

Intuitive Performance Engineering at the Exascale with TAU and TAU Commander

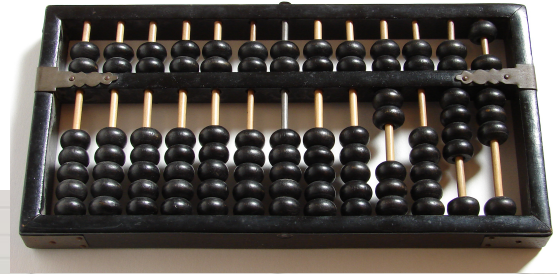
John C. Linford
ParaTools, Inc.

Argonne Extreme Scale Computing Training Program
10 August 2016, Pheasant Run “Resort”

Intuitive Performance Engineering

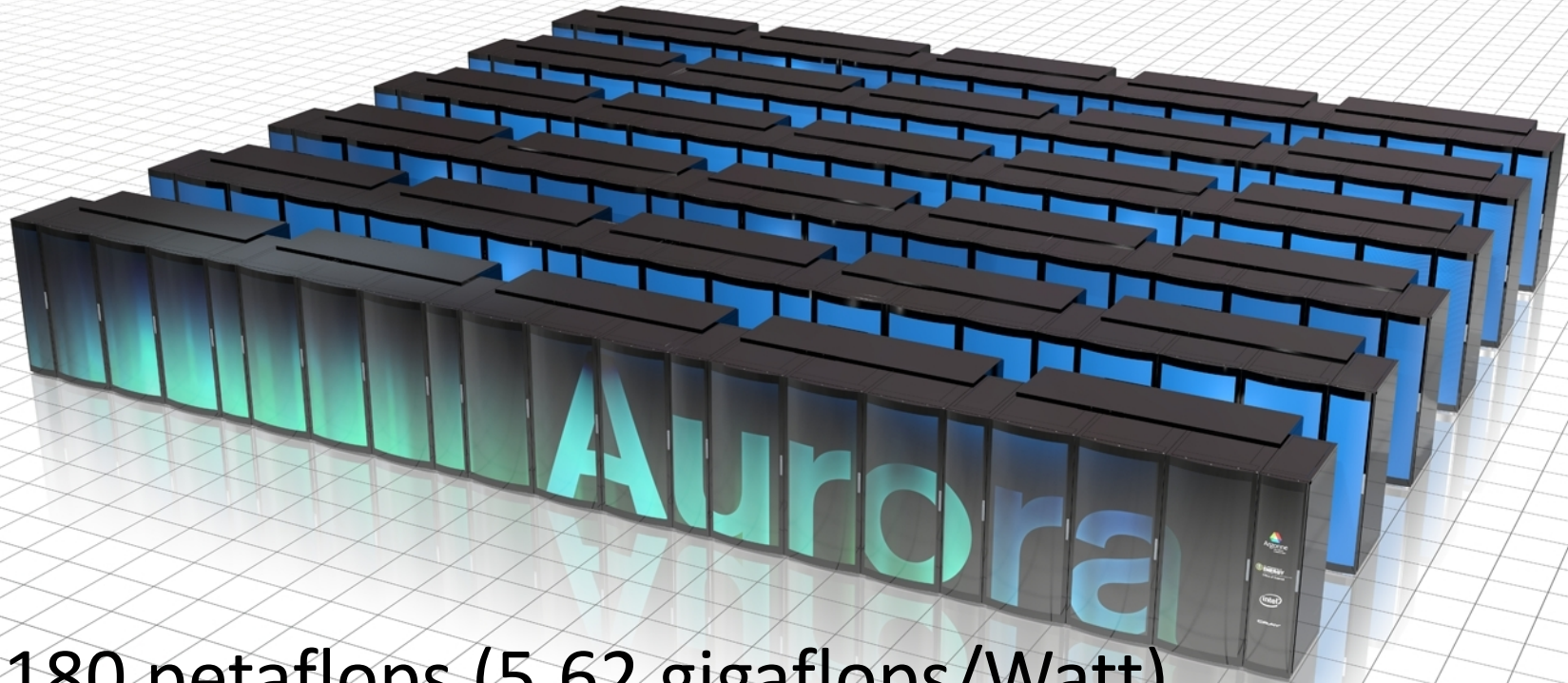
A SILLY INTRODUCTION

This is faster than an abacus



- 10 petaflops (2.08 gigaflops/Watt)
- 1.25 quadrillion abacuses
- ~172,000 abacuses per living person

This is even faster



- 180 petaflops (5.62 gigaflops/Watt)
- 22.5 quadrillion abacuses
- ~3.1M abacuses per living person

Unfreeze your science fish!

- Performance engineering tools are essential to success.
- Increase your capability to answer the hard questions.
- Reduce time-to-solution (i.e. publish first).
- Lower costs.
- Credibility, provenance, etc.

Delicious science



You

Intuitive Performance Engineering

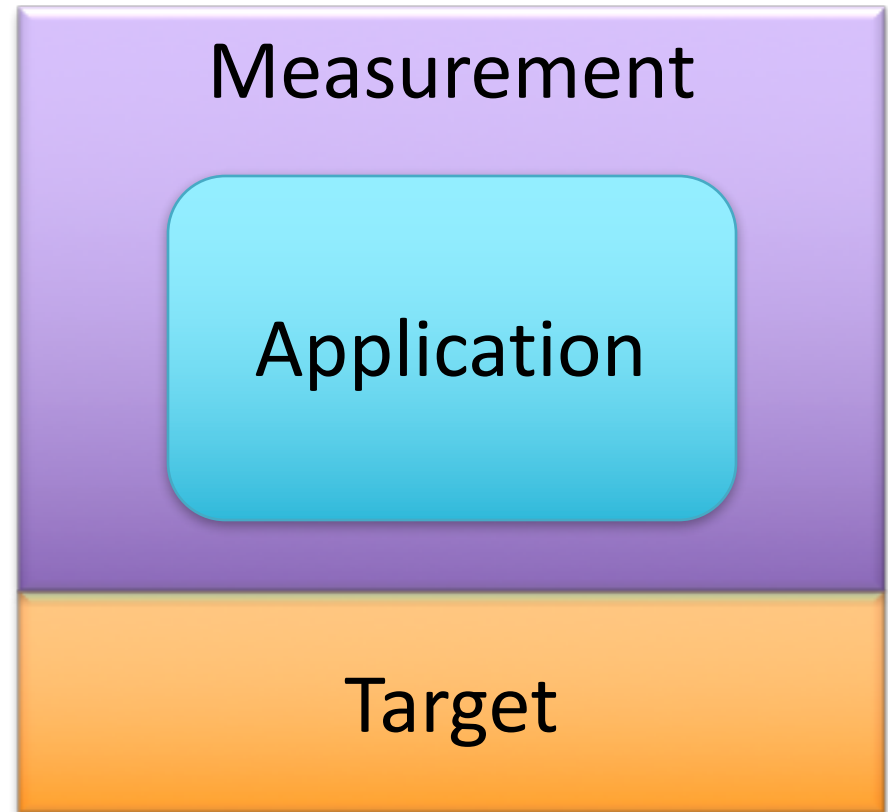
PERFORMANCE ENGINEERING

Define your project

- What are the **target** hardware and software?
 - Shared memory via OpenMP, pthreads, etc?
 - Distributed memory via MPI, SHMEM, etc?
 - Intel compilers? IBM compilers?
- What are the **application** features?
 - What language(s)?
 - MPI? OpenMP? CUDA?
 - Compute-intense? Data intense? I/O intense?
- What do I want to **measure**? And how?
 - Wall clock time?
 - Hardware performance counters?
 - Probes? Library wrapping? Sampling?
 - Profiles? Traces? Both?

T-A-M Model for Performance Engineering

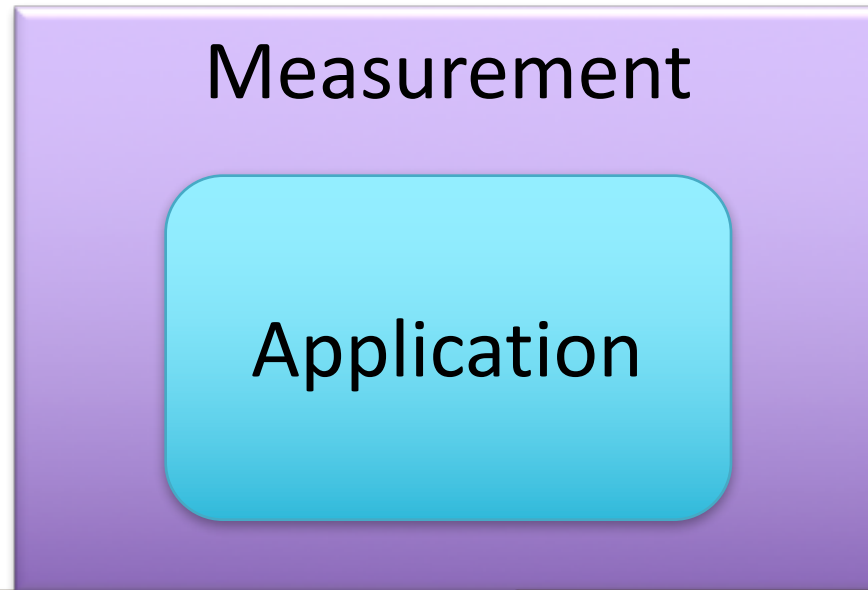
- Target
 - Installed software
 - Available compilers
 - Host architecture/OS
- Application
 - MPI, OpenMP, CUDA, OpenACC, etc.
- Measurement
 - Profile, trace, or both
 - Sample, source inst...



**Experiment =
(Target, Application, Measurement)**

Which platform is best for my application?

- **Many** targets
 - Possibly same hardware, different software
- **One** measurement
- **One** application



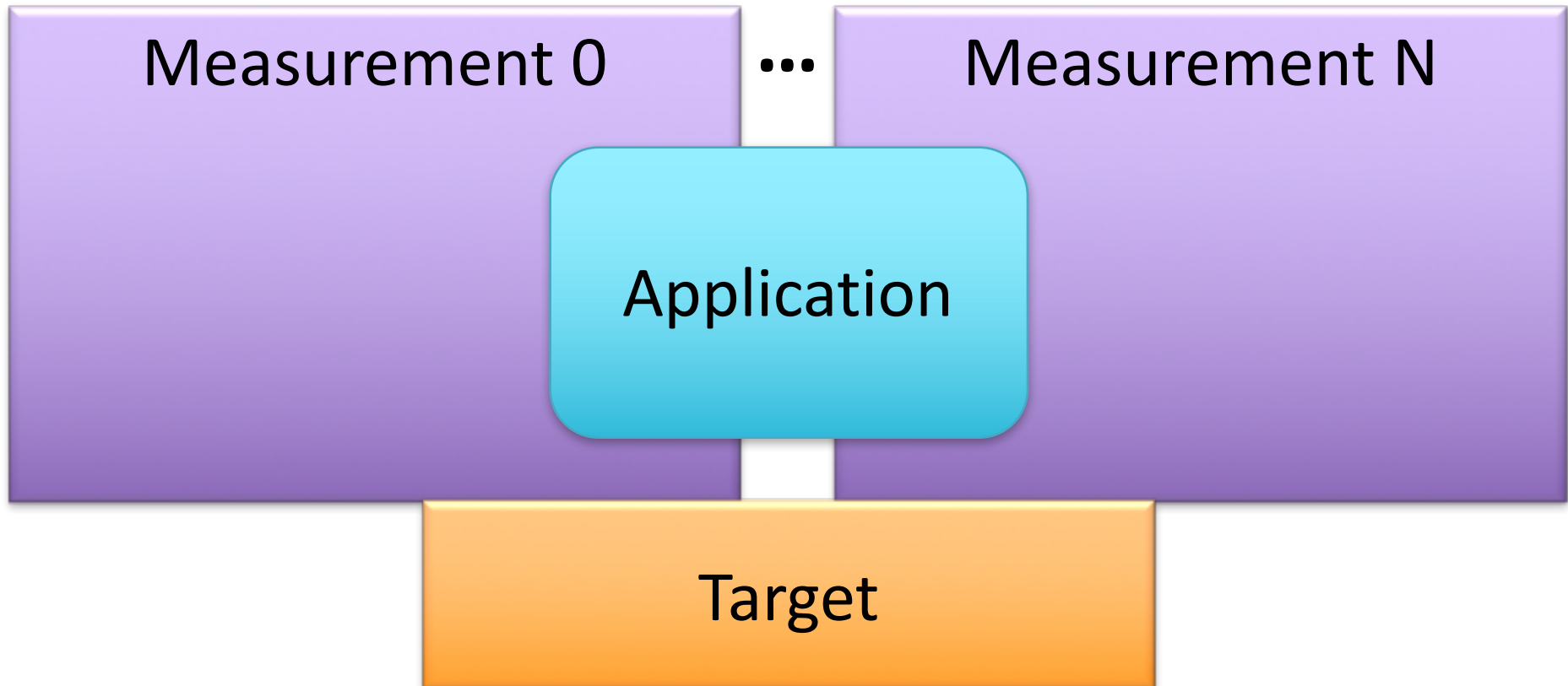
Target 0

...

Target N

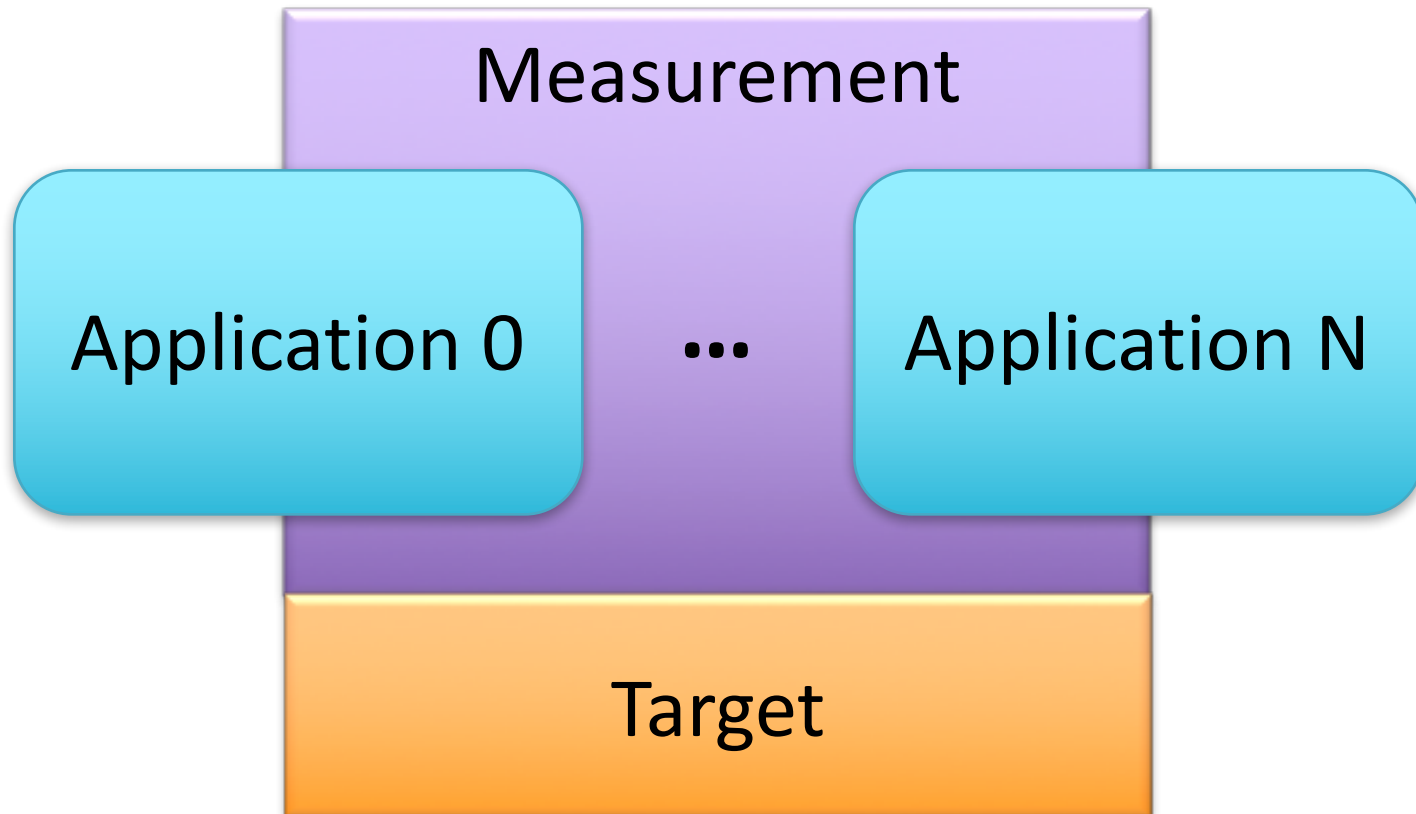
What are the performance characteristics of my application?

- **One** target
- **Many** measurements
- **One** application



How well does my target perform various tasks?

