Performance Engineering with TAU and TAU Commander

John C. Linford
ParaTools, Inc.

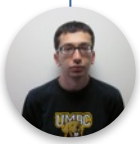
NASA Langley Research Center
25 May 2017

ParaTools


Dr. Allen Malony (CEO)


Dr. Sameer Shende (President)

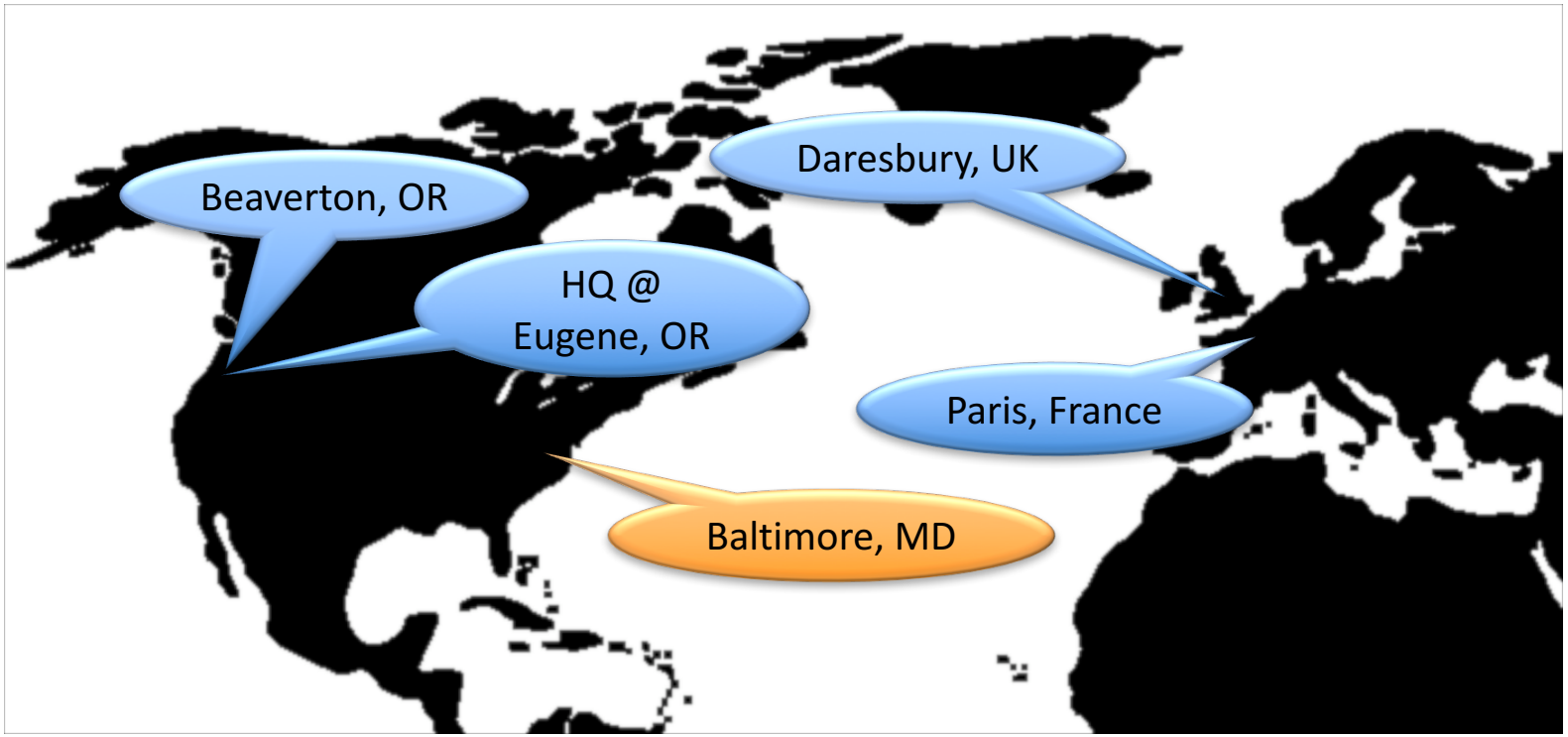

Dr. John Linford


Dr. Sam Khuvis


Mr. Izaak Beekman


Dr. David Mackay


Dr. Jean-Baptiste Besnard



FUN3D work at SC'16

Performance Engineering FUN3D at Scale with TAU Commander

John C. Linford*[1], Srinath Vadlamani[1], Sameer Shende[1], Allen D. Malony[1], William Jones[2], William Kyle Anderson[2], and Eric Nielsen[2]

[1]ParaTools, Inc. Eugene, OR 97405, U.S.A.
[2]NASA Langley Research Center, Hampton, VA 23681, U.S.A.

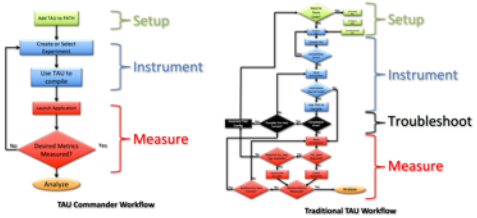
The FUN3D Solver

NASA Langley Research Center's FUN3D software is an unstructured-grid computational fluid dynamics suite used to tackle complex aerodynamics problems. The toolset enables multidisciplinary capabilities through coupling to variable fidelity models encompassing structural effects, multi-body dynamics, acoustics, radiation, optics, propulsion, and ablation. FUN3D provides the world's foremost adjoint-based design capability, enabling formal optimization of time-dependent moving-body simulations involving turbulent flows. The adjoint formulation is also used to perform mathematically-rigorous mesh adaptation and error estimation.

FUN3D is widely used to support major national research and engineering efforts at NASA and among groups across U.S. industry, other government agencies, and academia. A past collaboration with the Department of Energy received the prestigious Gordon Bell Prize, which recognizes outstanding achievements in high-performance computing.

TAU Commander

TAU Commander simplifies the TAU Performance System® by using a structured workflow approach that gives context to a TAU user's actions. This eliminates the troubleshooting step inherent in the traditional TAU workflow and avoids invalid TAU configurations.



TAU Commander Workflow — Traditional TAU Workflow

A study of 124 workflows demonstrated that using TAU Commander reduces the number of unique steps in the performance workflow by approximately 50% and reduces the number of commands a user must know from approximately eight to exactly one. TAU Commander is installed on several DoD DSRC systems.

Improve productivity by specifying the desired result, not step-by-step directions.

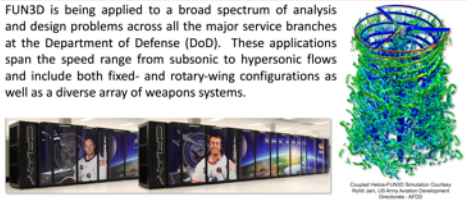
Acknowledgments

This work was developed in part by the User Productivity Enhancement, Technology Transfer and Training (PETTT) Project No. PETA-KY07-001 and by DoE SBIR Grant No. DE-SC0009593. The authors would like to acknowledge the computational resources and PETTT software support from the DoD High Performance Computing Modernization office under Contract No. GS04T09DBC0017.

* Corresponding author: jlinford@paratools.com

FUN3D at the Department of Defense

FUN3D is being applied to a broad spectrum of analysis and design problems across all the major service branches at the Department of Defense (DoD). These applications span the speed range from subsonic to hypersonic flows and include both fixed- and rotary-wing configurations as well as a diverse array of weapons systems.

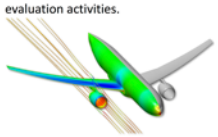


Coupled Helios-FUN3D Simulation Courtesy Rohit Jain, US Army Aviation Development Directorate - AFSD

FUN3D is in routine use across the various DoD Supercomputing Resource Centers (DSRCs). To accommodate an ever-increasing demand for larger and more complex simulations, the FUN3D development team is partnered with computational experts from ParaTools, Inc. through the DoD/Engility Productivity Enhancement, Technology Transfer and Training Project (PETTT). Through this collaboration, the team is effectively identifying and addressing computational barriers encountered at scale.

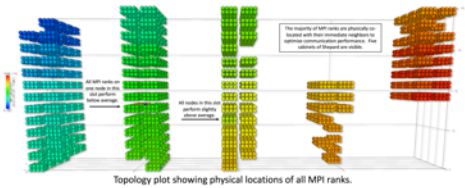
Project Goal

The team is using TAU commander and related tools on the DoD DSRC systems to study FUN3D computational performance and to guide optimization efforts. TAU Commander highlights source code regions that limit scalability through profiling, tracing, and aggregate summary statistics with respect to computational time, memory allocation, and memory access patterns. The analysis approach is being carefully documented to assist other DoD groups in similar performance evaluation activities.



- 1,651,089,924 grid points
- 5,902,801,476 tetrahedral elements
- 1,310,290,264 prismatic elements
- 14,400 Ivy Bridge cores

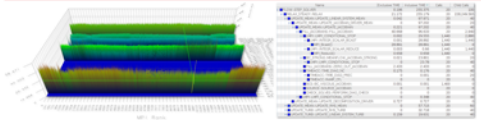
To establish a baseline database of FUN3D performance metrics, the team used TAU Commander to profile a high Reynolds number simulation of the flow over a wing-body-pylon-nacelle geometry. The computation was performed using 600 nodes (14,400 Intel® Xeon® Ivy Bridge cores) on *Shepard*, a Cray XC30 system located at the Navy DSRC.



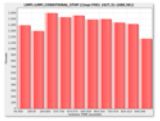
Topology plot showing physical locations of all MPI ranks.

DISTRIBUTION STATEMENT A. Approved for public release; distribution is unlimited.

Performance Analysis

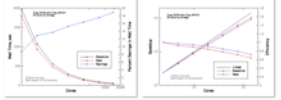


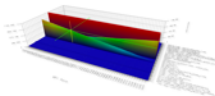
Load imbalance in `FILL_JACOBIAN` is easily visible by comparing against the lowest time in this routine and viewing the function histogram. On average, the routine took 96.6 seconds, of which 28.8 seconds are spent in `MPI_Bcast`. The broadcast operation takes place in `LMPI_CONDITIONAL_STOP`, which has dramatically different times across all MPI ranks.



Analyzing data from the 14.4k core run revealed a communication bottleneck in `LMPI_CONDITIONAL_STOP`. Rank 0 performs a data reduction on behalf of all other ranks, so all other ranks remain idle until Rank 0 finishes the calculation. The severity of the problem grew linearly in the number of MPI ranks.

The team implemented a new approach that did not require the data reduction operation. **This improvement reduced runtime by up to 33% in core solver routines and is included in the FUN3D 13.0 release.**





When writing the solution file to disk, there is a strong correlation between MPI rank and time spent in `MPI_Bcast` and `MPI_Send`. This operation occurs only once every few hours in production. **Further optimization should focus on load balance in the PDE solution.**

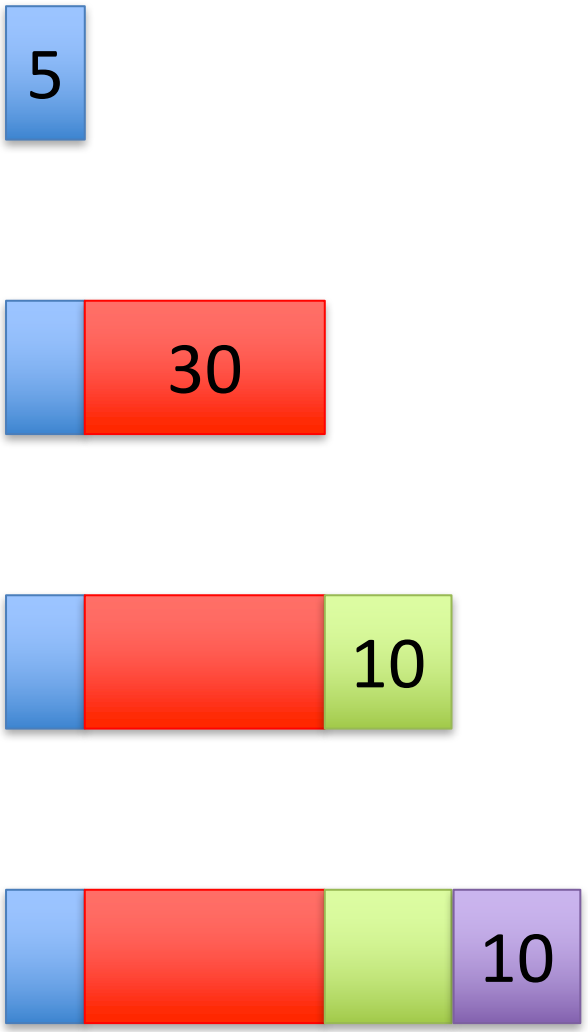
`MPI_Bcast` is called the same number of times with the same message size for all ranks, yet time spent in `MPI_Bcast` varies dramatically by rank. Rank 0 receives data from all others and writes it to disk, hence N-1 ranks spend the majority of their time in this operation waiting for I/O on Rank 0. This suggests implementing parallel I/O and distributed checkpoints.

Additional information, visualizations, and slides.

ParaTools ENGILITY DoD HPC

Overview

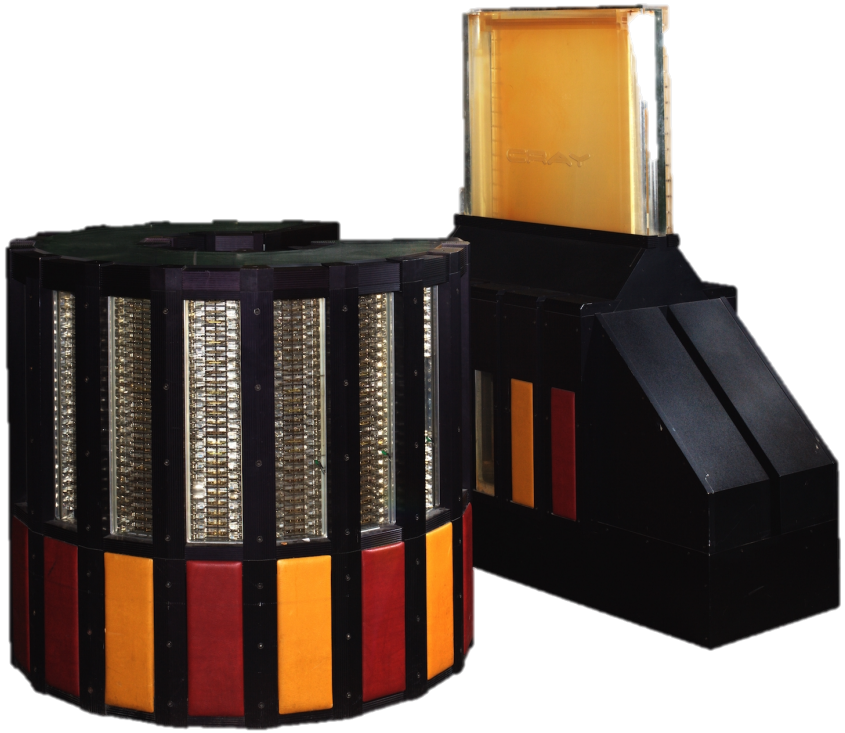
- Why Tools?
- TAU Overview
- TAU Commander
- Case Studies



Intuitive Performance Engineering

MOTIVATION

1985: Cray-2

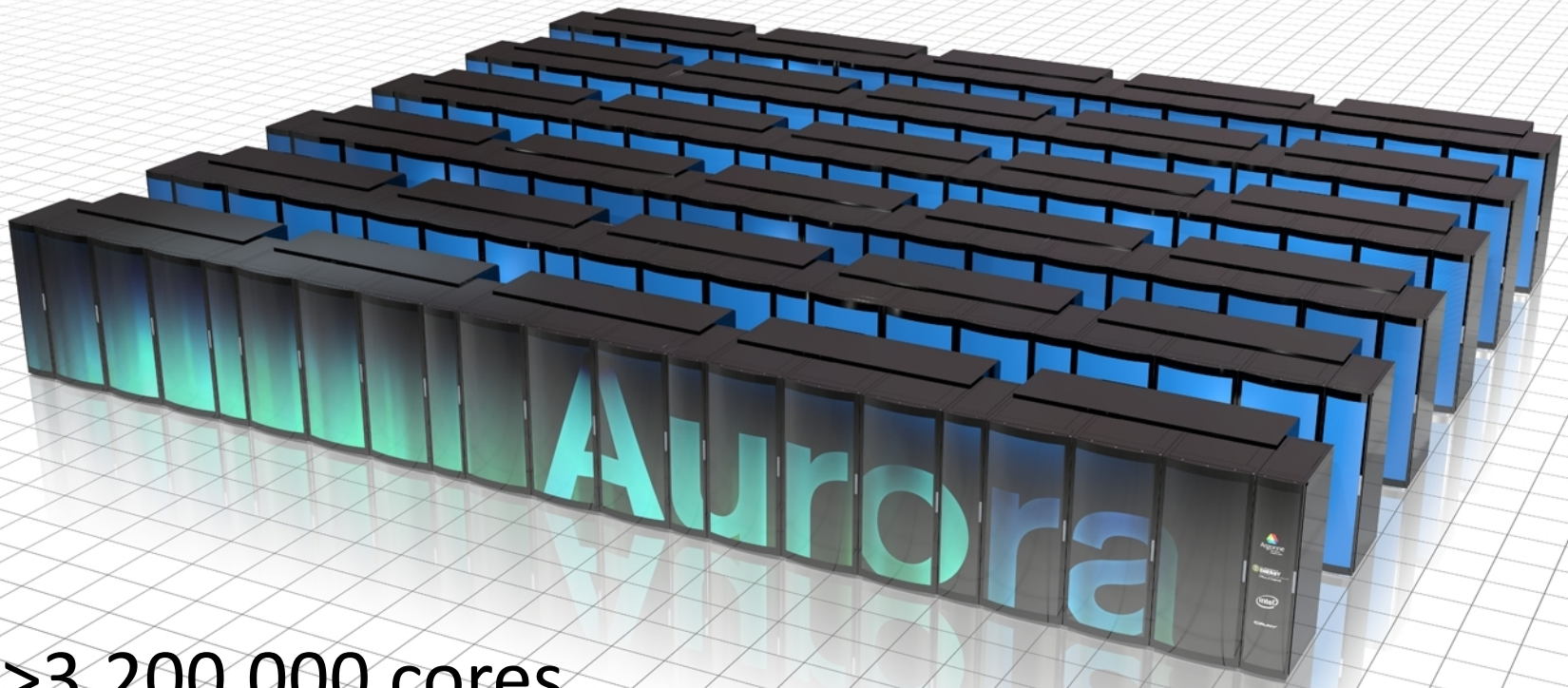


- 4 vector processors
- 1.9 gigaflops (0.0095 gigaflops/Watt)
- 237.5 million abacuses

2013: IBM “Mira” BlueGene/Q

- 784,432 cores
- 10 petaflops (2.08 gigaflops/Watt)
- 1.25 quadrillion abacuses

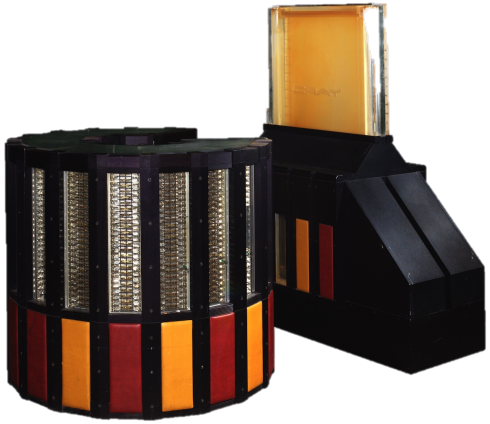
2019: Aurora

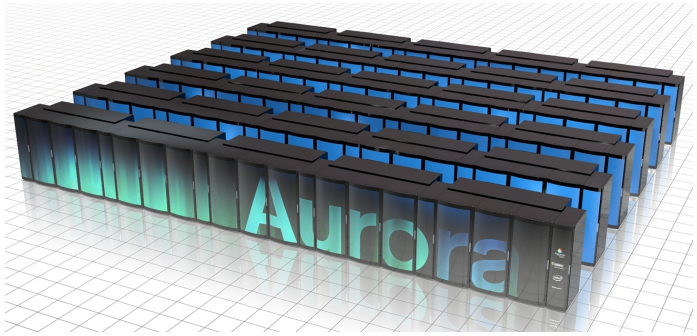


- >3,200,000 cores
- 180 petaflops (5.62 gigaflops/Watt)
- 22.5 quadrillion abacuses

Your brain is not enough

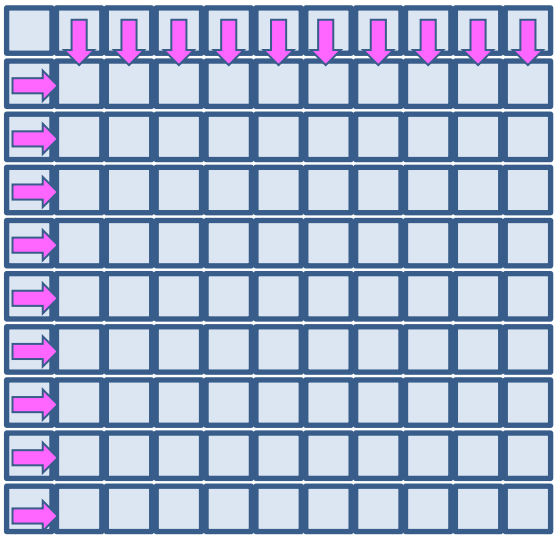
- Supercomputers are incomprehensibly complex.
- Naïve optimization may harm performance.
- **Performance engineering tools are essential for realizing performance at scale.**



