UNIVERSITÉ DE VERSAILLES ST-QUENTIN-EN-YVELINES

Virtual Institute – High Productivity Supercomputing

23 April 2012

Brian Wylie
Jülich Supercomputing Centre
b.wylie@fz-juelich.de

9th VI-HPS Tuning Workshop (UVSQ)

- Presenters
  - Jean-Baptiste Besnard (CEA)
  - Jens Doleschal (TU Dresden)
  - Thomas Guillet (Intel)
  - Emmanuel Oseret & Andres Charif-Rubial (UVSQ)
  - Ventsislav Petkov & Josef Wiedendorfer (TU München)
  - Marc Schlütter & Brian Wylie (Jülich Supercomputing Centre)
  - Sameer Shende (University of Oregon PRL)
  - Jan Treibig (RRZ Erlangen)
- Thanks
  - Local arrangements & facilities
    - Bettina Krammer (UVSQ)
    - Systems: UVSQ & CEA/TGCC
  - Sponsors: ECR, CEA, GENCI, Intel

Outline

Monday 23 April

- 09:00 (early registration & set-up, individual preparation)
- 12:00-13:00 (lunch)
- 13:00-13:30 (registration)
- Welcome & introduction to VI-HPS
- Introduction to parallel performance analysis
- 15:00-15:30 (break)
- Overview of VI-HPS tools
- Lab setup
- 17:30 (adjourn)

Outline of rest of week

Tuesday 24 April
- 09:00-10:30 **Scalasca**
- 11:00-12:30 **Vampir**

Wednesday 25 April
- 09:00-10:30 **TAU**
- 11:00-12:30 **Periscope**

Thursday 26 April
- 09:00-10:30 **Kcachegrind**
- 11:00-12:30 **MAQAO**
- 13:30-14:30 **Intel tools**

Friday 27 April
- 09:00-09:45 **MPC**
- 09:45-10:30 **Likwid**

- Hands-on exercises part of each tool presentation every morning session
- Hands-on coaching to apply tools to analyse & tune your own codes on workshop HPC systems each afternoon to 17:30
- Social dinner on Wednesday evening

Participant survey

We'd like to know a little about you, your application(s), and your expectations and desires from this tutorial

- What programming paradigms do you use in your app(s)?
  - only MPI, only OpenMP, mixed-mode/hybrid OpenMP/MPI, ...
  - Fortran, C, C++, multi-language, ...
- What platforms/systems *must* your app(s) run well on?
  - Cray XT/XE/XK, IBM BlueGene, SGI Altix, Linux cluster™, ...
- Who's already familiar with *serial* performance analysis?
  - Which tools have you used?
    - time, print/printf, prof/gprof, VTune, ...
- Who's already familiar with *parallel* performance analysis?
  - Which tools have you used?
    - time, print/printf, prof/gprof, Periscope, Scalasca, TAU, Vampir, ...

Prepare to analyse your own application codes

- Ensure your application codes build and run to completion with appropriate datasets
  - initial configuration should ideally run in less than 15 minutes with 1-4 compute nodes (up to 64 processes/threads)
    - to facilitate rapid turnaround and quick experimentation
  - larger/longer scalability configurations are also interesting
    - turnaround may be limited due to busyness of batch queues
- Compare your application performance on other systems
  - VI-HPS tools already installed on a number of HPC systems
    - if not, ask your system administrator to install them (or install a personal copy yourself)

Virtual Institute – High Productivity Supercomputing

VI-HPS

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
  - Training workshops
  - Service
    - Support email lists
    - Application engagement
  - Academic workshops

www.vi-hps.org

VI-HPS partners (founders)




- Forschungszentrum Jülich
  - Jülich Supercomputing Centre
- RWTH Aachen University
  - Centre for Computing & Communication
- Technical University of Dresden
  - Centre for Information Services & HPC
- University of Tennessee (Knoxville)
  - Innovative Computing Laboratory

VI-HPS partners (cont.)





German Research School
- Laboratory of Parallel Programming

Technical University of Munich
- Chair for Computer Architecture

University of Oregon
- Performance Research Laboratory

University of Stuttgart
- HPC Centre

University of Versailles St-Quentin
- LRC ITACA

Barcelona Supercomputing Center
- Centro Nacional de Supercomputación

Lawrence Livermore National Laboratory
- Centre for Applied Scientific Computing

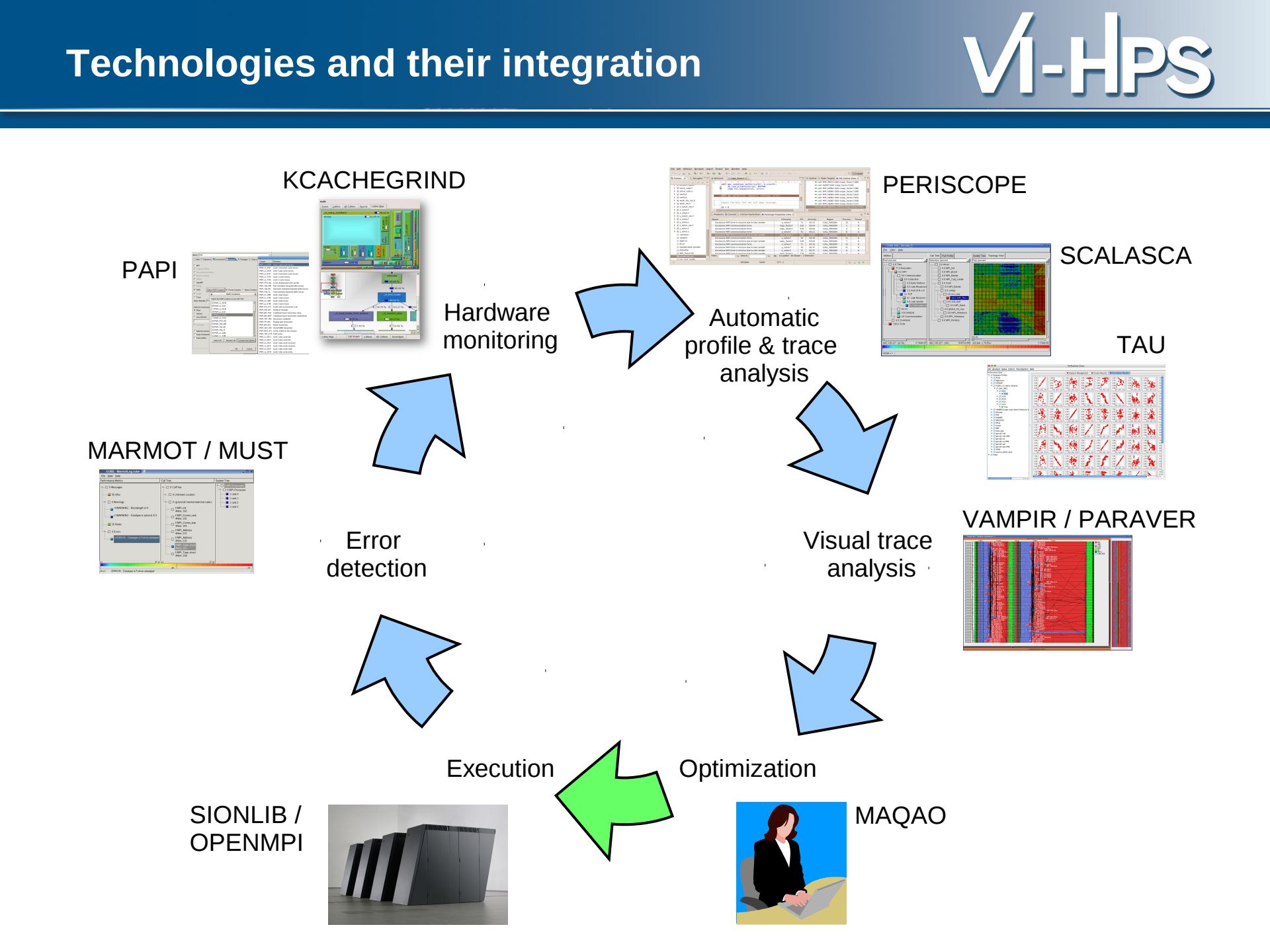
VI-HPS productivity tools

- Marmot/MUST
  - MPI correctness checking
- PAPI
  - Interfacing to hardware performance counters
- Periscope
  - Automatic analysis driven by on-line distributed search
- Scalasca
  - Large-scale parallel performance analysis
- TAU
  - Integrated parallel performance system
- Vampir/VampirTrace
  - Event tracing and graphical trace visualization & analysis

VI-HPS productivity tools (cont.)

- KCachegrind
  - Callgraph-based cache analysis [x86 only]
- MAQAO
  - Assembly instrumentation & optimization [x86 only]
- ompP
  - OpenMP profiling tool
- OpenMPI
  - Memory checking
- Paraver/Extrae
  - Event tracing and graphical trace visualization & analysis
- Score-P
  - Common instrumentation & measurement infrastructure
- SIONlib
  - Optimized native parallel file I/O

Technologies and their integration



VI-HPS Training & Tuning Workshops

- Goals
  - Give an overview of the programming tools suite
  - Explain the functionality of individual tools
  - Teach how to use the tools effectively
  - Offer hands-on experience and expert assistance using tools
  - Receive feedback from users to guide future development
- For best results, bring & analyse/tune your own code(s)!

- VI-HPS Tutorial series
  - SC'08, ICCS'09, SC'09, Cluster'10, SC'10, SC'11
- VI-HPS Tuning Workshop series
  - 2008 (Aachen & Dresden), 2009 (Jülich & Bremen), 2010 (Garching & Amsterdam), 2011 (Stuttgart & Aachen)
  - **2012/04/23-27 (St-Quentin)**, 2012/10/15-19 (Garching)

Upcoming VI-HPS training events

- PRACE Spring School (16-18 May 2012)
  - Krakow, Poland
  - One day hands-on tutorial with Scalasca + Vampir
- EU/US HPC Summer School (25-28 June 2012)
  - Dublin, Ireland
  - Half-day hands-on tutorial with PAPI, Scalasca, TAU, Vampir
- EuroMPI conference hands-on tutorial (23 Sep 2012)
  - Vienna, Austria
  - Periscope, Scalasca, TAU, Vampir (using Score-P)

Check www.vi-hps.org/training for announced events

- Contact us if you might be interested in hosting an event

Upcoming VI-HPS training events

- 10th VI-HPS Tuning Workshop (16-19 Oct 2012)
  - hosted by LRZ, Garching-bei-Muenchen, Germany
  - using PRACE Tier-0 *SuperMUC* iDataPlex system
  - Scalasca, Vampir, TAU, Periscope, KCachegrind, MAQAO, ...

