The TAU Performance System®

Kevin Huck, Sameer Shende, Allen Malony

khuck@cs.uoregon.edu

http://tau.uoregon.edu

U.S. DEPARTMENT OF ENERGY | Office of Science

Tuning and Analysis Utilities

RAPIDS

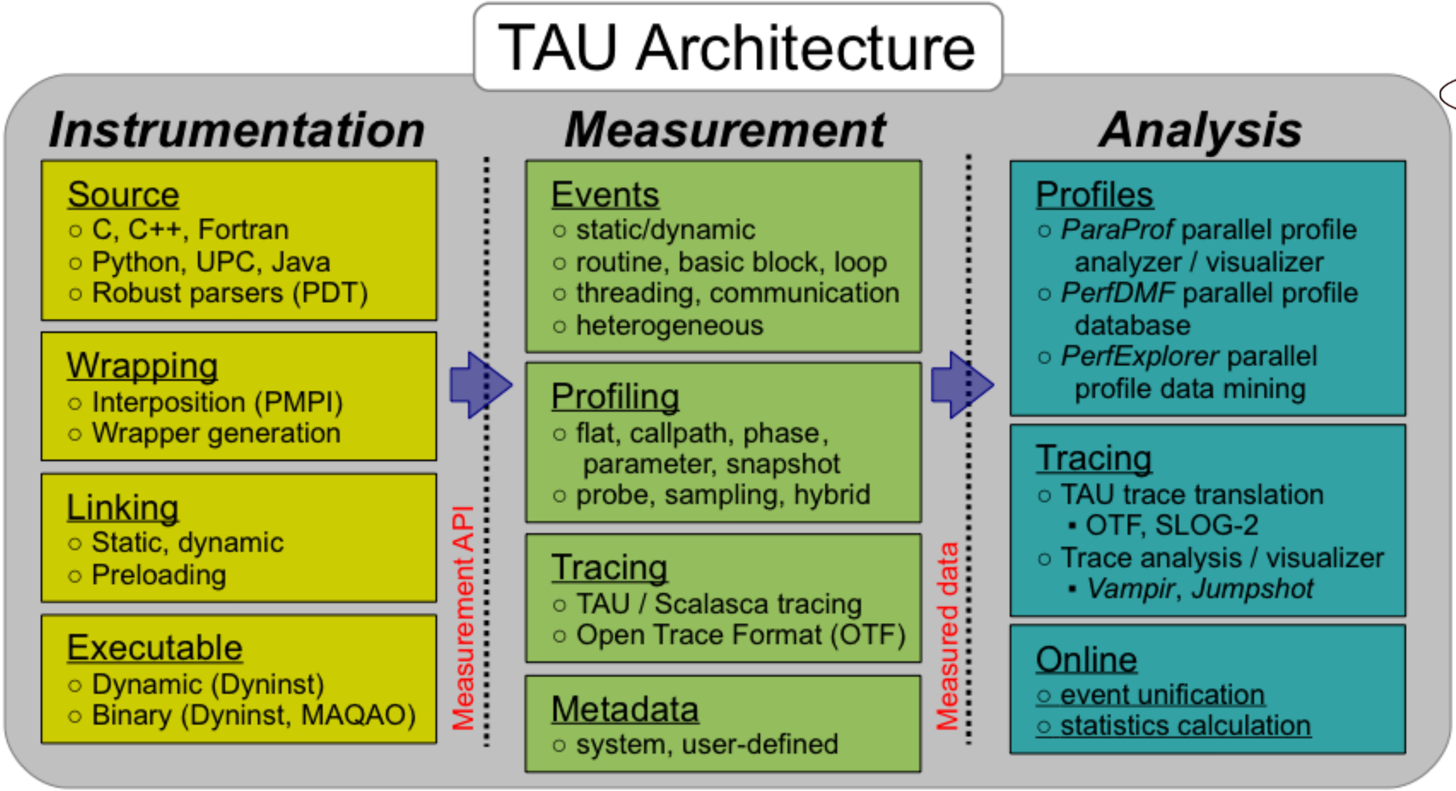
ECP EXASCALE COMPUTING PROJECT

UNIVERSITY OF OREGON

TAU : brief overview

- Tuning and Analysis Utilities (28+ year project)
- Integrated performance toolkit:
  - Multi-level performance instrumentation
  - Highly configurable
  - Widely ported performance profiling / tracing system
  - Portable (java, python) visualization / exploration / analysis tools
- Supports all major HPC programming models
  - MPI/SHMEM, OpenMP/ACC, CUDA, HIP, OneAPI, Kokkos...

TAU : brief overview



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

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

Analysis

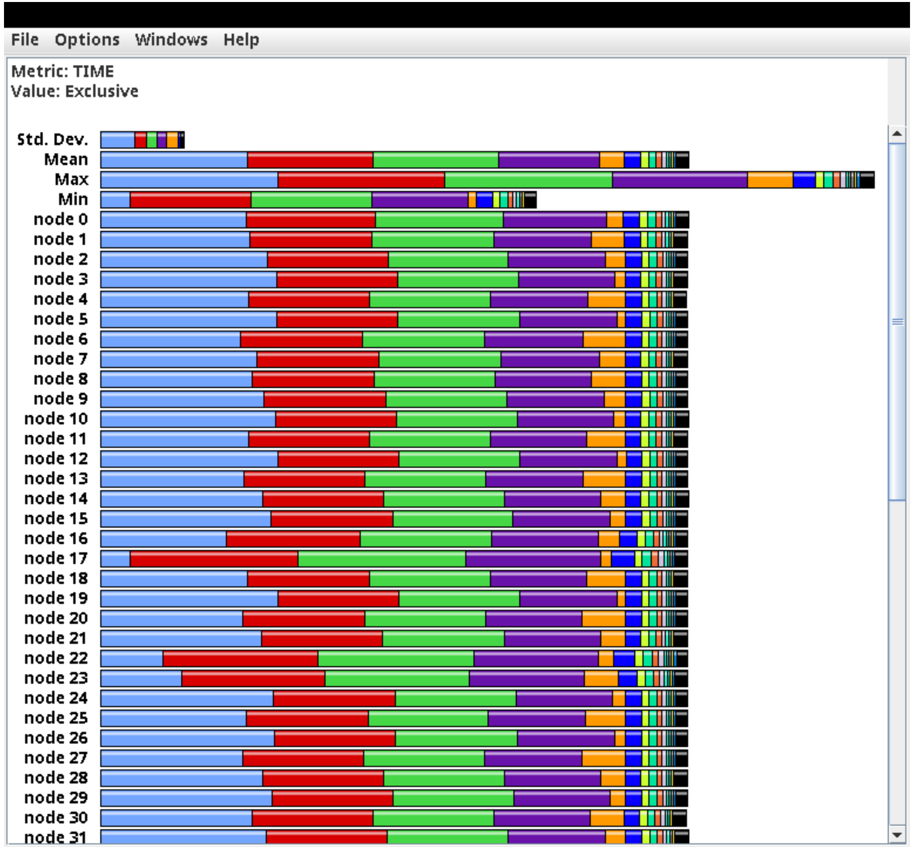
Profiles
- *ParaProf* parallel profile analyzer / visualizer
- *PerfDMF* parallel profile database
- *PerfExplorer* parallel profile data mining

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

Online
- event unification
- statistics calculation

ParaProf Profile Browser

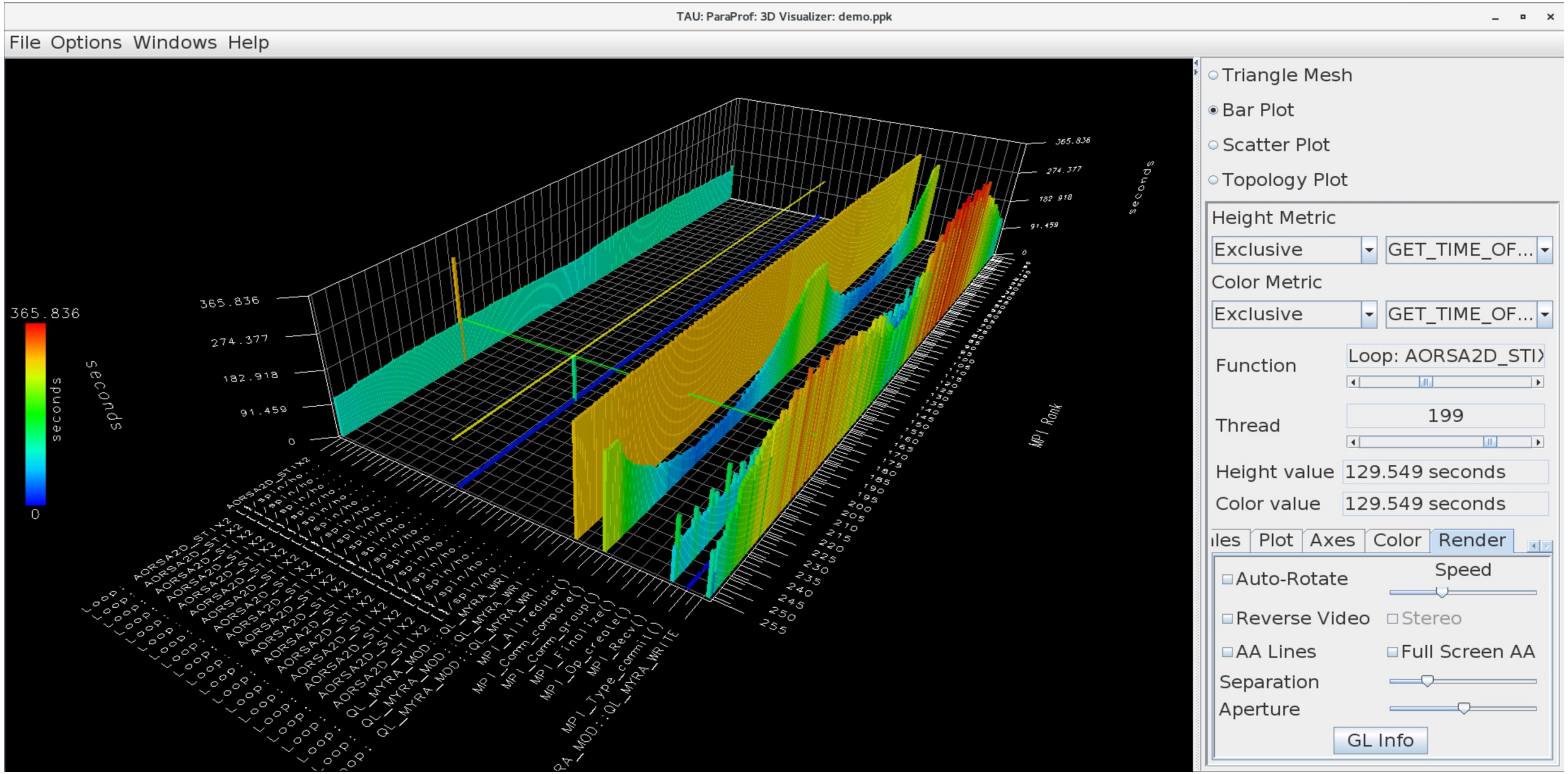
```
% paraprof
```



Metric: TIME
Value: Exclusive

Each line is a different process/thread of execution, each color is a different function

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

iles | Plot | Axes | Color | Render

□ Auto-Rotate Speed

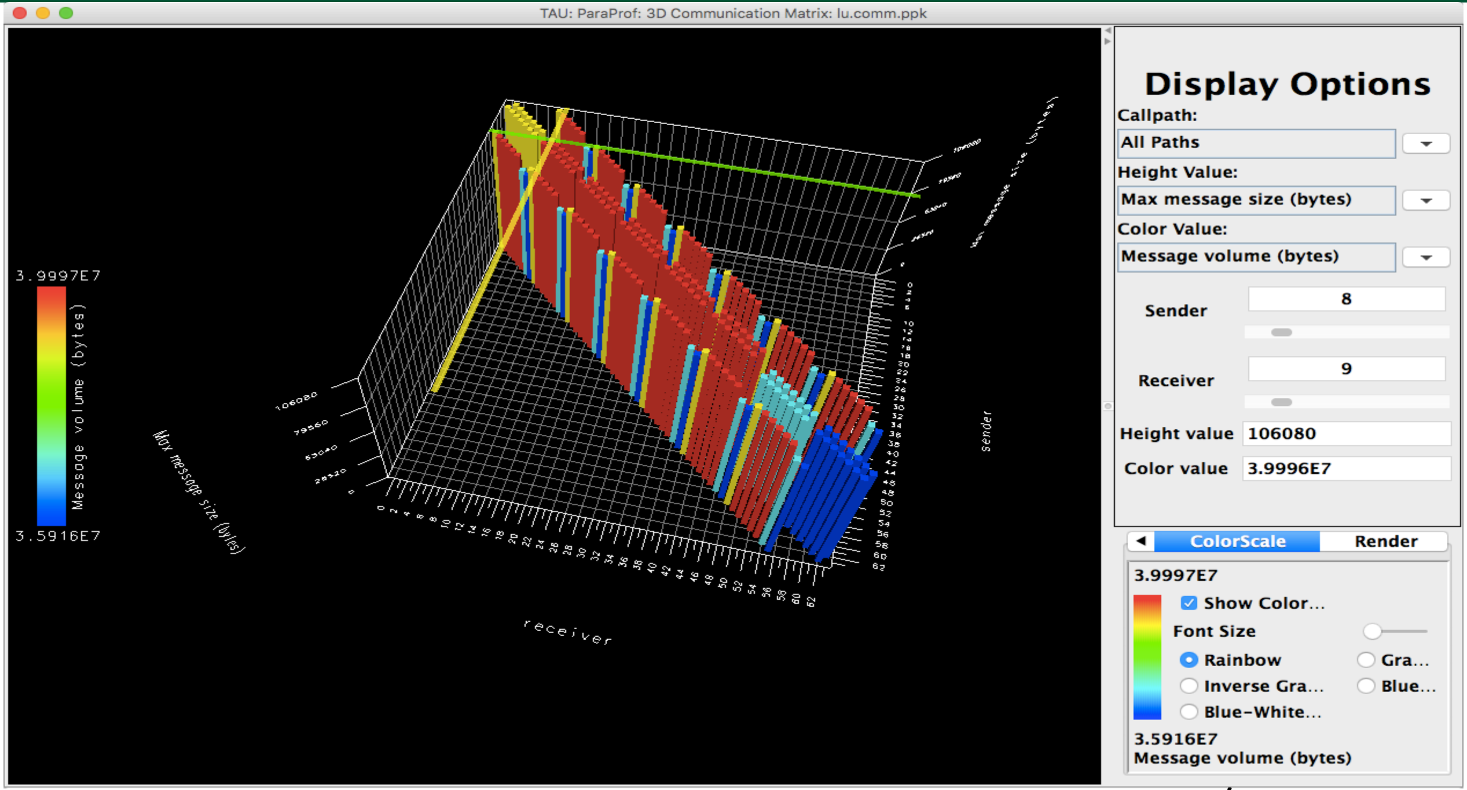
□ Reverse Video □ Stereo

□ AA Lines □ Full Screen AA

Separation

Aperture

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

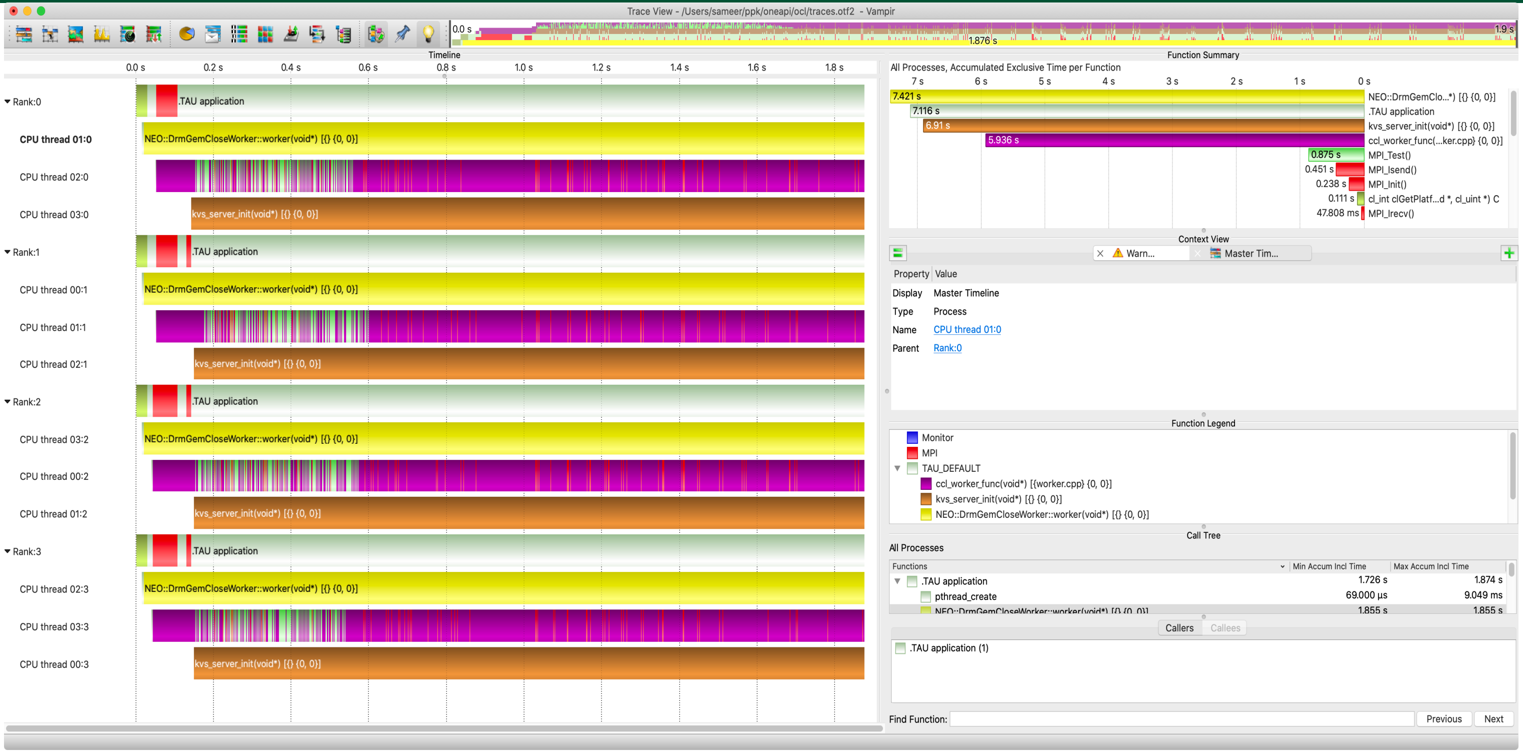
Receiver: 9

Height value: 106080

Color value: 3.9996E7

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

TAU and Vampir [TU Dresden]: Intel oneAPI OpenCL



Trace View - /Users/sameer/ppk/oneapi/ocl/traces.otf2 - Vampir

Function Summary

All Processes, Accumulated Exclusive Time per Function

Time	Function
7.421 s	NEO::DrmGemClo...*) [{} {0, 0}]
7.116 s	.TAU application
6.91 s	kvs_server_init(void*) [{} {0, 0}]
5.936 s	ccl_worker_func(...ker.cpp} {0, 0}]
0.875 s	MPI_Test()
0.451 s	MPI_Isend()
0.238 s	MPI_Init()
0.111 s	cl_int clGetPlatf...d *, cl_uint *) C
47.808 ms	MPI_Irecv()

Context View

Warn... | Master Tim...