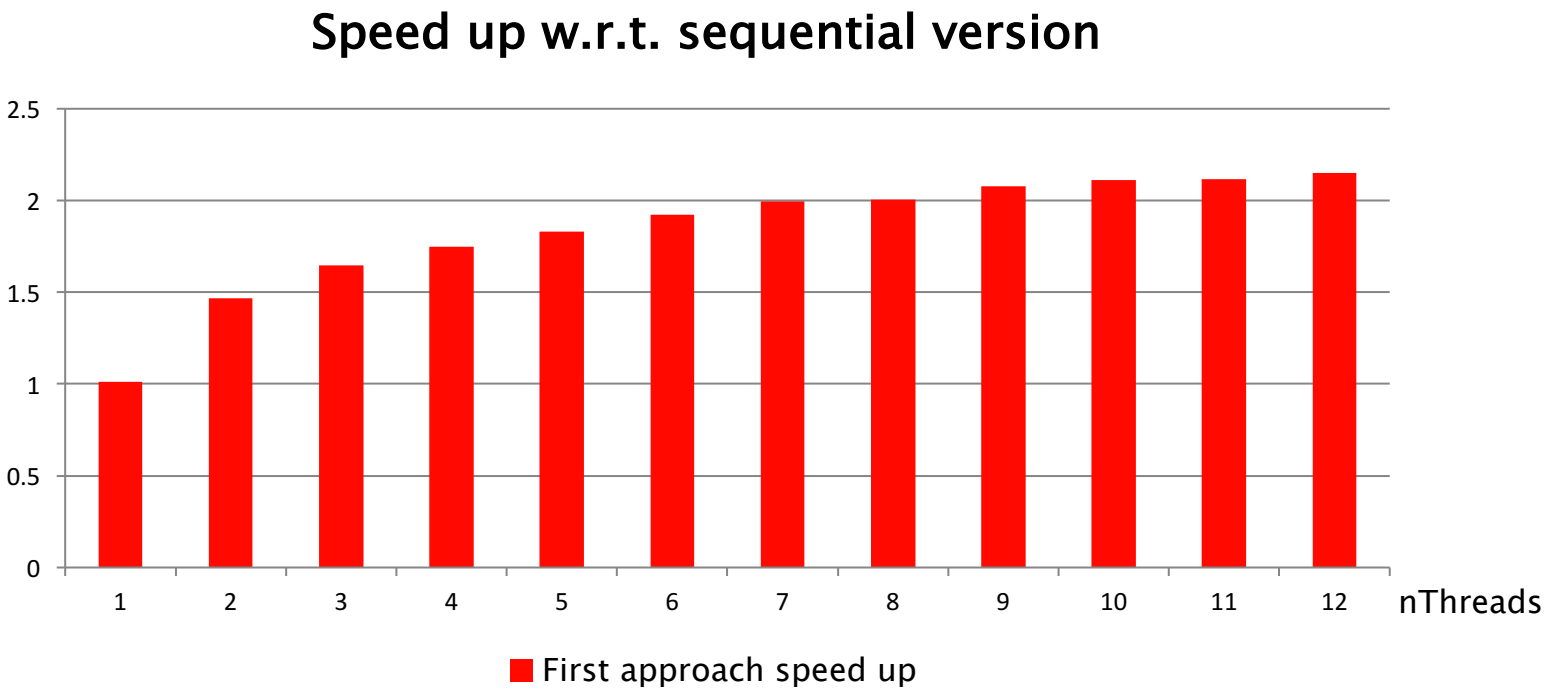


Naïve GE Optimization

```
for i=1 to n-1
    find pivotPos in column i
    if pivotPos ≠ i
        exchange rows(pivotPos,i)
    end if

    for j=i+1 to n
        A(i,j) = A(i,j)/A(i,i)
    end for j
!$omp parallel do private ( i ,j )
    for j=i+1 to n+1
        for k=i+1 to n
            A(k,j)=A(k,j)-A(k,i)×A(i,j)
        end for k
    end for j
end for i
```



First approach speed up

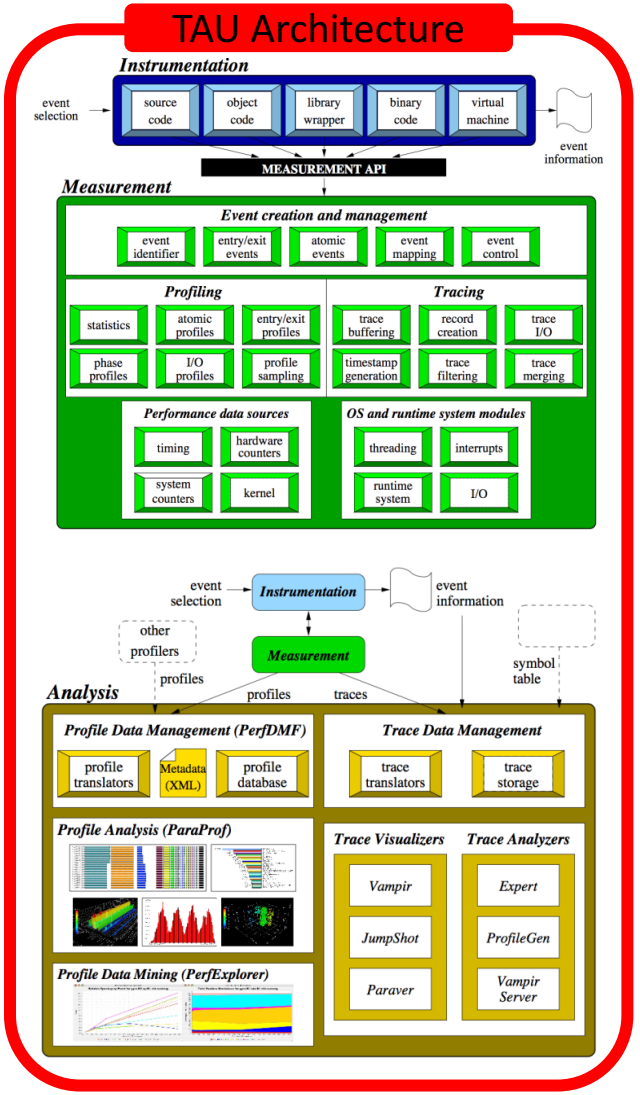


Intuitive Performance Engineering

THE TAU PERFORMANCE SYSTEM

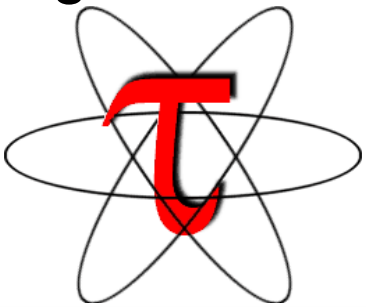
The TAU Performance System®

- *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

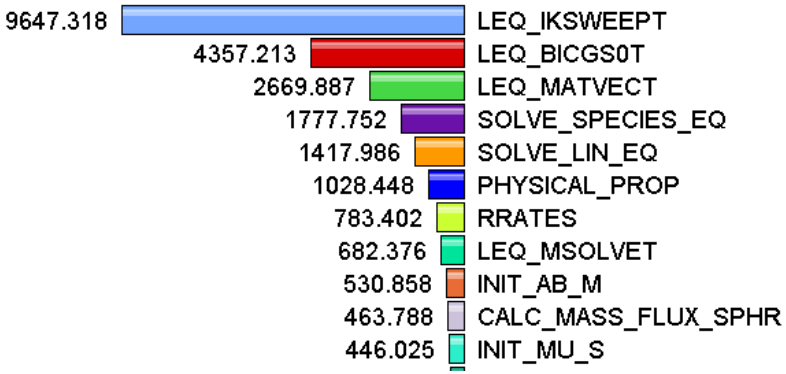


TAU Supports All HPC Platforms

C/C++
CUDA
UPC
Python
GPI
Fortran
OpenACC
Java
MPI
pthreads
Intel MIC
OpenMP
Intel
GNU
LLVM
PGI
Cray
Sun
MinGW
AIX
Linux
Windows
Insert yours here
BlueGene
Fujitsu
ARM
Android
MPC
OS X

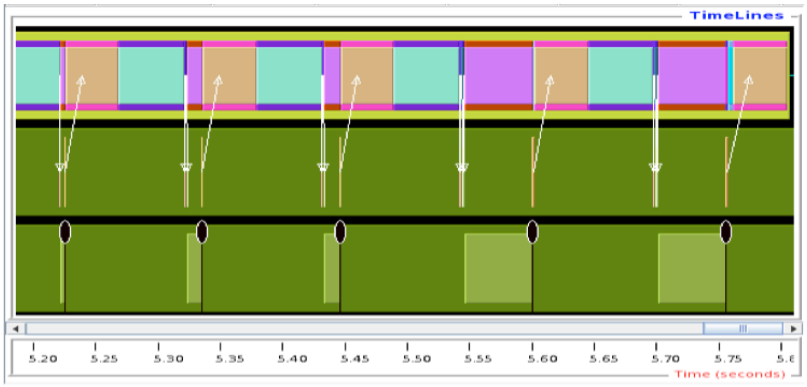
Measurement Approaches

Profiling



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

Tracing

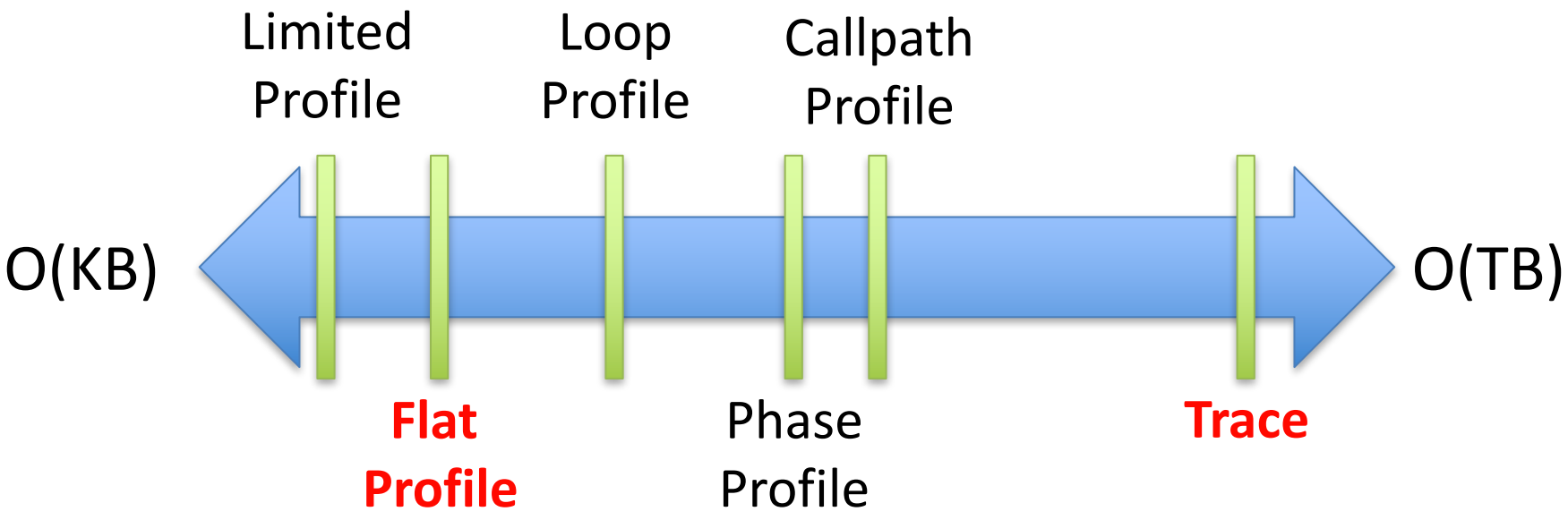


Shows **when** events take place on a timeline

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

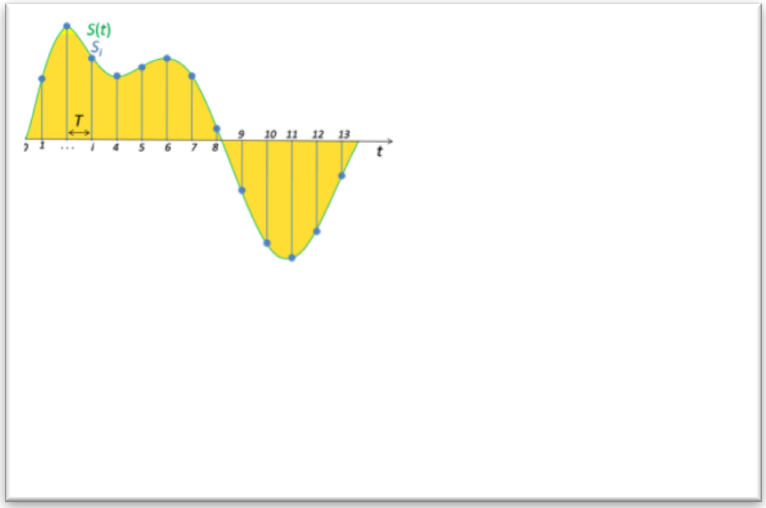
Performance Data Measurement

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
- 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

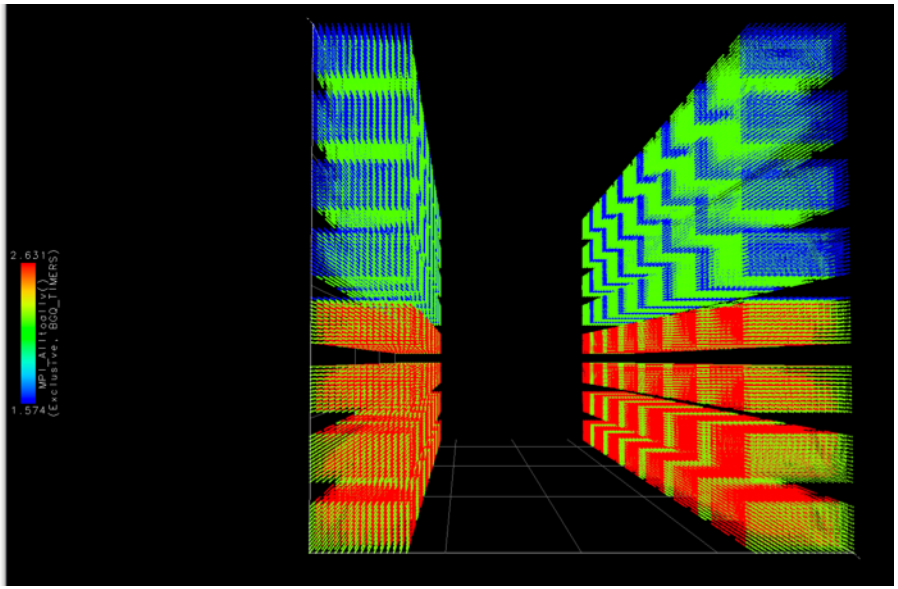
- *Source code* instrumentation
  - PDT parsers, pre-processors
- *Wrap* external libraries
  - I/O, MPI, Memory, CUDA, OpenCL, pthread
- *Rewrite* the binary executable
  - Dyninst, MAQAO

Measure: Gather Data

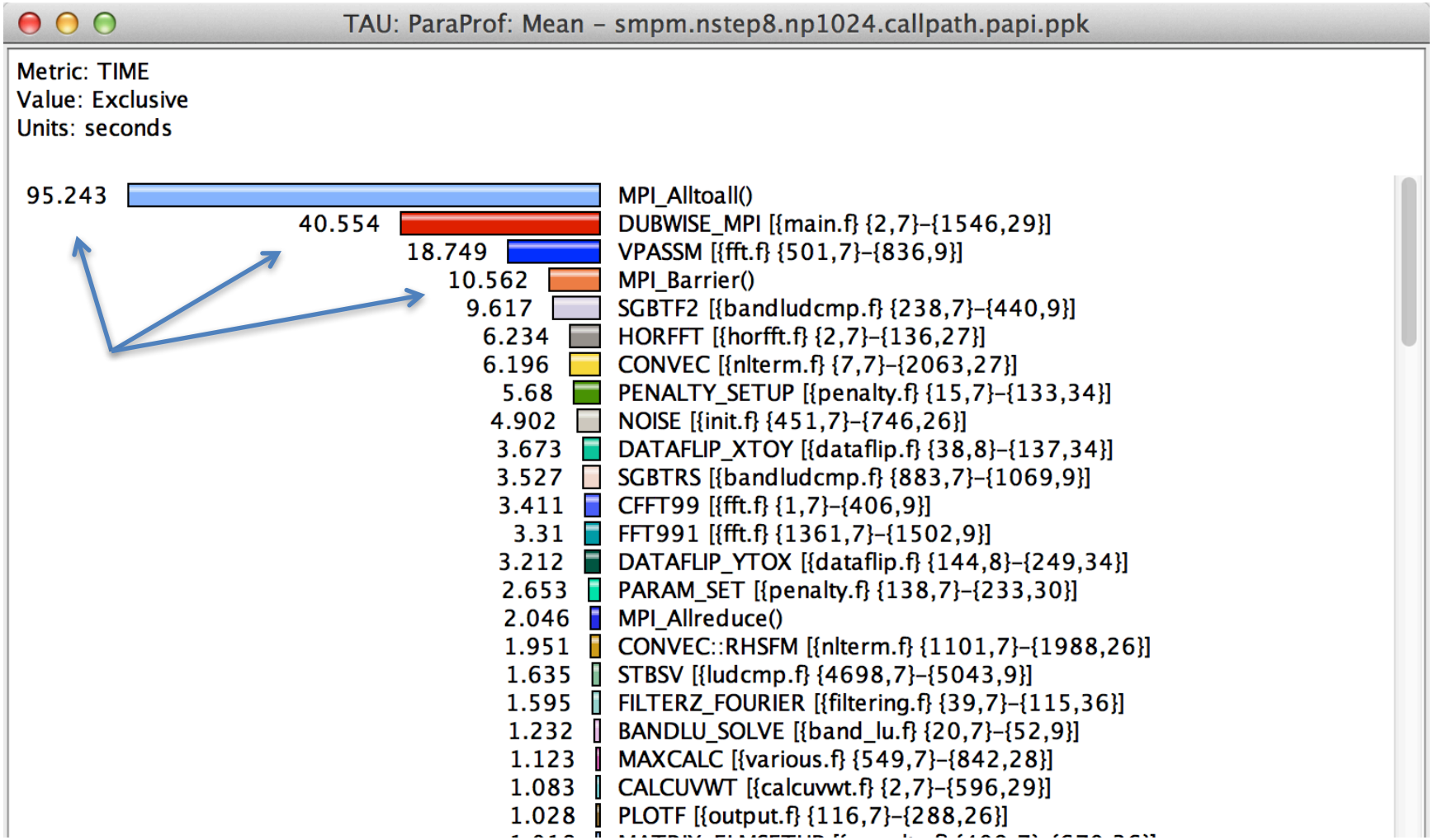
- Direct measurement via *probes*
- Indirect measurement via *sampling*
- Throttling and runtime control
- Interface with external packages (PAPI)

Analyze: Synthesize Knowledge

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

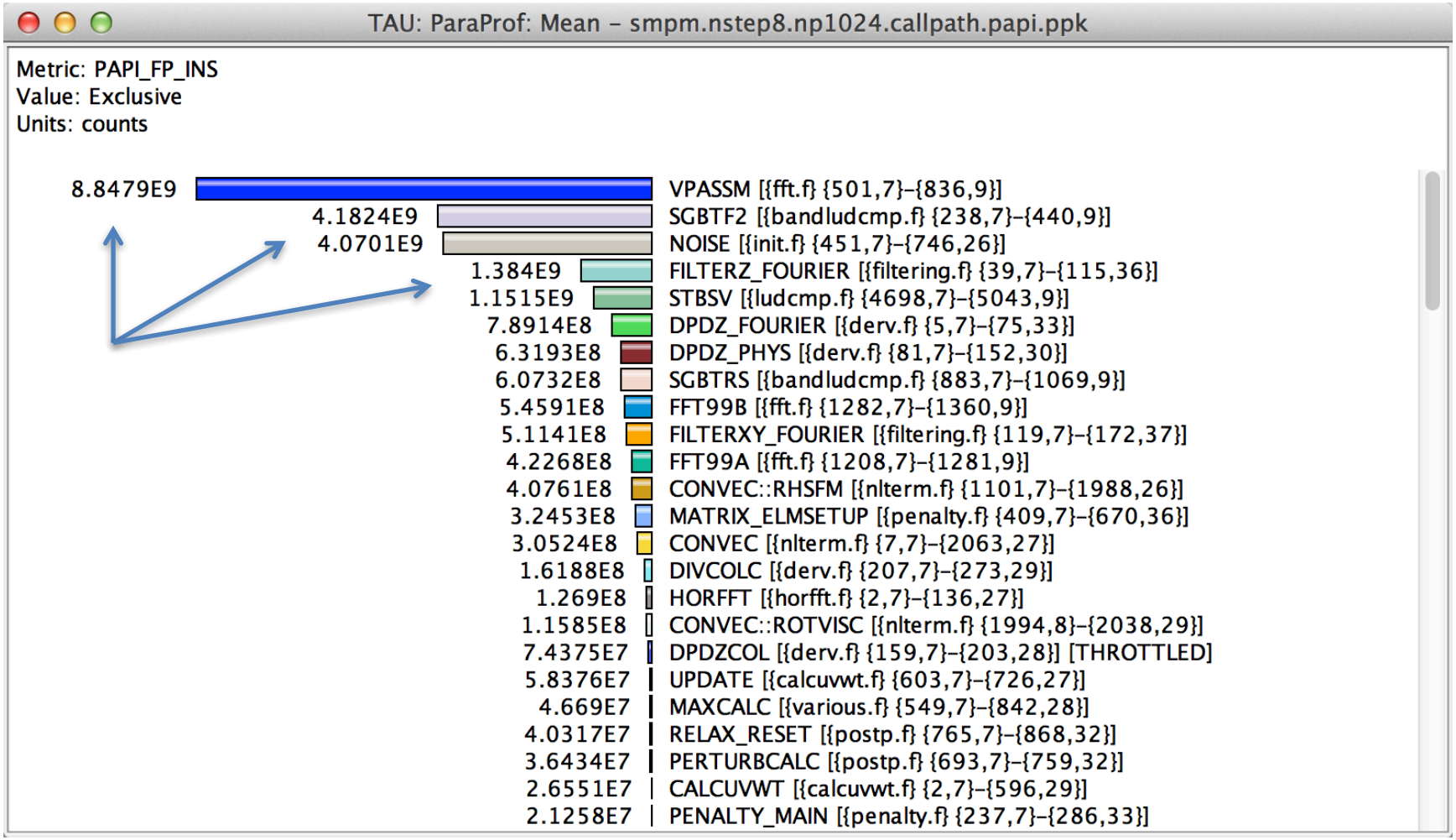


How Much Time per Code Region?



