

Enabling HPC software productivity with the TAU performance system

Hands-on

Jean-Baptiste BESNARD
<jbbesnard@paratools.fr>

PDC Summer School August 2023, KTH, Stockholm, Sweden.

Sourcing TAU

Guidelines to Load TAU Modules

Initial HOWTO

```
# Add TAU to MODULE PATH
. ~besnard2/TAU/tau.sh
# You now have two flavors of TAU

# Cray
module load TAU/2.32.1-cray

# AMD
module load TAU/2.32.1-amd

# Get some examples

cd && tar xf $TAU_TARBALL
export WORK=$HOME/pdc_tau
```

Tour of Supported Configurations

Cray Compilers and AMD ones

module load TAU/2.32.1-amd

On GPU due to ROCM dep

```
cc --version
```

```
AMD clang version 14.0.0 (https://github.com/RadeonOpenCompute/llvm-project roc-5.0.2 22065 030a405a181176f1a7749819092f4ef8ea5f0758)
```

```
Target: x86_64-unknown-linux-gnu
```

```
Thread model: posix
```

```
InstalledDir: /opt/rocm-5.0.2/llvm/bin
```

module load TAU/2.32.1-cray

On GPU and Main

```
cc --version
```

```
Cray clang version 14.0.1 (3a8780657c742829e80f36338fb6ec6578642bb7)
```

```
Target: x86_64-unknown-linux-gnu
```

```
Thread model: posix
```

```
InstalledDir: /opt/cray/pe/cce/14.0.1/cce-clang/x86_64/share/../bin
```

Sampling

Running with Event-Based Sampling

Flat Event-Based Sampling

```
# Setup Env
module load TAU/2.23.1-cray
# Go to sources
cd $WORK/CoMD/src-mpi/
# Compile without instrumentation
make CC=cc

#Now run
cd ../bin/
sh ./run-mpi.sh

# Show profile
paraprof
```

... tau_exec -ebs ...

Event Based Sampling

Paraprof Output

File Options Windows Help					
Name △					
		Exclusive TIME	Inclusive TIME	Calls	Child Calls
■	.TAU application	0,002	4,015	1	1
■	MPI Collective Sync	0,005	0,005	34	0
■	MPI_Allreduce()	0	0,004	33	33
■	MPI_Barrier()	0,002	0,002	5	0
■	MPI_Bcast()	0	0,001	1	1
■	MPI_Comm_rank()	0	0	1	0
■	MPI_Comm_size()	0	0	1	0
■	MPI_Finalize()	0,023	0,023	1	0
■	MPI_Get_count()	0	0	606	0
■	MPI_Init()	0,063	0,063	1	0
■	MPI_Sendrecv()	0,076	0,076	606	0
■	[CONTEXT] MPI_Sendrecv()	0	0,15	3	0
■	int taupreload_main(int, char **, char **)	3,844	4,013	1	1 255
■	[CONTEXT] int taupreload_main(int, char **, char **)	0	3,65	73	0
■	[SAMPLE] getBoxFromTuple [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src	0,05	0,05	1	0
■	[SUMMARY] ljForce [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/ljfc	3,5	3,5	70	0
■	[SAMPLE] ljForce [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/ljfc	0,3	0,3	6	0
■	[SAMPLE] ljForce [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/ljfc	0,05	0,05	1	0
■	[SAMPLE] ljForce [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/ljfc	0,05	0,05	1	0
■	[SAMPLE] ljForce [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/ljfc	0,75	0,75	15	0
■	[SAMPLE] ljForce [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/ljfc	0,25	0,25	5	0
■	[SAMPLE] ljForce [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/ljfc	0,4	0,4	8	0
■	[SAMPLE] ljForce [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/ljfc	0,15	0,15	3	0
■	[SAMPLE] ljForce [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/ljfc	0,2	0,2	4	0
■	[SAMPLE] ljForce [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/ljfc	0,05	0,05	1	0
■	[SAMPLE] ljForce [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/ljfc	0,2	0,2	4	0
■	[SAMPLE] ljForce [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/ljfc	0,2	0,2	4	0
■	[SAMPLE] ljForce [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/ljfc	0,8	0,8	16	0
■	[SAMPLE] ljForce [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/ljfc	0,05	0,05	1	0
■	[SAMPLE] ljForce [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/ljfc	0,05	0,05	1	0
■	[SUMMARY] loadAtomsBuffer [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/si	0,05	0,05	1	0
■	[SAMPLE] loadAtomsBuffer [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/si	0,05	0,05	1	0
■	[SUMMARY] sortAtomsInCell [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/sr	0,05	0,05	1	0
■	[SAMPLE] sortAtomsInCell [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/sr	0,05	0,05	1	0

Running with Event-Based Sampling

Add Stack Unwinding

```
# Setup Env
module load TAU/2.23.1-cray
# Go to sources
cd $WORK/CoMD/src-mpi/
# Compile without instrumentation
make CC=cc
```

```
#Now run
cd ../bin/
TAU_EBS_UNWIND=1 ./run-mpi.sh
```

```
# Show profile
paraprof
```

... tau_exec **-ebs** ...

