OAK RIDGE National Laboratory

How to use TAU for Performance Analysis

George S. Markomanolis

7 August 2019

ORNL is managed by UT-Battelle, LLC for the US Department of Energy

U.S. DEPARTMENT OF ENERGY

Outline

- Introduction to TAU
- How to compile
- Explaining functionalities of TAU/ParaProf
- Presenting basic steps of PerfExplorer

TAU

- Tuning and Analysis Utilities, developed at University of Oregon
- Scalable and flexible performance analysis toolkit
- Automatic instrumentation through Program Database Toolkit (PDT) for routines, loops, I/O, memory, phases, etc.
- Installed version on Summit: v2.28.1
- Module: tau
- Web site: https://www.cs.uoregon.edu/research/tau/home.php
- Email: tau-bugs@cs.uoregon.edu

Capability Matrix - TAU

Capability	Profiling	Tracing	Notes/Limitations
MPI, MPI-IO	Yes	Yes	
OpenMP CPU	Yes	Yes	
OpenMP GPU	Yes	Yes	Some restrictions apply regarding the CUPTI metrics
OpenACC	Yes	Yes	Some functionalities are not ready for production, no metrics available
CUDA	Yes	Yes	Some functionalities are not ready for production
POSIX I/O	Yes	Yes	
POSIX threads	Yes	Yes	
Memory – app-level	Yes	Yes	
Memory – func-level	Yes	Yes	
Hotspot Detection	Yes	Yes	
Variance Detection	Yes	Yes	
Hardware Counters	Yes	Yes	

Compilation

- There are mainly three approaches to use an application with TAU
  - Use TAU Wrappers
    - For C: replace the compiler with tau_cc.sh
    - For C++: replace the compiler with tau_cxx.sh
    - For Fortran: replace the compiler with tau_f90.sh/tau_f77.sh
  - Dynamic instrumentation, for example:
    - jsrun -n 4 –r 4 –a 1 –c1 tau_exec -T mpi ./test
  - Rewrite the binary (support for x86_64):
    - tau_rewrite –T papi,pdf a.out –o a.inst

Compilation (cont.)

Method	Requires recompil-ing	Requires PDT	Shows MPI events	Routine-level event	Low level events (loops, phases, etc...)	Throttling to reduce overhead	Ability to exclude file from in-strumenta-tion
Interposi-tion			Yes			Yes	
Compiler	Yes		Yes	Yes		Yes	Yes
Source	Yes	Yes	Yes	Yes	Yes	Yes	Yes

Interposition: tau_exec

Compiler:
tau_cc.sh –tau_options=-optCompInst
Set the TAU_MAKEFILE

Source:
tau_cc.sh
The TAU_MAKEFILE should include the PDT

tau_exec

tau_exec –help

Options:

-v	Verbose mode
-s	Show what will be done but don't actually do anything (dryrun)
-io	Track I/O
-memory	Track memory allocation/deallocation
-memory_debug	Enable memory debugger
-cuda	Track GPU events via CUDA
-cupti	Track GPU events via CUPTI (Also see env. variable TAU_CUPTI_API)
-opencl	Track GPU events via OpenCL
-openacc	Track GPU events via OpenACC (currently PGI only)
-rocm	Track ROCm events via rocprofiler
-ompt	Track OpenMP events via OMPT interface
-ebs	Enable event-based sampling
-ebs_period=<count>	Sampling period (default 1000)
-ebs_source=<counter>	Counter (default itimer)
-ebs_resolution=<file|function|line>	Choose sampling granularity.
-um	Enable Unified Memory events via CUPTI
-sass=<level>	Track GPU events via CUDA with Source Code Locator activity (kernel level or source level)
-csv	Outputs sass profile in CSV
-env	Track GPU environment activity (power utilization, SM, memory frequency, temperature)

-T <CUPTI,DISABLE,GNU,GNU_MEM,MPI,OPENMP,PAPI,PDT,PGI,PGI_MEM,PROFILE,SERIAL> : Specify TAU tags

TAU 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
TAU_TRACK_HEAP	0	Setting to 1 turns on heap memory/headroom at routine entry & exit
TAU_CALLPATH_DEPTH	2	Specifies depth of callpath
TAU_TRACK_IO_PARAMS	0	Setting to 1 with -optTrackIO
TAU_SAMPLING	1	Generates sample based profiles
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
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 snapshot per thread
TAU_METRICS	TIME	Setting to a comma separated list (TIME:PAPI_TOT_INS:PAPI_FP_OPS)

TAU Compile-Time Environment Variables

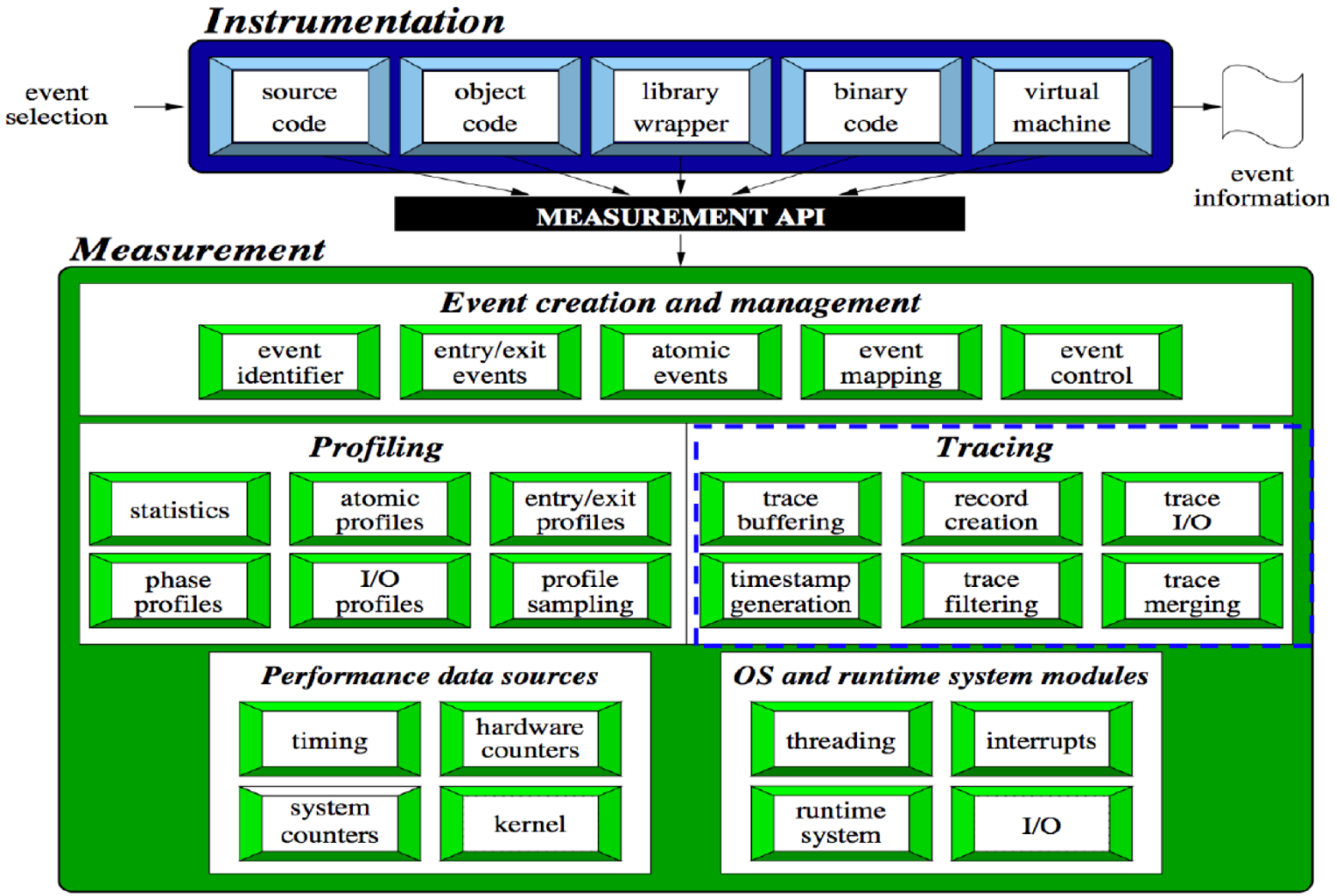
-optVerbose	Turn on verbose debugging messages
-optCompInst	Use compiler based instrumentation
-optNoCompInst	Do not revert to compiler instrumentation if source instrumentation fails
-optTrackIO	Wrap POSIX I/O call and calculates vol/bw of I/O operations
-optKeepFiles	Does not remove .pdb and .inst.* files
-optPreProcess	Preprocess Fortran sources 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
-optLinking=""	Options passed to the linker
-optCompile=""	Options passed to the compiler
-optPdtF95Opts=""	Add options for Fortran parser in PDT
-optPdtF95Reset=""	Reset options for Fortran parser in PDT
-optPdtCOpts=""	Options for C parser in PDT
-optPdtCxxOpts=""	Options for C++ parser in PDT

For using free format in .f files, use:
% export TAU_OPTIONS=`-optPdtF95Opts=``-R free'''

How TAU works?

- Instrumentation:
  - Adds probes to perform measurements
  - Source code instrumentation
  - Wrapping external libraries (I/O, CUDA, OpenACC, OpenCL)
  - Rewriting the binary executable
- Measurement:
  - Profiling or Tracing
  - Direct instrumentation
  - Sampling
  - Throttling
- Analysis:
  - Visualization of profiles and traces
  - 3D visualization
  - Trace conversion tools

TAU Instrumentation/Measurement



Tau_exec

Usage: tau_exec [options] [--] <exe> <exe options>
Options:

-v Verbose mode
-vv Very Verbose mode (enables TAU_VERBOSE=1)
-s Show what will be done but don't actually do anything (dryrun)

-io Track I/O
-memory Track memory allocation/deallocation
-memory_debug Enable memory debugger
-cuda Track GPU events via CUDA
-cupti Track GPU events via CUPTI (Also see env. variable TAU_CUPTI_API)
-opencl Track GPU events via OpenCL
-openacc Track GPU events via OpenACC (currently PGI only)
-rocm Track ROCm events via rocprofiler
-ompt Track OpenMP events via OMPT interface
-power Track power events via PAPI's perf RAPL interface|
-numa Track remote DRAM, total DRAM events (needs papi with recent perf support for x86_64)
-ebs Enable event-based sampling
-ebs_period=<count> Sampling period (default 1000)
-um Enable Unified Memory events via CUPTI
-sass=<level> Track GPU events via CUDA with Source Code Locator activity (kernel level or source level)
-csv Outputs sass profile in CSV

MiniWeather MPI compilation

- module load pgi
- module load tau
- export TAU_MAKEFILE/sw/summit/tau/2.28.1_patched/ibm64linux/lib/Makefile.tau-pgi-papi-mpi-pdt-pgi
- Replace mpicxx with tau_cxx.sh in the Makefile
- export TAU_OPTIONS='-optLinking=-lpnetcdf -optVerbose'
- make mpi

MiniWeather MPI – Execution - Profiling

```
export TAU_METRICS=TIME:PAPI_TOT_INS:PAPI_TOT_CYC:PAPI_FP_OPS
#export TAU_CALLPATH=1
#export TAU_CALLPATH_DEPTH=10
export TAU_PROFILE=1
export TAU_TRACK_MESSAGE=1
export TAU_COMM_MATRIX=1

jsrun -n 64 -r 8 -a 1 -c 1    ./miniWeather_mpi
```

Or if compiled with mpicxx

```
jsrun -n 64 -r 8 -a 1 -c 1  tau_exec ./miniWeather_mpi
```

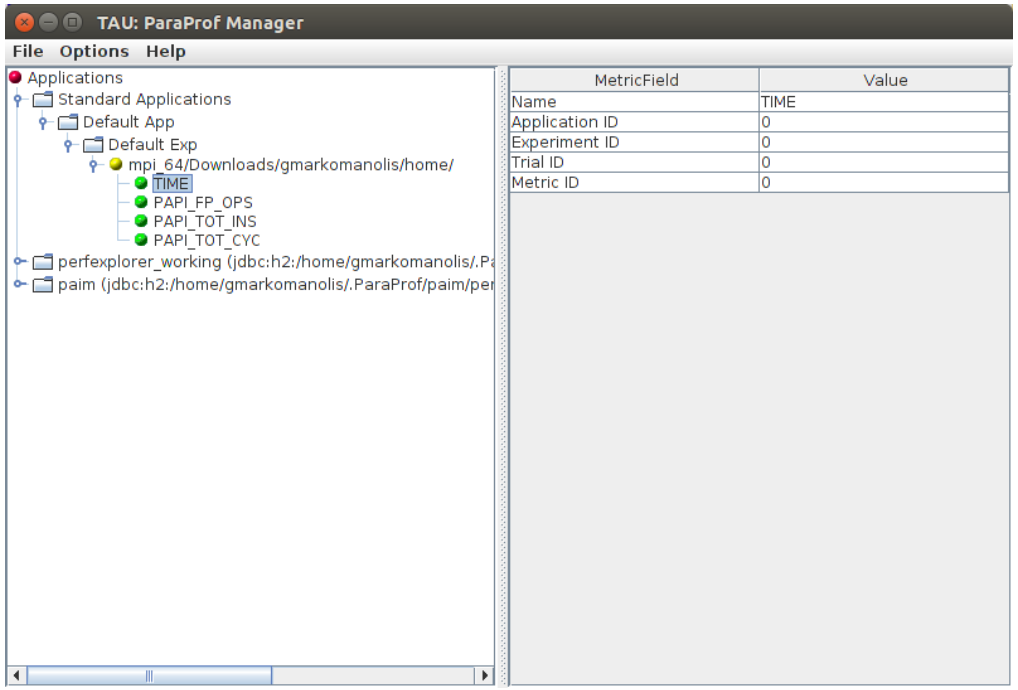
MiniWeather MPI - Execution

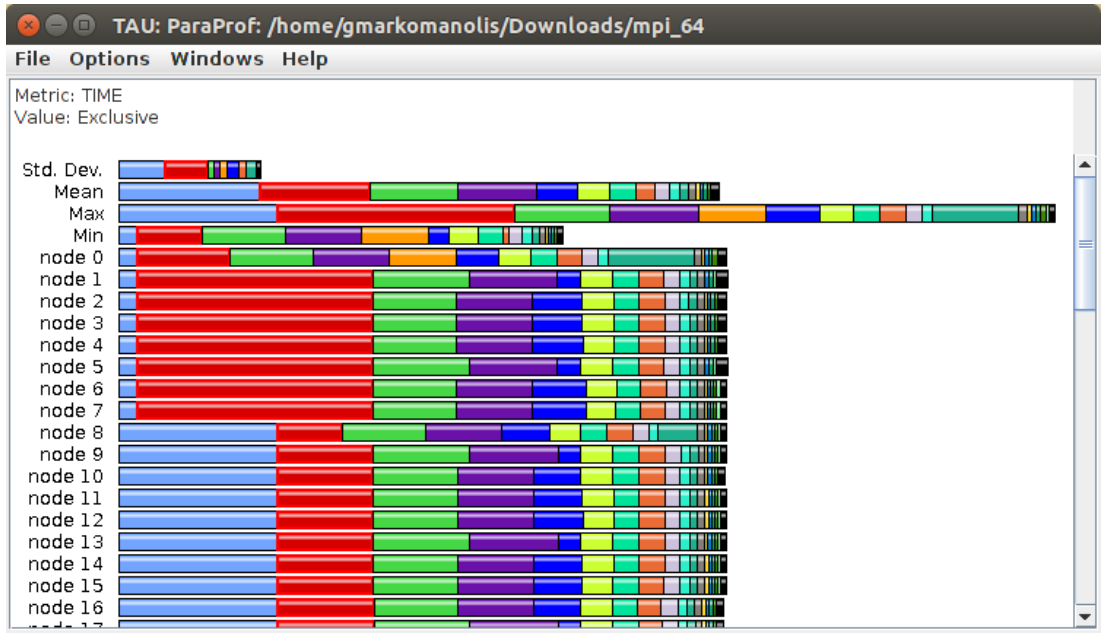
- When the execution finished, there is one folder for each TAU_METRICS declaration with the format MULTI__
- If there is no TAU_METRICS declared, then by default is used the metric TIME and the profiling files are not in a folder, in this case you need to pack them and execute paraprof:

```
summit> paraprof –pack name.ppk
summit> paraprof name.ppk
```

- To visualize the results execute paraprof (check also pprof for text mode)

MiniWeather MPI - Paraprof

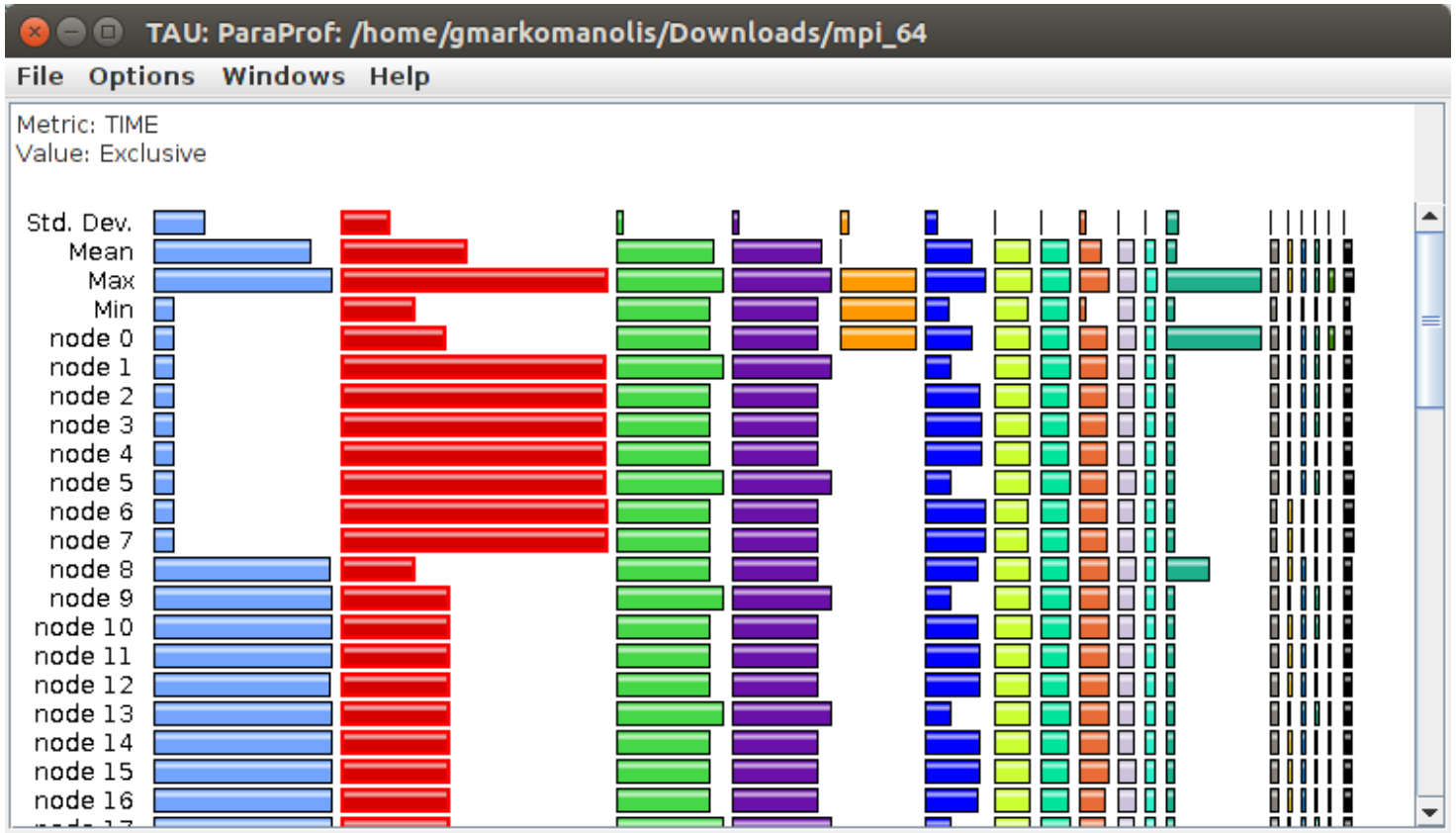




- The default metric is TIME
- Each color is a different call
- Each horizontal line is a process or Std.Dev./mean/max/min

