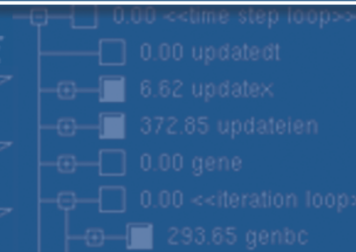


VI-HPS

SOFTWARE



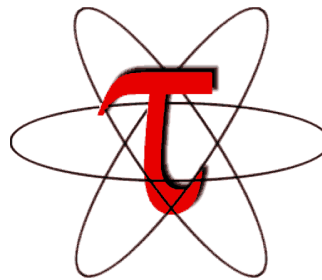
FAST SOLUTIONS

- ☒ PAPI_L1_DCM
- ☒ PAPI_L1_ICM
- ☐ PAPI_L2_DCM
- ☒ PAPI_L2_ICM
- ☒ PAPI_L3_ICM
- ☐ PAPI_L2_TCM

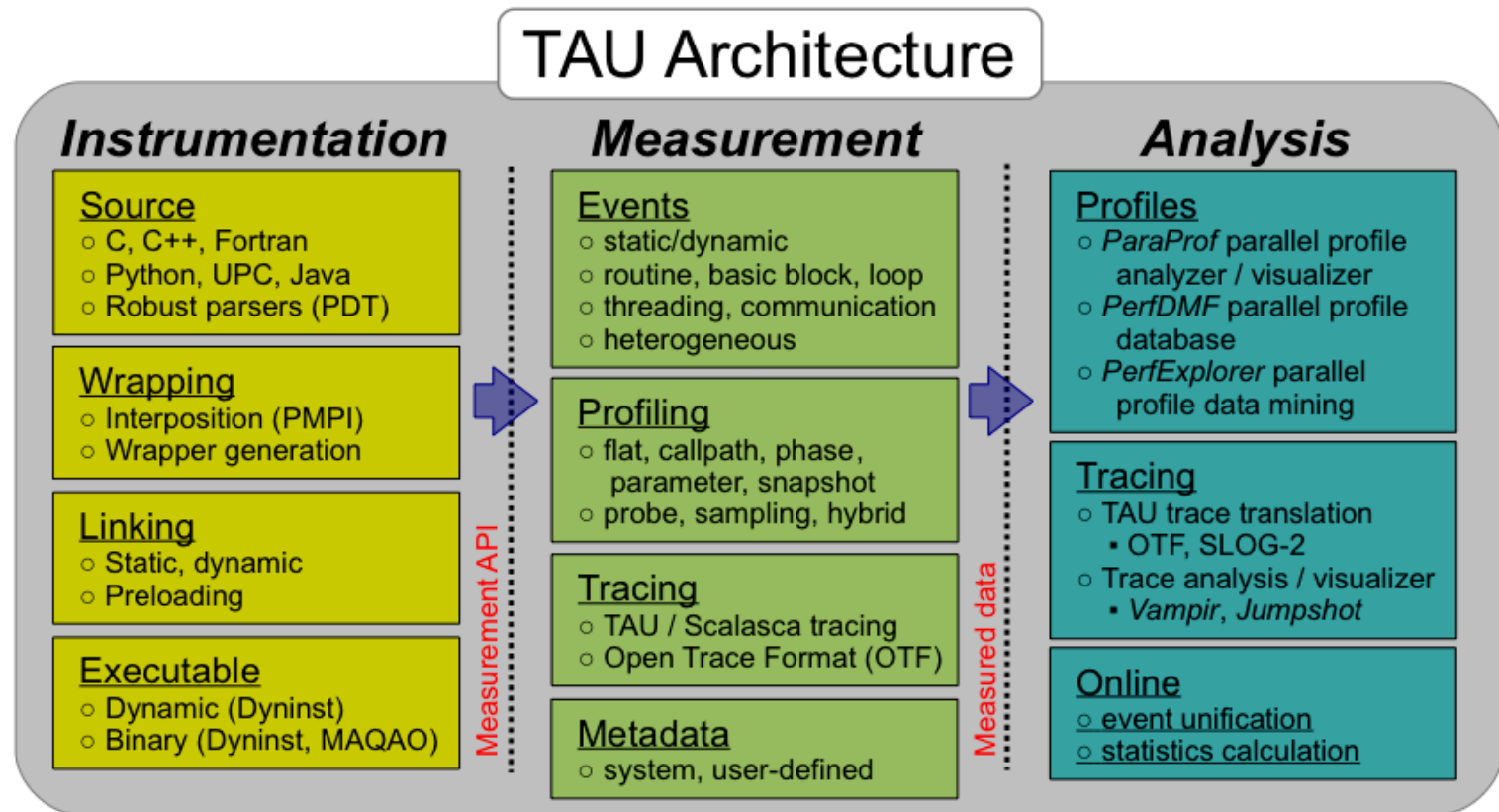
PRODUCTIVITY

Parallel Profile Analysis with TAU

Prof. Allen D. Malony
University of Oregon



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



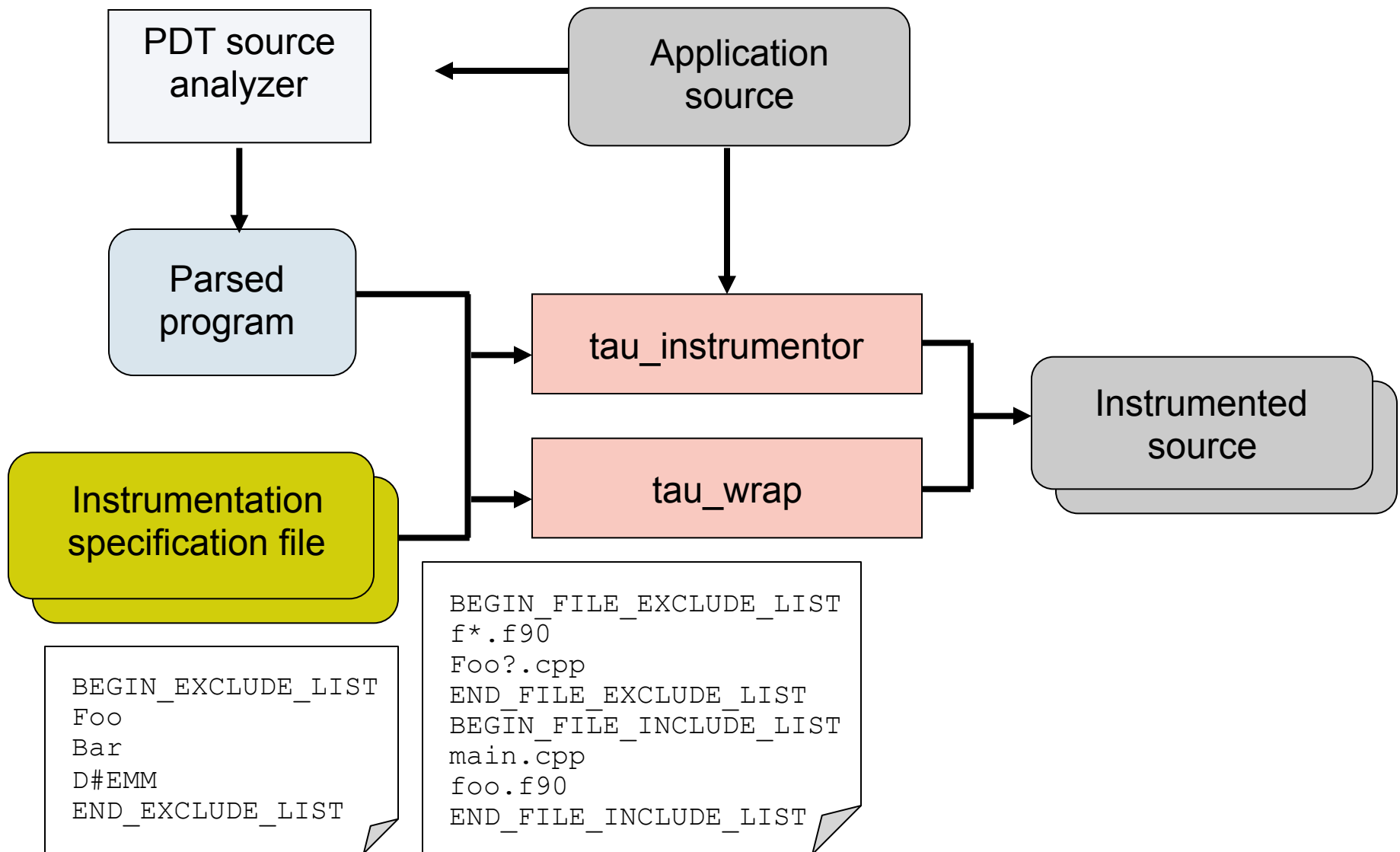
- TAU's (primary) methodology for parallel performance observation is based on the insertion of measurement *probes* into application, library, and runtime system
 - Code is instrumented to make visible certain events
 - Performance measurements occur when events are triggered
 - Known as probe-based (direct) measurement
- Performance experimentation workflow
 - Instrument application and other code components
 - Link / load TAU measurement library
 - Execute program to gather performance data
 - Analysis performance data with respect to events
 - Analyze multiple performance experiments
- Extend methodology with sampling-based techniques

- How much time is spent in each routine and outer loops?
 - Within loops, what is the contribution of each statement?
- How many instructions are executed in code regions?
 - Floating point, Level 1/2 data cache misses, hits, branches, ...
- What is the memory usage of the code?
 - When and where is memory allocated/de-allocated?
 - Are there any memory leaks?
- What are the I/O characteristics of the code?
 - What is the peak read/write BW 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?