- Further events to be determined
  - (one-day) tutorials
    - with guided exercises usually using Live DVD
  - (multi-day) training workshops
    - with your own applications on real HPC systems

Check www.vi-hps.org/training for announced events

- Contact us if you might be interested in hosting an event

POINT/VI-HPS Live-ISO/DVD

- Bootable Linux installation ISO (on DVD or USB drive)
- Includes everything needed to try out our parallel tools on an x86-architecture notebook computer
  - VI-HPS tools: KCachegrind, Marmot, PAPI, Periscope, Scalasca, TAU, VT/Vampir*
  - Also: Eclipse/PTP, TotalView*, etc.
    - * time/capability-limited evaluation licences provided for commercial products
  - GCC (w/ OpenMP), OpenMPI
  - Manuals/User Guides
  - Tutorial exercises & examples
- Produced by U. Oregon PRL
  - Sameer Shende

POINT/VI-HPS Live-ISO/DVD

- ISO image approximately 4GB
  - distributed on DVD or USB drive
  - or download from website
- Boot directly from disk
  - enables hardware counter access and offers best performance
- Boot within virtual machine
  - faster boot time and can save/resume state, but no hardware counter access
- Boots into Linux environment
  - supports building and running provided MPI and/or OpenMP parallel application codes
  - and experimentation with VI-HPS (and other) tools

KCachegrind

Cachegrind: cache analysis by simple cache simulation

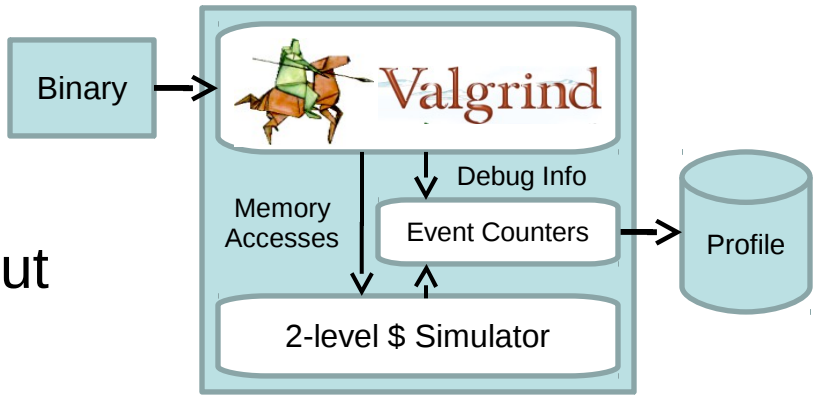
- Captures dynamic callgraph
- Based on valgrind dynamic binary instrumentation
- Runs on x86/PowerPC/ARM unmodified binaries
  - No root access required
- ASCII reports produced

[KQ]Cachegrind GUI

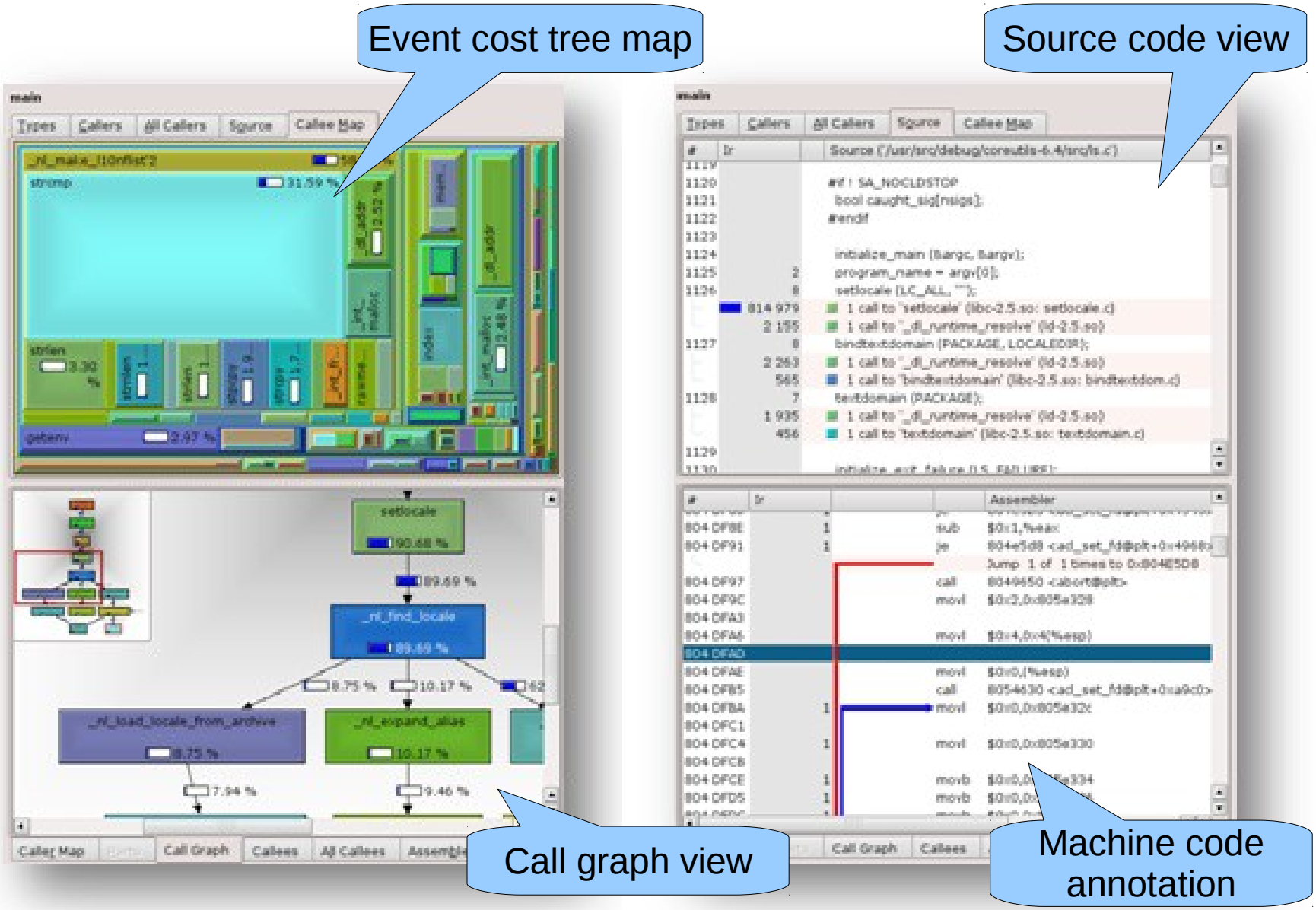
- Visualization of cachegrind output

Developed by TU Munich

- Released as GPL open-source
- http://kcachegrind.sf.net/



KCachegrind GUI



Marmot

Tool to check for correct MPI usage at runtime

- Checks conformance to MPI standard
  - Supports Fortran & C bindings of MPI-1.2
- Checks parameters passed to MPI
- Monitors MPI resource usage

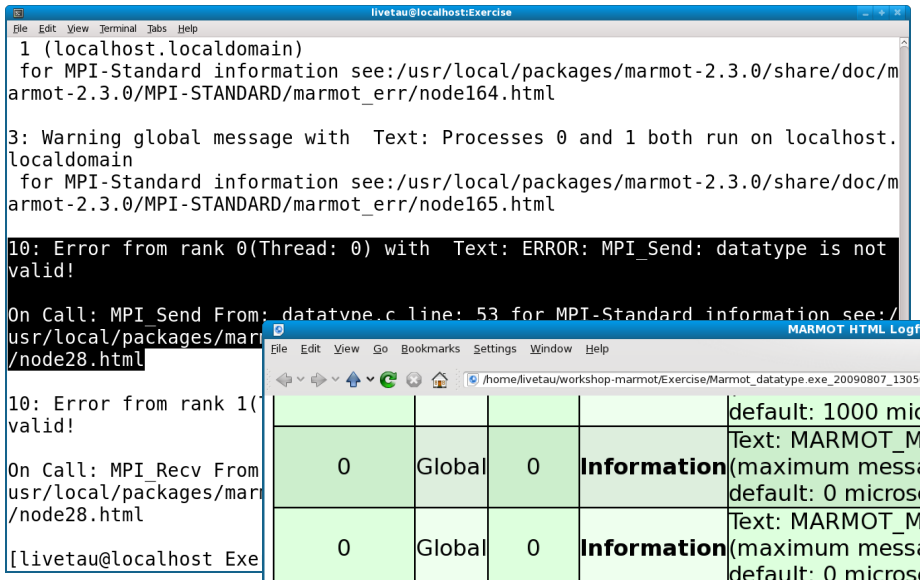
Implementation

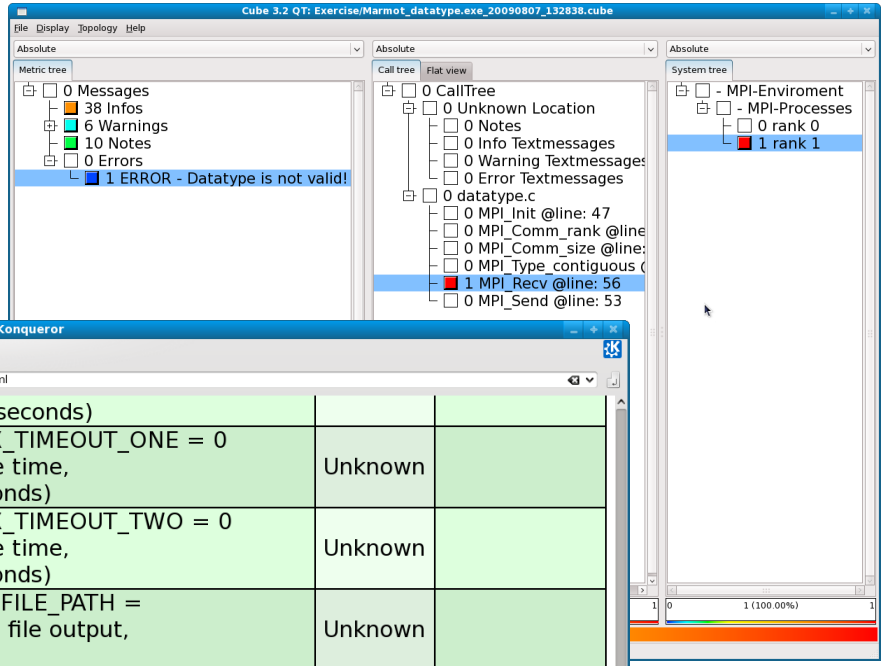
- C++ library gets linked to the application
- Does not require source code modifications
- Additional process used as DebugServer
- Results written in a log file (ASCII/HTML/CUBE)

