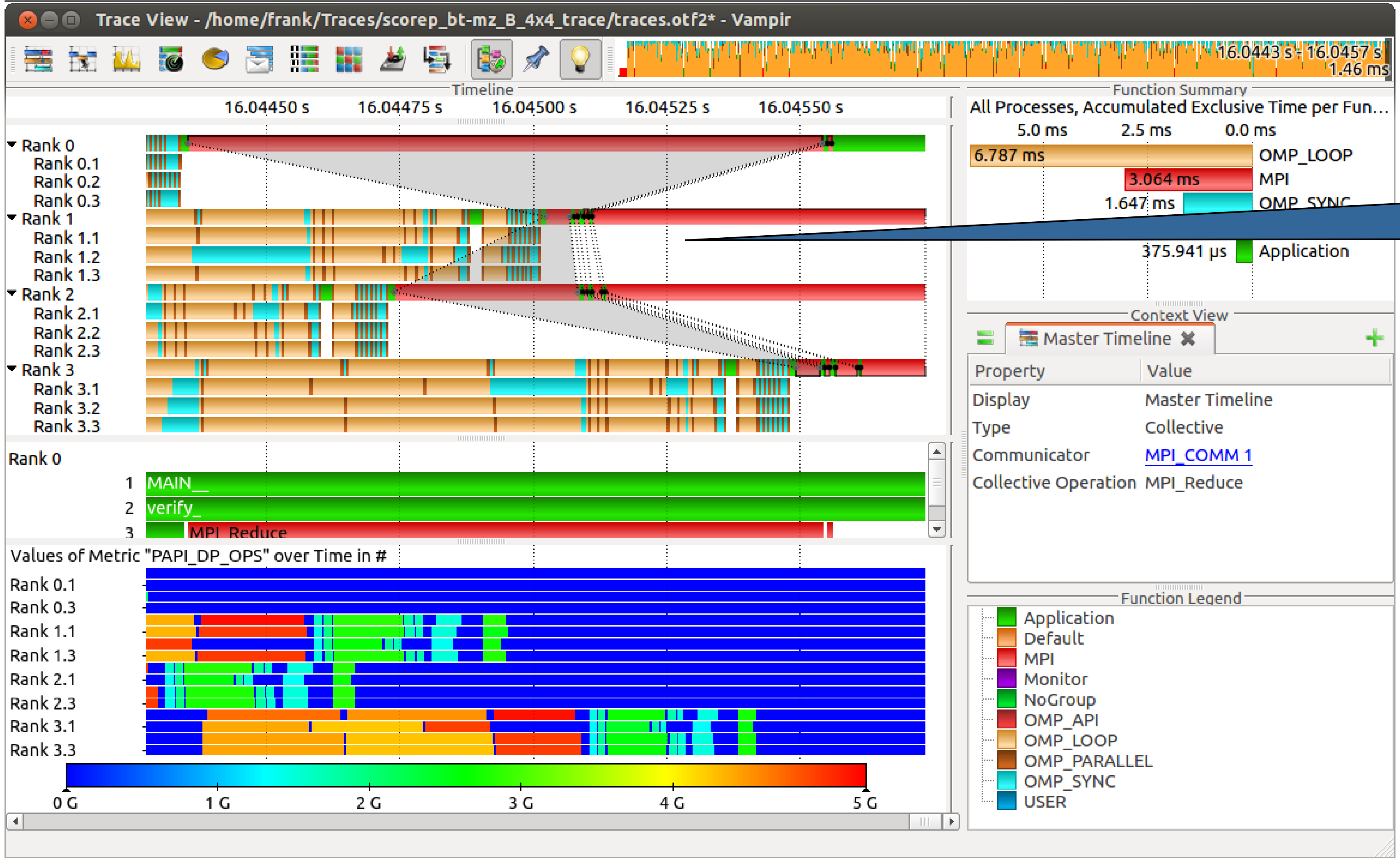
Values of Metric "PAPI_DP_OPS" over Time in #

Function Legend: Application, Default, MPI, Monitor, NoGroup, OMP_API, OMP_LOOP, OMP_PARALLEL, OMP_SYNC, USER

MPI communication results in lower floating point operations.

Visualization of the NPB-MZ-MPI / BT trace

Zoom in: Finalisation Phase



Trace View - /home/frank/Traces/scorep_bt-mz_B_4x4_trace/traces.otf2* - Vampir

16.0443 s - 16.0457 s 1.46 ms

6.787 ms	OMP_LOOP
3.064 ms	MPI
1.647 ms	OMP_SYNC
375.941 µs	Application

Context View — Master Timeline

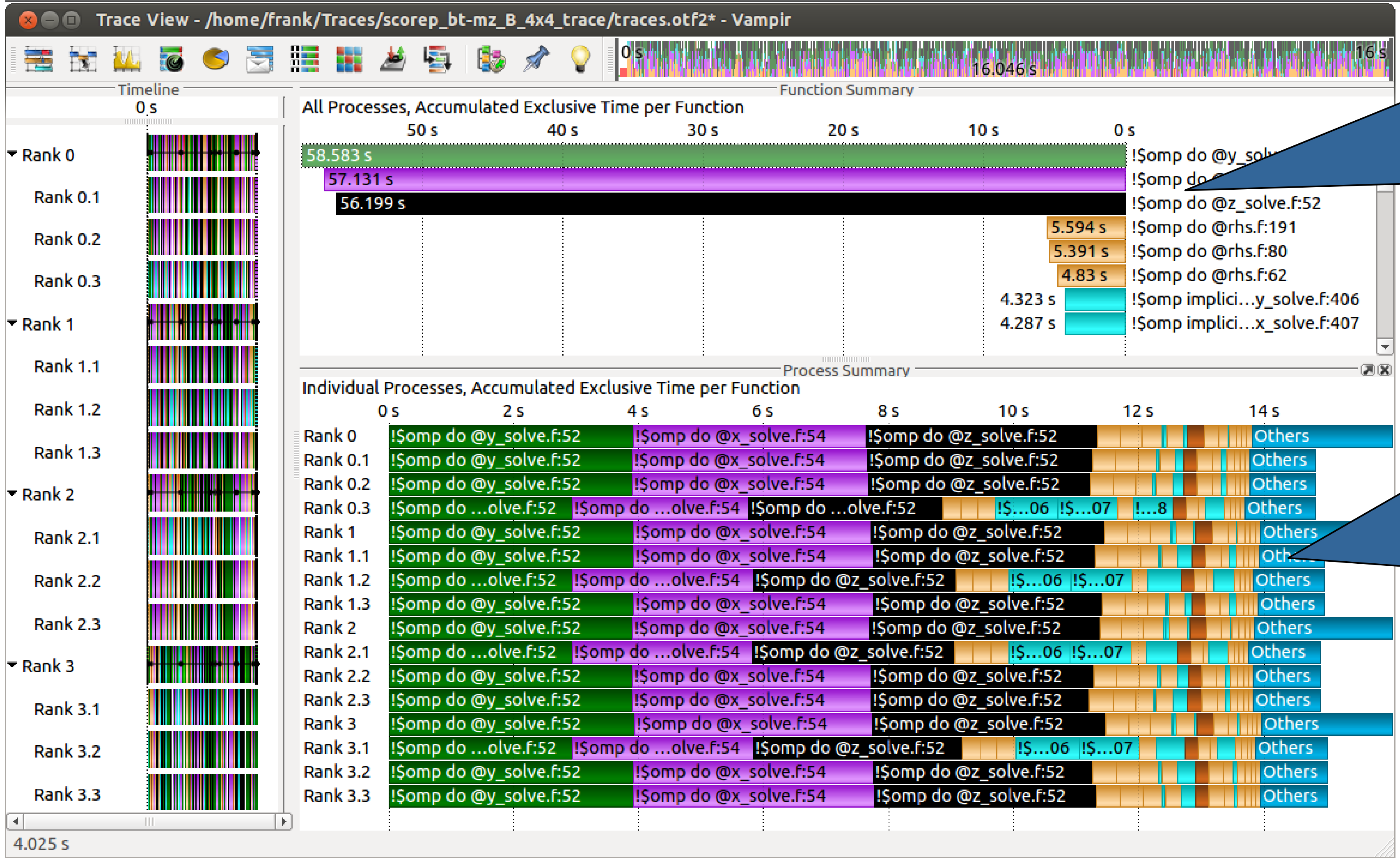
Property	Value
Display	Master Timeline
Type	Collective
Communicator	MPI_COMM 1
Collective Operation	MPI_Reduce

Function Legend

- 🟩 Application
- 🟧 Default
- 🟥 MPI
- 🟪 Monitor
- 🟩 NoGroup
- 🟥 OMP_API
- 🟧 OMP_LOOP
- 🟫 OMP_PARALLEL
- 🟦 OMP_SYNC
- 🟦 USER

"Early reduce" bottleneck.

Visualization of the NPB-MZ-MPI / BT trace
Process Summary


Trace View - /home/frank/Traces/scorep_bt-mz_B_4x4_trace/traces.otf2* - Vampir

Function Summary

All Processes, Accumulated Exclusive Time per Function

```
58.583 s   !$omp do @y_sol…
57.131 s   !$omp do …
56.199 s   !$omp do @z_solve.f:52
5.594 s    !$omp do @rhs.f:191
5.391 s    !$omp do @rhs.f:80
4.83 s     !$omp do @rhs.f:62
4.323 s    !$omp implici…y_solve.f:406
4.287 s    !$omp implici…x_solve.f:407
```

Process Summary

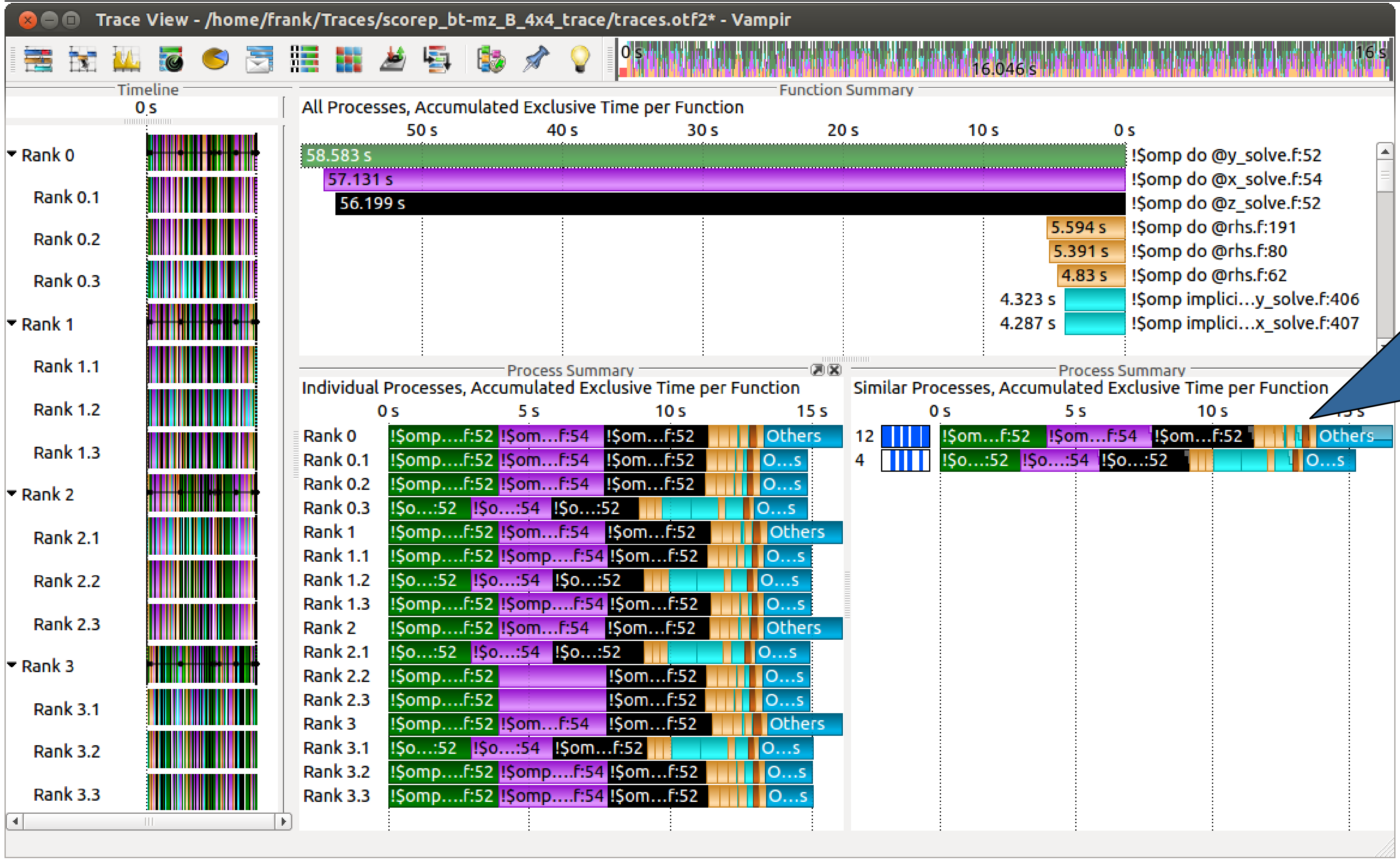
Individual Processes, Accumulated Exclusive Time per Function

```
Rank 0    !$omp do @y_solve.f:52  !$omp do @x_solve.f:54  !$omp do @z_solve.f:52  Others
Rank 0.1  !$omp do @y_solve.f:52  !$omp do @x_solve.f:54  !$omp do @z_solve.f:52  Others
Rank 0.2  !$omp do @y_solve.f:52  !$omp do @x_solve.f:54  !$omp do @z_solve.f:52  Others
Rank 0.3  !$omp do …olve.f:52  !$omp do …olve.f:54  !$omp do …olve.f:52  !$…06  !$…07  !…8  Others
Rank 1    !$omp do @y_solve.f:52  !$omp do @x_solve.f:54  !$omp do @z_solve.f:52  Others
Rank 1.1  !$omp do @y_solve.f:52  !$omp do @x_solve.f:54  !$omp do @z_solve.f:52  Oth…
Rank 1.2  !$omp do …olve.f:52  !$omp do …olve.f:54  !$omp do @z_solve.f:52  !$…06  !$…07  Others
Rank 1.3  !$omp do @y_solve.f:52  !$omp do @x_solve.f:54  !$omp do @z_solve.f:52  Others
Rank 2    !$omp do @y_solve.f:52  !$omp do @x_solve.f:54  !$omp do @z_solve.f:52  Others
Rank 2.1  !$omp do …olve.f:52  !$omp do …olve.f:54  !$omp do @z_solve.f:52  !$…06  !$…07  Others
Rank 2.2  !$omp do @y_solve.f:52  !$omp do @x_solve.f:54  !$omp do @z_solve.f:52  Others
Rank 2.3  !$omp do @y_solve.f:52  !$omp do @x_solve.f:54  !$omp do @z_solve.f:52  Others
Rank 3    !$omp do @y_solve.f:52  !$omp do @x_solve.f:54  !$omp do @z_solve.f:52  Others
Rank 3.1  !$omp do …olve.f:52  !$omp do …olve.f:54  !$omp do @z_solve.f:52  !$…06  !$…07  Others
Rank 3.2  !$omp do @y_solve.f:52  !$omp do @x_solve.f:54  !$omp do @z_solve.f:52  Others
Rank 3.3  !$omp do @y_solve.f:52  !$omp do @x_solve.f:54  !$omp do @z_solve.f:52  Others
```

Function Summary: Overview of the accumulated information across all functions and for a collection of processes.

Process Summary: Overview of the accumulated information across all functions and for every process independently.

Visualization of the NPB-MZ-MPI / BT trace
Process Summary


Trace View - /home/frank/Traces/scorep_bt-mz_B_4x4_trace/traces.otf2* - Vampir

Function Summary — All Processes, Accumulated Exclusive Time per Function

Time	Function
58.583 s	!$omp do @y_solve.f:52
57.131 s	!$omp do @x_solve.f:54
56.199 s	!$omp do @z_solve.f:52
5.594 s	!$omp do @rhs.f:191
5.391 s	!$omp do @rhs.f:80
4.83 s	!$omp do @rhs.f:62
4.323 s	!$omp implici…y_solve.f:406
4.287 s	!$omp implici…x_solve.f:407

Process Summary — Individual Processes, Accumulated Exclusive Time per Function

Process Summary — Similar Processes, Accumulated Exclusive Time per Function

Count	Functions
12	!$om…f:52, !$om…f:54, !$om…f:52, Others
4	!$o…:52, !$o…:54, !$o…:52, O…s

Find groups of similar processes and threads by using summarized function information.

Summary and Conclusion

Summary

- Vampir & VampirServer
  - Interactive trace visualization and analysis
  - Intuitive browsing and zooming
  - Scalable to large trace data sizes (20 TiByte)
  - Scalable to high parallelism (200,000 processes)
- Vampir for Linux, Windows, and Mac OS X

INDIANA UNIVERSITY

SPONSORED BY THE Federal Ministry of Education and Research

SILC

OAK RIDGE National Laboratory

LMAC

READEX Runtime Exploitation of Application Dynamism for Energy-efficient eXascale computing

eeClust Energy-Efficient Cluster Computing

nextgenio

Score-E

VAMPIR

European Commission | Horizon 2020 European Union funding for Research & Innovation

Microsoft®

ITEA 2 INFORMATION TECHNOLOGY FOR EUROPEAN ADVANCEMENT

SEVENTH FRAMEWORK PROGRAMME

//H4H

//ParMA

http://www.vampir.eu

vampirsupport@zih.tu-dresden.de

Score-P – A Joint Performance Measurement Run-Time Infrastructure for Periscope, Scalasca, TAU, and Vampir

VI-HPS Team

Scalable performance measurement infrastructure for parallel codes

Score-P: Specialized Measurements and Analyses

Mastering build systems

- Hooking up the Score-P instrumenter `scorep` into complex build environments like *Autotools* or *CMake* was always challenging
- Score-P provides new convenience wrapper scripts to simplify this (since Score-P 2.0)
- *Autotools* and *CMake* need the used compiler already in the *configure step,* but instrumentation should not happen in this step, only in the *build step*

```
% SCOREP_WRAPPER=off \
> cmake .. \
> -DCMAKE_C_COMPILER=scorep-icc \
> -DCMAKE_CXX_COMPILER=scorep-icpc
```

Disable instrumentation in the *configure step*

Specify the wrapper scripts as the compiler to use

- Allows to pass addition options to the Score-P instrumenter and the compiler via environment variables without modifying the *Makefiles*
- Run `scorep-wrapper --help` for a detailed description and the available wrapper scripts of the Score-P installation

Mastering C++ applications

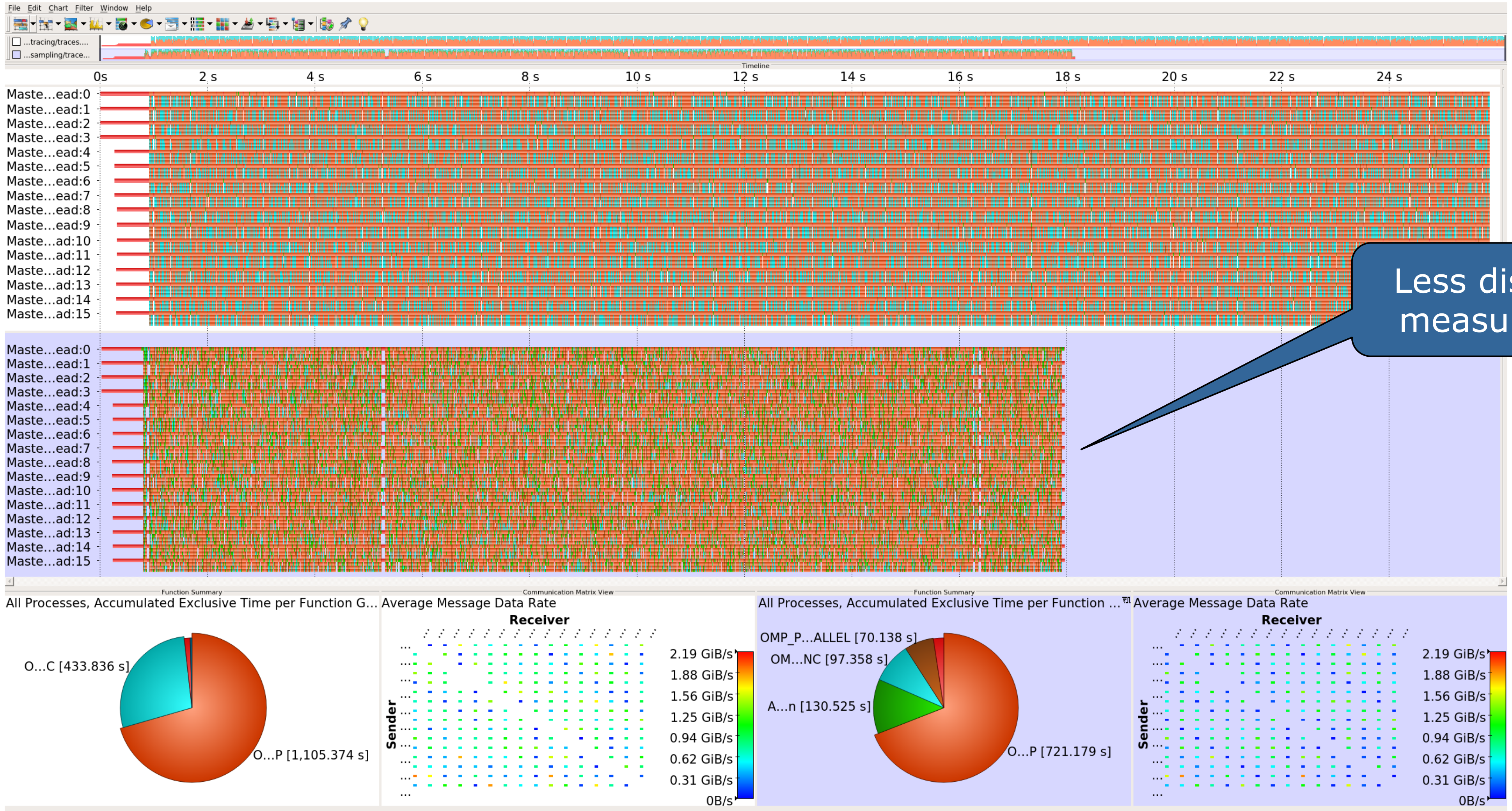
- Automatic compiler instrumentation greatly disturbs C++ applications because of frequent/short function calls => Use sampling instead
- Novel combination of sampling events and instrumentation of MPI, OpenMP, …
  - Sampling replaces compiler instrumentation (instrument with `--nocompiler` to further reduce overhead) => Filtering not needed anymore
  - Instrumentation is used to get accurate times for parallel activities to still be able to identifies patterns of inefficiencies
- Supports profile and trace generation

```
% export SCOREP_ENABLE_UNWINDING=true
% # use the default sampling frequency
% #export SCOREP_SAMPLING_EVENTS=perf_cycles@2000000

% OMP_NUM_THREADS=4 mpiexec -np 4 ./bt-mz_W.4
```

- Set new configuration variable to enable sampling

- Available since Score-P 2.0, only x86-64 supported currently

Mastering C++ applications



Wrapping calls to 3rd party libraries

- Enables users to install library wrappers for any C/C++ library
- Intercept calls to a library API
  - no need to either build the library with Score-P or add manual instrumentation to the application using the library
  - no need to access the source code of the library, header and library files suffice
- Score-P needs to be executed with `--libwrap=…`
- Execute `scorep-libwrap-init` for directions:

Step 1: Initialize the working directory
Step 2: Add library headers
Step 3: Create a simple example application
Step 4: Further configure the build parameters
Step 5: Build the wrapper
Step 6: Verify the wrapper
Step 7: Install the wrapper
Step 8: Verify the installed wrapper

Only once

Often

Step 9: Use the wrapper

Wrapping calls to 3rd party libraries

- Generate your own library wrappers by telling `scorep-libwrap-init` how you would compile and link an application, e.g. using FFTW

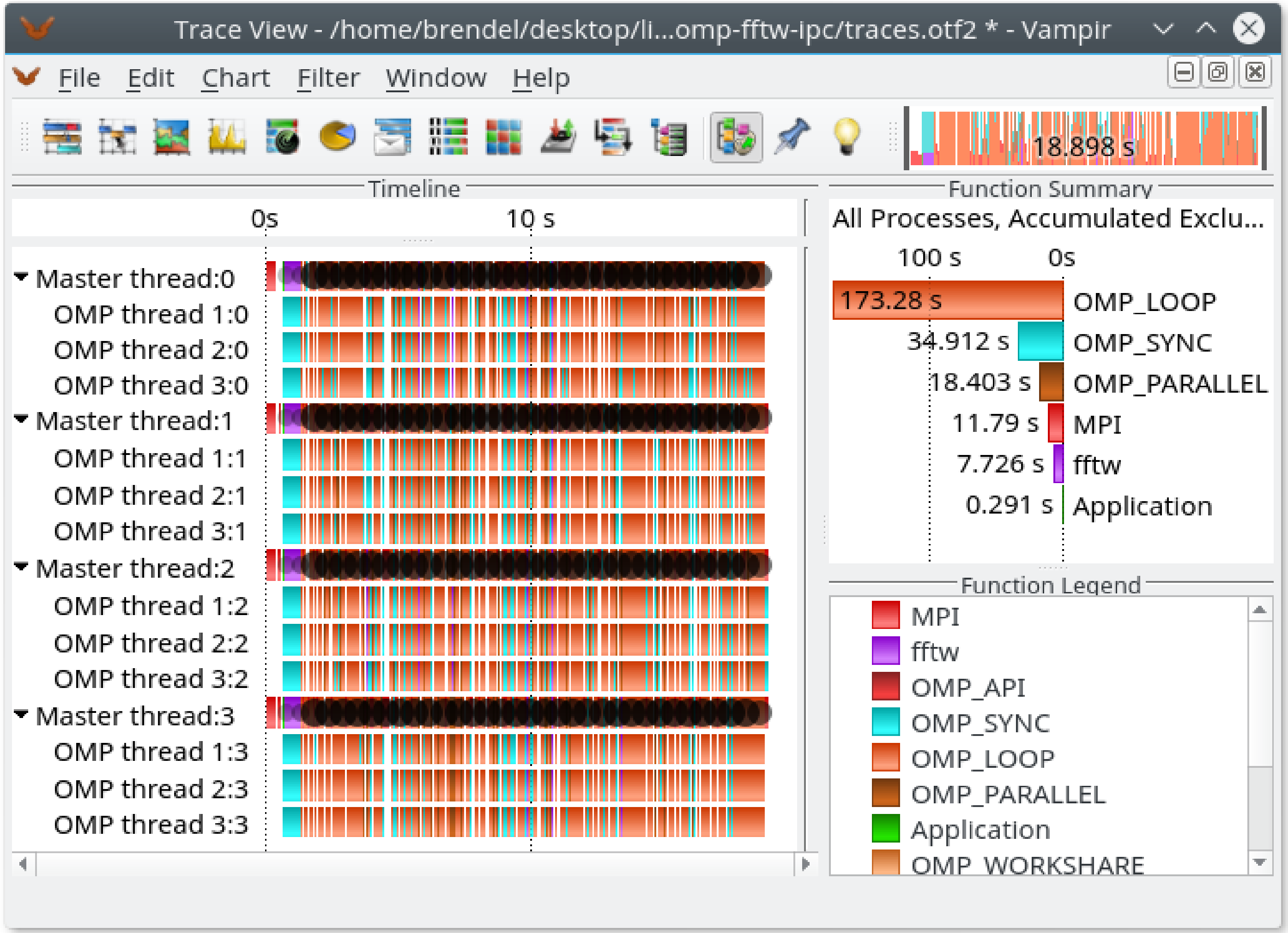
```
% scorep-libwrap-init         \
>   --name=fftw               \
>   --prefix=$PREFIX          \
>   -x c                      \
>   --cppflags="-O3 -DNDEBUG -openmp -I$FFTW_INC" \
>   --ldflags="-L$FFTW_LIB"   \
>   --libs="-lfftw3f -lfftw3" \
>   working_directory
```

- Generate and build wrapper

```
% cd working_directory
% ls                   # (Check README.md for instructions)
% make                 # Generate and build wrapper
% make check           # See if header analysis matches symbols
% make install         #
% make installcheck    # More checks: Linking etc.
```

Wrapping calls to 3rd party libraries

- MPI + OpenMP
- Calls to FFTW library



Mastering application memory usage

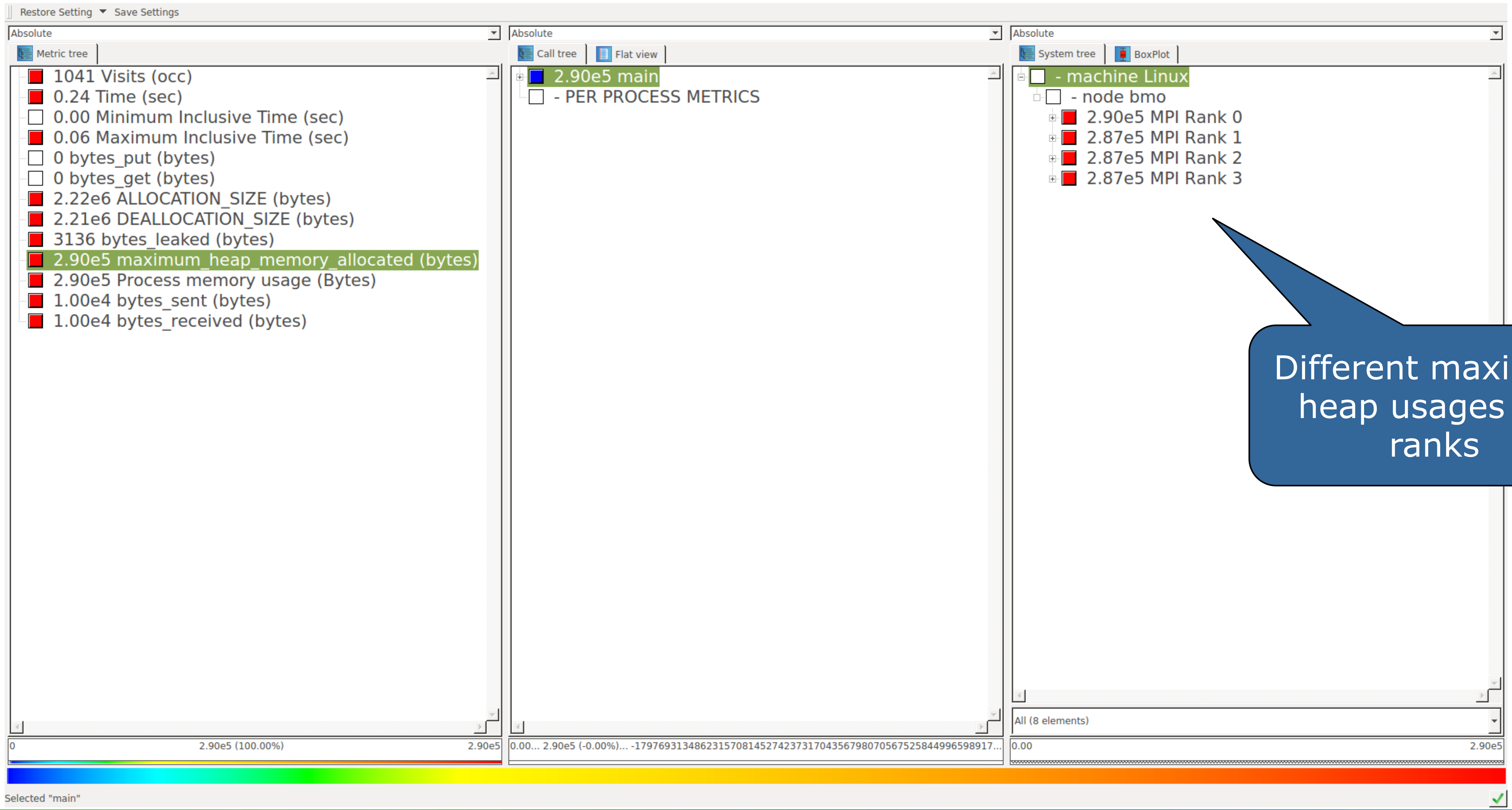
- Determine the maximum heap usage per process
- Find high frequent small allocation patterns
- Find memory leaks
- Support for:
  - C, C++, MPI, and SHMEM (Fortran only for GNU Compilers)
  - Profile and trace generation (profile recommended)
    - Memory leaks are recorded only in the profile
    - Resulting traces are not supported by Scalasca yet

```
% export SCOREP_MEMORY_RECORDING=true
% export SCOREP_MPI_MEMORY_RECORDING=true

% OMP_NUM_THREADS=4 mpiexec –np 4 ./bt-mz_W.4
```

- Set new configuration variable to enable memory recording

- Available since Score-P 2.0

Mastering application memory usage



- 🟥 1041 Visits (occ)
- 🟥 0.24 Time (sec)
- ☐ 0.00 Minimum Inclusive Time (sec)
- 🟥 0.06 Maximum Inclusive Time (sec)
- ☐ 0 bytes_put (bytes)
- ☐ 0 bytes_get (bytes)
- 🟥 2.22e6 ALLOCATION_SIZE (bytes)
- 🟥 2.21e6 DEALLOCATION_SIZE (bytes)
- 🟥 3136 bytes_leaked (bytes)
- 🟥 2.90e5 maximum_heap_memory_allocated (bytes)
- 🟥 2.90e5 Process memory usage (Bytes)
- 🟥 1.00e4 bytes_sent (bytes)
- 🟥 1.00e4 bytes_received (bytes)

- 🟦 2.90e5 main
- ☐ - PER PROCESS METRICS

- ☐ - machine Linux
  - ☐ - node bmo
    - 🟥 2.90e5 MPI Rank 0
    - 🟥 2.87e5 MPI Rank 1
    - 🟥 2.87e5 MPI Rank 2
    - 🟥 2.87e5 MPI Rank 3

All (8 elements)

0 2.90e5 (100.00%) 2.90e5

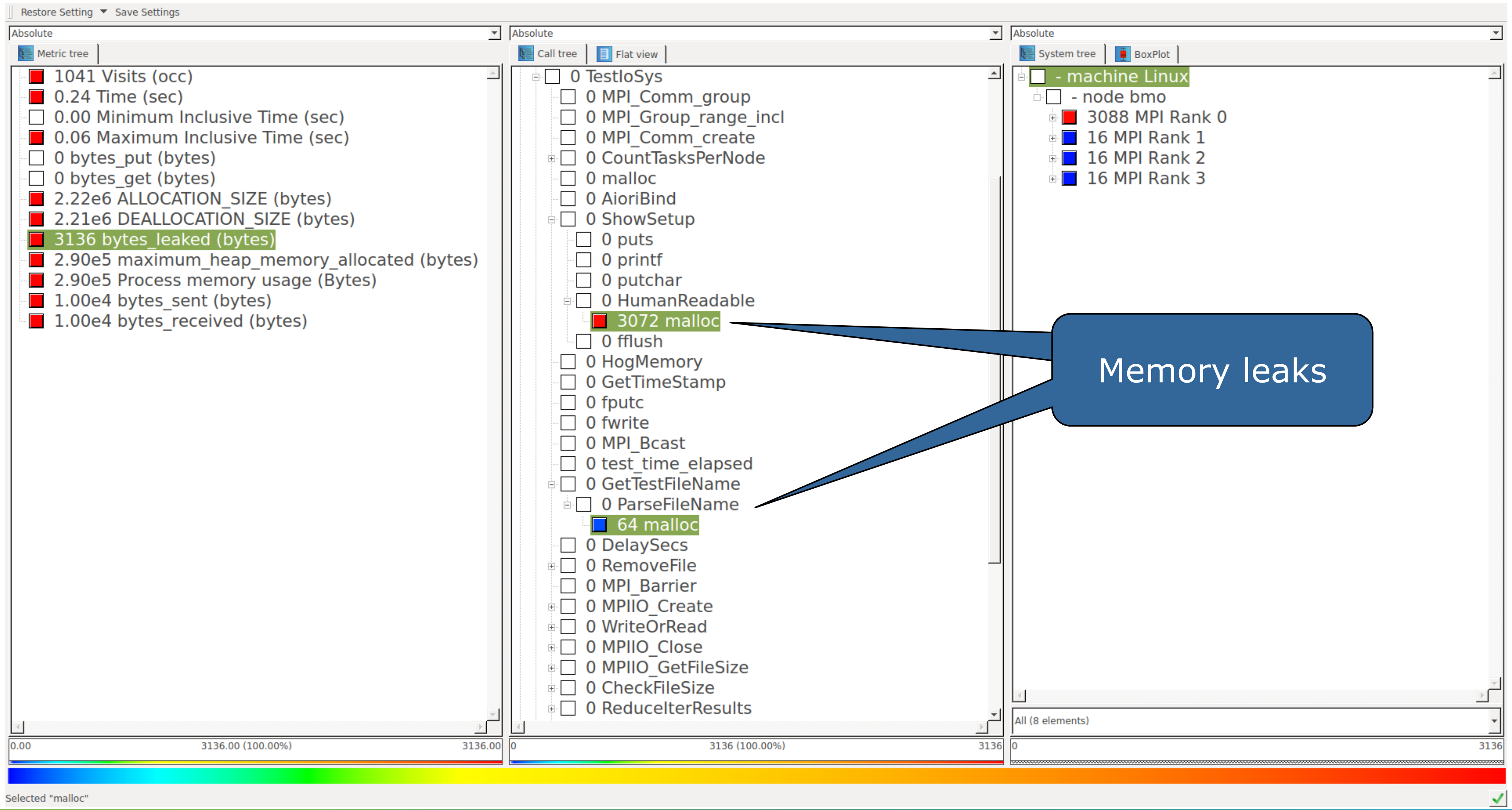
0.00... 2.90e5 (-0.00%)... -17976931348623157081452742373170435679807056752584496598917...

