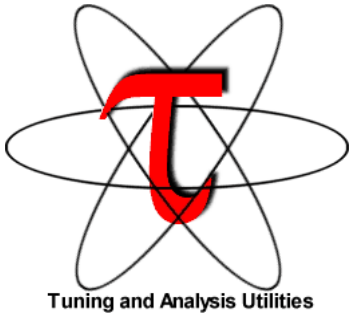
VI-HPS

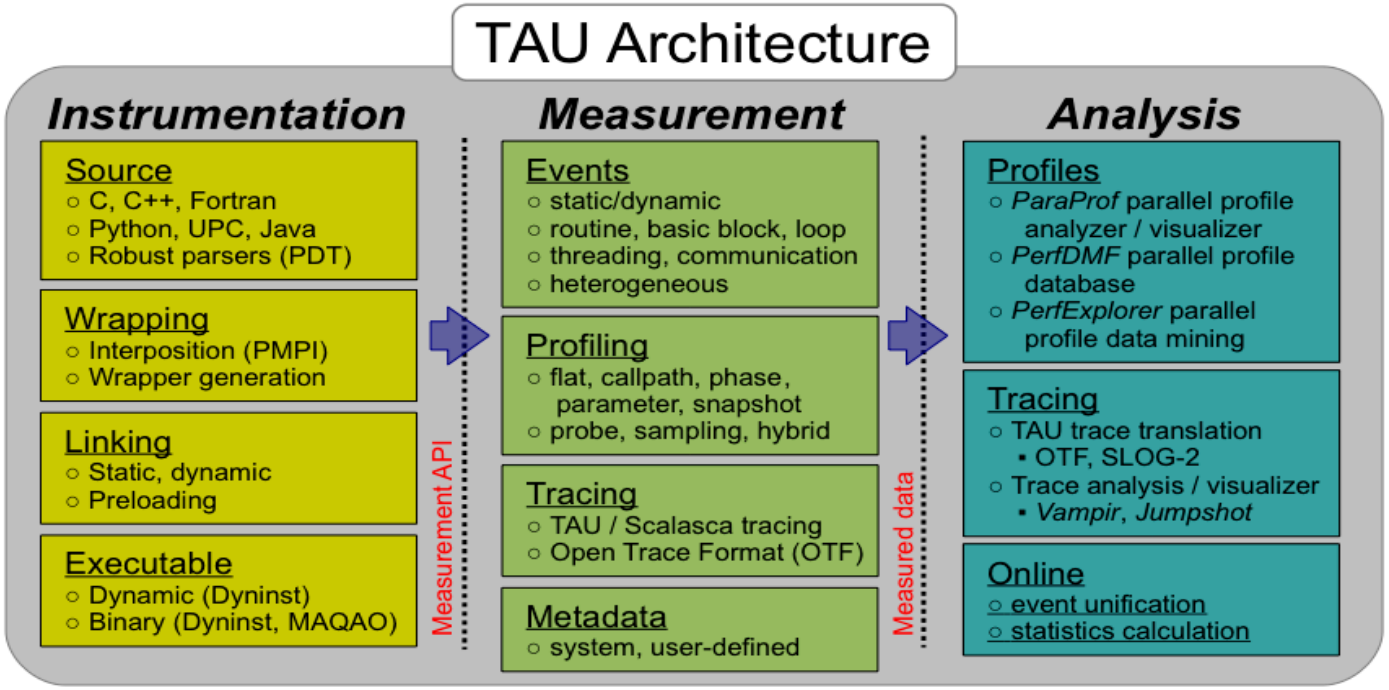
TAU Performance System®



Sameer Shende
sameer@cs.uoregon.edu
University of Oregon
http://tau.uoregon.edu/TAU_TW35.pdf

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, ROCm, HIP
  - 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

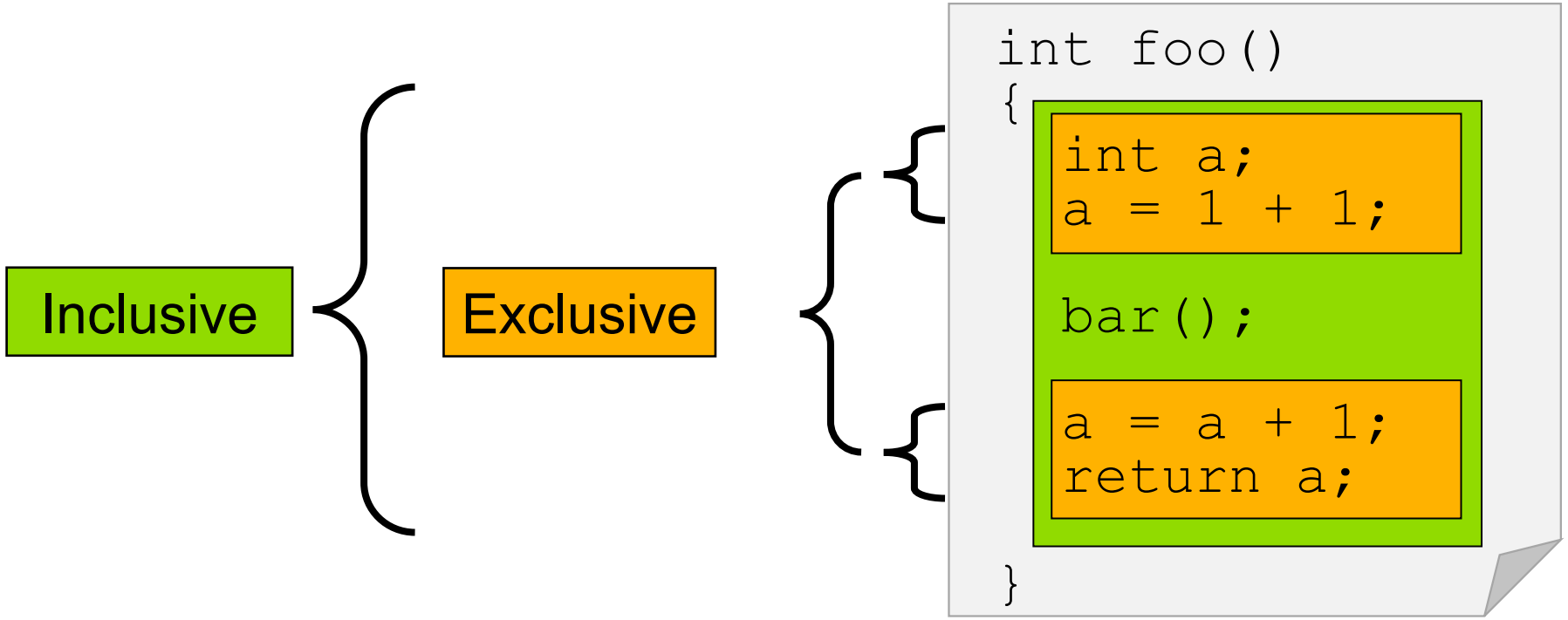
- TAU supports both sampling and direct instrumentation
- Memory debugging as well as I/O performance evaluation
- Profiling as well as tracing
- Interfaces with Score-P for more efficient measurements
- TAU's instrumentation covers:
  - Runtime library interposition (tau_exec)
  - Compiler-based instrumentation
  - 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 v5 (OMPT, tau_exec –T ompt,v5)
  - 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?

Inclusive vs. Exclusive values

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



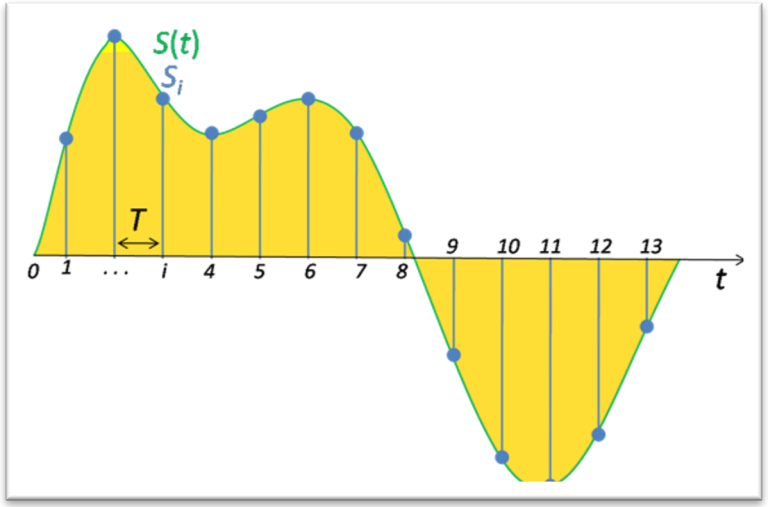
Performance Data Measurement

Direct via Probes

```
Call START('potential')
// code
Call STOP('potential')
```

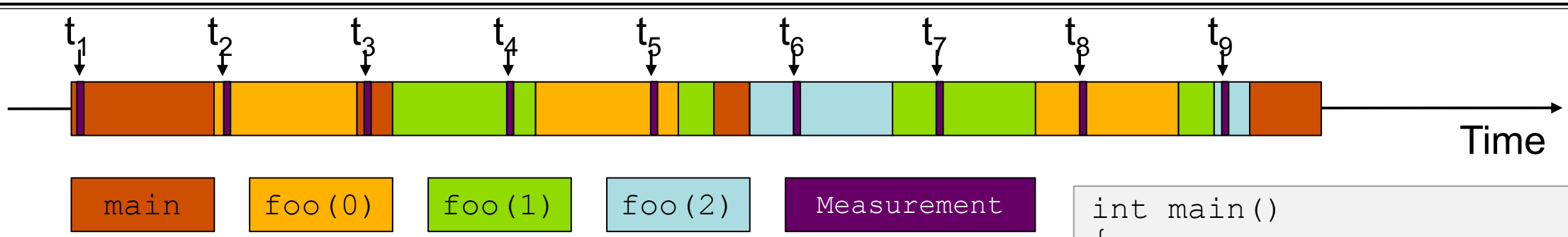
- Exact measurement
- Fine-grain control
- Calls inserted into code

Indirect via Sampling



- No code modification
- Minimal effort
- Relies on debug symbols (**-g**)

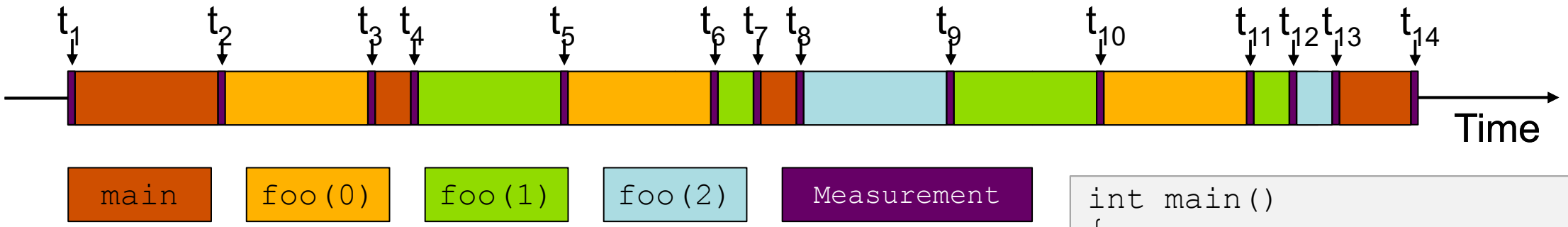
Event-Based Sampling (EBS)



- 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 (tau_exec –ebs)

```
int main()
{
  int i;

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

  return 0;
}

void foo(int i)
{

  if (i > 0)
    foo(i - 1);

}
```

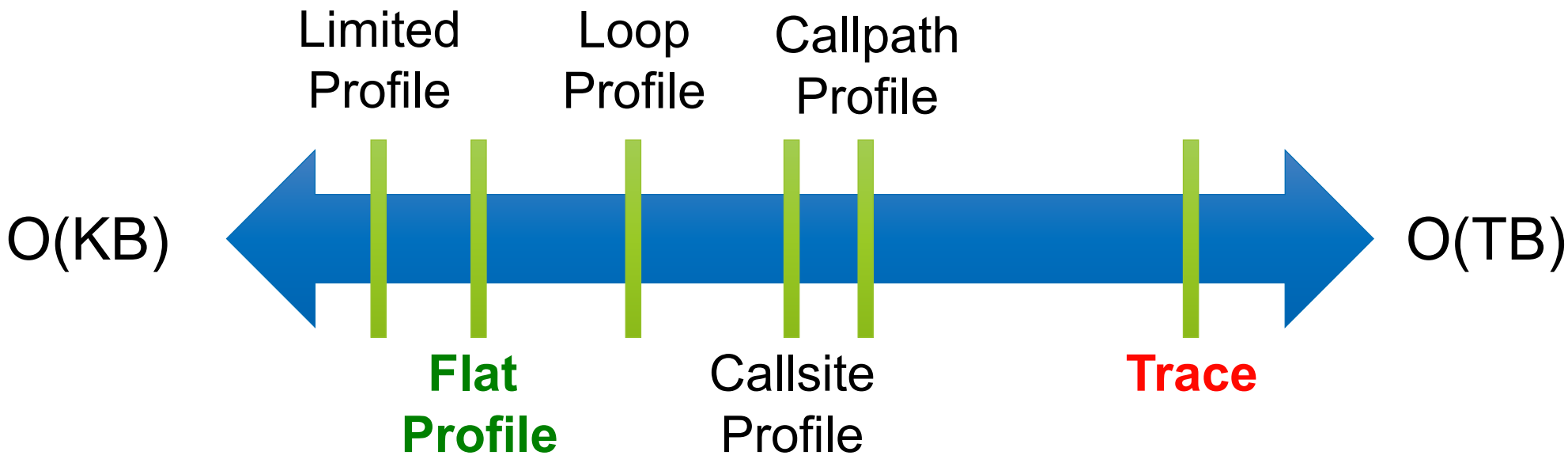
Instrumentation



- 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;
  TAU_START("main");
  for (i=0; i < 3; i++)
    foo(i);
  TAU_STOP("main");
  return 0;
}

void foo(int i)
{
  TAU_START("foo");
  if (i > 0)
    foo(i - 1);
  TAU_STOP("foo");
}
```

How much data do you want?



Types of Performance Profiles

- *Flat* profiles
  - Metric (e.g., time) spent in an event
  - Exclusive/inclusive, # of calls, child calls, …
- *Callpath* profiles
  - Time spent along a calling path (edges in callgraph)
  - "*main=> f1 => f2 => MPI_Send*"
  - Set the TAU_CALLPATH and TAU_CALLPATH_DEPTH environment variables
- *Callsite* profiles
  - Time spent along in an event at a given source location
  - Set the TAU_CALLSITE environment variable
- *Phase* profiles
  - Flat profiles under a phase (nested phases allowed)
  - Default "main" phase
  - Supports static or dynamic (e.g. per-iteration) phases

Using TAU's Runtime Preloading Tool: tau_exec

- Preload a wrapper that intercepts the runtime system call and substitutes with another
  - **MPI**
  - **OpenMP**
  - **POSIX I/O**
  - **Memory allocation/deallocation routines**
  - **Wrapper library for an external package**
- No modification to the binary executable!
- Enable other TAU options (communication matrix, OTF2, event-based sampling)
- Add tau_exec before the name of the binary
  - aprun tau_exec ./a.out
  - aprun tau_exec –T ompt,v5,mpi,papi -ompt ./a.out

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!

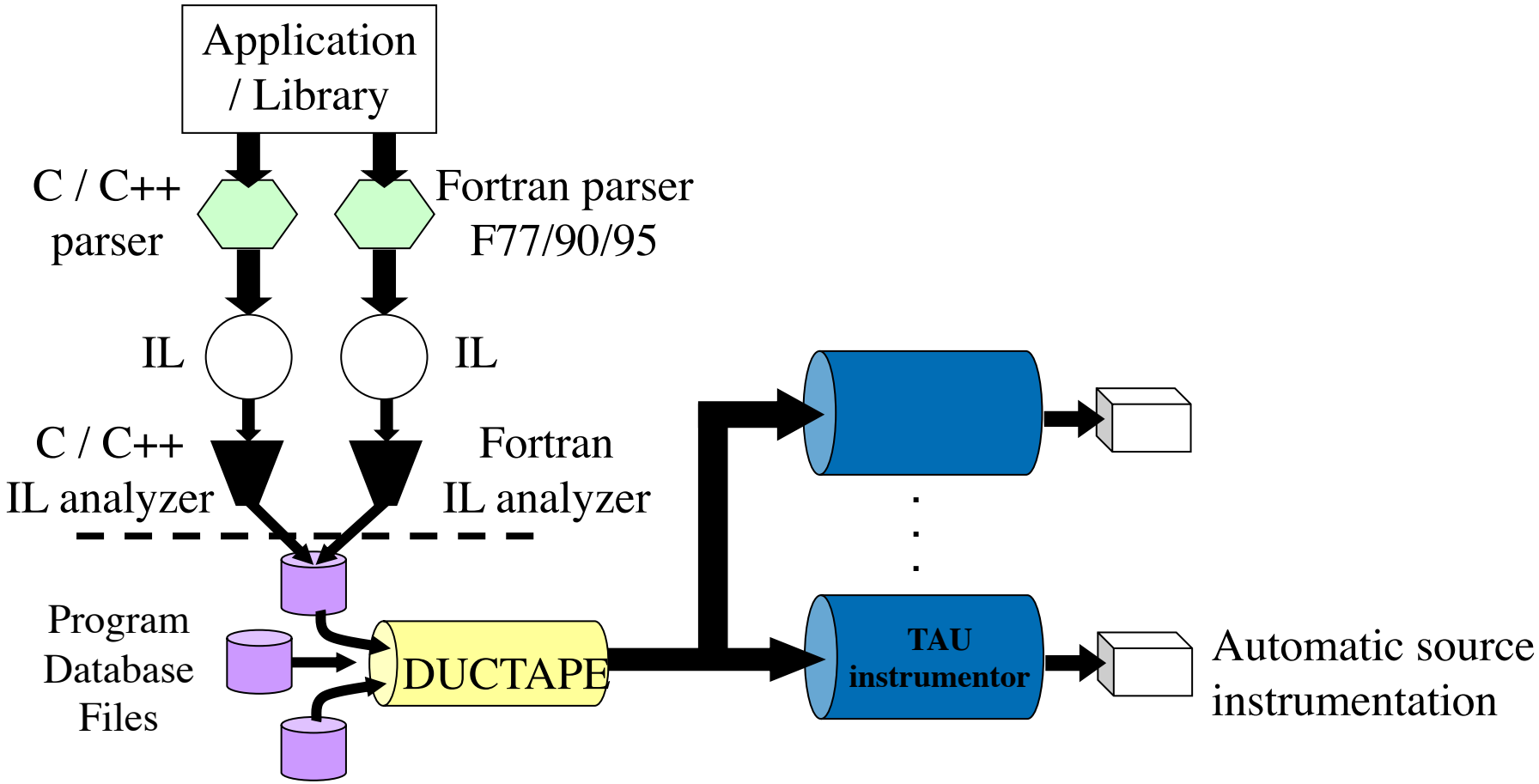
tau_exec Example (continued)

```
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
```

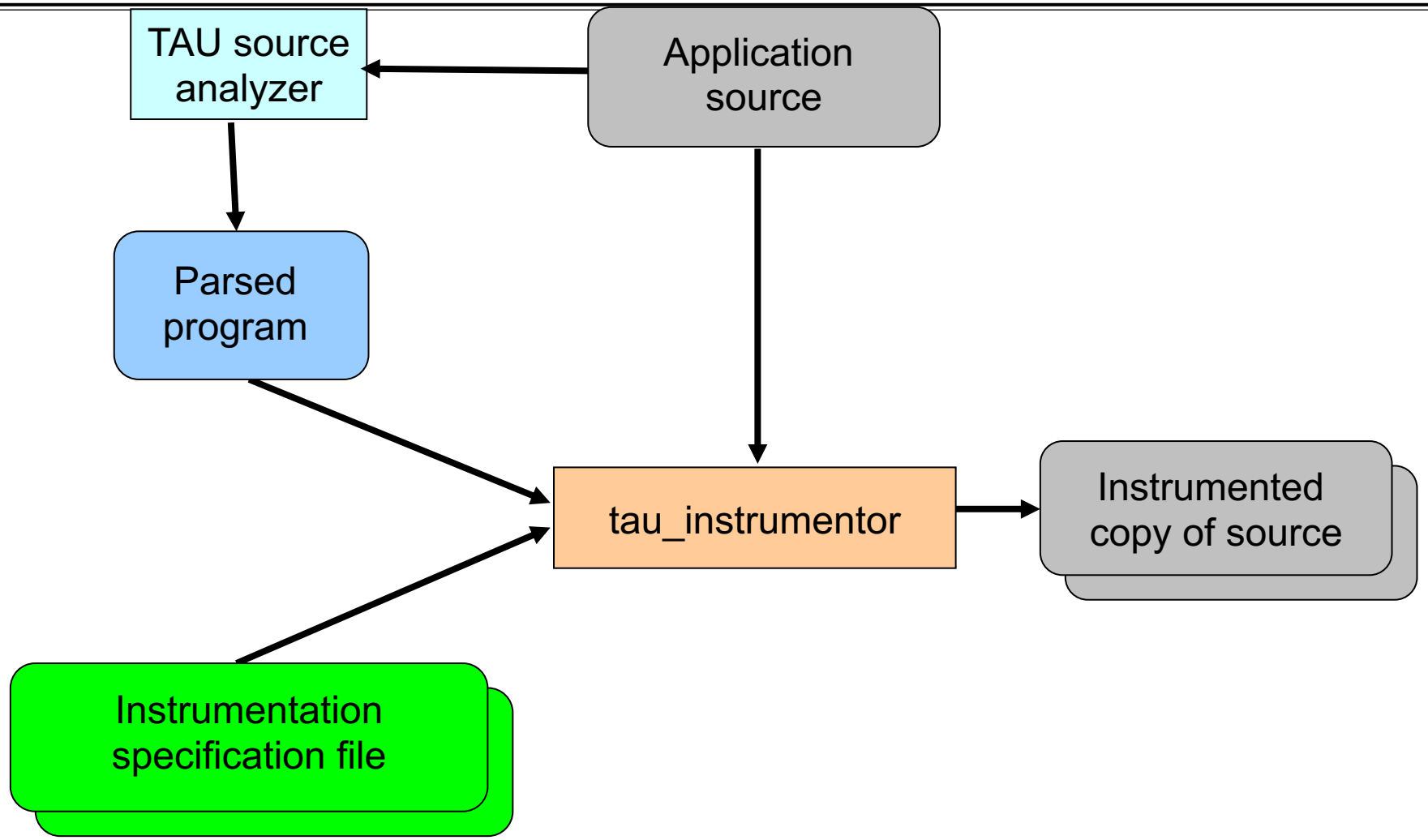
- tau_exec can enable event based sampling while launching the executable using the –ebs flag!

TAU's Source Instrumentation

TAU's Static Analysis System: Program Database Toolkit (PDT)



Automatic Source Instrumentation using PDT



Installing and Configuring TAU

- Installing PDT:
  - wget http://tau.uoregon.edu/pdt.tgz
  - ./configure; make ; make install
- Installing TAU:
  - wget http://tau.uoregon.edu/tau.tgz
  - ./configure –ompt –c++=mpiicpc –cc=mpiicc –fortran=mpiifort -mpi -bfd=download -pdt=<dir> -papi=<dir> ...
  - make install; export PATH=<taudir>/x86_64/bin:$PATH
- Using TAU for source instrumentation:
  - export TAU_MAKEFILE=<taudir>/x86_64/lib/Makefile.tau-<TAGS>
  - make CC=tau_cc.sh CXX=tau_cxx.sh F90=tau_f90.sh

Installing TAU on your laptop for paraprof (GUI)

- Microsoft Windows
  - Install Java from Oracle.com
  - http://tau.uoregon.edu/tau.exe
  - Install, click on a ppk file to launch paraprof
- macOS
  - Install Java 11.0.3:
    - Download http://tau.uoregon.edu/java.dmg
    - If you have multiple Java installations, add to your ~/.zshrc (or ~/.bashrc as appropriate):
    - export PATH=/Library/Java/JavaVirtualMachines/jdk-11.0.3.jdk/Contents/Home/bin:$PATH
    - java -version
  - Download and install TAU (copy to /Applications from dmg):
    - http://tau.uoregon.edu/tau.dmg
    - export PATH=/Applications/TAU/tau/apple/bin:$PATH
    - paraprof app.ppk &
- Linux (http://tau.uoregon.edu/tau.tgz)
  - ./configure; make install; export PATH=<taudir>/x86_64/bin:$PATH
  - paraprof app.ppk &

Using TAU on Hawk at HLRS

- Allocate a single node (Reservation: R_tw):
  - qsub -I –q R_tw -l select=1:mpiprocs=128 -l walltime=0:30:00
- Load the TAU module for the workshop:
  - module use /lustre/cray/ws9/6/ws/xhpsashe-TAU/modulefiles
  - module load tau
- Copy the workshop examples to your workspace:
  - cp -r /lustre/cray/ws9/6/ws/xhpsashe-TAU/workshop .
- Build an example and run with TAU:
  - cd workshop/NPB3.3-MZ-MPI
  - make clean; make -j8; cd bin; ./clean.sh; cat r.sh rt.sh
  - ./rt.sh
  - pprof –a | more ; paraprof --pack bt.ppk; <scp to your laptop and launch with paraprof>; paraprof bt.ppk

Using TAU's Source Code Instrumentation

- 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 use /lustre/cray/ws9/6/ws/xhpsashe-TAU/modulefiles
% module load tau
% export TAU_MAKEFILE=$TAU/Makefile.tau-papi-mpi-pdt-openmp-opari
% export TAU_OPTIONS='-optVerbose …' (see tau_compiler.sh )
```

  - Use tau_f90.sh, tau_f77.sh , tau_cxx.sh, tau_upc.sh, or tau_cc.sh as F90, F77, 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)

Makefiles for TAU Compiler and Runtime Options (Hawk)

```
% module use /lustre/cray/ws9/6/ws/xhpsashe-TAU/modulefiles ; module load tau
% echo $TAU
/lustre/cray/ws9/6/ws/xhpsashe-TAU/pkgs/tau-2.29.1/x86_64/lib
% ls $TAU/Makefile.*
```

- /lustre/cray/ws9/6/ws/xhpsashe-TAU/pkgs/tau-2.29.1/x86_64/lib/Makefile.tau-icpc-papi-ompt-v5-mpi-pdt-openmp
- /lustre/cray/ws9/6/ws/xhpsashe-TAU/pkgs/tau-2.29.1/x86_64/lib/Makefile.tau-mpi-pdt-openmp-opari
- /lustre/cray/ws9/6/ws/xhpsashe-TAU/pkgs/tau-2.29.1/x86_64/lib/Makefile.tau-mpi-pthread-pdt
- /lustre/cray/ws9/6/ws/xhpsashe-TAU/pkgs/tau-2.29.1/x86_64/lib/Makefile.tau-papi-mpi-pdt-openmp-opari
- /lustre/cray/ws9/6/ws/xhpsashe-TAU/pkgs/tau-2.29.1/x86_64/lib/Makefile.tau-papi-mpi-pthread-pdt
- /lustre/cray/ws9/6/ws/xhpsashe-TAU/pkgs/tau-2.29.1/x86_64/lib/Makefile.tau-pdt
- /lustre/cray/ws9/6/ws/xhpsashe-TAU/pkgs/tau-2.29.1/x86_64/lib/Makefile.tau-pthread-pdt

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

$TAU/Makefile.tau-icpc-papi-ompt-v5-mpi-pdt-openmp

Supports MPI instrumentation & PDT for automatic source instrumentation

```
% export TAU_MAKEFILE=$TAU/tau-icpc-papi-ompt-v5-mpi-pdt-openmp
% tau_f90.sh matmult.f90 -o matmult
% mpirun -np 16 ./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

-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.

…

Using TAU_OPTIONS

- 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

Selective Instrumentation File With Program Database Toolkit (PDT)

- To use an instrumentation specification file for source instrumentation:

  % export TAU_OPTIONS= '-optTauSelectFile=/path/to/select.tau –optVerbose '
  % cat select.tau

```
BEGIN_EXCLUDE_LIST
BINVCRHS
MATMUL_SUB
MATVEC_SUB
EXACT_SOLUTION
BINVRHS
LHS#INIT
TIMER_#
END_EXCLUDE_LIST
```

  - NOTE: paraprof can create this file from an earlier execution for you based on throttling.
  - File -> Create Selective Instrumentation File -> save
  - Selective instrumentation at runtime:

```
% export TAU_SELECT_FILE=select.tau
```

Simplifying TAU's usage (tau_exec)

- Uninstrumented execution linked with –dynamic (dynamic executables only!)
  % mpirun -np 16 ./a.out
- Track MPI performance
  % mpirun -np 16 tau_exec ./a.out
- Track OpenMP, and MPI performance (MPI enabled by default; OMPT in Clang 9+, Intel 19+)
  % export TAU_OMPT_SUPPORT_LEVEL=full;
  % mpirun -np 16 tau_exec –T mpi,pdt,ompt,v5,papi –ompt ./a.out
- Track memory operations
  % export TAU_TRACK_MEMORY_LEAKS=1
  % mpirun -np 16 tau_exec –memory_debug ./a.out (bounds check)
- Use event based sampling (compile with –g)
  % mpirun -np 16 tau_exec –ebs ./a.out
  Also –ebs_source=<PAPI_COUNTER> -ebs_period=<overflow_count> -ebs_resolution=<file|function|line>
- Load wrapper interposition library
  % mpirun -np 16 tau_exec –loadlib=<path/libwrapper.so> ./a.out
- Track GPGPU operations (-rocm, -opencl, -cupti, -cupti –um, -openacc):
  % mpirun -np 16 tau_exec –cupti ./a.out

Configuration tags for tau_exec

```
% ./configure -pdt=<dir> -mpi -papi=<dir>; make install
```
Creates in $TAU:
```
Makefile.tau-papi-mpi-pdt(Configuration parameters in stub makefile)
shared-papi-mpi-pdt/libTAU.so

% ./configure -pdt=<dir> -mpi; make install   creates
Makefile.tau-mpi-pdt
shared-mpi-pdt/libTAU.so
```
To explicitly choose preloading of shared-<options>/libTAU.so change:
```
% mpirun -np 256 ./a.out      to
% mpirun -np 256  tau_exec -T <comma_separated_options> ./a.out