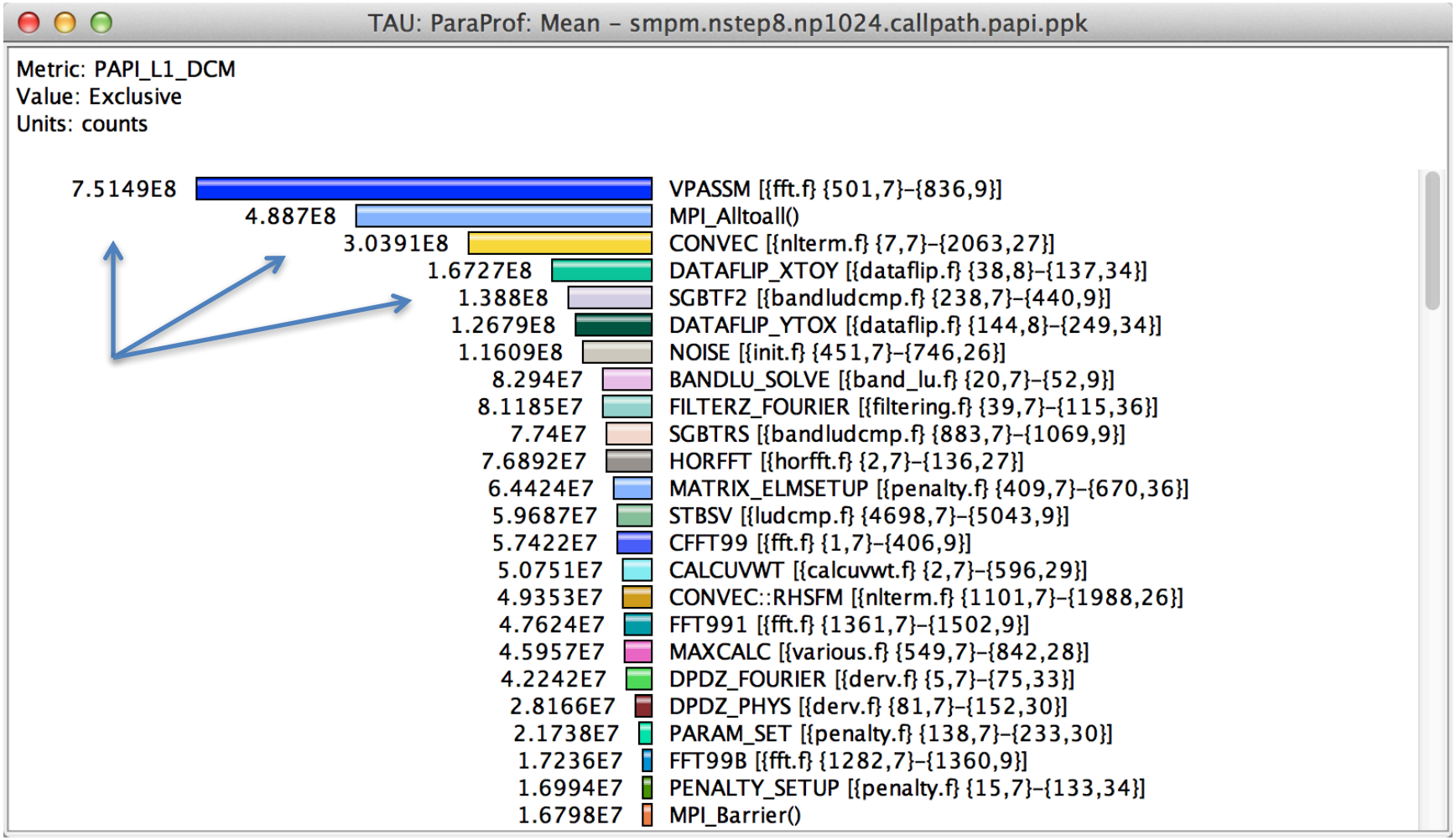
`% paraprof` **(Click on label, e.g. “Mean” or “node 0”)**

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?

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?

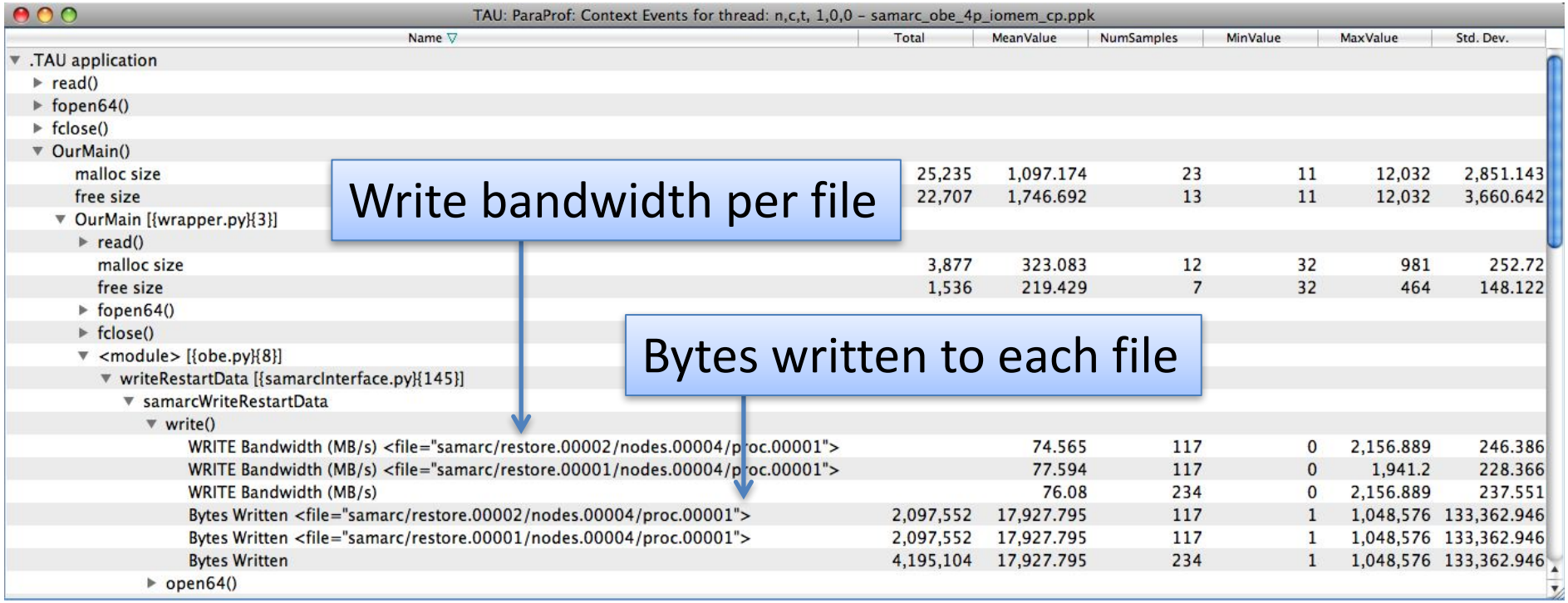
TAU: ParaProf: Mean Context Events – sphere_np32_nsteps5_mem.ppk

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?

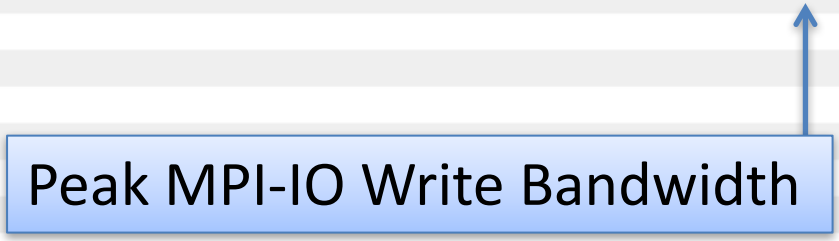


Name	Total	MeanValue	NumSamples	MinValue	MaxValue	Std. Dev.
.TAU application						
read()						
fopen64()						
fclose()						
OurMain()						
malloc size	25,235	1,097.174	23	11	12,032	2,851.143
free size	22,707	1,746.692	13	11	12,032	3,660.642
OurMain [{wrapper.py}{3}]						
read()						
malloc size	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()						

% `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()						



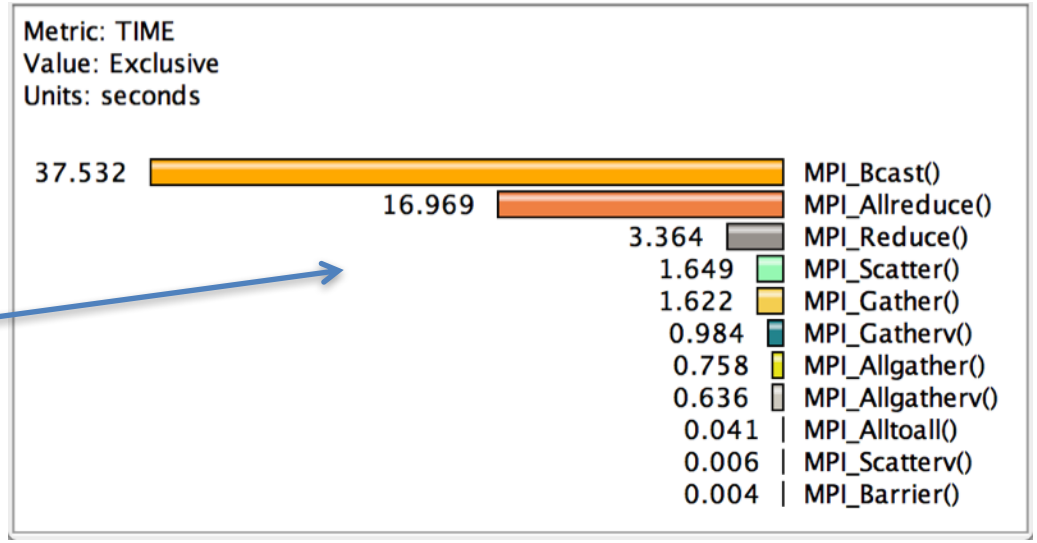
% `paraprof` (Right-click label [e.g "node 0"] → Show Context Event Window)

How Much Time is spent in Collectives?

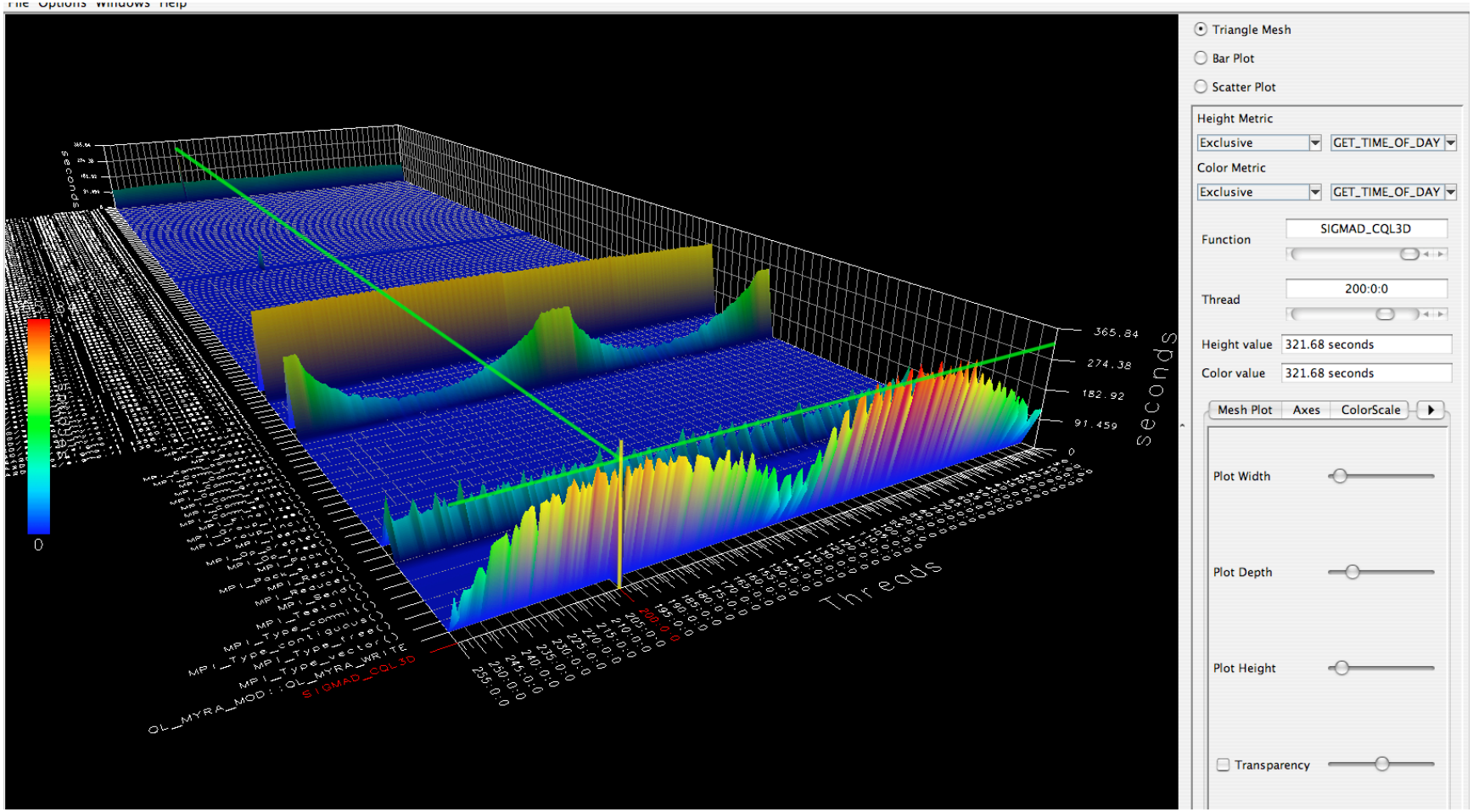
Name △	Total	Num...	MaxValue	MinValue	MeanValue	Std. Dev.
▶ MPI_Wait()						
▶ MPI_Waitall()						
Message size for all-gather	305,753,268	72	172,215,296	4	4,246,573.167	22,551,605.859
Message size for all-reduce	163,308	632	21,908	4	258.399	897.725
Message size for all-to-all	112	14	8	8	8	0
Message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

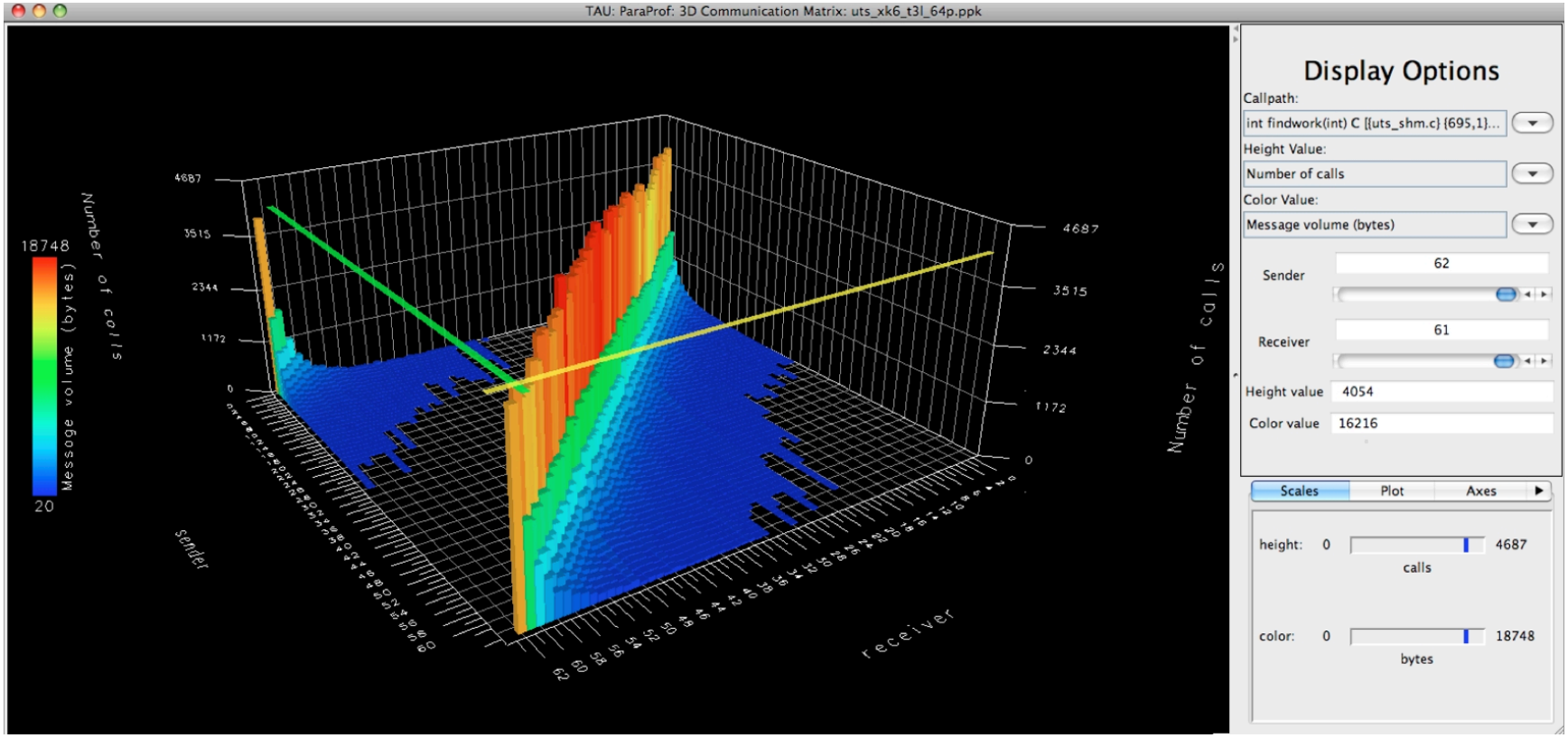


3D Profile Visualization



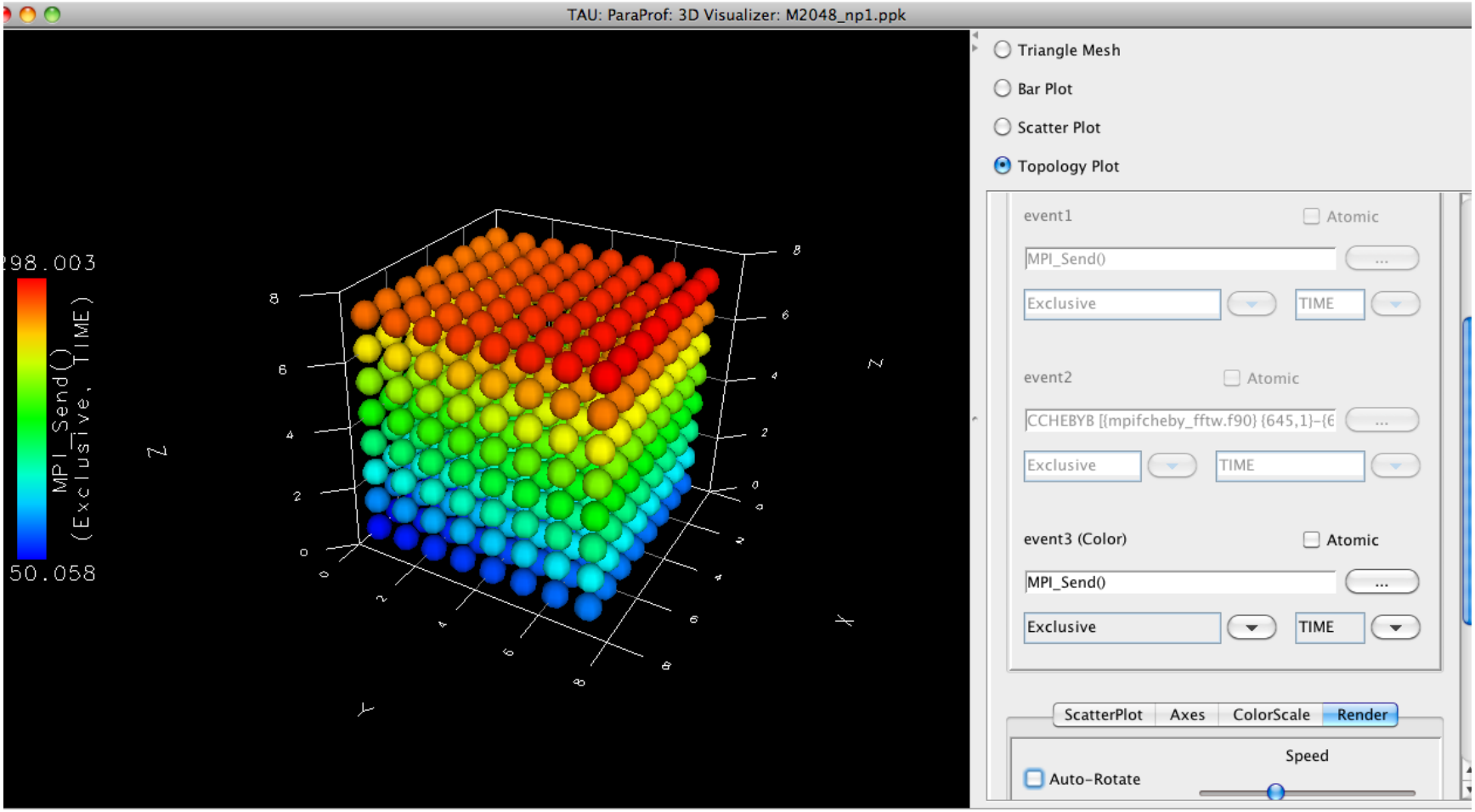
% paraprof (Windows → 3D Visualization)

3D Communication Visualization



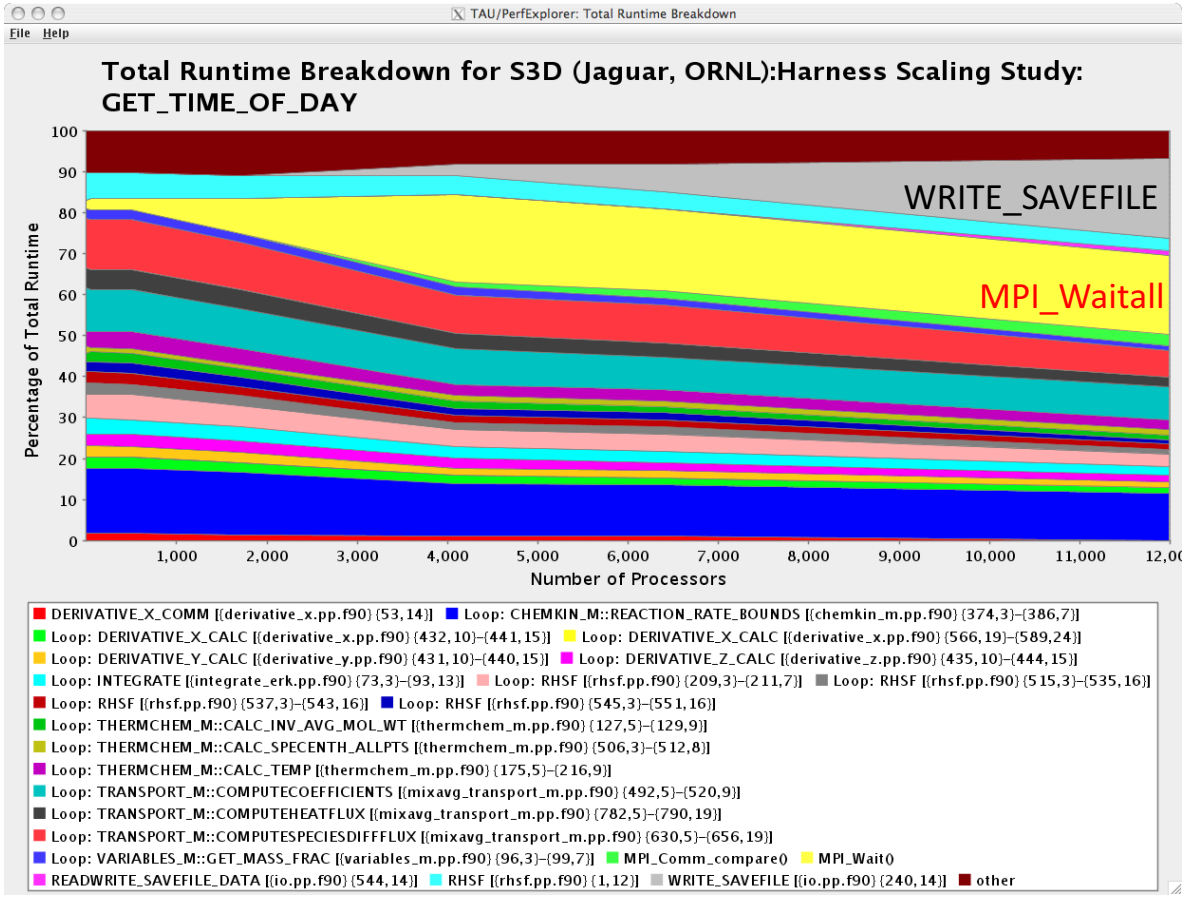

```
% qsub –env TAU_COMM_MATRIX=1 …
% paraprof  (Windows → 3D Communication Matrix)
```

