TAU Performance System®

IWOCL 2022

Prof. Sameer Shende
Research Associate Professor and Director,
Performance Research Laboratory, OACISS, University of Oregon
President and Director, ParaTools, Inc. and ParaTools, SAS

http://tau.uoregon.edu/TAU_IWOCL22.pdf

Motivation and Challenges

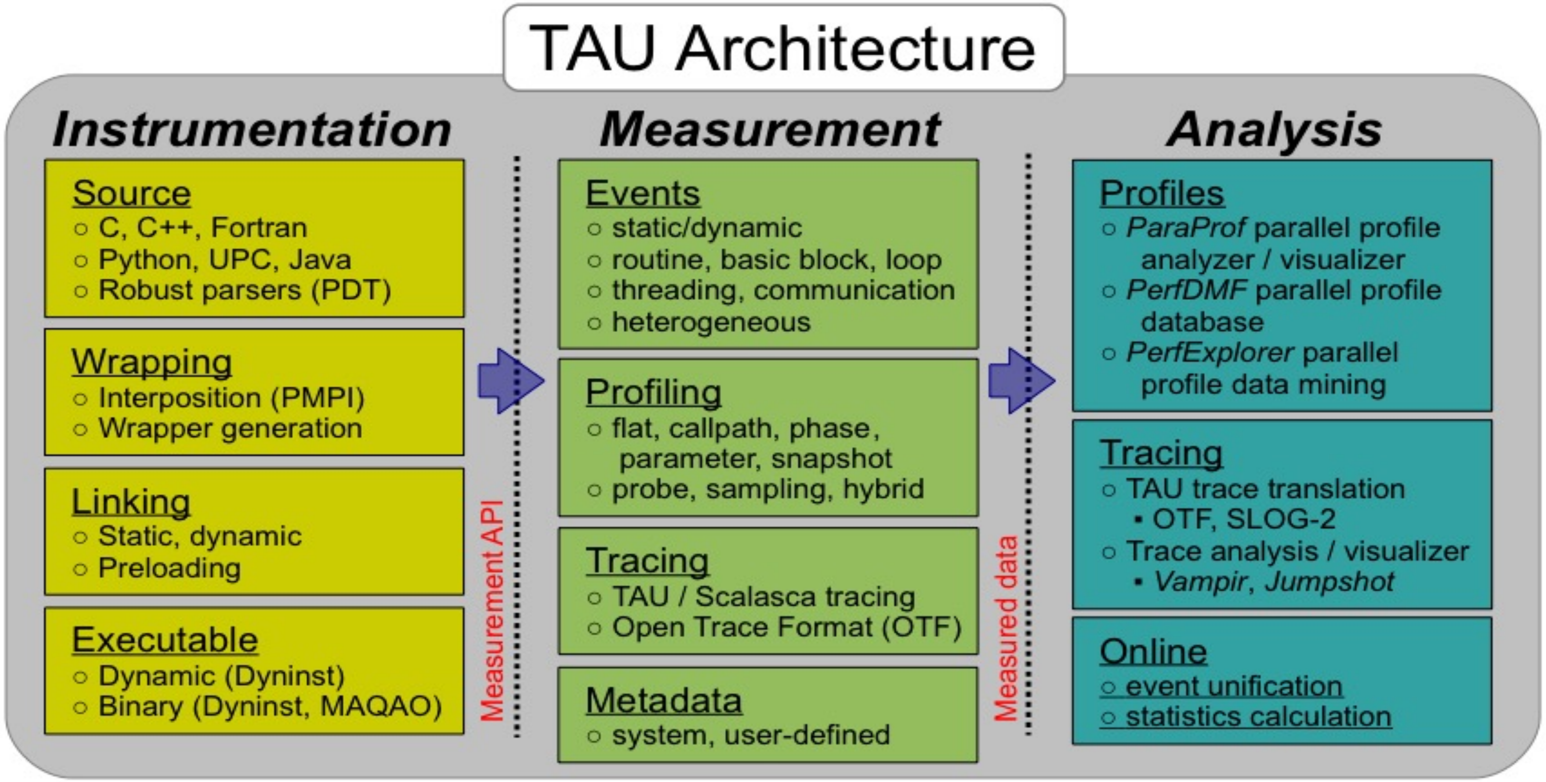
- With growing hardware complexity, it is getting harder to accurately measure and optimize the performance of our HPC and AI/ML workloads.
- TAU Performance System®:
  - Deliver a scalable, portable, performance evaluation toolkit for HPC and AI/ML workloads.
  - Supports OpenCL programming model across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

TAU Performance System®

Tuning and Analysis Utilities

Parallel performance framework and toolkit

- Aims to supports all HPC platforms, compilers, runtime system
- Provides portable instrumentation, measurement, analysis



TAU Architecture

Instrumentation

Source
- C, C++, Fortran
- Python, UPC, Java
- Robust parsers (PDT)

Wrapping
- Interposition (PMPI)
- Wrapper generation

Linking
- Static, dynamic
- Preloading

Executable
- Dynamic (Dyninst)
- Binary (Dyninst, MAQAO)

Measurement API

Measurement

Events
- static/dynamic
- routine, basic block, loop
- threading, communication
- heterogeneous

Profiling
- flat, callpath, phase, parameter, snapshot
- probe, sampling, hybrid

Tracing
- TAU / Scalasca tracing
- Open Trace Format (OTF)

Metadata
- system, user-defined

Measured data

Analysis

Profiles
- *ParaProf* parallel profile analyzer / visualizer
- *PerfDMF* parallel profile database
- *PerfExplorer* parallel profile data mining

Tracing
- TAU trace translation
  - OTF, SLOG-2
- Trace analysis / visualizer
  - *Vampir*, *Jumpshot*

Online
- event unification
- statistics calculation

TAU Performance System

Instrumentation

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

Measurement and analysis support

- MPI, OpenSHMEM, ARMCI, PGAS
- pthreads, OpenMP, OMPT interface, hybrid, other thread models
- GPU: OpenCL, DPC++/SYCL, ROCm, CUDA, OpenACC
- 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 long did it take to transfer data between host and device (GPU)?
- 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?

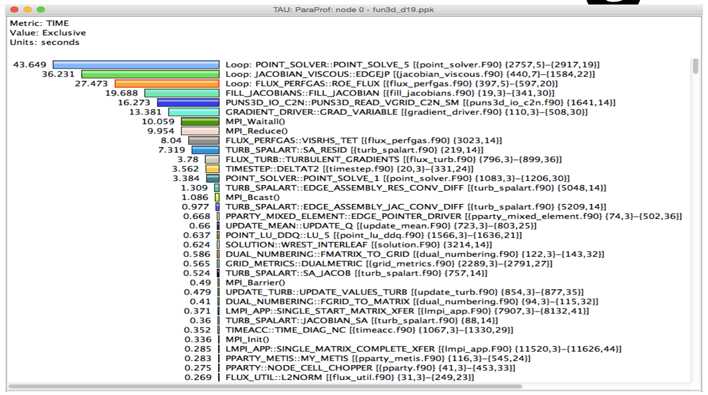
Instrumentation

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 **mpirun tau_exec ./app** with options.

Profiling and Tracing

Profiling

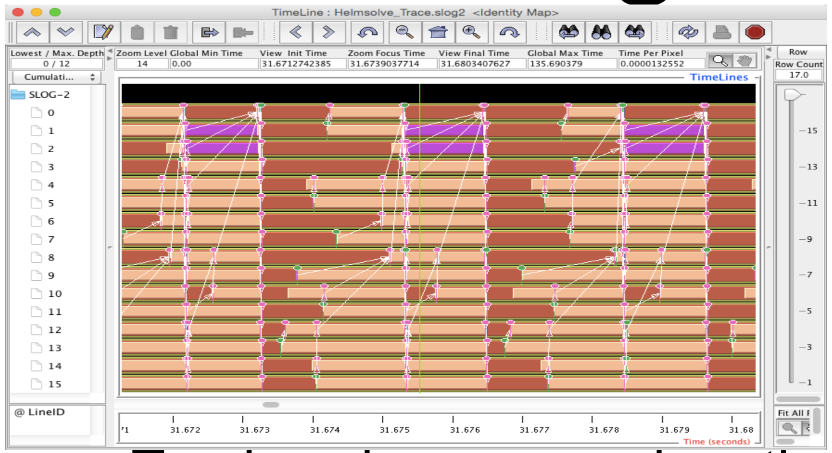


TAU: ParaProf: node 0 - fun3d_d19.ppk

```
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}]
```

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

Tracing



TimeLine : Helmsolve_Trace.slog2 <Identity Map>

Lowest / Max. Depth: 0 / 12
Zoom Level: 14
Global Min Time: 0.00
View Init Time: 31.6712742385
Zoom Focus Time: 31.6739037714
View Final Time: 31.6803407627
Global Max Time: 135.690379
Time Per Pixel: 0.0000132552
Row Count: 17.0

