OAK RIDGE National Laboratory

Performance Analysis with TAU, part II

George S. Markomanolis

8 August 2019

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

U.S. DEPARTMENT OF ENERGY

OAK RIDGE National Laboratory

MPI+OpenMP

MiniWeather MPI+OpenMP compilation

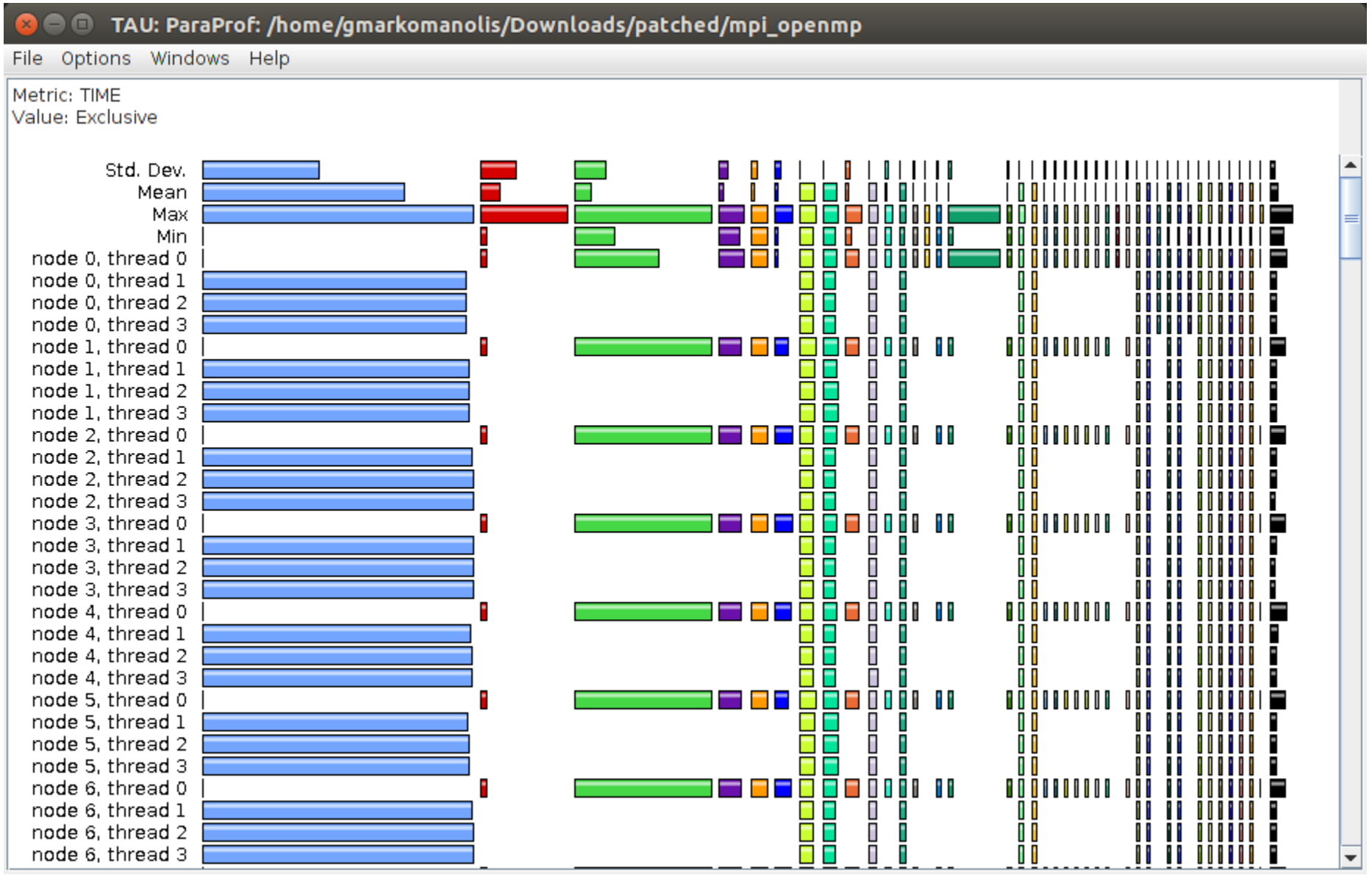
- module load pgi
- module load tau
- export TAU_MAKEFILE=/sw/summit/…/ibm64linux/lib/Makefile.tau-pgi-papi-mpi-pdt-openmp-opari-pgi
- Export TAU_OPTIONS='-optLinking=-lpnetcdf –optPreProcess'
- Replace mpicxx with tau_cxx.sh in the Makefile
- make openmp