- One target
- One measurement
- **Many** applications (e.g. benchmarks)



TAU Commander

```
mm -- ssh -- 154x53
jlinford@east03 ~/workspace/taucmdr/examples/mm $ tau dash
== Targets (/home/jlinford/.tau) ==
-----
| Name | Host OS | Host Arch | C | C++ | Fortran | In Projects |
-----+-----+-----+-----+-----+-----+-----
| localhost | Linux | x86_64 | /usr/bin/gcc | /usr/bin/g++ | /usr/bin/gfortran | ex-mm |
-----

== Applications (/home/jlinford/.tau) ==
-----
| Name | OpenMP | Pthreads | MPI | CUDA | MIC | SHMEM | MPC | In Projects |
-----+-----+-----+-----+-----+-----+-----+-----+-----
| ex-mm-serial | | | | | | | | ex-mm |
-----+-----+-----+-----+-----+-----+-----+-----+-----
| ex-mm-openmp | Yes | | | | | | | ex-mm |
-----+-----+-----+-----+-----+-----+-----+-----+-----
| ex-mm-openmp-mpi | Yes | | Yes | | | | | ex-mm |
-----+-----+-----+-----+-----+-----+-----+-----+-----

== Measurements (/home/jlinford/.tau) ==
-----
| Name | Profile | Trace | Sample | Source Inst. | Compiler Inst. | MPI | OpenMP | Callpath Depth | Mem. Usage | Mem. Alloc | In Projects |
-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----
| ex-profile | Yes | No | No | automatic | fallback | No | compiler_default | 0 | No | No | ex-mm |
-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----
| ex-trace | No | Yes | No | automatic | fallback | No | compiler_default | 0 | No | No | ex-mm |
-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----
| ex-sample | Yes | No | Yes | never | never | No | compiler_default | 0 | No | No | ex-mm |
-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----

== Projects (/home/jlinford/.tau) ==
-----
| Name | Targets | Applications | Measurements | Home |
-----+-----+-----+-----+-----
| ex-mm | localhost | ex-mm-serial | ex-profile | /home/jlinford/.tau |
| | | ex-mm-openmp | ex-trace | |
| | | ex-mm-openmp-mpi | ex-sample | |
-----+-----+-----+-----+-----

== ex-mm (localhost, ex-mm-openmp, ex-profile) Trials ==
-----
No trials. Use 'tau <command>' or 'tau trial create <command>' to create a new trial

jlinford@east03 ~/workspace/taucmdr/examples/mm $
```

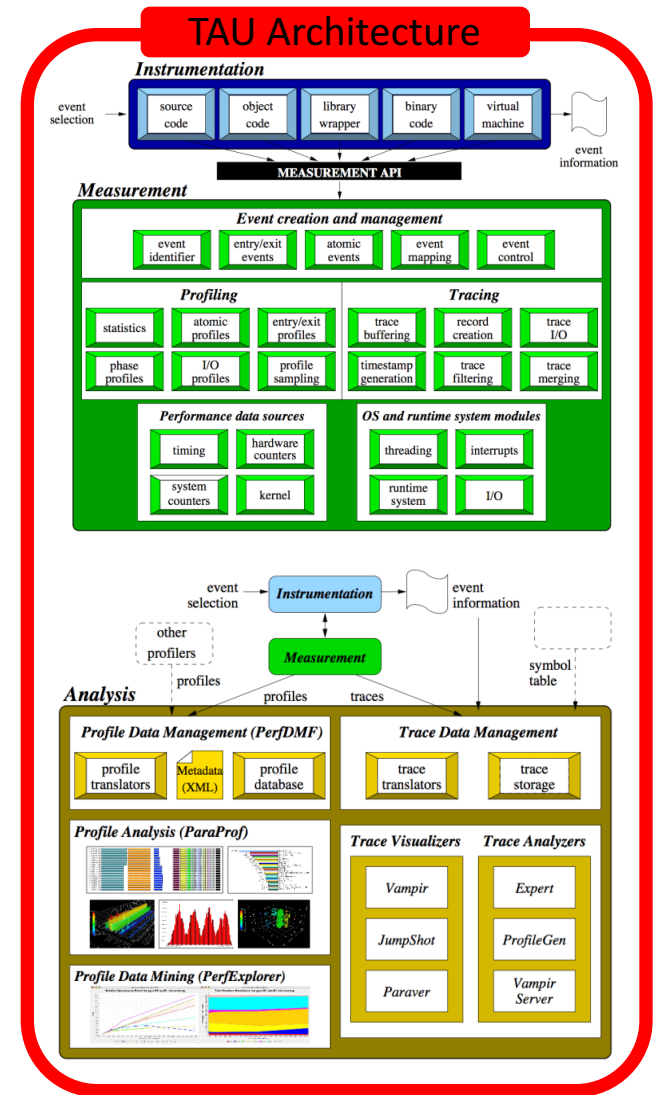


Intuitive Performance Engineering

THE TAU PERFORMANCE SYSTEM

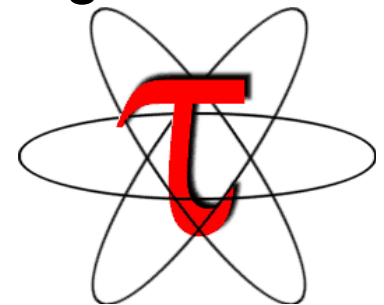
The TAU Performance System[®]

- *Integrated toolkit* for performance problem solving
 - Instrumentation, measurement, analysis, visualization
 - Portable profiling and tracing
 - Performance data management and data mining
- Direct and indirect measurement
- *Free, open source, BSD license*
- Available on all HPC platforms (and some non-HPC)
- <http://tau.uoregon.edu/>



The TAU Performance System[®]

- Tuning and Analysis Utilities (**20+ year project**)
- Comprehensive performance profiling and tracing
 - Integrated, scalable, flexible, portable
 - Targets all parallel programming/execution paradigms
- Integrated performance toolkit
 - Instrumentation, measurement, analysis, visualization
 - Widely-ported performance profiling / tracing system
 - Performance data management and data mining
 - Open source (BSD-style license)
- Integrates with application frameworks



TAU Supports All HPC Platforms

C/C++
Fortran
pthreads
Intel
MinGW

CUDA
OpenACC
Intel MIC
GNU
LLVM
Linux
BlueGene
Android

UPC
OpenACC
Intel MIC
PGI
Windows
Fujitsu
MPC

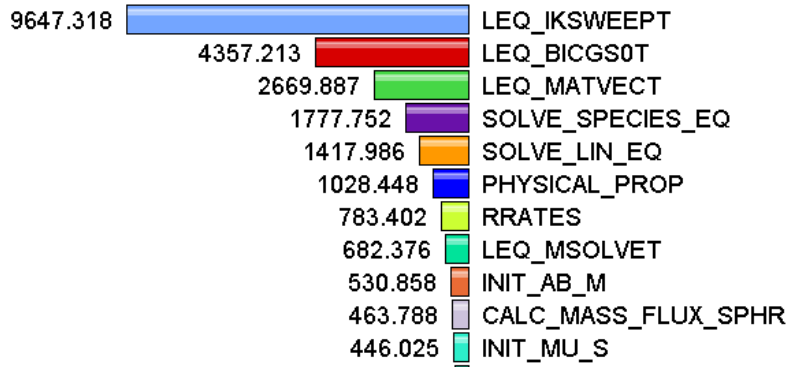
GPI
Java
OpenMP
Cray
AIX
ARM
OS X

Python
MPI
Sun

Insert
yours
here

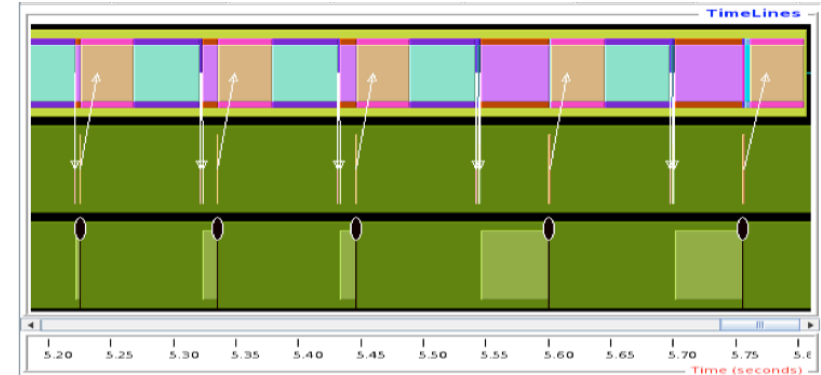
Measurement Approaches

Profiling



Shows
how much time
was spent in each
routine

Tracing

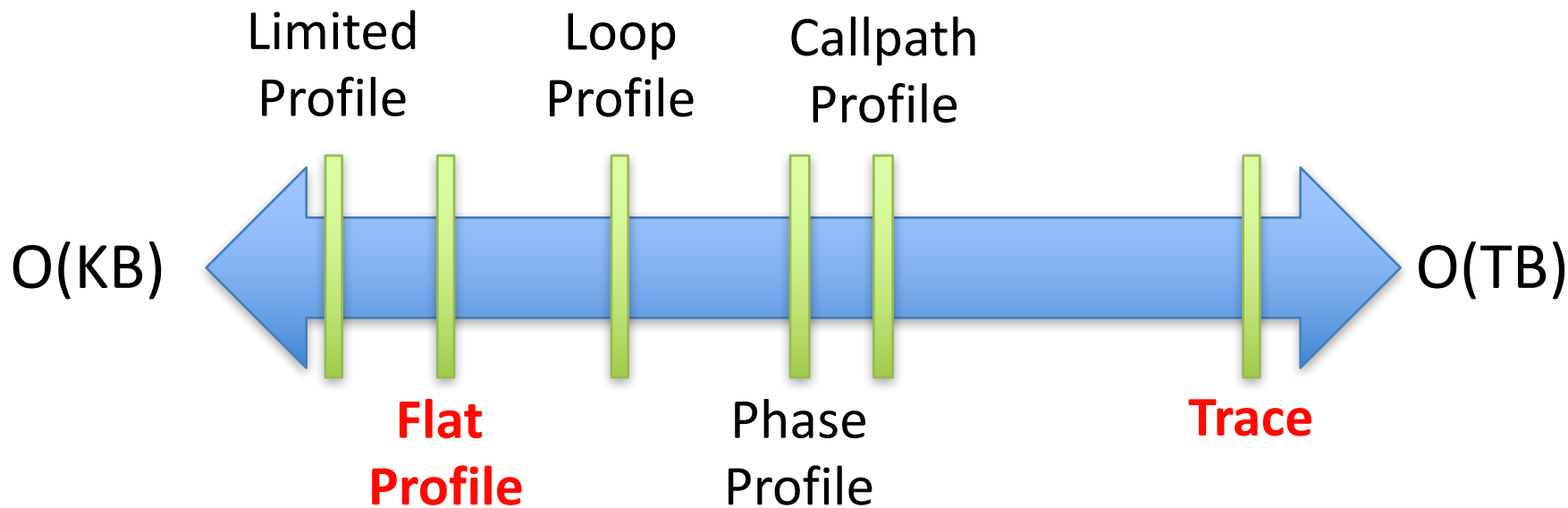


Shows
when events
take place on a
timeline

Types of Performance Profiles

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

How much data do you want?



All levels support multiple metrics/counters

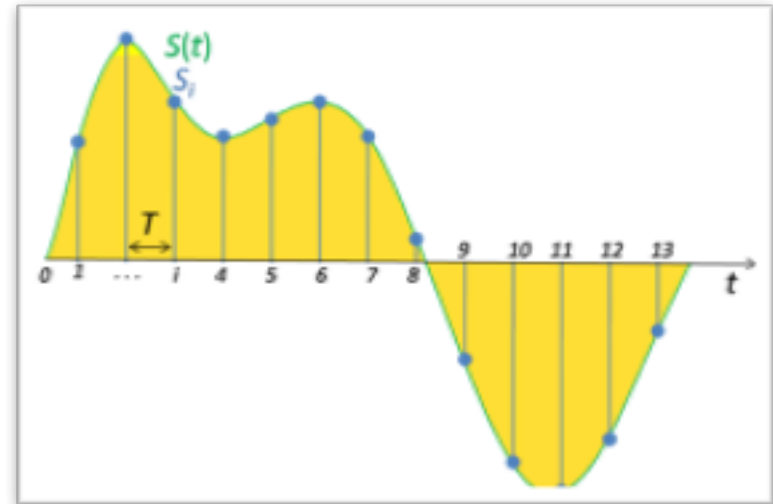
Performance Data Measurement

Direct via Probes

```
call TAU_START('potential')  
// code  
call TAU_STOP('potential')
```

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

Indirect via Sampling



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

Insert TAU API Calls Automatically

- Use TAU's compiler wrappers
 - Replace `cxx` with `tau_cxx.sh`, etc.
 - Automatically instruments source code, links with TAU libraries.
- Use `tau_cc.sh` for C, `tau_f90.sh` for Fortran, etc.

Makefile without TAU

```
CXX = mpicxx
F90 = mpif90
CXXFLAGS =
LIBS = -lm
OBJS = f1.o f2.o f3.o ... fn.o

app: $(OBJS)
    $(CXX) $(LDFLAGS) $(OBJS) -o $@
    $(LIBS)
.cpp.o:
    $(CXX) $(CXXFLAGS) -c $<
```

Makefile with TAU

```
CXX = tau_cxx.sh
F90 = tau_f90.sh
CXXFLAGS =
LIBS = -lm
OBJS = f1.o f2.o f3.o ... fn.o

app: $(OBJS)
    $(CXX) $(LDFLAGS) $(OBJS) -o $@
    $(LIBS)
.cpp.o:
    $(CXX) $(CXXFLAGS) -c $<
```

TAU Workflow

Instrumentation

Source

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

Library

- Interposition (PMPI, GASNET, ...)
- Wrapper generation

Linker

- Static, Dynamic
- Preloading (LD_PRELOAD)

Executable

- Dynamic (Dyninst)
- Binary (Dininst, MAQAO, PEBIL)

Measurement

Events

- Static, Dynamic
- Routine, Block, Loop
- Threading, Communication
- Heterogeneous

Profiling

- Flat, Callpath, Phase, Snapshot
- Probe, Sampling, Compiler, Hybrid

Tracing

- TAU, Scalasca, ScoreP
- Open Trace Format (OTF)

Metadata

- System
- User defined

Analysis

Profiles

- ParaProf analyzer & visualizer
 - 3D profile data visualization
 - Communication matrix
 - Callstack analysis
 - Graph generation
- PerfDMF
- PerfExplorer profile data miner

Traces

- OTF, SLOG-2
- Vampir
- Jumpshot

Online

- Event unification
- Statistics calculation

Instrument: Add Probes

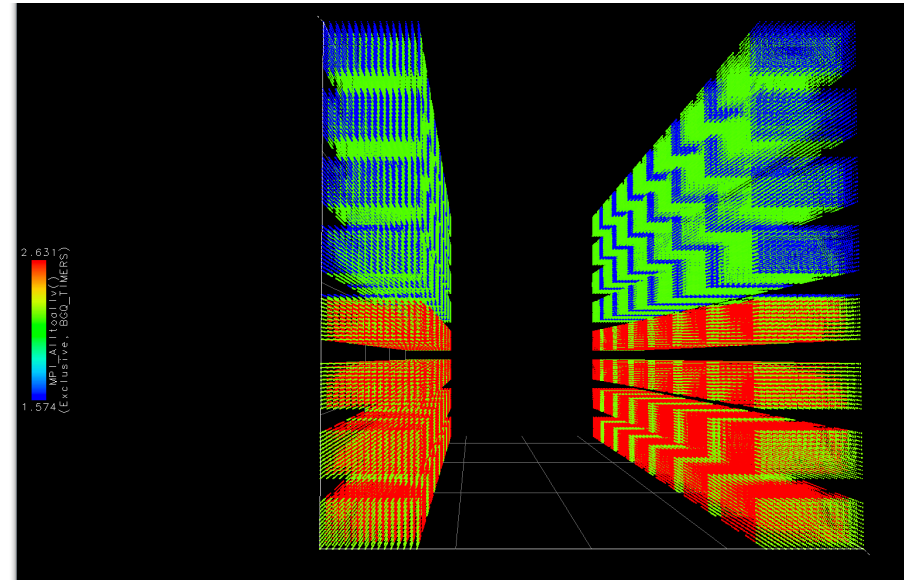
- *Source code* instrumentation
 - PDT parsers, pre-processors
- *Wrap* external libraries
 - I/O, MPI, Memory, CUDA, OpenCL, pthread
- *Rewrite* the binary executable
 - Dyninst, MAQAO

Measure: Gather Data

- Direct measurement via *probes*
- Indirect measurement via *sampling*
- Throttling and runtime control
- Interface with external packages (PAPI)

Analyze: Synthesize Knowledge

- Data *visualization*
- Data *mining*
- Statistical analysis
- Import/export performance data



Using TAU: A Brief Introduction

- Each configuration of TAU corresponds to a unique stub makefile (*TAU_MAKEFILE*) in the TAU installation directory

```
% soft add +tau-latest
```

```
% ls $TAU/Makefile.*
```

```
Makefile.tau-bgqtimers-papi-gnu-ompt-pdt-openmp
```

```
Makefile.tau-bgqtimers-papi-mpi-pdt
```

```
Makefile.tau-bgqtimers-papi-mpi-pdt-openmp-opari
```

```
Makefile.tau-bgqtimers-papi-mpi-pdt-scorep
```

```
Makefile.tau-bgqtimers-papi-mpi-pthread-pdt
```

```
Makefile.tau-bgqtimers-papi-pdt
```

```
Makefile.tau-gnu-bgqtimers-papi-mpi-pdt
```

```
Makefile.tau-gnu-bgqtimers-papi-mpi-pdt-openmp-opari
```

```
Makefile.tau-gnu-bgqtimers-papi-mpi-pthread-pdt
```

```
% export TAU_MAKEFILE=$TAU/Makefile.tau-bgqtimers-papi-mpi-pdt
```


Using TAU: A Brief Introduction

1. Choose an appropriate TAU_MAKEFILE:

```
% soft add +tau-latest
% export TAU_MAKEFILE=/soft/perfutils/tau/tau_latest/
   bgq/lib/Makefile.tau-bgqtimers-papi-mpi-pdt
% export TAU_OPTIONS='-optVerbose ...'
   # (see tau_compiler.sh -help for more options)
```

2. Use `tau_f90.sh`, `tau_cxx.sh`, etc. as Fortran, C++, etc. compiler:

```
% mpixlf90_r foo.f90
   changes to
% tau_f90.sh foo.f90
```

3. Execute application:

```
% qsub -A <queue> -q R.bc -n 256 -t 10 ./a.out
```

Note: If TAU_MAKEFILE has “**papi**” in its name, set **TAU_METRICS**:

```
% qsub --env TAU_METRICS=BGQ_TIMERS:PAPI_L2_DCM...
```

4. Analyze performance data:

pprof (for text based profile display)

paraprof (for GUI)

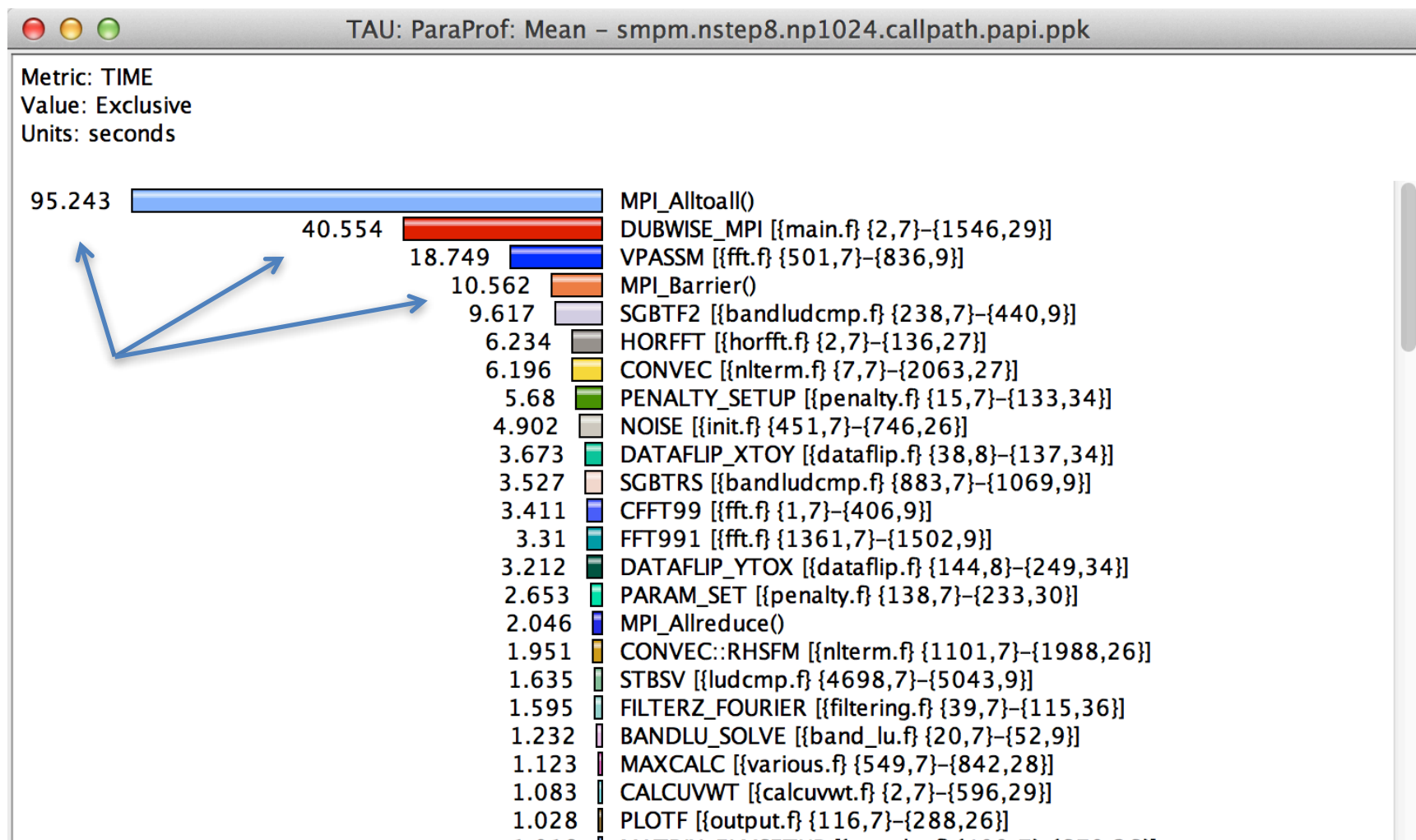
Hands-on (18:30 – 21:15)

```
% ssh mira.alcf.anl.gov
% tar xvzf /soft/perftools/tau/workshop.tgz
% cd workshop
% less README
```