0.00 2.90e5

Selected "main"

Different maximum heap usages per ranks

Mastering application memory usage



```
🟥 1041 Visits (occ)
🟥 0.24 Time (sec)
☐ 0.00 Minimum Inclusive Time (sec)
🟥 0.06 Maximum Inclusive Time (sec)
☐ 0 bytes_put (bytes)
☐ 0 bytes_get (bytes)
🟥 2.22e6 ALLOCATION_SIZE (bytes)
🟥 2.21e6 DEALLOCATION_SIZE (bytes)
🟥 3136 bytes_leaked (bytes)
🟥 2.90e5 maximum_heap_memory_allocated (bytes)
🟥 2.90e5 Process memory usage (Bytes)
🟥 1.00e4 bytes_sent (bytes)
🟥 1.00e4 bytes_received (bytes)
```

```
☐ 0 TestIoSys
  ☐ 0 MPI_Comm_group
  ☐ 0 MPI_Group_range_incl
  ☐ 0 MPI_Comm_create
  ☐ 0 CountTasksPerNode
  ☐ 0 malloc
  ☐ 0 AioriBind
  ☐ 0 ShowSetup
    ☐ 0 puts
    ☐ 0 printf
    ☐ 0 putchar
    ☐ 0 HumanReadable
      🟥 3072 malloc
    ☐ 0 fflush
  ☐ 0 HogMemory
  ☐ 0 GetTimeStamp
  ☐ 0 fputc
  ☐ 0 fwrite
  ☐ 0 MPI_Bcast
  ☐ 0 test_time_elapsed
  ☐ 0 GetTestFileName
    ☐ 0 ParseFileName
      🟦 64 malloc
  ☐ 0 DelaySecs
  ☐ 0 RemoveFile
  ☐ 0 MPI_Barrier
  ☐ 0 MPIIO_Create
  ☐ 0 WriteOrRead
  ☐ 0 MPIIO_Close
  ☐ 0 MPIIO_GetFileSize
  ☐ 0 CheckFileSize
  ☐ 0 ReduceIterResults
```

```
☐ - machine Linux
  ☐ - node bmo
    🟥 3088 MPI Rank 0
    🟦 16 MPI Rank 1
    🟦 16 MPI Rank 2
    🟦 16 MPI Rank 3
```

All (8 elements)

0.00 3136.00 (100.00%) 3136.00 | 0 3136 (100.00%) 3136 | 0 3136

Selected "malloc"

Mastering heterogeneous applications

- Record CUDA applications and device activities

```
% export SCOREP_CUDA_ENABLE=gpu,kernel,idle
```

- Record OpenCL applications and device activities

```
% export SCOREP_OPENCL_ENABLE=api,kernel
```

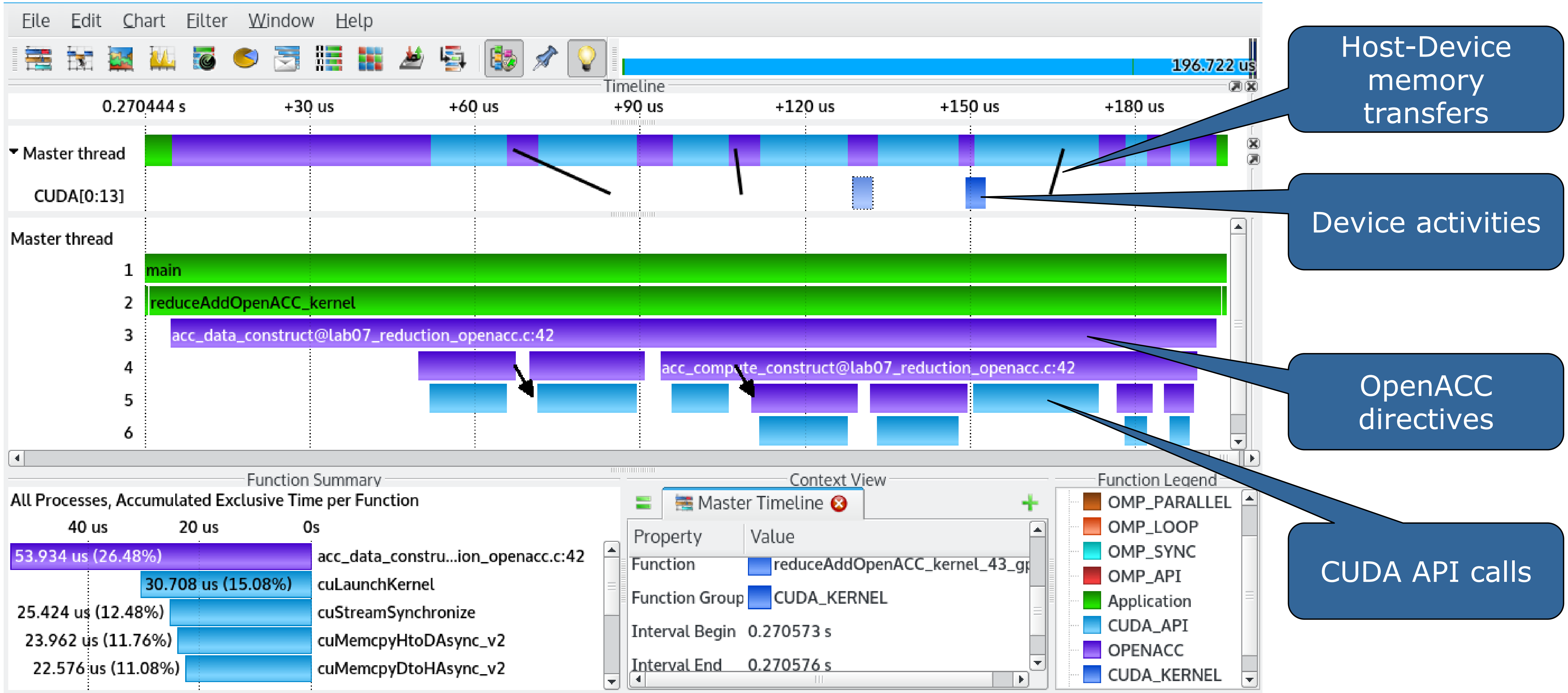
- Record OpenACC applications

```
% export SCOREP_OPENACC_ENABLE=yes
```

  - Can be combined with CUDA if it is a NVIDIA device

```
% export SCOREP_CUDA_ENABLE=kernel
```

Mastering heterogeneous applications



All Processes, Accumulated Exclusive Time per Function

40 us 20 us 0s

Time	Function
53.934 us (26.48%)	acc_data_constru...ion_openacc.c:42
30.708 us (15.08%)	cuLaunchKernel
25.424 us (12.48%)	cuStreamSynchronize
23.962 us (11.76%)	cuMemcpyHtoDAsync_v2
22.576 us (11.08%)	cuMemcpyDtoHAsync_v2

Property	Value
Function	reduceAddOpenACC_kernel_43_gp
Function Group	CUDA_KERNEL
Interval Begin	0.270573 s
Interval End	0.270576 s

Function Legend:
- OMP_PARALLEL
- OMP_LOOP
- OMP_SYNC
- OMP_API
- Application
- CUDA_API
- OPENACC
- CUDA_KERNEL

Enriching measurements with performance counters

- Record metrics from PAPI:

```
% export SCOREP_METRIC_PAPI=PAPI_TOT_CYC
% export SCOREP_METRIC_PAPI_PER_PROCESS=PAPI_L3_TCM
```

  - Use PAPI tools to get available metrics and valid combinations:

```
% papi_avail
% papi_native_avail
```

- Record metrics from Linux perf:

```
% export SCOREP_METRIC_PERF=cpu-cycles
% export SCOREP_METRIC_PERF_PER_PROCESS=LLC-load-misses
```

  - Use the `perf` tool to get available metrics and valid combinations:

```
% perf list
```

Only the master thread records the metric (assuming all threads of the process access the same L3 cache)

- Write your own metric plugin
  - Repository of available plugins: https://github.com/score-p

Score-P user instrumentation API

- No replacement for automatic compiler instrumentation

- Can be used to further subdivide functions
  - E.g., multiple loops inside a function
- Can be used to partition application into coarse grain phases
  - E.g., initialization, solver, & finalization

- Enabled with `--user` flag to Score-P instrumenter

- Available for Fortran / C / C++

Score-P user instrumentation API (Fortran)

```
#include "scorep/SCOREP_User.inc"

subroutine foo(…)
  ! Declarations
  SCOREP_USER_REGION_DEFINE( solve )

  ! Some code…
  SCOREP_USER_REGION_BEGIN( solve, “<solver>", \
                            SCOREP_USER_REGION_TYPE_LOOP )
  do i=1,100
    [...]
  end do
  SCOREP_USER_REGION_END( solve )
  ! Some more code…
end subroutine
```

- Requires processing by the C preprocessor
  - For most compilers, this can be automatically achieved by having an uppercase file extension, e.g., `main.F` or `main.F90`

Score-P user instrumentation API (C/C++)

```
#include "scorep/SCOREP_User.h"

void foo()
{
  /* Declarations */
  SCOREP_USER_REGION_DEFINE( solve )

  /* Some code… */
  SCOREP_USER_REGION_BEGIN( solve, “<solver>",
                            SCOREP_USER_REGION_TYPE_LOOP )
  for (i = 0; i < 100; i++)
  {
    [...]
  }
  SCOREP_USER_REGION_END( solve )
  /* Some more code… */
}
```

Score-P user instrumentation API (C++)

```
#include "scorep/SCOREP_User.h"

void foo()
{
  // Declarations

  // Some code…
  {
    SCOREP_USER_REGION( “<solver>",
                        SCOREP_USER_REGION_TYPE_LOOP )
    for (i = 0; i < 100; i++)
    {
      [...]
    }
  }
  // Some more code…
}
```

Score-P measurement control API

- Can be used to temporarily disable measurement for certain intervals
  - Annotation macros ignored by default
  - Enabled with `--user` flag

```
#include “scorep/SCOREP_User.inc”

subroutine foo(…)
  ! Some code…
  SCOREP_RECORDING_OFF()
  ! Loop will not be measured
  do i=1,100
    [...]
  end do
  SCOREP_RECORDING_ON()
  ! Some more code…
end subroutine
```

Fortran (requires C preprocessor)

```
#include “scorep/SCOREP_User.h”

void foo(…) {
  /* Some code… */
  SCOREP_RECORDING_OFF()
  /* Loop will not be measured */
  for (i = 0; i < 100; i++) {
    [...]
  }
  SCOREP_RECORDING_ON()
  /* Some more code… */
}
```

C / C++

Score-P: Conclusion and Outlook

Project management

- Ensure a single official release version at all times which will always work with the tools

- Allow experimental versions for new features or research

- Commitment to joint long-term cooperation
  - Development based on meritocratic governance model
  - Open for contributions and new partners

Future features

- Scalability to maximum available CPU core count

- Support for emerging architectures and new programming models

- Features currently worked on:
  - Hardware and MPI topologies
  - MPI-3 RMA support
  - OpenMP tool support (OMPT)
  - I/O recording
  - Basic support of measurements without re-compiling/-linking
  - Java recording
  - Persistent memory recording (e.g., PMEM, NVRAM, …)

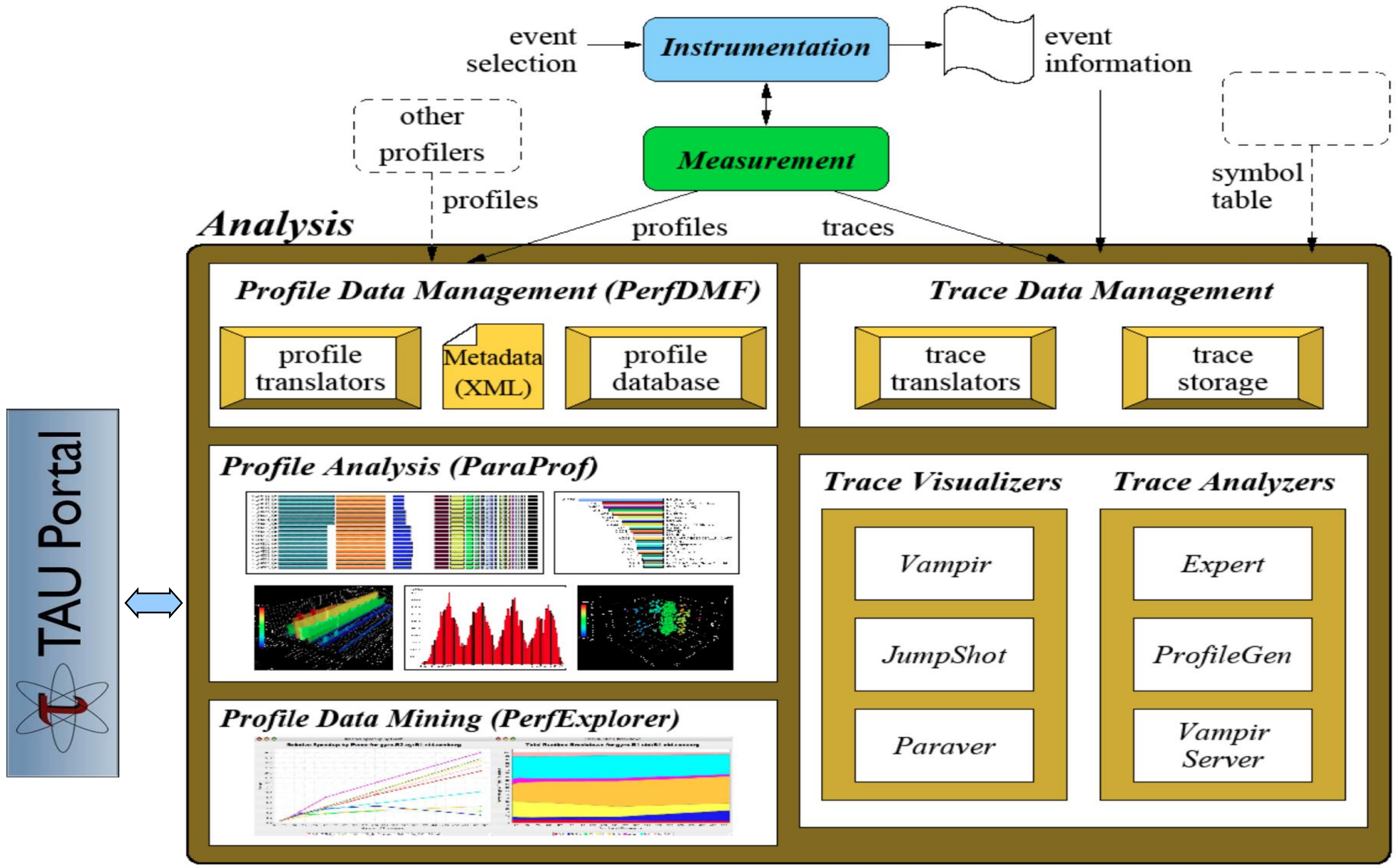
Further information

- Community instrumentation & measurement infrastructure
  - Instrumentation (various methods) and sampling
  - Basic and advanced profile generation
  - Event trace recording
  - Online access to profiling data
- Available under 3-clause BSD open-source license
- Documentation & Sources:
  - http://www.score-p.org
- User guide also part of installation:
  - `<prefix>/share/doc/scorep/{pdf,html}/`
- Support and feedback: support@score-p.org
- Subscribe to news@score-p.org, to be up to date

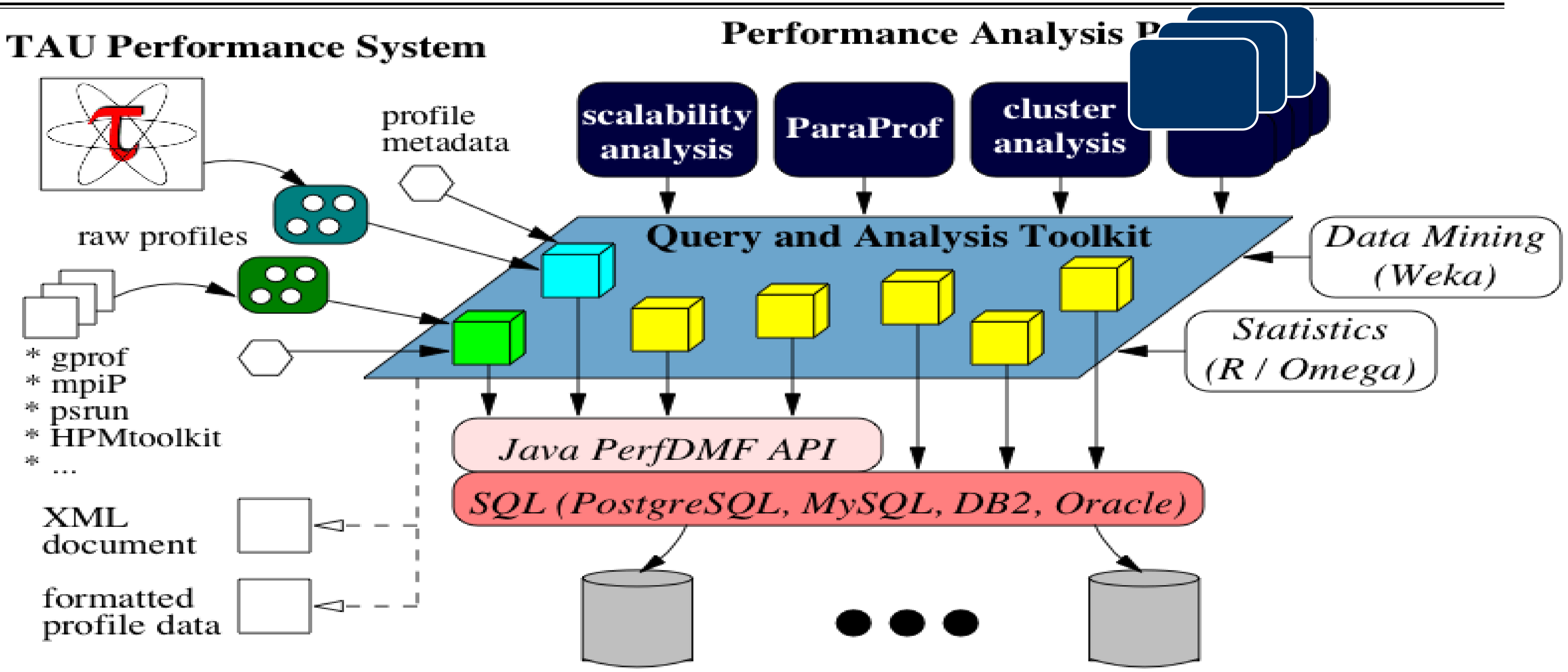
Performance data management with TAU PerfExplorer

Sameer Shende
sameer@cs.uoregon.edu
University of Oregon
http://tau.uoregon.edu

TAU Analysis



TAUdb: Performance Data Management Framework



Using TAUdb

- Configure TAUdb (Done by each user)

  % taudb_configure --create-default

  - Choose derby, PostgreSQL, MySQL, Oracle or DB2
  - Hostname
  - Username
  - Password
  - Say yes to downloading required drivers (we are not allowed to distribute these)
  - Stores parameters in your ~/.ParaProf/taudb.cfg file
- Configure PerfExplorer (Done by each user)

  % perfexplorer_configure
- Execute PerfExplorer

  % perfexplorer

Using PerfExplorer

```
% wget http://tau.uoregon.edu/data.tgz  (Contains CUBE profiles from Score-P)
% tar zxf data.tgz; cd data; cat README; cd tau; ./upload.sh; perfexplorer
Or manually:
% taudb_configure --create-default
(Chooses derby, blank user/passwd, yes to save passwd, defaults)
% perfexplorer_configure
(Yes to load schema, defaults)
% paraprof
(load each trial: DB -> Add Trial -> Type (Paraprof Packed Profile) -> OK) OR use taudb_loadtrial –
 a “app” –x “experiment” –n “name” file.ppk
Then,
% tar zxf $TAU/data.tgz; cd data/tau;
% taudb_loadtrial –a BT_MZ –x “Class_B” bt-mz_B.*.ppk
% perfexplorer
(Select experiment, Menu: Charts -> Speedup)
```

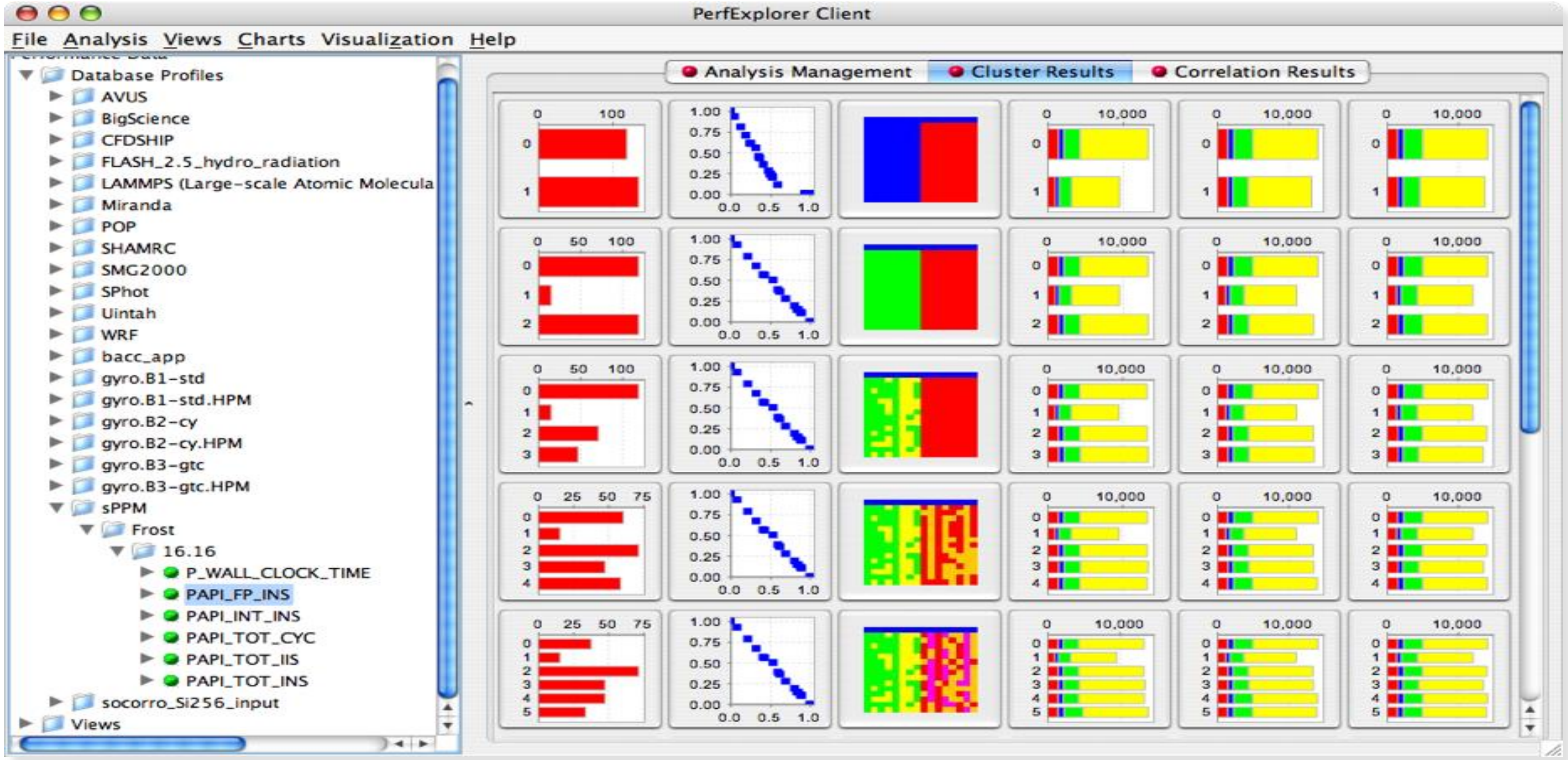
Performance Data Mining (PerfExplorer)

- Performance knowledge discovery framework
  - Data mining analysis applied to parallel performance data
    - comparative, clustering, correlation, dimension reduction, …
  - Use the existing TAU infrastructure
    - TAU performance profiles, taudb
  - Client-server based system architecture
- Technology integration
  - Java API and toolkit for portability
  - taudb
  - R-project/Omegahat, Octave/Matlab statistical analysis
  - WEKA data mining package
  - JFreeChart for visualization, vector output (EPS, SVG)

PerfExplorer: Using Cluster Analysis

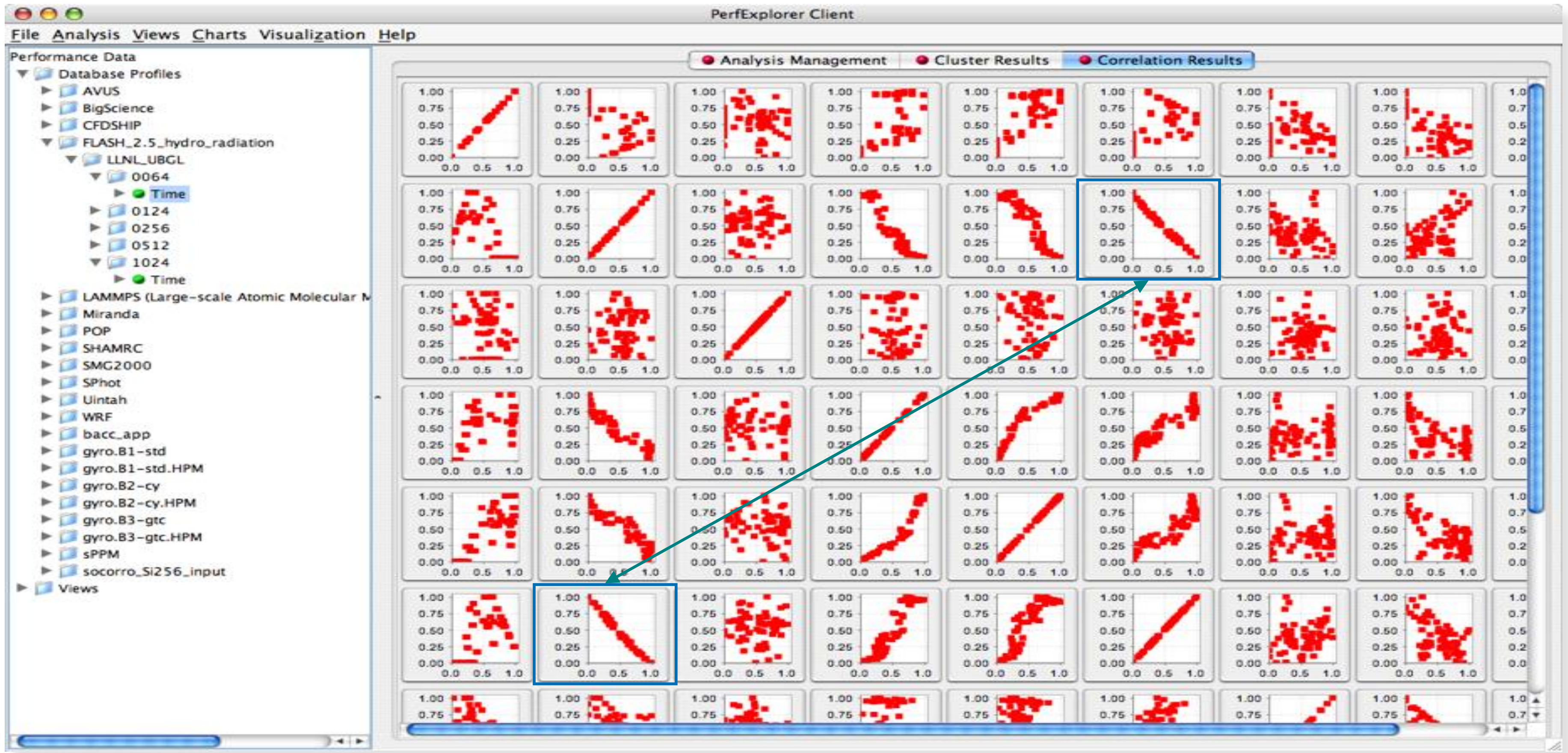
- Performance data represented as vectors - each dimension is the cumulative time for an event
- *k*-means: *k* random centers are selected and instances are grouped with the "closest" (Euclidean) center
- New centers are calculated and the process repeated until stabilization or max iterations
- Dimension reduction necessary for meaningful results
- Virtual topology, summaries constructed

PerfExplorer - Cluster Analysis (sPPM)



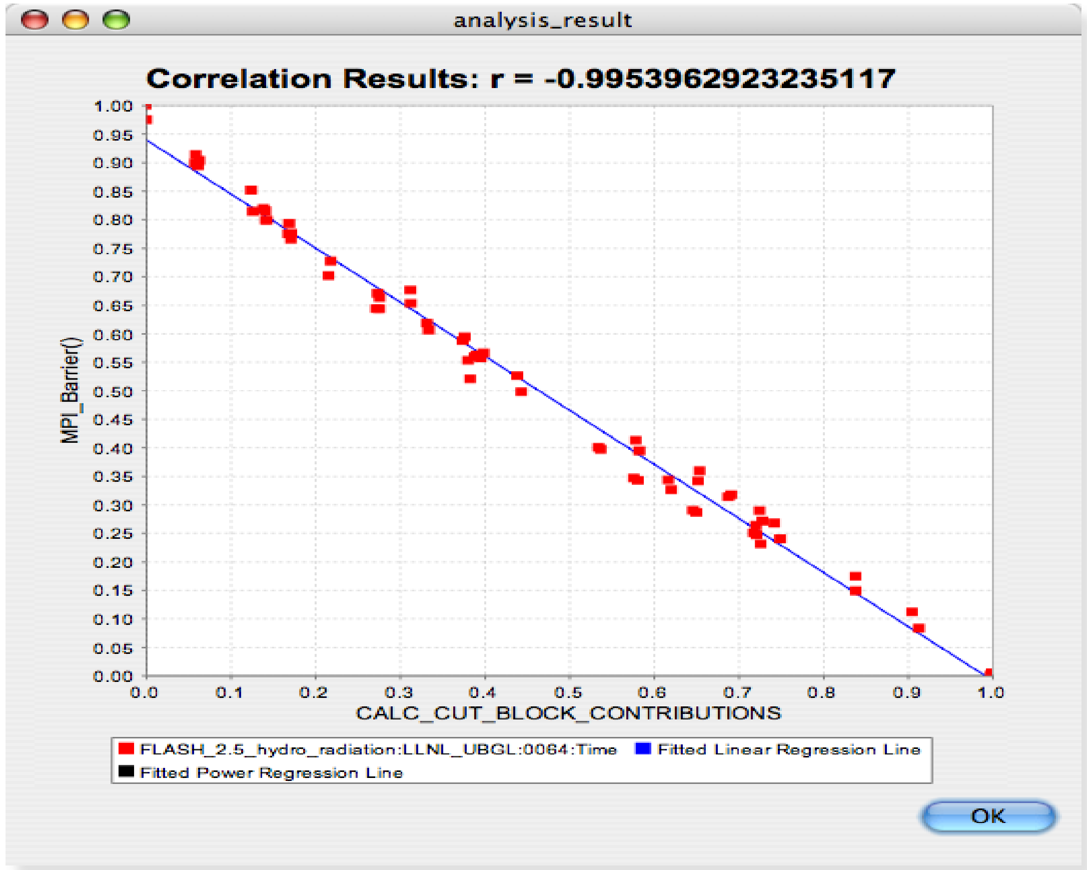
PerfExplorer - Correlation Analysis (Flash)

- Describes strength and direction of a linear relationship between two variables (events) in the data



PerfExplorer - Correlation Analysis (Flash)

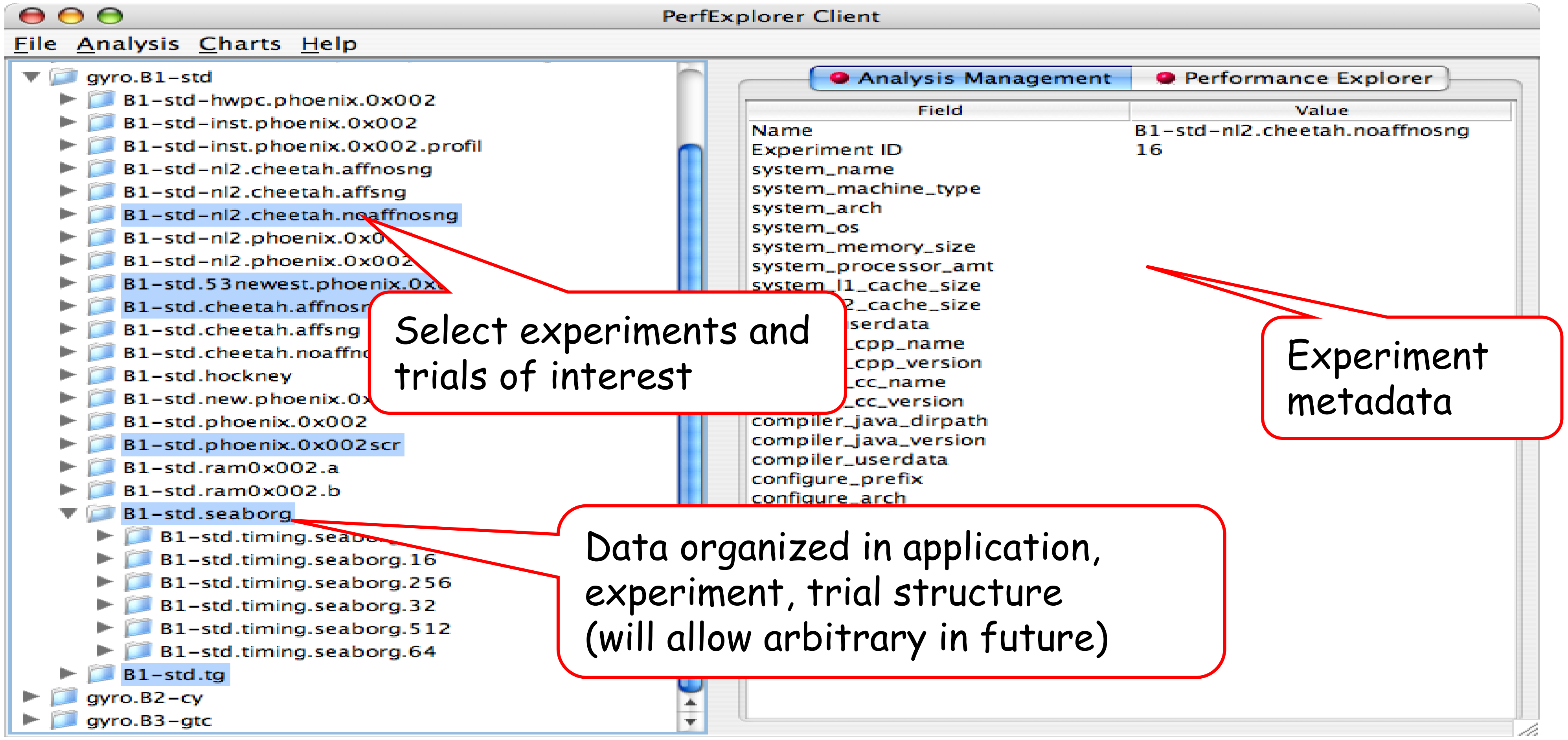
- -0.995 indicates strong, negative relationship
- As CALC_CUT_BLOCK_CONTRIBUTIONS() increases in execution time, MPI_Barrier() decreases