MiniWeather MPI compilation and execute

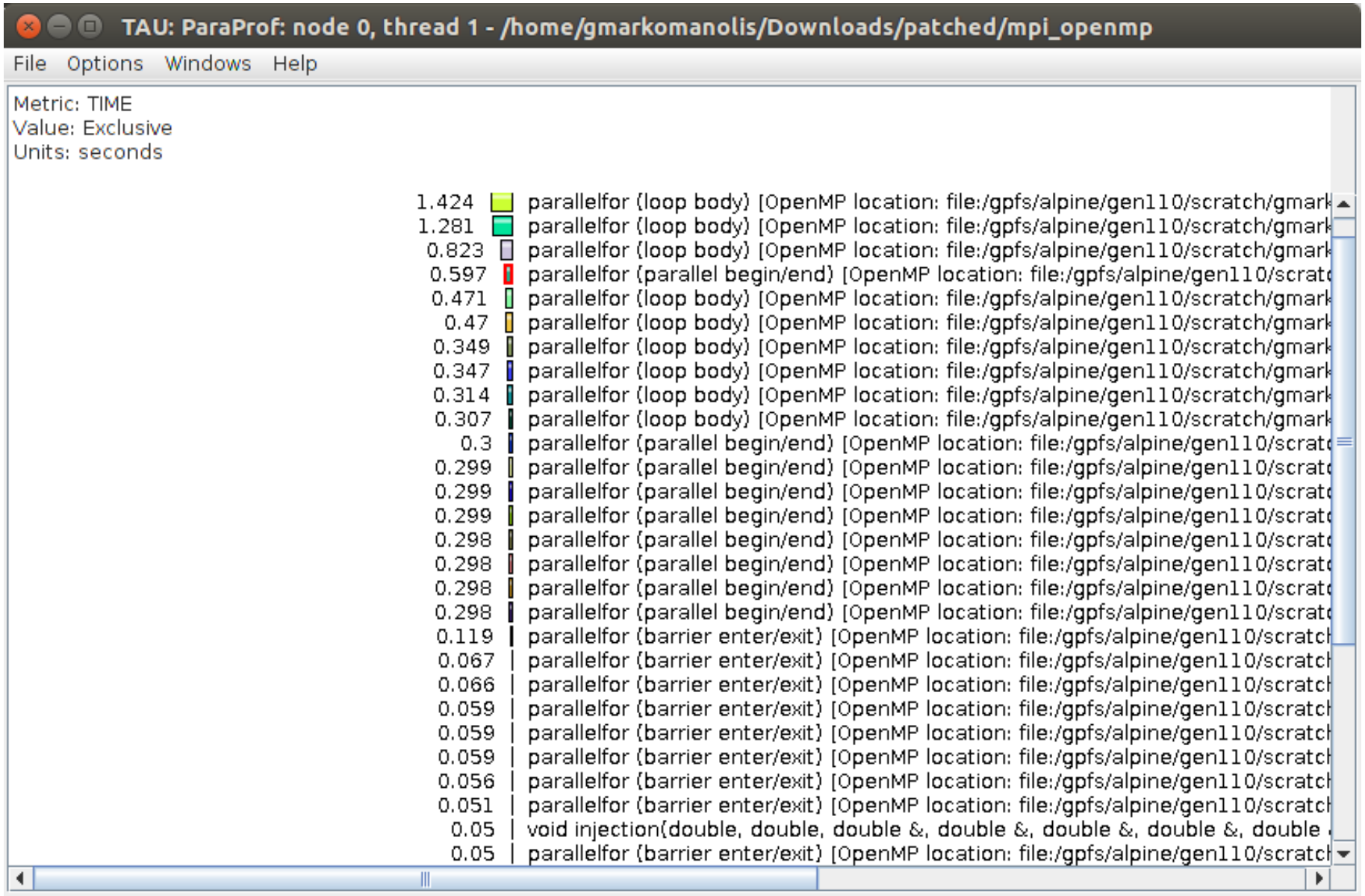
- Compile with: make openmp
- Execution:

```
export TAU_METRICS=TIME:PAPI_TOT_INS:PAPI_TOT_CYC:PAPI_FP_OPS
export TAU_PROFILE=1
export TAU_TRACK_MESSAGE=1
export TAU_COMM_MATRIX=1
#TAU_CALLPATH=1
#TAU_CALLPATH_DEPTH=10
jsrun -n 64 -r 8 -a 1 -c 4 -b packed:4 ./miniWeather_mpi_openmp
```