3D Topology Visualization



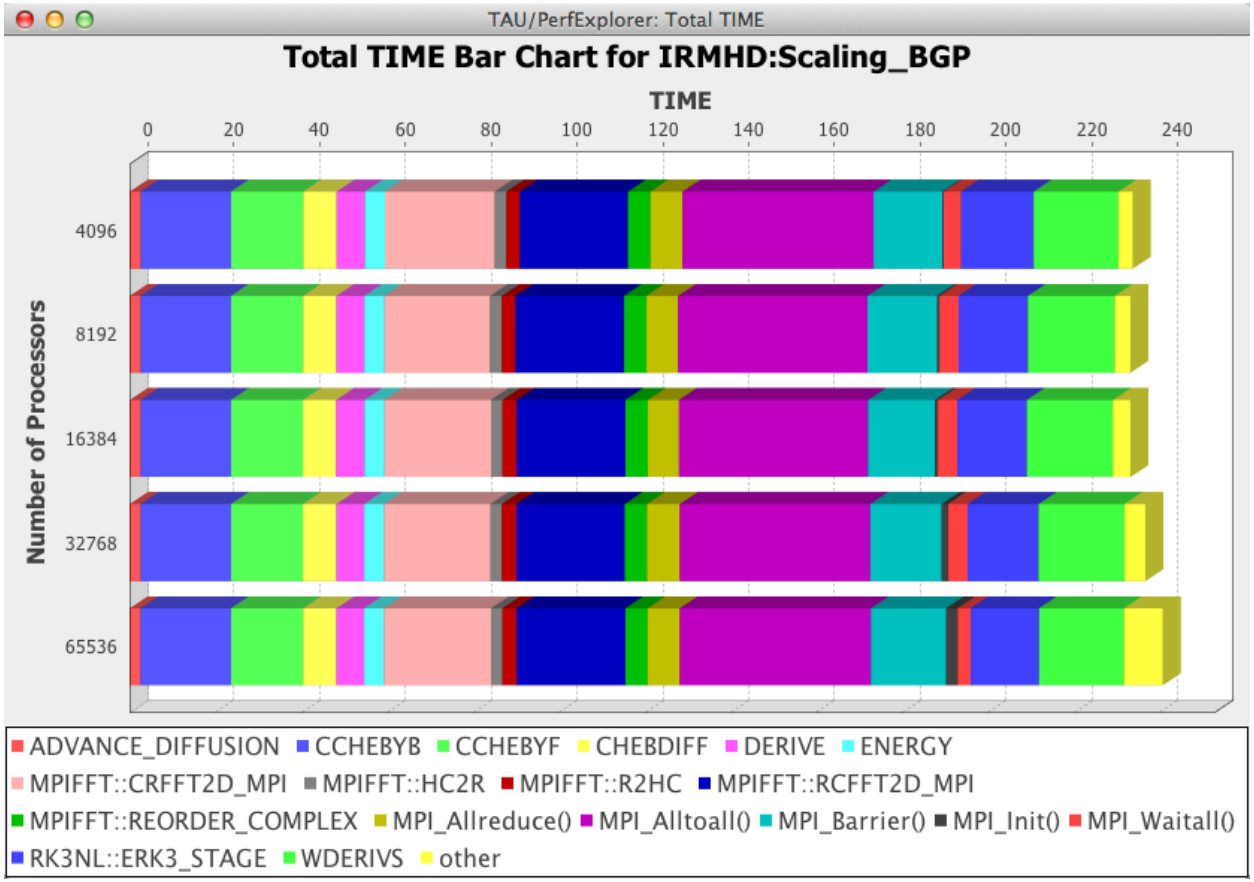
% paraprof (Windows → 3D Visualization → Topology Plot)

How Does Each Routine Scale?



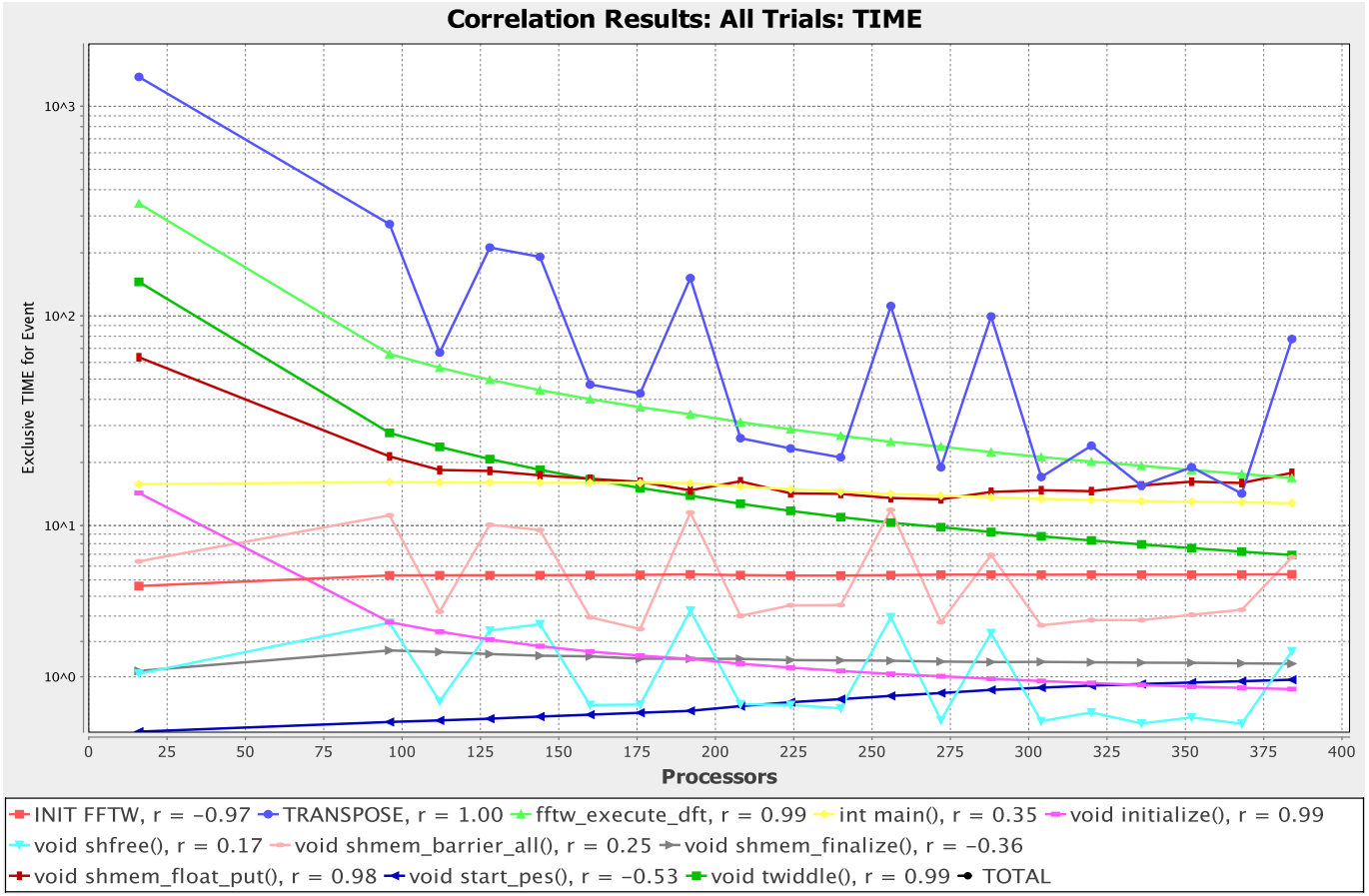
`% perfexplorer` **(Charts → Runtime Breakdown)**

How Does Each Routine Scale?



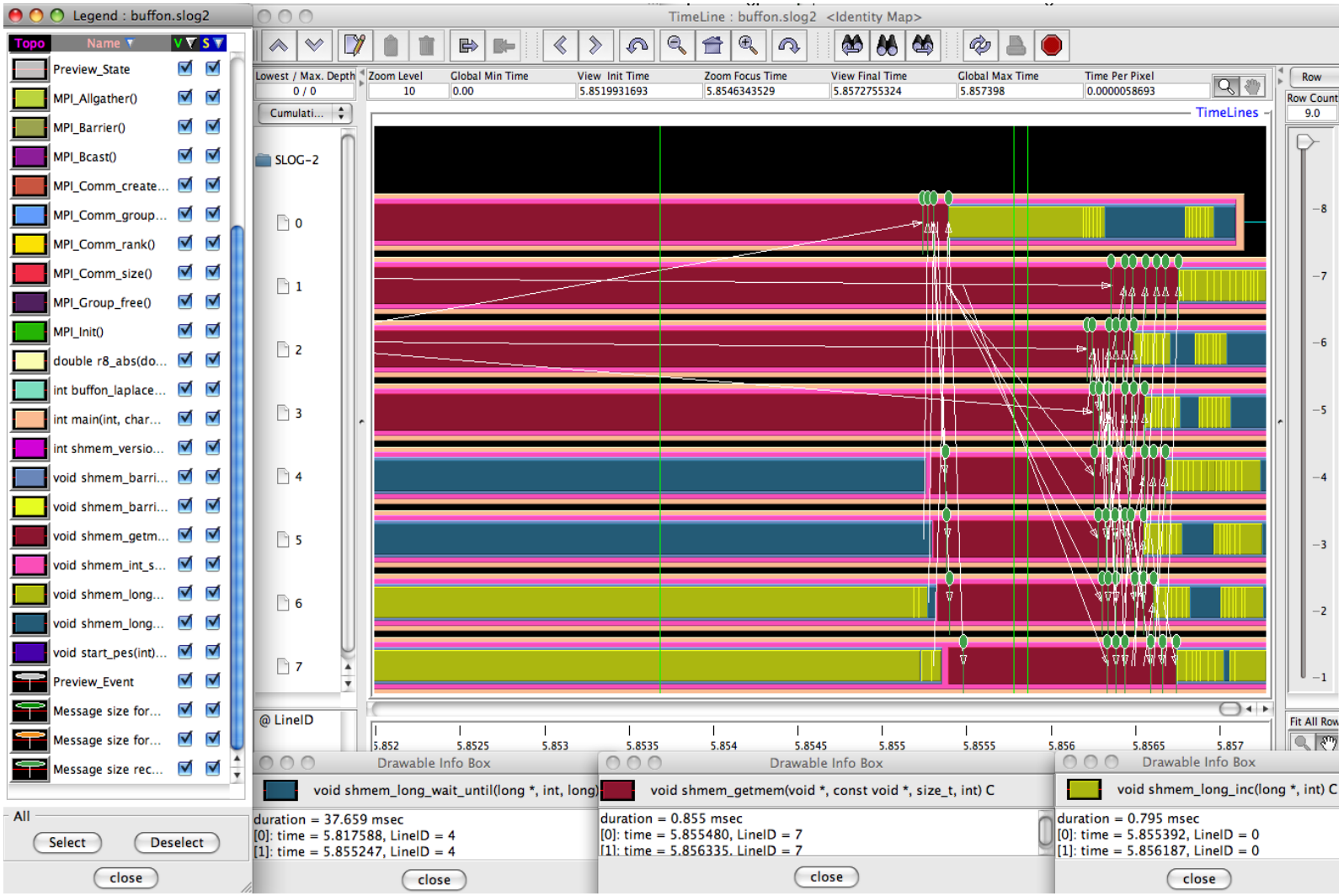
`% perfexplorer` **(Charts → Stacked Bar Chart)**

Which Events Correlate with Runtime?

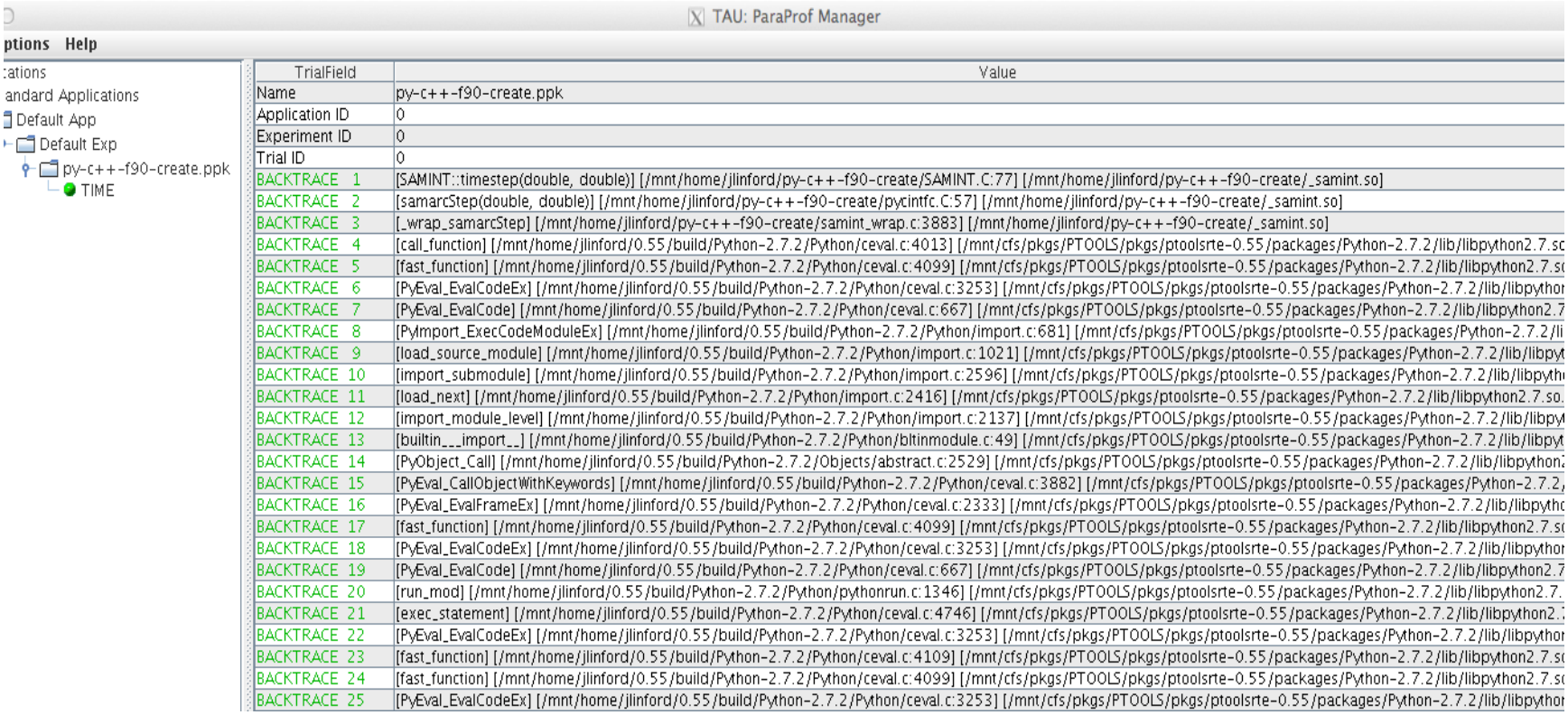


`% perfexplorer` **(Charts → Correlate Events with Total Runtime)**

When do Events Occur?



What Caused My Application to Crash?

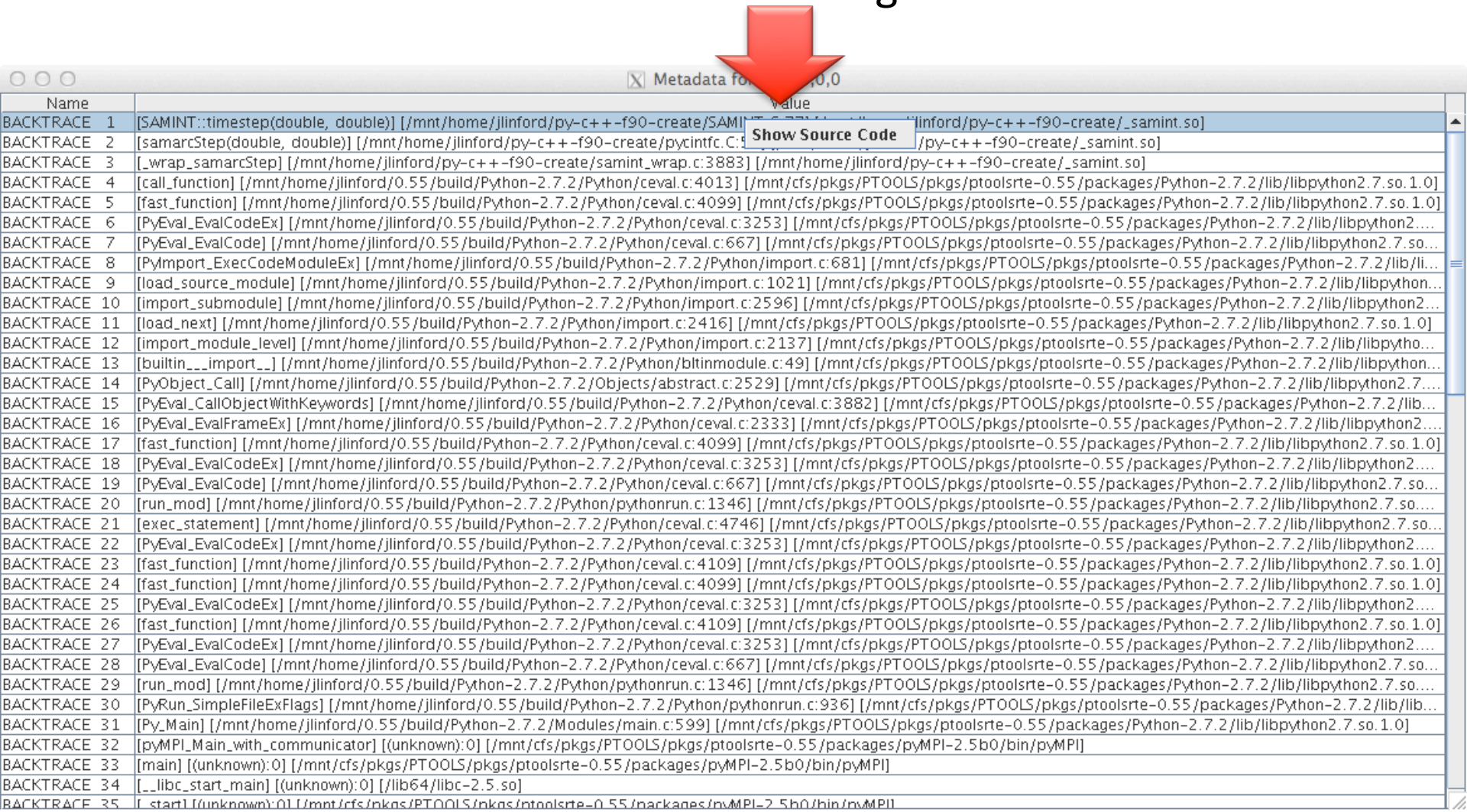


TrialField	Value
Name	py-c++-f90-create.ppk
Application ID	0
Experiment ID	0
Trial ID	0
BACKTRACE 1	[SAMINT::timestep(double, double)] [/mnt/home/jlinford/py-c++-f90-create/SAMINT.C:77] [/mnt/home/jlinford/py-c++-f90-create/_samint.so]
BACKTRACE 2	[samarcStep(double, double)] [/mnt/home/jlinford/py-c++-f90-create/pycintfc.C:57] [/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
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
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/libpython
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
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/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/libpyt
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/libpyth
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.
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/libpyt
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/libpyt
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
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/
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/libpytho
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
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/libpython
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
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.
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
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/libpython
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
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
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/libpython

```
% qsub –env TAU_TRACK_SIGNALS=1 …
% paraprof
```

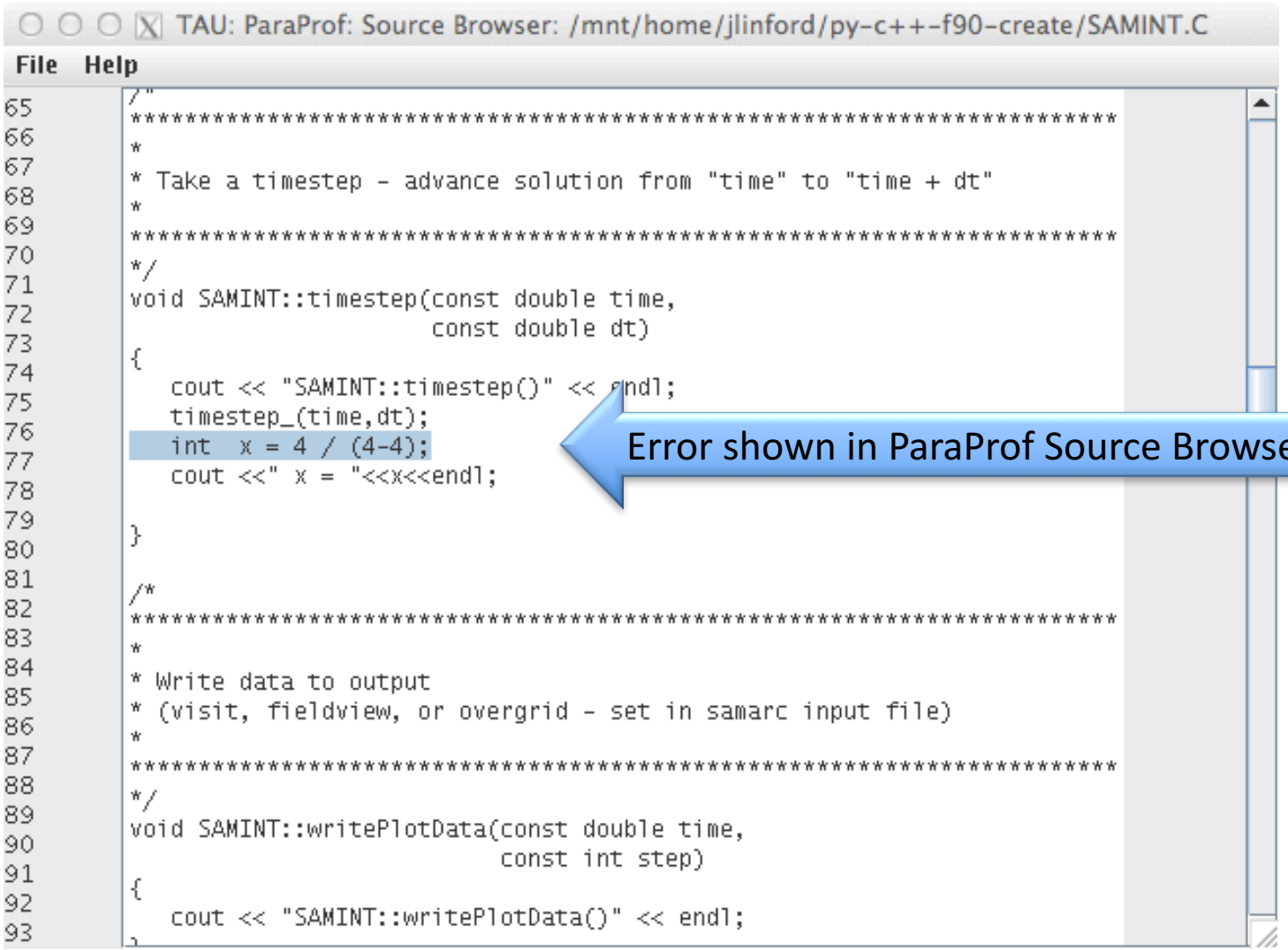
What Caused My Application to Crash?