PerfExplorer - Comparative Analysis

- Relative speedup, efficiency
  - total runtime, by event, one event, by phase
- Breakdown of total runtime
- Group fraction of total runtime
- Correlating events to total runtime
- Timesteps per second

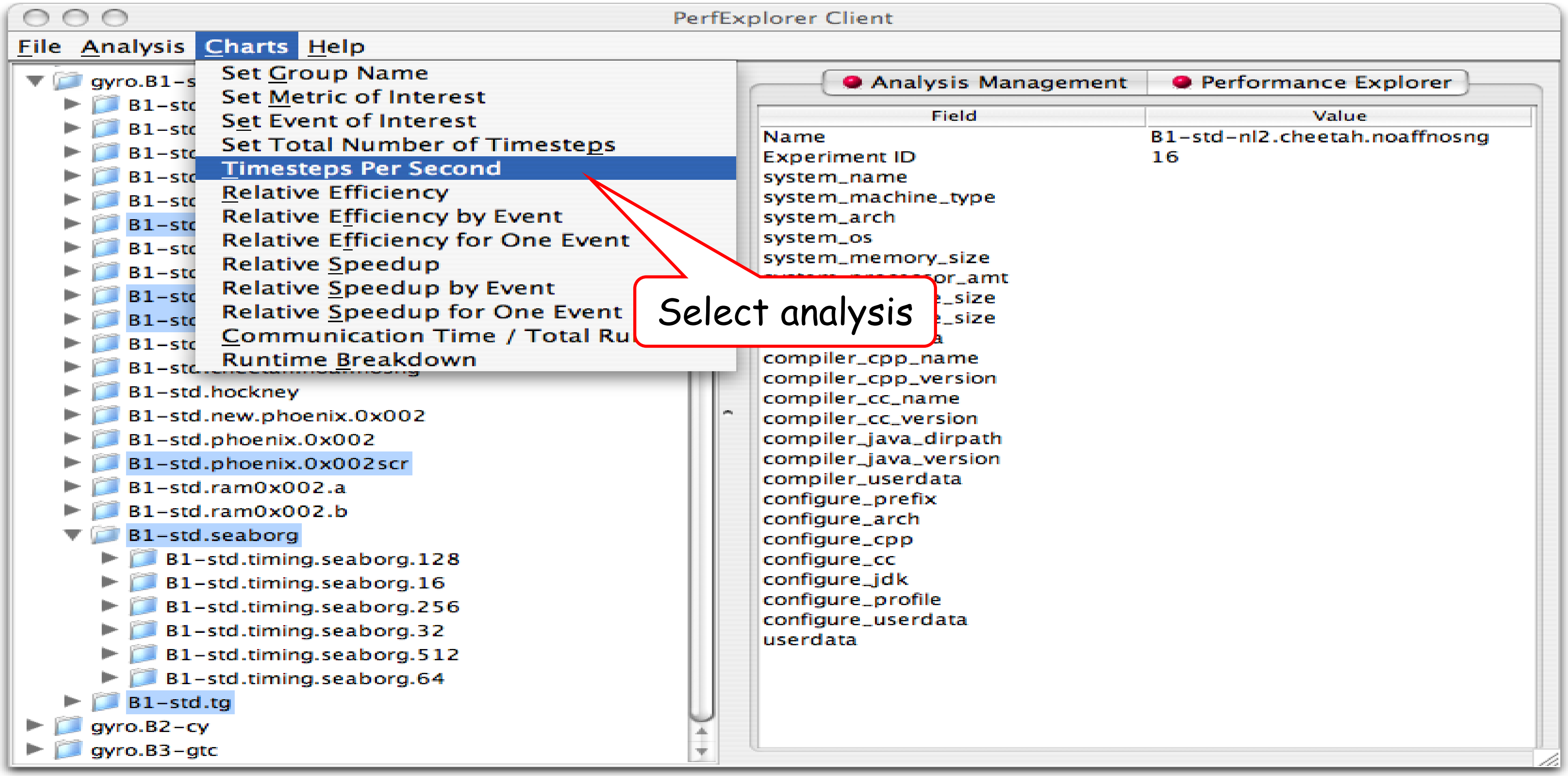
PerfExplorer - Interface


PerfExplorer Client
File Analysis Charts Help
Analysis Management
Performance Explorer
Select experiments and trials of interest
Experiment metadata
Data organized in application, experiment, trial structure (will allow arbitrary in future)

```
gyro.B1-std
  B1-std-hwpc.phoenix.0x002
  B1-std-inst.phoenix.0x002
  B1-std-inst.phoenix.0x002.profil
  B1-std-nl2.cheetah.affnosng
  B1-std-nl2.cheetah.affsng
  B1-std-nl2.cheetah.noaffnosng
  B1-std-nl2.phoenix.0x0
  B1-std-nl2.phoenix.0x002
  B1-std.53newest.phoenix.0x
  B1-std.cheetah.affnosr
  B1-std.cheetah.affsng
  B1-std.cheetah.noaffnc
  B1-std.hockney
  B1-std.new.phoenix.0x
  B1-std.phoenix.0x002
  B1-std.phoenix.0x002scr
  B1-std.ram0x002.a
  B1-std.ram0x002.b
  B1-std.seaborg
    B1-std.timing.seab
    B1-std.timing.seaborg.16
    B1-std.timing.seaborg.256
    B1-std.timing.seaborg.32
    B1-std.timing.seaborg.512
    B1-std.timing.seaborg.64
  B1-std.tg
gyro.B2-cy
gyro.B3-gtc
```

Field	Value
Name	B1-std-nl2.cheetah.noaffnosng
Experiment ID	16
system_name	
system_machine_type	
system_arch	
system_os	
system_memory_size	
system_processor_amt	
system_l1_cache_size	
2_cache_size	
serdata	
_cpp_name	
_cpp_version	
_cc_name	
_cc_version	
compiler_java_dirpath	
compiler_java_version	
compiler_userdata	
configure_prefix	
configure_arch	

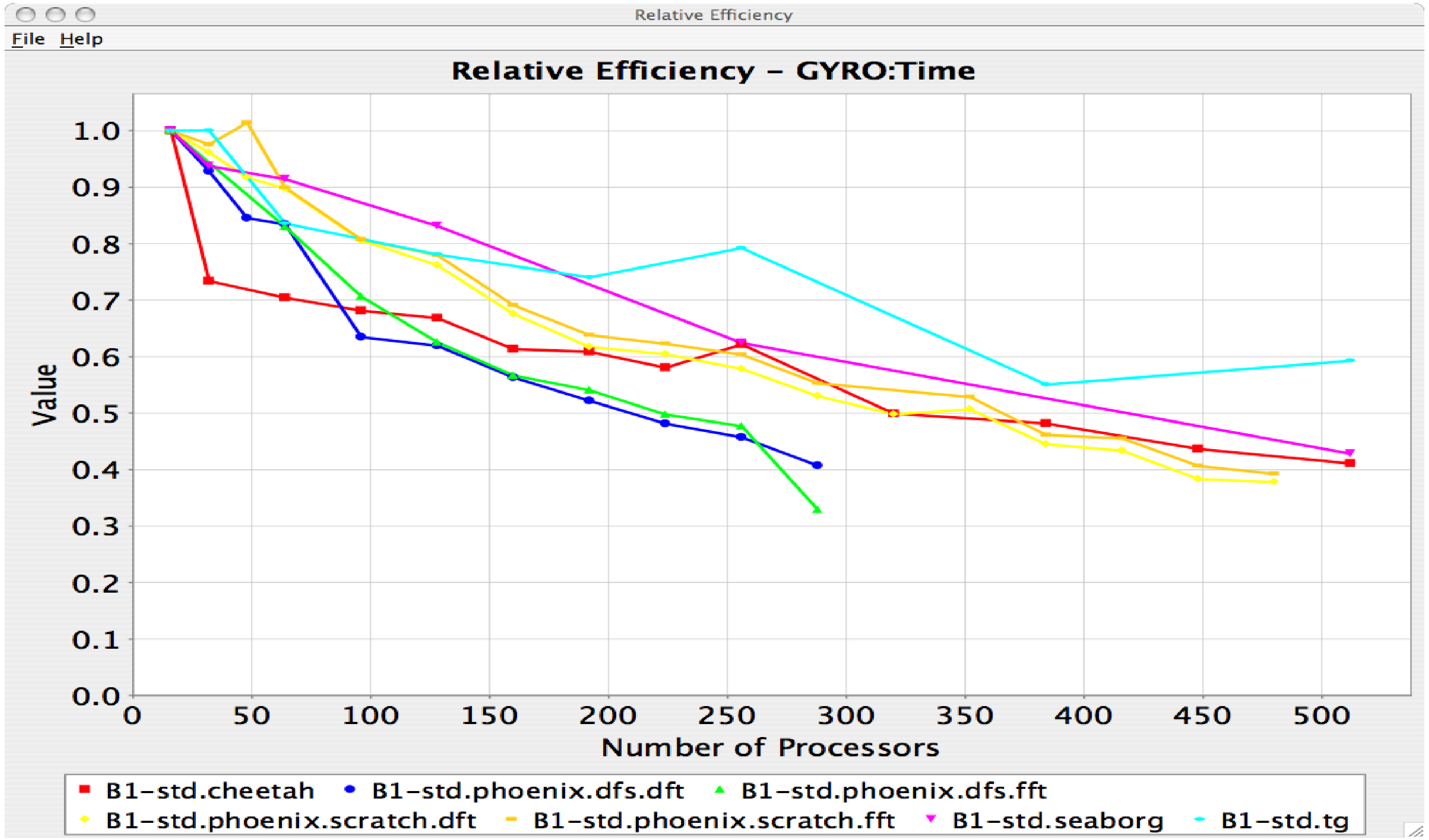
PerfExplorer - Interface



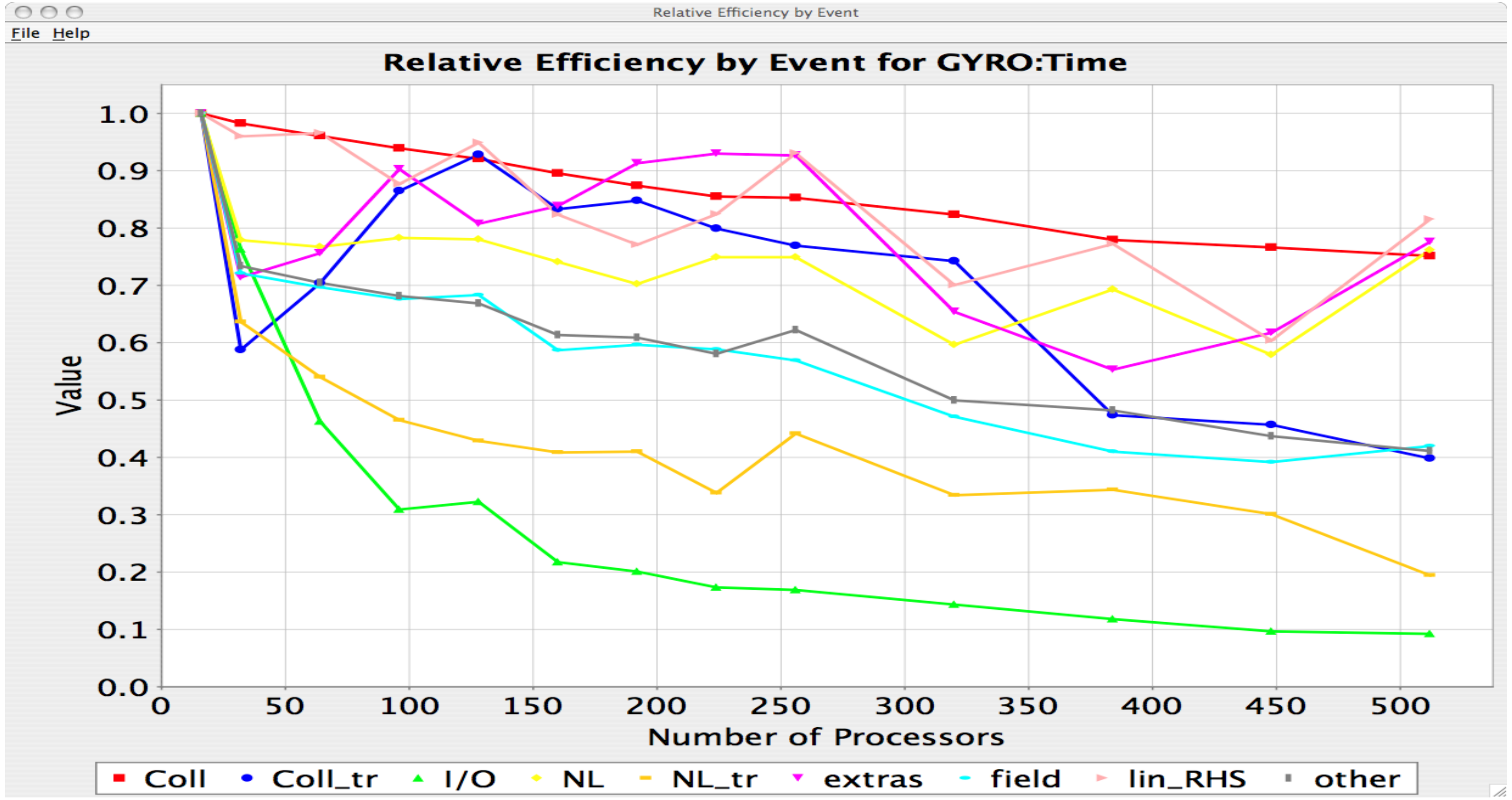
gyro.B1-s
B1-std.hockney
B1-std.new.phoenix.0x002
B1-std.phoenix.0x002
B1-std.phoenix.0x002scr
B1-std.ram0x002.a
B1-std.ram0x002.b
B1-std.seaborg
B1-std.timing.seaborg.128
B1-std.timing.seaborg.16
B1-std.timing.seaborg.256
B1-std.timing.seaborg.32
B1-std.timing.seaborg.512
B1-std.timing.seaborg.64
B1-std.tg
gyro.B2-cy
gyro.B3-gtc

Field	Value
Name	B1-std-nl2.cheetah.noaffnosng
Experiment ID	16
system_name	
system_machine_type	
system_arch	
system_os	
system_memory_size	
compiler_cpp_name	
compiler_cpp_version	
compiler_cc_name	
compiler_cc_version	
compiler_java_dirpath	
compiler_java_version	
compiler_userdata	
configure_prefix	
configure_arch	
configure_cpp	
configure_cc	
configure_jdk	
configure_profile	
configure_userdata	
userdata	

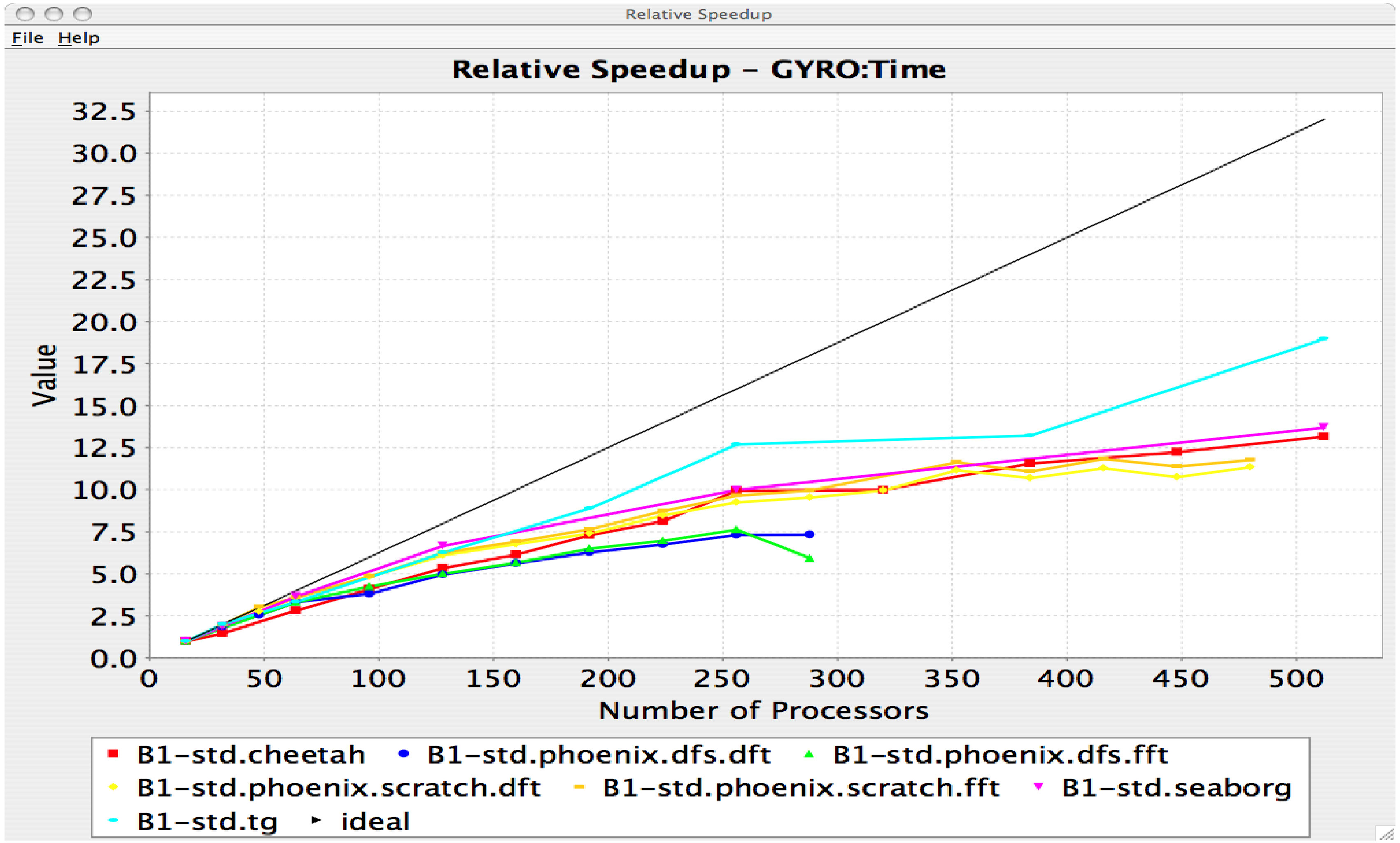
PerfExplorer - Relative Efficiency Plots



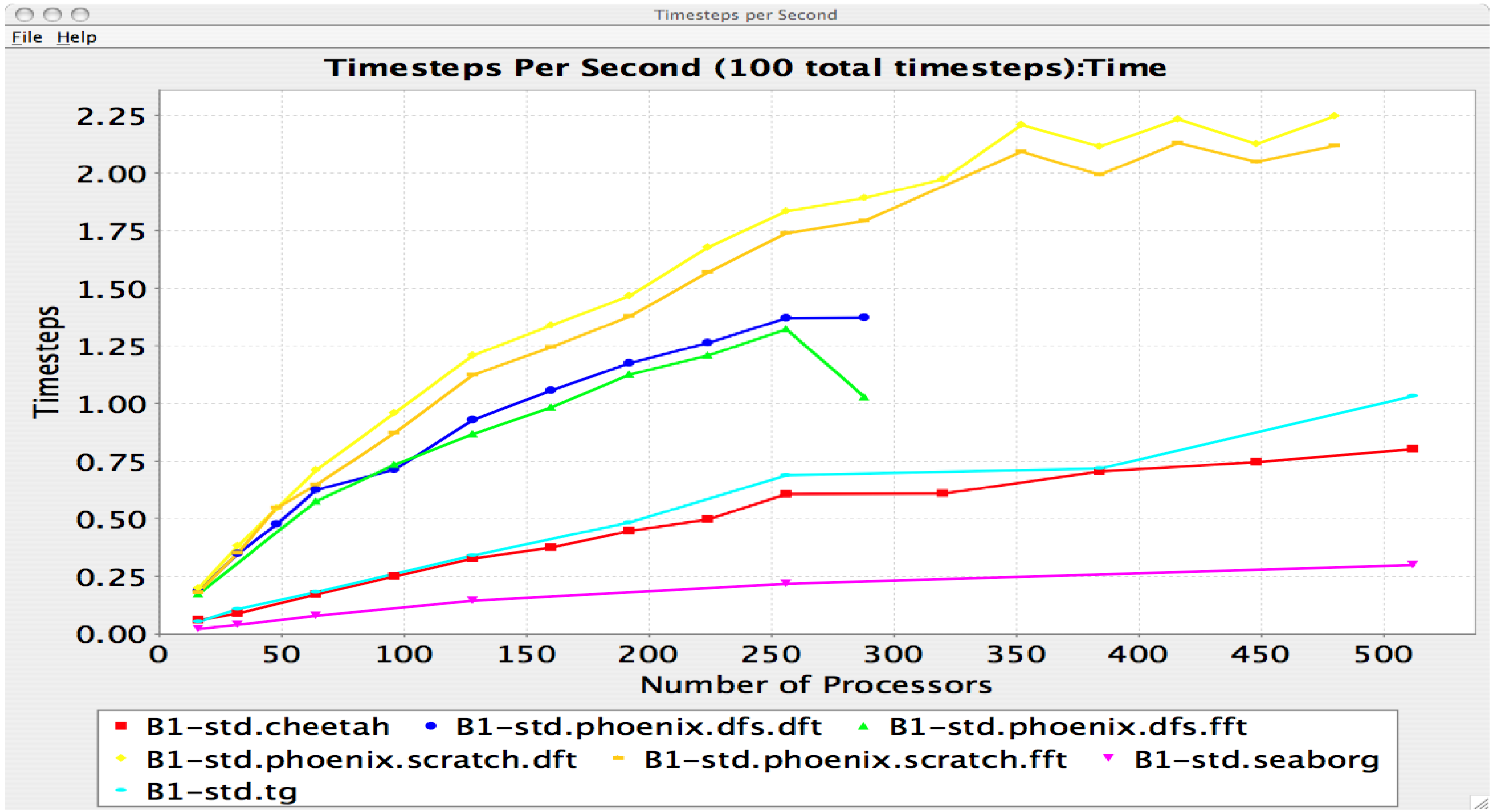
PerfExplorer - Relative Efficiency by Routine



PerfExplorer - Relative Speedup

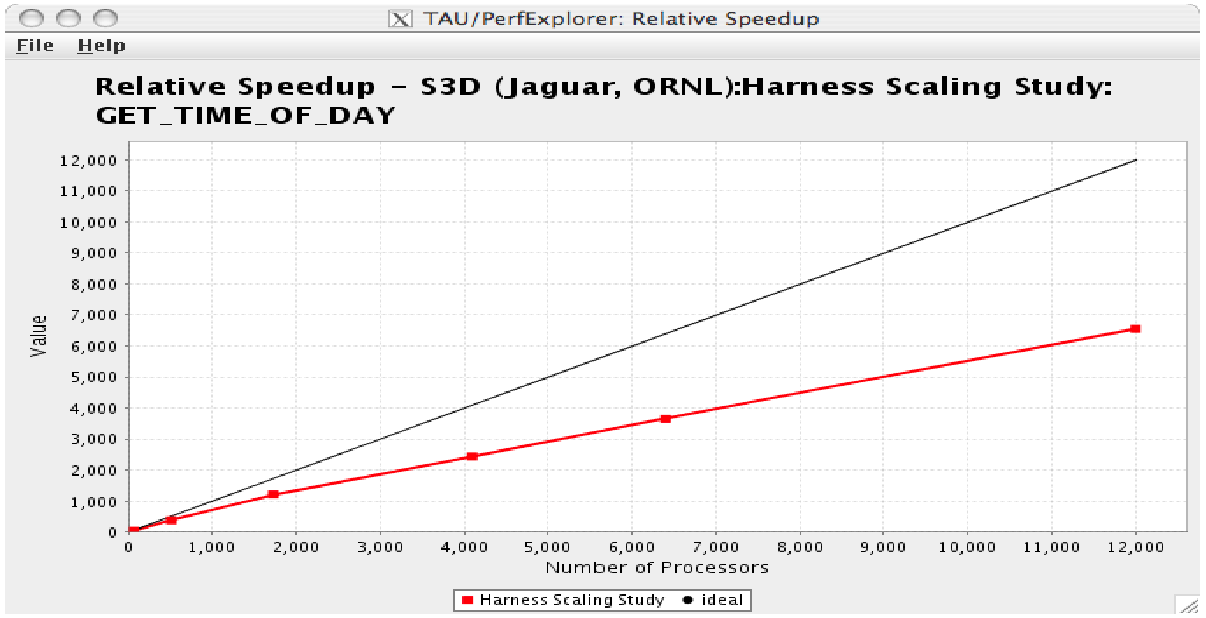


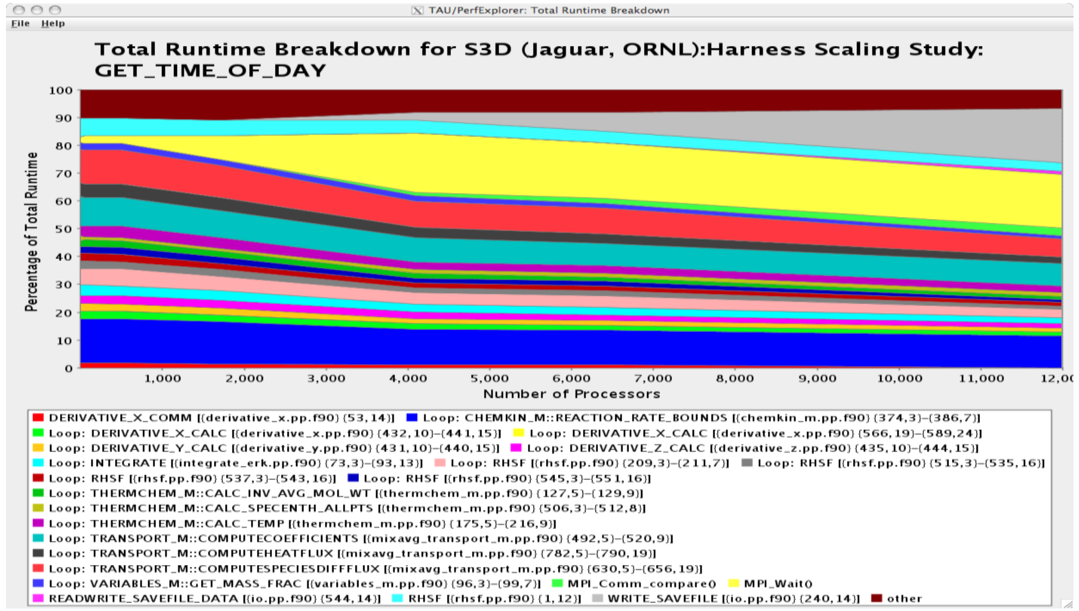
PerfExplorer - Timesteps Per Second



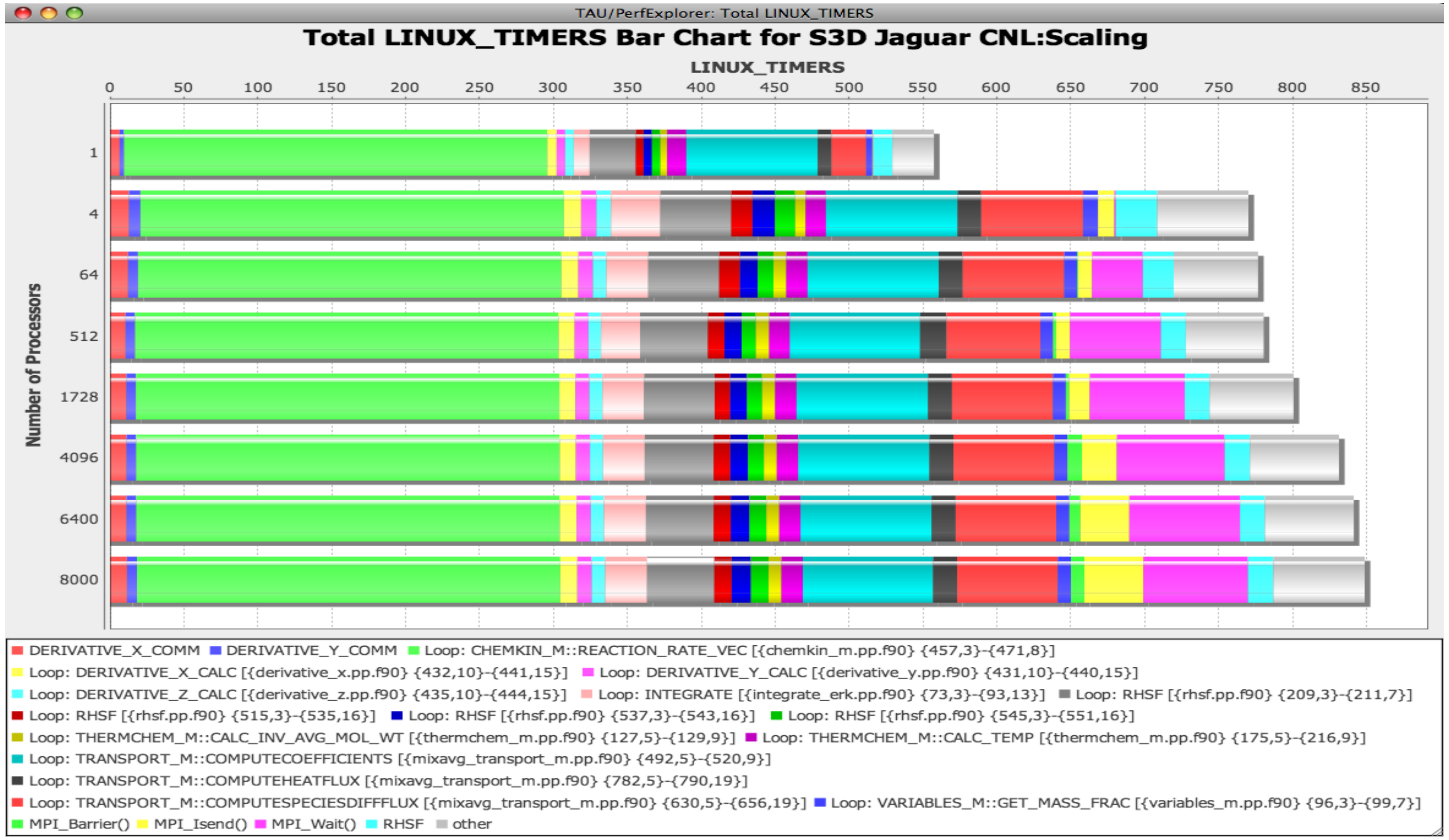
Evaluate Scalability

- Goal: How does my application scale? What bottlenecks occur at what core counts?
- Load profiles in taudb database and examine with PerfExplorer

