Intuitive Performance Engineering at the Exascale with TAU and TAU Commander

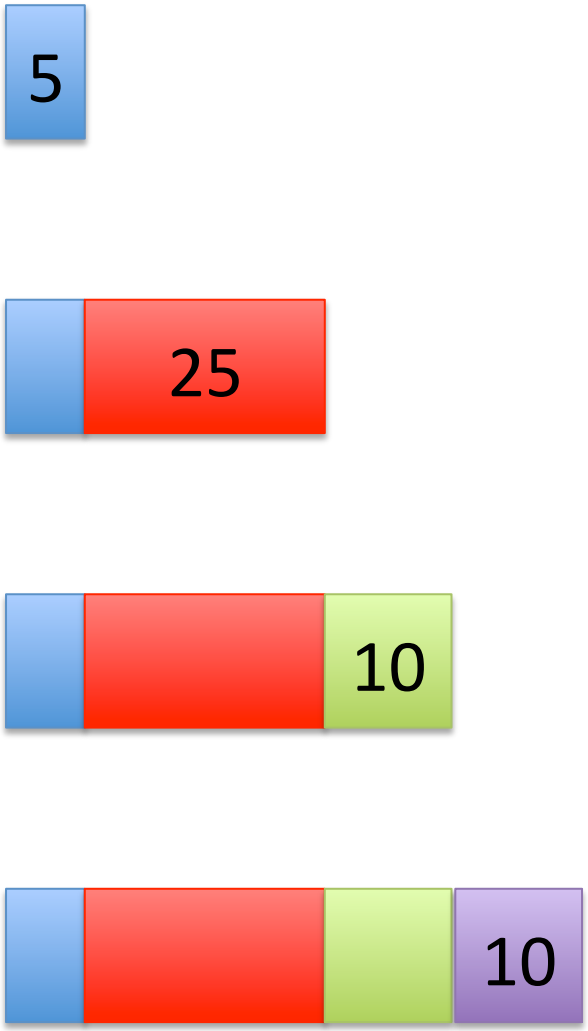
John C. Linford
ParaTools, Inc.

Argonne Extreme Scale Computing Training Program
11 August 2014, Pheasant Run "Resort"

ParaTools

Overview

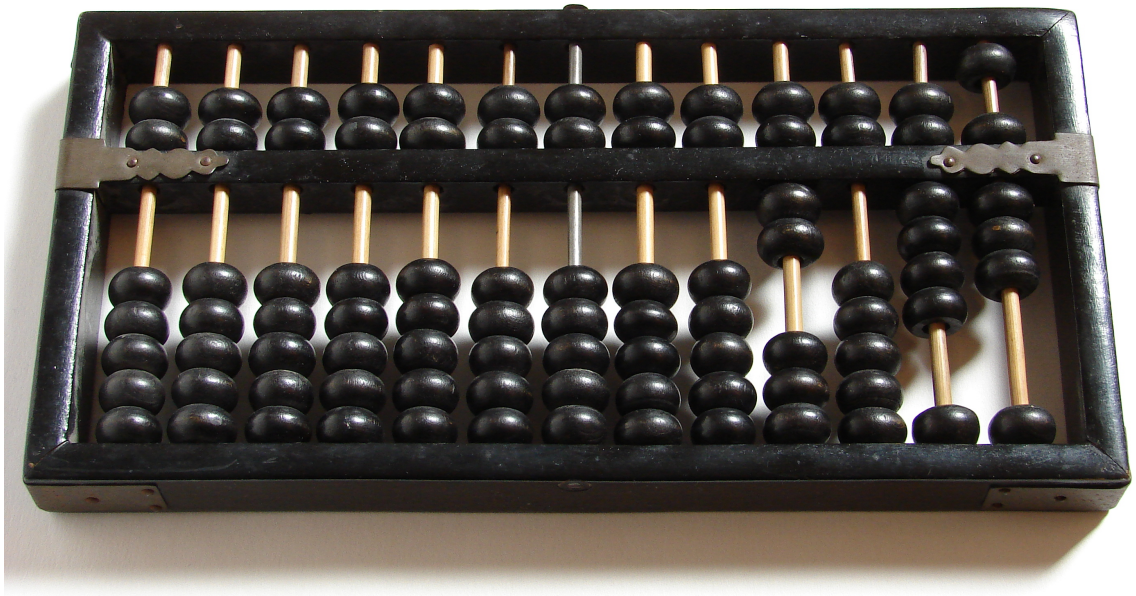
- Motivation
- TAU Overview
- TAU Commander
- Case Studies



Intuitive Performance Engineering

MOTIVATION

Why don't we just use this?



About 8 flops (0.1 flops/Watt)

Because this is better than an abacus

10 petaflops (2.17 gigaflops/Watt)

A lot better…

1.25 quadrillion abacuses

The Metrics We Care About

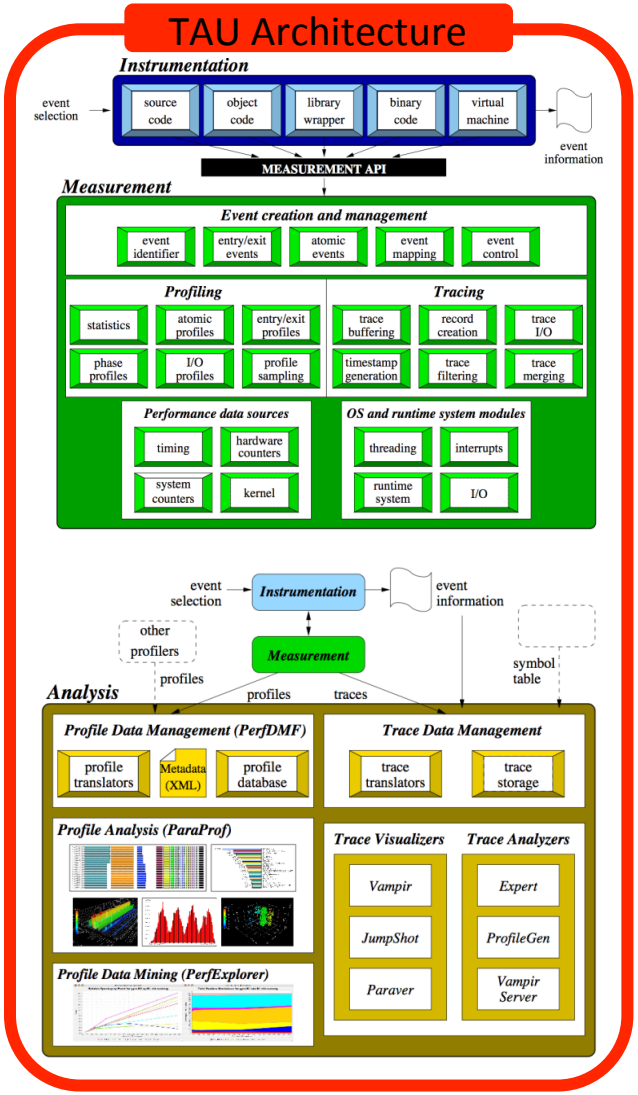
Performance

Efficiency

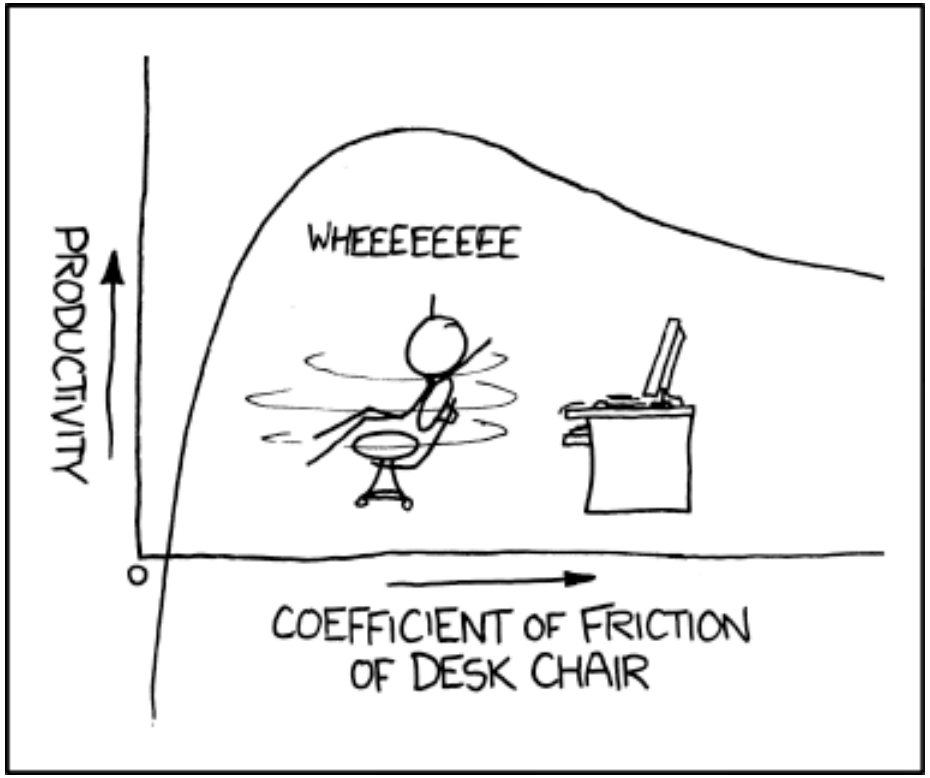
Productivity

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/



How do we Improve Productivity?



XKCD, Randall Munroe

How do we Improve Productivity?

TAU Commander

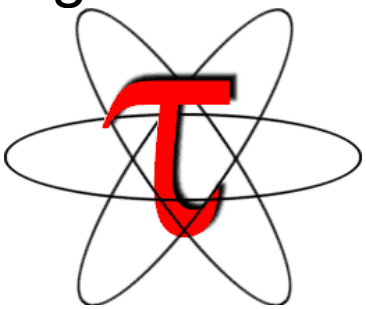
An intuitive interface to
the TAU Performance System