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

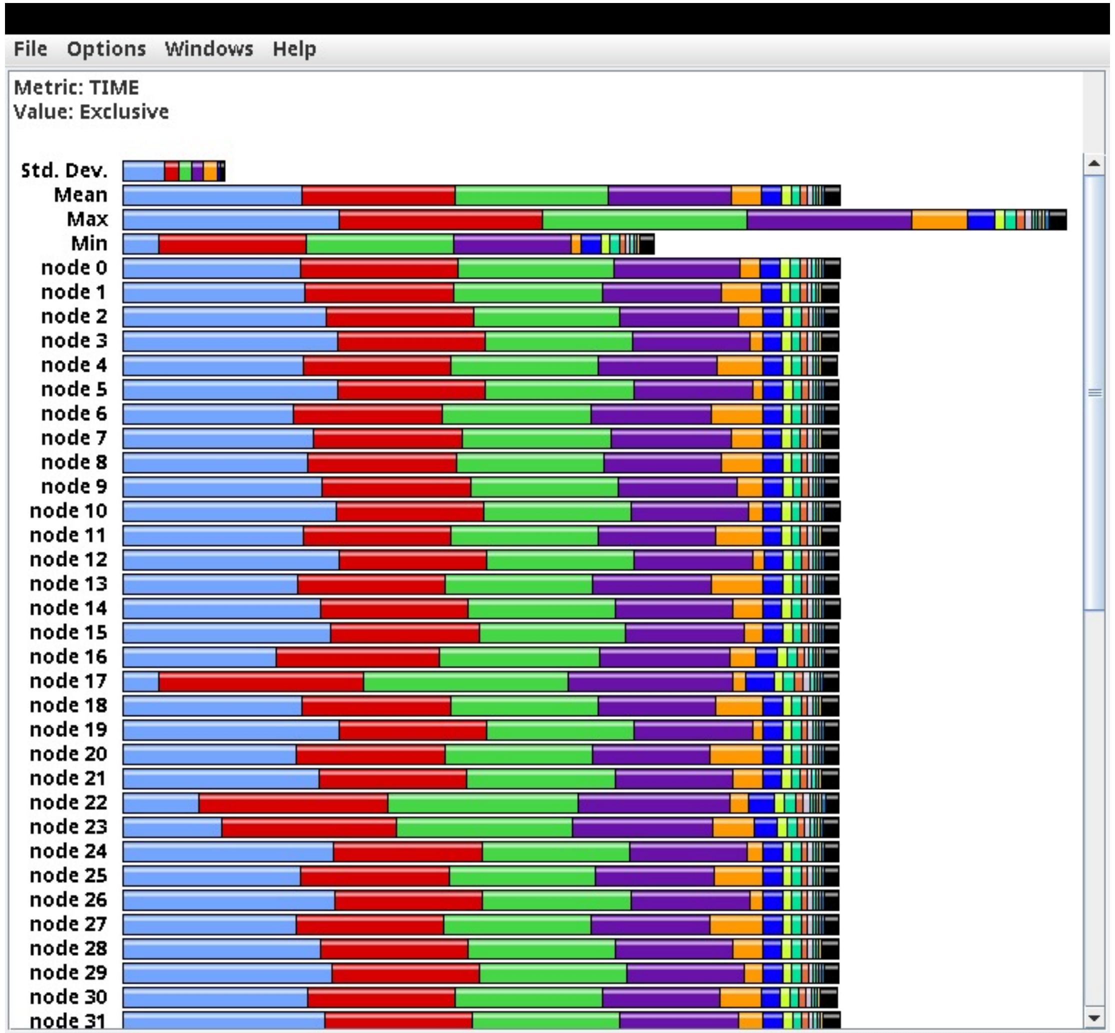
- Profiling and tracing

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

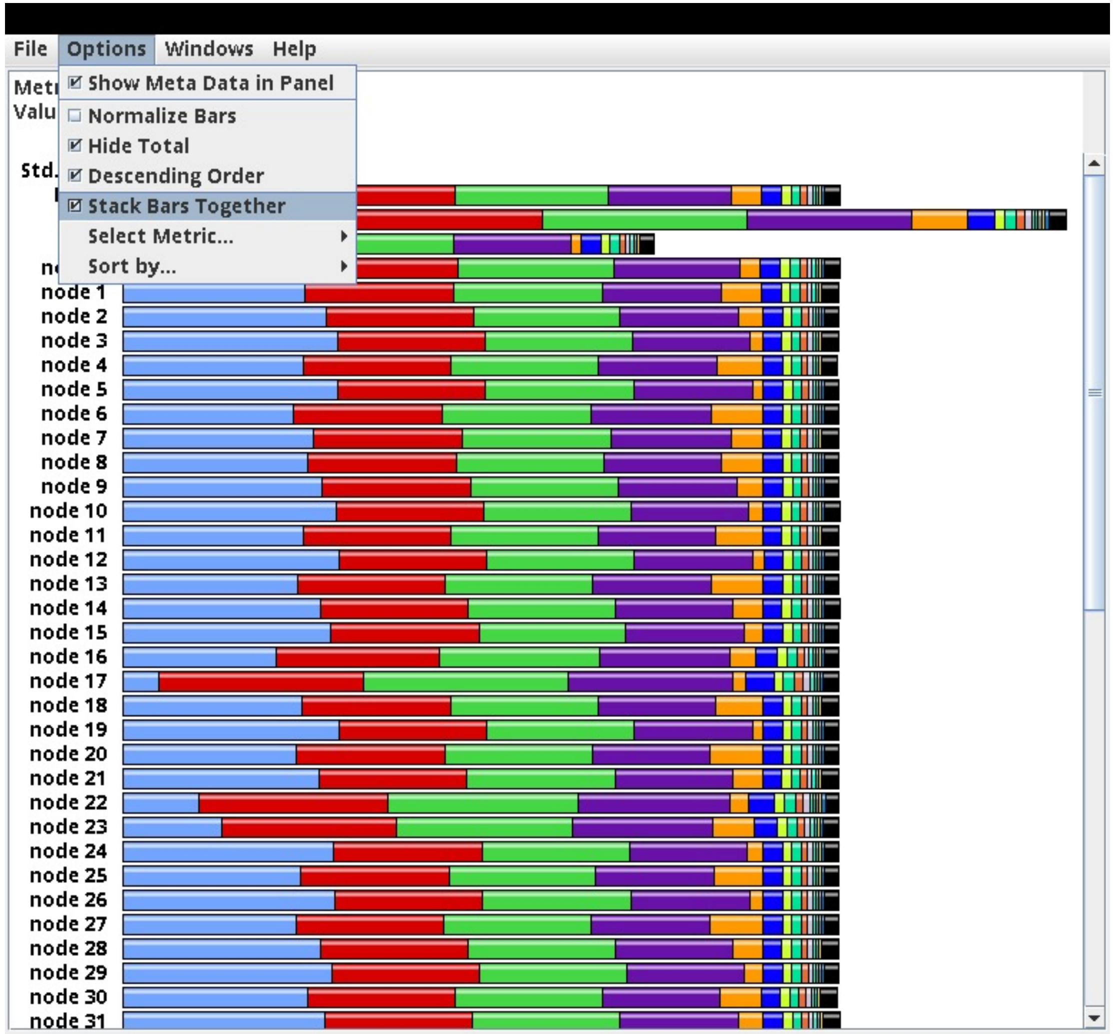
  Tracing shows you **when** the events take place on a timeline

ParaProf Profile Browser

```
% paraprof
```

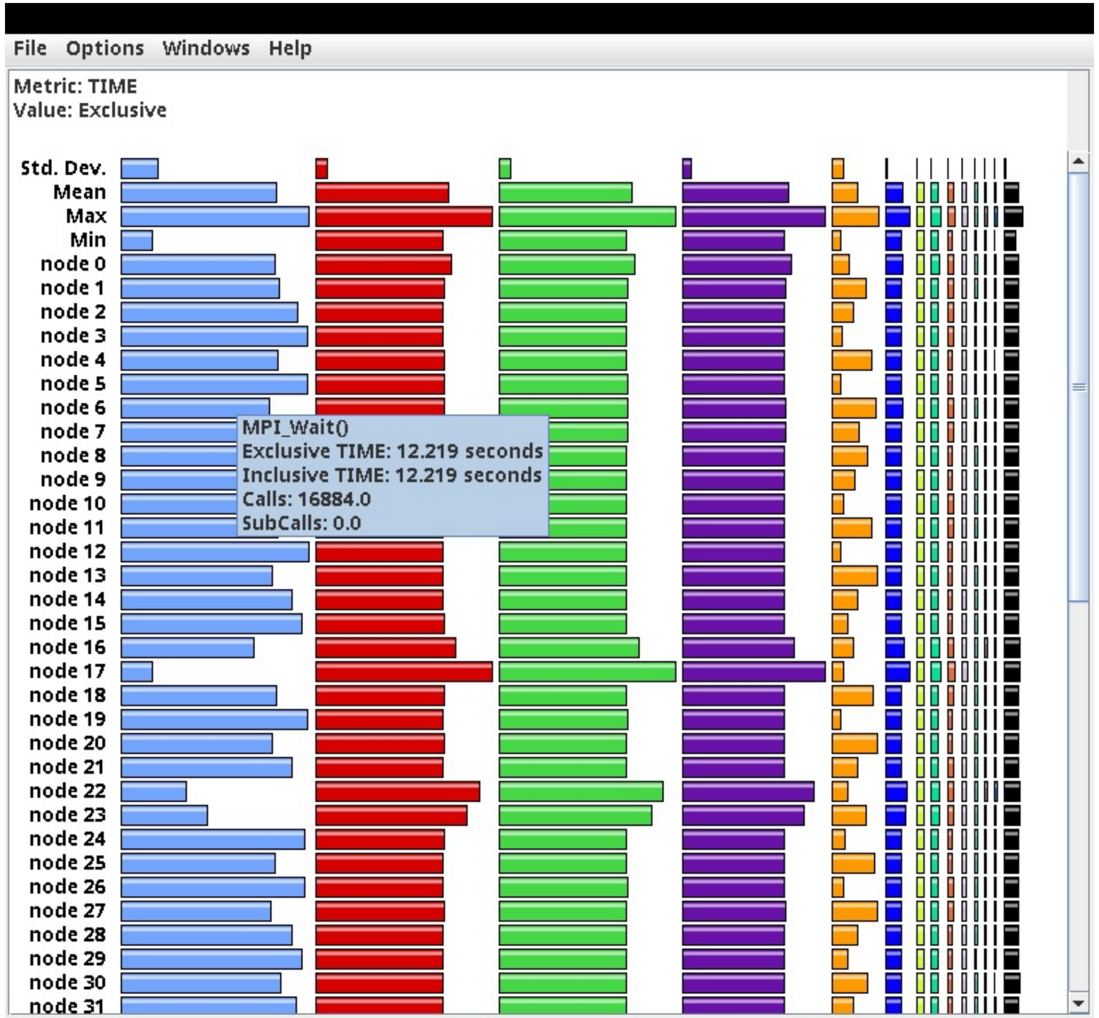


TAU’s ParaProf Profile Browser



- ☑ Show Meta Data in Panel
- ☐ Normalize Bars
- ☑ Hide Total
- ☑ Descending Order
- ☑ Stack Bars Together
- Select Metric...
- Sort by...

