TAU Performance System®

ALCF Computational Performance Workshop
TAU Breakout: May 4, 2021, 10:15am – 10:45am CT

Sameer Shende
Research Associate Professor and Director,
Performance Research Laboratory, OACISS,
University of Oregon
http://tau.uoregon.edu/TAU_ALCF21.pdf

Tuning and Analysis Utilities

UNIVERSITY OF OREGON

ParaTools

EXASCALE COMPUTING PROJECT

National Nuclear Security Administration

U.S. DEPARTMENT OF ENERGY | Office of Science

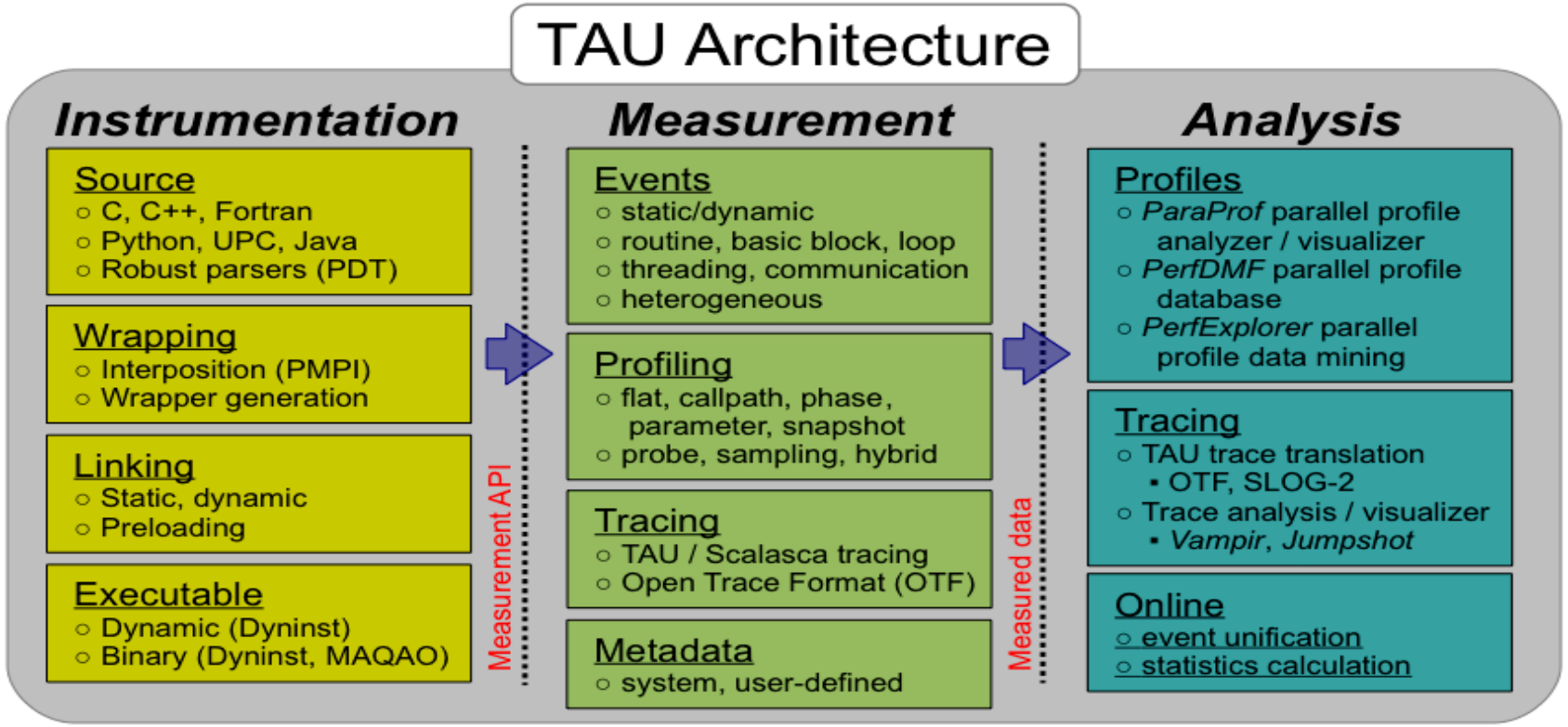
Challenges

- With growing hardware complexity, it is getting harder to accurately measure and optimize the performance of our HPC and AI/ML workloads.
- TAU Performance System®:
  - Deliver a scalable, portable, performance evaluation toolkit for HPC and AI/ML workloads
  - http://tau.uoregon.edu

TAU Performance System®

Parallel performance framework and toolkit

- Supports all HPC platforms, compilers, runtime system
- Provides portable instrumentation, measurement, analysis



TAU Performance System

Instrumentation

- Fortran, C++, C, UPC, Java, Python, Chapel, Spark
- Automatic instrumentation

Measurement and analysis support

- MPI, OpenSHMEM, ARMCI, uGNI, PGAS, DMAPP
- pthreads, OpenMP, OMPT interface, hybrid, other thread models
- GPU: CUPTI, ROCm, OpenCL, oneAPI DPC++/SYCL (Level Zero), OpenACC, Kokkos
- Parallel profiling and tracing

Analysis

- Parallel profile analysis (ParaProf), data mining (PerfExplorer)
- Performance database technology (TAUdb)
- 3D profile browser

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? In kernels on GPUs.
- 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?
- 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?
- How does the application *scale*? What is the efficiency, runtime breakdown of performance across different core counts?

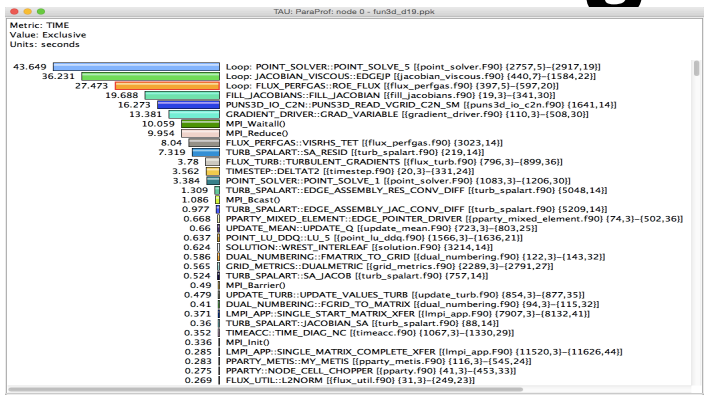
Instrumentation

Add hooks in the code to perform measurements

- **Source instrumentation using a preprocessor**
  - Add timer start/stop calls in a copy of the source code.
  - Use Program Database Toolkit (PDT) for parsing source code.
  - Requires recompiling the code using TAU shell scripts (tau_cc.sh, tau_f90.sh)
  - Selective instrumentation (filter file) can reduce runtime overhead and narrow instrumentation focus.
- **Compiler-based instrumentation**
  - Use system compiler to add a special flag to insert hooks at routine entry/exit.
  - Requires recompiling using TAU compiler scripts (tau_cc.sh, tau_f90.sh…)
- **Runtime preloading of TAU's Dynamic Shared Object (DSO)**
  - No need to recompile code! Use **aprun tau_exec ./app** with options.
  - Requires dynamic executable (link using **–dynamic** on Theta).

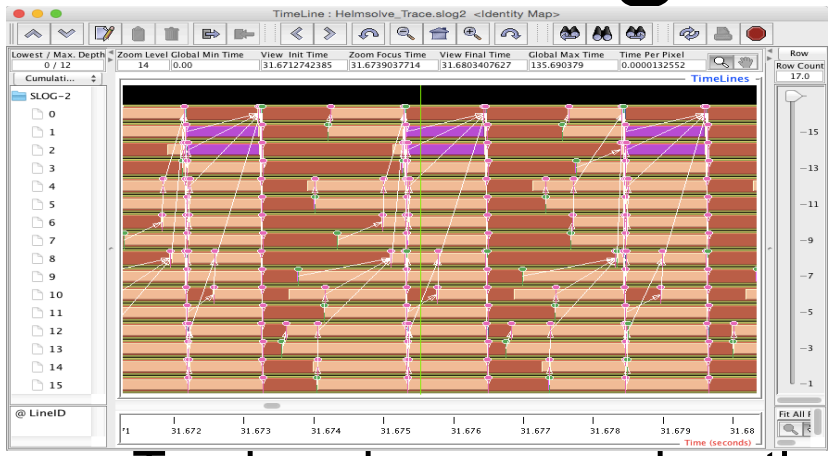
Profiling and Tracing

Profiling



- **Profiling** shows you **how much** (total) time was spent in each routine

Tracing



- Tracing shows you when the events take place on a timeline

- Profiling and tracing

  Profiling shows you **how much** (total) time was spent in each routine

  Tracing shows you **when** the events take place on a timeline

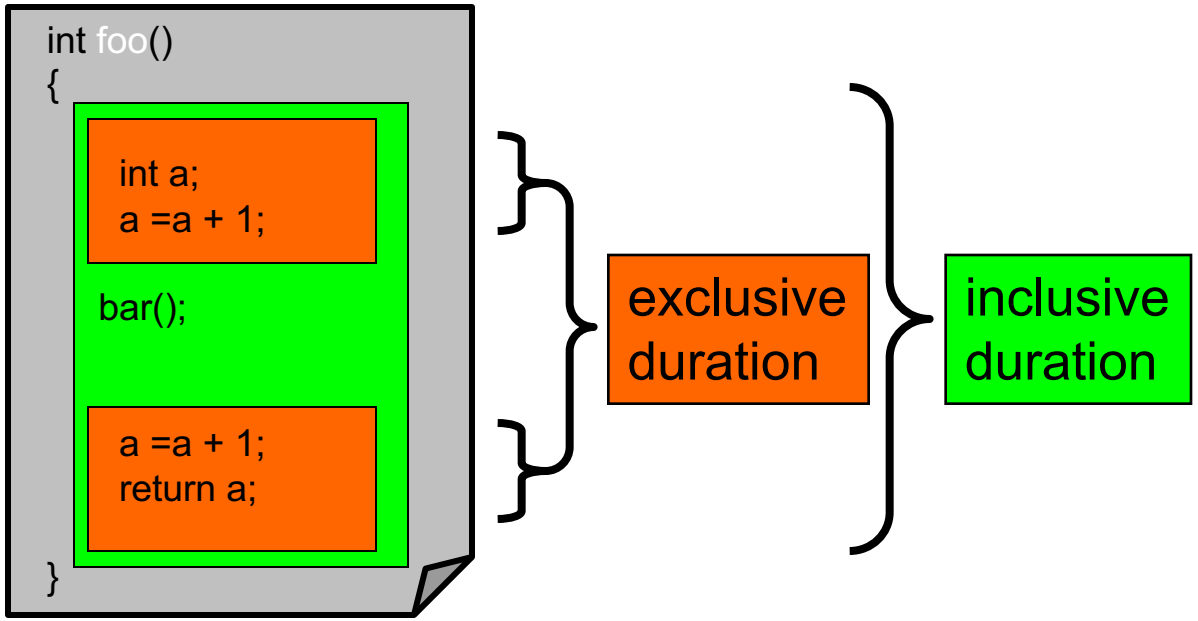
Instrumentation

- Direct and indirect performance observation
- Instrumentation invokes performance measurement
- Direct measurement with *probes*
- Indirect measurement with periodic sampling or hardware performance counter overflow interrupts
- Events measure performance data, metadata, context, etc.