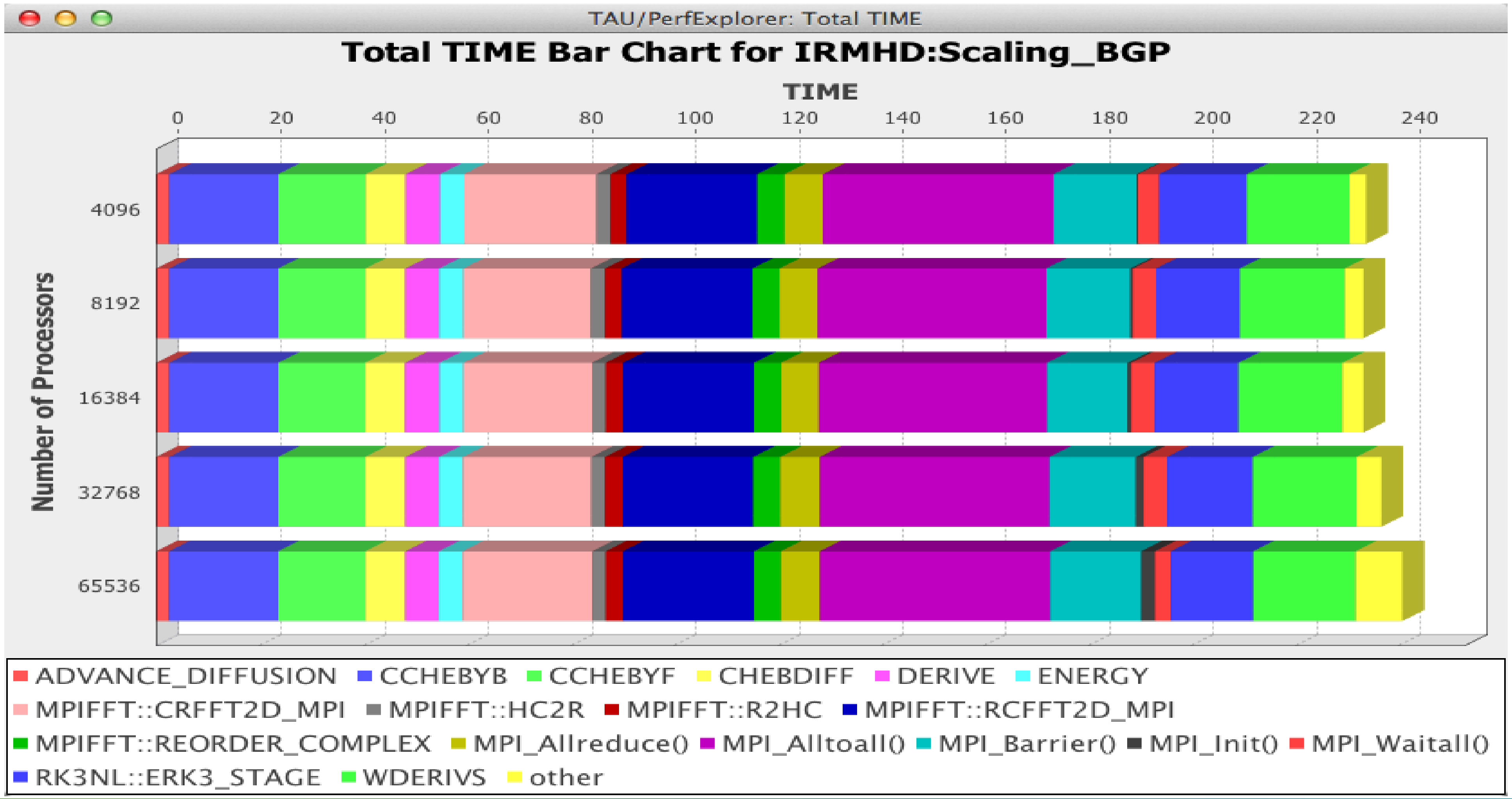




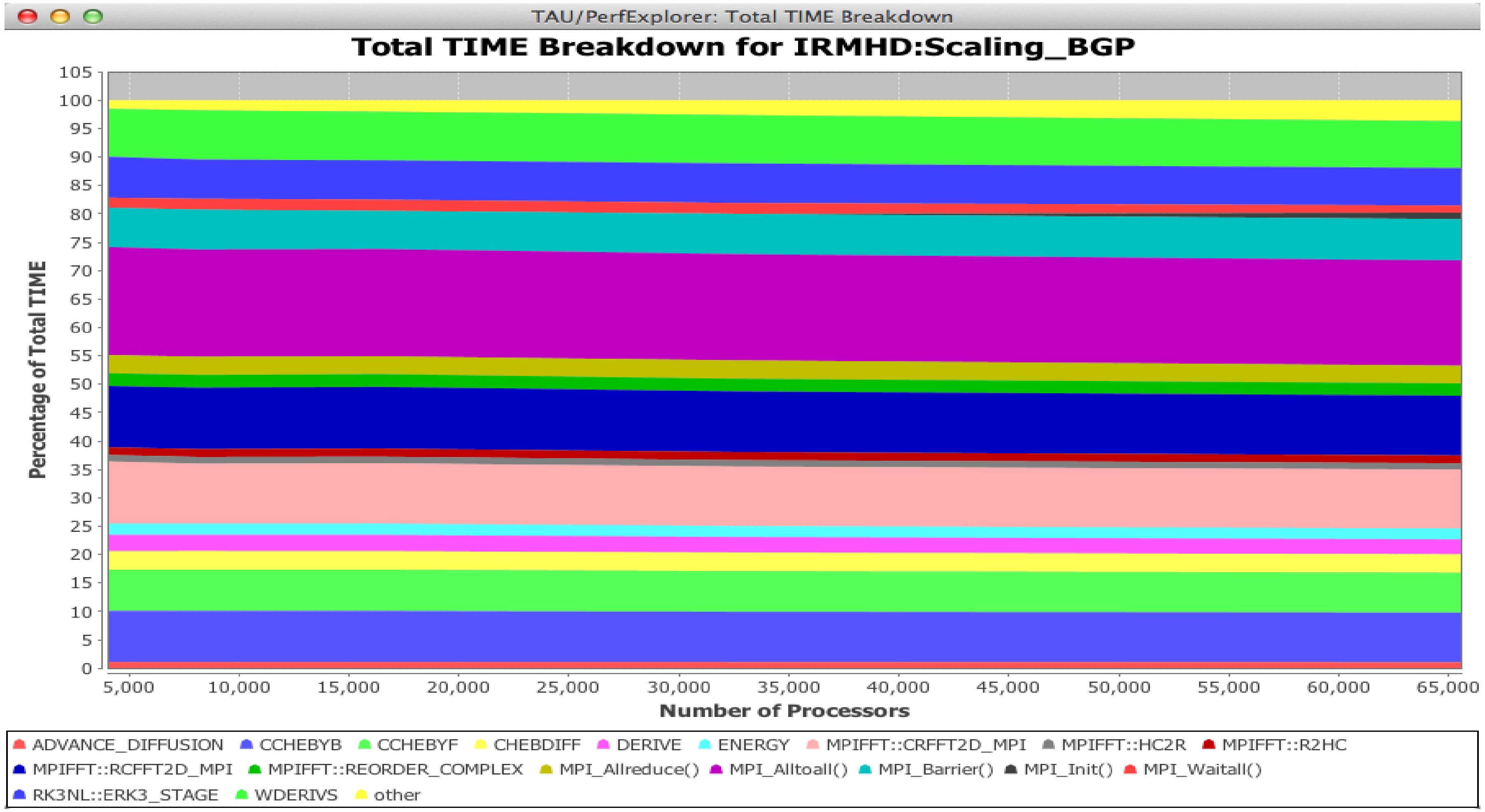
Evaluate Scalability



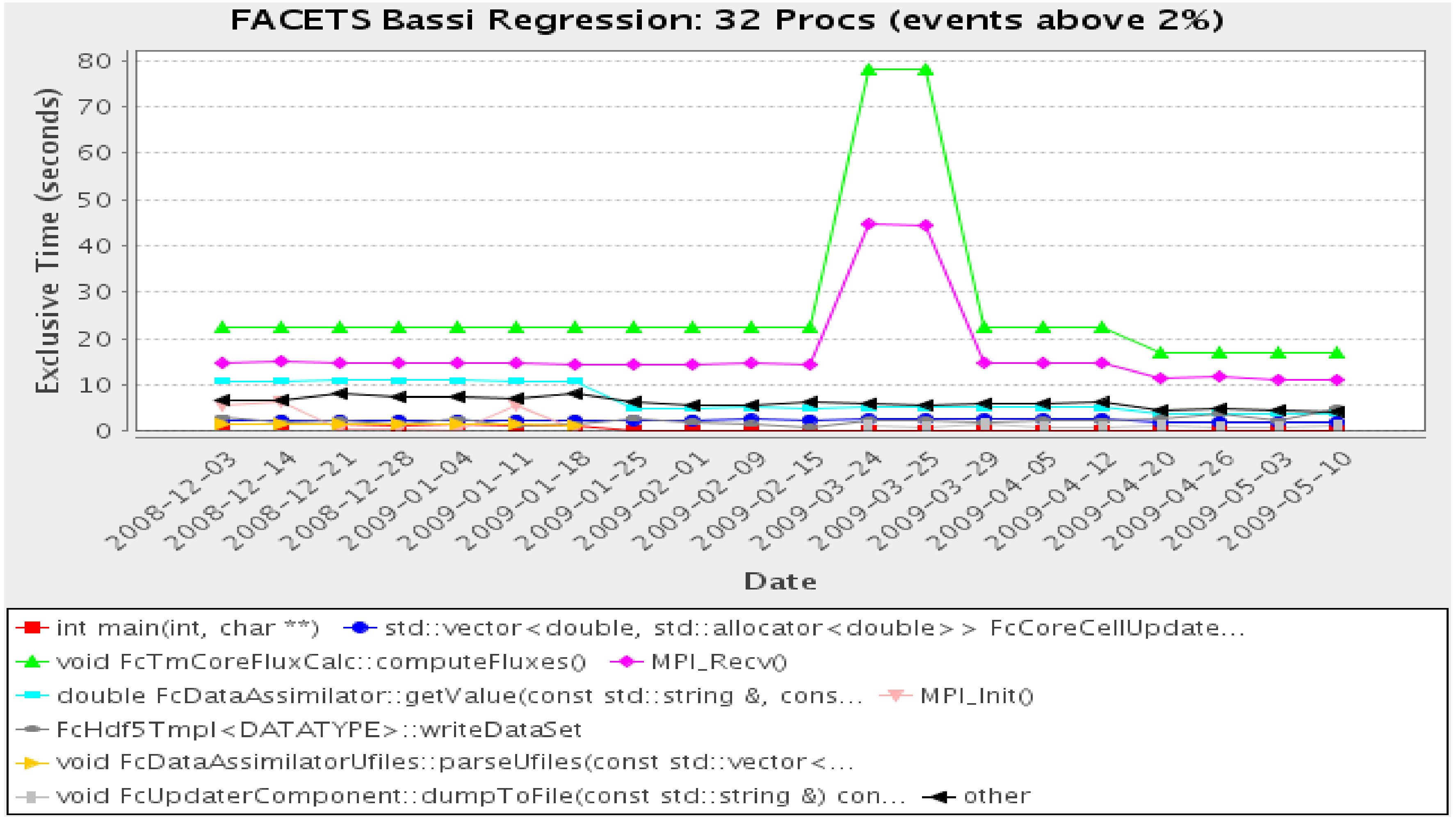
PerfExplorer



PerfExplorer



Performance Regression Testing



Download TAU from U. Oregon

http://tau.uoregon.edu

http://www.hpclinux.com [LiveDVD, OVA]

Free download, open source, BSD license

Parallel application performance analysis case studies

The VI-HPS Team

Outline

- Case I:
  - NPB3.3_MZ_MPI/**BT-MZ** (MPI+OpenMP) on *MARCONI-KNL*: load balancing
- Case II:
  - **k-Wave** (MPI+OpenMP) on *Salomon*: load-balancing in FFTW OpenMP parallel regions
- Case III:
  - **ICON** (MPI) on *Mistral*: automatic trace analysis of critical path of execution
- Case IV:
  - **PIConGPU** (MPI+CUDA): computation offload to multiple attached accelerator devices
- Case V:
  - **TensorFlow** (Python+CUDA): interpreted & compiled heterogeneous execution measurement

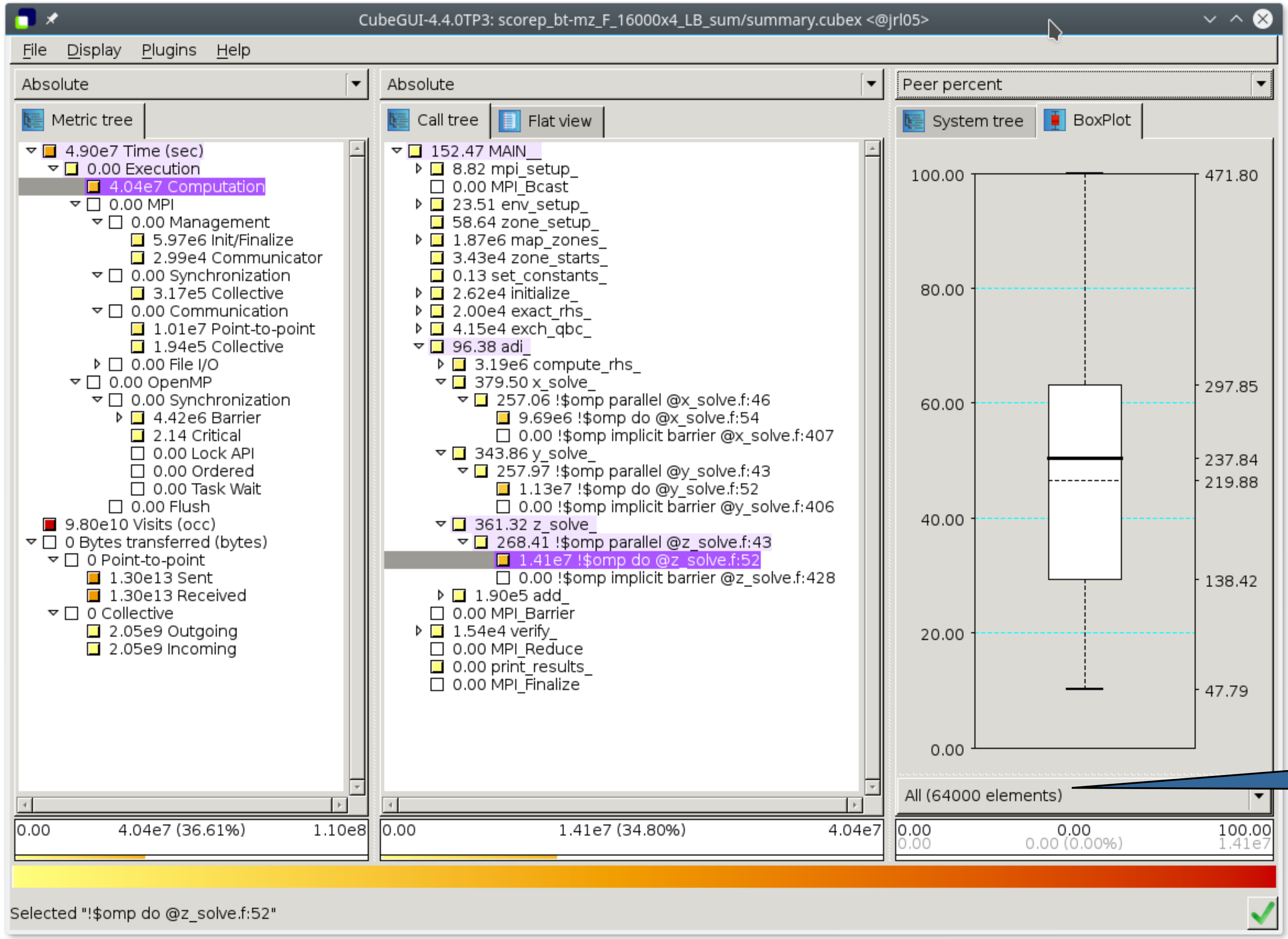
Case I: NPB3.3-MZ-MPI/BT-MZ: balancing OMP threads per process

- Same F77 benchmark code as used in tutorial exercise, CLASS=F (128x128 zones)
- Using Intel compilers and Intel MPI on *MARCONI-KNL* (68C), **`-xMIC-AVX512`**
- 4,000 MPI processes (4 ranks/KNL), OMP_NUM_THREADS=64
  - Default execution configuration "balances" number of OpenMP threads per MPI process
  - Threads reassigned from processes with simpler zones to those with more complex zones
- Intel compiler configuration file used when instrumenting with Score-P
  - Avoids instrumenting small/frequently-executed routines

- Since "balancing" scheme doesn't take account of threads (cores) per compute node, some KNL processors end up more over-subscribed
  - whereas 39 KNL nodes received 4 MPI processes each with 67 OpenMP threads (268 threads), two KNL nodes had 4 MPI processes each with only 62 OpenMP threads (248 threads)

⇒12% better performance exploiting hyper-threading and thread "balancing"

BT-MZ.F Score-P summary profile: 16p16000x4 “balancing” active


CubeGUI-4.4.0TP3: scorep_bt-mz_F_16000x4_LB_sum/summary.cubex <@jrl05>
File Display Plugins Help
Absolute
Absolute
Peer percent
Metric tree
Call tree
Flat view
System tree
BoxPlot
100.00
80.00
60.00
40.00
20.00
0.00
471.80
297.85
237.84
219.88
138.42
47.79
All (64000 elements)
0.00 4.04e7 (36.61%) 1.10e8
0.00 1.41e7 (34.80%) 4.04e7
0.00 0.00 0.00 (0.00%) 100.00 1.41e7

```
▽ 4.90e7 Time (sec)
  ▽ 0.00 Execution
      4.04e7 Computation
    ▽ 0.00 MPI
      ▽ 0.00 Management
          5.97e6 Init/Finalize
          2.99e4 Communicator
      ▽ 0.00 Synchronization
          3.17e5 Collective
      ▽ 0.00 Communication
          1.01e7 Point-to-point
          1.94e5 Collective
      ▷ 0.00 File I/O
    ▽ 0.00 OpenMP
      ▽ 0.00 Synchronization
        ▷ 4.42e6 Barrier
          2.14 Critical
          0.00 Lock API
          0.00 Ordered
          0.00 Task Wait
        0.00 Flush
  9.80e10 Visits (occ)
▽ 0 Bytes transferred (bytes)
  ▽ 0 Point-to-point
      1.30e13 Sent
      1.30e13 Received
  ▽ 0 Collective
      2.05e9 Outgoing
      2.05e9 Incoming
```

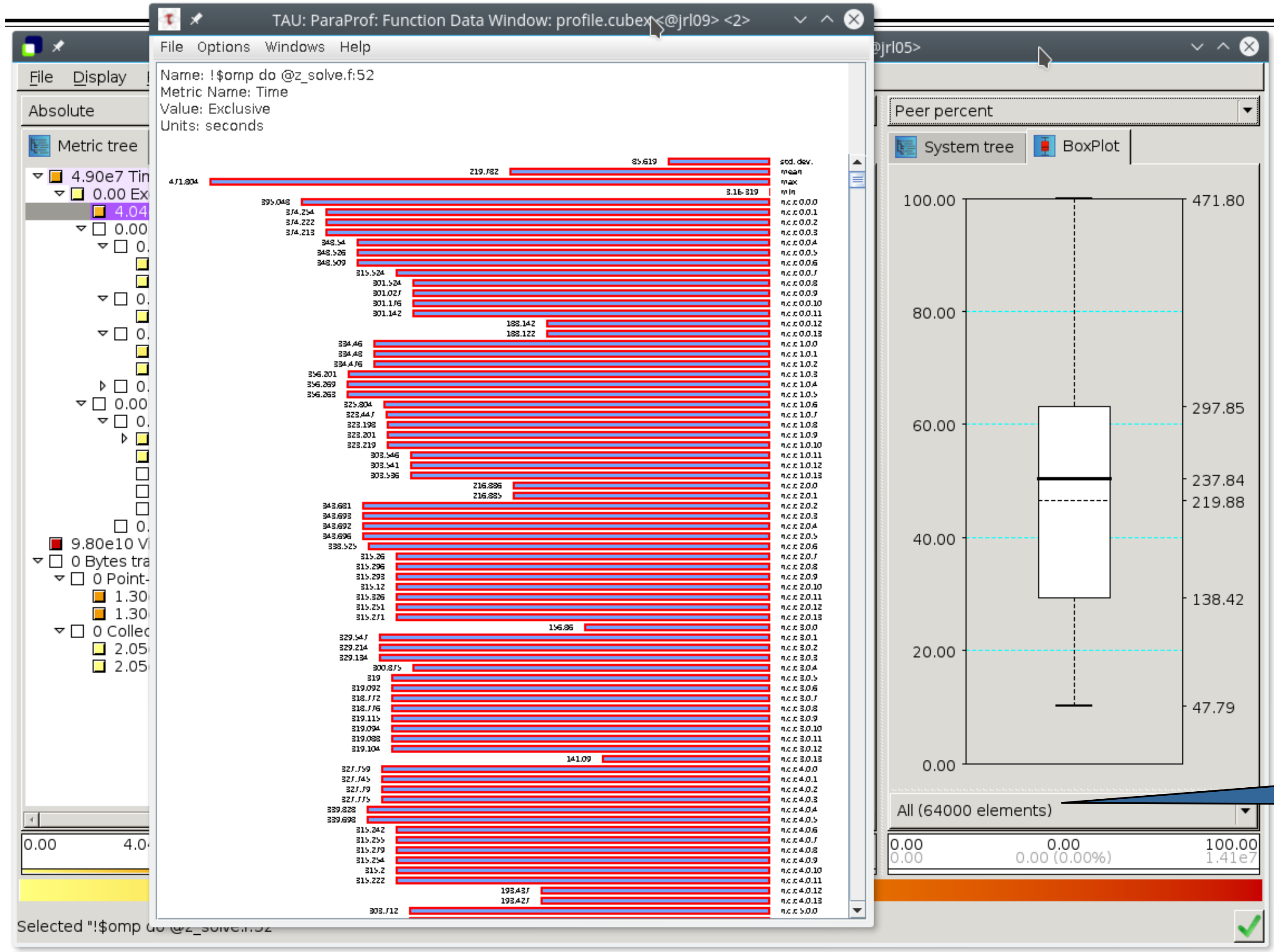
```
▽ 152.47 MAIN__
  ▷ 8.82 mpi_setup_
    0.00 MPI_Bcast
  ▷ 23.51 env_setup_
    58.64 zone_setup_
  ▷ 1.87e6 map_zones_
    3.43e4 zone_starts_
    0.13 set_constants_
  ▷ 2.62e4 initialize_
  ▷ 2.00e4 exact_rhs_
  ▷ 4.15e4 exch_qbc_
  ▽ 96.38 adi_
    ▷ 3.19e6 compute_rhs_
    ▽ 379.50 x_solve_
      ▽ 257.06 !$omp parallel @x_solve.f:46
          9.69e6 !$omp do @x_solve.f:54
          0.00 !$omp implicit barrier @x_solve.f:407
    ▽ 343.86 y_solve_
      ▽ 257.97 !$omp parallel @y_solve.f:43
          1.13e7 !$omp do @y_solve.f:52
          0.00 !$omp implicit barrier @y_solve.f:406
    ▽ 361.32 z_solve_
      ▽ 268.41 !$omp parallel @z_solve.f:43
          1.41e7 !$omp do @z_solve.f:52
          0.00 !$omp implicit barrier @z_solve.f:428
    ▷ 1.90e5 add_
    0.00 MPI_Barrier
  ▷ 1.54e4 verify_
    0.00 MPI_Reduce
    0.00 print_results_
    0.00 MPI_Finalize
```

Selected "!$omp do @z_solve.f:52"

- BT-MZ class F (12032x8960x250)
- Wall clock time 1221 seconds (18.3 Mop/s total)
  - 15.0% MPI, 4.0% OMP
- Large computational imbalance for threads in each ADI solver direction, despite attempt at “balancing”

64,000 OpenMP threads in total, running on 1000 KNL nodes

BT-MZ.F Score-P summary profile: alternative presentations



TAU: ParaProf: Function Data Window: profile.cubex <@jrl09> <2>

Name: !$omp do @z_solve.f:52
Metric Name: Time
Value: Exclusive
Units: seconds

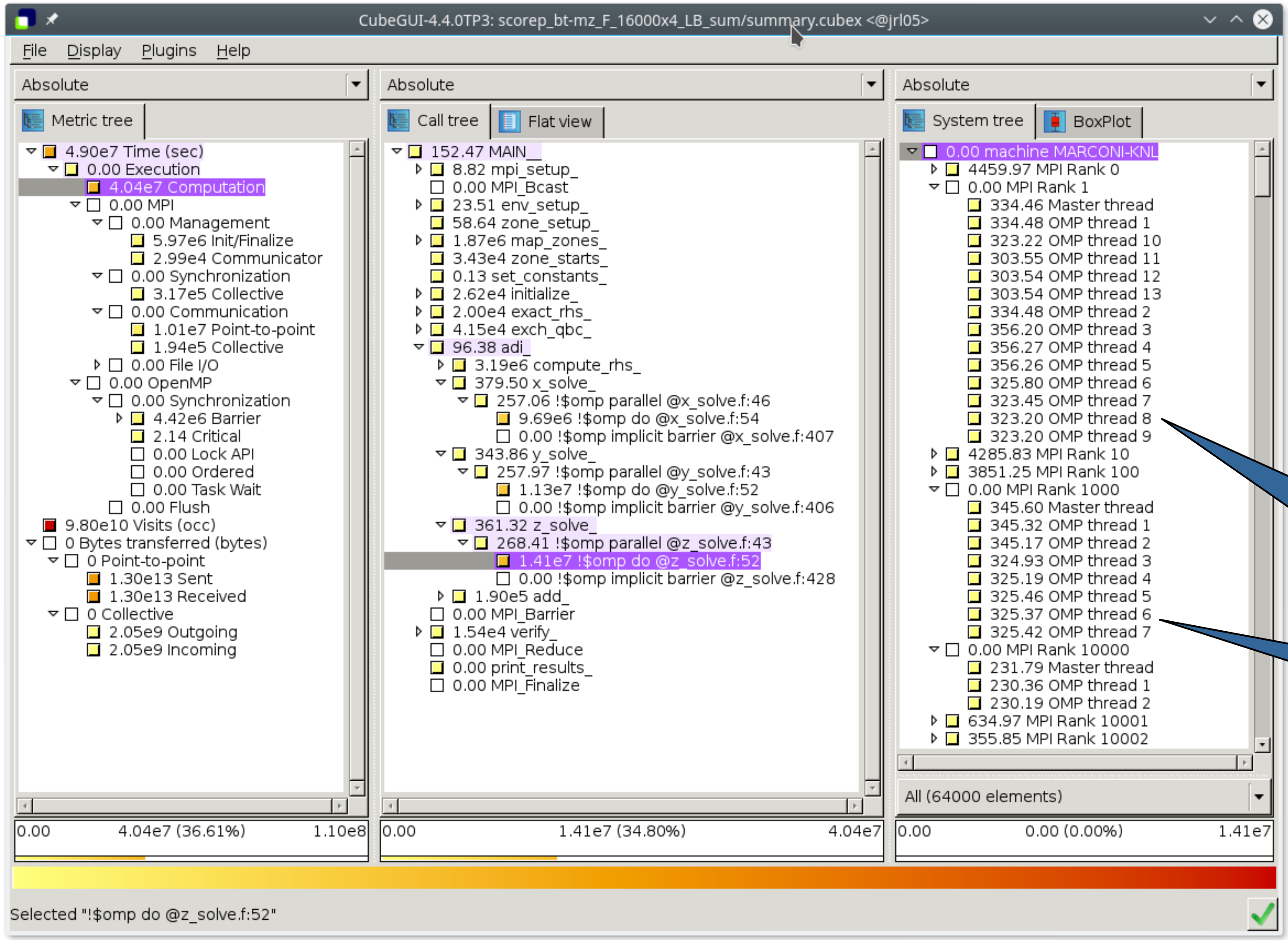
Selected "!$omp do @z_solve.f:52

`!$omp do @z_solve.f52` loop computation time by OpenMP thread

- CUBE system tree scrollable list limited to showing values of around 30 processes/threads at a time
- CUBE boxplot summarizes metric value distribution
  - min 47.8 to max 471.8 seconds
- ParaProf charts the same metric values graphically

64,000 OpenMP threads in total, running on 1000 KNL nodes

BT-MZ.F Score-P summary profile: 16p16000x4 “balancing” active


CubeGUI-4.4.0TP3: scorep_bt-mz_F_16000x4_LB_sum/summary.cubex <@jrl05>
File Display Plugins Help
Absolute
Metric tree
Call tree
Flat view
System tree
BoxPlot

```
Metric tree
4.90e7 Time (sec)
  0.00 Execution
    4.04e7 Computation
    0.00 MPI
      0.00 Management
        5.97e6 Init/Finalize
        2.99e4 Communicator
      0.00 Synchronization
        3.17e5 Collective
      0.00 Communication
        1.01e7 Point-to-point
        1.94e5 Collective
      0.00 File I/O
    0.00 OpenMP
      0.00 Synchronization
        4.42e6 Barrier
        2.14 Critical
        0.00 Lock API
        0.00 Ordered
        0.00 Task Wait
      0.00 Flush
9.80e10 Visits (occ)
0 Bytes transferred (bytes)
  0 Point-to-point
    1.30e13 Sent
    1.30e13 Received
  0 Collective
    2.05e9 Outgoing
    2.05e9 Incoming

0.00    4.04e7 (36.61%)    1.10e8
```

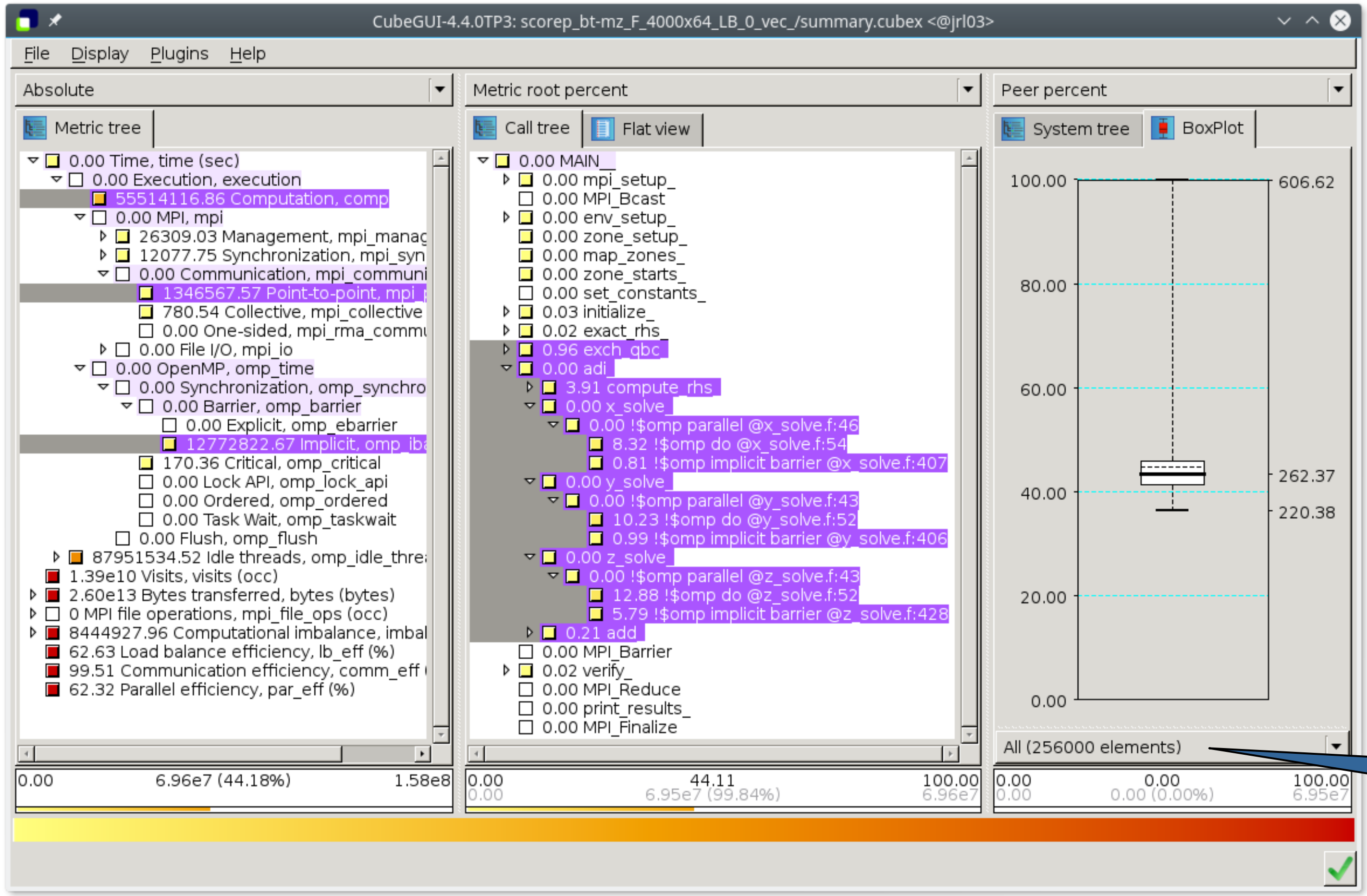
```
Call tree
152.47 MAIN__
  8.82 mpi_setup_
  0.00 MPI_Bcast
  23.51 env_setup_
  58.64 zone_setup_
  1.87e6 map_zones_
  3.43e4 zone_starts_
  0.13 set_constants_
  2.62e4 initialize_
  2.00e4 exact_rhs_
  4.15e4 exch_qbc_
  96.38 adi_
    3.19e6 compute_rhs_
    379.50 x_solve_
      257.06 !$omp parallel @x_solve.f:46
        9.69e6 !$omp do @x_solve.f:54
        0.00 !$omp implicit barrier @x_solve.f:407
    343.86 y_solve_
      257.97 !$omp parallel @y_solve.f:43
        1.13e7 !$omp do @y_solve.f:52
        0.00 !$omp implicit barrier @y_solve.f:406
    361.32 z_solve_
      268.41 !$omp parallel @z_solve.f:43
        1.41e7 !$omp do @z_solve.f:52
        0.00 !$omp implicit barrier @z_solve.f:428
    1.90e5 add_
  0.00 MPI_Barrier
  1.54e4 verify_
  0.00 MPI_Reduce
  0.00 print_results_
  0.00 MPI_Finalize

0.00    1.41e7 (34.80%)    4.04e7
```

```
System tree
0.00 machine MARCONI-KNL
  4459.97 MPI Rank 0
  0.00 MPI Rank 1
    334.46 Master thread
    334.48 OMP thread 1
    323.22 OMP thread 10
    303.55 OMP thread 11
    303.54 OMP thread 12
    303.54 OMP thread 13
    334.48 OMP thread 2
    356.20 OMP thread 3
    356.27 OMP thread 4
    356.26 OMP thread 5
    325.80 OMP thread 6
    323.45 OMP thread 7
    323.20 OMP thread 8
    323.20 OMP thread 9
  4285.83 MPI Rank 10
  3851.25 MPI Rank 100
  0.00 MPI Rank 1000
    345.60 Master thread
    345.32 OMP thread 1
    345.17 OMP thread 2
    324.93 OMP thread 3
    325.19 OMP thread 4
    325.46 OMP thread 5
    325.37 OMP thread 6
    325.42 OMP thread 7
  0.00 MPI Rank 10000
    231.79 Master thread
    230.36 OMP thread 1
    230.19 OMP thread 2
  634.97 MPI Rank 10001
  355.85 MPI Rank 10002

All (64000 elements)
0.00    0.00 (0.00%)    1.41e7

Selected "!$omp do @z_solve.f:52"
```

- BT-MZ class F (12032x8960x250)
- Wall clock time 1221 seconds (18.3 Mop/s total)
- 15% MPI, 4% OMP
- Computational imbalance for threads, despite attempt at “balancing”

Rank 1 has been assigned 14 threads, whereas rank 10000 only has 3 threads

BT-MZ.F Score-P summary profile: 4p4000x64 no “balancing”



CubeGUI-4.4.0TP3: scorep_bt-mz_F_4000x64_LB_0_vec_/summary.cubex <@jrl03>

File Display Plugins Help

Absolute | Metric root percent | Peer percent

Metric tree | Call tree | Flat view | System tree | BoxPlot

```
0.00 Time, time (sec)
  0.00 Execution, execution
    55514116.86 Computation, comp
    0.00 MPI, mpi
      26309.03 Management, mpi_manag
      12077.75 Synchronization, mpi_syn
      0.00 Communication, mpi_communi
        1346567.57 Point-to-point, mpi_
        780.54 Collective, mpi_collective
        0.00 One-sided, mpi_rma_commu
      0.00 File I/O, mpi_io
    0.00 OpenMP, omp_time
      0.00 Synchronization, omp_synchro
        0.00 Barrier, omp_barrier
          0.00 Explicit, omp_ebarrier
          12772822.67 Implicit, omp_iba
        170.36 Critical, omp_critical
        0.00 Lock API, omp_lock_api
        0.00 Ordered, omp_ordered
        0.00 Task Wait, omp_taskwait
      0.00 Flush, omp_flush
  87951534.52 Idle threads, omp_idle_threa
1.39e10 Visits, visits (occ)
2.60e13 Bytes transferred, bytes (bytes)
0 MPI file operations, mpi_file_ops (occ)
8444927.96 Computational imbalance, imbal
62.63 Load balance efficiency, lb_eff (%)
99.51 Communication efficiency, comm_eff (
62.32 Parallel efficiency, par_eff (%)
```

```
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 set_constants_
  0.03 initialize_
  0.02 exact_rhs_
  0.96 exch_qbc_
  0.00 adi_
    3.91 compute_rhs_
    0.00 x_solve_
      0.00 !$omp parallel @x_solve.f:46
        8.32 !$omp do @x_solve.f:54
        0.81 !$omp implicit barrier @x_solve.f:407
    0.00 y_solve_
      0.00 !$omp parallel @y_solve.f:43
        10.23 !$omp do @y_solve.f:52
        0.99 !$omp implicit barrier @y_solve.f:406
    0.00 z_solve_
      0.00 !$omp parallel @z_solve.f:43
        12.88 !$omp do @z_solve.f:52
        5.79 !$omp implicit barrier @z_solve.f:428
    0.21 add_
  0.00 MPI_Barrier
  0.02 verify_
  0.00 MPI_Reduce
  0.00 print_results_
  0.00 MPI_Finalize
```

All (256000 elements)

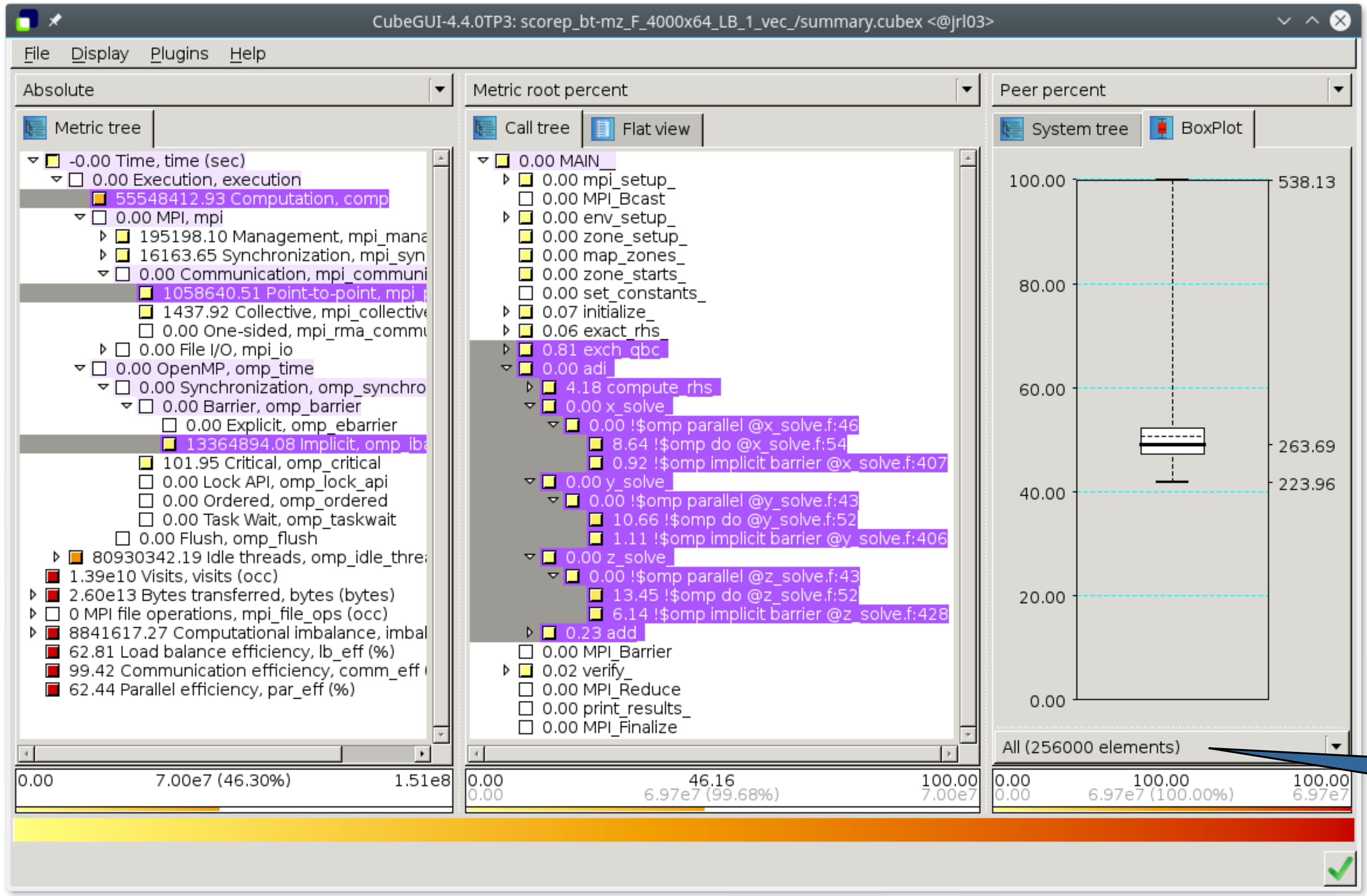
```
0.00        6.96e7 (44.18%)        1.58e8
0.00             44.11             100.00
0.00        6.95e7 (99.84%)        6.96e7
0.00             0.00              100.00
0.00        0.00 (0.00%)           6.95e7
```