Right-click to see source code



Name	Value
BACKTRACE 1	[SAMINT::timestep(double, double)] [/mnt/home/jlinford/py-c++-f90-create/SAM
BACKTRACE 2	[samarcStep(double, double)] [/mnt/home/jlinford/py-c++-f90-create/pycintfc.C:
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?

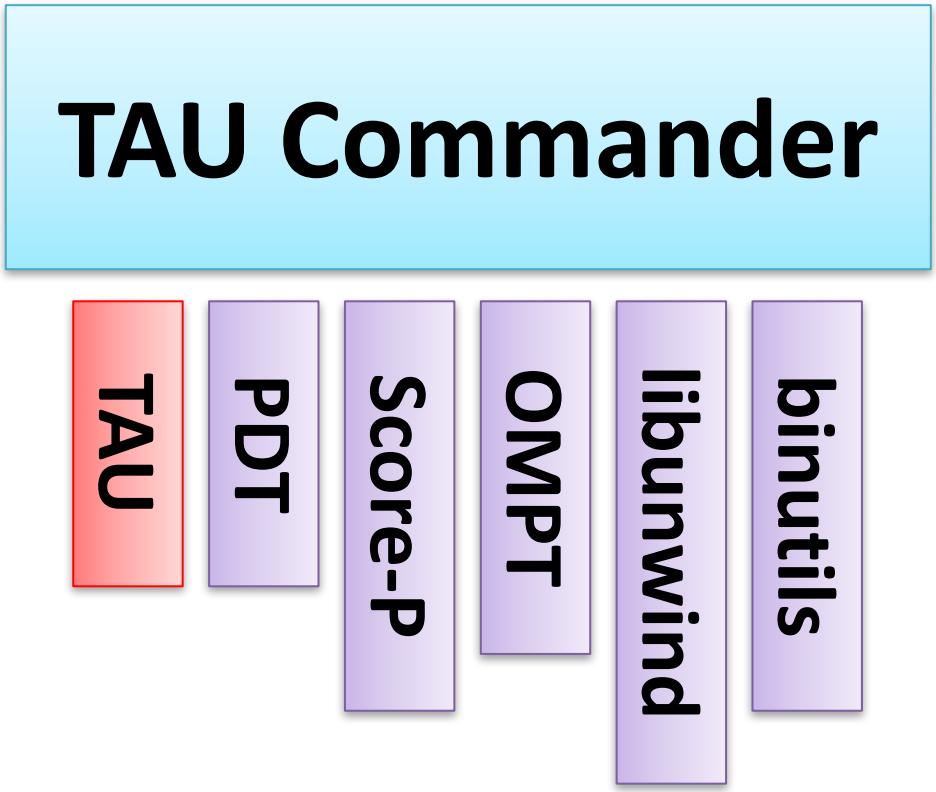


Intuitive Performance Engineering

TAU COMMANDER

TAU Commander

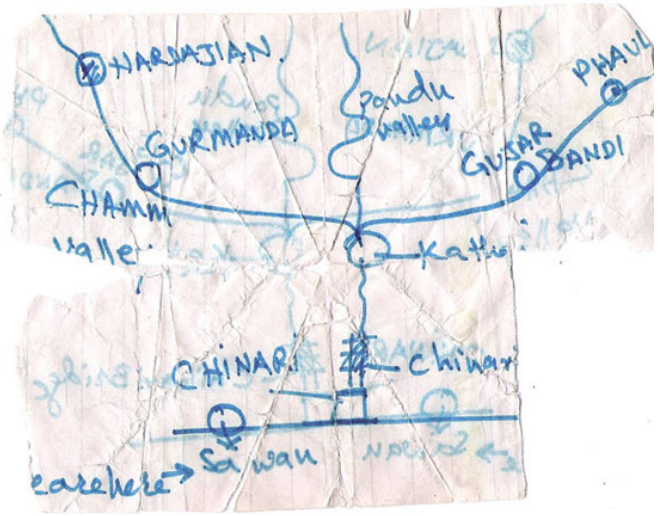
- Workflow manager and user interface for TAU.
- Automatically installs and (re)configures TAU and dependencies.
- Prevents invalid TAU configurations.
- Maintains a performance narrative to facilitate data provenance.



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 — 154×53

```
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   | Compiler  |    MPI    |  OpenMP   | Callpath  |   Mem.    |   Mem.    |    In     |
|           |           |           |           |   Inst.   |   Inst.   |           |           |   Depth   |   Usage   |   Alloc   | Projects  |
+===========+===========+===========+===========+===========+===========+===========+===========+===========+===========+===========+===========+
|       ex- |    Yes    |    No     |    No     | automatic | fallback  |    No     | compiler_ |     0     |    No     |    No     | ex-mm     |
|   profile |           |           |           |           |           |           |  default  |           |           |           |           |
+-----------+-----------+-----------+-----------+-----------+-----------+-----------+-----------+-----------+-----------+-----------+-----------+
|  ex-trace |    No     |    Yes    |    No     | automatic | fallback  |    No     | compiler_ |     0     |    No     |    No     | ex-mm     |
|           |           |           |           |           |           |           |  default  |           |           |           |           |
+-----------+-----------+-----------+-----------+-----------+-----------+-----------+-----------+-----------+-----------+-----------+-----------+
| ex-sample |    Yes    |    No     |    Yes    |   never   |   never   |    No     | compiler_ |     0     |    No     |    No     | ex-mm     |
|           |           |           |           |           |           |           |  default  |           |           |           |           |
+-----------+-----------+-----------+-----------+-----------+-----------+-----------+-----------+-----------+-----------+-----------+-----------+

== 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 $
```

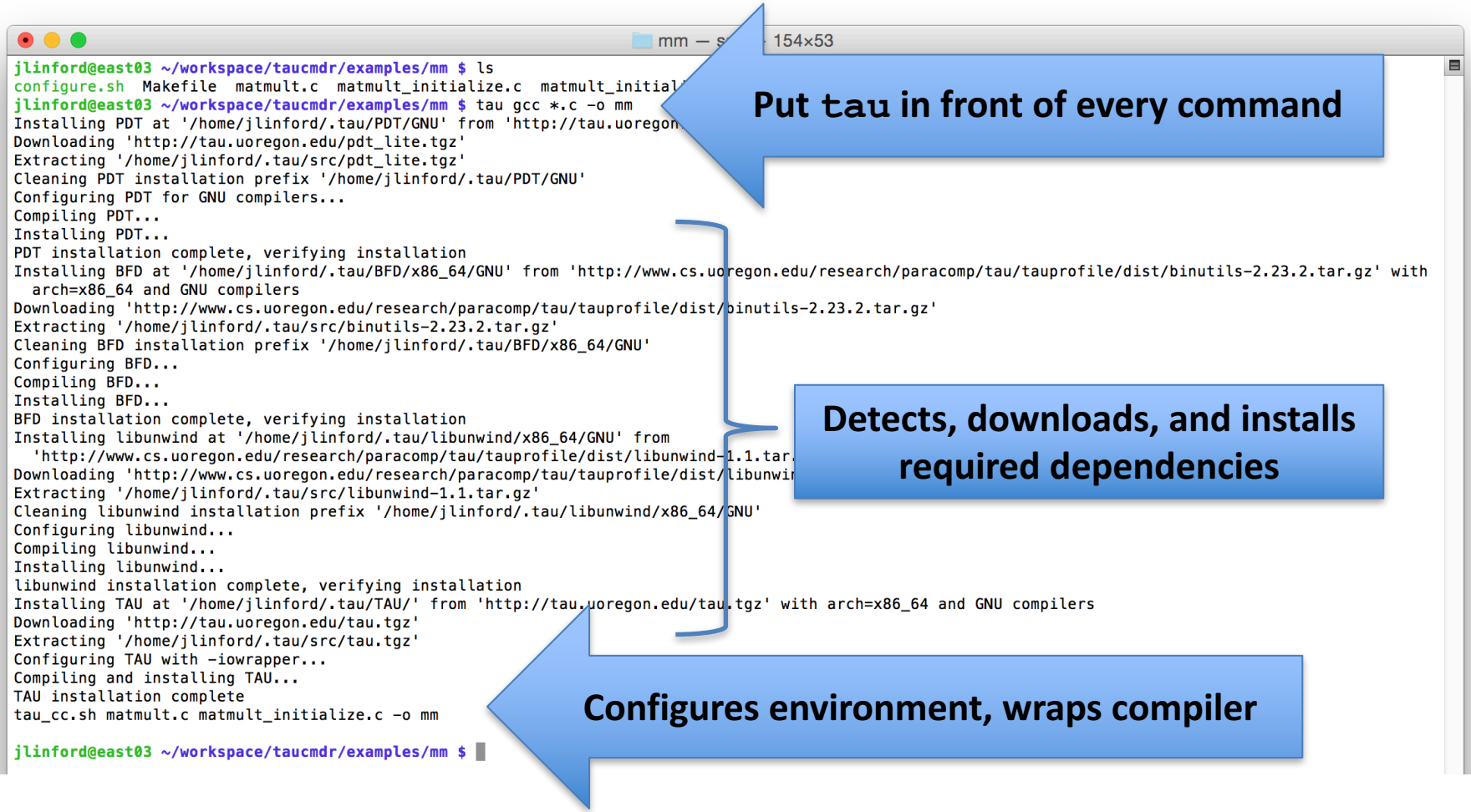
TAU Commander CLI

This command's usage

Subcommand usage

Shortcuts

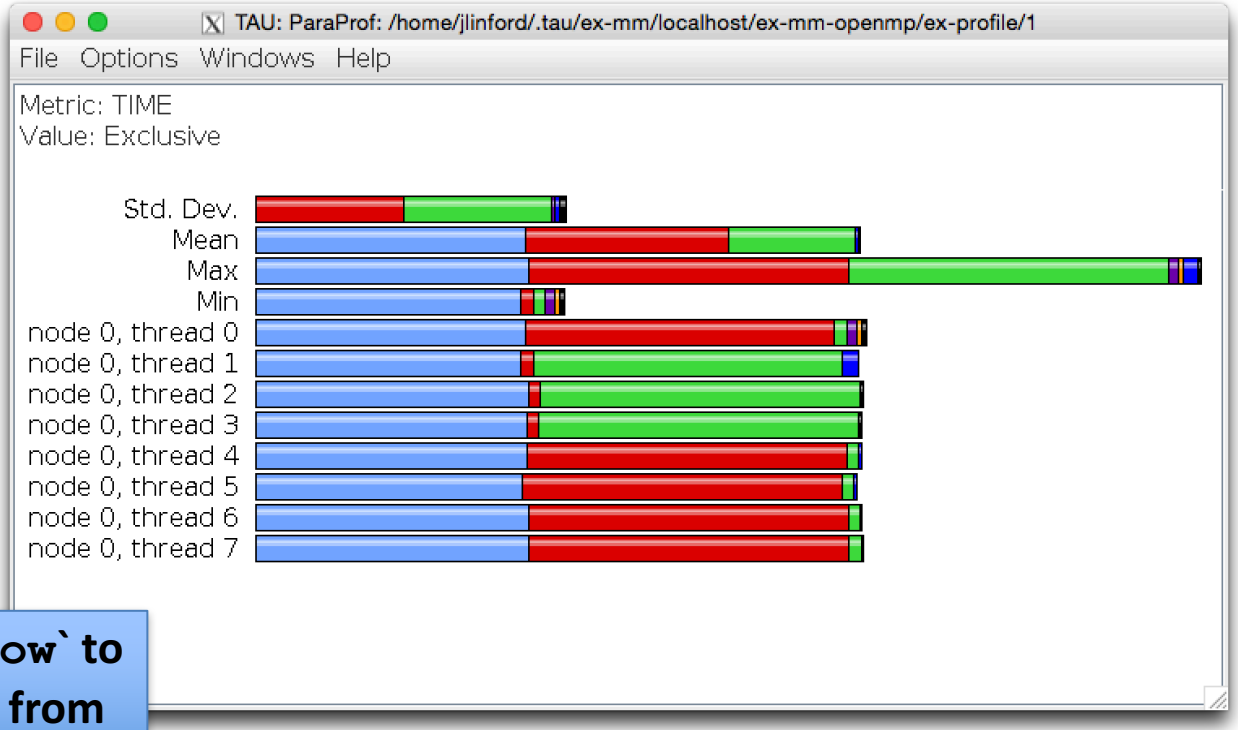
First use on a “vanilla” system



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 — 154×53

```
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   | Compiler  |    MPI    |  OpenMP   | Callpath  |   Mem.    |   Mem.    |    In     |
|           |           |           |           |   Inst.   |   Inst.   |           |           |   Depth   |   Usage   |   Alloc   | Projects  |
+===========+===========+===========+===========+===========+===========+===========+===========+===========+===========+===========+===========+
|       ex- |    Yes    |    No     |    No     | automatic | fallback  |    No     | compiler_ |     0     |    No     |    No     | ex-mm     |
|   profile |           |           |           |           |           |           |  default  |           |           |           |           |
+-----------+-----------+-----------+-----------+-----------+-----------+-----------+-----------+-----------+-----------+-----------+-----------+
|  ex-trace |    No     |    Yes    |    No     | automatic | fallback  |    No     | compiler_ |     0     |    No     |    No     | ex-mm     |
|           |           |           |           |           |           |           |  default  |           |           |           |           |
+-----------+-----------+-----------+-----------+-----------+-----------+-----------+-----------+-----------+-----------+-----------+-----------+
| ex-sample |    Yes    |    No     |    Yes    |   never   |   never   |    No     | compiler_ |     0     |    No     |    No     | ex-mm     |
|           |           |           |           |           |           |           |  default  |           |           |           |           |
+-----------+-----------+-----------+-----------+-----------+-----------+-----------+-----------+-----------+-----------+-----------+-----------+

== 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

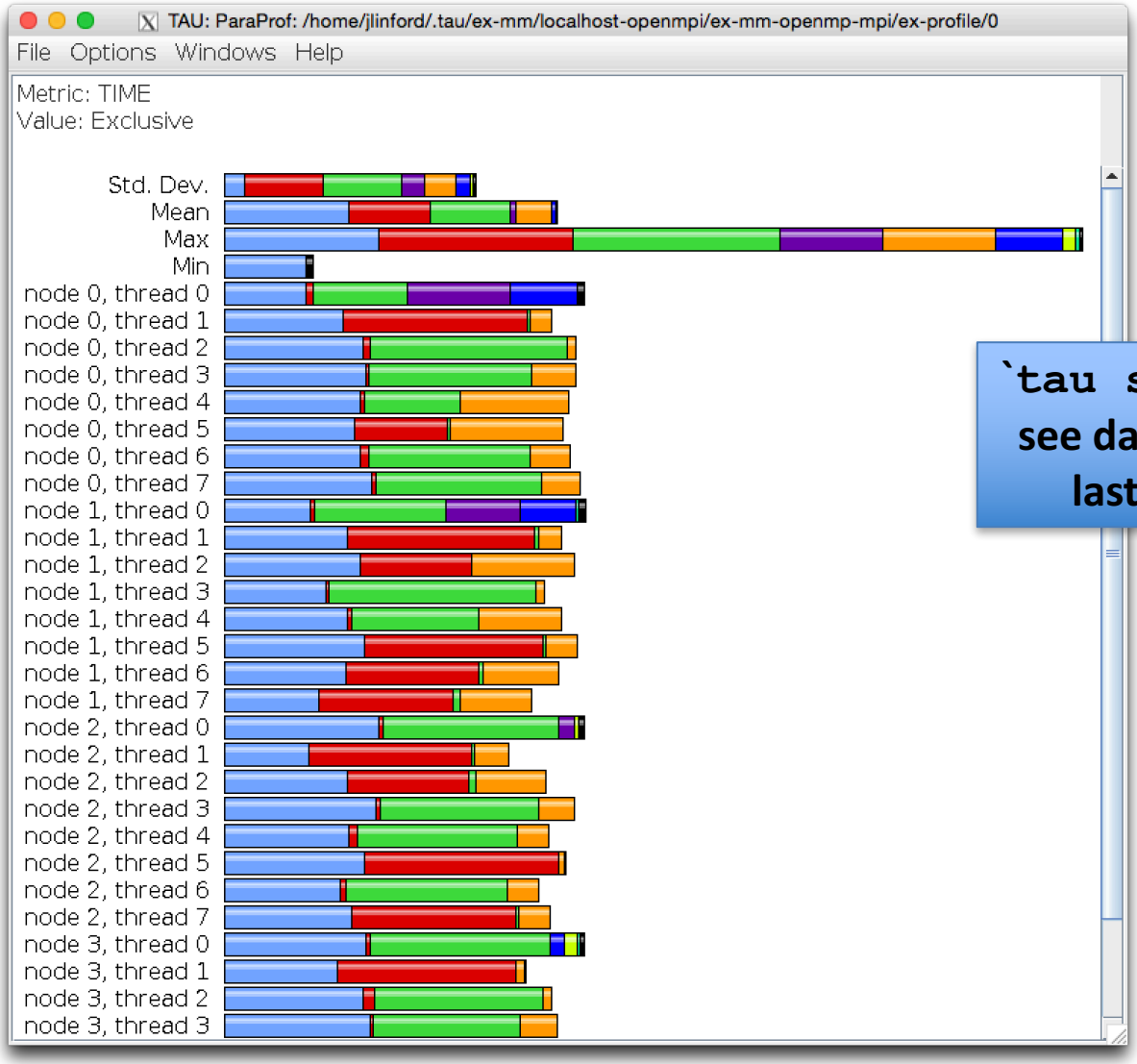
Changing from serial to MPI+OpenMP

```
== 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-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      86_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

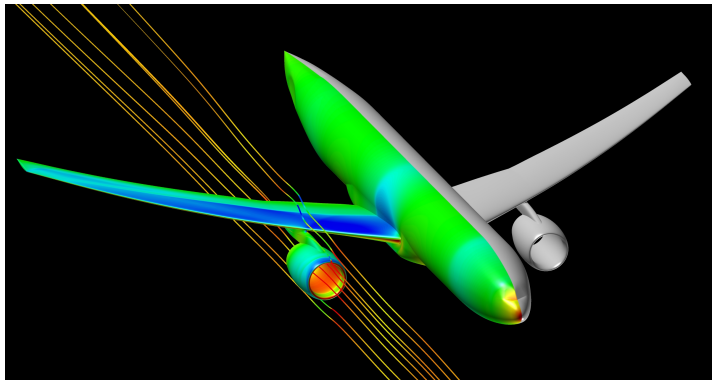


`tau show` to see data from last trial

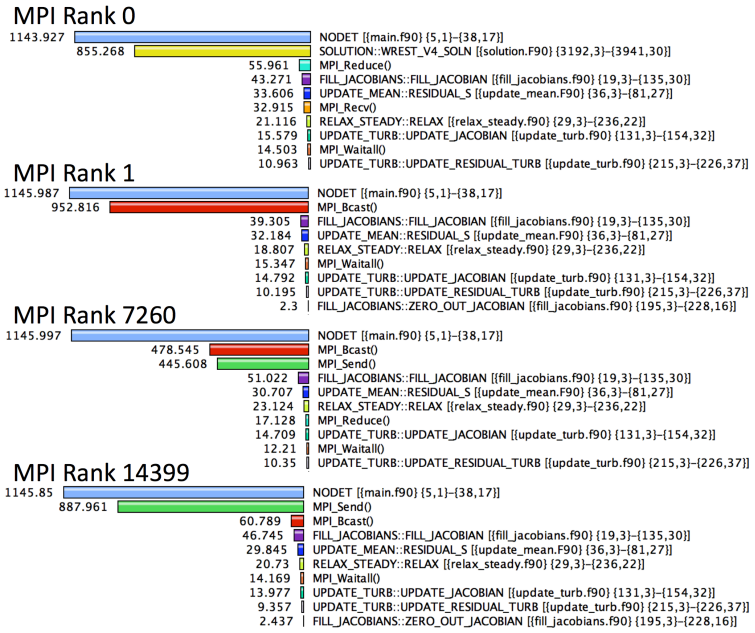
Parallel I/O Tuning for Scientific Applications

BOTTLENECK IDENTIFICATION IN FUN3D

FUN3D Wing-Body-Nacelle



- 1,651,089,924 grid points
- 5,902,801,476 tetrahedral elements
- 1,310,290,264 prismatic elements
- 14,400 Ivy Bridge cores



MPI Rank 0

Name	Total	NumSamples	MaxValue	MinValue	MeanValue	Std. Dev.
Message size for all-gather	2,046,555,416	1,132	19,240,000	4	1,807,911.145	3,307,678.302
Message size for broadcast	39,795,349,570	196,653	75,171,056	1	202,363.298	352,969.91
Message size for gather	16,586,550,740	57,602	82,050,416	4	287,950.952	492,059.37
Message size for reduce	606,584	36,136	344	4	16.786	64.327
Message size received from all nodes	106,507,952,672	223,816	5,038,840	12	475,872.827	1,136,221.111
Message size received in wait	845,737,888	166,100	311,840	12	5,091.739	17,266.492
Message size sent to all nodes	769,221,120	126,660	327,680	20	6,073.118	19,893.424