- User-defined events
  - ***Interval*** (start/stop) events to measure exclusive & inclusive duration
  - ***Atomic events*** take measurements at a single point
    - Measures total, samples, min/max/mean/std. deviation statistics
  - ***Context events*** are atomic events with executing context
    - Measures above statistics for a given calling path

Inclusive vs. Exclusive Measurements

- Performance with respect to code regions
- Exclusive measurements for region only
- Inclusive measurements includes child regions



Inclusive Measurements

TAU: ParaProf: node 0 - fun3d_d19.ppk

Metric: TIME
Value: Inclusive
Units: seconds

221.305	.TAU application
221.304	NODET [{main.f90} {4,1}-{35,17}]
197.989	FLOW::ITERATE [{flow.F90} {1692,14}]
195.577	FLOW::STEP_SOLVER [{flow.F90} {1845,14}]
195.569	RELAX_STEADY::RELAX [{relax_steady.f90} {30,3}-{307,22}]
61.927	UPDATE_MEAN::UPDATE_LINEAR_SYSTEM_MEAN [{update_mean.F90} {195,3}-{275,42}]
61.28	UPDATE_MEAN::UPDATE_JACOBIAN_DRIVER_MEAN [{update_mean.F90} {460,3}-{505,44}]
61.275	UPDATE_MEAN::UPDATE_JACOBIAN [{update_mean.F90} {513,3}-{588,32}]
61.258	FILL_JACOBIANS::FILL_JACOBIAN [{fill_jacobians.f90} {19,3}-{341,30}]
59.068	GCR_SOLVE::GCR_SOLVER_QSET [{gcr_solve.f90} {47,3}-{415,32}]
57.635	GCR_SOLVE_UTIL::GCR_PRECONDITIONER_QSET [{gcr_solve_util.f90} {40,3}-{131,40}]
57.152	POINT_SOLVER::POINT_SOLVE [{point_solver.F90} {31,3}-{214,28}]
56.882	UPDATE_MEAN::UPDATE_RHS_MEAN [{update_mean.F90} {102,3}-{185,32}]
54.402	RELAX_MEAN::RELAX [{relax_mean.f90} {22,3}-{84,22}]
53.103	LINEARSOLVE_NODIVCHECK::NODIVCHECK_RELAX_Q [{linearsolve_nodivcheck.F90} {56,14}]
52.867	UPDATE_MEAN::RESIDUAL_S [{update_mean.F90} {42,3}-{94,27}]
52.866	FUN3D_RES_FLOW::RES_FLOW [{fun3d_res_flow.f90} {27,3}-{279,25}]
52.756	FLUX::RESIDUAL_COMPRESSIBLE [{flux.f90} {25,3}-{592,38}]
52.747	POINT_SOLVER::POINT_SOLVE_5 [{point_solver.F90} {2700,3}-{2921,30}]
52.744	Loop: POINT_SOLVER::POINT_SOLVE_5 [{point_solver.F90} {2757,5}-{2917,19}]
36.232	JACOBIAN_VISCOUS::VISCOUS_JACOBIAN [{jacobian_viscous.f90} {20,14}]
36.231	JACOBIAN_VISCOUS::EDGEJP [{jacobian_viscous.f90} {324,14}]
36.231	Loop: JACOBIAN_VISCOUS::EDGEJP [{jacobian_viscous.f90} {440,7}-{1584,22}]
27.474	FLUX_PERFGAS::INVISCID_FLUX_DRIVER [{flux_perfgas.f90} {37,14}]
27.474	FLUX_PERFGAS::ROE_FLUX [{flux_perfgas.f90} {236,14}]
27.473	Loop: FLUX_PERFGAS::ROE_FLUX [{flux_perfgas.f90} {397,5}-{597,20}]
22.707	FLOW::INITIALIZE_DATA [{flow.F90} {465,14}]
22.694	FLOW::INITIALIZE_DATA2 [{flow.F90} {663,14}]
20.916	PPARTY_PREPROCESSOR::PPARTY_PREPROCESS [{pparty_preprocessor.f90} {28,14}]
16.726	PPARTY_PREPROCESSOR::PPARTY_READ_GRID [{pparty_preprocessor.f90} {735,14}]
16.726	PUNS3D_IO_C2N::PUNS3D_READ_VGRID_C2N [{puns3d_io_c2n.f90} {1543,14}]
16.657	PUNS3D_IO_C2N::PUNS3D_READ_VGRID_C2N_SM [{puns3d_io_c2n.f90} {1641,14}]
14.159	GRADIENT_DRIVER::GRAD_VARIABLE [{gradient_driver.f90} {110,3}-{508,30}]
13.852	UPDATE_TURB::UPDATE_RHS_TURB [{update_turb.f90} {742,3}-{845,32}]

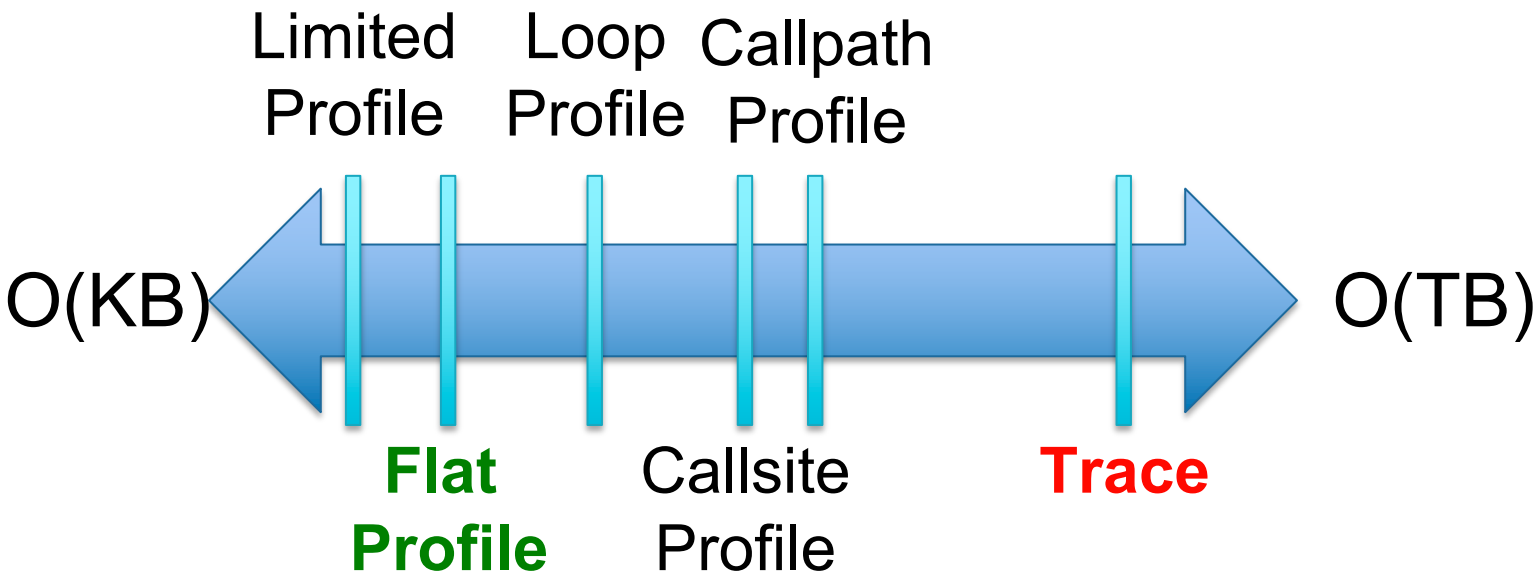
Exclusive Time

TAU: ParaProf: node 0 - fun3d_d19.ppk

Metric: TIME
Value: Exclusive
Units: seconds

43.649	Loop: POINT_SOLVER::POINT_SOLVE_5 [{point_solver.F90} {2757,5}-{2917,19}]
36.231	Loop: JACOBIAN_VISCOUS::EDGEJP [{jacobian_viscous.f90} {440,7}-{1584,22}]
27.473	Loop: FLUX_PERFGAS::ROE_FLUX [{flux_perfgas.f90} {397,5}-{597,20}]
19.688	FILL_JACOBIANS::FILL_JACOBIAN [{fill_jacobians.f90} {19,3}-{341,30}]
16.273	PUNS3D_IO_C2N::PUNS3D_READ_VGRID_C2N_SM [{puns3d_io_c2n.f90} {1641,14}]
13.381	GRADIENT_DRIVER::GRAD_VARIABLE [{gradient_driver.f90} {110,3}-{508,30}]
10.059	MPI_Waitall()
9.954	MPI_Reduce()
8.04	FLUX_PERFGAS::VISRHS_TET [{flux_perfgas.f90} {3023,14}]
7.319	TURB_SPALART::SA_RESID [{turb_spalart.f90} {219,14}]
3.78	FLUX_TURB::TURBULENT_GRADIENTS [{flux_turb.f90} {796,3}-{899,36}]
3.562	TIMESTEP::DELTAT2 [{timestep.f90} {20,3}-{331,24}]
3.384	POINT_SOLVER::POINT_SOLVE_1 [{point_solver.F90} {1083,3}-{1206,30}]
1.309	TURB_SPALART::EDGE_ASSEMBLY_RES_CONV_DIFF [{turb_spalart.f90} {5048,14}]
1.086	MPI_Bcast()
0.977	TURB_SPALART::EDGE_ASSEMBLY_JAC_CONV_DIFF [{turb_spalart.f90} {5209,14}]
0.668	PPARTY_MIXED_ELEMENT::EDGE_POINTER_DRIVER [{pparty_mixed_element.f90} {74,3}-{502,36}]
0.66	UPDATE_MEAN::UPDATE_Q [{update_mean.F90} {723,3}-{803,25}]
0.637	POINT_LU_DDQ::LU_5 [{point_lu_ddq.f90} {1566,3}-{1636,21}]
0.624	SOLUTION::WREST_INTERLEAF [{solution.F90} {3214,14}]
0.586	DUAL_NUMBERING::FMATRIX_TO_GRID [{dual_numbering.f90} {122,3}-{143,32}]
0.565	GRID_METRICS::DUALMETRIC [{grid_metrics.f90} {2289,3}-{2791,27}]
0.524	TURB_SPALART::SA_JACOB [{turb_spalart.f90} {757,14}]
0.49	MPI_Barrier()
0.479	UPDATE_TURB::UPDATE_VALUES_TURB [{update_turb.f90} {854,3}-{877,35}]
0.41	DUAL_NUMBERING::FGRID_TO_MATRIX [{dual_numbering.f90} {94,3}-{115,32}]
0.371	LMPI_APP::SINGLE_START_MATRIX_XFER [{lmpi_app.F90} {7907,3}-{8132,41}]
0.36	TURB_SPALART::JACOBIAN_SA [{turb_spalart.f90} {88,14}]
0.352	TIMEACC::TIME_DIAG_NC [{timeacc.f90} {1067,3}-{1330,29}]
0.336	MPI_Init()
0.285	LMPI_APP::SINGLE_MATRIX_COMPLETE_XFER [{lmpi_app.F90} {11520,3}-{11626,44}]
0.283	PPARTY_METIS::MY_METIS [{pparty_metis.F90} {116,3}-{545,24}]
0.275	PPARTY::NODE_CELL_CHOPPER [{pparty.f90} {41,3}-{453,33}]
0.269	FLUX_UTIL::L2NORM [{flux_util.f90} {31,3}-{249,23}]

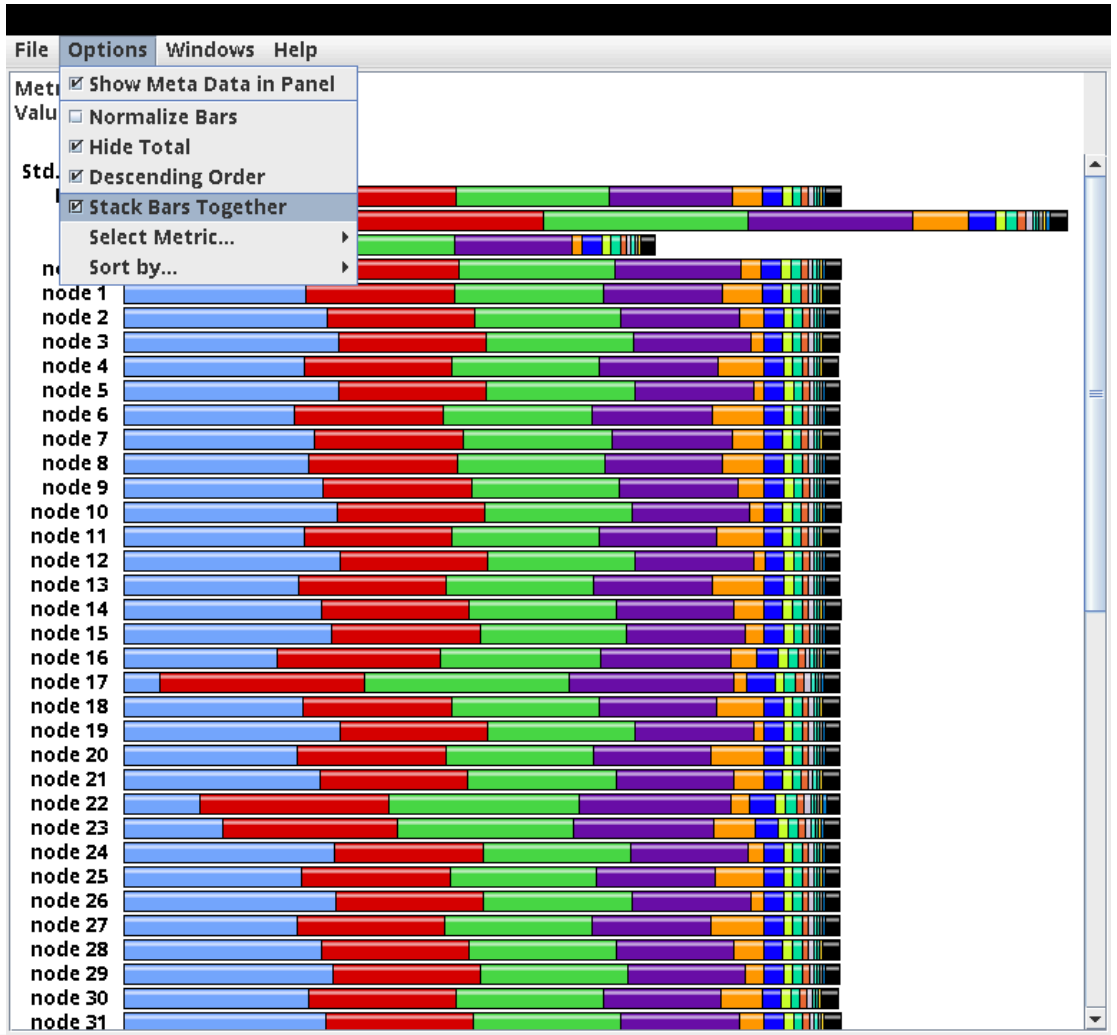
How much data do you want?