For an MPI+F90 application, you may want to start with:

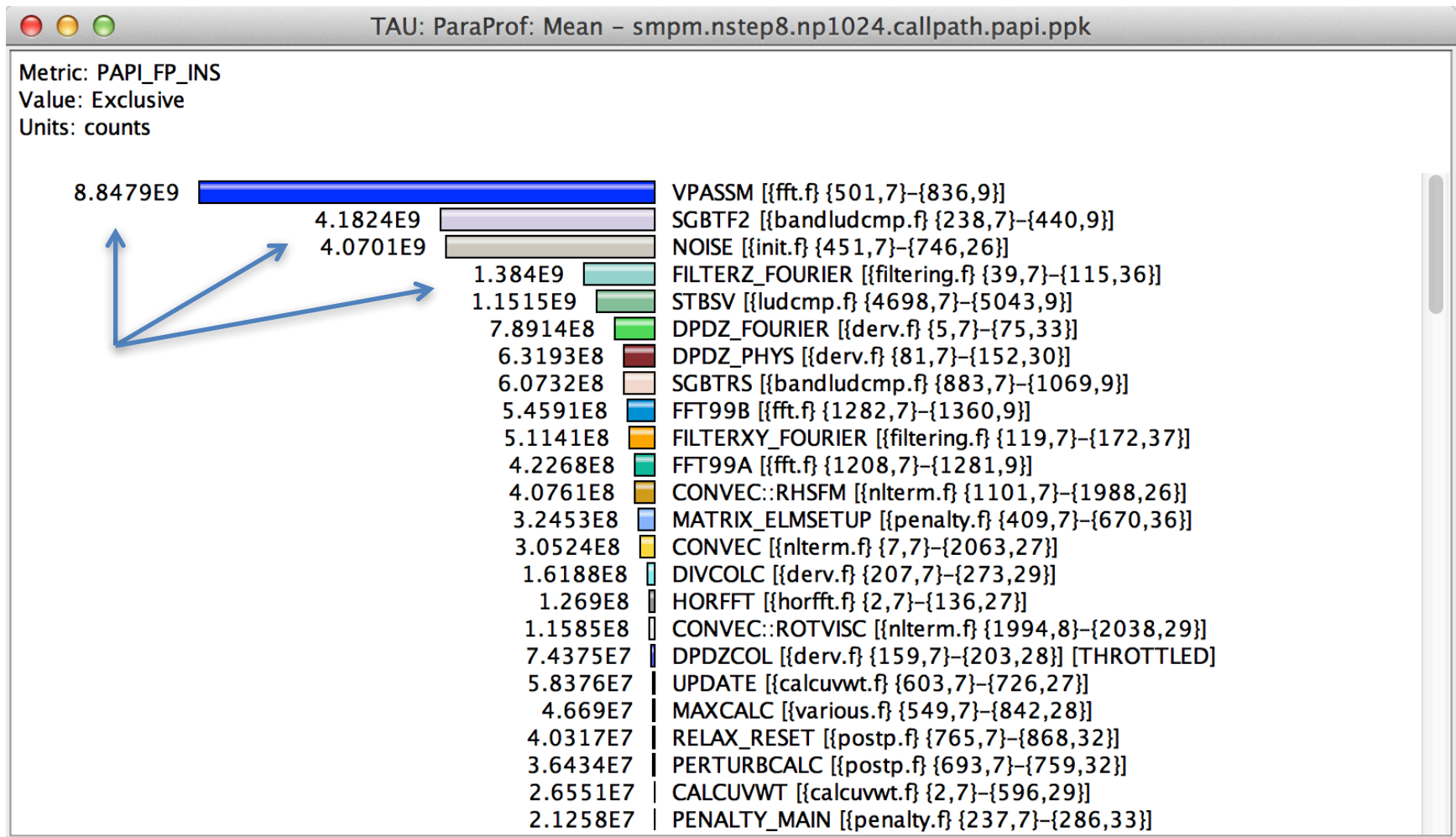
```
% soft add +tau-latest
% export TAU_MAKEFILE=
    $TAU/Makefile.tau-bgqtimers-papi-mpi-pdt
% make CC=tau_cc.sh CXX=tau_cxx.sh F90=tau_f90.sh
% qsub -q R.bc -n 2 --mode c16 -t 10 -A ... ./a.out
% paraprof
```

How Much Time per Code Region?



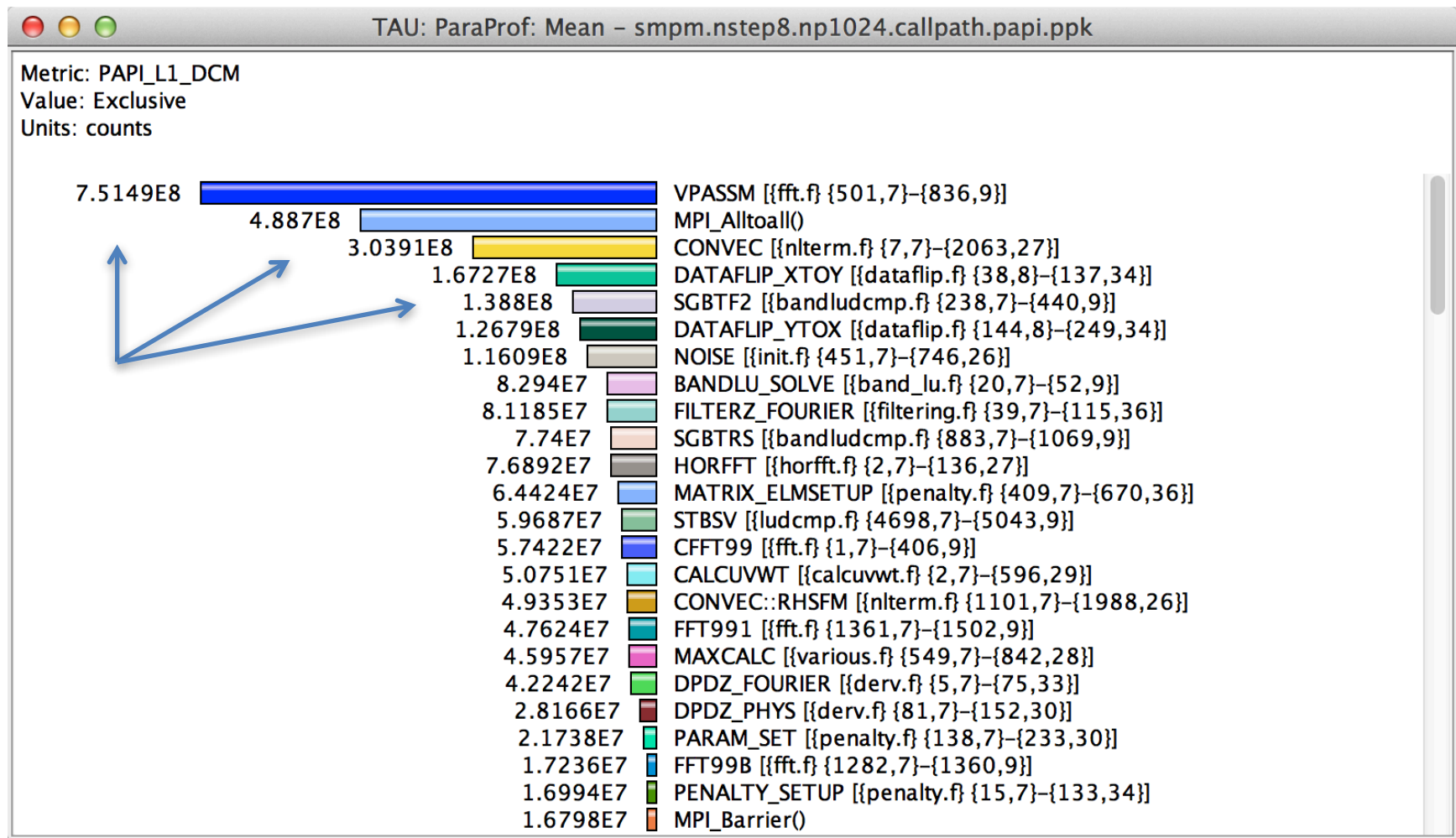
% **paraprof** (Click on label, e.g. "Mean" or "node 0")

How Many Instructions per Code Region?



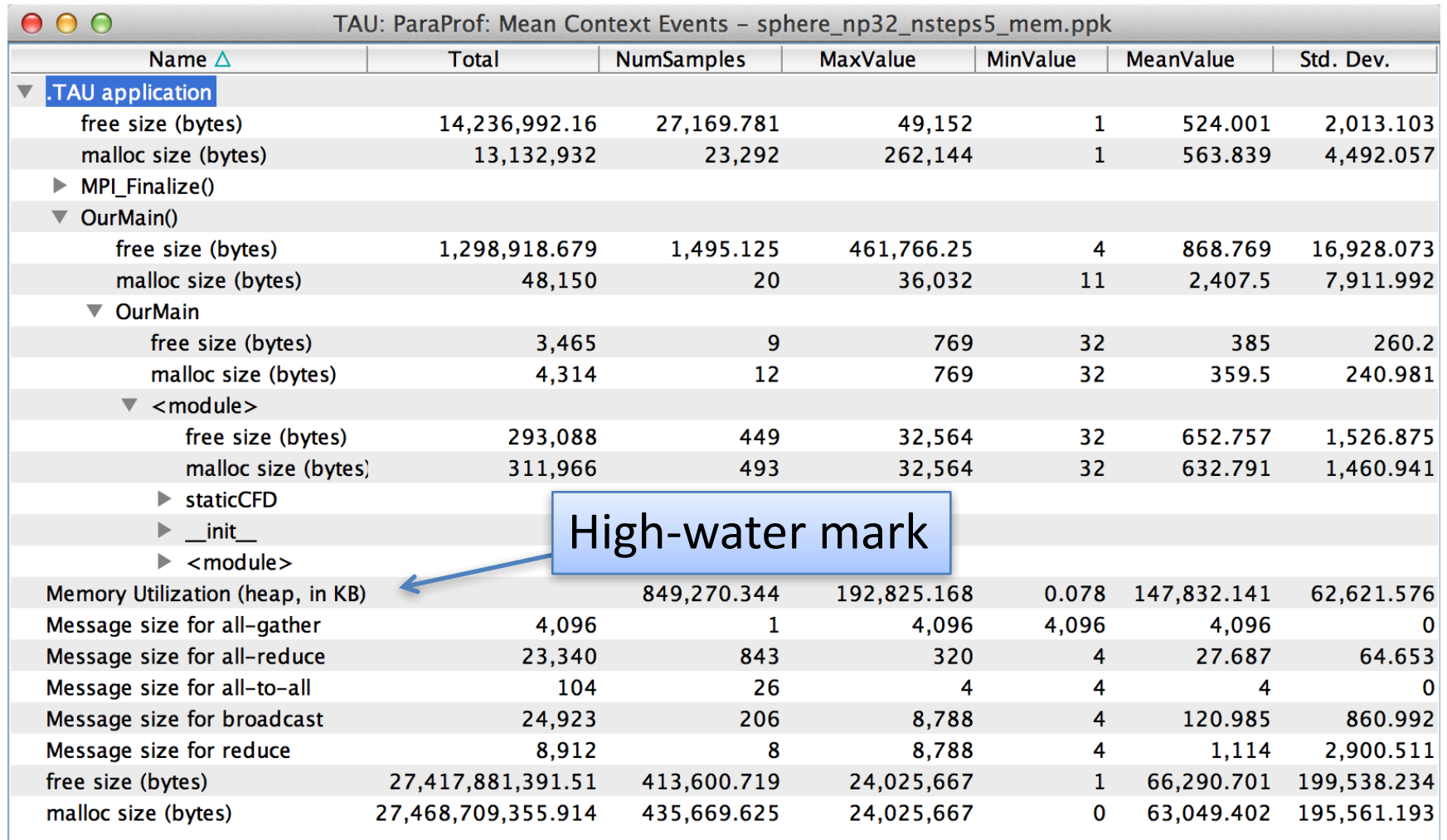
% paraprof (Options → Select Metric... → Exclusive... → PAPI_FP_INS)

How Many L1 or L2 Cache Misses?



% paraprof (Options → Select Metric... → Exclusive... → PAPI_L1_DCM)

How Much Memory Does the Code Use?



Name Δ	Total	NumSamples	MaxValue	MinValue	MeanValue	Std. Dev.
▼ TAU application						
free size (bytes)	14,236,992.16	27,169.781	49,152	1	524.001	2,013.103
malloc size (bytes)	13,132,932	23,292	262,144	1	563.839	4,492.057
▶ MPI_Finalize()						
▼ OurMain()						
free size (bytes)	1,298,918.679	1,495.125	461,766.25	4	868.769	16,928.073
malloc size (bytes)	48,150	20	36,032	11	2,407.5	7,911.992
▼ OurMain						
free size (bytes)	3,465	9	769	32	385	260.2
malloc size (bytes)	4,314	12	769	32	359.5	240.981
▼ <module>						
free size (bytes)	293,088	449	32,564	32	652.757	1,526.875
malloc size (bytes)	311,966	493	32,564	32	632.791	1,460.941
▶ staticCFD						
▶ __init__						
▶ <module>						
Memory Utilization (heap, in KB)		849,270.344	192,825.168	0.078	147,832.141	62,621.576
Message size for all-gather	4,096	1	4,096	4,096	4,096	0
Message size for all-reduce	23,340	843	320	4	27.687	64.653
Message size for all-to-all	104	26	4	4	4	0
Message size for broadcast	24,923	206	8,788	4	120.985	860.992
Message size for reduce	8,912	8	8,788	4	1,114	2,900.511
free size (bytes)	27,417,881,391.51	413,600.719	24,025,667	1	66,290.701	199,538.234
malloc size (bytes)	27,468,709,355.914	435,669.625	24,025,667	0	63,049.402	195,561.193

⌘ paraprof (Right-click label [e.g “node 0”] → Show Context Event Window)

How Much Memory Does the Code Use?

Name Δ	Total	NumSamples	MaxValue	MinValue	MeanValue	Std. Dev.
▼ TAU application						
free size (bytes)	14,236,992.16	27,169.781	49,152	1	524.001	2,013.103
malloc size (bytes)	13,132,932	23,292	262,144	1	563.839	4,492.057
▶ MPI_Finalize()						
▼ OurMain()						
free size (bytes)	1,298,918.679	1,495.125	461,766.25	4	868.769	16,928.073
malloc size (bytes)	48,150	20	36,032	11	2,407.5	7,911.992
▼ OurMain						
free size (bytes)	3,465	9	769	32	385	260.2
malloc size (bytes)	4,314	12	769	32	359.5	240.981
▼ <module>						
free size (bytes)	293,088	449	32,564	32	652.757	1,526.875
malloc size (bytes)	311,966	493	32,564	32	632.791	1,460.941
▶ staticCFD						
▶ __init__						
▶ <module>						
Memory Utilization (heap, in KB)		849,270.344	192,825.168	0.078	147,832.141	62,621.576
Message size for all-gather	4,096	1	4,096	4,096	4,096	0
Message size for all-reduce	23,340	843	320	4	27.687	64.653
Message size for all-to-all	104	26	4	4	4	0
Message size for broadcast	24,923	206	8,788	4	120.985	860.992
Message size for reduce	8,912	8	8,788	4	1,114	2,900.511
free size (bytes)	27,417,881,391.51	413,600.719	24,025,667	1	66,290.701	199,538.234
malloc size (bytes)	27,468,709,355.914	435,669.625	24,025,667	0	63,049.402	195,561.193

⌘ paraprof (Right-click label [e.g “node 0”] → Show Context Event Window)

Where is Memory Allocated / Deallocated?

TAU: ParaProf: Mean Context Events – sphere_np32_nsteps5_mem.ppk

Name	Total	NumSamples	MaxValue	MinValue	MeanValue	Std. Dev.
▼ TAU application						
free size (bytes)	14,236,992.16	27,169.781	49,152	1	524.001	2,013.103
malloc size (bytes)	13,132,932	23,292	262,144	1	563.839	4,492.057
▶ MPI_Finalize()						
▼ OurMain()						
free size (bytes)	1,298,918.679	1,495.125	461,766.25	4	868.769	16,928.073
malloc size (bytes)	48,150	20	36,032	11	2,407.5	7,911.992
▼ OurMain						
free size (bytes)	3,465	9	769	32	385	260.2
malloc size (bytes)	4,314	12	769	32	359.5	240.981
▼ <module>						
free size (bytes)	293,088	449	32,564	32	652.757	1,526.875
malloc size (bytes)	311,966	493	32,564	32	632.791	1,460.941
▶ staticCFD						
▶ __init__						
▶ <module>						
Memory Utilization (heap, in KB)		849,270.344	192,825.168	0.078	147,832.141	62,621.576
Message size for all-gather	4,096	1	4,096	4,096	4,096	0
Message size for all-reduce	23,340	843	320	4	27.687	64.653
Message size for all-to-all	104	26	4	4	4	0
Message size for broadcast	24,923	206	8,788	4	120.985	860.992
Message size for reduce	8,912	8	8,788	4	1,114	2,900.511
free size (bytes)	27,417,881,391.51	413,600.719	24,025,667	1	66,290.701	199,538.234
malloc size (bytes)	27,468,709,355.914	435,669.625	24,025,667	0	63,049.402	195,561.193

Allocation / Deallocation Events

⌘ paraprof (Right-click label [e.g “node 0”] → Show Context Event Window)

What are the I/O Characteristics?

TAU: ParaProf: Context Events for thread: n,c,t, 1,0,0 - samarc_obe_4p_iomem_cp.ppk


Name	Total	MeanValue	NumSamples	MinValue	MaxValue	Std. Dev.
▼ .TAU application						
▶ read()						
▶ fopen64()						
▶ fclose()						
▼ OurMain()						
malloc size	25,235	1,097.174	23	11	12,032	2,851.143
free size	22,707	1,746.692	13	11	12,032	3,660.642
▼ OurMain [{{wrapper.py}}{3}]						
▶ read()						
malloc size	3,877	323.083	12	32	981	252.72
free size	1,536	219.429	7	32	464	148.122
▶ fopen64()						
▶ fclose()						
▼ <module> [{{obe.py}}{8}]						
▼ writeRestartData [{{samarcInterface.py}}{145}]						
▼ samarcWriteRestartData						
▼ write()						
WRITE Bandwidth (MB/s) <file="samarc/restore.00002/nodes.00004/proc.00001">		74.565	117	0	2,156.889	246.386
WRITE Bandwidth (MB/s) <file="samarc/restore.00001/nodes.00004/proc.00001">		77.594	117	0	1,941.2	228.366
WRITE Bandwidth (MB/s)		76.08	234	0	2,156.889	237.551
Bytes Written <file="samarc/restore.00002/nodes.00004/proc.00001">	2,097,552	17,927.795	117	1	1,048,576	133,362.946
Bytes Written <file="samarc/restore.00001/nodes.00004/proc.00001">	2,097,552	17,927.795	117	1	1,048,576	133,362.946
Bytes Written	4,195,104	17,927.795	234	1	1,048,576	133,362.946
▶ open64()						


Write bandwidth per file

Bytes written to each file

% **paraprof** (Right-click label [e.g “node 0”] → Show Context Event Window)

What are the I/O Characteristics?