Intuitive Performance Engineering

THE TAU PERFORMANCE SYSTEM

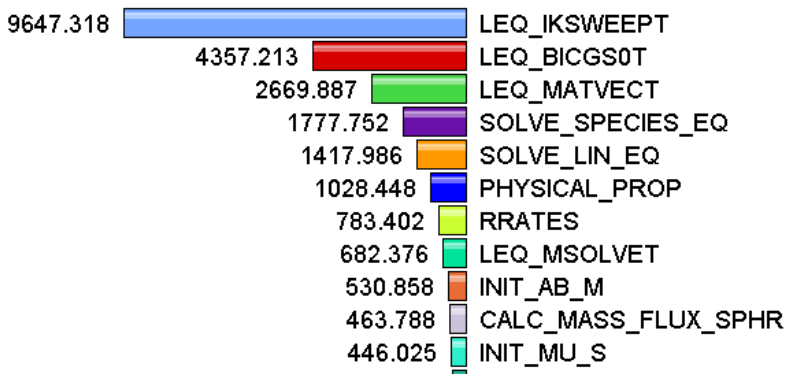
The TAU Performance System®

- 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



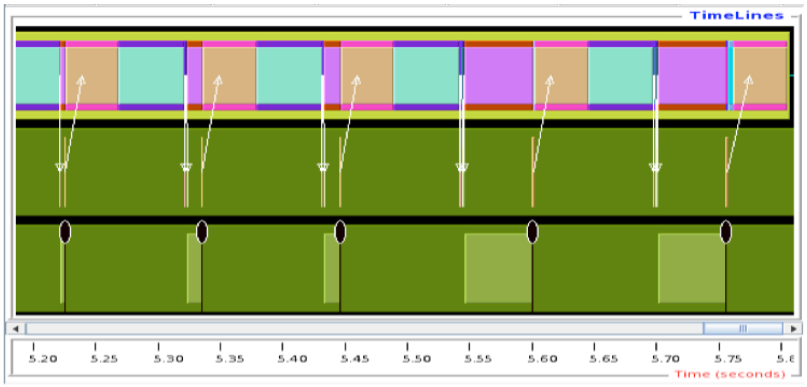
Measurement Approaches

Profiling



Shows **how much** time was spent in each routine

Tracing

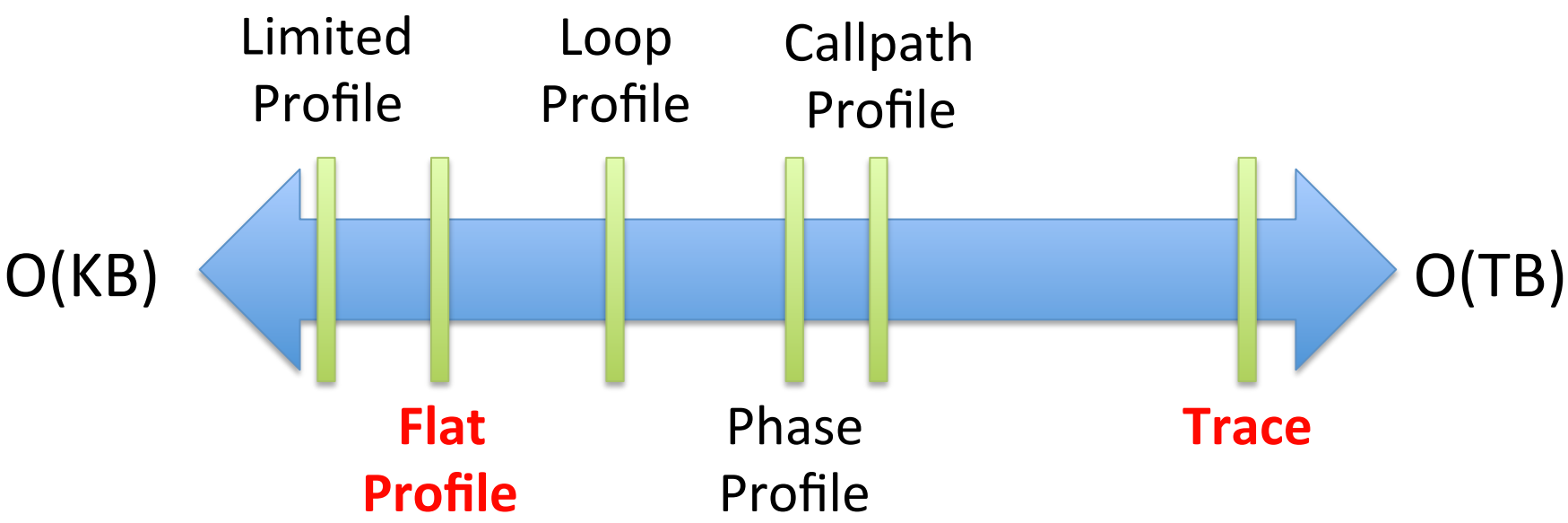


Shows **when** events take place on a timeline

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

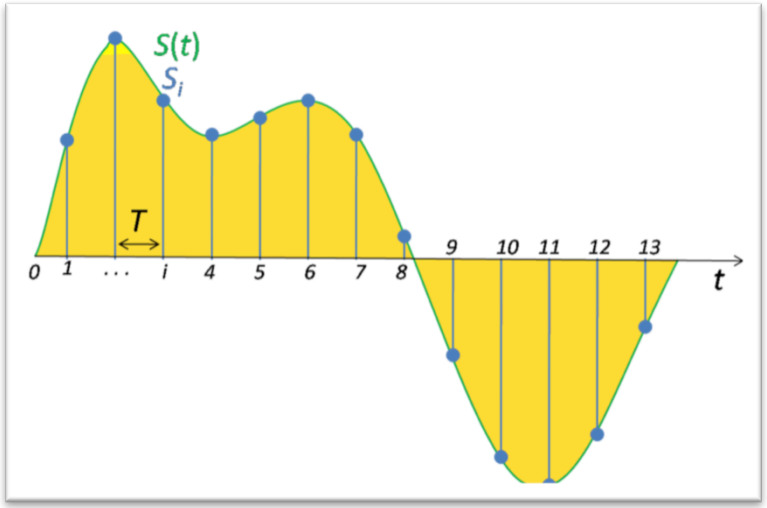
Performance Data Measurement

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 Supports All HPC Platforms

C/C++ CUDA UPC Python

GPI

Fortran OpenACC Java MPI

pthreads Intel MIC OpenMP

Intel GNU

LLVM PGI Cray Sun

MinGW

Linux Windows AIX

Insert yours here

BlueGene Fujitsu ARM

Android MPC OS X

TAU Architecture and 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
- Threadding, Communication
- Heterogeneous

Profiling
- Flat, Callpath, Phase, Snapshot
- Probe, Sampling, Compiler, Hybird

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

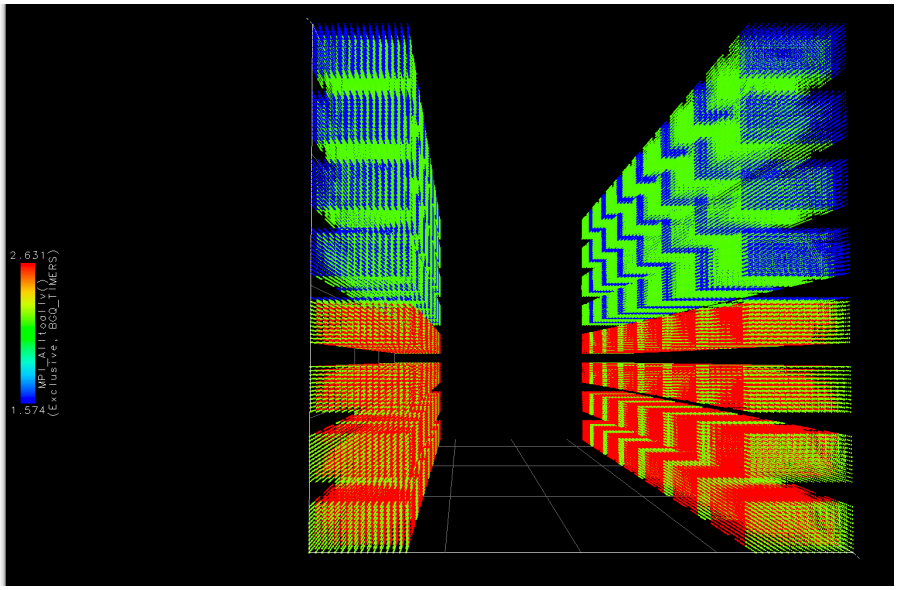
- *Source code* instrumentation
  - PDT parsers, pre-processors
- *Wrap* external libraries
  - I/O, MPI, Memory, CUDA, OpenCL, pthread
- *Rewrite* the binary executable
  - Dyninst, MAQAO

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

```
% ls /soft/perftools/tau/tau_latest/bgq/lib/Makefile.*
Makefile.tau-bgqtimers-mpi-pdt-openmp-opari
Makefile.tau-bgqtimers-mpi-pthread-pdt
Makefile.tau-bgqtimers-papi-mpi-pdt
Makefile.tau-bgqtimers-papi-mpi-pdt-openmp-opari
Makefile.tau-bgqtimers-papi-mpi-pthread-pdt
Makefile.tau-bgqtimers-pdt
Makefile.tau-papi-mpi-pdt-openmp-opari
Makefile.tau-papi-mpi-pdt-openmp-opari-scorep
Makefile.tau-papi-mpi-pdt-scorep
```

Using TAU: A Brief Introduction

1. Choose an appropriate `TAU_MAKEFILE`:

```
% soft add +tau-latest
% export TAU_MAKEFILE=/soft/perftools/tau/tau_latest/
      bgq/lib/Makefile.tau-bgqtimers-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)

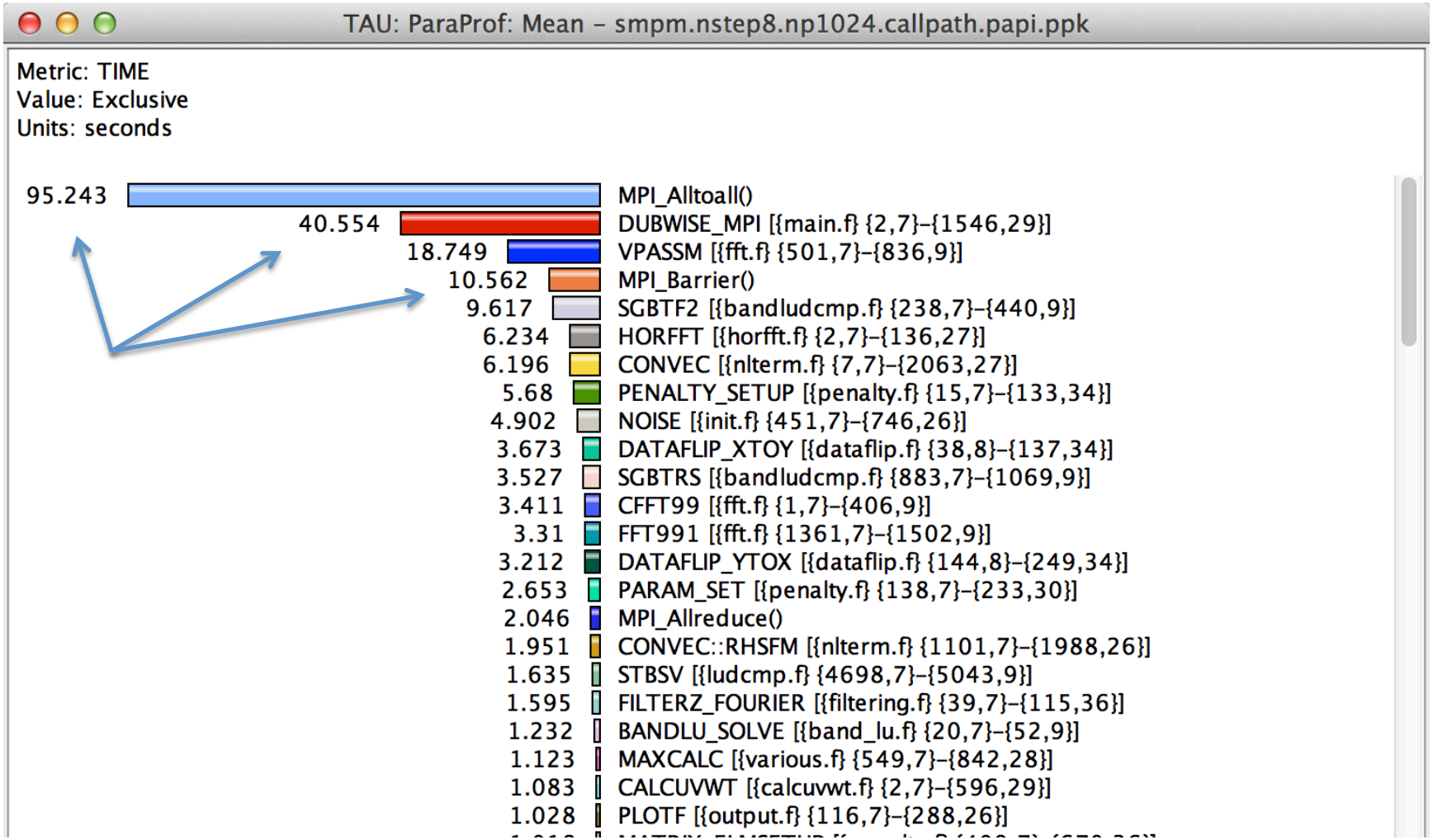
Hands-on (18:30 – 21:30)

```
% 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?

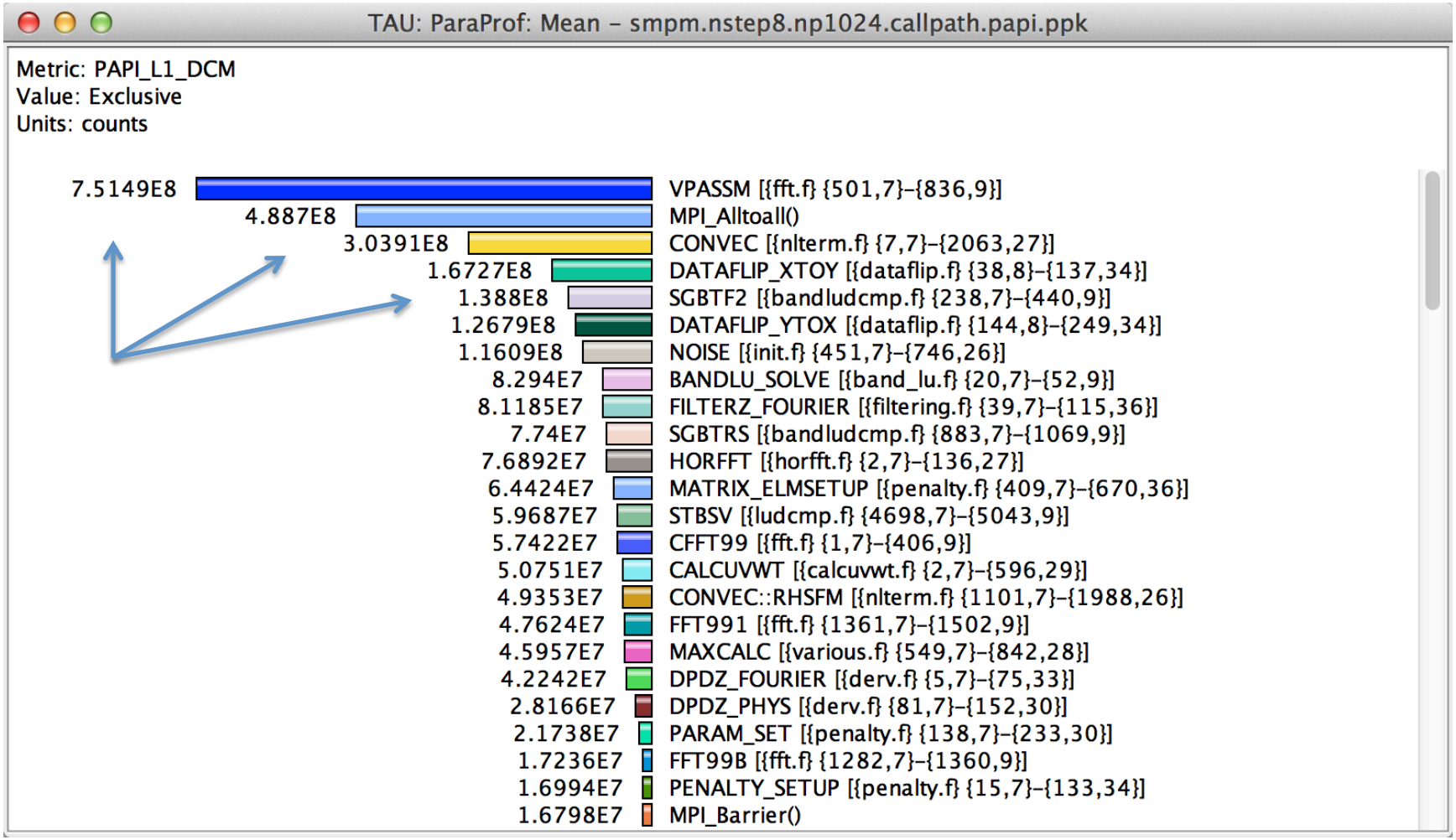
TAU: ParaProf: Mean – smpm.nstep8.np1024.callpath.papi.ppk