% mpirun -np 256  tau_exec -T papi,mpi,pdt ./a.out
```
Preloads `$TAU/shared-papi-mpi-pdt/libTAU.so`
```
% mpirun -np 256  tau_exec -T papi ./a.out
```
Preloads `$TAU/shared-papi-mpi-pdt/libTAU.so by matching.`
```
% aprun -n 256 tau_exec -T papi,mpi,pdt -s ./a.out
```
Does not execute the program. Just displays the library that it will preload if executed without the –s option.
NOTE: -mpi configuration is selected by default. Use –T serial for
Sequential programs.

TAU's Runtime Environment Variables

Environment Variable	Default	Description
TAU_TRACE	0	Setting to 1 turns on tracing
TAU_CALLPATH	0	Setting to 1 turns on callpath profiling
TAU_TRACK_MEMORY_FOOTPRINT	0	Setting to 1 turns on tracking memory usage by sampling periodically the resident set size and high water mark of memory usage
TAU_TRACK_POWER	0	Tracks power usage by sampling 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. Throttle a routine if it is called over 100000 times and takes less than 10 usec of inclusive time per call
TAU_CALLSITE	0	Setting to 1 enables callsite profiling that shows where an instrumented function was called. Also compatible with tracing.
TAU_PROFILE_FORMAT	Profile	Setting to "merged" generates a single file. "snapshot" generates xml format

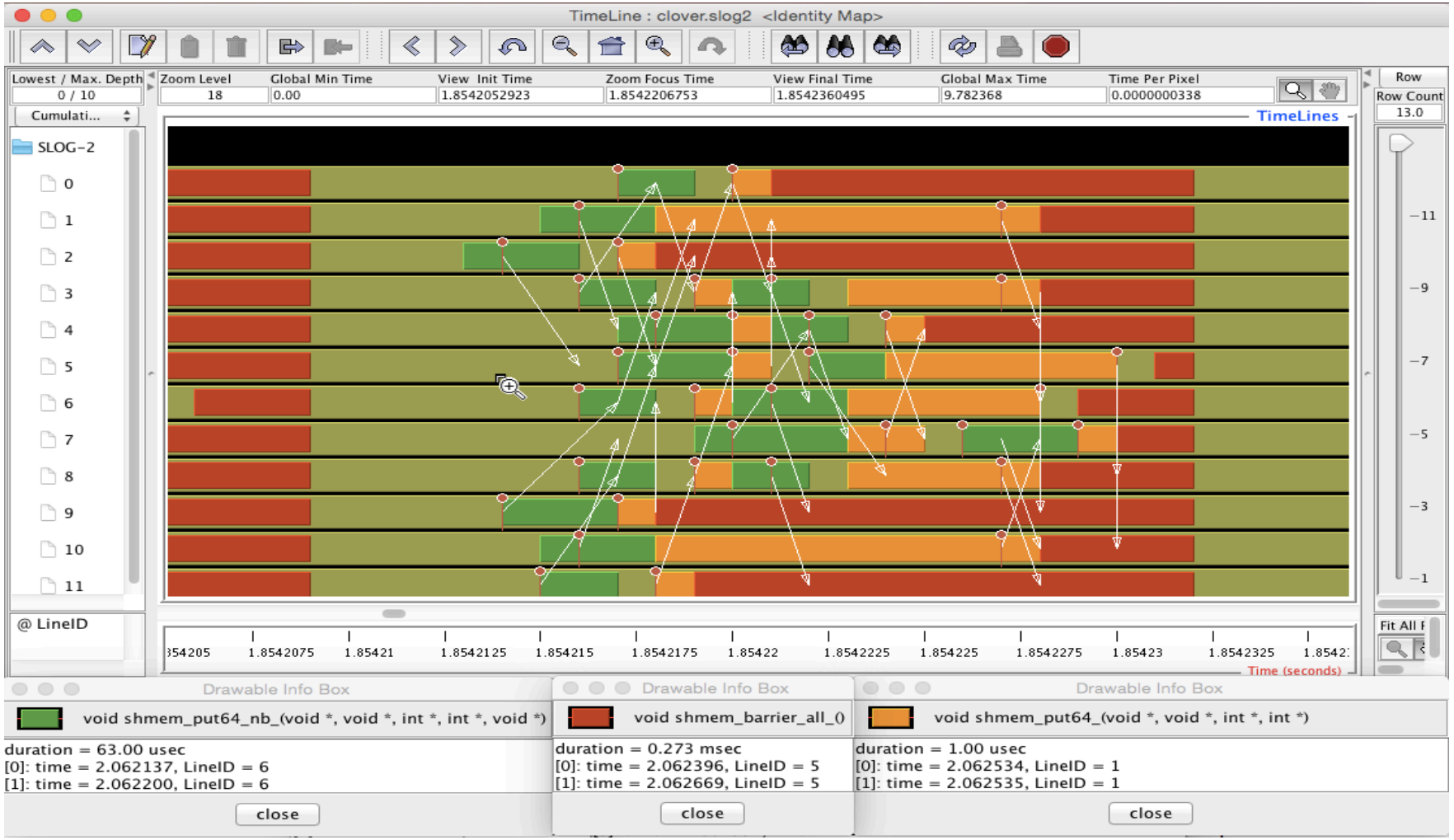
Runtime Environment Variables

Environment Variable	Default	Description
TAU_METRICS	TIME	Setting to a comma separated list generates other metrics. (e.g., ENERGY,TIME,P_VIRTUAL_TIME,PAPI_FP_INS,PAPI_NATIVE_<event>:<subevent>)
TAU_TRACE	0	Setting to 1 turns on tracing
TAU_TRACE_FORMAT	Default	Setting to “otf2” turns on TAU’s native OTF2 trace generation (configure with –otf=download)
TAU_EBS_UNWIND	0	Setting to 1 turns on unwinding the callstack during sampling (use with tau_exec –ebs or TAU_SAMPLING=1)
TAU_EBS_RESOLUTION	line	Setting to “function” or “file” changes the sampling resolution to function or file level respectively.
TAU_TRACK_LOAD	0	Setting to 1 tracks system load on the node
TAU_SELECT_FILE	Default	Setting to a file name, enables selective instrumentation based on exclude/include lists specified in the file.
TAU_OMPT_SUPPORT_LEVEL	basic	Setting to “full” improves resolution of OMPT TR6 regions on threads 1.. N-1. Also, “lowoverhead” option is available.
TAU_OMPT_RESOLVE_ADDRESS_EAGERLY	1	Setting to 1 is necessary for event based sampling to resolve addresses with OMPT. Setting to 0 allows the user to do offline address translation.
TAU_EVENT_THRESHOLD	0.5	Define a threshold value (e.g., .25 is 25%) to trigger marker events for min/max

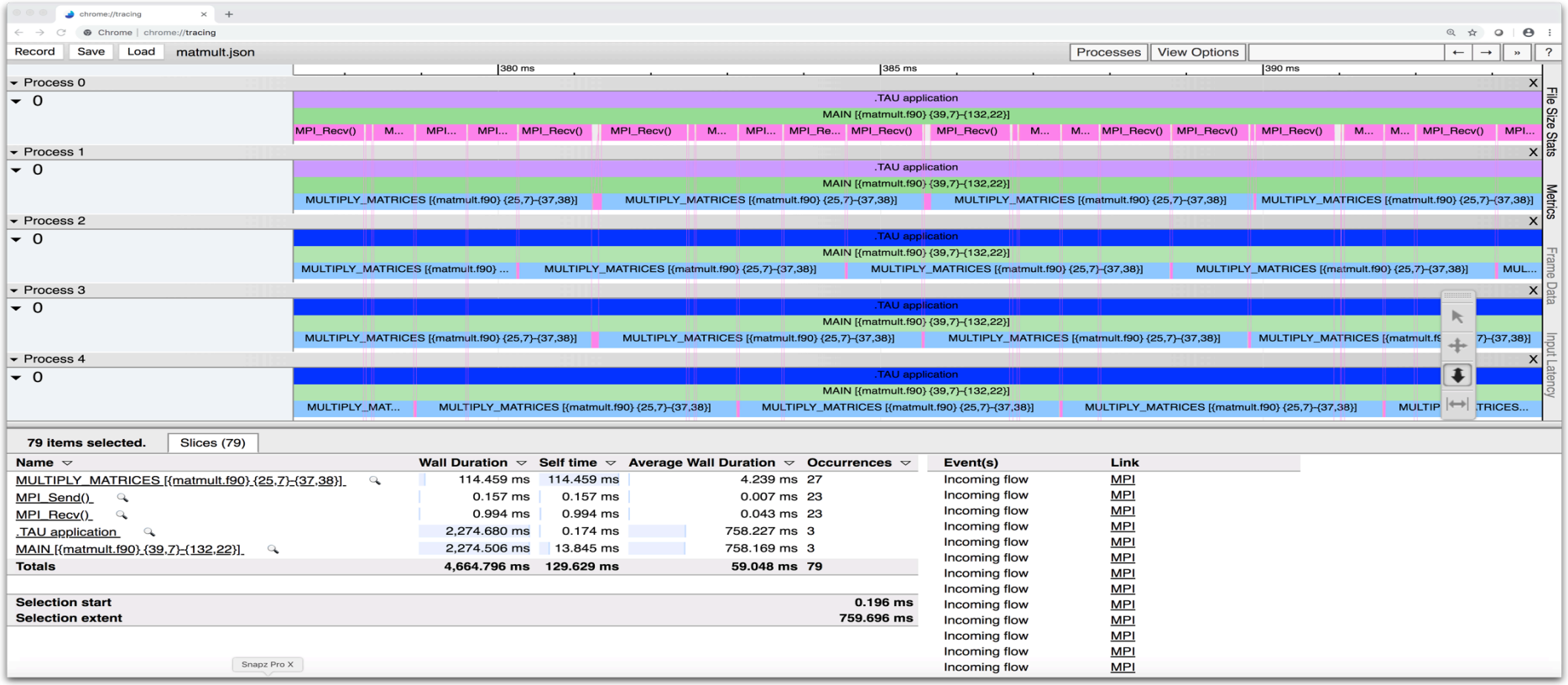
Runtime Environment Variables

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

Tracing: Jumpshot (ships with TAU)



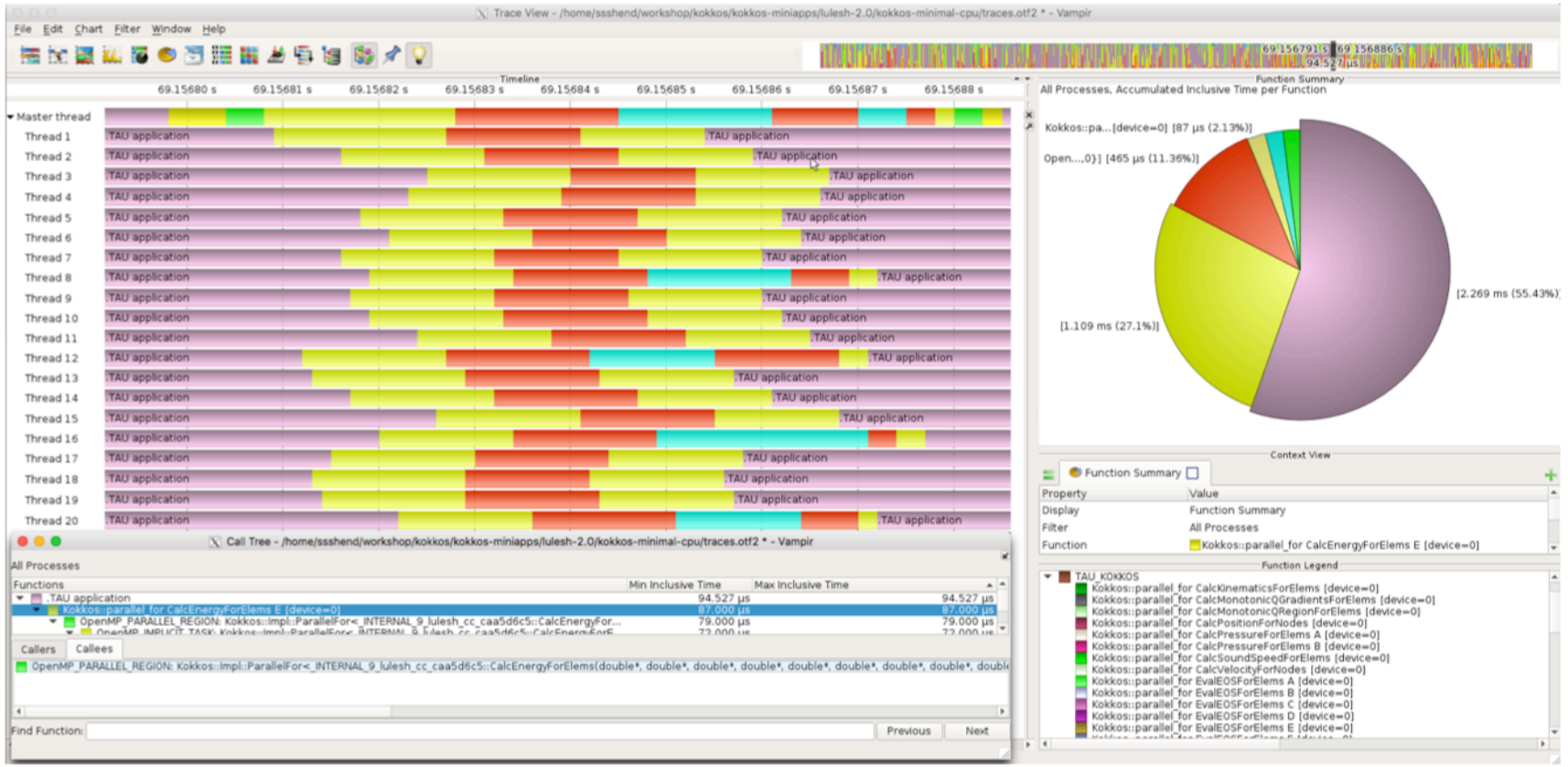
Tracing: Chrome Browser




```
% export TAU_TRACE=1
% mpirun –np 256 tau_exec ./a.out
% tau_treemerge.pl; tau_trace2json tau.trc tau.edf –chrome –ignoreatomic –o app.json
```

Chrome browser: chrome://tracing (Load -> app.json)

Vampir [TU Dresden] Timeline: Kokkos




```
% export TAU_TRACE=1; export TAU_TRACE_FORMAT=otf2
% tau_exec –ompt  ./a.out
% vampir traces.otf2 &
```

Kokkos

- Provides abstractions for node level parallelism (X in MPI+X)
- Productive, portable, and performant shared-memory programming model
- Helps you create single source performance portable codes
- Provides data abstractions
- C++ API for expressing parallelism in your program
- Aggressive compiler transformations using C++ templates
- Low level code targets backends such as OpenMP, Pthread, CUDA
- Creates a problem for performance evaluation tools
- Gap: performance data and higher-level abstractions
- Solution: Kokkos profiling API for mapping performance data

Kokkos API use in ExaMiniMD

20. sameer@pegasus:~/pkgs/ORNL/DEMO/BUILD/ExaMiniMD-pthread/ExaMiniMD/src/comm_types (ssh)

```
void CommMPI::update_halo() {

  Kokkos::Profiling::pushRegion("Comm::update_halo");

  N_ghost = 0;
  s=*system;

  pack_buffer_update = t_buffer_update((T_X_FLOAT*)pack_buffer.data(),pack_indicies_all.extent(1));
  unpack_buffer_update = t_buffer_update((T_X_FLOAT*)unpack_buffer.data(),pack_indicies_all.extent(1));

  for(phase = 0; phase<6; phase++) {
    pack_indicies = Kokkos::subview(pack_indicies_all,phase,Kokkos::ALL());
    if(proc_grid[phase/2]>1) {

      Kokkos::parallel_for("CommMPI::halo_update_pack",
         Kokkos::RangePolicy<TagHaloUpdatePack, Kokkos::IndexType<T_INT> >(0,proc_num_send[phase]),
         *this);
      MPI_Request request;
      MPI_Status status;
      MPI_Irecv(unpack_buffer.data(),proc_num_recv[phase]*sizeof(T_X_FLOAT)*3/sizeof(int),MPI_INT, proc_neighbors_recv[phase],100002,MPI_COMM_WORLD,&request);
      MPI_Send (pack_buffer.data(),proc_num_send[phase]*sizeof(T_X_FLOAT)*3/sizeof(int),MPI_INT, proc_neighbors_send[phase],100002,MPI_COMM_WORLD);
      s = *system;
      MPI_Wait(&request,&status);
      const int count = proc_num_recv[phase];
      if(unpack_buffer_update.extent(0)<count) {
        unpack_buffer_update = t_buffer_update((T_X_FLOAT*)unpack_buffer.data(),count);
      }
      Kokkos::parallel_for("CommMPI::halo_update_unpack",
                Kokkos::RangePolicy<TagHaloUpdateUnpack, Kokkos::IndexType<T_INT> >(0,proc_num_recv[phase]),
                *this);

    } else {
      //printf("HaloUpdateCopy: %i %i %i\n",phase,proc_num_send[phase],pack_indicies.extent(0));
      Kokkos::parallel_for("CommMPI::halo_update_self",
        Kokkos::RangePolicy<TagHaloUpdateSelf, Kokkos::IndexType<T_INT> >(0,proc_num_send[phase]),
        *this);
    }
    N_ghost += proc_num_recv[phase];
  }