Name 	Total	NumSamples	MaxValue	MinValue	MeanValue	Std. Dev.
▶ Incl						
▶ Initialize						
▶ LoadBodyEuler						
▶ LoadMesh						
MPI-IO Bytes Written	4,328,712	144	893,152	0	30,060.5	128,042.696
MPI-IO Write Bandwidth (MB/s)		144	196.86	0	3.421	16.87
▶ MPI_Allgatherv()						
▶ MPI_Bcast()						
▶ MPI_Comm_create()						
▶ MPI_File_close()						
▶ MPI_File_open()						
▶ MPI_File_write_all()						
▶ MPI_File_write_at()						
▶ MPI_Finalize()						
▶ MPI_Gather()						
▶ MPI_Gatherv()						



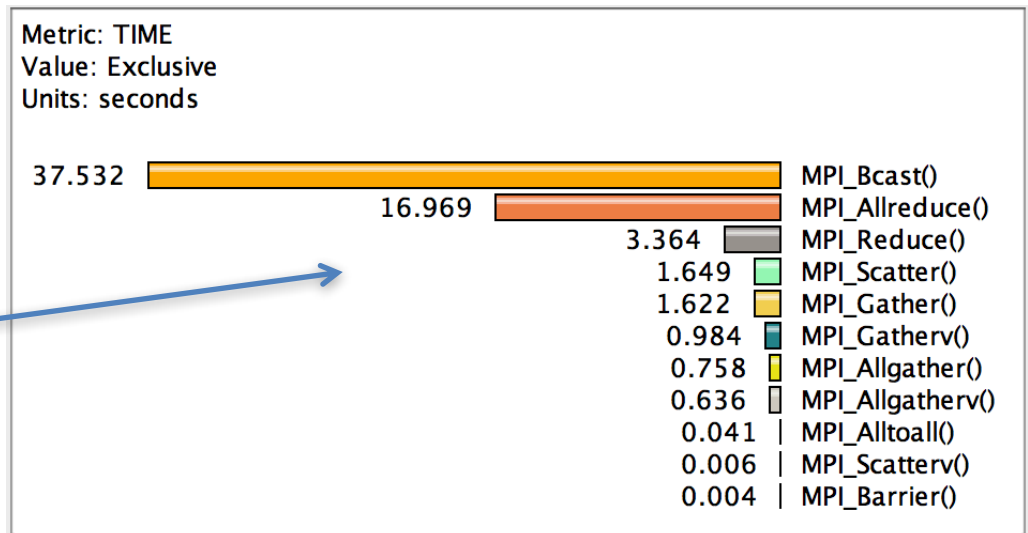
⌘ **paraprof** (Right-click label [e.g “node 0”] → Show Context Event Window)

How Much Time is spent in Collectives?

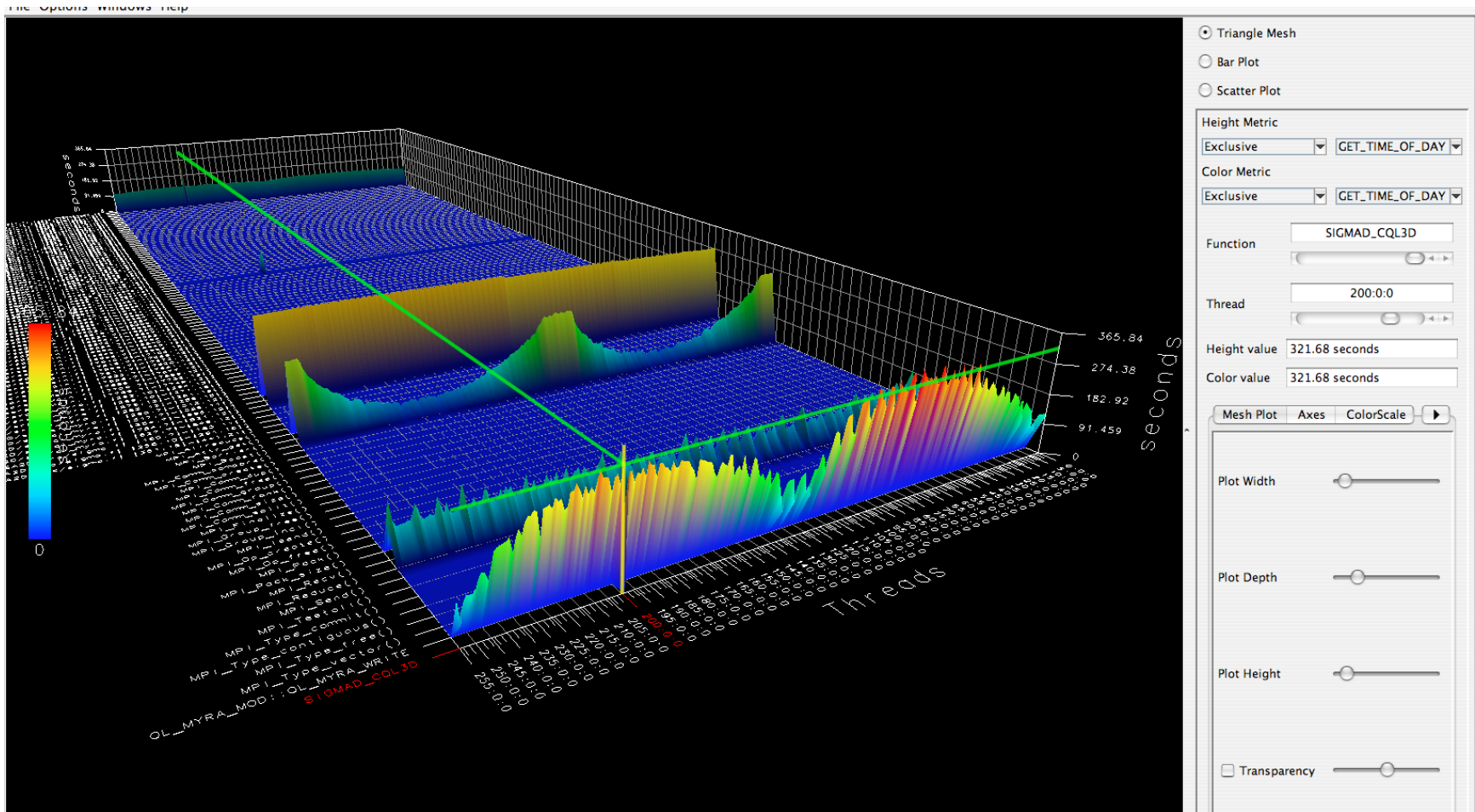
Name Δ	Total	Num...	MaxValue	MinValue	MeanValue	Std. Dev.
▶ MPI_Wait()						
▶ MPI_Waitall()						
Message size for all-gather	305,753,268	72	172,215,296	4	4,246,573.167	22,551,605.859
Message size for all-reduce	163,308	632	21,908	4	258.399	897.725
Message size for all-to-all	112	14	8	8	8	0
Message size for broadcast	692,208,045.5	3,346	18,117,620	0	206,876.284	1,284,673.036
Message size for gather	6,901,452.378	15.312	1,387,306.625	4	450,707.094	483,216.499
Message size for reduce	66,812	1,520	56	4	43.955	21.598
Message size for scatter	63,147.906	146	62,567.906	4	432.52	5,160.063

Message sizes

Time spent in collectives

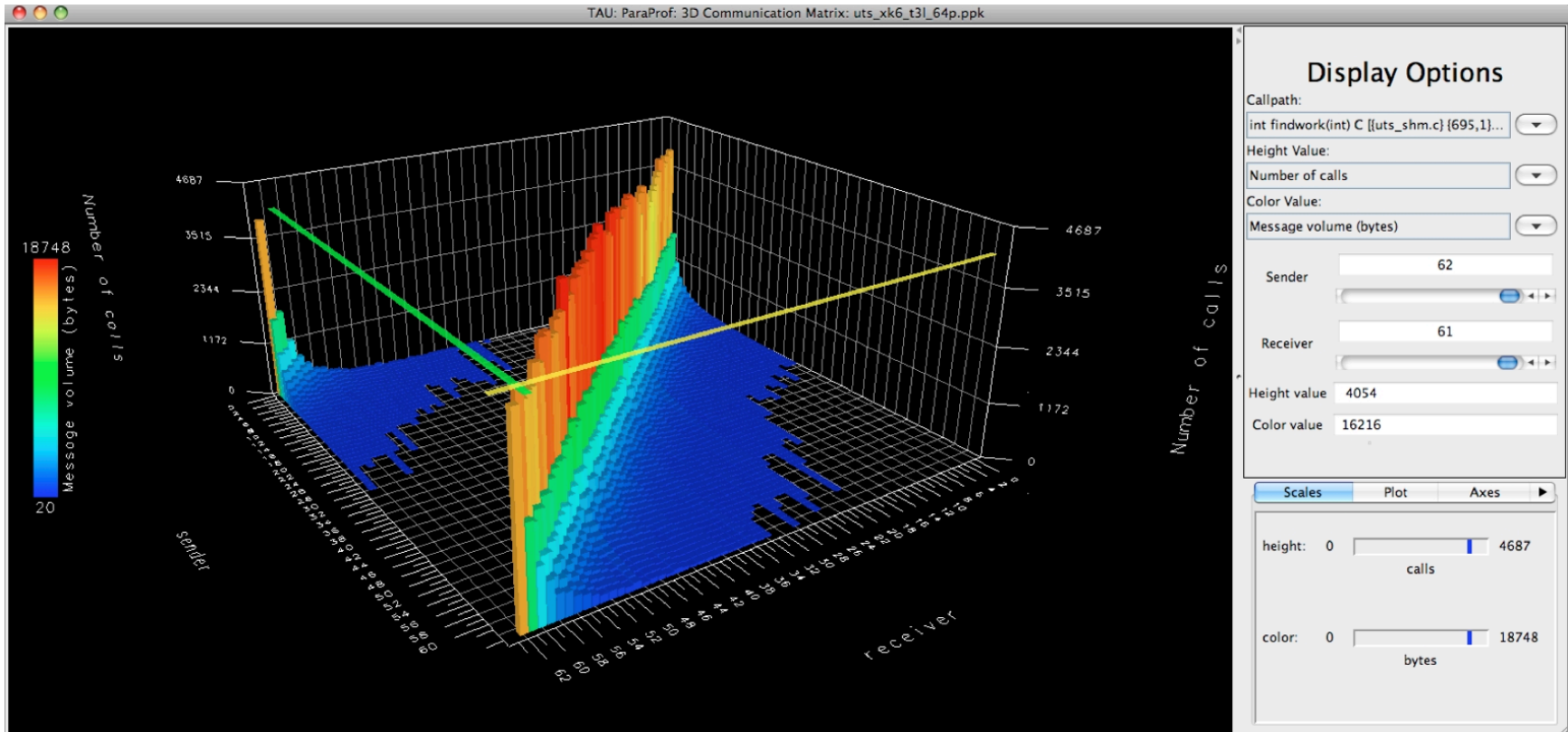


3D Profile Visualization



% paraprof (Windows → 3D Visualization)

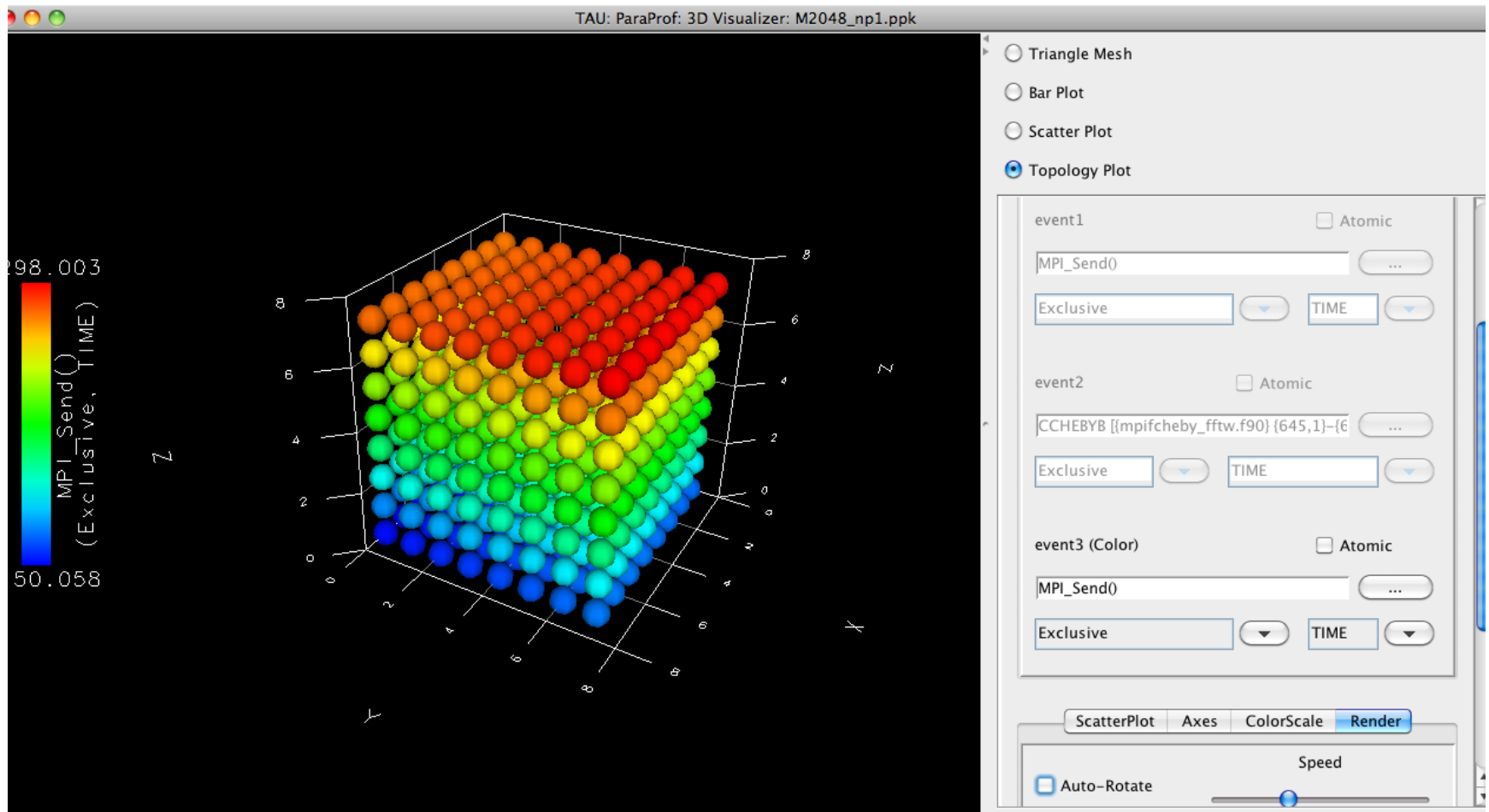
3D Communication Visualization



```
% qsub -env TAU_COMM_MATRIX=1 ...
```

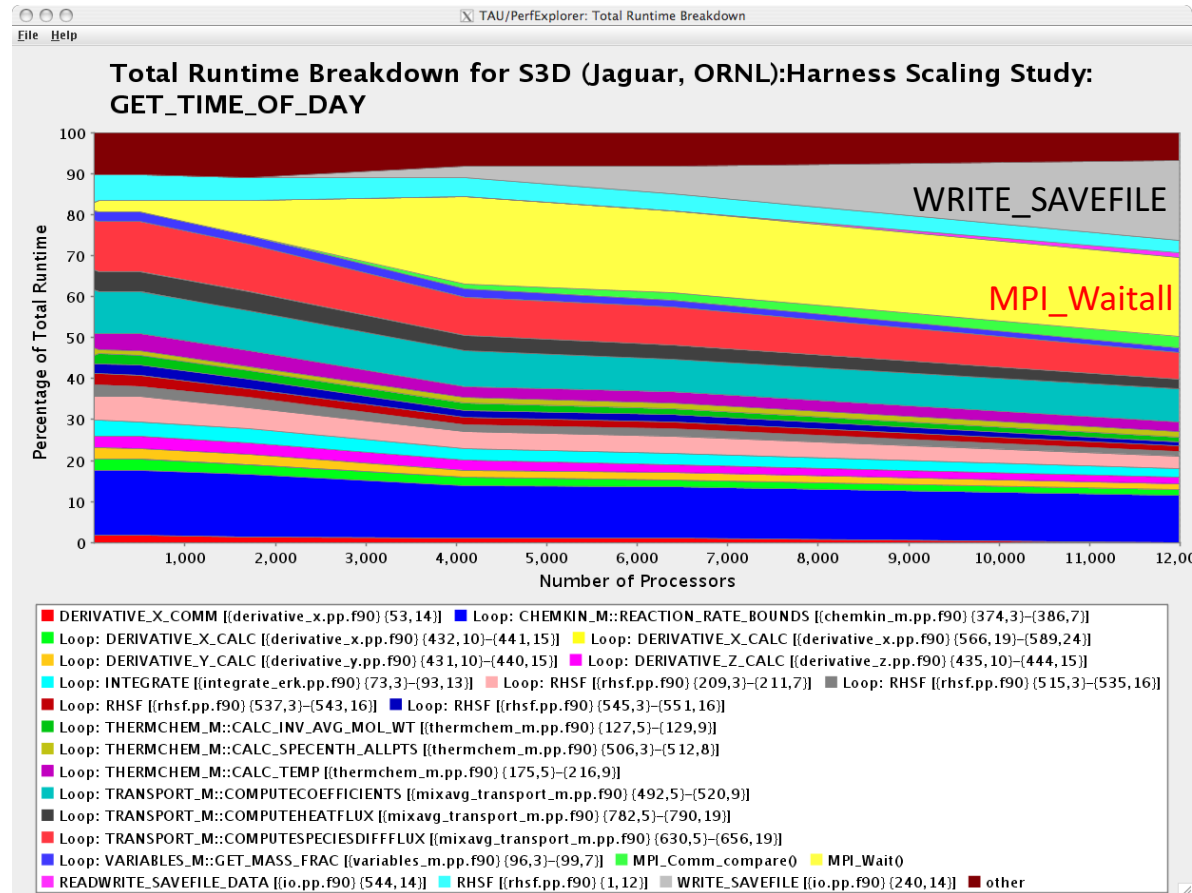
```
% paraprof (Windows → 3D Communication Matrix)
```

3D Topology Visualization



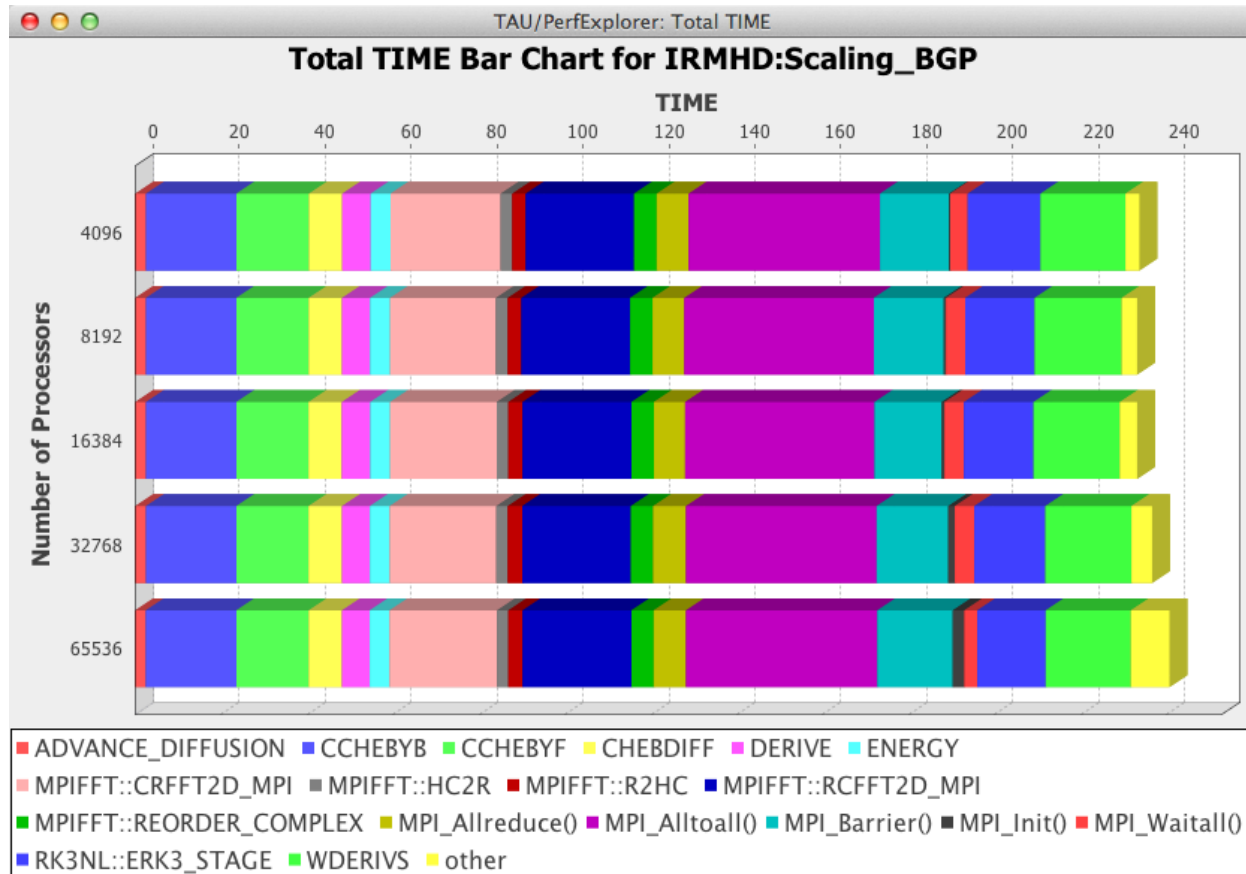
% **paraprof** (Windows → 3D Visualization → Topology Plot)

How Does Each Routine Scale?