Metric: PAPI_FP_INS
Value: Exclusive
Units: counts

8.8479E9	VPASSM [{fft.f} {501,7}-{836,9}]
4.1824E9	SGBTF2 [{bandludcmp.f} {238,7}-{440,9}]
4.0701E9	NOISE [{init.f} {451,7}-{746,26}]
1.384E9	FILTERZ_FOURIER [{filtering.f} {39,7}-{115,36}]
1.1515E9	STBSV [{ludcmp.f} {4698,7}-{5043,9}]
7.8914E8	DPDZ_FOURIER [{derv.f} {5,7}-{75,33}]
6.3193E8	DPDZ_PHYS [{derv.f} {81,7}-{152,30}]
6.0732E8	SGBTRS [{bandludcmp.f} {883,7}-{1069,9}]
5.4591E8	FFT99B [{fft.f} {1282,7}-{1360,9}]
5.1141E8	FILTERXY_FOURIER [{filtering.f} {119,7}-{172,37}]
4.2268E8	FFT99A [{fft.f} {1208,7}-{1281,9}]
4.0761E8	CONVEC::RHSFM [{nlterm.f} {1101,7}-{1988,26}]
3.2453E8	MATRIX_ELMSETUP [{penalty.f} {409,7}-{670,36}]
3.0524E8	CONVEC [{nlterm.f} {7,7}-{2063,27}]
1.6188E8	DIVCOLC [{derv.f} {207,7}-{273,29}]
1.269E8	HORFFT [{horfft.f} {2,7}-{136,27}]
1.1585E8	CONVEC::ROTVISC [{nlterm.f} {1994,8}-{2038,29}]
7.4375E7	DPDZCOL [{derv.f} {159,7}-{203,28}] [THROTTLED]
5.8376E7	UPDATE [{calcuvwt.f} {603,7}-{726,27}]
4.669E7	MAXCALC [{various.f} {549,7}-{842,28}]
4.0317E7	RELAX_RESET [{postp.f} {765,7}-{868,32}]
3.6434E7	PERTURBCALC [{postp.f} {693,7}-{759,32}]
2.6551E7	CALCUVWT [{calcuvwt.f} {2,7}-{596,29}]
2.1258E7	PENALTY_MAIN [{penalty.f} {237,7}-{286,33}]

% **`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?

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

High-water mark

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

How Much Memory Does the Code Use?

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

Total allocated/deallocated

% **`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()						

Peak MPI-IO Write Bandwidth

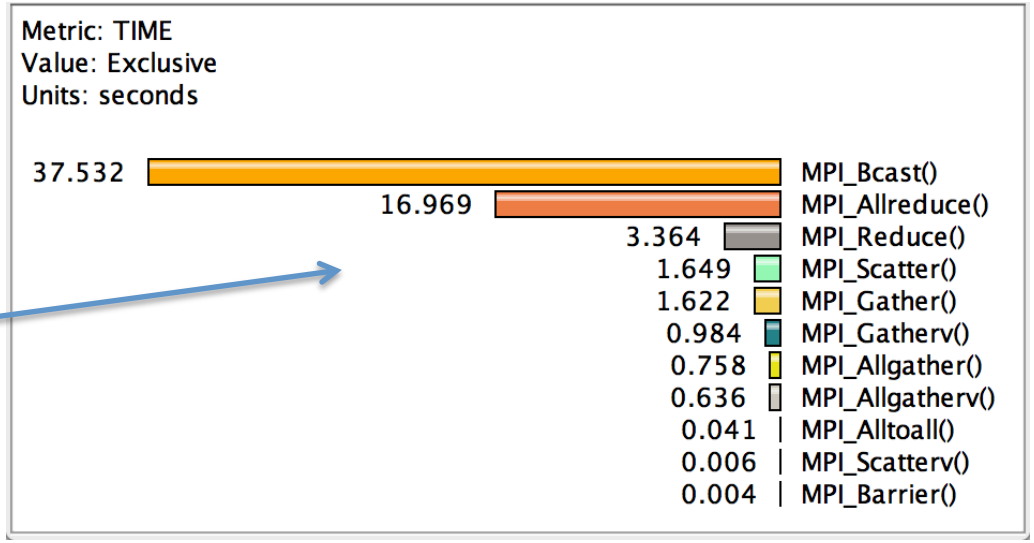
% **`paraprof`** **(Right-click label [e.g "node 0"] → Show Context Event Window)**

How Much Time is spent in Collectives?

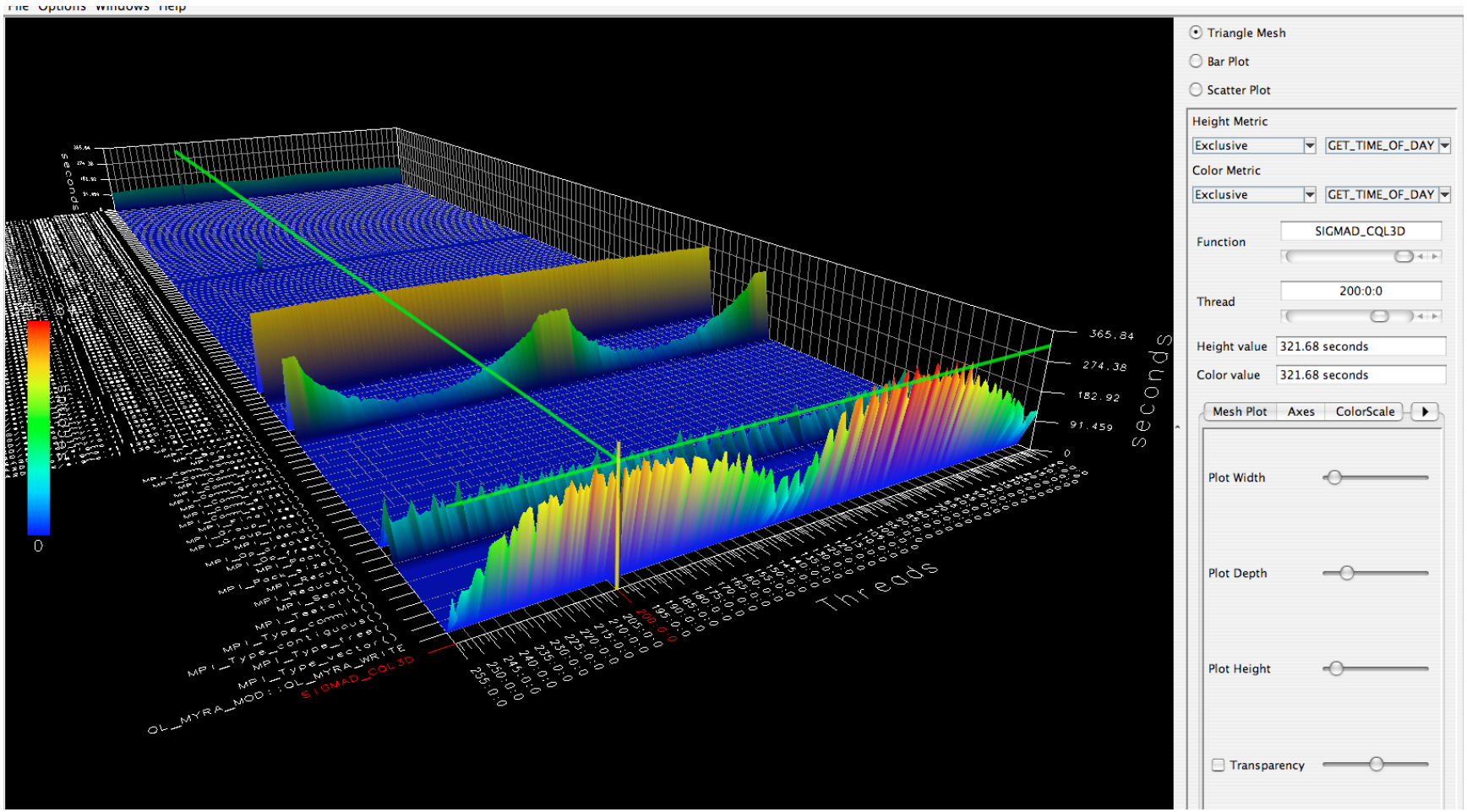
Name △	Total	Num...	MaxValue	MinValue	MeanValue	Std. Dev.
▶ MPI_Wait()						
▶ MPI_Waitall()						
Message size for all-gather	305,753,268	72	172,215,296	4	4,246,573.167	22,551,605.859
Message size for all-reduce	163,308	632	21,908	4	258.399	897.725
Message size for all-to-all	112	14	8	8	8	0
Message 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