- Instrumentation
 - Fortran, C/C++, OpenMP, UPC, Java, Python, Chapel
 - Source, compiler, library wrapping, binary rewriting
 - Automatic instrumentation
- Measurement and analysis support
 - Internode: MPI, OpenSHMEM, ARMCI, PGAS, DMAPP
 - Intranode: pthreads, OpenMP, hybrid, other thread models
 - Heterogeneous: GPU, MIC, CUDA, OpenCL, OpenACC, ...
 - Performance data (timing, counters) and metadata
 - Parallel profiling and tracing (with Score-P integration)
- Analysis
 - Parallel profile analysis and visualization (*ParaProf*)
 - Performance data mining / machine learning (*PerfExplorer*)
 - Performance database technology (*TAUdb*)
 - Empirical autotuning (integration with Active Harmony, CHiLL, Orio)

- Direct and indirect performance instrumentation
 - *Direct* instrumentation of program (system) code (probes)
 - *Indirect* support via sampling or interrupts
- Support for standard program code events
 - Routines, classes and templates
 - Statement-level blocks, loops
 - *Interval events* (start/stop)
- Support for user-defined events
 - Interval events specified by user
 - *Atomic events* (statistical measurement at a single point)
 - *Context events* (atomic events with calling path context)
- Provides static events and dynamic events


- Source code
 - Manual (TAU API, TAU component API)
 - Automatic (robust)
 - C, C++, F77/90/95, OpenMP (POMP/OPARI), UPC
 - Compiler (GNU, IBM, NAG, Intel, PGI, Pathscale, Cray, ...)
- Object code (library-level)
 - Statically- and dynamically-linked wrapper libraries
 - MPI, I/O, memory, ...
 - Powerful library wrapping of external libraries without source
- Executable code / runtime
 - Runtime preloading and interception of library calls (*tau_exec*)
 - Binary instrumentation (Dyninst, MAQAO, PEBIL)
 - OpenMP Tools Interface (OMPT, *tau_exec -T ompt*)
 - Sampling (TAU_SAMPLING=1, TAU_EBS_UNWIND=1)
 - CUDA CUPTI, OpenCL (*tau_exec -T cupti -cupti*)



- Portable and scalable parallel profiling solution
 - Multiple profiling types and options
 - Event selection and control (enabling/disabling, throttling)
 - Online profile access and sampling
 - Online performance profile overhead compensation
- Portable and scalable parallel tracing solution
 - Trace translation to OTF, EPILOG, Paraver, and SLOG2
 - Trace streams (OTF) and hierarchical trace merging
- Robust timing and hardware performance support
- Multiple counters (hardware, user-defined, system)
- Metadata (hardware/system, application, ...)
- Measurement approach also integrates with Score-P

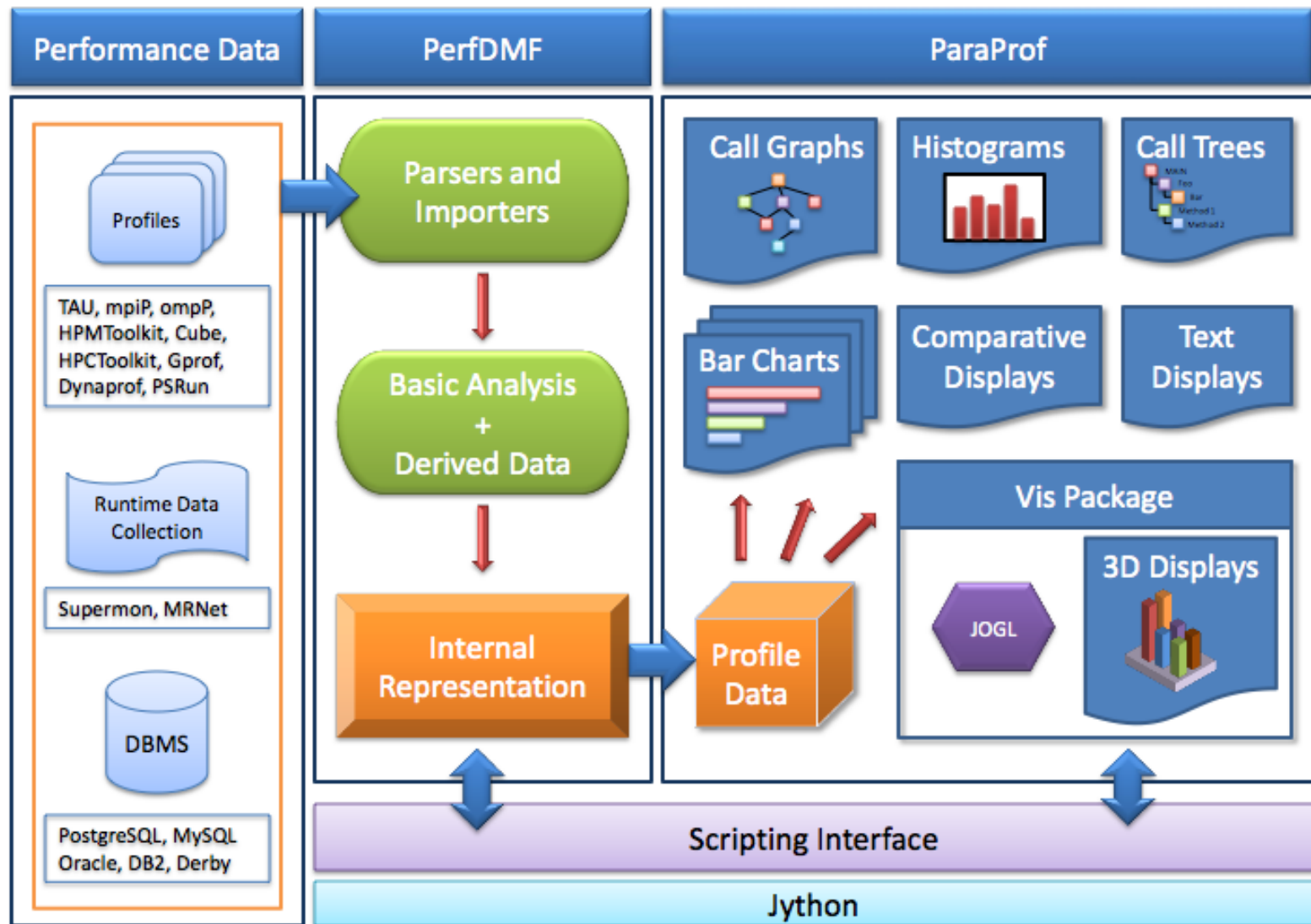
- Parallel profiling
 - Function-level, block-level, statement-level
 - Supports user-defined events and mapping events
 - Support for flat, callgraph/callpath, phase profiling
 - Support for parameter and context profiling
 - Support for tracking I/O and memory (library wrappers)
 - Parallel profile stored (dumped, snapshot) during execution
- Tracing
 - All profile-level events
 - Inter-process communication events
 - Inclusion of multiple counter data in traced events
- Utilizes Score-P for efficient callpath profiling and tracing

- Flat profiles
 - Metric (e.g., time) spent in an event (callgraph nodes)
 - Exclusive/inclusive, # of calls, child calls
- Callpath profiles (Callddepth profiles)
 - Time spent along a calling path (edges in callgraph)
 - “main=> f1 => f2 => MPI_Send” (event name)
 - TAU_CALLPATH_DEPTH environment variable
- Phase profiles
 - Flat profiles under a phase (nested phases are allowed)
 - Default “main” phase
 - Supports static or dynamic (per-iteration) phases
- Parameter and context profiling

- TAU supports several measurement and thread options
 - 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:
 - % export TAU_MAKEFILE=\$TAU/Makefile.tau-icpc-mpi-pdt
 - % export TAU_OPTIONS= ‘-optVerbose ...’ (see tau_compiler.sh)
 - % export PATH=\$TAU_ROOT/x86_64/bin:\$PATH
 - % export TAU=\$TAU_ROOT/x86_64/lib
 - Use *tau_f90.sh*, *tau_cxx.sh*, *tau_upc.sh*, or *tau_cc.sh* as compilers
 - % mpif90 foo.f90  % tau_f90.sh foo.f90
- Set environment variables, execute application, and analyze with *pprof* (text based profile display) or *paraprof* (GUI)