Event Based Sampling + Unwind

Paraprof Output

[CONTEXT] timestep (/cfs/klemming/home/b/besnard2/Public/TAU/x86_64/lib/shared-cray-mpi/libTAU-preload.so) {0}	0
int taupreload_main(int, char **, char **)	3,876
[CONTEXT] int taupreload_main(int, char **, char **)	0
[UNWIND] /lib64/libc-2.31.so.0 [@@] __libc_start_main [/{cfs/klemming/home/b/besnard2/Public/TAU/x86_64/lib/shared-cray-mpi/libTAU-preload.so} {0}]	0
[UNWIND] __libc_start_main [/{lib64/libc-2.31.so} {0}]	0
[UNWIND] /cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/CoMD.c.122 [@@] taupreload_main [/{cfs/klemming/home/b/besnard2/Public/TAU/x86_64/lib/shared-cray-mpi/libTAU-preload.so} {0}]	0
[UNWIND] main [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/CoMD.c} {122}]	0
[UNWIND] /cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/.mytype.h.23 [@@] computeForce [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/ljForce.c} {189}]	0
[UNWIND] /cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/ljForce.c.0 [@@] computeForce [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/ljForce.c} {189}]	0
[UNWIND] /cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/ljForce.c.173 [@@] computeForce [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/ljForce.c} {189}]	0
[UNWIND] /cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/ljForce.c.189 [@@] computeForce [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/ljForce.c} {189}]	0
[UNWIND] /cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/ljForce.c.193 [@@] computeForce [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/ljForce.c} {189}]	0
[UNWIND] /cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/ljForce.c.198 [@@] computeForce [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/ljForce.c} {189}]	0
[UNWIND] /cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/ljForce.c.199 [@@] computeForce [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/ljForce.c} {189}]	0
[UNWIND] /cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/ljForce.c.202 [@@] computeForce [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/ljForce.c} {189}]	0
[UNWIND] /cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/ljForce.c.207 [@@] computeForce [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/ljForce.c} {189}]	0
[UNWIND] /cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/ljForce.c.208 [@@] computeForce [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/ljForce.c} {189}]	0
[UNWIND] /cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/ljForce.c.209 [@@] computeForce [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/ljForce.c} {189}]	0
[UNWIND] /cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/ljForce.c.214 [@@] computeForce [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/ljForce.c} {189}]	0
[UNWIND] /cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/ljForce.c.220 [@@] computeForce [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/ljForce.c} {189}]	0
[UNWIND] /cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/ljForce.c.224 [@@] computeForce [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/ljForce.c} {189}]	0
[UNWIND] __libc_start_main [/{cfs/klemming/home/b/besnard2/Public/TAU/x86_64/lib/shared-cray-mpi/libTAU-preload.so} {0}]	0
[UNWIND] /cfs/klemming/home/b/besnard2/Public/TAU/x86_64/lib/shared-cray-mpi/libTAU-preload.so.0 [@@] __libc_start_main [/{lib64/libc-2.31.so} {0}]	0
[UNWIND] taupreload_main [/{cfs/klemming/home/b/besnard2/Public/TAU/x86_64/lib/shared-cray-mpi/libTAU-preload.so} {0}]	0
[UNWIND] /cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/ljForce.c.189 [@@] main [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/timestep.c} {73}]	0
[UNWIND] /cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/timestep.c.44 [@@] main [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/timestep.c} {73}]	0
[UNWIND] /cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/timestep.c.73 [@@] main [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-mpi/timestep.c} {73}]	0

Callsite Profiling

Callsites

Flat Profile with MPI events

```
# Setup Env
module load TAU/2.23.1-cray
# Go to sources
cd $WORK/CoMD/src-mpi/
# Compile without instrumentation
make CC=cc
#Now run
cd ../bin/
# Edit run-mpi.sh and remove -ebs
./run-mpi.sh
# Show profile
paraprof
```

... tau_exec ...

MPI Flat Profile

Paraprof Output

– .TAU application	0,002
– MPI Collective Sync	0,013
– MPI_Allreduce()	0
– MPI_Barrier()	0,002
– MPI_Bcast()	0,001
– MPI_Comm_rank()	0
– MPI_Comm_size()	0
– MPI_Finalize()	0,006
– MPI_Get_count()	0
– MPI_Init()	0,061
– MPI_Sendrecv()	0,08
– int taupreload_main(int, char **, char **)	3,798

Callsites with Stacks






























Add location of the calls

```
# Setup Env
module load TAU/2.23.1-cray
# Go to sources
cd $WORK/CoMD/src-mpi/
# Compile without instrumentation
make CC=cc
#Now run
cd ../bin/
# Edit run-mpi.sh and remove -ebs
TAU_CALLPATH=1 ./run-mpi.sh
# Show profile
paraprof
```

... tau_exec ... (-mpi is by default)

MPI Flat Profile + Callpath

Paraprof Output

Name 		Exclusi
  .TAU application		
  MPI_Allreduce()		
  MPI_Bcast()		
  MPI Collective Sync		
  int taupreload_main(int, char **, char **)		
  MPI_Allreduce()		
  MPI_Barrier()		
  MPI_Bcast()		
  MPI_Comm_rank()		
  MPI_Comm_size()		
  MPI_Finalize()		
  MPI_Get_count()		
  MPI_Init()		
  MPI_Sendrecv()		

Looking at MPI Traces

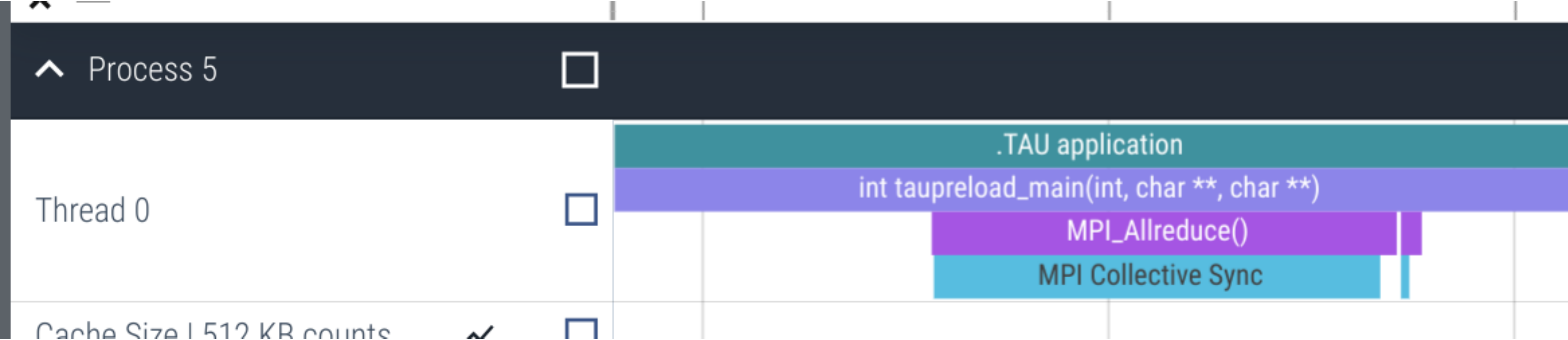
How to understand temporal behavior

```
# Setup Env
module load TAU/2.23.1-cray
# Go to sources
cd $WORK/CoMD/src-mpi/
# Compile without instrumentation
make CC=cc
#Now run
cd ../bin/
TAU_TRACE=1 ./run-mpi.sh
# Merge per process traces
tau_treemerge.pl
# Generate CHROME trace
tau_trace2json ./tau.trc ./tau.edf -o ./out.json -chrome [-ignoreatomic]
# Copy traces to your machine
# Load in https://ui.perfetto.dev/
```