  Kokkos::Profiling::popRegion();
};
```

pushRegion("Comm::update_halo")

Kokkos::parallel_for

popRegion

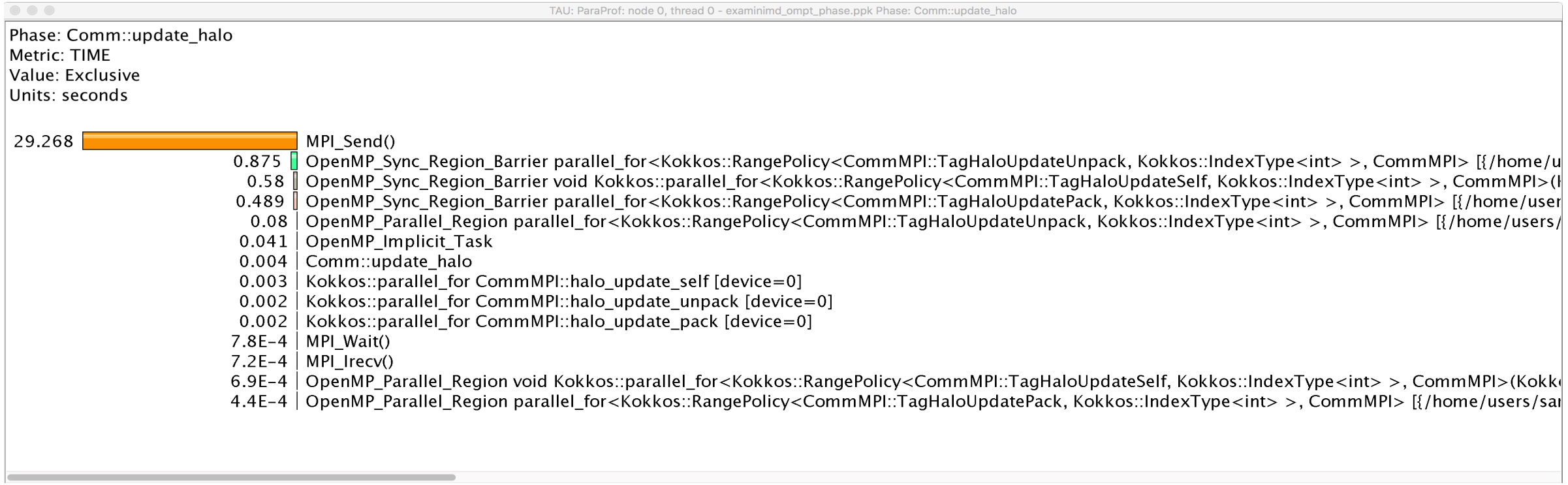
ExaMiniMD: TAU Phase

TAU: ParaProf: Statistics for: node 0, thread 0 - examinimd_ompt_phase.ppk

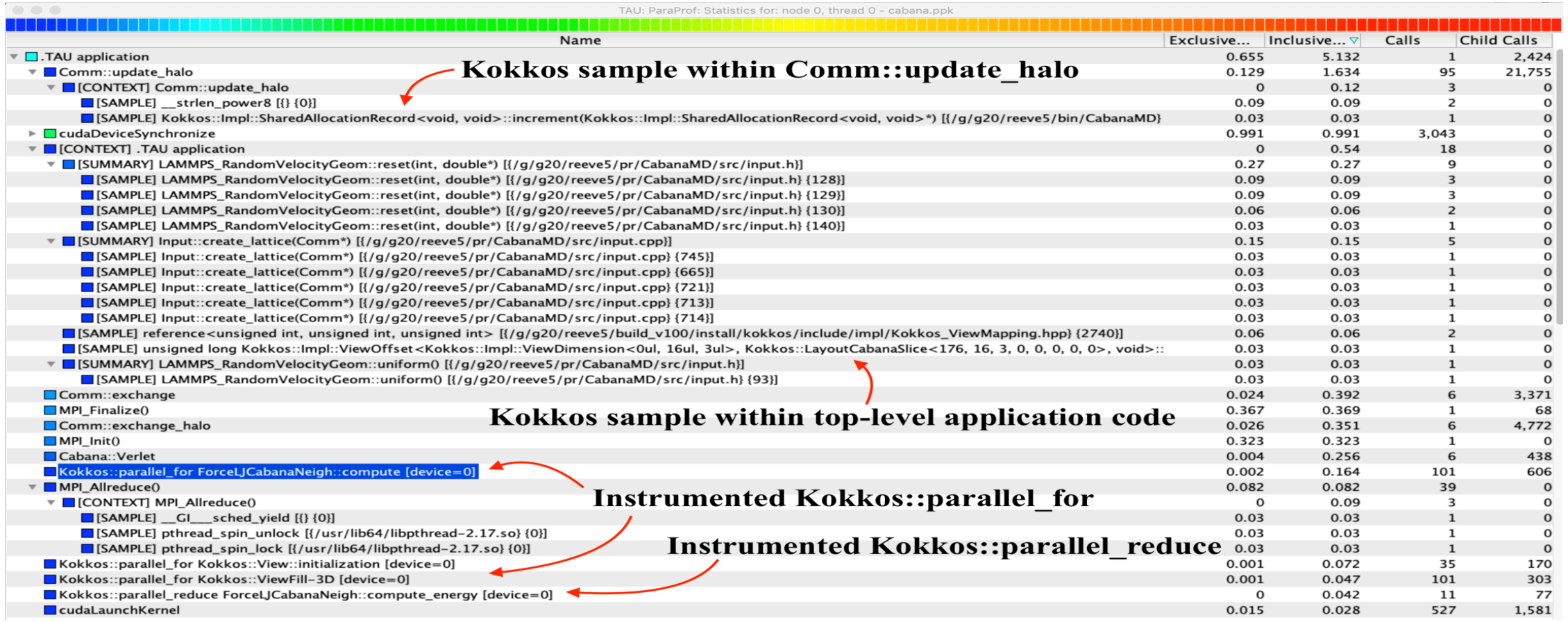
Name	Exclusive TIME	Inclusive TIME	Calls	Child Calls
.TAU application	0.143	96.743	1	832
Comm::exchange	0.001	0.967	6	142
Comm::exchange_halo	0.001	4.702	6	184
Comm::update_halo	0.004	31.347	95	1,330
Kokkos::parallel_for CommMPI::halo_update_pack [device=0]	0.002	0.506	190	190
Kokkos::parallel_for CommMPI::halo_update_self [device=0]	0.003	0.597	380	380
Kokkos::parallel_for CommMPI::halo_update_unpack [device=0]	0.002	0.97	190	190
MPI_Irecv()	0.001	0.001	190	0
MPI_Send()	29.268	29.268	190	0
MPI_Wait()	0.001	0.001	190	0
OpenMP_Implicit_Task	0.041	1.985	760	760
OpenMP_Parallel_Region parallel_for<Kokkos::RangePolicy<CommMPI::Ta	0	0.504	190	190
OpenMP_Parallel_Region parallel_for<Kokkos::RangePolicy<CommMPI::Ta	0.08	0.968	190	190
OpenMP_Parallel_Region void Kokkos::parallel_for<Kokkos::RangePolicy<(	0.001	0.594	380	380
OpenMP_Sync_Region_Barrier parallel_for<Kokkos::RangePolicy<CommMF	0.489	0.489	190	0
OpenMP_Sync_Region_Barrier parallel_for<Kokkos::RangePolicy<CommMF	0.875	0.875	190	0
OpenMP_Sync_Region_Barrier void Kokkos::parallel_for<Kokkos::RangePol	0.58	0.58	380	0

Comm::update_halo phase in TAU ParaProf's Thread Statistics Table

ExaMiniMD: ParaProf Node Window



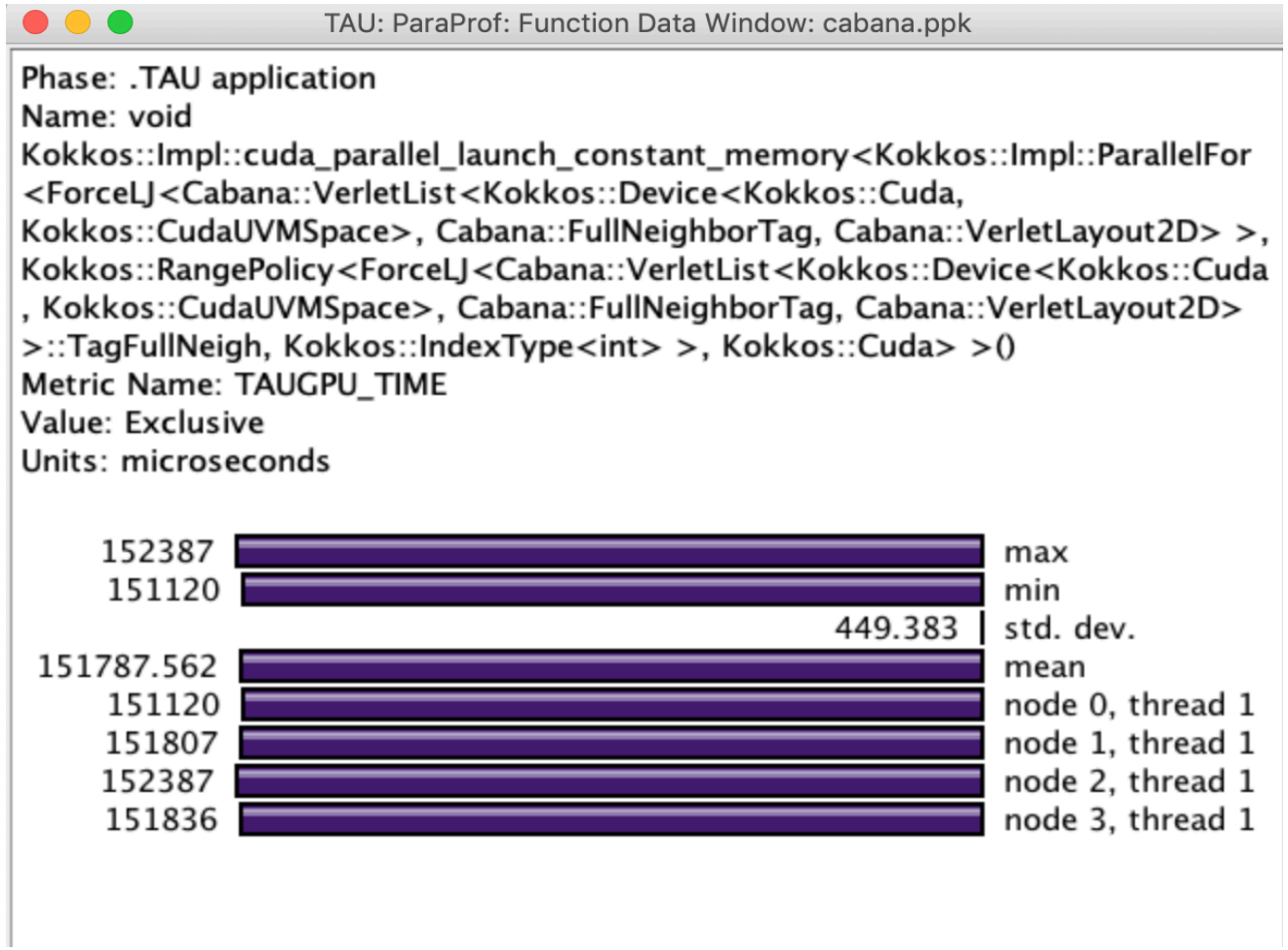
Event-based Sampling (EBS): CabanaMD with Kokkos



Name	Exclusive...	Inclusive...	Calls	Child Calls
.TAU application	0.655	5.132	1	2,424
Comm::update_halo	0.129	1.634	95	21,755
[CONTEXT] Comm::update_halo	0	0.12	3	0
[SAMPLE] __strlen_power8 [{} {0}]	0.09	0.09	2	0
[SAMPLE] Kokkos::Impl::SharedAllocationRecord<void, void>::increment(Kokkos::Impl::SharedAllocationRecord<void, void>*) [{/g/g20/reeve5/bin/CabanaMD}	0.03	0.03	1	0
cudaDeviceSynchronize	0.991	0.991	3,043	0
[CONTEXT] .TAU application	0	0.54	18	0
[SUMMARY] LAMMPS_RandomVelocityGeom::reset(int, double*) [{/g/g20/reeve5/pr/CabanaMD/src/input.h}]	0.27	0.27	9	0
[SAMPLE] LAMMPS_RandomVelocityGeom::reset(int, double*) [{/g/g20/reeve5/pr/CabanaMD/src/input.h} {128}]	0.09	0.09	3	0
[SAMPLE] LAMMPS_RandomVelocityGeom::reset(int, double*) [{/g/g20/reeve5/pr/CabanaMD/src/input.h} {129}]	0.09	0.09	3	0
[SAMPLE] LAMMPS_RandomVelocityGeom::reset(int, double*) [{/g/g20/reeve5/pr/CabanaMD/src/input.h} {130}]	0.06	0.06	2	0
[SAMPLE] LAMMPS_RandomVelocityGeom::reset(int, double*) [{/g/g20/reeve5/pr/CabanaMD/src/input.h} {140}]	0.03	0.03	1	0
[SUMMARY] Input::create_lattice(Comm*) [{/g/g20/reeve5/pr/CabanaMD/src/input.cpp}]	0.15	0.15	5	0
[SAMPLE] Input::create_lattice(Comm*) [{/g/g20/reeve5/pr/CabanaMD/src/input.cpp} {745}]	0.03	0.03	1	0
[SAMPLE] Input::create_lattice(Comm*) [{/g/g20/reeve5/pr/CabanaMD/src/input.cpp} {665}]	0.03	0.03	1	0
[SAMPLE] Input::create_lattice(Comm*) [{/g/g20/reeve5/pr/CabanaMD/src/input.cpp} {721}]	0.03	0.03	1	0
[SAMPLE] Input::create_lattice(Comm*) [{/g/g20/reeve5/pr/CabanaMD/src/input.cpp} {713}]	0.03	0.03	1	0
[SAMPLE] Input::create_lattice(Comm*) [{/g/g20/reeve5/pr/CabanaMD/src/input.cpp} {714}]	0.03	0.03	1	0
[SAMPLE] reference<unsigned int, unsigned int, unsigned int> [{/g/g20/reeve5/build_v100/install/kokkos/include/impl/Kokkos_ViewMapping.hpp} {2740}]	0.06	0.06	2	0
[SAMPLE] unsigned long Kokkos::Impl::ViewOffset<Kokkos::Impl::ViewDimension<0ul, 16ul, 3ul>, Kokkos::LayoutCabanaSlice<176, 16, 3, 0, 0, 0, 0, 0>, void>::	0.03	0.03	1	0
[SUMMARY] LAMMPS_RandomVelocityGeom::uniform() [{/g/g20/reeve5/pr/CabanaMD/src/input.h}]	0.03	0.03	1	0
[SAMPLE] LAMMPS_RandomVelocityGeom::uniform() [{/g/g20/reeve5/pr/CabanaMD/src/input.h} {93}]	0.03	0.03	1	0
Comm::exchange	0.024	0.392	6	3,371
MPI_Finalize()	0.367	0.369	1	68
Comm::exchange_halo	0.026	0.351	6	4,772
MPI_Init()	0.323	0.323	1	0
Cabana::Verlet	0.004	0.256	6	438
Kokkos::parallel_for ForceLJCabanaNeigh::compute [device=0]	0.002	0.164	101	606
MPI_Allreduce()	0.082	0.082	39	0
[CONTEXT] MPI_Allreduce()	0	0.09	3	0
[SAMPLE] __GI___sched_yield [{} {0}]	0.03	0.03	1	0
[SAMPLE] pthread_spin_unlock [{/usr/lib64/libpthread-2.17.so} {0}]	0.03	0.03	1	0
[SAMPLE] pthread_spin_lock [{/usr/lib64/libpthread-2.17.so} {0}]	0.03	0.03	1	0
Kokkos::parallel_for Kokkos::View::initialization [device=0]	0.001	0.072	35	170
Kokkos::parallel_for Kokkos::ViewFill-3D [device=0]	0.001	0.047	101	303
Kokkos::parallel_reduce ForceLJCabanaNeigh::compute_energy [device=0]	0	0.042	11	77
cudaLaunchKernel	0.015	0.028	527	1,581

Event-based sampling (EBS) with Kokkos API

CabanaMD: CUDA Events



TAU's Support for Runtime Systems

- *MPI*
  - PMPI profiling interface
  - MPI_T tools interface using performance and control variables
- *Pthread*
  - Captures time spent in routines per thread of execution
- *OpenMP*
  - OMPT tools interface to track salient OpenMP runtime events
  - Opari source rewriter
  - Preloading wrapper OpenMP runtime library when OMPT is not supported
- *OpenACC*
  - OpenACC instrumentation API
  - Track data transfers between host and device (per-variable)
  - Track time spent in kernels

TAU's Support for Runtime Systems (contd.)

- *OpenCL*
  - OpenCL profiling interface
  - Track timings of kernels
- *CUDA*
  - Cuda Profiling Tools Interface (CUPTI)
  - Track data transfers between host and GPU
  - Track access to uniform shared memory between host and GPU
- *ROCm*
  - Rocprofiler and Roctracer instrumentation interfaces
  - Track data transfers and kernel execution between host and GPU
- *Kokkos*
  - Kokkos profiling API
  - Push/pop interface for region, kernel execution interface
- *Python*
  - Python interpreter instrumentation API
  - Tracks Python routine transitions as well as Python to C transitions

Examples of Multi-Level Instrumentation

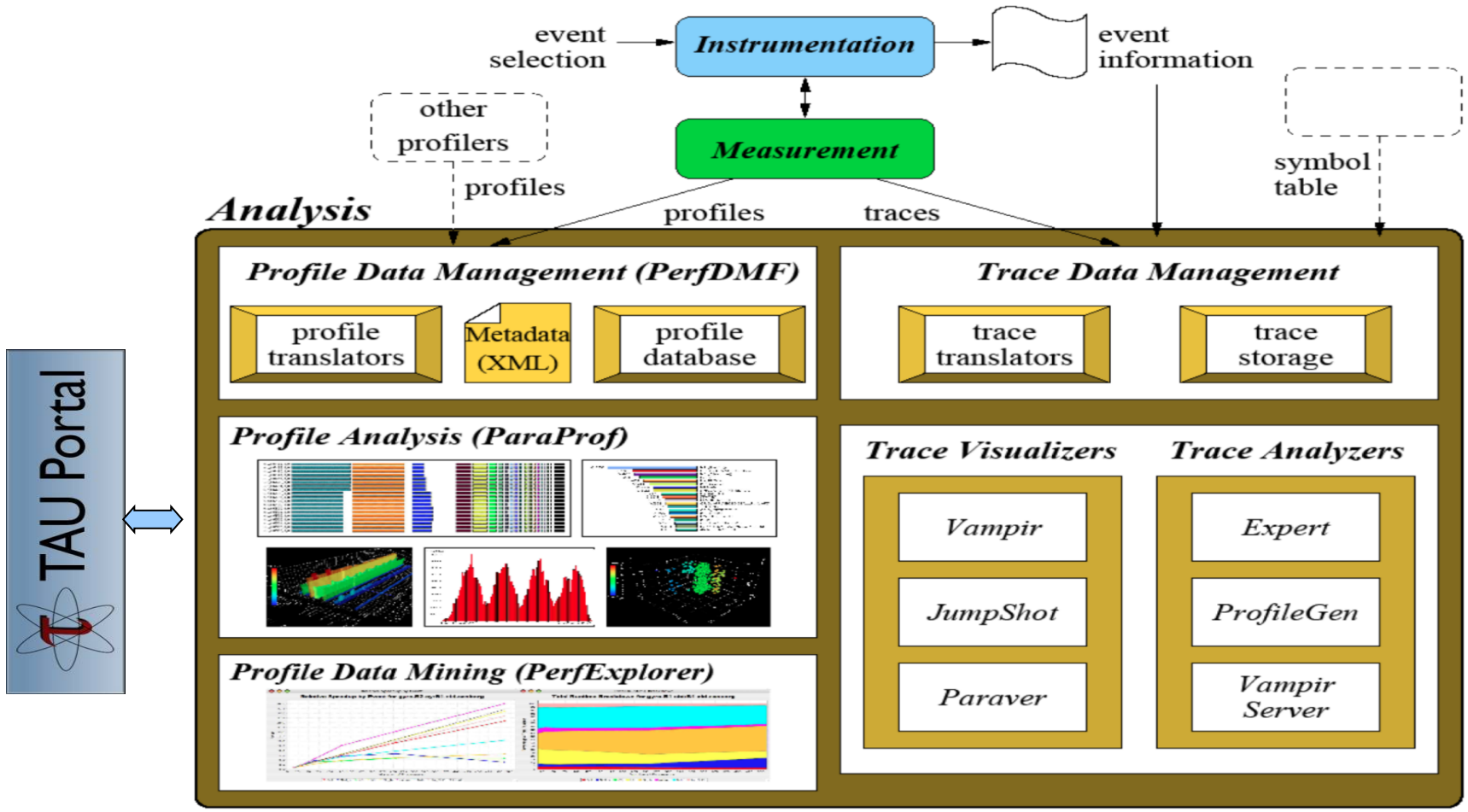
- *MPI + OpenMP*
  - MPI_T + PMPI + OMPT may be used to track MPI and OpenMP
- *MPI + CUDA*
  - PMPI + CUPTI interfaces
- *OpenCL + ROCm*
  - Rocprofiler + OpenCL instrumentation interfaces
- *Kokkos + OpenMP*
  - Kokkos profiling API + OMPT to transparently track events
- *Kokkos + pthread + MPI*
  - Kokkos + pthread wrapper interposition library + PMPI layer
- *Python + CUDA + MPI*
  - Python + CUPTI + pthread profiling interfaces (e.g., Tensorflow, PyTorch) + MPI
- *MPI + OpenCL*
  - PMPI + OpenCL profiling interfaces

TAU Execution Command (tau_exec)

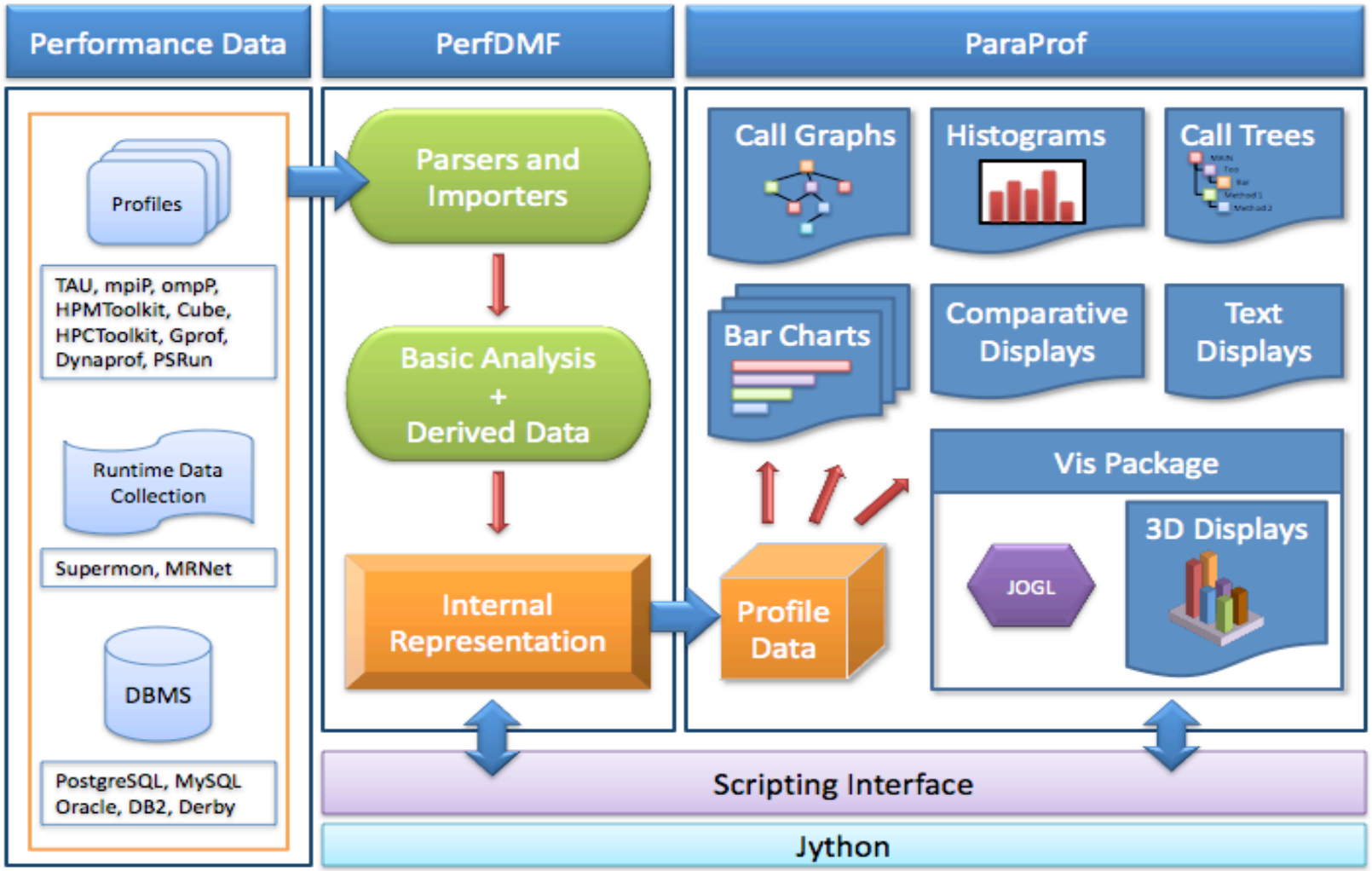
- Uninstrumented execution
  - % mpirun -np 256 ./a.out
- Track GPU operations
  - % mpirun –np 256 tau_exec –rocm ./a.out
  - % mpirun –np 256 tau_exec –cupti ./a.out
  - % mpirun –np 256 tau_exec –opencl ./a.out
  - % mpirun –np 256 tau_exec –openacc ./a.out
- Track MPI performance
  - % mpirun -np 256 tau_exec ./a.out
- Track I/O, and MPI performance (MPI enabled by default)
  - % mpirun -np 256 tau_exec -io ./a.out
- Track OpenMP and MPI execution (using OMPT for Intel v19+ or Clang 8+)
  - % export TAU_OMPT_SUPPORT_LEVEL=full;
  - % mpirun –np 256 tau_exec –T ompt,v5,mpi -ompt ./a.out
- Track memory operations
  - % export TAU_TRACK_MEMORY_LEAKS=1
  - % mpirun –np 256 tau_exec –memory_debug ./a.out (bounds check)
- Use event based sampling (compile with –g)
  - % mpirun –np 256 tau_exec –ebs ./a.out
  - Also -ebs_source=<PAPI_COUNTER> -ebs_period=<overflow_count> -ebs_resolution=<file | function | line>

TAU's Analysis Tools: ParaProf

TAU Analysis



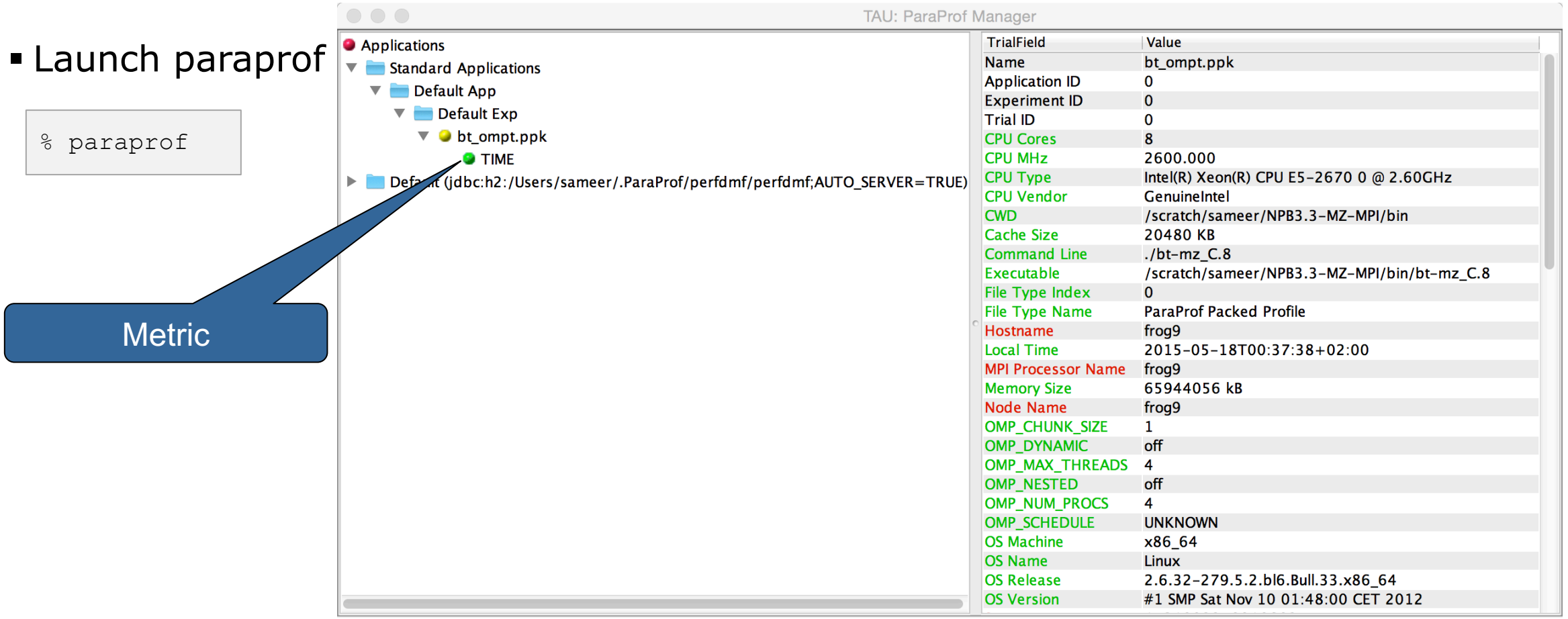
ParaProf Profile Analysis Framework



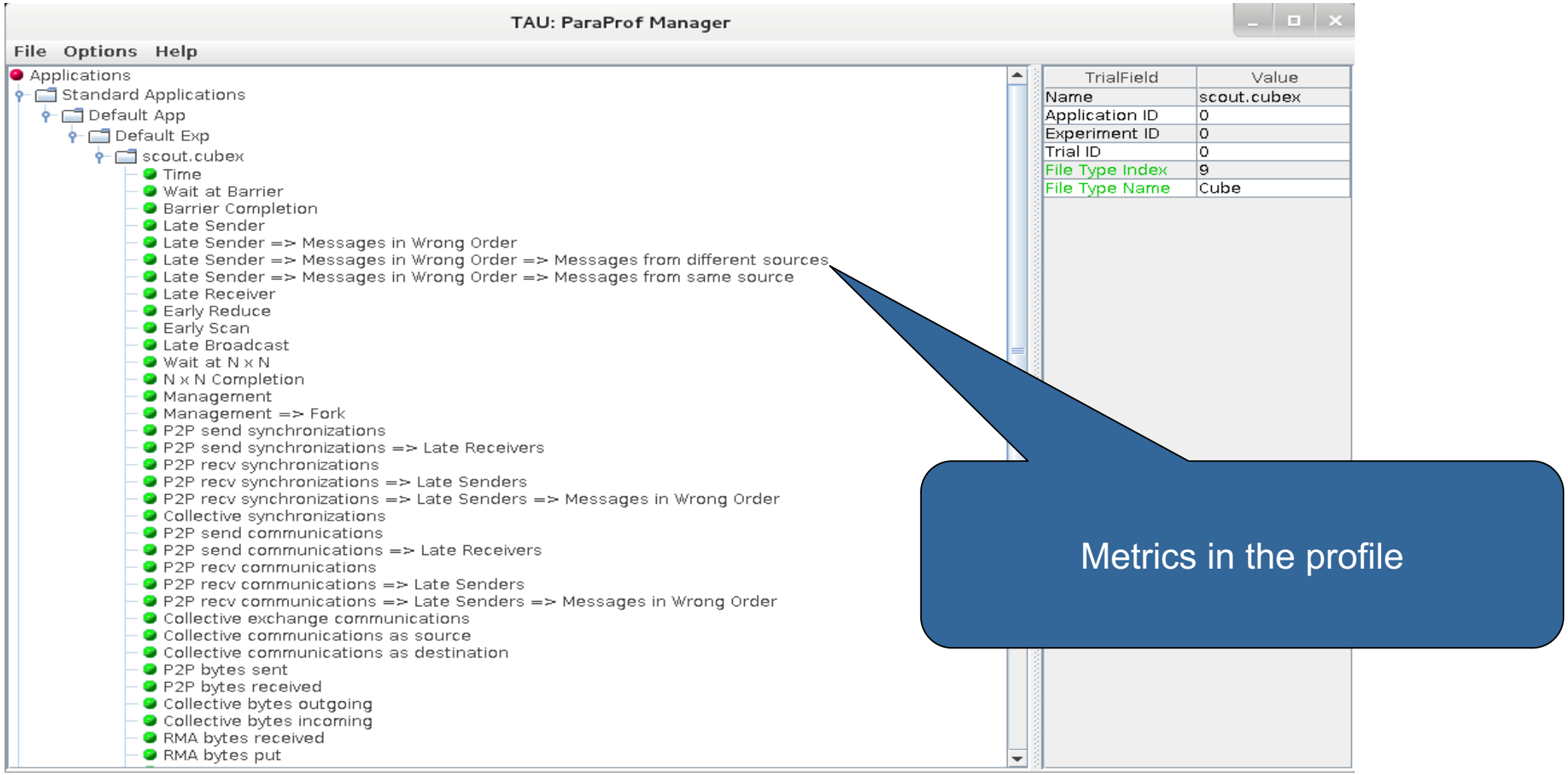
TAU Analysis Tools: paraprof

- Launch paraprof