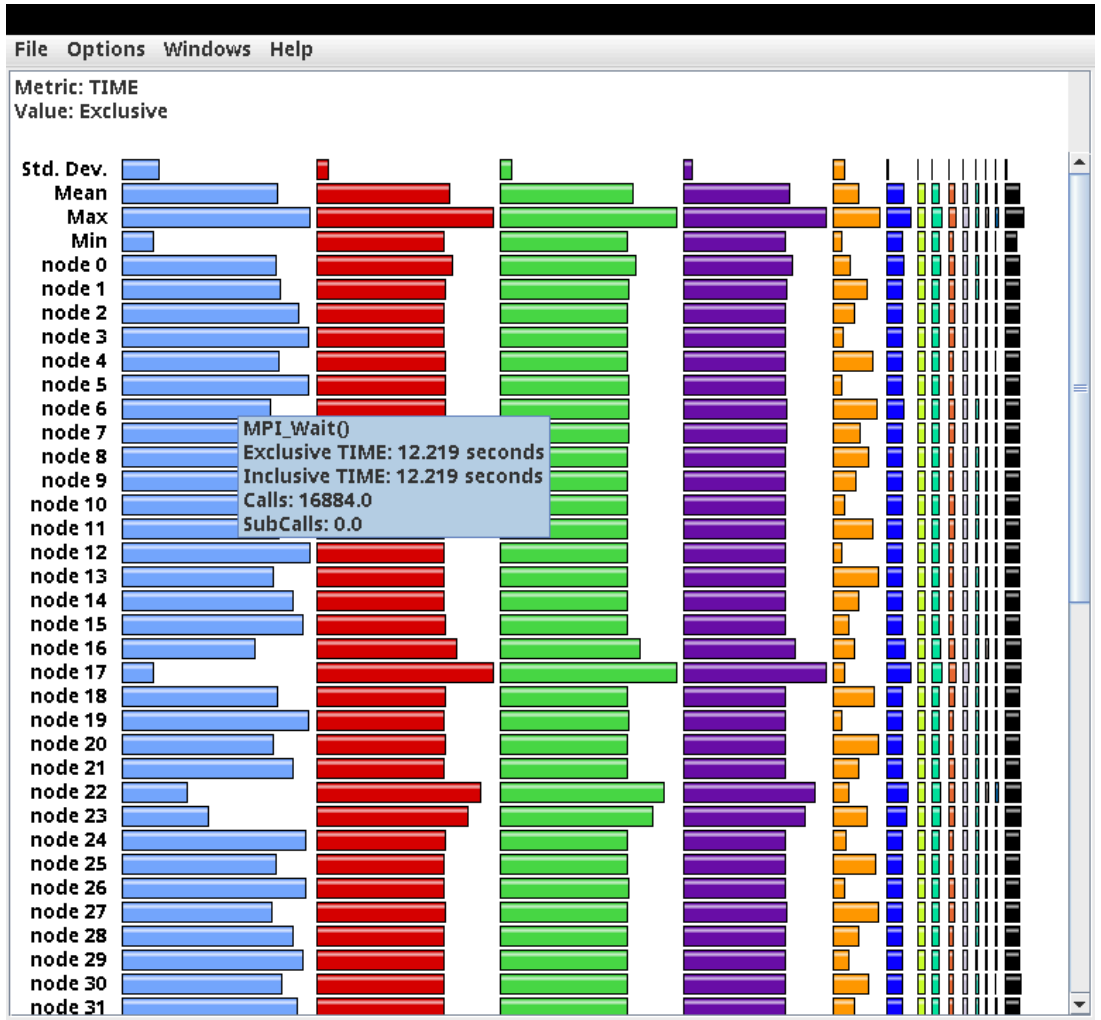


ParaProf Profile Browser

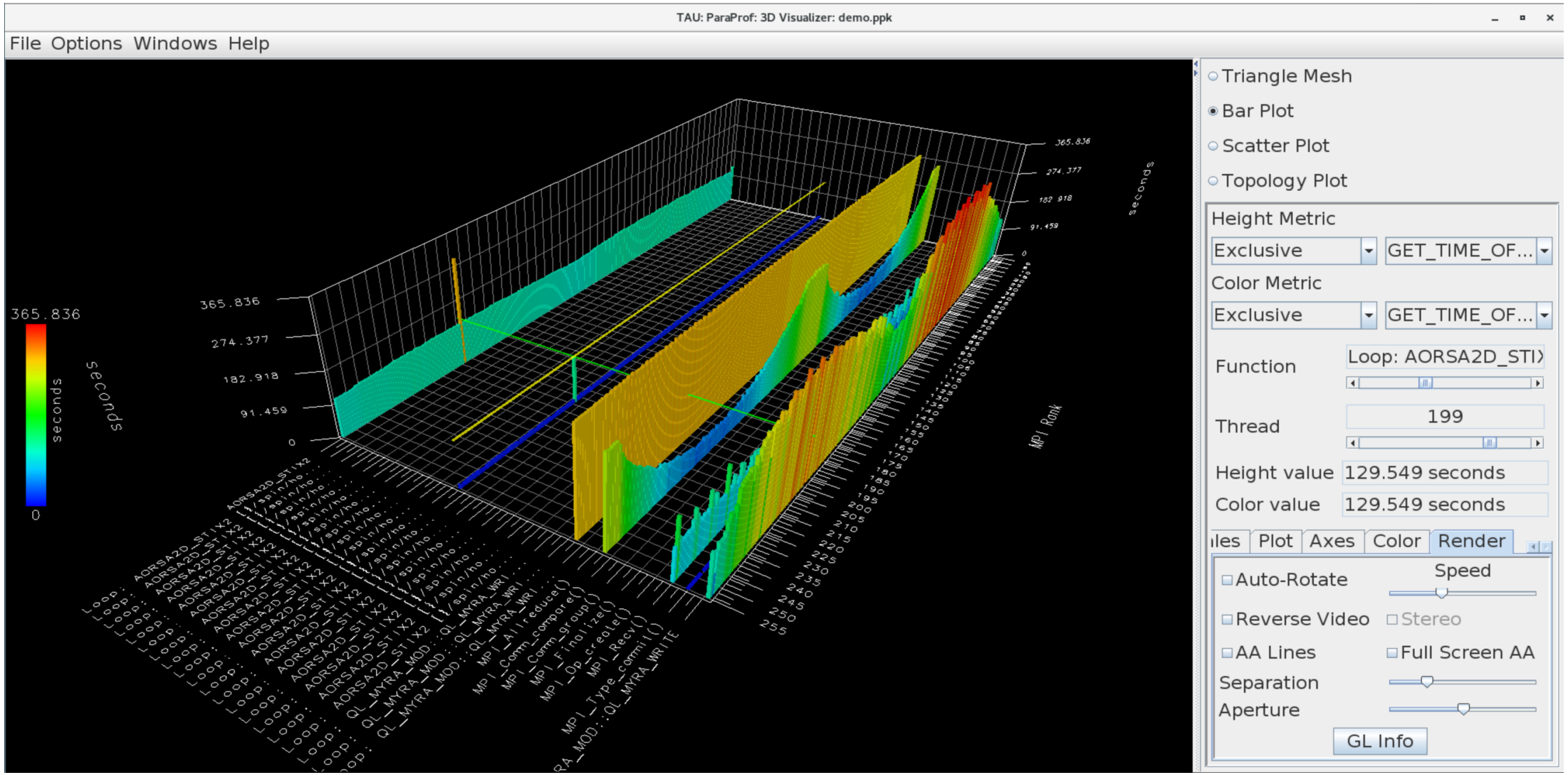
```
% paraprof
```

ParaProf Profile Browser

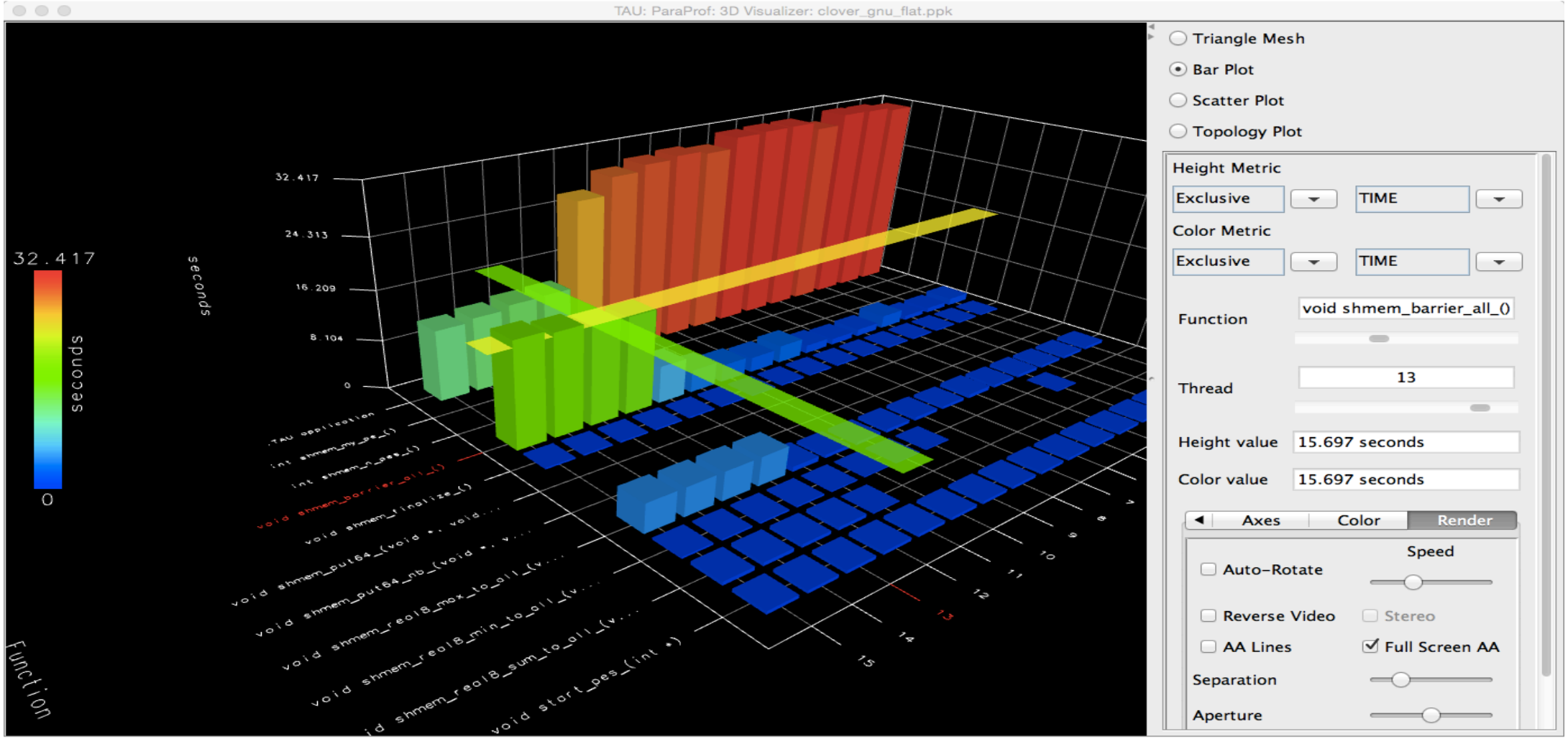




ParaProf 3D Profile Browser

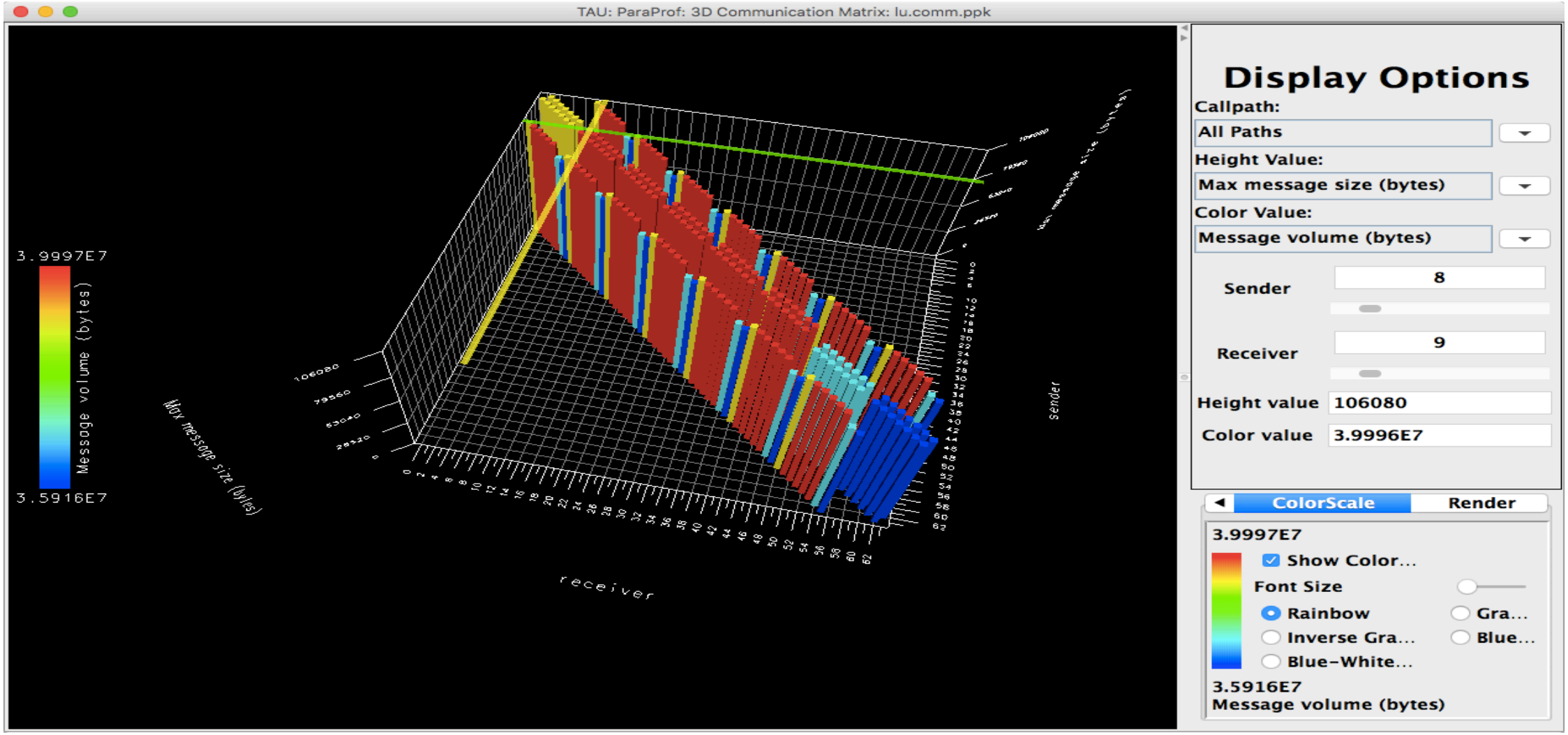


TAU – ParaProf 3D Visualization



% paraprof app.ppk
Windows -> 3D Visualization -> Bar Plot (right pane)

TAU – 3D Communication Window



% export TAU_COMM_MATRIX=1; mpirun … tau_exec ./a.out
% paraprof ; Windows -> 3D Communication Matrix

TAU: Intel oneAPI DPC++ on an Intel Gen12LP or DG1 GPU

TAU: ParaProf: Statistics for: node 0, thread 1 - iso3dfd_dpcpp.ppk

Name	Exclusive TAUGPU_TI...	Inclusive TAUGPU_TIME	Calls	Child Calls
.TAU application	0.18	22.279	1	10,002
_ZTSZZ13Iso3dfdDeviceRN2cl4sycl5queueEPfS3_S3_S3_mmmmmmmjENKUlRT_E313_16clINS0_7handlerEEEDaS5_EUlS4_E399_58	11.063	11.063	5,000	0
_ZTSZZ13Iso3dfdDeviceRN2cl4sycl5queueEPfS3_S3_S3_mmmmmmmjENKUlRT_E313_16clINS0_7handlerEEEDaS5_EUlS4_E407_58	11.033	11.033	5,000	0
zeCommandListAppendMemoryCopy	0.003	0.003	2	0

TAU: ParaProf: Statistics for: node 0, thread 0 - iso3dfd_dpcpp.ppk

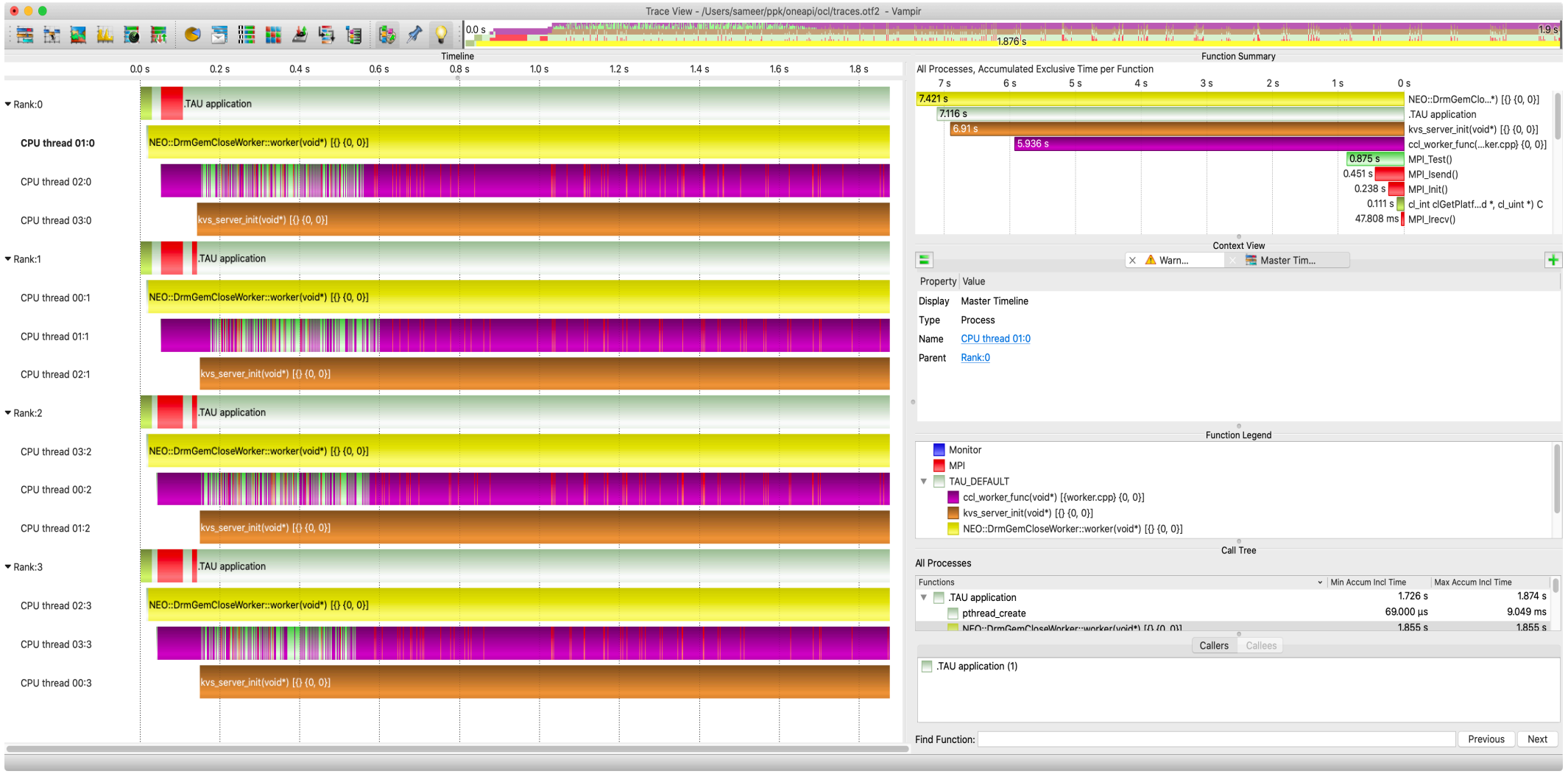
Name	Exclusive TAUGPU_...	Inclusive TAUGPU_...	Calls	Child Calls
pthread_create	0	0	1	0
.TAU application	22.73	22.73	1	1
[CONTEXT] .TAU application	0	22.71	729	0
[SAMPLE] std::_Sp_counted_ptr_inplace<cl::sycl::detail::event_impl, std::allocator<cl::sycl::detail::event_impl>, (__gnu_cxx::_Lock_policy)2	0.03	0.03	1	0
[SAMPLE] cl::sycl::detail::pi::emitFunctionEndTrace(unsigned long, char const*) [{crtstuff.c} {0}]	0.09	0.09	2	0
[SAMPLE] cl::sycl::detail::Scheduler::GraphBuilder::cleanupCommandsForRecord(cl::sycl::detail::MemObjRecord*) [{crtstuff.c} {0}]	0.03	0.03	1	0
[SAMPLE] cl::sycl::detail::LeavesCollection::push_back(cl::sycl::detail::Command*) [{crtstuff.c} {0}]	0.03	0.03	1	0
[SAMPLE] cl::sycl::detail::ExecCGCommand::enqueueImp() [{crtstuff.c} {0}]	0.03	0.03	1	0
[SAMPLE] cl::sycl::detail::ExecCGCommand::SetKernelParamsAndLaunch(cl::sycl::detail::CGExecKernel*, _pi_kernel*, cl::sycl::detail::NDRDes	0.03	0.03	1	0