... tau_exec ... (-mpi is by default)

MPI Trace

Perfetto UI Output



Profiling I/O

Capturing I/O Calls

How to look at atomic events

```
# Setup Env
module load TAU/2.23.1-cray
# Go to sources
cd $WORK/mpiposix/
# Compile without instrumentation
make CC=cc
#Now run
sh ./run-mpi.sh
# Show profile
paraprof
```

Go to Window/Thread/User Event Statistics

... tau_exec -T cray,mpi -io ...

IO Atomic Events

Paraprof Output

Max	Min	Mean	Std. Dev	Name
40000	2	39749,78	3152,478	Bytes Written
1904,762	0,01	1379,365	206,721	Write Bandwidth (MB/s)
1904,762	220,994	1387,329	178,836	Write Bandwidth (MB/s) : int taupreload_m
40000	40000	40000	0	Bytes Written : int taupreload_main(int,
1904,762	220,994	1387,329	178,836	Write Bandwidth (MB/s) <file=out.0.dat>
40000	40000	40000	0	Bytes Written <file=out.0.dat>
1904,762	220,994	1387,329	178,836	Write Bandwidth (MB/s) <file=out.0.dat> :
40000	40000	40000	0	Bytes Written <file=out.0.dat> : int taup
39,429	0,167	0,829	3,163	Read Bandwidth (MB/s)
276	1	2,821	22,305	Bytes Read
1	0,167	0,572	0,267	Read Bandwidth (MB/s) : int taupreload_ma
1	1	1	0	Bytes Read : int taupreload_main(int, cha
32	0,01	6,351	7,712	Write Bandwidth (MB/s) <file=/dev/cxi0>
88	8	35,228	29,853	Bytes Written <file=/dev/cxi0>
32	0,043	7,569	7,815	Write Bandwidth (MB/s) : int taupreload_m
88	8	53,935	28,735	Bytes Written : int taupreload_main(int,
32	0,043	7,569	7,815	Write Bandwidth (MB/s) <file=/dev/cxi0> :
88	8	53,935	28,735	Bytes Written <file=/dev/cxi0> : int taup

Tracing I/O Calls

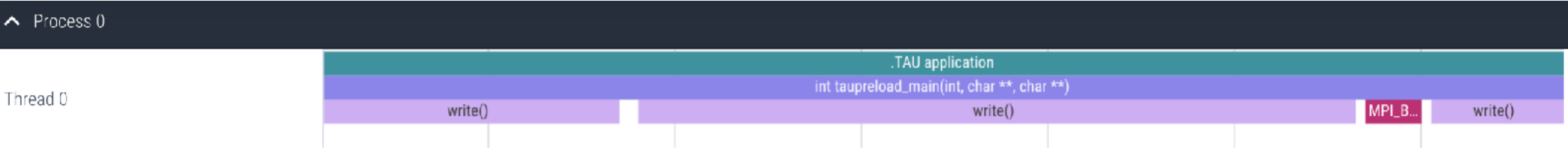
How to look at atomic events

```
# Setup Env
module load TAU/2.23.1-cray
# Go to sources
cd $WORK/mpiposix/
# Compile without instrumentation
make CC=cc
#Now run
TAU_TRACE=1 sh ./run-mpi.sh
# Merge per process traces
tau_treemerge.pl
# Generate CHROME trace
tau_trace2json ./tau.trc ./tau.edf -o ./out.json -chrome [-ignoreatomic]
# Copy traces to your machine
# Load in https://ui.perfetto.dev/
```

... tau_exec -T cray,mpi **-io** ...

MPI Trace

Perfetto UI Output



Instrumenting OpenMP

OpenMP via OMPT

Requires LLVM's OMP runtime

```
# Setup Env
module load TAU/2.23.1-amd
# Go to sources
cd $WORK/CoMD/src-openmp/
# Compile without instrumentation
make CC=cc
#Now run
cd ../bin/
export TAU_OMPT_SUPPORT_LEVEL=full
./run-omp.sh
# Show profile
paraprof
```

... tau_exec -T amd,ompt -ompt -ebs ...

MPI Flat Profile + Callpath

Paraprof Output

■ MPI_Sendrecv()	
■ OpenMP_Parallel_Region computeVcm [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-openmp/initAtoms.c}{218, 0}]	
■ OpenMP_Parallel_Region kineticEnergy [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-openmp/timestep.c}{120, 0}]	
■ OpenMP_Parallel_Region ljForce [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-openmp/ljForce.c}{0, 0}]	
■ OpenMP_Parallel_Region ljForce [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-openmp/ljForce.c}{172, 0}]	
■ OpenMP_Parallel_Region randomDisplacements [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-openmp/initAtoms.c}{205, 0}]	
■ OpenMP_Parallel_Region redistributeAtoms [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-openmp/timestep.c}{155, 0}]	
■ OpenMP_Parallel_Region setTemperature [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-openmp/initAtoms.c}{167, 0}]	
■ OpenMP_Parallel_Region setTemperature [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-openmp/initAtoms.c}{184, 0}]	
■ OpenMP_Parallel_Region setVcm [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-openmp/initAtoms.c}{136, 0}]	
■ OpenMP_Parallel_Region timestep [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-openmp/timestep.c}{39, 0}]	
■ OpenMP_Parallel_Region timestep [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-openmp/timestep.c}{43, 0}]	
■ OpenMP_Parallel_Region timestep [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-openmp/timestep.c}{47, 0}]	
■ OpenMP_Parallel_Region timestep [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-openmp/timestep.c}{55, 0}]	
■ OpenMP_Sync_Region_Barrier_Implicit computeVcm [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-openmp/initAtoms.c}{218, 0}]	
■ OpenMP_Sync_Region_Barrier_Implicit kineticEnergy [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-openmp/timestep.c}{120, 0}]	
■ OpenMP_Sync_Region_Barrier_Implicit ljForce [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-openmp/ljForce.c}{0, 0}]	
■ OpenMP_Sync_Region_Barrier_Implicit ljForce [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-openmp/ljForce.c}{172, 0}]	
■ OpenMP_Sync_Region_Barrier_Implicit randomDisplacements [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-openmp/initAtoms.c}{205, 0}]	
■ OpenMP_Sync_Region_Barrier_Implicit redistributeAtoms [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-openmp/timestep.c}{155, 0}]	
■ OpenMP_Sync_Region_Barrier_Implicit setTemperature [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-openmp/initAtoms.c}{167, 0}]	
■ OpenMP_Sync_Region_Barrier_Implicit setTemperature [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-openmp/initAtoms.c}{184, 0}]	
■ OpenMP_Sync_Region_Barrier_Implicit setVcm [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-openmp/initAtoms.c}{136, 0}]	
■ OpenMP_Sync_Region_Barrier_Implicit timestep [/{cfs/klemming/home/b/besnard2/pdt_tau_workshop/CoMD/src-openmp/timestep.c}{39, 0}]	