```
% paraprof
```



ParaProf Manager Widow: scout.cubex

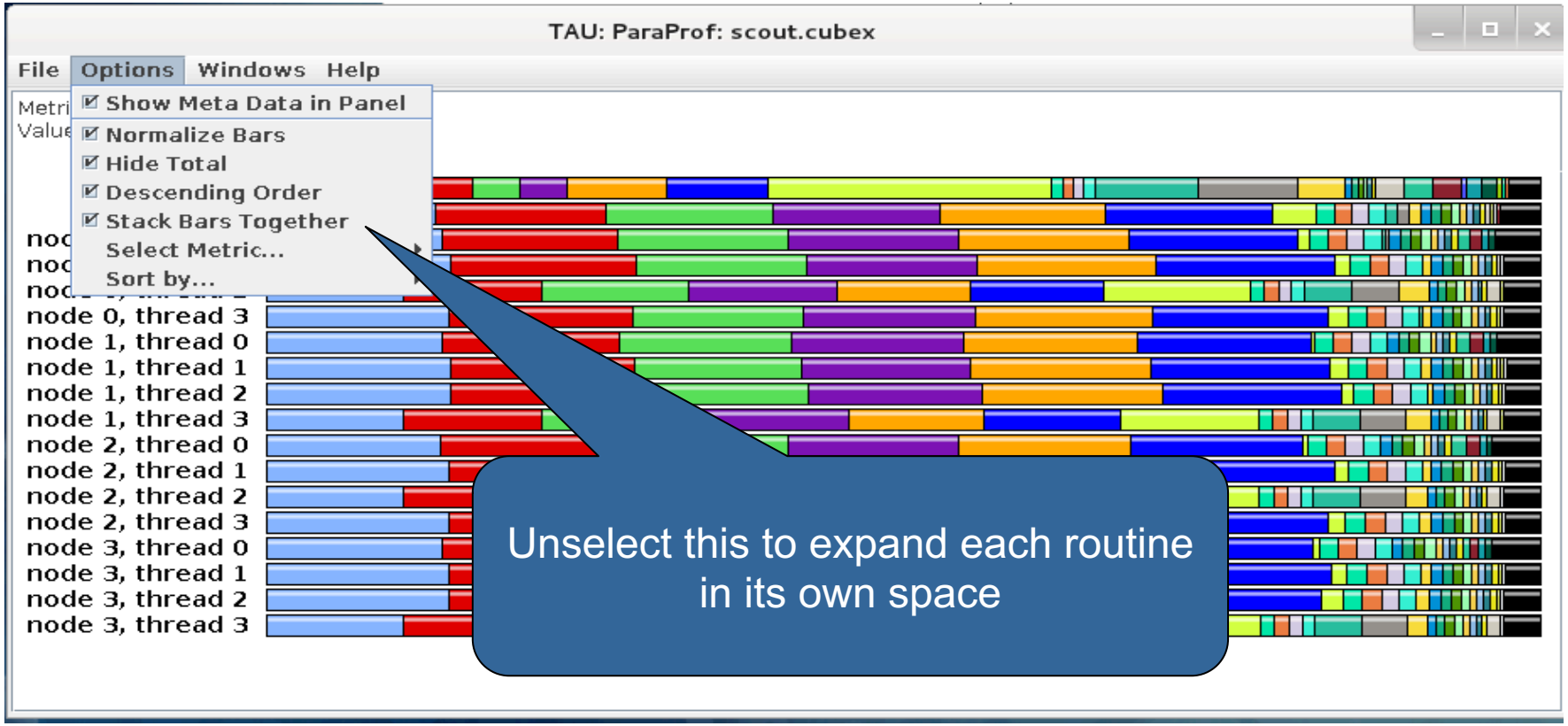


Paraprof main window

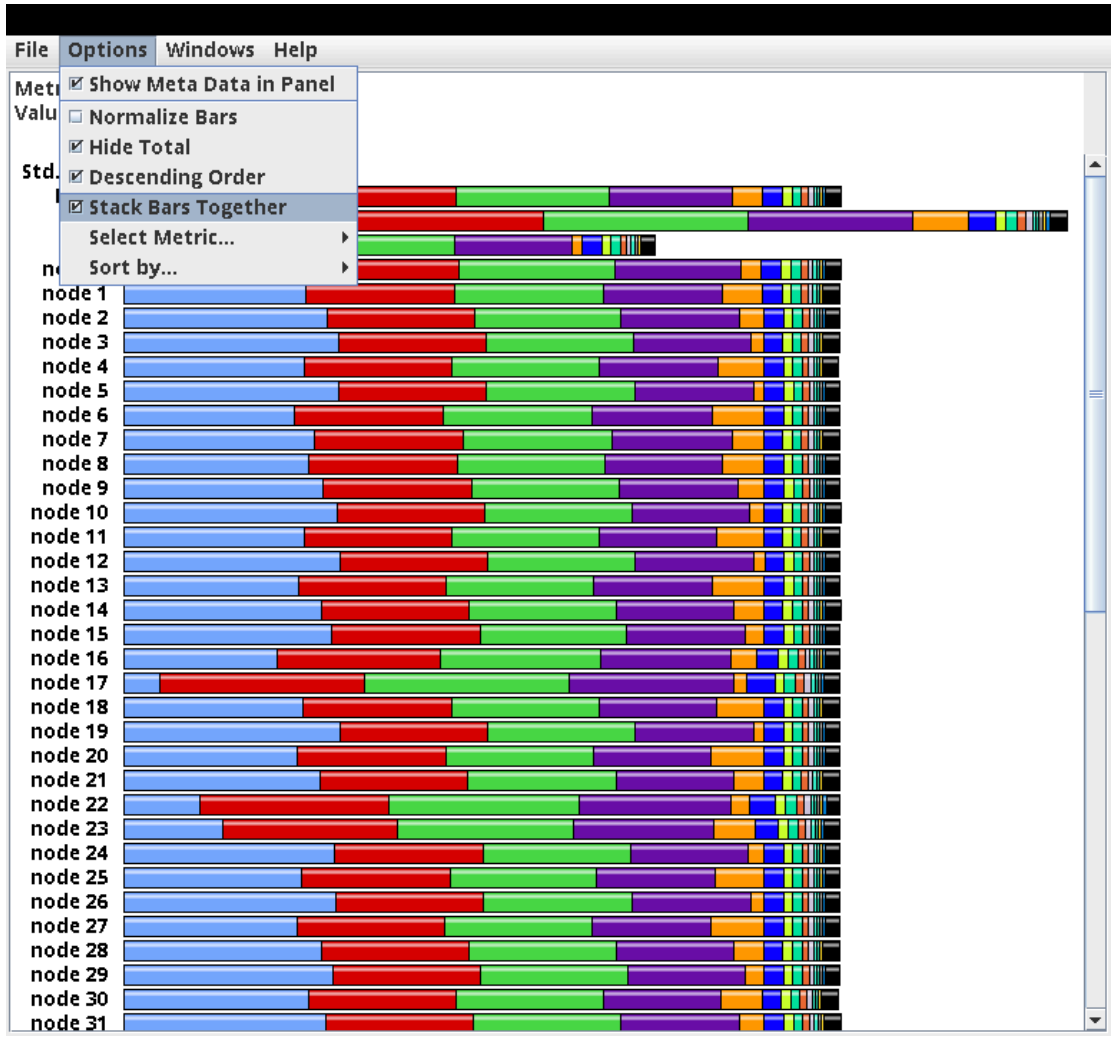
Colors represent code regions

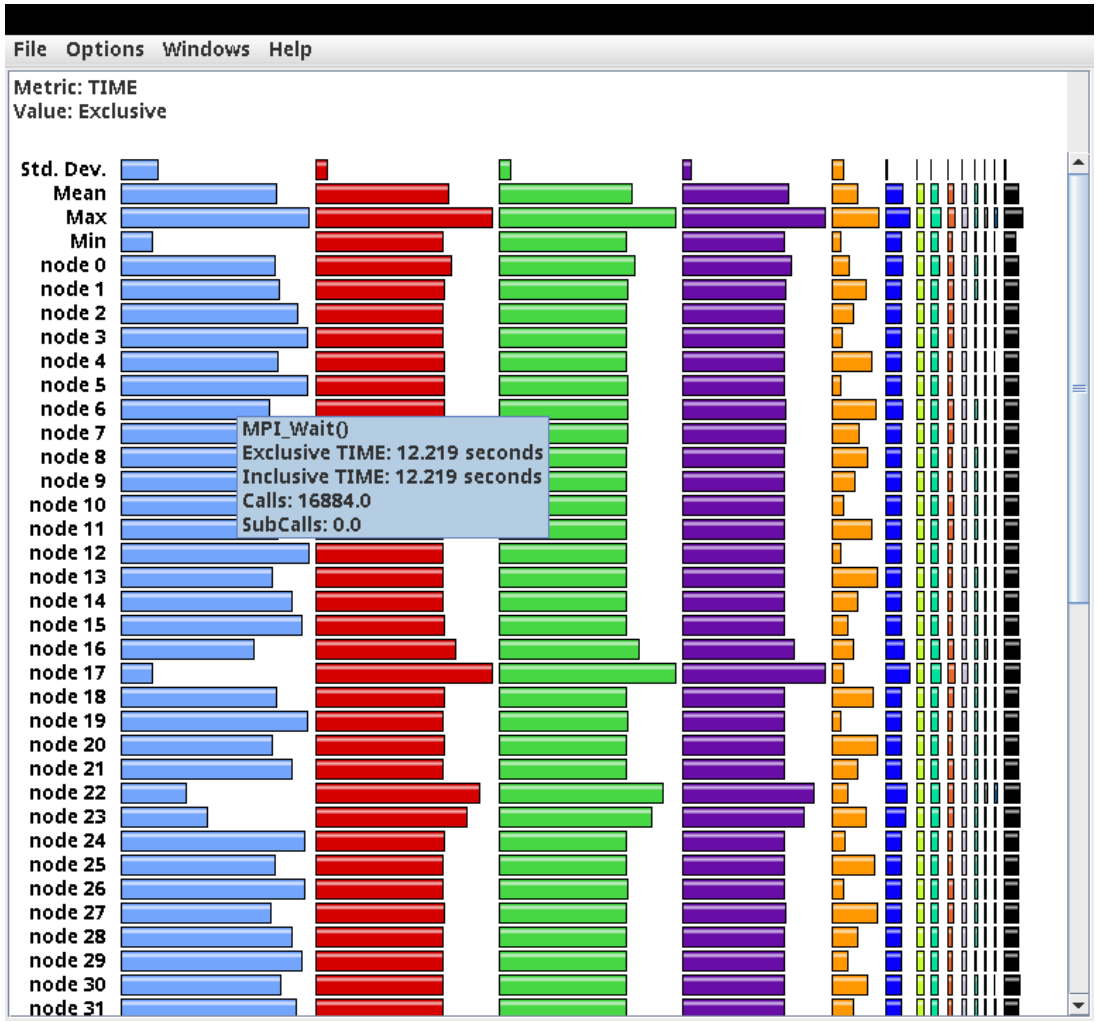
Options -> uncheck Stack Bars Together

Paraprof main window

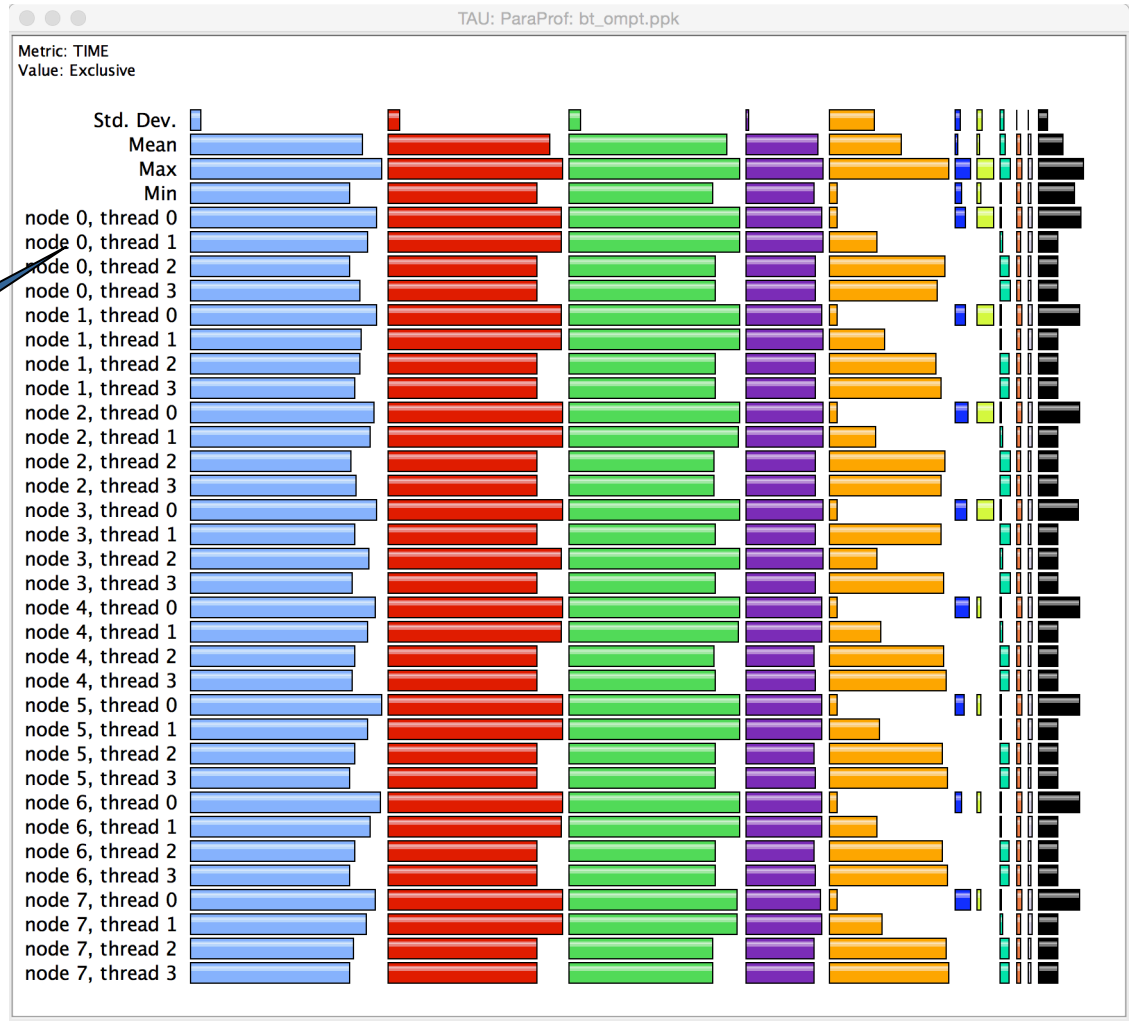


ParaProf Profile Browser





Paraprof main window

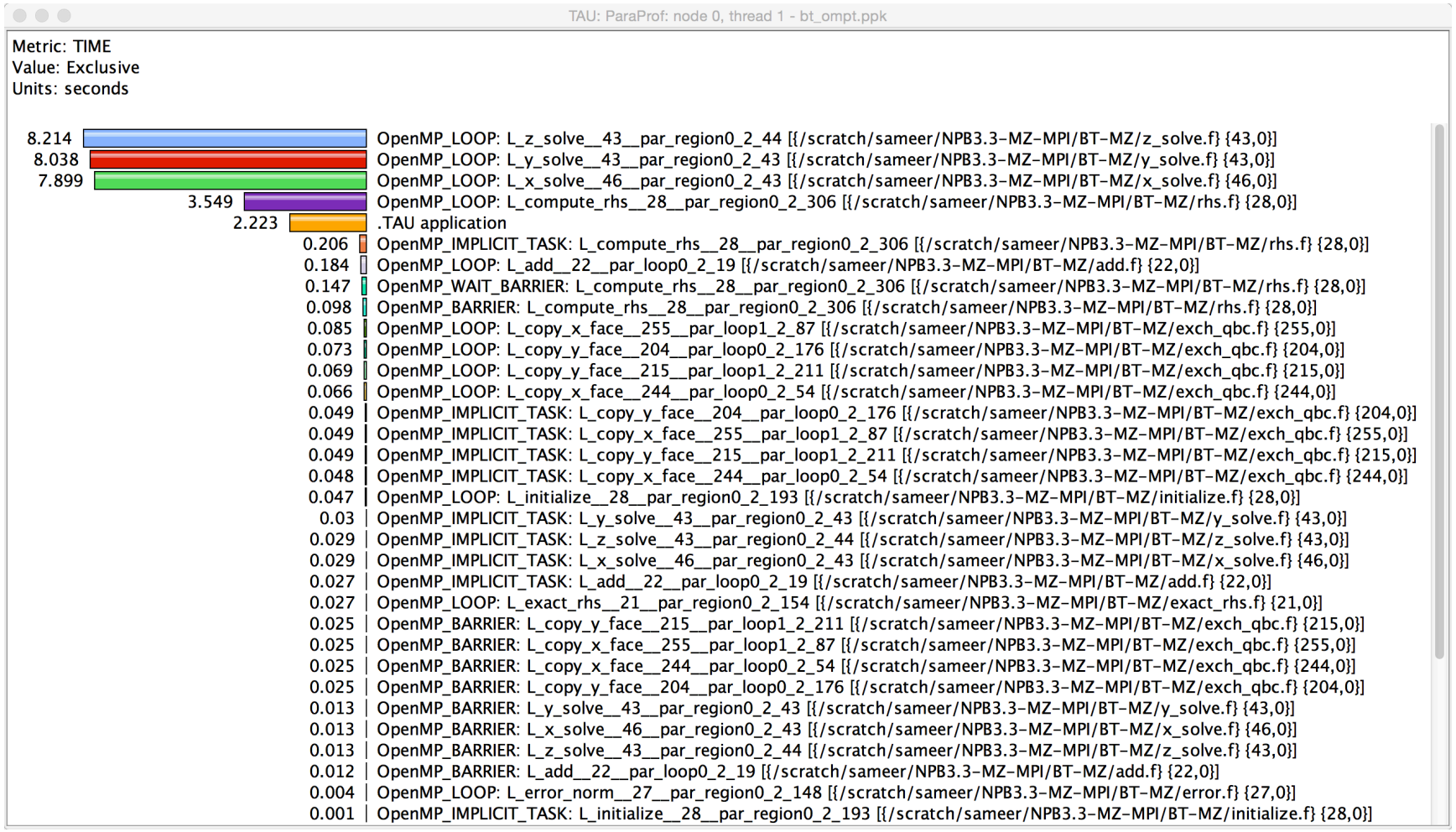


Left/right click here

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

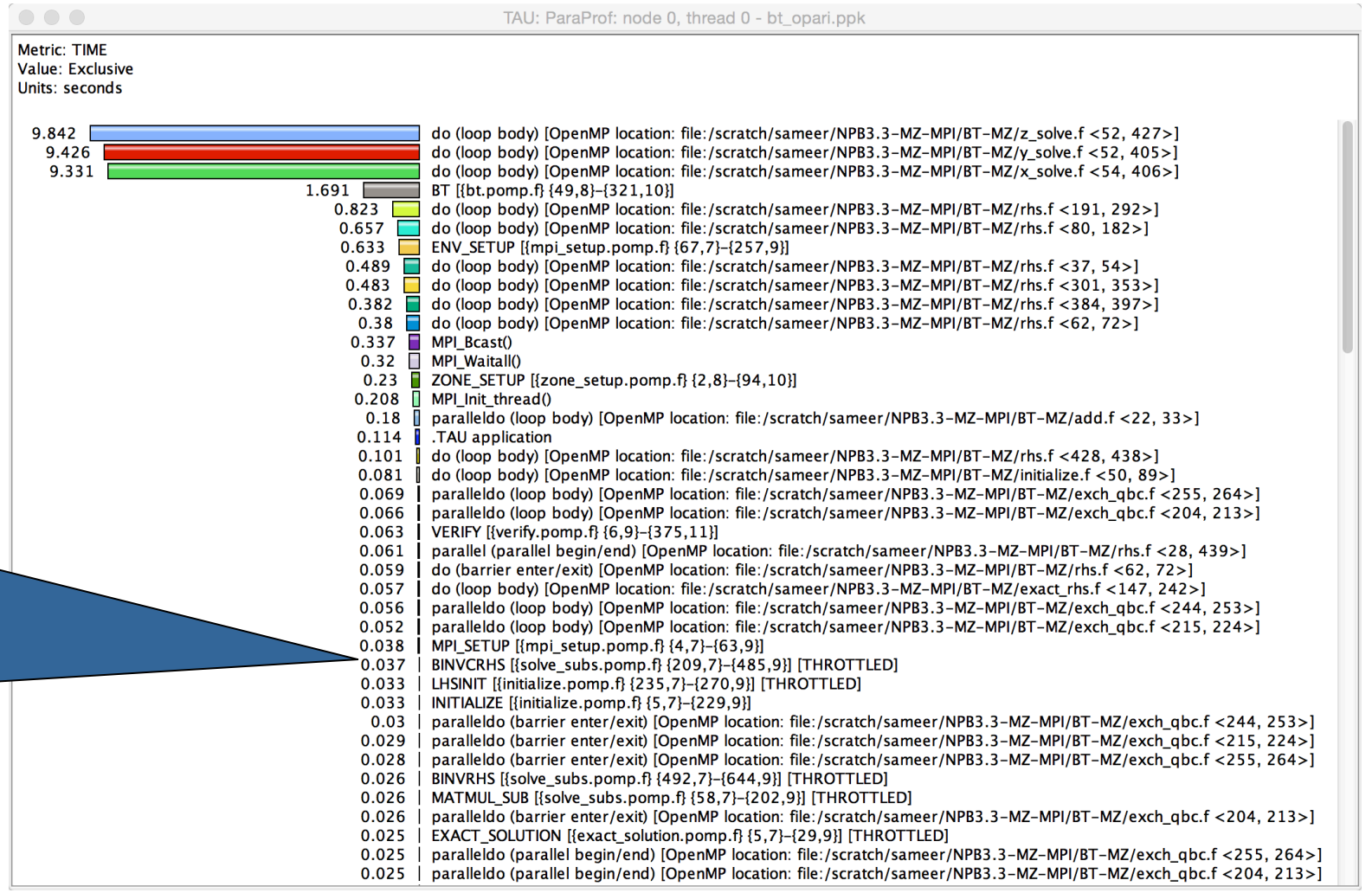
Paraprof node window (function barchart window)

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

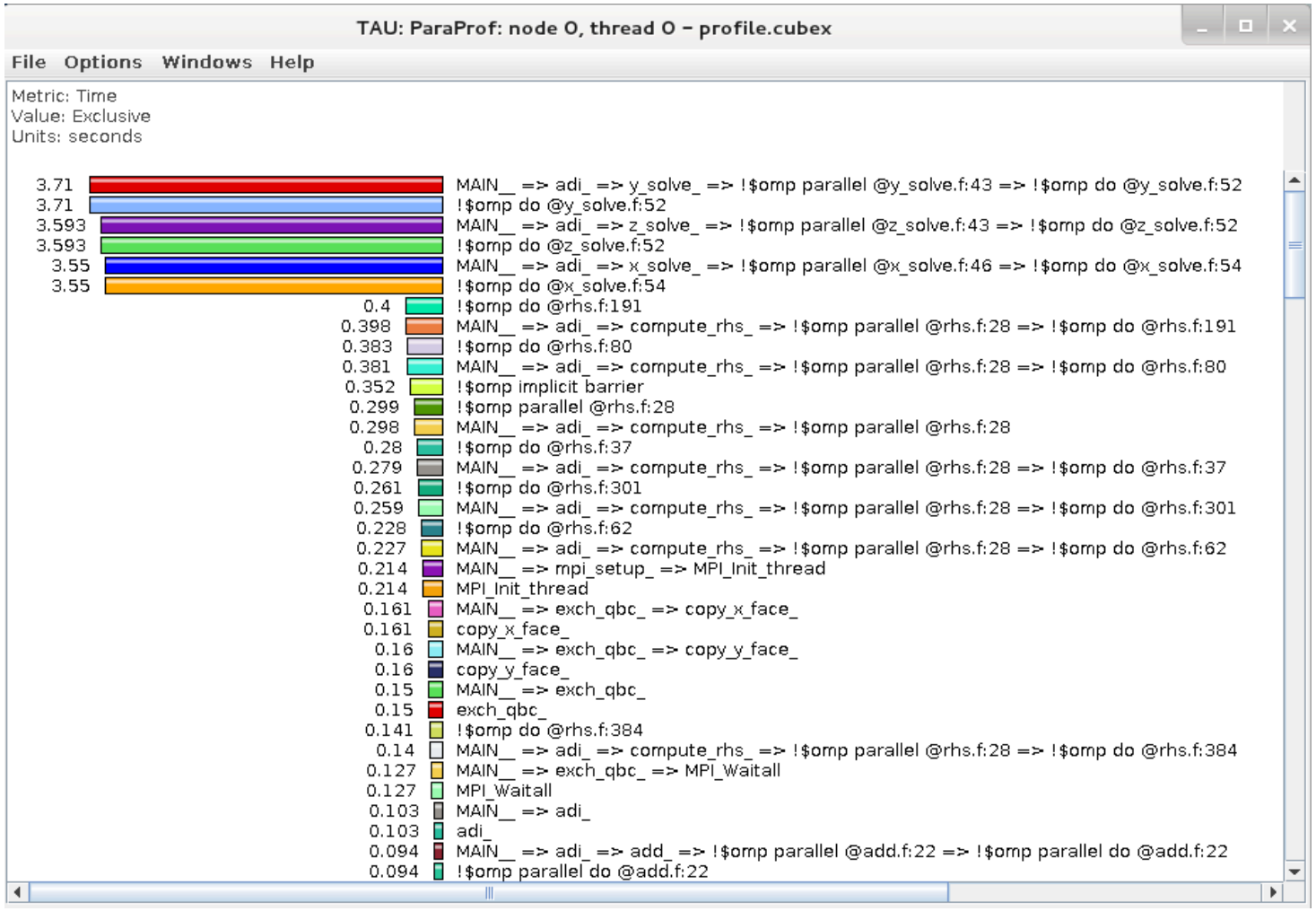


Instrumenting Source Code with PDT and Opari

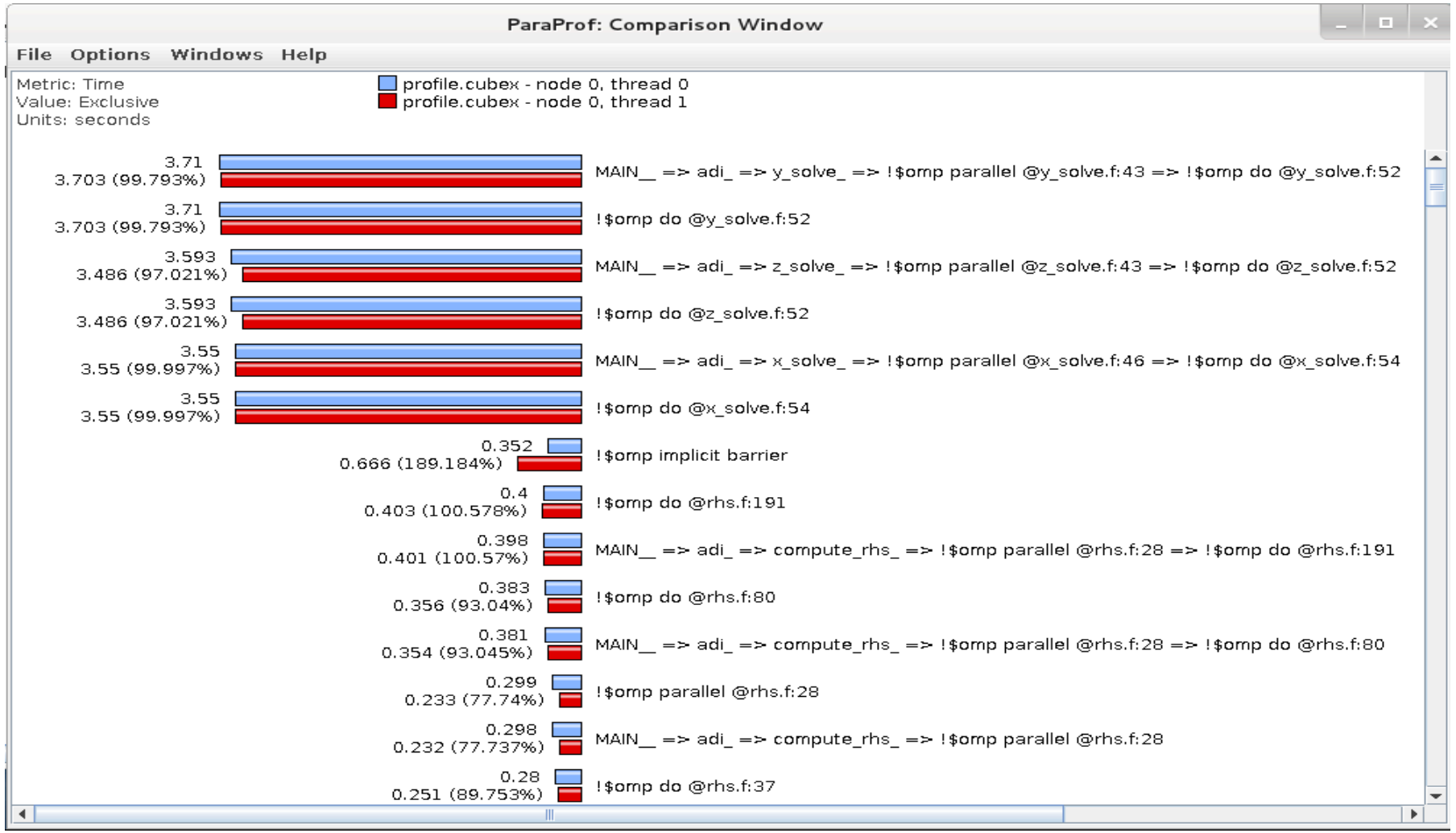
Frequently executing lightweight routines are automatically throttled at runtime. Reduces runtime dilation.



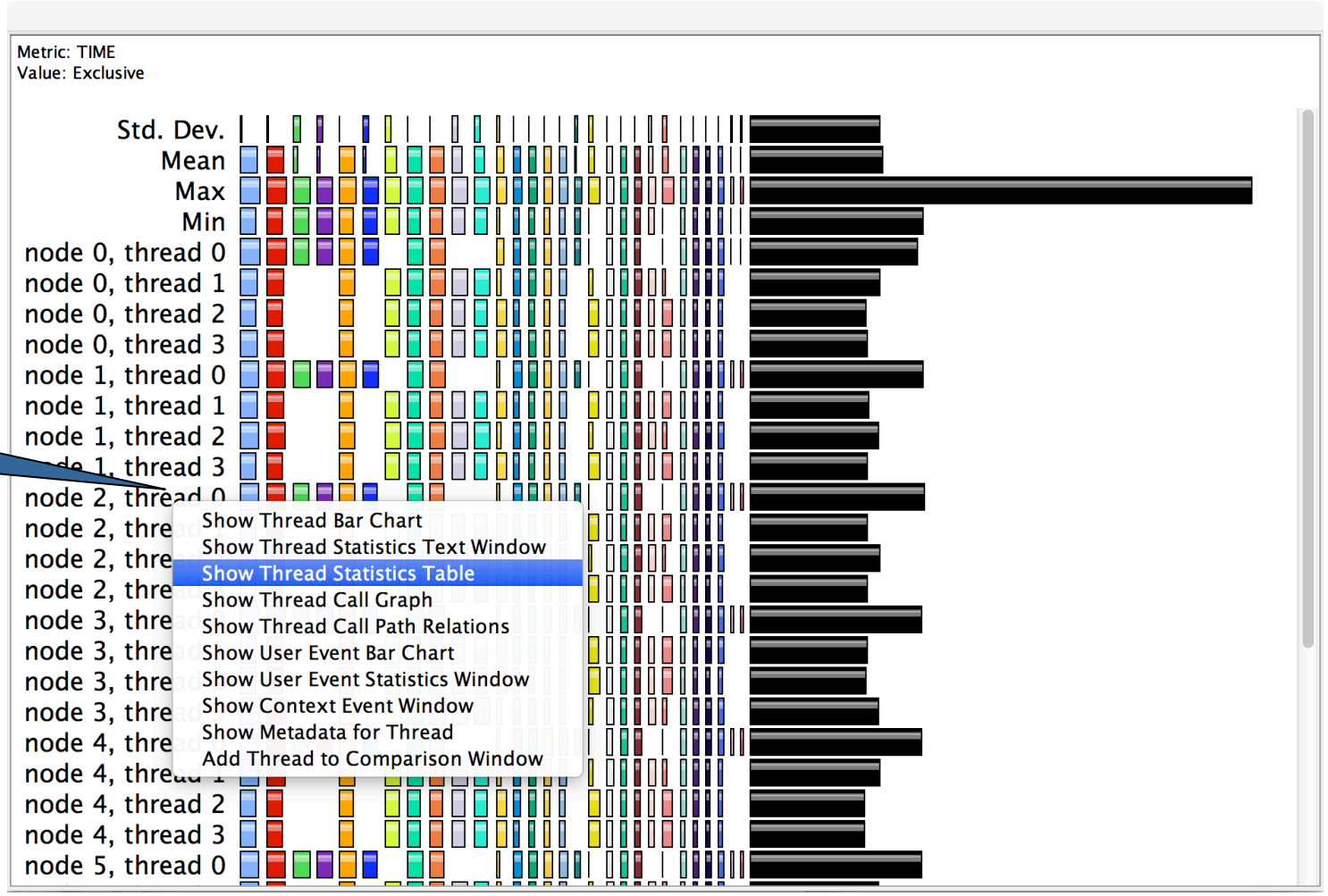
ParaProf: Node view in a callpath profile



ParaProf: Add thread to comparison window

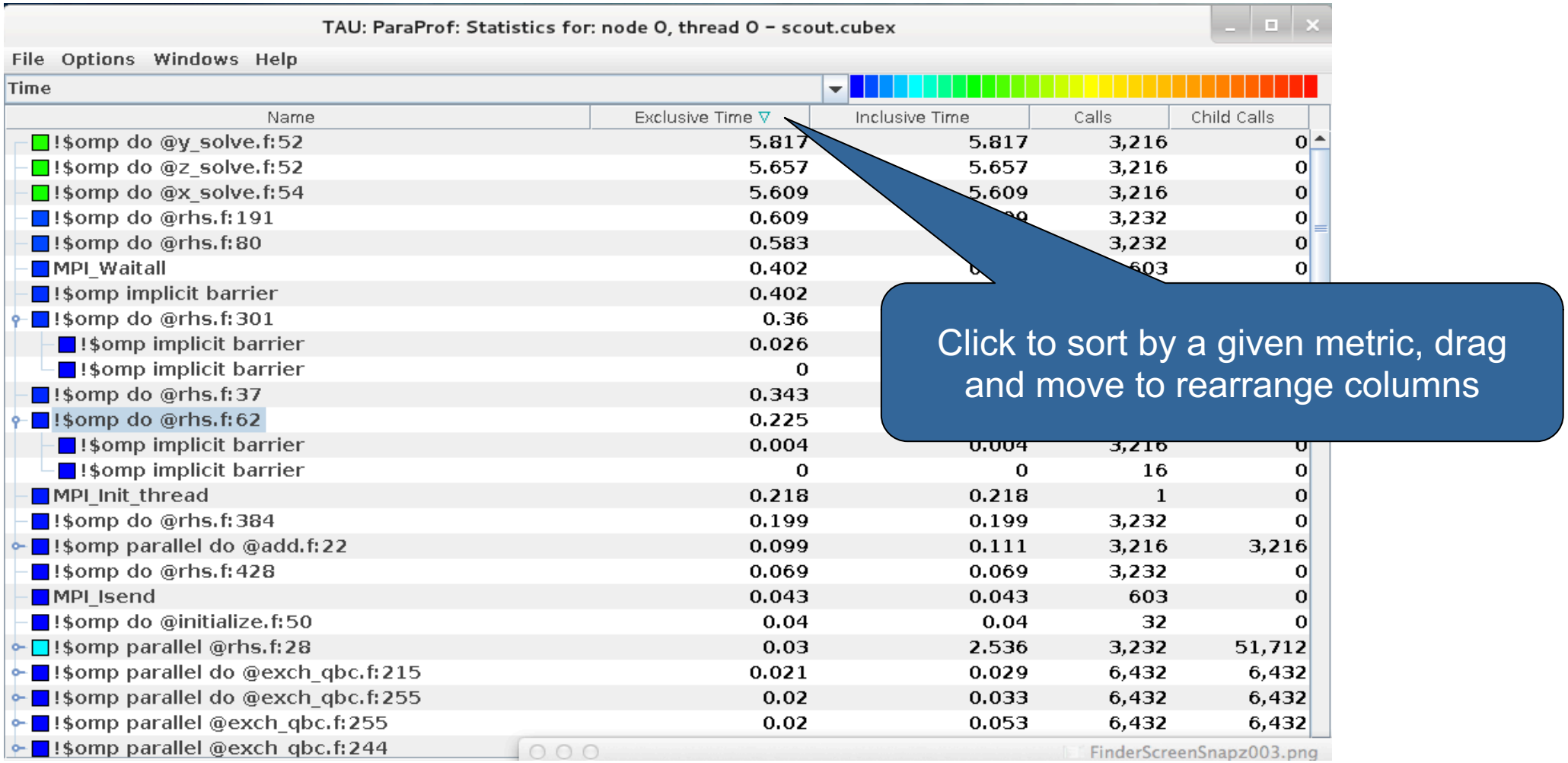


Paraprof Thread Statistics Table with TAU_SAMPLING=1



Right click here

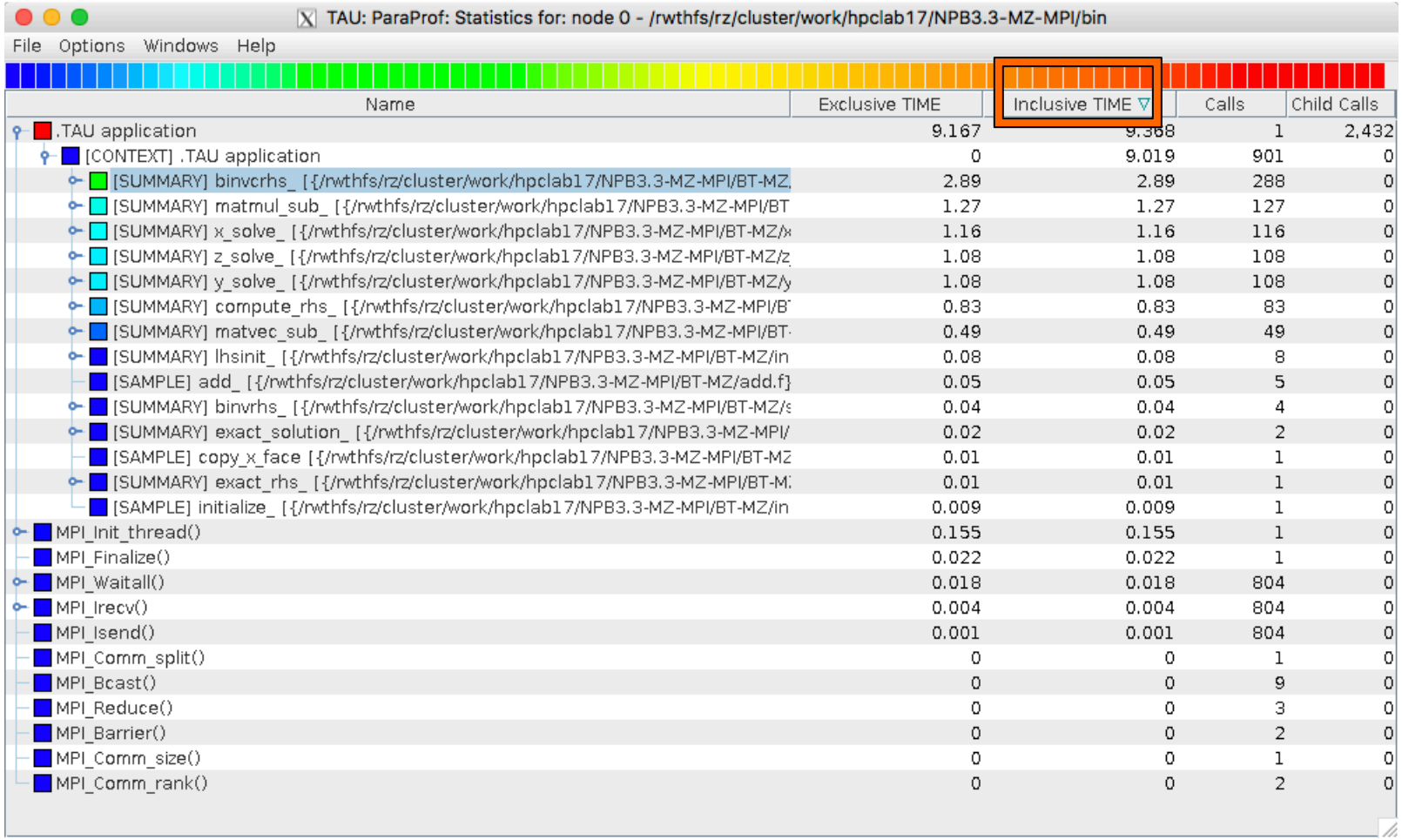
ParaProf: Thread Statistics Table



ParaProf

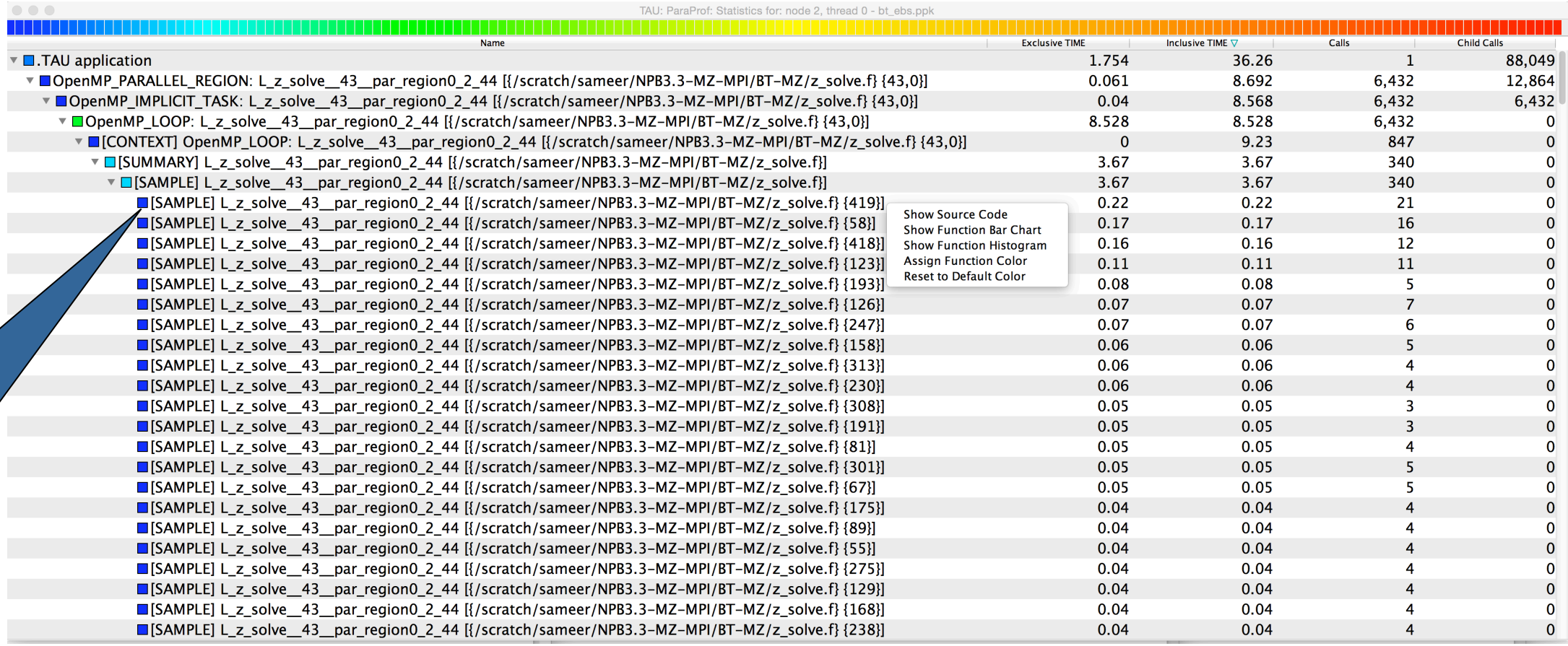
- Click on Columns:
- to sort by incl time

- Open binvcrhs
- Click on Sample



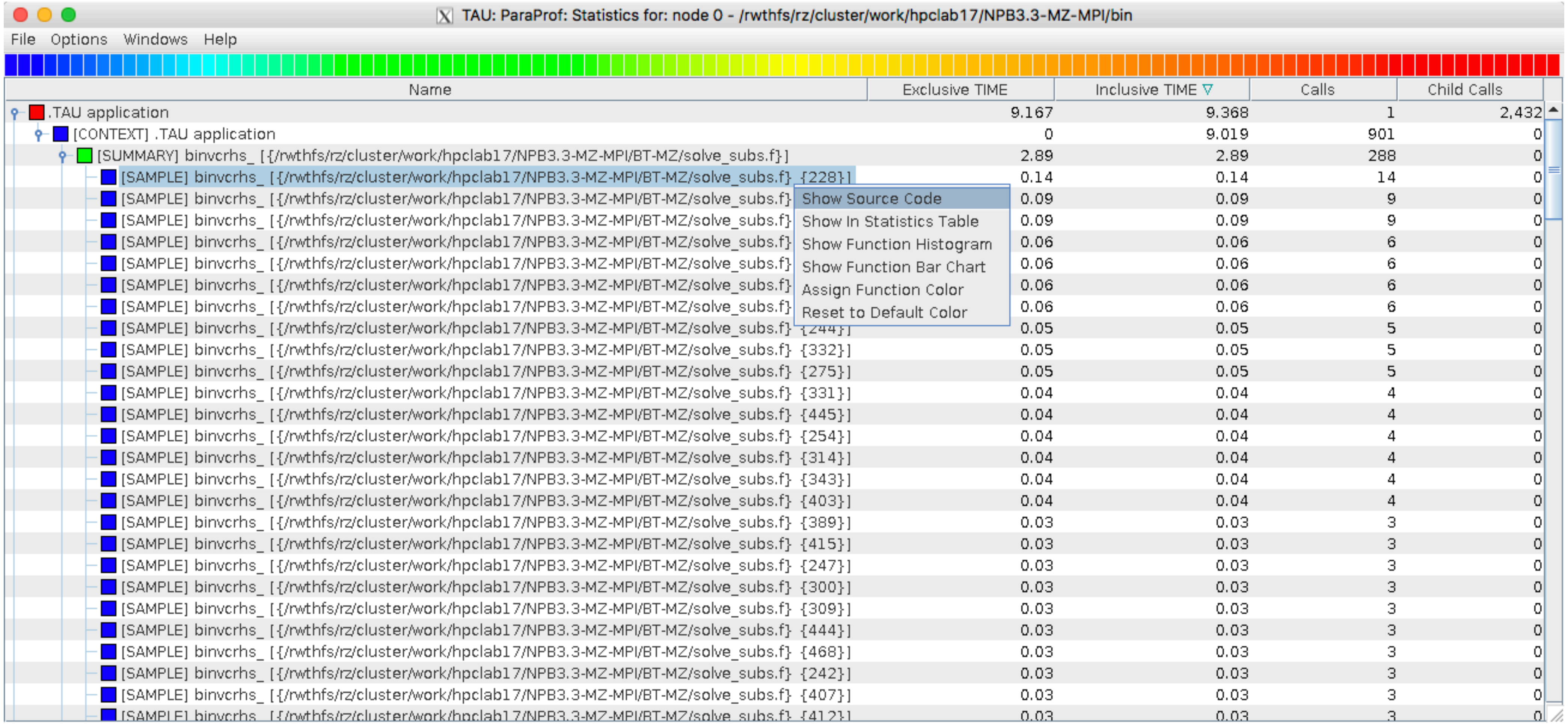
Name	Exclusive TIME	Inclusive TIME ▽	Calls	Child Calls
.TAU application	9.167	9.388	1	2,432
[CONTEXT] .TAU application	0	9.019	901	0
[SUMMARY] binvcrhs_ [{/rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ	2.89	2.89	288	0
[SUMMARY] matmul_sub_ [{/rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT	1.27	1.27	127	0
[SUMMARY] x_solve_ [{/rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/x	1.16	1.16	116	0
[SUMMARY] z_solve_ [{/rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/z	1.08	1.08	108	0
[SUMMARY] y_solve_ [{/rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/y	1.08	1.08	108	0
[SUMMARY] compute_rhs_ [{/rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/B	0.83	0.83	83	0
[SUMMARY] matvec_sub_ [{/rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT	0.49	0.49	49	0
[SUMMARY] lhsinit_ [{/rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/in	0.08	0.08	8	0
[SAMPLE] add_ [{/rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/add.f}	0.05	0.05	5	0
[SUMMARY] binvrhs_ [{/rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/s	0.04	0.04	4	0
[SUMMARY] exact_solution_ [{/rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/	0.02	0.02	2	0
[SAMPLE] copy_x_face [{/rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ	0.01	0.01	1	0
[SUMMARY] exact_rhs_ [{/rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-M	0.01	0.01	1	0
[SAMPLE] initialize_ [{/rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/in	0.009	0.009	1	0
MPI_Init_thread()	0.155	0.155	1	0
MPI_Finalize()	0.022	0.022	1	0
MPI_Waitall()	0.018	0.018	804	0
MPI_Irecv()	0.004	0.004	804	0
MPI_Isend()	0.001	0.001	804	0
MPI_Comm_split()	0	0	1	0
MPI_Bcast()	0	0	9	0
MPI_Reduce()	0	0	3	0
MPI_Barrier()	0	0	2	0
MPI_Comm_size()	0	0	1	0
MPI_Comm_rank()	0	0	2	0

Paraprof Thread Statistics Table



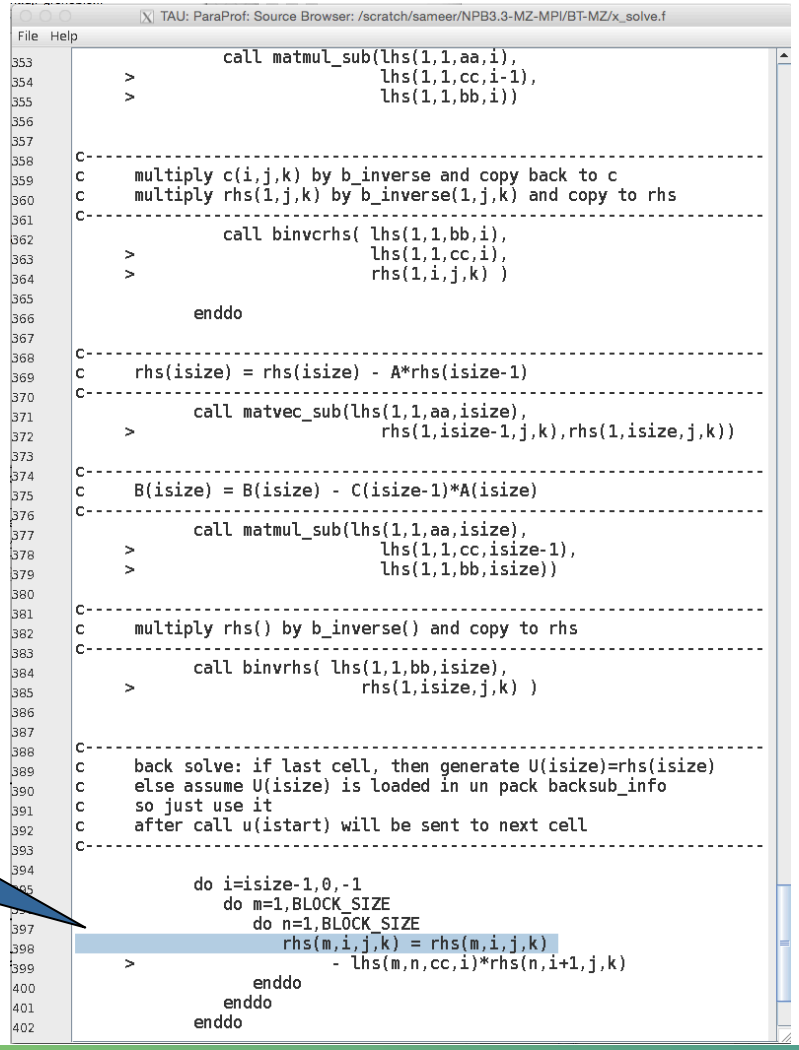
Right click here and choose “Show Source Code” for a sample

ParaProf



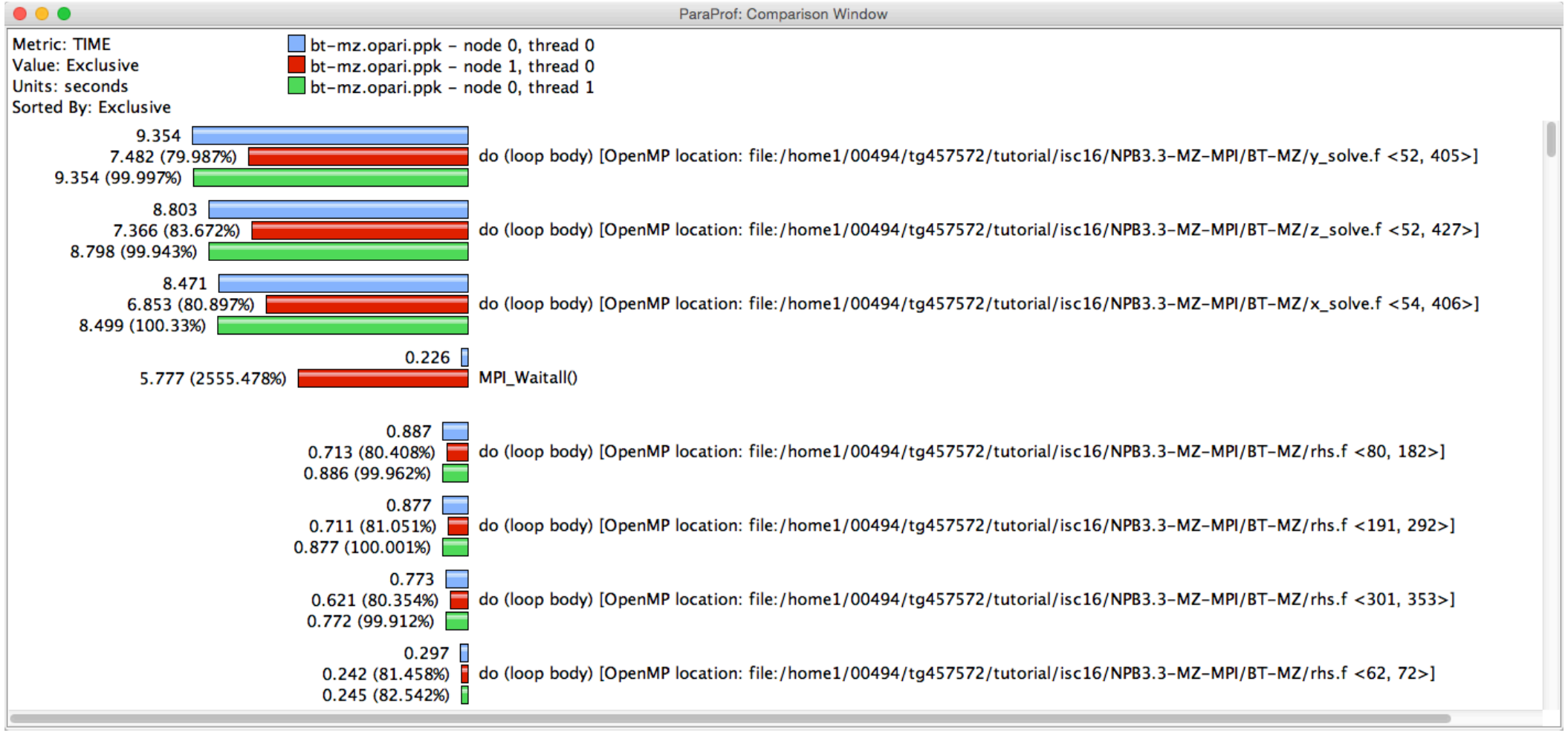
Name	Exclusive TIME	Inclusive TIME	Calls	Child Calls
.TAU application	9.167	9.368	1	2,432
[CONTEXT] .TAU application	0	9.019	901	0
[SUMMARY] binvcrhs_ [{/rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f}]	2.89	2.89	288	0
[SAMPLE] binvcrhs_ [{/rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f} {228}]	0.14	0.14	14	0
[SAMPLE] binvcrhs_ [{/rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f}	0.09	0.09	9	0
[SAMPLE] binvcrhs_ [{/rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f}	0.09	0.09	9	0
[SAMPLE] binvcrhs_ [{/rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f}	0.06	0.06	6	0
[SAMPLE] binvcrhs_ [{/rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f}	0.06	0.06	6	0
[SAMPLE] binvcrhs_ [{/rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f}	0.06	0.06	6	0
[SAMPLE] binvcrhs_ [{/rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f}	0.06	0.06	6	0
[SAMPLE] binvcrhs_ [{/rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f} {244}]	0.05	0.05	5	0
[SAMPLE] binvcrhs_ [{/rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f} {332}]	0.05	0.05	5	0
[SAMPLE] binvcrhs_ [{/rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f} {275}]	0.05	0.05	5	0
[SAMPLE] binvcrhs_ [{/rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f} {331}]	0.04	0.04	4	0
[SAMPLE] binvcrhs_ [{/rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f} {445}]	0.04	0.04	4	0
[SAMPLE] binvcrhs_ [{/rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f} {254}]	0.04	0.04	4	0
[SAMPLE] binvcrhs_ [{/rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f} {314}]	0.04	0.04	4	0
[SAMPLE] binvcrhs_ [{/rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f} {343}]	0.04	0.04	4	0
[SAMPLE] binvcrhs_ [{/rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f} {403}]	0.04	0.04	4	0
[SAMPLE] binvcrhs_ [{/rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f} {389}]	0.03	0.03	3	0
[SAMPLE] binvcrhs_ [{/rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f} {415}]	0.03	0.03	3	0
[SAMPLE] binvcrhs_ [{/rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f} {247}]	0.03	0.03	3	0
[SAMPLE] binvcrhs_ [{/rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f} {300}]	0.03	0.03	3	0
[SAMPLE] binvcrhs_ [{/rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f} {309}]	0.03	0.03	3	0
[SAMPLE] binvcrhs_ [{/rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f} {444}]	0.03	0.03	3	0
[SAMPLE] binvcrhs_ [{/rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f} {468}]	0.03	0.03	3	0
[SAMPLE] binvcrhs_ [{/rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f} {242}]	0.03	0.03	3	0
[SAMPLE] binvcrhs_ [{/rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f} {407}]	0.03	0.03	3	0
[SAMPLE] binvcrhs_ [{/rwthfs/rz/cluster/work/hpclab17/NPB3.3-MZ-MPI/BT-MZ/solve_subs.f} {412}]	0.03	0.03	3	0

Statement Level Profiling with TAU

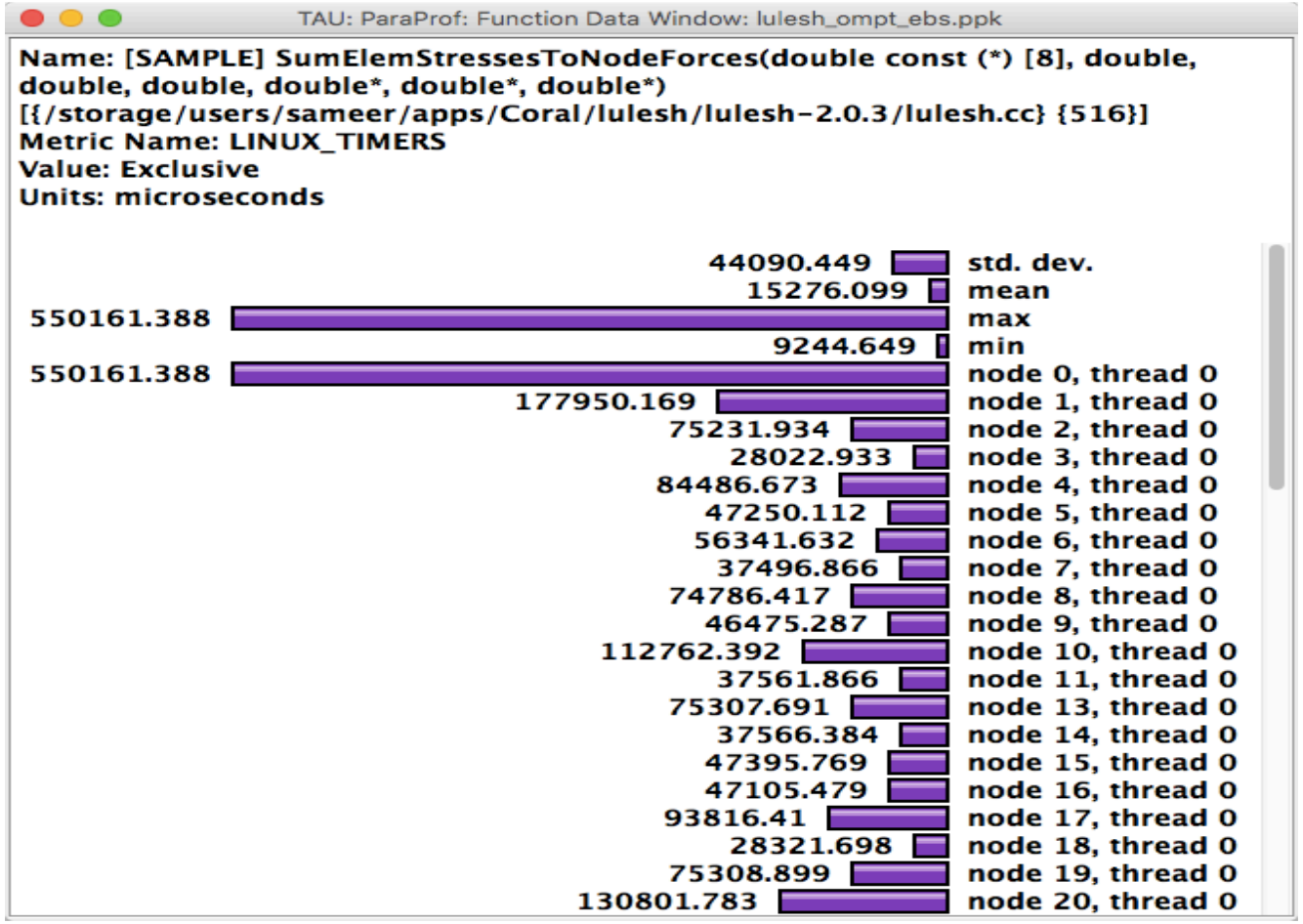


Source location where samples are taken. Compute intensive region.

ParaProf Comparison Window



TAU – Event Based Sampling (EBS)



% export TAU_SAMPLING=1

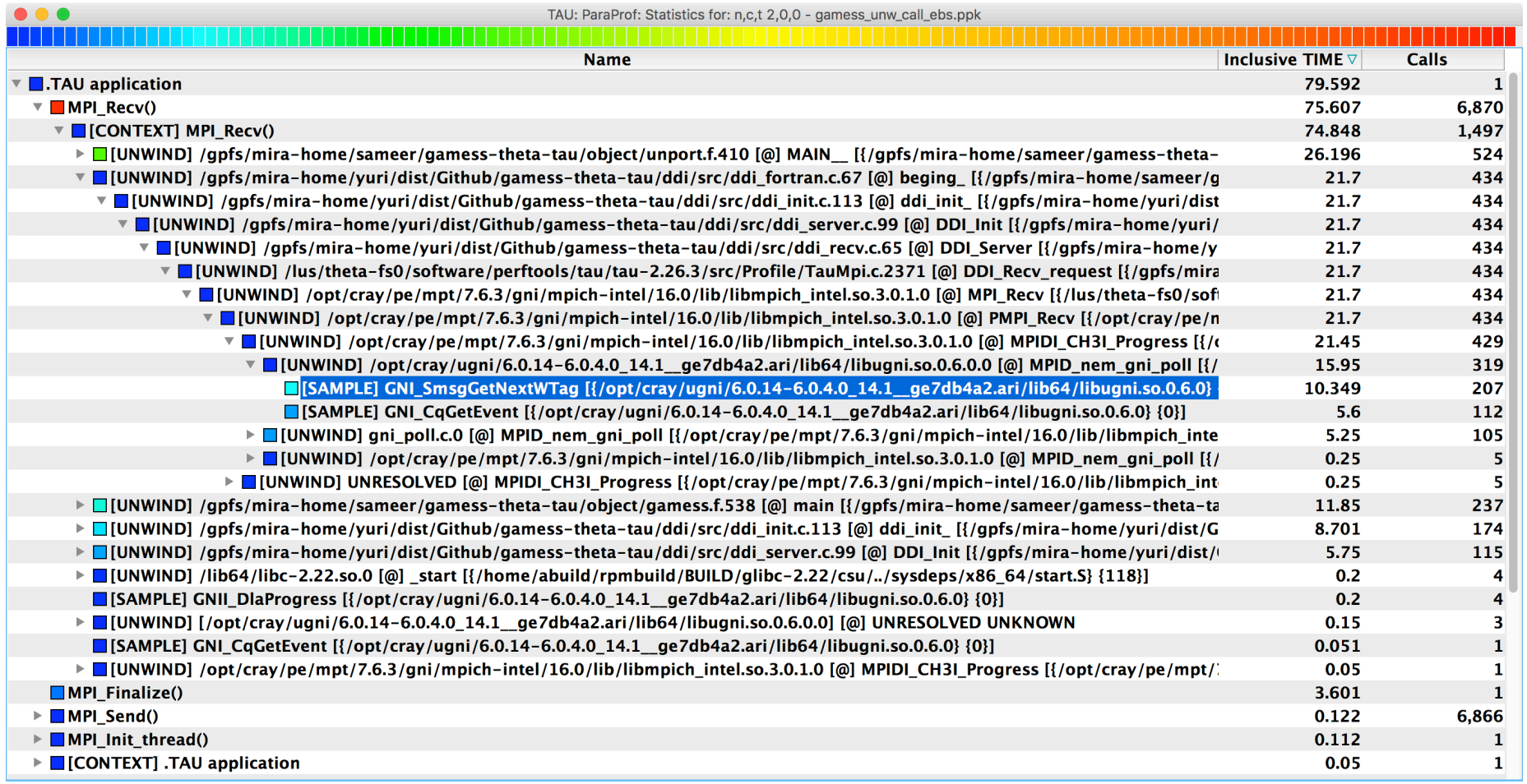
Examples: Callstack Sampling in TAU

TAU: ParaProf: Statistics for: n,c,t 2,0,0 - gamess_unw_call_ebs.ppk

Name	Inclusive TIME	Calls
.TAU application	79.592	1
MPI_Recv()	75.607	6,870
[CONTEXT] MPI_Recv()	74.848	1,497
[UNWIND] /gpfs/mira-home/sameer/gamess-theta-tau/object/unport.f.410 [@] MAIN__ [{/gpfs/mira-home/sameer/gamess-theta-t	26.196	524
[UNWIND] /gpfs/mira-home/yuri/dist/Github/gamess-theta-tau/ddi/src/ddi_fortran.c.67 [@] beging_ [{/gpfs/mira-home/sameer/g	21.7	434
[UNWIND] /gpfs/mira-home/sameer/gamess-theta-tau/object/gamess.f.538 [@] main [{/gpfs/mira-home/sameer/gamess-theta-ta	11.85	237
[UNWIND] /gpfs/mira-home/yuri/dist/Github/gamess-theta-tau/ddi/src/ddi_init.c.113 [@] ddi_init_ [{/gpfs/mira-home/yuri/dist/Gi	8.701	174
[UNWIND] /gpfs/mira-home/yuri/dist/Github/gamess-theta-tau/ddi/src/ddi_server.c.99 [@] DDI_Init [{/gpfs/mira-home/yuri/dist/C	5.75	115
[UNWIND] /lib64/libc-2.22.so.0 [@] _start [{/home/abuild/rpmbuild/BUILD/glibc-2.22/csu/../sysdeps/x86_64/start.S} {118}]	0.2	4
[SAMPLE] GNII_DlaProgress [{/opt/cray/ugni/6.0.14-6.0.4.0_14.1__ge7db4a2.ari/lib64/libugni.so.0.6.0} {0}]	0.2	4
[UNWIND] [/opt/cray/ugni/6.0.14-6.0.4.0_14.1__ge7db4a2.ari/lib64/libugni.so.0.6.0.0] [@] UNRESOLVED UNKNOWN	0.15	3
[SAMPLE] GNI_CqGetEvent [{/opt/cray/ugni/6.0.14-6.0.4.0_14.1__ge7db4a2.ari/lib64/libugni.so.0.6.0} {0}]	0.051	1
[UNWIND] /opt/cray/pe/mpt/7.6.3/gni/mpich-intel/16.0/lib/libmpich_intel.so.3.0.1.0 [@] MPIDI_CH3I_Progress [{/opt/cray/pe/mpt/7	0.05	1
MPI_Finalize()	3.601	1
MPI_Send()	0.122	6,866
MPI_Init_thread()	0.112	1
[CONTEXT] .TAU application	0.05	1
MPI_Bcast()	0.014	6
MPI_Allgather()	0.004	3
MPI_Barrier()	0.003	7
MPI_Comm_create()	0.002	4
MPI_Gather()	0.002	1
MPI_Comm_split()	0.002	1
MPI_Group_intersection()	0.001	1
MPI_Comm_group()	0.001	1
MPI_Group_incl()	0	3
MPI_Comm_rank()	0	6
MPI_Comm_size()	0	2

% export TAU_SAMPLING=1; export TAU_EBS_UNWIND=1

UNWINDING CALLSTACKS



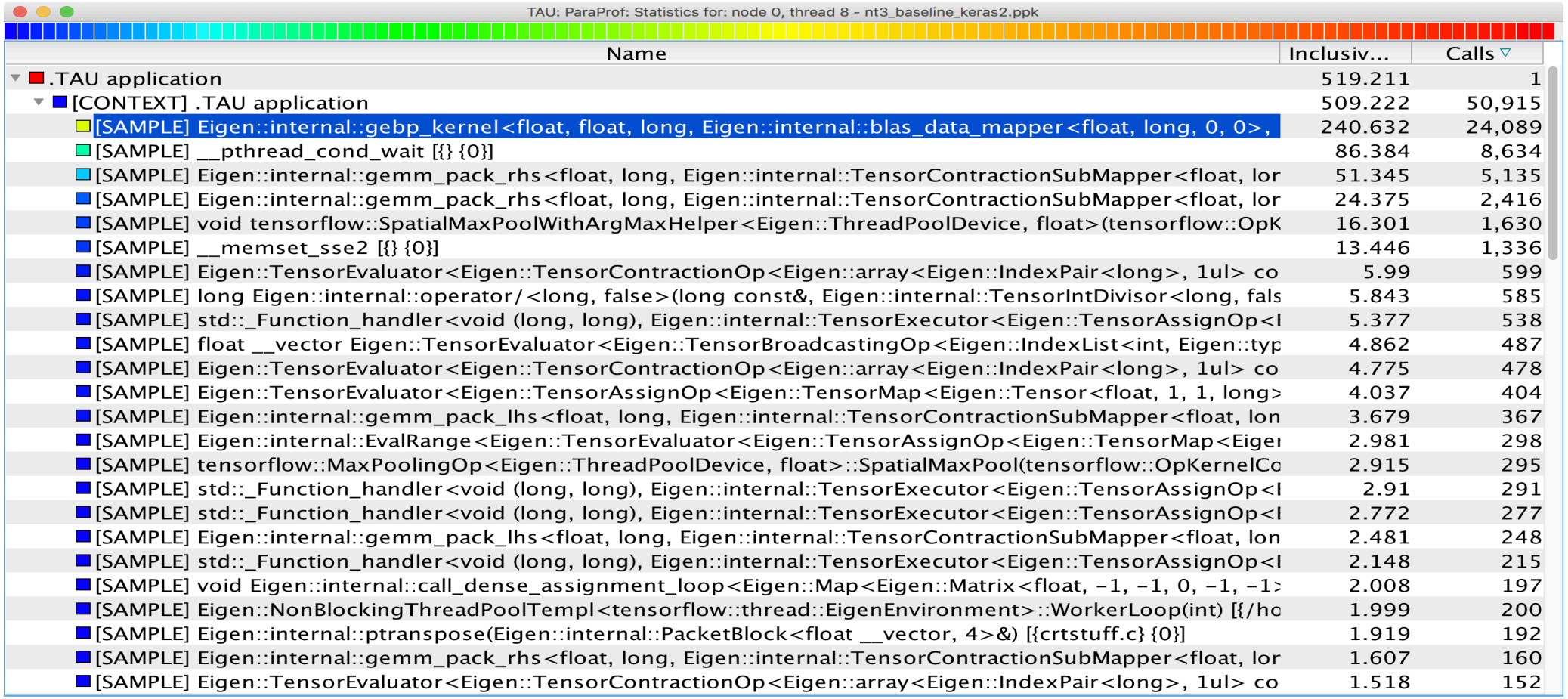
Name	Inclusive TIME	Calls
.TAU application	79.592	1
MPI_Recv()	75.607	6,870
[CONTEXT] MPI_Recv()	74.848	1,497
[UNWIND] /gpfs/mira-home/sameer/gamess-theta-tau/object/unport.f.410 [@] MAIN__ [{/gpfs/mira-home/sameer/gamess-theta-	26.196	524
[UNWIND] /gpfs/mira-home/yuri/dist/Github/gamess-theta-tau/ddi/src/ddi_fortran.c.67 [@] beging_ [{/gpfs/mira-home/sameer/g	21.7	434