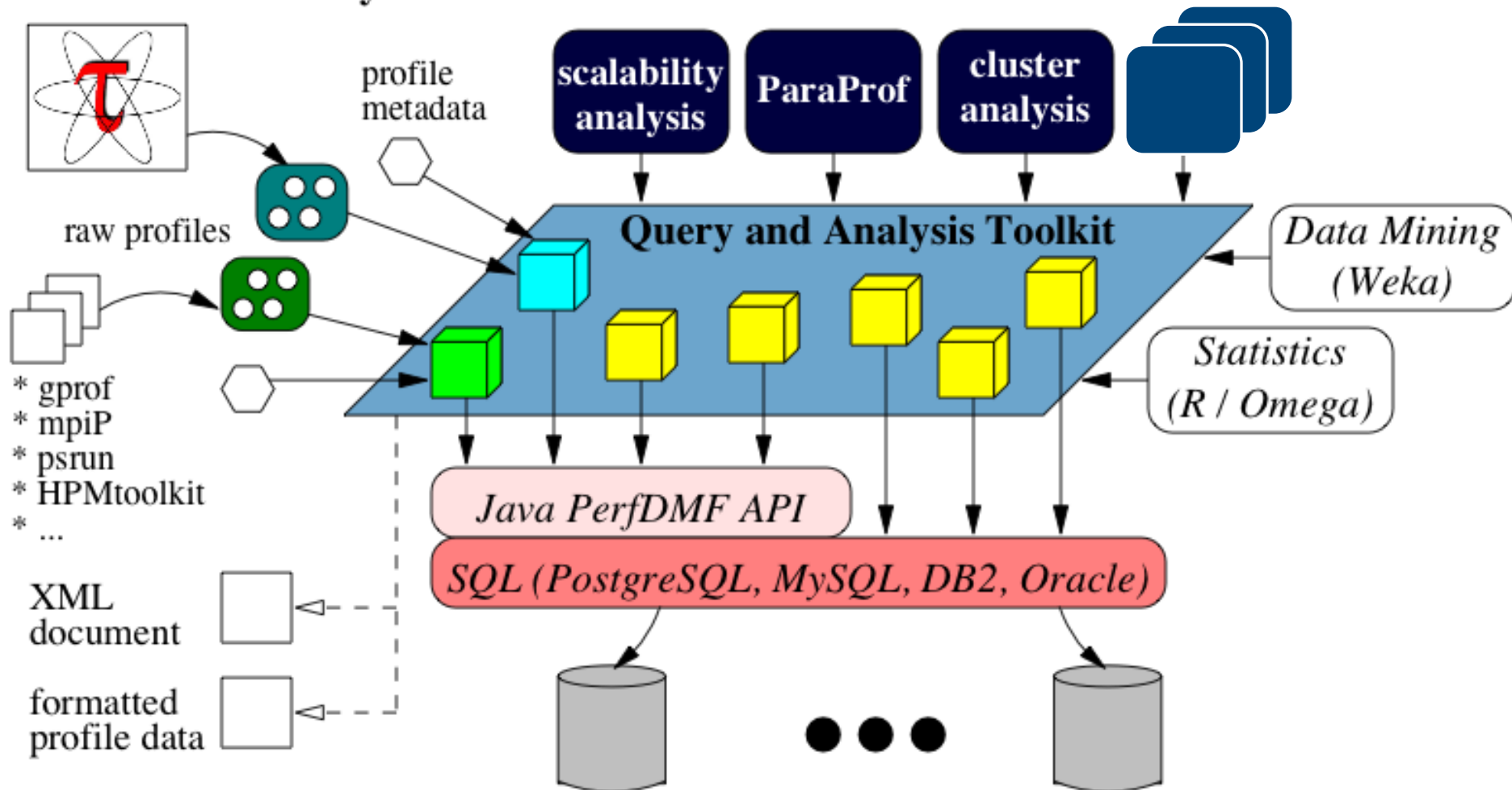
```
% export TAU=$TAU_ROOT/x86_64/lib
% export TAU_MAKEFILE=
    $TAU/Makefile.tau-icpc-papi-mpi-pdt-openmp-opari-scorep
% export OMP_NUM_THREADS=10
% make CC=tau_cc.sh CXX=tau_cxx.sh F90=tau_f90.sh
% mpirun -np 4 ./matmult
% cd score*; paraprof profile.cubex &
```

- Analysis of parallel profile and trace measurement
- Parallel profile analysis (*ParaProf*)
 - Java-based analysis and visualization tool
 - Support for large-scale parallel profiles
- Performance data management (*TAUdb*)
- Performance data mining (PerfExplorer)
- Parallel trace analysis
 - Translation to VTF (V3.0), EPILOG, OTF formats
 - Integration with Vampir / Vampir Server (TU Dresden)
- Integration with CUBE browser (Scalasca, UTK / FZJ)
- Scalable runtime fault isolation with callstack debugging
- Efficient parallel runtime bounds checking

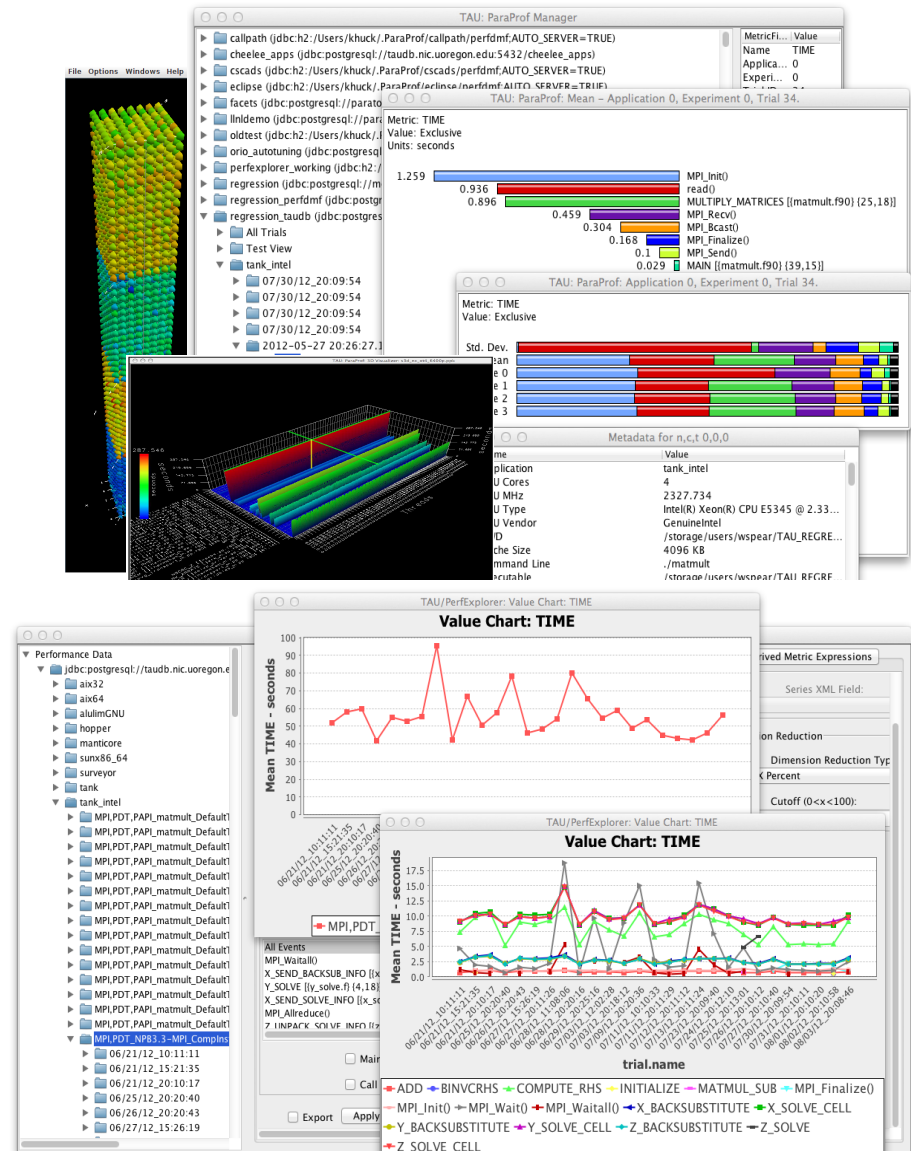


TAU Performance System

Performance Analysis Programs



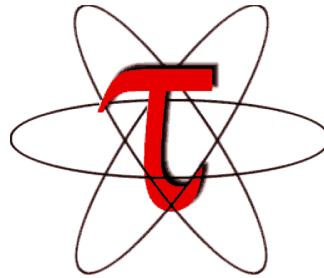
- ParaProf
 - Parallel profile analyzer
 - visual pprof
 - 2, 3+D visualizations
 - Single and comparative experiment analysis
- PerfExplorer
 - Data mining framework
 - Clustering, correlation
 - Multi-experiment analysis
 - Scripting engine
 - Expert system



VI-HPS



Demo: NPB-MZ-MPI / BT



- Tutorial contains Score-P experiments of BT-MZ
 - MPI + OpenMP
 - Class “B”, 4 processes with 4 OpenMP threads each
 - Collected on SuperMUC HPC system at LRZ, Munich, Germany

```
% wget http://tau.uoregon.edu/data.tgz
% tar xzf data.tgz
% cd data/scorep
```

- Load trials into TAUdb

```
% taudb_loadtrial -a BT_MZ -x "Class_B" -n "16p" -f cube 2p/profile.cubex
% taudb_loadtrial -a BT_MZ -x "Class_B" -n "16p" -f cube 4p/profile.cubex
% taudb_loadtrial -a BT_MZ -x "Class_B" -n "16p" -f cube 8p/profile.cubex
% taudb_loadtrial -a BT_MZ -x "Class_B" -n "16p" -f cube 16p/profile.cubex
```

- Start TAU's paraprof and perfexplorer GUI

```
% paraprof &
% perfexplorer &
```