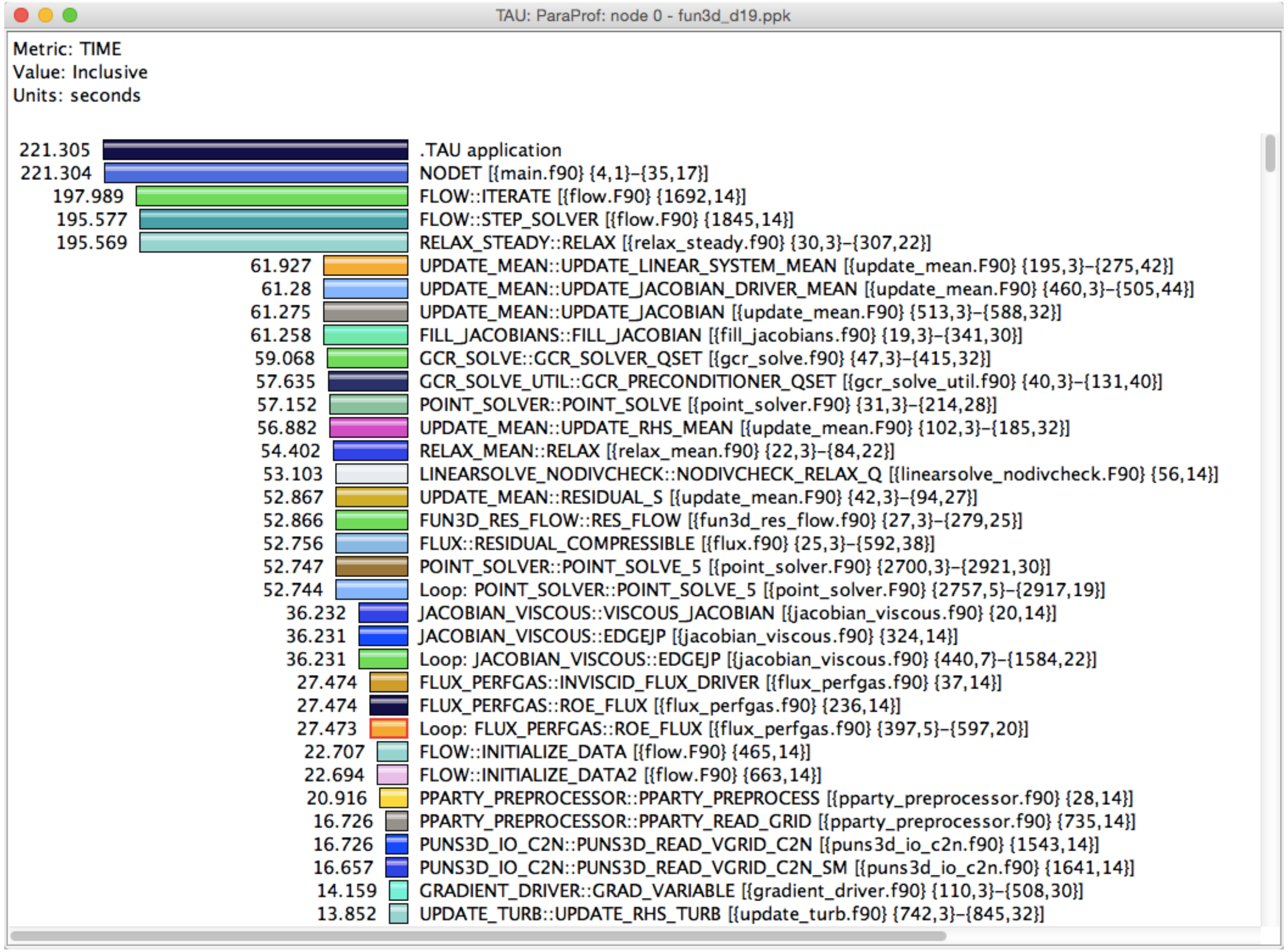
→



Metric: TIME
Value: Exclusive

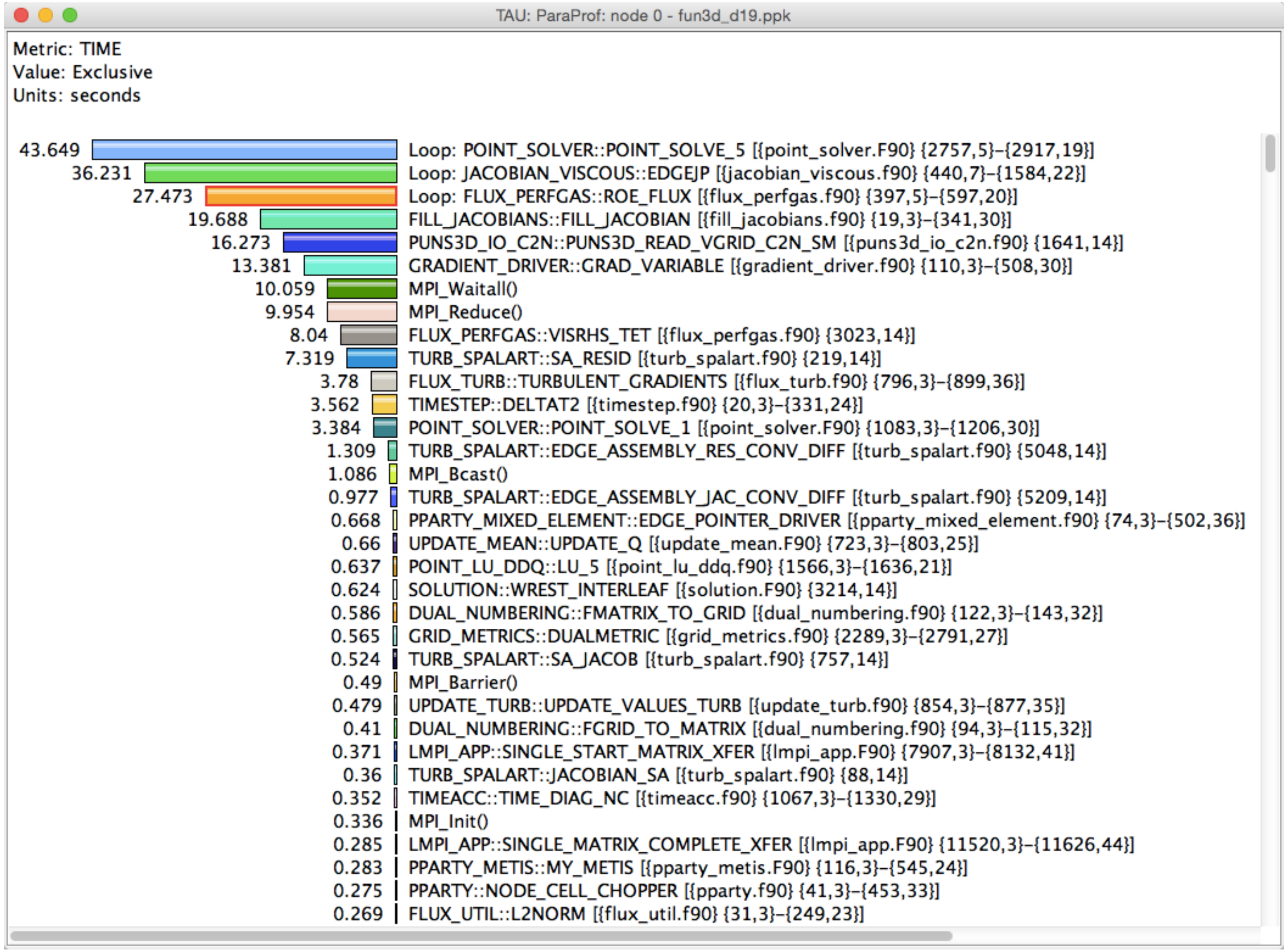
MPI_Wait()
Exclusive TIME: 12.219 seconds
Inclusive TIME: 12.219 seconds
Calls: 16884.0
SubCalls: 0.0

Inclusive Measurements



```
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}]
```

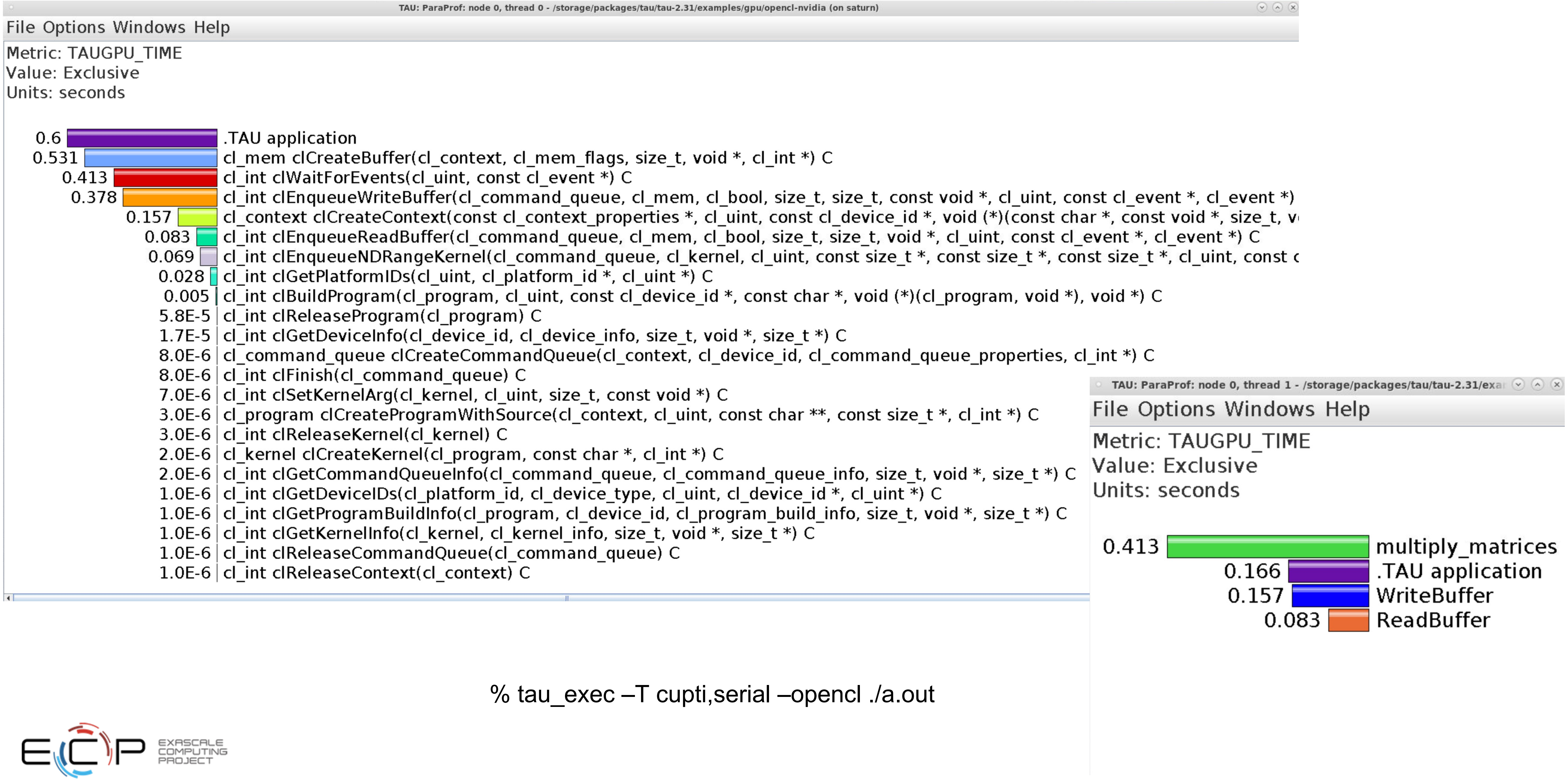
Exclusive Time



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}]
```

TAU: OpenCL profiling on NVIDIA A100 GPU

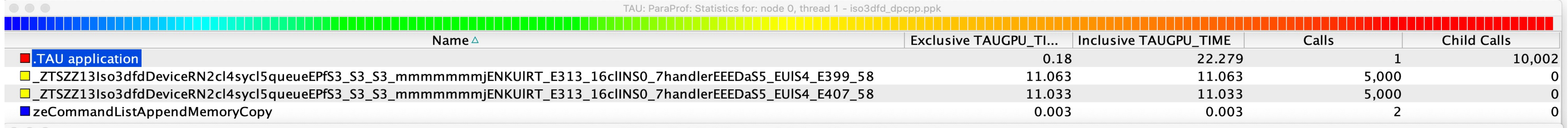

TAU: ParaProf: node 0, thread 0 - /storage/packages/tau/tau-2.31/examples/gpu/opencl-nvidia (on saturn)
File Options Windows Help
TAU: ParaProf: node 0, thread 1 - /storage/packages/tau/tau-2.31/exa
File Options Windows Help

```
Metric: TAUGPU_TIME
Value: Exclusive
Units: seconds

    0.6 .TAU application
  0.531 cl_mem clCreateBuffer(cl_context, cl_mem_flags, size_t, void *, cl_int *) C
  0.413 cl_int clWaitForEvents(cl_uint, const cl_event *) C
  0.378 cl_int clEnqueueWriteBuffer(cl_command_queue, cl_mem, cl_bool, size_t, size_t, const void *, cl_uint, const cl_event *, cl_event *)
  0.157 cl_context clCreateContext(const cl_context_properties *, cl_uint, const cl_device_id *, void (*)(const char *, const void *, size_t, v
  0.083 cl_int clEnqueueReadBuffer(cl_command_queue, cl_mem, cl_bool, size_t, size_t, void *, cl_uint, const cl_event *, cl_event *) C
  0.069 cl_int clEnqueueNDRangeKernel(cl_command_queue, cl_kernel, cl_uint, const size_t *, const size_t *, const size_t *, cl_uint, const c
  0.028 cl_int clGetPlatformIDs(cl_uint, cl_platform_id *, cl_uint *) C
  0.005 cl_int clBuildProgram(cl_program, cl_uint, const cl_device_id *, const char *, void (*)(cl_program, void *), void *) C
 5.8E-5 cl_int clReleaseProgram(cl_program) C
 1.7E-5 cl_int clGetDeviceInfo(cl_device_id, cl_device_info, size_t, void *, size_t *) C
 8.0E-6 cl_command_queue clCreateCommandQueue(cl_context, cl_device_id, cl_command_queue_properties, cl_int *) C
 8.0E-6 cl_int clFinish(cl_command_queue) C
 7.0E-6 cl_int clSetKernelArg(cl_kernel, cl_uint, size_t, const void *) C
 3.0E-6 cl_program clCreateProgramWithSource(cl_context, cl_uint, const char **, const size_t *, cl_int *) C
 3.0E-6 cl_int clReleaseKernel(cl_kernel) C
 2.0E-6 cl_kernel clCreateKernel(cl_program, const char *, cl_int *) C
 2.0E-6 cl_int clGetCommandQueueInfo(cl_command_queue, cl_command_queue_info, size_t, void *, size_t *) C
 1.0E-6 cl_int clGetDeviceIDs(cl_platform_id, cl_device_type, cl_uint, cl_device_id *, cl_uint *) C
 1.0E-6 cl_int clGetProgramBuildInfo(cl_program, cl_device_id, cl_program_build_info, size_t, void *, size_t *) C
 1.0E-6 cl_int clGetKernelInfo(cl_kernel, cl_kernel_info, size_t, void *, size_t *) C
 1.0E-6 cl_int clReleaseCommandQueue(cl_command_queue) C
 1.0E-6 cl_int clReleaseContext(cl_context) C
```