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

Function Legend

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

Call Tree

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

Callers | Callees

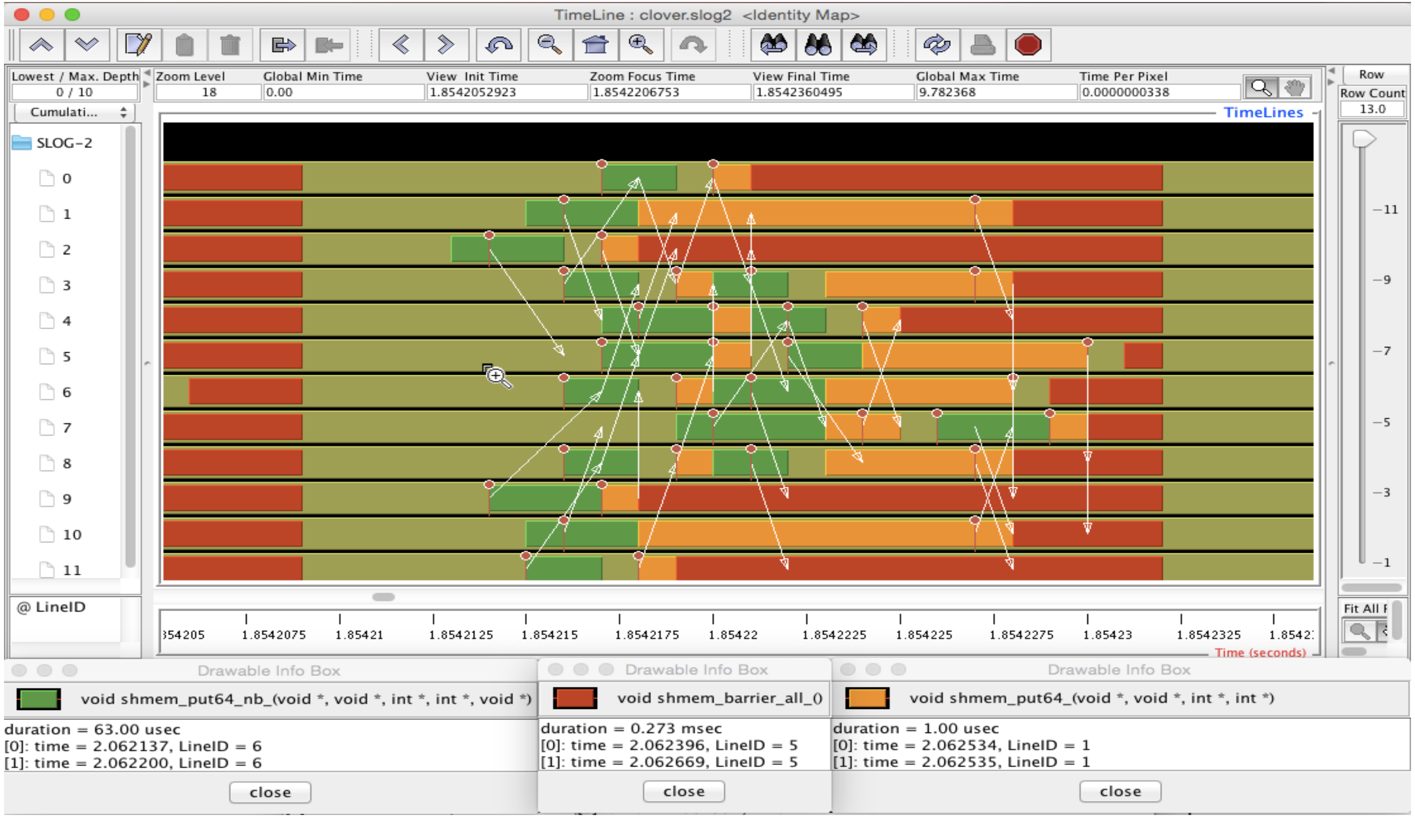
.TAU application (1)

Find Function: Previous Next

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

UNIVERSITY OF OREGON

Tracing: Jumpshot (ships with TAU)



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

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

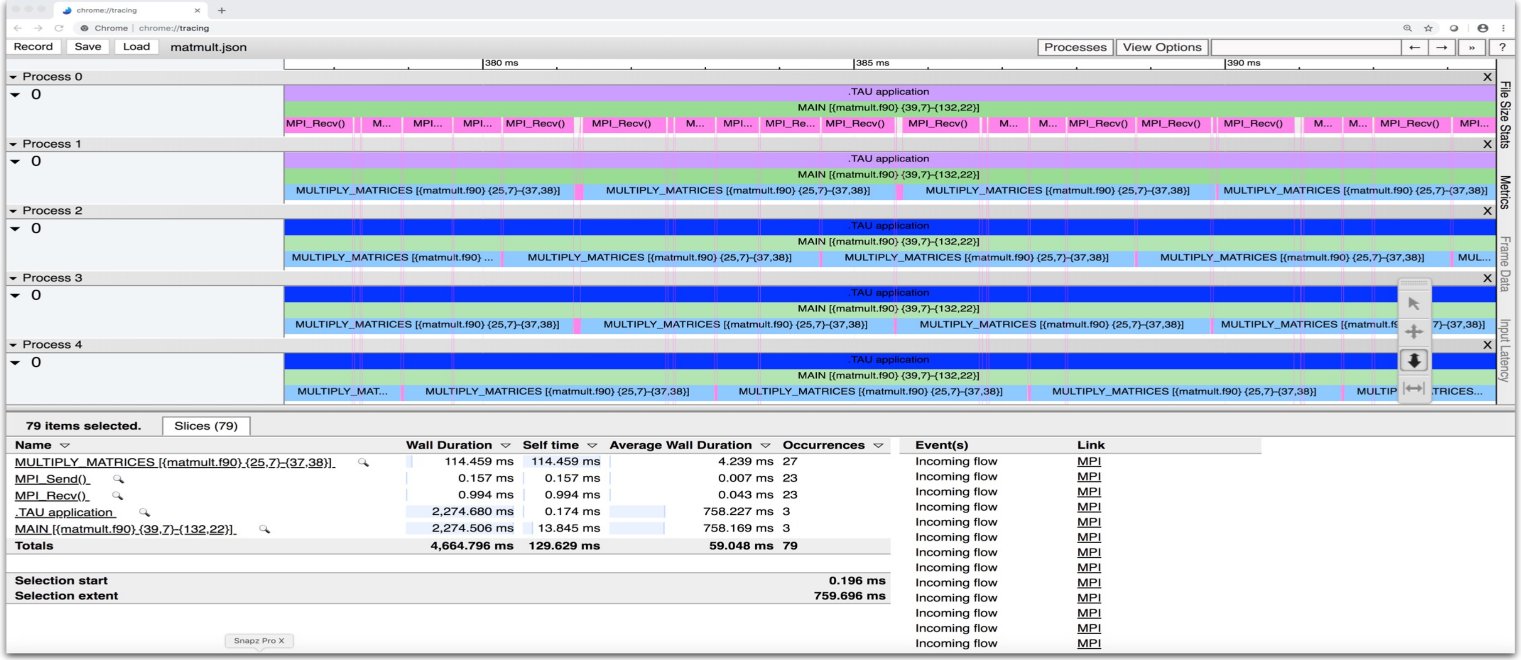
🟧 void shmem_put64_(void *, void *, int *, int *)

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

close

UNIVERSITY OF OREGON

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)

Or visit https://ui.perfetto.dev/ to use Perfetto

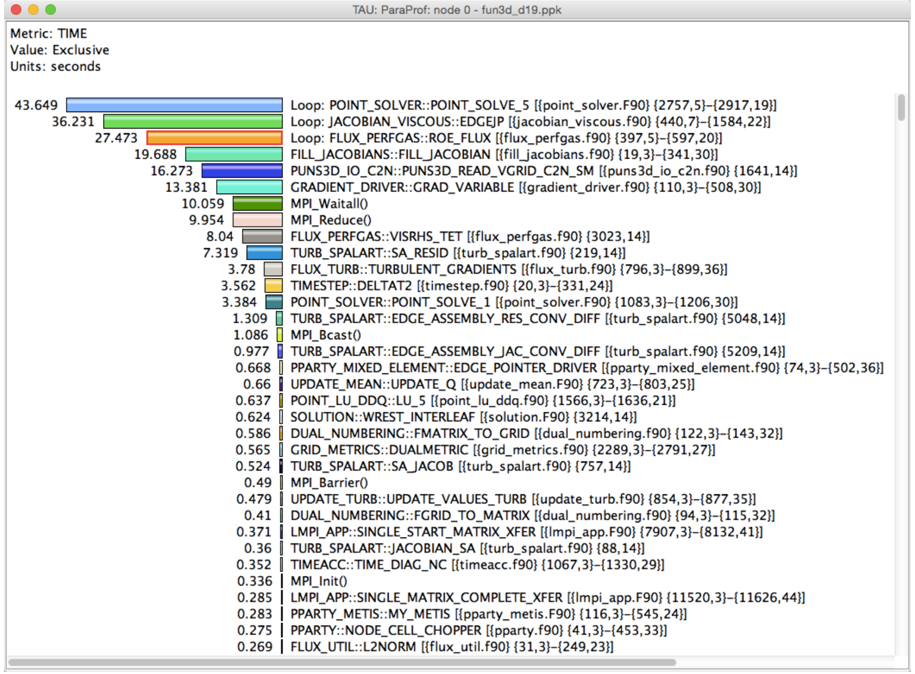
Performance Measurement

- **Timers**
  - Requires instrumentation of some kind
    - Manual, automated
    - Source, compiler provided, binary
    - Library callbacks, API wrappers, weak symbol replacement
  - Simple to implement
- **Sampling**
  - Requires specialized system libraries / support
    - Periodic signals, signal handler
  - No modification to executable/library needed
  - Potential to interfere with system support (signal handlers)

Profiling and Tracing

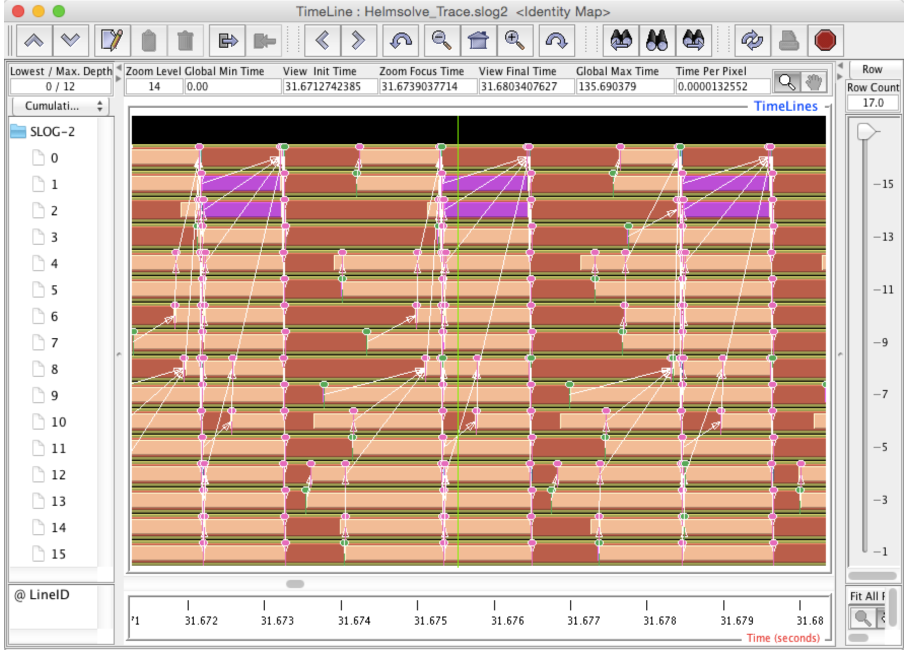
- **Profiling**: how much time was spent in each measured function on each thread in each process?
  - Collapses the time axis
  - No ordering or causal event information
  - Small summary per thread/process, regardless of execution time – only grows with number of timers & threads/processes
- **Tracing**: record all function entry & exit events on a timeline
  - Detailed view of what happened
  - The longer the program runs, the bigger the trace

Profiling and Tracing



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}]
```



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

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

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

Integrating TAU

Compile Time

- Compile with TAU compiler wrappers (see next slides)
- Link with TAU libraries

Runtime

- Execute with `tau_exec`
- Preloads the TAU shared object library and instantiates measurement support for different models
- More later…

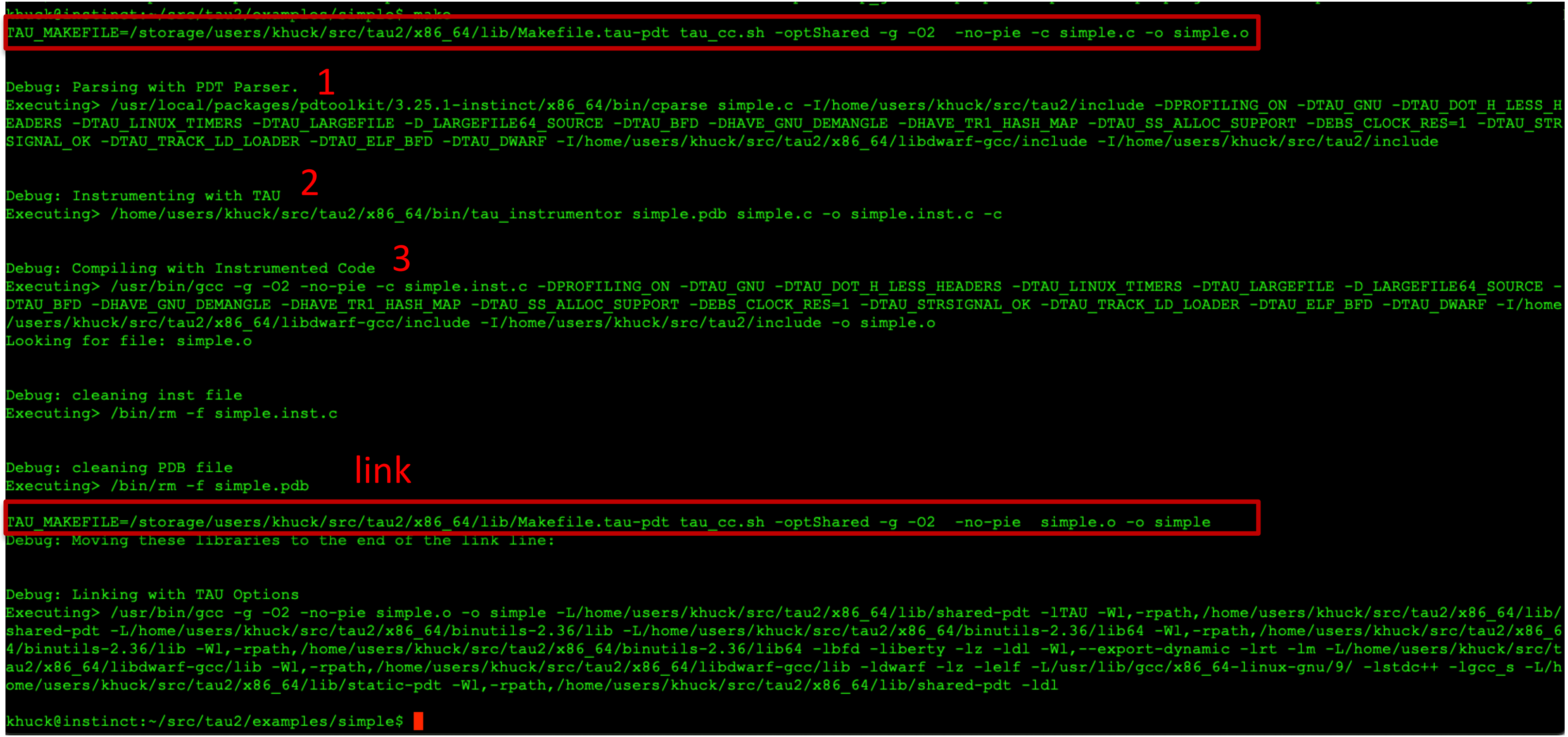
Instrumentation Approaches

- Manual
  - Add TAU API calls to the code by hand: https://www.cs.uoregon.edu/research/tau/docs/newguide/bk05rn01.html
- Automated:
  - PDT – optional TAU configuration
  - Compiler based instrumentation – comes with TAU
  - LLVM plugin – comes with TAU
  - Binary instrumentation - using Dyninst, PIN, or MAQAO
    - Optional TAU configuration, not covered in this tutorial
- PerfStubs API: https://github.com/UO-OACISS/perfstubs

TAU compiler wrappers with PDT

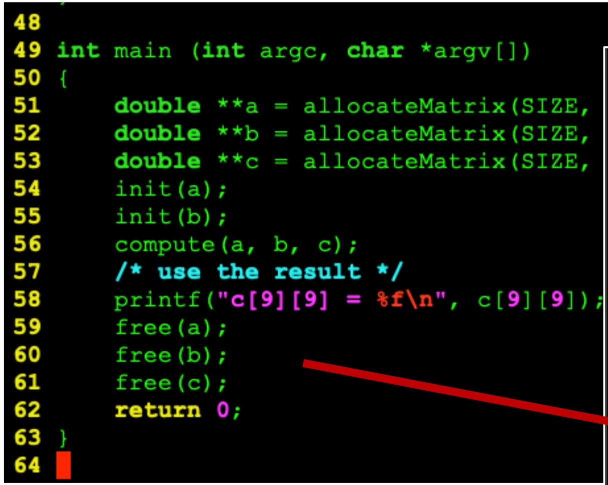
- `tau_cc.sh`, `tau_cxx.sh`, `tau_f90.sh`
- Usually does 3 passes to compile:
  - PDT parses the source file, writes a .pdb file
  - tau_instrumentor reads the source file, the pdb file, writes an instrumented source file
  - The instrumented source file is passed to the regular compiler
  - The instrumented source file is deleted
- Selective instrumentation by file, function (include/exclude)
- At link time, tau_*.sh will add the TAU libraries to the link

PDT Example



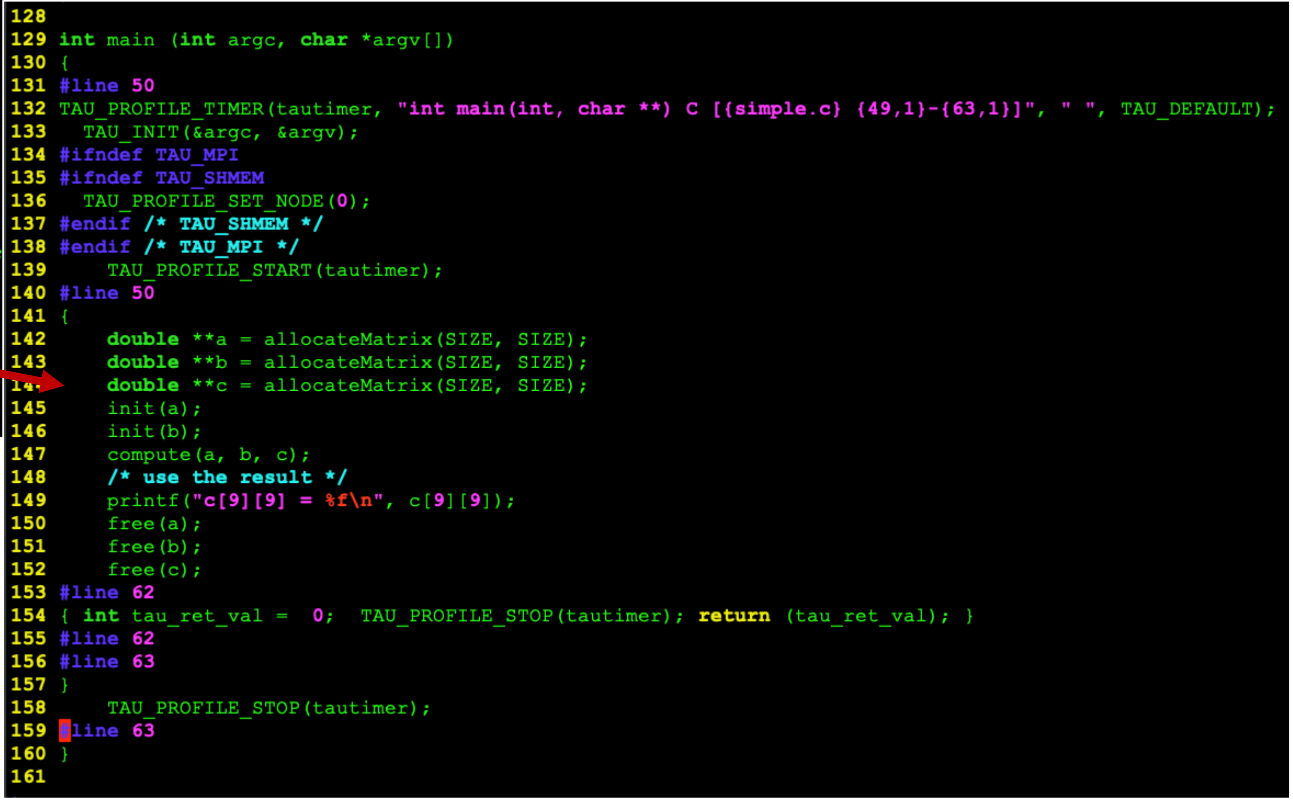
```
khuck@instinct:~/src/tau2/examples/simple$ make
TAU_MAKEFILE=/storage/users/khuck/src/tau2/x86_64/lib/Makefile.tau-pdt tau_cc.sh -optShared -g -O2  -no-pie -c simple.c -o simple.o


Debug: Parsing with PDT Parser.
Executing> /usr/local/packages/pdtoolkit/3.25.1-instinct/x86_64/bin/cparse simple.c -I/home/users/khuck/src/tau2/include -DPROFILING_ON -DTAU_GNU -DTAU_DOT_H_LESS_H
EADERS -DTAU_LINUX_TIMERS -DTAU_LARGEFILE -D_LARGEFILE64_SOURCE -DTAU_BFD -DHAVE_GNU_DEMANGLE -DHAVE_TR1_HASH_MAP -DTAU_SS_ALLOC_SUPPORT -DEBS_CLOCK_RES=1 -DTAU_STR
SIGNAL_OK -DTAU_TRACK_LD_LOADER -DTAU_ELF_BFD -DTAU_DWARF -I/home/users/khuck/src/tau2/x86_64/libdwarf-gcc/include -I/home/users/khuck/src/tau2/include