- BT-MZ class F (12032x8960x250): 4 MPI ranks/node, **64** OpenMP threads/rank =256 threads/node
- No load “balancing”
- 4000 KNL nodes using HW threading

- Wall clock time 603 seconds (37.0 Mop/s total)
  - time ranging from 220 to 606s

256,000 OpenMP threads in total, running on 4,000 KNL nodes

BT-MZ.F Score-P summary profile: 4p4000x64 “balancing” active



CubeGUI-4.4.0TP3: scorep_bt-mz_F_4000x64_LB_1_vec_/summary.cubex <@jrl03>

File Display Plugins Help

Absolute | Metric root percent | Peer percent

Metric tree | Call tree | Flat view | System tree | BoxPlot

```
-0.00 Time, time (sec)
  0.00 Execution, execution
    55548412.93 Computation, comp
    0.00 MPI, mpi
      195198.10 Management, mpi_mana
      16163.65 Synchronization, mpi_syn
      0.00 Communication, mpi_communi
        1058640.51 Point-to-point, mpi_
        1437.92 Collective, mpi_collective
        0.00 One-sided, mpi_rma_commu
      0.00 File I/O, mpi_io
    0.00 OpenMP, omp_time
      0.00 Synchronization, omp_synchro
        0.00 Barrier, omp_barrier
          0.00 Explicit, omp_ebarrier
          13364894.08 Implicit, omp_iba
        101.95 Critical, omp_critical
        0.00 Lock API, omp_lock_api
        0.00 Ordered, omp_ordered
        0.00 Task Wait, omp_taskwait
      0.00 Flush, omp_flush
  80930342.19 Idle threads, omp_idle_threa
1.39e10 Visits, visits (occ)
2.60e13 Bytes transferred, bytes (bytes)
0 MPI file operations, mpi_file_ops (occ)
8841617.27 Computational imbalance, imbal
62.81 Load balance efficiency, lb_eff (%)
99.42 Communication efficiency, comm_eff
62.44 Parallel efficiency, par_eff (%)
```

```
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 set_constants_
  0.07 initialize_
  0.06 exact_rhs_
  0.81 exch_qbc_
  0.00 adi_
    4.18 compute_rhs_
    0.00 x_solve_
      0.00 !$omp parallel @x_solve.f:46
        8.64 !$omp do @x_solve.f:54
        0.92 !$omp implicit barrier @x_solve.f:407
    0.00 y_solve_
      0.00 !$omp parallel @y_solve.f:43
        10.66 !$omp do @y_solve.f:52
        1.11 !$omp implicit barrier @y_solve.f:406
    0.00 z_solve_
      0.00 !$omp parallel @z_solve.f:43
        13.45 !$omp do @z_solve.f:52
        6.14 !$omp implicit barrier @z_solve.f:428
    0.23 add_
  0.00 MPI_Barrier
  0.02 verify_
  0.00 MPI_Reduce
  0.00 print_results_
  0.00 MPI_Finalize
```

All (256000 elements)

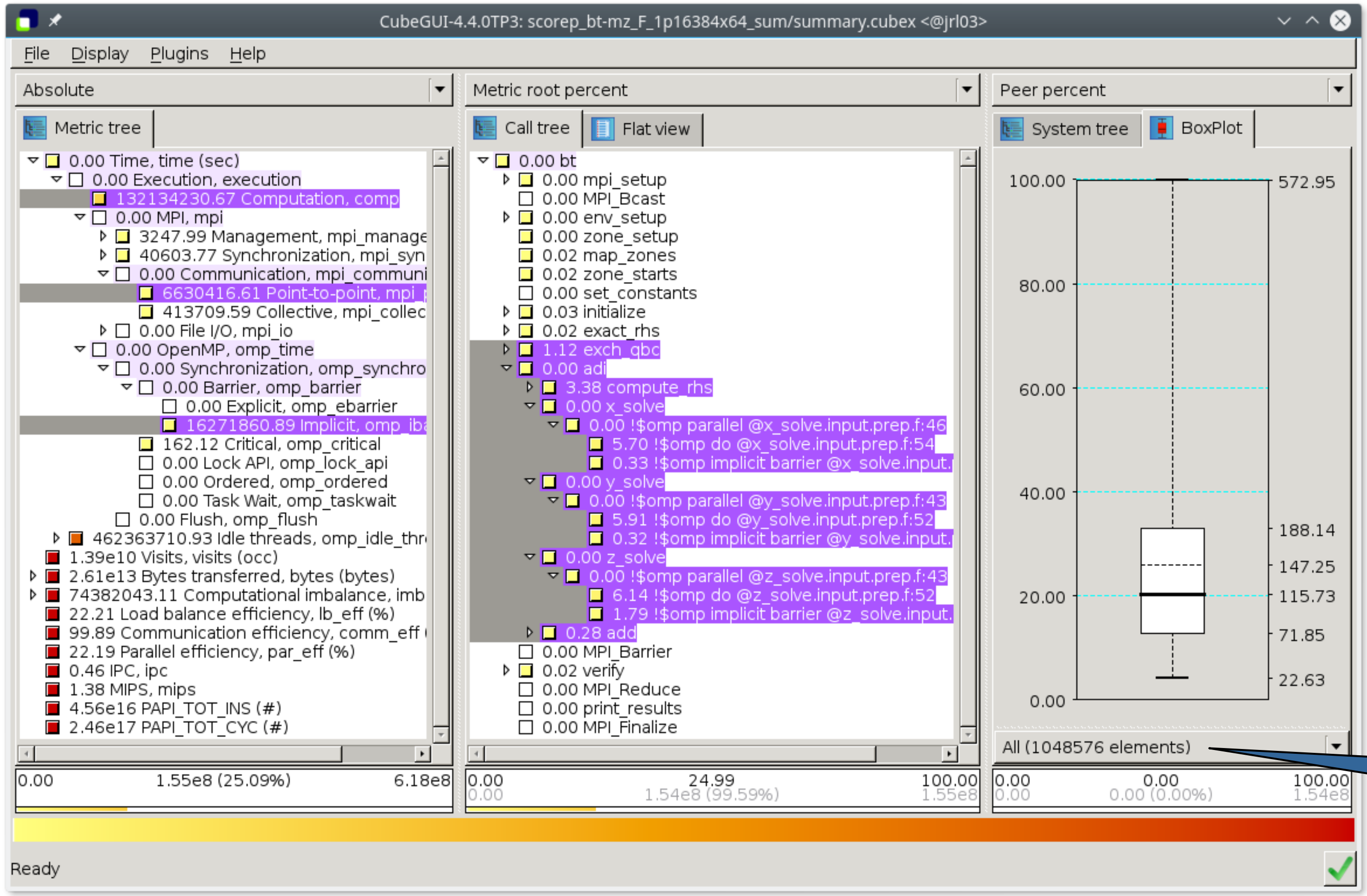
```
0.00      7.00e7 (46.30%)      1.51e8
0.00      46.16                100.00
0.00      6.97e7 (99.68%)      7.00e7
0.00      100.00               100.00
0.00      6.97e7 (100.00%)     6.97e7
```

- BT-MZ class F (12032x8960x250): 4 MPI ranks/node, **62-67** OpenMP threads/rank ~256 threads/node
- Static “balancing”
- 4000 KNL nodes using HW threading

- Wall clock time 531 seconds (42.0 Mop/s total)
  - time ranging from 224 to 538s
- **12% gain** from static balancing of OpenMP threads per MPI process

256,000 OpenMP threads in total, running on 4,000 KNL nodes

BT-MZ.F Score-P summary profile: 1p16364x64 on *JUQUEEN* BG/Q



CubeGUI-4.4.0TP3: scorep_bt-mz_F_1p16384x64_sum/summary.cubex <@jrl03>

File Display Plugins Help

Absolute | Metric root percent | Peer percent

Metric tree

```
▽ 0.00 Time, time (sec)
  ▽ 0.00 Execution, execution
      132134230.67 Computation, comp
    ▽ 0.00 MPI, mpi
      ▷ 3247.99 Management, mpi_manage
      ▷ 40603.77 Synchronization, mpi_syn
      ▽ 0.00 Communication, mpi_communi
          6630416.61 Point-to-point, mpi_
          413709.59 Collective, mpi_collec
      ▷ 0.00 File I/O, mpi_io
    ▽ 0.00 OpenMP, omp_time
      ▽ 0.00 Synchronization, omp_synchro
        ▽ 0.00 Barrier, omp_barrier
            0.00 Explicit, omp_ebarrier
            16271860.89 Implicit, omp_ib
          162.12 Critical, omp_critical
          0.00 Lock API, omp_lock_api
          0.00 Ordered, omp_ordered
          0.00 Task Wait, omp_taskwait
        0.00 Flush, omp_flush
  ▷ 462363710.93 Idle threads, omp_idle_thr
  1.39e10 Visits, visits (occ)
▷ 2.61e13 Bytes transferred, bytes (bytes)
▷ 74382043.11 Computational imbalance, imb
  22.21 Load balance efficiency, lb_eff (%)
  99.89 Communication efficiency, comm_eff
  22.19 Parallel efficiency, par_eff (%)
  0.46 IPC, ipc
  1.38 MIPS, mips
  4.56e16 PAPI_TOT_INS (#)
  2.46e17 PAPI_TOT_CYC (#)
```

```
0.00        1.55e8 (25.09%)        6.18e8
```

Call tree | Flat view

```
▽ 0.00 bt
  ▷ 0.00 mpi_setup
    0.00 MPI_Bcast
  ▷ 0.00 env_setup
    0.00 zone_setup
    0.02 map_zones
    0.02 zone_starts
    0.00 set_constants
  ▷ 0.03 initialize
  ▷ 0.02 exact_rhs
  ▷ 1.12 exch_qbc
  ▽ 0.00 adi
    ▷ 3.38 compute_rhs
    ▽ 0.00 x_solve
      ▽ 0.00 !$omp parallel @x_solve.input.prep.f:46
          5.70 !$omp do @x_solve.input.prep.f:54
          0.33 !$omp implicit barrier @x_solve.input.
    ▽ 0.00 y_solve
      ▽ 0.00 !$omp parallel @y_solve.input.prep.f:43
          5.91 !$omp do @y_solve.input.prep.f:52
          0.32 !$omp implicit barrier @y_solve.input.
    ▽ 0.00 z_solve
      ▽ 0.00 !$omp parallel @z_solve.input.prep.f:43
          6.14 !$omp do @z_solve.input.prep.f:52
          1.79 !$omp implicit barrier @z_solve.input.
    ▷ 0.28 add
    0.00 MPI_Barrier
  ▷ 0.02 verify
    0.00 MPI_Reduce
    0.00 print_results
    0.00 MPI_Finalize
```

```
0.00           24.99              100.00
0.00      1.54e8 (99.59%)         1.55e8
```

System tree | BoxPlot

All (1048576 elements)

```
0.00           0.00              100.00
0.00       0.00 (0.00%)          1.54e8
```

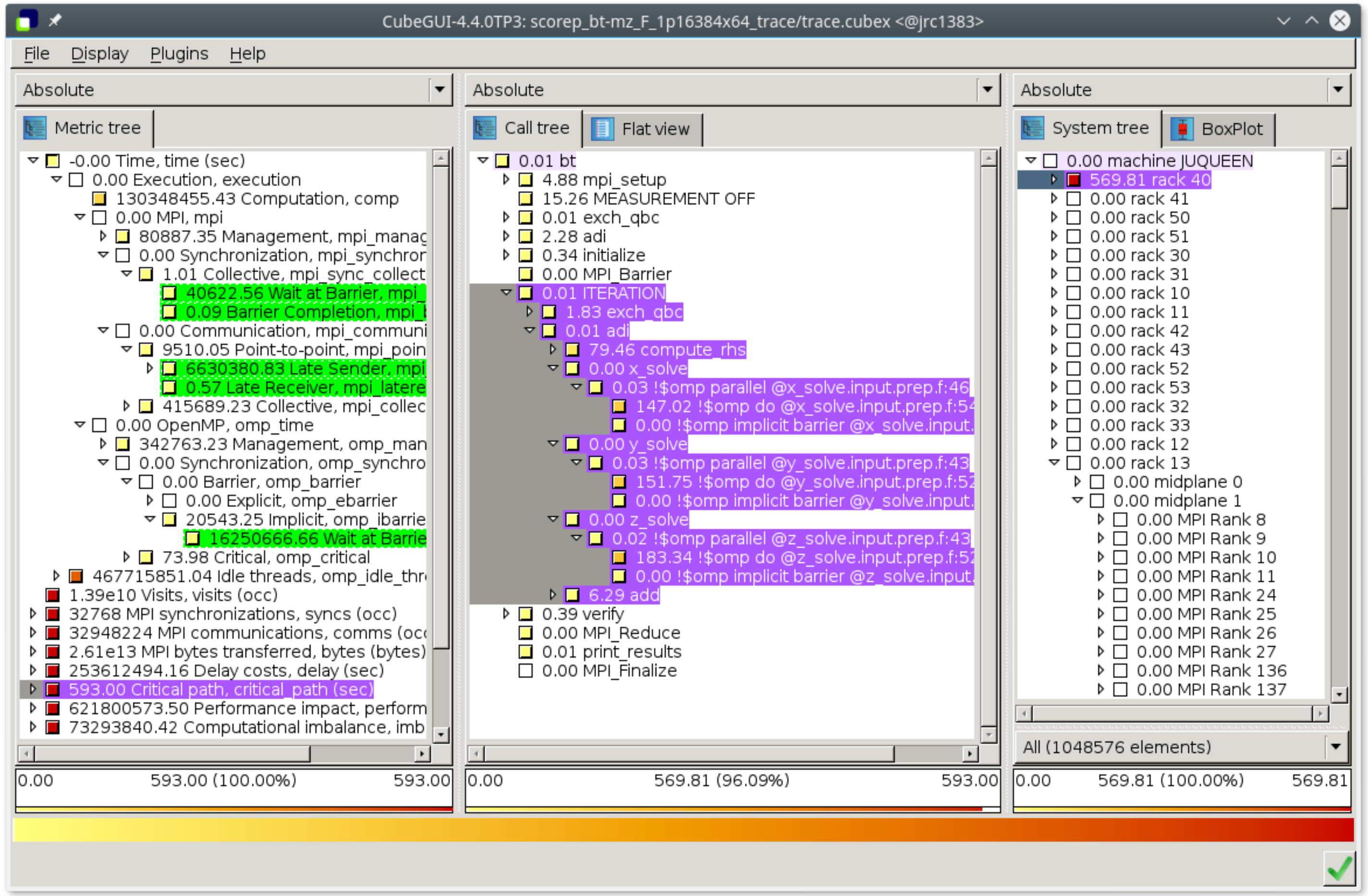
Ready

- BT-MZ class F (12032x8960x250): 1 MPI rank/node, 64 OpenMP threads/rank
- No load “balancing”
- 16,384 PowerPC A2 compute nodes (16 racks) of IBM Blue Gene/Q

- IBM XL compiler instrumentation with measurement filter
- Wall clock time 573 seconds (39.1 Mop/s total)
- 7% measurement dilation (including 2 hardware counters)

1M OpenMP threads in total, running on 16,384 nodes

BT-MZ.F Scalasca trace analysis: 1p16364x64 on *JUQUEEN* BG/Q


CubeGUI-4.4.0TP3: scorep_bt-mz_F_1p16384x64_trace/trace.cubex <@jrc1383>

File Display Plugins Help

```
Absolute                                   | Absolute                                           | Absolute
Metric tree                                | Call tree   Flat view                              | System tree   BoxPlot
-0.00 Time, time (sec)                     | 0.01 bt                                            | 0.00 machine JUQUEEN
  0.00 Execution, execution                |   4.88 mpi_setup                                   |   569.81 rack 40
    130348455.43 Computation, comp         |   15.26 MEASUREMENT OFF                            |   0.00 rack 41
    0.00 MPI, mpi                          |   0.01 exch_qbc                                    |   0.00 rack 50
      80887.35 Management, mpi_manag       |   2.28 adi                                         |   0.00 rack 51
      0.00 Synchronization, mpi_synchror   |   0.34 initialize                                  |   0.00 rack 30
        1.01 Collective, mpi_sync_collect  |   0.00 MPI_Barrier                                 |   0.00 rack 31
          40622.56 Wait at Barrier, mpi_   |   0.01 ITERATION                                   |   0.00 rack 10
          0.09 Barrier Completion, mpi_    |     1.83 exch_qbc                                  |   0.00 rack 11
      0.00 Communication, mpi_communi      |     0.01 adi                                       |   0.00 rack 42
        9510.05 Point-to-point, mpi_poin   |       79.46 compute_rhs                            |   0.00 rack 43
          6630380.83 Late Sender, mpi      |       0.00 x_solve                                 |   0.00 rack 52
          0.57 Late Receiver, mpi_latere   |         0.03 !$omp parallel @x_solve.input.prep.f:46|   0.00 rack 53
        415689.23 Collective, mpi_collec   |           147.02 !$omp do @x_solve.input.prep.f:5  |   0.00 rack 32
    0.00 OpenMP, omp_time                  |           0.00 !$omp implicit barrier @x_solve.input.| 0.00 rack 33
      342763.23 Management, omp_man        |       0.00 y_solve                                 |   0.00 rack 12
      0.00 Synchronization, omp_synchro    |         0.03 !$omp parallel @y_solve.input.prep.f:43|   0.00 rack 13
        0.00 Barrier, omp_barrier          |           151.75 !$omp do @y_solve.input.prep.f:5  |     0.00 midplane 0
          0.00 Explicit, omp_ebarrier      |           0.00 !$omp implicit barrier @y_solve.input.|   0.00 midplane 1
          20543.25 Implicit, omp_ibarrie   |       0.00 z_solve                                 |       0.00 MPI Rank 8
            16250666.66 Wait at Barrie     |         0.02 !$omp parallel @z_solve.input.prep.f:43|       0.00 MPI Rank 9
        73.98 Critical, omp_critical       |           183.34 !$omp do @z_solve.input.prep.f:5  |       0.00 MPI Rank 10
    467715851.04 Idle threads, omp_idle_thr|           0.00 !$omp implicit barrier @z_solve.input.|     0.00 MPI Rank 11
1.39e10 Visits, visits (occ)               |       6.29 add                                     |       0.00 MPI Rank 24
32768 MPI synchronizations, syncs (occ)    |   0.39 verify                                      |       0.00 MPI Rank 25
32948224 MPI communications, comms (occ    |   0.00 MPI_Reduce                                  |       0.00 MPI Rank 26
2.61e13 MPI bytes transferred, bytes (bytes)| 0.01 print_results                                |       0.00 MPI Rank 27
253612494.16 Delay costs, delay (sec)      |   0.00 MPI_Finalize                                |       0.00 MPI Rank 136
593.00 Critical path, critical_path (sec)  |                                                    |       0.00 MPI Rank 137
621800573.50 Performance impact, perform   |                                                    |
73293840.42 Computational imbalance, imb   |                                                    | All (1048576 elements)
0.00   593.00 (100.00%)   593.00           | 0.00   569.81 (96.09%)   593.00                    | 0.00   569.81 (100.00%)   569.81
```

- BT-MZ class F (12032x8960x250):
  1 MPI rank/node,
  64 OpenMP threads/rank
- No load “balancing”
- 16,384 PowerPC A2 compute nodes (16 racks) of IBM Blue Gene/Q

- 0.5 TiB event data written in 3.3s (using one SIONlib file per IONode)
- Scalasca automatic trace analysis
  - distinguishes waiting times for comm&synch operations as 3.7% of total CPU time, resulting in 75% idle threads
  - quantifies callpath contributions to critical path of execution (all on MPI rank 0)

Outline

- Case I:
  - NPB3.3_MZ_MPI/**BT-MZ** (MPI+OpenMP) on *MARCONI-KNL*: load balancing
- Case II:
  - **k-Wave** (MPI+OpenMP) on *Salomon*: load-balancing in FFTW OpenMP parallel regions
- Case III:
  - **ICON** (MPI) on *Mistral*: automatic trace analysis of critical path of execution
- Case IV:
  - **PIConGPU** (MPI+CUDA): computation offload to multiple attached accelerator devices
- Case V:
  - **TensorFlow** (Python+CUDA): interpreted & compiled heterogeneous execution measurement

Case II: k-Wave: load balancing in FFTW OpenMP parallel regions

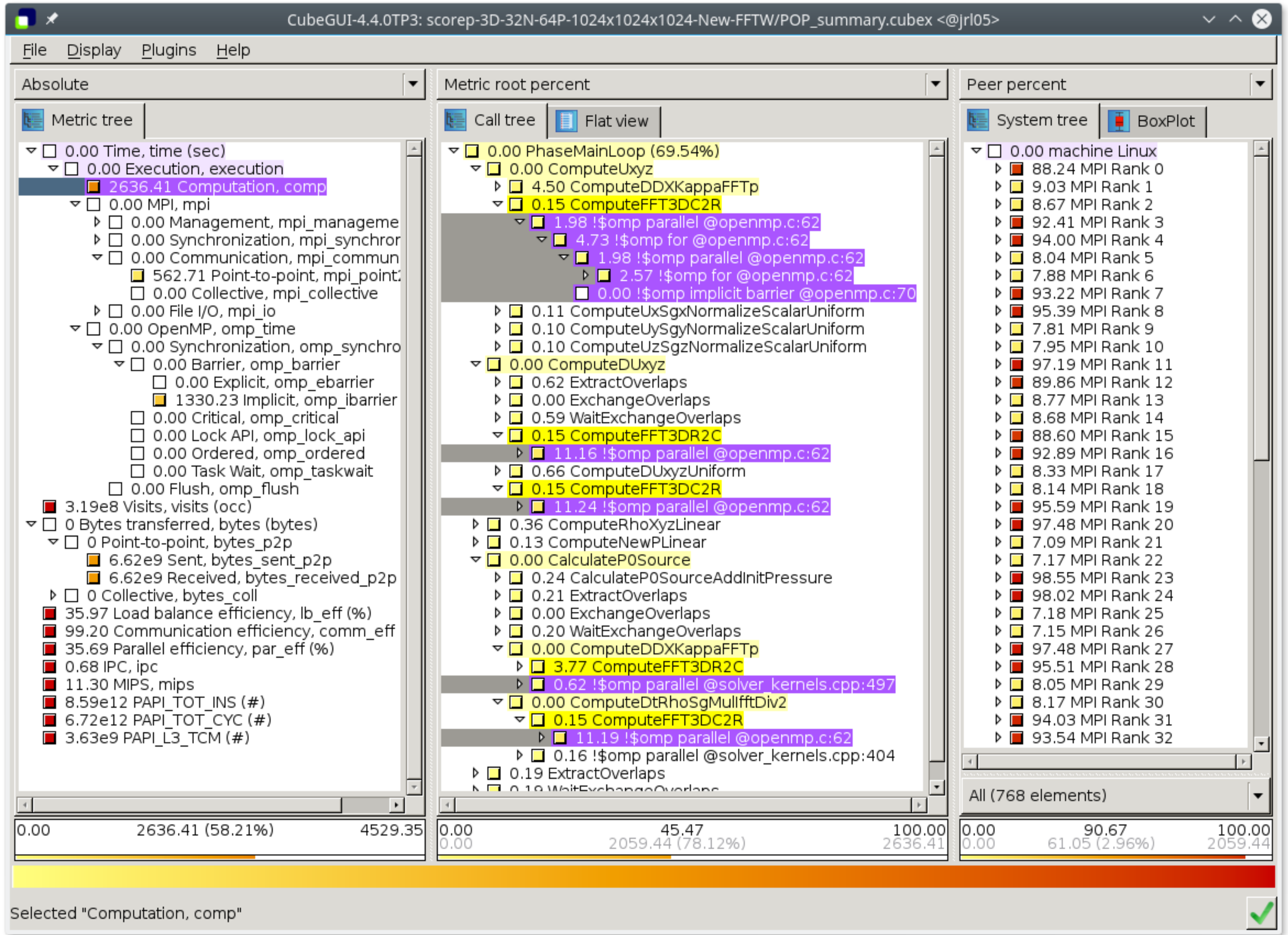
- Toolbox for time-domain acoustic and ultrasound simulations in complex and tissue-realistic media, developed by Brno University of Technology (CZ)
- C++ code parallelized with MPI and OpenMP [+ CUDA unused]
  - FFTW library using OpenMP parallelization; parallel HDF5 file I/O
  - GCC compiler and OpenMPI library
- Executed on *Salomon* Intel Xeon compute nodes (IT4Innovations/CZ)
  - 64 MPI processes (2 per compute node), 12 OpenMP threads per process
  - Score-P runtime measurement filter used to eliminate FFTW computation routines

www.k-wave.org

- 3D domain decomposition (1024^3 on 4x4x4 processes) suffered major load imbalance
  - exterior MPI processes with fewer grid cells took 4x longer than interior
  - OpenMP-parallelized FFTs were much less efficient for (smaller) grid dimensions of exterior, requiring many more small and poorly-balanced nested parallel loops
- Revised to use a periodic domain with identical (padded) halo zones for each MPI rank

  ⇒ ***improved kernel by a factor of 6 and overall execution time by a factor of 2***

k-Wave summary profile (initial version): parallel region imbalance



CubeGUI-4.4.0TP3: scorep-3D-32N-64P-1024x1024x1024-New-FFTW/POP_summary.cubex <@jrl05>

File Display Plugins Help

Absolute | Metric root percent | Peer percent

Metric tree | Call tree | Flat view | System tree | BoxPlot

```
0.00 Time, time (sec)
  0.00 Execution, execution
    2636.41 Computation, comp
    0.00 MPI, mpi
      0.00 Management, mpi_manageme
      0.00 Synchronization, mpi_synchror
      0.00 Communication, mpi_commun
        562.71 Point-to-point, mpi_point:
        0.00 Collective, mpi_collective
      0.00 File I/O, mpi_io
    0.00 OpenMP, omp_time
      0.00 Synchronization, omp_synchro
        0.00 Barrier, omp_barrier
          0.00 Explicit, omp_ebarrier
          1330.23 Implicit, omp_ibarrier
        0.00 Critical, omp_critical
        0.00 Lock API, omp_lock_api
        0.00 Ordered, omp_ordered
        0.00 Task Wait, omp_taskwait
      0.00 Flush, omp_flush
3.19e8 Visits, visits (occ)
0 Bytes transferred, bytes (bytes)
  0 Point-to-point, bytes_p2p
    6.62e9 Sent, bytes_sent_p2p
    6.62e9 Received, bytes_received_p2p
  0 Collective, bytes_coll
35.97 Load balance efficiency, lb_eff (%)
99.20 Communication efficiency, comm_eff
35.69 Parallel efficiency, par_eff (%)
0.68 IPC, ipc
11.30 MIPS, mips
8.59e12 PAPI_TOT_INS (#)
6.72e12 PAPI_TOT_CYC (#)
3.63e9 PAPI_L3_TCM (#)
```

```
0.00 PhaseMainLoop (69.54%)
  0.00 ComputeUxyz
    4.50 ComputeDDXKappaFFTp
    0.15 ComputeFFT3DC2R
      1.98 !$omp parallel @openmp.c:62
        4.73 !$omp for @openmp.c:62
          1.98 !$omp parallel @openmp.c:62
            2.57 !$omp for @openmp.c:62
            0.00 !$omp implicit barrier @openmp.c:70
    0.11 ComputeUxSgxNormalizeScalarUniform
    0.10 ComputeUySgyNormalizeScalarUniform
    0.10 ComputeUzSgzNormalizeScalarUniform
  0.00 ComputeDUxyz
    0.62 ExtractOverlaps
    0.00 ExchangeOverlaps
    0.59 WaitExchangeOverlaps
    0.15 ComputeFFT3DR2C
      11.16 !$omp parallel @openmp.c:62
    0.66 ComputeDUxyzUniform
    0.15 ComputeFFT3DC2R
      11.24 !$omp parallel @openmp.c:62
  0.36 ComputeRhoXyzLinear
  0.13 ComputeNewPLinear
  0.00 CalculateP0Source
    0.24 CalculateP0SourceAddInitPressure
    0.21 ExtractOverlaps
    0.00 ExchangeOverlaps
    0.20 WaitExchangeOverlaps
    0.00 ComputeDDXKappaFFTp
      3.77 ComputeFFT3DR2C
      0.62 !$omp parallel @solver_kernels.cpp:497
    0.00 ComputeDtRhoSgMulIfftDiv2
      0.15 ComputeFFT3DC2R
        11.19 !$omp parallel @openmp.c:62
      0.16 !$omp parallel @solver_kernels.cpp:404
  0.19 ExtractOverlaps
  0.19 WaitExchangeOverlaps
```

```
0.00 machine Linux
  88.24 MPI Rank 0
  9.03 MPI Rank 1
  8.67 MPI Rank 2
  92.41 MPI Rank 3
  94.00 MPI Rank 4
  8.04 MPI Rank 5
  7.88 MPI Rank 6
  93.22 MPI Rank 7
  95.39 MPI Rank 8
  7.81 MPI Rank 9
  7.95 MPI Rank 10
  97.19 MPI Rank 11
  89.86 MPI Rank 12
  8.77 MPI Rank 13
  8.68 MPI Rank 14
  88.60 MPI Rank 15
  92.89 MPI Rank 16
  8.33 MPI Rank 17
  8.14 MPI Rank 18
  95.59 MPI Rank 19
  97.48 MPI Rank 20
  7.09 MPI Rank 21
  7.17 MPI Rank 22
  98.55 MPI Rank 23
  98.02 MPI Rank 24
  7.18 MPI Rank 25
  7.15 MPI Rank 26
  97.48 MPI Rank 27
  95.51 MPI Rank 28
  8.05 MPI Rank 29
  8.17 MPI Rank 30
  94.03 MPI Rank 31
  93.54 MPI Rank 32
```

All (768 elements)

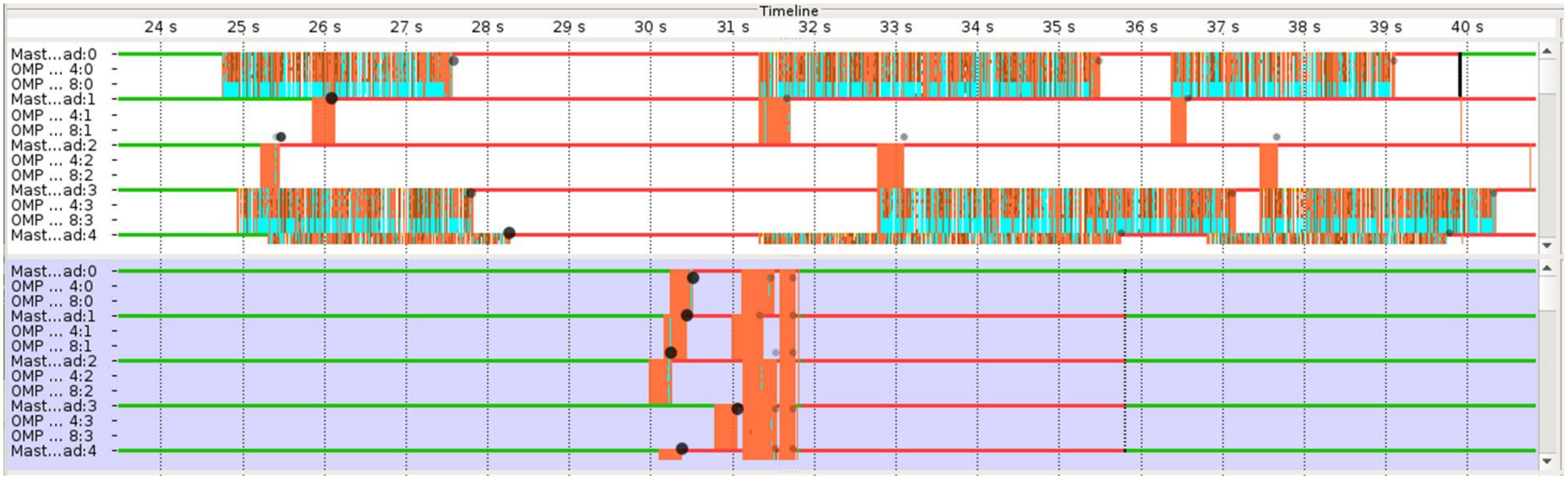
```
0.00      2636.41 (58.21%)      4529.35
0.00          45.47             100.00
0.00      2059.44 (78.12%)     2636.41
0.00          90.67             100.00
0.00      61.05 (2.96%)        2059.44
```

Selected "Computation, comp"

PhaseMainLoop routine extract

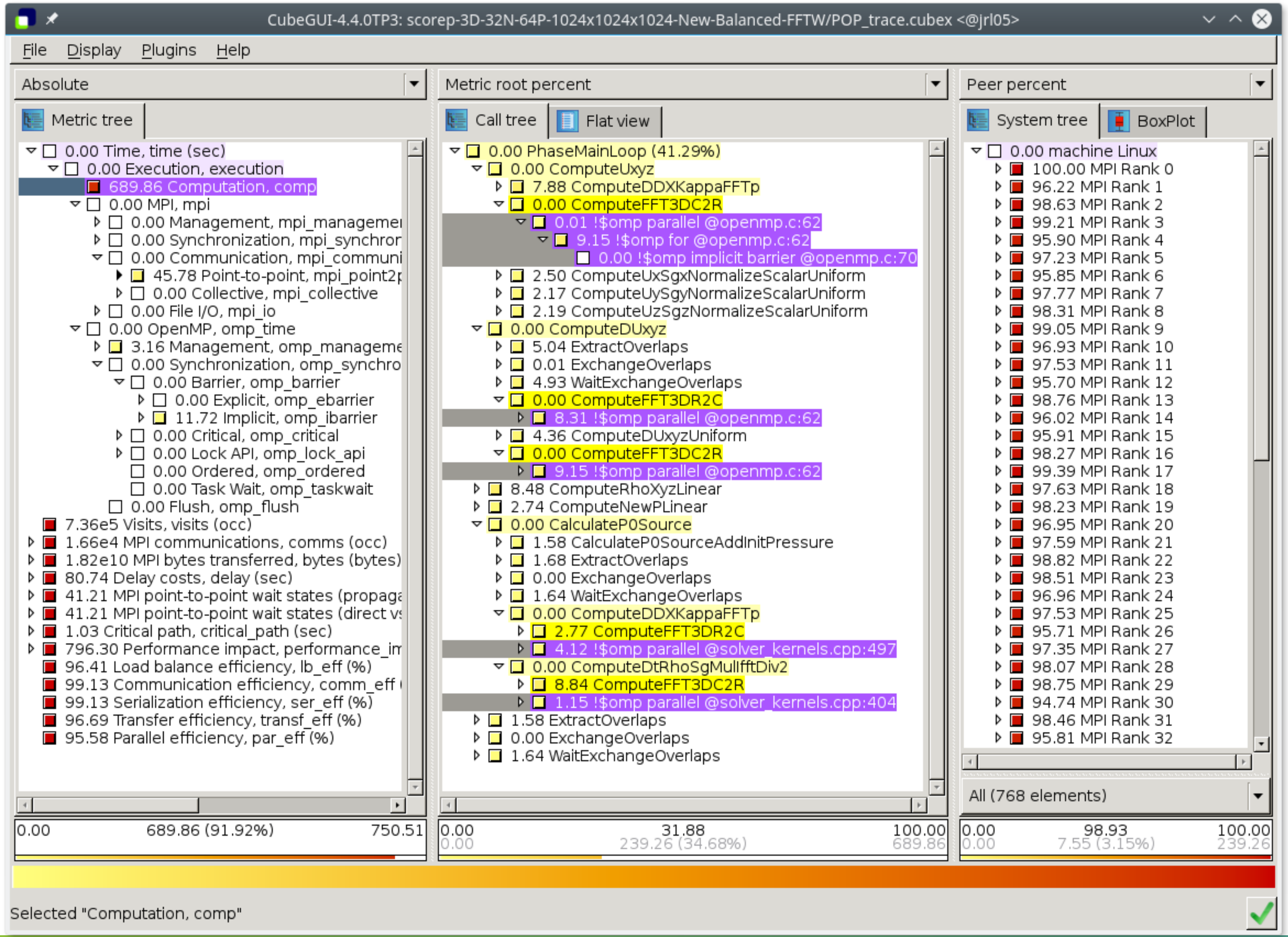
- 58% Computation time
  - 29% OMP + 12% MPI overheads
- 50% of Computation time in five ComputeFFT3D routines each with OpenMP parallel regions
  - with nested parallel regions inside
- Huge computation imbalance
  - half of the MPI ranks (1,2,5,6,9,…) ten times faster than the others
  - results in huge amounts of OpenMP implicit barrier synchronization time at end of parallel regions
- Only 35% parallel efficiency

k-Wave Vampir trace time-line comparison (original & revised)