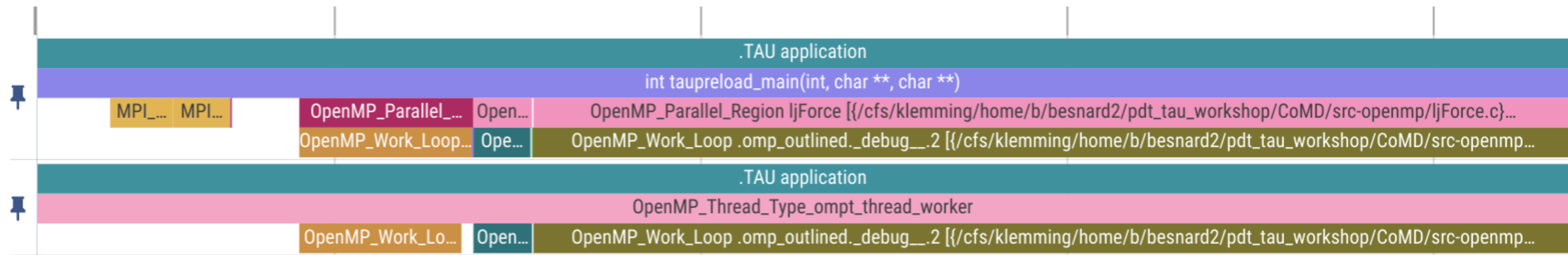
Looking at OpenMP Traces

How to understand temporal behavior

```
# Setup Env
module load TAU/2.23.1-amd
# Go to sources
cd $WORK/CoMD/src-openmp/
# Compile without instrumentation
make CC=cc
#Now run
cd ../bin/
export TAU_OMPT_SUPPORT_LEVEL=full
TAU_TRACE=1 ./run-omp.sh
# Merge per process traces
tau_treemerge.pl
# Generate CHROME trace
tau_trace2json ./tau.trc ./tau.edf -o ./out.json -chrome [-ignoreatomic]
# Copy traces to your machine
# Load in https://ui.perfetto.dev/
```