```
Metric: TAUGPU_TIME
Value: Exclusive
Units: seconds

0.413 multiply_matrices
0.166 .TAU application
0.157 WriteBuffer
0.083 ReadBuffer
```

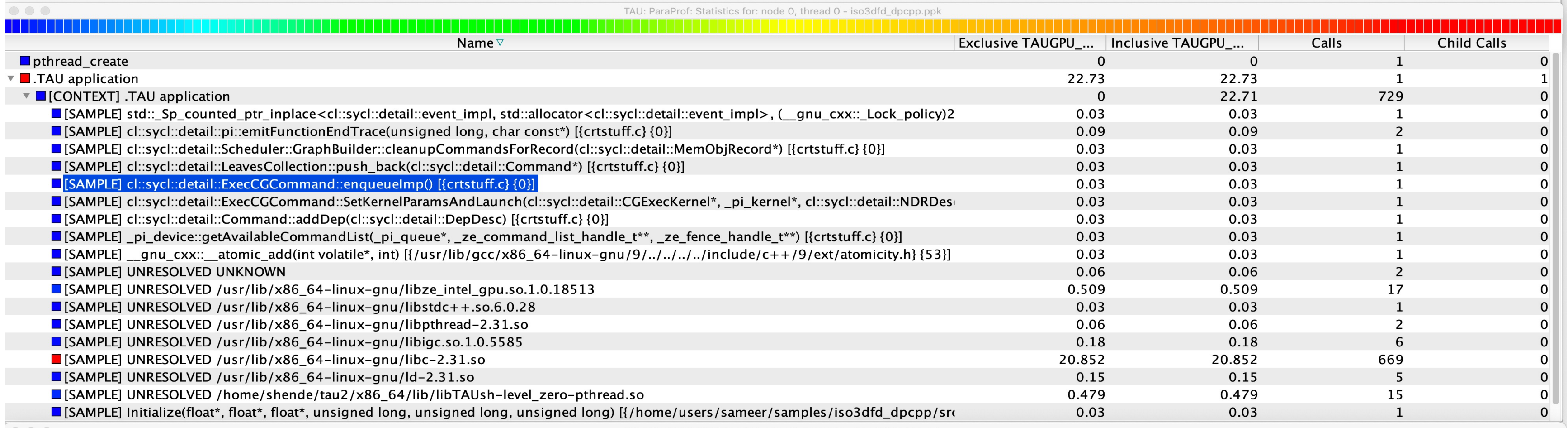
```
% tau_exec –T cupti,serial –opencl ./a.out
```

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



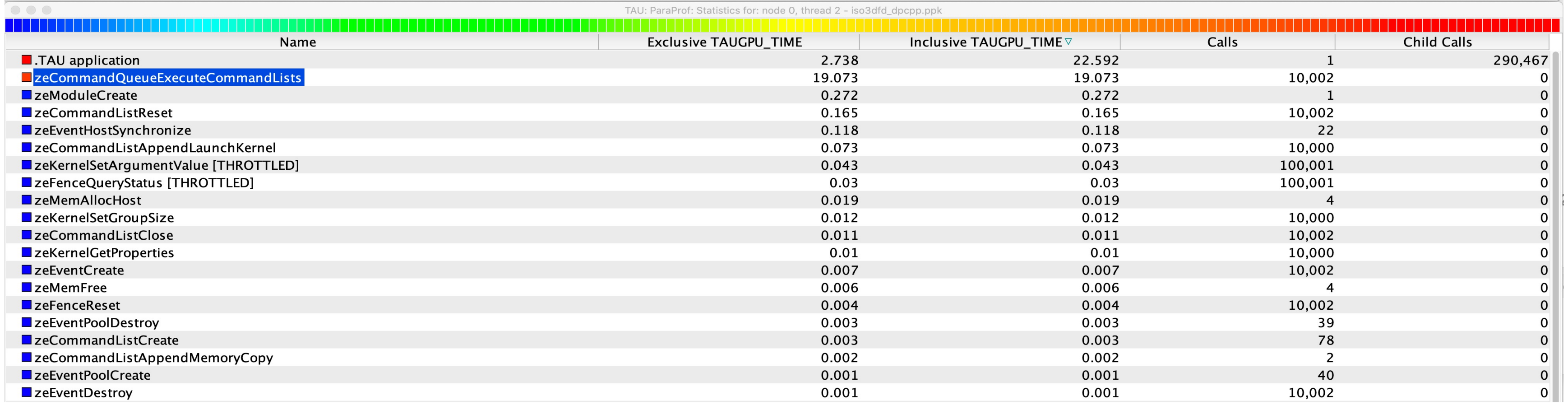
TAU: ParaProf: Statistics for: node 0, thread 1 - iso3dfd_dpcpp.ppk

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

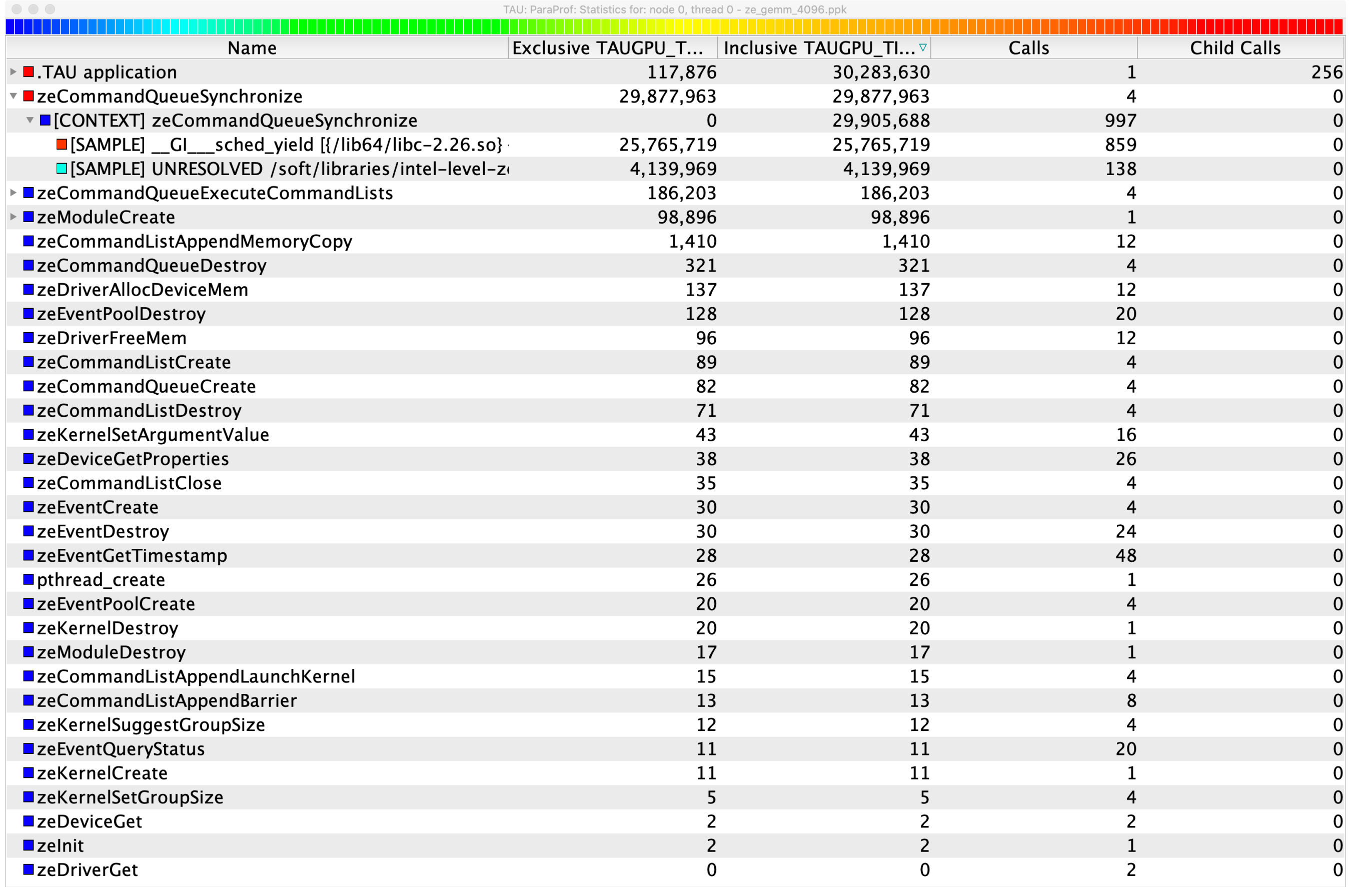


TAU: ParaProf: Statistics for: node 0, thread 2 - iso3dfd_dpcpp.ppk

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
```

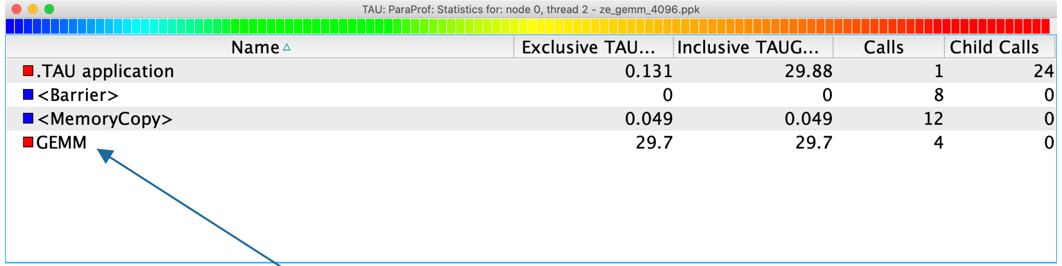
Intel Level Zero (TigerLake Gen12LP integrated CPUs or DG1)



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

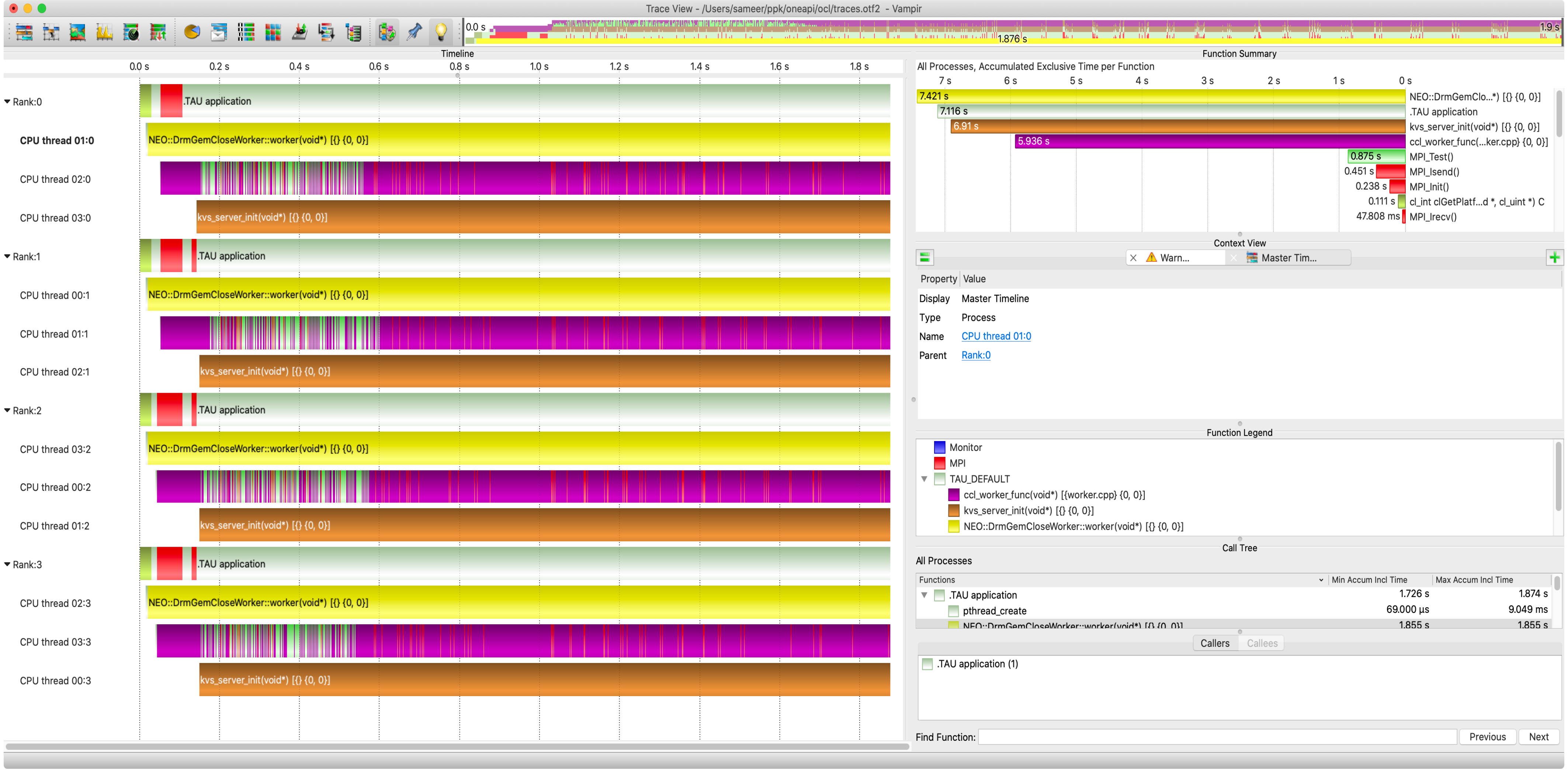
Name	Exclusive TAU...	Inclusive TAUG...	Calls	Child Calls
.TAU application	0.131	29.88	1	24
<Barrier>	0	0	8	0
<MemoryCopy>	0.049	0.049	12	0
GEMM	29.7	29.7	4	0