Developed by HLRS & TU Dresden

- Released as open-source
- http://www.hlrs.de/organization/av/amt/projects/marmot

Marmot logfiles





MARMOT HTML Logfile - Konqueror

File Edit View Go Bookmarks Settings Window Help

/home/livetau/workshop-marmot/Exercise/Marmot_datatype.exe_20090807_130509.html

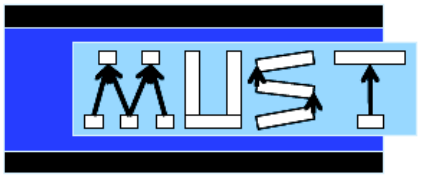
				default: 1000 microseconds)		
0	Global	0	**Information**	Text: MARMOT_MAX_TIMEOUT_ONE = 0 (maximum message time, default: 0 microseconds)	Unknown	
0	Global	0	**Information**	Text: MARMOT_MAX_TIMEOUT_TWO = 0 (maximum message time, default: 0 microseconds)	Unknown	
0	Global	0	**Information**	Text: MARMOT_LOGFILE_PATH = (path of Marmot log file output, default:)	Unknown	
0	Global	0	**Information**	Text: MARMOT_ERRCODES_SET = (not set) (not functional yet)	Unknown	
0	Global	0	**Information**	Text: End of the environmental variables info.	Unknown	
0	Global	0	**Information**	Text: Thread Syncronisation is disabled.If you are using multiple threads errors might occur	Unknown	
3	Global	0	**Warning**	Text: Debugserver runs on same node as process 0 (localhost.localdomain)	Unknown	Infos see MPI-Standard
3	Global	0	**Warning**	Text: Debugserver runs on same node as process 1 (localhost.localdomain)	Unknown	Infos see MPI-Standard
3	Global	0	**Warning**	Text: Processes 0 and 1 both run on localhost.localdomain	Unknown	Infos see MPI-Standard
10	0	0	**Error**	Text: ERROR: MPI_Send: datatype is not valid! Call: MPI_Send	datatype.c line: 53	Infos see MPI-Standard
10	1	0	**Error**	Text: ERROR: MPI_Recv: datatype is not valid! Call: MPI_Recv	datatype.c line: 56	Infos see MPI-Standard

Next generation MPI runtime error detection tool

- Successor of the Marmot and Umpire tools
- Initial merge of Marmot's many local checks with Umpire's non-local checks
- Improved scalability expected in future
- Exploits CMake, GTI & PnMPI infrastructure

Developed by TU Dresden, LLNL & LANL

- BSD license open-source initial release in November 2011
- http://tu-dresden.de/zih/must/



Portable performance counter library & utilities

- Configures and accesses hardware/system counters
- Predefined events derived from available native counters
- Core component for CPU/processor counters
  - instructions, floating point operations, branches predicted/taken, cache accesses/misses, TLB misses, cycles, stall cycles, …
  - performs transparent multiplexing when required
- Extensible components for off-processor counters
  - InfiniBand network, Lustre filesystem, system hardware health, …
- Used by multi-platform performance measurement tools
  - Periscope, Scalasca, TAU, VampirTrace, ...

Developed by UTK-ICL

- Available as open-source for most modern processors
  http://icl.cs.utk.edu/papi/

PAPI preset counters (and their definitions)

```
juropa$ papi_avail
Available events and hardware information.
-------------------------------------------------
PAPI Version             : 4.1.0.0
Vendor string and code   : GenuineIntel (1)
Model string and code    : Intel(R) Xeon(R) CPU
                           X5570 @ 2.93GHz (26)
CPU Revision             : 5.000000
CPUID Info               : Family: 6  Model: 26
                           Stepping: 5
CPU Megahertz            : 1600.000000
CPU Clock Megahertz      : 1600
Hdw Threads per core     : 2
Cores per Socket         : 4
NUMA Nodes               : 2
CPU's per Node           : 8
Total CPU's              : 16
Number Hardware Counters : 16
Max Multiplex Counters   : 512
-------------------------------------------------
    Name        Code    Avail Deriv Description
PAPI_L1_DCM  0x80000000  Yes   No
                     Level 1 data cache misses
PAPI_L1_ICM  0x80000001  Yes   No
               Level 1 instruction cache misses
...
-------------------------------------------------
Of 107 possible events, 35 are available, of
which 9 are derived.
```

```
juropa$ papi_avail -d
...
Symbol       Event Code  Count   |Short Descr.|
 |Long Description|
 |Developer's Notes|
 |Derived|
 |PostFix|
 Native Code[n]: <hex> |name|
PAPI_L1_DCM    0x80000000  1 |L1D cache misses|
 |Level 1 data cache misses|
 ||
 |NOT_DERIVED|
 ||
 Native Code[0]: 0x40002028 |L1D:REPL|
PAPI_L1_ICM    0x80000001  1 |L1I cache misses|
 |Level 1 instruction cache misses|
 ||
 |NOT_DERIVED|
 ||
 Native Code[0]: 0x40001031 |L1I:MISSES|
PAPI_L2_DCM    0x80000002  2 |L2D cache misses|
 |Level 2 data cache misses|
 ||
 |DERIVED_SUB|
 ||
 Native Code[0]: 0x40000437 |L2_RQSTS:MISS|
 Native Code[1]: 0x40002037 |
L2_RQSTS:IFETCH_MISS|

...
```

PAPI native counters (and qualifiers)

```
juropa$ papi_native_avail
Available native events and hardware information.
...
Event Code   Symbol  | Long Description |
--------------------------------------------------------------------------------
0x40000000   UNHALTED_CORE_CYCLES  | count core clock cycles whenever the cloc |
           | k signal on the specific core is running (not halted). Alias to e |
           | vent CPU_CLK_UNHALTED:THREAD                                      |
--------------------------------------------------------------------------------
0x40000001   INSTRUCTION_RETIRED  | count the number of instructions at retire |
           | ment. Alias to event INST_RETIRED:ANY_P                           |
--------------------------------------------------------------------------------
...
--------------------------------------------------------------------------------
0x40000086   UNC_SNP_RESP_TO_REMOTE_HOME  | Remote home snoop response - LLC d |
           | oes not have cache line                                           |
  40000486   :I_STATE  | Remote home snoop response - LLC does not have cache  |
           | line                                                              |
  40000886   :S_STATE  | Remote home snoop response - LLC has  cache line in S |
           |  state                                                            |
  40001086   :FWD_S_STATE  | Remote home snoop response - LLC forwarding cache |
           |  line in S state.                                                 |
  40002086   :FWD_I_STATE  | Remote home snoop response - LLC has forwarded a  |
           | modified cache line                                               |
  40004086   :CONFLICT  | Remote home conflict snoop response                   |
  40008086   :WB  | Remote home snoop response - LLC has cache line in the M s |
           | tate                                                              |
  40010086   :HITM  | Remote home snoop response - LLC HITM                     |
--------------------------------------------------------------------------------
Total events reported: 135
```

Periscope

Automated profile-based performance analysis

- Iterative on-line performance analysis
  - Multiple distributed hierarchical agents
- Automatic search for bottlenecks based on properties formalizing expert knowledge
  - MPI wait states, OpenMP overheads and imbalances
  - Processor utilization hardware counters
- Clustering of processes/threads with similar properties
- Eclipse-based integrated environment

Supports

- SGI Altix Itanium2, IBM Power and x86-based architectures

Developed by TU Munich

- Released as open-source
- http://www.lrr.in.tum.de/periscope

Periscope properties & strategies (examples)

MPI

- Excessive MPI communication time
- Excessive MPI time due to many small messages
- Excessive MPI time in receive due to late sender
- …