[SAMPLE] cl::sycl::detail::Command::addDep(cl::sycl::detail::DepDesc) [{crtstuff.c} {0}]	0.03	0.03	1	0
[SAMPLE] _pi_device::getAvailableCommandList(_pi_queue*, _ze_command_list_handle_t**, _ze_fence_handle_t**) [{crtstuff.c} {0}]	0.03	0.03	1	0
[SAMPLE] __gnu_cxx::__atomic_add(int volatile*, int) [{/usr/lib/gcc/x86_64-linux-gnu/9/../../../include/c++/9/ext/atomicity.h} {53}]	0.03	0.03	1	0
[SAMPLE] UNRESOLVED UNKNOWN	0.06	0.06	2	0
[SAMPLE] UNRESOLVED /usr/lib/x86_64-linux-gnu/libze_intel_gpu.so.1.0.18513	0.509	0.509	17	0
[SAMPLE] UNRESOLVED /usr/lib/x86_64-linux-gnu/libstdc++.so.6.0.28	0.03	0.03	1	0
[SAMPLE] UNRESOLVED /usr/lib/x86_64-linux-gnu/libpthread-2.31.so	0.06	0.06	2	0
[SAMPLE] UNRESOLVED /usr/lib/x86_64-linux-gnu/libigc.so.1.0.5585	0.18	0.18	6	0
[SAMPLE] UNRESOLVED /usr/lib/x86_64-linux-gnu/libc-2.31.so	20.852	20.852	669	0
[SAMPLE] UNRESOLVED /usr/lib/x86_64-linux-gnu/ld-2.31.so	0.15	0.15	5	0
[SAMPLE] UNRESOLVED /home/shende/tau2/x86_64/lib/libTAUsh-level_zero-pthread.so	0.479	0.479	15	0
[SAMPLE] Initialize(float*, float*, float*, unsigned long, unsigned long, unsigned long) [{/home/users/sameer/samples/iso3dfd_dpcpp/sr	0.03	0.03	1	0

TAU: ParaProf: Statistics for: node 0, thread 2 - iso3dfd_dpcpp.ppk

Name	Exclusive TAUGPU_TIME	Inclusive TAUGPU_TIME	Calls	Child Calls
.TAU application	2.738	22.592	1	290,467
zeCommandQueueExecuteCommandLists	19.073	19.073	10,002	0
zeModuleCreate	0.272	0.272	1	0
zeCommandListReset	0.165	0.165	10,002	0
zeEventHostSynchronize	0.118	0.118	22	0
zeCommandListAppendLaunchKernel	0.073	0.073	10,000	0
zeKernelSetArgumentValue [THROTTLED]	0.043	0.043	100,001	0
zeFenceQueryStatus [THROTTLED]	0.03	0.03	100,001	0
zeMemAllocHost	0.019	0.019	4	0
zeKernelSetGroupSize	0.012	0.012	10,000	0
zeCommandListClose	0.011	0.011	10,002	0
zeKernelGetProperties	0.01	0.01	10,000	0
zeEventCreate	0.007	0.007	10,002	0
zeMemFree	0.006	0.006	4	0
zeFenceReset	0.004	0.004	10,002	0
zeEventPoolDestroy	0.003	0.003	39	0
zeCommandListCreate	0.003	0.003	78	0
zeCommandListAppendMemoryCopy	0.002	0.002	2	0
zeEventPoolCreate	0.001	0.001	40	0
zeEventDestroy	0.001	0.001	10,002	0