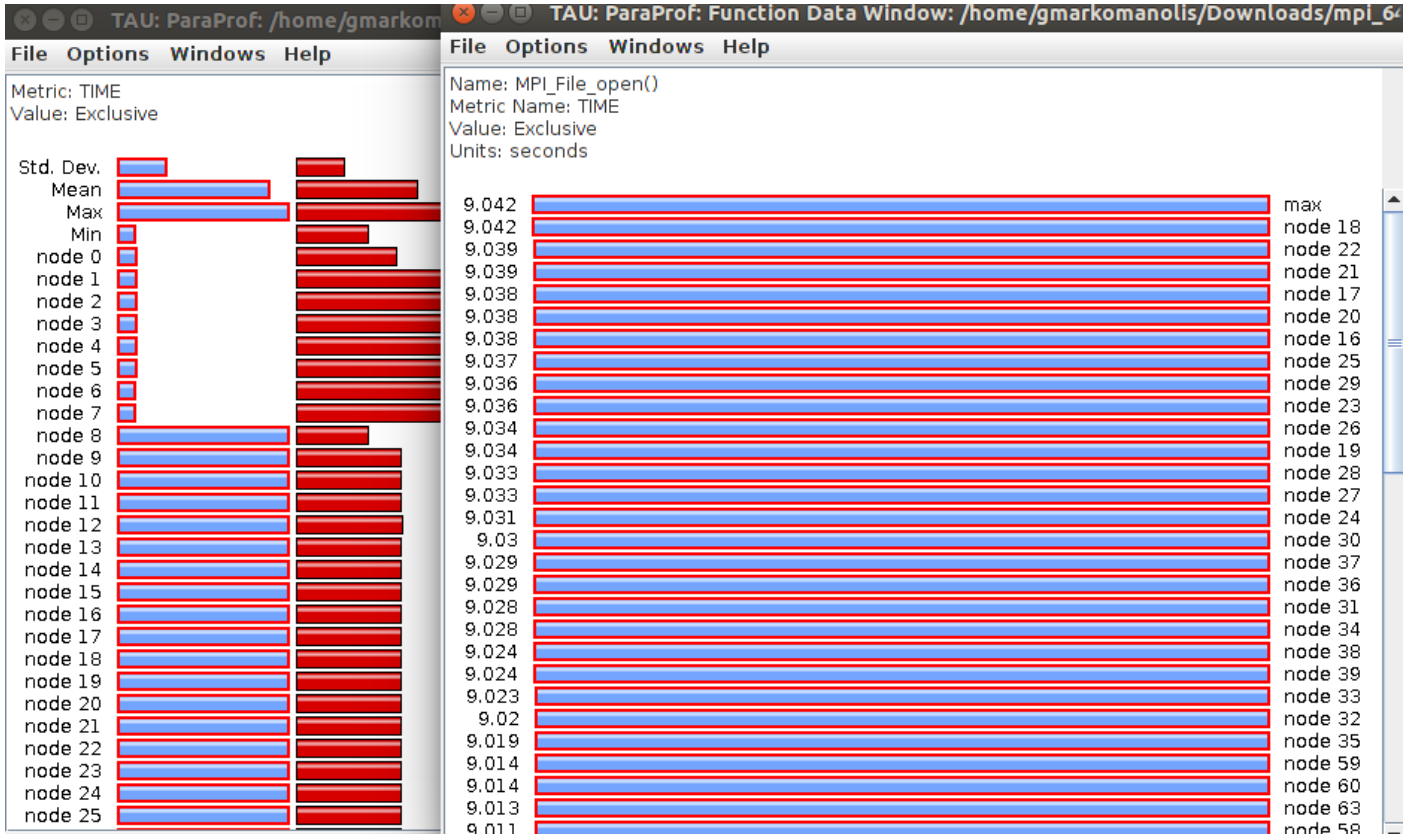
Exploring Paraprof

- Options -> Uncheck Stack Bars Together
- It is easier to check the load imbalance
- We will call this window as the main one



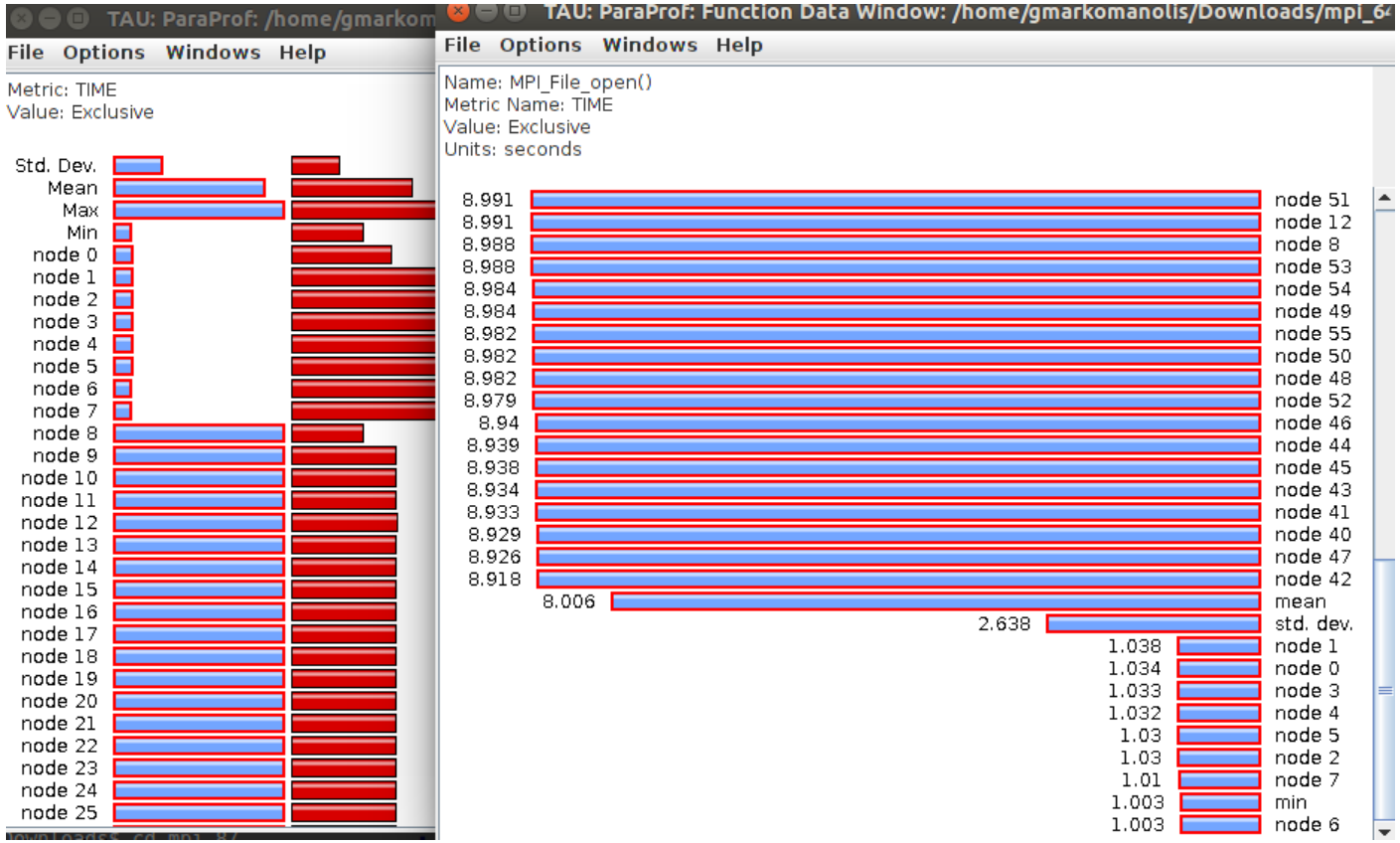
Exploring Paraprof

- Click on any color, values per process, name of routine with callpath (if activated), units in seconds, value exclusive, max, min, mean, std, values.



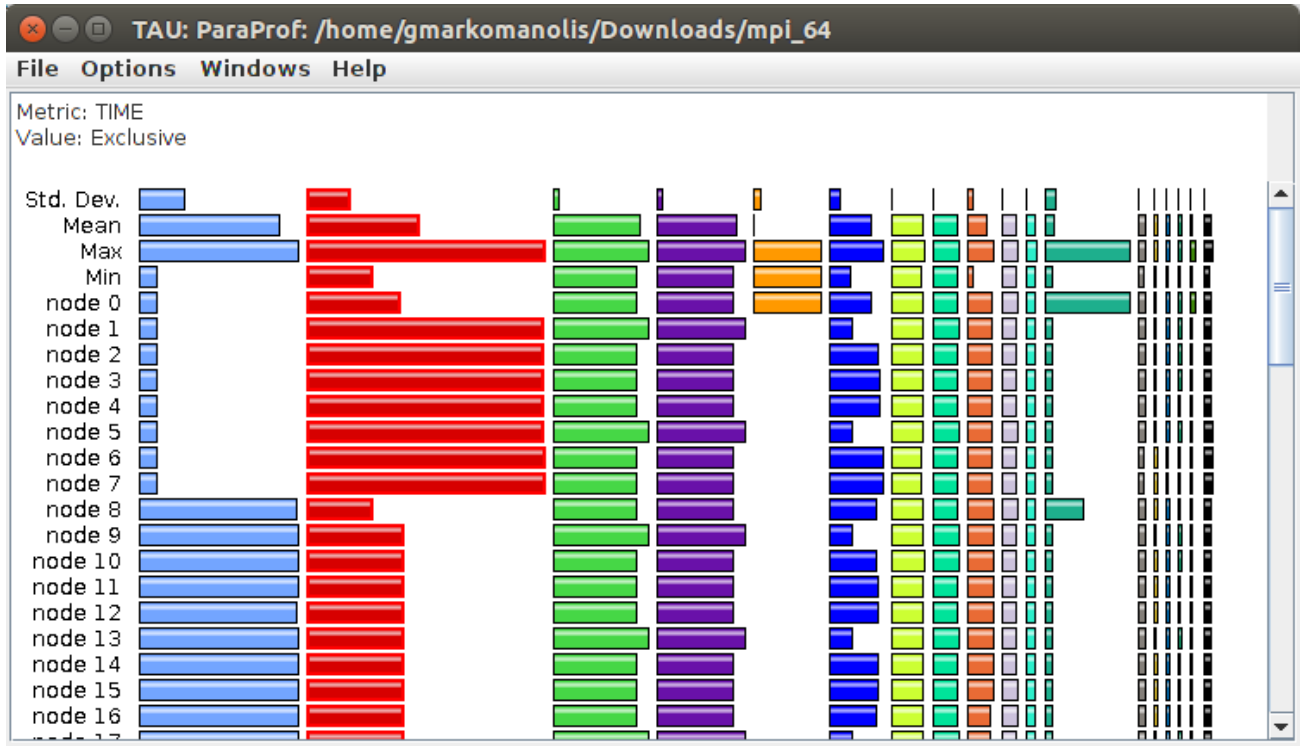
Exploring Paraprof

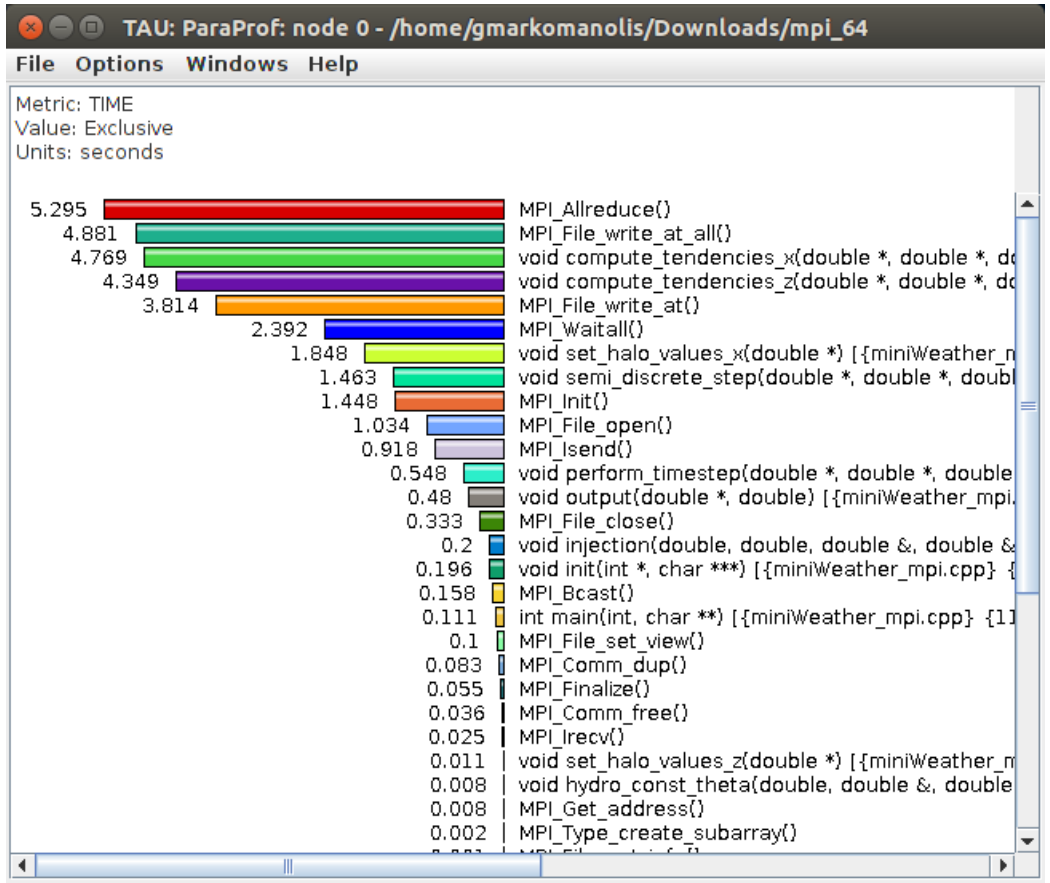
- Scroll down



Exploring Paraprof

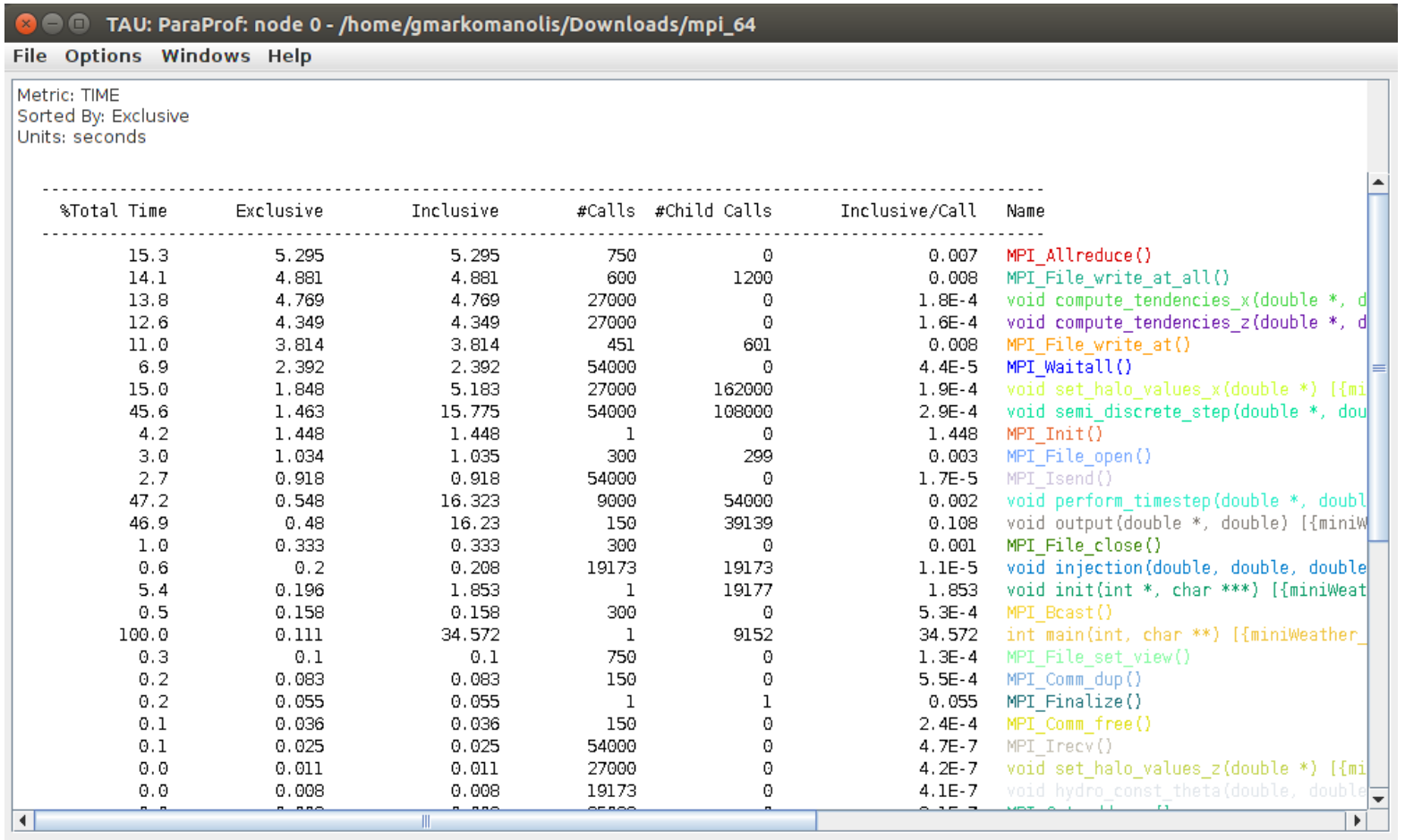
- Click on any label on the left (node 0, mean, etc.). You can see immediately which calls take more time