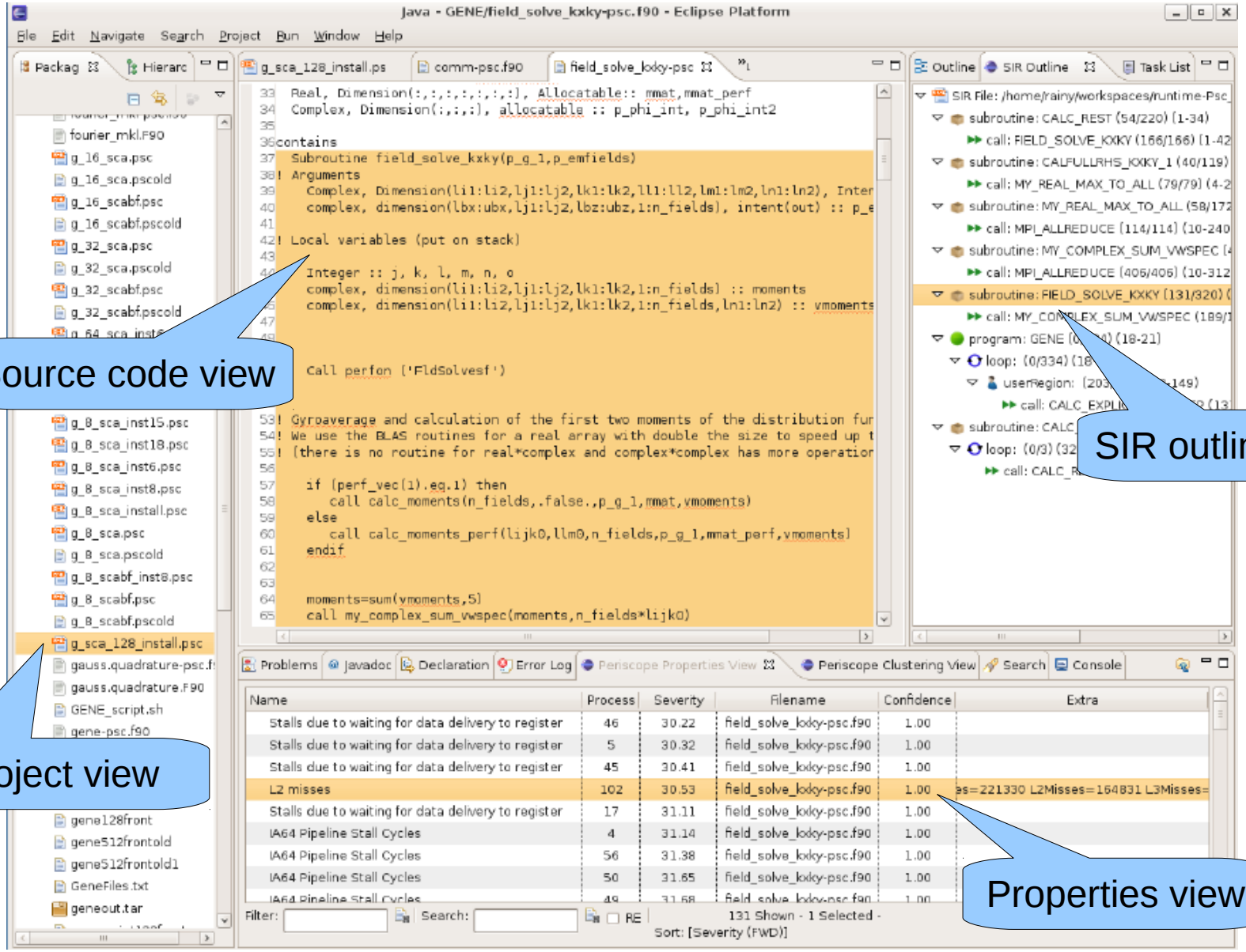
OpenMP

- Load imbalance in parallel region/section
- Sequential computation in master/single/ordered region
- ...

Hardware performance counters (platform-specific)

- Cycles lost due to cache misses
  - High L1/L2/L3 demand load miss rate
- Cycles lost due to no instruction to dispatch
- ...

Periscope plug-in to Eclipse environment



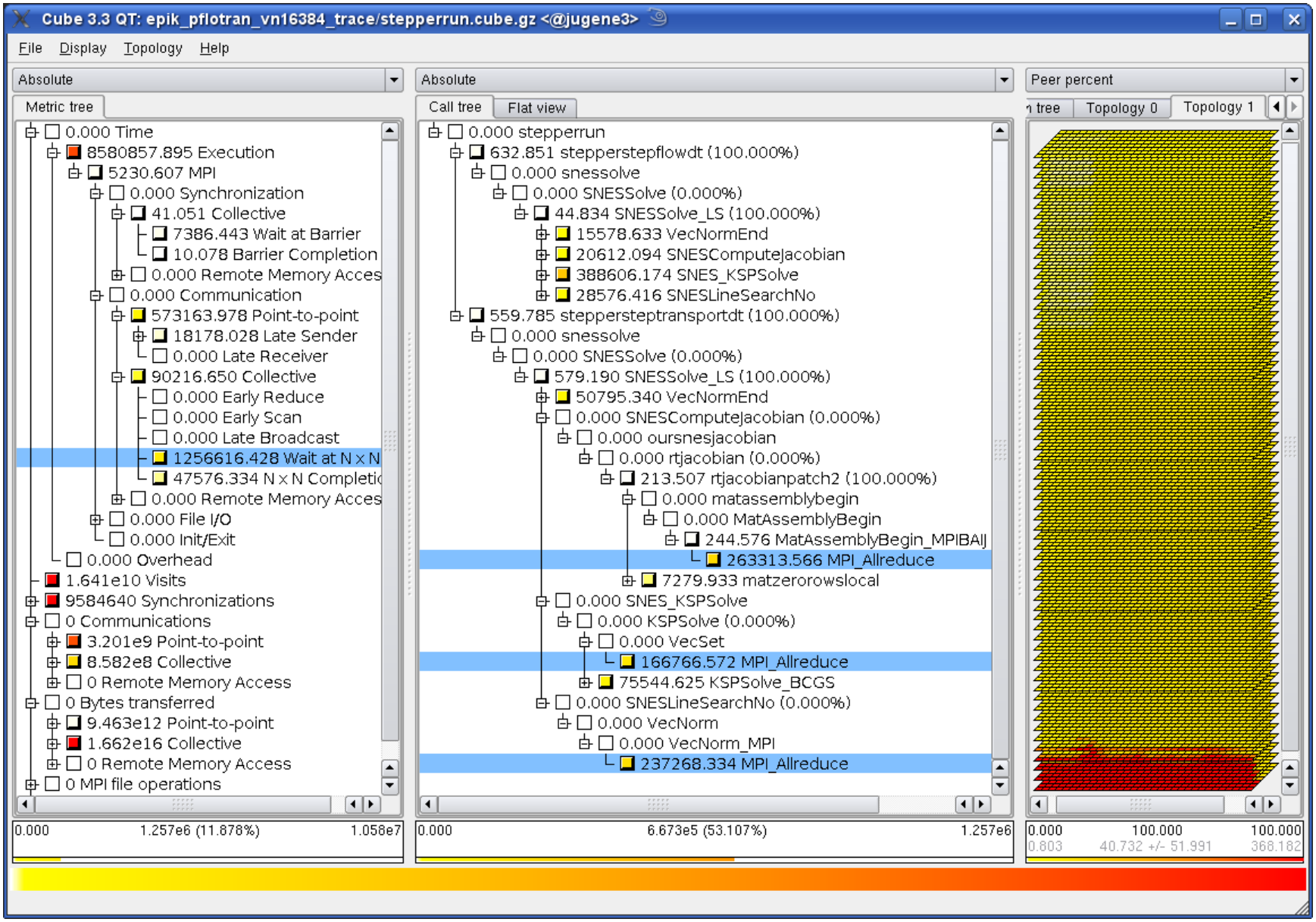
Automatic performance analysis toolset

- Scalable performance analysis of large-scale applications
  - particularly focused on MPI & OpenMP paradigms
  - analysis of communication & synchronization overheads
- Automatic and manual instrumentation capabilities
- Runtime summarization and/or event trace analyses
- Automatic search of event traces for patterns of inefficiency
  - Scalable trace analysis based on parallel replay
- Interactive exploration GUI and algebra utilities for XML callpath profile analysis reports