Debug: Instrumenting with TAU
Executing> /home/users/khuck/src/tau2/x86_64/bin/tau_instrumentor simple.pdb simple.c -o simple.inst.c -c


Debug: Compiling with Instrumented Code
Executing> /usr/bin/gcc -g -O2 -no-pie -c simple.inst.c -DPROFILING_ON -DTAU_GNU -DTAU_DOT_H_LESS_HEADERS -DTAU_LINUX_TIMERS -DTAU_LARGEFILE -D_LARGEFILE64_SOURCE -
DTAU_BFD -DHAVE_GNU_DEMANGLE -DHAVE_TR1_HASH_MAP -DTAU_SS_ALLOC_SUPPORT -DEBS_CLOCK_RES=1 -DTAU_STRSIGNAL_OK -DTAU_TRACK_LD_LOADER -DTAU_ELF_BFD -DTAU_DWARF -I/home
/users/khuck/src/tau2/x86_64/libdwarf-gcc/include -I/home/users/khuck/src/tau2/include -o simple.o
Looking for file: simple.o


Debug: cleaning inst file
Executing> /bin/rm -f simple.inst.c


Debug: cleaning PDB file
Executing> /bin/rm -f simple.pdb

TAU_MAKEFILE=/storage/users/khuck/src/tau2/x86_64/lib/Makefile.tau-pdt tau_cc.sh -optShared -g -O2  -no-pie  simple.o -o simple
Debug: Moving these libraries to the end of the link line:


Debug: Linking with TAU Options
Executing> /usr/bin/gcc -g -O2 -no-pie simple.o -o simple -L/home/users/khuck/src/tau2/x86_64/lib/shared-pdt -lTAU -Wl,-rpath,/home/users/khuck/src/tau2/x86_64/lib/
shared-pdt -L/home/users/khuck/src/tau2/x86_64/binutils-2.36/lib -L/home/users/khuck/src/tau2/x86_64/binutils-2.36/lib64 -Wl,-rpath,/home/users/khuck/src/tau2/x86_6
4/binutils-2.36/lib -Wl,-rpath,/home/users/khuck/src/tau2/x86_64/binutils-2.36/lib64 -lbfd -liberty -lz -ldl -Wl,--export-dynamic -lrt -lm -L/home/users/khuck/src/t
au2/x86_64/libdwarf-gcc/lib -Wl,-rpath,/home/users/khuck/src/tau2/x86_64/libdwarf-gcc/lib -ldwarf -lz -lelf -L/usr/lib/gcc/x86_64-linux-gnu/9/ -lstdc++ -lgcc_s -L/h
ome/users/khuck/src/tau2/x86_64/lib/static-pdt -Wl,-rpath,/home/users/khuck/src/tau2/x86_64/lib/shared-pdt -ldl

khuck@instinct:~/src/tau2/examples/simple$ 
```

PDT Instrumentation Example



```
int main (int argc, char *argv[])
{
    double **a = allocateMatrix(SIZE,
    double **b = allocateMatrix(SIZE,
    double **c = allocateMatrix(SIZE,
    init(a);
    init(b);
    compute(a, b, c);
    /* use the result */
    printf("c[9][9] = %f\n", c[9][9]);
    free(a);
    free(b);
    free(c);
    return 0;
}
```

tau_cc.sh -optKeepFiles -g -O2 -c simple.c -o simple.o

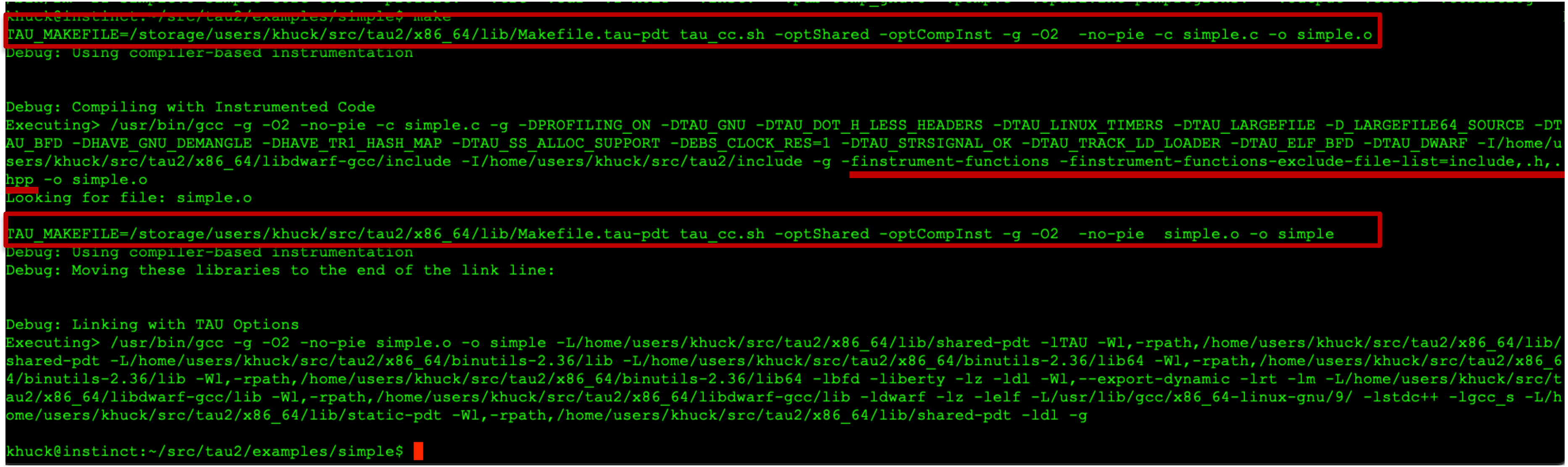


```

int main (int argc, char *argv[])
{
#line 50
TAU_PROFILE_TIMER(tautimer, "int main(int, char **) C [{simple.c} {49,1}-{63,1}]", " ", TAU_DEFAULT);
  TAU_INIT(&argc, &argv);
#ifndef TAU_MPI
#ifndef TAU_SHMEM
  TAU_PROFILE_SET_NODE(0);
#endif /* TAU_SHMEM */
#endif /* TAU_MPI */
    TAU_PROFILE_START(tautimer);
#line 50
{
    double **a = allocateMatrix(SIZE, SIZE);
    double **b = allocateMatrix(SIZE, SIZE);
    double **c = allocateMatrix(SIZE, SIZE);
    init(a);
    init(b);
    compute(a, b, c);
    /* use the result */
    printf("c[9][9] = %f\n", c[9][9]);
    free(a);
    free(b);
    free(c);
#line 62
{ int tau_ret_val =  0;  TAU_PROFILE_STOP(tautimer); return (tau_ret_val); }
#line 62
#line 63
}
    TAU_PROFILE_STOP(tautimer);
#line 63
}

```

TAU compiler wrappers without PDT

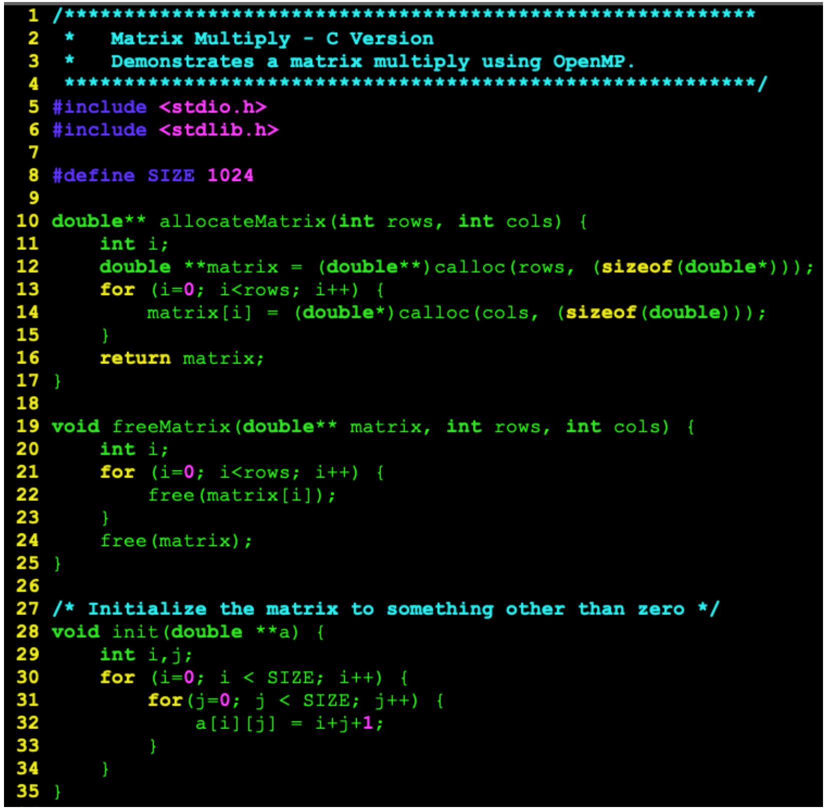
- Same `tau_cc.sh`, `tau_cxx.sh`, `tau_f90.sh`
- Usually does 1 pass to compile:
  - Extra flags are added to the compiler:
    - Compiler based instrumentation (-finstrument-functions)
      - Tool has to implement two special functions:
        - » void __cyg_profile_func_enter (void *this_fn, void *call_site);
        - » void __cyg_profile_func_exit (void *this_fn, void *call_site);
    - Call a compiler plugin (LLVM only)
      - tau_instrumentor adds TAU API calls directly
- At link time, tau_*.sh will add the TAU libraries to the link
- Can be forced with -optCompInst flag

Compiler Based Example

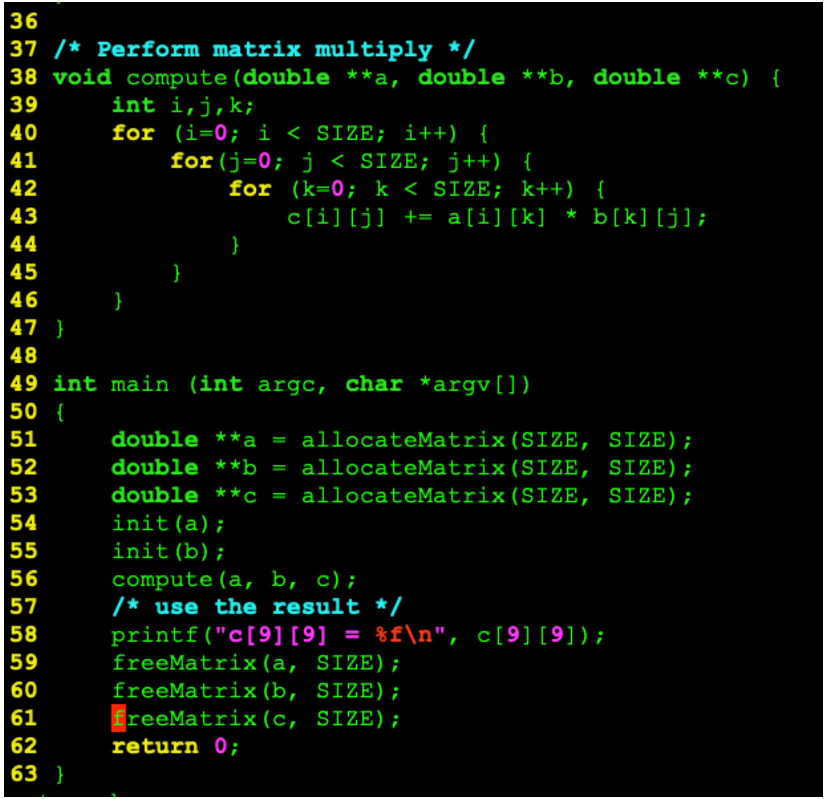


```
khuck@instinct:~/src/tau2/examples/simple$ make
TAU_MAKEFILE=/storage/users/khuck/src/tau2/x86_64/lib/Makefile.tau-pdt tau_cc.sh -optShared -optCompInst -g -O2  -no-pie -c simple.c -o simple.o
Debug: Using compiler-based instrumentation


Debug: Compiling with Instrumented Code
Executing> /usr/bin/gcc -g -O2 -no-pie -c simple.c -g -DPROFILING_ON -DTAU_GNU -DTAU_DOT_H_LESS_HEADERS -DTAU_LINUX_TIMERS -DTAU_LARGEFILE -D_LARGEFILE64_SOURCE -DT
AU_BFD -DHAVE_GNU_DEMANGLE -DHAVE_TR1_HASH_MAP -DTAU_SS_ALLOC_SUPPORT -DEBS_CLOCK_RES=1 -DTAU_STRSIGNAL_OK -DTAU_TRACK_LD_LOADER -DTAU_ELF_BFD -DTAU_DWARF -I/home/u
sers/khuck/src/tau2/x86_64/libdwarf-gcc/include -I/home/users/khuck/src/tau2/include -g -finstrument-functions -finstrument-functions-exclude-file-list=include,.h,.
hpp -o simple.o
Looking for file: simple.o

TAU_MAKEFILE=/storage/users/khuck/src/tau2/x86_64/lib/Makefile.tau-pdt tau_cc.sh -optShared -optCompInst -g -O2  -no-pie  simple.o -o simple
Debug: Using compiler-based instrumentation
Debug: Moving these libraries to the end of the link line:


Debug: Linking with TAU Options
Executing> /usr/bin/gcc -g -O2 -no-pie simple.o -o simple -L/home/users/khuck/src/tau2/x86_64/lib/shared-pdt -lTAU -Wl,-rpath,/home/users/khuck/src/tau2/x86_64/lib/
shared-pdt -L/home/users/khuck/src/tau2/x86_64/binutils-2.36/lib -L/home/users/khuck/src/tau2/x86_64/binutils-2.36/lib64 -Wl,-rpath,/home/users/khuck/src/tau2/x86_6
4/binutils-2.36/lib -Wl,-rpath,/home/users/khuck/src/tau2/x86_64/binutils-2.36/lib64 -lbfd -liberty -lz -ldl -Wl,--export-dynamic -lrt -lm -L/home/users/khuck/src/t
au2/x86_64/libdwarf-gcc/lib -Wl,-rpath,/home/users/khuck/src/tau2/x86_64/libdwarf-gcc/lib -ldwarf -lz -lelf -L/usr/lib/gcc/x86_64-linux-gnu/9/ -lstdc++ -lgcc_s -L/h
ome/users/khuck/src/tau2/x86_64/lib/static-pdt -Wl,-rpath,/home/users/khuck/src/tau2/x86_64/lib/shared-pdt -ldl -g

khuck@instinct:~/src/tau2/examples/simple$
```

Simple example

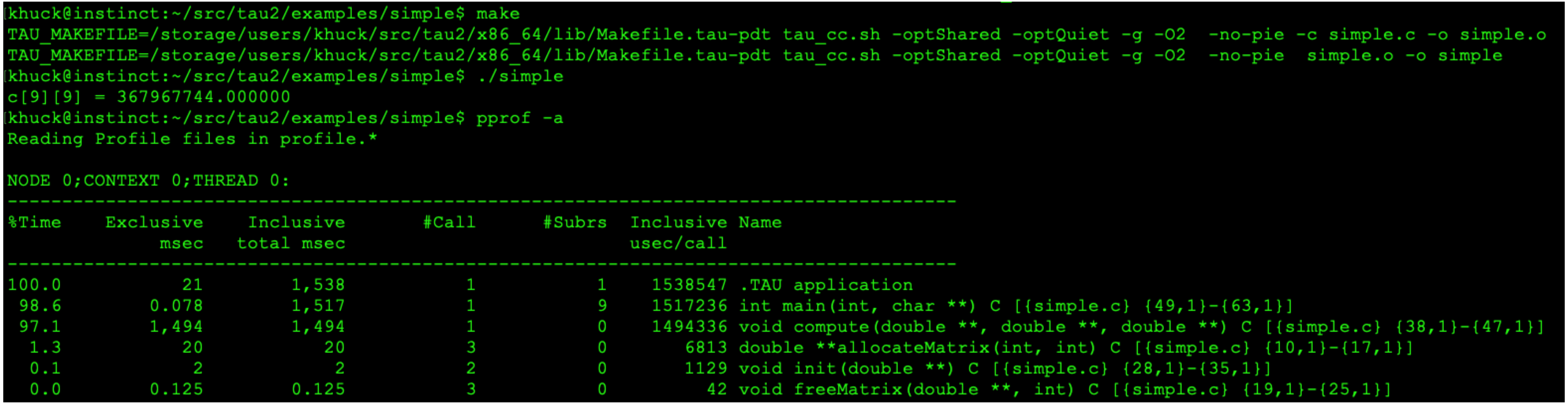


```c
/****************************************************************
 *   Matrix Multiply - C Version
 *   Demonstrates a matrix multiply using OpenMP.
 ****************************************************************/
#include <stdio.h>
#include <stdlib.h>

#define SIZE 1024

double** allocateMatrix(int rows, int cols) {
    int i;
    double **matrix = (double**)calloc(rows, (sizeof(double*)));
    for (i=0; i<rows; i++) {
        matrix[i] = (double*)calloc(cols, (sizeof(double)));
    }
    return matrix;
}

void freeMatrix(double** matrix, int rows, int cols) {
    int i;
    for (i=0; i<rows; i++) {
        free(matrix[i]);
    }
    free(matrix);
}

/* Initialize the matrix to something other than zero */
void init(double **a) {
    int i,j;
    for (i=0; i < SIZE; i++) {
        for(j=0; j < SIZE; j++) {
            a[i][j] = i+j+1;
        }
    }
}
```



```c

/* Perform matrix multiply */
void compute(double **a, double **b, double **c) {
    int i,j,k;
    for (i=0; i < SIZE; i++) {
        for(j=0; j < SIZE; j++) {
            for (k=0; k < SIZE; k++) {
                c[i][j] += a[i][k] * b[k][j];
            }
        }
    }
}

int main (int argc, char *argv[])
{
    double **a = allocateMatrix(SIZE, SIZE);
    double **b = allocateMatrix(SIZE, SIZE);
    double **c = allocateMatrix(SIZE, SIZE);
    init(a);
    init(b);
    compute(a, b, c);
    /* use the result */
    printf("c[9][9] = %f\n", c[9][9]);
    freeMatrix(a, SIZE);
    freeMatrix(b, SIZE);
    freeMatrix(c, SIZE);
    return 0;
}
```

PDT Instrumentation

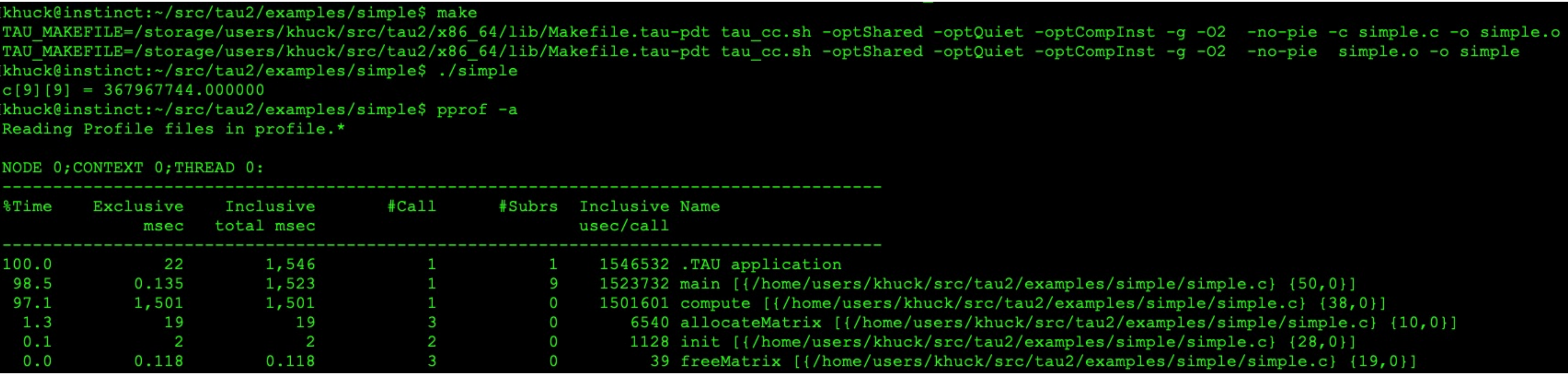


```
[khuck@instinct:~/src/tau2/examples/simple$ make
TAU_MAKEFILE=/storage/users/khuck/src/tau2/x86_64/lib/Makefile.tau-pdt tau_cc.sh -optShared -optQuiet -g -O2  -no-pie -c simple.c -o simple.o
TAU_MAKEFILE=/storage/users/khuck/src/tau2/x86_64/lib/Makefile.tau-pdt tau_cc.sh -optShared -optQuiet -g -O2  -no-pie  simple.o -o simple
[khuck@instinct:~/src/tau2/examples/simple$ ./simple
c[9][9] = 367967744.000000
[khuck@instinct:~/src/tau2/examples/simple$ pprof -a
Reading Profile files in profile.*