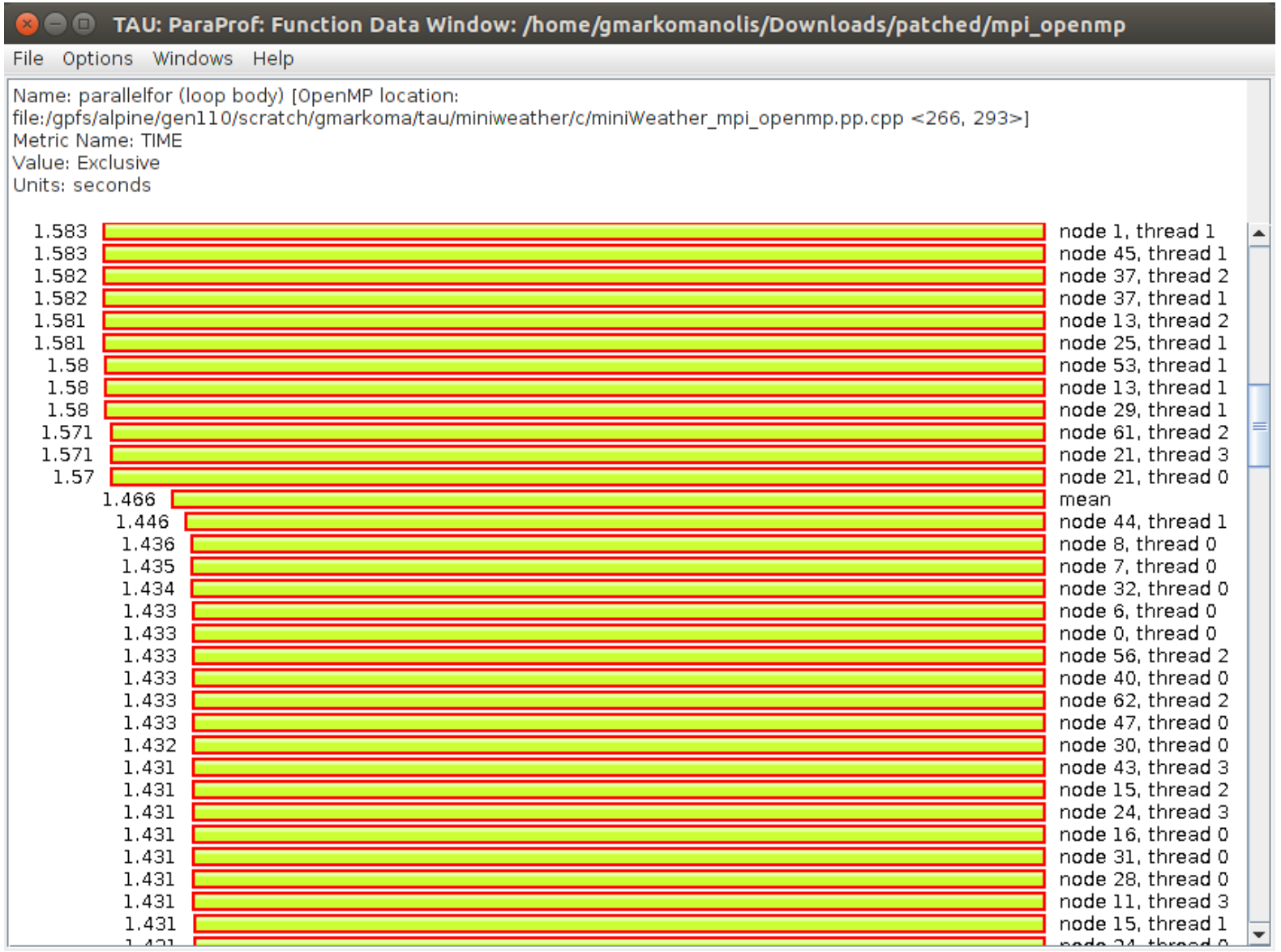
Paraprof - OpenMP



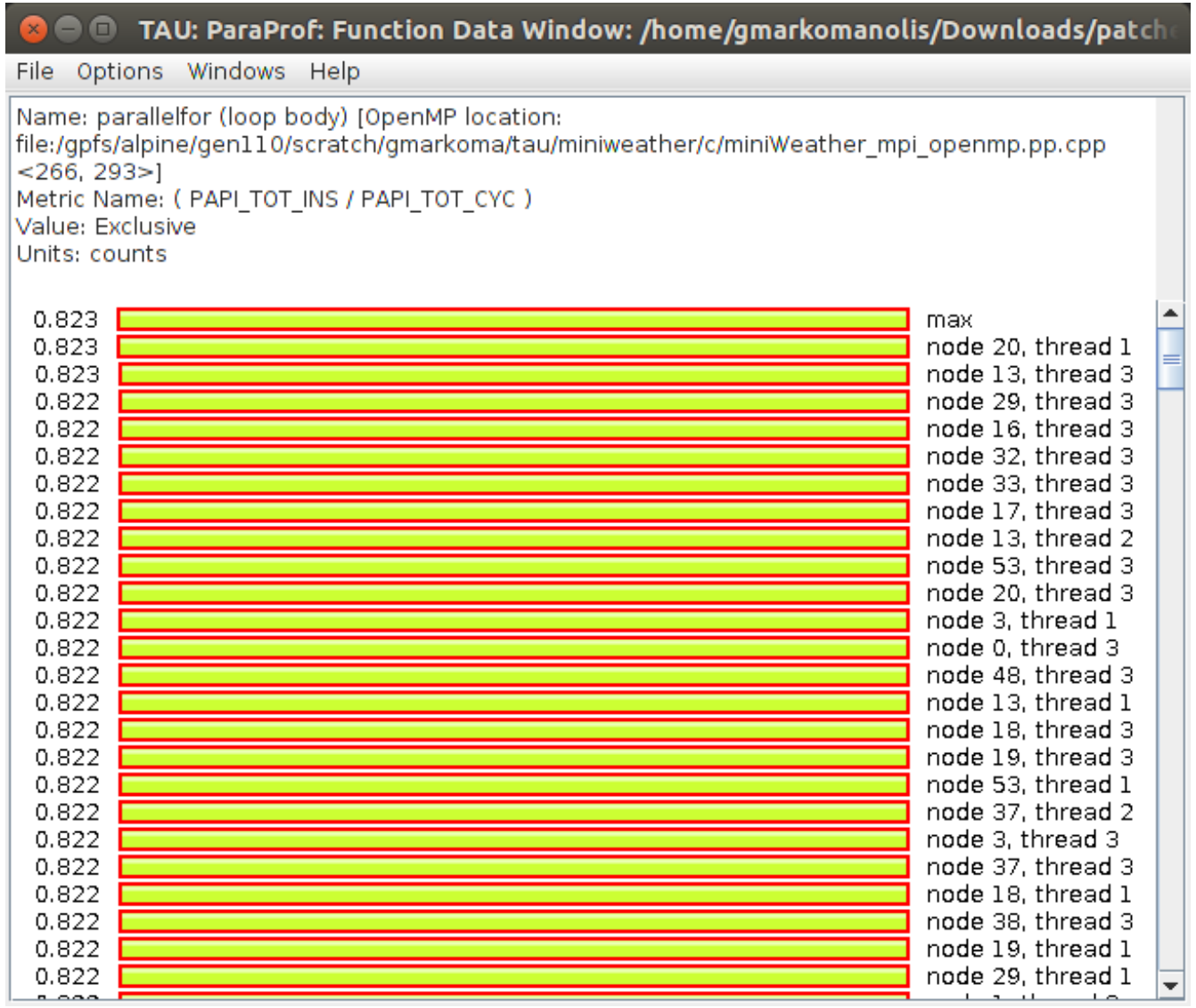
Paraprof – OpenMP - TIME



Paraprof – OpenMP - parallel loop



Paraprof – OpenMP - IPC



OAK RIDGE National Laboratory

GPU

MiniWeather MPI+OpenACC compilation

- module load tau
- Use mpicxx in the Makefile **not** tau_cxx.sh (for now)
- make openacc