Paraprof – Thread Statistics Text Window

- Right click on any label of the main window, select “Show Thread Statistics Text Window”



OAK RIDGE National Laboratory

Paraprof – Thread Statistics Table

- Right click on any label of the main window, select “Show Thread Statistics Table”

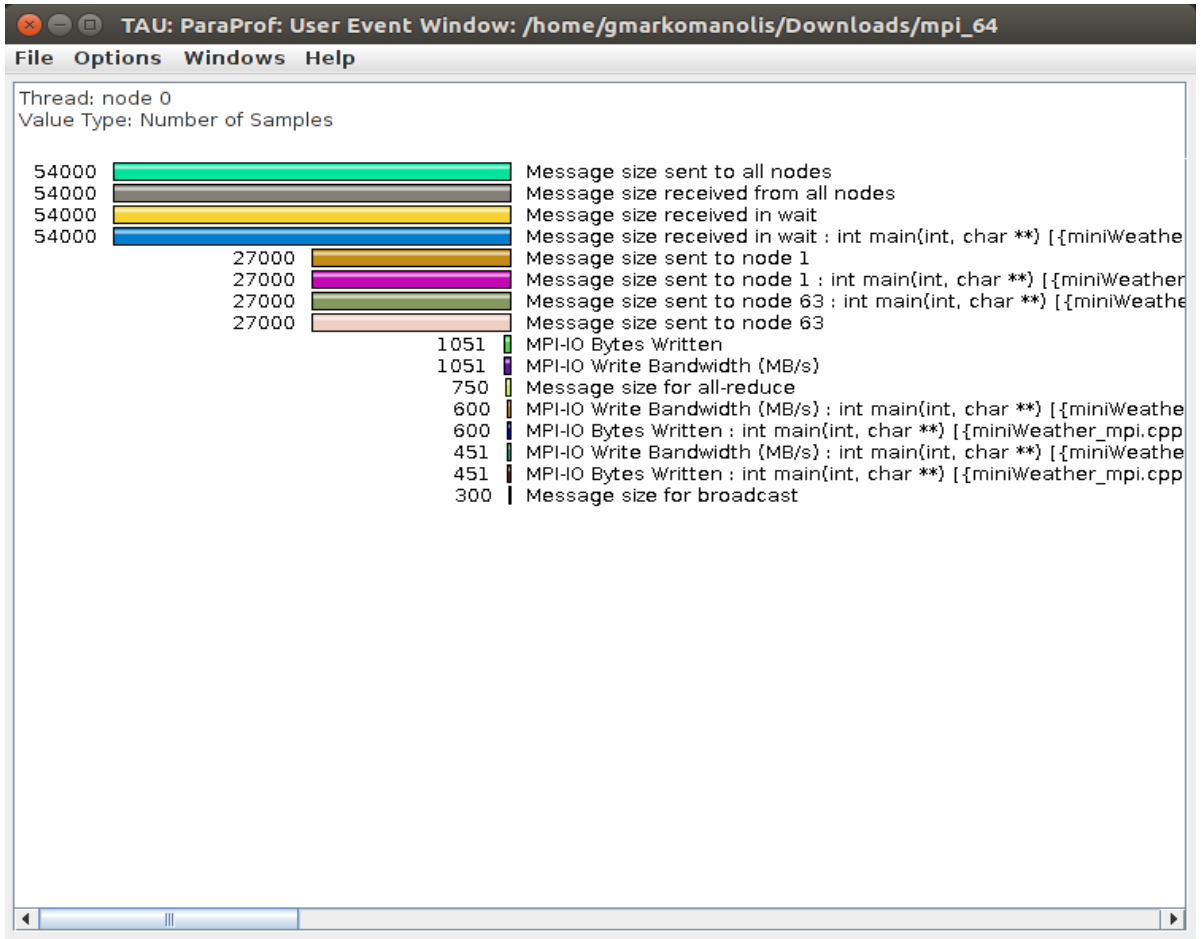
TAU: ParaProf: Statistics for: node 0 - /home/gmarkomanolis/Downloads/mpi_64

File Options Windows Help

Name △	Exclusive TIME	Inclusive TIME	Calls	Child Calls
.TAU application	0	34.572	1	1
MPI_Allreduce()	5.295	5.295	750	0
MPI_Barrier()	0.001	0.001	1	0
MPI_Bcast()	0.158	0.158	300	0
MPI_Comm_dup()	0.083	0.083	150	0
MPI_Comm_free()	0.036	0.036	150	0
MPI_Comm_get_attr()	0	0	299	0
MPI_Comm_rank()	0.001	0.001	1,504	0
MPI_Comm_size()	0	0	2	0
MPI_File_close()	0.333	0.333	300	0
MPI_File_get_info()	0.001	0.001	150	0
MPI_File_open()	1.034	1.035	300	299
MPI_File_set_view()	0.1	0.1	750	0
MPI_File_write_at()	3.814	3.814	451	601
MPI_File_write_at_all()	4.881	4.881	600	1,200
MPI_Finalize()	0.055	0.055	1	1
MPI_Get_address()	0.008	0.008	25,032	0
MPI_Get_count()	0	0	149	0
MPI_Info_free()	0.001	0.001	150	0
MPI_Init()	1.448	1.448	1	0
MPI_Irecv()	0.025	0.025	54,000	0
MPI_Isend()	0.918	0.918	54,000	0
MPI_Status_set_elements()	0.001	0.001	1,950	0
MPI_Type_commit()	0.001	0.001	1,800	0
MPI_Type_create_hvector()	0.001	0.001	600	0
MPI_Type_create_struct()	0.001	0.001	600	0
MPI_Type_create_subarray()	0.002	0.002	600	0
MPI_Type_free()	0.001	0.001	1,800	0
MPI_Type_free_keyval()	0	0	1	0

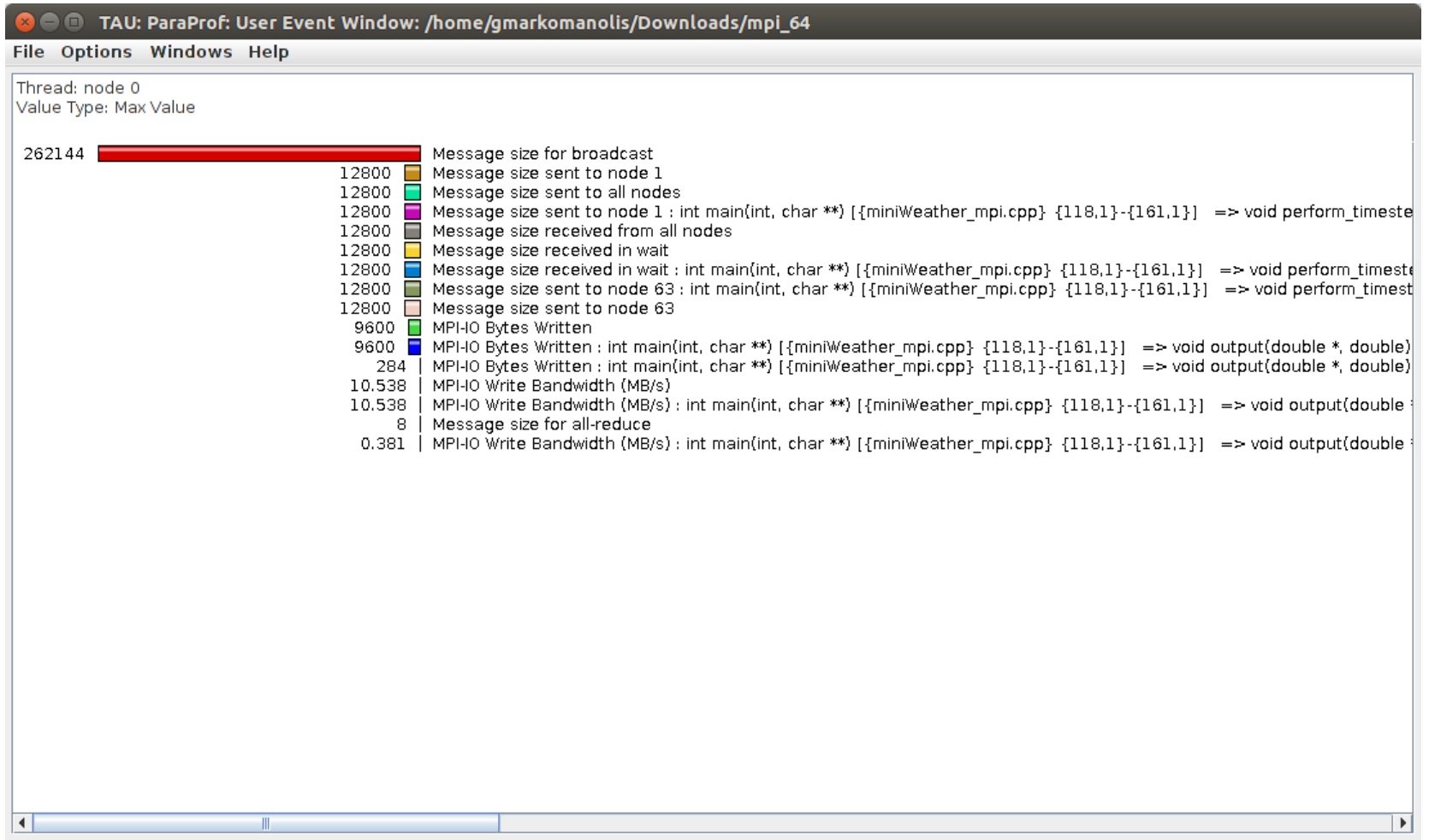
Paraprof – User Bar Chart

- Right click on any label of the main window, select “Show User Bar Chart”



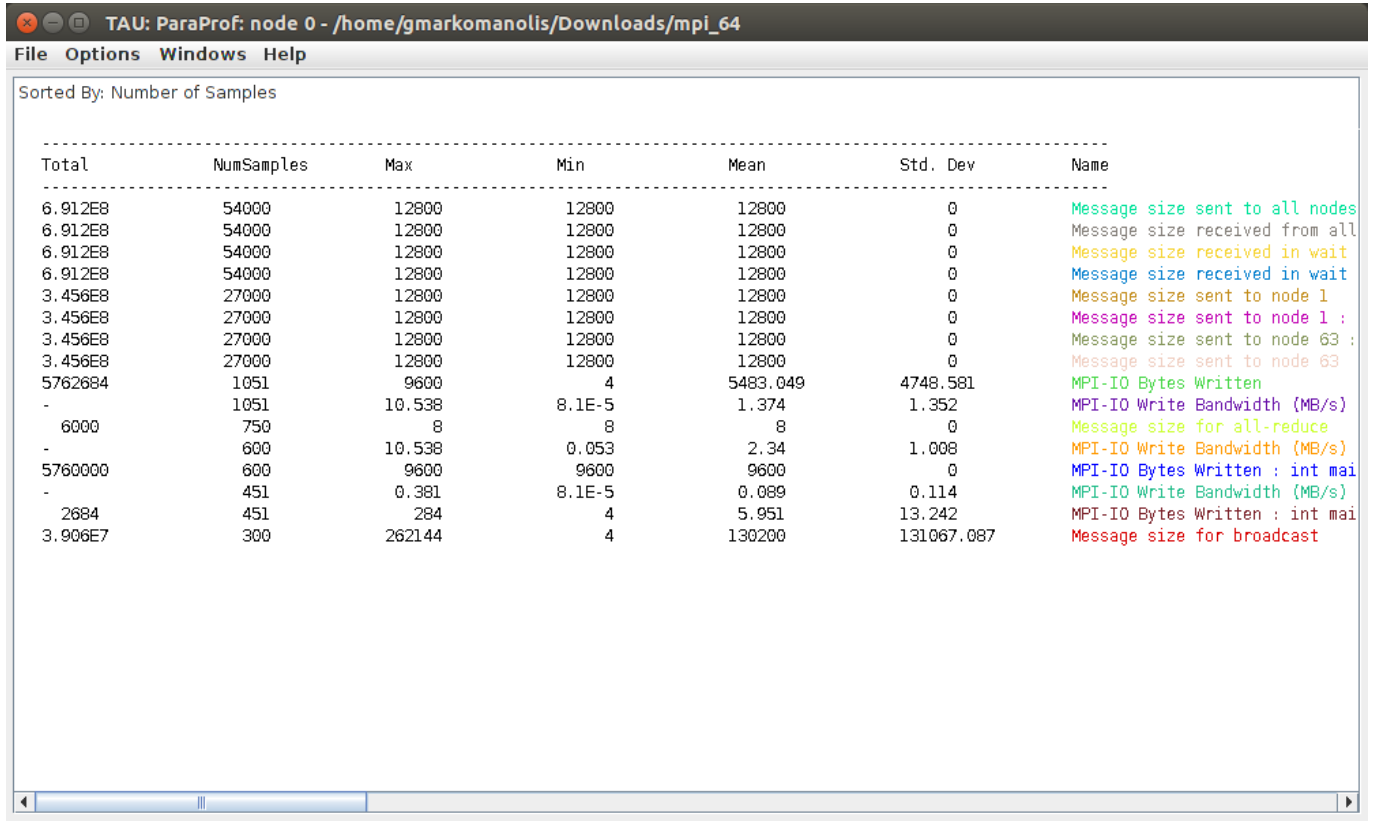
Paraprof – User Event

- Options -> Select Value Type -> Max. Value



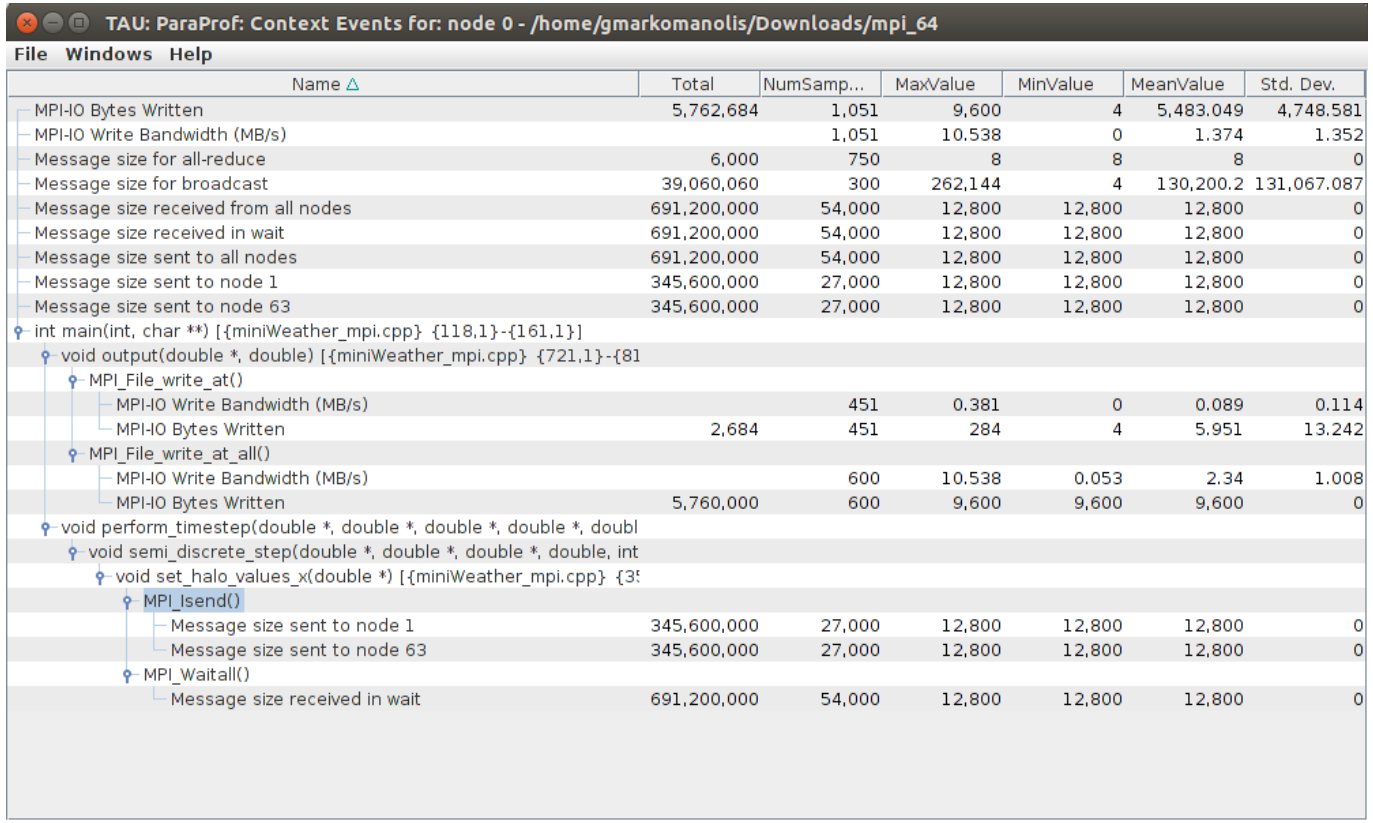
Paraprof – User Event Statistics Window

- Right click on any label of the main window, select “Show User Event Statistics Window”