NODE 0;CONTEXT 0;THREAD 0:
---------------------------------------------------------------------------------------
%Time    Exclusive    Inclusive       #Call      #Subrs  Inclusive Name
              msec   total msec                          usec/call
---------------------------------------------------------------------------------------
100.0           21        1,538           1           1    1538547 .TAU application
 98.6        0.078        1,517           1           9    1517236 int main(int, char **) C [{simple.c} {49,1}-{63,1}]
 97.1        1,494        1,494           1           0    1494336 void compute(double **, double **, double **) C [{simple.c} {38,1}-{47,1}]
  1.3           20           20           3           0       6813 double **allocateMatrix(int, int) C [{simple.c} {10,1}-{17,1}]
  0.1            2            2           2           0       1129 void init(double **) C [{simple.c} {28,1}-{35,1}]
  0.0        0.125        0.125           3           0         42 void freeMatrix(double **, int) C [{simple.c} {19,1}-{25,1}]
```

Timer names have full signatures, start & end lines/columns

- all information available from parsing original source file with PDT

Compiler Instrumentation



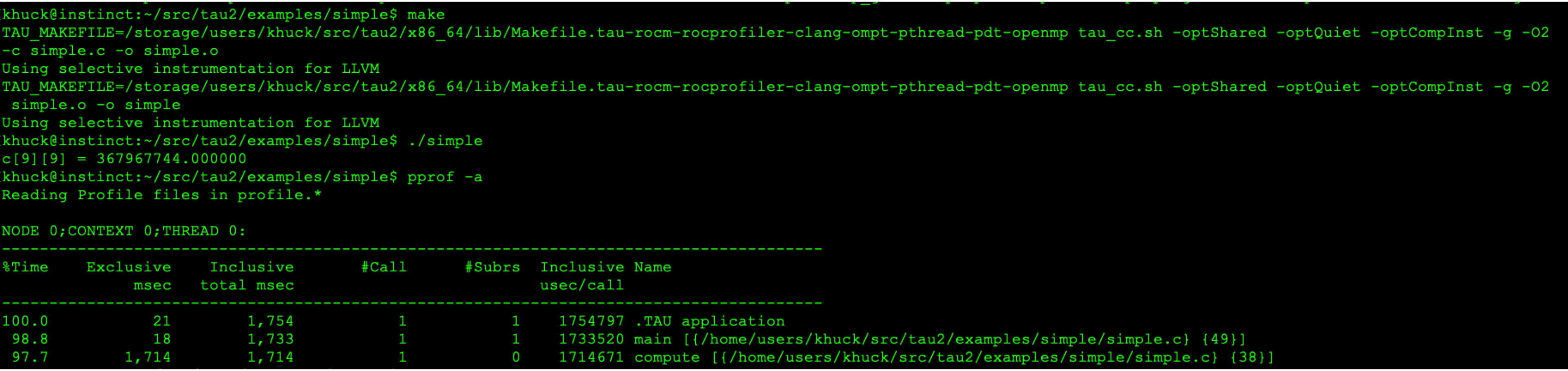
```
khuck@instinct:~/src/tau2/examples/simple$ make
TAU_MAKEFILE=/storage/users/khuck/src/tau2/x86_64/lib/Makefile.tau-pdt tau_cc.sh -optShared -optQuiet -optCompInst -g -O2  -no-pie -c simple.c -o simple.o
TAU_MAKEFILE=/storage/users/khuck/src/tau2/x86_64/lib/Makefile.tau-pdt tau_cc.sh -optShared -optQuiet -optCompInst -g -O2  -no-pie  simple.o -o simple
khuck@instinct:~/src/tau2/examples/simple$ ./simple
c[9][9] = 367967744.000000
khuck@instinct:~/src/tau2/examples/simple$ pprof -a
Reading Profile files in profile.*

NODE 0;CONTEXT 0;THREAD 0:
---------------------------------------------------------------------------------------
%Time    Exclusive    Inclusive       #Call      #Subrs  Inclusive Name
              msec   total msec                          usec/call
---------------------------------------------------------------------------------------
100.0           22        1,546           1           1    1546532 .TAU application
 98.5        0.135        1,523           1           9    1523732 main [{/home/users/khuck/src/tau2/examples/simple/simple.c} {50,0}]
 97.1        1,501        1,501           1           0    1501601 compute [{/home/users/khuck/src/tau2/examples/simple/simple.c} {38,0}]
  1.3           19           19           3           0       6540 allocateMatrix [{/home/users/khuck/src/tau2/examples/simple/simple.c} {10,0}]
  0.1            2            2           2           0       1128 init [{/home/users/khuck/src/tau2/examples/simple/simple.c} {28,0}]
  0.0        0.118        0.118           3           0         39 freeMatrix [{/home/users/khuck/src/tau2/examples/simple/simple.c} {19,0}]
```

Timer names have name only, start line only

- function entry/exit callback only has address of function and return address
- all source information retrieved during program execution with binutils (libbfd)

LLVM Plugin Instrumentation

Different TAU configuration with clang++/clang/flang



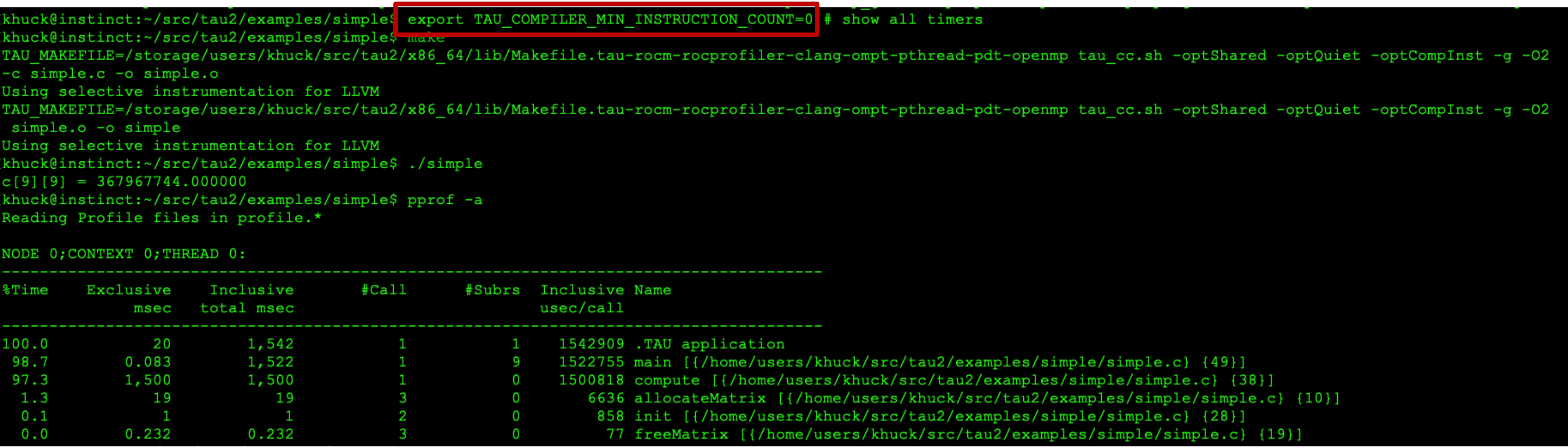
```
khuck@instinct:~/src/tau2/examples/simple$ make
TAU_MAKEFILE=/storage/users/khuck/src/tau2/x86_64/lib/Makefile.tau-rocm-rocprofiler-clang-ompt-pthread-pdt-openmp tau_cc.sh -optShared -optQuiet -optCompInst -g -O2
-c simple.c -o simple.o
Using selective instrumentation for LLVM
TAU_MAKEFILE=/storage/users/khuck/src/tau2/x86_64/lib/Makefile.tau-rocm-rocprofiler-clang-ompt-pthread-pdt-openmp tau_cc.sh -optShared -optQuiet -optCompInst -g -O2
 simple.o -o simple
Using selective instrumentation for LLVM
khuck@instinct:~/src/tau2/examples/simple$ ./simple
c[9][9] = 367967744.000000
khuck@instinct:~/src/tau2/examples/simple$ pprof -a
Reading Profile files in profile.*

NODE 0;CONTEXT 0;THREAD 0:
---------------------------------------------------------------------------------------
%Time    Exclusive    Inclusive       #Call      #Subrs  Inclusive Name
              msec   total msec                          usec/call
---------------------------------------------------------------------------------------
100.0           21        1,754           1           1    1754797 .TAU application
 98.8           18        1,733           1           1    1733520 main [{/home/users/khuck/src/tau2/examples/simple/simple.c} {49}]
 97.7        1,714        1,714           1           0    1714671 compute [{/home/users/khuck/src/tau2/examples/simple/simple.c} {38}]
```

Where are the other timers?...

TAU_COMPILER_MIN_INSTRUCTION_COUNT defaults to 50, so they were filtered out

LLVM Plugin Instrumentation

Different TAU configuration with clang++/clang/flang



```
khuck@instinct:~/src/tau2/examples/simple$ export TAU_COMPILER_MIN_INSTRUCTION_COUNT=0 # show all timers
khuck@instinct:~/src/tau2/examples/simple$ make
TAU_MAKEFILE=/storage/users/khuck/src/tau2/x86_64/lib/Makefile.tau-rocm-rocprofiler-clang-ompt-pthread-pdt-openmp tau_cc.sh -optShared -optQuiet -optCompInst -g -O2
-c simple.c -o simple.o
Using selective instrumentation for LLVM
TAU_MAKEFILE=/storage/users/khuck/src/tau2/x86_64/lib/Makefile.tau-rocm-rocprofiler-clang-ompt-pthread-pdt-openmp tau_cc.sh -optShared -optQuiet -optCompInst -g -O2
 simple.o -o simple
Using selective instrumentation for LLVM
khuck@instinct:~/src/tau2/examples/simple$ ./simple
c[9][9] = 367967744.000000
khuck@instinct:~/src/tau2/examples/simple$ pprof -a
Reading Profile files in profile.*

NODE 0;CONTEXT 0;THREAD 0:
---------------------------------------------------------------------------------------
%Time    Exclusive    Inclusive       #Call      #Subrs  Inclusive Name
              msec   total msec                          usec/call
---------------------------------------------------------------------------------------
100.0           20        1,542           1           1    1542909 .TAU application
 98.7        0.083        1,522           1           9    1522755 main [{/home/users/khuck/src/tau2/examples/simple/simple.c} {49}]
 97.3        1,500        1,500           1           0    1500818 compute [{/home/users/khuck/src/tau2/examples/simple/simple.c} {38}]
  1.3           19           19           3           0       6636 allocateMatrix [{/home/users/khuck/src/tau2/examples/simple/simple.c} {10}]
  0.1            1            1           2           0        858 init [{/home/users/khuck/src/tau2/examples/simple/simple.c} {28}]
  0.0        0.232        0.232           3           0         77 freeMatrix [{/home/users/khuck/src/tau2/examples/simple/simple.c} {19}]
```

TAU_COMPILER_MIN_INSTRUCTION_COUNT=0 disables filtering

Instrumentation pros/cons

Pros

- Simple to implement
- Works universally
- Instruments everything – no blind spots
- Selective instrumentation available (by file or function name, or instruction count)

Cons

- Instruments *everything*
- Potentially high overhead – especially with C++
- Changes program behavior
- Potentially interferes with optimizations

Sampling

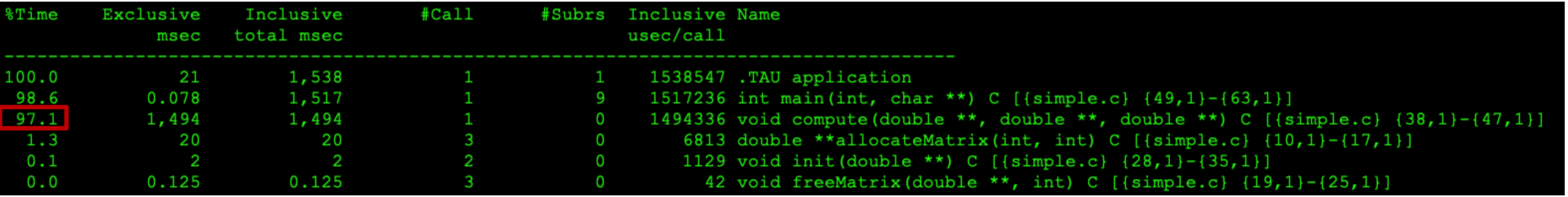
- Run the application with `tau_exec -ebs`
  - Preloads the TAU library, instantiates a signal handler and periodic interrupt to process the signal
  - The signal handler will record the current instruction pointer, all requested metrics, and optionally unwind the callstack
  - At the end of execution, all addresses are resolved to symbols in the application using binutils/libdwarf
- Some things that help:
  - For best support, build application with debug (-g) - all other optimizations are fine

Using TAU's Runtime Preloading Tool: tau_exec

- <mpirun> `tau_exec -T` <config> <options> <executable>
- `tau-config --list-matching` <mpi/serial> will show available configs
- Preload a wrapper that intercepts the runtime calls and substitutes with another (using dlsym() or weak symbol replacement)
  - 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)

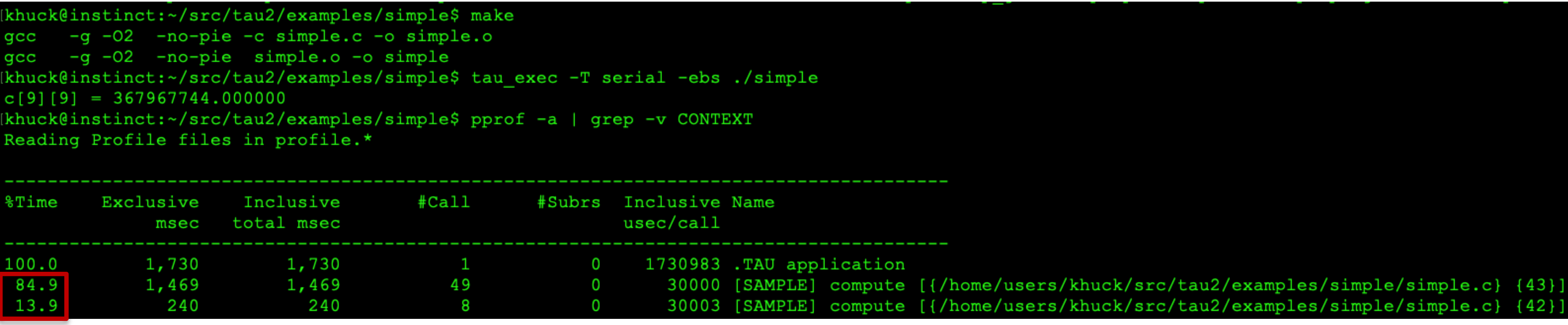
Sampled Measurement

Previous instrumentation example:



```
%Time    Exclusive    Inclusive       #Call      #Subrs  Inclusive Name
              msec   total msec                          usec/call
---------------------------------------------------------------------------------------
100.0           21        1,538           1           1    1538547 .TAU application
 98.6        0.078        1,517           1           9    1517236 int main(int, char **) C [{simple.c} {49,1}-{63,1}]
 97.1        1,494        1,494           1           0    1494336 void compute(double **, double **, double **) C [{simple.c} {38,1}-{47,1}]
  1.3           20           20           3           0       6813 double **allocateMatrix(int, int) C [{simple.c} {10,1}-{17,1}]
  0.1            2            2           2           0       1129 void init(double **) C [{simple.c} {28,1}-{35,1}]
  0.0        0.125        0.125           3           0         42 void freeMatrix(double **, int) C [{simple.c} {19,1}-{25,1}]
```

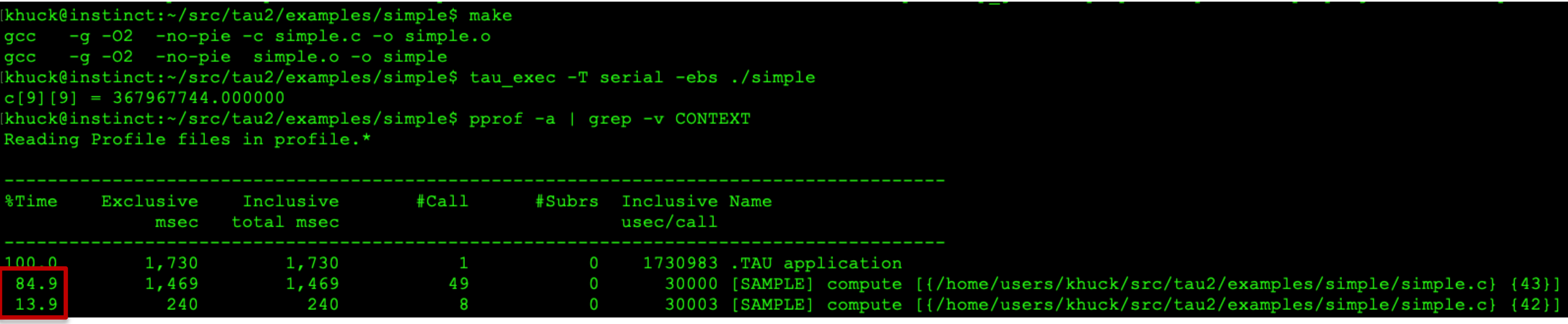
Sampling example:



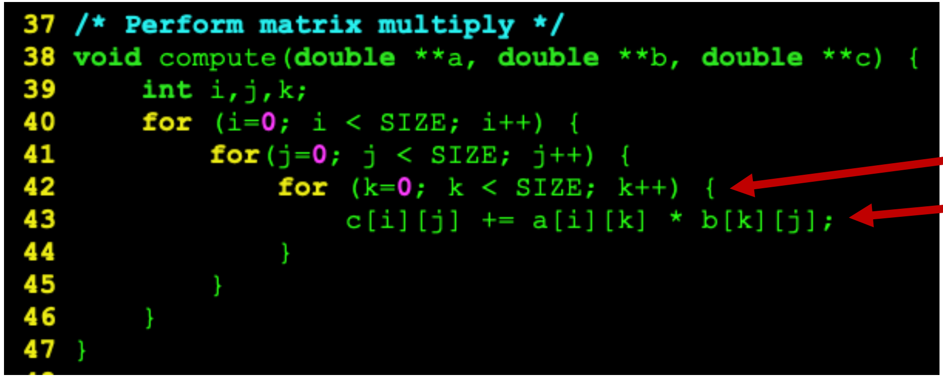
```
khuck@instinct:~/src/tau2/examples/simple$ make
gcc    -g -O2  -no-pie -c simple.c -o simple.o
gcc    -g -O2  -no-pie  simple.o -o simple
khuck@instinct:~/src/tau2/examples/simple$ tau_exec -T serial -ebs ./simple
c[9][9] = 367967744.000000
khuck@instinct:~/src/tau2/examples/simple$ pprof -a | grep -v CONTEXT
Reading Profile files in profile.*

---------------------------------------------------------------------------------------
%Time    Exclusive    Inclusive       #Call      #Subrs  Inclusive Name
              msec   total msec                          usec/call
---------------------------------------------------------------------------------------
100.0        1,730        1,730           1           0    1730983 .TAU application
 84.9        1,469        1,469          49           0      30000 [SAMPLE] compute [{/home/users/khuck/src/tau2/examples/simple/simple.c} {43}]
 13.9          240          240           8           0      30003 [SAMPLE] compute [{/home/users/khuck/src/tau2/examples/simple/simple.c} {42}]
```

Both *more* and *less* information at the same time…

Sampled Measurement



```
[khuck@instinct:~/src/tau2/examples/simple$ make
gcc   -g -O2  -no-pie -c simple.c -o simple.o
gcc   -g -O2  -no-pie  simple.o -o simple
[khuck@instinct:~/src/tau2/examples/simple$ tau_exec -T serial -ebs ./simple
c[9][9] = 367967744.000000
[khuck@instinct:~/src/tau2/examples/simple$ pprof -a | grep -v CONTEXT
Reading Profile files in profile.*

---------------------------------------------------------------------------------------
%Time    Exclusive    Inclusive       #Call      #Subrs  Inclusive Name
              msec   total msec                          usec/call
---------------------------------------------------------------------------------------
100.0        1,730        1,730           1           0    1730983 .TAU application
 84.9        1,469        1,469          49           0      30000 [SAMPLE] compute [{/home/users/khuck/src/tau2/examples/simple/simple.c} {43}]
 13.9          240          240           8           0      30003 [SAMPLE] compute [{/home/users/khuck/src/tau2/examples/simple/simple.c} {42}]
```

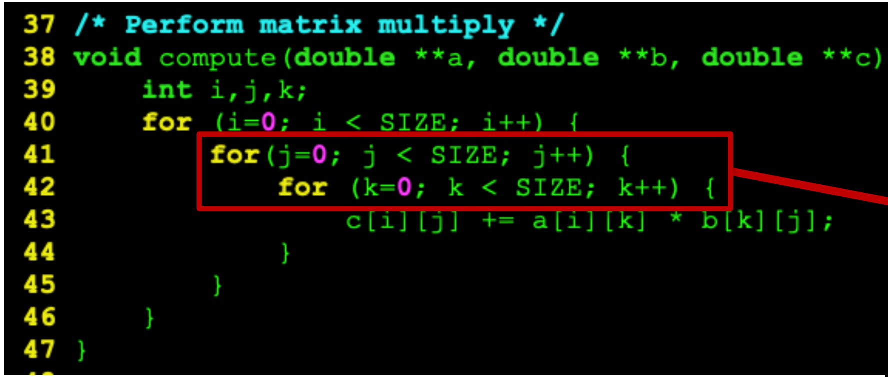