MPI Trace

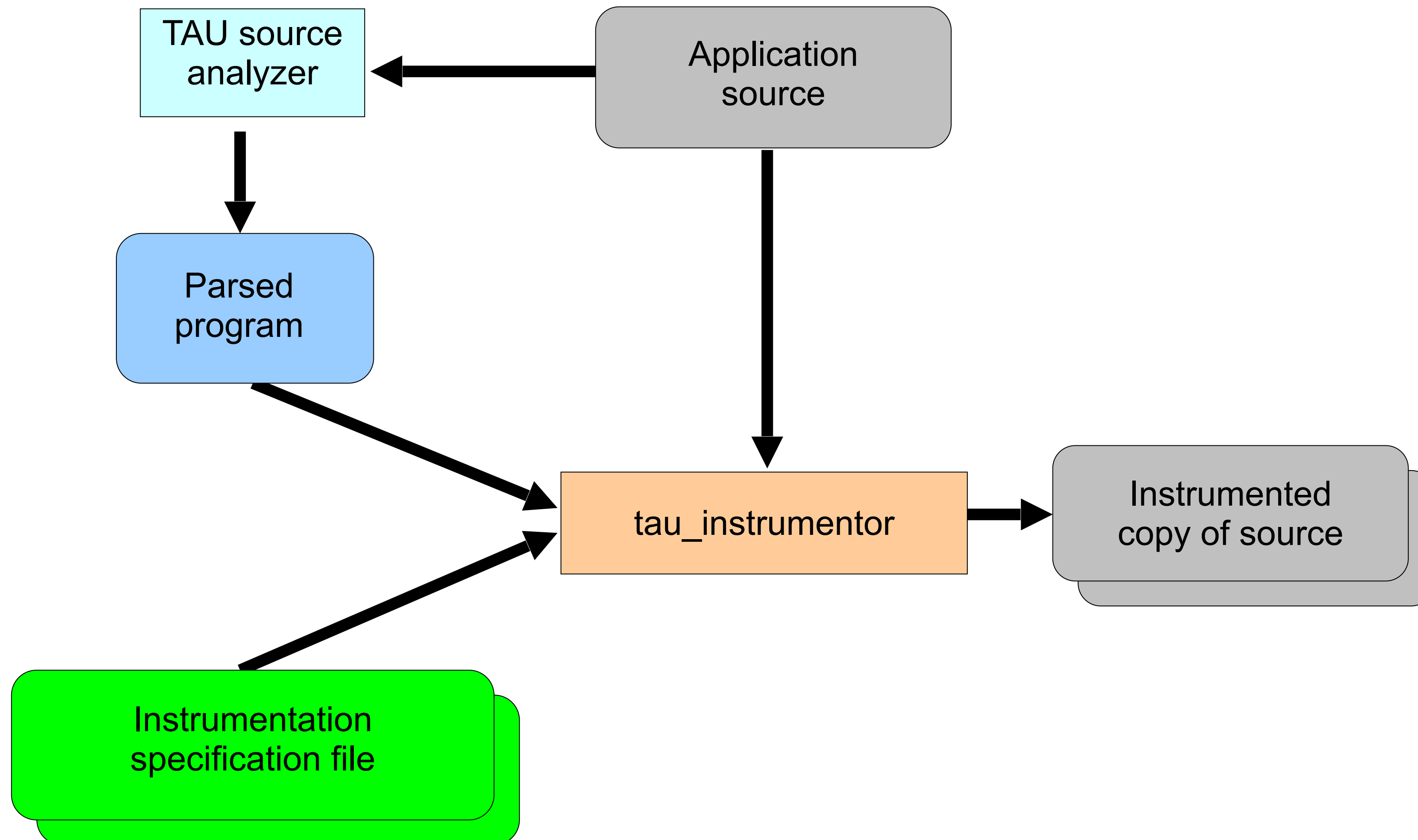
Perfetto UI Output



Compiler Based Instrumentation

Tracking More at Compile Time

Using PDT (source to source)



Tracking More at Compile Time

Using PDT

```
# Setup Env
module load TAU/2.23.1-amd
# Go to sources
cd $WORK/CoMD/src-mpi/
# Set the TAU Makefile
export TAU_MAKEFILE=$TAU/Makefile.tau-cray-mpi-pdt
# Compile with clang instrumentation
unset TAU_OPTION
make CC=tau_cc.sh
#Now run
cd ../bin/
# We add call path not to get a flat profile
TAU_CALLPATH=1 ./run-mpi-bare.sh
# Show profile
paraprof
```