Paraprof – Context Event Window

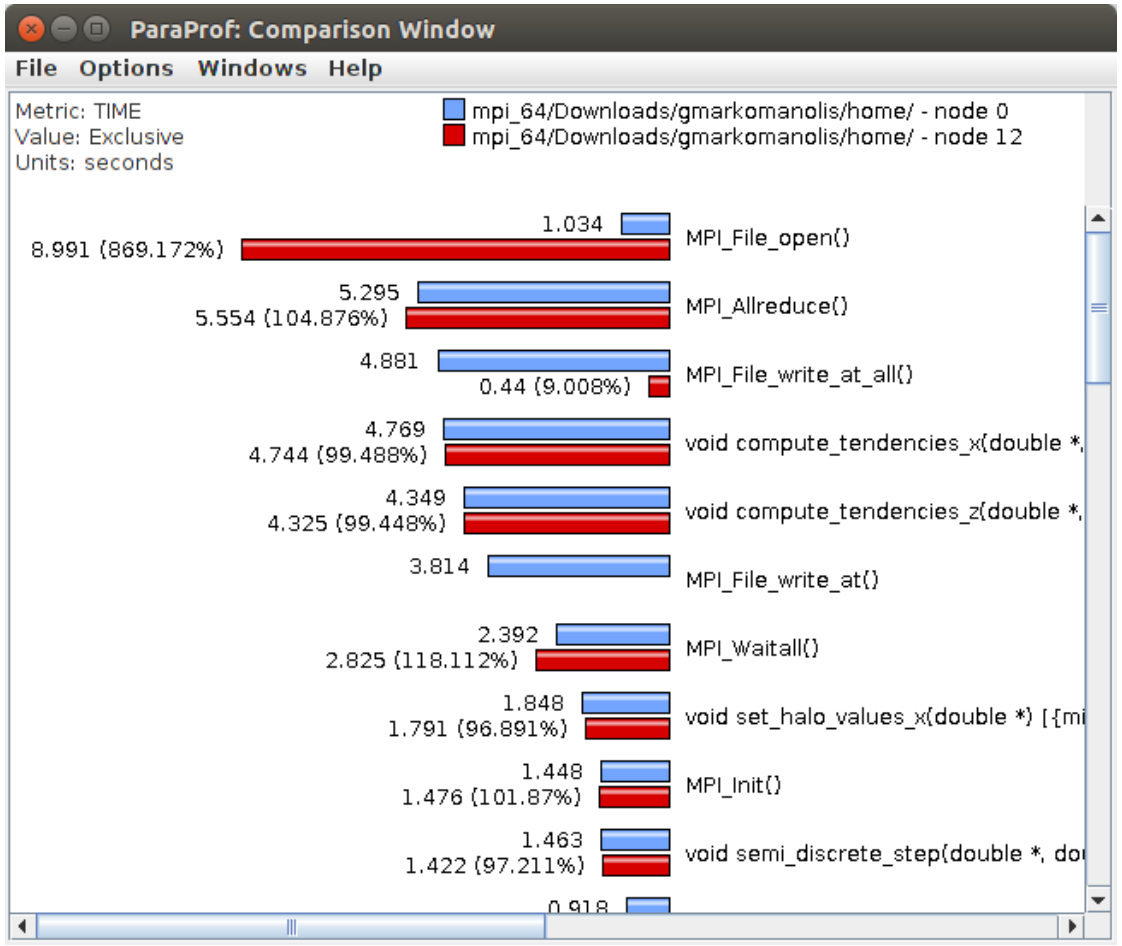
- Right click on any label of the main window, select “Show Context Event Window” (with callpath)



Name △	Total	NumSamp...	MaxValue	MinValue	MeanValue	Std. Dev.
MPI-IO Bytes Written	5,762,684	1,051	9,600	4	5,483.049	4,748.581
MPI-IO Write Bandwidth (MB/s)		1,051	10.538	0	1.374	1.352
Message size for all-reduce	6,000	750	8	8	8	0
Message size for broadcast	39,060,060	300	262,144	4	130,200.2	131,067.087
Message size received from all nodes	691,200,000	54,000	12,800	12,800	12,800	0
Message size received in wait	691,200,000	54,000	12,800	12,800	12,800	0
Message size sent to all nodes	691,200,000	54,000	12,800	12,800	12,800	0
Message size sent to node 1	345,600,000	27,000	12,800	12,800	12,800	0
Message size sent to node 63	345,600,000	27,000	12,800	12,800	12,800	0
int main(int, char **) [{miniWeather_mpi.cpp} {118,1}-{161,1}]						
void output(double *, double) [{miniWeather_mpi.cpp} {721,1}-{81						
MPI_File_write_at()						
MPI-IO Write Bandwidth (MB/s)		451	0.381	0	0.089	0.114
MPI-IO Bytes Written	2,684	451	284	4	5.951	13.242
MPI_File_write_at_all()						
MPI-IO Write Bandwidth (MB/s)		600	10.538	0.053	2.34	1.008
MPI-IO Bytes Written	5,760,000	600	9,600	9,600	9,600	0
void perform_timestep(double *, double *, double *, double *, doubl						
void semi_discrete_step(double *, double *, double *, double, int						
void set_halo_values_x(double *) [{miniWeather_mpi.cpp} {3						
MPI_Isend()						
Message size sent to node 1	345,600,000	27,000	12,800	12,800	12,800	0
Message size sent to node 63	345,600,000	27,000	12,800	12,800	12,800	0
MPI_Waitall()						
Message size received in wait	691,200,000	54,000	12,800	12,800	12,800	0

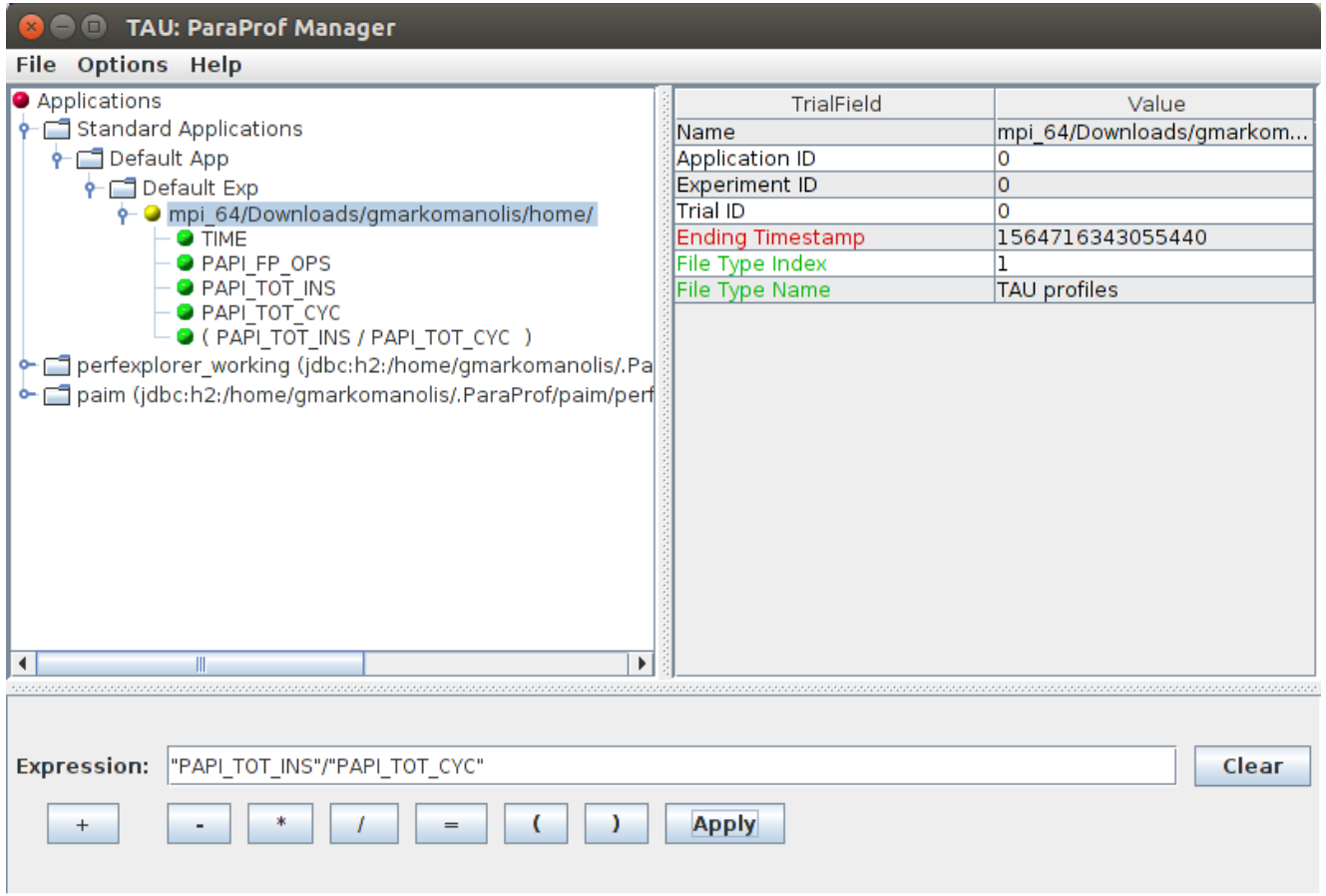
Paraprof – Add Thread to Comparison Window

- Right click on node 0 and select “Add Thread to Comparison Window”, similar for node 12. You could use any number of processes that you prefer.



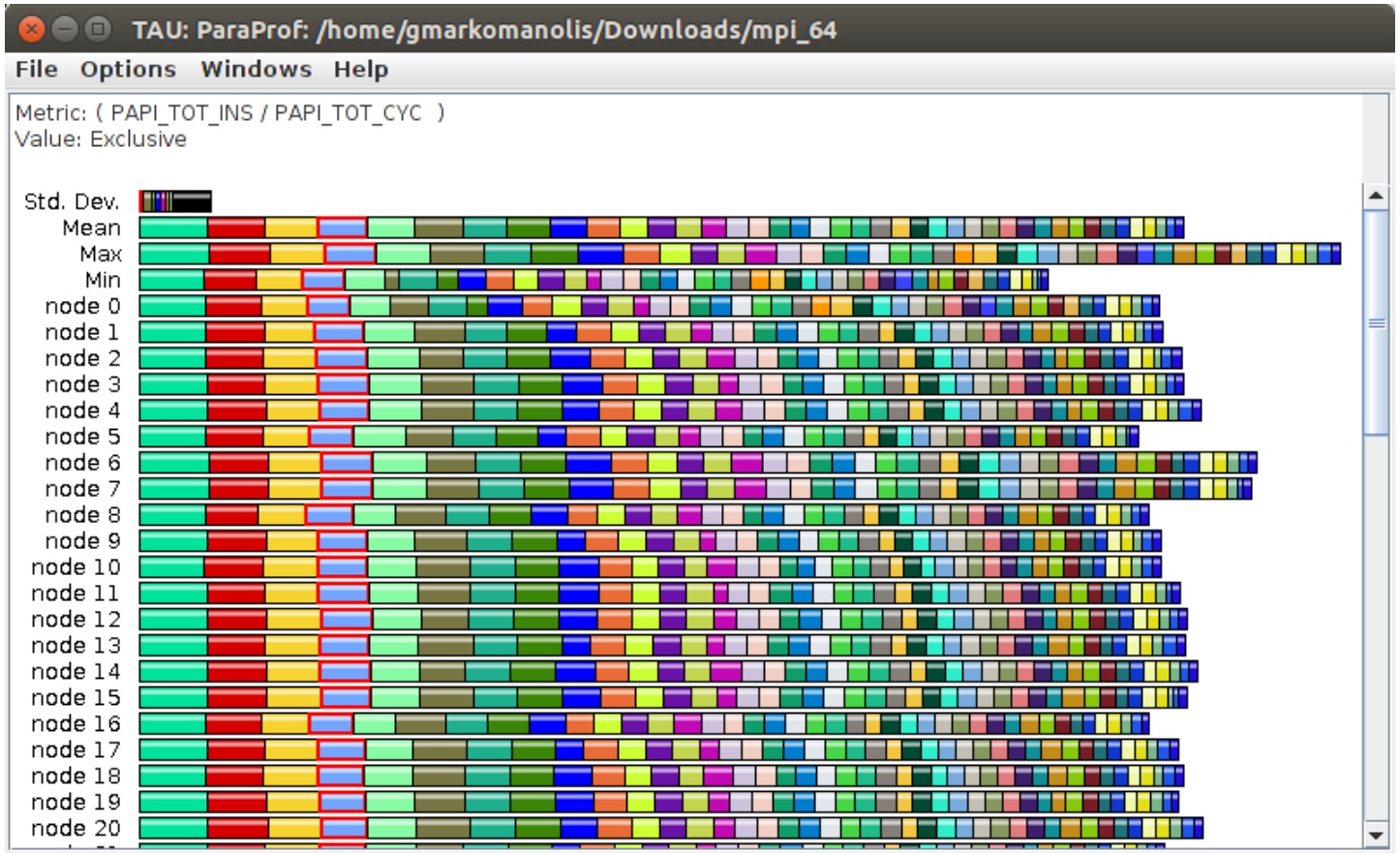
Derived Metrics

Options -> Show Derived Metric Panel, select the metrics and then operator and then click Apply. Then uncheck the Show Derived Metric



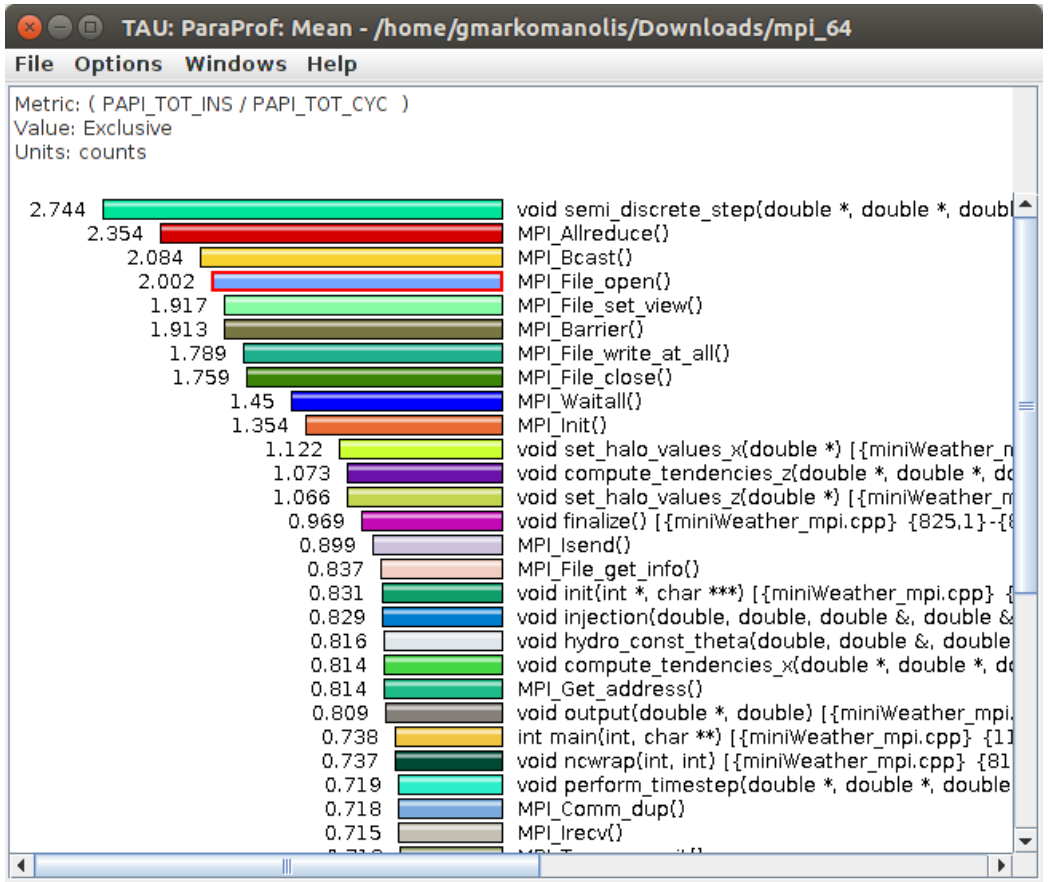
Paraprof - IPC

- Click on the new metric, PAPI_TOT_INS/PAPI_TOT_CYC



Paraprof – Mean IPC

- Click on the label mean



Paraprof – IPC for thread 0

- From the main window with the PAPI_TOT_INS/PAPI_TOT_CYC metric, right click on node 0 and select Show Thread Statistics Table

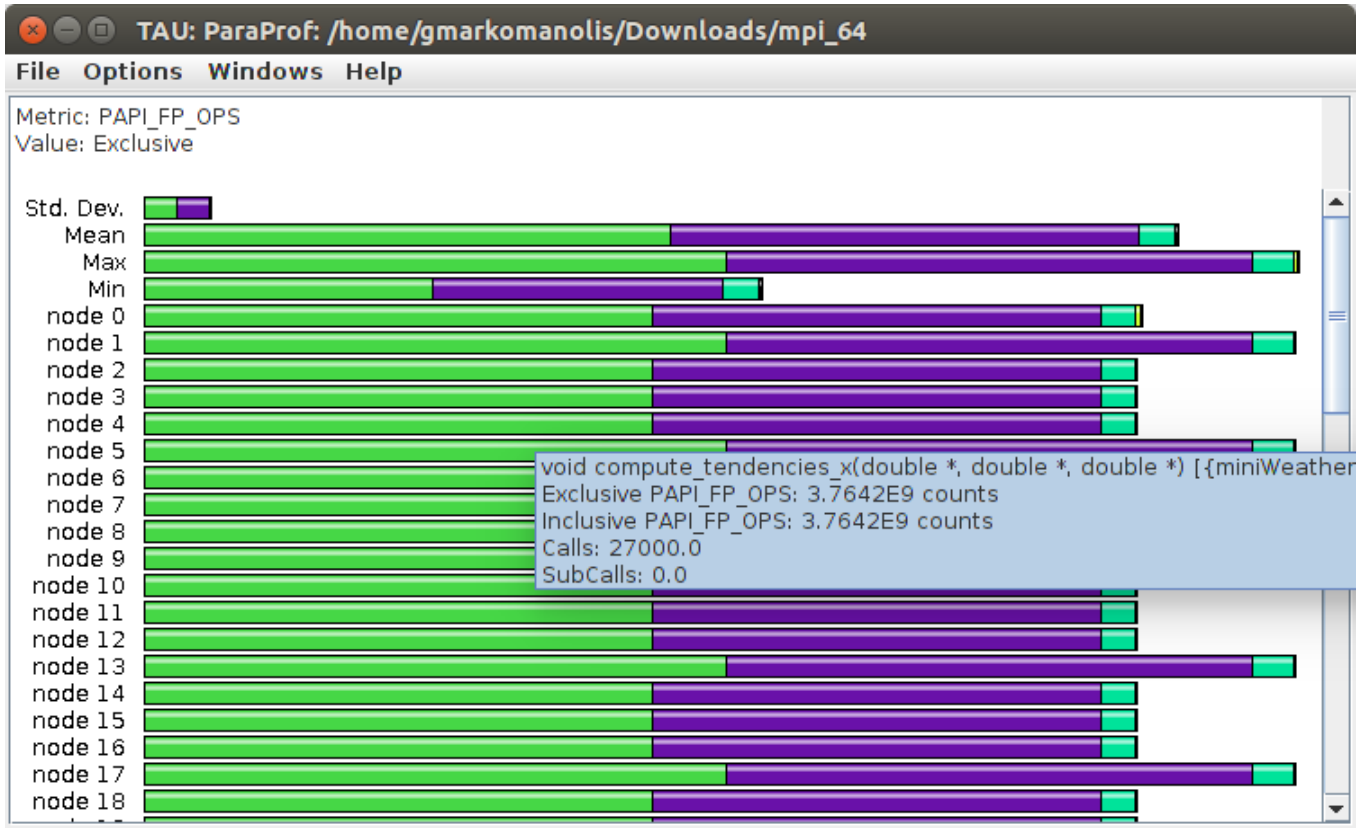
TAU: ParaProf: Statistics for: node 0 - /home/gmarkomanolis/Downloads/mpi_64