```
37 /* Perform matrix multiply */
38 void compute(double **a, double **b, double **c) {
39     int i,j,k;
40     for (i=0; i < SIZE; i++) {
41         for(j=0; j < SIZE; j++) {
42             for (k=0; k < SIZE; k++) {
43                 c[i][j] += a[i][k] * b[k][j];
44             }
45         }
46     }
47 }
```

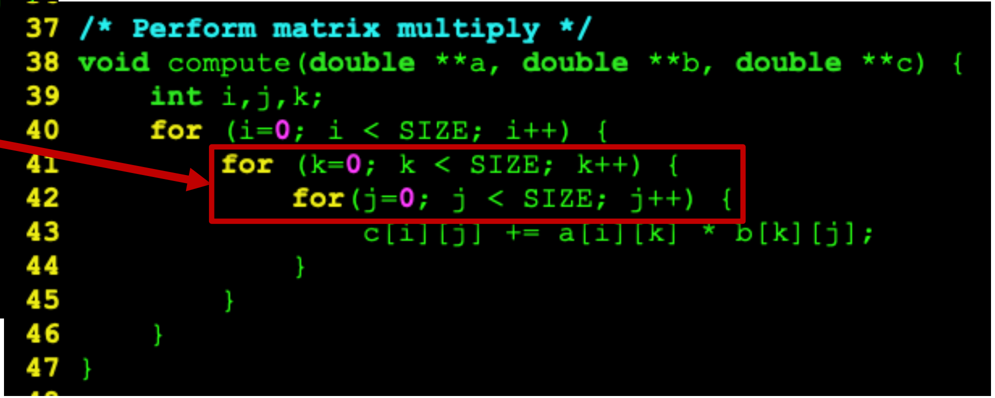
14% spent here

85% spent here

Simple Transformation – loop inversion

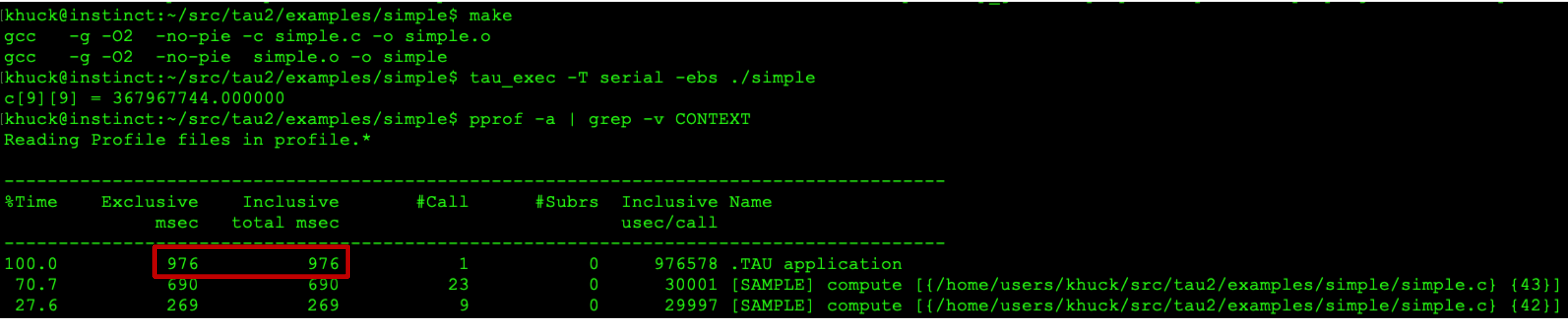


```
37 /* Perform matrix multiply */
38 void compute(double **a, double **b, double **c)
39      int i,j,k;
40      for (i=0; i < SIZE; i++) {
41          for(j=0; j < SIZE; j++) {
42              for (k=0; k < SIZE; k++) {
43                  c[i][j] += a[i][k] * b[k][j];
44              }
45          }
46      }
47 }
```



```
37 /* Perform matrix multiply */
38 void compute(double **a, double **b, double **c) {
39      int i,j,k;
40      for (i=0; i < SIZE; i++) {
41          for (k=0; k < SIZE; k++) {
42              for(j=0; j < SIZE; j++) {
43                  c[i][j] += a[i][k] * b[k][j];
44              }
45          }
46      }
47 }
```

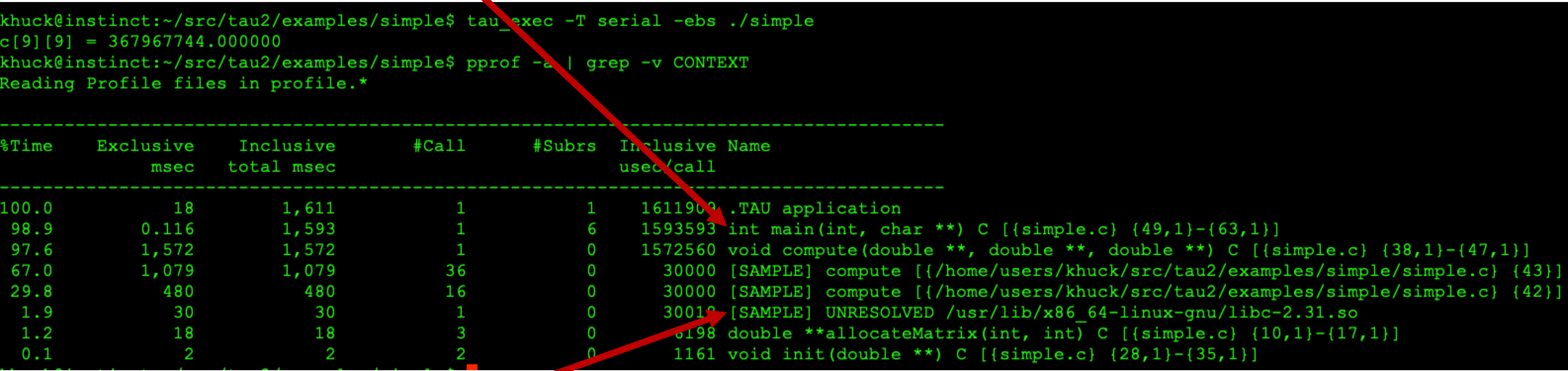
Reduced from 1.73 seconds



```
khuck@instinct:~/src/tau2/examples/simple$ make
gcc   -g -O2  -no-pie -c simple.c -o simple.o
gcc   -g -O2  -no-pie  simple.o -o simple
khuck@instinct:~/src/tau2/examples/simple$ tau_exec -T serial -ebs ./simple
c[9][9] = 367967744.000000
khuck@instinct:~/src/tau2/examples/simple$ pprof -a | grep -v CONTEXT
Reading Profile files in profile.*

---------------------------------------------------------------------------------------
%Time    Exclusive    Inclusive       #Call      #Subrs  Inclusive Name
              msec   total msec                          usec/call
---------------------------------------------------------------------------------------
100.0          976          976           1           0     976578 .TAU application
 70.7          690          690          23           0      30001 [SAMPLE] compute [{/home/users/khuck/src/tau2/examples/simple/simple.c} {43}]
 27.6          269          269           9           0      29997 [SAMPLE] compute [{/home/users/khuck/src/tau2/examples/simple/simple.c} {42}]
```

Both together!

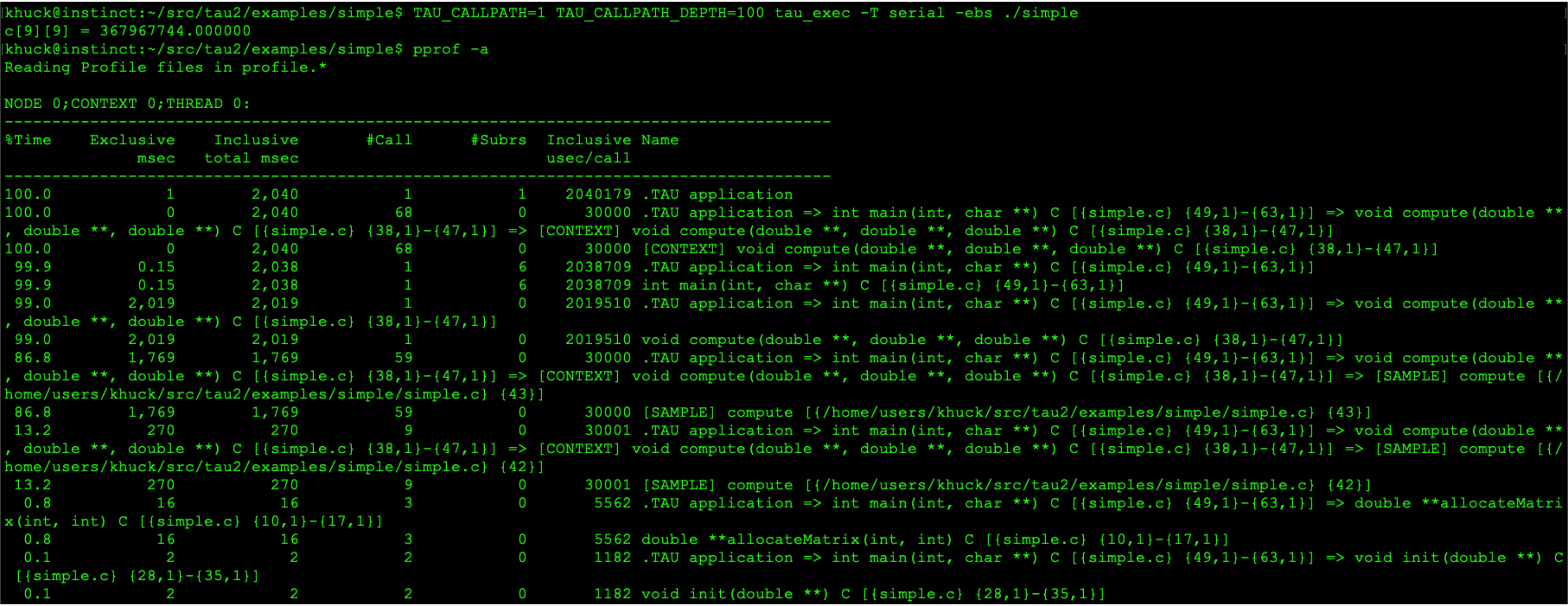
Timers



```
khuck@instinct:~/src/tau2/examples/simple$ tau_exec -T serial -ebs ./simple
c[9][9] = 367967744.000000
khuck@instinct:~/src/tau2/examples/simple$ pprof -a | grep -v CONTEXT
Reading Profile files in profile.*

---------------------------------------------------------------------------------------
%Time    Exclusive    Inclusive       #Call      #Subrs  Inclusive Name
              msec   total msec                          usec/call
---------------------------------------------------------------------------------------
100.0           18        1,611           1           1    1611909 .TAU application
 98.9        0.116        1,593           1           6    1593593 int main(int, char **) C [{simple.c} {49,1}-{63,1}]
 97.6        1,572        1,572           1           0    1572560 void compute(double **, double **, double **) C [{simple.c} {38,1}-{47,1}]
 67.0        1,079        1,079          36           0      30000 [SAMPLE] compute [{/home/users/khuck/src/tau2/examples/simple/simple.c} {43}]
 29.8          480          480          16           0      30000 [SAMPLE] compute [{/home/users/khuck/src/tau2/examples/simple/simple.c} {42}]
  1.9           30           30           1           0      30018 [SAMPLE] UNRESOLVED /usr/lib/x86_64-linux-gnu/libc-2.31.so
  1.2           18           18           3           0       6198 double **allocateMatrix(int, int) C [{simple.c} {10,1}-{17,1}]
  0.1            2            2           2           0       1161 void init(double **) C [{simple.c} {28,1}-{35,1}]
```

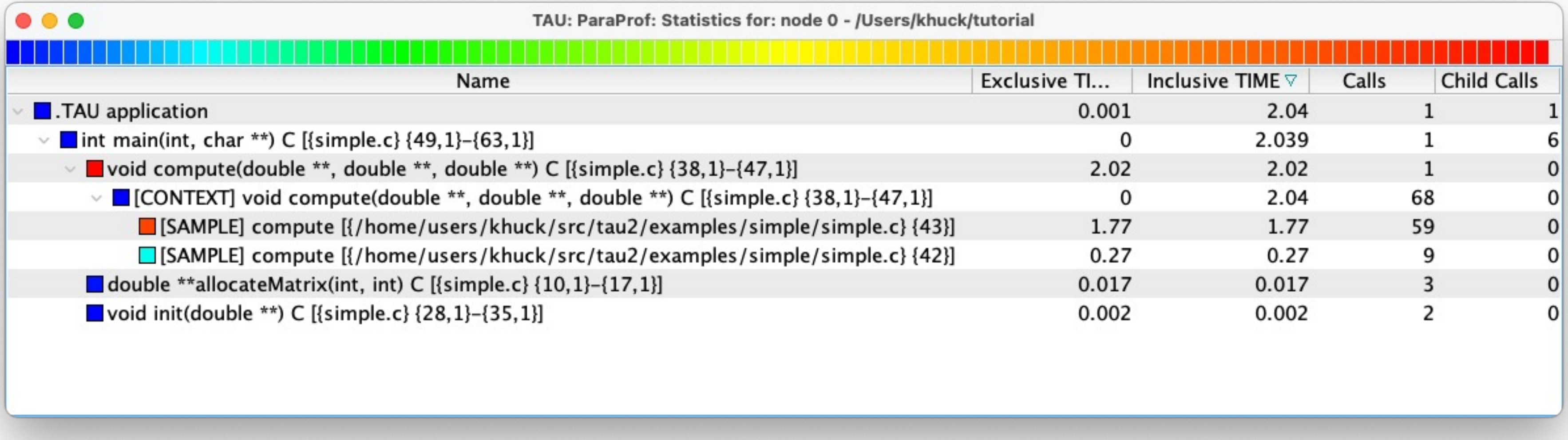
Samples

…with callpath profiling



```
khuck@instinct:~/src/tau2/examples/simple$ TAU_CALLPATH=1 TAU_CALLPATH_DEPTH=100 tau_exec -T serial -ebs ./simple
c[9][9] = 367967744.000000
khuck@instinct:~/src/tau2/examples/simple$ pprof -a
Reading Profile files in profile.*

NODE 0;CONTEXT 0;THREAD 0:
---------------------------------------------------------------------------------------
%Time    Exclusive    Inclusive       #Call      #Subrs  Inclusive Name
              msec   total msec                          usec/call
---------------------------------------------------------------------------------------
100.0            1        2,040           1           1    2040179 .TAU application
100.0            0        2,040          68           0      30000 .TAU application => int main(int, char **) C [{simple.c} {49,1}-{63,1}] => void compute(double **
, double **, double **) C [{simple.c} {38,1}-{47,1}] => [CONTEXT] void compute(double **, double **, double **) C [{simple.c} {38,1}-{47,1}]
100.0            0        2,040          68           0      30000 [CONTEXT] void compute(double **, double **, double **) C [{simple.c} {38,1}-{47,1}]
 99.9         0.15        2,038           1           6    2038709 .TAU application => int main(int, char **) C [{simple.c} {49,1}-{63,1}]
 99.9         0.15        2,038           1           6    2038709 int main(int, char **) C [{simple.c} {49,1}-{63,1}]
 99.0        2,019        2,019           1           0    2019510 .TAU application => int main(int, char **) C [{simple.c} {49,1}-{63,1}] => void compute(double **
, double **, double **) C [{simple.c} {38,1}-{47,1}]
 99.0        2,019        2,019           1           0    2019510 void compute(double **, double **, double **) C [{simple.c} {38,1}-{47,1}]
 86.8        1,769        1,769          59           0      30000 .TAU application => int main(int, char **) C [{simple.c} {49,1}-{63,1}] => void compute(double **
, double **, double **) C [{simple.c} {38,1}-{47,1}] => [CONTEXT] void compute(double **, double **, double **) C [{simple.c} {38,1}-{47,1}] => [SAMPLE] compute [{/
home/users/khuck/src/tau2/examples/simple/simple.c} {43}]
 86.8        1,769        1,769          59           0      30000 [SAMPLE] compute [{/home/users/khuck/src/tau2/examples/simple/simple.c} {43}]
 13.2          270          270           9           0      30001 .TAU application => int main(int, char **) C [{simple.c} {49,1}-{63,1}] => void compute(double **
, double **, double **) C [{simple.c} {38,1}-{47,1}] => [CONTEXT] void compute(double **, double **, double **) C [{simple.c} {38,1}-{47,1}] => [SAMPLE] compute [{/
home/users/khuck/src/tau2/examples/simple/simple.c} {42}]
 13.2          270          270           9           0      30001 [SAMPLE] compute [{/home/users/khuck/src/tau2/examples/simple/simple.c} {42}]
  0.8           16           16           3           0       5562 .TAU application => int main(int, char **) C [{simple.c} {49,1}-{63,1}] => double **allocateMatri
x(int, int) C [{simple.c} {10,1}-{17,1}]
  0.8           16           16           3           0       5562 double **allocateMatrix(int, int) C [{simple.c} {10,1}-{17,1}]
  0.1            2            2           2           0       1182 .TAU application => int main(int, char **) C [{simple.c} {49,1}-{63,1}] => void init(double **) C
 [{simple.c} {28,1}-{35,1}]
  0.1            2            2           2           0       1182 void init(double **) C [{simple.c} {28,1}-{35,1}]
```

…easier to view in ParaProf



TAU: ParaProf: Statistics for: node 0 - /Users/khuck/tutorial

Name	Exclusive TI...	Inclusive TIME ▽	Calls	Child Calls
🟦 .TAU application	0.001	2.04	1	1
🟦 int main(int, char **) C [{simple.c} {49,1}-{63,1}]	0	2.039	1	6
🟥 void compute(double **, double **, double **) C [{simple.c} {38,1}-{47,1}]	2.02	2.02	1	0
🟦 [CONTEXT] void compute(double **, double **, double **) C [{simple.c} {38,1}-{47,1}]	0	2.04	68	0
🟧 [SAMPLE] compute [{/home/users/khuck/src/tau2/examples/simple/simple.c} {43}]	1.77	1.77	59	0
🟦 [SAMPLE] compute [{/home/users/khuck/src/tau2/examples/simple/simple.c} {42}]	0.27	0.27	9	0
🟦 double **allocateMatrix(int, int) C [{simple.c} {10,1}-{17,1}]	0.017	0.017	3	0
🟦 void init(double **) C [{simple.c} {28,1}-{35,1}]	0.002	0.002	2	0

Other measurement support

- Many programming models provide “hooks” for tools
- Often, instrumentation isn’t necessary!
  - **MPI**, SHMEM, Charm++
  - Pthreads, **OpenMP**, Kokkos
  - CUDA, **HIP/ROCm**, OneAPI, OpenACC, OpenCL, OpenMP offload
  - Python
  - Wrappers: POSIX, Chapel, UPC, memory, ARMCI, GASNet…
  - Java

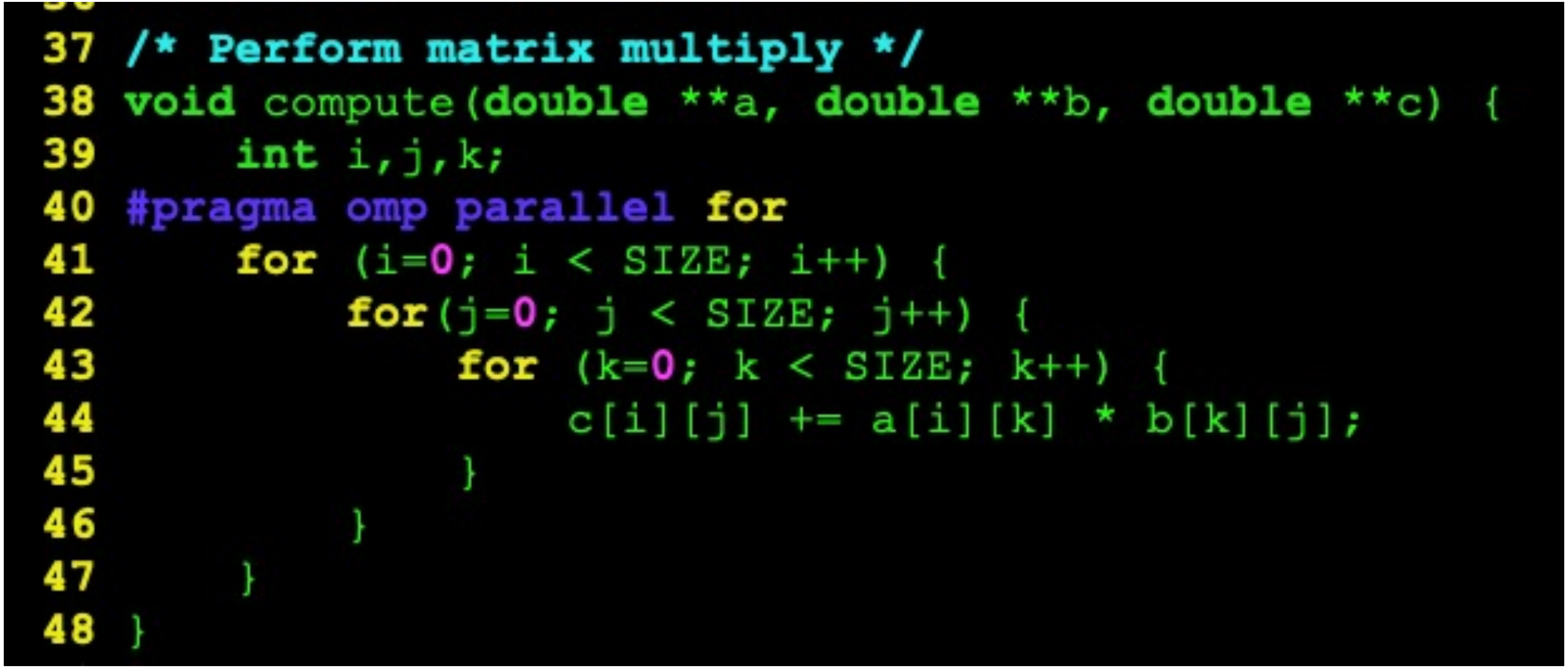
Other TAU features

- Binary instrumentation
  - Dyninst, MAQAO, PIN
- Hardware counter support
  - PAPI, LIKWID
- Tracing support (native or converters)
  - Vampir (OTF2), Perfetto (JSON), Jumpshot (SLOG2), …
- Plugins
  - OS/HW monitoring, ADIOS2, SOS, Mochi, SQLite3, …

OpenMP

- https://www.openmp.org
- Pragma-based language extension to facilitate threading
- OpenMP 5.0 standard includes OpenMP Tools (OMPT/OMPD) specification for providing callbacks from the runtime to performance/debugging tools
- Provided by Intel, LLVM, IBM compilers
- GCC can use drop-in replacement (LLVM 8.0 runtime)
- TAU provides OPARI legacy support (when using PDT)

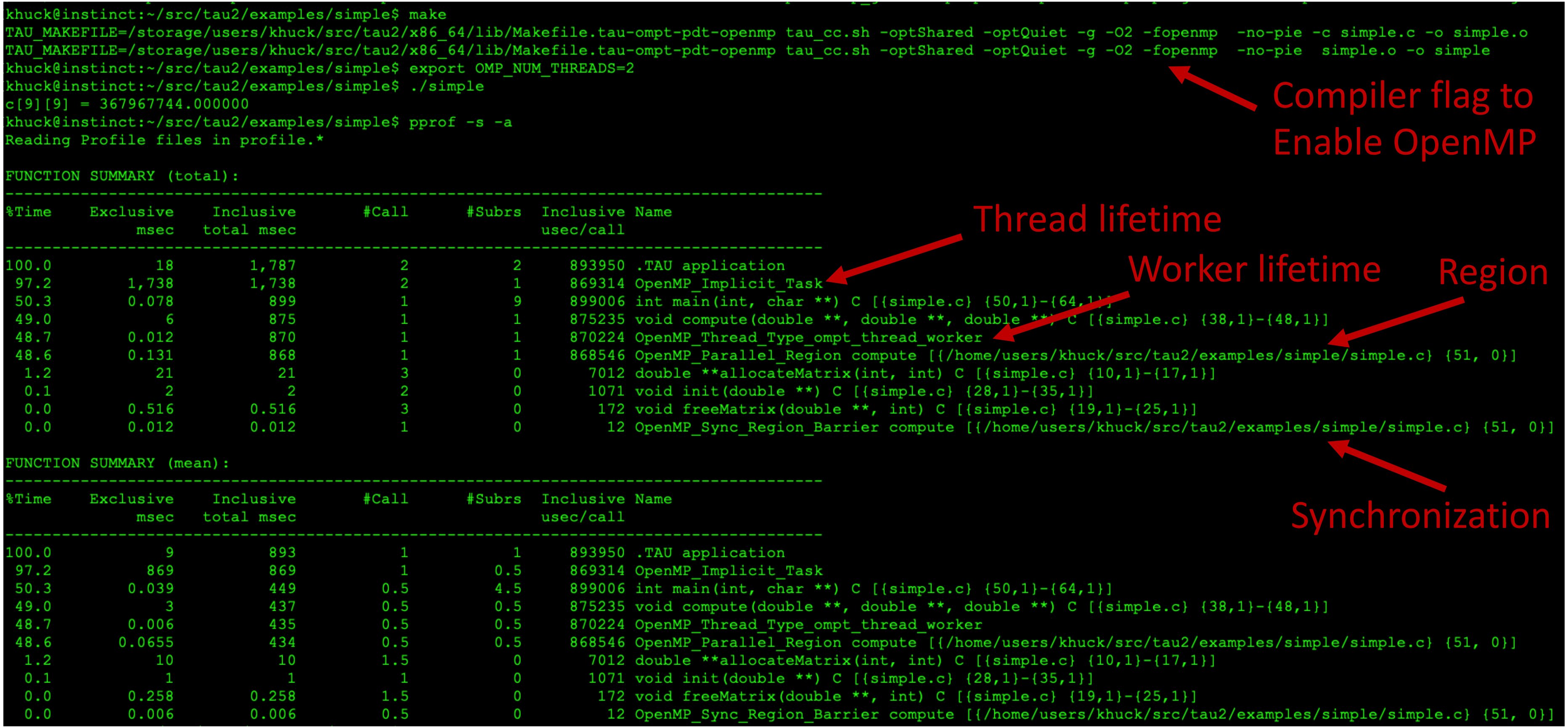
Adding OpenMP