TAU: ParaProf Manager

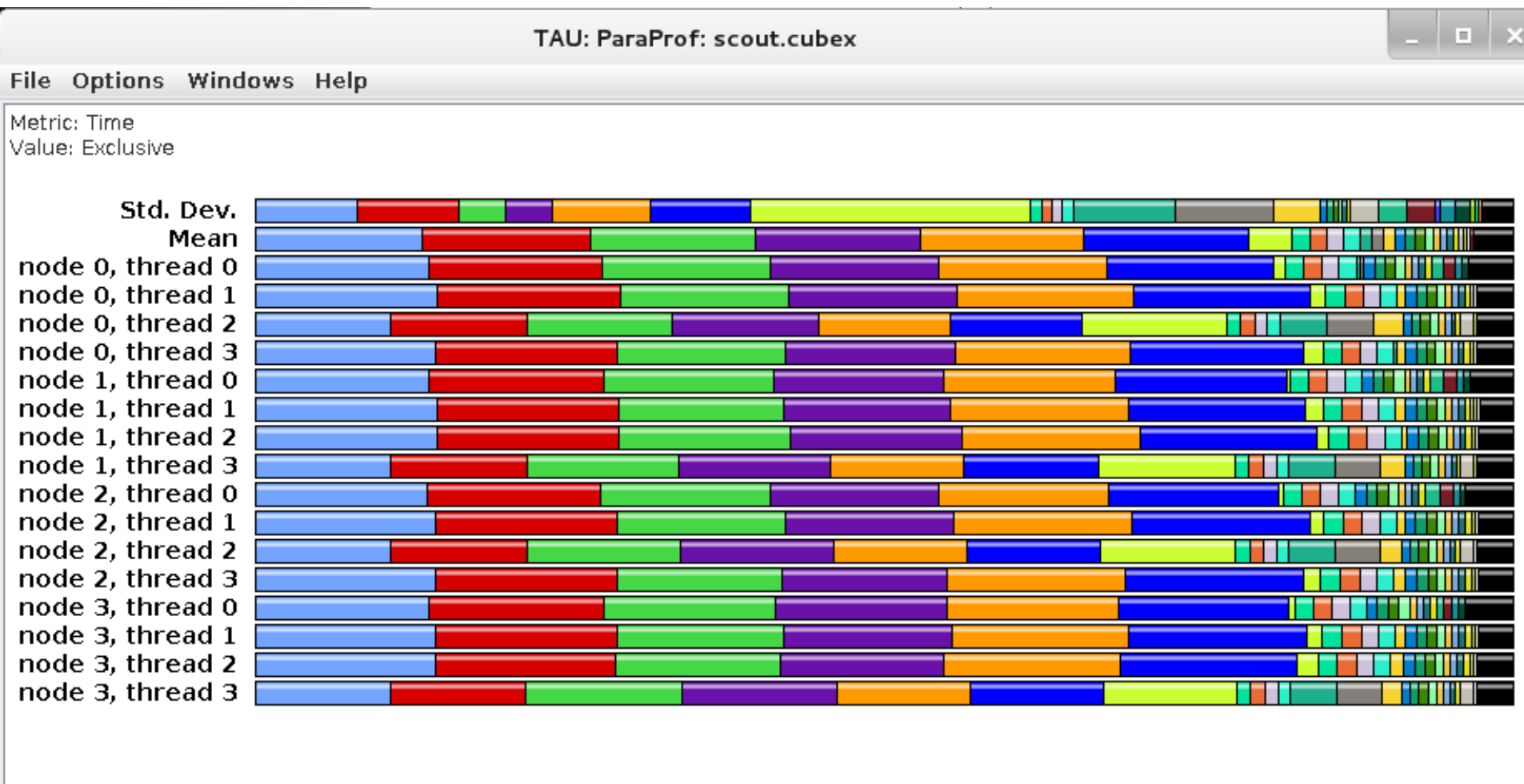
File Options Help

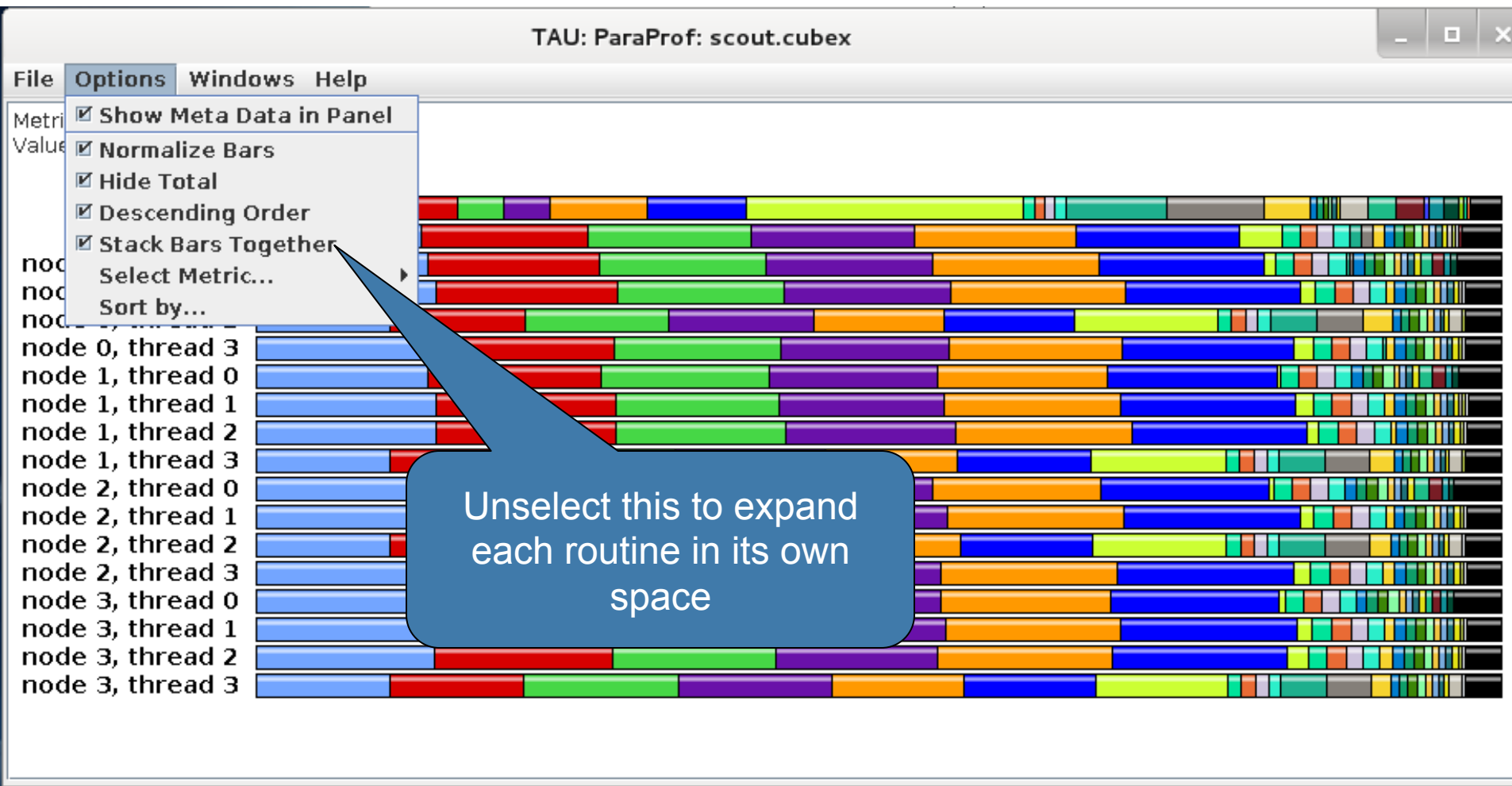
Applications

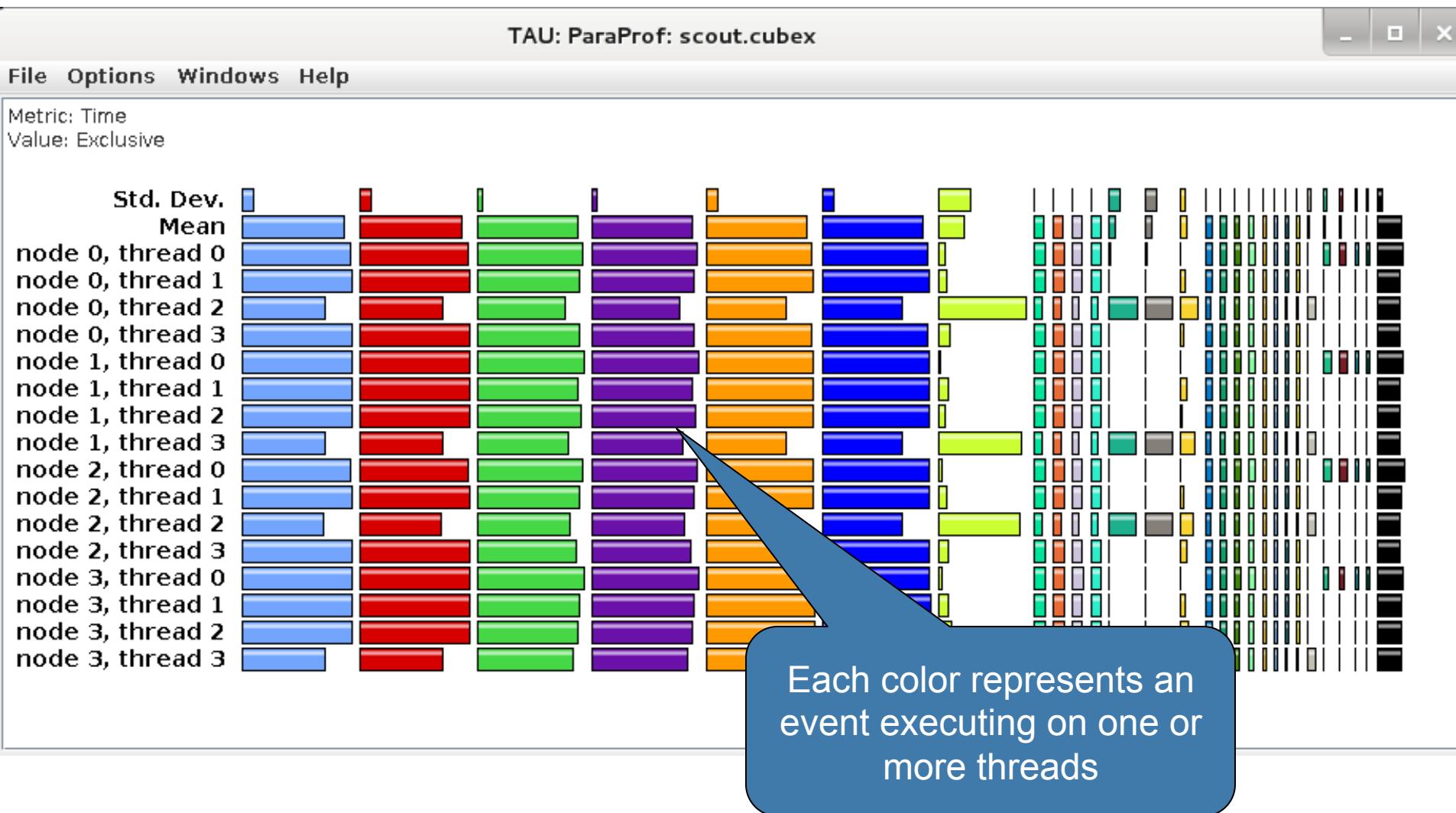
- Standard Applications
 - Default App
 - Default Exp
 - scout.cubex
 - Time
 - Wait at Barrier
 - Barrier Completion
 - Late Sender
 - Late Sender => Messages in Wrong Order
 - Late Sender => Messages in Wrong Order => Messages from different sources
 - Late Sender => Messages in Wrong Order => Messages 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