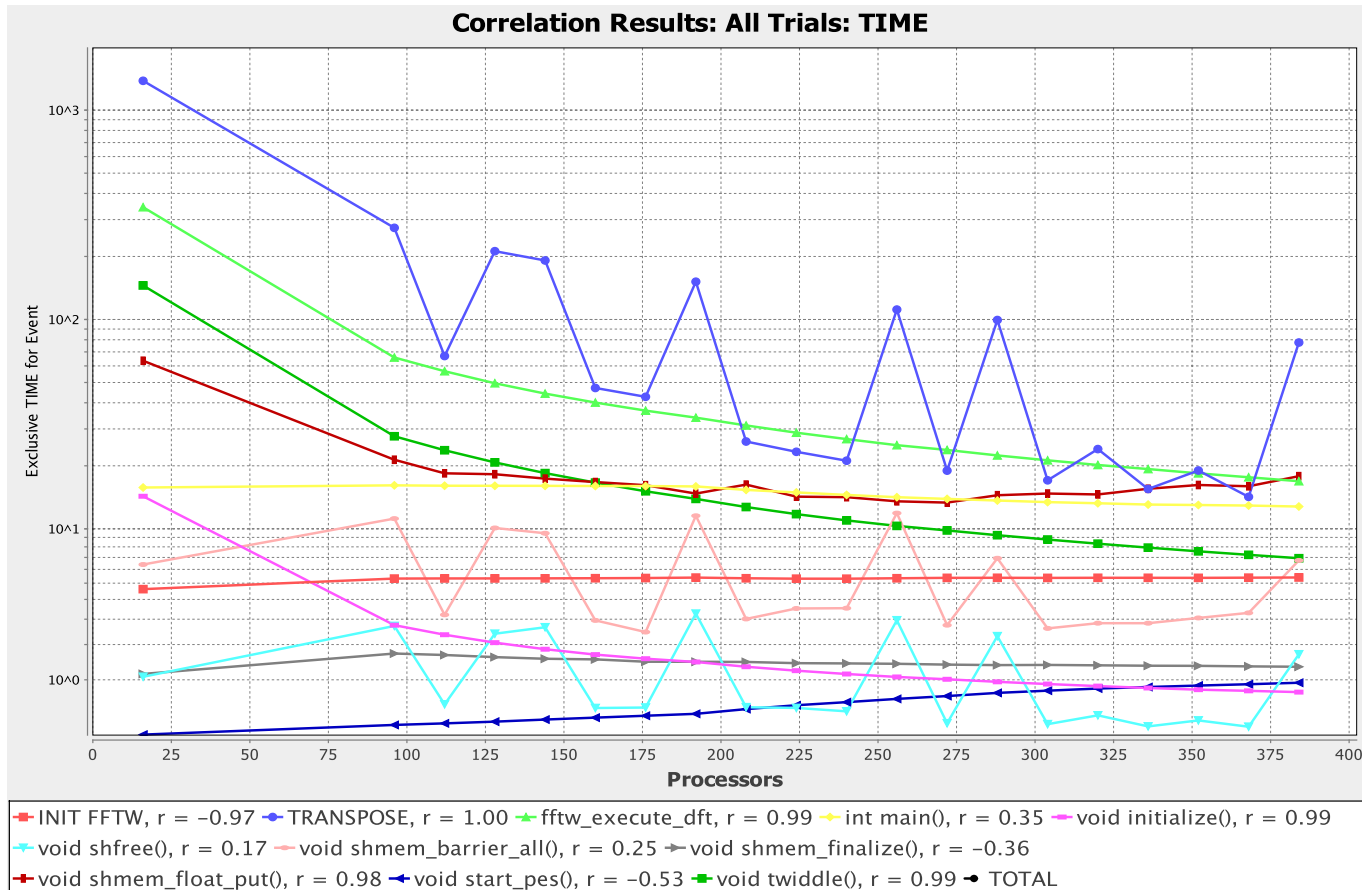
% perfexplorer (Charts → Runtime Breakdown)

How Does Each Routine Scale?



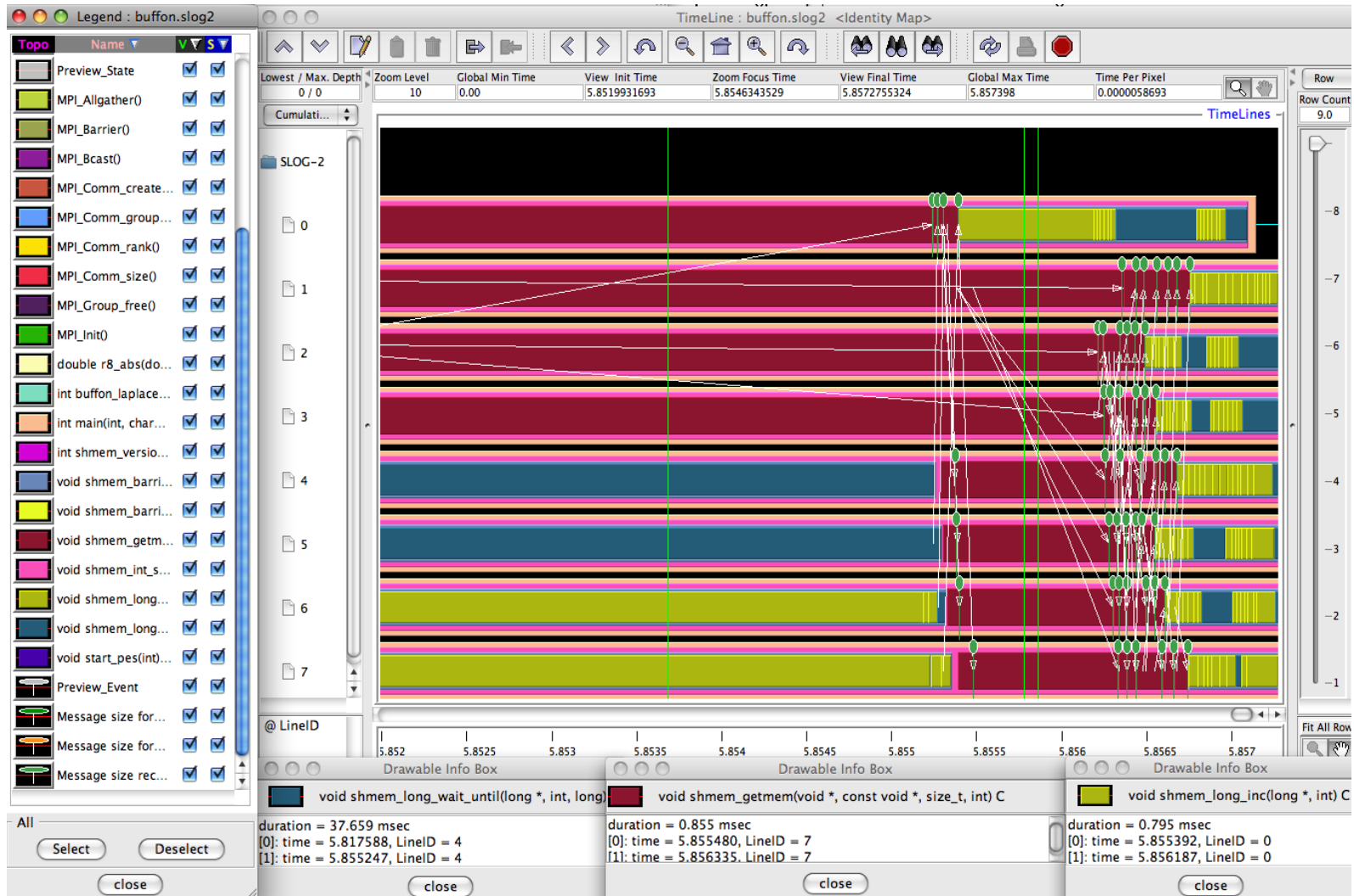
% **perfexplorer** (Charts → Stacked Bar Chart)

Which Events Correlate with Runtime?



% **perfexplorer** (Charts → Correlate Events with Total Runtime)

When do Events Occur?

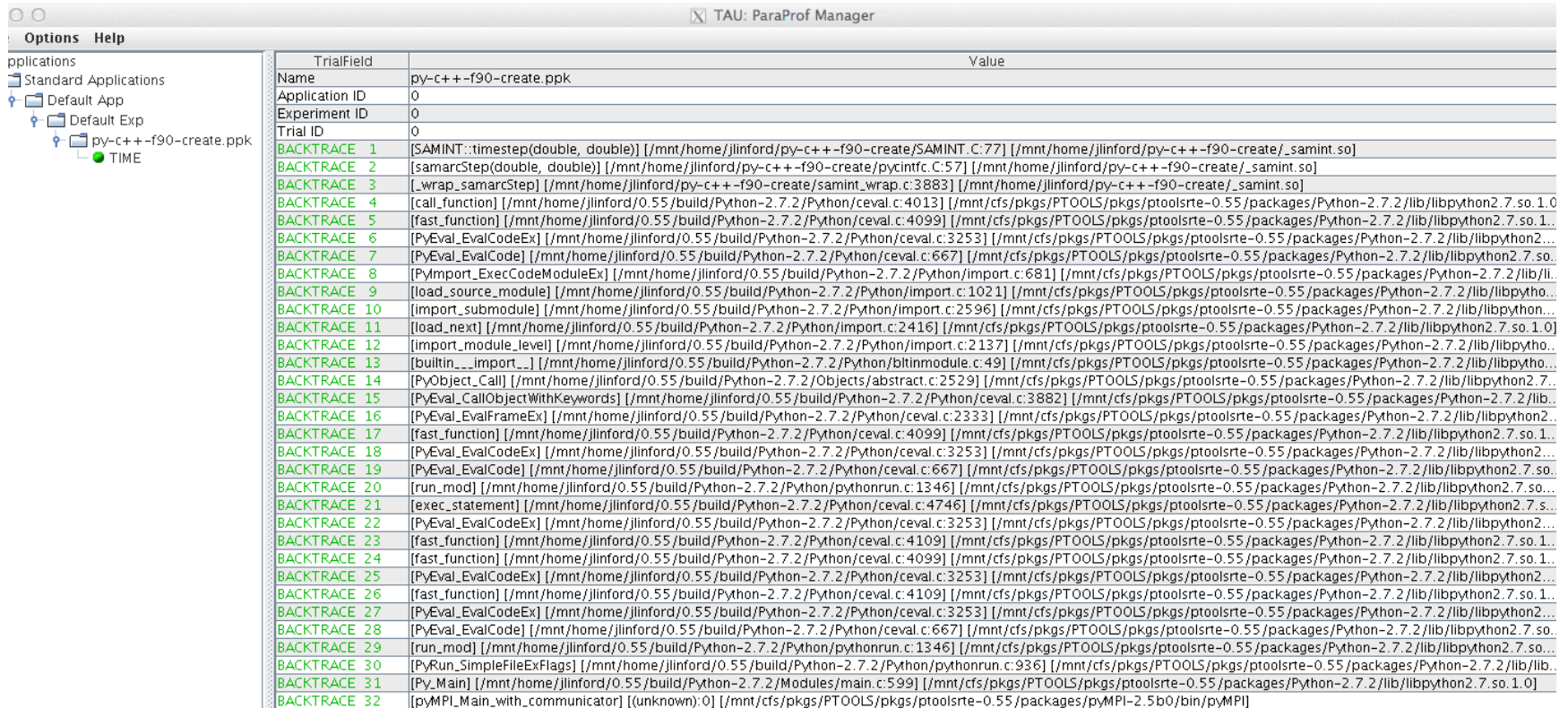


When do Events Occur?

To generate a trace and visualize it in Jumpshot:

```
% qsub -env TAU_TRACE=1 ...  
% tau_treemerge.pl  
% tau2slog2 tau.trc tau.edf -o app.slog2  
% jumpshot app.slog2
```

What Caused My Application to Crash?



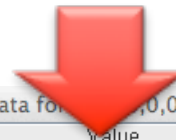
TrialField	Value
Name	py-c++-f90-create.ppk
Application ID	0
Experiment ID	0
Trial ID	0
BACKTRACE 1	[SAMINT::timestep(double, double)] [/mnt/home/jlinford/py-c++-f90-create/SAMINT.C:77] [/mnt/home/jlinford/py-c++-f90-create/_samint.so]
BACKTRACE 2	[samarcStep(double, double)] [/mnt/home/jlinford/py-c++-f90-create/pycintf.C:57] [/mnt/home/jlinford/py-c++-f90-create/_samint.so]
BACKTRACE 3	[_wrap_samarcStep] [/mnt/home/jlinford/py-c++-f90-create/samint_wrap.c:3883] [/mnt/home/jlinford/py-c++-f90-create/_samint.so]
BACKTRACE 4	[call_function] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/ceval.c:4013] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2.7.so.1.0]
BACKTRACE 5	[fast_function] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/ceval.c:4099] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2.7.so.1.0]
BACKTRACE 6	[PyEval_EvalCodeEx] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/ceval.c:3253] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2.7.so.1.0]
BACKTRACE 7	[PyEval_EvalCode] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/ceval.c:667] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2.7.so.1.0]
BACKTRACE 8	[PyImport_ExecCodeModuleEx] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/import.c:681] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2.7.so.1.0]
BACKTRACE 9	[load_source_module] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/import.c:1021] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2.7.so.1.0]
BACKTRACE 10	[import_submodule] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/import.c:2596] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2.7.so.1.0]
BACKTRACE 11	[load_next] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/import.c:2416] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2.7.so.1.0]
BACKTRACE 12	[import_module_level] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/import.c:2137] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2.7.so.1.0]
BACKTRACE 13	[builtin___import_] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/builtinmodule.c:49] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2.7.so.1.0]
BACKTRACE 14	[PyObject_Call] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Objects/abstract.c:2529] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2.7.so.1.0]
BACKTRACE 15	[PyEval_CallObjectWithKeywords] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/ceval.c:3882] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2.7.so.1.0]
BACKTRACE 16	[PyEval_EvalFrameEx] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/ceval.c:2333] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2.7.so.1.0]
BACKTRACE 17	[fast_function] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/ceval.c:4099] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2.7.so.1.0]
BACKTRACE 18	[PyEval_EvalCodeEx] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/ceval.c:3253] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2.7.so.1.0]
BACKTRACE 19	[PyEval_EvalCode] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/ceval.c:667] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2.7.so.1.0]
BACKTRACE 20	[run_mod] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/pythonrun.c:1346] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2.7.so.1.0]
BACKTRACE 21	[exec_statement] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/ceval.c:4746] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2.7.so.1.0]
BACKTRACE 22	[PyEval_EvalCodeEx] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/ceval.c:3253] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2.7.so.1.0]
BACKTRACE 23	[fast_function] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/ceval.c:4109] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2.7.so.1.0]
BACKTRACE 24	[fast_function] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/ceval.c:4099] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2.7.so.1.0]
BACKTRACE 25	[PyEval_EvalCodeEx] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/ceval.c:3253] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2.7.so.1.0]
BACKTRACE 26	[fast_function] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/ceval.c:4109] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2.7.so.1.0]
BACKTRACE 27	[PyEval_EvalCodeEx] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/ceval.c:3253] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2.7.so.1.0]
BACKTRACE 28	[PyEval_EvalCode] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/ceval.c:667] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2.7.so.1.0]
BACKTRACE 29	[run_mod] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/pythonrun.c:1346] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2.7.so.1.0]
BACKTRACE 30	[PyRun_SimpleFileExFlags] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/pythonrun.c:936] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2.7.so.1.0]
BACKTRACE 31	[Py_Main] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Modules/main.c:599] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2.7.so.1.0]
BACKTRACE 32	[pyMPI_Main_with_communicator] [(unknown):0] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/pyMPI-2.5b0/bin/pyMPI]

```
% qsub -env TAU_TRACK_SIGNALS=1 ...
```

```
% paraprof
```

What Caused My Application to Crash?