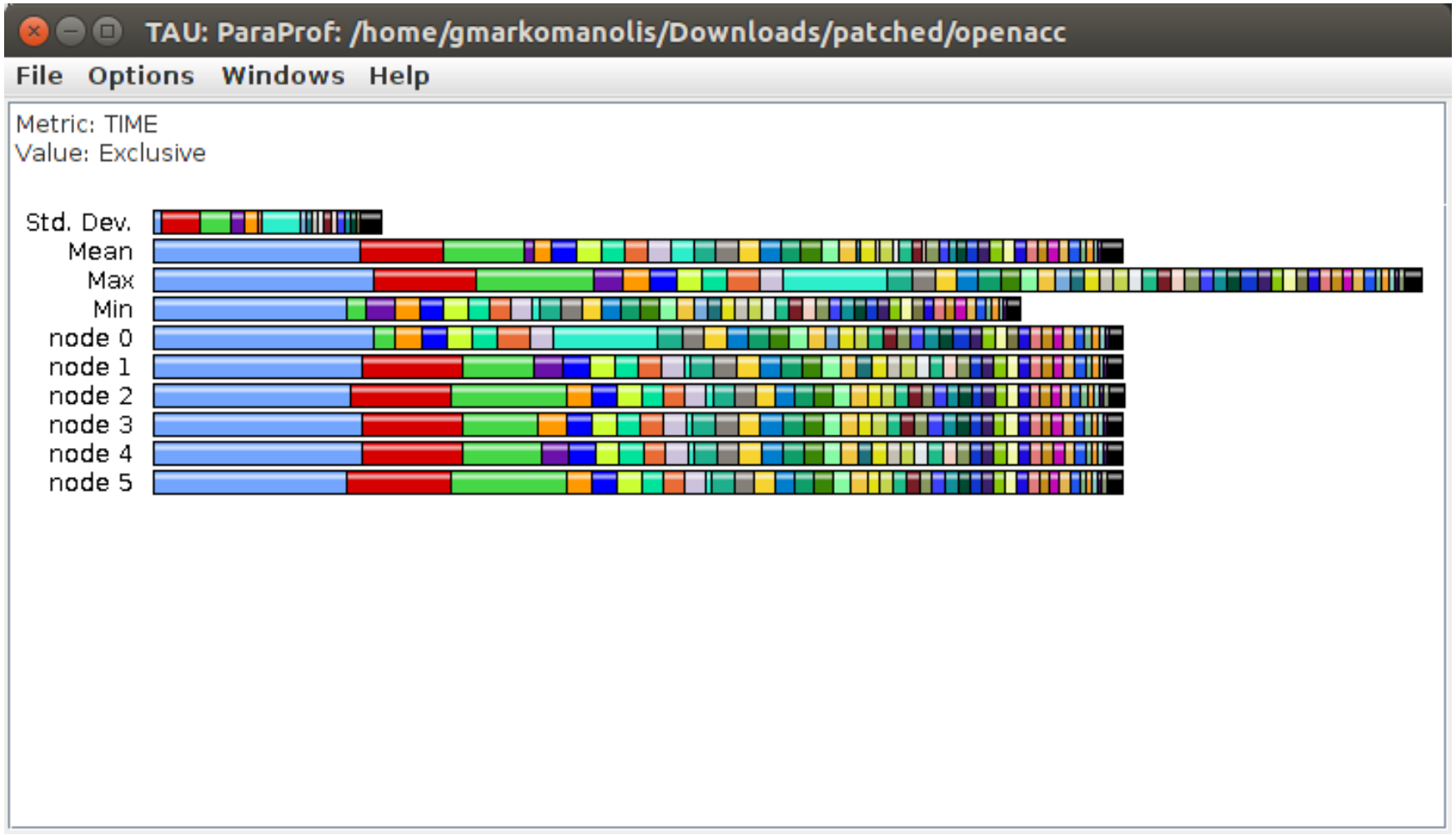
MiniWeather MPI compilation and execute

- Compile with: make openacc
- Execution:

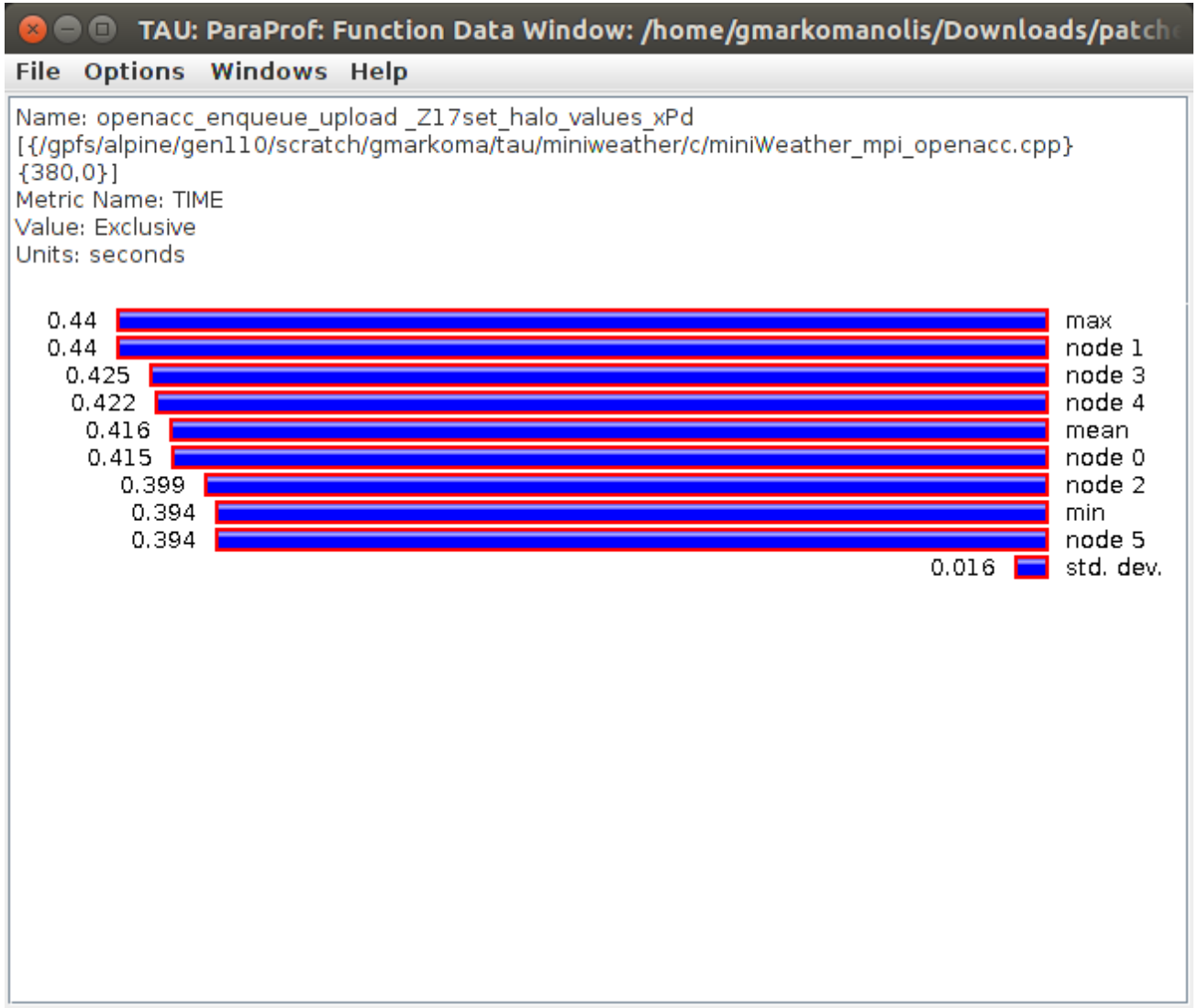
```
export TAU_METRICS=TIME

export TAU_PROFILE=1

export TAU_TRACK_MESSAGE=1

export TAU_COMM_MATRIX=1

jsrun -n 6 -r 6 --smpiargs="-gpu" -g 1  tau_exec -T mpi,pgi,pdt -openacc
./miniWeather_mpi_openacc
```