File Options Windows Help

Name	Exclusive (PAPI_TOT_I...	Inclusive (PAPI_TOT_IN...	Calls	Child Calls
MPI_Finalize()	0.588	0.588	1	1
MPI_Get_address()	0.791	0.791	25,032	0
MPI_Get_count()	0.612	0.612	149	0
MPI_Info_free()	0.47	0.47	150	0
MPI_Init()	1.197	1.197	1	0
MPI_Irecv()	0.73	0.73	54,000	0
MPI_Isend()	0.869	0.869	54,000	0
MPI_Status_set_elements()	0.581	0.581	1,950	0
MPI_Type_commit()	0.73	0.73	1,800	0
MPI_Type_create_hvector()	0.7	0.7	600	0
MPI_Type_create_struct()	0.683	0.683	600	0
MPI_Type_create_subarray()	0.624	0.624	600	0
MPI_Type_free()	0.685	0.685	1,800	0
MPI_Type_free_keyval()	0.425	0.425	1	0
MPI_Type_size()	0.721	0.721	750	0
MPI_Waitall()	1.406	1.406	54,000	0
int main(int, char **) [{miniWeather_mpi.cpp} {118,1}-{161,1}]	0.956	1.421	1	9,152
void compute_tendencies_x(double *, double *, double *) [{miniWe	0.812	0.812	27,000	0
void compute_tendencies_z(double *, double *, double *) [{miniWe	1.069	1.069	27,000	0
void finalize() [{miniWeather_mpi.cpp} {825,1}-{841,1}]	0.619	0.588	1	1
void hydro_const_theta(double, double &, double &) [{miniWeathe	0.825	0.825	19,173	0
void init(int *, char ***) [{miniWeather_mpi.cpp} {448,1}-{583,1}]	0.838	1.047	1	19,177
void injection(double, double, double &, double &, double &, doubl	0.835	0.831	19,173	19,173
void ncwrap(int, int) [{miniWeather_mpi.cpp} {816,1}-{822,1}]	0.755	0.755	2,104	0
void output(double *, double) [{miniWeather_mpi.cpp} {721,1}-{8	0.787	1.97	150	39,139
void perform_timestep(double *, double *, double *, double *, doub	0.728	1.075	9,000	54,000
void semi_discrete_step(double *, double *, double *, double, int, d	2.715	1.076	54,000	108,000
void set_halo_values_x(double *) [{miniWeather_mpi.cpp} {351,1}-	1.178	1.248	27,000	162,000
void set_halo_values_z(double *) [{miniWeather_mpi.cpp} {411,1}-	1.044	1.044	27,000	0

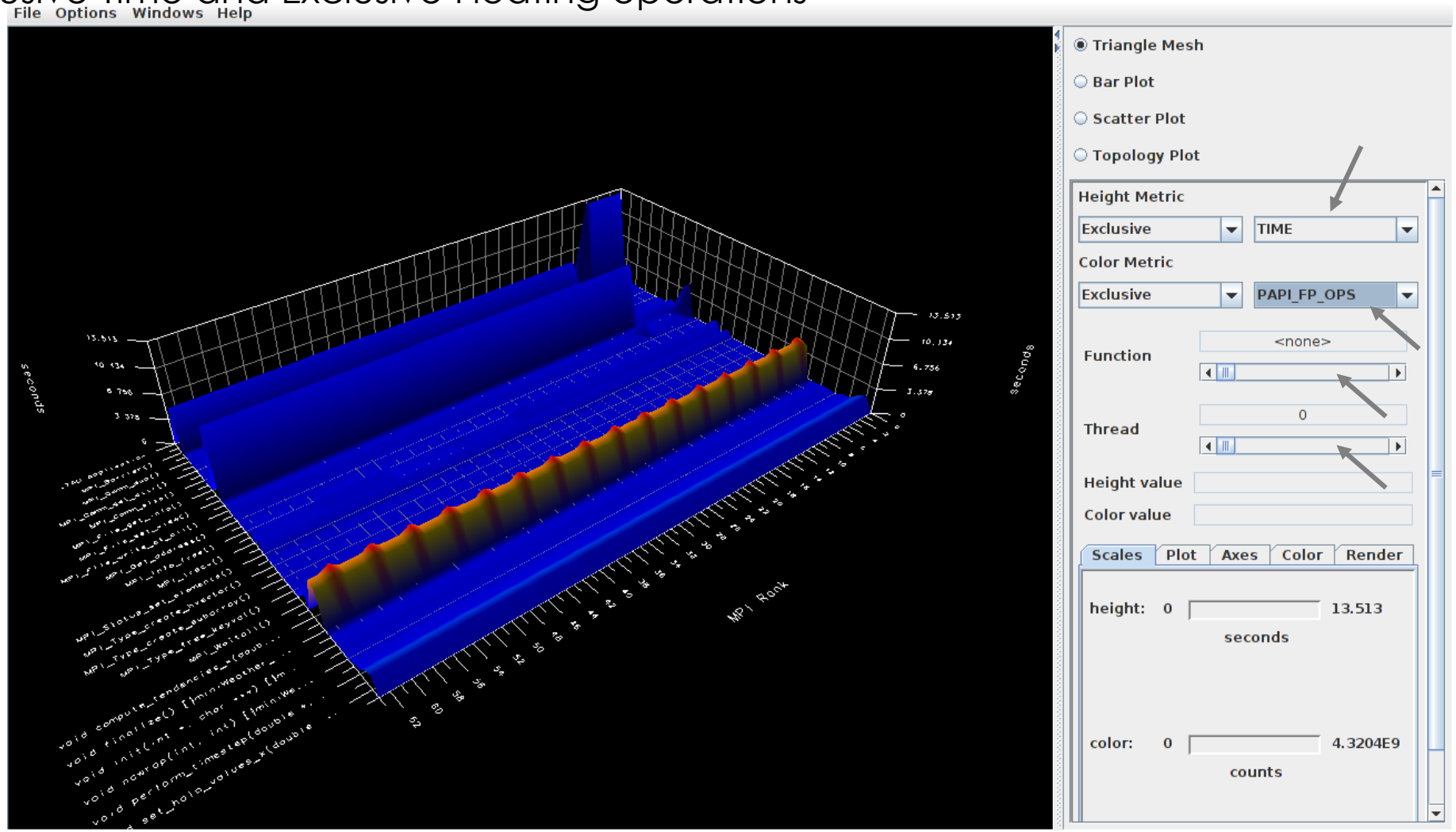
Paraprof

- From the main window select Options -> Select Metric… -> Exclusive -> PAPI_FP_OPS



Paraprof – 3D Visualization

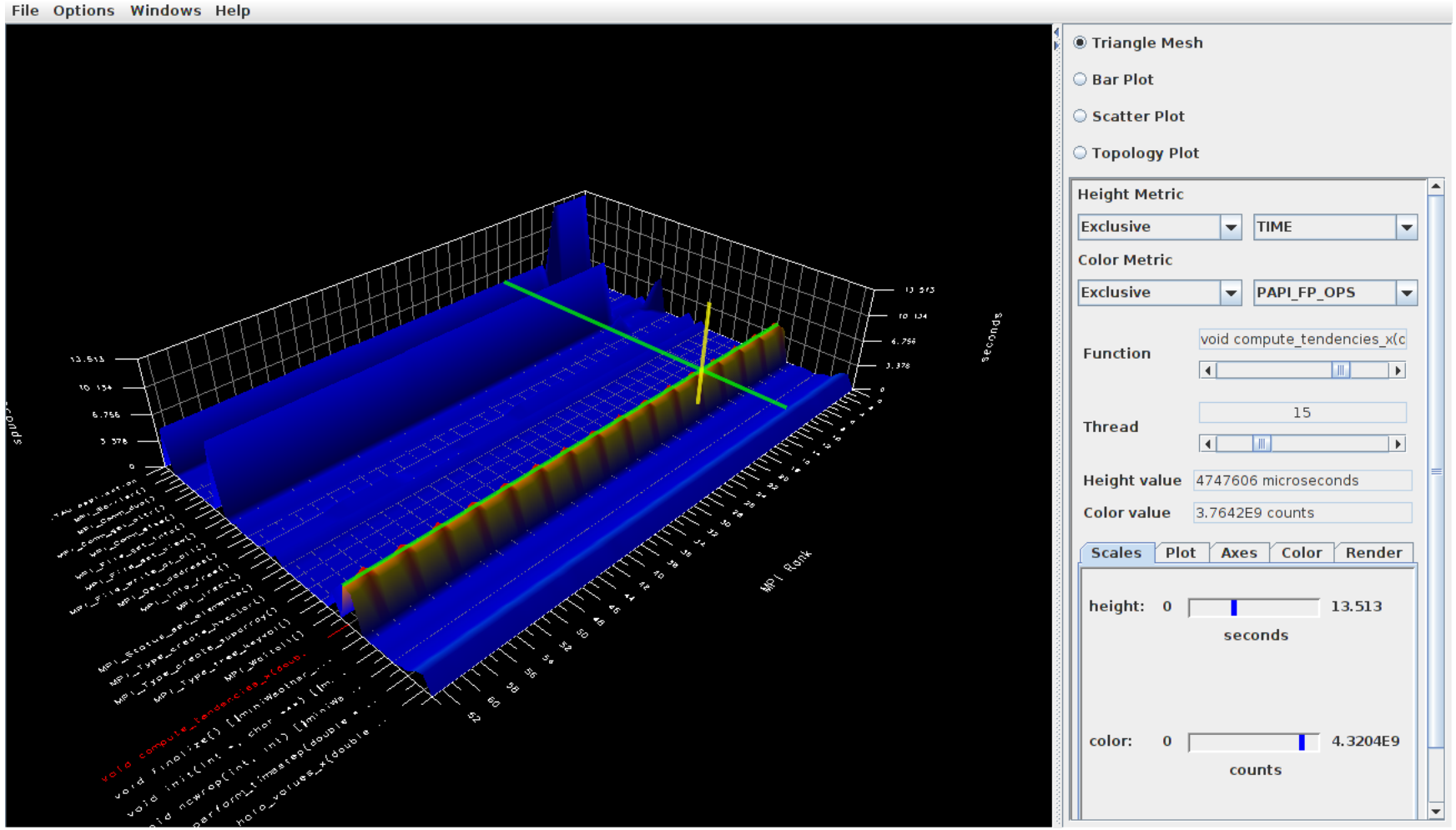
- Menu Windows -> 3D Visualization (3D demands OpenGL)
- Exclusive Time and Exclusive Floating operations



OAK RIDGE National Laboratory

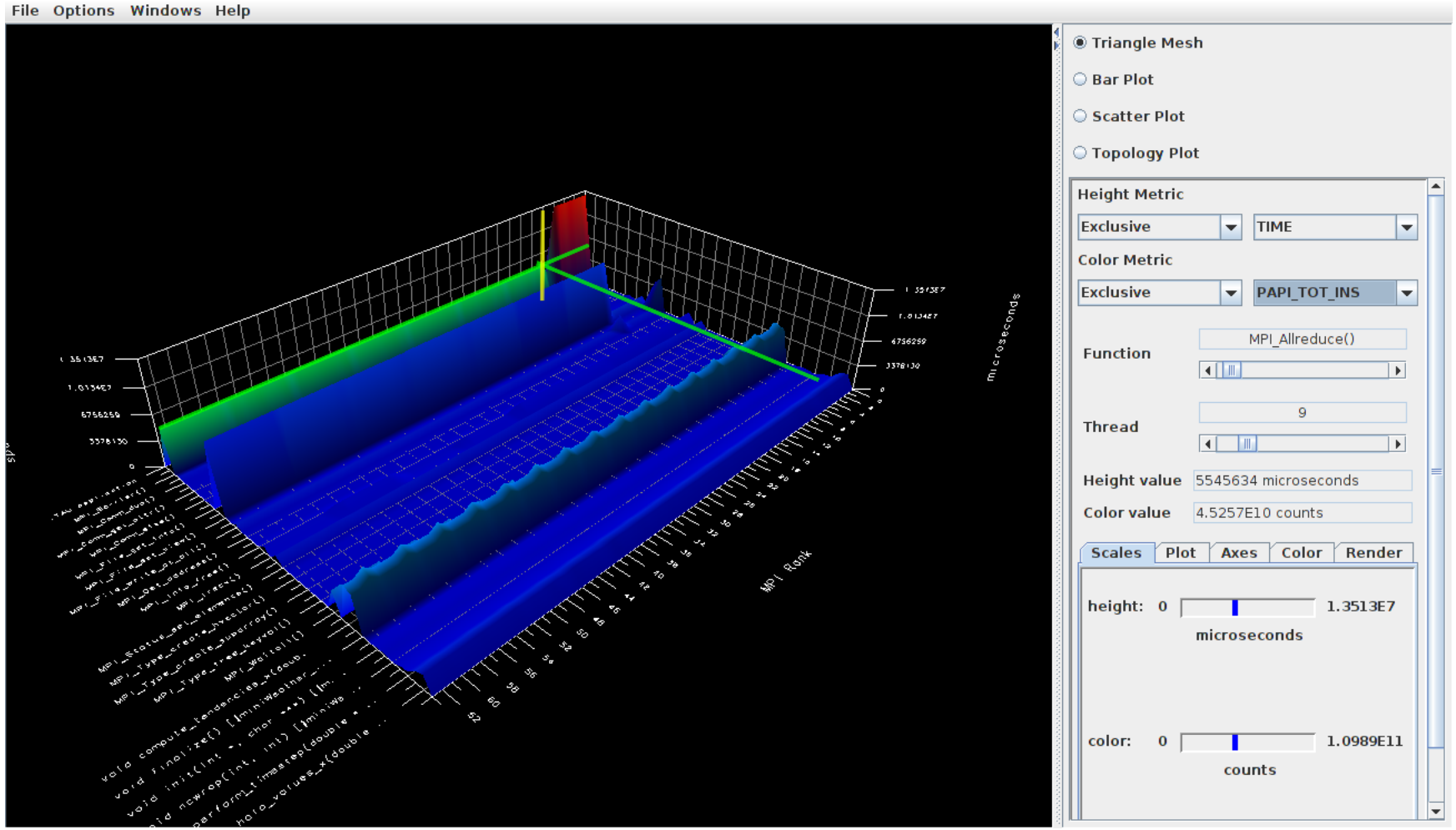
Paraprof – 3D Visualization

- Menu Windows -> 3D Visualization (3D demands OpenGL)
- Specific routine and thread



Paraprof – 3D Visualization

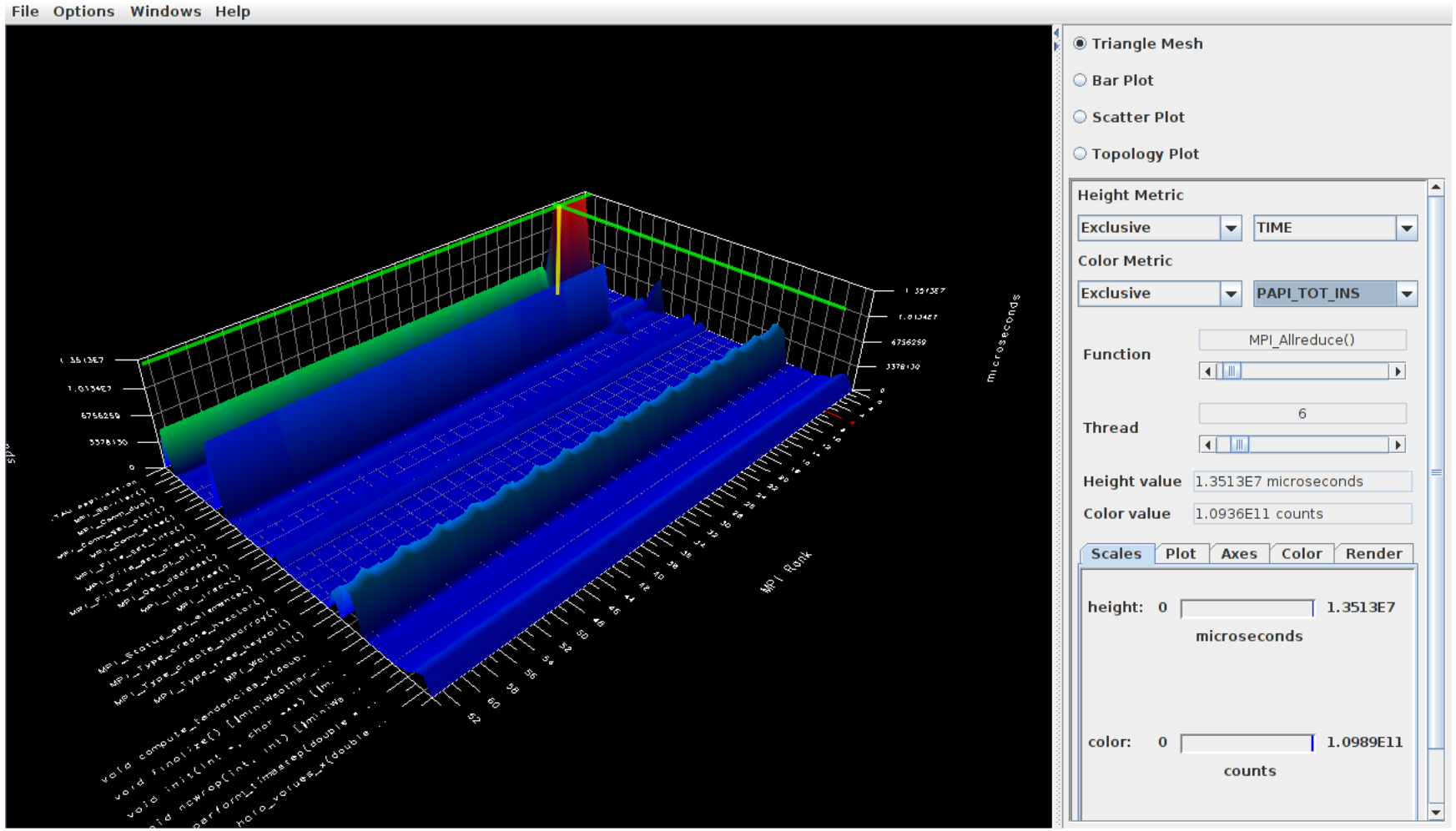
- Menu Windows -> 3D Visualization (3D demands OpenGL)
- Exclusive time and total instructions