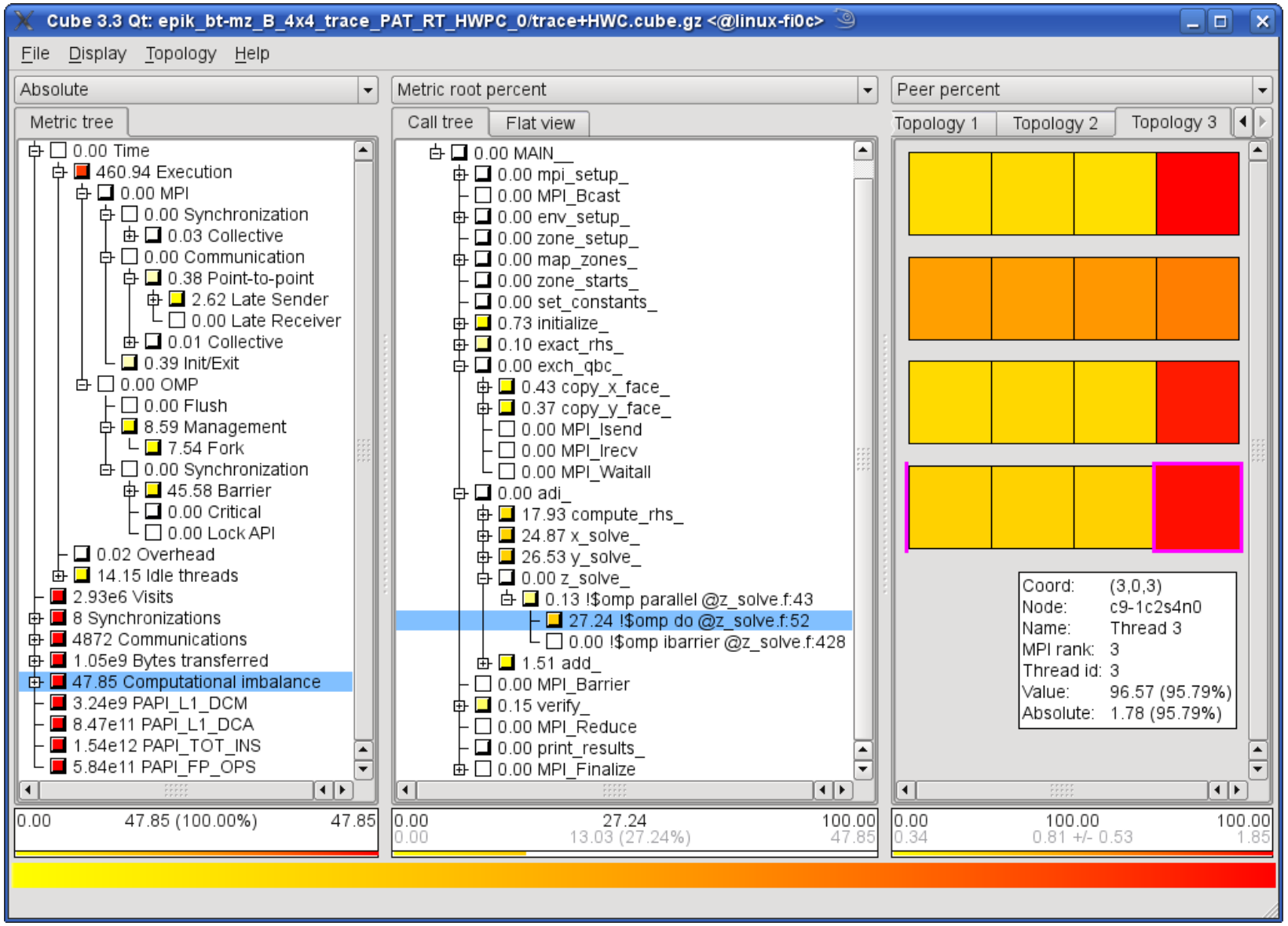
Developed by JSC & GRS

- Released as open-source
- http://www.scalasca.org/

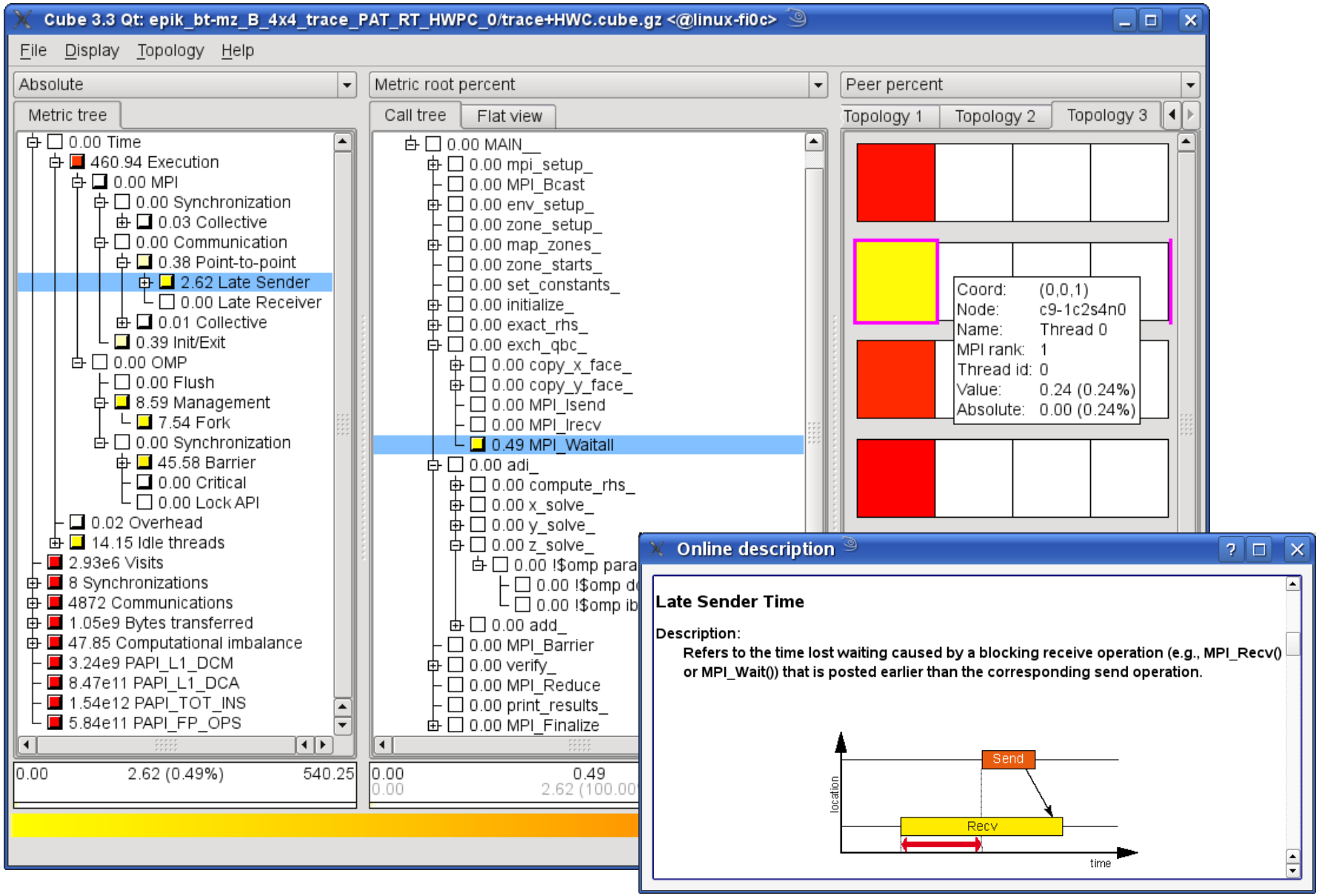
Scalasca automatic trace analysis report



Scalasca hybrid analysis report



Scalasca automatic trace analysis report



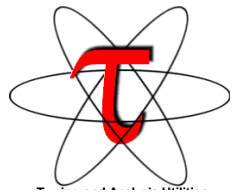
TAU Performance System

Integrated performance toolkit

- Instrumentation, measurement, analysis & visualization
  - Highly customizable installation, API, envvars & GUI
  - Supports multiple profiling & tracing capabilities
- Performance data management & data mining
- Targets all parallel programming/execution paradigms
  - Ported to a wide range of computer systems
- Performance problem solving framework for HPC
- Extensive bridges to/from other performance tools
  - PerfSuite, Scalasca, Vampir, ...

Developed by U. Oregon/PRL

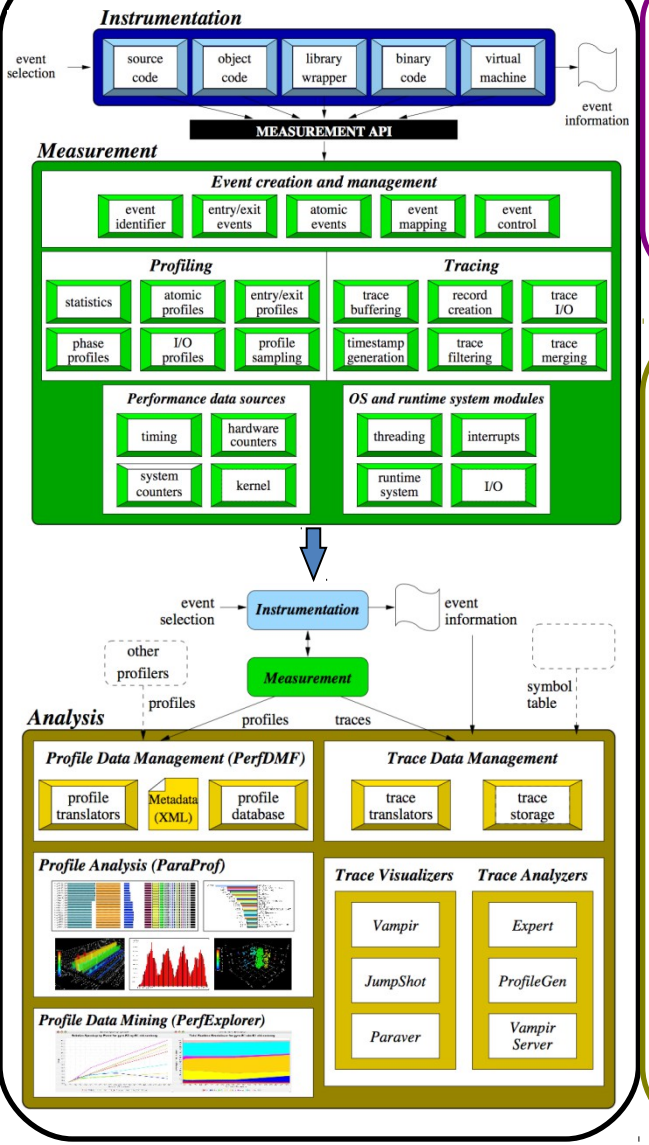
- Broadly deployed open-source software
- http://tau.uoregon.edu/