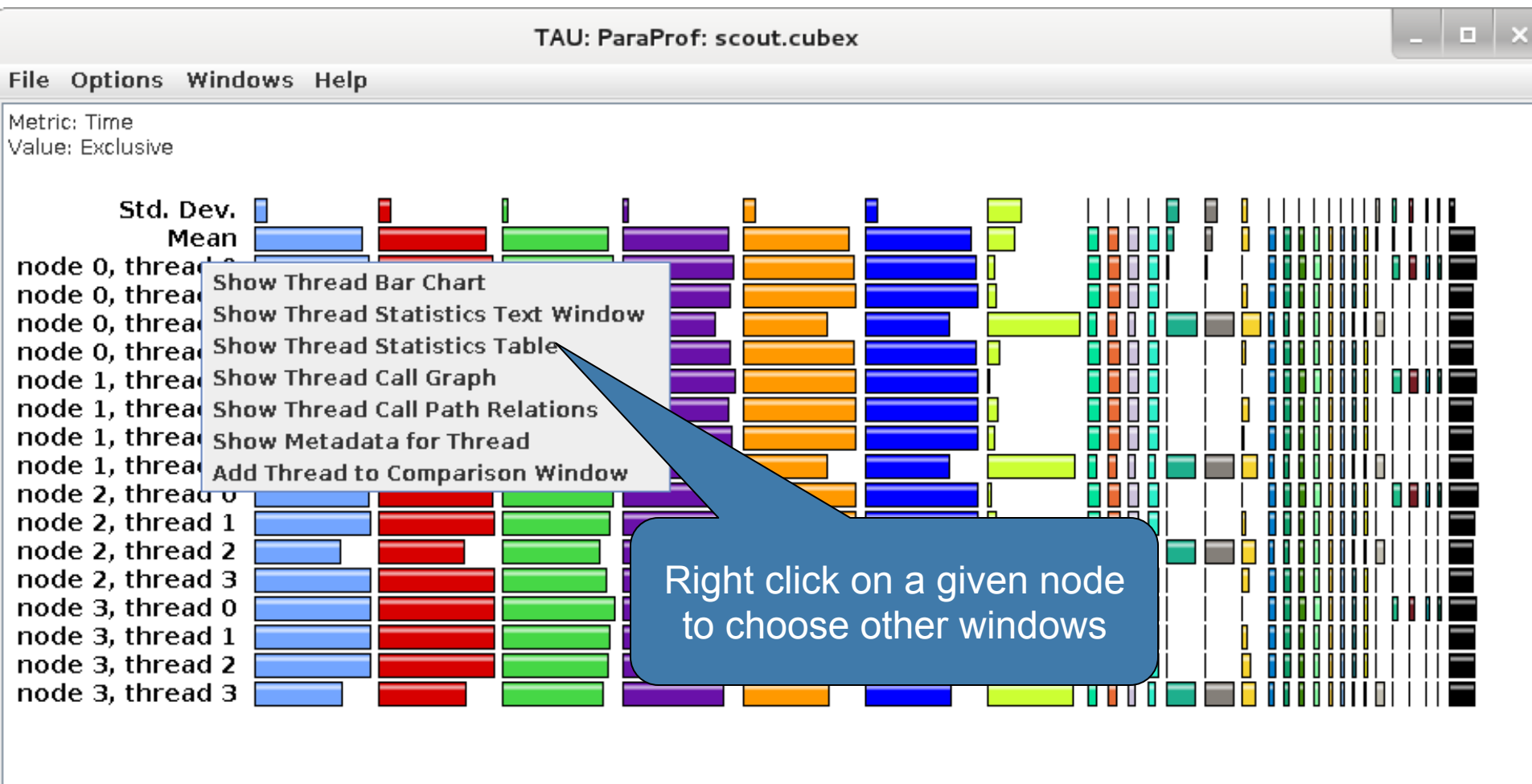
TrialField	Value
Name	scout.cubex
Application ID	0
Experiment ID	0
Trial ID	0
File Type Index	9
File Type Name	Cube

Metrics in the profile









ParaProf: Thread Statistics Table

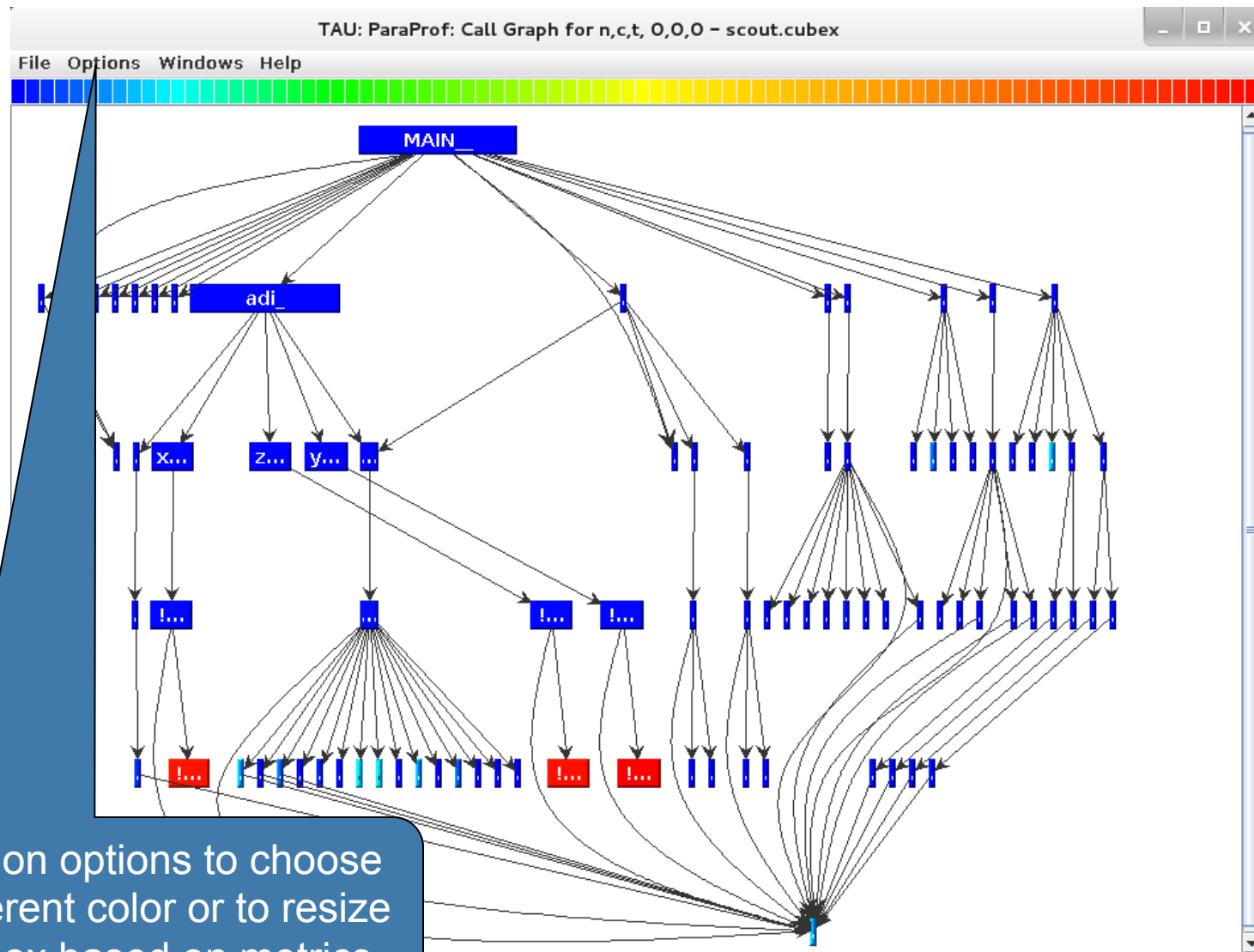
TAU: ParaProf: Statistics for: node 0, thread 0 - scout.cubex

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	0.609	3,232	0
!\$omp do @rhs.f:80	0.583	0.583	3,232	0
MPI_Waitall	0.402	0.402	1	0
!\$omp implicit barrier	0.402	0.402	1	0
!\$omp do @rhs.f:301	0.36	0.36	1	0
!\$omp implicit barrier	0.026	0.026	1	0
!\$omp implicit barrier	0	0	1	0
!\$omp do @rhs.f:37	0.343	0.343	1	0
!\$omp do @rhs.f:62	0.225	0.228	3,232	3,232
!\$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				