OAK RIDGE National Laboratory

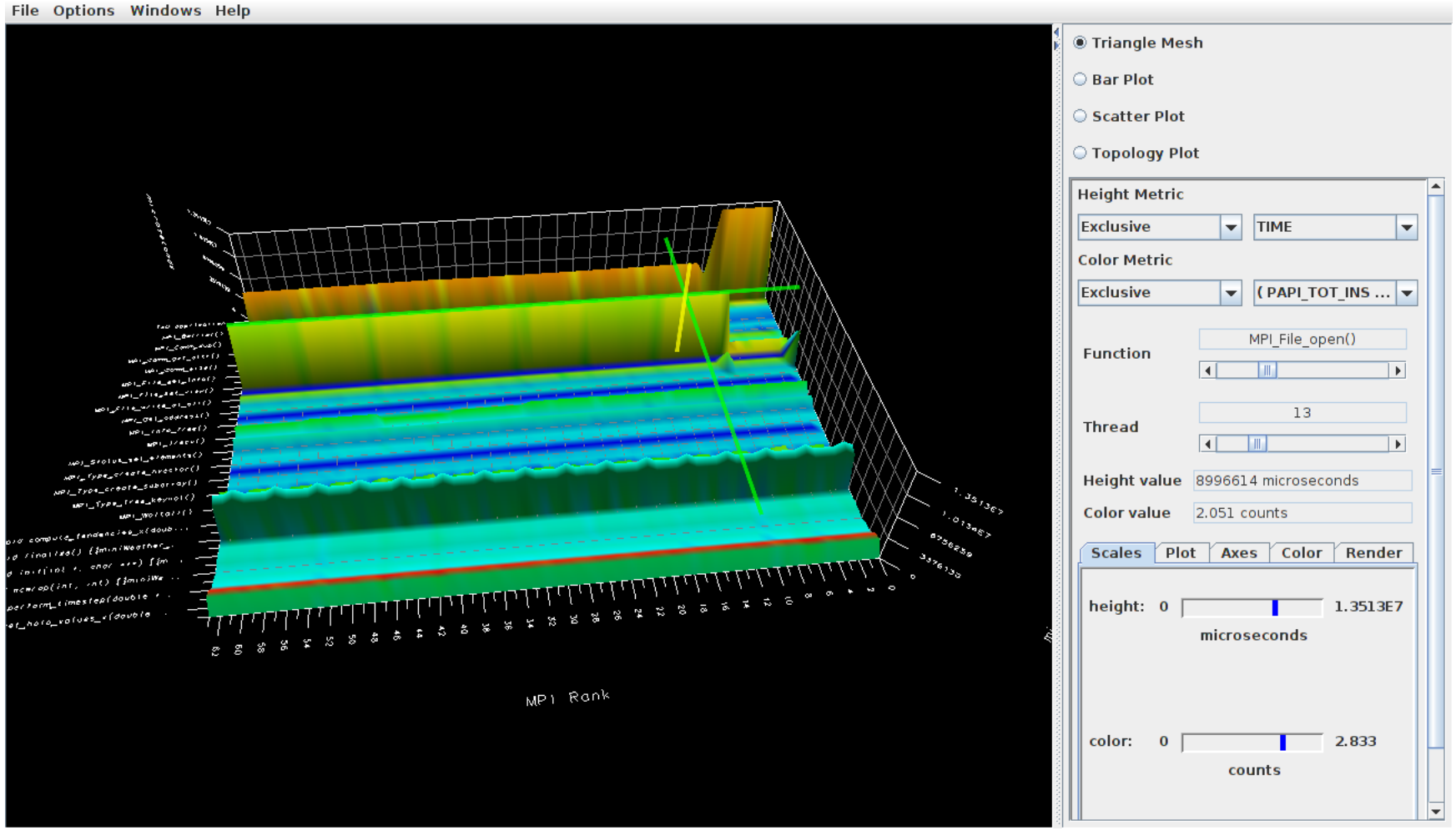
Paraprof – 3D Visualization

Menu Windows -> 3D Visualization (3D demands OpenGL)



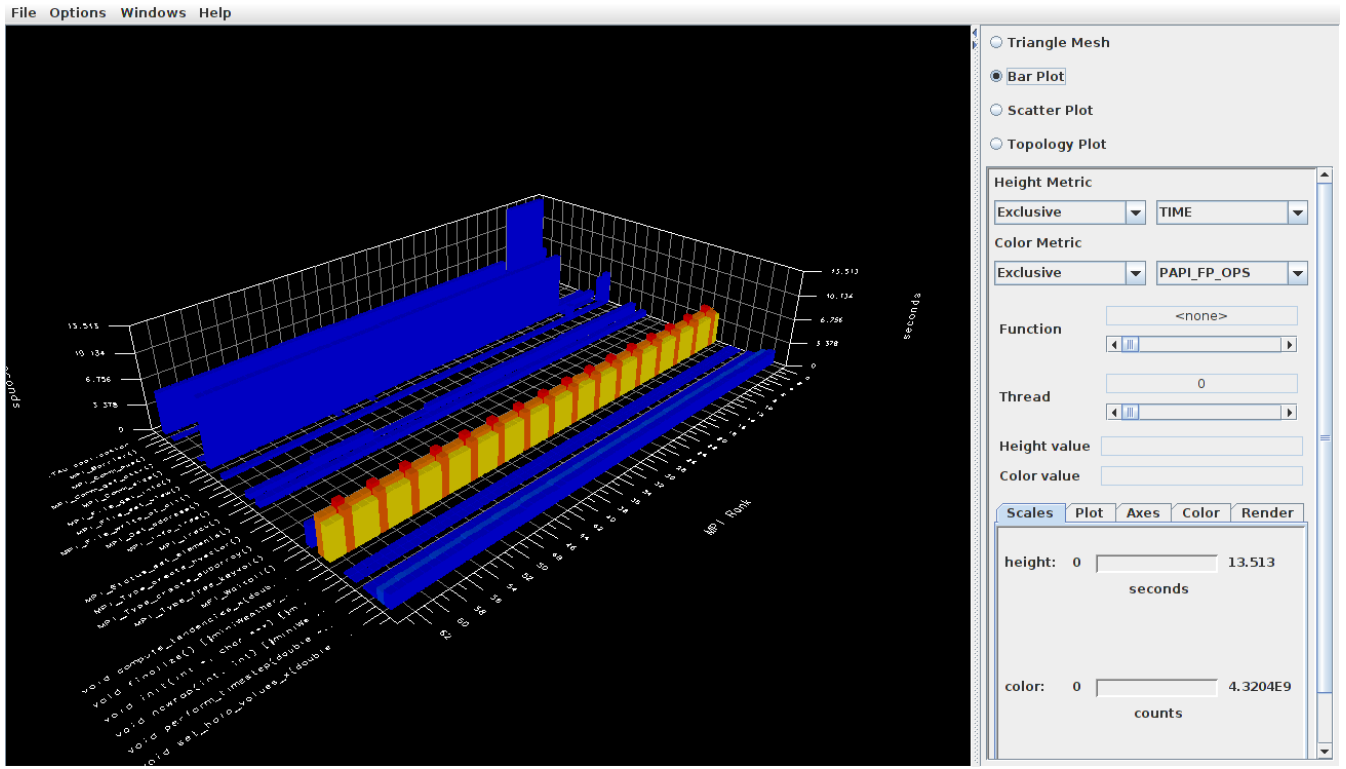
Paraprof – 3D Visualization

- Menu Windows -> 3D Visualization (3D demands OpenGL)
- Exclusive time and instructions per cycle



Paraprof – 3D Visualization

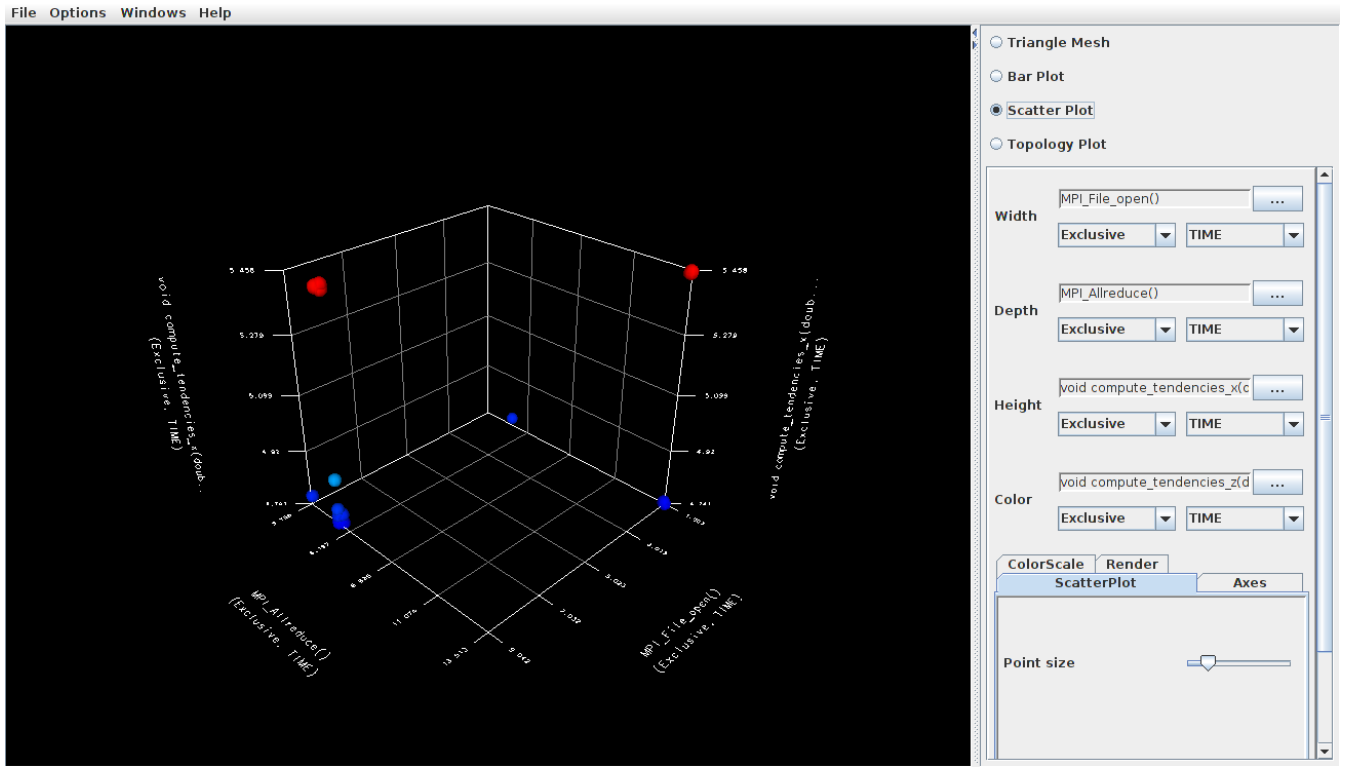
- Menu Windows -> 3D Visualization (3D demands OpenGL)
- Bar Plot