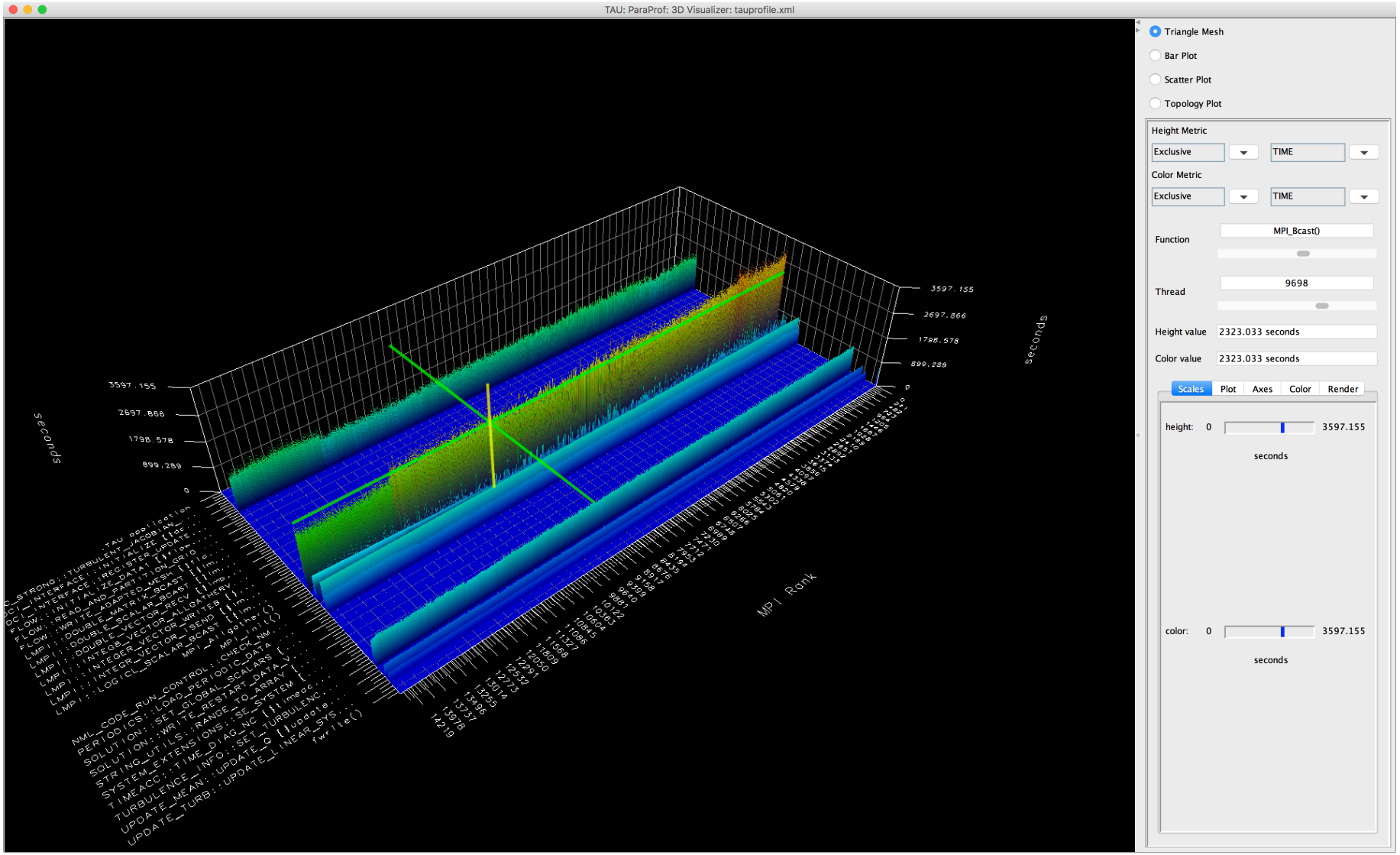
MPI Rank 1

Name	Total	NumSamples	MaxValue	MinValue	MeanValue	Std. Dev.
Message size for all-gather	2,046,555,416	1,132	19,240,000	4	1,807,911.145	3,307,678.302
Message size for broadcast	39,795,349,570	196,653	75,171,056	1	202,363.298	352,969.91
Message size for reduce	606,584	36,136	344	4	16.786	64.327
Message size received from all nodes	546,987,316	137,400	245,040	20	3,980.985	12,778.438
Message size received in wait	546,987,316	137,400	245,040	20	3,980.985	12,778.438
Message size sent to all nodes	503,971,872	105,844	4,637,520	20	4,761.459	20,996.819
Message size sent to node 0	7,420,032	4	4,637,520	927,504	1,855,008	1,606,484.052

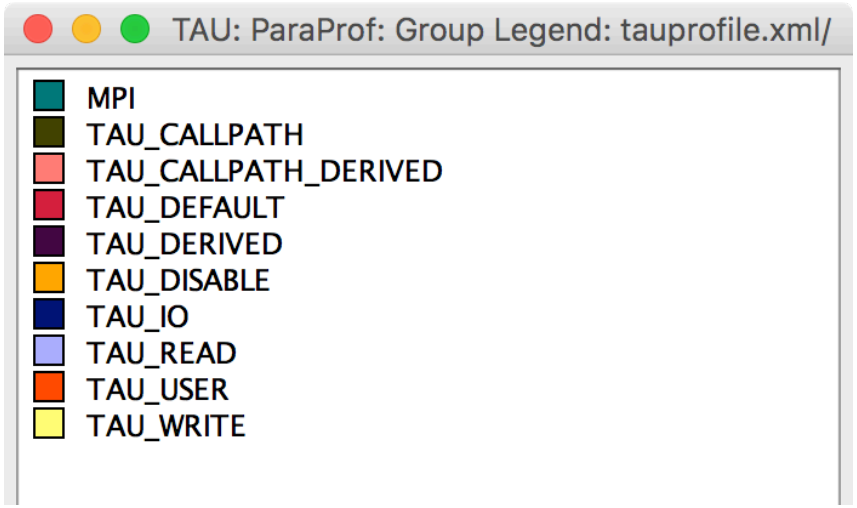
MPI Rank 14399

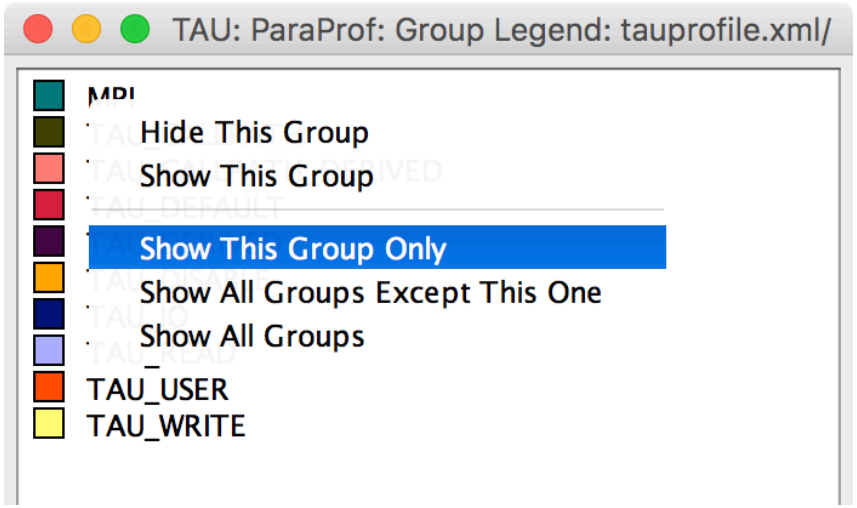
Name	Total	NumSamples	MaxValue	MinValue	MeanValue	Std. Dev.
Message size for all-gather	2,046,555,416	1,132	19,240,000	4	1,807,911.145	3,307,678.302
Message size for broadcast	39,795,349,570	196,653	75,171,056	1	202,363.298	352,969.91
Message size for reduce	606,584	36,136	344	4	16.786	64.327
Message size received from all nodes	495,423,720	112,100	199,280	20	4,419.48	13,658.624
Message size received in wait	495,423,720	112,100	199,280	20	4,419.48	13,658.624
Message size sent to all nodes	455,176,512	122,464	4,632,120	20	3,716.819	18,857.997
Message size sent to node 0	7,411,392	4	4,632,120	926,424	1,852,848	1,604,613.437

FUN3D Profile shows a hot spot



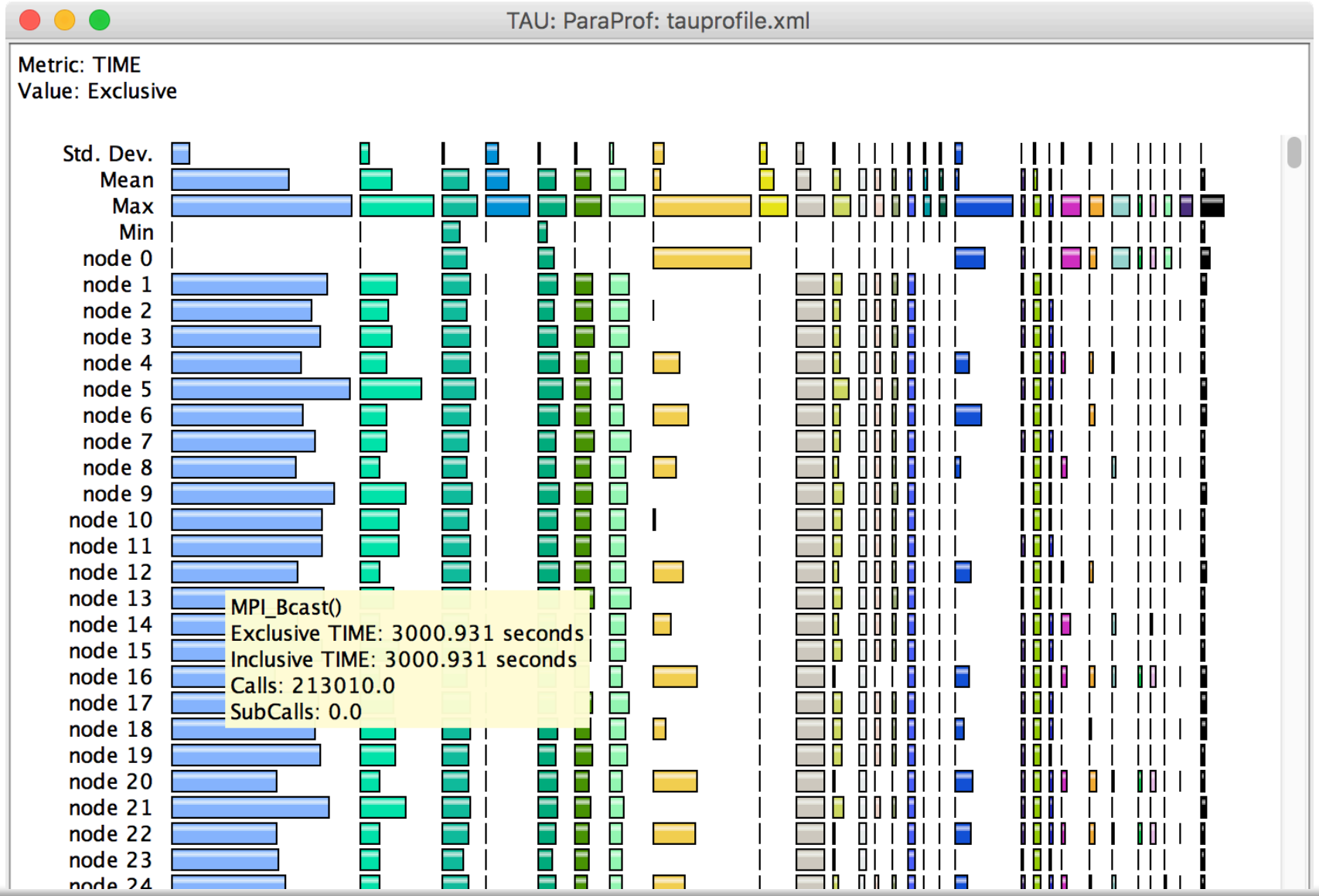
Windows | Group Legend



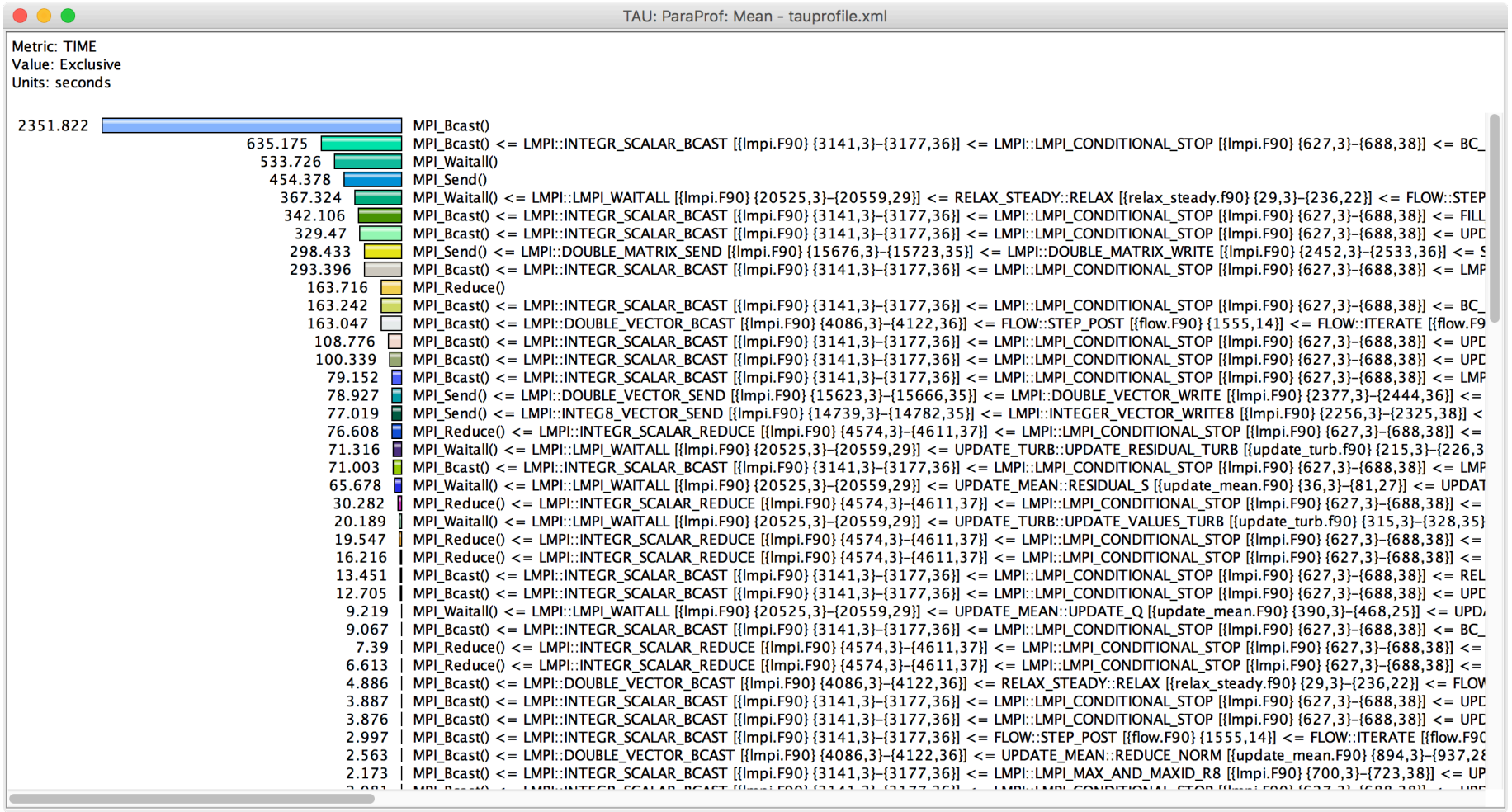


Right-click to show options

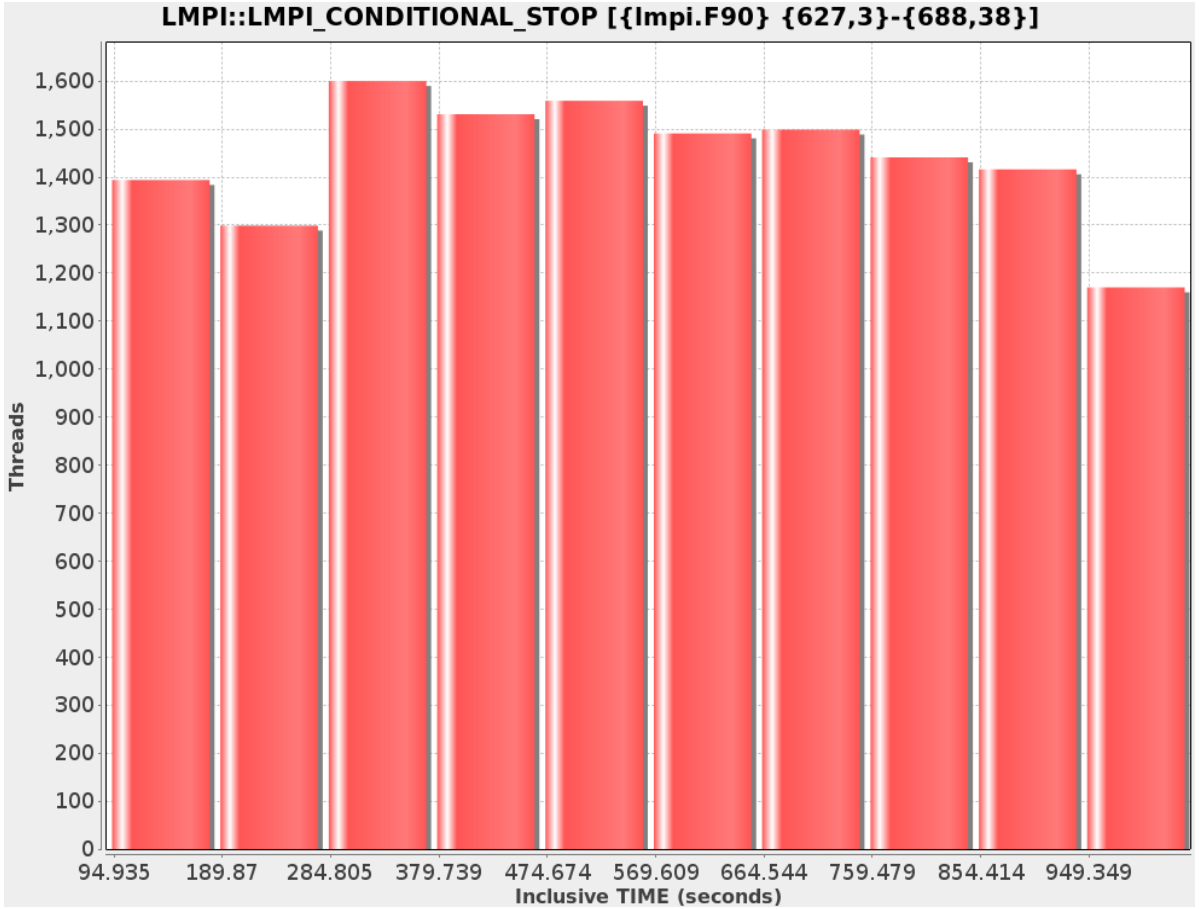
MPI_Bcast in Unstacked Bar Chart



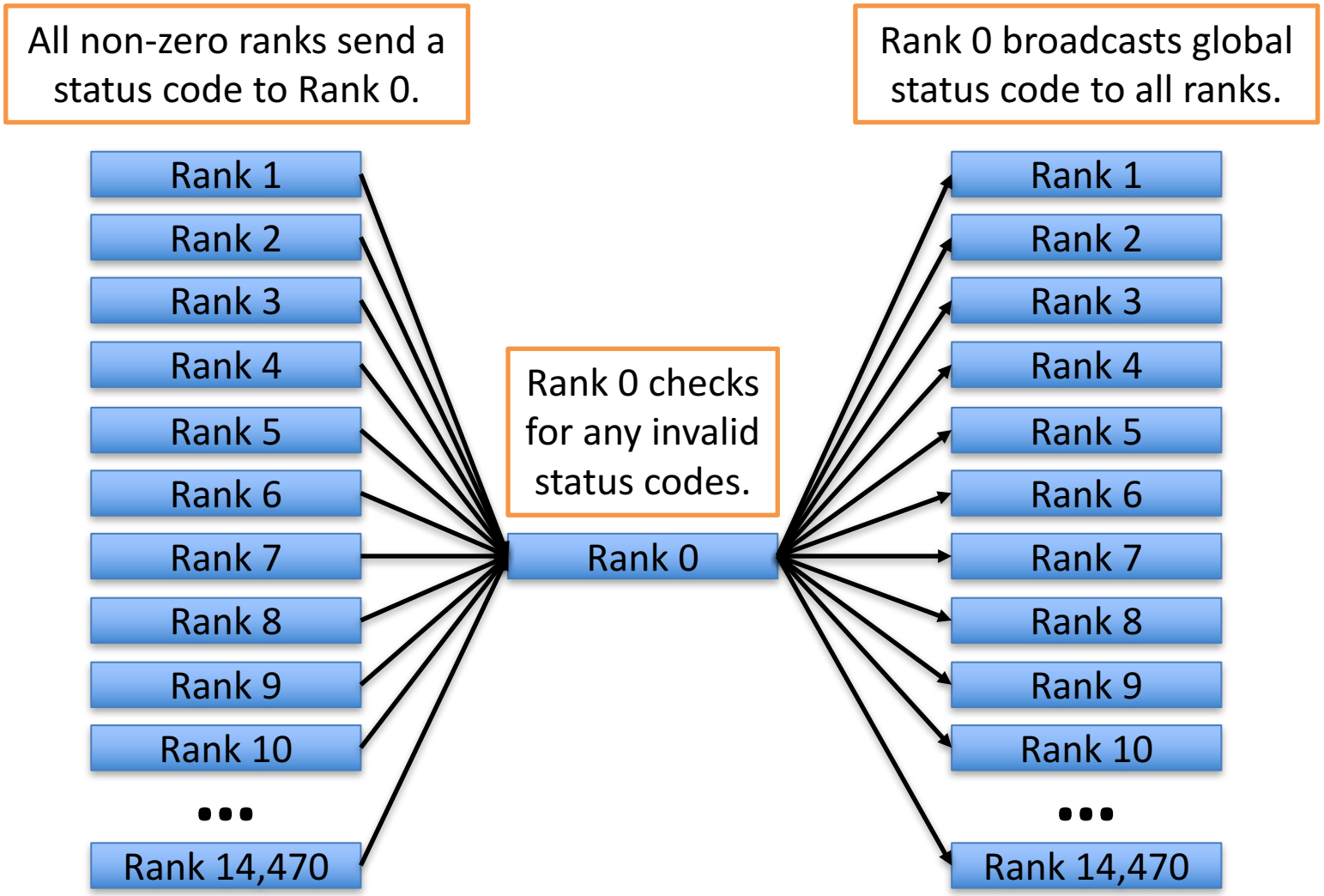
Callpath Profile



Broad histogram indicates bottleneck.