[UNWIND] /gpfs/mira-home/yuri/dist/Github/gamess-theta-tau/ddi/src/ddi_init.c.113 [@] ddi_init_ [{/gpfs/mira-home/yuri/dist	21.7	434
[UNWIND] /gpfs/mira-home/yuri/dist/Github/gamess-theta-tau/ddi/src/ddi_server.c.99 [@] DDI_Init [{/gpfs/mira-home/yuri/	21.7	434
[UNWIND] /gpfs/mira-home/yuri/dist/Github/gamess-theta-tau/ddi/src/ddi_recv.c.65 [@] DDI_Server [{/gpfs/mira-home/y	21.7	434
[UNWIND] /lus/theta-fs0/software/perftools/tau/tau-2.26.3/src/Profile/TauMpi.c.2371 [@] DDI_Recv_request [{/gpfs/mira	21.7	434
[UNWIND] /opt/cray/pe/mpt/7.6.3/gni/mpich-intel/16.0/lib/libmpich_intel.so.3.0.1.0 [@] MPI_Recv [{/lus/theta-fs0/soft	21.7	434
[UNWIND] /opt/cray/pe/mpt/7.6.3/gni/mpich-intel/16.0/lib/libmpich_intel.so.3.0.1.0 [@] PMPI_Recv [{/opt/cray/pe/n	21.7	434
[UNWIND] /opt/cray/pe/mpt/7.6.3/gni/mpich-intel/16.0/lib/libmpich_intel.so.3.0.1.0 [@] MPIDI_CH3I_Progress [{/	21.45	429
[UNWIND] /opt/cray/ugni/6.0.14-6.0.4.0_14.1__ge7db4a2.ari/lib64/libugni.so.0.6.0.0 [@] MPID_nem_gni_poll [{/	15.95	319
[SAMPLE] GNI_SmsgGetNextWTag [{/opt/cray/ugni/6.0.14-6.0.4.0_14.1__ge7db4a2.ari/lib64/libugni.so.0.6.0}	10.349	207
[SAMPLE] GNI_CqGetEvent [{/opt/cray/ugni/6.0.14-6.0.4.0_14.1__ge7db4a2.ari/lib64/libugni.so.0.6.0} {0}]	5.6	112
[UNWIND] gni_poll.c.0 [@] MPID_nem_gni_poll [{/opt/cray/pe/mpt/7.6.3/gni/mpich-intel/16.0/lib/libmpich_inte	5.25	105
[UNWIND] /opt/cray/pe/mpt/7.6.3/gni/mpich-intel/16.0/lib/libmpich_intel.so.3.0.1.0 [@] MPID_nem_gni_poll [{/	0.25	5
[UNWIND] UNRESOLVED [@] MPIDI_CH3I_Progress [{/opt/cray/pe/mpt/7.6.3/gni/mpich-intel/16.0/lib/libmpich_int	0.25	5
[UNWIND] /gpfs/mira-home/sameer/gamess-theta-tau/object/gamess.f.538 [@] main [{/gpfs/mira-home/sameer/gamess-theta-ta	11.85	237
[UNWIND] /gpfs/mira-home/yuri/dist/Github/gamess-theta-tau/ddi/src/ddi_init.c.113 [@] ddi_init_ [{/gpfs/mira-home/yuri/dist/G	8.701	174
[UNWIND] /gpfs/mira-home/yuri/dist/Github/gamess-theta-tau/ddi/src/ddi_server.c.99 [@] DDI_Init [{/gpfs/mira-home/yuri/dist/	5.75	115
[UNWIND] /lib64/libc-2.22.so.0 [@] _start [{/home/abuild/rpmbuild/BUILD/glibc-2.22/csu/../sysdeps/x86_64/start.S} {118}]	0.2	4
[SAMPLE] GNII_DlaProgress [{/opt/cray/ugni/6.0.14-6.0.4.0_14.1__ge7db4a2.ari/lib64/libugni.so.0.6.0} {0}]	0.2	4
[UNWIND] [/opt/cray/ugni/6.0.14-6.0.4.0_14.1__ge7db4a2.ari/lib64/libugni.so.0.6.0.0] [@] UNRESOLVED UNKNOWN	0.15	3
[SAMPLE] GNI_CqGetEvent [{/opt/cray/ugni/6.0.14-6.0.4.0_14.1__ge7db4a2.ari/lib64/libugni.so.0.6.0} {0}]	0.051	1
[UNWIND] /opt/cray/pe/mpt/7.6.3/gni/mpich-intel/16.0/lib/libmpich_intel.so.3.0.1.0 [@] MPIDI_CH3I_Progress [{/opt/cray/pe/mpt/	0.05	1
MPI_Finalize()	3.601	1
MPI_Send()	0.122	6,866
MPI_Init_thread()	0.112	1
[CONTEXT] .TAU application	0.05	1

% export TAU_SAMPLING=1; export TAU_EBS_UNWIND=1

UNWINDING CALLSTACKS

Name	Inclusive TIME ▽	Calls
.TAU application	79.592	1
MPI_Recv()	75.607	6,870
[CONTEXT] MPI_Recv()	74.848	1,497
[UNWIND] /gpfs/mira-home/sameer/gamess-theta-tau/object/unport.f.410 [@] MAIN__ [{/gpfs/mira-home/sameer/gamess-theta-	26.196	524
[UNWIND] /gpfs/mira-home/yuri/dist/Github/gamess-theta-tau/ddi/src/ddi_fortran.c.67 [@] beging_ [{/gpfs/mira-home/sameer/g	21.7	434
[UNWIND] /gpfs/mira-home/sameer/gamess-theta-tau/object/gamess.f.538 [@] main [{/gpfs/mira-home/sameer/gamess-theta-ta	11.85	237
[UNWIND] /gpfs/mira-home/sameer/gamess-theta-tau/object/unport.f.410 [@] MAIN__ [{/gpfs/mira-home/sameer/gamess-thet	11.85	237
[UNWIND] /gpfs/mira-home/yuri/dist/Github/gamess-theta-tau/ddi/src/ddi_fortran.c.67 [@] beging_ [{/gpfs/mira-home/sam	11.85	237
[UNWIND] /gpfs/mira-home/yuri/dist/Github/gamess-theta-tau/ddi/src/ddi_init.c.113 [@] ddi_init_ [{/gpfs/mira-home/yur	11.85	237
[UNWIND] /gpfs/mira-home/yuri/dist/Github/gamess-theta-tau/ddi/src/ddi_server.c.99 [@] DDI_Init [{/gpfs/mira-home/	11.85	237
[UNWIND] /gpfs/mira-home/yuri/dist/Github/gamess-theta-tau/ddi/src/ddi_recv.c.65 [@] DDI_Server [{/gpfs/mira-ho	11.85	237
[UNWIND] /lus/theta-fs0/software/perftools/tau/tau-2.26.3/src/Profile/TauMpi.c.2371 [@] DDI_Recv_request [{/gpfs	11.85	237
[UNWIND] /opt/cray/pe/mpt/7.6.3/gni/mpich-intel/16.0/lib/libmpich_intel.so.3.0.1.0 [@] MPI_Recv [{/lus/theta-fs(	11.85	237
[UNWIND] /opt/cray/pe/mpt/7.6.3/gni/mpich-intel/16.0/lib/libmpich_intel.so.3.0.1.0 [@] PMPI_Recv [{/opt/cray/	11.7	234
[SAMPLE] MPIDI_CH3I_Progress [{/opt/cray/pe/mpt/7.6.3/gni/mpich-intel/16.0/lib/libmpich_intel.so.3.0.1} {0	11.3	226
[SAMPLE] MPIDU_Sched_are_pending [{/opt/cray/pe/mpt/7.6.3/gni/mpich-intel/16.0/lib/libmpich_intel.so.3.0	0.2	4
[SAMPLE] MPID_nem_gni_poll [{/opt/cray/pe/mpt/7.6.3/gni/mpich-intel/16.0/lib/libmpich_intel.so.3.0.1} {0}]	0.15	3
[SAMPLE] MPID_nem_network_poll [{/opt/cray/pe/mpt/7.6.3/gni/mpich-intel/16.0/lib/libmpich_intel.so.3.0.1	0.05	1
[UNWIND] ch3_progress.c.0 [@] PMPI_Recv [{/opt/cray/pe/mpt/7.6.3/gni/mpich-intel/16.0/lib/libmpich_intel.so.	0.15	3
[UNWIND] /gpfs/mira-home/yuri/dist/Github/gamess-theta-tau/ddi/src/ddi_init.c.113 [@] ddi_init_ [{/gpfs/mira-home/yuri/dist/G	8.701	174
[UNWIND] /gpfs/mira-home/yuri/dist/Github/gamess-theta-tau/ddi/src/ddi_server.c.99 [@] DDI_Init [{/gpfs/mira-home/yuri/dist/(	5.75	115
[UNWIND] /lib64/libc-2.22.so.0 [@] _start [{/home/abuild/rpmbuild/BUILD/glibc-2.22/csu/../sysdeps/x86_64/start.S} {118}]	0.2	4
[SAMPLE] GNII_DlaProgress [{/opt/cray/ugni/6.0.14-6.0.4.0_14.1__ge7db4a2.ari/lib64/libugni.so.0.6.0} {0}]	0.2	4
[UNWIND] [/opt/cray/ugni/6.0.14-6.0.4.0_14.1__ge7db4a2.ari/lib64/libugni.so.0.6.0.0] [@] UNRESOLVED UNKNOWN	0.15	3
[SAMPLE] GNI_CqGetEvent [{/opt/cray/ugni/6.0.14-6.0.4.0_14.1__ge7db4a2.ari/lib64/libugni.so.0.6.0} {0}]	0.051	1
[UNWIND] /opt/cray/pe/mpt/7.6.3/gni/mpich-intel/16.0/lib/libmpich_intel.so.3.0.1.0 [@] MPIDI_CH3I_Progress [{/opt/cray/pe/mpt/	0.05	1
MPI_Finalize()	3.601	1
MPI_Send()	0.122	6,866
MPI_Init_thread()	0.112	1
[CONTEXT] .TAU application	0.05	1

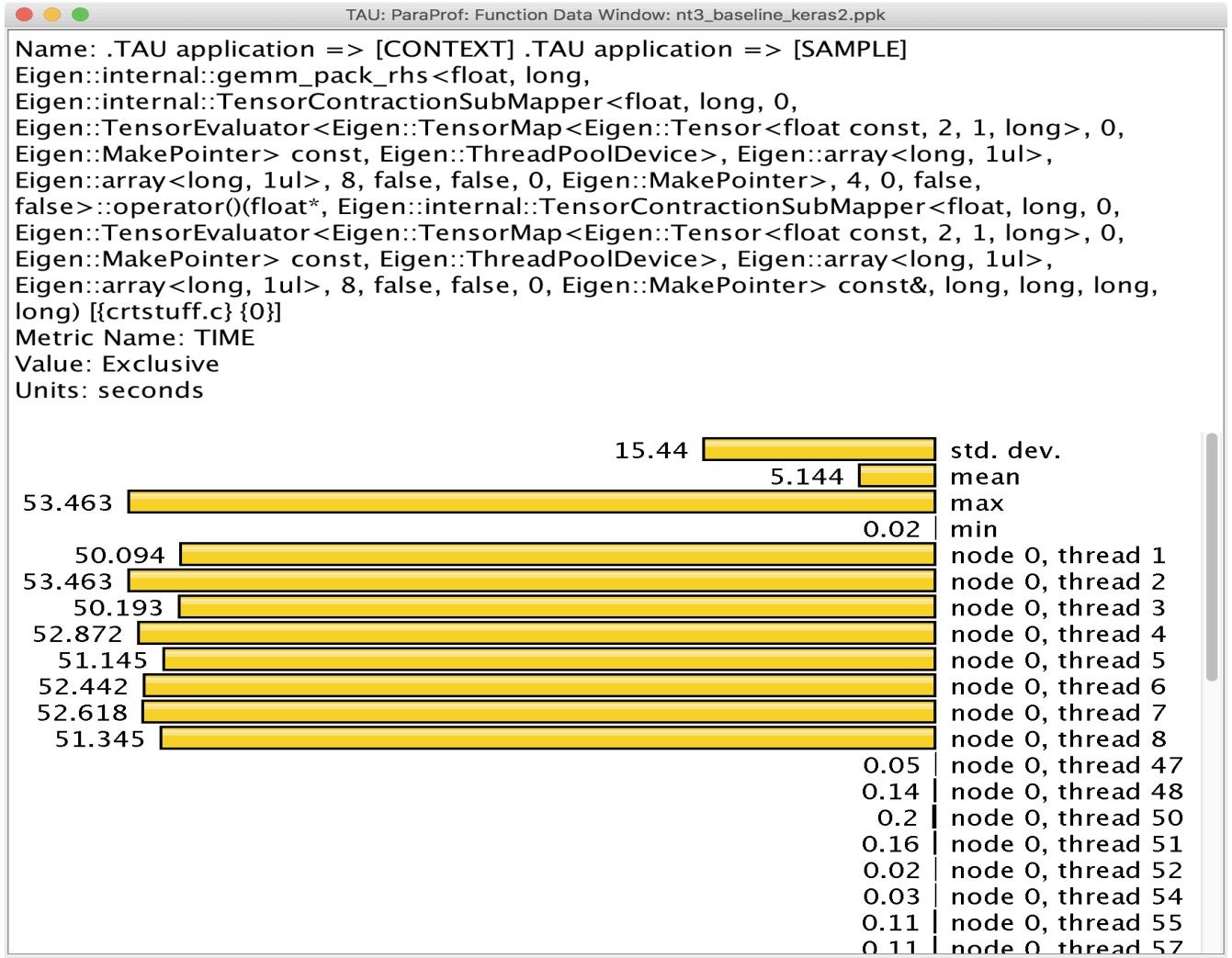
Deep Learning: Tensorflow



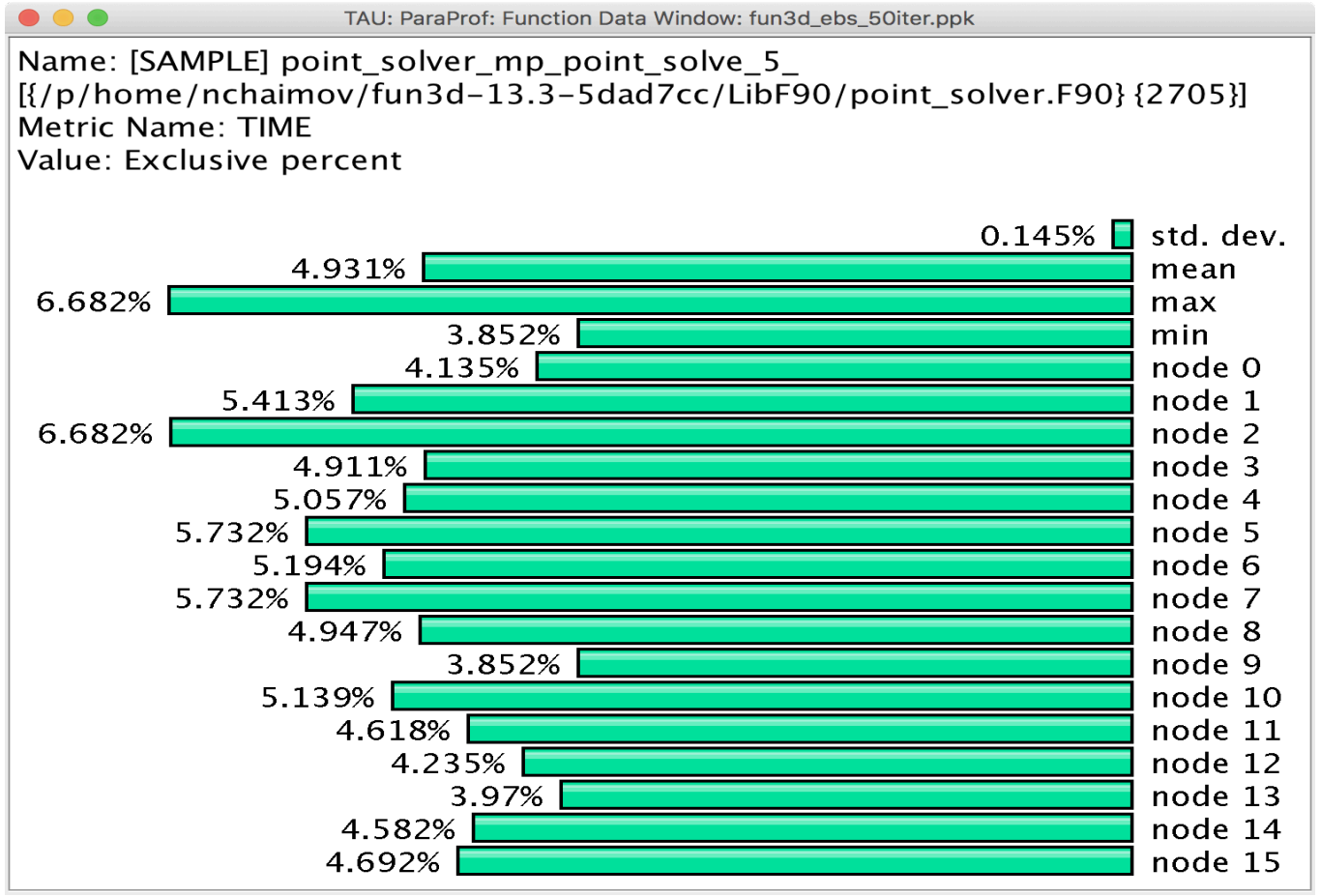
Name	Inclusiv...	Calls
.TAU application	519.211	1
[CONTEXT] .TAU application	509.222	50,915
[SAMPLE] Eigen::internal::gebp_kernel<float, float, long, Eigen::internal::blas_data_mapper<float, long, 0, 0>,	240.632	24,089
[SAMPLE] __pthread_cond_wait [{} {0}]	86.384	8,634
[SAMPLE] Eigen::internal::gemm_pack_rhs<float, long, Eigen::internal::TensorContractionSubMapper<float, lor	51.345	5,135
[SAMPLE] Eigen::internal::gemm_pack_rhs<float, long, Eigen::internal::TensorContractionSubMapper<float, lor	24.375	2,416
[SAMPLE] void tensorflow::SpatialMaxPoolWithArgMaxHelper<Eigen::ThreadPoolDevice, float>(tensorflow::OpK	16.301	1,630
[SAMPLE] __memset_sse2 [{} {0}]	13.446	1,336
[SAMPLE] Eigen::TensorEvaluator<Eigen::TensorContractionOp<Eigen::array<Eigen::IndexPair<long>, 1ul> co	5.99	599
[SAMPLE] long Eigen::internal::operator/<long, false>(long const&, Eigen::internal::TensorIntDivisor<long, fals	5.843	585
[SAMPLE] std::_Function_handler<void (long, long), Eigen::internal::TensorExecutor<Eigen::TensorAssignOp<I	5.377	538
[SAMPLE] float __vector Eigen::TensorEvaluator<Eigen::TensorBroadcastingOp<Eigen::IndexList<int, Eigen::typ	4.862	487
[SAMPLE] Eigen::TensorEvaluator<Eigen::TensorContractionOp<Eigen::array<Eigen::IndexPair<long>, 1ul> co	4.775	478
[SAMPLE] Eigen::TensorEvaluator<Eigen::TensorAssignOp<Eigen::TensorMap<Eigen::Tensor<float, 1, 1, long>	4.037	404
[SAMPLE] Eigen::internal::gemm_pack_lhs<float, long, Eigen::internal::TensorContractionSubMapper<float, lon	3.679	367
[SAMPLE] Eigen::internal::EvalRange<Eigen::TensorEvaluator<Eigen::TensorAssignOp<Eigen::TensorMap<Eiger	2.981	298
[SAMPLE] tensorflow::MaxPoolingOp<Eigen::ThreadPoolDevice, float>::SpatialMaxPool(tensorflow::OpKernelCo	2.915	295
[SAMPLE] std::_Function_handler<void (long, long), Eigen::internal::TensorExecutor<Eigen::TensorAssignOp<I	2.91	291
[SAMPLE] std::_Function_handler<void (long, long), Eigen::internal::TensorExecutor<Eigen::TensorAssignOp<I	2.772	277
[SAMPLE] Eigen::internal::gemm_pack_lhs<float, long, Eigen::internal::TensorContractionSubMapper<float, lon	2.481	248
[SAMPLE] std::_Function_handler<void (long, long), Eigen::internal::TensorExecutor<Eigen::TensorAssignOp<I	2.148	215
[SAMPLE] void Eigen::internal::call_dense_assignment_loop<Eigen::Map<Eigen::Matrix<float, -1, -1, 0, -1, -1>	2.008	197
[SAMPLE] Eigen::NonBlockingThreadPoolTempl<tensorflow::thread::EigenEnvironment>::WorkerLoop(int) [{/ho	1.999	200
[SAMPLE] Eigen::internal::ptranspose(Eigen::internal::PacketBlock<float __vector, 4>&) [{crtstuff.c} {0}]	1.919	192
[SAMPLE] Eigen::internal::gemm_pack_rhs<float, long, Eigen::internal::TensorContractionSubMapper<float, lor	1.607	160
[SAMPLE] Eigen::TensorEvaluator<Eigen::TensorContractionOp<Eigen::array<Eigen::IndexPair<long>, 1ul> co	1.518	152