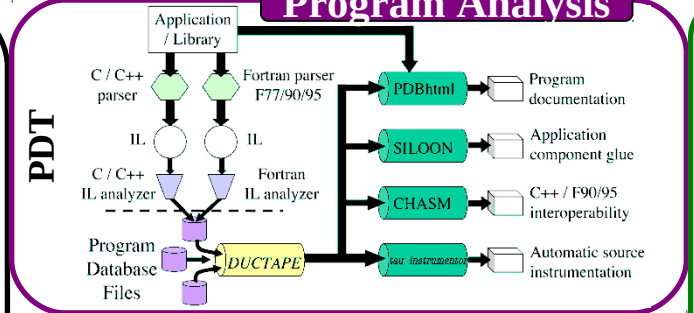
TAU Performance System components

VI-HPS

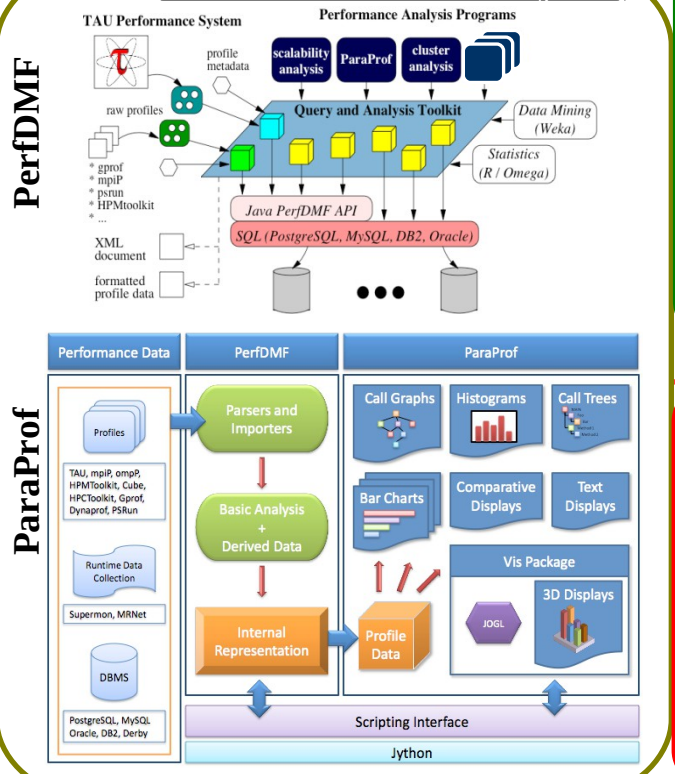
TAU Architecture



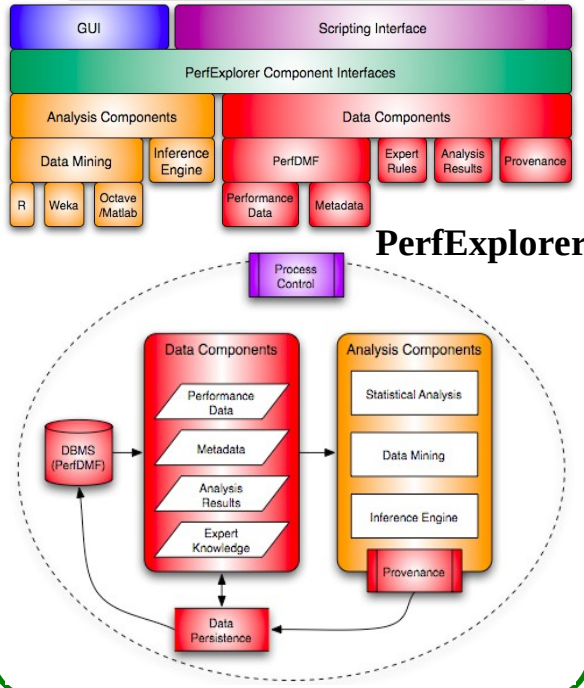
Program Analysis



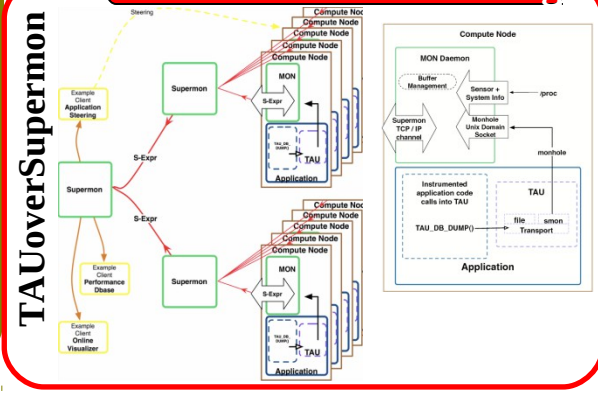
Parallel Profile Analysis



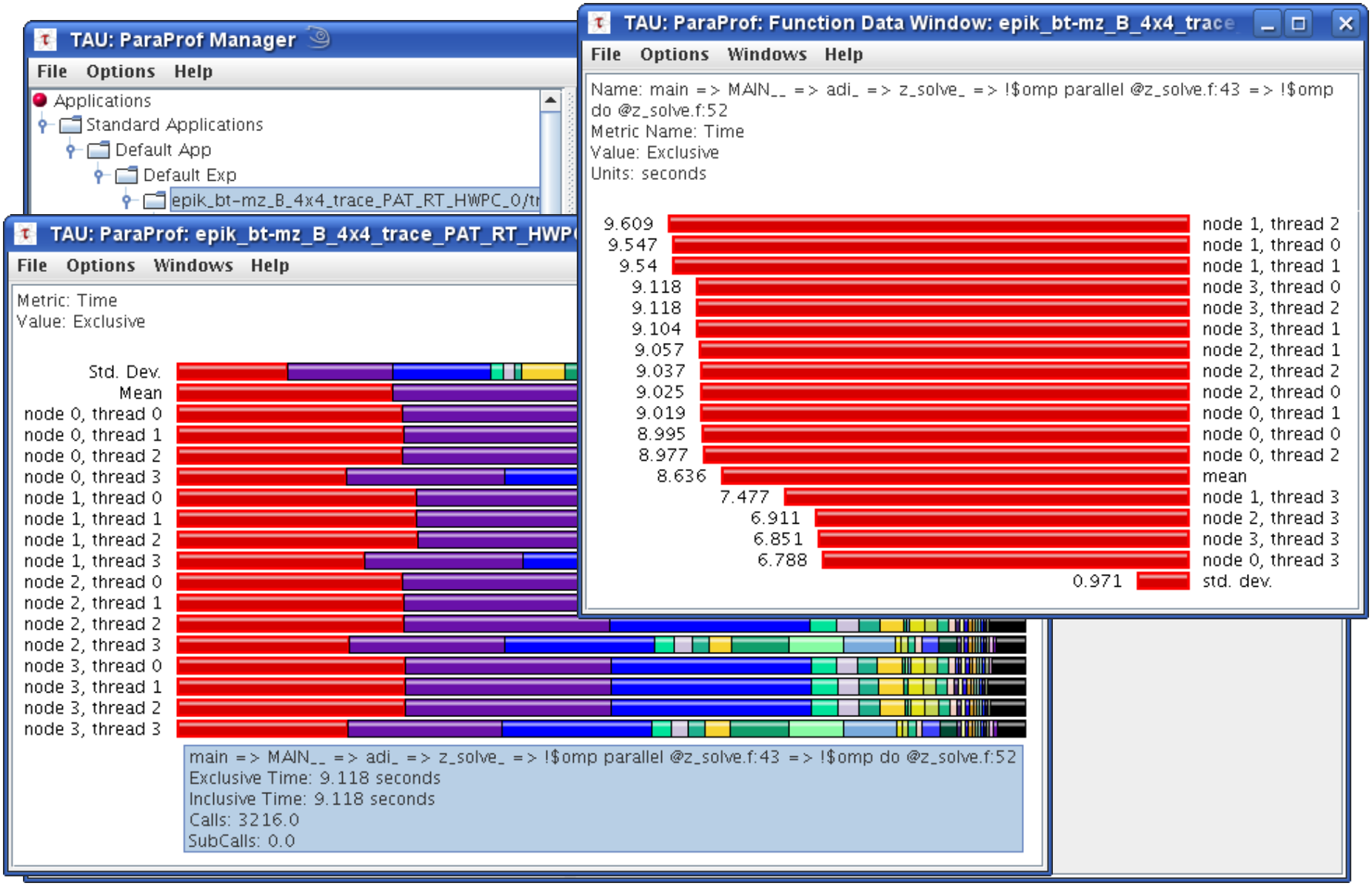
Performance Data Mining



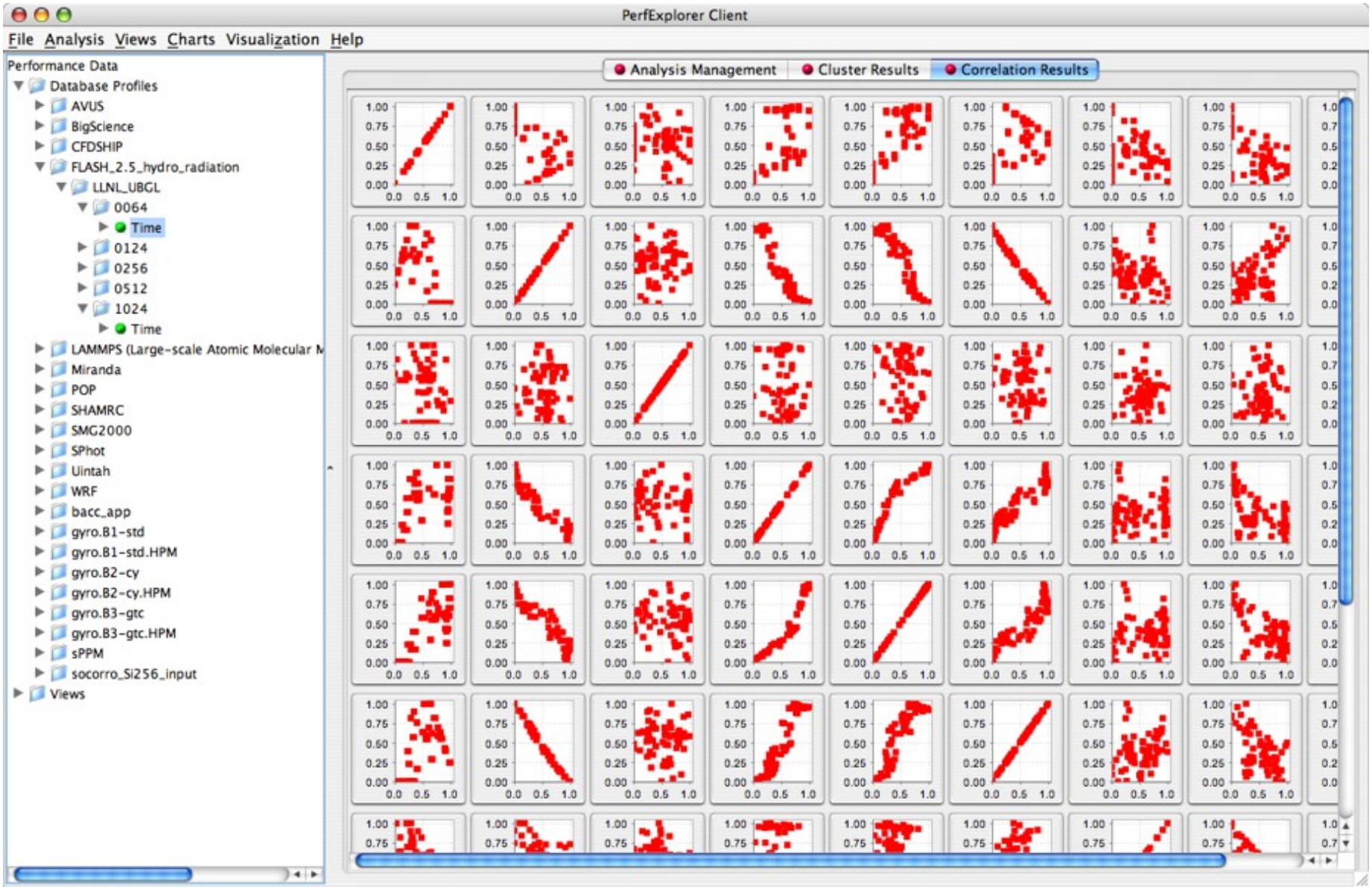
Performance Monitoring



TAU ParaProf GUI displays (selected)