Units: seconds

Time spent in GEMM kernel

```
% mpirun –np 64 tau_exec –l0 ./a.out
```

TAU and Vampir [TU Dresden]: Intel oneAPI OpenCL



Property	Value
Display	Master Timeline
Type	Process
Name	CPU thread 01:0
Parent	Rank:0

- 🟦 Monitor
- 🟥 MPI
- TAU_DEFAULT
  - ccl_worker_func(void*) [{worker.cpp} {0, 0}]
  - kvs_server_init(void*) [{} {0, 0}]
  - NEO::DrmGemCloseWorker::worker(void*) [{} {0, 0}]

All Processes

Functions	Min Accum Incl Time	Max Accum Incl Time
.TAU application	1.726 s	1.874 s
pthread_create	69.000 µs	9.049 ms
NEO::DrmGemCloseWorker::worker(void*) [{} {0, 0}]	1.855 s	1.855 s

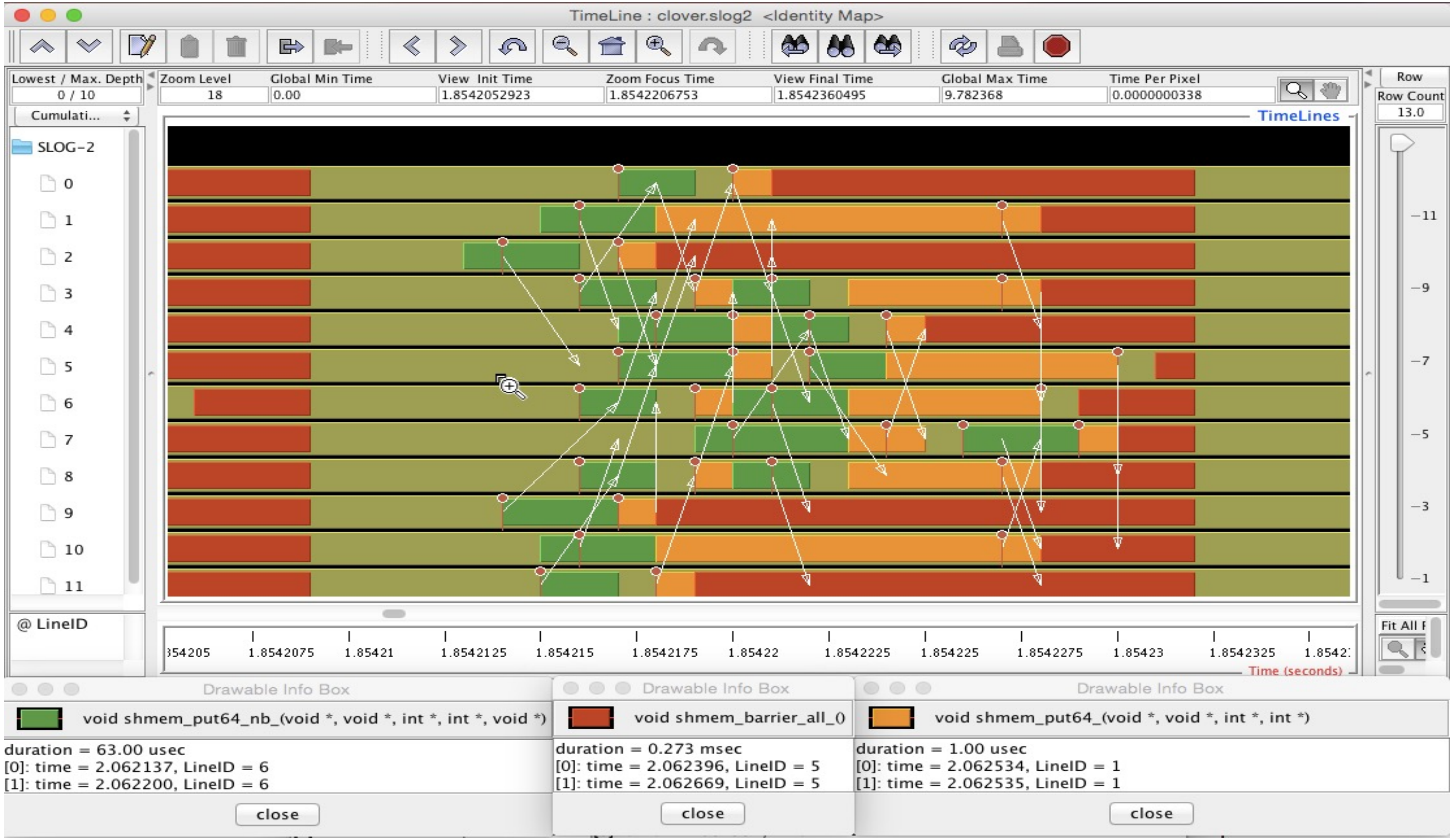
.TAU application (1)

% export TAU_TRACE=1; export TAU_TRACE_FORMAT=otf2

% mpirun –np 4 tau_exec –T level_zero –opencl ./a.out

EXASCALE COMPUTING PROJECT

Tracing: Jumpshot (ships with TAU)



TimeLine : clover.slog2 <Identity Map>

Lowest / Max. Depth: 0 / 10

Zoom Level: 18

Global Min Time: 0.00

View Init Time: 1.8542052923

Zoom Focus Time: 1.8542206753

View Final Time: 1.8542360495

Global Max Time: 9.782368

Time Per Pixel: 0.0000000338

Row Count: 13.0

Drawable Info Box

🟩 void shmem_put64_nb_(void *, void *, int *, int *, void *)

duration = 63.00 usec
[0]: time = 2.062137, LineID = 6
[1]: time = 2.062200, LineID = 6

close

Drawable Info Box

🟥 void shmem_barrier_all_()

duration = 0.273 msec
[0]: time = 2.062396, LineID = 5
[1]: time = 2.062669, LineID = 5

close

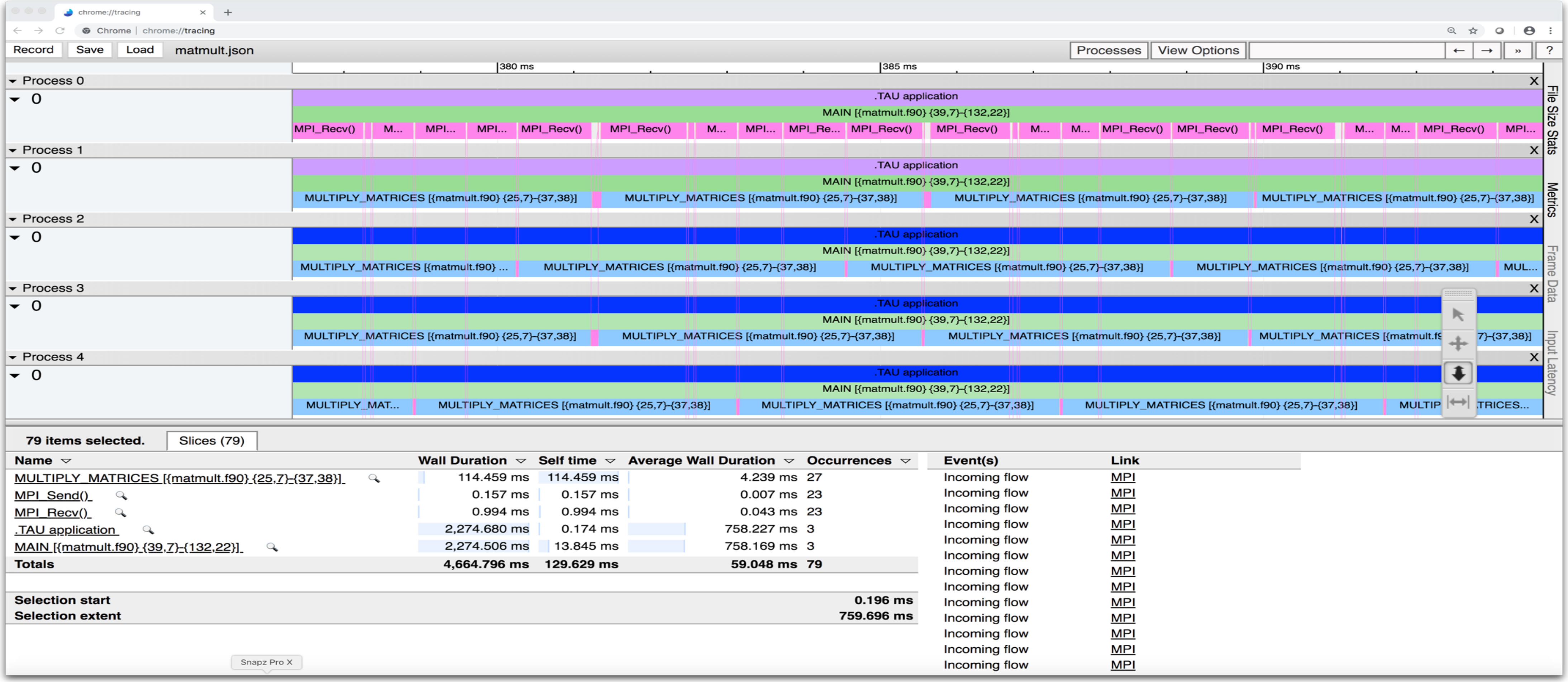
Drawable Info Box

🟧 void shmem_put64_(void *, void *, int *, int *)

duration = 1.00 usec
[0]: time = 2.062534, LineID = 1
[1]: time = 2.062535, LineID = 1

close

Tracing: Chrome Browser or Perfetto.dev



79 items selected. Slices (79)

Name	Wall Duration	Self time	Average Wall Duration	Occurrences
MULTIPLY_MATRICES [{matmult.f90} {25,7}-{37,38}]	114.459 ms	114.459 ms	4.239 ms	27
MPI_Send()	0.157 ms	0.157 ms	0.007 ms	23
MPI_Recv()	0.994 ms	0.994 ms	0.043 ms	23
.TAU application	2,274.680 ms	0.174 ms	758.227 ms	3
MAIN [{matmult.f90} {39,7}-{132,22}]	2,274.506 ms	13.845 ms	758.169 ms	3
Totals	**4,664.796 ms**	**129.629 ms**	**59.048 ms**	**79**