```
% tau_python -ebs nt3_baseline_keras2.py (CANDLE)
```

Sampling Tensorflow



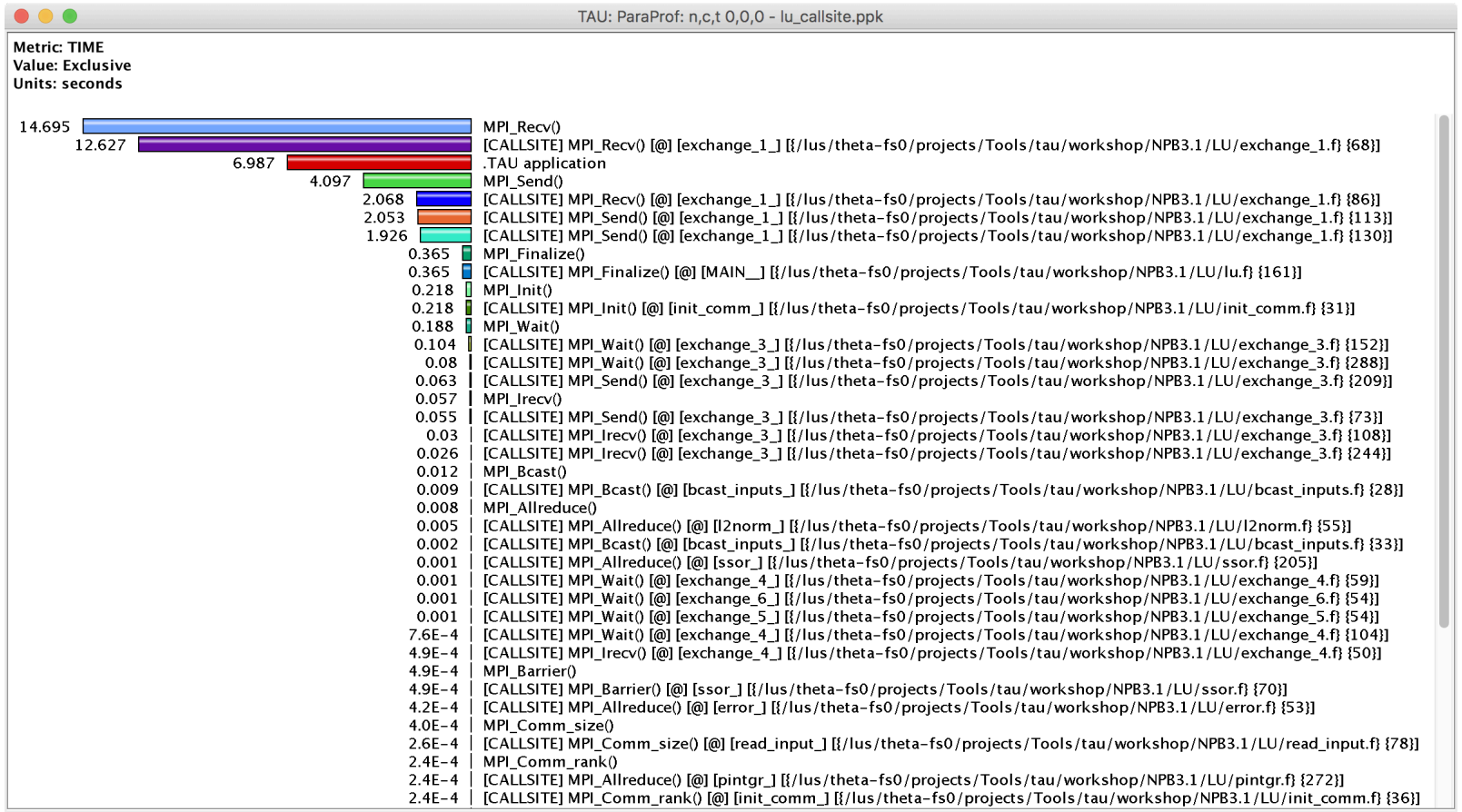
Event Based Sampling (EBS)



Uninstrumented!