TAU PerfExplorer data mining



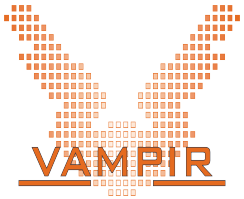
Vampir & VampirTrace

Interactive event trace analysis

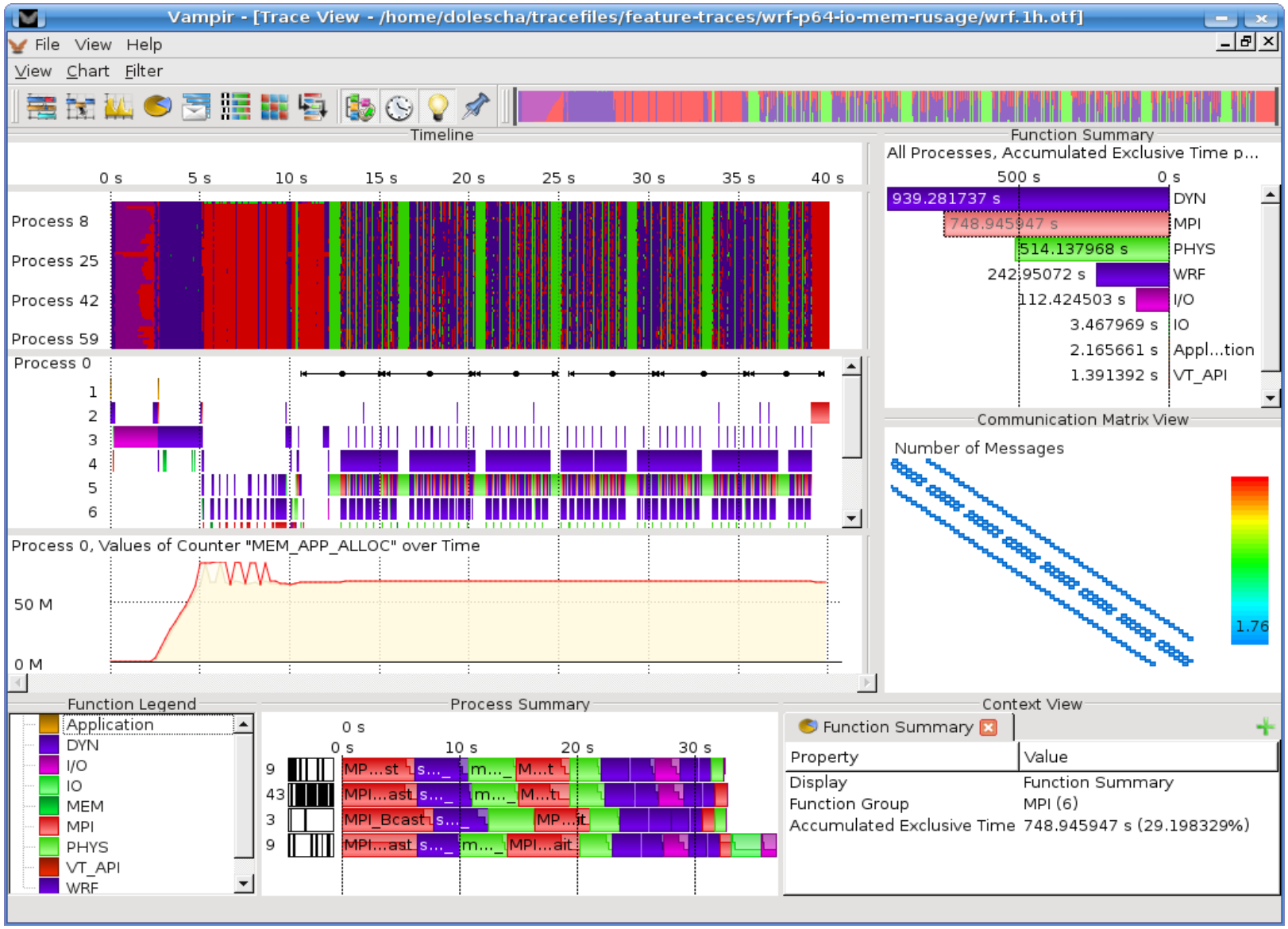
- Alternative & supplement to automatic trace analysis
- Visual presentation of dynamic runtime behaviour
  - event timeline chart for states & interactions of processes/threads
  - communication statistics, summaries & more
- Interactive browsing, zooming, selecting
  - linked displays & statistics adapt to selected time interval (zoom)
  - scalable server runs in parallel to handle larger traces

Developed by TU Dresden ZIH

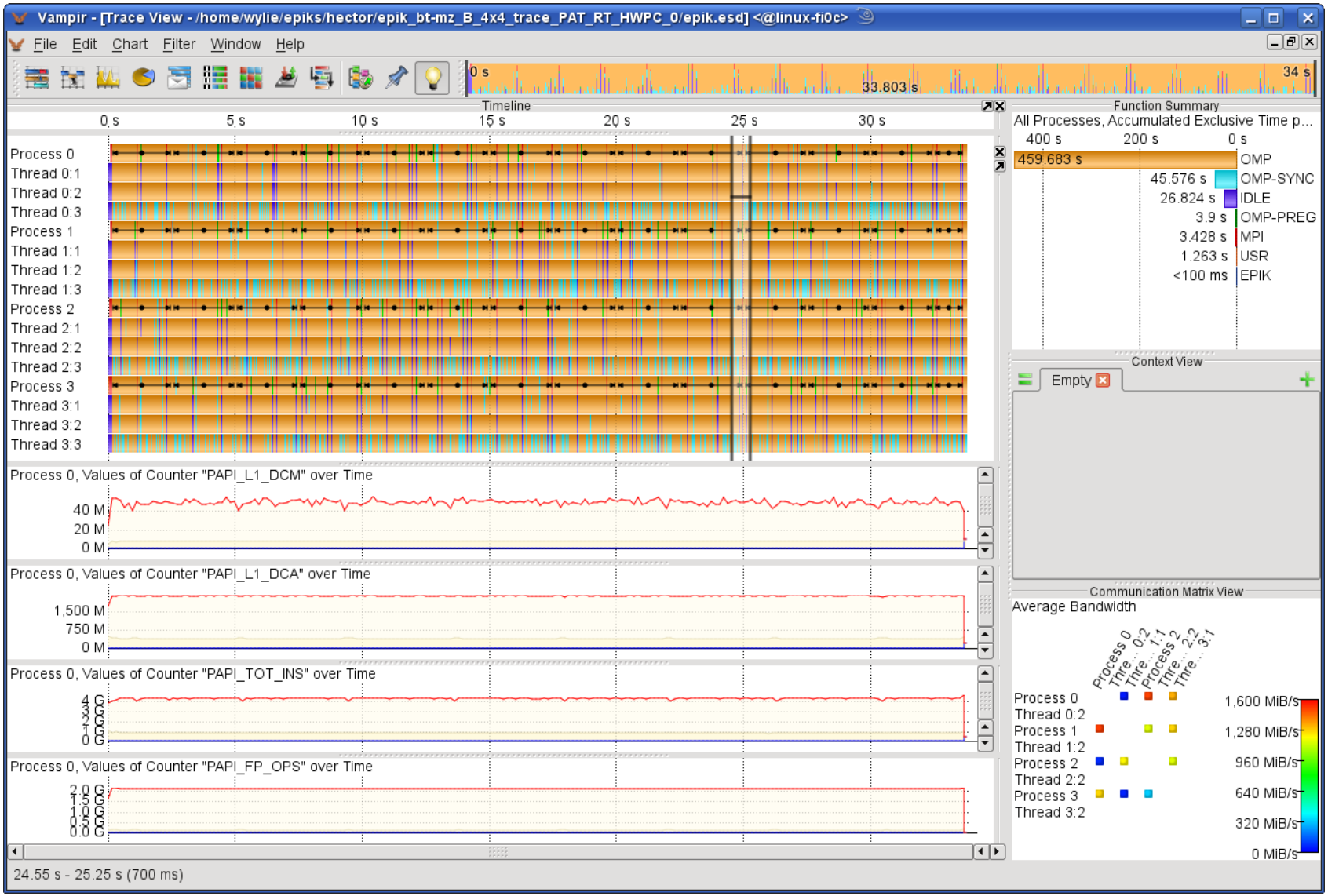
- Open-source VampirTrace library bundled with OpenMPI 1.3
- http://www.tu-dresden.de/zih/vampirtrace/
- Vampir Server & GUI have a commercial license
- http://www.vampir.eu/