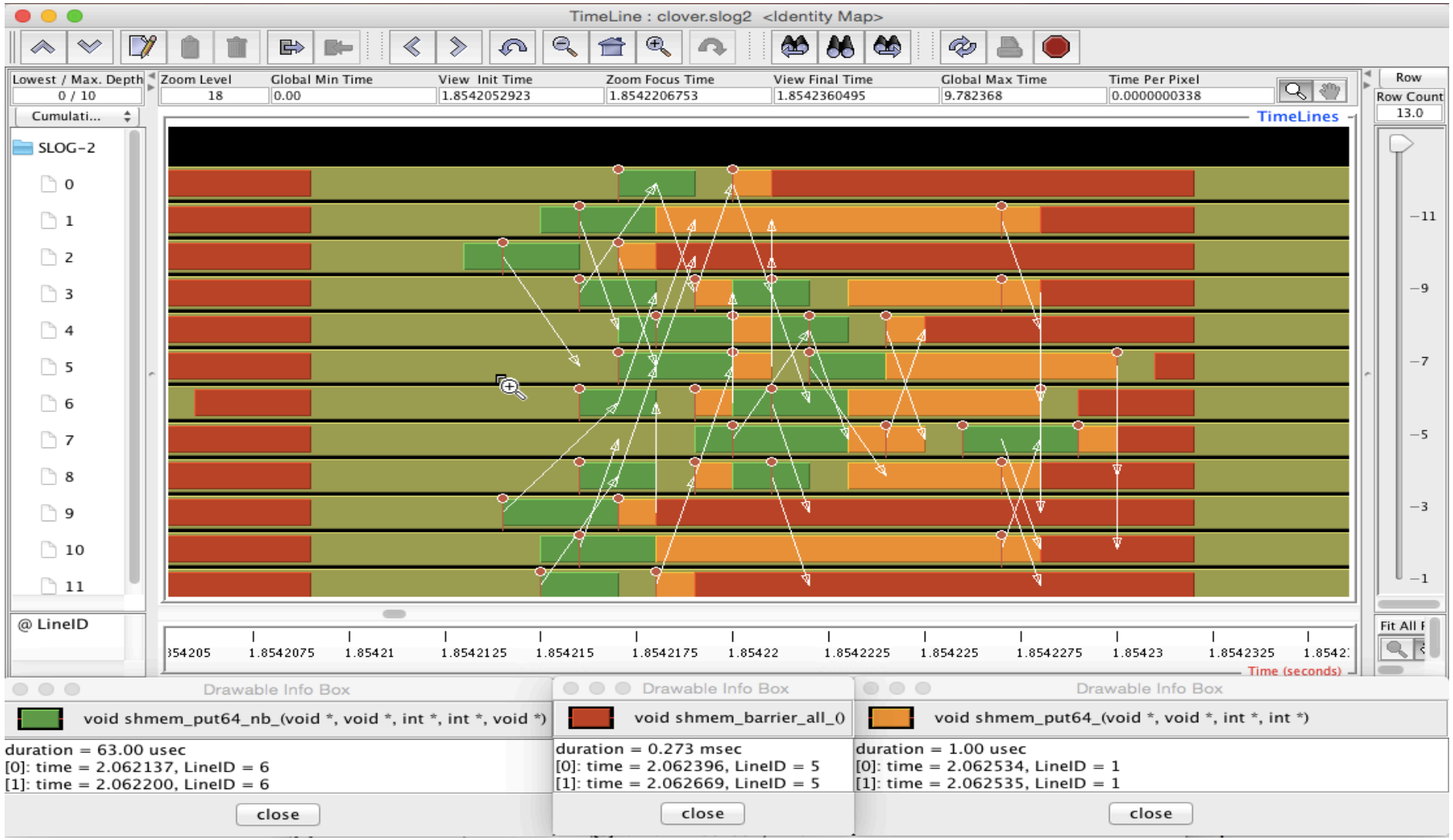
% tau_exec –T level_zero,serial –l0 ./a.out

TAU and Vampir [TU Dresden]: Intel oneAPI OpenCL (oneCCL)



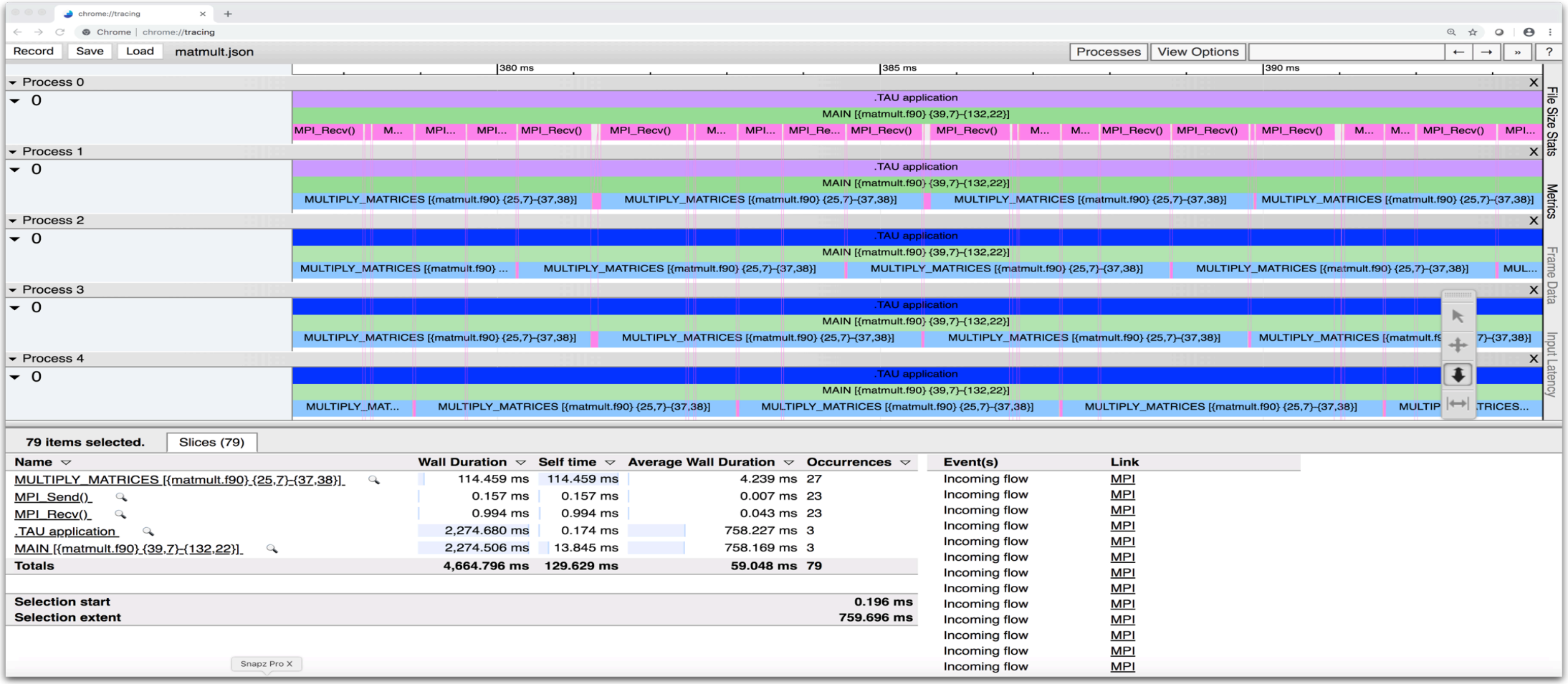

```
% export TAU_TRACE=1; export TAU_TRACE_FORMAT=otf2
% mpirun –np 4  tau_exec –T level_zero,serial –opencl ./a.out
```

Tracing: Jumpshot (ships with TAU)



EXASCALE COMPUTING PROJECT

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)

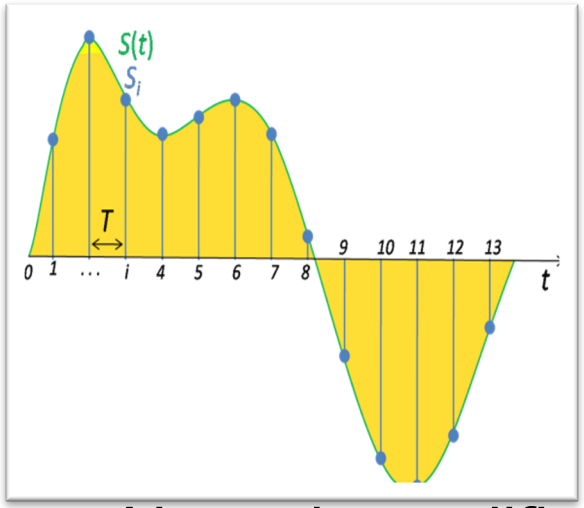
Performance Data Measurement

Direct via Probes

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

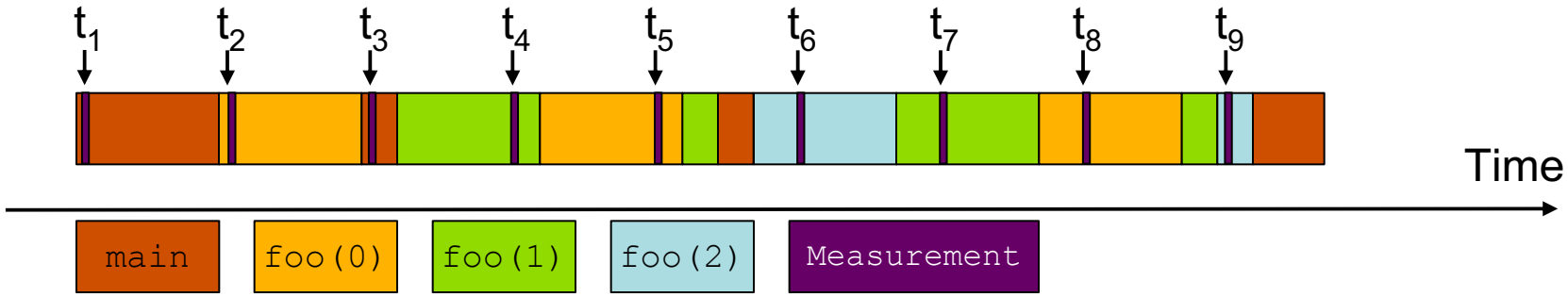
- Exact measurement
- Fine-grain control
- Calls inserted into code

Indirect via Sampling



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

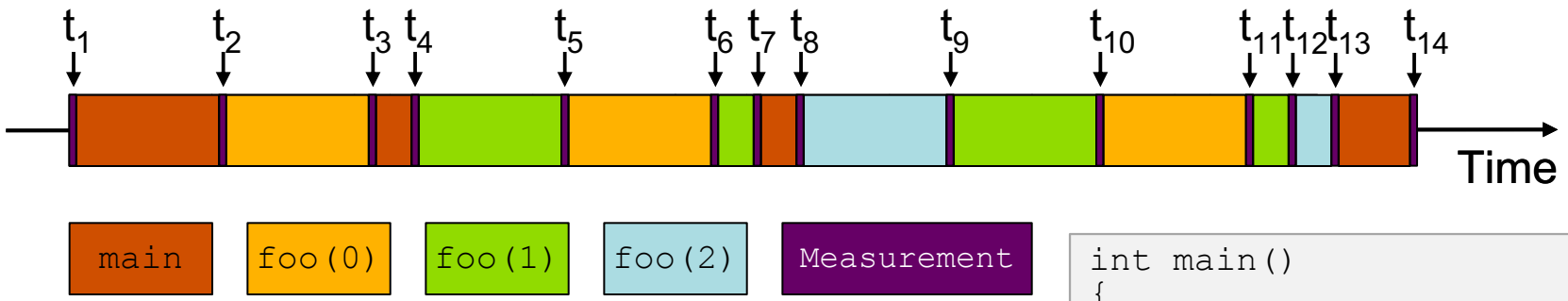
Sampling



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

```
int main()
{
  int i;

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

  return 0;
}

void foo(int i)
{

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

}
```