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

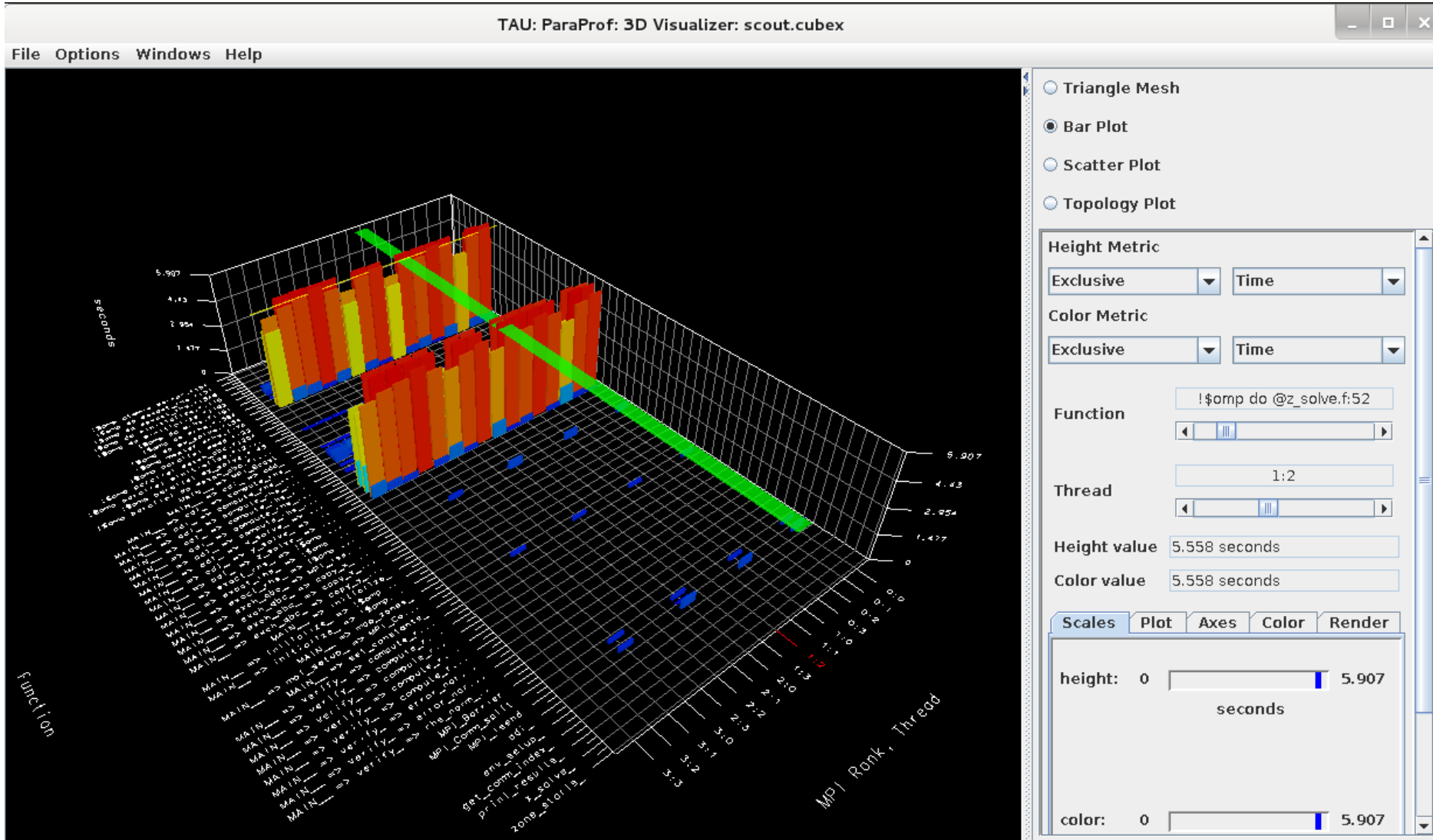


ParaProf: Callpath Thread Relations Window



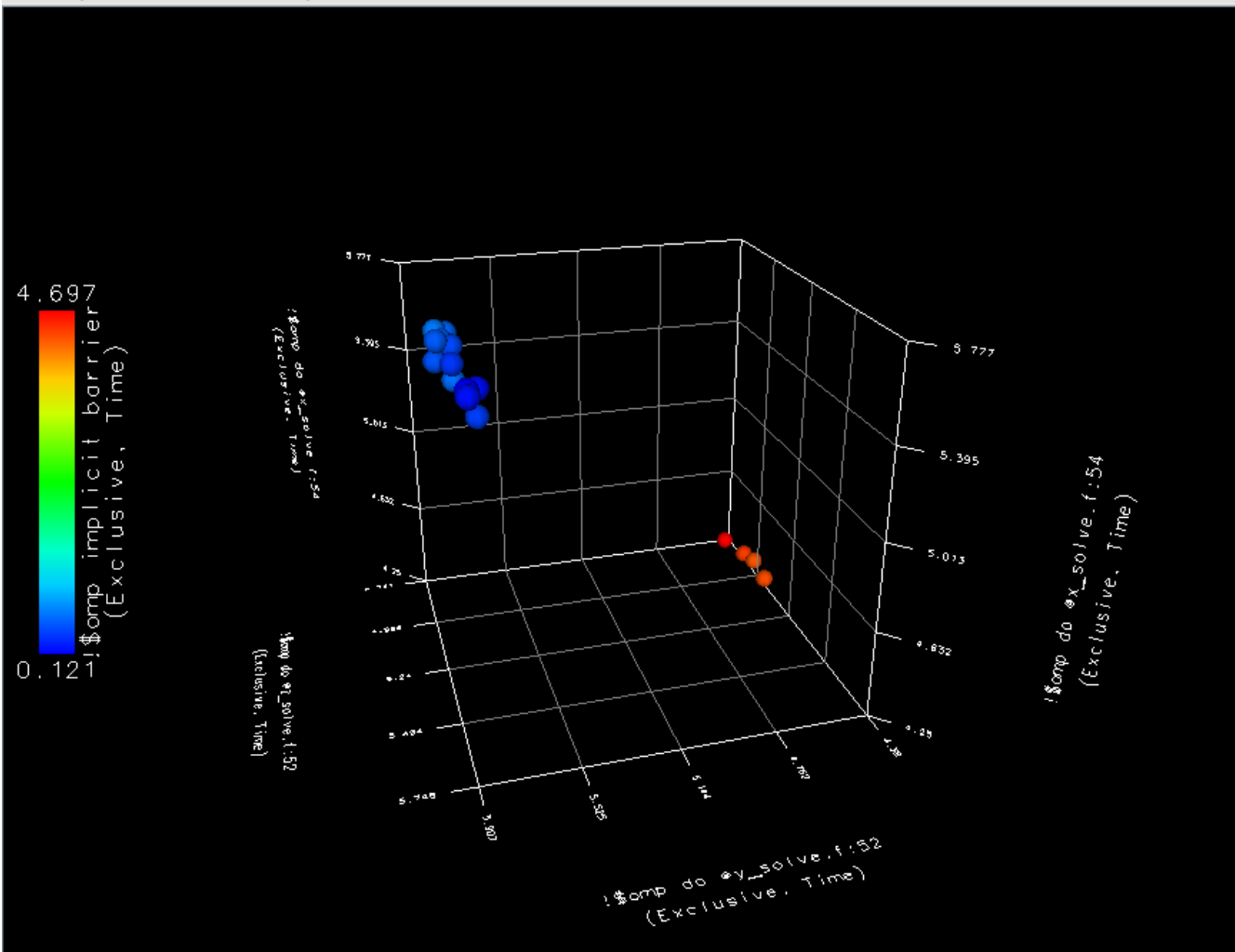
TAU: ParaProf: Call Path Data n,c,t, 0,0,0 - scout.cubex				
File Options Windows Help				
Metric Name: Time				
Sorted By: Exclusive				
Units: seconds				
-->	0.04	0.04	32/32	!\$omp parallel @initialize.f:28
	0.04	0.04	32	!\$omp do @initialize.f:50
-->	0.03	2.536	3232/3232	compute_rhs_
	0.03	2.536	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:Windows -> 3D Visualization -> Bar Plot