Right-click to see source code



Name	Value
BACKTRACE 1	[SAMINT::timestep(double, double)] [/mnt/home/jlinford/py-c++-f90-create/SAMI... jlinford/py-c++-f90-create/_samint.so]
BACKTRACE 2	[samarcStep(double, double)] [/mnt/home/jlinford/py-c++-f90-create/pycintfc.C... Show Source Code /py-c++-f90-create/_samint.so]
BACKTRACE 3	[_wrap_samarcStep] [/mnt/home/jlinford/py-c++-f90-create/samint_wrap.c:3883] [/mnt/home/jlinford/py-c++-f90-create/_samint.so]
BACKTRACE 4	[call_function] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/ceval.c:4013] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2.7.so.1.0]
BACKTRACE 5	[fast_function] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/ceval.c:4099] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2.7.so.1.0]
BACKTRACE 6	[PyEval_EvalCodeEx] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/ceval.c:3253] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2....]
BACKTRACE 7	[PyEval_EvalCode] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/ceval.c:667] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2.7.so....]
BACKTRACE 8	[PyImport_ExecCodeModuleEx] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/import.c:681] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/li...]
BACKTRACE 9	[load_source_module] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/import.c:1021] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython...]
BACKTRACE 10	[import_submodule] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/import.c:2596] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2...]
BACKTRACE 11	[load_next] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/import.c:2416] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2.7.so.1.0]
BACKTRACE 12	[import_module_level] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/import.c:2137] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython...]
BACKTRACE 13	[builtin__import_] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/builtinmodule.c:49] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython...]
BACKTRACE 14	[PyObject_Call] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Objects/abstract.c:2529] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2.7....]
BACKTRACE 15	[PyEval_CallObjectWithKeywords] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/ceval.c:3882] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib...]
BACKTRACE 16	[PyEval_EvalFrameEx] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/ceval.c:2333] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2....]
BACKTRACE 17	[fast_function] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/ceval.c:4099] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2.7.so.1.0]
BACKTRACE 18	[PyRun_EvalCodeEx] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/ceval.c:3253] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2....]
BACKTRACE 19	[PyEval_EvalCode] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/ceval.c:667] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2.7.so....]
BACKTRACE 20	[run_mod] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/pythonrun.c:1346] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2.7.so....]
BACKTRACE 21	[exec_statement] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/ceval.c:4746] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2.7.so....]
BACKTRACE 22	[PyEval_EvalCodeEx] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/ceval.c:3253] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2....]
BACKTRACE 23	[fast_function] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/ceval.c:4109] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2.7.so.1.0]
BACKTRACE 24	[fast_function] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/ceval.c:4099] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2.7.so.1.0]
BACKTRACE 25	[PyEval_EvalCodeEx] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/ceval.c:3253] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2....]
BACKTRACE 26	[fast_function] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/ceval.c:4109] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2.7.so.1.0]
BACKTRACE 27	[PyEval_EvalCodeEx] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/ceval.c:3253] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2....]
BACKTRACE 28	[PyEval_EvalCode] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/ceval.c:667] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2.7.so....]
BACKTRACE 29	[run_mod] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/pythonrun.c:1346] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2.7.so....]
BACKTRACE 30	[PyRun_SimpleFileExFlags] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/pythonrun.c:936] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/li...]
BACKTRACE 31	[Py_Main] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Modules/main.c:599] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpython2.7.so.1.0]
BACKTRACE 32	[pyMPI_Main_with_communicator] [(unknown):0] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/pyMPI-2.5b0/bin/pyMPI]
BACKTRACE 33	[main] [(unknown):0] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/pyMPI-2.5b0/bin/pyMPI]
BACKTRACE 34	[_libc_start_main] [(unknown):0] [/lib64/libc-2.5.so]
BACKTRACE 35	[start] [(unknown):0] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/pyMPI-2.5b0/bin/pyMPI]

What Caused My Application to Crash?

```
TAU: ParaProf: Source Browser: /mnt/home/jlinford/py-c++-f90-create/SAMINT.C
File Help
65  /*
66  *****
67  *
68  * Take a timestep - advance solution from "time" to "time + dt"
69  *
70  *****
71  */
72  void SAMINT::timestep(const double time,
73                      const double dt)
74  {
75      cout << "SAMINT::timestep()" << endl;
76      timestep_(time,dt);
77      int x = 4 / (4-4);
78      cout << " x = " << x << endl;
79  }
80
81  /*
82  *****
83  *
84  * Write data to output
85  * (visit, fieldview, or overgrid - set in samarc input file)
86  *
87  *****
88  */
89  void SAMINT::writePlotData(const double time,
90                            const int step)
91  {
92      cout << "SAMINT::writePlotData()" << endl;
93  }
```



Intuitive Performance Engineering

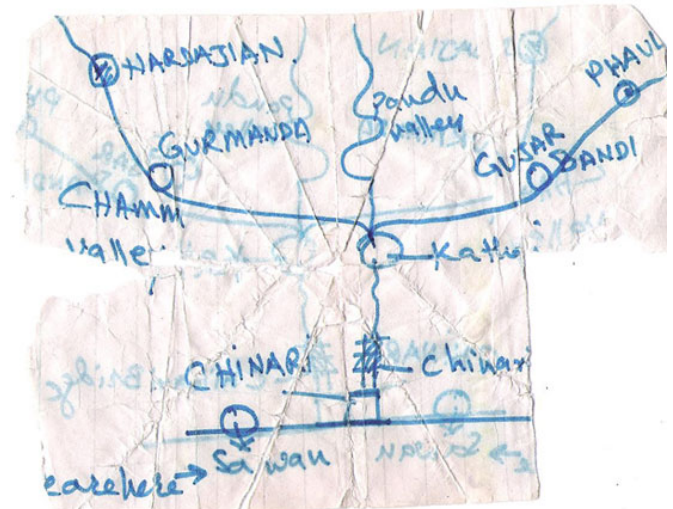
TAU COMMANDER (ALPHA RELEASE!)

The TAU Commander Approach

- Say where you're going, not how to get there
- **TAU Projects** give context to the user's actions
 - Defines desired metrics and measurement approach
 - Defines operating environment
 - Establishes a baseline for error checking



VS.



TAU Commander CLI Dashboard

```
mm -- ssh -- 154x53
jlinford@east03 ~/workspace/taucmdr/examples/mm $ tau dash
== Targets (/home/jlinford/.tau) =====
-----
| Name | Host OS | Host Arch | C | C++ | Fortran | In Projects |
-----
| localhost | Linux | x86_64 | /usr/bin/gcc | /usr/bin/g++ | /usr/bin/gfortran | ex-mm |
-----

== Applications (/home/jlinford/.tau) =====
-----
| Name | OpenMP | Pthreads | MPI | CUDA | MIC | SHMEM | MPC | In Projects |
-----
| ex-mm-serial | | | | | | | | ex-mm |
-----
| ex-mm-openmp | Yes | | | | | | | ex-mm |
-----
| ex-mm-openmp-mpi | Yes | | Yes | | | | | ex-mm |
-----

== Measurements (/home/jlinford/.tau) =====
-----
| Name | Profile | Trace | Sample | Source Inst. | Compiler Inst. | MPI | OpenMP | Callpath Depth | Mem. Usage | Mem. Alloc | In Projects |
-----
| ex-profile | Yes | No | No | automatic | fallback | No | compiler_default | 0 | No | No | ex-mm |
-----
| ex-trace | No | Yes | No | automatic | fallback | No | compiler_default | 0 | No | No | ex-mm |
-----
| ex-sample | Yes | No | Yes | never | never | No | compiler_default | 0 | No | No | ex-mm |
-----

== Projects (/home/jlinford/.tau) =====
-----
| Name | Targets | Applications | Measurements | Home |
-----
| ex-mm | localhost | ex-mm-serial | ex-profile | /home/jlinford/.tau |
| | | ex-mm-openmp | ex-trace | |
| | | ex-mm-openmp-mpi | ex-sample | |
-----

== ex-mm (localhost, ex-mm-openmp, ex-profile) Trials =====
No trials. Use 'tau <command>' or 'tau trial create <command>' to create a new trial

jlinford@east03 ~/workspace/taucmdr/examples/mm $
```

TAU Commander CLI

This command's usage

Subcommand usage

Shortcuts

```
mm -- ssh -- 89x45
jlinford@east03 ~/workspace/taucmdr/examples/mm $ tau --help
usage:
  tau [arguments] <subcommand> [options]

TAU Commander [ http://www.taucommander.com/ ]

positional arguments:
  <subcommand>  See subcommand descriptions below
  [options]     Options to be passed to <subcommand>

optional arguments:
  -h, --help    show this help message and exit
  -v, --verbose Set logging level to DEBUG
                - default: INFO

configuration subcommands:
  application  Create and manage application configurations.
  measurement  Create and manage measurement configurations.
  project      Create and manage project configurations.
  target       Create and manage target configurations.

subcommands:
  build        Instrument programs during compilation and/or linking.
  dashboard    Show all projects and their components.
  help         Show help for a command or suggest actions for a file.
  make         Instrument programs during compilation and/or linking with `make`.
  trial        Create and manage experiment trials.

shortcuts:
  tau <compiler>  Execute a compiler command
                  - Example: tau gcc *.c -o a.out
                  - Alias for 'tau build <compiler>'
  tau <program>   Gather data from a program
                  - Example: tau ./a.out
                  - Alias for 'tau trial create <program>'
  tau run <program> Gather data from a program
                  - Example: tau ./a.out
                  - Alias for 'tau trial create <program>'
  tau show       Show data from the most recent trial
                  - An alias for 'tau trial show'

See 'tau help <subcommand>' for more information on <subcommand>.
jlinford@east03 ~/workspace/taucmdr/examples/mm $
```



First use on a “vanilla” system

```
jlinford@east03 ~/workspace/taucmdr/examples/mm $ ls
configure.sh Makefile matmult.c matmult_initialize.c matmult_initialize.o
jlinford@east03 ~/workspace/taucmdr/examples/mm $ tau gcc *.c -o mm
Installing PDT at '/home/jlinford/.tau/PDT/GNU' from 'http://tau.uoregon.edu/tau/pdt_lite.tgz'
Downloading 'http://tau.uoregon.edu/tau/pdt_lite.tgz'
Extracting '/home/jlinford/.tau/src/pdt_lite.tgz'
Cleaning PDT installation prefix '/home/jlinford/.tau/PDT/GNU'
Configuring PDT for GNU compilers...
Compiling PDT...
Installing PDT...
PDT installation complete, verifying installation
Installing BFD at '/home/jlinford/.tau/BFD/x86_64/GNU' from 'http://www.cs.uoregon.edu/research/paracomp/tau/tauprofile/dist/binutils-2.23.2.tar.gz' with
arch=x86_64 and GNU compilers
Downloading 'http://www.cs.uoregon.edu/research/paracomp/tau/tauprofile/dist/binutils-2.23.2.tar.gz'
Extracting '/home/jlinford/.tau/src/binutils-2.23.2.tar.gz'
Cleaning BFD installation prefix '/home/jlinford/.tau/BFD/x86_64/GNU'
Configuring BFD...
Compiling BFD...
Installing BFD...
BFD installation complete, verifying installation
Installing libunwind at '/home/jlinford/.tau/libunwind/x86_64/GNU' from
'http://www.cs.uoregon.edu/research/paracomp/tau/tauprofile/dist/libunwind-1.1.tar.gz'
Downloading 'http://www.cs.uoregon.edu/research/paracomp/tau/tauprofile/dist/libunwind-1.1.tar.gz'
Extracting '/home/jlinford/.tau/src/libunwind-1.1.tar.gz'
Cleaning libunwind installation prefix '/home/jlinford/.tau/libunwind/x86_64/GNU'
Configuring libunwind...
Compiling libunwind...
Installing libunwind...
libunwind installation complete, verifying installation
Installing TAU at '/home/jlinford/.tau/TAU/' from 'http://tau.uoregon.edu/tau.tgz' with arch=x86_64 and GNU compilers
Downloading 'http://tau.uoregon.edu/tau.tgz'
Extracting '/home/jlinford/.tau/src/tau.tgz'
Configuring TAU with -iowrapper...
Compiling and installing TAU...
TAU installation complete
tau_cc.sh matmult.c matmult_initialize.c -o mm
jlinford@east03 ~/workspace/taucmdr/examples/mm $
```

Put tau in front of every command

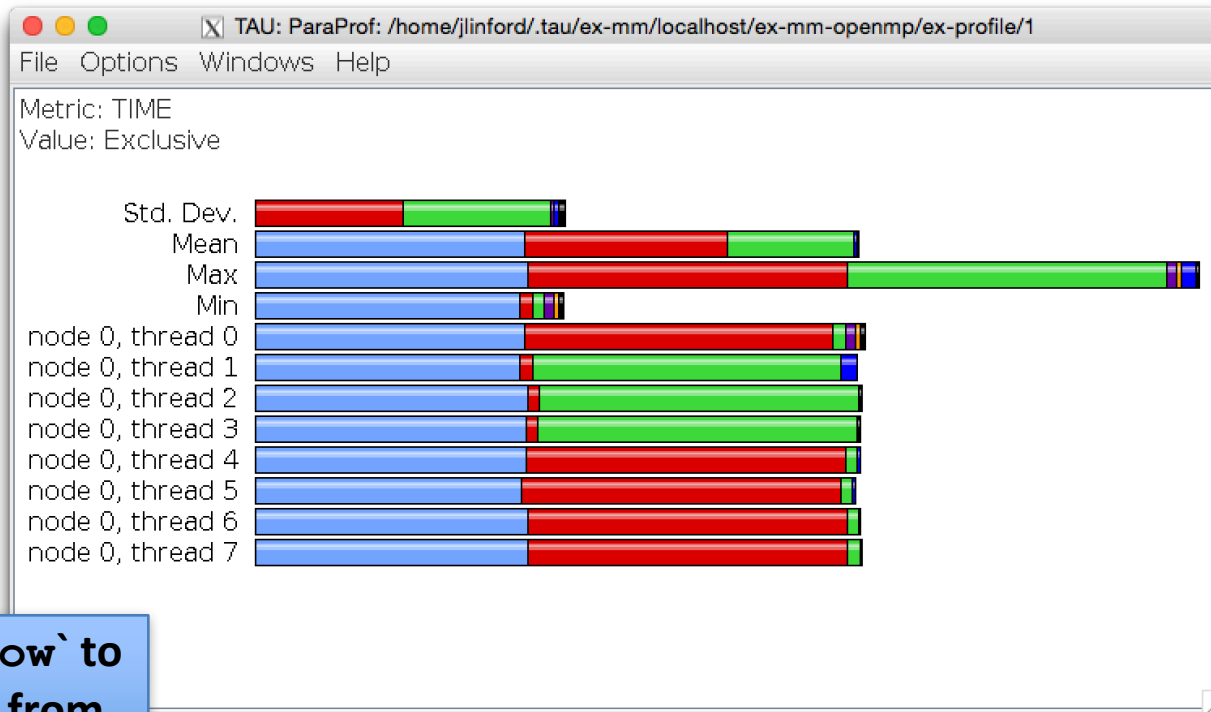
Detects, downloads, and installs required dependencies

Configures environment, wraps compiler

Executions create experiment trials

```
mm -- ssh -- 154x5
jlinford@east03 ~/workspace/taucmdr/examples/mm $ export OMP_NUM_THREADS=8
jlinford@east03 ~/workspace/taucmdr/examples/mm $ tau ./mm
== BEGIN ex-mm (localhost, ex-mm-openmp, ex-profile) (2015-08-12 04:18:35.372290) ==
./mm
Done.
Found 8 profile files. Adding to trial...
== END ex-mm (localhost, ex-mm-openmp, ex-profile) (2015-08-12 04:18:36.593917) =====
jlinford@east03 ~/workspace/taucmdr/examples/mm $
```

Put tau in front of every command



``tau show`` to see data from last trial

Executions create experiment trials

```
mm - ssh - 154x53
jlinford@east03 ~/workspace/taucmdr/examples/mm $ tau dash
== Targets (/home/jlinford/.tau) ==
-----
| Name | Host OS | Host Arch | C | C++ | Fortran | In Projects |
-----
| localhost | Linux | x86_64 | /usr/bin/gcc | /usr/bin/g++ | /usr/bin/gfortran | ex-mm |
-----

== Applications (/home/jlinford/.tau) ==
-----
| Name | OpenMP | Pthreads | MPI | CUDA | MIC | SHMEM | MPC | In Projects |
-----
| ex-mm-serial | | | | | | | | ex-mm |
-----
| ex-mm-openmp | Yes | | | | | | | ex-mm |
-----
| ex-mm-openmp-mpi | Yes | | Yes | | | | | ex-mm |
-----

== Measurements (/home/jlinford/.tau) ==
-----
| Name | Profile | Trace | Sample | Source Inst. | Compiler Inst. | MPI | OpenMP | Callpath Depth | Mem. Usage | Mem. Alloc | In Projects |
-----
| ex-profile | Yes | No | No | automatic | fallback | No | compiler_default | 0 | No | No | ex-mm |
-----
| ex-trace | No | Yes | No | automatic | fallback | No | compiler_default | 0 | No | No | ex-mm |
-----
| ex-sample | Yes | No | Yes | never | never | No | compiler_default | 0 | No | No | ex-mm |
-----

== Projects (/home/jlinford/.tau) ==
-----
| Name | Targets | Applications | Measurements | Home |
-----
| ex-mm | localhost | ex-mm-serial | ex-profile | /home/jlinford/.tau |
| | | ex-mm-openmp | ex-trace | |
| | | ex-mm-openmp-mpi | ex-sample | |
-----

== ex-mm (localhost, ex-mm-openmp, ex-profile) Trials ==
-----
2 trials of 'mm' (22.6KiB). Use `tau trial list` to see details.
jlinford@east03 ~/workspace/taucmdr/examples/mm $
```



Each execution is a new trial

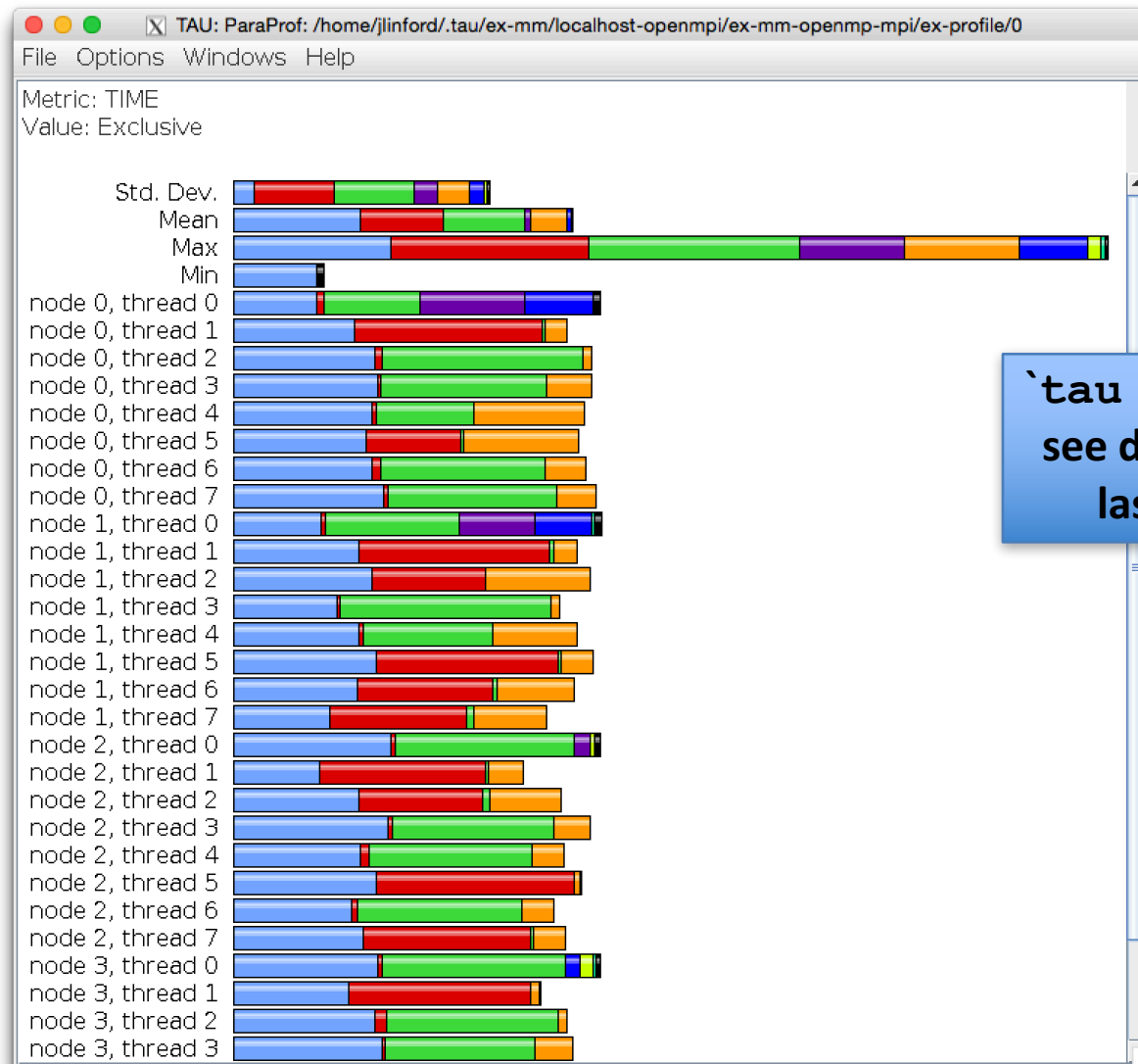
Changing from serial to MPI+OpenMP

```
== Projects (/home/jlinford/.tau) =====  
-----  
+-----+-----+-----+-----+-----+  
| Name | Targets | Applications | Measurements | Home |  
+-----+-----+-----+-----+-----+  
| ex-mm | localhost | ex-mm-serial | ex-profile | /home/jlinford/.tau |  
| | localhost-openmpi | ex-mm-openmp | ex-trace | |  
| | | ex-mm-openmp-mpi | ex-sample | |  
+-----+-----+-----+-----+-----+  
-----  
== ex-mm (localhost-openmpi, ex-mm-openmp-mpi, ex-ex-profile) Trials =====  
-----  
No trials. Use 'tau <command>' or 'tau trial create <command>' to create a new tri  
  
jlinford@east03 ~/workspace/taucmdr/examples/mm $ tau mpicc *.c -fopenmp -o mm  
Installing TAU at '/home/jlinford/.tau/TAU/' from 'http://tau.uoregon.edu/tau.tgz' with 386_64 and MPI compilers  
Using TAU source archive at '/home/jlinford/.tau/src/tau.tgz'  
Reusing TAU source files found at '/home/jlinford/.tau/src/./tau-2.24.1'  
Configuring TAU with -iowrapper...  
Compiling and installing TAU...  
TAU installation complete  
tau_cc.sh matmult.c matmult_initialize.c -fopenmp -o mm  
  
jlinford@east03 ~/workspace/taucmdr/examples/mm $
```