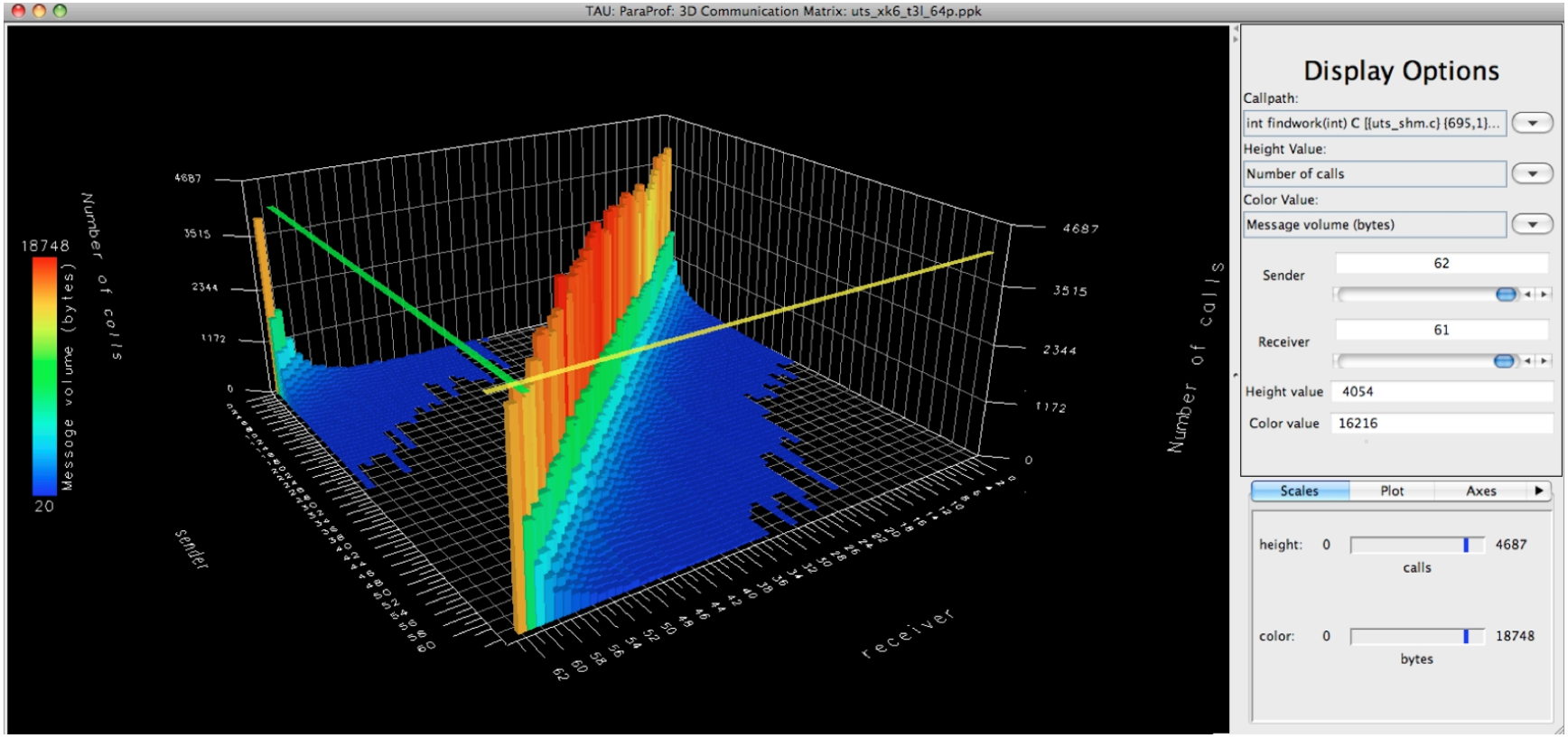


3D Profile Visualization



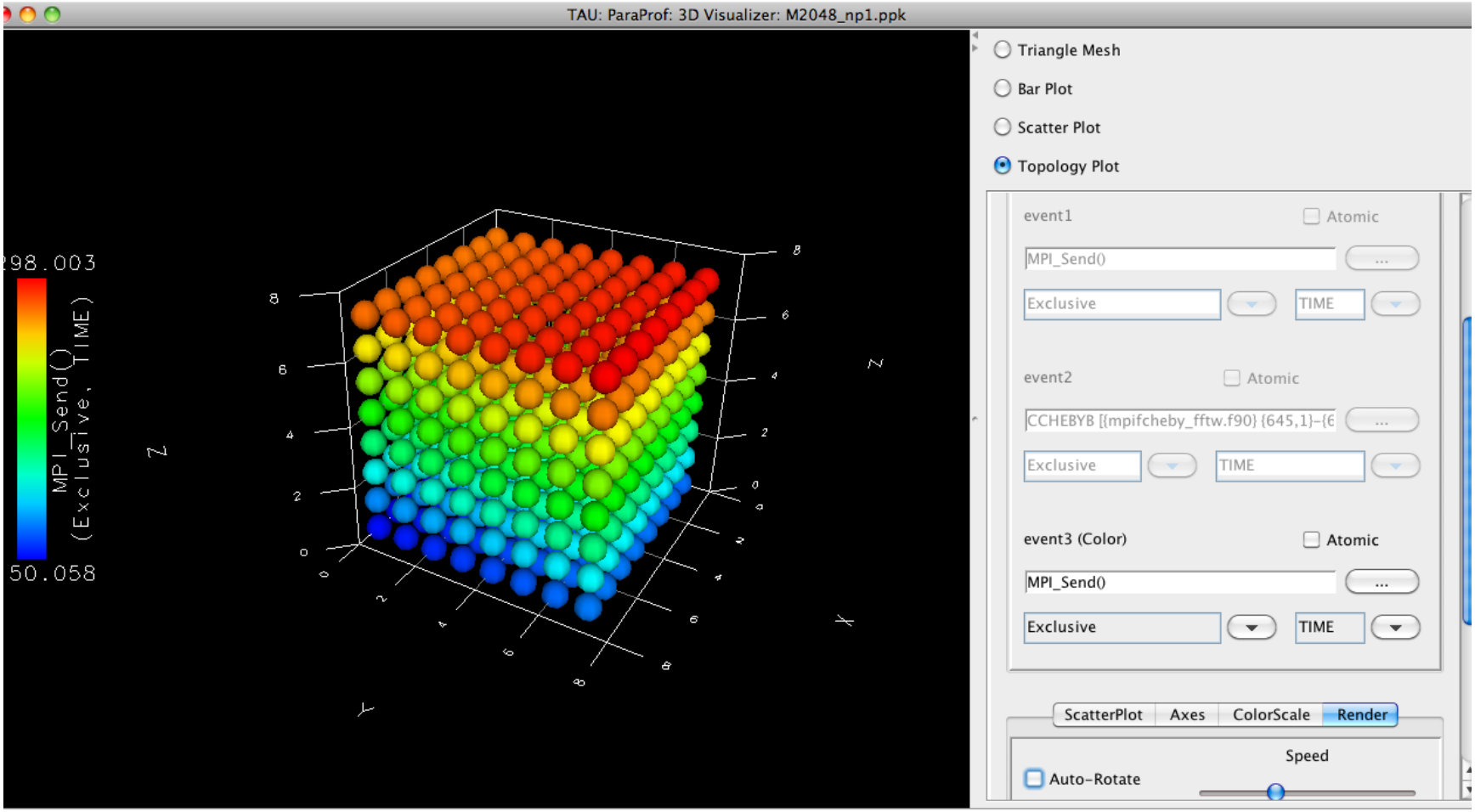
% `paraprof` (Windows → 3D Visualization)

3D Communication Visualization



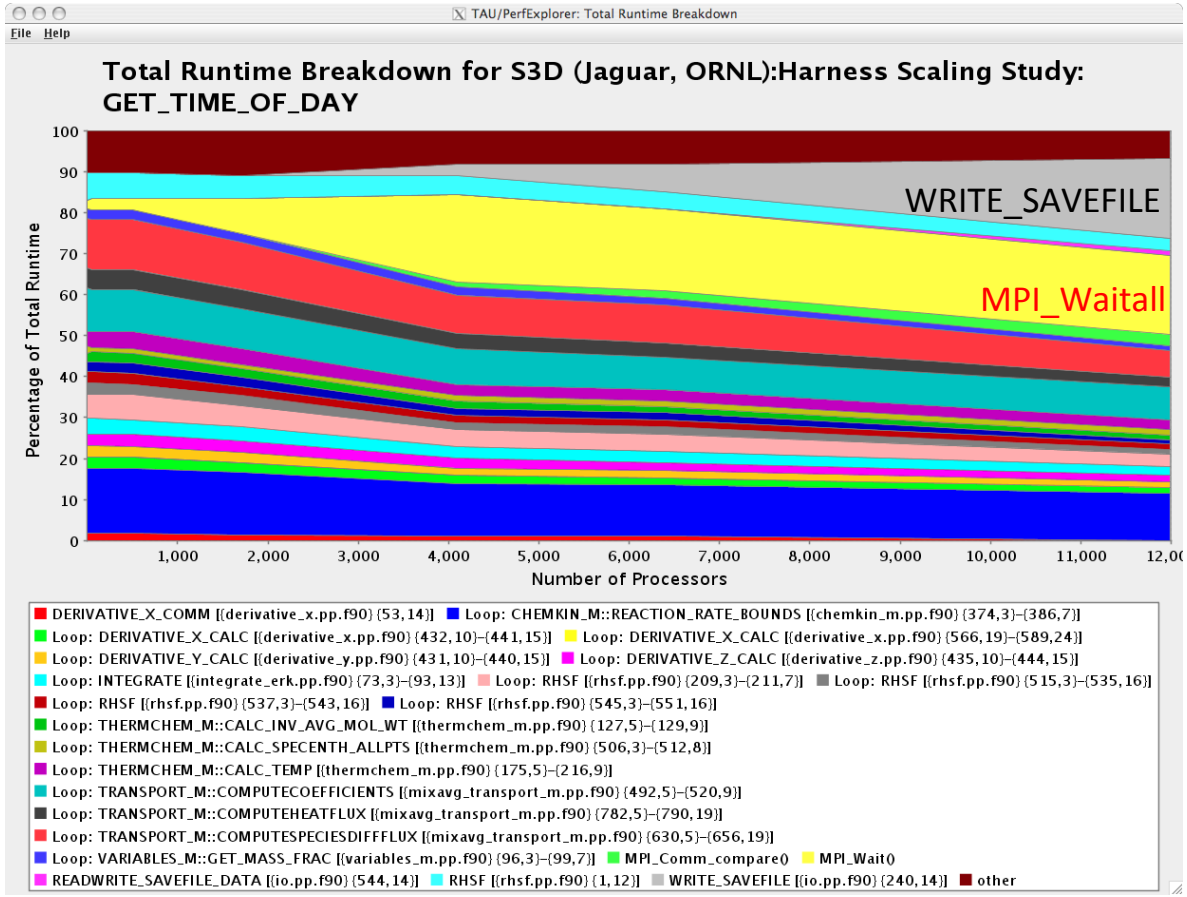

```
% qsub –env TAU_COMM_MATRIX=1 …
% paraprof (Windows → 3D Communication Matrix)
```

3D Topology Visualization



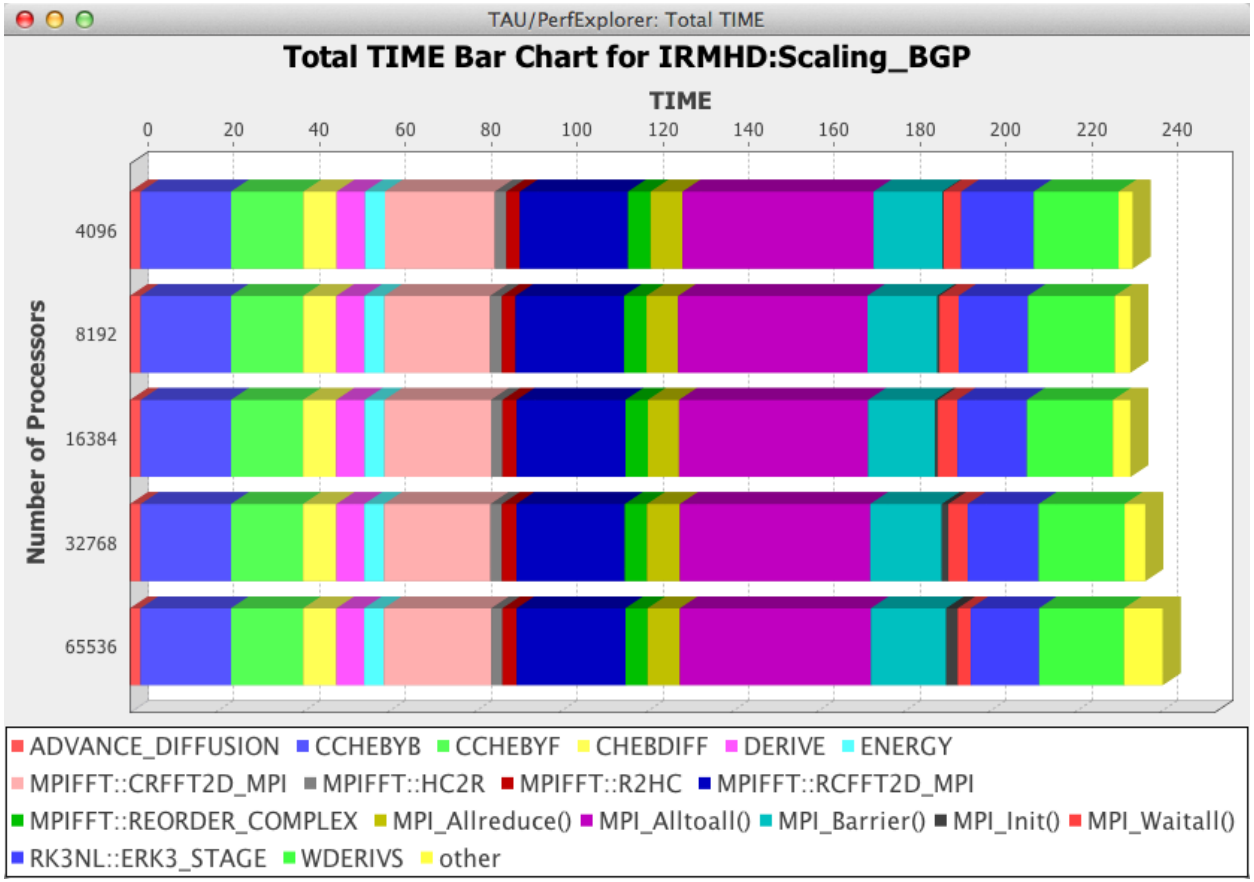
% `paraprof` (Windows → 3D Visualization → Topology Plot)

How Does Each Routine Scale?



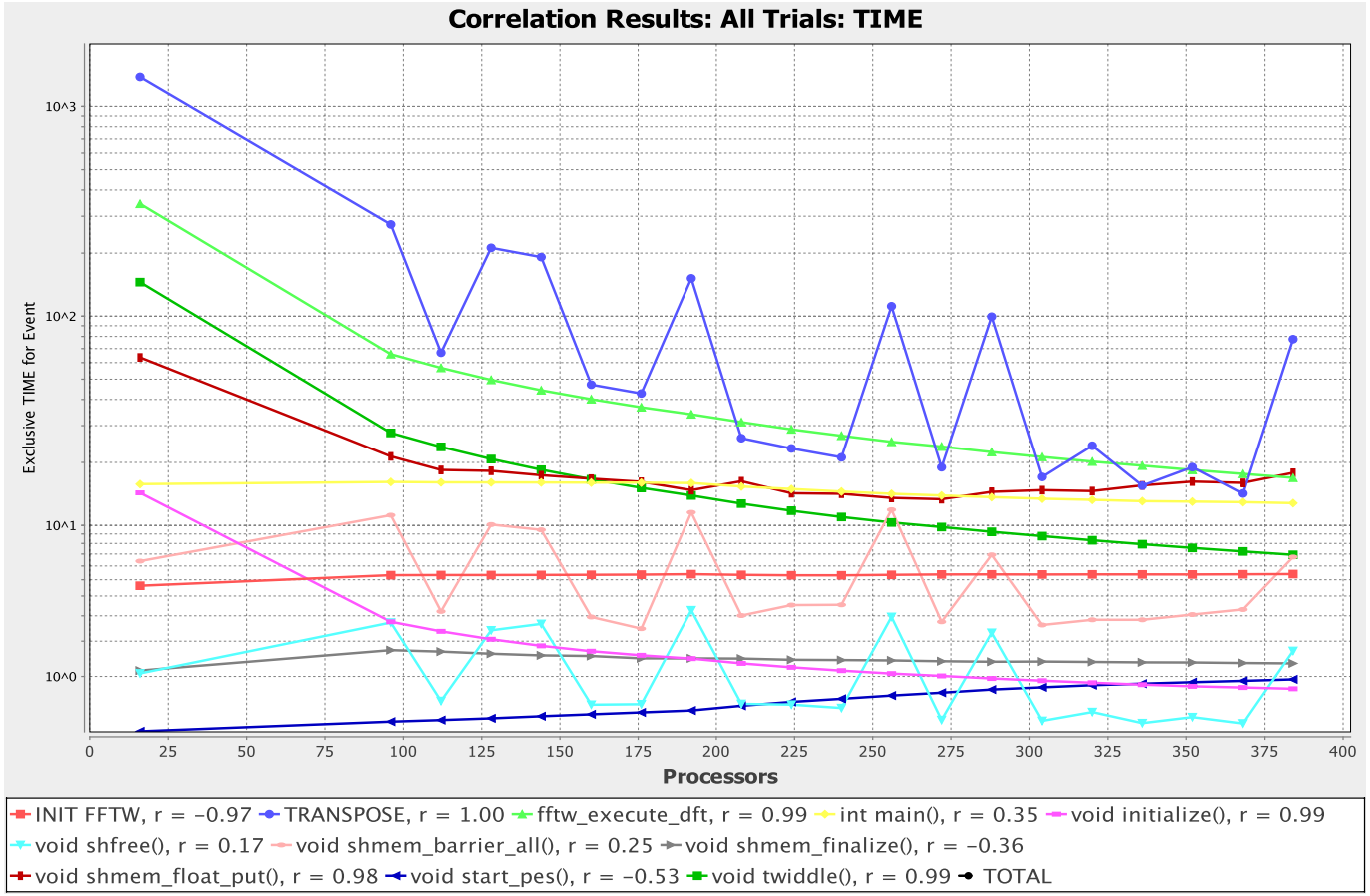
`% perfexplorer` (Charts → Runtime Breakdown)

How Does Each Routine Scale?