Selection start	**0.196 ms**
Selection extent	**759.696 ms**

Event(s)	Link
Incoming flow	MPI
Incoming flow	MPI
Incoming flow	MPI
Incoming flow	MPI
Incoming flow	MPI
Incoming flow	MPI
Incoming flow	MPI
Incoming flow	MPI
Incoming flow	MPI
Incoming flow	MPI
Incoming flow	MPI
Incoming flow	MPI
Incoming flow	MPI

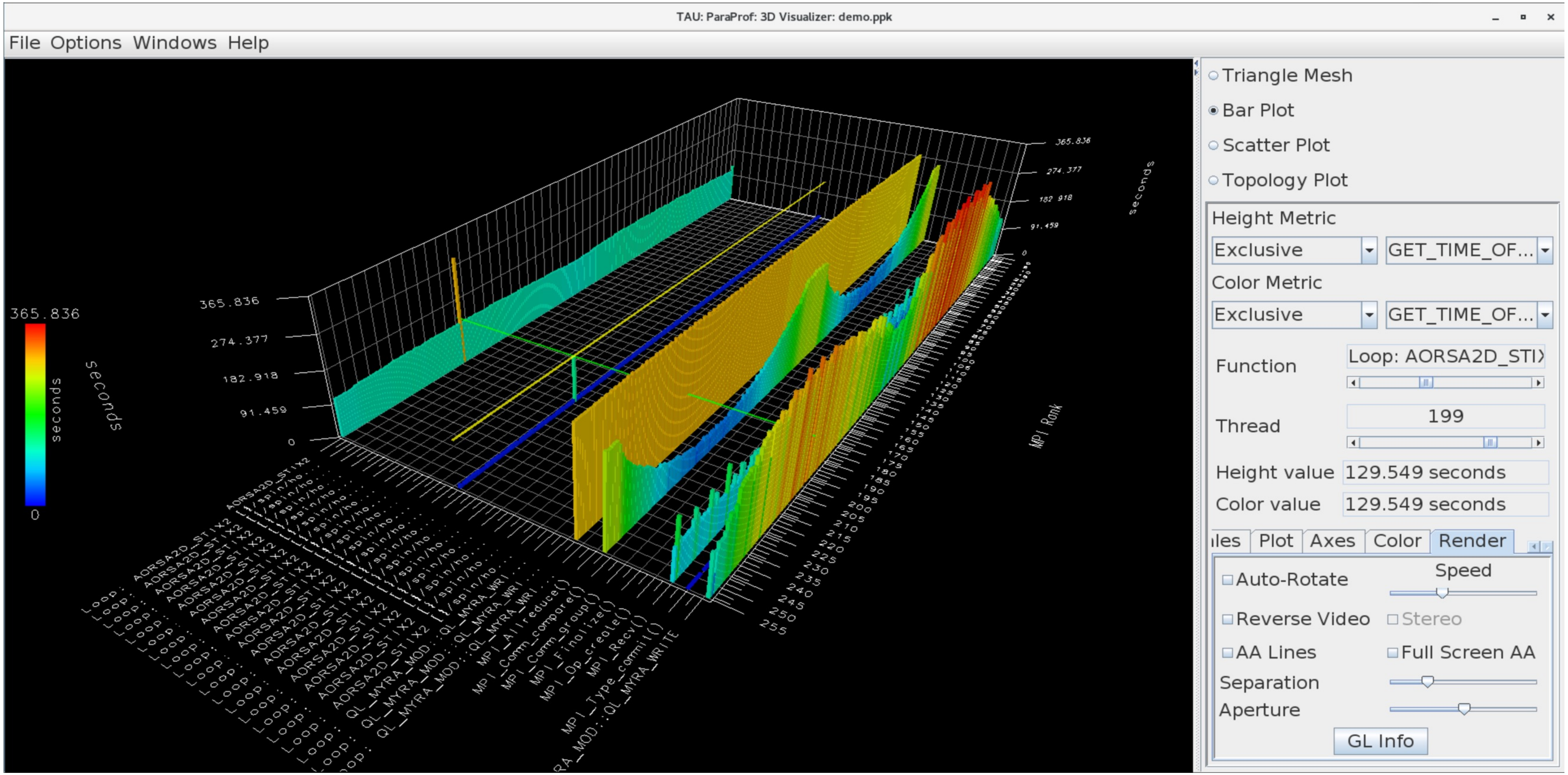
```
% export TAU_TRACE=1
% mpirun –np 256 tau_exec ./a.out
% tau_treemerge.pl; tau_trace2json tau.trc tau.edf –chrome –ignoreatomic –o app.json
```

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

Perfetto.dev (open the UI)

EXASCALE COMPUTING PROJECT

ParaProf 3D Profile Browser



○ Triangle Mesh

● Bar Plot

○ Scatter Plot

○ Topology Plot

Height Metric

Exclusive ▾ GET_TIME_OF... ▾

Color Metric

Exclusive ▾ GET_TIME_OF... ▾

Function: Loop: AORSA2D_STI)

Thread: 199

Height value: 129.549 seconds

Color value: 129.549 seconds

□ Auto-Rotate Speed

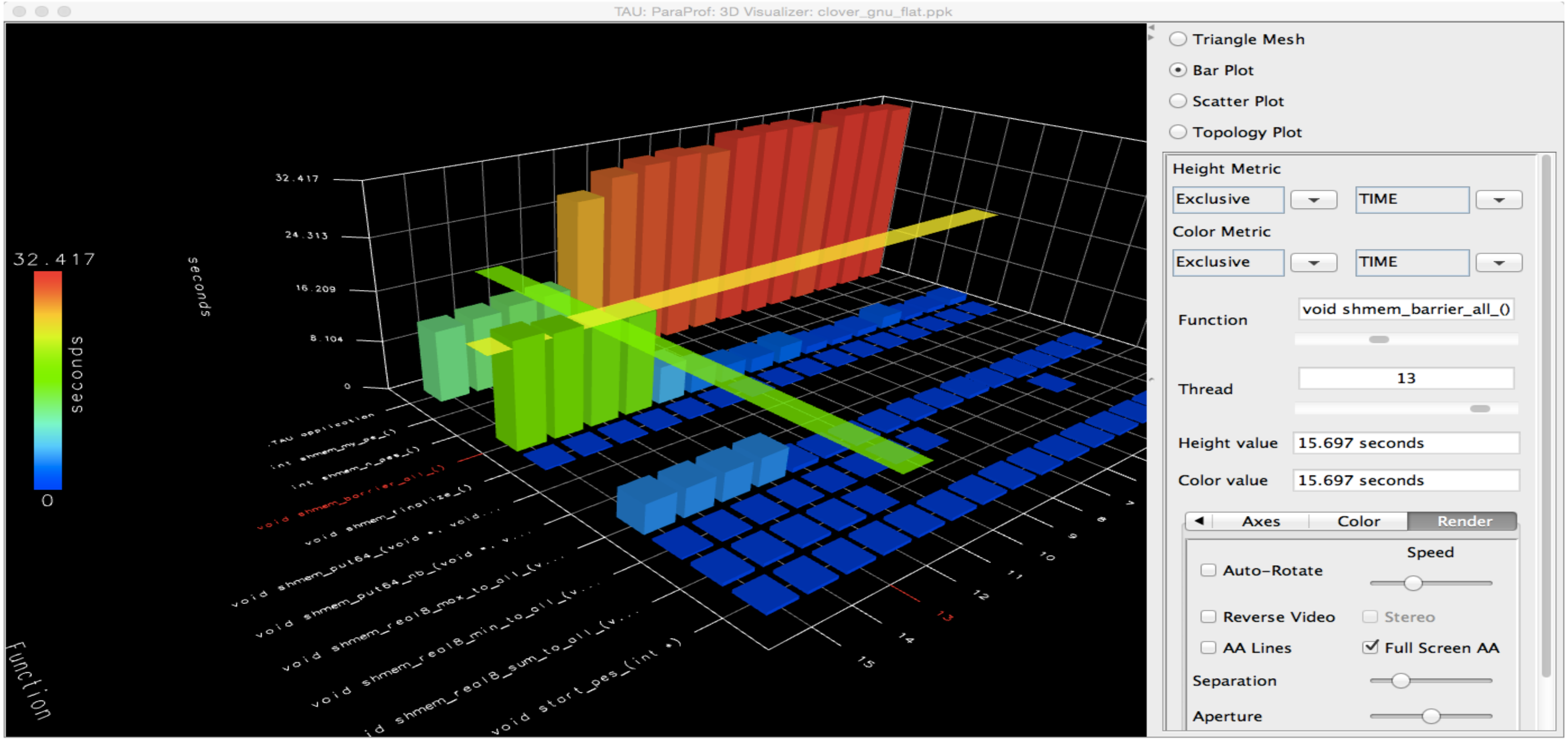
□ Reverse Video □ Stereo

□ AA Lines □ Full Screen AA

Separation

Aperture

TAU – ParaProf 3D Visualization


TAU: ParaProf: 3D Visualizer: clover_gnu_flat.ppk
32.417
seconds
0
32.417
24.313
16.209
8.104
0
seconds
Function
.TAU application
int shmem_my_pe_()
int shmem_n_pes_()
void shmem_barrier_all_()
void shmem_finalize_()
void shmem_put64_(void *, void...
void shmem_put64_nb_(void *, v...
void shmem_real8_max_to_all_(v...
void shmem_real8_min_to_all_(v...
void shmem_real8_sum_to_all_(v...
void start_pes_(int *)
id shmem_
void
7
8
9
10
11
12
13
14
15
Axes
Color
Render

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

Height Metric: Exclusive, TIME

Color Metric: Exclusive, TIME

Function: void shmem_barrier_all_()

Thread: 13

Height value: 15.697 seconds

Color value: 15.697 seconds

Speed

□ Auto-Rotate

□ Reverse Video □ Stereo

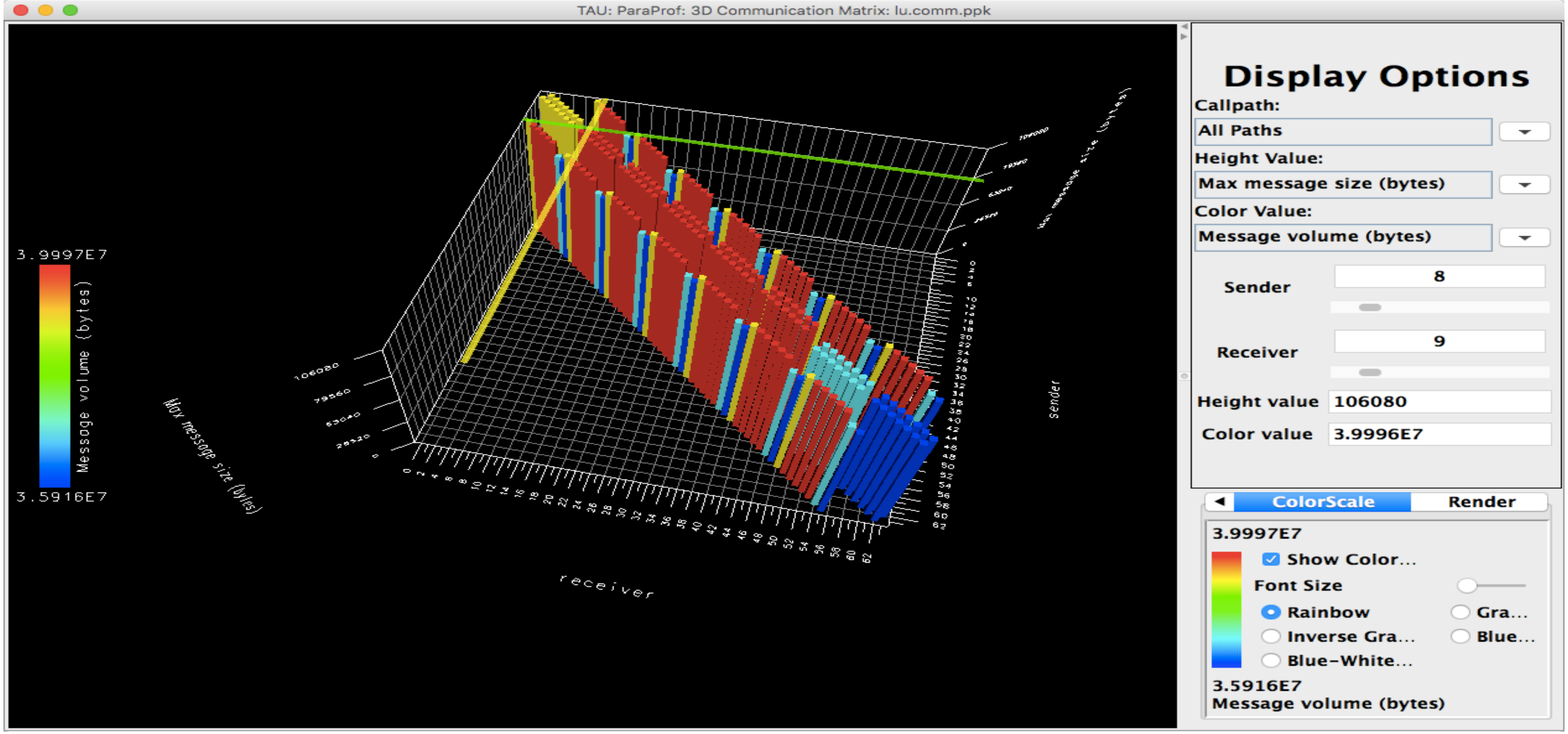
□ AA Lines ☑ Full Screen AA

Separation

Aperture

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

TAU – 3D Communication Window



TAU: ParaProf: 3D Communication Matrix: lu.comm.ppk

Display Options

Callpath: All Paths

Height Value: Max message size (bytes)

Color Value: Message volume (bytes)

Sender 8

Receiver 9

Height value 106080

Color value 3.9996E7

3.9997E7

☑ Show Color...