- executions before [upper] and after [lower] balancing grid-points per MPI process
- showing processes for corner ranks (0&3) and edge ranks (1&2) of 4x4x4 geometry
- MPI synchronization in red, OpenMP synchronization in cyan

k-Wave summary profile (revised version): balanced parallel regions



```
CubeGUI-4.4.0TP3: scorep-3D-32N-64P-1024x1024x1024-New-Balanced-FFTW/POP_trace.cubex <@jrl05>
File  Display  Plugins  Help

Absolute                                   | Metric root percent                                  | Peer percent
Metric tree                                | Call tree   Flat view                                | System tree   BoxPlot
0.00 Time, time (sec)                      | 0.00 PhaseMainLoop (41.29%)                          | 0.00 machine Linux
  0.00 Execution, execution                |   0.00 ComputeUxyz                                   |   100.00 MPI Rank 0
    689.86 Computation, comp               |     7.88 ComputeDDXKappaFFTp                         |   96.22 MPI Rank 1
    0.00 MPI, mpi                          |     0.00 ComputeFFT3DC2R                             |   98.63 MPI Rank 2
      0.00 Management, mpi_manageme        |       0.01 !$omp parallel @openmp.c:62               |   99.21 MPI Rank 3
      0.00 Synchronization, mpi_synchror   |         9.15 !$omp for @openmp.c:62                  |   95.90 MPI Rank 4
      0.00 Communication, mpi_communi      |           0.00 !$omp implicit barrier @openmp.c:70   |   97.23 MPI Rank 5
        45.78 Point-to-point, mpi_point2p  |     2.50 ComputeUxSgxNormalizeScalarUniform          |   95.85 MPI Rank 6
        0.00 Collective, mpi_collective    |     2.17 ComputeUySgyNormalizeScalarUniform          |   97.77 MPI Rank 7
      0.00 File I/O, mpi_io                |     2.19 ComputeUzSgzNormalizeScalarUniform          |   98.31 MPI Rank 8
    0.00 OpenMP, omp_time                  |   0.00 ComputeDUxyz                                  |   99.05 MPI Rank 9
      3.16 Management, omp_manageme        |     5.04 ExtractOverlaps                             |   96.93 MPI Rank 10
      0.00 Synchronization, omp_synchro    |     0.01 ExchangeOverlaps                            |   97.53 MPI Rank 11
        0.00 Barrier, omp_barrier          |     4.93 WaitExchangeOverlaps                        |   95.70 MPI Rank 12
          0.00 Explicit, omp_ebarrier      |     0.00 ComputeFFT3DR2C                             |   98.76 MPI Rank 13
          11.72 Implicit, omp_ibarrier     |       8.31 !$omp parallel @openmp.c:62               |   96.02 MPI Rank 14
        0.00 Critical, omp_critical        |     4.36 ComputeDUxyzUniform                         |   95.91 MPI Rank 15
        0.00 Lock API, omp_lock_api        |     0.00 ComputeFFT3DC2R                             |   98.27 MPI Rank 16
        0.00 Ordered, omp_ordered          |       9.15 !$omp parallel @openmp.c:62               |   99.39 MPI Rank 17
        0.00 Task Wait, omp_taskwait       |   8.48 ComputeRhoXyzLinear                           |   97.63 MPI Rank 18
      0.00 Flush, omp_flush                |   2.74 ComputeNewPLinear                             |   98.23 MPI Rank 19
7.36e5 Visits, visits (occ)                |   0.00 CalculateP0Source                             |   96.95 MPI Rank 20
1.66e4 MPI communications, comms (occ)     |     1.58 CalculateP0SourceAddInitPressure            |   97.59 MPI Rank 21
1.82e10 MPI bytes transferred, bytes (bytes)|    1.68 ExtractOverlaps                             |   98.82 MPI Rank 22
80.74 Delay costs, delay (sec)             |     0.00 ExchangeOverlaps                            |   98.51 MPI Rank 23
41.21 MPI point-to-point wait states (propaga|   1.64 WaitExchangeOverlaps                          |   96.96 MPI Rank 24
41.21 MPI point-to-point wait states (direct vs| 0.00 ComputeDDXKappaFFTp                          |   97.53 MPI Rank 25
1.03 Critical path, critical_path (sec)    |       2.77 ComputeFFT3DR2C                           |   95.71 MPI Rank 26
796.30 Performance impact, performance_im  |       4.12 !$omp parallel @solver_kernels.cpp:497    |   97.35 MPI Rank 27
96.41 Load balance efficiency, lb_eff (%)  |     0.00 ComputeDtRhoSgMullfftDiv2                   |   98.07 MPI Rank 28
99.13 Communication efficiency, comm_eff ( |       8.84 ComputeFFT3DC2R                           |   98.75 MPI Rank 29
99.13 Serialization efficiency, ser_eff (%)|       1.15 !$omp parallel @solver_kernels.cpp:404    |   94.74 MPI Rank 30
96.69 Transfer efficiency, transf_eff (%)  |   1.58 ExtractOverlaps                               |   98.46 MPI Rank 31
95.58 Parallel efficiency, par_eff (%)     |   0.00 ExchangeOverlaps                              |   95.81 MPI Rank 32
                                           |   1.64 WaitExchangeOverlaps                          |
                                           |                                                      | All (768 elements)
0.00      689.86 (91.92%)      750.51      | 0.00        31.88               100.00               | 0.00      98.93          100.00
                                           | 0.00        239.26 (34.68%)     689.86               | 0.00      7.55 (3.15%)   239.26

Selected "Computation, comp"
```

- **PhaseMainLoop** routine
  Execution time reduced 6-fold from 4530 to 750 seconds
- Now 92% Computation time
  - 2% OMP + 6% MPI overhead
- 32% of Computation time now in ComputeFFT3D routines
  - simplified OpenMP parallel regions no longer nested
- Greatly improved load balance
  - 1.4% standard deviation
- Over 95% parallel efficiency

Outline

- Case I:
  - NPB3.3_MZ_MPI/**BT-MZ** (MPI+OpenMP) on *MARCONI-KNL*: load balancing
- Case II:
  - **k-Wave** (MPI+OpenMP) on *Salomon*: load-balancing in FFTW OpenMP parallel regions
- Case III:
  - **ICON** (MPI) on *Mistral*: automatic trace analysis of critical path of execution
- Case IV:
  - **PIConGPU** (MPI+CUDA): computation offload to multiple attached accelerator devices
- Case V:
  - **TensorFlow** (Python+CUDA): interpreted & compiled heterogeneous execution measurement

Case III: ICON

- Icosahedral non-hydrostatic unified weather forecasting and climate model jointly developed by
  - Max Planck Institute for Meteorology (MPI-M)
  - Germany's National Meteorological Service (DWD)
- ICON source code and test case provided by H. Bockelmann (DKRZ)
  - Mostly Fortran 90, some C; parallelized with MPI [+ OpenMP unused]
  - Intel compiler and bullx MPI library
  - 24-hour physical simulation in 10 min increments
- Executed on *Mistral* Intel Xeon compute nodes (DKRZ):
  - 2x 12-core Intel Xeon E5-2680 v3 (Haswell) @ 2.5GHz
  - 24 MPI processes/node, experiments with 4/8/16/32 compute nodes

⇒ ***Identification & quantification of impact of periodic additional computations***

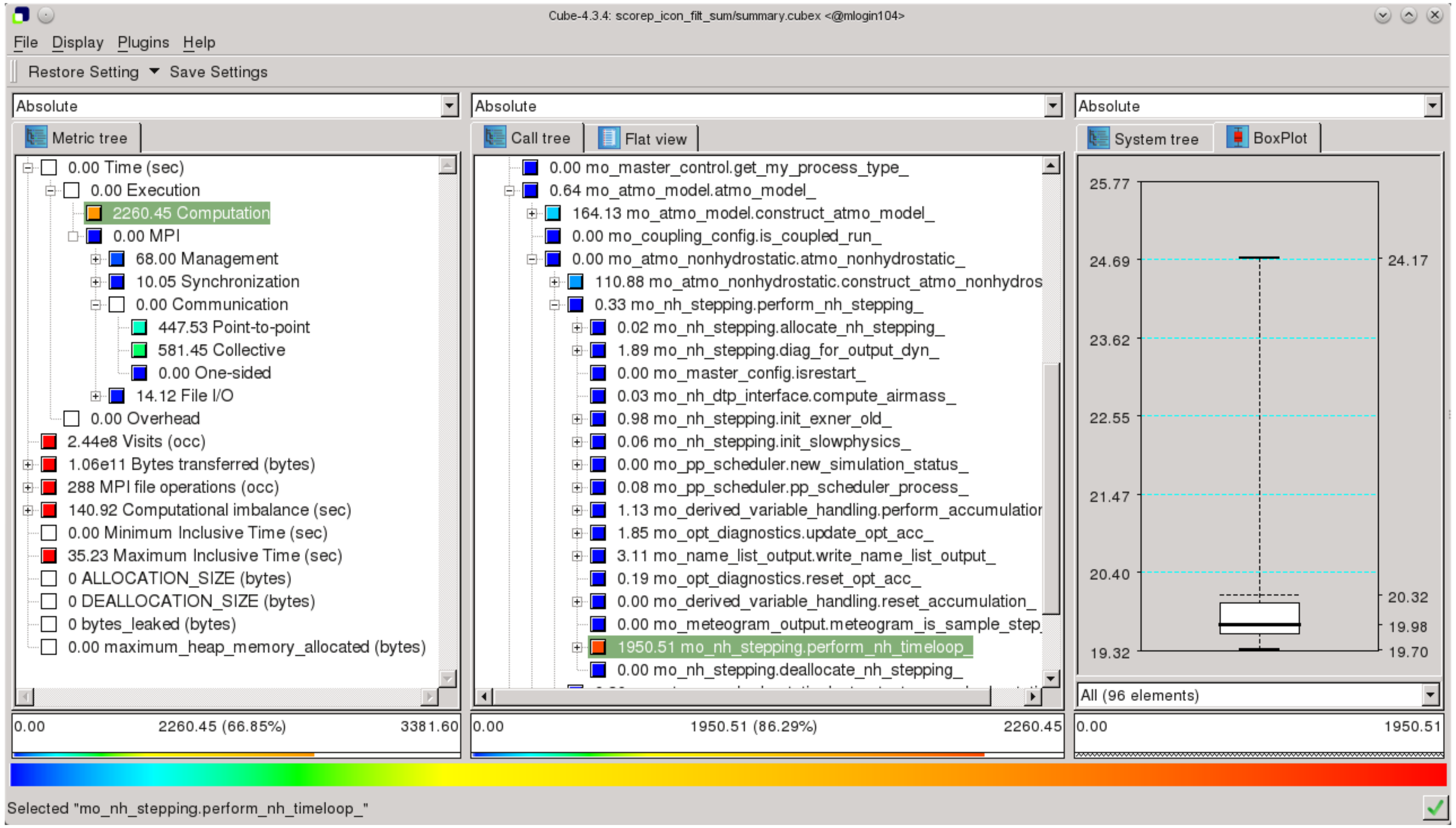
ICON instrumentation

- After configuration, adjusted compiler variables in top-level Makefile
  - Code parts written in C **not** instrumented

```
CC  = icc
FC  = scorep --user --mpp=mpi --thread=none ifort
F77 = scorep --user --mpp=mpi --thread=none ifort
```

- Initial instrumented run incurred ~120% overhead
- Preparing a good filter required several iterations
  - Filter out enough routines to achieve reasonable overhead…
  - …but not loose important information
- Overhead of filtered run <20%
  - Not perfect, but OK

ICON summary profile analysis (4 compute nodes)


Cube-4.3.4: scorep_icon_filt_sum/summary.cubex <@mlogin104>

File Display Plugins Help

Restore Setting Save Settings

Absolute | Absolute | Absolute

Metric tree | Call tree | Flat view | System tree | BoxPlot

```
0.00 Time (sec)
  0.00 Execution
    2260.45 Computation
    0.00 MPI
      68.00 Management
      10.05 Synchronization
      0.00 Communication
        447.53 Point-to-point
        581.45 Collective
        0.00 One-sided
      14.12 File I/O
  0.00 Overhead
2.44e8 Visits (occ)
1.06e11 Bytes transferred (bytes)
288 MPI file operations (occ)
140.92 Computational imbalance (sec)
0.00 Minimum Inclusive Time (sec)
35.23 Maximum Inclusive Time (sec)
0 ALLOCATION_SIZE (bytes)
0 DEALLOCATION_SIZE (bytes)
0 bytes_leaked (bytes)
0.00 maximum_heap_memory_allocated (bytes)
```

```
0.00 mo_master_control.get_my_process_type_
0.64 mo_atmo_model.atmo_model_
  164.13 mo_atmo_model.construct_atmo_model_
  0.00 mo_coupling_config.is_coupled_run_
  0.00 mo_atmo_nonhydrostatic.atmo_nonhydrostatic_
    110.88 mo_atmo_nonhydrostatic.construct_atmo_nonhydros
    0.33 mo_nh_stepping.perform_nh_stepping_
      0.02 mo_nh_stepping.allocate_nh_stepping_
      1.89 mo_nh_stepping.diag_for_output_dyn_
      0.00 mo_master_config.isrestart_
      0.03 mo_nh_dtp_interface.compute_airmass_
      0.98 mo_nh_stepping.init_exner_old_
      0.06 mo_nh_stepping.init_slowphysics_
      0.00 mo_pp_scheduler.new_simulation_status_
      0.08 mo_pp_scheduler.pp_scheduler_process_
      1.13 mo_derived_variable_handling.perform_accumulatior
      1.85 mo_opt_diagnostics.update_opt_acc_
      3.11 mo_name_list_output.write_name_list_output_
      0.19 mo_opt_diagnostics.reset_opt_acc_
      0.00 mo_derived_variable_handling.reset_accumulation_
      0.00 mo_meteogram_output.meteogram_is_sample_step
      1950.51 mo_nh_stepping.perform_nh_timeloop_
      0.00 mo_nh_stepping.deallocate_nh_stepping_
```

All (96 elements)

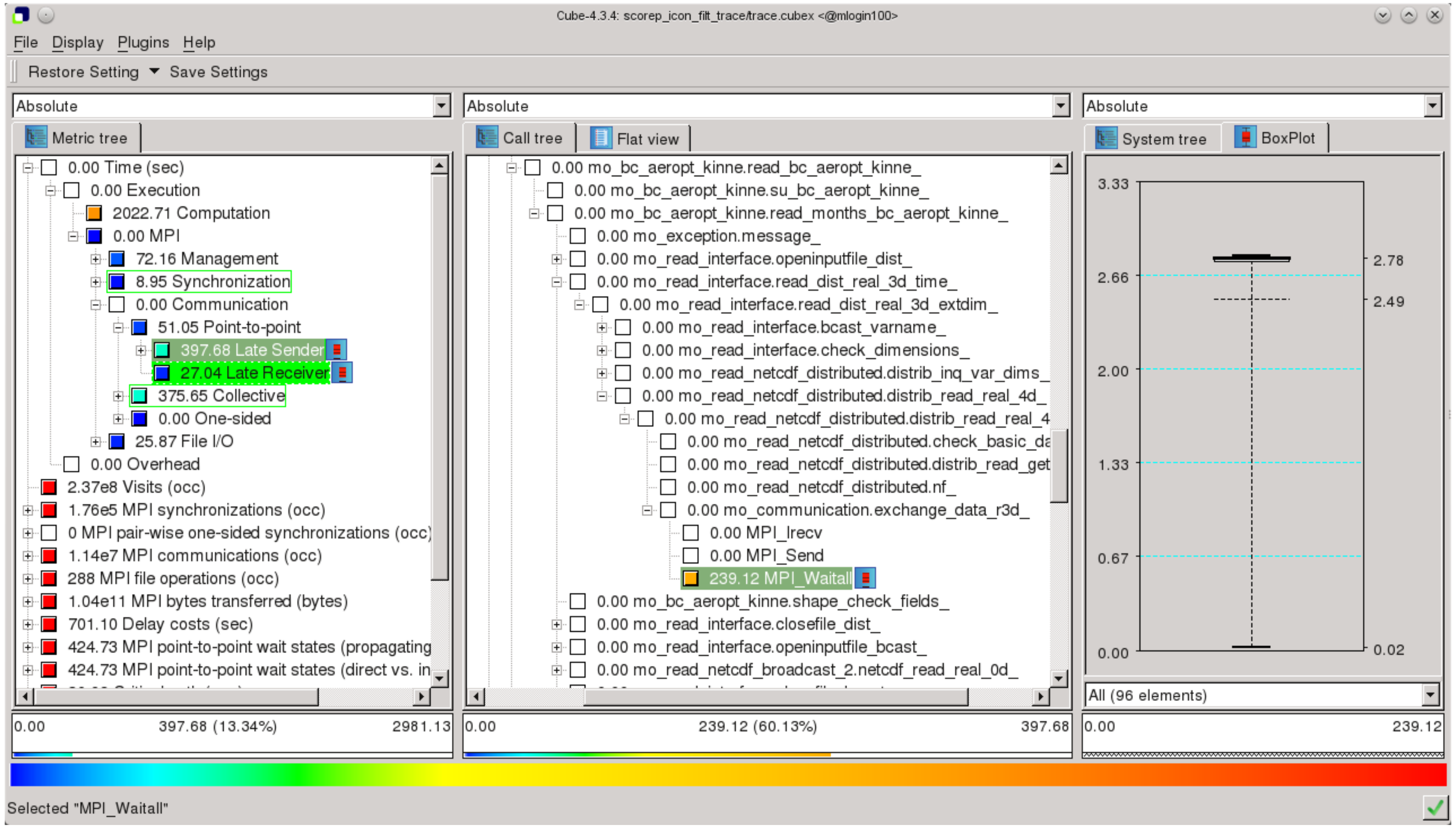
0.00	2260.45 (66.85%)	3381.60
0.00	1950.51 (86.29%)	2260.45
0.00		1950.51

Selected "mo_nh_stepping.perform_nh_timeloop_"

- 67% of time is computation
- >30% is MPI communication

- 87% of computation is spent in timestep loop
  - …with quite some variation across ranks

ICON Scalasca trace analysis (4 compute nodes)


Cube-4.3.4: scorep_icon_filt_trace/trace.cubex <@mlogin100>

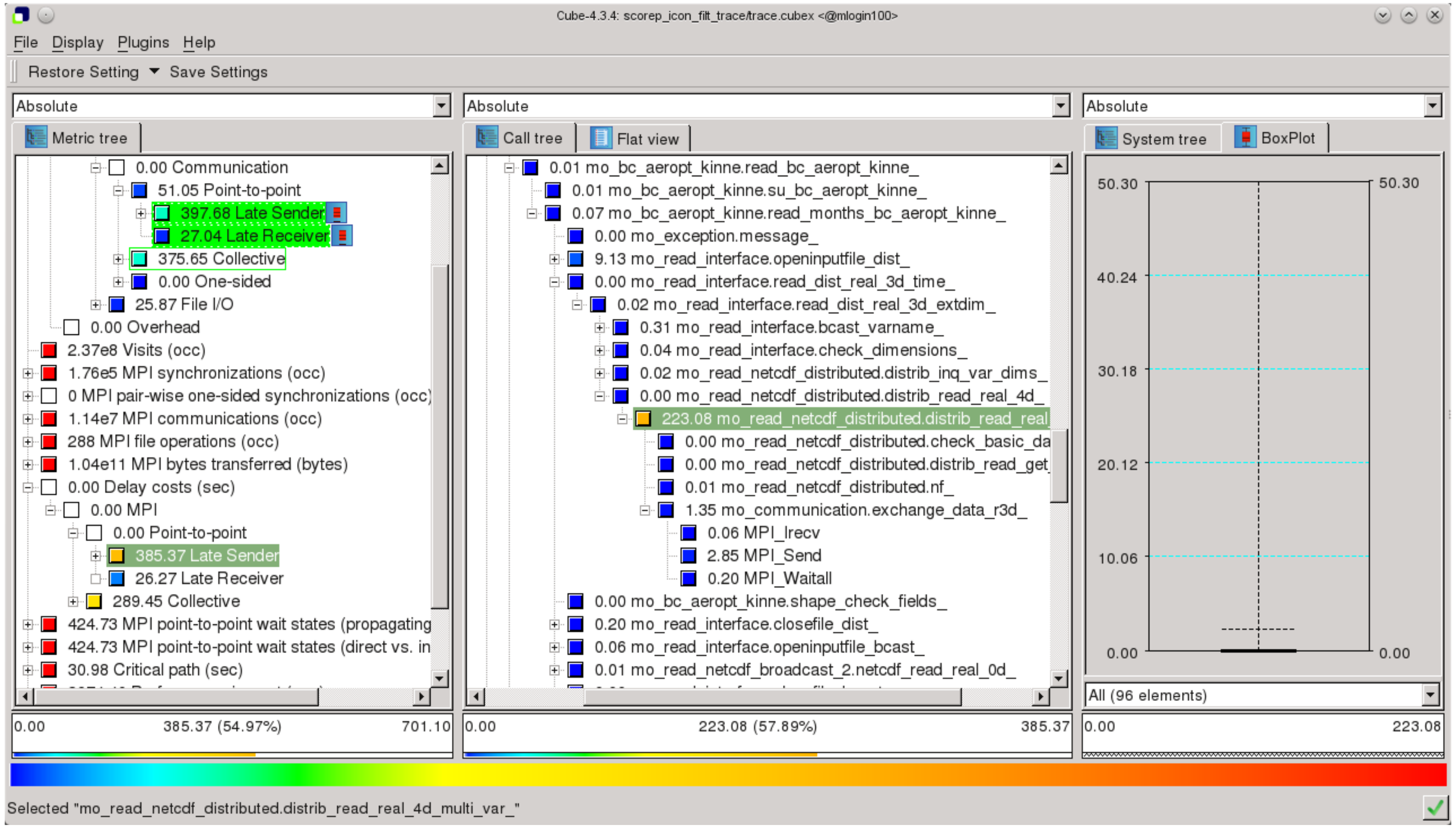
File Display Plugins Help

Restore Setting Save Settings

```
Absolute                                      Absolute                                                   Absolute
Metric tree                                   Call tree | Flat view                                      System tree | BoxPlot

0.00 Time (sec)                               0.00 mo_bc_aeropt_kinne.read_bc_aeropt_kinne_
  0.00 Execution                                0.00 mo_bc_aeropt_kinne.su_bc_aeropt_kinne_
    2022.71 Computation                         0.00 mo_bc_aeropt_kinne.read_months_bc_aeropt_kinne_
    0.00 MPI                                      0.00 mo_exception.message_
      72.16 Management                            0.00 mo_read_interface.openinputfile_dist_
      8.95 Synchronization                        0.00 mo_read_interface.read_dist_real_3d_time_
      0.00 Communication                            0.00 mo_read_interface.read_dist_real_3d_extdim_
        51.05 Point-to-point                          0.00 mo_read_interface.bcast_varname_
          397.68 Late Sender                          0.00 mo_read_interface.check_dimensions_
          27.04 Late Receiver                         0.00 mo_read_netcdf_distributed.distrib_inq_var_dims_
        375.65 Collective                             0.00 mo_read_netcdf_distributed.distrib_read_real_4d_
        0.00 One-sided                                  0.00 mo_read_netcdf_distributed.distrib_read_real_4
      25.87 File I/O                                      0.00 mo_read_netcdf_distributed.check_basic_da
  0.00 Overhead                                           0.00 mo_read_netcdf_distributed.distrib_read_get
2.37e8 Visits (occ)                                       0.00 mo_read_netcdf_distributed.nf_
1.76e5 MPI synchronizations (occ)                         0.00 mo_communication.exchange_data_r3d_
0 MPI pair-wise one-sided synchronizations (occ)            0.00 MPI_Irecv
1.14e7 MPI communications (occ)                             0.00 MPI_Send
288 MPI file operations (occ)                               239.12 MPI_Waitall
1.04e11 MPI bytes transferred (bytes)             0.00 mo_bc_aeropt_kinne.shape_check_fields_
701.10 Delay costs (sec)                          0.00 mo_read_interface.closefile_dist_
424.73 MPI point-to-point wait states (propagating  0.00 mo_read_interface.openinputfile_bcast_
424.73 MPI point-to-point wait states (direct vs. in 0.00 mo_read_netcdf_broadcast_2.netcdf_read_real_0d_

                                                                                                         All (96 elements)
0.00     397.68 (13.34%)     2981.13          0.00       239.12 (60.13%)       397.68                     0.00        239.12

Selected "MPI_Waitall"
```

- Significant amount of *Late Sender* wait states
  - 13% of overall time
- 8% in **`MPI_Waitall`** called from **`exchange_data_r3d`**

ICON Scalasca trace analysis (cont.)


Cube-4.3.4: scorep_icon_filt_trace/trace.cubex <@mlogin100>

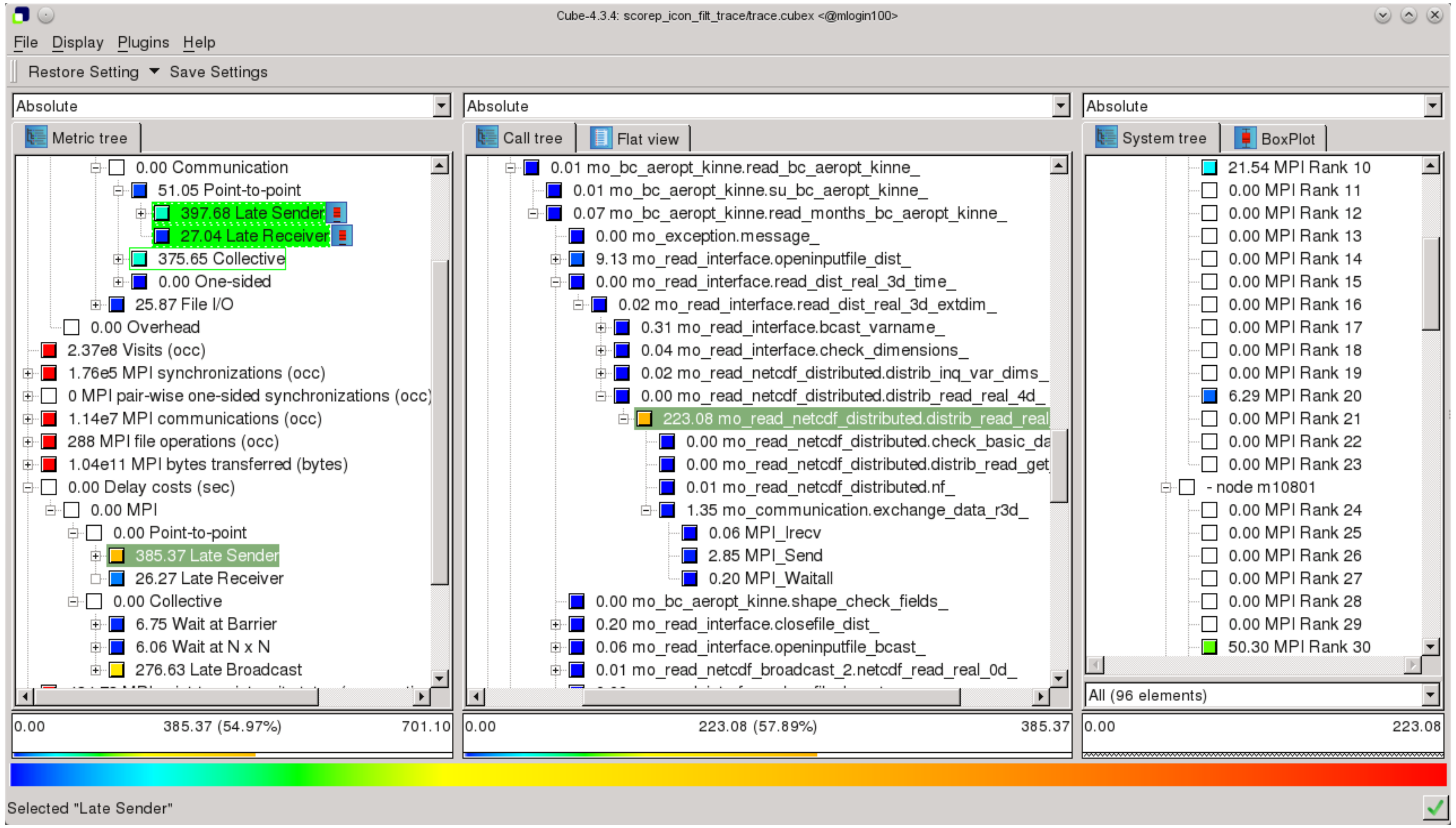
```
Metric tree:
      0.00 Communication
        51.05 Point-to-point
          397.68 Late Sender
          27.04 Late Receiver
        375.65 Collective
        0.00 One-sided
      25.87 File I/O
    0.00 Overhead
  2.37e8 Visits (occ)
  1.76e5 MPI synchronizations (occ)
  0 MPI pair-wise one-sided synchronizations (occ)
  1.14e7 MPI communications (occ)
  288 MPI file operations (occ)
  1.04e11 MPI bytes transferred (bytes)
  0.00 Delay costs (sec)
    0.00 MPI
      0.00 Point-to-point
        385.37 Late Sender
        26.27 Late Receiver
      289.45 Collective
  424.73 MPI point-to-point wait states (propagating
  424.73 MPI point-to-point wait states (direct vs. in
  30.98 Critical path (sec)

Call tree:
0.01 mo_bc_aeropt_kinne.read_bc_aeropt_kinne_
  0.01 mo_bc_aeropt_kinne.su_bc_aeropt_kinne_
  0.07 mo_bc_aeropt_kinne.read_months_bc_aeropt_kinne_
    0.00 mo_exception.message_
    9.13 mo_read_interface.openinputfile_dist_
    0.00 mo_read_interface.read_dist_real_3d_time_
      0.02 mo_read_interface.read_dist_real_3d_extdim_
        0.31 mo_read_interface.bcast_varname_
        0.04 mo_read_interface.check_dimensions_
        0.02 mo_read_netcdf_distributed.distrib_inq_var_dims_
        0.00 mo_read_netcdf_distributed.distrib_read_real_4d_
          223.08 mo_read_netcdf_distributed.distrib_read_real
            0.00 mo_read_netcdf_distributed.check_basic_da
            0.00 mo_read_netcdf_distributed.distrib_read_get
            0.01 mo_read_netcdf_distributed.nf_
            1.35 mo_communication.exchange_data_r3d_
              0.06 MPI_Irecv
              2.85 MPI_Send
              0.20 MPI_Waitall
    0.00 mo_bc_aeropt_kinne.shape_check_fields_
    0.20 mo_read_interface.closefile_dist_
    0.06 mo_read_interface.openinputfile_bcast_
    0.01 mo_read_netcdf_broadcast_2.netcdf_read_real_0d_
```

```
0.00      385.37 (54.97%)      701.10
0.00      223.08 (57.89%)      385.37
0.00                           223.08
```

Selected "mo_read_netcdf_distributed.distrib_read_real_4d_multi_var_"

- …mostly due to imbalance in netCDF `distrib_read_real_4d_multi_var` …

ICON Scalasca trace analysis (cont.)



Cube-4.3.4: scorep_icon_filt_trace/trace.cubex <@mlogin100>

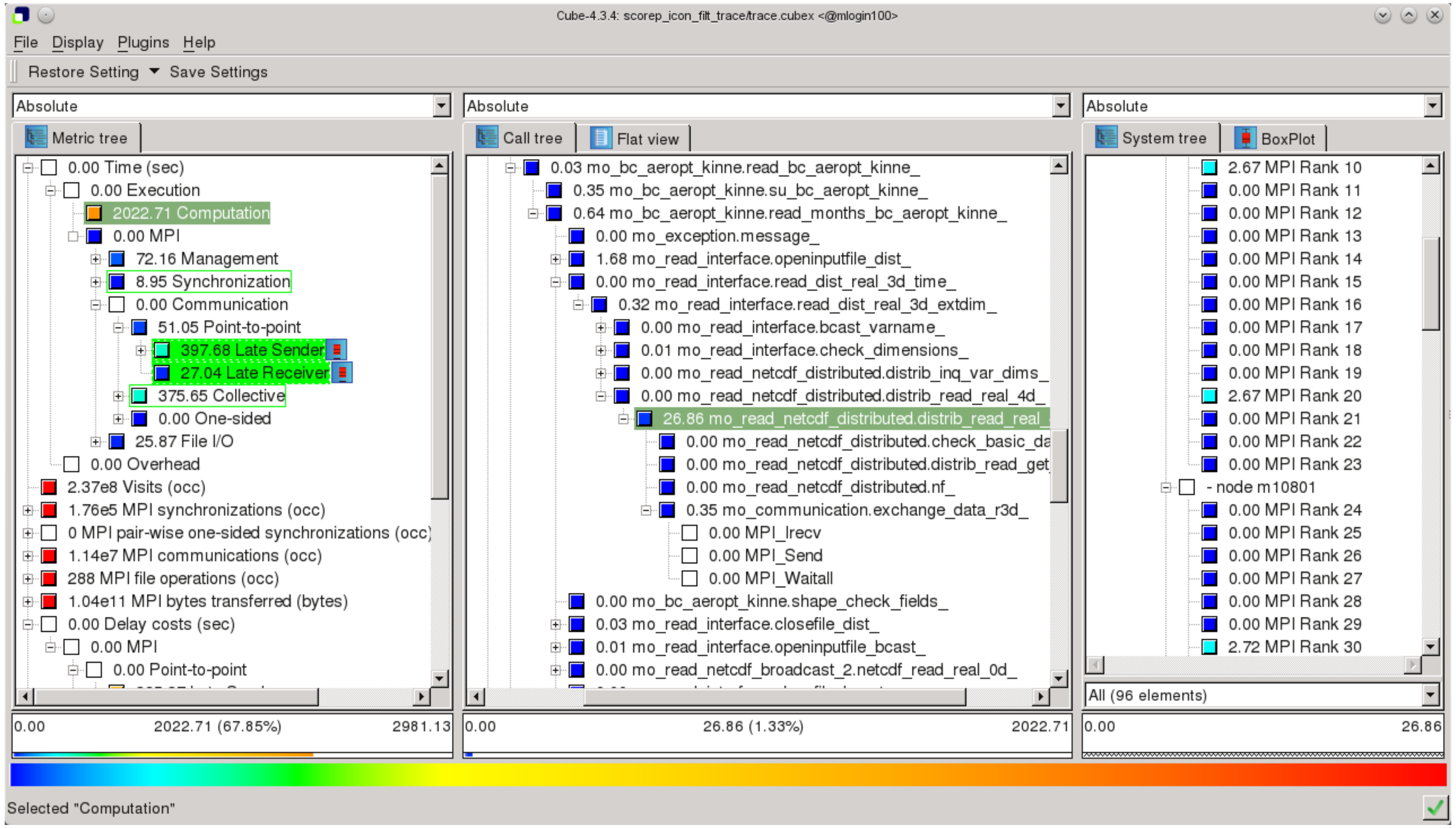
File Display Plugins Help

Restore Setting Save Settings

```
Absolute                                   Absolute                                                   Absolute
Metric tree                                Call tree | Flat view                                      System tree | BoxPlot

0.00 Communication                         0.01 mo_bc_aeropt_kinne.read_bc_aeropt_kinne_              21.54 MPI Rank 10
  51.05 Point-to-point                       0.01 mo_bc_aeropt_kinne.su_bc_aeropt_kinne_              0.00 MPI Rank 11
    397.68 Late Sender                       0.07 mo_bc_aeropt_kinne.read_months_bc_aeropt_kinne_     0.00 MPI Rank 12
    27.04 Late Receiver                        0.00 mo_exception.message_                             0.00 MPI Rank 13
  375.65 Collective                            9.13 mo_read_interface.openinputfile_dist_             0.00 MPI Rank 14
  0.00 One-sided                               0.00 mo_read_interface.read_dist_real_3d_time_         0.00 MPI Rank 15
25.87 File I/O                                   0.02 mo_read_interface.read_dist_real_3d_extdim_     0.00 MPI Rank 16
0.00 Overhead                                      0.31 mo_read_interface.bcast_varname_              0.00 MPI Rank 17
2.37e8 Visits (occ)                                0.04 mo_read_interface.check_dimensions_           0.00 MPI Rank 18
1.76e5 MPI synchronizations (occ)                  0.02 mo_read_netcdf_distributed.distrib_inq_var_dims_  0.00 MPI Rank 19
0 MPI pair-wise one-sided synchronizations (occ)   0.00 mo_read_netcdf_distributed.distrib_read_real_4d_  6.29 MPI Rank 20
1.14e7 MPI communications (occ)                      223.08 mo_read_netcdf_distributed.distrib_read_real  0.00 MPI Rank 21
288 MPI file operations (occ)                          0.00 mo_read_netcdf_distributed.check_basic_da     0.00 MPI Rank 22
1.04e11 MPI bytes transferred (bytes)                  0.00 mo_read_netcdf_distributed.distrib_read_get   0.00 MPI Rank 23
0.00 Delay costs (sec)                                 0.01 mo_read_netcdf_distributed.nf_              - node m10801
  0.00 MPI                                             1.35 mo_communication.exchange_data_r3d_           0.00 MPI Rank 24
    0.00 Point-to-point                                  0.06 MPI_Irecv                                   0.00 MPI Rank 25
      385.37 Late Sender                                 2.85 MPI_Send                                    0.00 MPI Rank 26
      26.27 Late Receiver                                0.20 MPI_Waitall                                 0.00 MPI Rank 27
    0.00 Collective                            0.00 mo_bc_aeropt_kinne.shape_check_fields_            0.00 MPI Rank 28
      6.75 Wait at Barrier                     0.20 mo_read_interface.closefile_dist_                 0.00 MPI Rank 29
      6.06 Wait at N x N                       0.06 mo_read_interface.openinputfile_bcast_            50.30 MPI Rank 30
      276.63 Late Broadcast                    0.01 mo_read_netcdf_broadcast_2.netcdf_read_real_0d_

                                                                                                      All (96 elements)
0.00     385.37 (54.97%)     701.10        0.00        223.08 (57.89%)        385.37                  0.00            223.08

Selected "Late Sender"
```

- …where every 10th rank causes wait states…

ICON Scalasca trace analysis (cont.)