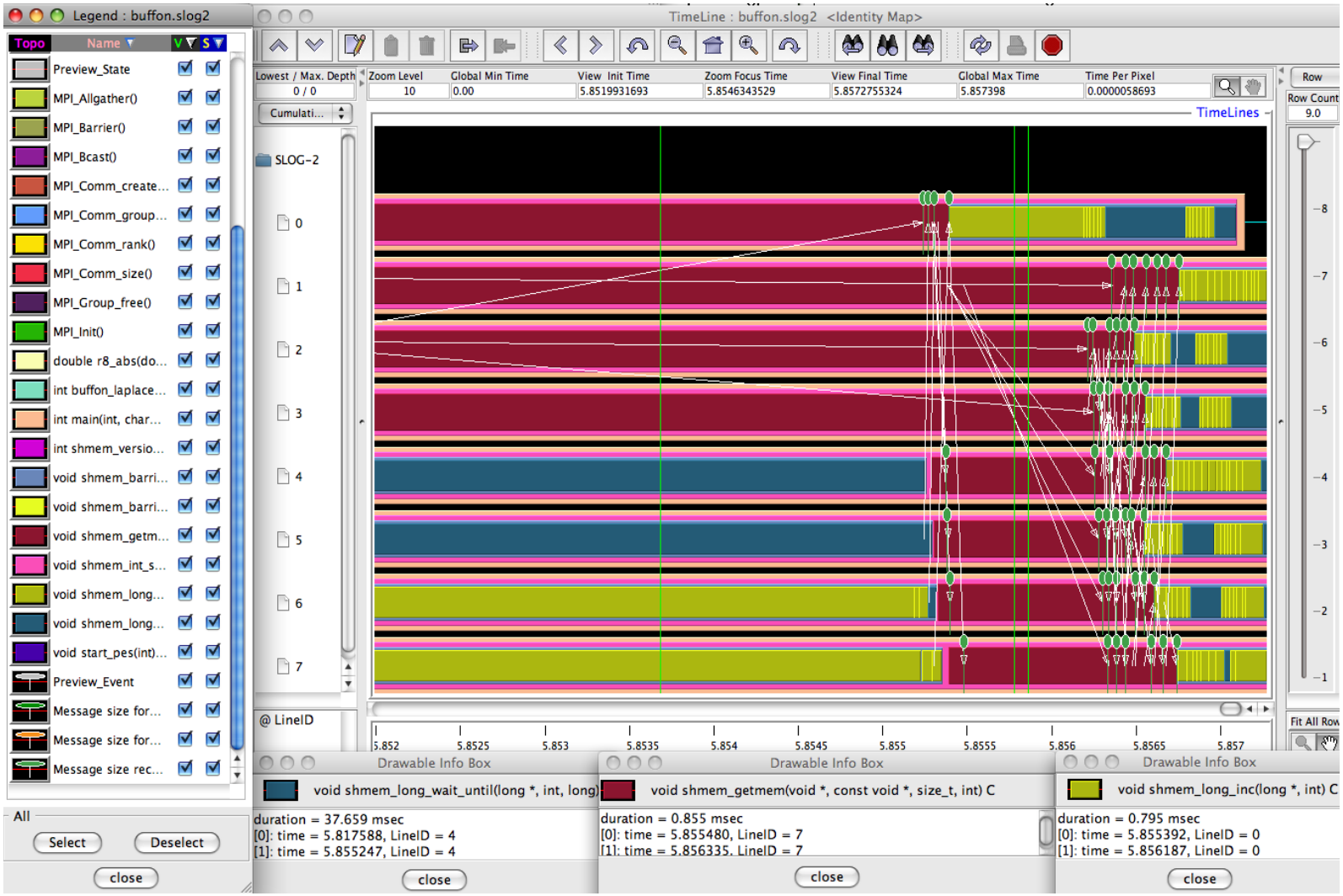
`% perfexplorer` **(Charts → Stacked Bar Chart)**

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?

TAU: ParaProf Manager

Options Help

pplications
- Standard Applications
- Default App
  - Default Exp
    - py-c++-f90-create.ppk
      - TIME

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/pycintfc.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...
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/libpytho...
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/libpython...
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/libpytho...
BACKTRACE 13	[builtin___import__] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/bltinmodule.c:49] [/mnt/cfs/pkgs/PTOOLS/pkgs/ptoolsrte-0.55/packages/Python-2.7.2/lib/libpytho...
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...
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...
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.s...
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...
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...
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...
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/lib...
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

Metadata fo 0,0

Show Source Code

Name	Value
BACKTRACE 1	[SAMINT::timestep(double, double)] [/mnt/home/jlinford/py-c++-f90-create/SAMI linford/py-c++-f90-create/_samint.so]
BACKTRACE 2	[samarcStep(double, double)] [/mnt/home/jlinford/py-c++-f90-create/pycintfc.C: /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/libpytho...
BACKTRACE 13	[builtin__import__] [/mnt/home/jlinford/0.55/build/Python-2.7.2/Python/bltinmodule.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	[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 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/lib...
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?

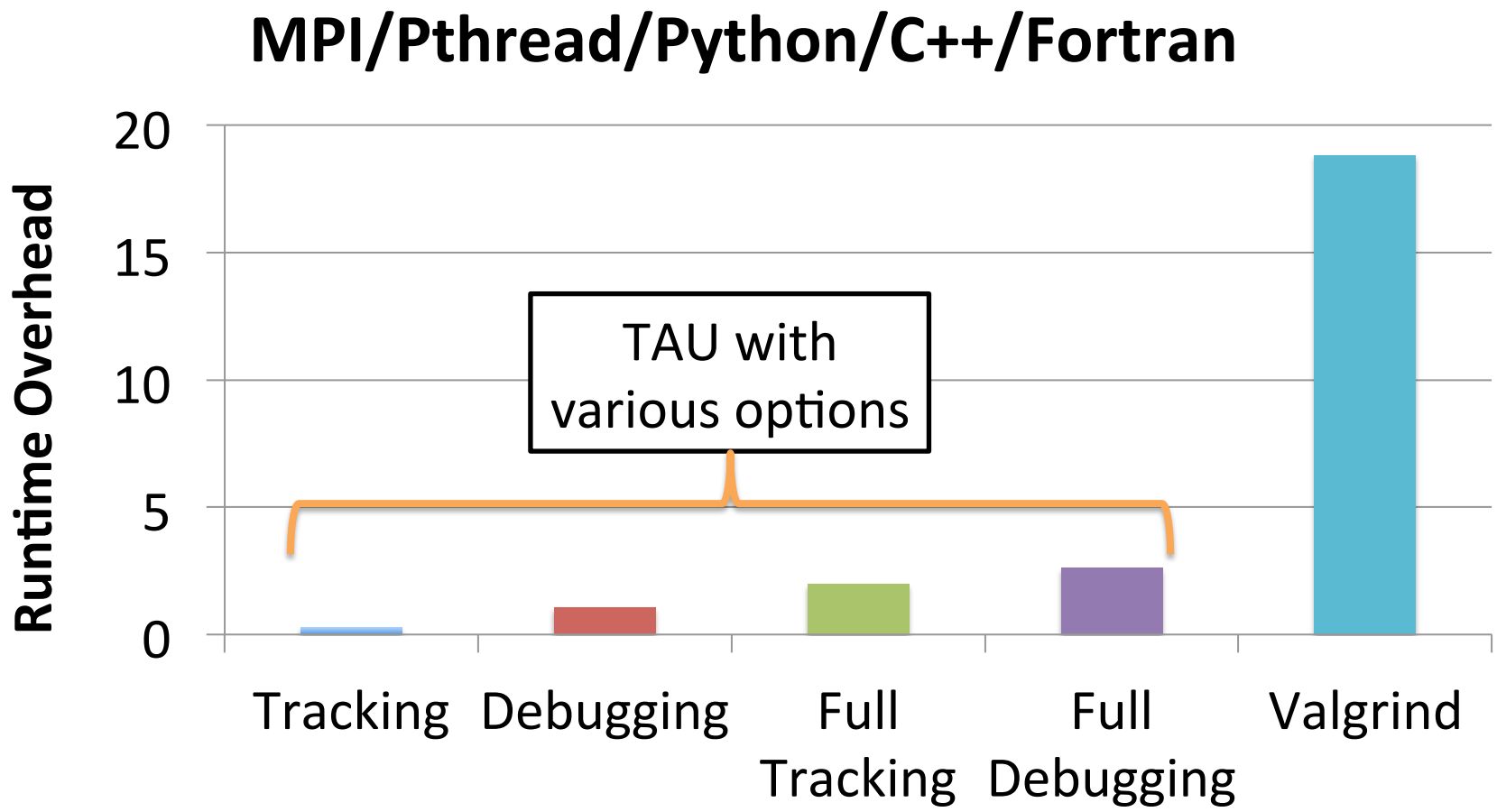
TAU: ParaProf: Source Browser: /mnt/home/jlinford/py-c++-f90-create/SAMINT.C