TAU: ParaProf: 3D Visualizer: scout.cubex

File Options Windows Help



☐ Triangle Mesh

☐ Bar Plot

☒ Scatter Plot

☐ Topology Plot

Width
!\$omp do @y_solve.f:52 ...
Exclusive Time

Depth
!\$omp do @z_solve.f:52 ...
Exclusive Time

Height
!\$omp do @x_solve.f:54 ...
Exclusive Time

Color
!\$omp implicit barrier ...
Exclusive Time

ColorScale Render

ScatterPlot

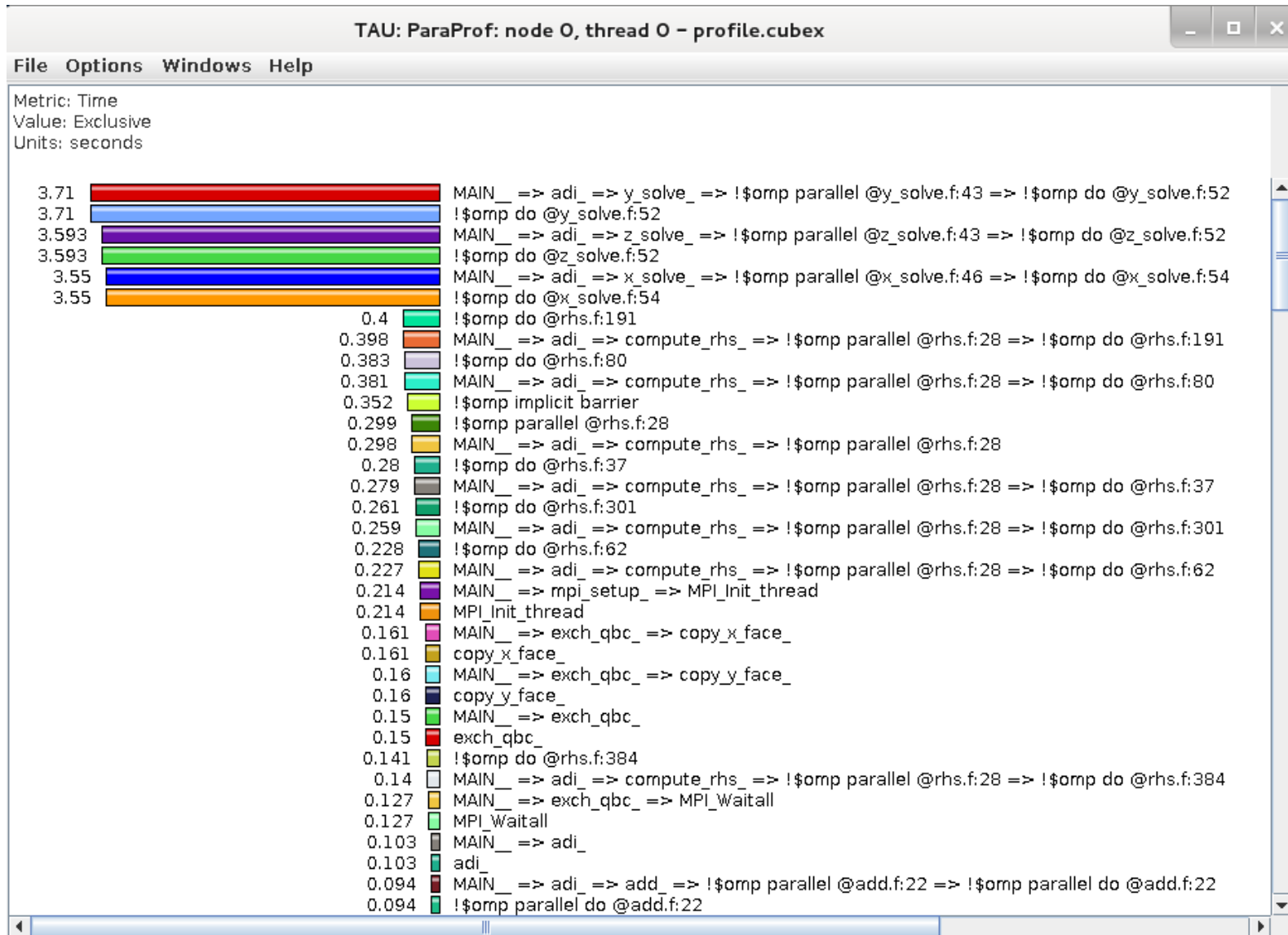
Axes

☐ Auto-Rotate

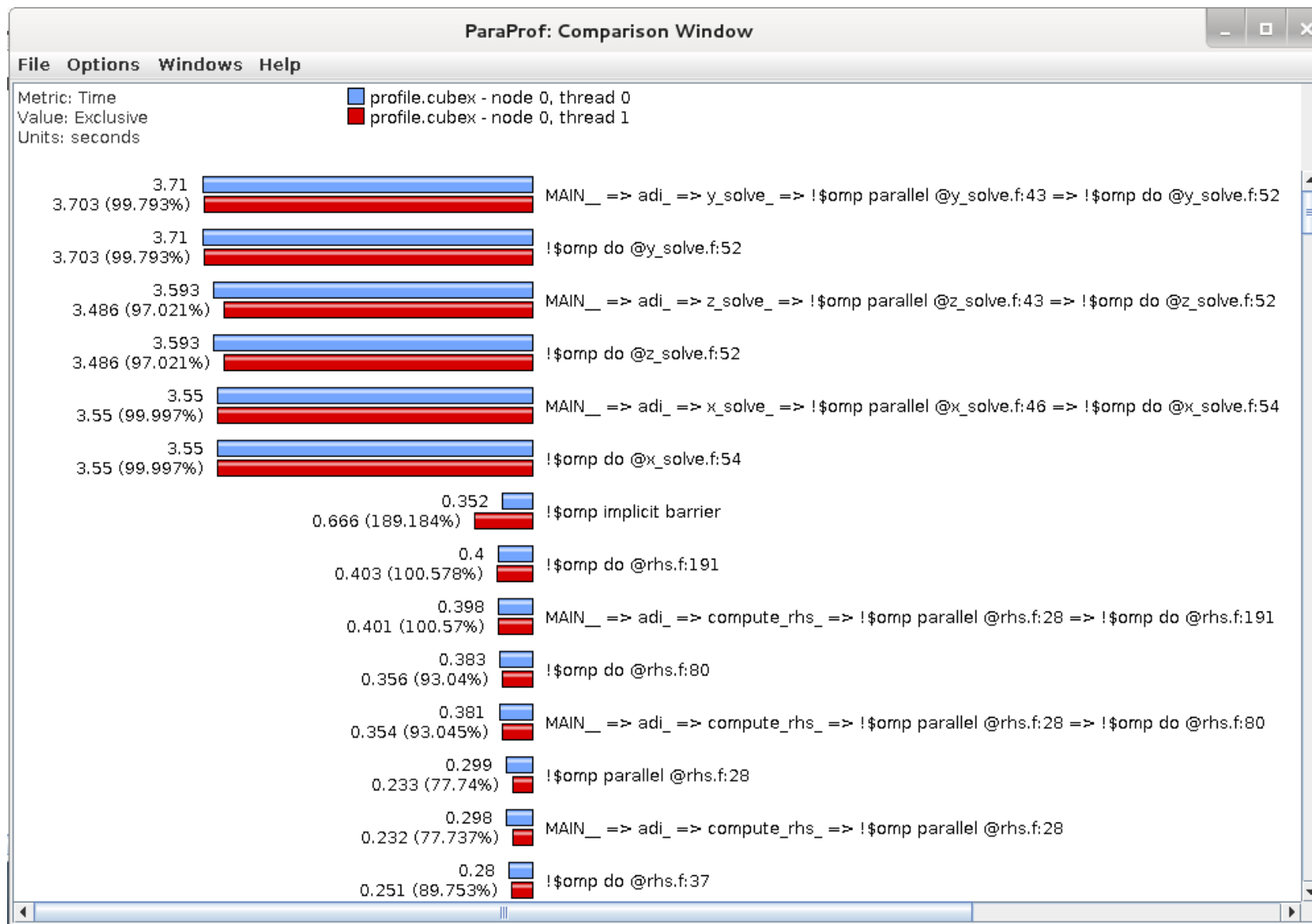
Speed

☐ Reverse Video

☐ Stereo



ParaProf: Add Thread to Comparison Window



ParaProf: Options -> Derived Metric Panel

TAU: ParaProf Manager

File Options Help

Applications

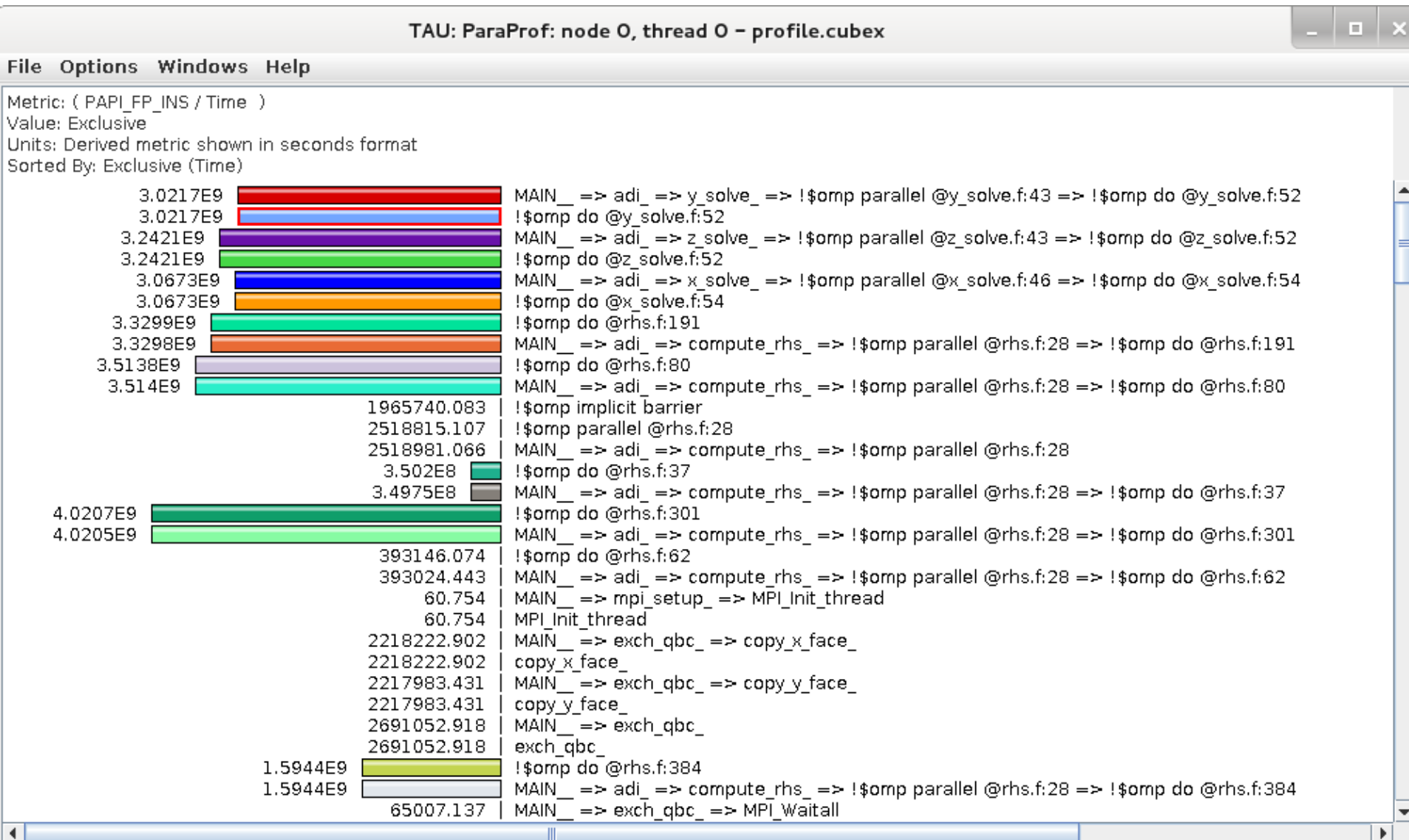
- Standard Applications
 - Default App
 - Default Exp
 - profile.cubex
 - Time**
 - Minimum Inclusive Time
 - Maximum Inclusive Time
 - PAPI_TOT_CYC
 - PAPI_TOT_INS
 - PAPI_FP_INS
 - ru_ftime
 - ru_stime
 - ru_maxrss
 - ru_ixrss
 - ru_idrss
 - ru_isrss
 - ru_minflt
 - ru_majflt
 - ru_nswap
 - ru_inblock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