What's going on here?



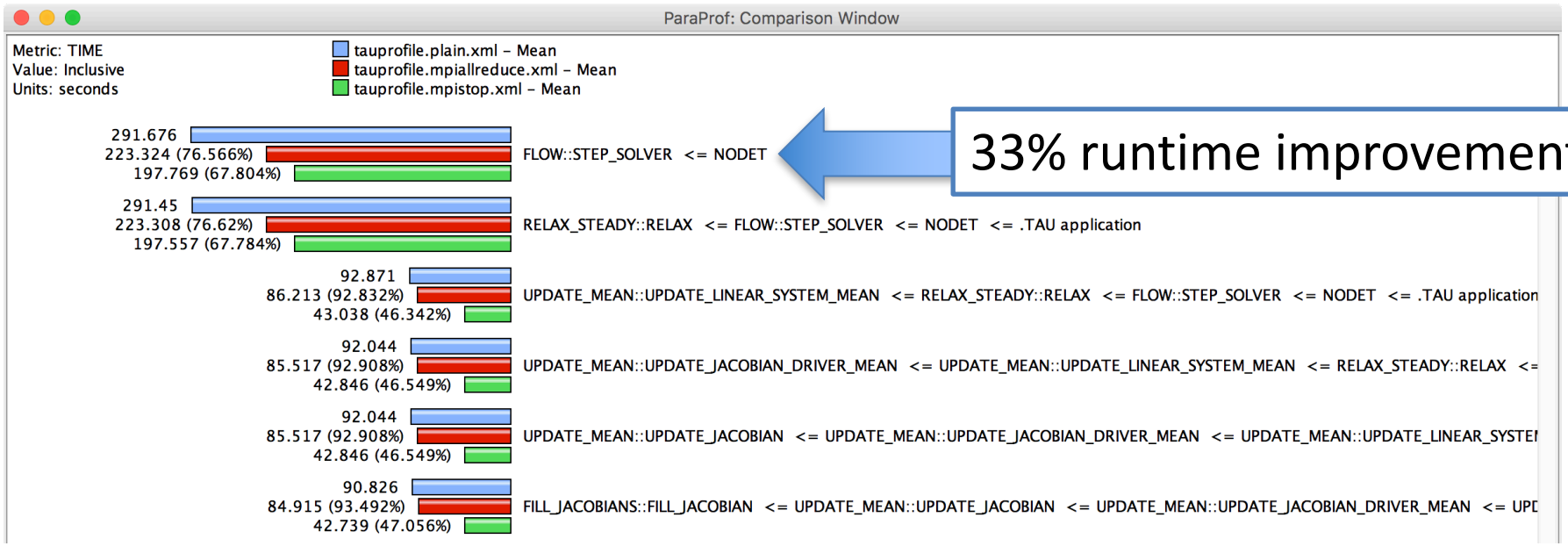
Two Solutions

MPI_Allreduce()

- $O(\log_2 N)$ vs. $O(N)$

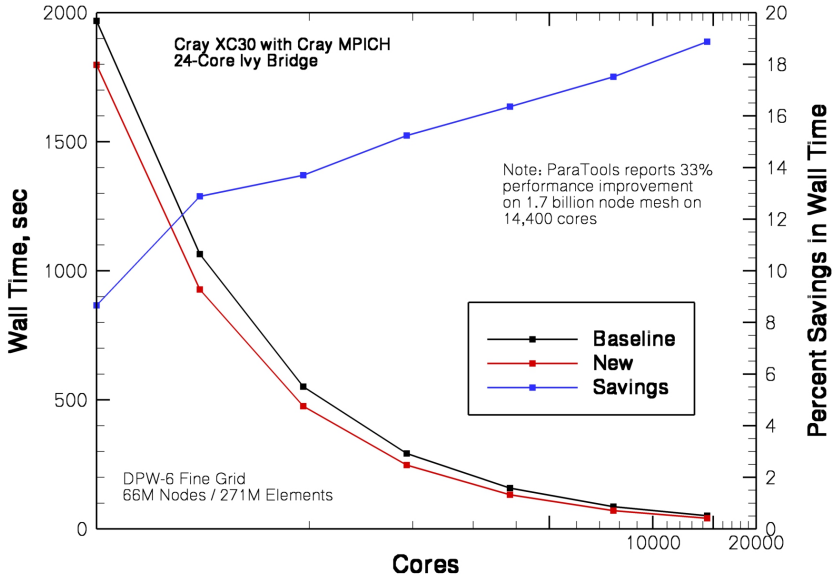
MPI_Abort()

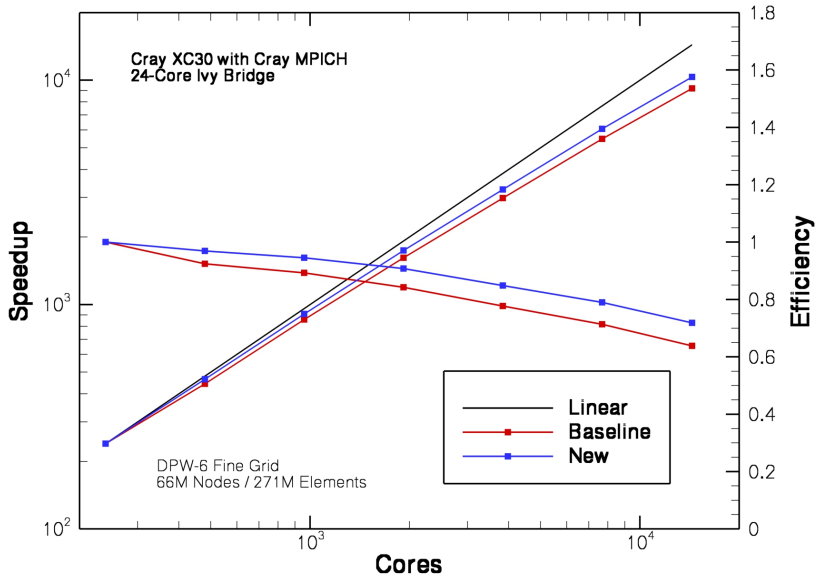
- $O(1)$ vs. $O(N)$



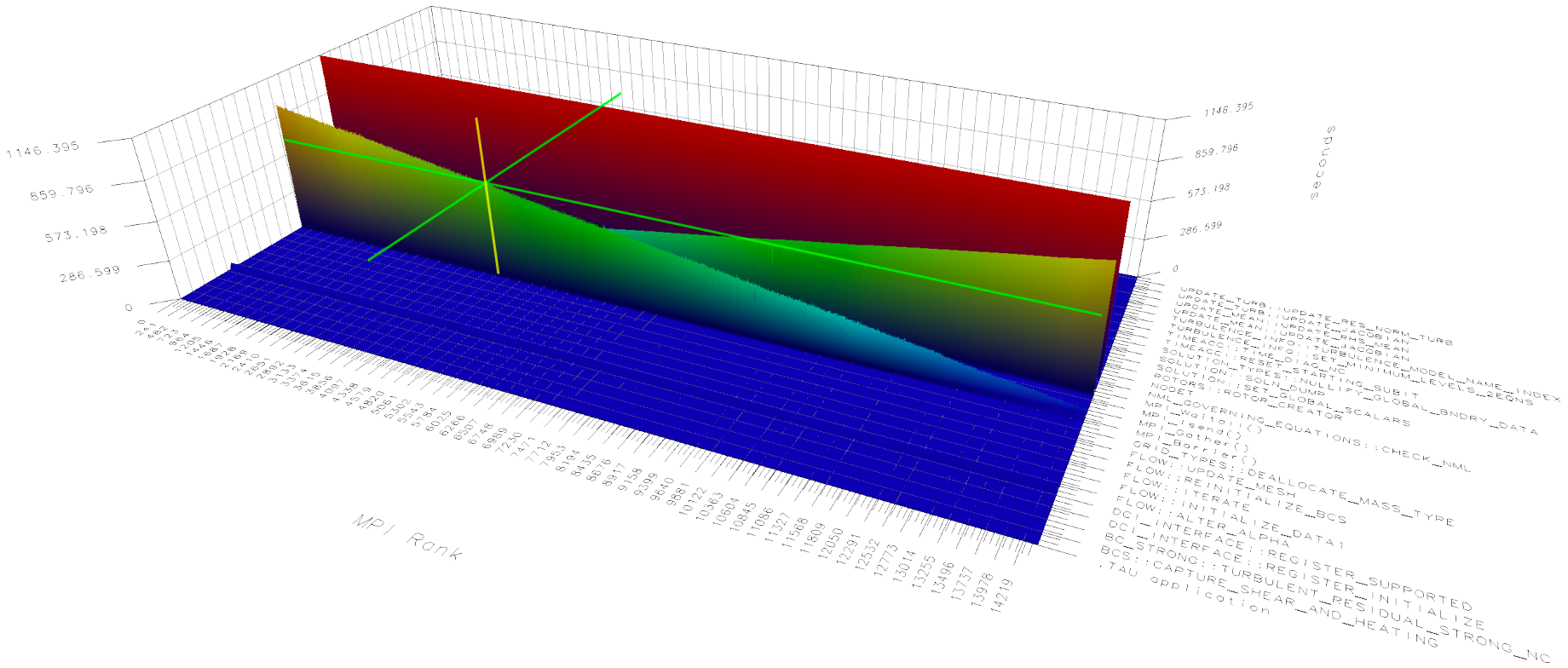
FUN3D Wing-Body-Nacelle

"These days I get excited about 1-2% speedups that I find....quite unusual to find something of this magnitude these days, especially with just a 2-line fix in the code! :)"

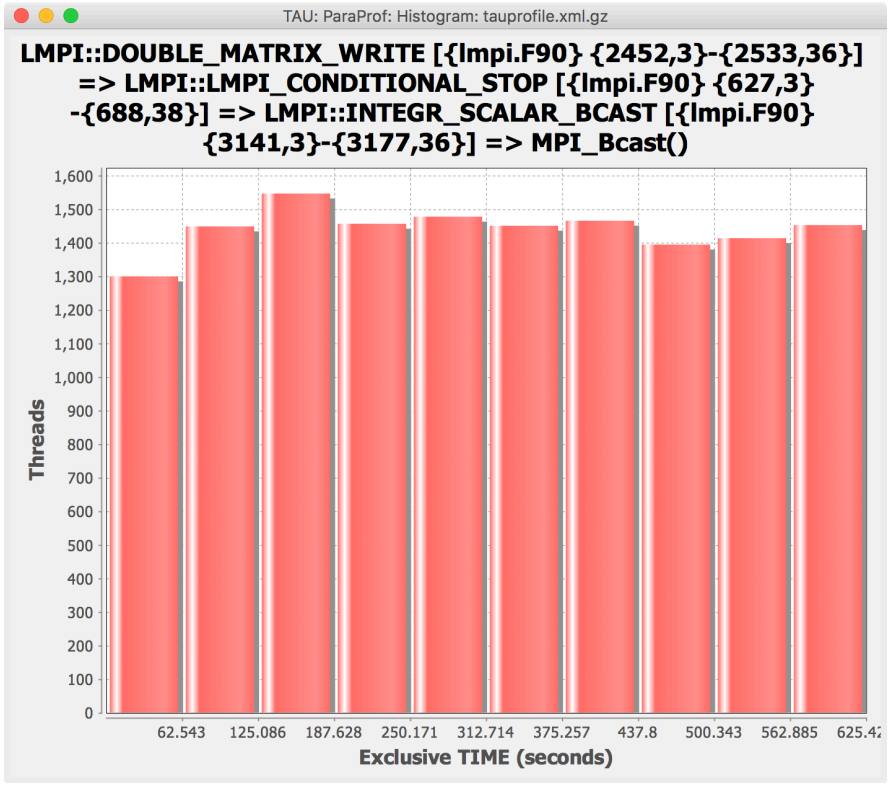


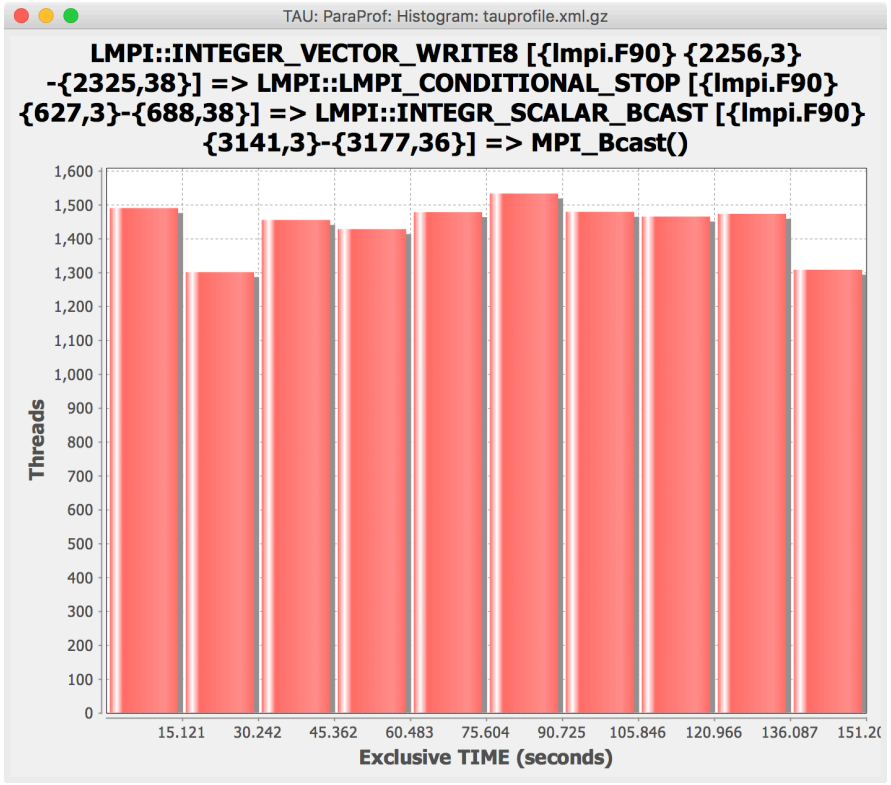


I/O Bottleneck on FUN3D Shutdown



Matrix and Vector I/O shows Bottleneck





BOTTLENECK IDENTIFICATION IN CREATE-AV HELIOS / SAMCART

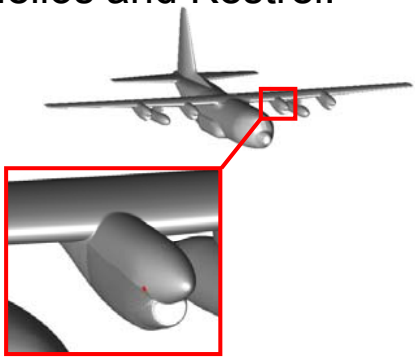
Strand Technology

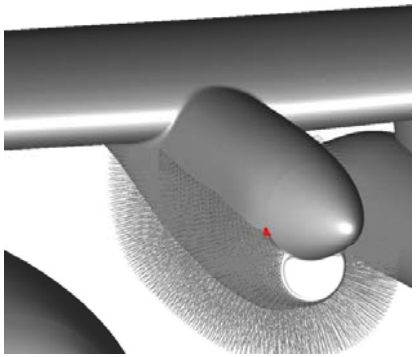
Technology Drivers

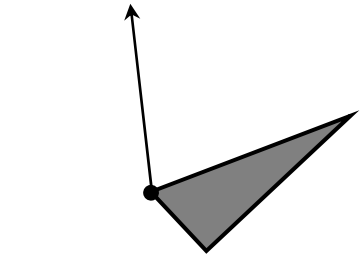
- Timeliness (automation of mesh generation)
- Timeliness (automation and scalability of domain connectivity)
- Timeliness/Physical accuracy (computational efficiency and scalability of aerodynamic solvers)
- Processor architecture (small memory footprint maps well to hierarchical memory architectures,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

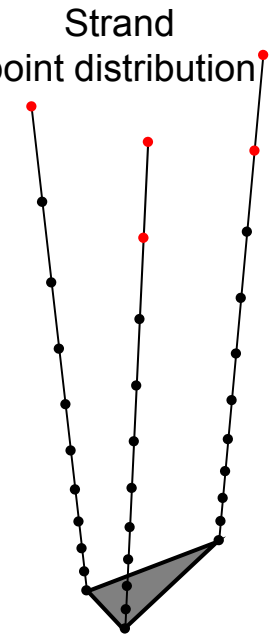
CREATE-AV Example

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]

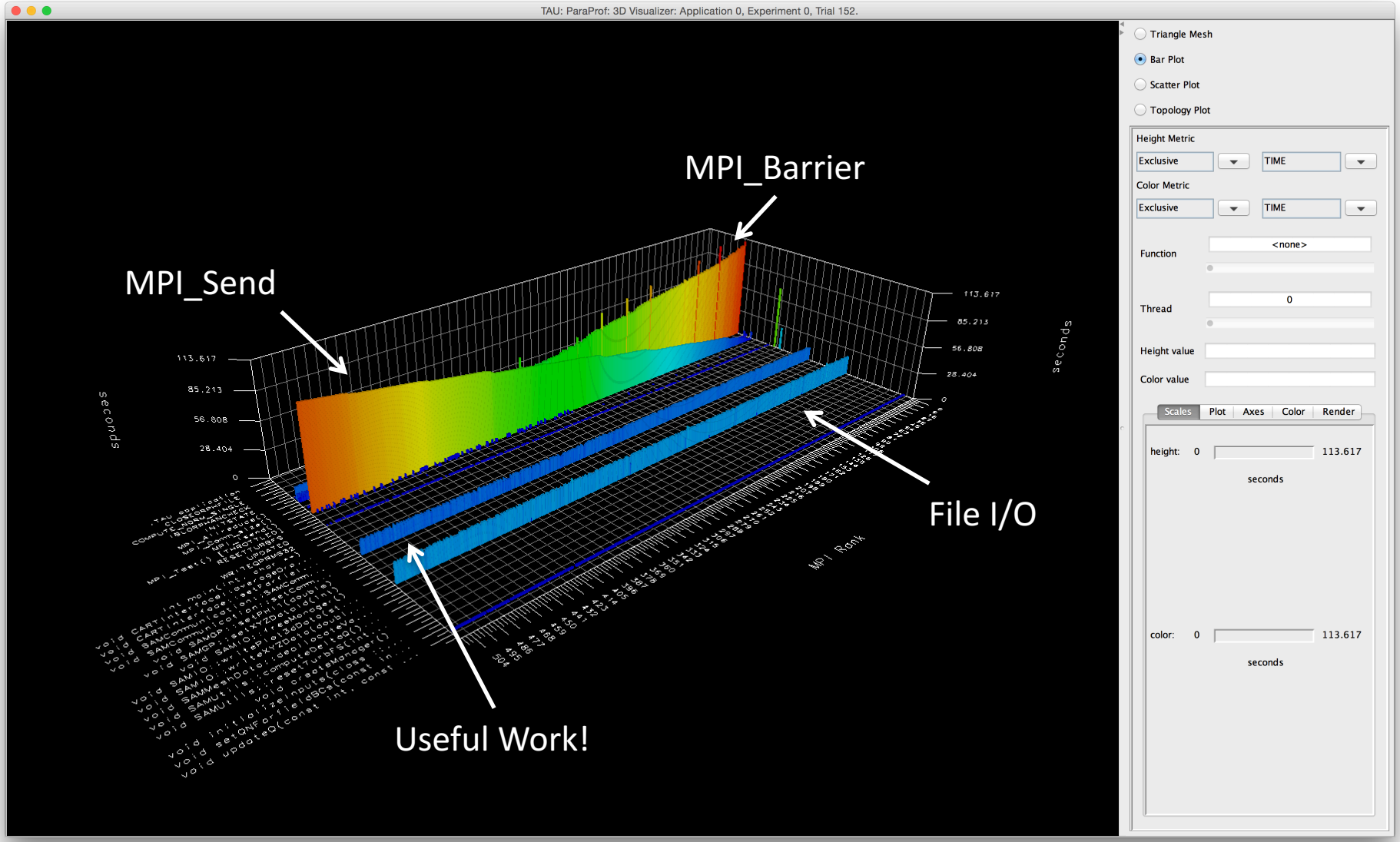
Cori

Haise [iDataPlex]

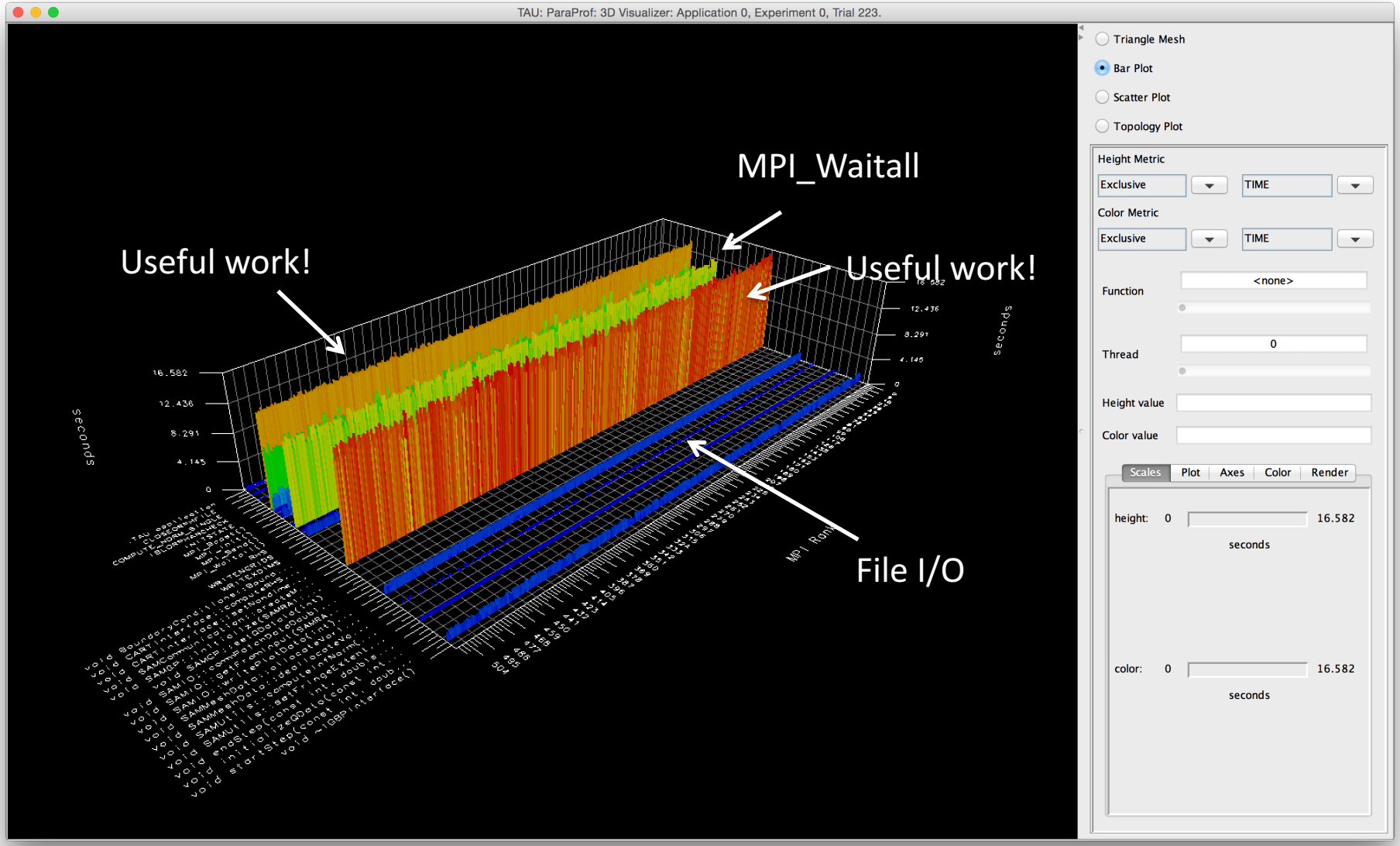
Lightning [XC30]