Instrumentation



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

```
int main()
{
  int i;
  Start("main");
  for (i=0; i < 3; i++)
    foo(i);
  Stop("main");
  return 0;
}

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

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

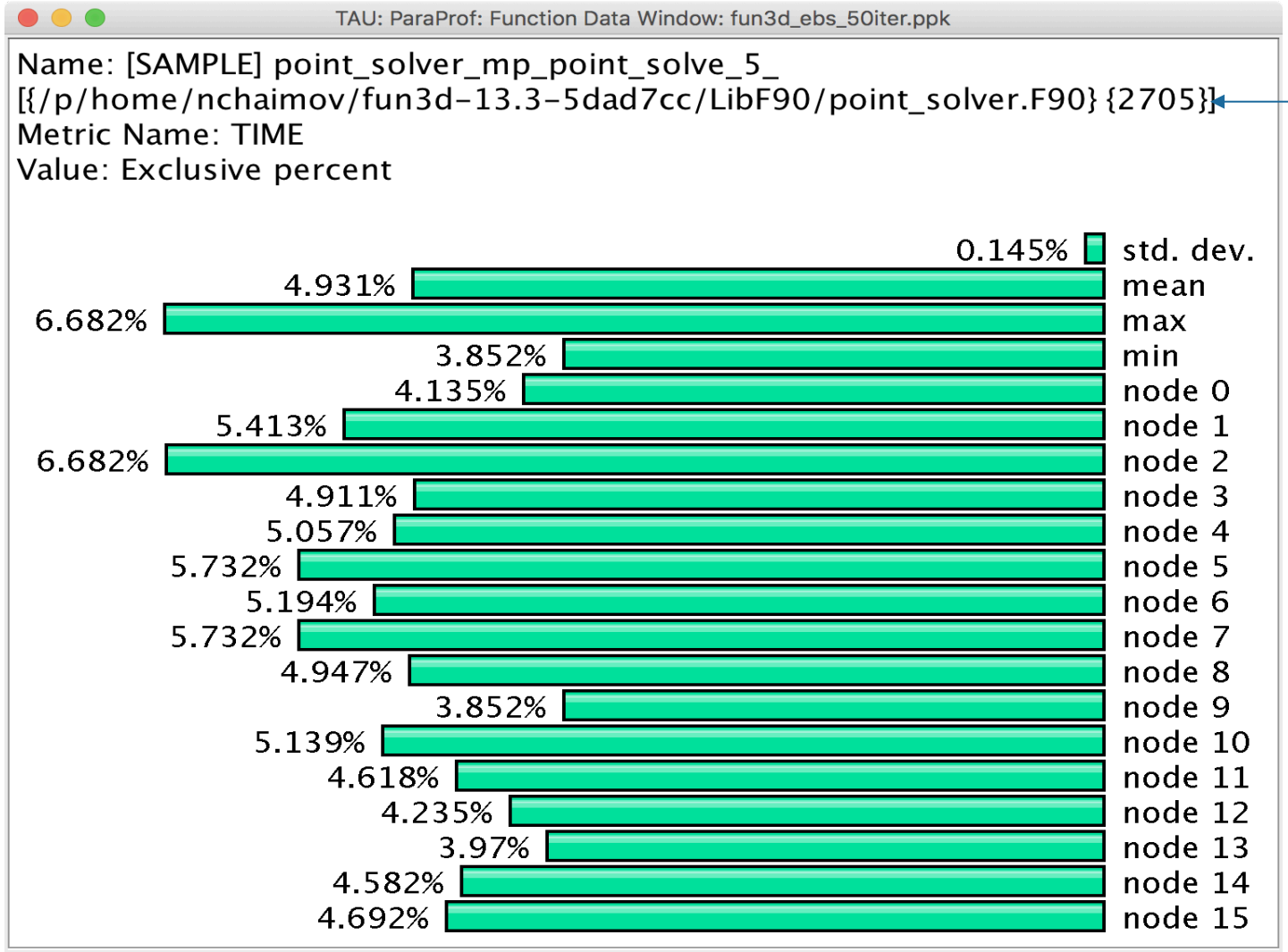
Memory allocation/deallocation routines

Wrapper library for an external package

No modification to the binary executable!

Enable other TAU options (communication matrix, OTF2, event-based sampling)

Event Based Sampling (EBS)



File: point_solver.F90
Line: 2705

Uninstrumented!

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

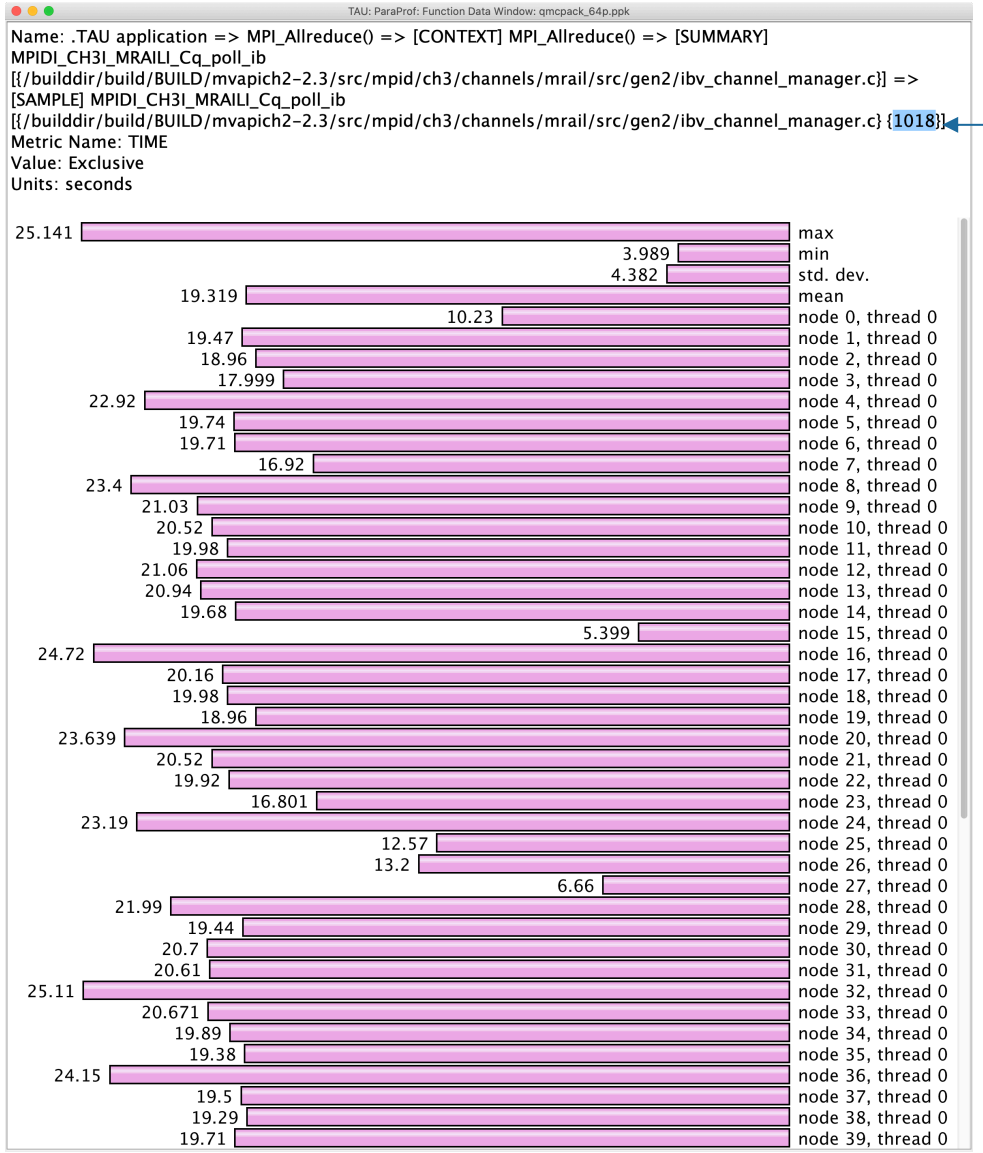
Event Based Sampling (EBS): QMCPack

TAU: ParaProf: Statistics for: node 0, thread 0 - qmcpack_64p.ppk

Name	Exclusive TIME	Inclusive TIME	Calls	Child Calls
.TAU application	979.071	1,066.528	1	493
[CONTEXT] .TAU application	0	975.151	32,481	0
MPI_Allreduce()	35.346	[illegible]	52	0
[CONTEXT] MPI_Allreduce()	0	35.37	1,179	0
[SUMMARY] MPIDI_CH3I_MRAILI_Get_next_vbuf [{/builddir/build/BUILD/mvapich2-2.3/src/m	16.109	16.109	537	0
[SUMMARY] MPIDI_CH3I_MRAILI_Cq_poll_ib [{/builddir/build/BUILD/mvapich2-2.3/src/mpid	10.71	10.71	357	0
[SAMPLE] MPIDI_CH3I_MRAILI_Cq_poll_ib [{/builddir/build/BUILD/mvapich2-2.3/src/mpid/	10.23	10.23	341	0
[SAMPLE] MPIDI_CH3I_MRAILI_Cq_poll_ib [{/builddir/build/BUILD/mvapich2-2.3/src/mpid/	0.27	0.27	9	0
[SAMPLE] MPIDI_CH3I_MRAILI_Cq_poll_ib [{/builddir/build/BUILD/mvapich2-2.3/src/mpid/	0.09	0.09	3	0
[SAMPLE] MPIDI_CH3I_MRAILI_Cq_poll_ib [{/builddir/build/BUILD/mvapich2-2.3/src/mpid/	0.03	0.03	1	0
[SAMPLE] MPIDI_CH3I_MRAILI_Cq_poll_ib [{/builddir/build/BUILD/mvapich2-2.3/src/mpid/	0.03	0.03	1	0
[SAMPLE] MPIDI_CH3I_MRAILI_Cq_poll_ib [{/builddir/build/BUILD/mvapich2-2.3/src/mpid/	0.03	0.03	1	0
[SAMPLE] MPIDI_CH3I_MRAILI_Cq_poll_ib [{/builddir/build/BUILD/mvapich2-2.3/src/mpid/	0.03	0.03	1	0
[SUMMARY] MPIDI_CH3I_SMP_read_progress [{/builddir/build/BUILD/mvapich2-2.3/src/mpi	1.98	1.98	66	0
[SUMMARY] MPIDI_CH3I_SMP_pull_header [{/builddir/build/BUILD/mvapich2-2.3/src/mpid/	1.77	1.77	59	0
[SAMPLE] GetSeqNumVbuf [{/builddir/build/BUILD/mvapich2-2.3/src/mpid/ch3/channels/m	1.23	1.23	41	0
[SAMPLE] UNRESOLVED /usr/lib64/libmlx5.so.1.0.0	1.05	1.05	35	0
[SAMPLE] ibv_poll_cq [{/usr/include/infiniband/verbs.h} {2456}]	0.96	0.96	32	0
[SUMMARY] MPIDI_CH3I_Progress [{/builddir/build/BUILD/mvapich2-2.3/src/mpid/ch3/cha	0.6	0.6	20	0
[SUMMARY] MPIDI_CH3I_MRAILI_Test_pkt [{/builddir/build/BUILD/mvapich2-2.3/src/mpid/c	0.24	0.24	8	0
[SUMMARY] MPIDU_Sched_progress [{/builddir/build/BUILD/mvapich2-2.3/src/mpid/comm	0.21	0.21	7	0
[SAMPLE] pthread_spin_lock [{/usr/lib64/libpthread-2.17.so} {0}]	0.18	0.18	6	0
[SUMMARY] MPIDI_CH3I_read_progress [{/builddir/build/BUILD/mvapich2-2.3/src/mpid/ch3	0.15	0.15	5	0
[SUMMARY] MPIDU_Sched_progress_state [{/builddir/build/BUILD/mvapich2-2.3/src/mpid/	0.09	0.09	3	0
[SUMMARY] MPIDI_CH3I_SHMEM_COLL_GetShmemBuf [{/builddir/build/BUILD/mvapich2-2.3	0.09	0.09	3	0
MPI_Reduce()	31.996	31.996	9	0
MPI_Gatherv()	9.433	9.433	230	0
MPI_Bcast()	5.452	5.452	71	0
MPI_Init()	2.356	2.356	1	2
MPI_Finalize()	1.333	1.351	1	4

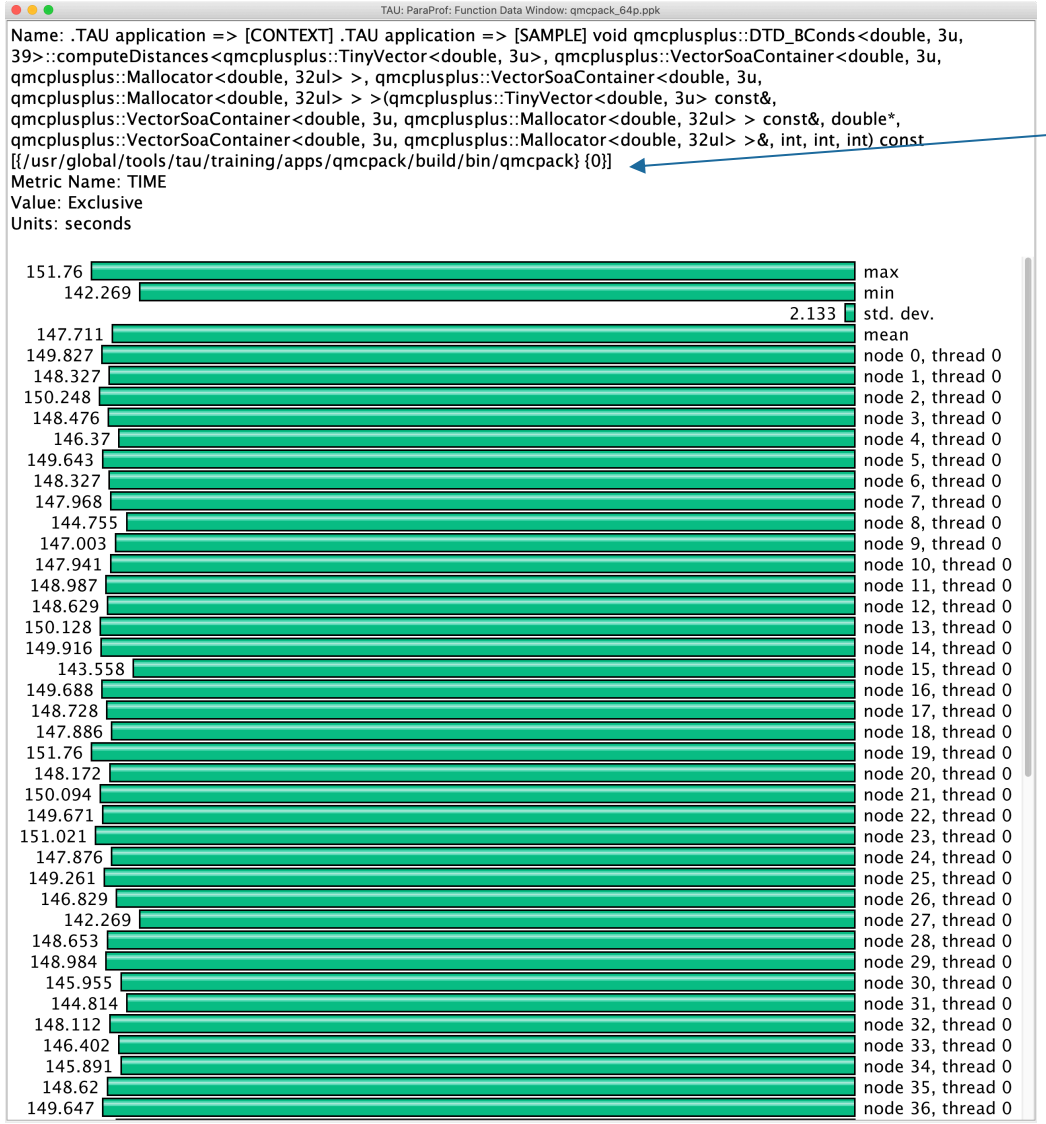
Ran for 1066.528 seconds. Outside of MPI calls, TAU can explain 975.151 seconds out of 979.071 seconds of exclusive time using EBS!

Event Based Sampling (EBS) shows statement level information



File: ibv_channel_manager.c
Line: 1018

Event Based Sampling without symbol information (-g): QMCPack



Line number is 0!

~147.71 seconds are spent in Qmcplusplus:DTD_Bconds:: computeDistances method

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
- *Intel® OneAPI*
  - Level Zero
  - Track time spent in kernels executing on GPU
  - Track time spent in OneAPI runtime calls
- *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 + pthread*
  - PMPI + pthread interfaces
- *MPI + Intel ® oneAPI DPC++/SYCL*
- PMPI + Level Zero interfaces
- *OpenCL + Python*
  - OpenCL + Python instrumentation interfaces
- *Kokkos + OpenMP*
  - Kokkos profiling API + OMPT to transparently track events
- *Kokkos + pthread + MPI*
  - Kokkos + pthread wrapper interposition library + PMPI layer
- *MPI + OpenCL*
  - PMPI + OpenCL profiling interfaces

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, DPC++/SYCL
- Creates a problem for performance evaluation tools
- Gap: performance data and higher-level abstractions
- Solution: Kokkos profiling API for mapping performance data
- https://kokkos.org Sandia National Laboratories, NM

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

TAU Execution Command (tau_exec)

Uninstrumented execution