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      *
85      * Write data to output
86      * (visit, fieldview, or overgrid - set in samarc input file)
87      *
88      ************************************************************************
89      */
90      void SAMINT::writePlotData(const double time,
91                                 const int step)
92      {
93         cout << "SAMINT::writePlotData()" << endl;
```

Error shown in ParaProf Source Browser

Memory debugging

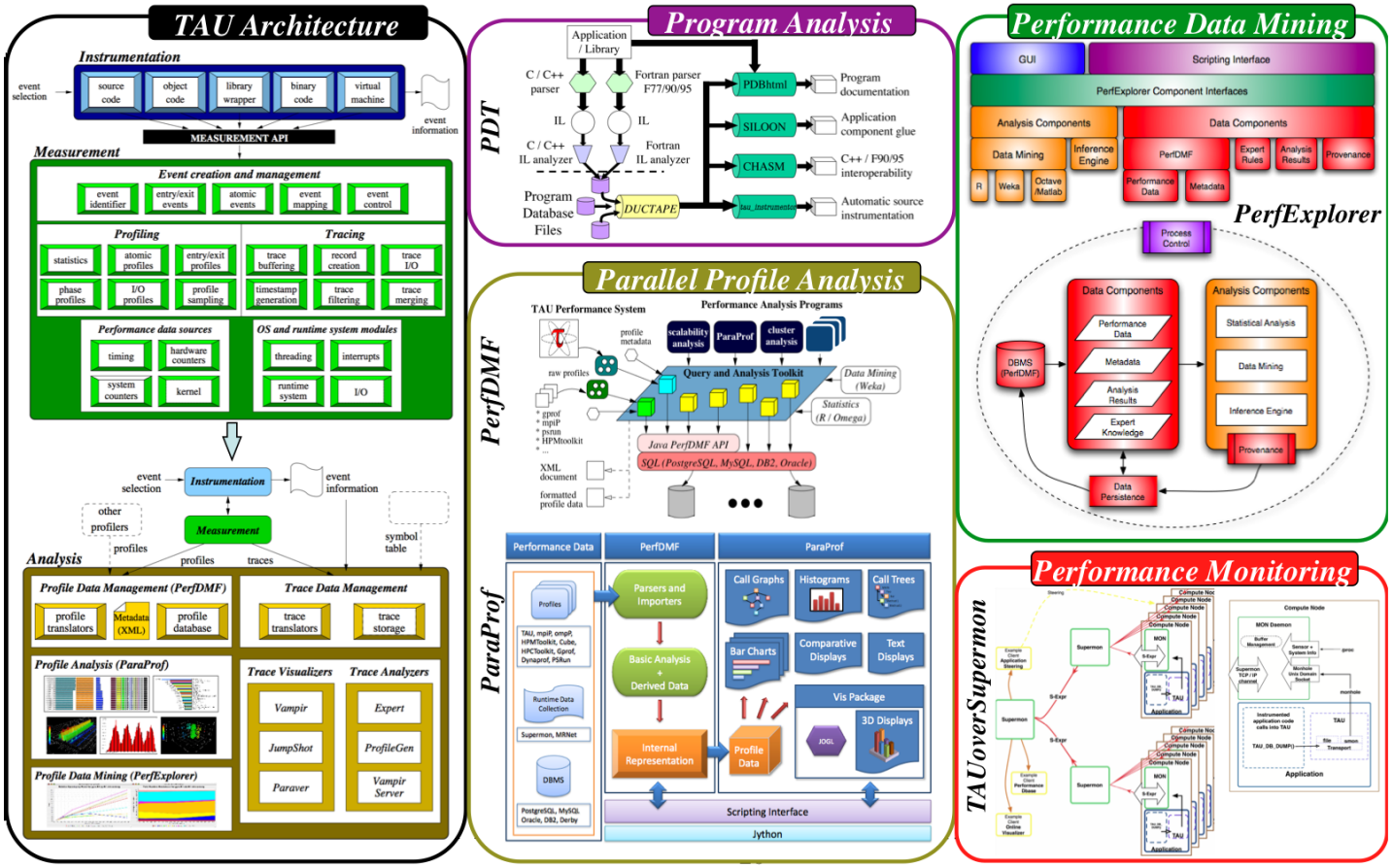


Note: Requires working mprotect() so BGQ not supported

Intuitive Performance Engineering

TAU COMMANDER

TAU: Powerful and Complex



How do we navigate?

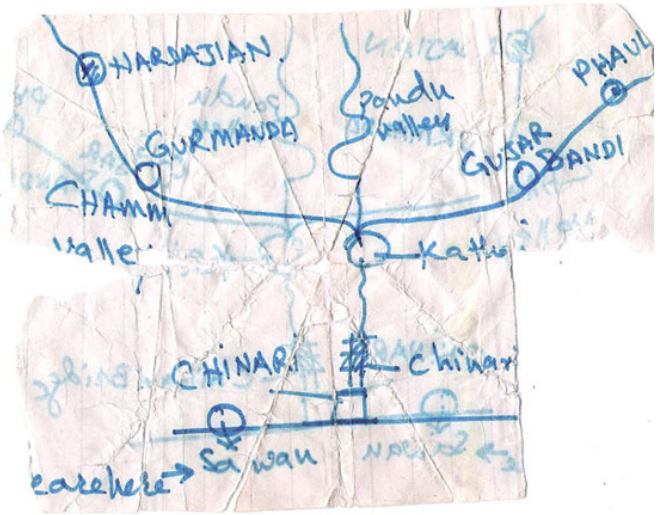
TAU Commander

- Two-year $1M project, DOE SBIR Phase II
- Simplify TAU usage
  - Develop a common framework for TAU user interface
  - Develop an intuitive, high-productivity user interface
- Phase I results:
  - Study of 124 workflows involving eleven test cases on six computing systems
  - TAU Commander reduces number of steps in TAU workflow by approximately 50%
  - Reduces the number of commands a user must know from approximately eight to exactly one

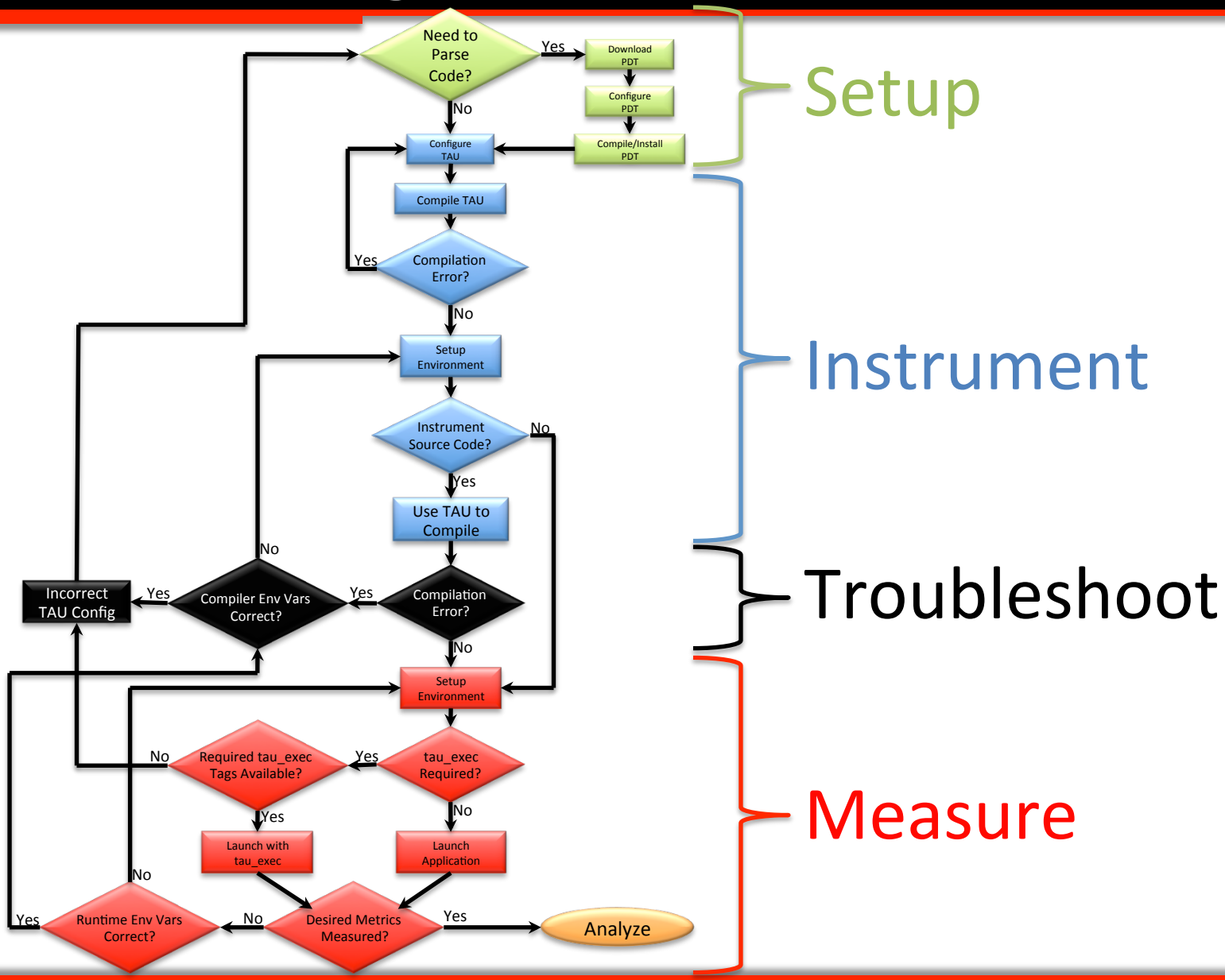
The TAU Commander Approach

- 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

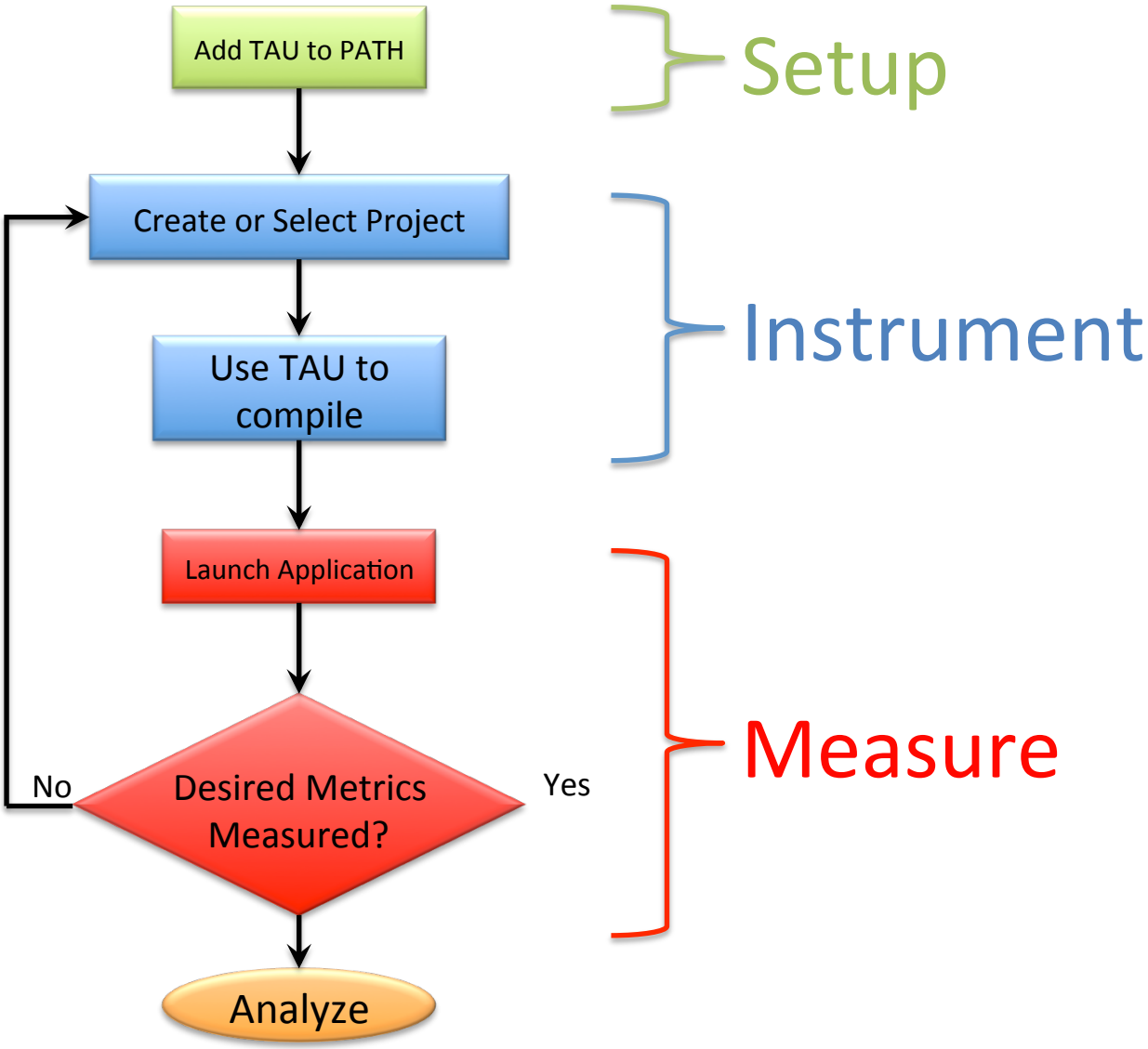
vs.



TAU Usage (SBIR Phase I)



TAU Commander Usage (SBIR Phase I)



TAU Commander User Interface

jlinford — jlinford@mavericks-pro:~ — bash — 82×36

```
jlinford@mavericks-pro ~ $ export PATH=/Users/jlinford/workspace/taucmd/bin:$PATH
jlinford@mavericks-pro ~ $ tau --help
The Tau Performance System (2.22.3-git)
http://tau.uoregon.edu/

Usage:
  tau [options] <command> [<args>...]
  tau -h | --help
  tau --version

Options:
  --verbose        Same as --log=DEBUG
  --quiet          Same as --log=CRITICAL
  --log=<level>    Output level, one of CRITICAL, ERRROR, WARNING, INFO, or DEBUG.

Commands:
  build            Instrument programs during compilation and/or linking.
  help             Get help with a command.
  make             Build your application with 'make' and the TAU compilers.
  pack             Package profile files into a PPK file.
  project          Create and manage TAU projects.
  run              Gather measurements from an application.
  show             Display application profile or trace data.