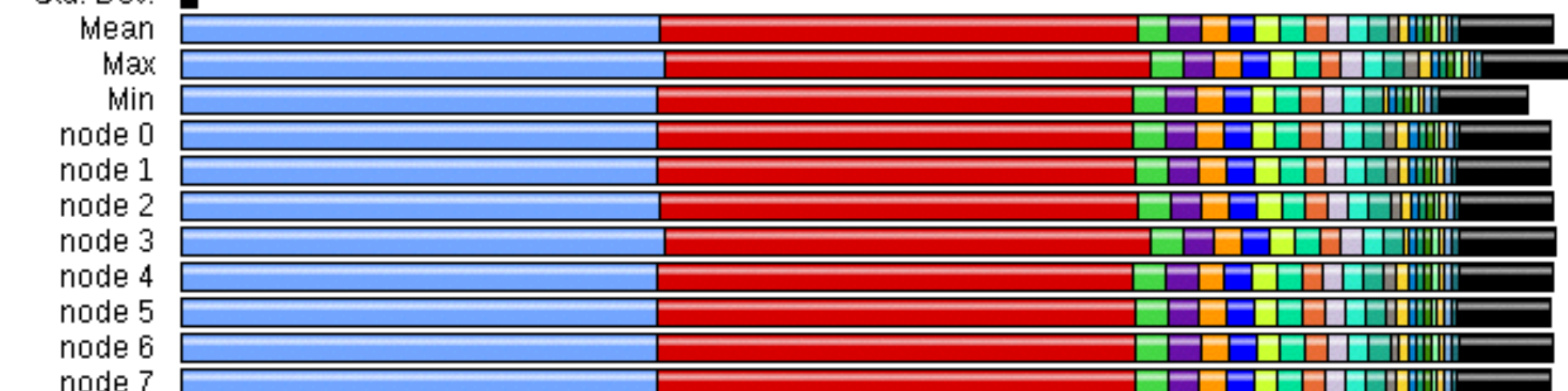

Looking at The Output

A look in Paraprof

File Options Windows Help

Metric: TIME
Value: Exclusive

Std. Dev. █



File Options Windows Help

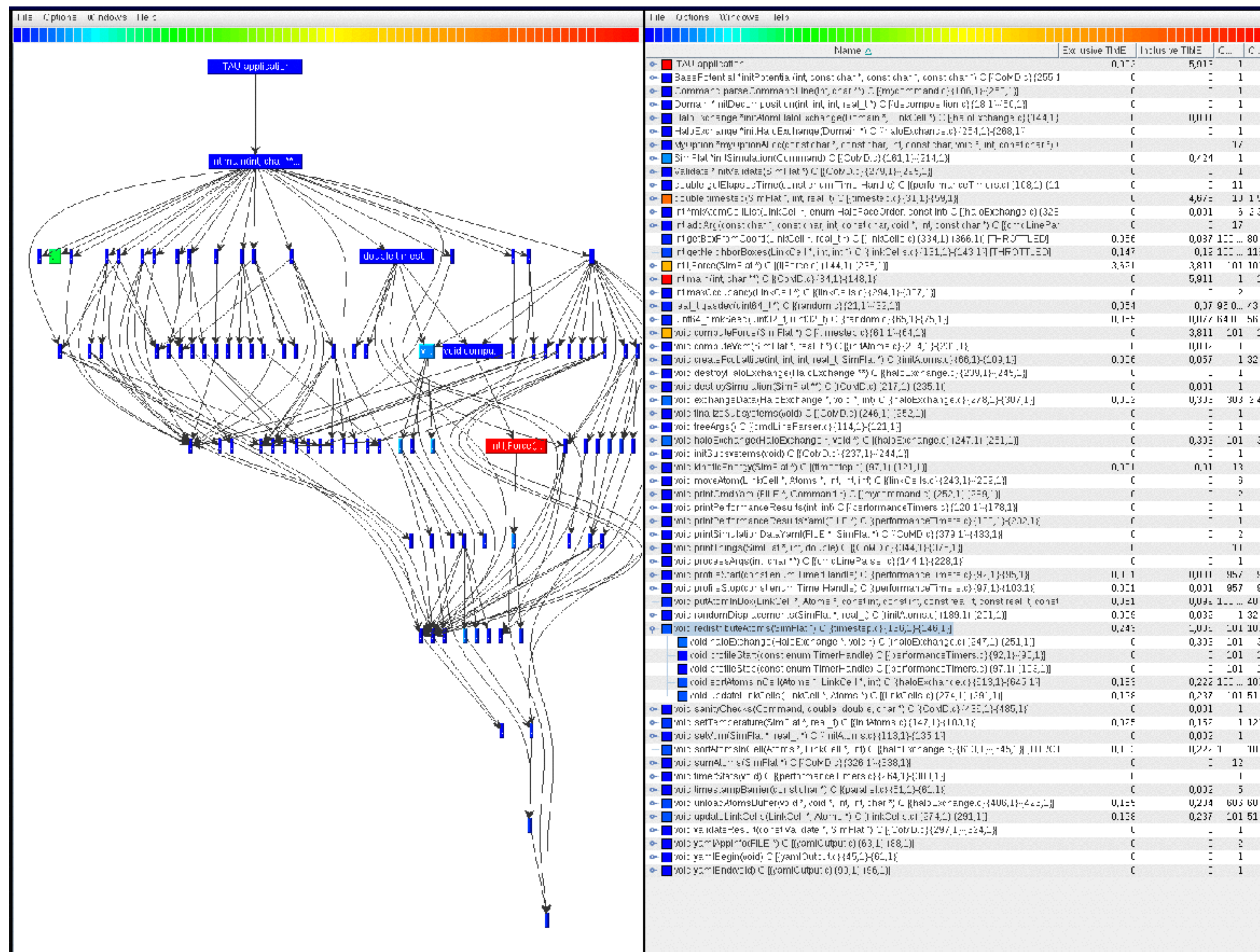
Name				Exclusiv...	Inclusiv...
int getBoxFromCoord(LinkCell *, real_t *) C [{linkCells.c} {334,1}-{366,1}] [THR				0,054	0,084
int getNeighborBoxes(LinkCell *, int, int *) C [{linkCells.c} {131,1}-{143,1}] [THR				0,155	0,198
void putAtomInBox(LinkCell *, Atoms *, const int, const int, const real_t, const re				0,051	0,097
void sortAtomsInCell(Atoms *, LinkCell *, int) C [{haloExchange.c} {613,1}-{645,				0,191	0,228
real_t gasdev(uint64_t *) C [{random.c} {21,1}-{32,1}]				0,051	0,067
uint64_t mkSeed(uint32_t, uint32_t) C [{random.c} {65,1}-{75,1}]				0,052	0,073
void profileStart(const enum TimerHandle) C [{performanceTimers.c} {92,1}-{9!				0,001	0,001
void profileStop(const enum TimerHandle) C [{performanceTimers.c} {97,1}-{1!				0,001	0,001
int sendReceiveParallel(void *, int, int, void *, int, int) C [{parallel.c} {93,1}-{111,1				0,001	0,081
void unloadAtomsBuffer(void *, void *, int, int, char *) C [{haloExchange.c} {406,1				0,162	0,211
void exchangeData(HaloExchange *, void *, int) C [{haloExchange.c} {278,1}-{3!				0,001	0,318
int ljForce(SimFlat *) C [{ljForce.c} {144,1}-{235,1}]				3,641	3,839
int getNeighborBoxes(LinkCell *, int, int *) C [{linkCells.c} {131,1}-{143,1}]				0,155	0,198
void computeForce(SimFlat *) C [{timestep.c} {61,1}-{64,1}]				0	3,84
void haloExchange(HaloExchange *, void *) C [{haloExchange.c} {247,1}-{251,1				0	0,318
void redistributeAtoms(SimFlat *) C [{timestep.c} {136,1}-{146,1}]				0,247	1,038
void updateLinkCells(LinkCell *, Atoms *) C [{linkCells.c} {274,1}-{291,1}]				0,207	0,245
MPI_Allreduce()				0	0,01
MyOption *myOptionAlloc(const char *, const char, int, const char, void *, int, cor				0	0
int addArg(const char *, const char, int, const char, void *, int, const char *) C [{c				0	0
void addRealParallel(real_t *, real_t *, int) C [{parallel.c} {123,1}-{131,1}]				0	0,009
void addIntParallel(int *, int *, int) C [{parallel.c} {113,1}-{121,1}]				0	0,001
void kineticEnergy(SimFlat *) C [{timestep.c} {97,1}-{121,1}]				0,001	0,008
void sumAtoms(SimFlat *) C [{CoMD.c} {326,1}-{338,1}]				0	0
double getElapsedTime(const enum TimerHandle) C [{performanceTimers.c} {				0	0
void printThings(SimFlat *, int, double) C [{CoMD.c} {344,1}-{375,1}]				0	0
double timestep(SimFlat *, int, real_t) C [{timestep.c} {31,1}-{59,1}]				0	4,741
int *mkAtomCellList(LinkCell *, enum HaloFaceOrder, const int) C [{haloExchar				0	0,001	6	...
void barrierParallel() C [{parallel.c} {79,1}-{84,1}]				0	0,002	5	5
void moveAtom(LinkCell *, Atoms *, int, int, int) C [{linkCells.c} {243,1}-{259,1}]				0	0	5	...
void timestampBarrier(const char *) C [{parallel.c} {51,1}-{61,1}]				0	0,002	5	...
int maxOccupancy(LinkCell *) C [{linkCells.c} {294,1}-{307,1}]				0	0	2	6
void addDoubleParallel(double *, double *, int) C [{parallel.c} {133,1}-{141,1}]				0	0	2	2
void maxIntParallel(int *, int *, int) C [{parallel.c} {143,1}-{151,1}]				0	0	2	2

Tracking More at Compile Time

Using LLVM Plugin (preview !)

```
# Setup Env
module load TAU/2.23.1-amd
# Go to sources
cd $WORK/CoMD/src-mpi/
# Set the TAU Makefile
export TAU_MAKEFILE=$TAU/Makefile.tau-amd-clang-mpi
export TAU_OPTION="-optCompInst -optVerbose"
export TAU_COMPILER_MIN_INSTRUCTION_COUNT=1
# Compile with PDT instrumentation
make CC=tau_cc.sh
#Now run
cd ../bin/
# We add call path not to get a flat profile
TAU_CALLPATH=1 ./run-mpi-bare.sh
# Show profile
paraprof
```


A look in Paraprof



Looking at The Output

Similar to LLVM (less verbose)