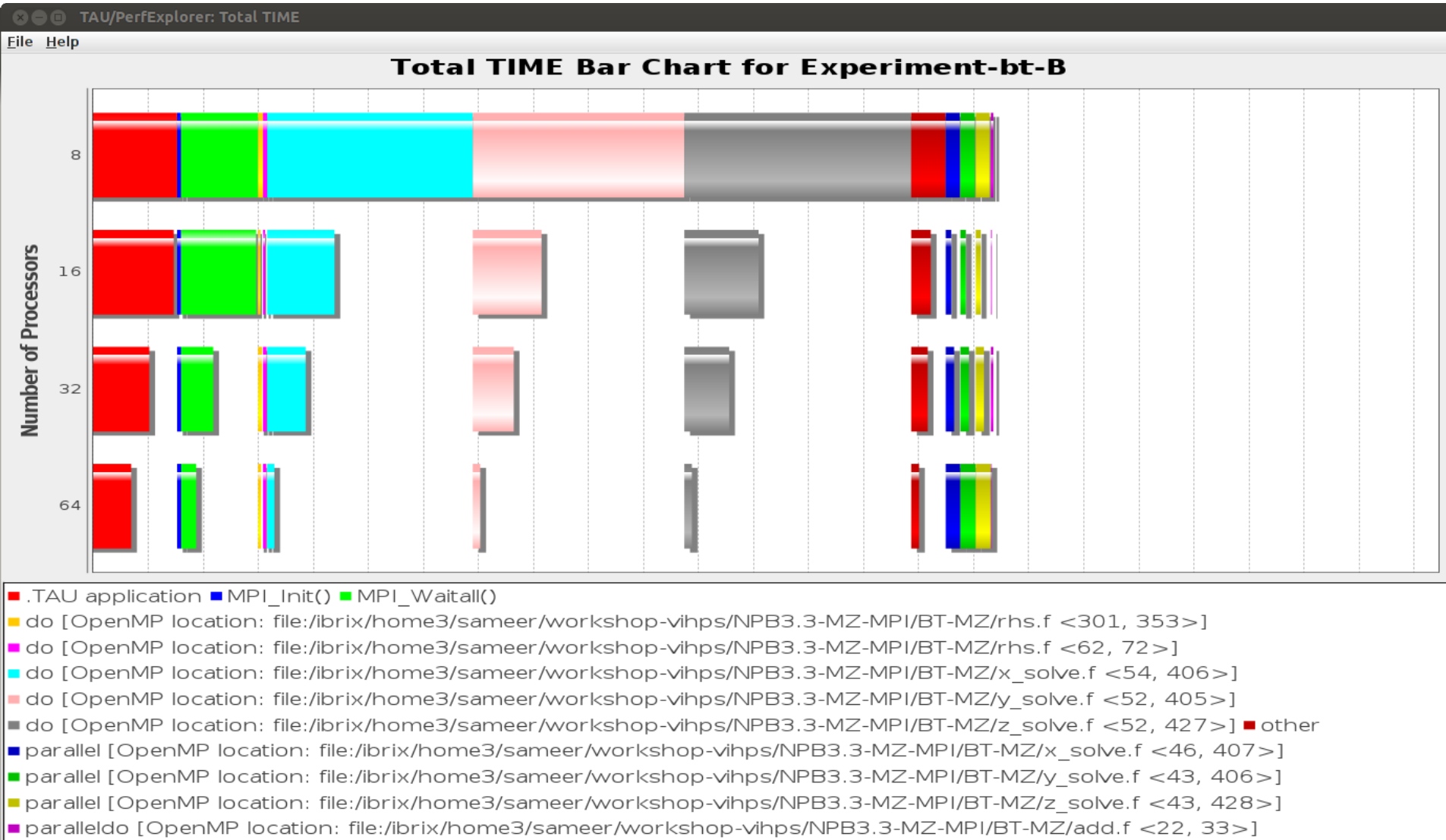
MetricField	Value
Name	Time
Application ID	0
Experiment ID	0
Trial ID	0
Metric ID	0

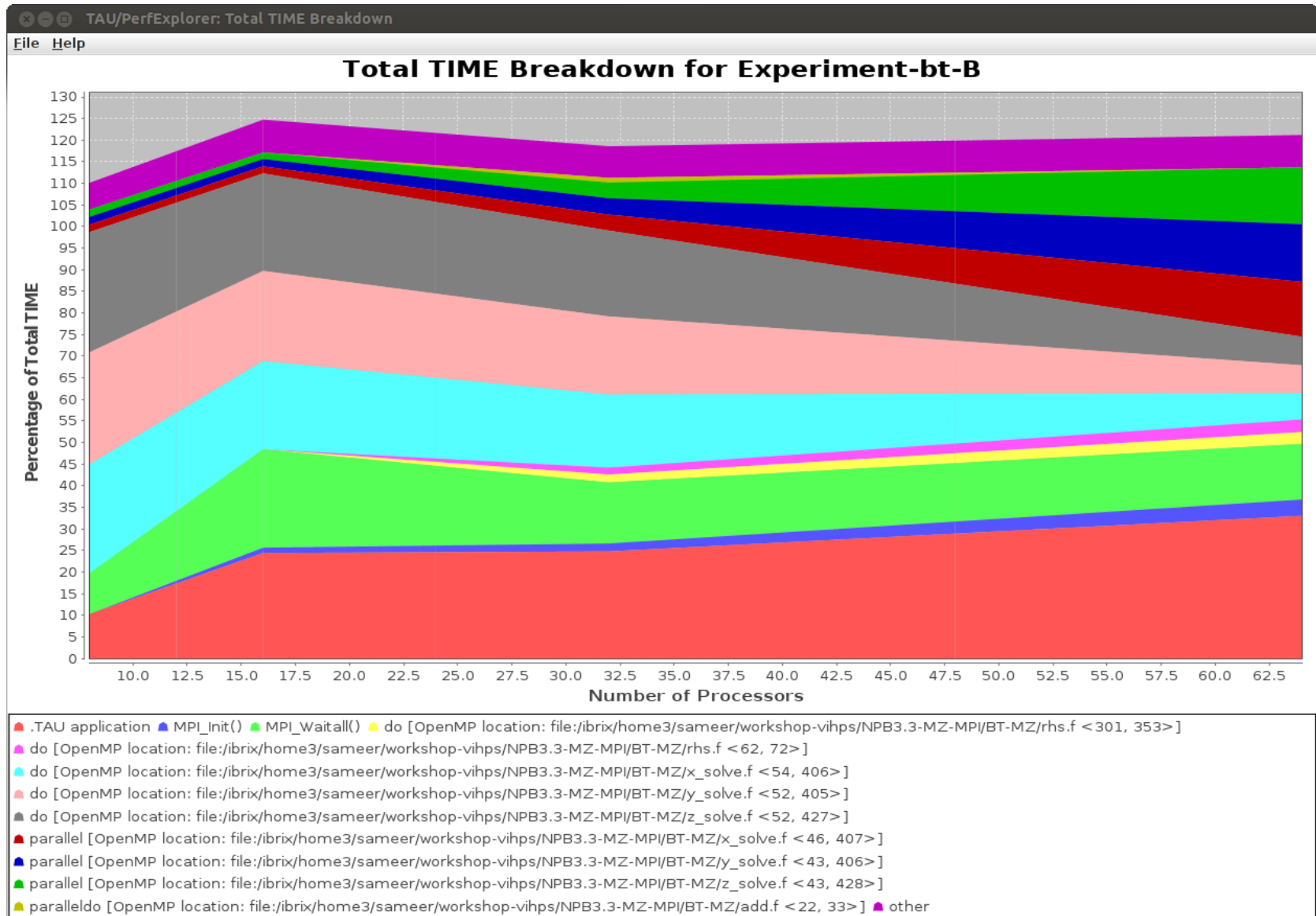
Expression: "PAPI_FP_INS"/"Time"

+ - * / = { } Apply Clear

Sorting Derived Flops Metric by Exclusive Time







- U.S. Department of Energy (DOE)
 - Advanced Scientific Computing Research
 - Office of Science
 - ASC/NNSA, Tri-labs (LLNL, LANL, SNL)
- U.S. Department of Defense (DoD)
 - HPC Modernization Office (HPCMO)
- NSF Office of Cyberinfrastructure
- Juelich Supercomputing Center, NIC
- Argonne National Laboratory
- Technical University Dresden
- ParaTools, Inc.



ParaTools