```
% mpirun -np 16 tau_exec –ebs a.out
```

Callsite Profiling and Tracing



% export TAU_CALLSITE=1

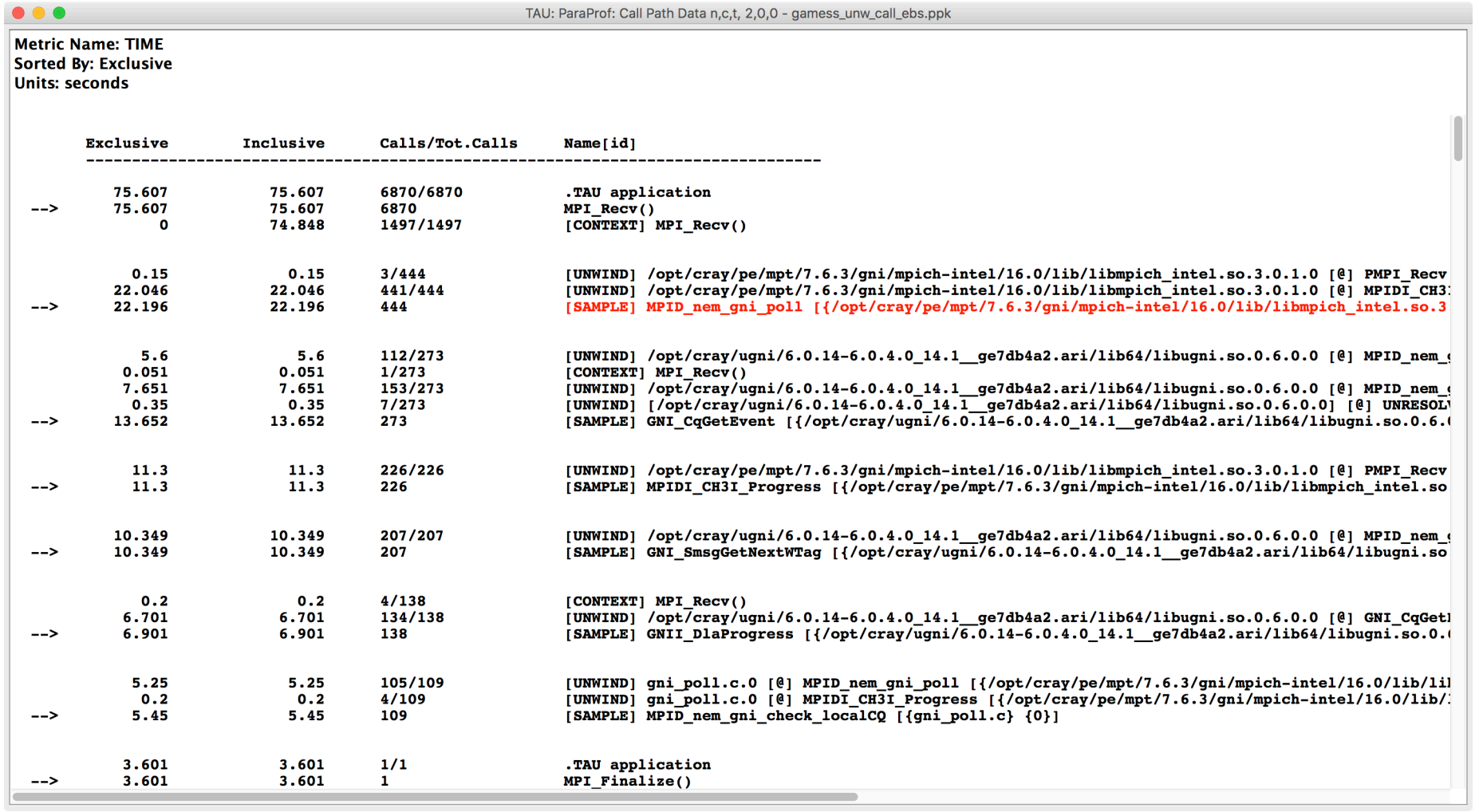
CALLPATH THREAD RELATIONS WINDOW

TAU: ParaProf: Call Path Data n,c,t, 2,0,0 - gamess_unw_call_ebs.ppk

Metric Name: TIME
Sorted By: Inclusive
Units: seconds

	Exclusive	Inclusive	Calls/Tot.Calls	Name[id]
-->	0.121	79.592	1	.TAU application
	0.002	0.002	1/1	MPI_Gather()
	0.004	0.004	3/3	MPI_Allgather()
	0.122	0.122	6866/6866	MPI_Send()
	0.002	0.002	1/1	MPI_Comm_split()
	8.9E-5	8.9E-5	2/2	MPI_Comm_size()
	4.6E-4	4.6E-4	3/3	MPI_Group_incl()
	75.607	75.607	6870/6870	MPI_Recv()
	0.002	0.002	4/4	MPI_Comm_create()
	9.5E-5	9.5E-5	6/6	MPI_Comm_rank()
	5.4E-4	5.4E-4	1/1	MPI_Comm_group()
	0.003	0.003	7/7	MPI_Barrier()
	0.112	0.112	1/1	MPI_Init_thread()
	6.3E-4	6.3E-4	1/1	MPI_Group_intersection()
	0	0.05	1/1	[CONTEXT] .TAU application
	3.601	3.601	1/1	MPI_Finalize()
	0.014	0.014	6/6	MPI_Bcast()
	75.607	75.607	6870/6870	.TAU application
-->	75.607	75.607	6870	MPI_Recv()
	0	74.848	1497/1497	[CONTEXT] MPI_Recv()
	0	74.848	1497/1497	MPI_Recv()
-->	0	74.848	1497	[CONTEXT] MPI_Recv()
	0	8.701	174/1371	[UNWIND] /gpfs/mira-home/yuri/dist/Github/gamess-theta-tau/ddi/src/ddi_init.c.113 [@] ddi_i
	0	26.196	524/763	[UNWIND] /gpfs/mira-home/sameer/gamess-theta-tau/object/unport.f.410 [@] MAIN__ [{/gpfs/mir
	0.2	0.2	4/138	[SAMPLE] GNII_DlaProgress [{/opt/cray/ugni/6.0.14-6.0.4.0_14.1__ge7db4a2.ari/lib64/libugni.
	0	5.75	115/1484	[UNWIND] /gpfs/mira-home/yuri/dist/Github/gamess-theta-tau/ddi/src/ddi_server.c.99 [@] DDI_
	0	0.2	4/5	[UNWIND] /lib64/libc-2.22.so.0 [@] _start [{/home/abuild/rpmbuild/BUILD/glibc-2.22/csu/../s
	0	11.85	237/239	[UNWIND] /gpfs/mira-home/sameer/gamess-theta-tau/object/gamess.f.538 [@] main [{/gpfs/mira-
	0.051	0.051	1/273	[SAMPLE] GNI_CqGetEvent [{/opt/cray/ugni/6.0.14-6.0.4.0_14.1__ge7db4a2.ari/lib64/libugni.so
	0	0.05	1/1197	[UNWIND] /opt/cray/pe/mpt/7.6.3/gni/mpich-intel/16.0/lib/libmpich_intel.so.3.0.1.0 [@] MPID
	0	0.15	3/7	[UNWIND] [/opt/cray/ugni/6.0.14-6.0.4.0_14.1__ge7db4a2.ari/lib64/libugni.so.0.6.0.0] [@] UN
	0	21.7	434/1197	[UNWIND] /gpfs/mira-home/yuri/dist/Github/gamess-theta-tau/ddi/src/ddi_fortran.c.67 [@] beg

CALLPATH THREAD RELATIONS WINDOW



ParaProf: Callpath Thread Relations Window

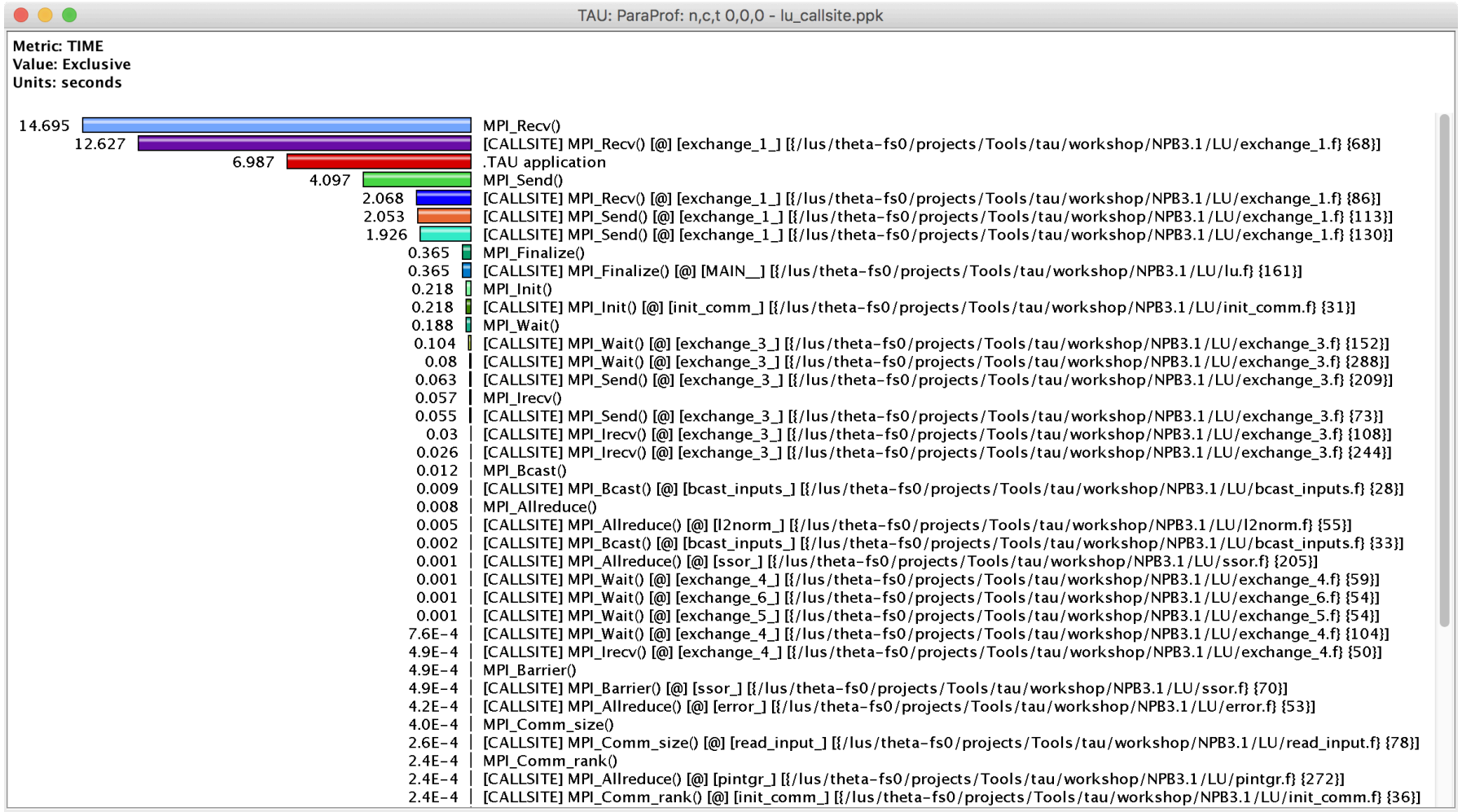
TAU: ParaProf: Call Path Data n,c,t, 0,0,0 – scout.cubex

File Options Windows Help

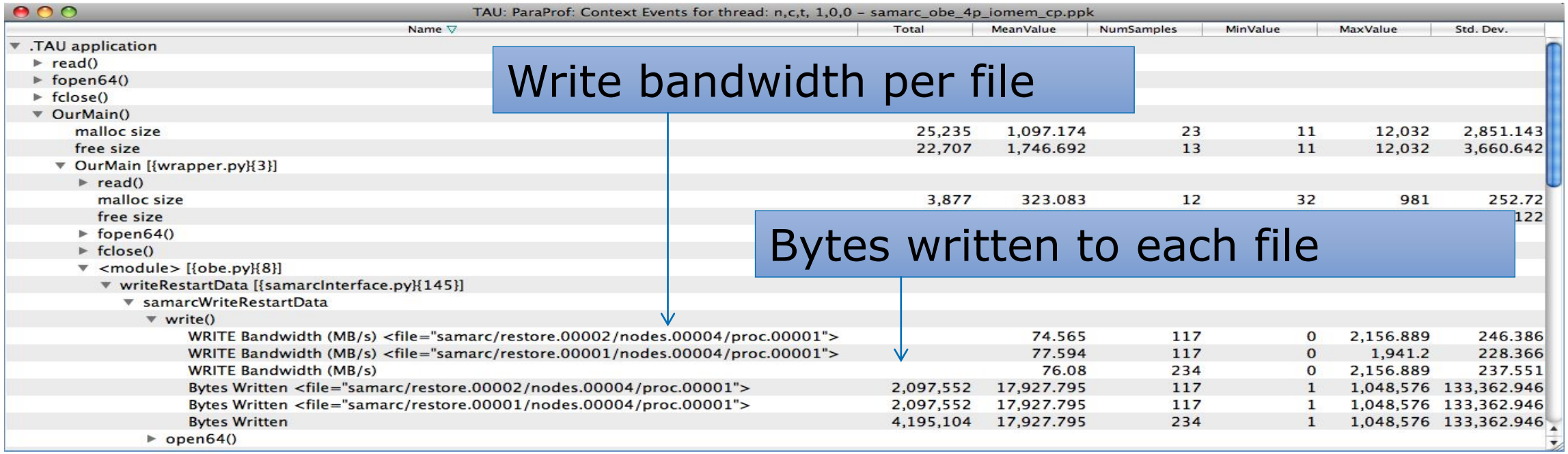
Metric Name: Time
Sorted By: Exclusive
Units: seconds

	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

Callsite Profiling and Tracing (TAU_CALLSITE=1)

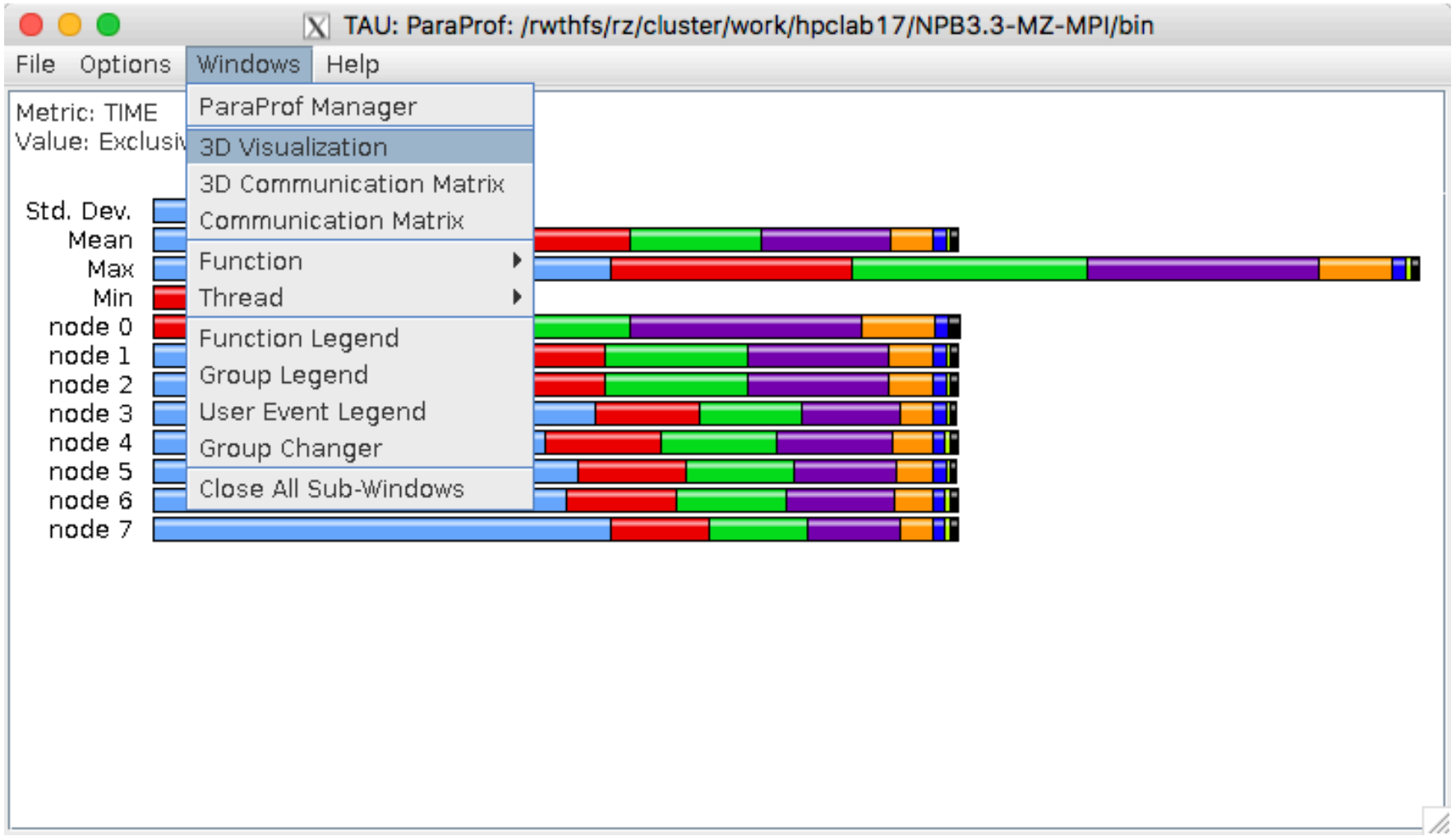


TAU – Context Events

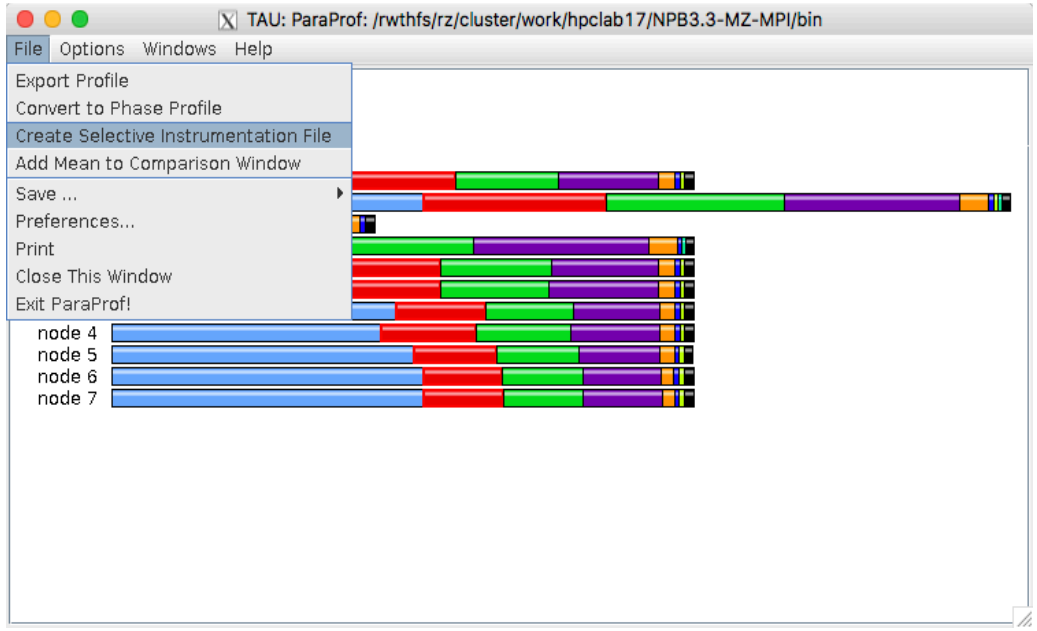


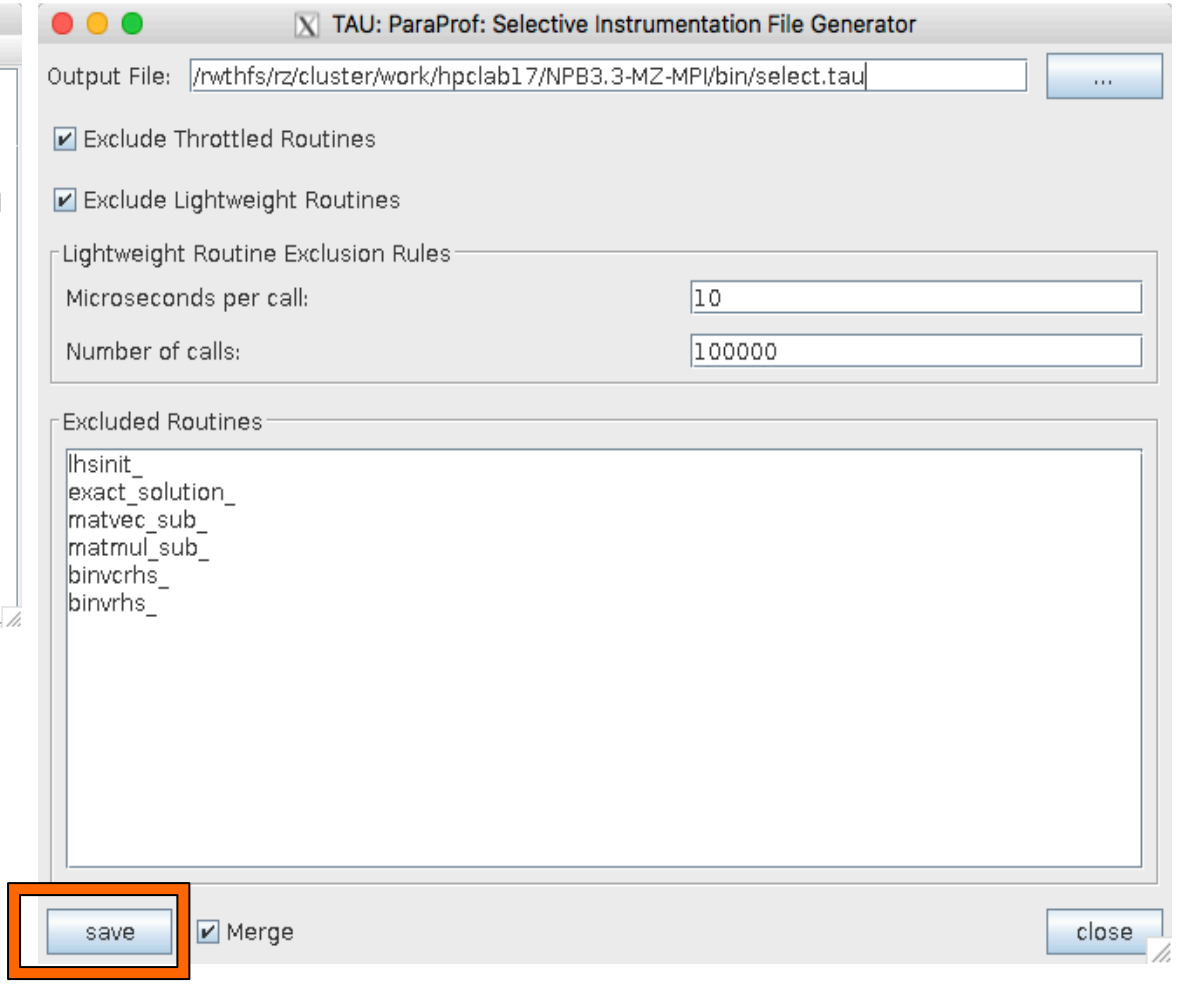
Name	Total	MeanValue	NumSamples	MinValue	MaxValue	Std. Dev.
.TAU application						
read()						
fopen64()						
fclose()						
OurMain()						
malloc size	25,235	1,097.174	23	11	12,032	2,851.143
free size	22,707	1,746.692	13	11	12,032	3,660.642
OurMain [{wrapper.py}{3}]						
read()						
malloc size	3,877	323.083	12	32	981	252.72
free size						122
fopen64()						
fclose()						
<module> [{obe.py}{8}]						
writeRestartData [{samarcInterface.py}{145}]						