```
% mpirun -np 256  ./a.out
```

Track GPU operations

```
% mpirun –np 256  tau_exec –l0      ./a.out
% mpirun –np 256 tau_exec –T serial,cupti –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_resolution=<file | function | line>
```

Intel Level Zero (TigerLake Gen12LP integrated CPUs or DG1)

TAU: ParaProf: Statistics for: node 0, thread 0 - ze_gemm_4096.ppk

Name	Exclusive TAUGPU_T...	Inclusive TAUGPU_TI...	Calls	Child Calls
.TAU application	117,876	30,283,630	1	256
zeCommandQueueSynchronize	29,877,963	29,877,963	4	0
[CONTEXT] zeCommandQueueSynchronize	0	29,905,688	997	0
[SAMPLE] __GI___sched_yield [{/lib64/libc-2.26.so}	25,765,719	25,765,719	859	0
[SAMPLE] UNRESOLVED /soft/libraries/intel-level-z	4,139,969	4,139,969	138	0
zeCommandQueueExecuteCommandLists	186,203	186,203	4	0
zeModuleCreate	98,896	98,896	1	0
zeCommandListAppendMemoryCopy	1,410	1,410	12	0
zeCommandQueueDestroy	321	321	4	0
zeDriverAllocDeviceMem	137	137	12	0
zeEventPoolDestroy	128	128	20	0
zeDriverFreeMem	96	96	12	0
zeCommandListCreate	89	89	4	0
zeCommandQueueCreate	82	82	4	0
zeCommandListDestroy	71	71	4	0
zeKernelSetArgumentValue	43	43	16	0
zeDeviceGetProperties	38	38	26	0
zeCommandListClose	35	35	4	0
zeEventCreate	30	30	4	0
zeEventDestroy	30	30	24	0
zeEventGetTimestamp	28	28	48	0
pthread_create	26	26	1	0
zeEventPoolCreate	20	20	4	0
zeKernelDestroy	20	20	1	0
zeModuleDestroy	17	17	1	0
zeCommandListAppendLaunchKernel	15	15	4	0
zeCommandListAppendBarrier	13	13	8	0
zeKernelSuggestGroupSize	12	12	4	0
zeEventQueryStatus	11	11	20	0
zeKernelCreate	11	11	1	0
zeKernelSetGroupSize	5	5	4	0
zeDeviceGet	2	2	2	0
zeInit	2	2	1	0
zeDriverGet	0	0	2	0

Units: microseconds

TAU: ParaProf: Statistics for: node 0, thread 2 - ze_gemm_4096.ppk

Name	Exclusive TAU...	Inclusive TAUG...	Calls	Child Calls
.TAU application	0.131	29.88	1	24
<Barrier>	0	0	8	0
<MemoryCopy>	0.049	0.049	12	0
GEMM	29.7	29.7	4	0

Units: seconds

Time spent in GEMM kernel

% mpirun –np 64 tau_exec –l0 ./a.out

MPI Tools Interface: PVARs and CVARs

TAU: ParaProf: Context Events for: node 2, thread 0 - comb_mpit_ebs.ppk

Name	Total	NumSamples	MaxValue	MinValue	MeanValue	Std. Dev.
Message size for broadcast	4	1	4	4	4	0
Message size for reduce	608	20	48	8	30.4	15.513
mpit_progress_poll (CH3 RDMA progress engine polling count)	474,499	1	474,499	474,499	474,499	0
mv2_coll_allgather_bytes_recv (Number of bytes recv by default algorithm of allgathe	612	1	612	612	612	0
mv2_coll_allgather_bytes_send (Number of bytes send by default algorithm of allgath	252	1	252	252	252	0
mv2_coll_allgather_count_recv (Count of messages recv by default algorithm of allgat	21	1	21	21	21	0
mv2_coll_allgather_count_send (Count of messages send by default algorithm of allga	21	1	21	21	21	0
mv2_coll_allgather_rd (Number of times recursive doubling Allgather was invoked)	7	1	7	7	7	0
mv2_coll_allgather_rd_bytes_recv (Number of bytes recv by rd algorithm of allgather	612	1	612	612	612	0
mv2_coll_allgather_rd_bytes_send (Number of bytes send by rd algorithm of allgather	252	1	252	252	252	0
mv2_coll_allgather_rd_count_recv (Count of messages recv by rd algorithm of allgath	21	1	21	21	21	0
mv2_coll_allgather_rd_count_send (Count of messages send by rd algorithm of allgat	21	1	21	21	21	0
mv2_coll_allreduce_2lvl (Number of times MV2 two-level allreduce algorithm was invo	5	1	5	5	5	0
mv2_coll_allreduce_bytes_recv (Number of bytes recv by allreduce collective)	7,080	1	7,080	7,080	7,080	0
mv2_coll_allreduce_bytes_send (Number of bytes send by allreduce collective)	7,080	1	7,080	7,080	7,080	0
mv2_coll_allreduce_count_recv (Count of messages recv by allreduce collective)	42	1	42	42	42	0
mv2_coll_allreduce_count_send (Count of messages send by allreduce collective)	42	1	42	42	42	0
mv2_coll_allreduce_pt2pt_rd_bytes_recv (Number of bytes recv by pt2pt rd algorithm	7,080	1	7,080	7,080	7,080	0
mv2_coll_allreduce_pt2pt_rd_bytes_send (Number of bytes send by pt2pt rd algorithm	7,080	1	7,080	7,080	7,080	0
mv2_coll_allreduce_pt2pt_rd_count_recv (Count of messages recv by pt2pt rd algorith	42	1	42	42	42	0
mv2_coll_allreduce_pt2pt_rd_count_send (Count of messages send by pt2pt rd algorit	42	1	42	42	42	0
mv2_coll_allreduce_shm_intra (Number of times MV2 shm intra allreduce algorithm w	3	1	3	3	3	0
mv2_coll_allreduce_shm_rd (Number of times MV2 shm rd allreduce algorithm was in	14	1	14	14	14	0
mv2_coll_allreduce_subcomm (Number of times MV2 allreduce was invoked at a sub-	10	1	10	10	10	0
mv2_coll_barrier_count_recv (Count of messages recv by barrier collective)	24	1	24	24	24	0
mv2_coll_barrier_count_send (Count of messages send by barrier collective)	24	1	24	24	24	0
mv2_coll_barrier_pairwise (Number of times pairwise barrier was invoked)	8	1	8	8	8	0
mv2_coll_barrier_pairwise_count_recv (Count of messages recv by pairwise algorithm	24	1	24	24	24	0
mv2_coll_barrier_pairwise_count_send (Count of messages send by pairwise algorithm	24	1	24	24	24	0