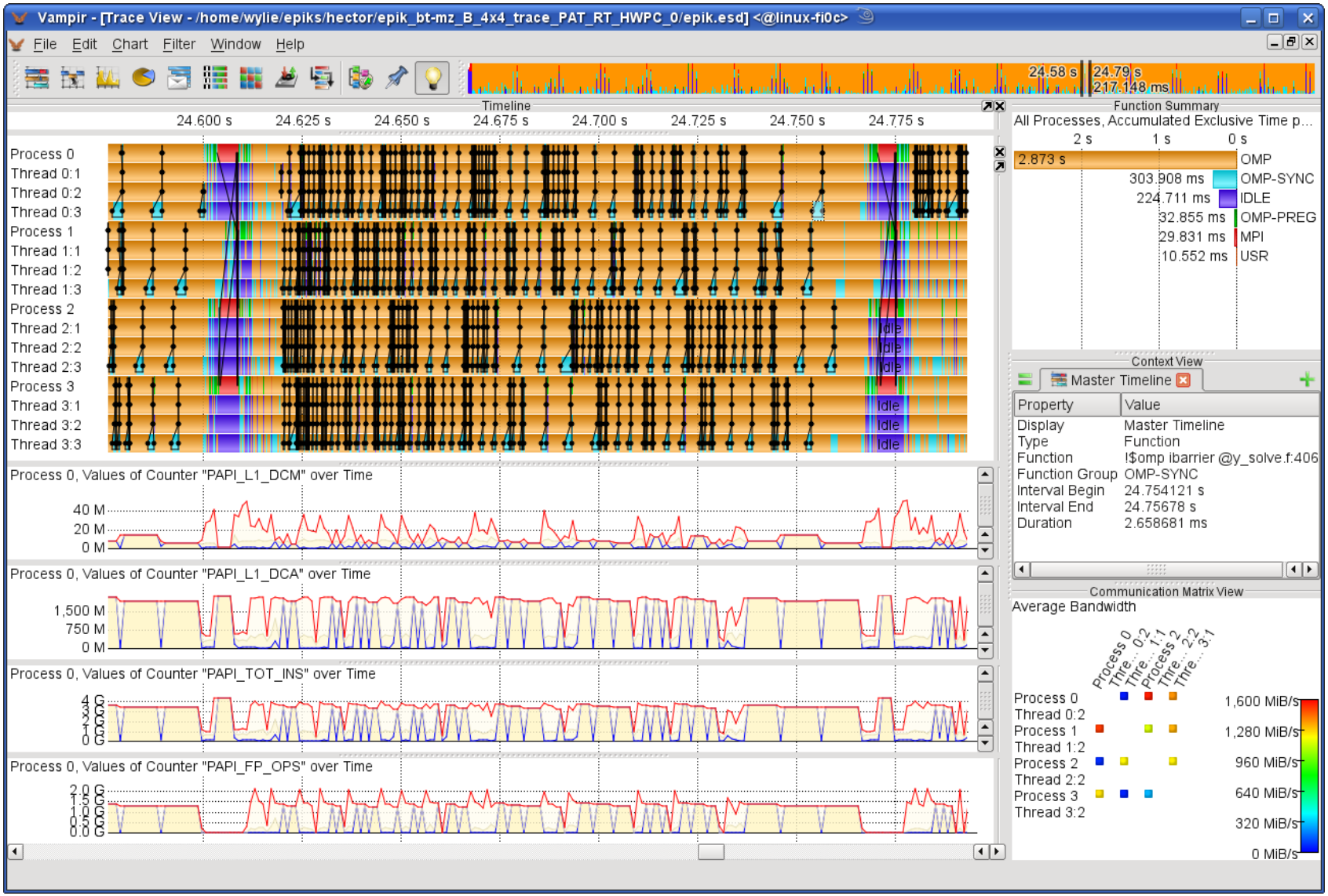
Vampir interactive trace analysis GUI



Vampir interactive trace analysis GUI



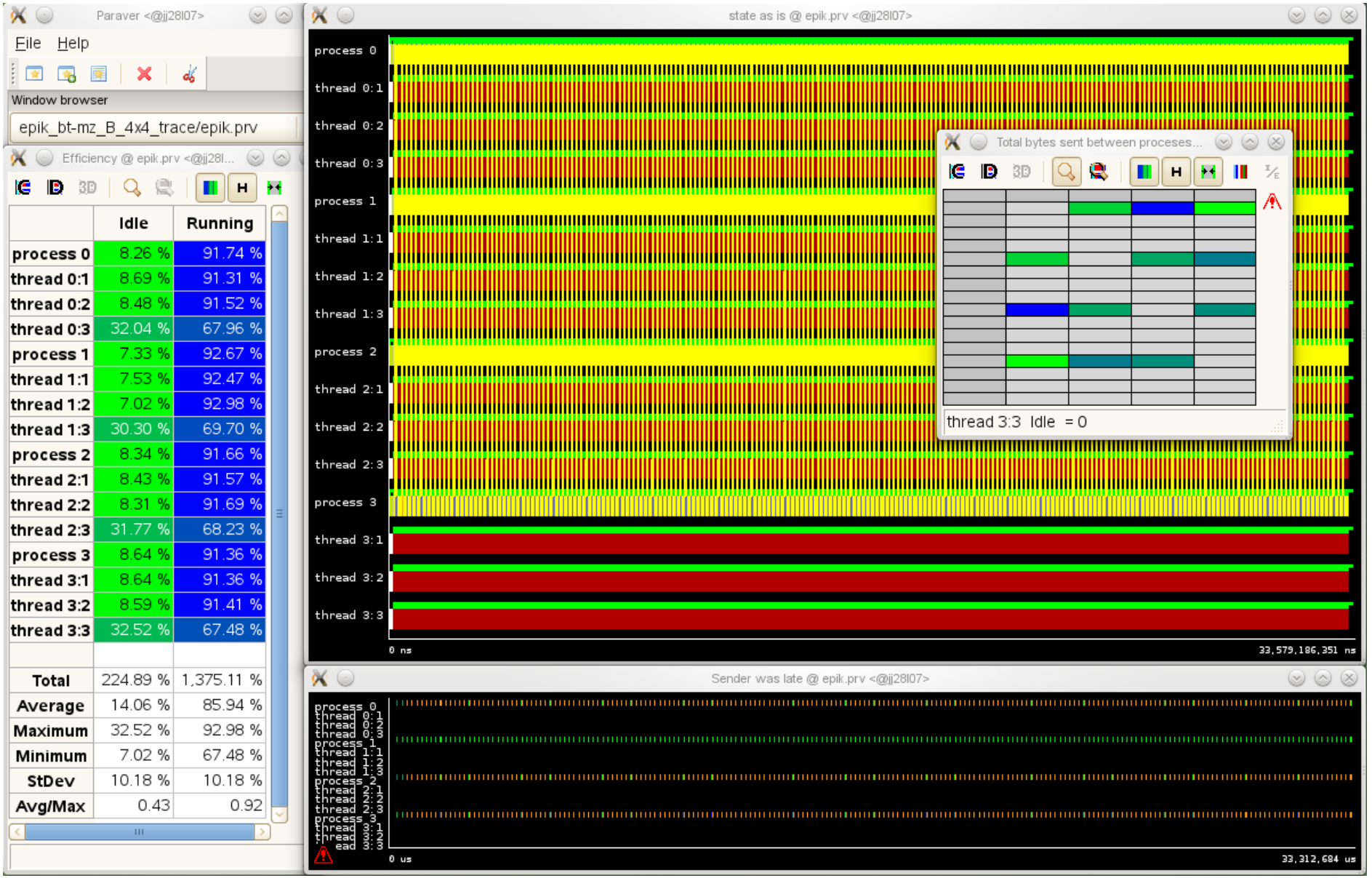
Vampir interactive trace analysis GUI (zoom)



Paraver & Extrae

- Interactive event trace analysis
  - Visual presentation of dynamic runtime behaviour
    - event timeline chart for states & interactions of processes
    - Interactive browsing, zooming, selecting
  - Large variety of highly configurable analyses & displays
- Developed by Barcelona Supercomputing Center
  - Paraver trace analyser and Extrae measurement library
  - Open source available from http://www.bsc.es/paraver/

Paraver interactive trace analysis GUI



- Modular Assembler Quality Analyzer & Optimizer
  - Framework for binary manipulation
    - using plugins and scripting language
  - Tool exploiting framework to produce reports
    - fast prototyping and batch interface
  - STAN static performance model
  - MIL instrumentation language for dynamic analysis
    - building custom performance evaluation tools using HWCs
    - instrumentation of functions, loops, blocks & instructions
- Developed by UVSQ Exascale Computing Research lab
  - Supports Intel x86_64 microarchitecture
  - Available from www.maqao.org

VI-HPS component technologies

Key tool components also provided as open-source

- Program development environment
  - Eclipse PTP ETFw, UNITE
- Program/library instrumentation
  - COBI, OPARI, PDToolkit
- Runtime measurement systems
  - Score-P, UniMCI
- Scalable I/O
  - SIONlib
- Libraries & tools for handling (and converting) traces
  - EPILOG, OTF, PEARL
- Analysis algebra & hierarchical/topological presentation
  - CUBE

Scalable performance measurement infrastructure

- Supports instrumentation, profiling & trace collection, as well as online analysis of HPC parallel applications
- Works with Periscope, Scalasca, TAU & Vampir prototypes
- Based on updated tool components
  - CUBE4 profile data utilities & GUI
  - OA online access interface to performance measurements
  - OPARI2 OpenMP & pragma instrumenter
  - OTF2 open trace format

Created by German BMBF SILC & US DOE PRIMA projects

- JSC, RWTH, TUD, TUM, GNS, GRS, GWT & UO PRL
- Available as BSD open-source from http://www.score-p.org/

Portable native parallel I/O library & utilities

- Scalable massively-parallel I/O to task-local files
- Manages single or multiple physical files on disk
  - optimizes bandwidth available from I/O servers by matching blocksizes/alignment, reduces metadata-server contention
- POSIX-I/O-compatible sequential & parallel API
  - adoption requires minimal source-code changes
- Tuned for common parallel filesystems
  - GPFS (BlueGene), Lustre (Cray), ...
- Convenient for application I/O, checkpointing,
  - Used by Scalasca tracing (when configured)

Developed by JSC

- Available as open-source from
  http://www.fz-juelich.de/jsc/sionlib/

Uniform integrated tool environment

- Manages installation & access to program development tools
  - based on software environment management “modules”
  - commonly used on most cluster and HPC systems
  - configurable for multiple MPI libraries & compiler suites
- Specifies how & where tools packages get installed
  - including integrating tools where possible
- Defines standard module names and different versions
- Supplies pre-defined module files
- Configurable to co-exist with local installations & policies

Developed by JSC, RWTH & TUD

- Available as open-source from http://www.vi-hps.org/projects/unite/

Disclaimer

Tools will ***not*** automatically make you, your applications or computer systems more *productive*.

However, they can help you understand ***how*** your parallel code executes and ***when / where*** it's necessary to work on *correctness* and *performance* issues.