samarcWriteRestartData						
write()						
WRITE Bandwidth (MB/s) <file="samarc/restore.00002/nodes.00004/proc.00001">		74.565	117	0	2,156.889	246.386
WRITE Bandwidth (MB/s) <file="samarc/restore.00001/nodes.00004/proc.00001">		77.594	117	0	1,941.2	228.366
WRITE Bandwidth (MB/s)		76.08	234	0	2,156.889	237.551
Bytes Written <file="samarc/restore.00002/nodes.00004/proc.00001">	2,097,552	17,927.795	117	1	1,048,576	133,362.946
Bytes Written <file="samarc/restore.00001/nodes.00004/proc.00001">	2,097,552	17,927.795	117	1	1,048,576	133,362.946
Bytes Written	4,195,104	17,927.795	234	1	1,048,576	133,362.946
open64()						

ParaProf with Optimized Instrumentation

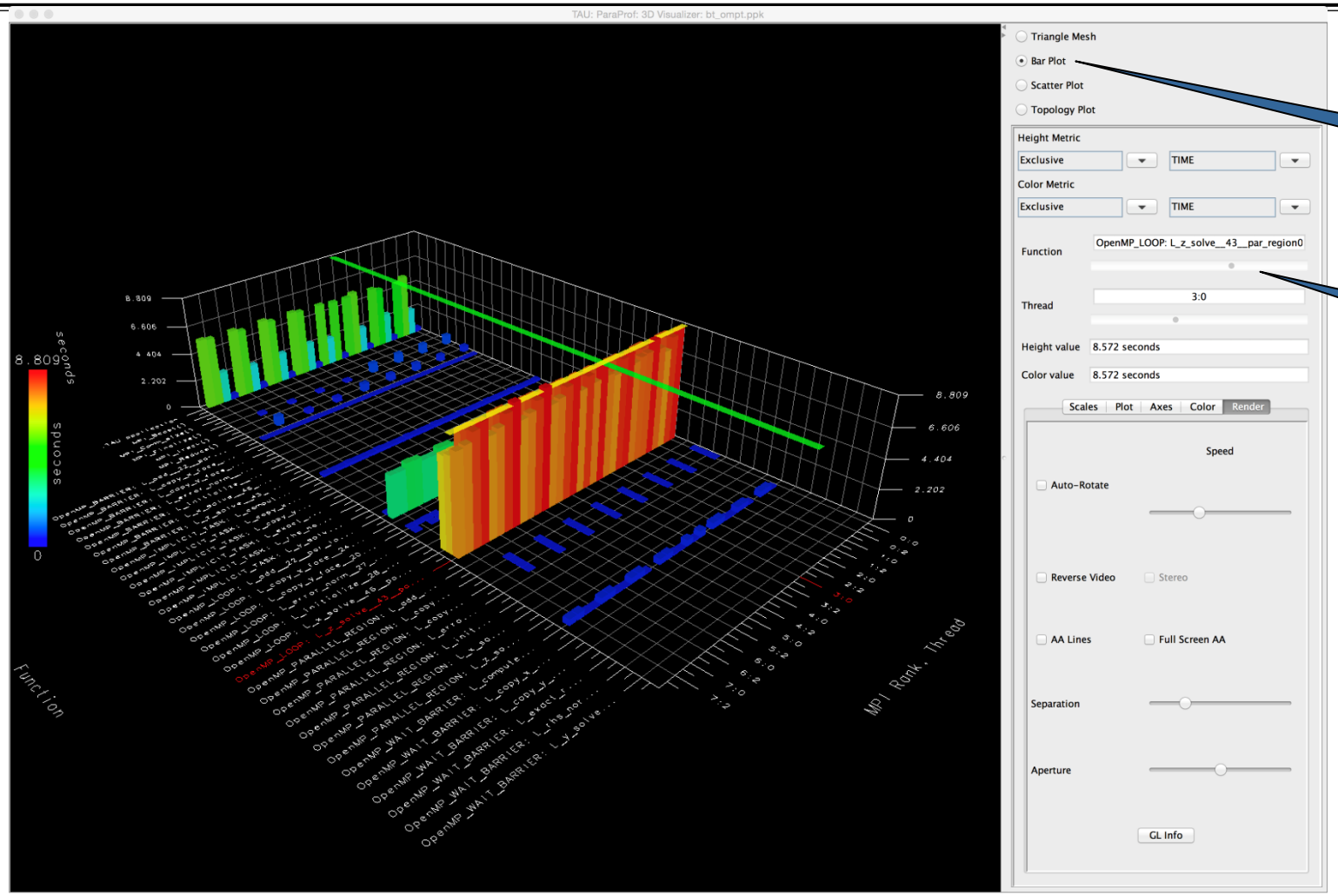


Create a Selective Instrumentation File, Re-instrument, Re-run





Paraprof 3D visualization window

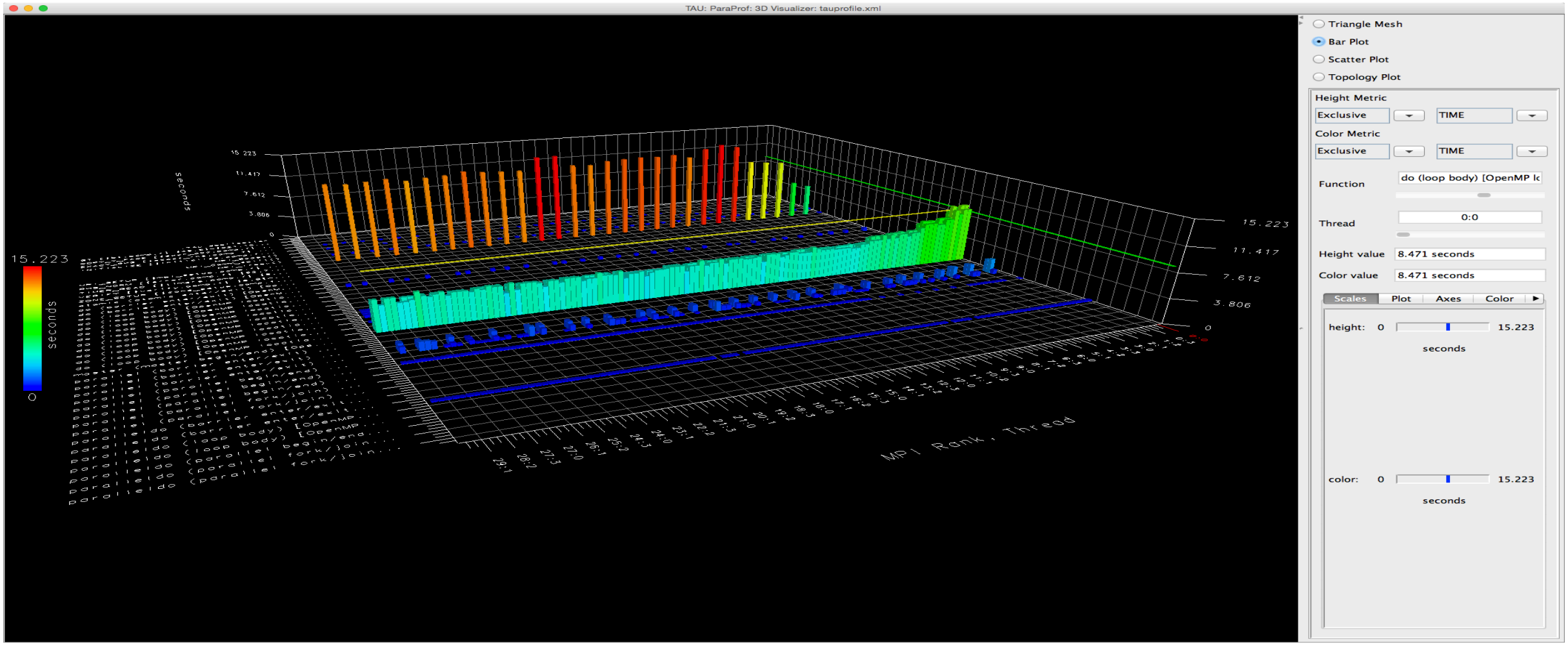


Click Bar Plot

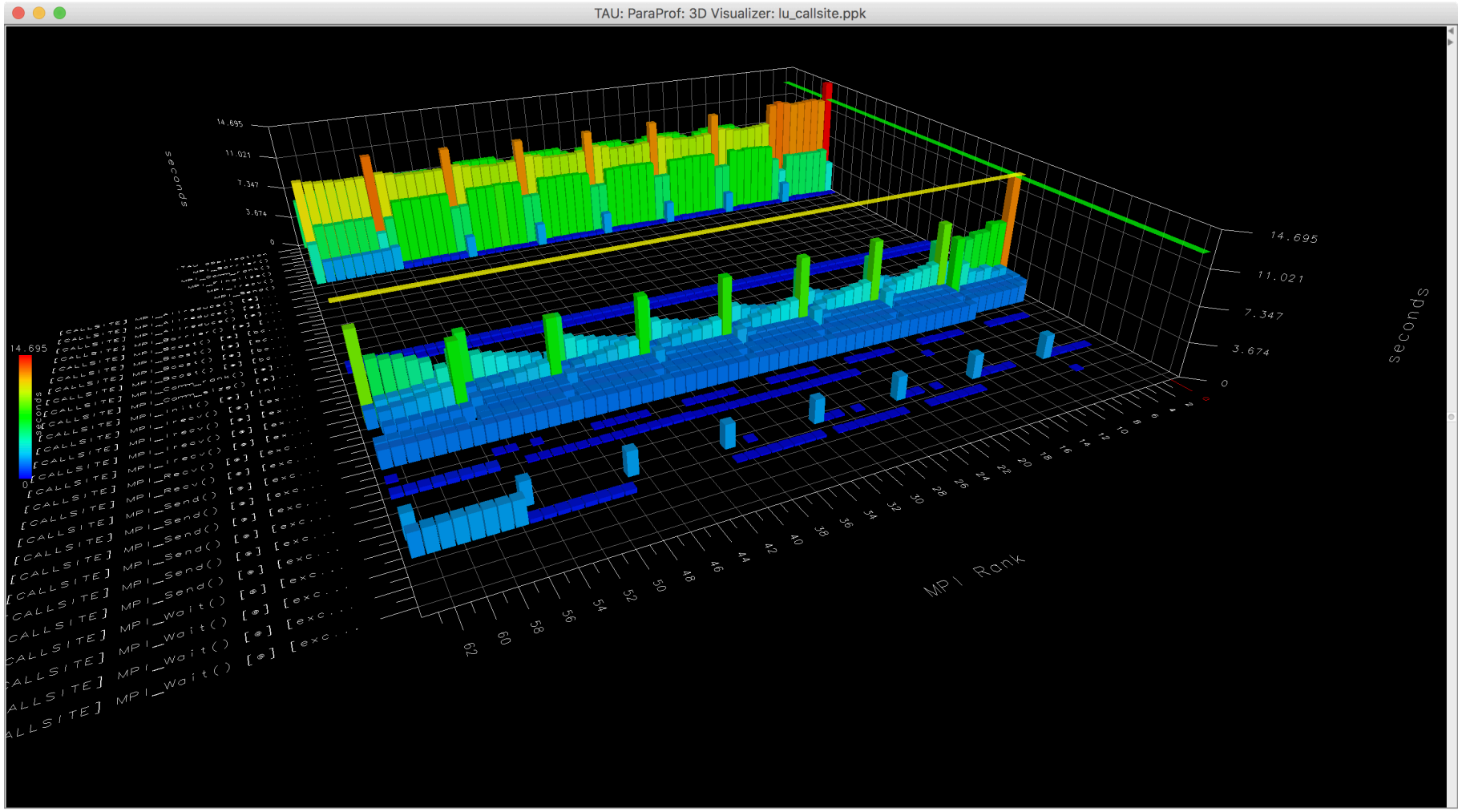
Move Function and Thread Sliders

Windows -> 3D visualization

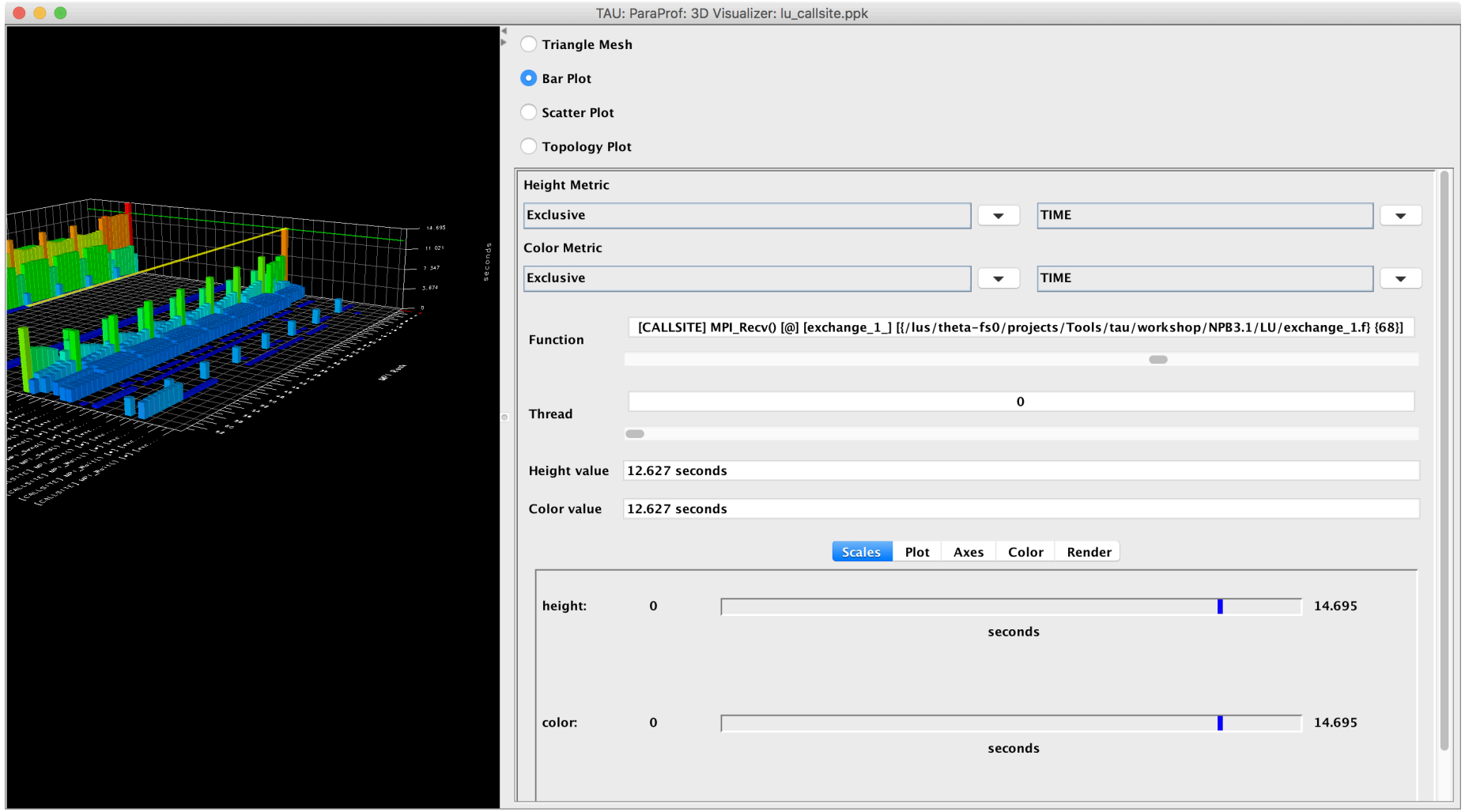
ParaProf: 3D Visualization Window Showing Entire Profile



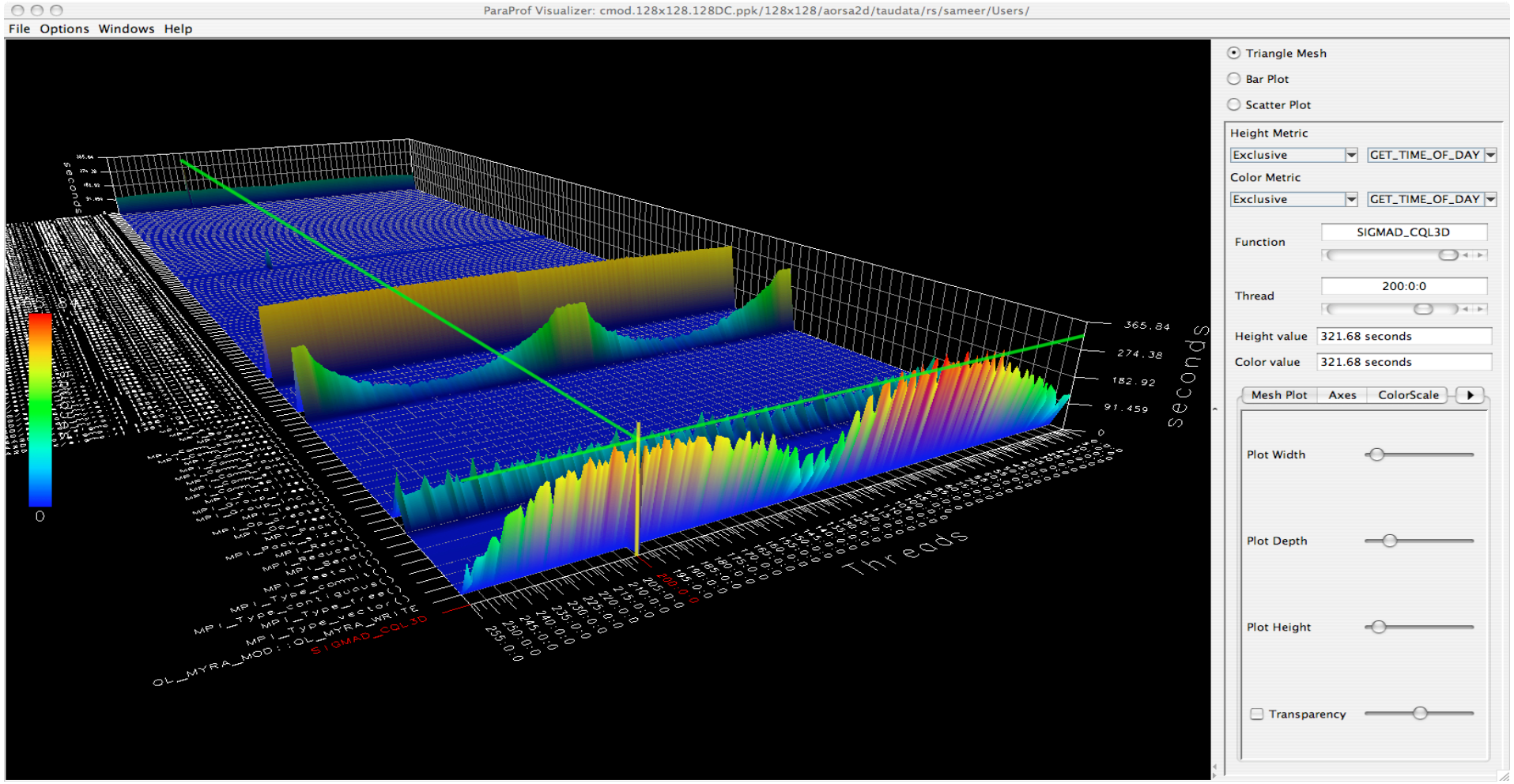
Callsite Profiling and Tracing



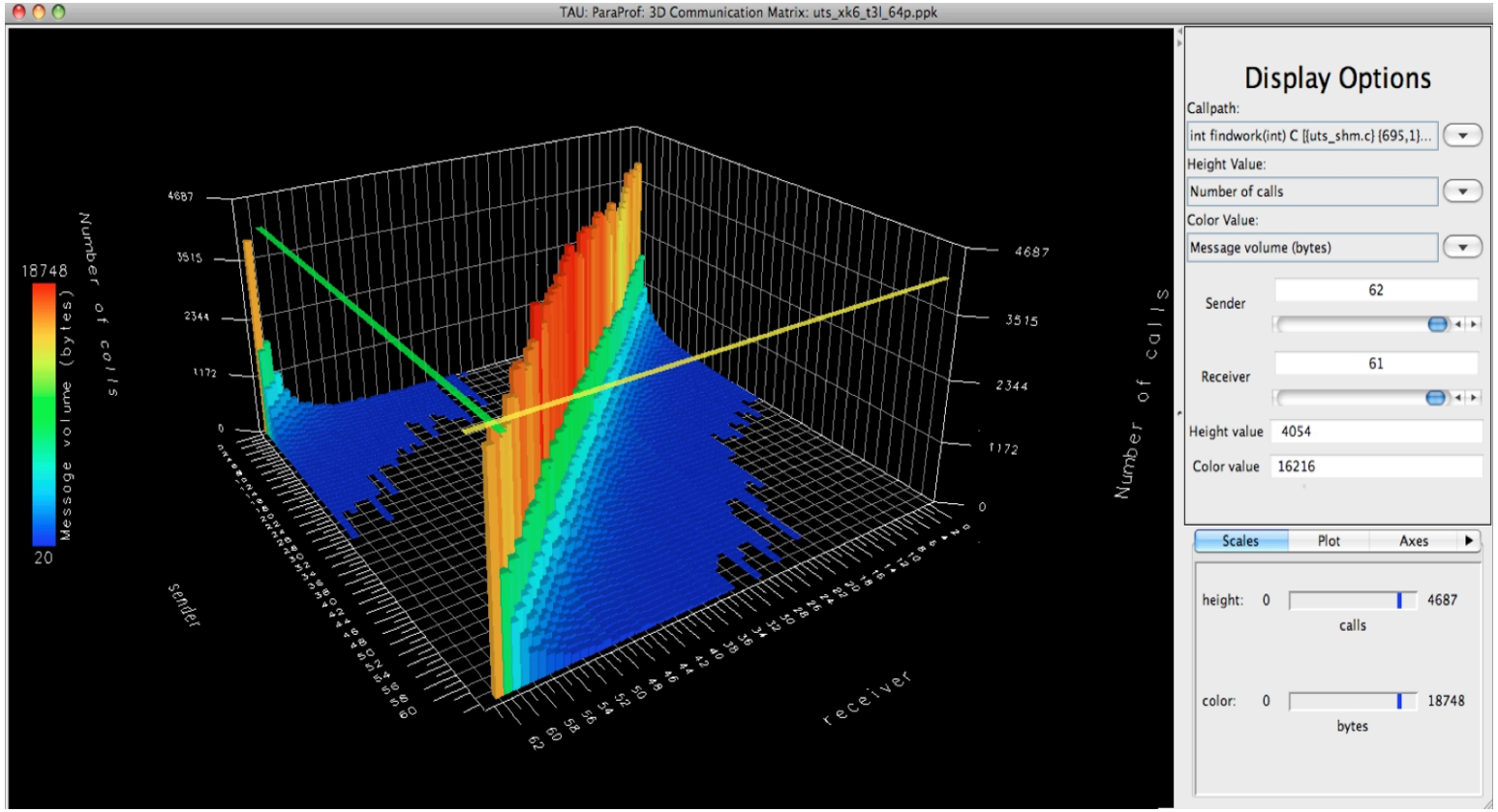
Callsite Profiling and Tracing



Parallel Profile Visualization: ParaProf

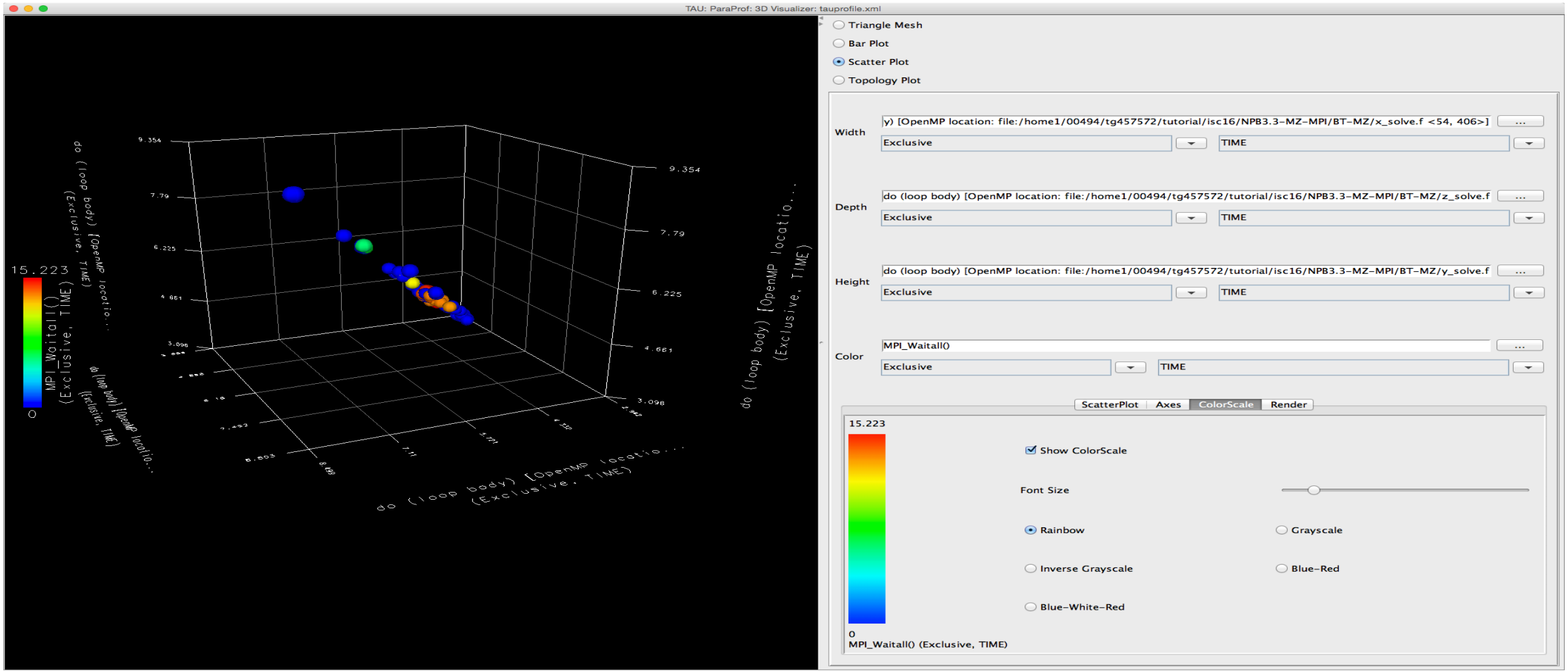


ParaProf 3D Communication Matrix

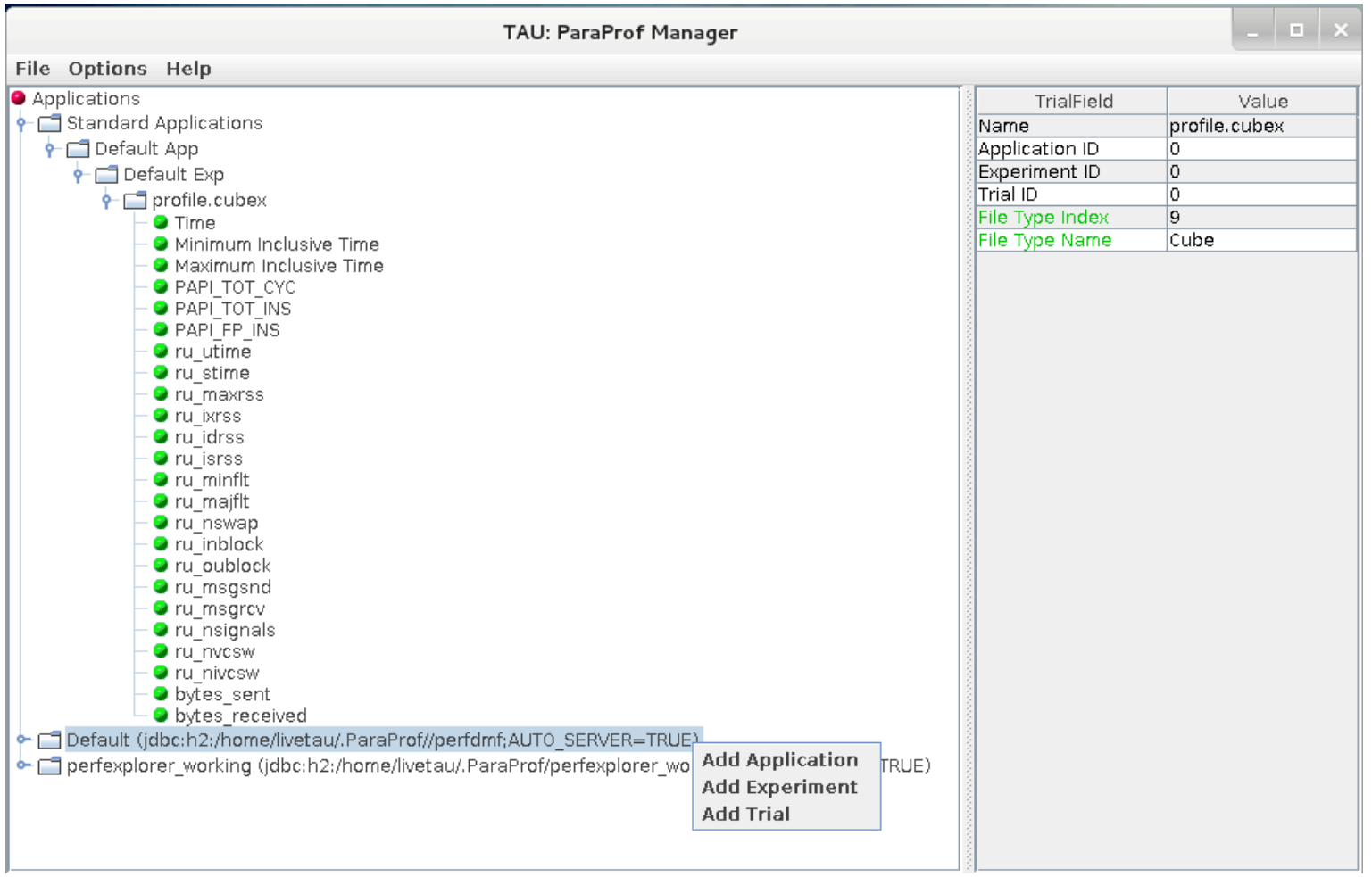


% export TAU_COMM_MATRIX=1

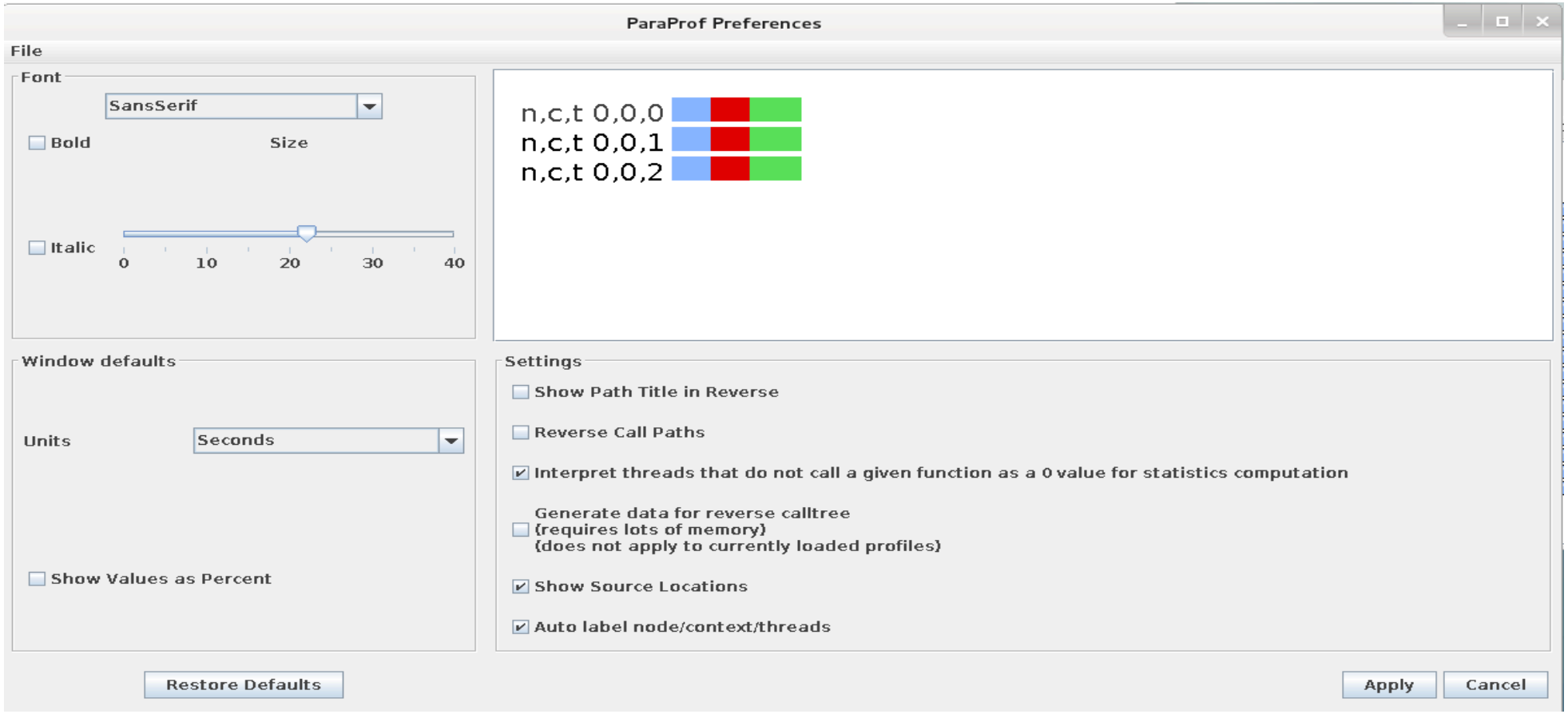
ParaProf: 3D Scatter Plot



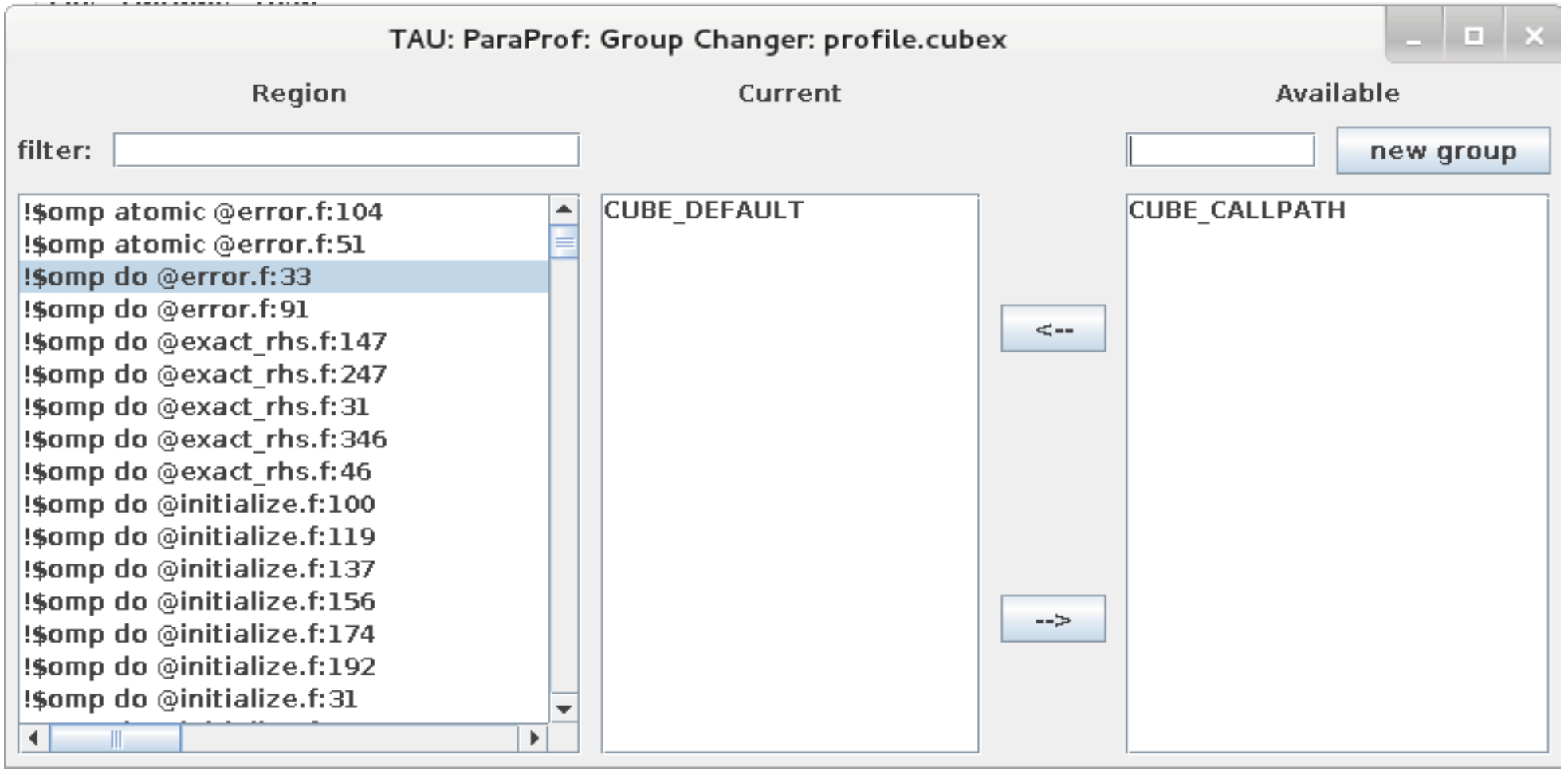
ParaProf: Score-P Profile Files, Database



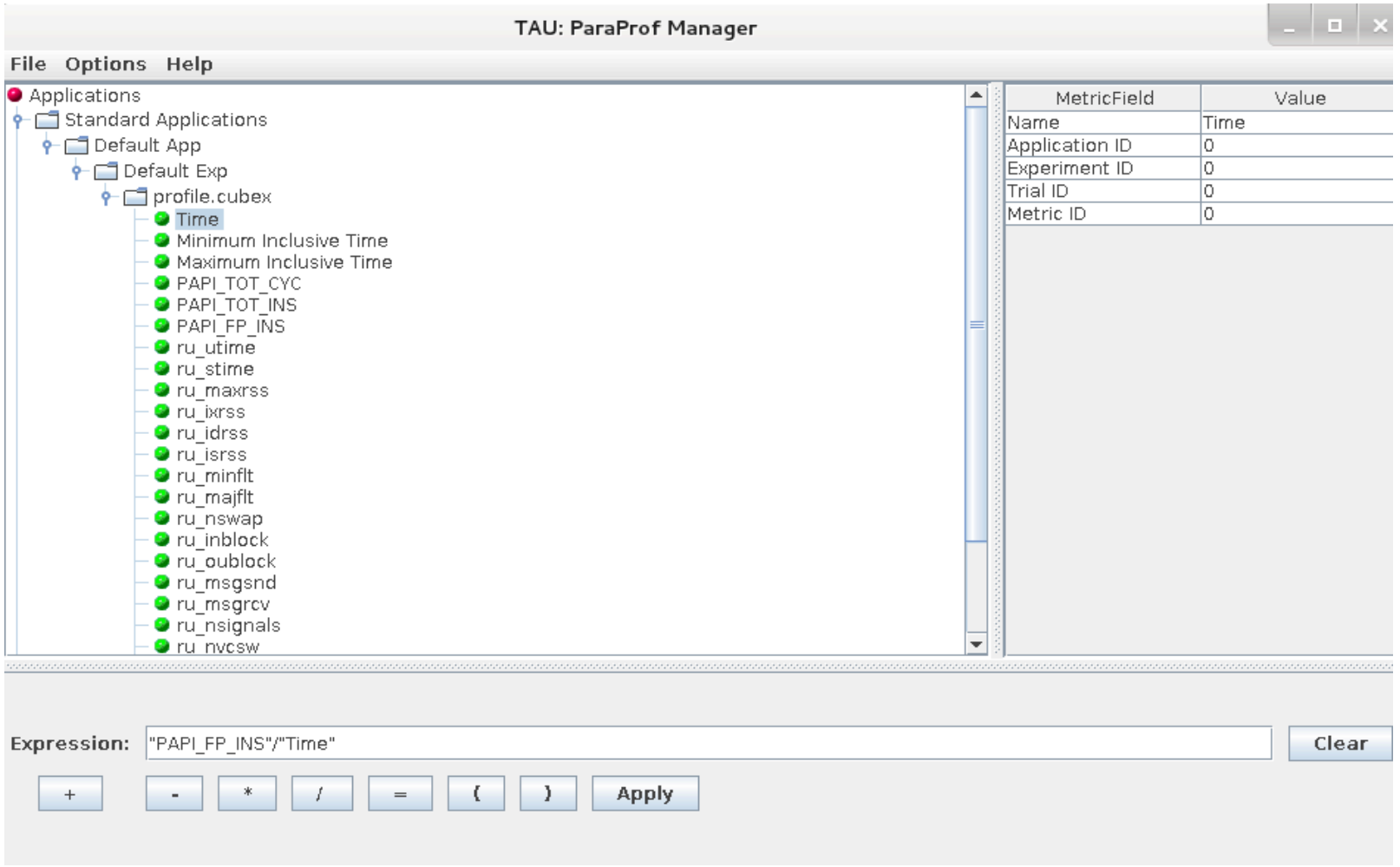
ParaProf: File Preferences Window



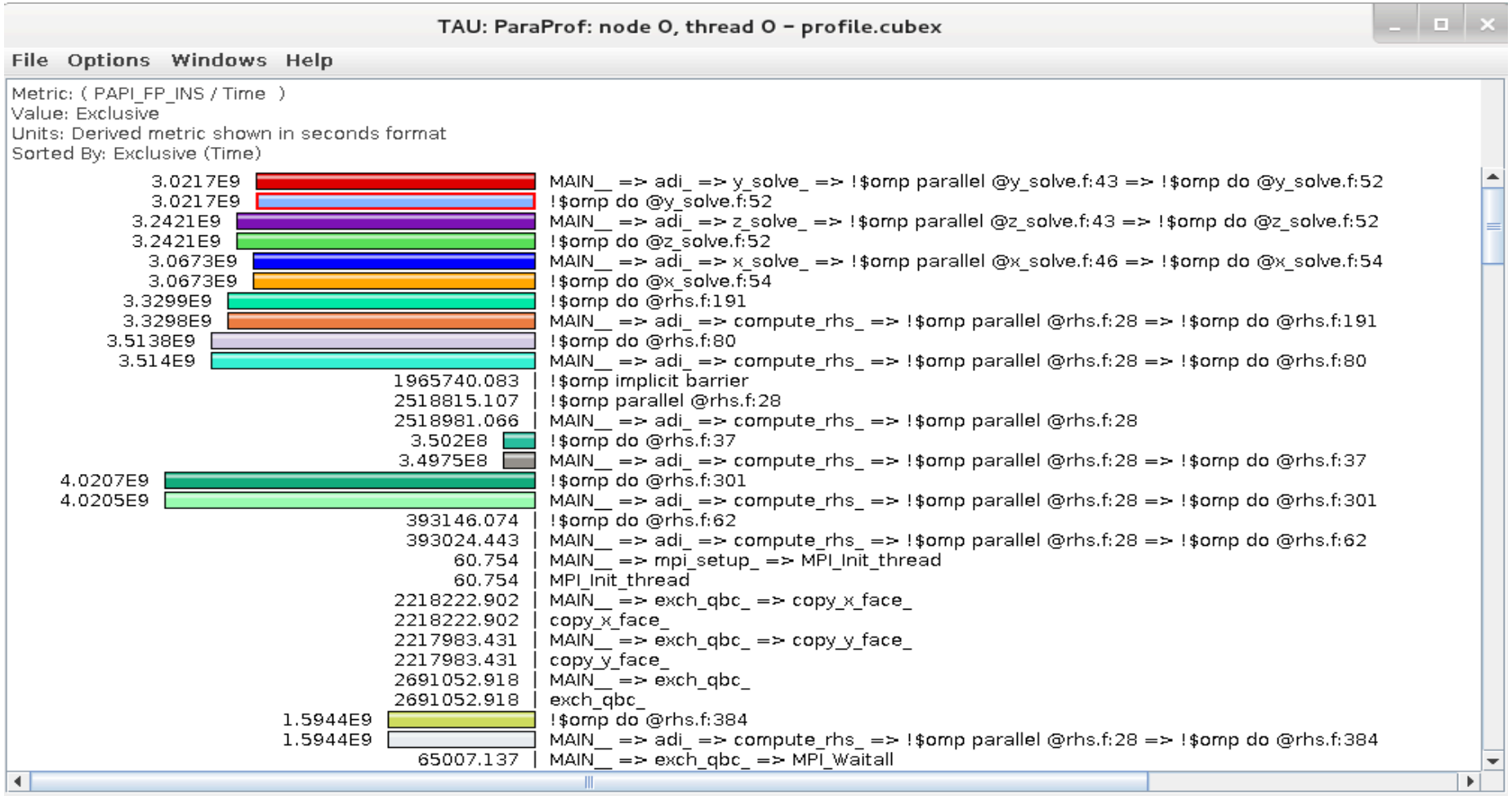
ParaProf: Group Changer Window



ParaProf: Derived Metric Panel in Manager Window



Sorting Derived FLOPS metric by Exclusive Time



TAU hands-on exercises

Performance Research Lab, University of Oregon, Eugene, USA

Support Acknowledgments

- US Department of Energy (DOE)
  - Office of Science contracts, ECP
  - SciDAC, LBL contracts
  - LLNL-LANL-SNL ASC/NNSA contract
  - Battelle, PNNL contract
  - ANL, ORNL contract
- Department of Defense (DoD)
  - PETTT, HPCMP
- National Science Foundation (NSF)
  - Glassbox, SI-2
- NASA
- CEA, France
- Partners:
  - University of Oregon
  - ParaTools, Inc., ParaTools, SAS
  - The Ohio State University
  - University of Tennessee, Knoxville
  - T.U. Dresden, GWT
  - Juelich Supercomputing Center















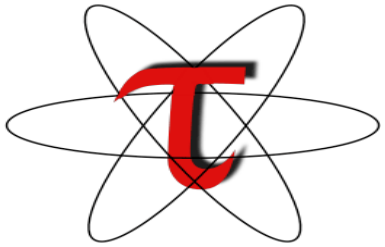




Acknowledgement

This research was supported by the Exascale Computing Project (17-SC-20-SC), a collaborative effort of two U.S. Department of Energy organizations (Office of Science and the National Nuclear Security Administration) responsible for the planning and preparation of a capable exascale ecosystem, including software, applications, hardware, advanced system engineering, and early testbed platforms, in support of the nation's exascale computing imperative.

Download TAU from U. Oregon



http://tau.uoregon.edu

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

https://e4s.io [Containers for Extreme-Scale Scientific Software Stack]

Free download, open source, BSD license