Shortcuts:
  <compiler>       A compiler command, e.g. gcc, mpif90, upcc, nvcc, etc.
                   An alias for 'tau build <compiler>'
  <executable>     A program executable, e.g. ./a.out
                   An alias for 'tau execute <executable>'
  <profile>        View profile data (*.ppk, *.xml, profile.*, etc.) in ParaProf
                   An alias for 'tau show <profile>'
  <trace>          View trace data (*.otf *.slog2, etc.) in Jumpshot
                   An alias for 'tau show <trace>'

See 'tau <command> --help' for more information on a specific command.
jlinford@mavericks-pro ~ $
```

TAU Commander Example Usage

Step	Action	Commands
1	Add TAU to path	`export PATH=$HOME/workspace/taucmd:$PATH`

2	Create project	`cd matmult`
		`tau project create matmult --openmp --mpi \`
		`  --callpath=100 --memory`

3	Instrument with TAU	`tau mpif90 *.f90 -o matmult`
4	Execute instrumented application	`tau qsub –A <queue> –q R.bc –n 256 –t 10 ./matmult`
5	Visualize data	`tau show`

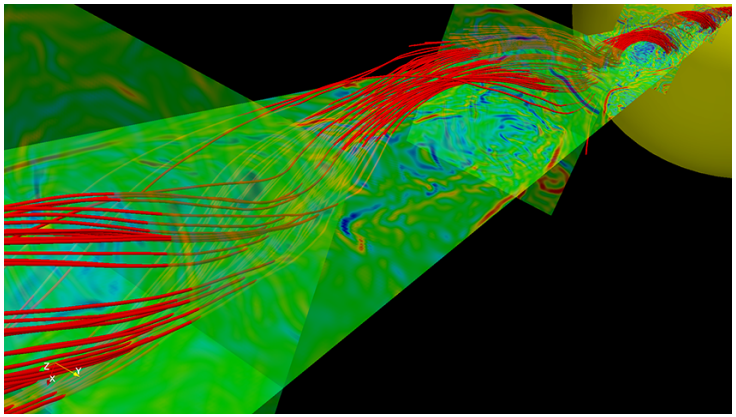
Note: the 'tau' command is in beta. Its usage may change.

Intuitive Performance Engineering

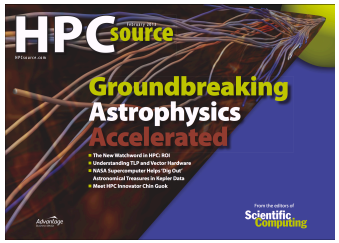
CASE STUDIES

IRMHD on Intrepid and Mira

- INCITE magnetohydrodynamcis 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)

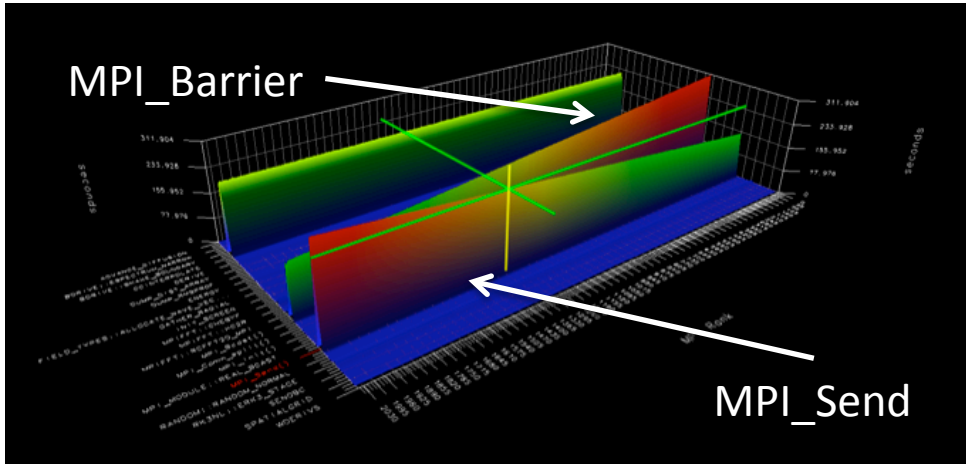


- IRMHD (Inhomogeneous Reduced Magnetohydrodynamics)
  - Fortran 90 and MPI
  - Excellent weak and strong scaling properties
  - Tested and benchmarked on Intrepid and Mira
- HPC Source article and ALCF news
  https://www.alcf.anl.gov/articles/furthering-understanding-coronal-heating-and-solar-wind-origin



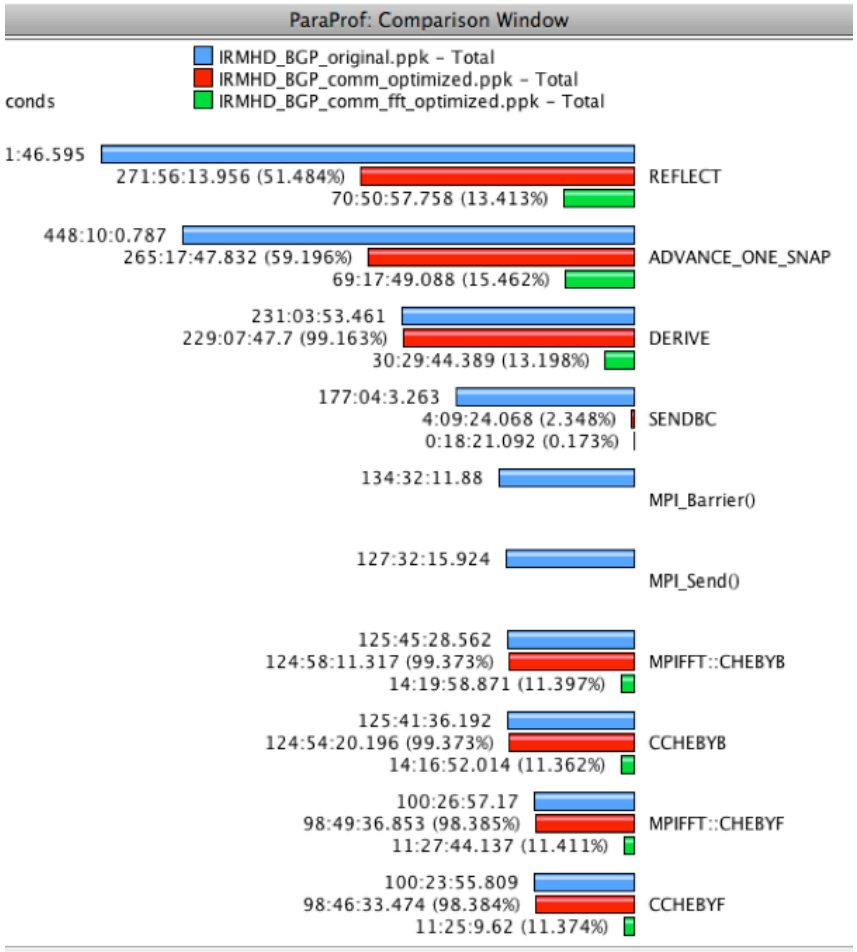
IRMHD Communication Analysis

- 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



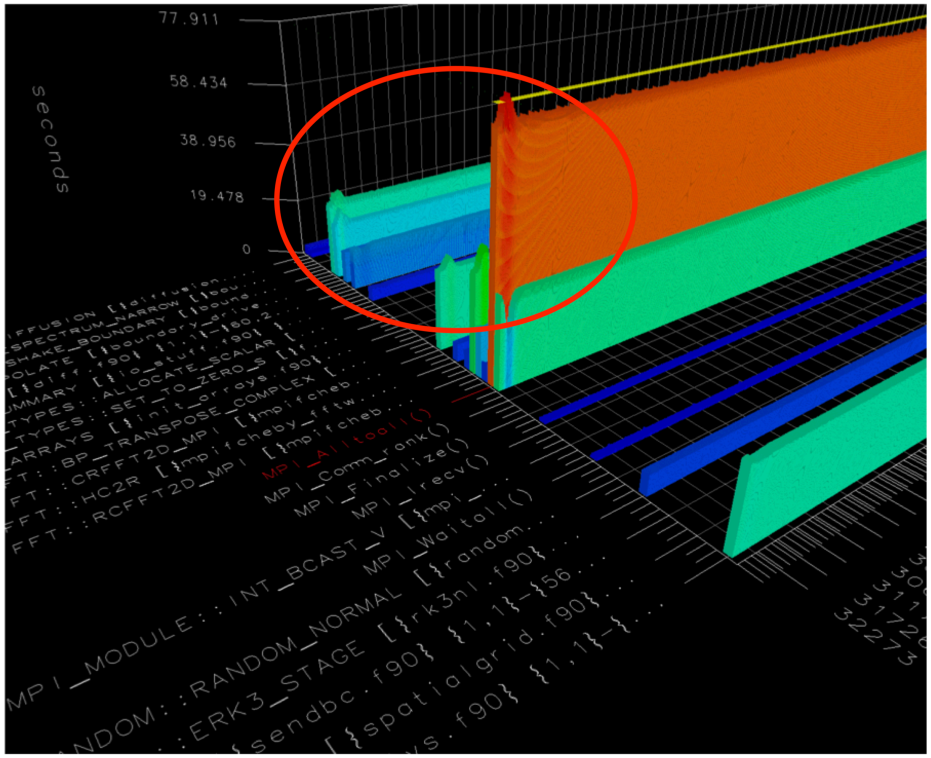
IRMHD Optimization on Intrepid (BG/P)

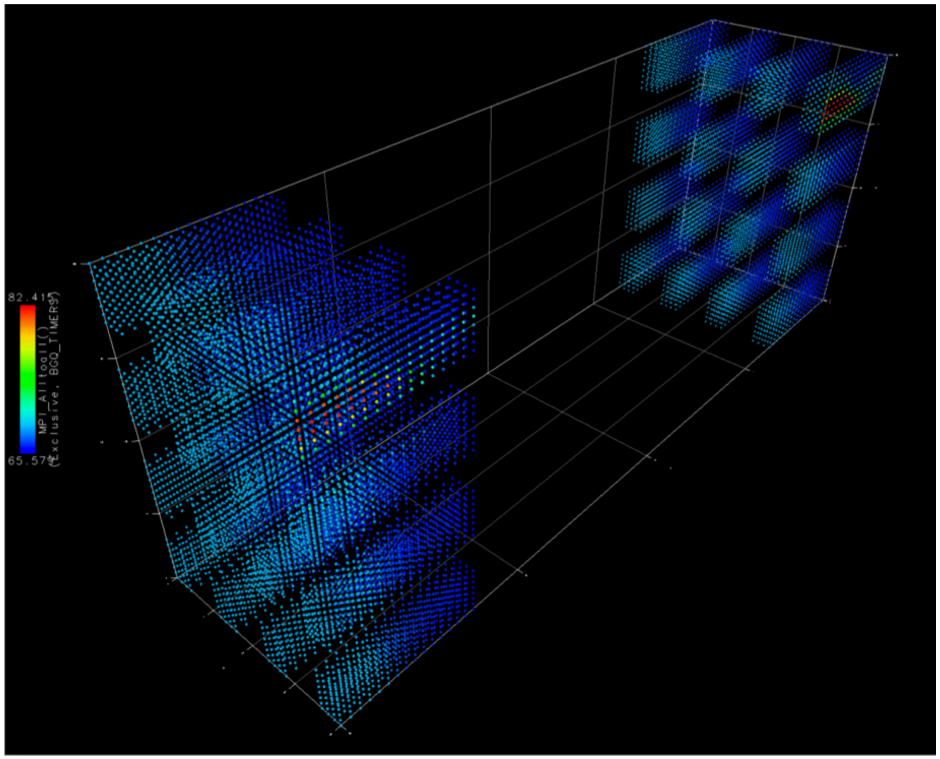
- On 2,408 cores, overall execution time reduced from 528.18 core hours to 70.8 core hours **(>7x improvement)**
- Non-blocking communication substrate
- More efficient implementation of underlying FFT



IRMHD Optimization on Mira (BG/Q)

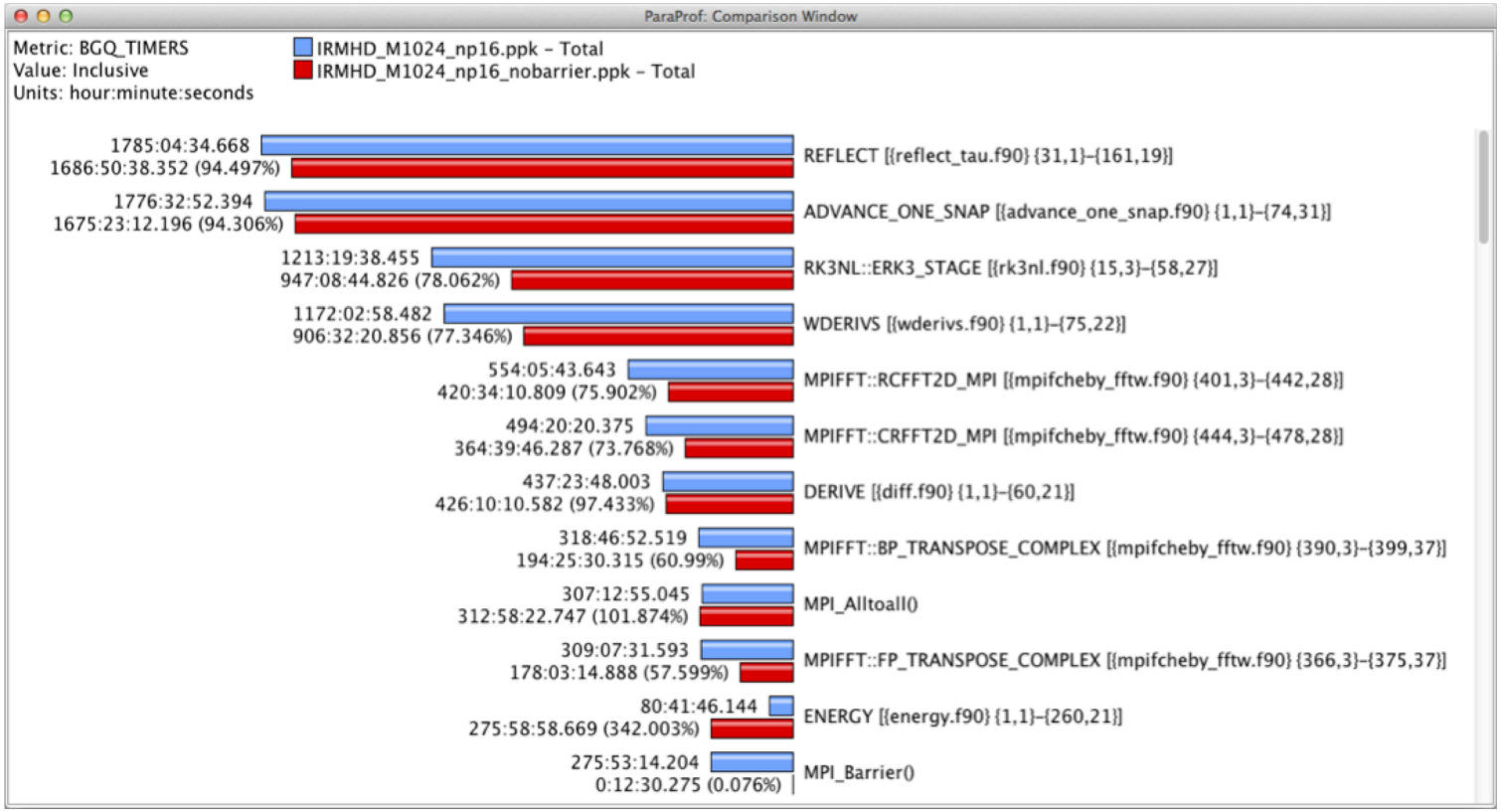
- At 32K cores, load imbalance visible in topology
  - Negative impact on MPI_Alltoall performance





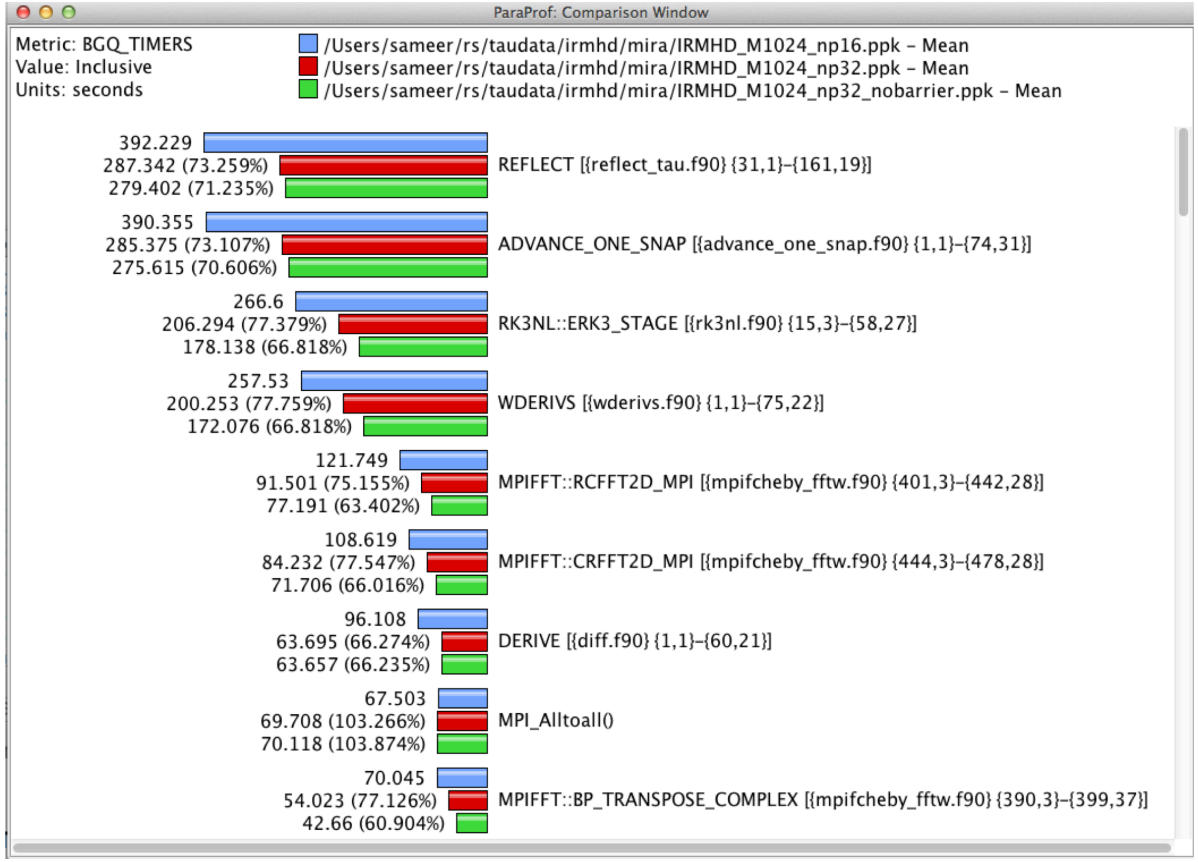
IRMHD Optimization on MIRA (BG/Q)

- Remove unnecessary MPI_Barrier calls



IRMHD Optimization on MIRA (BG/Q)

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



NWChem

- A leading chemistry modeling code
- Relies on global arrays (GA)
  - Unified view of physically distributed arrays