Create a filter file « select.tau » :

```
BEGIN_INCLUDE_LIST
int timestep#
int MPI_#
END_INCLUDE_LIST
```

Sample syntax:

```
BEGIN_INCLUDE_LIST
int main#
int dgemm#
END_INCLUDE_LIST
BEGIN_FILE_INCLUDE_LIST
Main.c
Blas/*.f77
END_FILE_INCLUDE_LIST
# replace include with exclude list
```

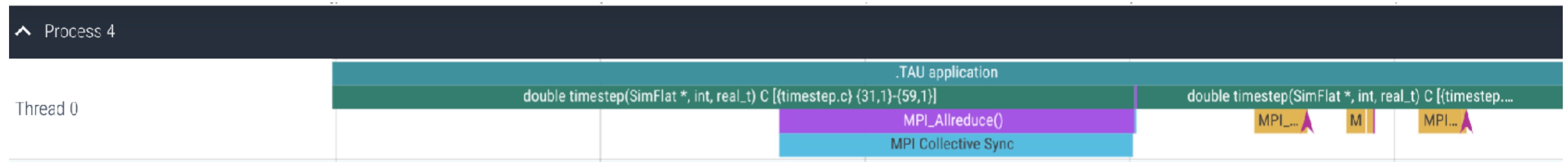
Selective Instrumentation for Tracing

Starting from the PDT case ..

```
# Setup Env
module load TAU/2.23.1-cray
# Go to sources
cd $WORK/CoMD/src-mpi/
# Set the TAU Makefile
export TAU_MAKEFILE=$TAU/Makefile.tau-cray-mpi-pdt
# Compile with selective instrumentation
export TAU_OPTION="-optTauSelectFile=select.tau"
make CC=tau_cc.sh
#Now run
cd ../bin/
# We add call path not to get a flat profile
TAU_TRACE=1 ./run-mpi-bare.sh
# Merge per process traces
tau_treemerge.pl
# Generate CHROME trace
tau_trace2json ./tau.trc ./tau.edf -o ./out.json -chrome [-ignoreatomic]
# Copy traces to your machine
# Load in https://ui.perfetto.dev/
```

Looking at The Output

A look in Perfetto



GPU Instrumentation (AMD/ROCM)

GPU Instrumentation (AMD)

Track ROCM Events

```
# Setup Env
module load TAU/2.23.1-amd
# Go to sources
cd $WORK/hip_add4/
# Look at content just makes on the nodes
./compile.sh
#Now run
./run.sh
# Show profile
paraprof
```

... tau_exec -T rocm,serial -rocm -ebs ...

Looking at The Output

A look in Paraprof

■ .TAU application	0,002
■ int taupreload_main(int, char **, char **)	9,793
■ [CONTEXT] int taupreload_main(int, char **, char **)	0
■ void add<double>(double const*, double const*, double const*, double const*, double*) [clone .kd]	0,358
■ void copy<double>(double const*, double*) [clone .kd]	0,12
■ void mul<double>(double*, double const*) [clone .kd]	0,12
■ void triad<double>(double*, double const*, double const*) [clone .kd]	0,193

GPU Instrumentation (AMD)

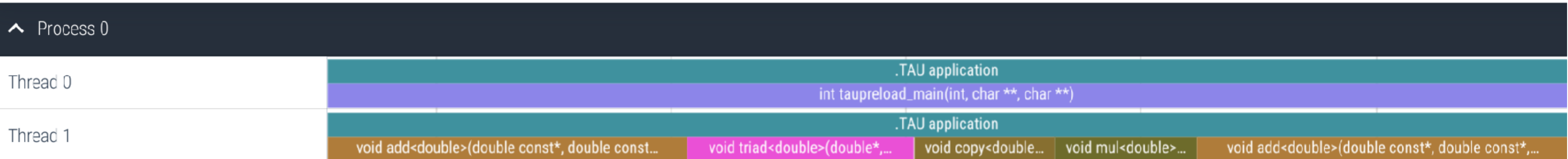
Trace ROCM Events

```
# Setup Env
module load TAU/2.23.1-amd
# Go to sources
cd $WORK/hip_add4/
# Look at content just makes on the nodes
./compile.sh
#Now run
TAU_TRACE=1 ./run.sh
# Merge per process traces
tau_treemerge.pl
# Generate CHROME trace
tau_trace2json ./tau.trc ./tau.edf -o ./out.json -chrome [-ignoreatomic]
# Copy traces to your machine
# Load in https://ui.perfetto.dev/
```

... tau_exec -T rocm,serial -rocm -ebs ...