Kilrain [iDataPlex]

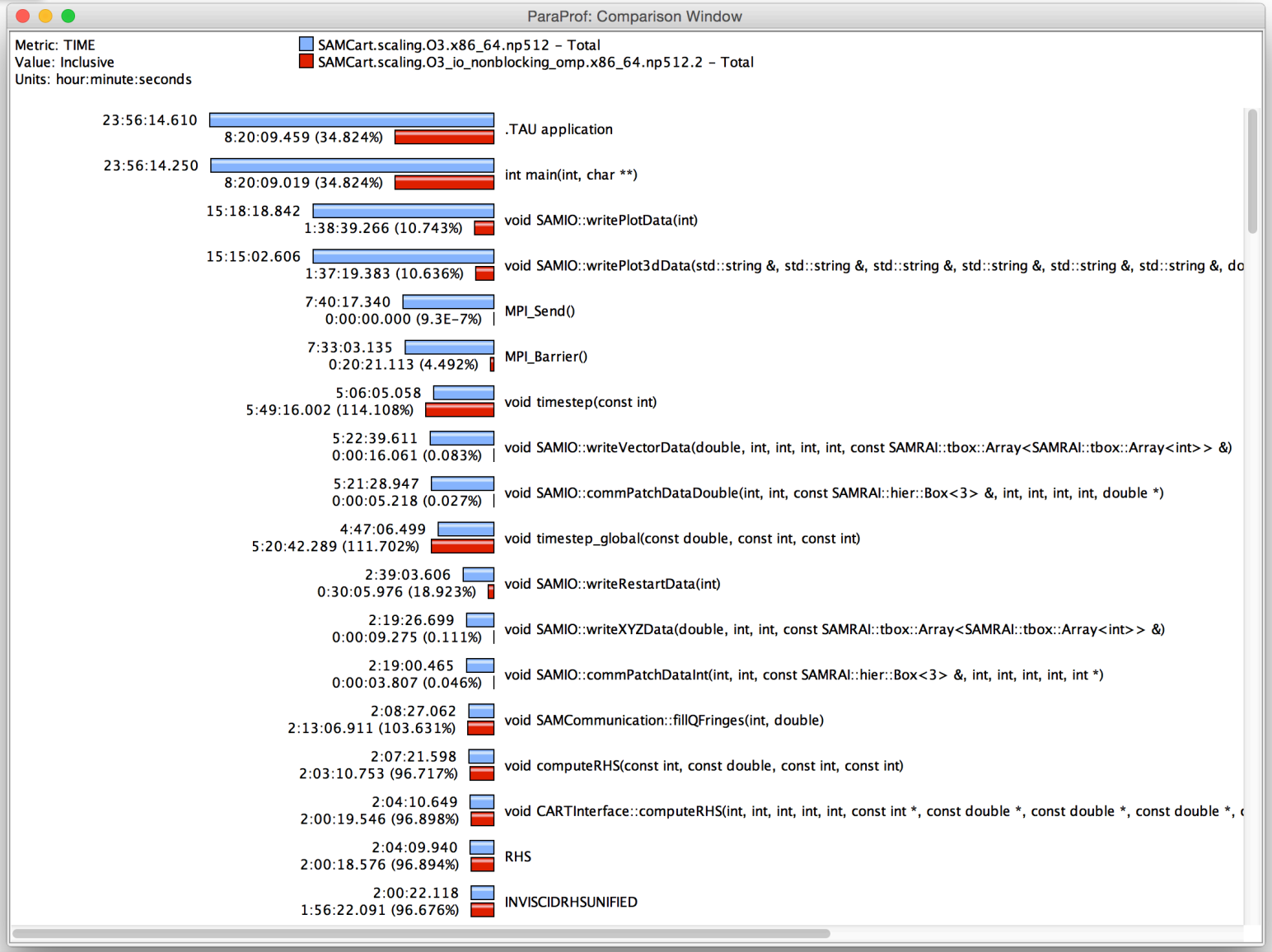
Initial Profile on Babbage



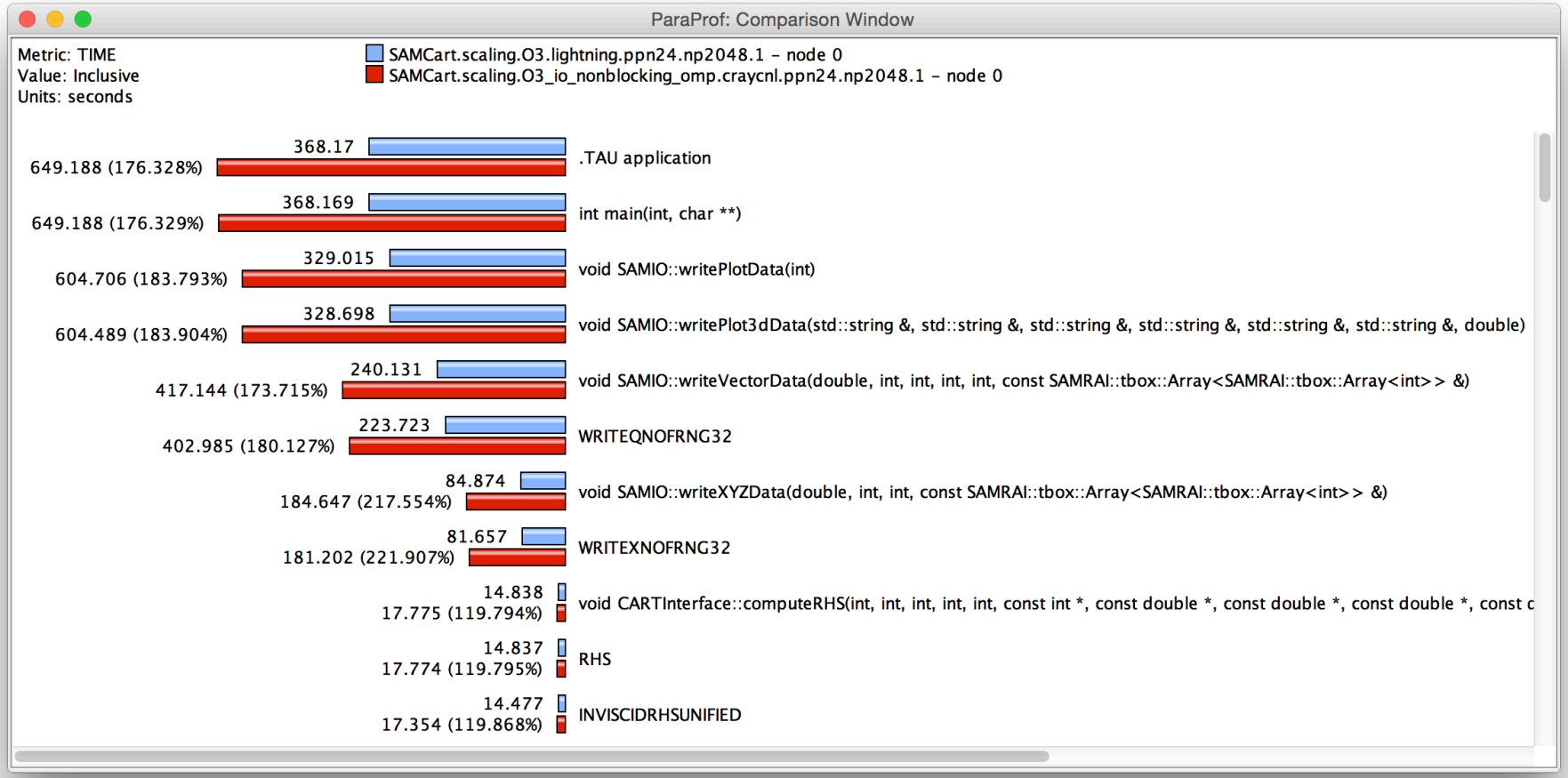
Implement Non-blocking Comm.



65% Runtime Reduction (~2x faster)

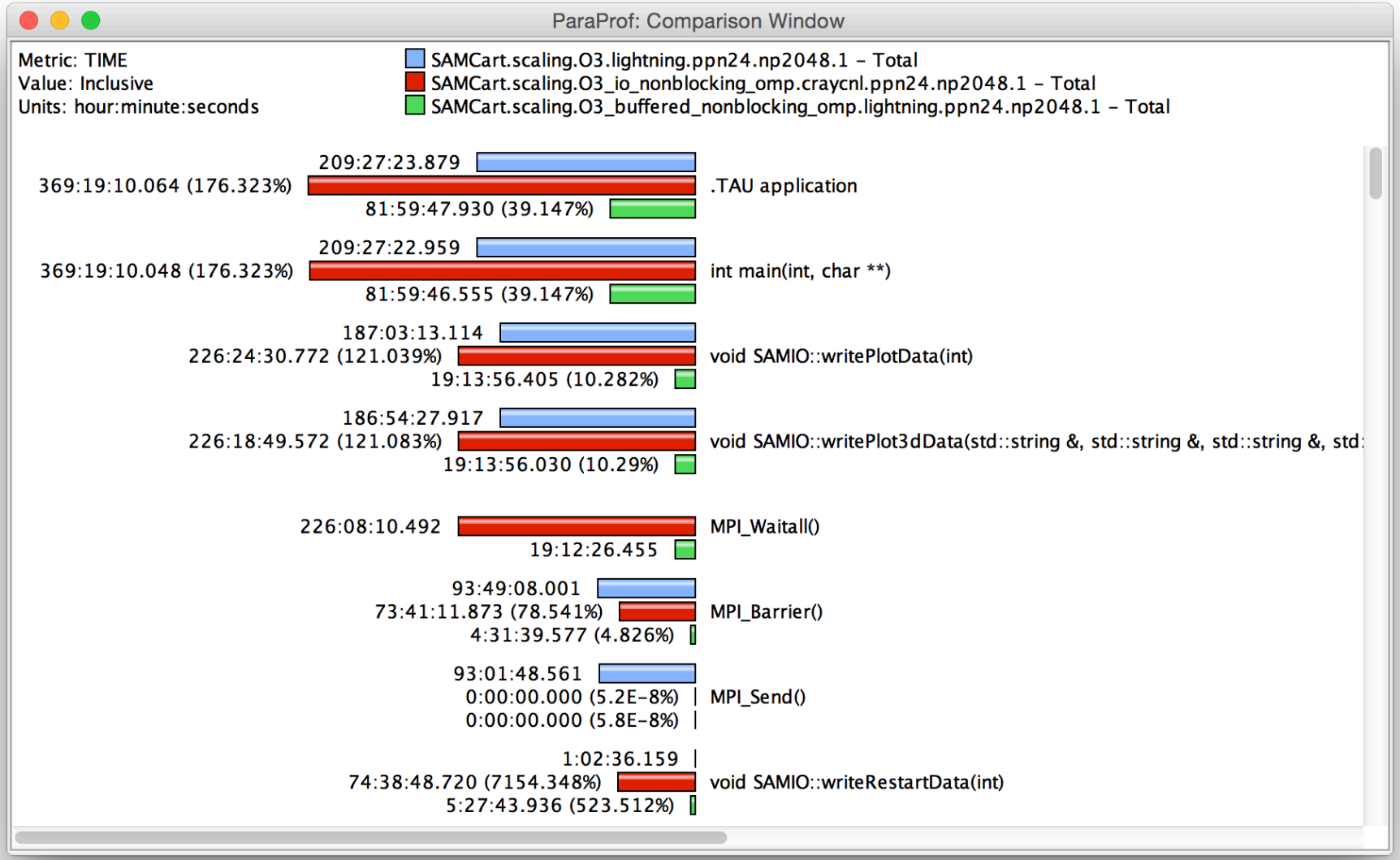


Lightning (a.k.a. watch your I/O)



Slower! What happened???

Buffered I/O, change stripe size

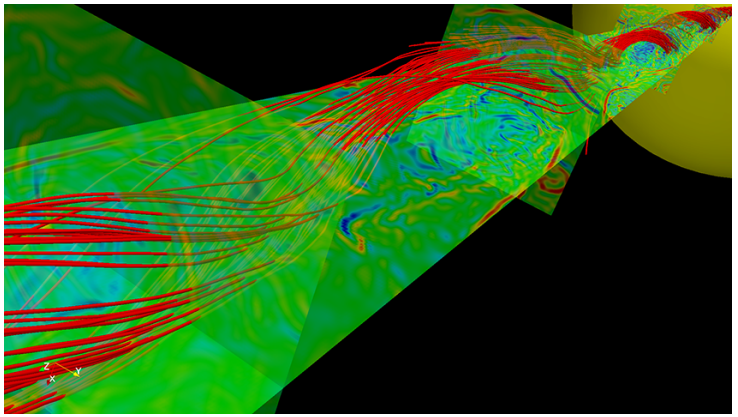


Intuitive Performance Engineering

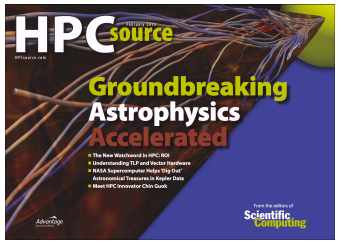
IRMHD ON BLUEGENE/Q

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)



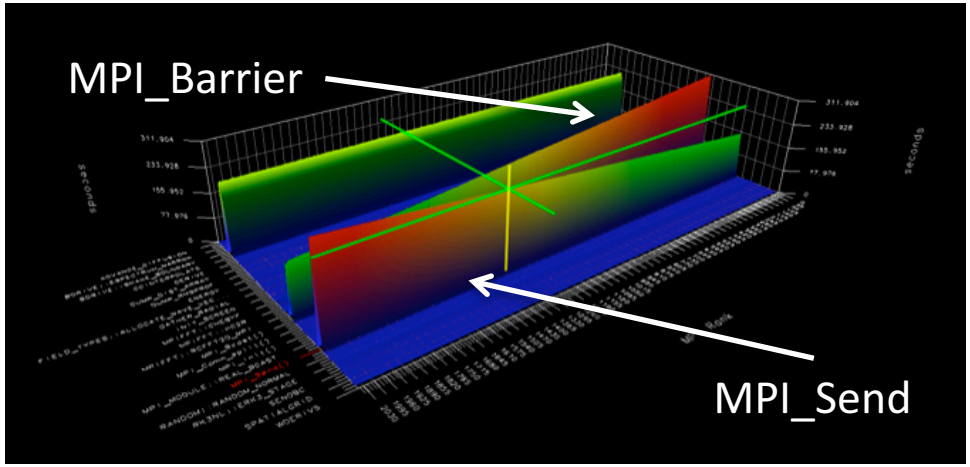
- IRMHD (Inhomogeneous Reduced Magnetohydrodynamics)
  - Fortran 90 and MPI
  - Excellent weak and strong scaling properties
  - Tested and benchmarked on Intrepid and Mira



- HPC Source article and ALCF news
  https://www.alcf.anl.gov/articles/furthering-understanding-coronal-heating-and-solar-wind-origin

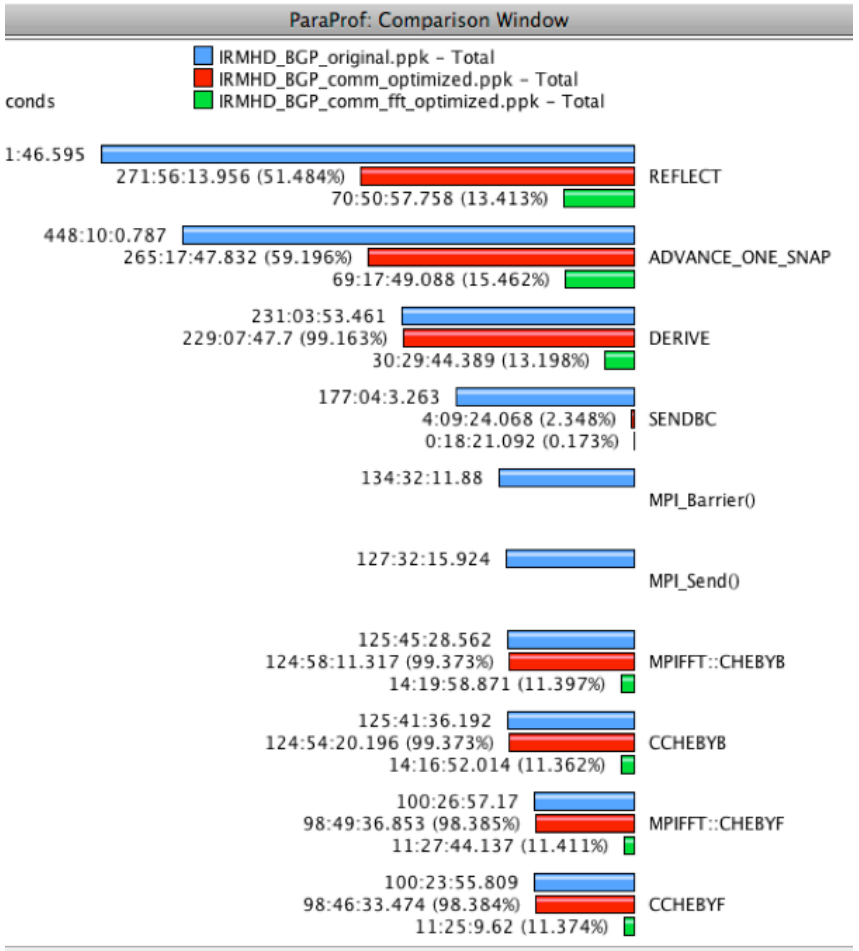
IRMHD Communication Analysis

- 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



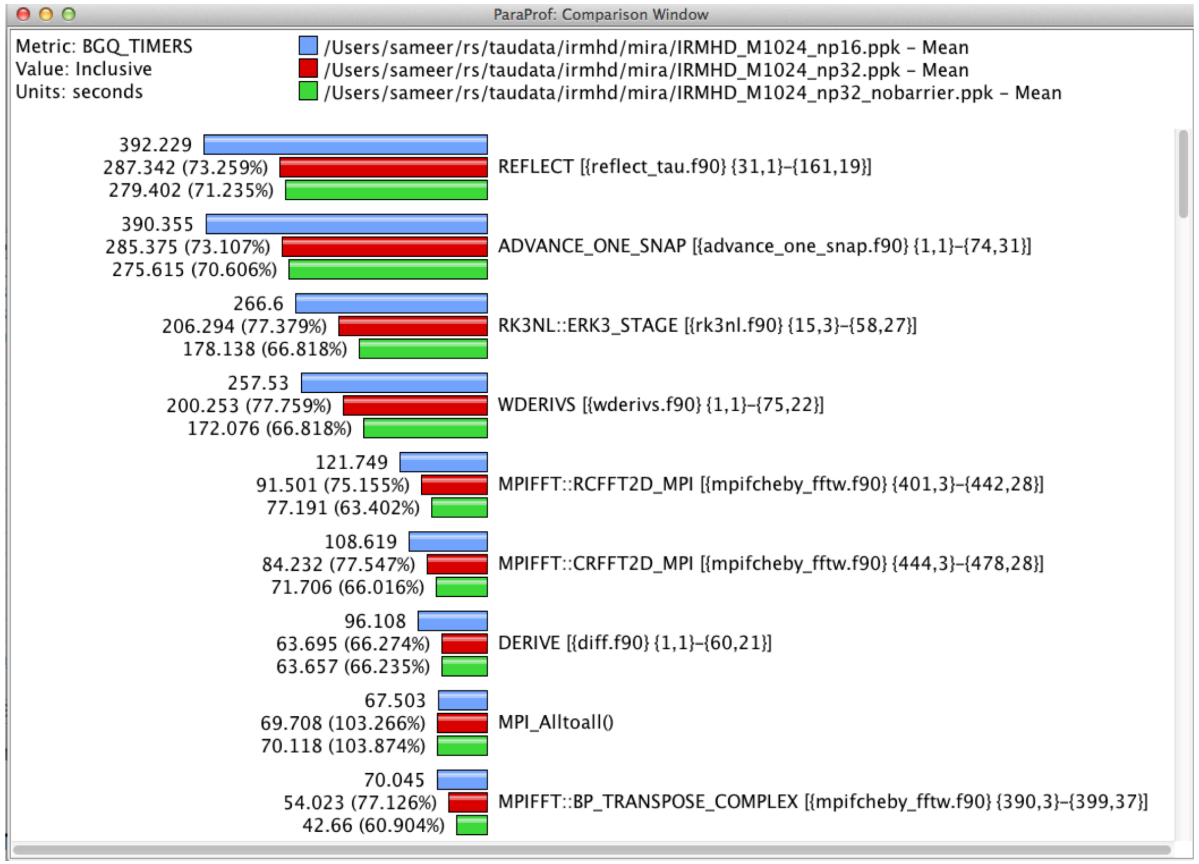
IRMHD Optimization on Intrepid (BG/P)

- 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)

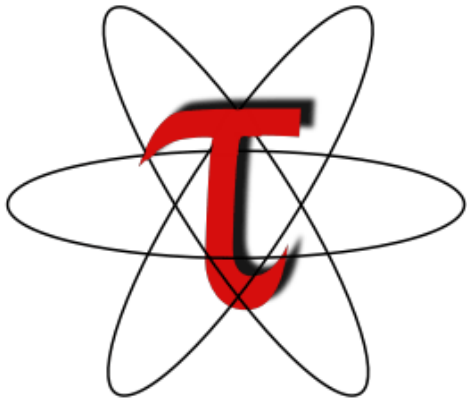
- Oversubscribe nodes: 32k ranks vs. 16k per node
- Overall time improvement: 71.23% of original



Intuitive Performance Engineering

CONCLUSION

Downloads



http://taucommander.paratools.com

http://tau.uoregon.edu

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