Put tau in front of every command

Automatically reconfigures TAU for MPI+OpenMP

Workflow is unchanged



``tau show`` to see data from last trial

Intuitive Performance Engineering

CASE STUDIES

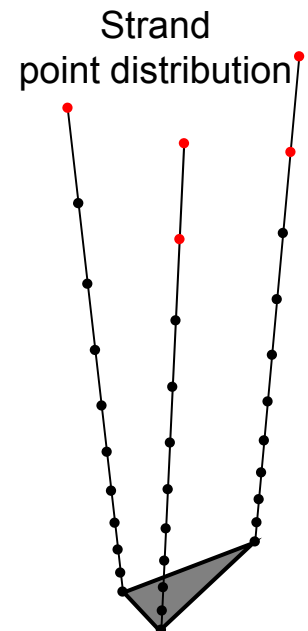
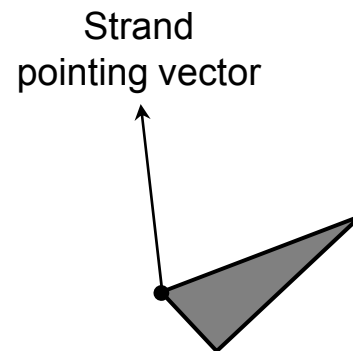
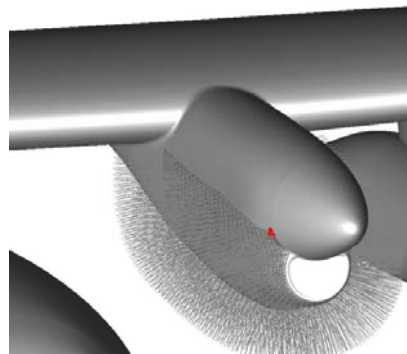
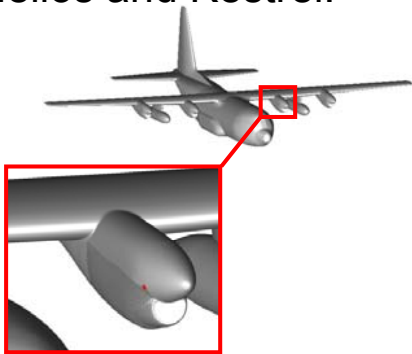
Strand Technology

Technology Drivers

- Timeliness (automation of mesh generation)
- Timeliness (automation and scalability of domain connectivity)
- Timeliness/Physical accuracy (computational efficiency and scalability of aerodynamic solvers)
- Processor architecture (small memory footprint maps well to hierarchical memory architectures, e.g., multi-core, GPU)

CREATE-AV Example

This is a new meshing paradigm introduced in 2007 by current members of the CREATE-AV technical staff. The technology is being matured in the Helios product and will be deployed through both Helios and Kestrel.

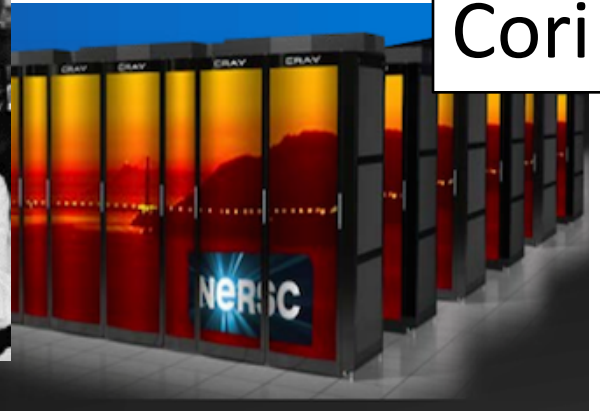


Target Platforms

Armstrong [XC30]



Cori



Haise [iDataPlex]



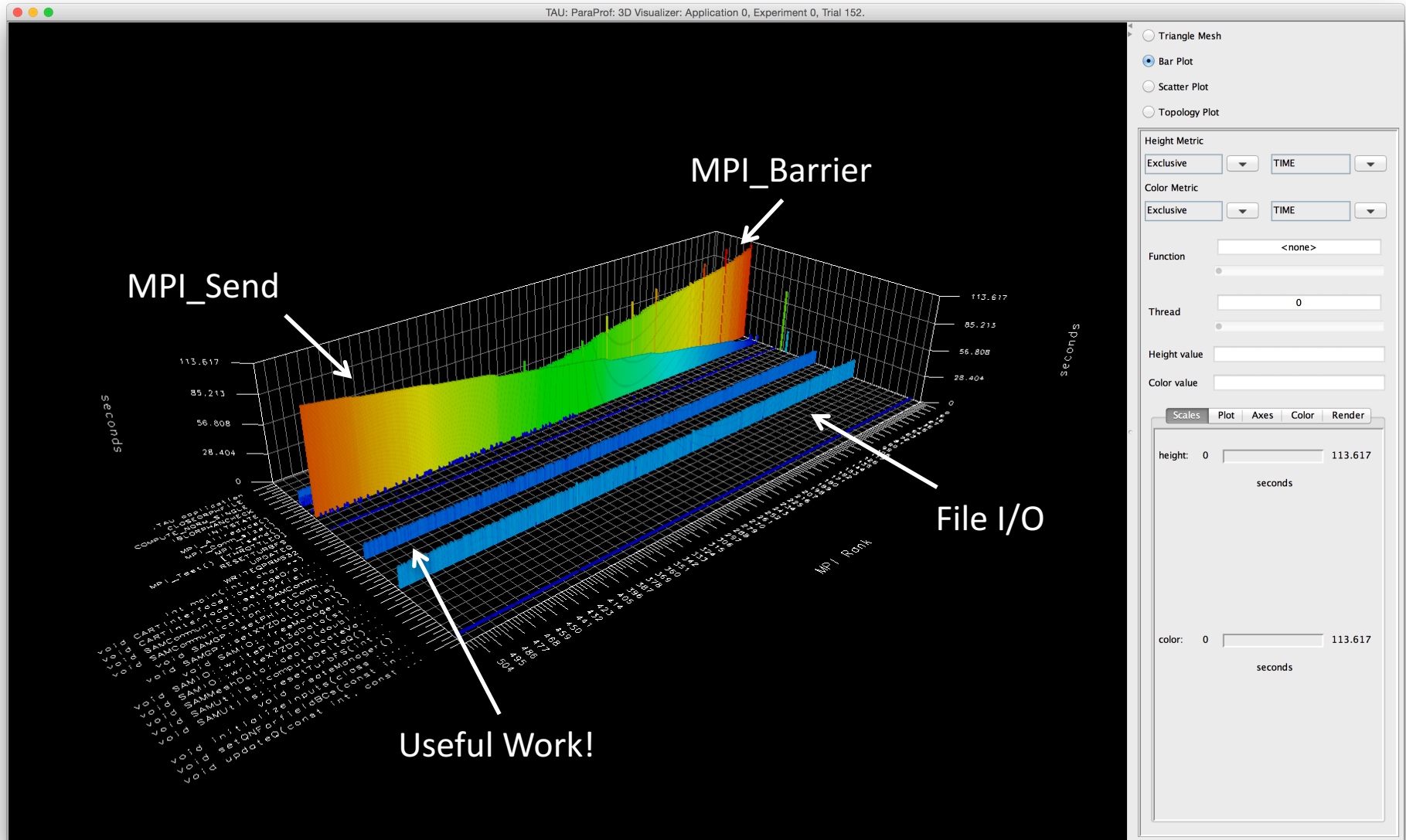
Lightning [XC30]



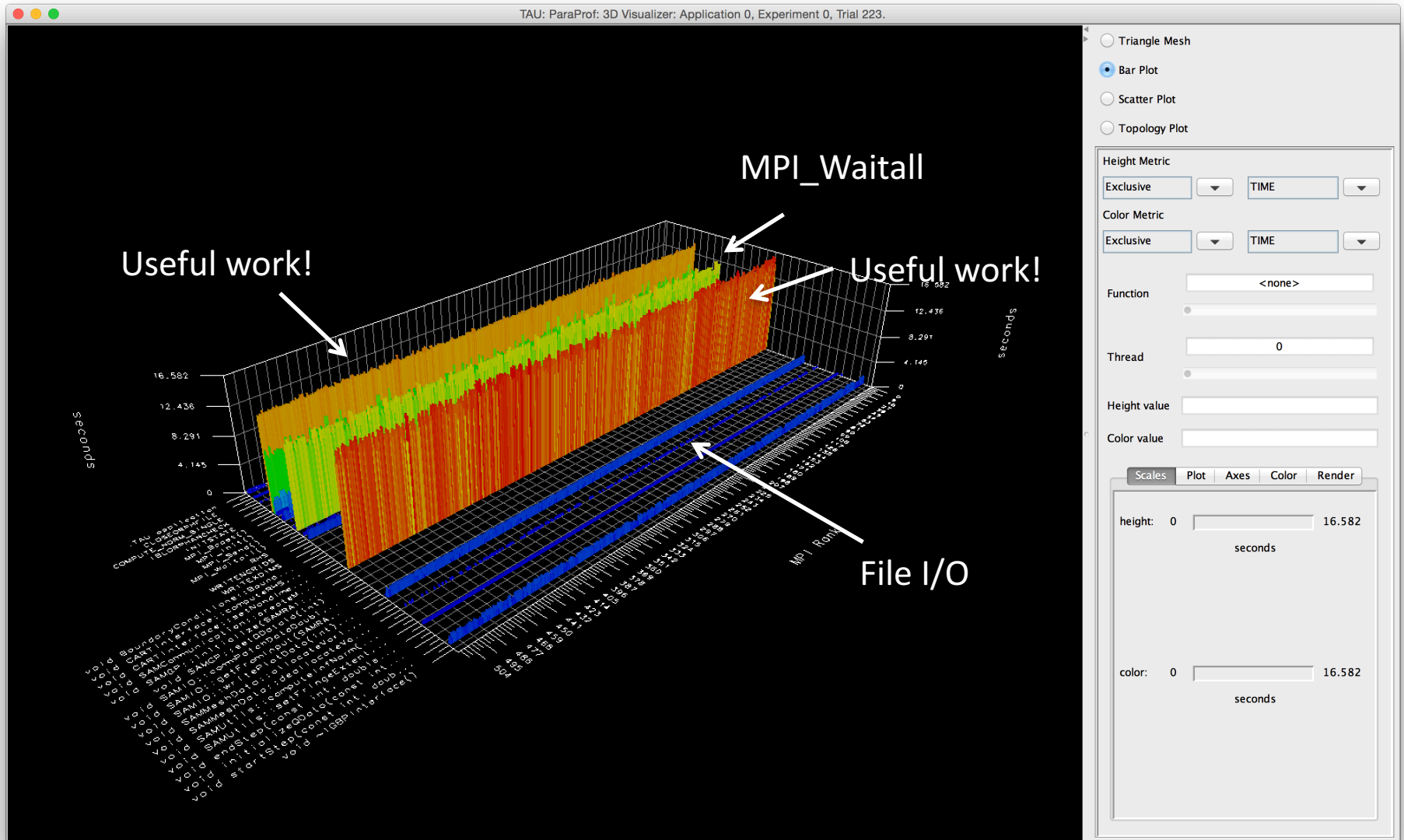
Kilrain [iDataPlex]



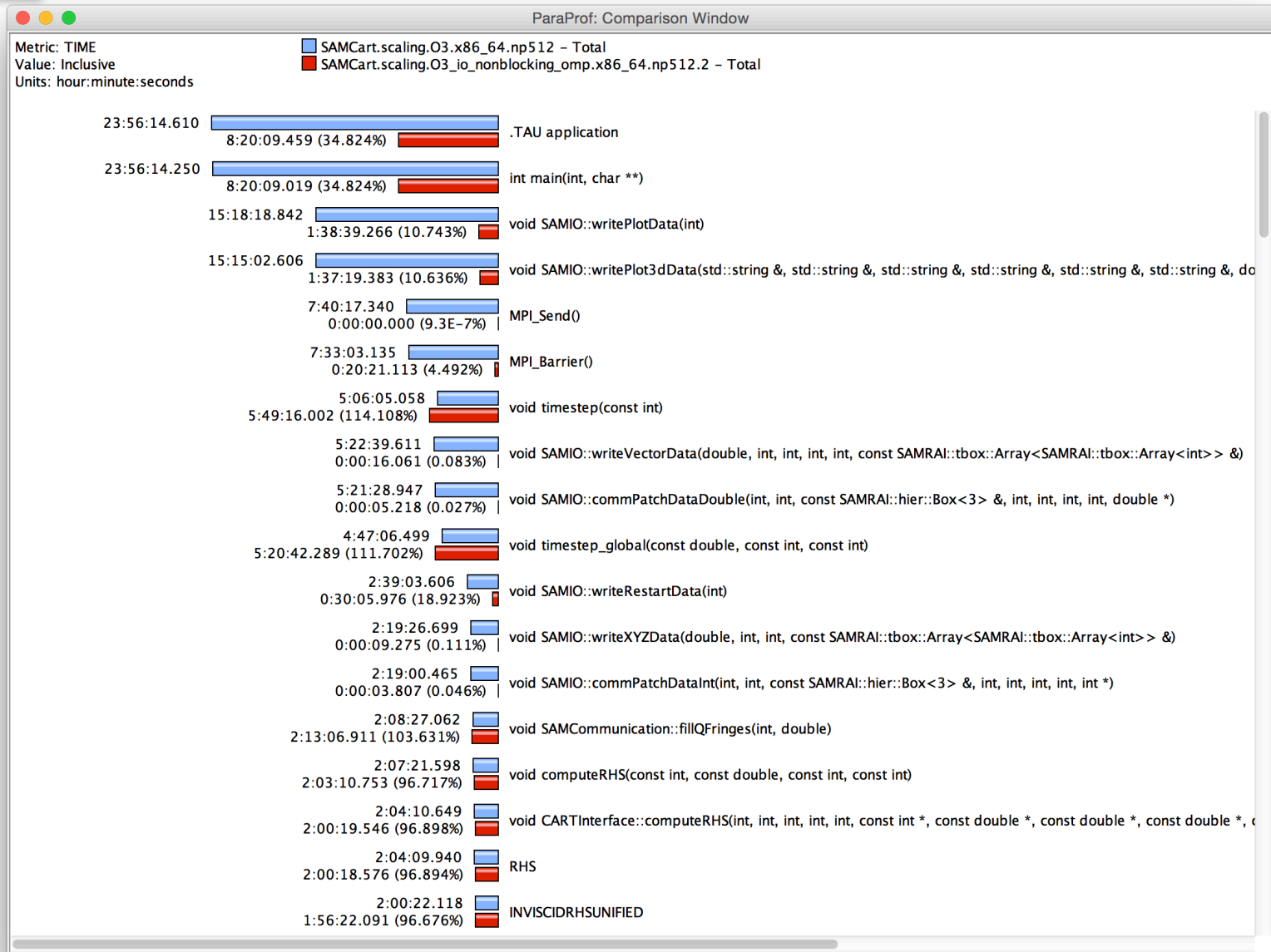
Initial Profile on Babbage



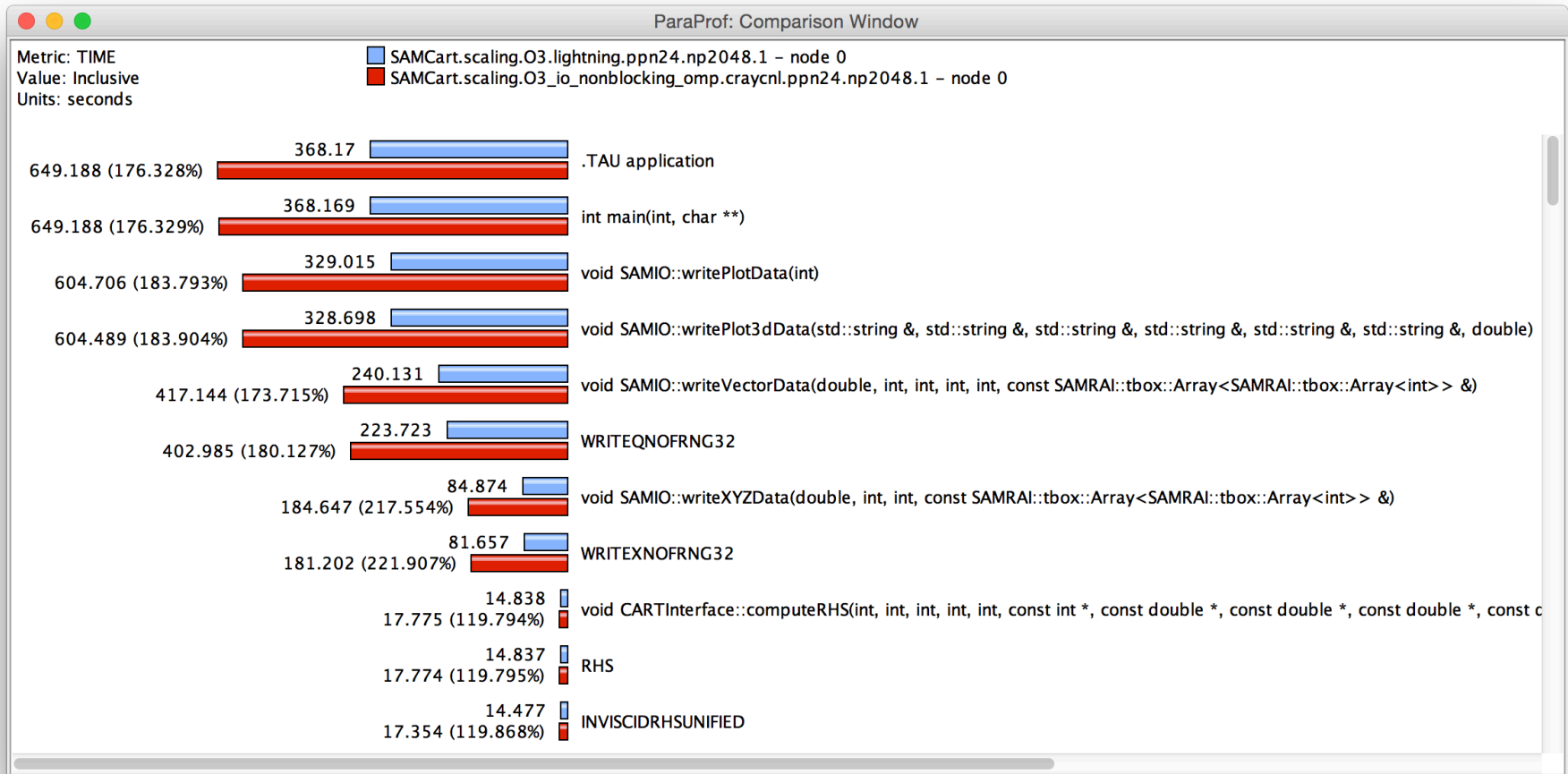
Hot Spot Optimization



65% Runtime Reduction (~2x faster)