```c
/* Perform matrix multiply */
void compute(double **a, double **b, double **c) {
    int i,j,k;
#pragma omp parallel for
    for (i=0; i < SIZE; i++) {
        for(j=0; j < SIZE; j++) {
            for (k=0; k < SIZE; k++) {
                c[i][j] += a[i][k] * b[k][j];
            }
        }
    }
}
```

If OMP_NUM_THREADS=4, SIZE=1024, then iteration space will be split into 4 of chunk size 256 each – 4x speedup

Compiling, Running, Reporting



```
khuck@instinct:~/src/tau2/examples/simple$ make
TAU_MAKEFILE=/storage/users/khuck/src/tau2/x86_64/lib/Makefile.tau-ompt-pdt-openmp tau_cc.sh -optShared -optQuiet -g -O2 -fopenmp  -no-pie -c simple.c -o simple.o
TAU_MAKEFILE=/storage/users/khuck/src/tau2/x86_64/lib/Makefile.tau-ompt-pdt-openmp tau_cc.sh -optShared -optQuiet -g -O2 -fopenmp  -no-pie  simple.o -o simple
khuck@instinct:~/src/tau2/examples/simple$ export OMP_NUM_THREADS=2
khuck@instinct:~/src/tau2/examples/simple$ ./simple
c[9][9] = 367967744.000000
khuck@instinct:~/src/tau2/examples/simple$ pprof -s -a
Reading Profile files in profile.*

FUNCTION SUMMARY (total):
-----------------------------------------------------------------------------------------
%Time    Exclusive    Inclusive       #Call      #Subrs  Inclusive Name
              msec   total msec                          usec/call
-----------------------------------------------------------------------------------------
100.0           18        1,787           2           2     893950 .TAU application
 97.2        1,738        1,738           2           1     869314 OpenMP_Implicit_Task
 50.3        0.078          899           1           9     899006 int main(int, char **) C [{simple.c} {50,1}-{64,1}]
 49.0            6          875           1           1     875235 void compute(double **, double **, double **) C [{simple.c} {38,1}-{48,1}]
 48.7        0.012          870           1           1     870224 OpenMP_Thread_Type_ompt_thread_worker
 48.6        0.131          868           1           1     868546 OpenMP_Parallel_Region compute [{/home/users/khuck/src/tau2/examples/simple/simple.c} {51, 0}]
  1.2           21           21           3           0       7012 double **allocateMatrix(int, int) C [{simple.c} {10,1}-{17,1}]
  0.1            2            2           2           0       1071 void init(double **) C [{simple.c} {28,1}-{35,1}]
  0.0        0.516        0.516           3           0        172 void freeMatrix(double **, int) C [{simple.c} {19,1}-{25,1}]
  0.0        0.012        0.012           1           0         12 OpenMP_Sync_Region_Barrier compute [{/home/users/khuck/src/tau2/examples/simple/simple.c} {51, 0}]

FUNCTION SUMMARY (mean):
-----------------------------------------------------------------------------------------
%Time    Exclusive    Inclusive       #Call      #Subrs  Inclusive Name
              msec   total msec                          usec/call
-----------------------------------------------------------------------------------------
100.0            9          893           1           1     893950 .TAU application
 97.2          869          869           1         0.5     869314 OpenMP_Implicit_Task
 50.3        0.039          449         0.5         4.5     899006 int main(int, char **) C [{simple.c} {50,1}-{64,1}]
 49.0            3          437         0.5         0.5     875235 void compute(double **, double **, double **) C [{simple.c} {38,1}-{48,1}]
 48.7        0.006          435         0.5         0.5     870224 OpenMP_Thread_Type_ompt_thread_worker
 48.6       0.0655          434         0.5         0.5     868546 OpenMP_Parallel_Region compute [{/home/users/khuck/src/tau2/examples/simple/simple.c} {51, 0}]
  1.2           10           10         1.5           0       7012 double **allocateMatrix(int, int) C [{simple.c} {10,1}-{17,1}]
  0.1            1            1           1           0       1071 void init(double **) C [{simple.c} {28,1}-{35,1}]
  0.0        0.258        0.258         1.5           0        172 void freeMatrix(double **, int) C [{simple.c} {19,1}-{25,1}]
  0.0        0.006        0.006         0.5           0         12 OpenMP_Sync_Region_Barrier compute [{/home/users/khuck/src/tau2/examples/simple/simple.c} {51, 0}]
```

Compiler flag to Enable OpenMP

Thread lifetime

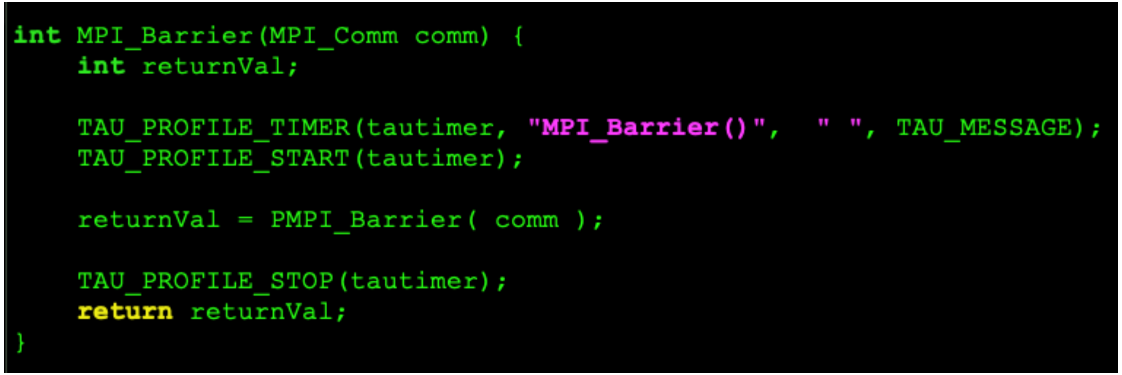
Worker lifetime

Region

Synchronization

MPI Support

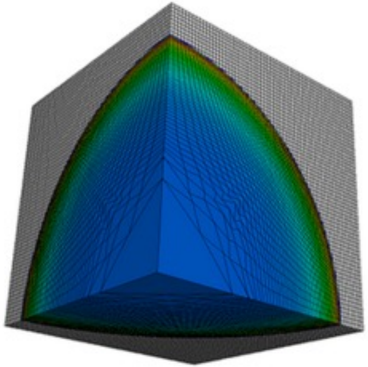
- MPI standard includes tool support
  - MPI_* functions are thin, weak wrappers around PMPI_* API
  - Tools create their own wrappers to replace them and intercept MPI calls
  - Tool library is preloaded or linked ahead of MPI library(ies)
  - Example:



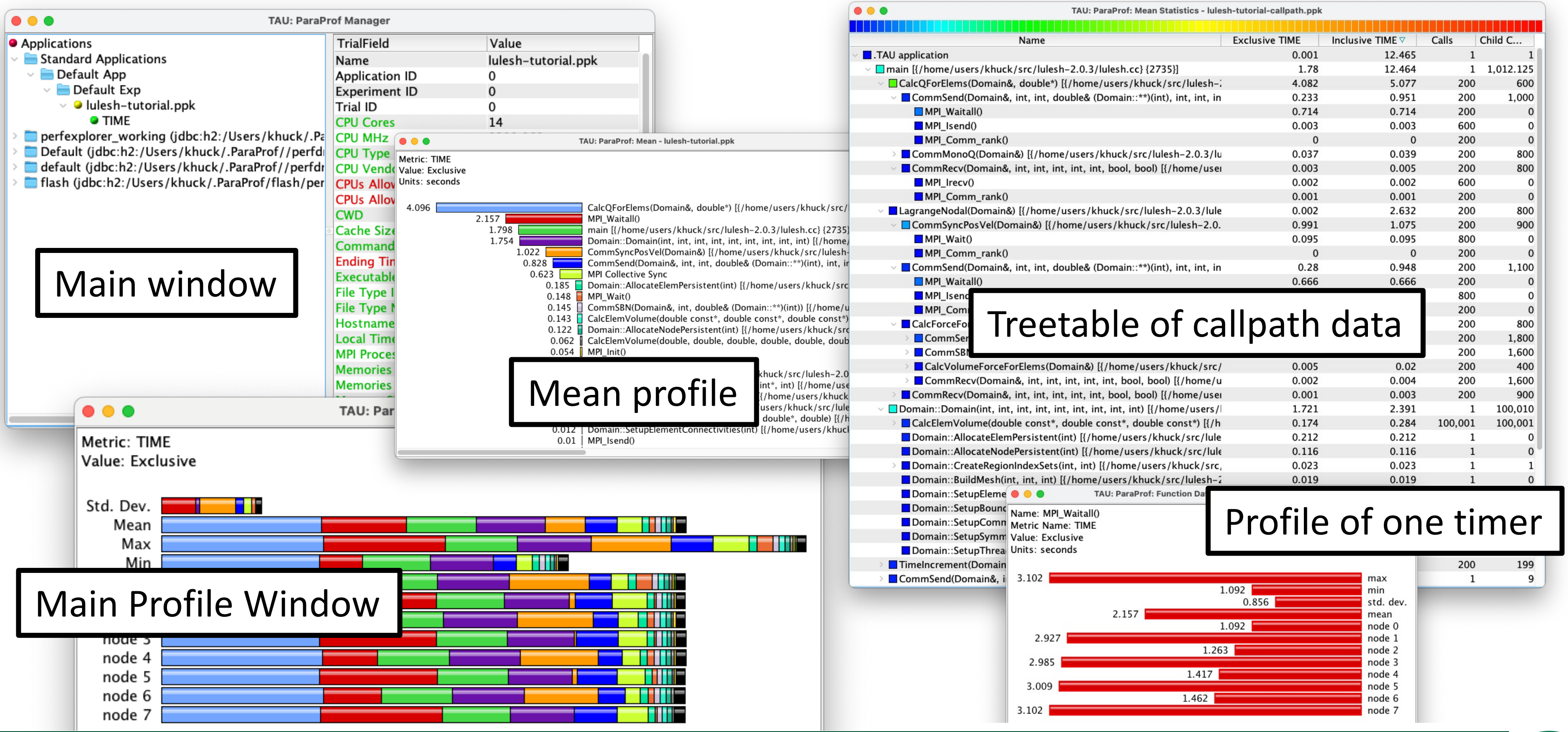
```
int MPI_Barrier(MPI_Comm comm) {
    int returnVal;

    TAU_PROFILE_TIMER(tautimer, "MPI_Barrier()",  " ", TAU_MESSAGE);
    TAU_PROFILE_START(tautimer);

    returnVal = PMPI_Barrier( comm );

    TAU_PROFILE_STOP(tautimer);
    return returnVal;
}
```

MPI example – Lulesh

- Lulesh 2.0.3 https://asc.llnl.gov/codes/proxy-apps/lulesh
- “The Shock Hydrodynamics Challenge Problem was originally defined and implemented by LLNL as one of five challenge problems in the DARPA UHPC program and has since become a widely studied proxy application in DOE co-design efforts for exascale.”
- C++, Serial, OpenMP, MPI
- CUDA, OpenACC, OpenCL, other models



Lulesh Profile - ParaProf



TAU: ParaProf Manager

- Applications
  - Standard Applications
    - Default App
      - Default Exp
        - lulesh-tutorial.ppk
          - TIME
  - perfexplorer_working (jdbc:h2:/Users/khuck/.Pa
  - Default (jdbc:h2:/Users/khuck/.ParaProf//perfd
  - default (jdbc:h2:/Users/khuck/.ParaProf//perfd
  - flash (jdbc:h2:/Users/khuck/.ParaProf/flash/per

TrialField	Value
Name	lulesh-tutorial.ppk
Application ID	0
Experiment ID	0
Trial ID	0
CPU Cores	14
CPU MHz	
CPU Type	
CPU Vend	
CPUs Allov	
CPUs Allov	
CWD	
Cache Size	
Command	
Ending Tin	
Executable	
File Type I	
File Type N	
Hostname	
Local Time	
MPI Proces	
Memories	
Memories	

Main window

TAU: ParaProf: Mean - lulesh-tutorial.ppk

Metric: TIME
Value: Exclusive
Units: seconds

```
4.096  CalcQForElems(Domain&, double*) [{/home/users/khuck/src/
2.157  MPI_Waitall()
1.798  main [{/home/users/khuck/src/lulesh-2.0.3/lulesh.cc} {2735}
1.754  Domain::Domain(int, int, int, int, int, int, int, int, int) [{/home/
1.022  CommSyncPosVel(Domain&) [{/home/users/khuck/src/lulesh-
0.828  CommSend(Domain&, int, int, double& (Domain::**)(int), int, in
0.623  MPI Collective Sync
0.185  Domain::AllocateElemPersistent(int) [{/home/users/khuck/src
0.148  MPI_Wait()
0.145  CommSBN(Domain&, int, double& (Domain::**)(int)) [{/home/u
0.143  CalcElemVolume(double const*, double const*, double const*)
0.122  Domain::AllocateNodePersistent(int) [{/home/users/khuck/src
0.062  CalcElemVolume(double, double, double, double, double, doub
0.054  MPI_Init()
       khuck/src/lulesh-2.0
       int*, int) [{/home/use
       [{/home/users/khuck
       users/khuck/src/lule
       double*, double) [{/h
0.012  Domain::SetupElementConnectivities(int) [{/home/users/khuc
0.01   MPI_Isend()
```

Mean profile

TAU: ParaProf: Mean Statistics - lulesh-tutorial-callpath.ppk

Name	Exclusive TIME	Inclusive TIME ▽	Calls	Child C...
.TAU application	0.001	12.465	1	1
main [{/home/users/khuck/src/lulesh-2.0.3/lulesh.cc} {2735}]	1.78	12.464	1	1,012.125
CalcQForElems(Domain&, double*) [{/home/users/khuck/src/lulesh-:	4.082	5.077	200	600
CommSend(Domain&, int, int, double& (Domain::**)(int), int, int, in	0.233	0.951	200	1,000
MPI_Waitall()	0.714	0.714	200	0
MPI_Isend()	0.003	0.003	600	0
MPI_Comm_rank()	0	0	200	0
CommMonoQ(Domain&) [{/home/users/khuck/src/lulesh-2.0.3/lu	0.037	0.039	200	800
CommRecv(Domain&, int, int, int, int, int, bool, bool) [{/home/usei	0.003	0.005	200	800
MPI_Irecv()	0.002	0.002	600	0
MPI_Comm_rank()	0.001	0.001	200	0
LagrangeNodal(Domain&) [{/home/users/khuck/src/lulesh-2.0.3/lule	0.002	2.632	200	800
CommSyncPosVel(Domain&) [{/home/users/khuck/src/lulesh-2.0.	0.991	1.075	200	900
MPI_Wait()	0.095	0.095	800	0
MPI_Comm_rank()	0	0	200	0
CommSend(Domain&, int, int, double& (Domain::**)(int), int, int, in	0.28	0.948	200	1,100
MPI_Waitall()	0.666	0.666	200	0
MPI_Isend			800	0
MPI_Com			200	0
CalcForceFo			200	800
CommSer			200	1,800
CommSBI			200	1,600
CalcVolumeForceForElems(Domain&) [{/home/users/khuck/src/	0.005	0.02	200	400
CommRecv(Domain&, int, int, int, int, int, bool, bool) [{/home/u	0.002	0.004	200	1,600
CommRecv(Domain&, int, int, int, int, int, bool, bool) [{/home/usei	0.001	0.003	200	900
Domain::Domain(int, int, int, int, int, int, int, int, int) [{/home/users/l	1.721	2.391	1	100,010
CalcElemVolume(double const*, double const*, double const*) [{/h	0.174	0.284	100,001	100,001
Domain::AllocateElemPersistent(int) [{/home/users/khuck/src/lule	0.212	0.212	1	0
Domain::AllocateNodePersistent(int) [{/home/users/khuck/src/lule	0.116	0.116	1	0
Domain::CreateRegionIndexSets(int, int) [{/home/users/khuck/src,	0.023	0.023	1	1
Domain::BuildMesh(int, int, int) [{/home/users/khuck/src/lulesh-2	0.019	0.019	1	0
Domain::SetupEleme				
Domain::SetupBounc				
Domain::SetupComm				
Domain::SetupSymm				
Domain::SetupThrea				
TimeIncrement(Domain			200	199
CommSend(Domain&, i			1	9

Treetable of callpath data

TAU: ParaProf: Function Da

Name: MPI_Waitall()
Metric Name: TIME
Value: Exclusive
Units: seconds