Cube-4.3.4: scorep_icon_filt_trace/trace.cubex <@mlogin100>

File Display Plugins Help

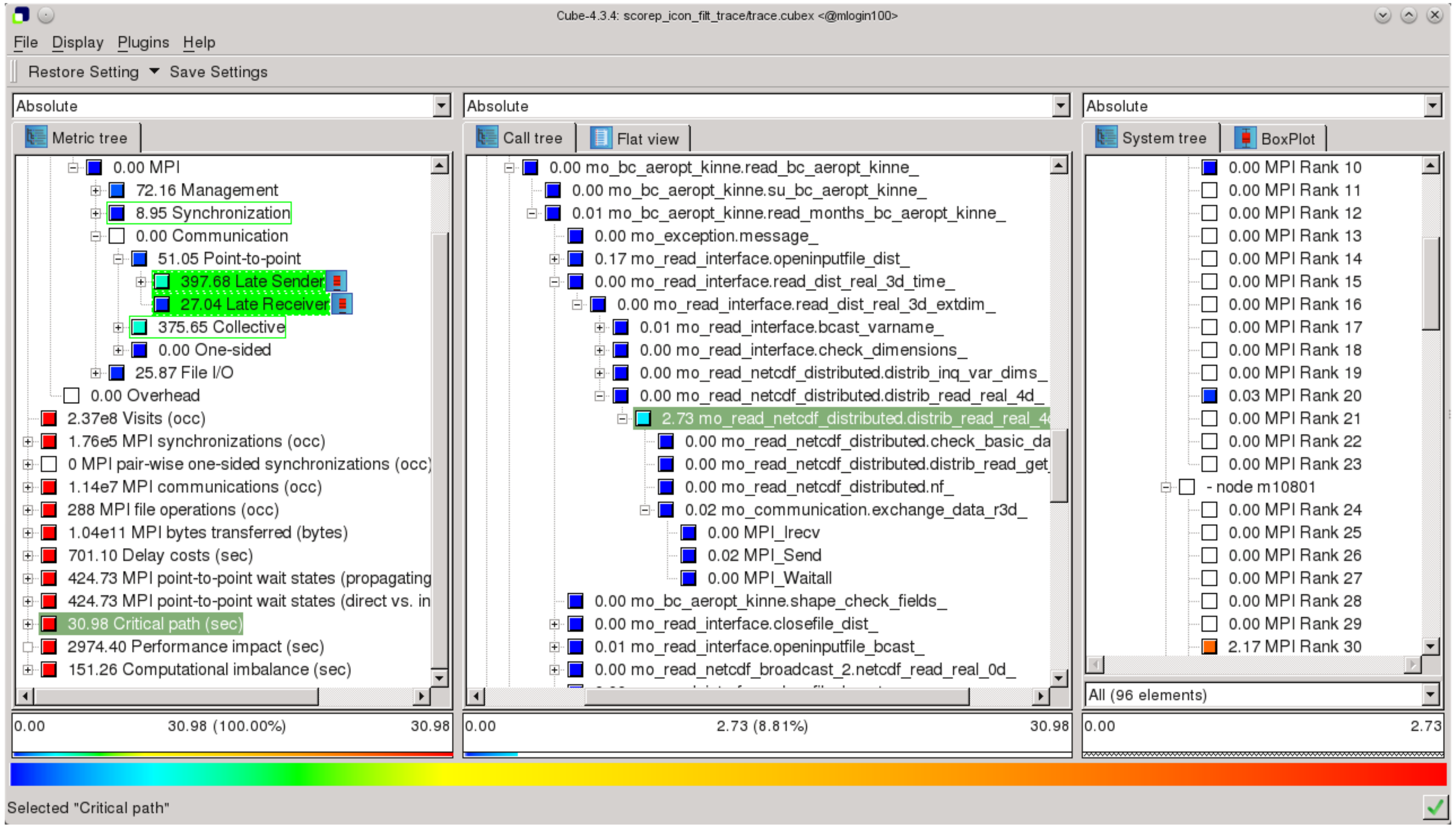
Restore Setting ▾ Save Settings

```
Absolute                                    Absolute                                                  Absolute
Metric tree                                 Call tree | Flat view                                     System tree | BoxPlot

0.00 Time (sec)                             0.03 mo_bc_aeropt_kinne.read_bc_aeropt_kinne_             2.67 MPI Rank 10
  0.00 Execution                              0.35 mo_bc_aeropt_kinne.su_bc_aeropt_kinne_             0.00 MPI Rank 11
    2022.71 Computation                       0.64 mo_bc_aeropt_kinne.read_months_bc_aeropt_kinne_    0.00 MPI Rank 12
    0.00 MPI                                    0.00 mo_exception.message_                            0.00 MPI Rank 13
      72.16 Management                          1.68 mo_read_interface.openinputfile_dist_            0.00 MPI Rank 14
      8.95 Synchronization                      0.00 mo_read_interface.read_dist_real_3d_time_        0.00 MPI Rank 15
      0.00 Communication                          0.32 mo_read_interface.read_dist_real_3d_extdim_    0.00 MPI Rank 16
        51.05 Point-to-point                        0.00 mo_read_interface.bcast_varname_             0.00 MPI Rank 17
          397.68 Late Sender                        0.01 mo_read_interface.check_dimensions_          0.00 MPI Rank 18
          27.04 Late Receiver                       0.00 mo_read_netcdf_distributed.distrib_inq_var_dims_  0.00 MPI Rank 19
        375.65 Collective                           0.00 mo_read_netcdf_distributed.distrib_read_real_4d_  2.67 MPI Rank 20
        0.00 One-sided                                26.86 mo_read_netcdf_distributed.distrib_read_real_  0.00 MPI Rank 21
      25.87 File I/O                                    0.00 mo_read_netcdf_distributed.check_basic_da    0.00 MPI Rank 22
  0.00 Overhead                                         0.00 mo_read_netcdf_distributed.distrib_read_get  0.00 MPI Rank 23
2.37e8 Visits (occ)                                     0.00 mo_read_netcdf_distributed.nf_           - node m10801
1.76e5 MPI synchronizations (occ)                       0.35 mo_communication.exchange_data_r3d_          0.00 MPI Rank 24
0 MPI pair-wise one-sided synchronizations (occ)          0.00 MPI_Irecv                                  0.00 MPI Rank 25
1.14e7 MPI communications (occ)                           0.00 MPI_Send                                   0.00 MPI Rank 26
288 MPI file operations (occ)                             0.00 MPI_Waitall                                0.00 MPI Rank 27
1.04e11 MPI bytes transferred (bytes)           0.00 mo_bc_aeropt_kinne.shape_check_fields_           0.00 MPI Rank 28
0.00 Delay costs (sec)                          0.03 mo_read_interface.closefile_dist_                0.00 MPI Rank 29
  0.00 MPI                                      0.01 mo_read_interface.openinputfile_bcast_           2.72 MPI Rank 30
    0.00 Point-to-point                         0.00 mo_read_netcdf_broadcast_2.netcdf_read_real_0d_
                                                                                                      All (96 elements)

0.00      2022.71 (67.85%)      2981.13     0.00      26.86 (1.33%)      2022.71                      0.00      26.86
```

Selected "Computation"

- ...much larger than the imbalance itself!

ICON Scalasca trace analysis (cont.)



Cube-4.3.4: scorep_icon_filt_trace/trace.cubex <@mlogin100>

File Display Plugins Help

Restore Setting Save Settings

Absolute | Absolute | Absolute

Metric tree

```
0.00 MPI
  72.16 Management
  8.95 Synchronization
  0.00 Communication
    51.05 Point-to-point
      397.68 Late Sender
      27.04 Late Receiver
    375.65 Collective
    0.00 One-sided
  25.87 File I/O
0.00 Overhead
2.37e8 Visits (occ)
1.76e5 MPI synchronizations (occ)
0 MPI pair-wise one-sided synchronizations (occ)
1.14e7 MPI communications (occ)
288 MPI file operations (occ)
1.04e11 MPI bytes transferred (bytes)
701.10 Delay costs (sec)
424.73 MPI point-to-point wait states (propagating
424.73 MPI point-to-point wait states (direct vs. in
30.98 Critical path (sec)
2974.40 Performance impact (sec)
151.26 Computational imbalance (sec)
```

Call tree | Flat view

```
0.00 mo_bc_aeropt_kinne.read_bc_aeropt_kinne_
  0.00 mo_bc_aeropt_kinne.su_bc_aeropt_kinne_
  0.01 mo_bc_aeropt_kinne.read_months_bc_aeropt_kinne_
    0.00 mo_exception.message_
    0.17 mo_read_interface.openinputfile_dist_
    0.00 mo_read_interface.read_dist_real_3d_time_
      0.00 mo_read_interface.read_dist_real_3d_extdim_
        0.01 mo_read_interface.bcast_varname_
        0.00 mo_read_interface.check_dimensions_
        0.00 mo_read_netcdf_distributed.distrib_inq_var_dims_
        0.00 mo_read_netcdf_distributed.distrib_read_real_4d_
          2.73 mo_read_netcdf_distributed.distrib_read_real_4
            0.00 mo_read_netcdf_distributed.check_basic_da
            0.00 mo_read_netcdf_distributed.distrib_read_get
            0.00 mo_read_netcdf_distributed.nf_
            0.02 mo_communication.exchange_data_r3d_
              0.00 MPI_Irecv
              0.02 MPI_Send
              0.00 MPI_Waitall
    0.00 mo_bc_aeropt_kinne.shape_check_fields_
    0.00 mo_read_interface.closefile_dist_
    0.01 mo_read_interface.openinputfile_bcast_
    0.00 mo_read_netcdf_broadcast_2.netcdf_read_real_0d_
```

System tree | BoxPlot

```
0.00 MPI Rank 10
0.00 MPI Rank 11
0.00 MPI Rank 12
0.00 MPI Rank 13
0.00 MPI Rank 14
0.00 MPI Rank 15
0.00 MPI Rank 16
0.00 MPI Rank 17
0.00 MPI Rank 18
0.00 MPI Rank 19
0.03 MPI Rank 20
0.00 MPI Rank 21
0.00 MPI Rank 22
0.00 MPI Rank 23
- node m10801
  0.00 MPI Rank 24
  0.00 MPI Rank 25
  0.00 MPI Rank 26
  0.00 MPI Rank 27
  0.00 MPI Rank 28
  0.00 MPI Rank 29
  2.17 MPI Rank 30
```

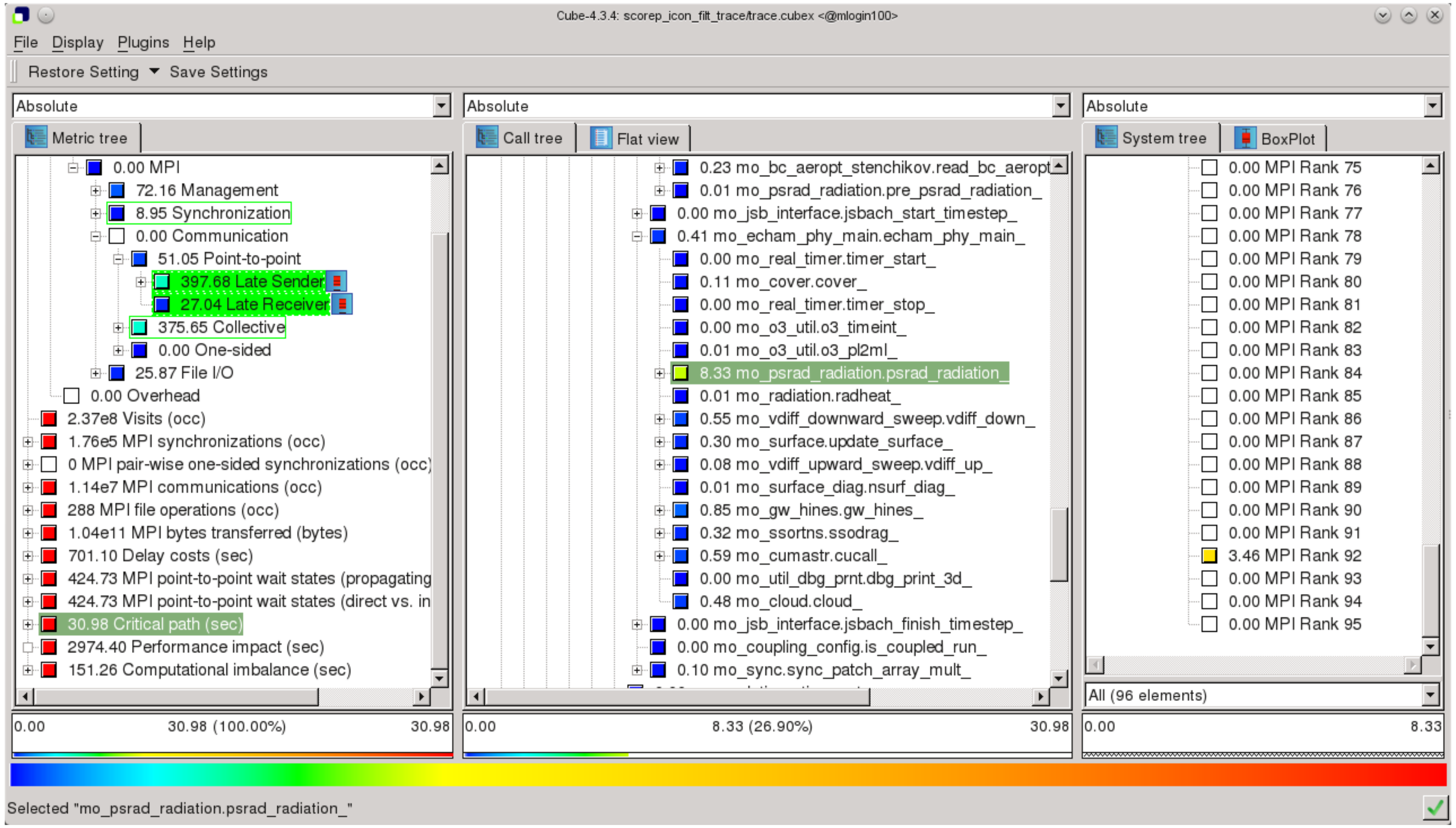
All (96 elements)

0.00 30.98 (100.00%) 30.98 | 0.00 2.73 (8.81%) 30.98 | 0.00 2.73

Selected "Critical path"

- This imbalance is also highlighted by the critical path…

ICON Scalasca trace analysis (cont.)


Cube-4.3.4: scorep_icon_filt_trace/trace.cubex <@mlogin100>
File Display Plugins Help
Restore Setting Save Settings
Absolute
Metric tree
Absolute
Call tree
Flat view
Absolute
System tree
BoxPlot

```
0.00 MPI
  72.16 Management
  8.95 Synchronization
  0.00 Communication
    51.05 Point-to-point
      397.68 Late Sender
      27.04 Late Receiver
    375.65 Collective
    0.00 One-sided
  25.87 File I/O
0.00 Overhead
2.37e8 Visits (occ)
1.76e5 MPI synchronizations (occ)
0 MPI pair-wise one-sided synchronizations (occ)
1.14e7 MPI communications (occ)
288 MPI file operations (occ)
1.04e11 MPI bytes transferred (bytes)
701.10 Delay costs (sec)
424.73 MPI point-to-point wait states (propagating
424.73 MPI point-to-point wait states (direct vs. in
30.98 Critical path (sec)
2974.40 Performance impact (sec)
151.26 Computational imbalance (sec)
```

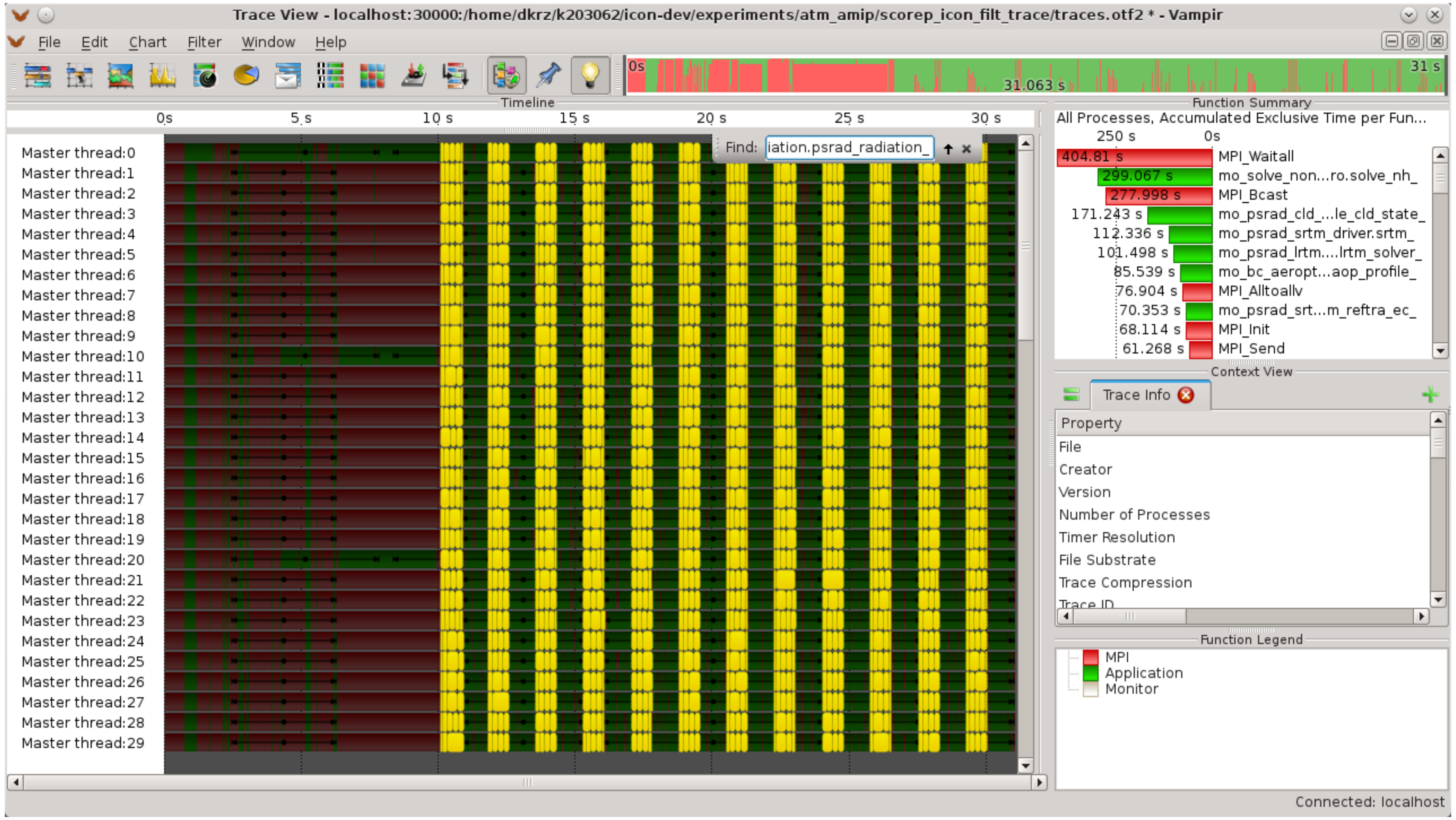
```
  0.23 mo_bc_aeropt_stenchikov.read_bc_aeropt
  0.01 mo_psrad_radiation.pre_psrad_radiation_
0.00 mo_jsb_interface.jsbach_start_timestep_
0.41 mo_echam_phy_main.echam_phy_main_
  0.00 mo_real_timer.timer_start_
  0.11 mo_cover.cover_
  0.00 mo_real_timer.timer_stop_
  0.00 mo_o3_util.o3_timeint_
  0.01 mo_o3_util.o3_pl2ml_
  8.33 mo_psrad_radiation.psrad_radiation_
  0.01 mo_radiation.radheat_
  0.55 mo_vdiff_downward_sweep.vdiff_down_
  0.30 mo_surface.update_surface_
  0.08 mo_vdiff_upward_sweep.vdiff_up_
  0.01 mo_surface_diag.nsurf_diag_
  0.85 mo_gw_hines.gw_hines_
  0.32 mo_ssortns.ssodrag_
  0.59 mo_cumastr.cucall_
  0.00 mo_util_dbg_prnt.dbg_print_3d_
  0.48 mo_cloud.cloud_
0.00 mo_jsb_interface.jsbach_finish_timestep_
0.00 mo_coupling_config.is_coupled_run_
0.10 mo_sync.sync_patch_array_mult_
```

```
0.00 MPI Rank 75
0.00 MPI Rank 76
0.00 MPI Rank 77
0.00 MPI Rank 78
0.00 MPI Rank 79
0.00 MPI Rank 80
0.00 MPI Rank 81
0.00 MPI Rank 82
0.00 MPI Rank 83
0.00 MPI Rank 84
0.00 MPI Rank 85
0.00 MPI Rank 86
0.00 MPI Rank 87
0.00 MPI Rank 88
0.00 MPI Rank 89
0.00 MPI Rank 90
0.00 MPI Rank 91
3.46 MPI Rank 92
0.00 MPI Rank 93
0.00 MPI Rank 94
0.00 MPI Rank 95
All (96 elements)
```

```
0.00        30.98 (100.00%)     30.98
0.00        8.33 (26.90%)       30.98
0.00                            8.33
Selected "mo_psrad_radiation.psrad_radiation_"
```

- …as well as the **`psrad_radiation`** routine which consumes >10% of the critical path time

ICON Vampir trace analysis



Trace View - localhost:30000:/home/dkrz/k203062/icon-dev/experiments/atm_amip/scorep_icon_filt_trace/traces.otf2 * - Vampir

All Processes, Accumulated Exclusive Time per Fun...

250 s 0s

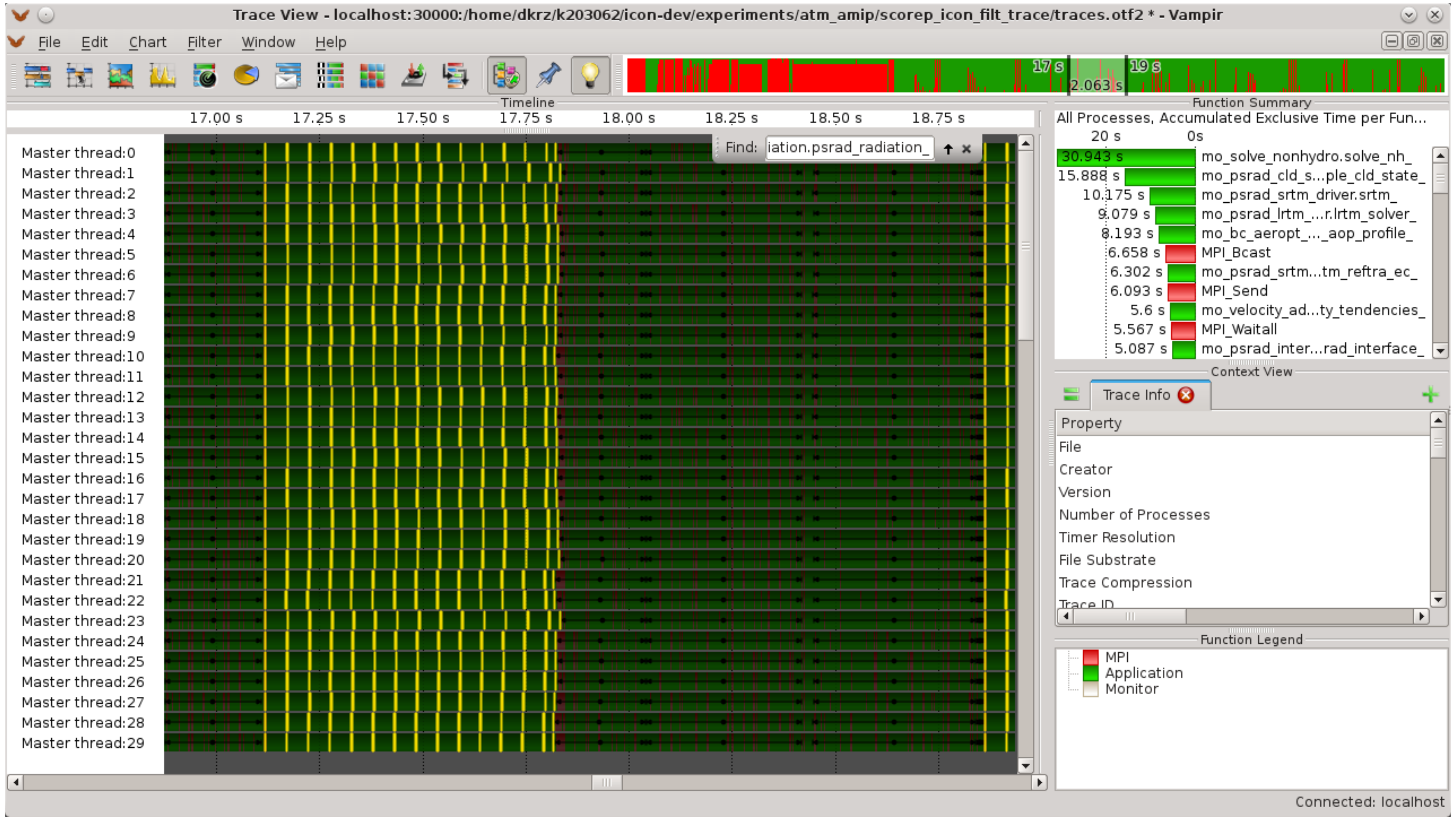
404.81 s	MPI_Waitall
299.067 s	mo_solve_non...ro.solve_nh_
277.998 s	MPI_Bcast
171.243 s	mo_psrad_cld_...le_cld_state_
112.336 s	mo_psrad_srtm_driver.srtm_
101.498 s	mo_psrad_lrtm....lrtm_solver_
85.539 s	mo_bc_aeropt...aop_profile_
76.904 s	MPI_Alltoallv
70.353 s	mo_psrad_srt...m_reftra_ec_
68.114 s	MPI_Init
61.268 s	MPI_Send

Property
File
Creator
Version
Number of Processes
Timer Resolution
File Substrate
Trace Compression
Trace ID

🟥 MPI
🟩 Application
⬜ Monitor

- **psrad_radiation** routine highlighted in trace

ICON Vampir trace analysis (cont.)


Trace View - localhost:30000:/home/dkrz/k203062/icon-dev/experiments/atm_amip/scorep_icon_filt_trace/traces.otf2 * - Vampir
File Edit Chart Filter Window Help
Timeline
Find: iation.psrad_radiation_
Function Summary
Context View
Trace Info
Function Legend
Connected: localhost

All Processes, Accumulated Exclusive Time per Fun...

Time	Function
30.943 s	mo_solve_nonhydro.solve_nh_
15.888 s	mo_psrad_cld_s...ple_cld_state_
10.175 s	mo_psrad_srtm_driver.srtm_
9.079 s	mo_psrad_lrtm_...r.lrtm_solver_
8.193 s	mo_bc_aeropt_..._aop_profile_
6.658 s	MPI_Bcast
6.302 s	mo_psrad_srtm...tm_reftra_ec_
6.093 s	MPI_Send
5.6 s	mo_velocity_ad...ty_tendencies_
5.567 s	MPI_Waitall
5.087 s	mo_psrad_inter...rad_interface_

Property: File, Creator, Version, Number of Processes, Timer Resolution, File Substrate, Trace Compression, Trace ID

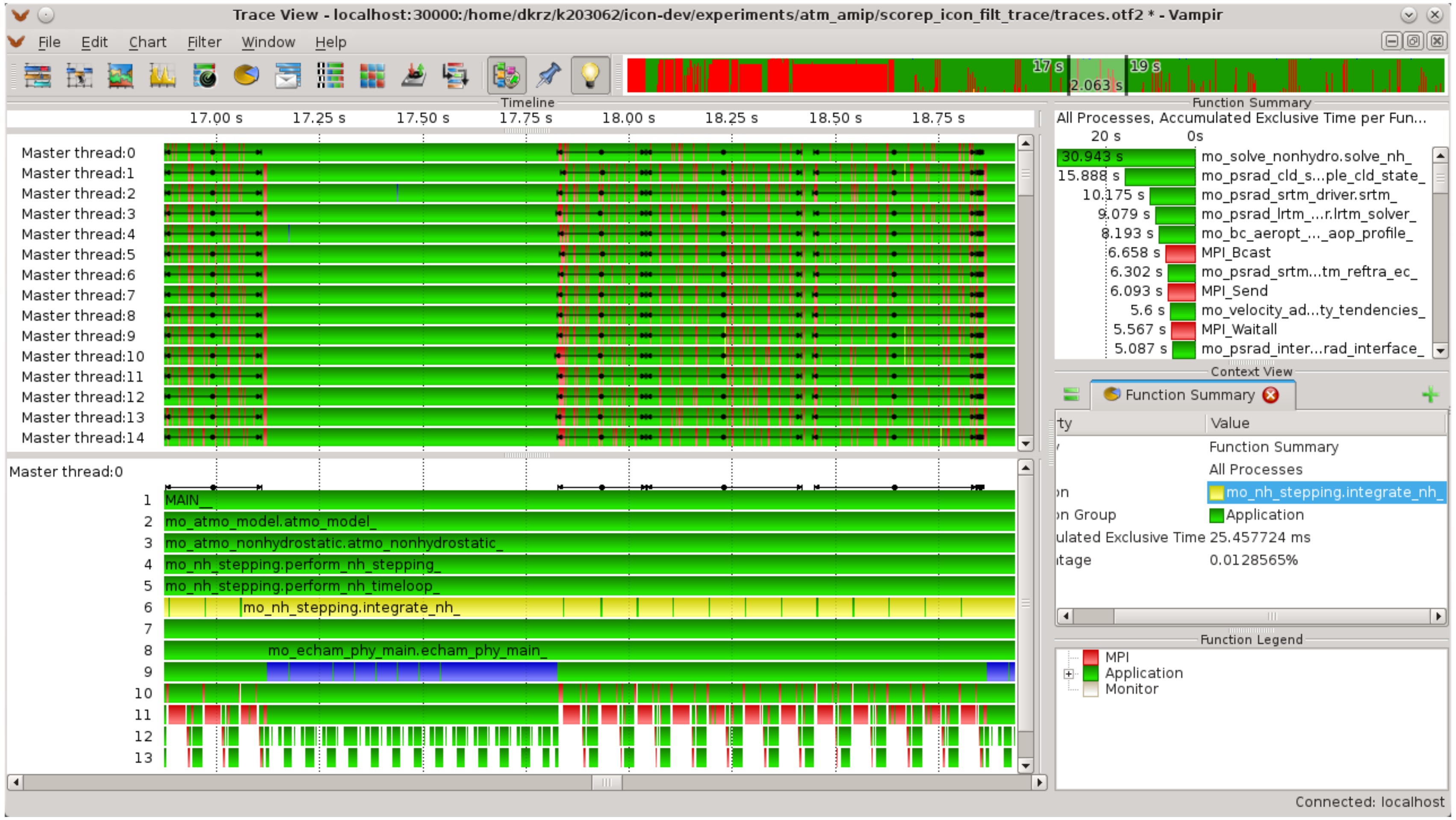
Function Legend: 🟥 MPI, 🟩 Application, ⬜ Monitor

- …zoomed on one iteration block

Outline

- Case I:
  - NPB3.3_MZ_MPI/**BT-MZ** (MPI+OpenMP) on *MARCONI-KNL*: load balancing
- Case II:
  - **k-Wave** (MPI+OpenMP) on *Salomon*: load-balancing in FFTW OpenMP parallel regions
- Case III:
  - **ICON** (MPI) on *Mistral*: automatic trace analysis of critical path of execution
- Case IV:
  - **PIConGPU** (MPI+CUDA): computation offload to multiple attached accelerator devices
- Case V:
  - **TensorFlow** (Python+CUDA): interpreted & compiled heterogeneous execution measurement

ICON Vampir trace analysis (cont.)


Trace View - localhost:30000:/home/dkrz/k203062/icon-dev/experiments/atm_amip/scorep_icon_filt_trace/traces.otf2 * - Vampir

Function Summary

All Processes, Accumulated Exclusive Time per Fun...