● Rainbow ○ Gra...

○ Inverse Gra... ○ Blue...

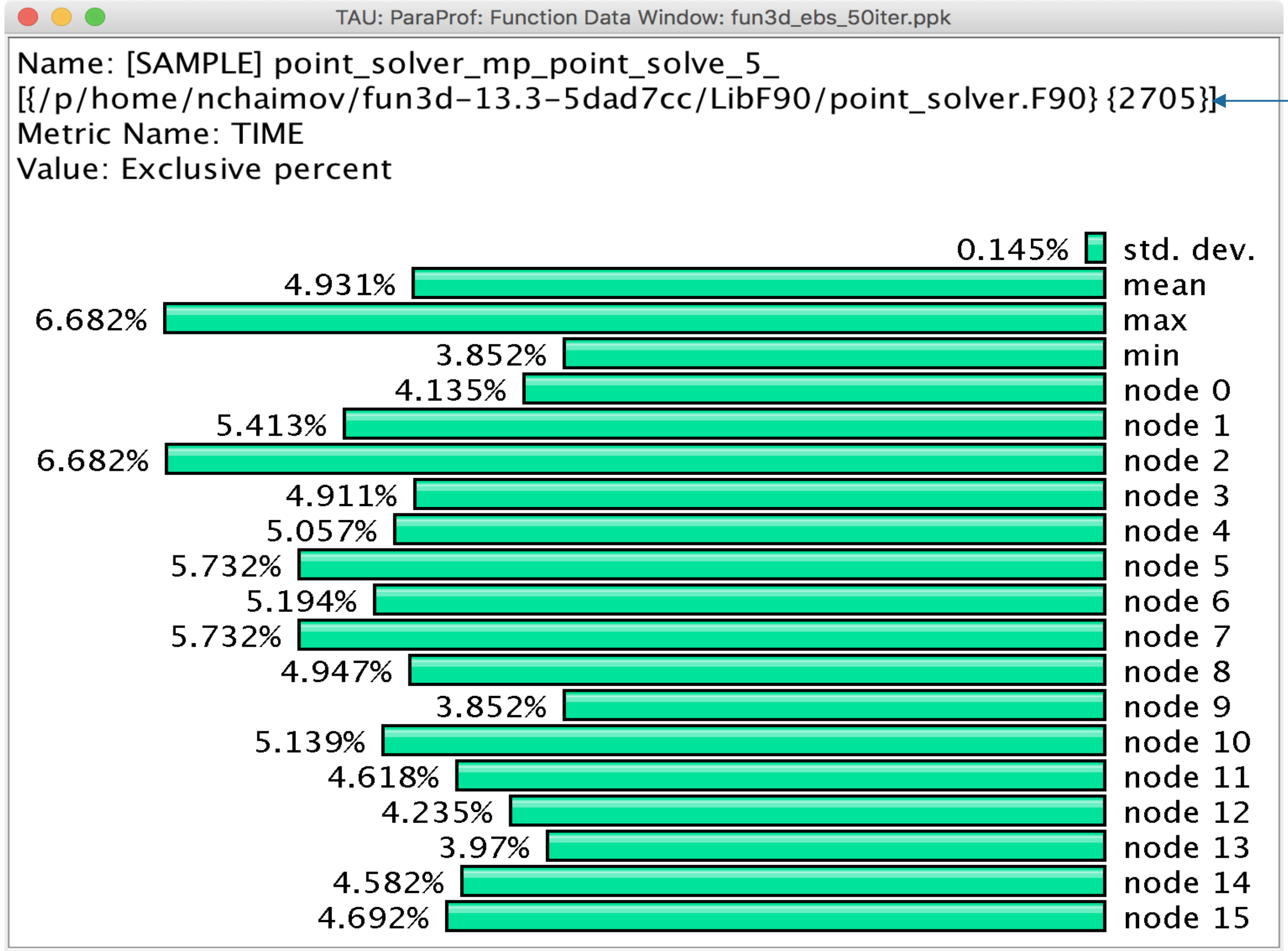
○ Blue-White...

3.5916E7

Message volume (bytes)

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

Event Based Sampling (EBS)



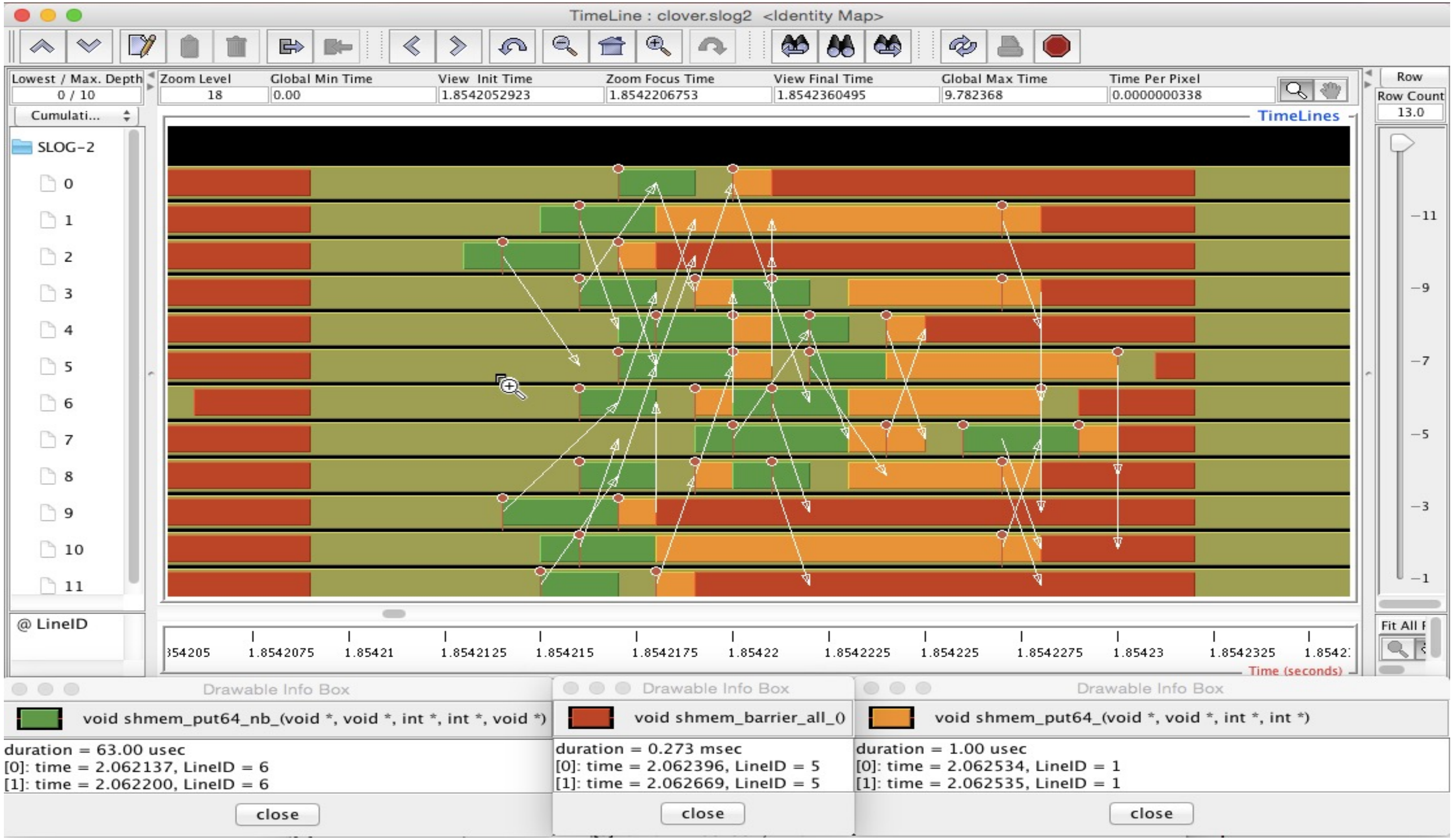
Name: [SAMPLE] point_solver_mp_point_solve_5_
[{/p/home/nchaimov/fun3d-13.3-5dad7cc/LibF90/point_solver.F90} {2705}]
Metric Name: TIME
Value: Exclusive percent

File: point_solver.F90
Line: 2705

Uninstrumented!

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

Tracing: Jumpshot (ships with TAU)



TimeLine : clover.slog2 <Identity Map>

Lowest / Max. Depth	Zoom Level	Global Min Time	View Init Time	Zoom Focus Time	View Final Time	Global Max Time	Time Per Pixel	Row Count
0 / 10	18	0.00	1.8542052923	1.8542206753	1.8542360495	9.782368	0.0000000338	13.0

Drawable Info Box

🟩 void shmem_put64_nb_(void *, void *, int *, int *, void *)

duration = 63.00 usec
[0]: time = 2.062137, LineID = 6
[1]: time = 2.062200, LineID = 6

Drawable Info Box

🟥 void shmem_barrier_all_()