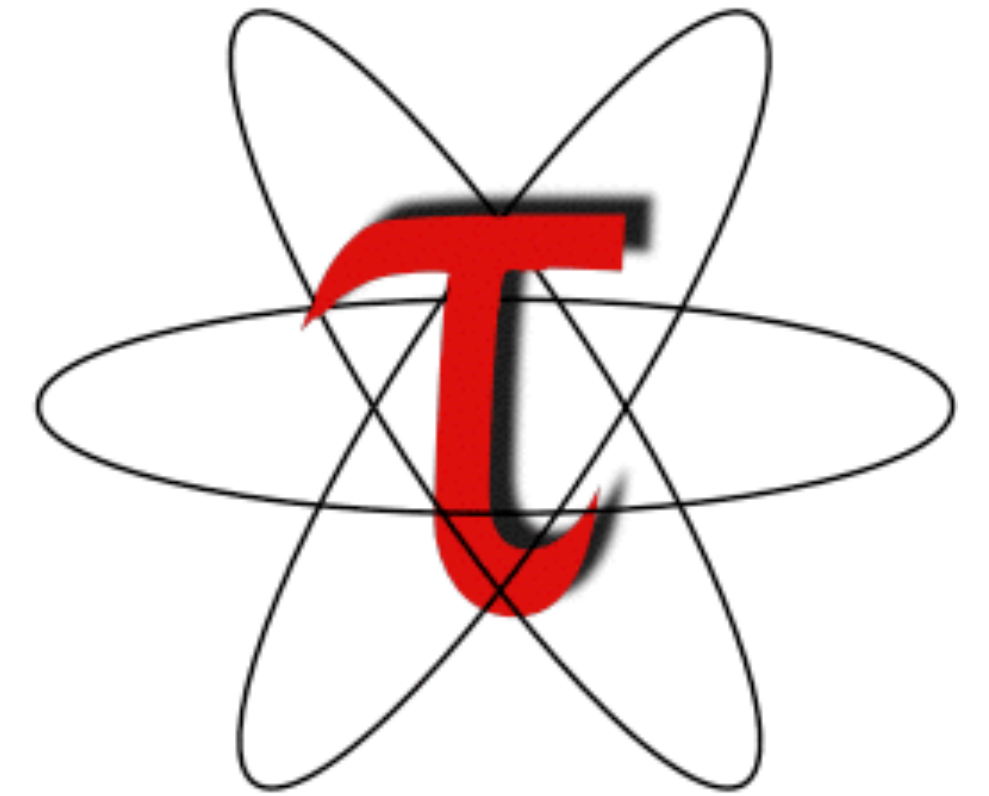
Looking at The Output

A look in Perfetto



Cheat-Sheet

Download TAU from U. Oregon



<http://tau.uoregon.edu>

<http://taucommander.com>

<http://www.hpclinux.com> [OVA for VirtualBox]

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

Free download, open source, BSD license

Setup: Installing TAU on Laptops

Prerequisites: Java in your path

Microsoft Windows

- Install Java from Oracle.com
 - <http://tau.uoregon.edu/tau.exe>
 - Install, click on a ppk file to launch paraprof

macOS (x86_64)

- Install Java 11.0.3:
 - Download and install <http://tau.uoregon.edu/java.dmg>
 - If you have multiple Java installations, add to your ~/.zshrc (or ~/.bashrc as appropriate):
 - `export PATH=/Library/Java/JavaVirtualMachines/jdk-11.0.3.jdk/Contents/Home/bin:$PATH`
 - `java -version`
- Download and install TAU (copy to /Applications from dmg):
 - <http://tau.uoregon.edu/tau.dmg>
 - `export PATH=/Applications/TAU/tau/apple/bin:$PATH`
 - `paraprof app.ppk &`

macOS (arm64, M1,M2)

- http://tau.uoregon.edu/java_arm64.dmg
- http://tau.uoregon.edu/tau_arm64.dmg

Linux (<http://tau.uoregon.edu/tau.tgz>)

- `./configure -mpi -bfd=download -iowrapper -dwarf=download -unwind=download -otf=download ; make install`

Installing and Configuring TAU

•Installing PDT:

- `wget tau.uoregon.edu/pdt_lite.tgz`
- `./configure; make ; make install`

•Installing TAU:

- `wget tau.uoregon.edu/tau.tgz; tar xzf tau.tgz; cd tau-2.<ver>`
- `wget http://tau.uoregon.edu/ext.tgz ; tar xf ext.tgz`
- `./configure -bfd=download -pdt=<dir> -papi=<dir> -mpi
-pthread -c++=mpicxx -cc=mpicc -fortran=mpif90
-dwarf=download -unwind=download -otf=download
-iowrapper -papi=<dir>`
- `make install`

•Using TAU:

- `export TAU_MAKEFILE=<taudir>/x86_64/lib/Makefile.tau-<TAGS>`
- `make CC=tau_cc.sh CXX=tau_cxx.sh F90=tau_f90.sh`

Compile-Time Options

Optional parameters for the TAU_OPTIONS environment variable:

% tau_compiler.sh

- optVerbose Turn on verbose debugging messages
- optComplnst Use compiler based instrumentation
- optNoComplnst Do not revert to compiler instrumentation if source instrumentation fails.
- optTrackIO Wrap POSIX I/O call and calculates vol/bw of I/O operations (Requires TAU to be configured with *-iowrapper*)
- optTrackGOMP Enable tracking GNU OpenMP runtime layer (used without *-opari*)
- optMemDbg Enable runtime bounds checking (see TAU_MEMDBG_* env vars)
- optKeepFiles Does not remove intermediate .pdb and .inst.* files
- optPreProcess Preprocess sources (OpenMP, Fortran) before instrumentation
- optTauSelectFile="*<file>*" Specify selective instrumentation file for *tau_instrumentor*
- optTauWrapFile="*<file>*" Specify path to *link_options.tau* generated by *tau_gen_wrapper*
- optHeaderInst Enable Instrumentation of headers
- optTrackUPCR Track UPC runtime layer routines (used with tau_upc.sh)
- optLinking="" Options passed to the linker. Typically
\$(TAU_MPI_FLIBS) \$(TAU_LIBS) \$(TAU_CXXLIBS)
- optCompile="" Options passed to the compiler. Typically
\$(TAU_MPI_INCLUDE) \$(TAU_INCLUDE) \$(TAU_DEFS)
- optPdtF95Opts="" Add options for Fortran parser in PDT (f95parse/gfparse) ...

Compile-Time Options (contd.)

Optional parameters for the TAU_OPTIONS environment variable:

% tau_compiler.sh

- optShared Use TAU's shared library (libTAU.so) instead of static library (default)
- optPdtCxxOpts="" Options for C++ parser in PDT (cxxparse).
- optPdtF90Parser="" Specify a different Fortran parser
- optPdtCleanscapeParser Specify the Cleanscape Fortran parser instead of GNU gfparsers
- optTau="" Specify options to the tau_instrumentor
- optTrackDMAPP Enable instrumentation of low-level DMAPP API calls on Cray
- optTrackPthread Enable instrumentation of pthread calls

See tau_compiler.sh for a full list of TAU_OPTIONS.

...

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 <code>–optMemDbg</code> or <code>tau_exec –memory</code>)
TAU_EBS_SOURCE	TIME	Allows using PAPI hardware counters for periodic interrupts for EBS (e.g., <code>TAU_EBS_SOURCE=PAPI_TOT_INS</code> when <code>TAU_SAMPLING=1</code>)
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 <code>TAU_MEMDBG_PROTECT_*</code>)
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 <code>–optMemDbg</code> while building or <code>tau_exec –memory</code>)
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 <code>–optMemDbg</code> or <code>tau_exec –memory</code>)
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