OAK RIDGE National Laboratory

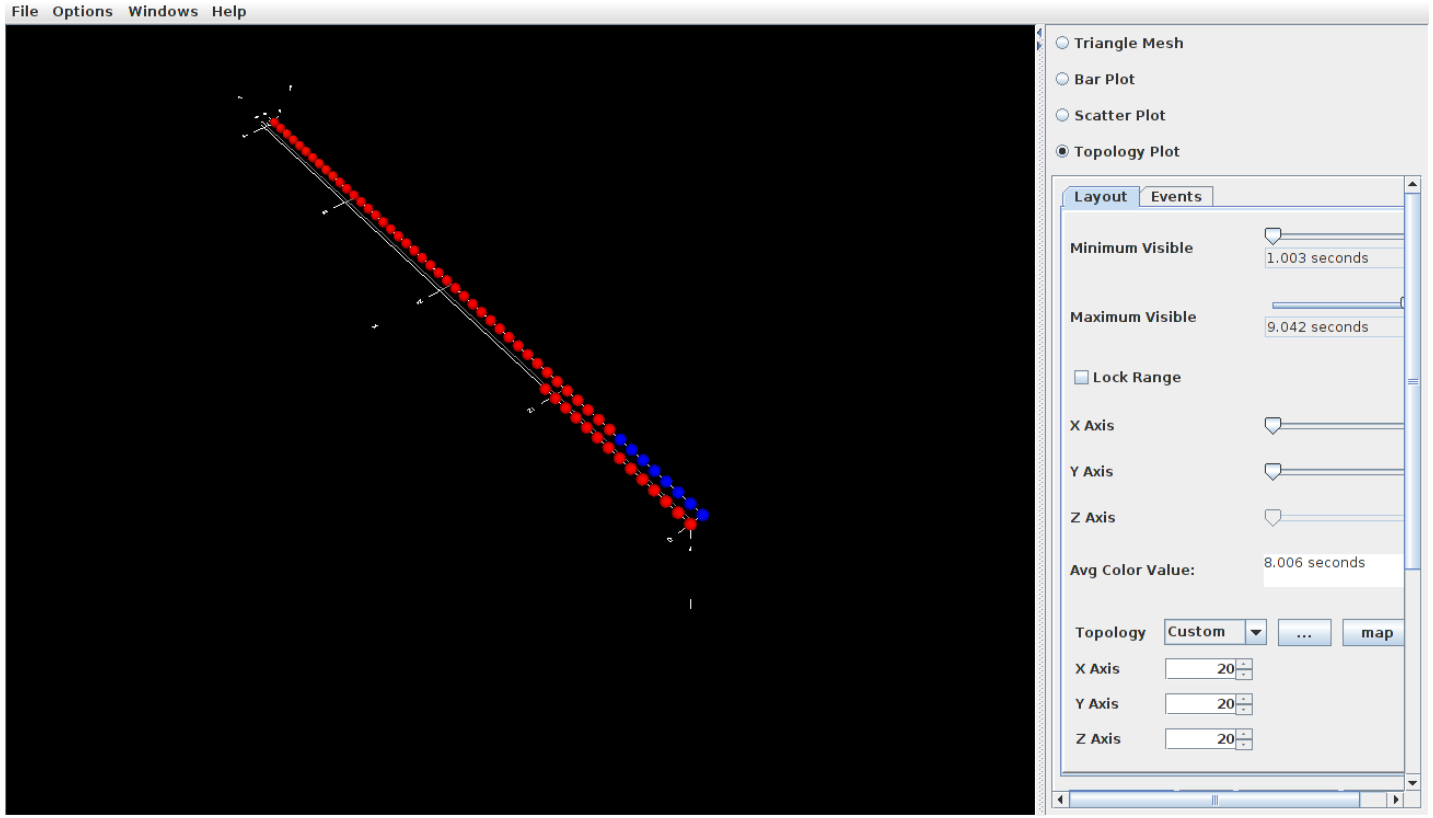
Paraprof – 3D Visualization

- Menu Windows -> 3D Visualization (3D demands OpenGL)
- Scatter Plot



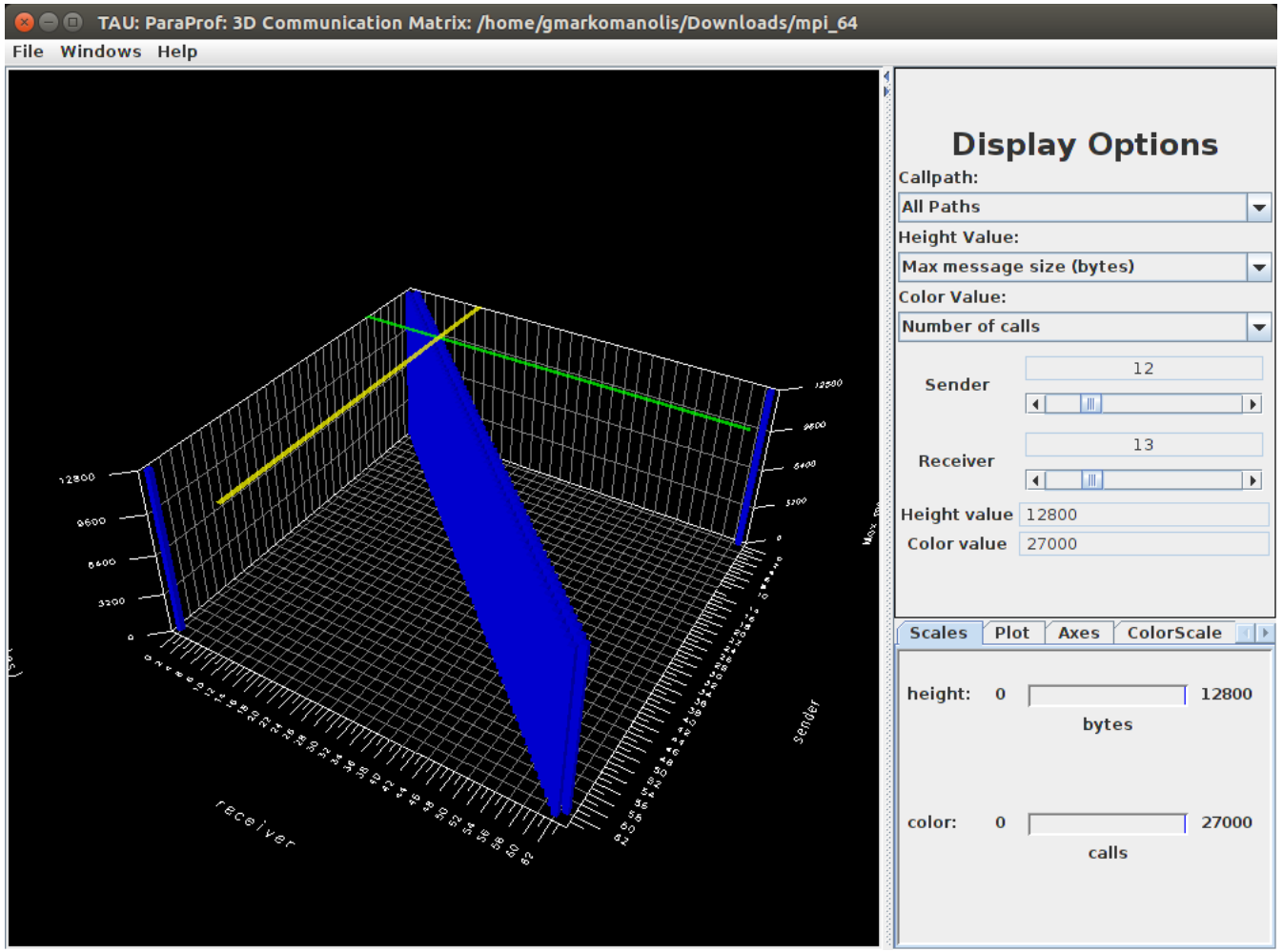
Paraprof – 3D Visualization

- Menu Windows -> 3D Visualization (3D demands OpenGL)
- Topology Plot



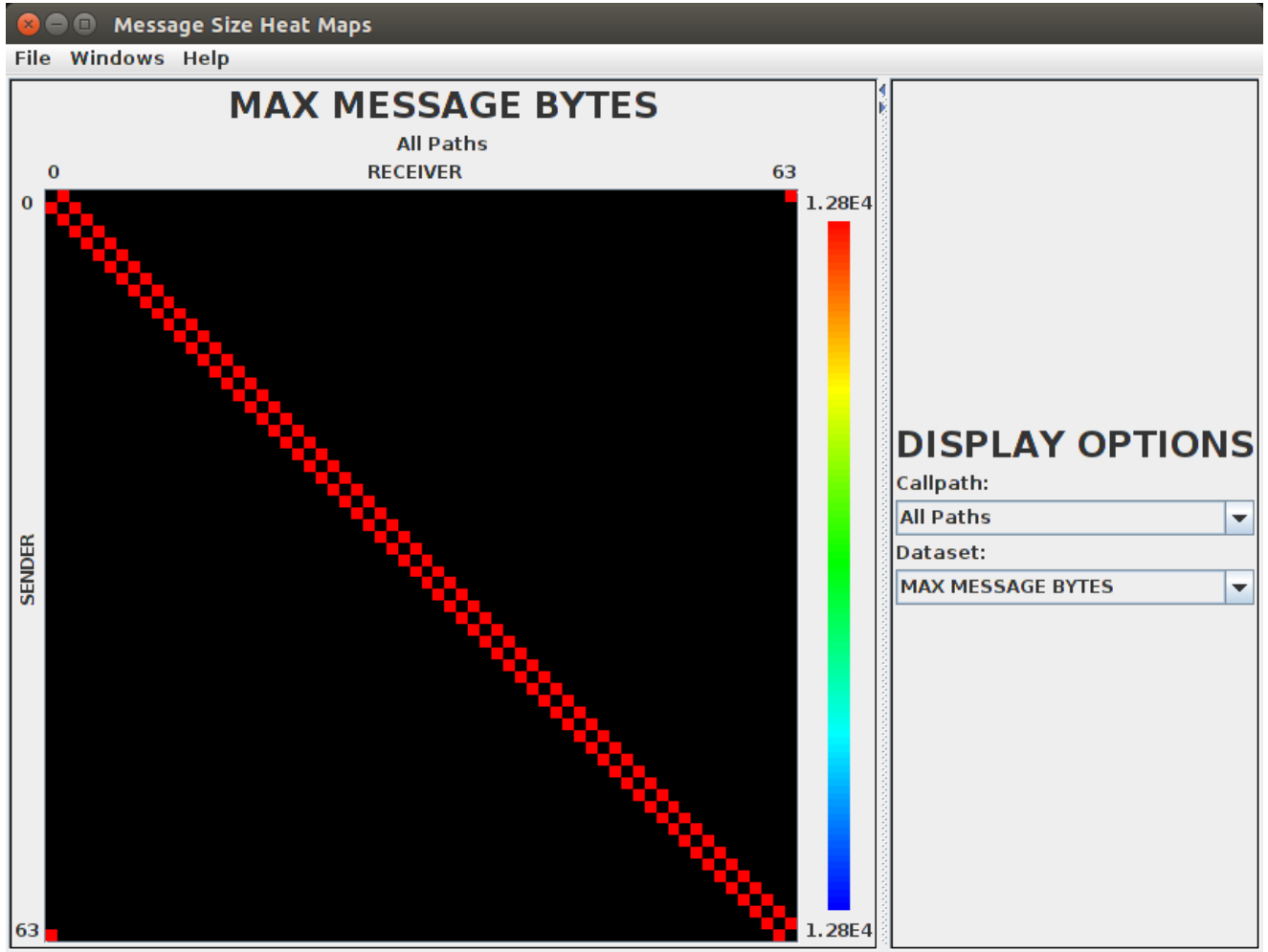
Paraprof – 3D Communication Matrix

- Menu Windows -> 3D Visualization (3D demands OpenGL)
- Max message size vs Number of calls



Paraprof

Menu Windows -> Communication Matrix



Which loops require the most time?

- File select.tau:

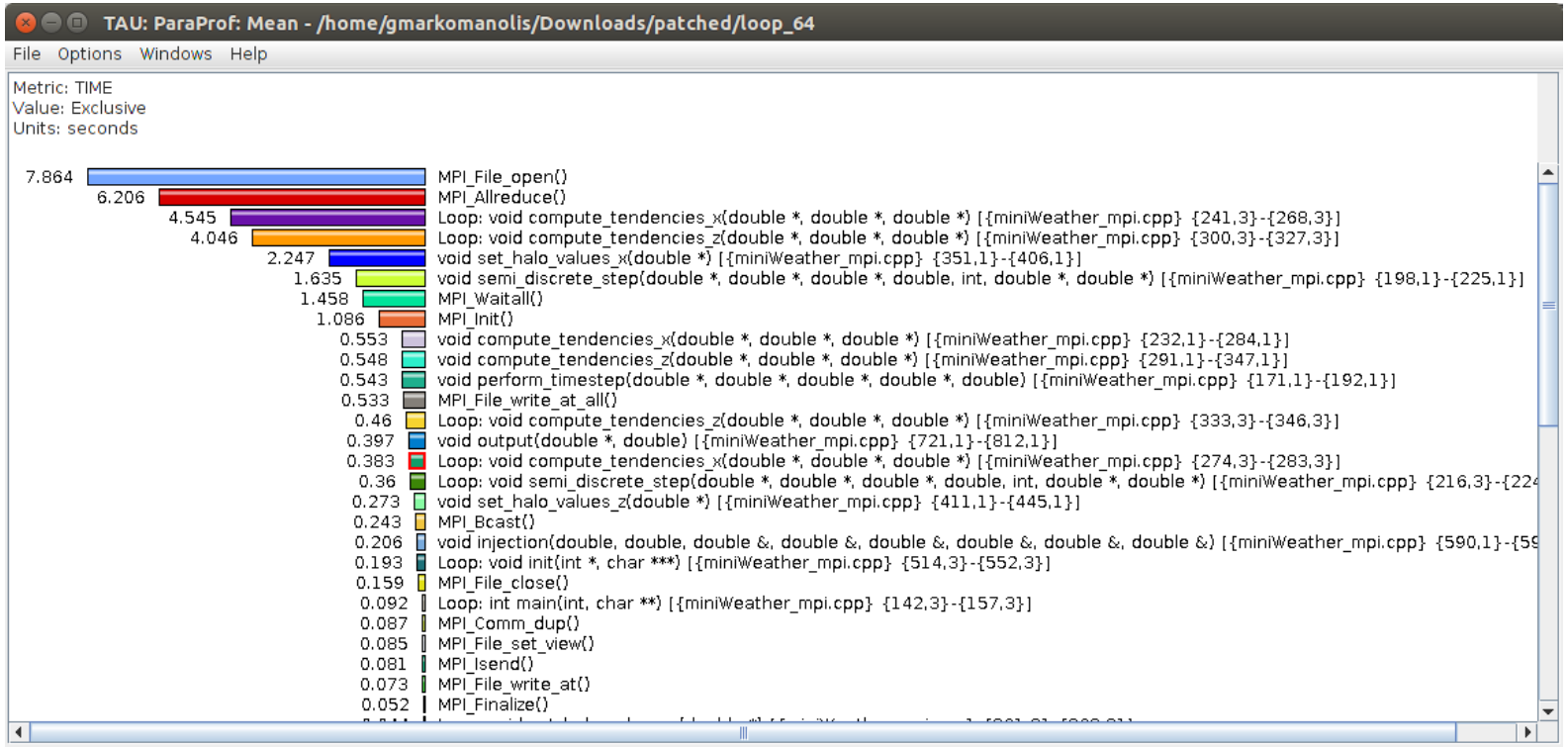
```
BEGIN_INSTRUMENT_SECTION
loops routine="#"
END_INSTRUMENT_SECTION
```

- Declare TAU options:

```
export TAU_OPTIONS="-optTauSelectFile=select.tau -optLinking=-lpnetcdf -optVerbose"
```

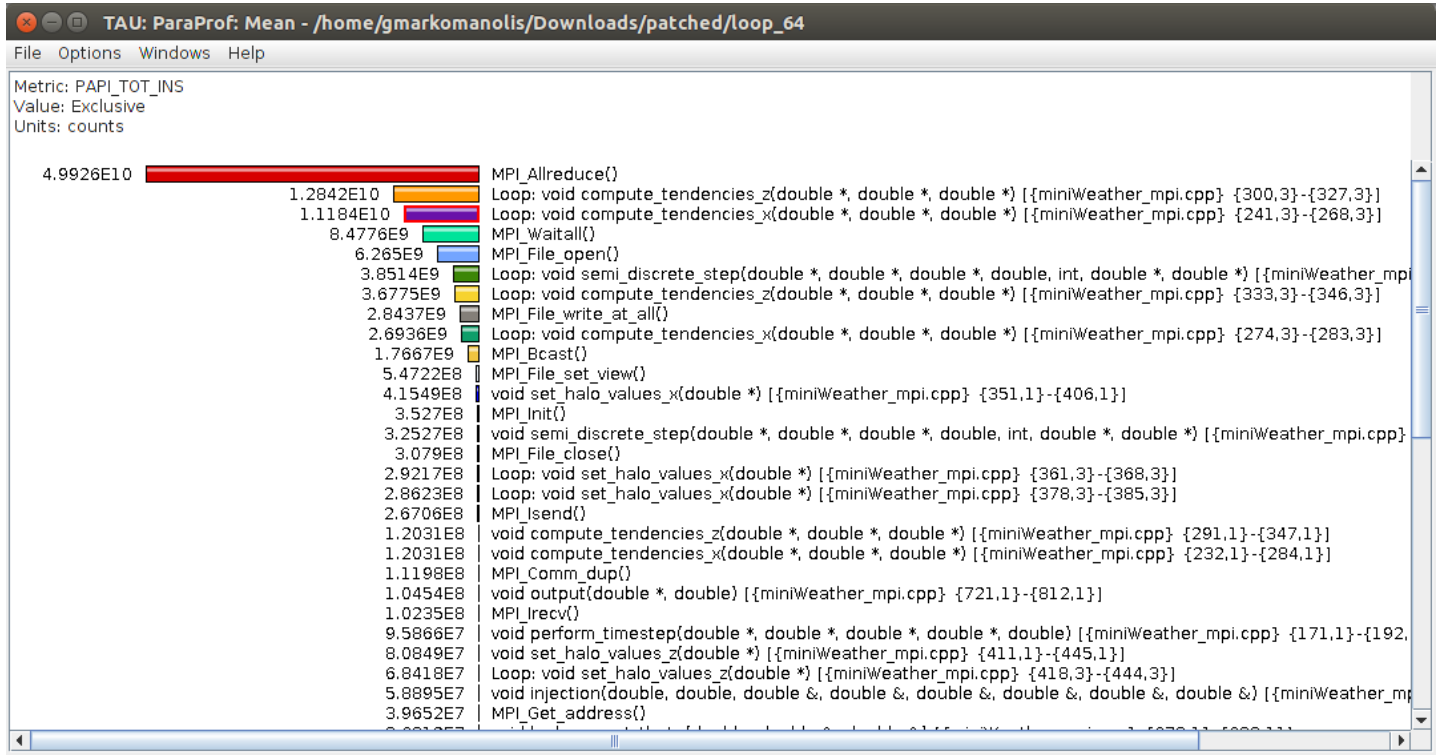
- Do not forget to unset TAU_OPTIONS when not required
- Execute as before