```
3.102  max
1.092  min
0.856  std. dev.
2.157  mean
1.092  node 0
2.927  node 1
1.263  node 2
2.985  node 3
1.417  node 4
3.009  node 5
1.462  node 6
3.102  node 7
```

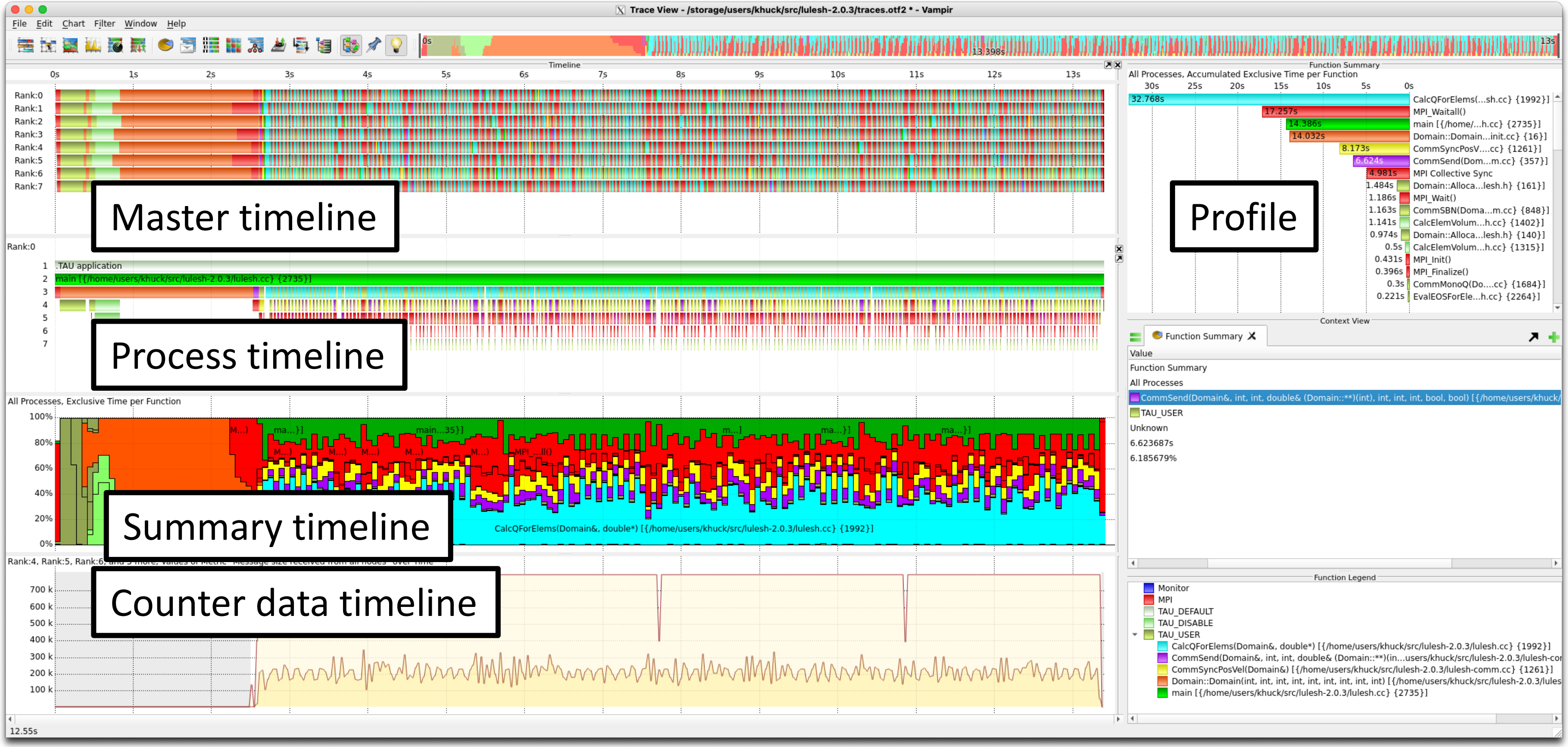
Profile of one timer

TAU: Par

Metric: TIME
Value: Exclusive

Main Profile Window

Lulesh Trace – Vampir


Trace View - /storage/users/khuck/src/lulesh-2.0.3/traces.otf2 * - Vampir
File
Edit
Chart
Filter
Window
Help
0s
13.398s
13s
Timeline
0s
1s
2s
3s
4s
5s
6s
7s
8s
9s
10s
11s
12s
13s
Rank:0
Rank:1
Rank:2
Rank:3
Rank:4
Rank:5
Rank:6
Rank:7
Rank:0
1
2
3
4
5
6
7
.TAU application
main [{/home/users/khuck/src/lulesh-2.0.3/lulesh.cc} {2735}]
All Processes, Exclusive Time per Function
100%
80%
60%
40%
20%
0%
M...)
ma...}]
main...35}]
m...]
ma...}]
ma...}]
MPI_...ll()
CalcQForElems(Domain&, double*) [{/home/users/khuck/src/lulesh-2.0.3/lulesh.cc} {1992}]
Rank:4, Rank:5, Rank:6, and 3 more, Values of Metric "Message size received from all nodes" over Time
700 k
600 k
500 k
400 k
300 k
200 k
100 k
12.55s
Function Summary
All Processes, Accumulated Exclusive Time per Function
30s
25s
20s
15s
10s
5s
0s
Context View
Function Summary
Function Legend

Master timeline

Process timeline

Summary timeline

Counter data timeline

Profile

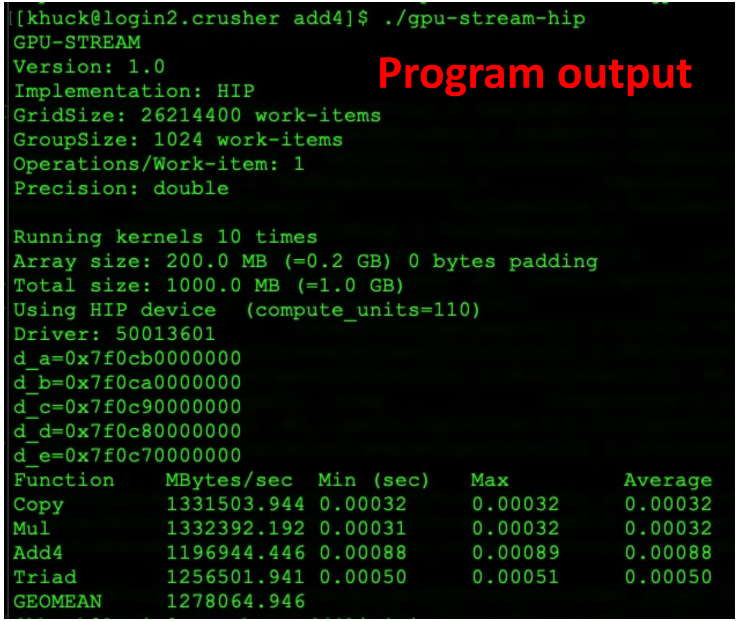
32.768s	CalcQForElems(...sh.cc} {1992}]
17.257s	MPI_Waitall()
14.386s	main [{/home/...h.cc} {2735}]
14.032s	Domain::Domain...init.cc} {16}]
8.173s	CommSyncPosV....cc} {1261}]
6.624s	CommSend(Dom...m.cc} {357}]
4.981s	MPI Collective Sync
1.484s	Domain::Alloca...lesh.h} {161}]
1.186s	MPI_Wait()
1.163s	CommSBN(Doma...m.cc} {848}]
1.141s	CalcElemVolum...h.cc} {1402}]
0.974s	Domain::Alloca...lesh.h} {140}]
0.5s	CalcElemVolum...h.cc} {1315}]
0.431s	MPI_Init()
0.396s	MPI_Finalize()
0.3s	CommMonoQ(Do....cc} {1684}]
0.221s	EvalEOSForEle...h.cc} {2264}]

Value
Function Summary
All Processes
CommSend(Domain&, int, int, double& (Domain::**)(int), int, int, int, bool, bool) [{/home/users/khuck/
TAU_USER
Unknown
6.623687s
6.185679%

- 🟦 Monitor
- 🟥 MPI
- TAU_DEFAULT
- TAU_DISABLE
- TAU_USER
  - CalcQForElems(Domain&, double*) [{/home/users/khuck/src/lulesh-2.0.3/lulesh.cc} {1992}]
  - CommSend(Domain&, int, int, double& (Domain::**)(in...users/khuck/src/lulesh-2.0.3/lulesh-cor
  - CommSyncPosVel(Domain&) [{/home/users/khuck/src/lulesh-2.0.3/lulesh-comm.cc} {1261}]
  - Domain::Domain(int, int, int, int, int, int, int, int, int) [{/home/users/khuck/src/lulesh-2.0.3/lules
  - main [{/home/users/khuck/src/lulesh-2.0.3/lulesh.cc} {2735}]

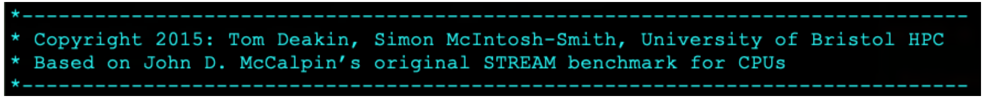
Measuring HIP kernel performance

- Hip-stream – small program with 4+ kernels

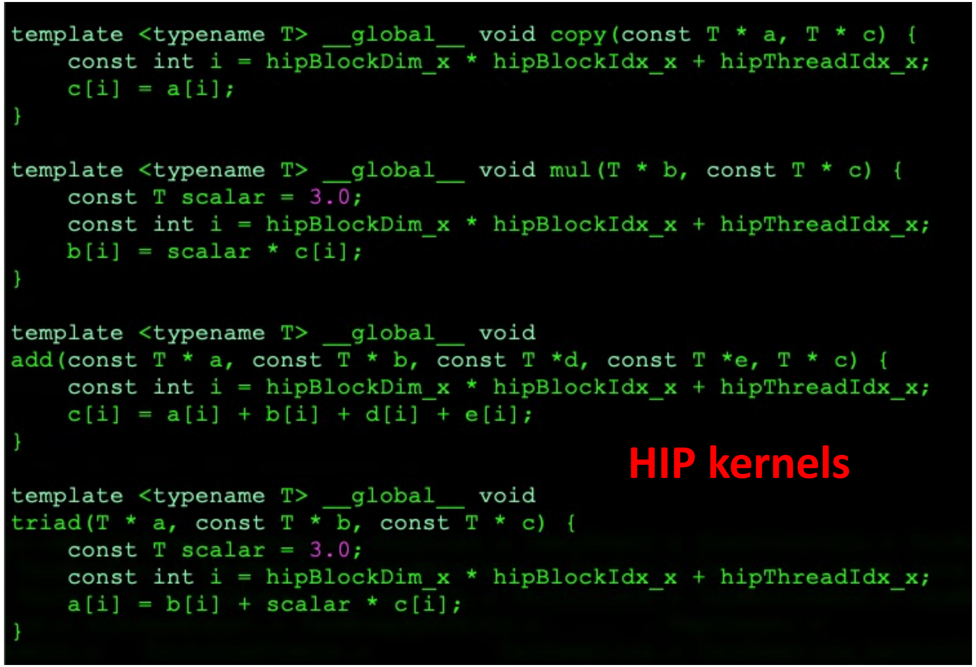


Program output

```
[khuck@login2.crusher add4]$ ./gpu-stream-hip
GPU-STREAM
Version: 1.0
Implementation: HIP
GridSize: 26214400 work-items
GroupSize: 1024 work-items
Operations/Work-item: 1
Precision: double

Running kernels 10 times
Array size: 200.0 MB (=0.2 GB) 0 bytes padding
Total size: 1000.0 MB (=1.0 GB)
Using HIP device  (compute_units=110)
Driver: 50013601
d_a=0x7f0cb0000000
d_b=0x7f0ca0000000
d_c=0x7f0c90000000
d_d=0x7f0c80000000
d_e=0x7f0c70000000
Function     MBytes/sec  Min (sec)    Max         Average
Copy         1331503.944 0.00032      0.00032     0.00032
Mul          1332392.192 0.00031      0.00032     0.00032
Add4         1196944.446 0.00088      0.00089     0.00088
Triad        1256501.941 0.00050      0.00051     0.00050
GEOMEAN      1278064.946
```



```
*-----------------------------------------------------------------------
* Copyright 2015: Tom Deakin, Simon McIntosh-Smith, University of Bristol HPC
* Based on John D. McCalpin's original STREAM benchmark for CPUs
*-----------------------------------------------------------------------
```



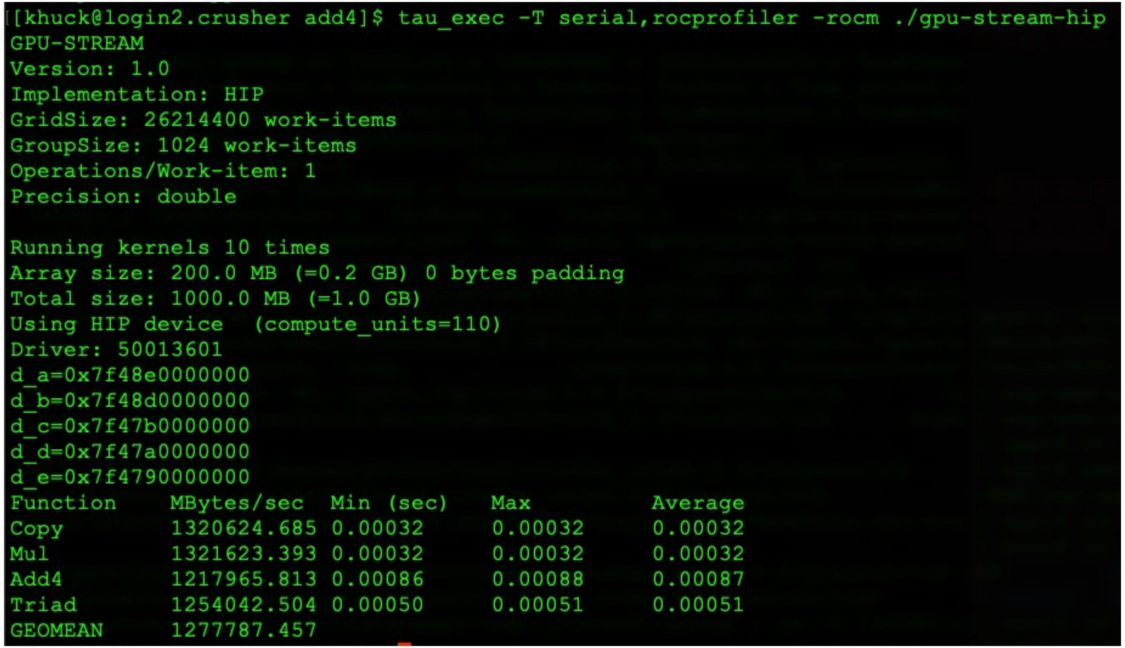
```
template <typename T> __global__ void copy(const T * a, T * c) {
    const int i = hipBlockDim_x * hipBlockIdx_x + hipThreadIdx_x;
    c[i] = a[i];
}

template <typename T> __global__ void mul(T * b, const T * c) {
    const T scalar = 3.0;
    const int i = hipBlockDim_x * hipBlockIdx_x + hipThreadIdx_x;
    b[i] = scalar * c[i];
}

template <typename T> __global__ void
add(const T * a, const T * b, const T *d, const T *e, T * c) {
    const int i = hipBlockDim_x * hipBlockIdx_x + hipThreadIdx_x;
    c[i] = a[i] + b[i] + d[i] + e[i];
}

template <typename T> __global__ void
triad(T * a, const T * b, const T * c) {
    const T scalar = 3.0;
    const int i = hipBlockDim_x * hipBlockIdx_x + hipThreadIdx_x;
    a[i] = b[i] + scalar * c[i];
}
```

HIP kernels

Measuring HIP kernel performance

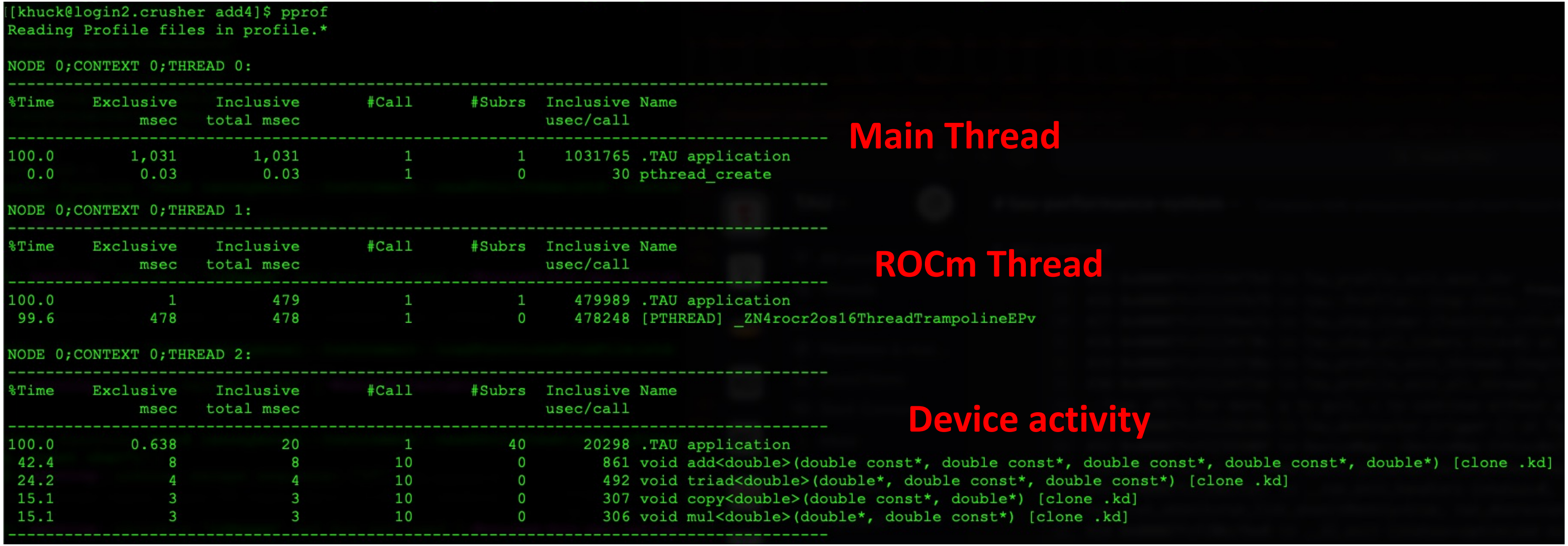
- Just add **`tau_exec`** and arguments to the command (between srun/mpirun and application when applicable)
- **`tau-config`** shows available configs

"use serial,rocprofiler configuration with HIP/ROCm support enabled"



```
[[khuck@login2.crusher add4]$ tau_exec -T serial,rocprofiler -rocm ./gpu-stream-hip
GPU-STREAM
Version: 1.0
Implementation: HIP
GridSize: 26214400 work-items
GroupSize: 1024 work-items
Operations/Work-item: 1
Precision: double