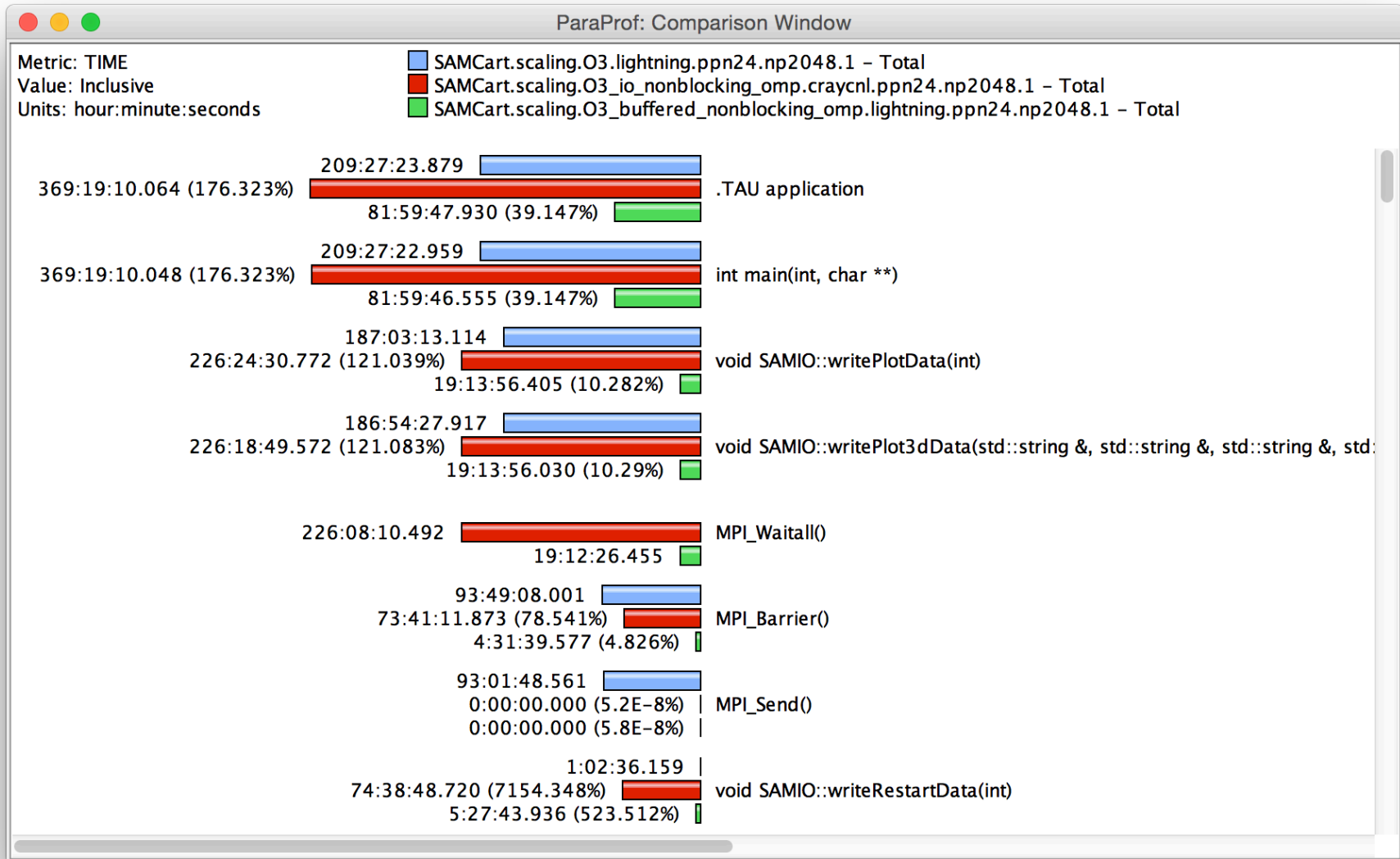


Cray XC30



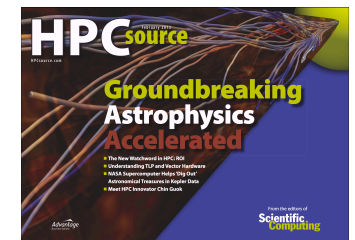
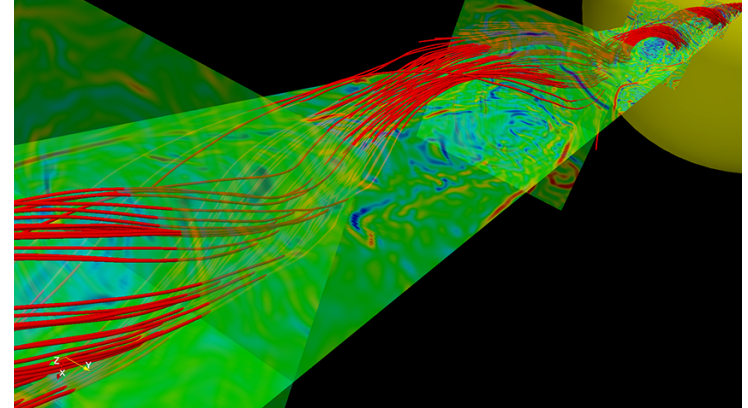
Slower! What happened???

No worries, I fix it



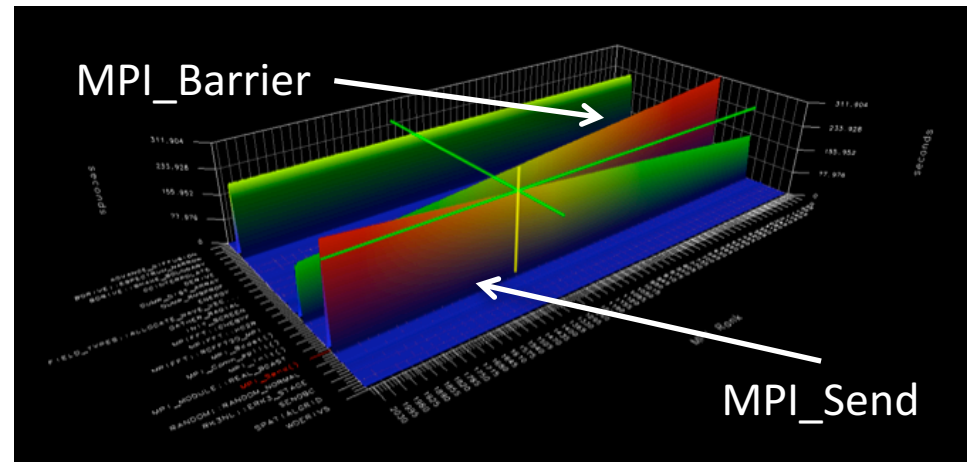
IRMHD on Intrepid and Mira

- INCITE magnetohydrodynamic simulation to understand solar winds and coronal heating
 - First direct numerical simulations of Alfvén wave (AW) turbulence in extended solar atmosphere accounting for inhomogeneities
 - Team
 - University of New Hampshire (Jean Perez and Benjamin Chandran)
 - ALCF (Tim Williams)
 - University of Oregon (Sameer Shende)
- IRMHD (Inhomogeneous Reduced Magnetohydrodynamics)
 - Fortran 90 and MPI
 - Excellent weak and strong scaling properties
 - Tested and benchmarked on Intrepid and Mira
- HPC Source article and ALCF news
<https://www.alcf.anl.gov/articles/furthering-understanding-coronal-heating-and-solar-wind-origin>



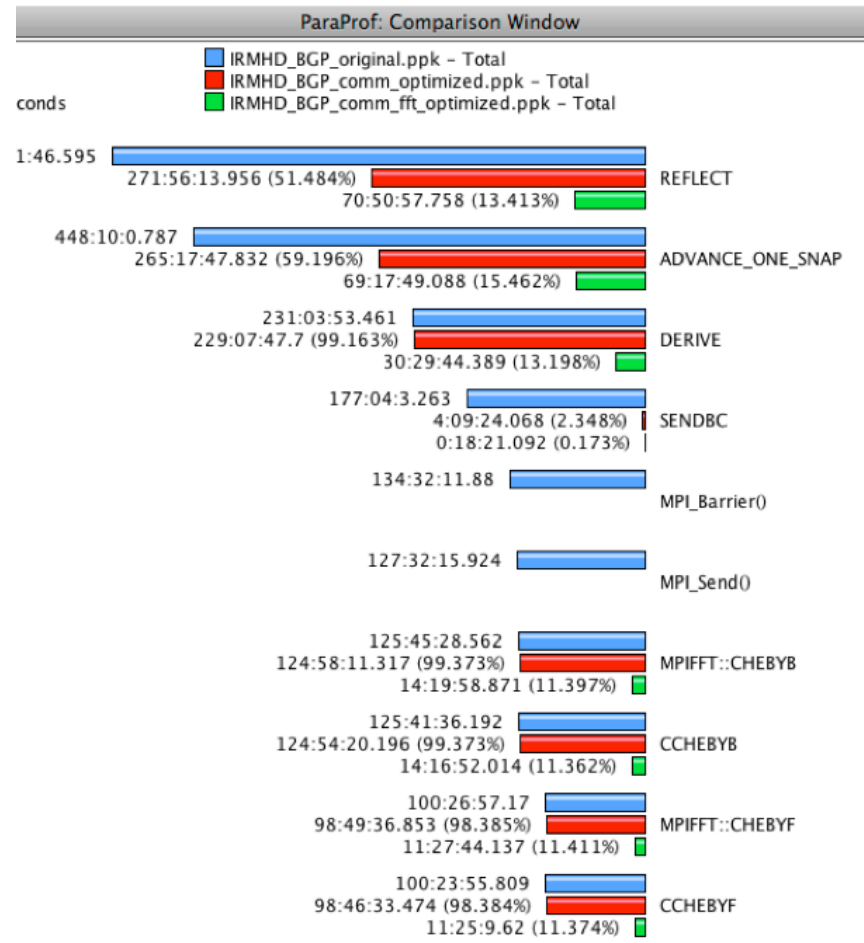
IRMHD Communication Analysis

- Source-based (direct) instrumentation
- MPI instrumentation and volume measurement
- IRMHD exhibited significant synchronous communication bottlenecks
- On 2,408 cores of BG/P:
 - **MPI_Send** and **MPI_Bcast** take significant time
 - Opportunities for communication/computation overlap
 - Identified possible targets for computation improvements



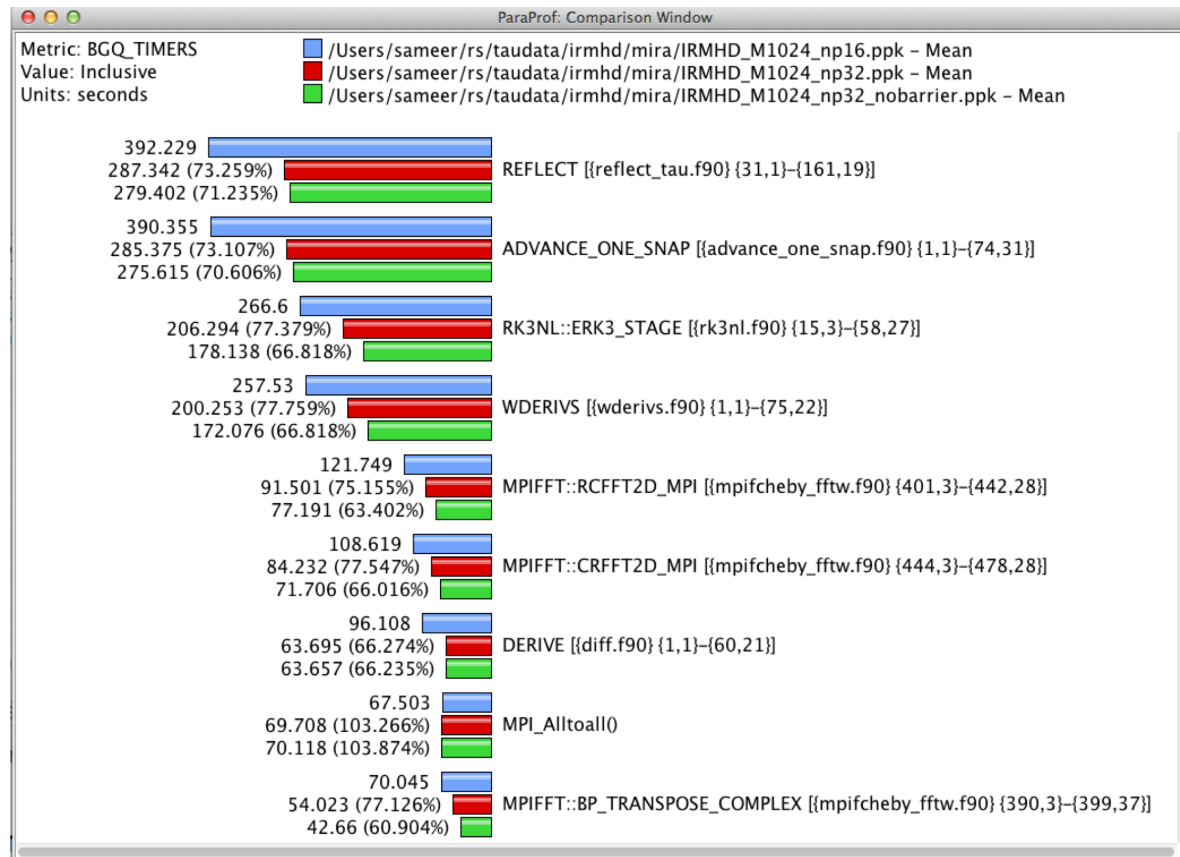
IRMHD Optimization on Intrepid (BG/P)

- On 2,408 cores, overall execution time reduced from 528.18 core hours to 70.8 core hours (**>7x improvement**)
- Non-blocking communication substrate
- More efficient implementation of underlying FFT



IRMHD Optimization on MIRA (BG/Q)

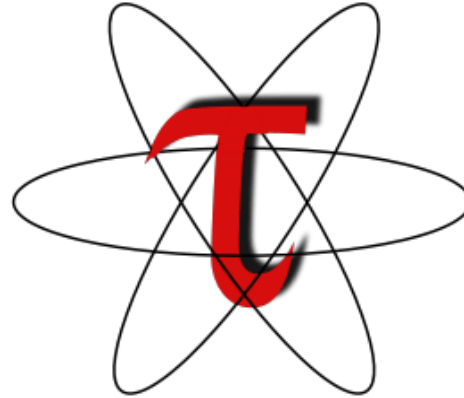
- Oversubscribe nodes: 32k ranks vs. 16k per node
- Overall time improvement: 71.23% of original



Intuitive Performance Engineering

CONCLUSION

Downloads



<http://tau.uoregon.edu>

<http://github.com/ParaToolsInc/taucmdr>

<http://www.hpclinux.com>

Free download, open source, BSD license

Acknowledgements

- Department of Energy
 - Office of Science
 - Argonne National Laboratory
 - Oak Ridge National Laboratory
 - NNSA/ASC Trilabs (SNL, LLNL, LANL)
- HPCMP DoD PETTT Program
- National Science Foundation
 - Glassbox, SI-2
- University of Tennessee
- University of New Hampshire
 - Jean Perez, Benjamin Chandran
- University of Oregon
 - Allen D. Malony, Sameer Shende
 - Kevin Huck, Wyatt Spear
- TU Dresden
 - Holger Brunst, Andreas Knupfer
 - Wolfgang Nagel
- Research Centre Jülich
 - Bernd Mohr
 - Felix Wolf



UNIVERSITY
OF OREGON



Intuitive Performance Engineering

REFERENCE

Online References

- PAPI:
 - PAPI documentation is available from the PAPI website:
<http://icl.cs.utk.edu/papi/>
- TAU:
 - TAU Users Guide and papers available from the TAU website:
<http://tau.uoregon.edu/>
- VAMPIR:
 - VAMPIR website:
<http://www.vampir.eu/>
- Scalasca:
 - Scalasca documentation page:
<http://www.scalasca.org/>
- Eclipse PTP:
 - Documentation available from the Eclipse PTP website:
<http://www.eclipse.org/ptp/>

Compiling Fortran Codes with TAU

- **If your Fortran code uses free format in .f files (fixed is default for .f):**
% export TAU_OPTIONS='-optPdtF95Opts="-R free" -optVerbose'
- **To use the compiler based instrumentation instead of PDT (source-based):**
% export TAU_OPTIONS='-optComplnst -optVerbose'
- **If your Fortran code uses C preprocessor directives (#include, #ifdef, #endif):**
% export TAU_OPTIONS='-optPreProcess -optVerbose'
- **To use an instrumentation specification file:**
% export TAU_OPTIONS=
 '-optTauSelectFile=select.tau -optVerbose -optPreProcess'

Example select.tau file

```
BEGIN_INSTRUMENT_SECTION  
loops file="*" routine="#"  
memory file="foo.f90" routine="#"  
io file="abc.f90" routine="FOO"  
END_INSTRUMENT_SECTION
```

Generate a PAPI profile with 2 or more counters

```
% export TAU_MAKEFILE=$TAU/Makefile.tau-bgqtimers-papi-mpi-pdt
% export TAU_OPTIONS='-optTauSelectFile=select.tau -optVerbose'
% cat select.tau
BEGIN_INSTRUMENT_SECTION
loops routine="#"
END_INSTRUMENT_SECTION

% export PATH=$TAU_ROOT/bin:$PATH
% make F90=tau_f90.sh
(Or edit Makefile and change F90=tau_f90.sh)
%
% qsub --env TAU_METRICS=TIME:PAPI_FP_INS:PAPI_L1_DCM -n 4 -t 15 ./a.out
% paraprof --pack app.ppk
Move the app.ppk file to your desktop.
% paraprof app.ppk
Choose Options -> Show Derived Metrics Panel -> "PAPI_FP_INS", click "/", "TIME", click
"Apply" and choose the derived metric.
```

Tracking I/O in static binaries

```
% export TAU_MAKEFILE=$TAU/Makefile.tau-bgqtimers-papi-mpi-pdt
% export PATH=$TAU_ROOT/bin:$PATH
% export TAU_OPTIONS='-optTrackIO -optVerbose'
% make CC=tau_cc.sh CXX=tau_cxx.sh F90=tau_f90.sh
% mpirun -n 4 ./a.out
% paraprof -pack ioprofile.ppk
% export TAU_TRACK_IO_PARAMS 1
% mpirun -n 4 ./a.out (to track parameters used in POSIX I/O calls as
  context events)
```

Installing and Configuring TAU

•Installing PDT:

- `wget http://tau.uoregon.edu/pdt.tgz`
- `./configure --prefix=<dir>; make ; make install`

•Installing TAU:

- `wget http://tau.uoregon.edu/tau.tgz`
- `./configure -bfd=download -pdt=<dir> -papi=<dir> ...`
- `make install`

•Using TAU:

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

Compile-Time Options (TAU_OPTIONS)

% 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
-optMemDbg	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=" <i><file></i> "	Specify selective instrumentation file for <i>tau_instrumentor</i>
-optTauWrapFile=" <i><file></i> "	Specify path to <i>link_options.tau</i> generated by <i>tau_gen_wrapper</i>
-optHeaderInst	Enable Instrumentation of headers
-optTrackUPCR	Track UPC runtime layer routines (used with tau_upc.sh)
-optPdtF95Opts=""	Add options for Fortran parser in PDT (f95parse/gfparse) ...

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_LEAKS	0	Setting to 1 turns on leak detection (for use with <code>-optMemDbg</code> or <code>tau_exec</code>)
TAU_MEMDBG_PROTECT_ABOVE	0	Setting to 1 turns on bounds checking for dynamically allocated arrays. (Use with <code>-optMemDbg</code> or <code>tau_exec -memory_debug</code>).
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_TRACK_IO_PARAMS	0	Setting to 1 with <code>-optTrackIO</code> or <code>tau_exec -io</code> captures arguments of I/O calls
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. Enabled by default to remove 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_COMPENSATE	0	Setting to 1 enables runtime compensation of instrumentation overhead
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., TIME:P_VIRTUAL_TIME:PAPI_FP_INS:PAPI_NATIVE_<event>\ :<subevent>)