```
% export TAU_TRACK_MPI_T_PVARS=1
% mpirun –np 64 tau_exec –T mpit ./a.out
```

MPI Tools Interface: Control Variables (CVARs)

TAU: ParaProf Manager

- Applications
  - Standard Applications
    - Default App
      - Default Exp
        - comb_mpit_ebs.ppk
          - TIME

TrialField	Value
Name	comb_mpit_ebs.ppk
Application ID	0
Experiment ID	0
Trial ID	0
CPU Cores	24
CPU MHz	2000.000
CPU Type	AMD EPYC 7401 24-Core Processor
CPU Vendor	AuthenticAMD
CWD	/usr/global/tools/tau/training/apps/COMB_LLNL/Comb/build_lc_toss3_gcc_8_3_1/bin
Cache Size	512 KB
Command Line	./comb -comm post_recv wait_all -comm post_send wait_all -comm wait_recv wait_all -comm wait_send wait_all 200_200_20...
Ending Timestamp	1612559352834401
Executable	/usr/global/tools/tau/training/apps/COMB_LLNL/Comb/build_lc_toss3_gcc_8_3_1/bin/comb
File Type Index	0
File Type Name	ParaProf Packed Profile
Hostname	corona107
Local Time	2021-02-05T13:09:08-08:00
MPI Processor Name	corona107
MPI_T CVAR: MPIR_CVAR_ABORT_ON_LEAKED_HANDLES	If true, MPI will call MPI_Abort at MPI_Finalize if any MPI object handles have been leaked. For example, if MPI_Comm_dup is c...
MPI_T CVAR: MPIR_CVAR_ALLGATHERV_PIPELINE_MSG_SIZE	The smallest message size that will be used for the pipelined, large-message, ring algorithm in the MPI_Allgatherv implement...
MPI_T CVAR: MPIR_CVAR_ALLGATHER_COLLECTIVE_ALGORITHM	This CVAR selects proper collective algorithm for allgather operation.
MPI_T CVAR: MPIR_CVAR_ALLGATHER_LONG_MSG_SIZE	For MPI_Allgather and MPI_Allgatherv, the long message algorithm will be used if the send buffer size is >= this value (in byte...
MPI_T CVAR: MPIR_CVAR_ALLGATHER_SHORT_MSG_SIZE	For MPI_Allgather and MPI_Allgatherv, the short message algorithm will be used if the send buffer size is < this value (in bytes...
MPI_T CVAR: MPIR_CVAR_ALLREDUCE_COLLECTIVE_ALGORITHM	This CVAR selects proper collective algorithm for allreduce operation.
MPI_T CVAR: MPIR_CVAR_ALLREDUCE_SHORT_MSG_SIZE	the short message algorithm will be used if the send buffer size is <= this value (in bytes)
MPI_T CVAR: MPIR_CVAR_ALLTOALLV_COLLECTIVE_ALGORITHM	This CVAR selects proper collective algorithm for alltoallv operation.
MPI_T CVAR: MPIR_CVAR_ALLTOALL_COLLECTIVE_ALGORITHM	This CVAR selects proper collective algorithm for alltoall operation.
MPI_T CVAR: MPIR_CVAR_ALLTOALL_MEDIUM_MSG_SIZE	the medium message algorithm will be used if the per-destination message size (sendcount*size(sendtype)) is <= this value a...
MPI_T CVAR: MPIR_CVAR_ALLTOALL_SHORT_MSG_SIZE	the short message algorithm will be used if the per-destination message size (sendcount*size(sendtype)) is <= this value (See...
MPI_T CVAR: MPIR_CVAR_ALLTOALL_THROTTLE	max no. of irecvs/isends posted at a time in some alltoall algorithms. Setting it to 0 causes all irecvs/isends to be posted at o...
MPI_T CVAR: MPIR_CVAR_ASYNC_PROGRESS	If set to true, MPICH will initiate an additional thread to make asynchronous progress on all communication operations includi...
MPI_T CVAR: MPIR_CVAR_BCAST_COLLECTIVE_ALGORITHM	This CVAR selects proper collective algorithm for broadcast operation.
MPI_T CVAR: MPIR_CVAR_BCAST_LONG_MSG_SIZE	Let's define short messages as messages with size < MPIR_CVAR_BCAST_SHORT_MSG_SIZE, and medium messages as message...
MPI_T CVAR: MPIR_CVAR_BCAST_MIN_PROCS	Let's define short messages as messages with size < MPIR_CVAR_BCAST_SHORT_MSG_SIZE, and medium messages as message...
MPI_T CVAR: MPIR_CVAR_BCAST_SHORT_MSG_SIZE	Let's define short messages as messages with size < MPIR_CVAR_BCAST_SHORT_MSG_SIZE, and medium messages as message...
MPI_T CVAR: MPIR_CVAR_CH3_EAGER_MAX_MSG_SIZE	This cvar controls the message size at which CH3 switches from eager to rendezvous mode.
MPI_T CVAR: MPIR_CVAR_CH3_ENABLE_HCOLL	If true, enable HCOLL collectives.
MPI_T CVAR: MPIR_CVAR_CH3_INTERFACE_HOSTNAME	If non-NULL, this cvar specifies the IP address that other processes should use when connecting to this process. This cvar is m...
MPI_T CVAR: MPIR_CVAR_CH3_NOLOCAL	If true, force all processes to operate as though all processes are located on another node. For example, this disables shared ...

MPI Tools Interface: Performance Variables (PVARs)

TAU: ParaProf Manager

- Applications
  - Standard Applications
    - Default App
      - Default Exp
        - comb_mpit_ebs.ppk
          - TIME

TrialField	Value
MPI_T CVAR: MPIR_CVAR_USE_CUDA	This option enables the CUDA library.
MPI_T CVAR: MPIR_CVAR_USE_GPUDIRECT	This option enables use of GPUDIRECT in CUDA library.
MPI_T CVAR: MPIR_CVAR_USE_GPUDIRECT_GDRCOPY	This option enables use of GPUDIRECT_GDRCOPY in CUDA library.
MPI_T CVAR: MPIR_CVAR_USE_GPUDIRECT_GDRCOPY_LIMIT	This option sets limit on Gpudirect GDRCOPY in CUDA library.
MPI_T CVAR: MPIR_CVAR_USE_GPUDIRECT_LOOPBACK	This option enables use of GPUDIRECT_LOOPBACK in CUDA library.
MPI_T CVAR: MPIR_CVAR_USE_GPUDIRECT_LOOPBACK_LIMIT	This option sets limit on GPUDIRECT LOOPBACK in CUDA library.
MPI_T CVAR: MPIR_CVAR_USE_GPUDIRECT_RECEIVE_LIMIT	This option sets limit on MV2_USE_GPUDIRECT_RECEIVE_LIMIT in CUDA library.
MPI_T CVAR: MPIR_CVAR_USE_IWARP_MODE	This parameter enables the library to run in iWARP mode.
MPI_T CVAR: MPIR_CVAR_USE_MCAST	Set this to 1, to enable hardware multicast support in collective communication.
MPI_T CVAR: MPIR_CVAR_USE_RDMA_CM	This parameter enables the use of RDMA CM for establishing the connections.
MPI_T CVAR: MPIR_CVAR_USE_SHARED_MEM	Use shared memory for intra-node communication.
MPI_T PVAR[0]: mv2_pt2pt_mpid_send	bucket level counters for mpid send
MPI_T PVAR[100]: mv2_coll_timer_alltoall_rd	total time spent on the MV2 alltoall_rd algorithm
MPI_T PVAR[101]: mv2_coll_timer_alltoall_rd	total time spent on the MV2 alltoall_rd algorithm
MPI_T PVAR[102]: mv2_coll_timer_alltoall_sd	total time spent on the MV2 alltoall_sd algorithm
MPI_T PVAR[103]: mv2_coll_timer_alltoall_sd	total time spent on the MV2 alltoall_sd algorithm
MPI_T PVAR[104]: mv2_coll_timer_alltoall_pw	total time spent on the MV2 alltoall_pw algorithm
MPI_T PVAR[105]: mv2_coll_timer_alltoall_pw	total time spent on the MV2 alltoall_pw algorithm
MPI_T PVAR[106]: mv2_coll_alltoall_inplace	Number of times MV2 in-place alltoall algorithm was invoked
MPI_T PVAR[107]: mv2_coll_alltoall_bruck	Number of times MV2 brucks alltoall algorithm was invoked
MPI_T PVAR[108]: mv2_coll_alltoall_rd	Number of times MV2 recursive-doubling alltoall algorithm was invoked
MPI_T PVAR[109]: mv2_coll_alltoall_sd	Number of times MV2 scatter-destination alltoall algorithm was invoked
MPI_T PVAR[10]: mv2_smp_read_progress_poll_success	Unsuccessful CH3 SMP read progress engine polling count
MPI_T PVAR[110]: mv2_coll_alltoall_pw	Number of times MV2 pairwise alltoall algorithm was invoked
MPI_T PVAR[111]: mv2_coll_alltoall_inplace_bytes_send	Number of bytes send by inplace algorithm of alltoall collective
MPI_T PVAR[112]: mv2_coll_alltoall_bruck_bytes_send	Number of bytes send by bruck algorithm of alltoall collective
MPI_T PVAR[113]: mv2_coll_alltoall_sd_bytes_send	Number of bytes send by sd algorithm of alltoall collective
MPI_T PVAR[114]: mv2_coll_alltoall_pw_bytes_send	Number of bytes send by pw algorithm of alltoall collective
MPI_T PVAR[115]: mv2_coll_alltoall_inplace_bytes_recv	Number of bytes recv by inplace algorithm of alltoall collective
MPI_T PVAR[116]: mv2_coll_alltoall_bruck_bytes_recv	Number of bytes recv by bruck algorithm of alltoall collective
MPI_T PVAR[117]: mv2_coll_alltoall_sd_bytes_recv	Number of bytes recv by sd algorithm of alltoall collective
MPI_T PVAR[118]: mv2_coll_alltoall_pw_bytes_recv	Number of bytes recv by pw algorithm of alltoall collective
MPI_T PVAR[119]: mv2_coll_alltoall_inplace_count_send	Count of messages send by inplace algorithm of alltoall collective
MPI_T PVAR[11]: mv2_smp_write_progress_poll_success	Unsuccessful CH3 SMP write progress engine polling count
MPI_T PVAR[120]: mv2_coll_alltoall_bruck_count_send	Count of messages send by bruck algorithm of alltoall collective
MPI_T PVAR[121]: mv2_coll_alltoall_sd_count_send	Count of messages send by sd algorithm of alltoall collective
MPI_T PVAR[122]: mv2_coll_alltoall_pw_count_send	Count of messages send by pw algorithm of alltoall collective
MPI_T PVAR[123]: mv2_coll_alltoall_inplace_count_recv	Count of messages recv by inplace algorithm of alltoall collective
MPI_T PVAR[124]: mv2_coll_alltoall_bruck_count_recv	Count of messages recv by bruck algorithm of alltoall collective

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
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>)

Runtime Environment Variables

Environment Variable	Default	Description
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.

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
TAU_EVENT_THRESHOLD	0.5	Define a threshold value (e.g., .25 is 25%) to trigger marker events for min/max

TAU Performance System®

Hands-on exercises on Theta and ThetaGPU

EXASCALE COMPUTING PROJECT

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 (x86_64)

  Install Java 11.0.3:

  Download and install 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 &

- macOS (arm64, M1)

  http://tau.uoregon.edu/java_arm64.dmg

- Linux (http://tau.uoregon.edu/tau.tgz)

  ./configure; make install; export PATH=<taudir>/x86_64/bin:$PATH; paraprof app.ppk &

TAU: Quickstart Guide on Theta @ ALCF

Setup:

- `% module load tau(on Theta) % module use /soft/perftools/tau/modulefiles/thetagpu; module load tau (ThetaGPU)`

Profiling with an un-instrumented application:

- `MPI:` `% aprun -np 64 tau_exec -ebs ./a.out`
- `MPI+OpenMP with Intel 19.1+:` `% export TAU_OMPT_SUPPORT_LEVEL=full;`
  `% aprun -np 64 tau_exec -T ompt,v5 -ompt ./a.out`
- `Pthread:` `% aprun -np 64 tau_exec -T mpi,pthread -ebs ./a.out`
- `Python+MPI+Sampling:` `% aprun -np 64 tau_python -ebs ./a.py`
- `Python+MPI+OpenCL:` `% aprun -np 64 tau_python -opencl ./a.py`
- `DPC++/SYCL (no MPI):` `% tau_exec -T level_zero,serial -l0 ./foo`
- `CUDA+Sampling (no MPI):` `% tau_exec -T cupti,serial -cupti -ebs ./a.out`

Analysis: `% pprof -a -m | more;  % paraprof (GUI)`

Tracing:

- `Vampir: MPI:` `% export TAU_TRACE=1; export TAU_TRACE_FORMAT=otf2`
  `% mpirun -np 64 tau_exec ./a.out; vampir traces.otf2 &`
- `Chrome:` `% export TAU_TRACE=1; mpirun -np 64 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)`
- `Jumpshot:` `% export TAU_TRACE=1; mpirun -np 64 tau_exec ./a.out; tau_treemerge.pl;`
  `% tau2slog2 tau.trc tau.edf -o app.slog2; jumpshot app.slog2 &`

TAU Breakout Session – CoMD on Theta

Setup preferred program environment compilers

Default set Intel Compilers with Intel MPI. You must compile with **–dynamic –g**

```
% module load tau;
% tar zxf /soft/perftools/tau/workshop.tgz
% cd workshop/CoMD/src-openmp;
% make clean
% make
% cd ../bin; cat rompt.sh
In a second window:
% qsub -I -n 1 -A comp_perf_workshop -q comp_perf_workshop -t 50 -q debug-cache-quad

% cd workshop/CoMD/src-openmp/bin;
% ./rompt.sh
% paraprof --pack ex1.ppk
In the first window:
% paraprof ex1.ppk &
```

TAU Breakout Session – CoMD on Theta

Setup preferred program environment compilers

Default set Intel Compilers with Intel MPI. You must compile with **–dynamic –g**

```
% module load tau;
% tar zxf /soft/perftools/tau/workshop.tgz
% cd workshop/CoMD/src-openmp;
% make clean
% make
% cd ../bin; cat rompt.sh
In a second window:
% qsub -I -n 1 -A comp_perf_workshop -q comp_perf_workshop -t 50 -q debug-cache-quad

% cd workshop/CoMD/src-openmp/bin;
% ./rompt.sh
% paraprof --pack ex1.ppk
In the first window:
% paraprof ex1.ppk &
```

TAU Breakout Session – CoMD MPI on ThetaGPU

Setup preferred program environment compilers

```
% module load cobalt/cobalt-gpu; qsub -I -n 1 -A comp_perf_workshop -t 50
% module use /soft/perftools/tau/modulefiles/thetagpu; module load tau;
% tar zxf /soft/perftools/tau/workshop.tgz
% cd workshop/CoMD/src-mpi;
% make clean
% make
% cd ../bin; cat r.sh; cat rt.sh
% ./r.sh;
% ./rt.sh
% paraprof --pack comd_mpi.ppk
Bring ppk file to your desktop:
% paraprof comd_mpi.ppk &
```

TAU Breakout Session – CUDA on ThetaGPU

Setup preferred program environment compilers

```
% module load cobalt/cobalt-gpu; qsub -I -n 1 -A comp_perf_workshop -t 50
% module use /soft/perftools/tau/modulefiles/thetagpu; module load tau;
% tar zxf /soft/perftools/tau/workshop.tgz
% cd workshop/cuda;
% make clean
% make
% ./matmult
% cat ./rt
% ./rt
% pprof -a | more
% paraprof --pack cupti_ex.ppk
Bring ppk file to your desktop:
% paraprof cupti_ex.ppk &
```

TAU contacts and links

- GPU profiling and tracing: Sameer Shende sameer@cs.uoregon.edu
- E4S: Sameer Shende sameer@cs.uoregon.edu
- TAU Performance System®: http://tau.uoregon.edu
- E4S: https://e4s.io

Download TAU from U. Oregon

Tuning and Analysis Utilities

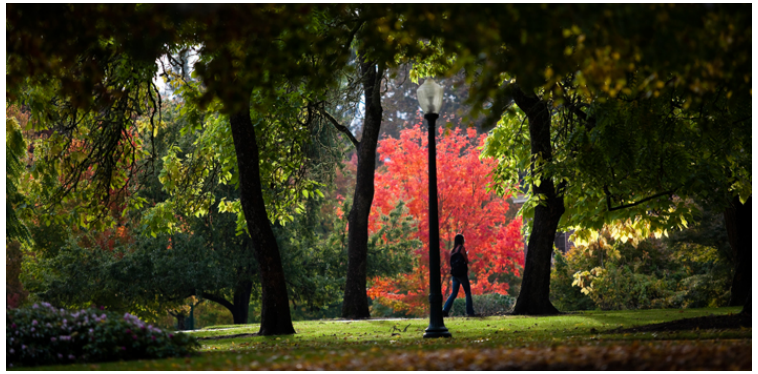
http://tau.uoregon.edu

https://e4s.io [TAU in Docker/Singularity containers]

for more information

Free download, open source, BSD license

Performance Research Laboratory, University of Oregon, Eugene



www.uoregon.edu

Support Acknowledgements

- US Department of Energy (DOE)
  - ANL
  - Office of Science contracts, ECP
  - SciDAC, LBL contracts
  - LLNL-LANL-SNL ASC/NNSA contract
  - Battelle, PNNL and ORNL contract
- Department of Defense (DoD)
  - PETTT, HPCMP
- National Science Foundation (NSF)
  - SI2-SSI, Glassbox
- NASA
- Intel
- CEA, France
- Partners:
  - University of Oregon
  - The Ohio State University
  - ParaTools, Inc.
  - University of Tennessee, Knoxville
  - T.U. Dresden, GWT
  - Jülich Supercomputing Center

Acknowledgment

"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."