Running kernels 10 times
Array size: 200.0 MB (=0.2 GB) 0 bytes padding
Total size: 1000.0 MB (=1.0 GB)
Using HIP device  (compute_units=110)
Driver: 50013601
d_a=0x7f48e0000000
d_b=0x7f48d0000000
d_c=0x7f47b0000000
d_d=0x7f47a0000000
d_e=0x7f4790000000
Function    MBytes/sec  Min (sec)   Max         Average
Copy        1320624.685 0.00032     0.00032     0.00032
Mul         1321623.393 0.00032     0.00032     0.00032
Add4        1217965.813 0.00086     0.00088     0.00087
Triad       1254042.504 0.00050     0.00051     0.00051
GEOMEAN     1277787.457
```

Pprof output, timers



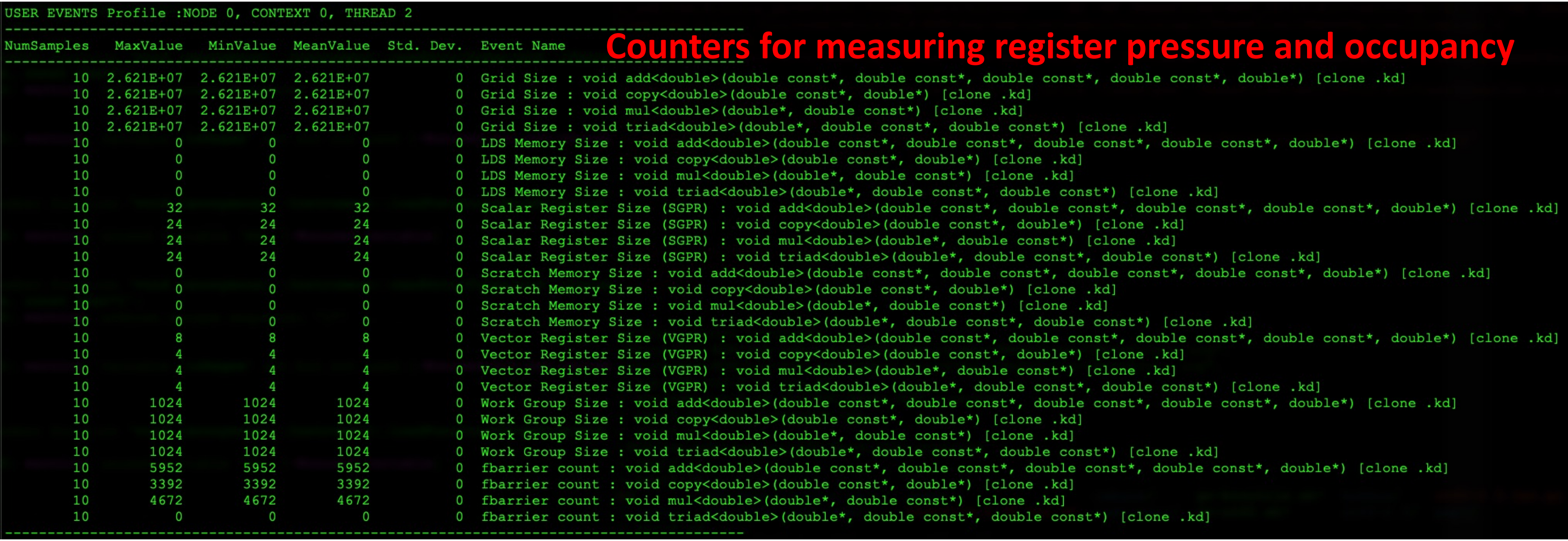
```
[khuck@login2.crusher add4]$ pprof
Reading Profile files in profile.*

NODE 0;CONTEXT 0;THREAD 0:
---------------------------------------------------------------------------------------
%Time    Exclusive    Inclusive       #Call      #Subrs  Inclusive Name
              msec   total msec                          usec/call
---------------------------------------------------------------------------------------
100.0        1,031        1,031           1           1    1031765 .TAU application
  0.0         0.03         0.03           1           0         30 pthread_create

NODE 0;CONTEXT 0;THREAD 1:
---------------------------------------------------------------------------------------
%Time    Exclusive    Inclusive       #Call      #Subrs  Inclusive Name
              msec   total msec                          usec/call
---------------------------------------------------------------------------------------
100.0            1          479           1           1     479989 .TAU application
 99.6          478          478           1           0     478248 [PTHREAD] _ZN4rocr2os16ThreadTrampolineEPv

NODE 0;CONTEXT 0;THREAD 2:
---------------------------------------------------------------------------------------
%Time    Exclusive    Inclusive       #Call      #Subrs  Inclusive Name
              msec   total msec                          usec/call
---------------------------------------------------------------------------------------
100.0        0.638           20           1          40      20298 .TAU application
 42.4            8            8          10           0        861 void add<double>(double const*, double const*, double const*, double const*, double*) [clone .kd]
 24.2            4            4          10           0        492 void triad<double>(double*, double const*, double const*) [clone .kd]
 15.1            3            3          10           0        307 void copy<double>(double const*, double*) [clone .kd]
 15.1            3            3          10           0        306 void mul<double>(double*, double const*) [clone .kd]
---------------------------------------------------------------------------------------
```

Main Thread

ROCm Thread

Device activity

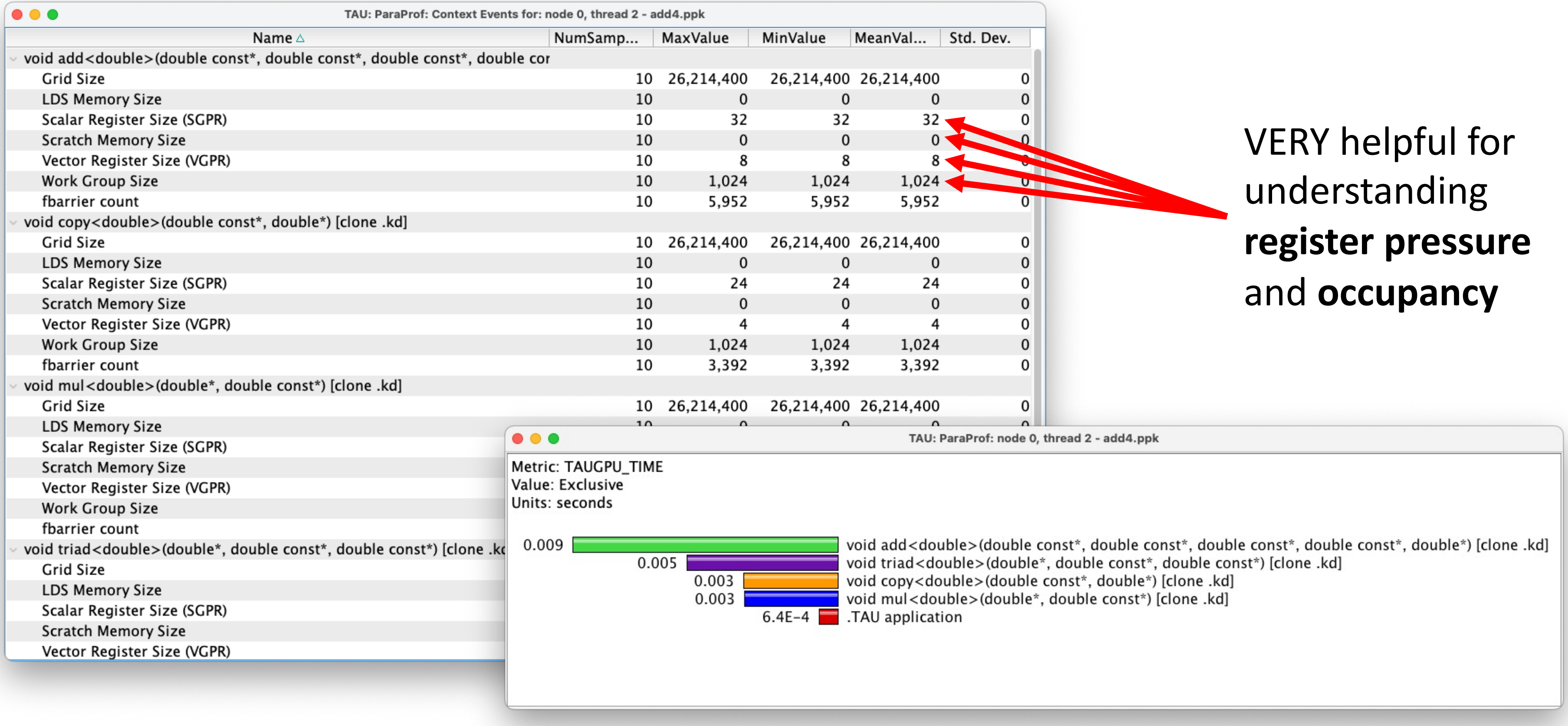
Pprof output, counters



Counters for measuring register pressure and occupancy

```
USER EVENTS Profile :NODE 0, CONTEXT 0, THREAD 2
---------------------------------------------------------------------------------------
NumSamples   MaxValue   MinValue  MeanValue  Std. Dev.  Event Name
---------------------------------------------------------------------------------------
        10  2.621E+07  2.621E+07  2.621E+07          0  Grid Size : void add<double>(double const*, double const*, double const*, double const*, double*) [clone .kd]
        10  2.621E+07  2.621E+07  2.621E+07          0  Grid Size : void copy<double>(double const*, double*) [clone .kd]
        10  2.621E+07  2.621E+07  2.621E+07          0  Grid Size : void mul<double>(double*, double const*) [clone .kd]
        10  2.621E+07  2.621E+07  2.621E+07          0  Grid Size : void triad<double>(double*, double const*, double const*) [clone .kd]
        10          0          0          0          0  LDS Memory Size : void add<double>(double const*, double const*, double const*, double const*, double*) [clone .kd]
        10          0          0          0          0  LDS Memory Size : void copy<double>(double const*, double*) [clone .kd]
        10          0          0          0          0  LDS Memory Size : void mul<double>(double*, double const*) [clone .kd]
        10          0          0          0          0  LDS Memory Size : void triad<double>(double*, double const*, double const*) [clone .kd]
        10         32         32         32          0  Scalar Register Size (SGPR) : void add<double>(double const*, double const*, double const*, double const*, double*) [clone .kd]
        10         24         24         24          0  Scalar Register Size (SGPR) : void copy<double>(double const*, double*) [clone .kd]
        10         24         24         24          0  Scalar Register Size (SGPR) : void mul<double>(double*, double const*) [clone .kd]
        10         24         24         24          0  Scalar Register Size (SGPR) : void triad<double>(double*, double const*, double const*) [clone .kd]
        10          0          0          0          0  Scratch Memory Size : void add<double>(double const*, double const*, double const*, double const*, double*) [clone .kd]
        10          0          0          0          0  Scratch Memory Size : void copy<double>(double const*, double*) [clone .kd]
        10          0          0          0          0  Scratch Memory Size : void mul<double>(double*, double const*) [clone .kd]
        10          0          0          0          0  Scratch Memory Size : void triad<double>(double*, double const*, double const*) [clone .kd]
        10          8          8          8          0  Vector Register Size (VGPR) : void add<double>(double const*, double const*, double const*, double const*, double*) [clone .kd]
        10          4          4          4          0  Vector Register Size (VGPR) : void copy<double>(double const*, double*) [clone .kd]
        10          4          4          4          0  Vector Register Size (VGPR) : void mul<double>(double*, double const*) [clone .kd]
        10          4          4          4          0  Vector Register Size (VGPR) : void triad<double>(double*, double const*, double const*) [clone .kd]
        10       1024       1024       1024          0  Work Group Size : void add<double>(double const*, double const*, double const*, double const*, double*) [clone .kd]
        10       1024       1024       1024          0  Work Group Size : void copy<double>(double const*, double*) [clone .kd]
        10       1024       1024       1024          0  Work Group Size : void mul<double>(double*, double const*) [clone .kd]
        10       1024       1024       1024          0  Work Group Size : void triad<double>(double*, double const*, double const*) [clone .kd]
        10       5952       5952       5952          0  fbarrier count : void add<double>(double const*, double const*, double const*, double const*, double*) [clone .kd]
        10       3392       3392       3392          0  fbarrier count : void copy<double>(double const*, double*) [clone .kd]
        10       4672       4672       4672          0  fbarrier count : void mul<double>(double*, double const*) [clone .kd]
        10          0          0          0          0  fbarrier count : void triad<double>(double*, double const*, double const*) [clone .kd]
---------------------------------------------------------------------------------------
```

ParaProf view of same data



TAU: ParaProf: Context Events for: node 0, thread 2 - add4.ppk

Name △	NumSamp...	MaxValue	MinValue	MeanVal...	Std. Dev.
void add<double>(double const*, double const*, double const*, double cor					
Grid Size	10	26,214,400	26,214,400	26,214,400	0
LDS Memory Size	10	0	0	0	0
Scalar Register Size (SGPR)	10	32	32	32	0
Scratch Memory Size	10	0	0	0	0
Vector Register Size (VGPR)	10	8	8	8	0
Work Group Size	10	1,024	1,024	1,024	0
fbarrier count	10	5,952	5,952	5,952	0
void copy<double>(double const*, double*) [clone .kd]					
Grid Size	10	26,214,400	26,214,400	26,214,400	0
LDS Memory Size	10	0	0	0	0
Scalar Register Size (SGPR)	10	24	24	24	0
Scratch Memory Size	10	0	0	0	0
Vector Register Size (VGPR)	10	4	4	4	0
Work Group Size	10	1,024	1,024	1,024	0
fbarrier count	10	3,392	3,392	3,392	0
void mul<double>(double*, double const*) [clone .kd]					
Grid Size	10	26,214,400	26,214,400	26,214,400	0
LDS Memory Size	10	0	0	0	0
Scalar Register Size (SGPR)					
Scratch Memory Size					
Vector Register Size (VGPR)					
Work Group Size					
fbarrier count					
void triad<double>(double*, double const*, double const*) [clone .kc					
Grid Size					
LDS Memory Size					
Scalar Register Size (SGPR)					
Scratch Memory Size					
Vector Register Size (VGPR)					

TAU: ParaProf: node 0, thread 2 - add4.ppk

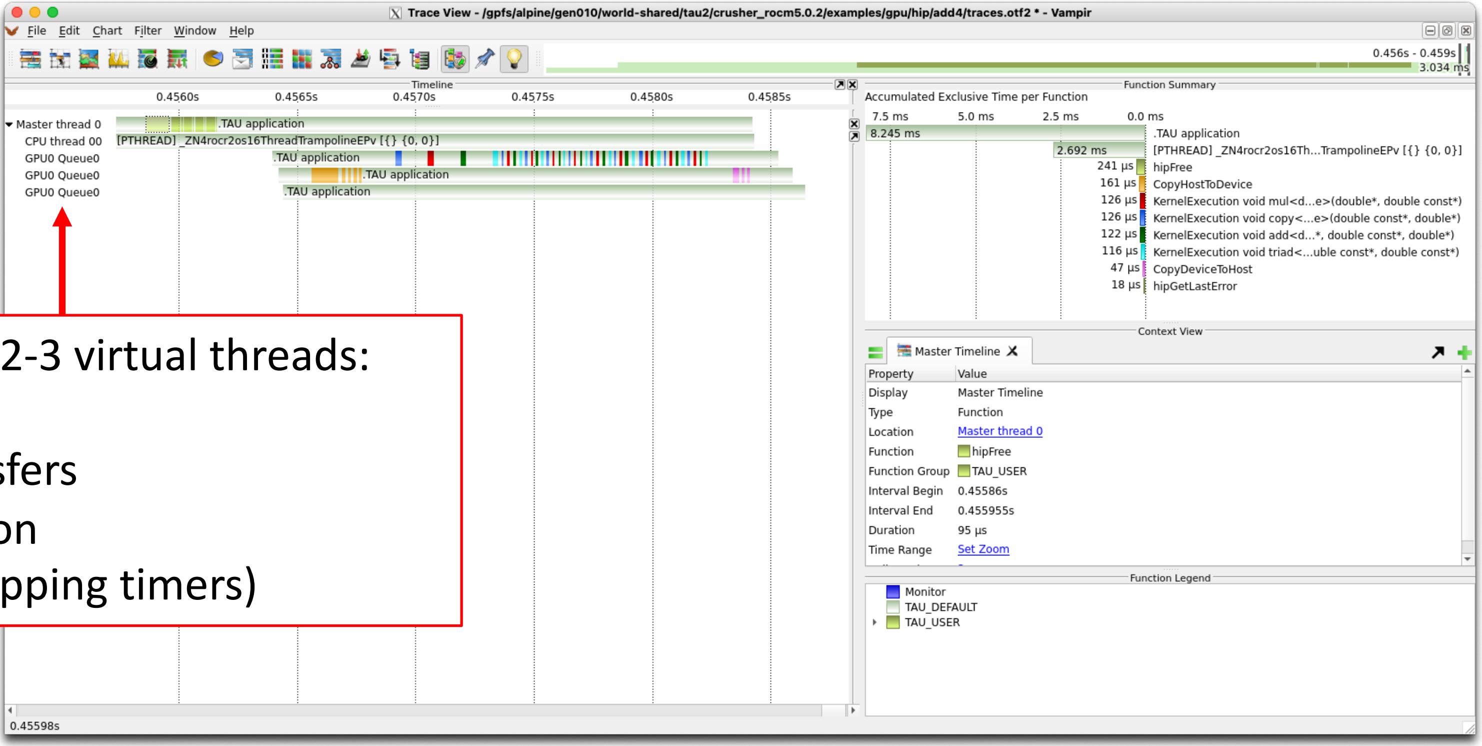
Metric: TAUGPU_TIME
Value: Exclusive
Units: seconds

0.009	void add<double>(double const*, double const*, double const*, double const*, double*) [clone .kd]
0.005	void triad<double>(double*, double const*, double const*) [clone .kd]
0.003	void copy<double>(double const*, double*) [clone .kd]
0.003	void mul<double>(double*, double const*) [clone .kd]
6.4E-4	.TAU application

VERY helpful for understanding **register pressure** and **occupancy**

Tracing support uses Roctracer

```
$ TAU_TRACE=1 TAU_TRACE_FORMAT=otf2 tau_exec -T serial,roctracer ./gpu-stream-hip
```



Trace View - /gpfs/alpine/gen010/world-shared/tau2/crusher_rocm5.0.2/examples/gpu/hip/add4/traces.otf2 * - Vampir

Function Summary

Accumulated Exclusive Time per Function

7.5 ms 5.0 ms 2.5 ms 0.0 ms

Time	Function
8.245 ms	.TAU application
2.692 ms	[PTHREAD] _ZN4rocr2os16Th...TrampolineEPv [{} {0, 0}]
241 µs	hipFree
161 µs	CopyHostToDevice
126 µs	KernelExecution void mul<d...e>(double*, double const*)
126 µs	KernelExecution void copy<...e>(double const*, double*)
122 µs	KernelExecution void add<d...*, double const*, double*)
116 µs	KernelExecution void triad<...uble const*, double const*)
47 µs	CopyDeviceToHost
18 µs	hipGetLastError

Context View

Master Timeline

Property	Value
Display	Master Timeline
Type	Function
Location	Master thread 0
Function	hipFree
Function Group	TAU_USER
Interval Begin	0.45586s
Interval End	0.455955s
Duration	95 µs
Time Range	Set Zoom

Function Legend

- Monitor
- TAU_DEFAULT
- TAU_USER

Each device has 2-3 virtual threads:
1) kernels,
2) memory transfers
3) synchronization
(prevents overlapping timers)

tau_exec command reference

- Uninstrumented execution
  - % mpirun -np 256 ./a.out
- Track GPU operations
  - % mpirun –np 256 tau_exec –l0   ./a.out
  - % mpirun –np 256 tau_exec –opencl ./a.out
  - % mpirun –np 256 tau_exec –openacc ./a.out
  - % mpirun –np 256 tau_exec –cupti ./a.out
  - % mpirun –np 256 tau_exec –rocm ./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,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 export TAU_METRICS=TIME,PAPI_L1_DCM… -ebs_resolution=<file | function | line>
- Non-MPI execution: use –T serial
  - % tau_exec –T serial,level_zero –l0 –ebs ./a.out

TAU 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>)

TAU 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.

TAU 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

For more info…

- https://tau.uoregon.edu
- https://github.com/UO-OACISS/tau2
- https://github.com/UO-OACISS/tau2/wiki
- https://github.com/UO-OACISS/tau2/wiki/Frequently-Asked-Questions-%28FAQ%29
- Email tau-bugs@cs.uoregon.edu

Acknowledgements

Parts of 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.

This research used resources of the Oak Ridge Leadership Computing Facility at the Oak Ridge National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725.

U.S. DEPARTMENT OF ENERGY | Office of Science

RAPIDS

ECP EXASCALE COMPUTING PROJECT

Current/Previous Acknowledgements

Argonne
NATIONAL LABORATORY

Office of Science
U.S. DEPARTMENT OF ENERGY

DEPARTMENT OF ENERGY · UNITED STATES OF AMERICA

ECP
EXASCALE COMPUTING PROJECT

TECHNISCHE UNIVERSITÄT DRESDEN

O
UNIVERSITY OF OREGON

DEPARTMENT OF DEFENSE · UNITED STATES OF AMERICA

NSF

Sandia National Laboratories

ASC™

THE UNIVERSITY of TENNESSEE UT

OAK RIDGE National Laboratory

Los Alamos
NATIONAL LABORATORY

JÜLICH
FORSCHUNGSZENTRUM

Pacific Northwest
NATIONAL LABORATORY

intel

DE LA RECHERCHE À L'INDUSTRIE
cea

NASA

O
THE OHIO STATE UNIVERSITY

ParaTools