  - One-sided random access
  - Use Aggregate Remote Memory Copy Interface (ARMCI)

NWChem One-sided Communication and Scaling

- What is the performance of representative workloads on different platforms?
- Understand data server / compute node interplay as a function of scaling
- Understand core allocation tradeoffs
  - All to computation vs. some to communication

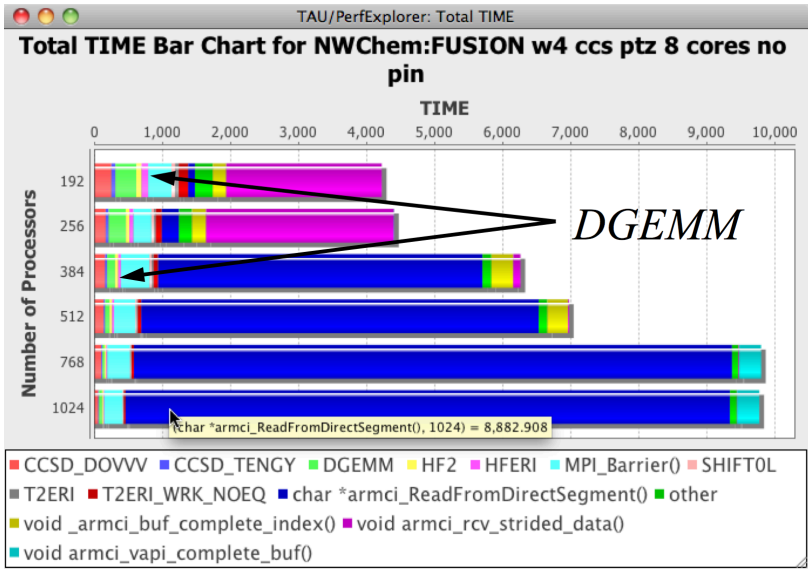
J. Hammond, S. Krishnamoorthy, S. Shende, N. Romero, A. Malony, "Performance Characterization of Global Address Space Applications: a Case Study with NWChem," *Concurrency and Computation: Practice and Experience,* 24(2), pp. 135-154, 2012.

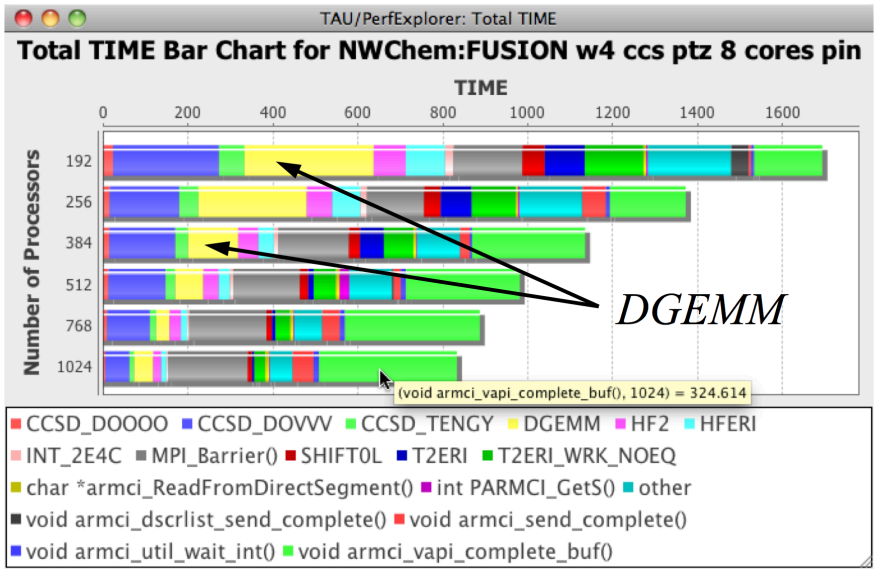
NWChem Instrumentation

- Source-based (direct) instrumentation
- ARMCI interposition library with TAU hooks (TAU PARMCI)
- Wrapped external libraries, e.g. BLAS
- Test Systems:
  - Fusion: Pacific Northwest National Lab
  - Intrepid: Argonne National Lab
- Mira and hopper will scale greater but will likely show similar effects

NWChem with Pinning on Fusion

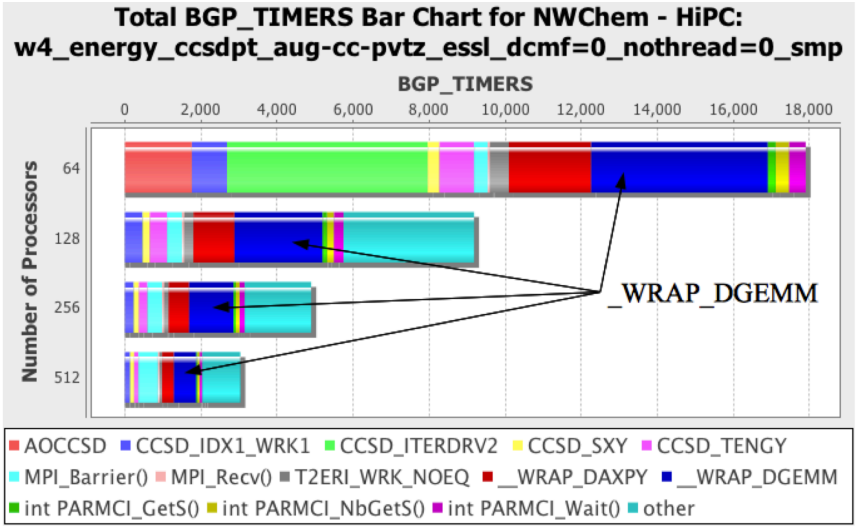
- Test on 8 cores (no separate data server thread)
- With no pinning, ARMCI communication overhead increases dramatically and no scaling is observed
- Pinning communication buffers shows dramatic effects
- Communication overhead increases, but not dramatically

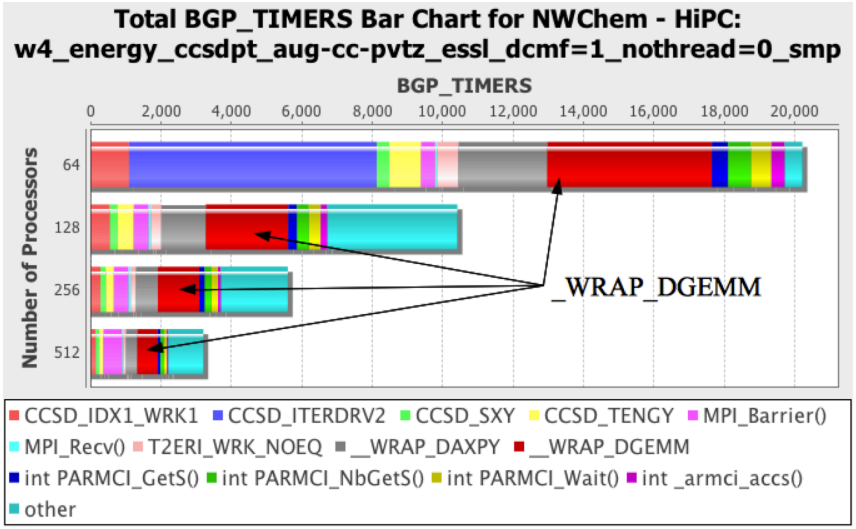




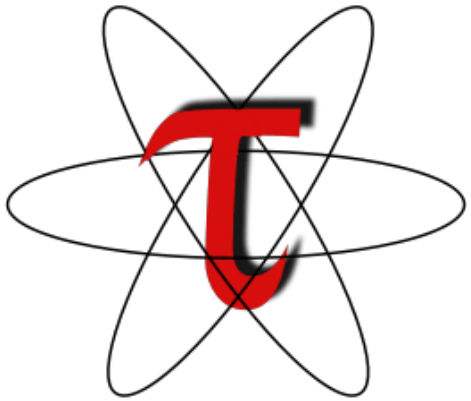
NWChem with Pinning on Intrepid

- Tests with interrupt or communication helper thread (CHT)
  - One core dedicated to CHT
- ARMCI calls are barely noticeable
- DAXPY calculation shows up more
- CHT performs better in both SMP and DUAL modes





Download TAU from U. Oregon



http://tau.uoregon.edu

http://www.hpclinux.com [LiveDVD]

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

TAU Performance System

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='-optCompInst -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_lite.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
-optCompInst	Use compiler based instrumentation
-optNoCompInst	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="<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)
-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 –optMemDbg or tau_exec)
TAU_MEMDBG_PROTECT_ABOVE	0	Setting to 1 turns on bounds checking for dynamically allocated arrays. (Use with –optMemDbg or tau_exec –memory_debug).
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 –optTrackIO or tau_exec –io 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>)