Paraprof - Loops



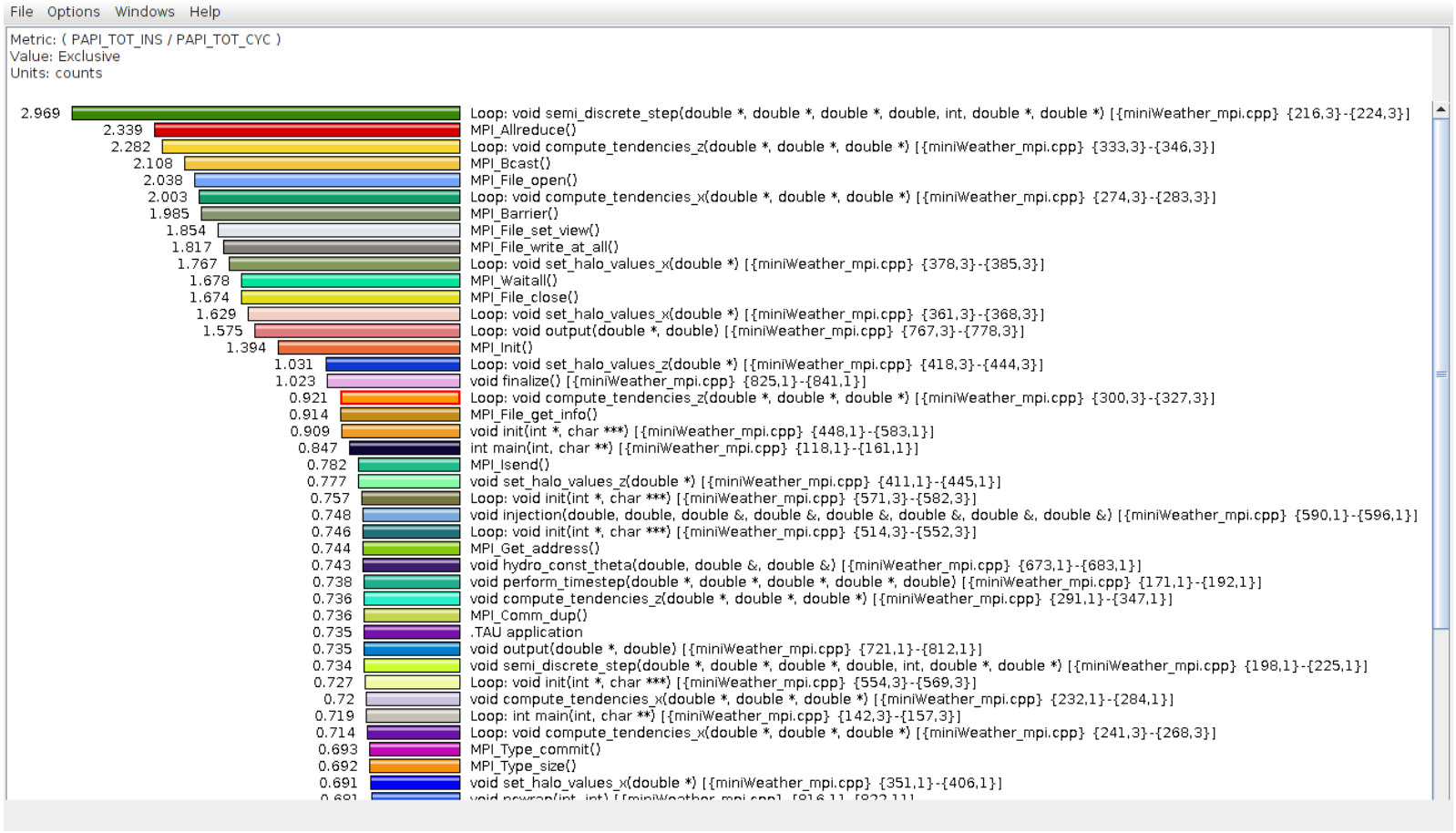
Paraprof - Loops

Select Options -> Select Metric… -> Exclusive… -> PAPI_TOT_INS



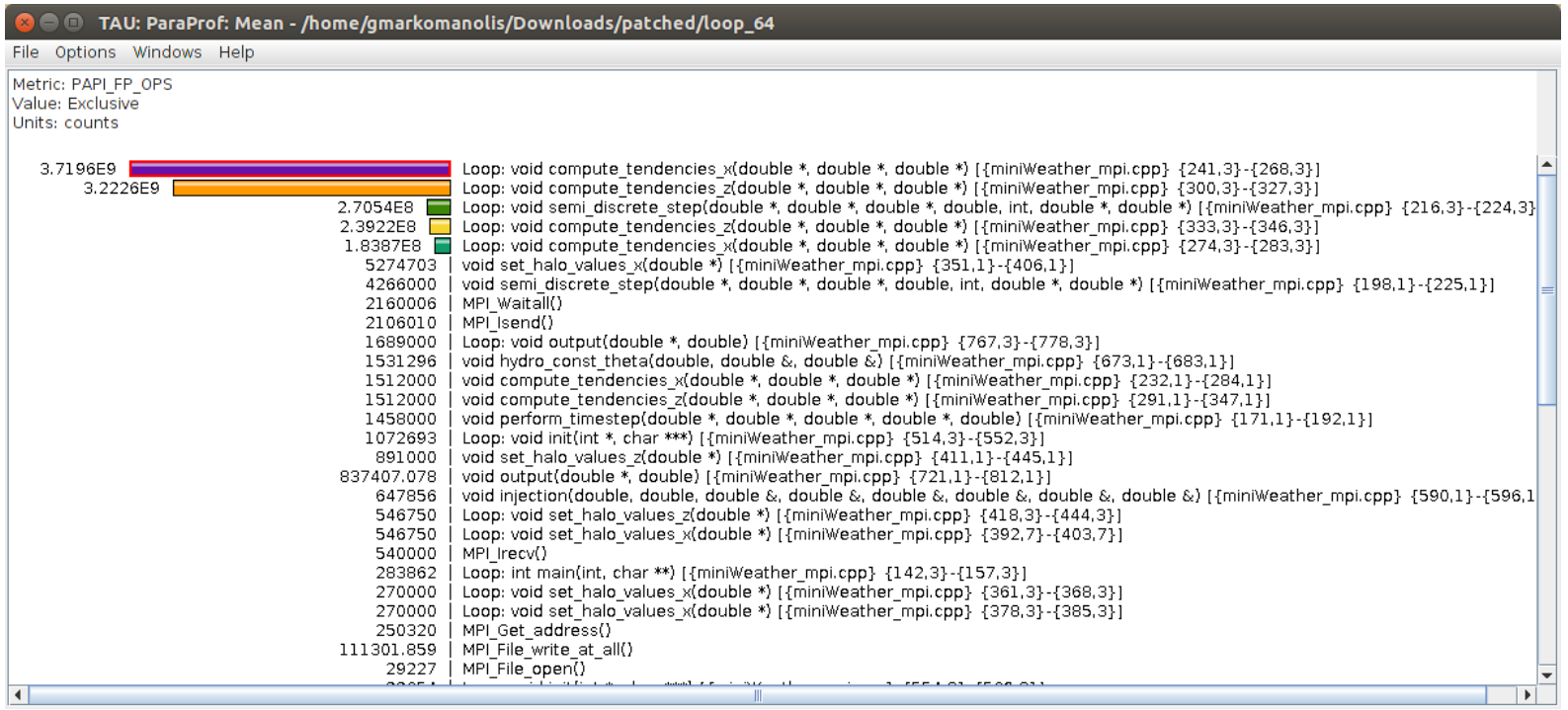
Paraprof - Loops

Select Options -> Select Metric… -> Exclusive… -> PAPI_TOT_INS/PAPI_TOT_CYC



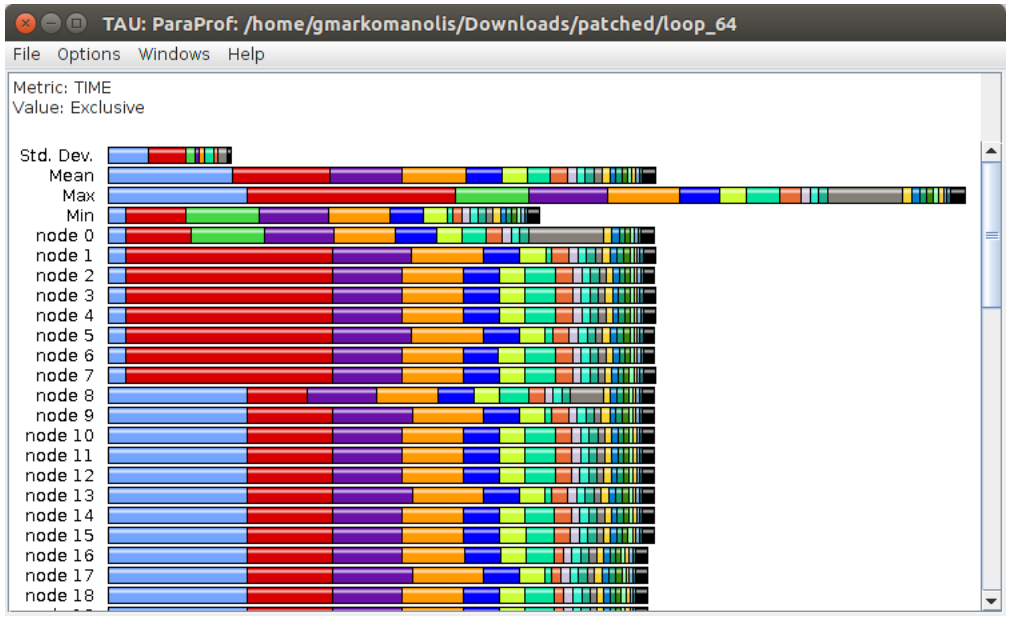
Paraprof - Loops

Select Options -> Select Metric… -> Exclusive… -> PAPI_FP_OPS

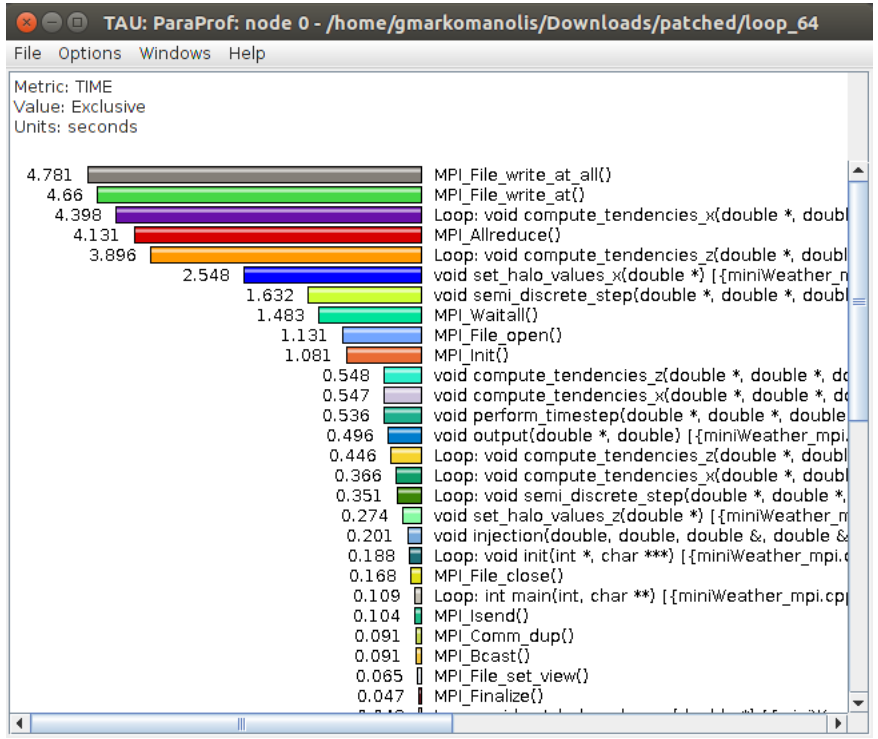


Paraprof

From the main window select a node

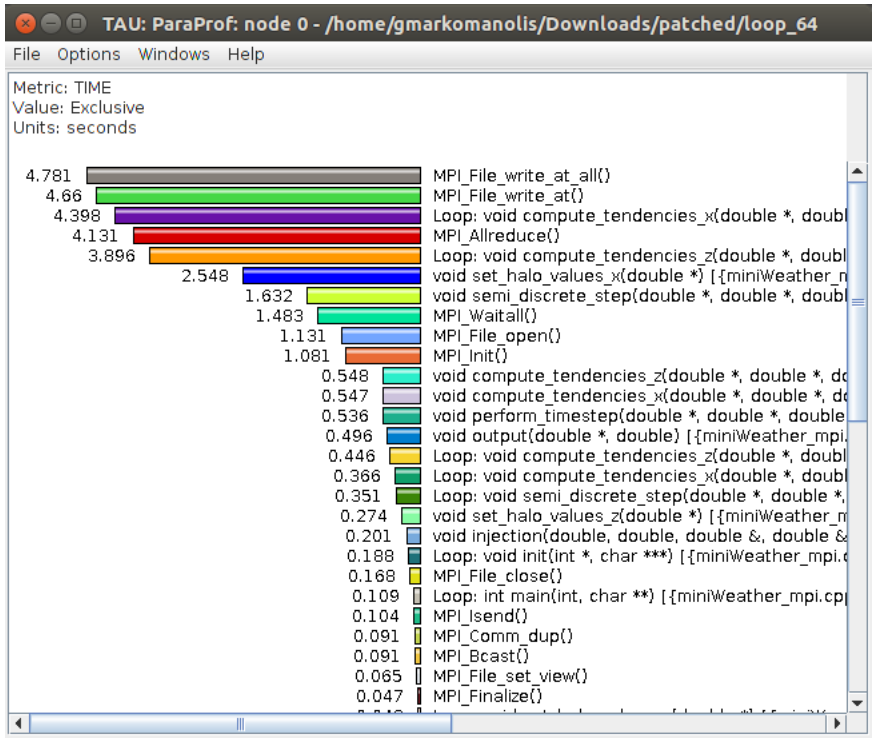


Click on node 0

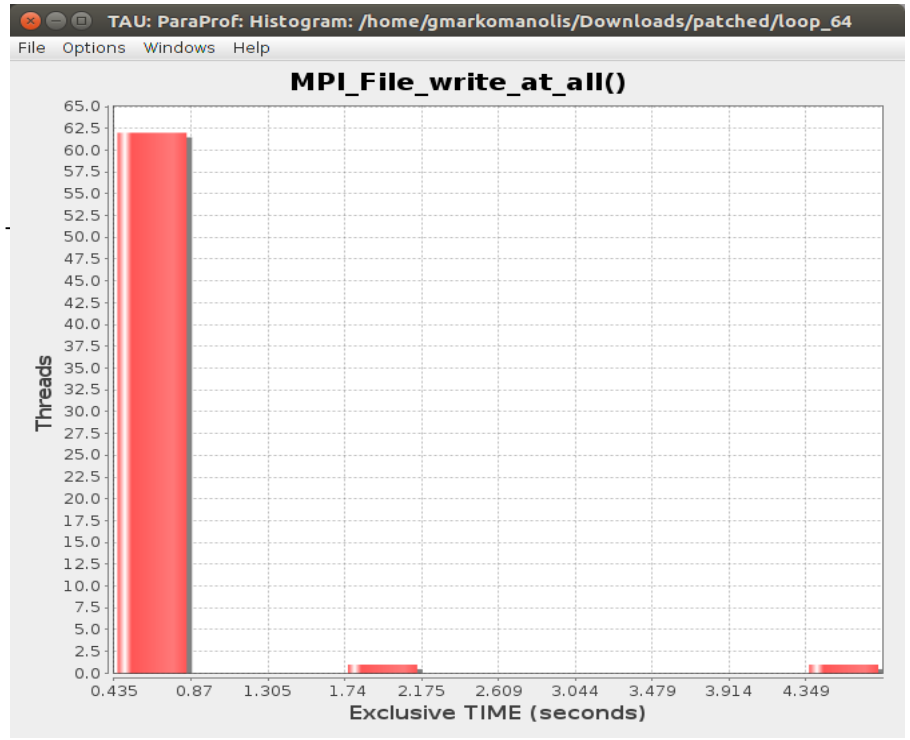


Paraprof – Function Histogram

From the main window select a node



Right click on MPI_File_write_at_all() - Show Function Histogram



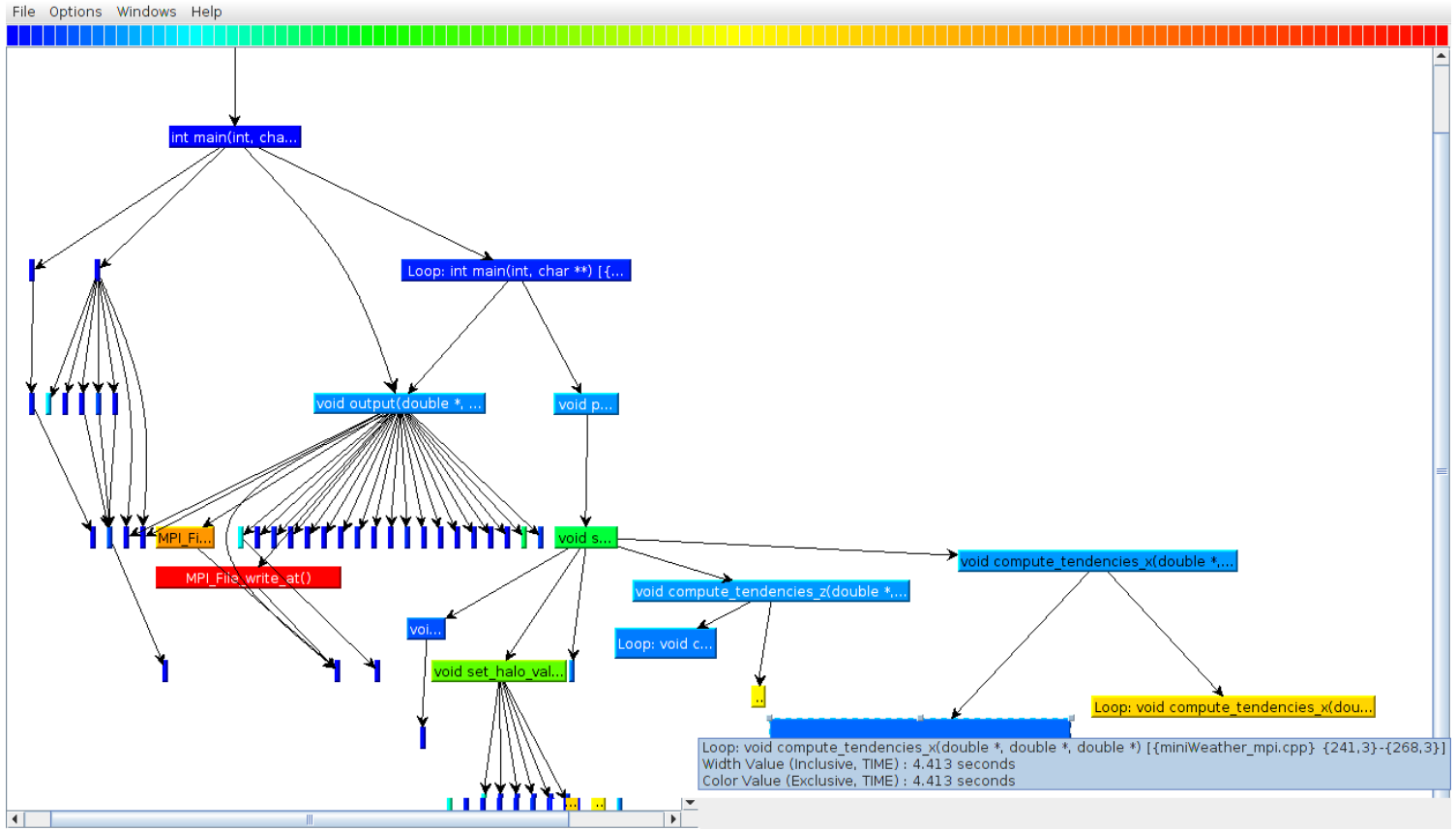
Callpath

```
export TAU_METRICS=TIME:PAPI_TOT_INS:PAPI_TOT_CYC:PAPI_FP_OPS
export TAU_CALLPATH=1
export TAU_CALLPATH_DEPTH=10
export TAU_PROFILE=1
export TAU_TRACK_MESSAGE=1
export TAU_COMM_MATRIX=1

jsrun -n 64 -r 8 -a 1 -c 1  ./miniWeather_mpi
```

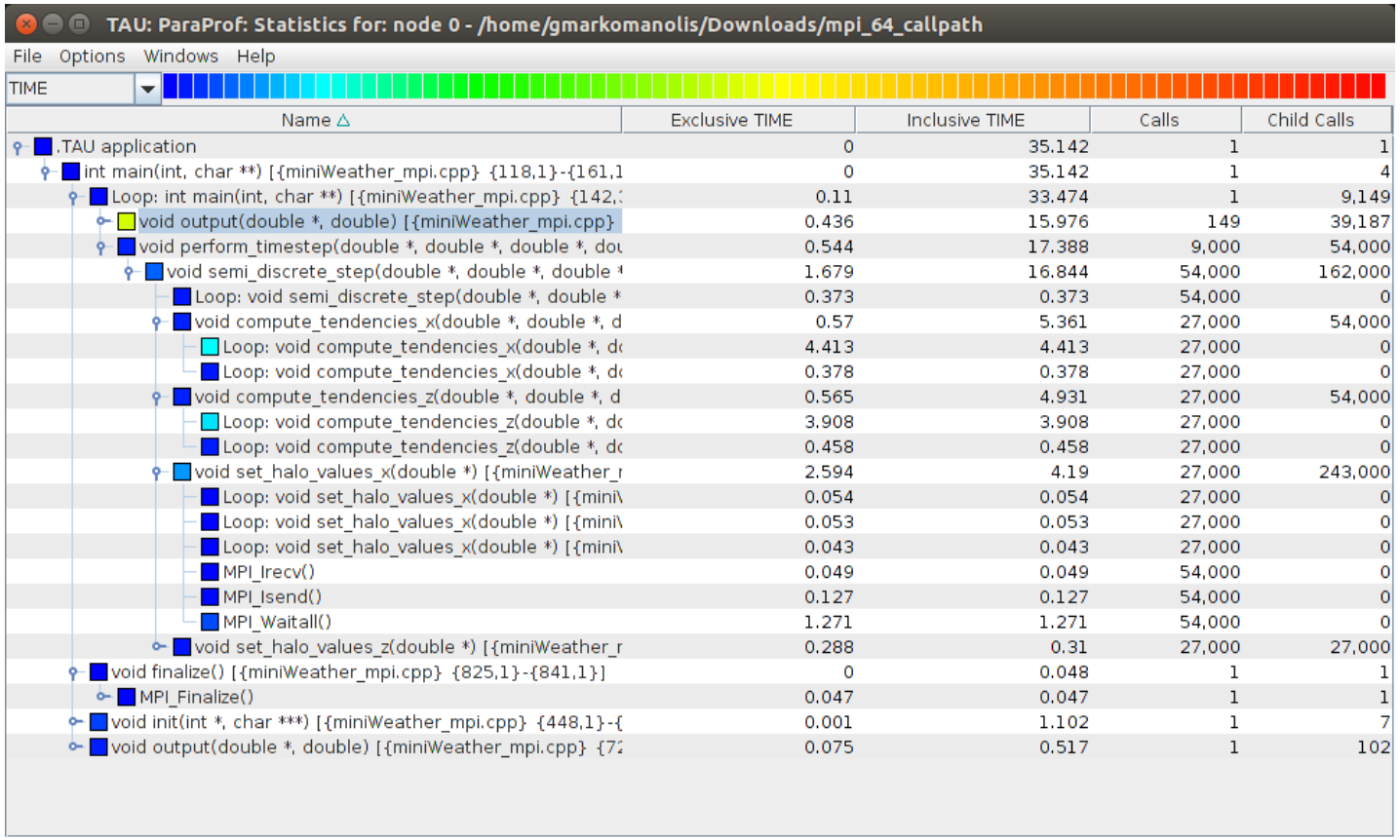
Paraprof - Callpath

From the main Window right click on any label (node 0, mean etc.) and select “Show Thread Call Graph”



Paraprof - Callpath

From the main Window right click on any label (node 0, mean etc.) and select “Show Thread Statistics Table”



Name	Exclusive TIME	Inclusive TIME	Calls	Child Calls
.TAU application	0	35.142	1	1
int main(int, char **) [{miniWeather_mpi.cpp} {118,1}-{161,1	0	35.142	1	4
Loop: int main(int, char **) [{miniWeather_mpi.cpp} {142,	0.11	33.474	1	9,149
void output(double *, double) [{miniWeather_mpi.cpp}	0.436	15.976	149	39,187
void perform_timestep(double *, double *, double *, dou	0.544	17.388	9,000	54,000
void semi_discrete_step(double *, double *, double *	1.679	16.844	54,000	162,000
Loop: void semi_discrete_step(double *, double *	0.373	0.373	54,000	0
void compute_tendencies_x(double *, double *, d	0.57	5.361	27,000	54,000
Loop: void compute_tendencies_x(double *, d	4.413	4.413	27,000	0
Loop: void compute_tendencies_x(double *, d	0.378	0.378	27,000	0
void compute_tendencies_z(double *, double *, d	0.565	4.931	27,000	54,000
Loop: void compute_tendencies_z(double *, d	3.908	3.908	27,000	0
Loop: void compute_tendencies_z(double *, d	0.458	0.458	27,000	0
void set_halo_values_x(double *) [{miniWeather_	2.594	4.19	27,000	243,000
Loop: void set_halo_values_x(double *) [{mini\	0.054	0.054	27,000	0
Loop: void set_halo_values_x(double *) [{mini\	0.053	0.053	27,000	0
Loop: void set_halo_values_x(double *) [{mini\	0.043	0.043	27,000	0
MPI_Irecv()	0.049	0.049	54,000	0
MPI_Isend()	0.127	0.127	54,000	0
MPI_Waitall()	1.271	1.271	54,000	0
void set_halo_values_z(double *) [{miniWeather_r	0.288	0.31	27,000	27,000
void finalize() [{miniWeather_mpi.cpp} {825,1}-{841,1}]	0	0.048	1	1
MPI_Finalize()	0.047	0.047	1	1
void init(int *, char ***) [{miniWeather_mpi.cpp} {448,1}-{	0.001	1.102	1	7
void output(double *, double) [{miniWeather_mpi.cpp} {7	0.075	0.517	1	102