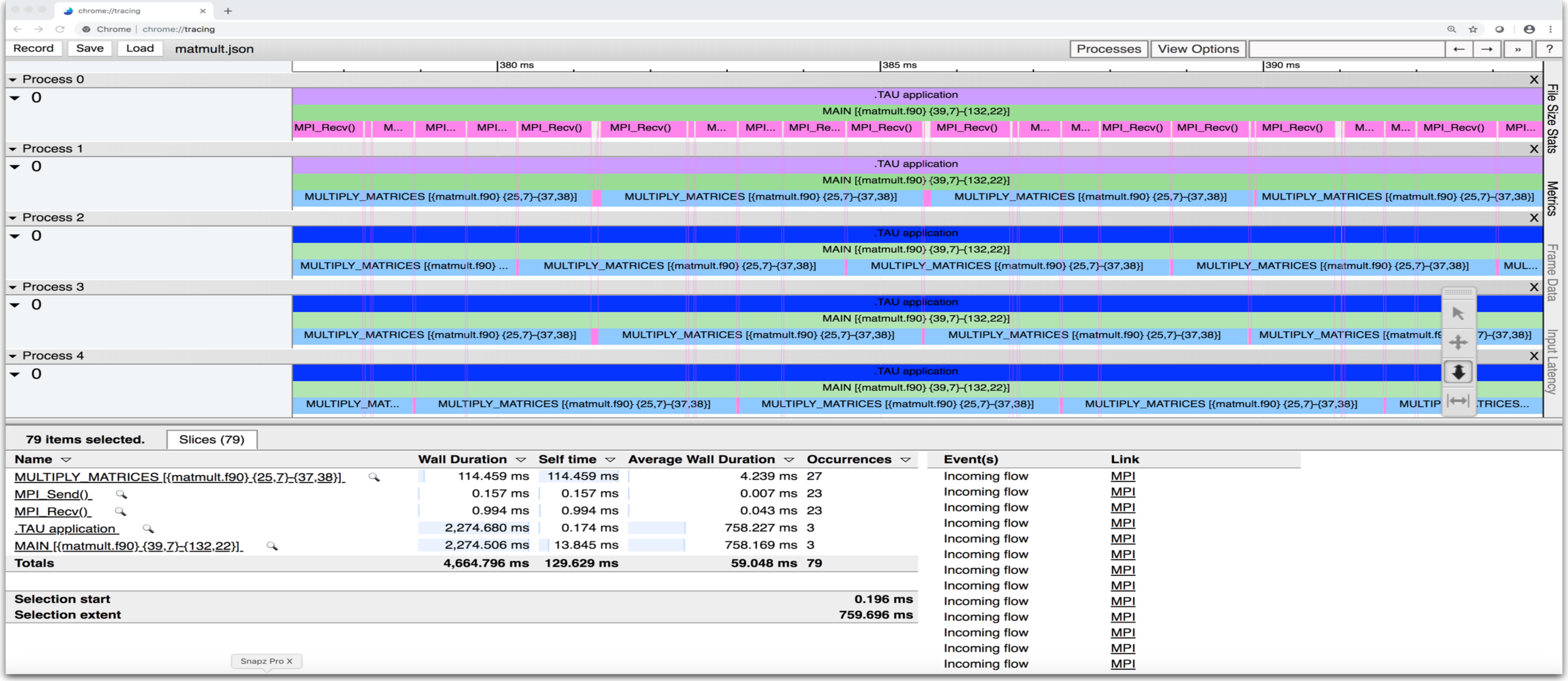
duration = 0.273 msec
[0]: time = 2.062396, LineID = 5
[1]: time = 2.062669, LineID = 5

Drawable Info Box

🟧 void shmem_put64_(void *, void *, int *, int *)

duration = 1.00 usec
[0]: time = 2.062534, LineID = 1
[1]: time = 2.062535, LineID = 1

Tracing: Chrome Browser



79 items selected. Slices (79)

Name	Wall Duration	Self time	Average Wall Duration	Occurrences
MULTIPLY_MATRICES [{matmult.f90} {25,7}-{37,38}]	114.459 ms	114.459 ms	4.239 ms	27
MPI_Send()	0.157 ms	0.157 ms	0.007 ms	23
MPI_Recv()	0.994 ms	0.994 ms	0.043 ms	23
.TAU application	2,274.680 ms	0.174 ms	758.227 ms	3
MAIN [{matmult.f90} {39,7}-{132,22}]	2,274.506 ms	13.845 ms	758.169 ms	3
Totals	**4,664.796 ms**	**129.629 ms**	**59.048 ms**	**79**

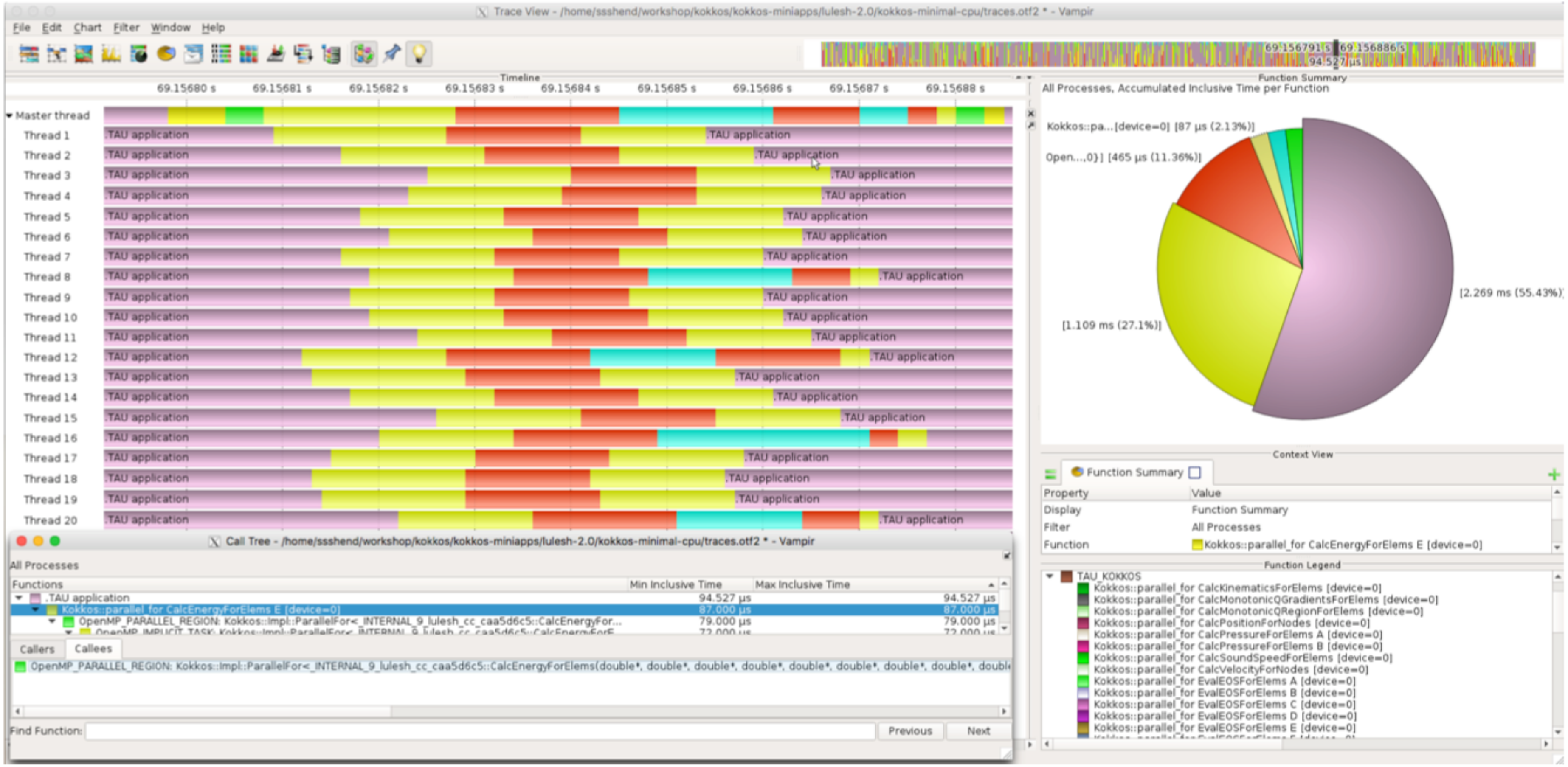
Selection start	**0.196 ms**
Selection extent	**759.696 ms**

Event(s)	Link
Incoming flow	MPI
Incoming flow	MPI
Incoming flow	MPI
Incoming flow	MPI
Incoming flow	MPI
Incoming flow	MPI
Incoming flow	MPI
Incoming flow	MPI
Incoming flow	MPI
Incoming flow	MPI
Incoming flow	MPI
Incoming flow	MPI
Incoming flow	MPI

```
% 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



Call Tree - /home/ssshend/workshop/kokkos/kokkos-miniapps/lulesh-2.0/kokkos-minimal-cpu/traces.otf2 * - Vampir

All Processes

Functions	Min Inclusive Time	Max Inclusive Time
.TAU application	94.527 µs	94.527 µs
Kokkos::parallel_for CalcEnergyForElems E [device=0]	87.000 µs	87.000 µs
OpenMP_PARALLEL_REGION: Kokkos::Impl::ParallelFor<_INTERNAL_9_lulesh_cc_caa5d6c5::CalcEnergyFor...	79.000 µs	79.000 µs
OpenMP_IMPLICIT_TASK: Kokkos::Impl::ParallelFor<_INTERNAL_9_lulesh_cc_caa5d6c5::CalcEnergyForE...	72.000 µs	72.000 µs

Callers Callees

OpenMP_PARALLEL_REGION: Kokkos::Impl::ParallelFor<_INTERNAL_9_lulesh_cc_caa5d6c5::CalcEnergyForElems(double*, double*, double*, double*, double*, double*, double*, double*, doubl

Find Function:

Property	Value
Display	Function Summary
Filter	All Processes
Function	Kokkos::parallel_for CalcEnergyForElems E [device=0]

TAU_KOKKOS
- Kokkos::parallel_for CalcKinematicsForElems [device=0]
- Kokkos::parallel_for CalcMonotonicQGradientsForElems [device=0]
- Kokkos::parallel_for CalcMonotonicQRegionForElems [device=0]
- Kokkos::parallel_for CalcPositionForNodes [device=0]
- Kokkos::parallel_for CalcPressureForElems A [device=0]
- Kokkos::parallel_for CalcPressureForElems B [device=0]
- Kokkos::parallel_for CalcSoundSpeedForElems [device=0]
- Kokkos::parallel_for CalcVelocityForNodes [device=0]
- Kokkos::parallel_for EvalEOSForElems A [device=0]
- Kokkos::parallel_for EvalEOSForElems B [device=0]
- Kokkos::parallel_for EvalEOSForElems C [device=0]
- Kokkos::parallel_for EvalEOSForElems D [device=0]
- Kokkos::parallel_for EvalEOSForElems E [device=0]

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

EXASCALE COMPUTING PROJECT

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

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

Level Zero (DPC++/SYCL Intel oneAPI)

- Track execution of kernels on GPU
- Track time spent in level zero runtime system calls

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

Level zero + MPI

Intel oneAPI level zero (DPC++) + PMPI MPI interface

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

Installing TAU with support for OpenCL for NVIDIA, Intel, AMD GPUs

```
% wget http://tau.uoregon.edu/tau.tgz; tar xf tau.tgz; cd tau-<version>;
% ./configure -bfd=download -cuda=/packages/cuda/11.4.1 -tag=nvidia ; make install -j
% ./configure -bfd=download -opencl -tag=intel; make install -j
% ./configure -bfd=download -opencl=/opt/rocm-5.1.0/opencl -tag=amd; make install -j
% ls x86_64/lib/Makefile*
Makefile.tau-amd
Makefile.tau-intel
Makefile.tau-nvidia-cupti

% cd examples/gpu/opencl; make;
% tau_exec -T serial,[amd,intel,nvidia] -opencl ./matmult
% pprof -a | more
% paraprof &
```

Using TAU’s Runtime Preloading Tool: tau_exec

- Preload a wrapper that intercepts the runtime system call and substitutes with another
  - **MPI**
  - **OpenMP**
  - **OpenCL**
  - **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)

TAU Execution Command (tau_exec)

Uninstrumented execution

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

Track GPU operations

```
% mpirun –np 256  tau_exec –cupti ./a.out
% mpirun –np 256  tau_exec –opencl ./a.out
% mpirun –np 256  tau_exec –rocm ./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 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

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
- CEA, France
- Industry: AMD, ARM, Intel, IBM, NVIDIA
- Partners:
  - University of Oregon
  - The Ohio State University
  - ParaTools, Inc.
  - University of Tennessee, Knoxville
  - T.U. Dresden, GWT
  - Jülich Supercomputing Center

Thank you

https://www.exascaleproject.org

This research was supported by the Exascale Computing Project (17-SC-20-SC), a joint project of the U.S. Department of Energy's Office of Science and National Nuclear Security Administration, responsible for delivering a capable exascale ecosystem, including software, applications, and hardware technology, to support the nation's exascale computing imperative.

ECP

EXASCALE COMPUTING PROJECT

Thank you to all collaborators in the ECP and broader computational science communities. The work discussed in this presentation represents creative contributions of many people who are passionately working toward next-generation computational science.