Time	Function
30.943 s	mo_solve_nonhydro.solve_nh_
15.888 s	mo_psrad_cld_s...ple_cld_state_
10.175 s	mo_psrad_srtm_driver.srtm_
9.079 s	mo_psrad_lrtm_...r.lrtm_solver_
8.193 s	mo_bc_aeropt_..._aop_profile_
6.658 s	MPI_Bcast
6.302 s	mo_psrad_srtm...tm_reftra_ec_
6.093 s	MPI_Send
5.6 s	mo_velocity_ad...ty_tendencies_
5.567 s	MPI_Waitall
5.087 s	mo_psrad_inter...rad_interface_

Context View

ty	Value
/	Function Summary
	All Processes
on	mo_nh_stepping.integrate_nh_
on Group	Application
ulated Exclusive Time	25.457724 ms
tage	0.0128565%

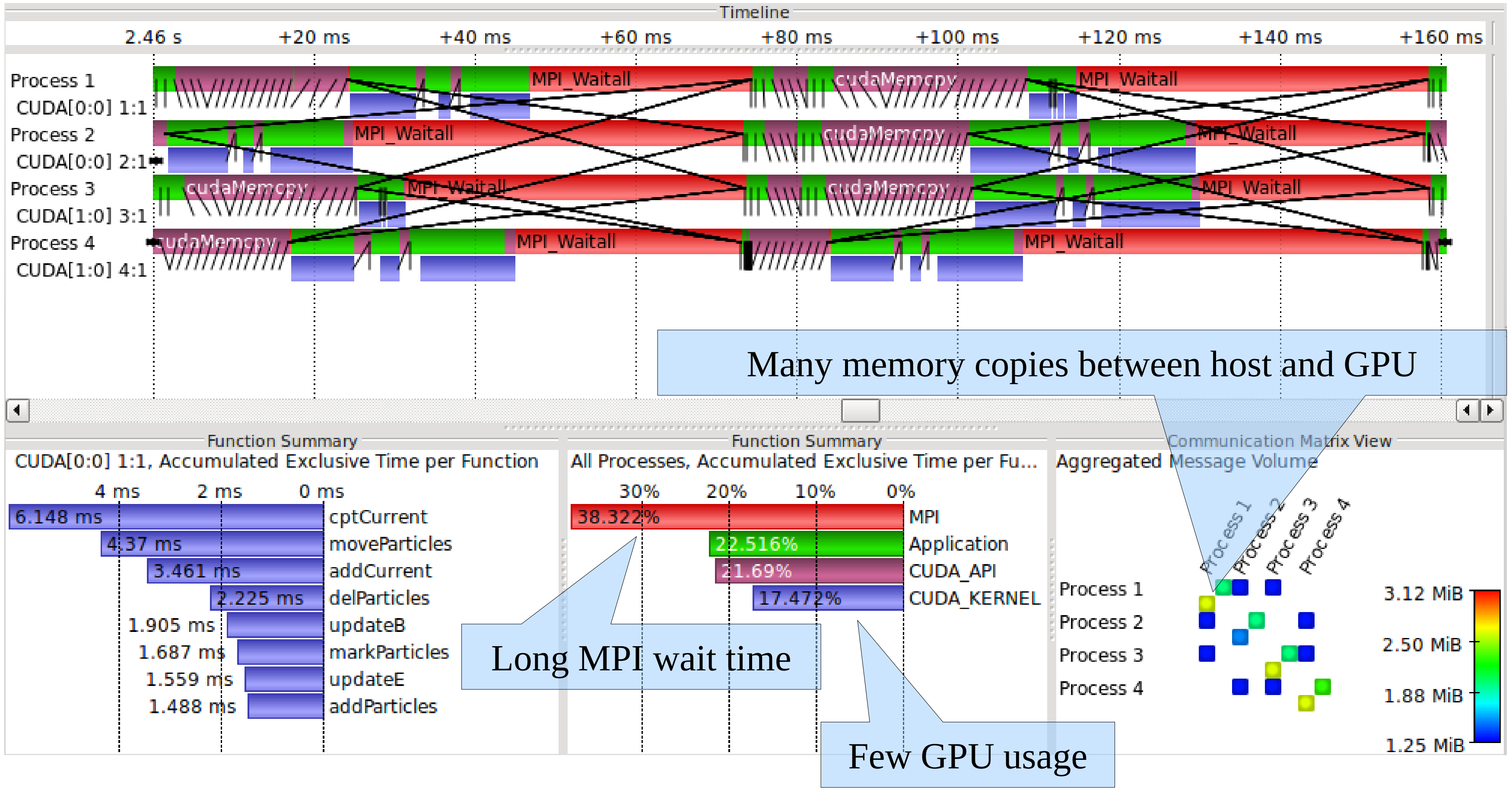
Connected: localhost

- ...with process timeline showing that **psrad_radiation** (blue) is called every 12th **integrate_nh** iteration (yellow)

Case IV: PIConGPU

- A fully-relativistic 3D3V plasma physics particle-in-cell code for many GPGPUs, developed by HZDR in collaboration with ZIH, TU Dresden
  https://github.com/ComputationalRadiationPhysics/picongpu
- Incremental software evolution
  - C++ & CUDA with MPI
- Continuous performance analysis and optimization
  - 2013 Gordon Bell Prize finalist for outstanding performance and scalability to over 18,000 GPGPUs

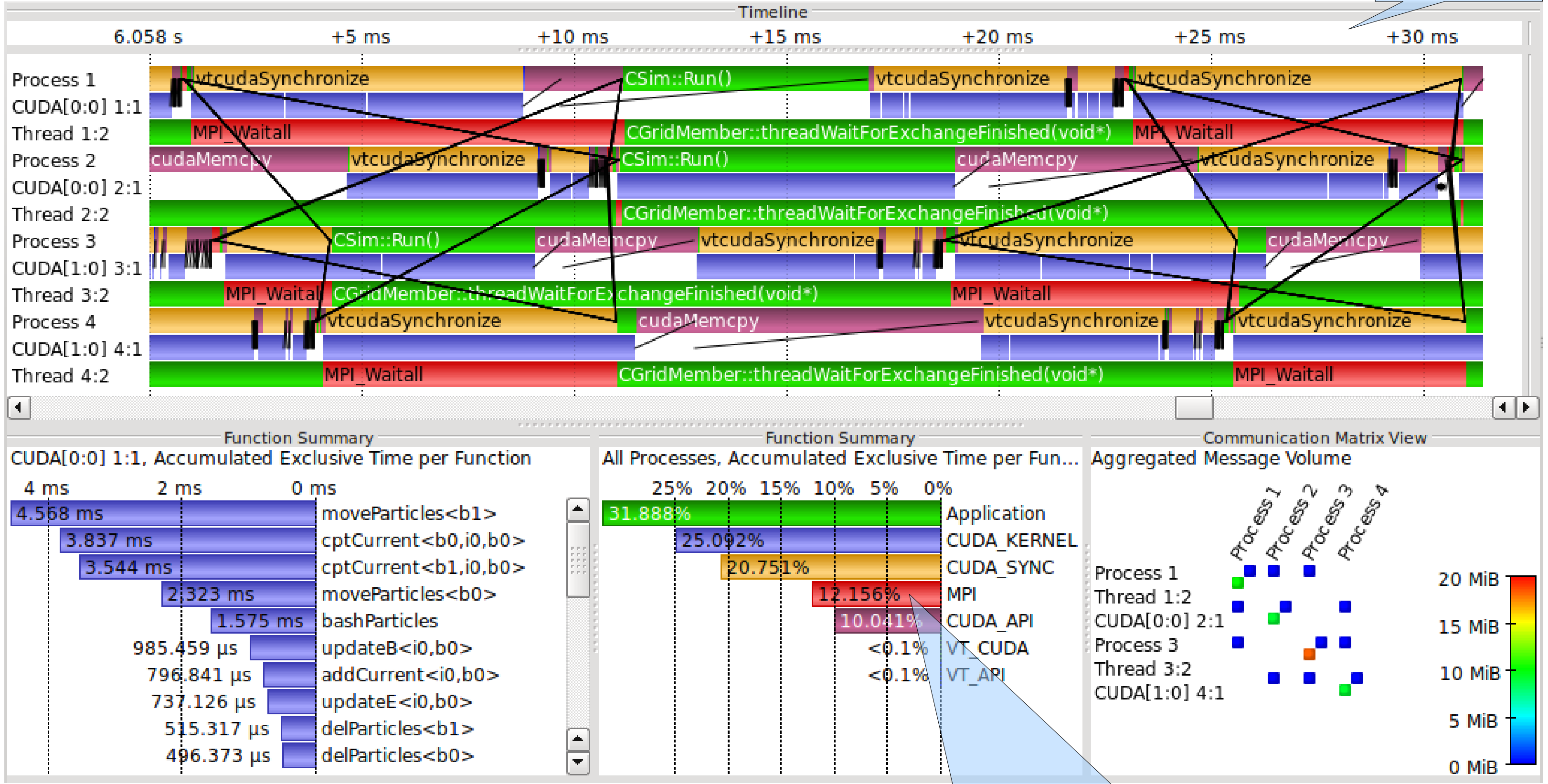
First parallel PIConGPU implementation (1 run step)



PIConGPU I (1 run step)

- General software design improvements

Sp ≈ 4



Function Summary
CUDA[0:0] 1:1, Accumulated Exclusive Time per Function

Time	Function
4.568 ms	moveParticles<b1>
3.837 ms	cptCurrent<b0,i0,b0>
3.544 ms	cptCurrent<b1,i0,b0>
2.323 ms	moveParticles<b0>
1.575 ms	bashParticles
985.459 µs	updateB<i0,b0>
796.841 µs	addCurrent<i0,b0>
737.126 µs	updateE<i0,b0>
515.317 µs	delParticles<b1>
496.373 µs	delParticles<b0>

Function Summary
All Processes, Accumulated Exclusive Time per Fun...

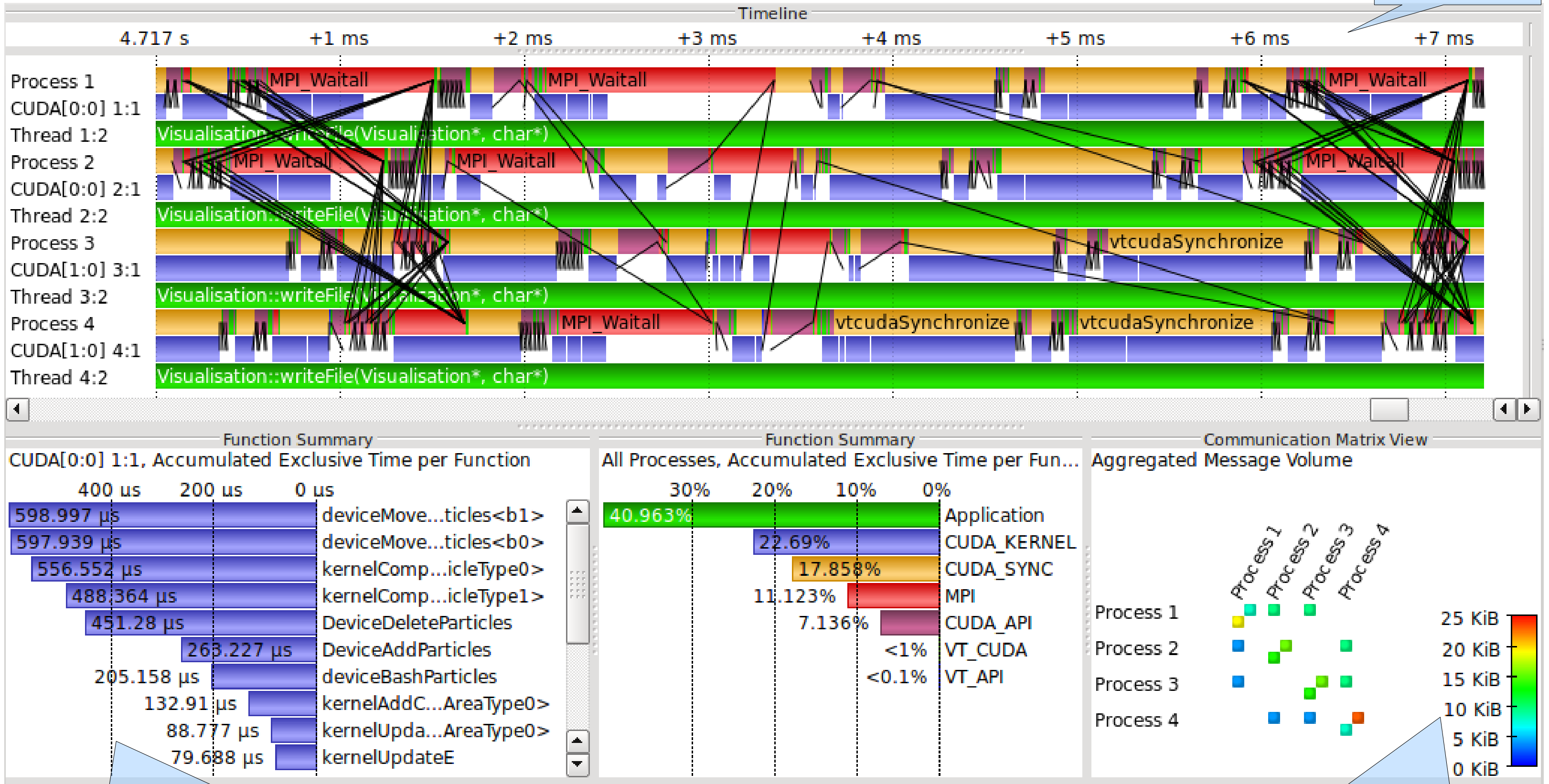
Share	Group
31.888%	Application
25.092%	CUDA_KERNEL
20.751%	CUDA_SYNC
12.156%	MPI
10.041%	CUDA_API
<0.1%	VT_CUDA
<0.1%	VT_API

Dramatically reduced MPI wait time

PICon GPU II (1 run step)

- Software re-design

Sp ≈ 4



Function Summary

CUDA[0:0] 1:1, Accumulated Exclusive Time per Function

Time	Function
598.997 µs	deviceMove...ticles<b1>
597.939 µs	deviceMove...ticles<b0>
556.552 µs	kernelComp...icleType0>
488.364 µs	kernelComp...icleType1>
451.28 µs	DeviceDeleteParticles
263.227 µs	DeviceAddParticles
205.158 µs	deviceBashParticles
132.91 µs	kernelAddC...AreaType0>
88.777 µs	kernelUpda...AreaType0>
79.688 µs	kernelUpdateE

Function Summary

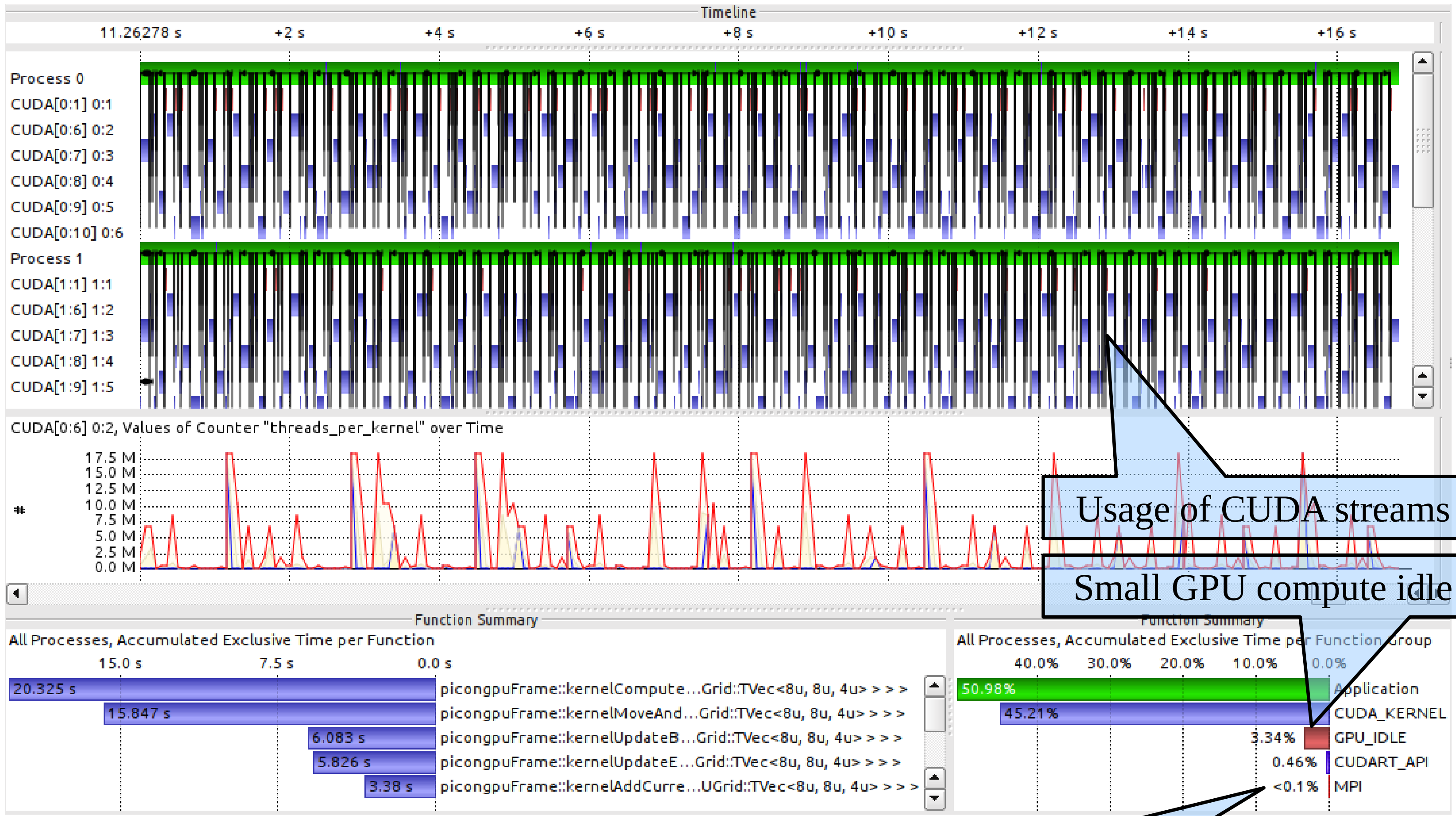
All Processes, Accumulated Exclusive Time per Fun...

Share	Function
40.963%	Application
22.69%	CUDA_KERNEL
17.858%	CUDA_SYNC
11.123%	MPI
7.136%	CUDA_API
<1%	VT_CUDA
<0.1%	VT_API

shorter kernel execution times

smaller messages (1 order)

PIConGPU – Today



Function Summary

All Processes, Accumulated Exclusive Time per Function

Time	Function
20.325 s	picongpuFrame::kernelCompute…Grid::TVec<8u, 8u, 4u> > > >
15.847 s	picongpuFrame::kernelMoveAnd…Grid::TVec<8u, 8u, 4u> > > >
6.083 s	picongpuFrame::kernelUpdateB…Grid::TVec<8u, 8u, 4u> > > >
5.826 s	picongpuFrame::kernelUpdateE…Grid::TVec<8u, 8u, 4u> > > >
3.38 s	picongpuFrame::kernelAddCurre…UGrid::TVec<8u, 8u, 4u> > > >

Share	Function Group
50.98%	Application
45.21%	CUDA_KERNEL
3.34%	GPU_IDLE
0.46%	CUDART_API
<0.1%	MPI

Usage of CUDA streams

Small GPU compute idle

No MPI wait, just short MPI_Test

Outline

- Case I:
  - NPB3.3_MZ_MPI/**BT-MZ** (MPI+OpenMP) on *MARCONI-KNL*: load balancing
- Case II:
  - **k-Wave** (MPI+OpenMP) on *Salomon*: load-balancing in FFTW OpenMP parallel regions
- Case III:
  - **ICON** (MPI) on *Mistral*: automatic trace analysis of critical path of execution
- Case IV:
  - **PIConGPU** (MPI+CUDA): computation offload to multiple attached accelerator devices
- Case V:
  - **TensorFlow** (Python+CUDA): interpreted & compiled heterogeneous execution measurement

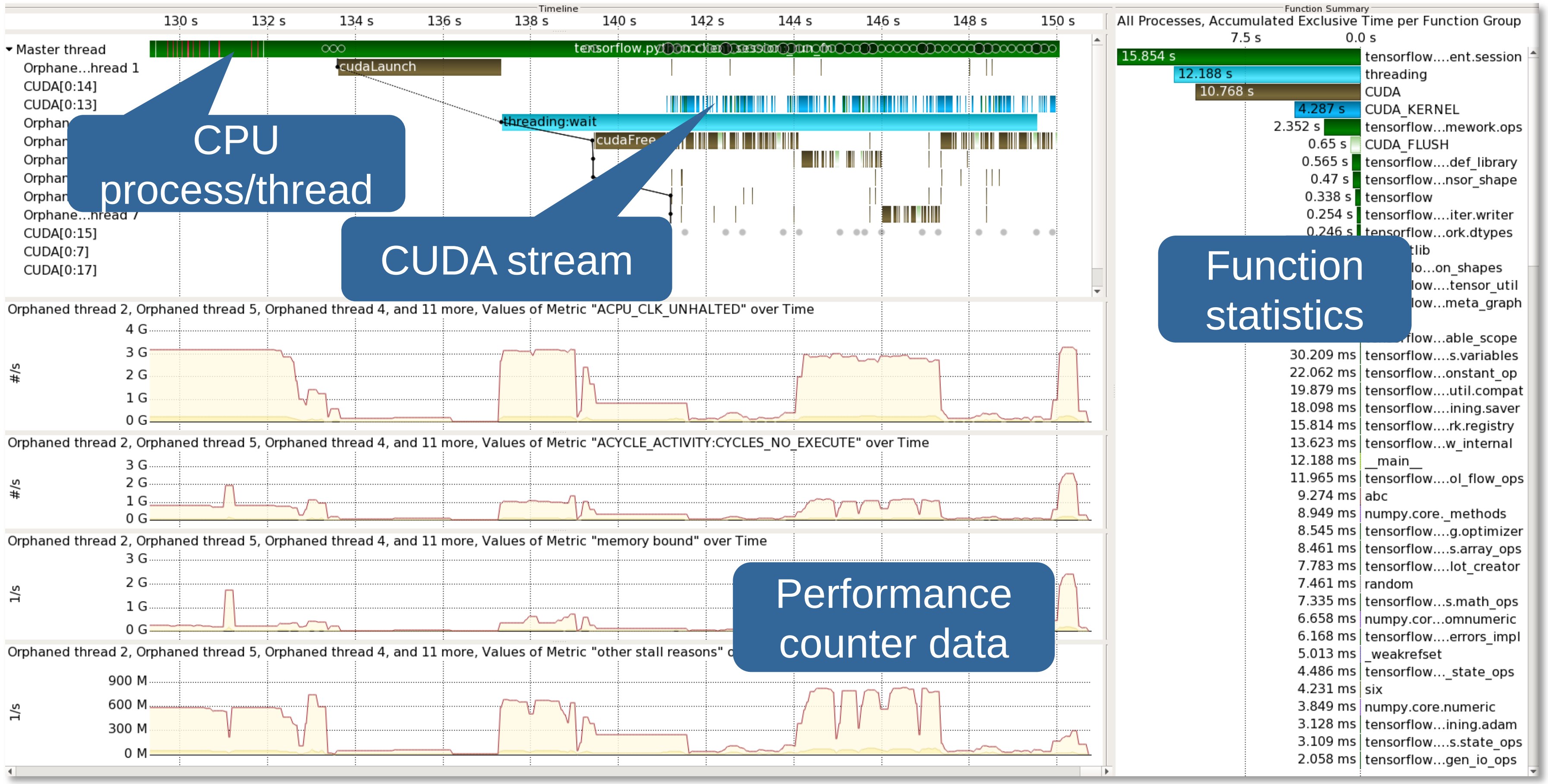
Case V: TensorFlow

- TensorFlow is one of the most popular Deep Learning frameworks
  - also the foundation of other tools, e.g., Keras
- Use additional Python bindings for Score-P to obtain execution performance data
  - available at https://github.com/score-p/scorep_binding_python
  - CUDA activities are recorded using CUPTI
- Execution of a single TensorFlow process on a workstation with a single GPU device, forking multiple threads

⇒ ***Optimized execution using NumPy array of doubles vs. native array***

TensorFlow

- Application process, its threads, and CUDA streams with corresponding performance counter data



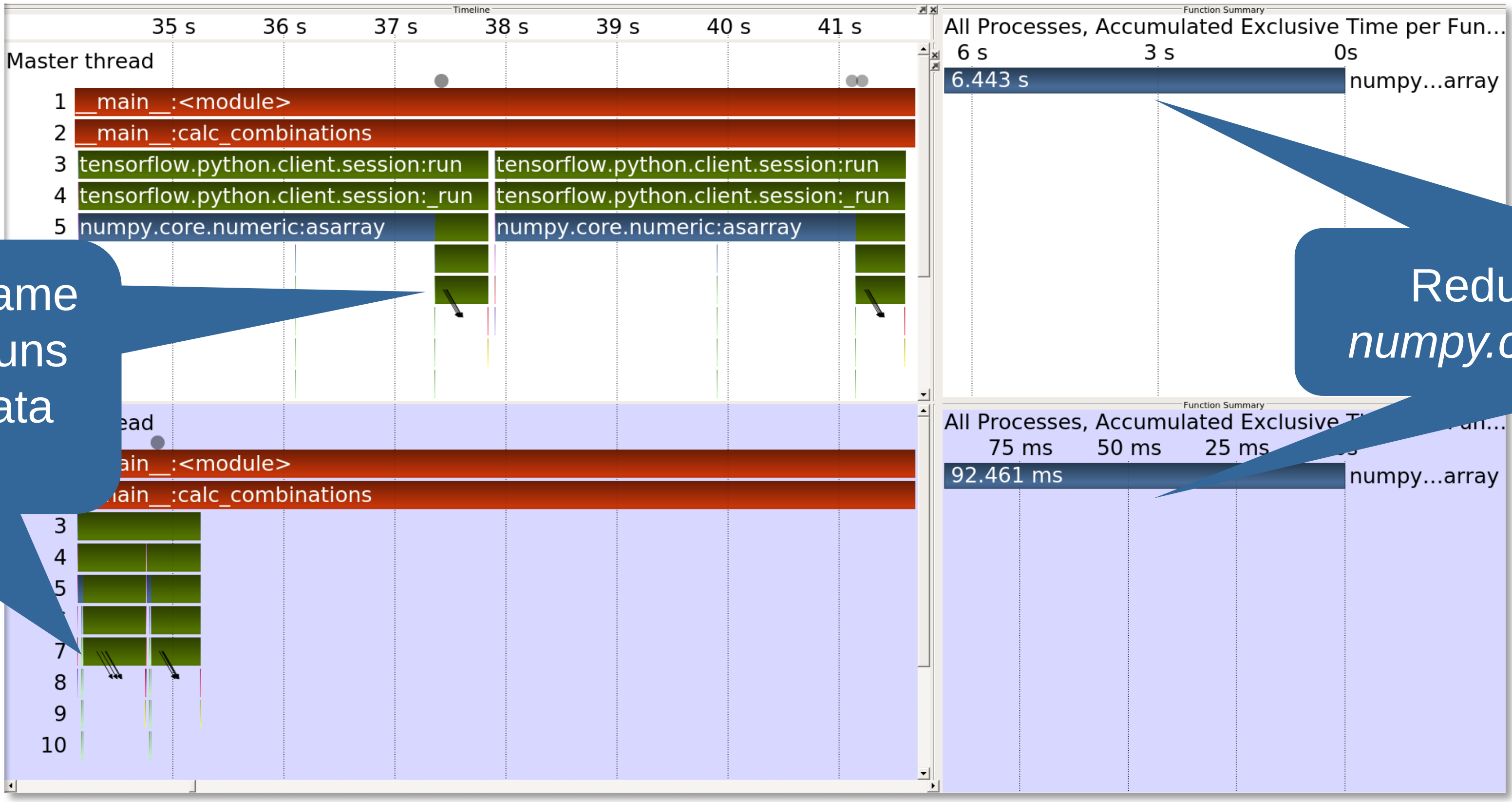
Timeline
130 s
132 s
134 s
136 s
138 s
140 s
142 s
144 s
146 s
148 s
150 s
Master thread
Orphane...hread 1
CUDA[0:14]
CUDA[0:13]
Orphan
Orphan
Orphan
Orphan
Orphan
Orphane...hread 7
CUDA[0:15]
CUDA[0:7]
CUDA[0:17]
tensorflow.python.client.session_run_fn
cudaLaunch
threading:wait
cudaFree
CPU process/thread
CUDA stream
Orphaned thread 2, Orphaned thread 5, Orphaned thread 4, and 11 more, Values of Metric "ACPU_CLK_UNHALTED" over Time
#/s
4 G
3 G
2 G
1 G
0 G
Orphaned thread 2, Orphaned thread 5, Orphaned thread 4, and 11 more, Values of Metric "ACYCLE_ACTIVITY:CYCLES_NO_EXECUTE" over Time
#/s
3 G
2 G
1 G
0 G
Orphaned thread 2, Orphaned thread 5, Orphaned thread 4, and 11 more, Values of Metric "memory bound" over Time
1/s
3 G
2 G
1 G
0 G
Orphaned thread 2, Orphaned thread 5, Orphaned thread 4, and 11 more, Values of Metric "other stall reasons" o
1/s
900 M
600 M
300 M
0 M
Performance counter data
Function Summary
Function statistics
7.5 s
0.0 s

All Processes, Accumulated Exclusive Time per Function Group

15.854 s	tensorflow....ent.session
12.188 s	threading
10.768 s	CUDA
4.287 s	CUDA_KERNEL
2.352 s	tensorflow...mework.ops
0.65 s	CUDA_FLUSH
0.565 s	tensorflow....def_library
0.47 s	tensorflow...nsor_shape
0.338 s	tensorflow
0.254 s	tensorflow....iter.writer
0.246 s	tensorflow...ork.dtypes
	tlib
	lo...on_shapes
	low....tensor_util
	low...meta_graph
	flow...able_scope
30.209 ms	tensorflow....s.variables
22.062 ms	tensorflow...onstant_op
19.879 ms	tensorflow....util.compat
18.098 ms	tensorflow....ining.saver
15.814 ms	tensorflow....rk.registry
13.623 ms	tensorflow....w_internal
12.188 ms	__main__
11.965 ms	tensorflow....ol_flow_ops
9.274 ms	abc
8.949 ms	numpy.core._methods
8.545 ms	tensorflow....g.optimizer
8.461 ms	tensorflow....s.array_ops
7.783 ms	tensorflow....lot_creator
7.461 ms	random
7.335 ms	tensorflow...s.math_ops
6.658 ms	numpy.cor...omnumeric
6.168 ms	tensorflow....errors_impl
5.013 ms	_weakrefset
4.486 ms	tensorflow..._state_ops
4.231 ms	six
3.849 ms	numpy.core.numeric
3.128 ms	tensorflow...ining.adam
3.109 ms	tensorflow....s.state_ops
2.058 ms	tensorflow...gen_io_ops

TensorFlow

- Comparison view of the original (top) and optimized (bottom) application run.



One iteration of the same application, the two runs differ in their input data structures

Reduced time spent in *numpy.core.numeric:asarray*

Summary

- Score-P instrumentation & measurement infrastructure is proven to be extremely scalable, portable and flexible
- Basis for diverse execution performance analyses with Scalasca, TAU & Vampir
- Successfully used with a wide variety of parallel applications

- Small representative selection:
  - NPB3.3_MZ_MPI/**BT-MZ** (MPI+OpenMP) on *MARCONI-KNL*: load balancing
  - **k-Wave** (MPI+OpenMP) on *Salomon*: load-balancing in FFTW OpenMP parallel regions
  - **ICON** (MPI) on *Mistral*: automatic trace analysis of critical path of execution
  - **PIConGPU** (MPI+CUDA): computation offload to multiple attached accelerator devices
  - **TensorFlow** (Python+CUDA): interpreted & compiled heterogeneous execution measurement

Review

Markus Geimer
Jülich Supercomputing Centre

Summary

You've been introduced to a variety of tools

- with hints to apply and use the tools effectively

Tools provide complementary capabilities

- computational kernel & processor analyses
- communication/synchronization analyses
- load-balance, scheduling, scaling, …

Tools are designed with various trade-offs

- general-purpose versus specialized
- platform-specific versus agnostic
- simple/basic versus complex/powerful

Tool selection

Which tools you use and when you use them likely to depend on the situation

- which are available on (or for) your computer system
- which support your programming paradigms and languages
- which you are familiar (comfortable) with using
- which type of issue you suspect
- which question you want to have answered

Being aware of (potentially) available tools and their capabilities can help finding the most appropriate tools

Workflow (getting started)

First ensure that the parallel application runs correctly

- no-one will care how quickly you can get invalid answers or produce a set of corefiles
- parallel debuggers help isolate known problems
- correctness checking tools can identify other issues
- (that might not cause problems right now, but will eventually)
  - e.g., race conditions, invalid/non-compliant usage

Best to start with an overview of execution performance

- fraction of time spent in computation vs comm/synch vs I/O
- which sections of the application/library code are most costly
- Example profilers: **Score-P + Cube/ParaProf, TAU**

and how it changes with scale or different configurations

- processes vs threads, mappings, bindings

Workflow (communication/synchronization)

Communication issues generally apply to every computer system (to different extents) and typically grow with the number of processes/threads

- Weak scaling: fixed computation per thread, and perhaps fixed localities, but increasingly distributed
- Strong scaling: constant total computation, increasingly divided amongst threads, while communication grows
- Collective communication (particularly of type “all-to-all”) result in increasing data movement
- Synchronizations of larger groups are increasingly costly
- Load-balancing becomes increasingly challenging, and imbalances more expensive
  - generally manifests as waiting time at following collective ops

Workflow (wasted waiting time)

Waiting times are difficult to determine in basic profiles

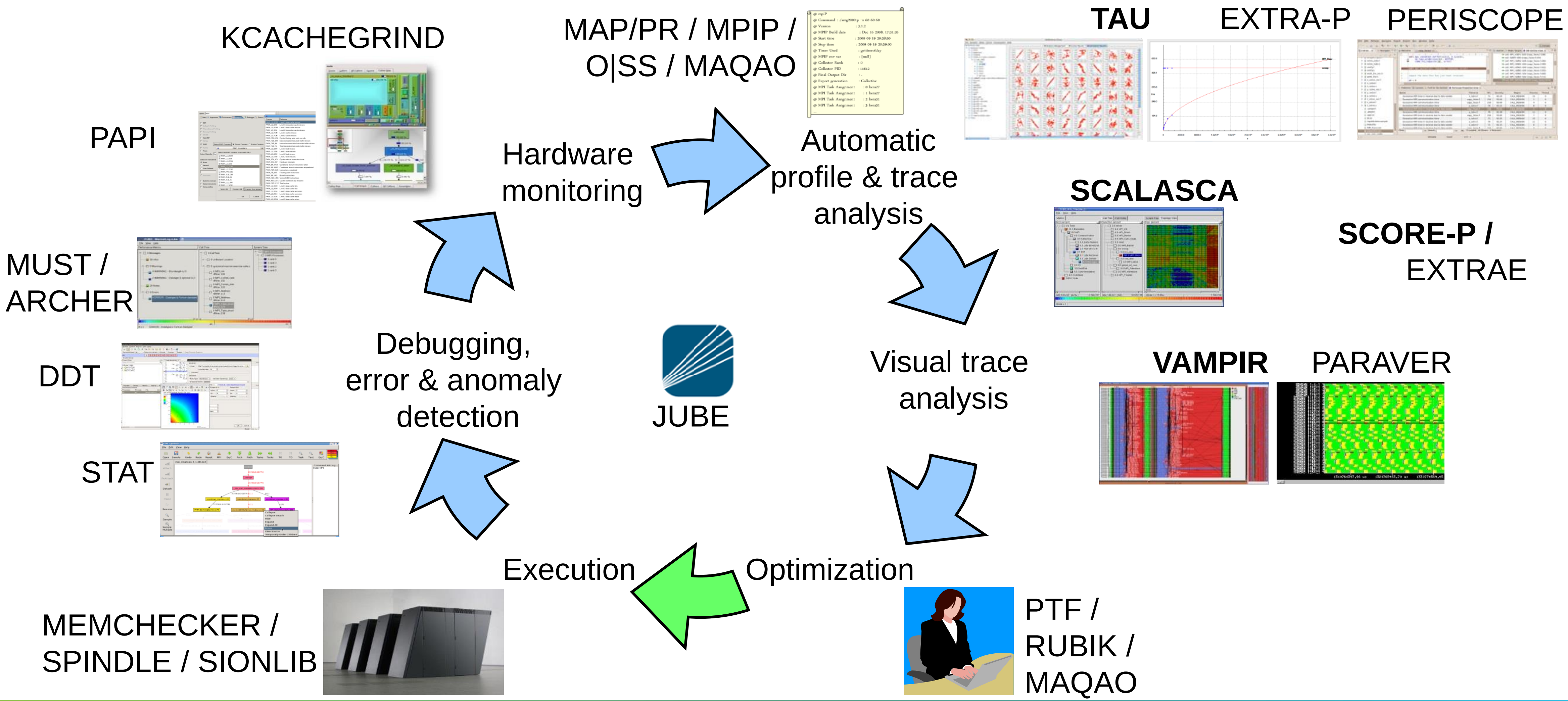
- Part of the time each process/thread spends in communication & synchronization operations may be wasted waiting time
- Need to correlate event times between processes/threads
  - Periscope uses augmented messages to transfer timestamps plus on-line analysis processes
  - Post-mortem event trace analysis avoids interference and provides a complete history
  - **Scalasca** automates trace analysis and ensures waiting times are completely quantified
  - **Vampir** allows interactive exploration and detailed examination of reasons for inefficiencies

Workflow (core computation)

Effective computation within processors/cores is also vital

- Optimized libraries may already be available
- Optimizing compilers can also do a lot
  - provided the code is clearly written and not too complex
  - appropriate directives and other hints can also help
- Processor hardware counters can also provide insight
  - although hardware-specific interpretation required
- Tools available from processor and system vendors help navigate and interpret processor-specific performance issues

Technologies and their integration



Further information

Website

- Introductory information about the VI-HPS portfolio of tools for high-productivity parallel application development
  - VI-HPS Tools Guide
  - links to individual tools sites for details and download
- Training material
  - tutorial slides
  - latest ISO image of VI-HPS Linux DVD with productivity tools
  - user guides and reference manuals for tools
- News of upcoming events
  - tutorials and workshops
  - mailing-list sign-up for announcements

http://www.vi-hps.org

Acknowledgement

Many thanks to the

Texas Advanced Computing Center (TACC)

for sponsoring this tutorial by providing
training accounts and compute time!