- **CUPTI metrics for OpenACC to be available up to SC19**

Paraprof – OpenACC



Paraprof – OpenACC



Paraprof – OpenACC

From the main window right click one label and select "Show User Event Statistics Window"

TAU: ParaProf: node 0 - /home/gmarkomanolis/Downloads/patched/openacc

File Options Windows Help

Sorted By: Number of Samples

Total	NumSamples	Max	Min	Mean	Std. Dev	Name
6.912E8	54000	12800	12800	12800	0	Message size sent to all nodes
6.912E8	54000	12800	12800	12800	0	Message size received from all nodes
6.912E8	54000	12800	12800	12800	0	Message size received in wait : MPI_Waitall()
6.912E8	54000	12800	12800	12800	0	Message size received in wait
3.456E8	27000	12800	12800	12800	0	Message size sent to node 5 : MPI_Isend()
3.456E8	27000	12800	12800	12800	0	Data transfer from device to host <variable=sendbuf_l>
3.456E8	27000	12800	12800	12800	0	Data transfer from device to host <variable=sendbuf_r>
3.456E8	27000	12800	12800	12800	0	Message size sent to node 1
3.456E8	27000	12800	12800	12800	0	Message size sent to node 1 : MPI_Isend()
3.456E8	27000	12800	12800	12800	0	Data transfer from host to device <variable=recvbuf_r>
3.456E8	27000	12800	12800	12800	0	Data transfer from host to device <variable=recvbuf_l>
3.456E8	27000	12800	12800	12800	0	Message size sent to node 5
6.4323E7	1051	107200	4	61201.412	53055.677	MPI-IO Bytes Written
-	1051	268	0.051	121.076	119.294	MPI-IO Write Bandwidth (MB/s)
6000	750	8	8	8	0	Message size for all-reduce
-	600	268	1.391	211.826	75.737	MPI-IO Write Bandwidth (MB/s) : MPI_File_write_at_all()
6.432E7	600	107200	107200	107200	0	MPI-IO Bytes Written : MPI_File_write_at_all()
2684	451	284	4	5.951	13.242	MPI-IO Bytes Written : MPI_File_write_at()
-	451	1.442	0.051	0.344	0.228	MPI-IO Write Bandwidth (MB/s) : MPI_File_write_at()
3.906E7	300	262144	4	130200	131067.087	Message size for broadcast
6.9523E7	150	463488	463488	463488	0	Data transfer from device to host <variable=state>
463488	1	463488	463488	463488	0	Data transfer from host to device <variable=state_tmp>
1632	1	1632	1632	1632	0	Data transfer from host to device <variable=hy_dens_cell>
1632	1	1632	1632	1632	0	Data transfer from host to device <variable=hy_dens_theta_cell>
1608	1	1608	1608	1608	0	Data transfer from host to device <variable=hy_dens_int>
1608	1	1608	1608	1608	0	Data transfer from host to device <variable=hy_dens_theta_int>
1608	1	1608	1608	1608	0	Data transfer from host to device <variable=hy_pressure_int>
463488	1	463488	463488	463488	0	Data transfer from host to device <variable=state>

CUPTI Metrics

- https://docs.nvidia.com/cupti/Cupti/r_main.html#metrics-reference

Metric Name	Description	Scope
achieved_occupancy	Ratio of the average active warps per active cycle to the maximum number of warps supported on a multiprocessor	Multi-context
alu_fu_utilization	The utilization level of the multiprocessor function units that execute integer and floating-point arithmetic instructions on a scale of 0 to 10	Multi-context
atomic_replay_overhead	Average number of replays due to atomic and reduction bank conflicts for each instruction executed	Multi-context
atomic_throughput	Global memory atomic and reduction throughput	Multi-context
atomic_transactions	Global memory atomic and reduction transactions	Multi-context
atomic_transactions_per_request	Average number of global memory atomic and reduction transactions performed for each atomic and reduction instruction	Multi-context
branch_efficiency	Ratio of non-divergent branches to total branches expressed as percentage. This is available for compute capability 3.0.	Multi-context
cf_executed	Number of executed control-flow instructions	Multi-context
cf_fu_utilization	The utilization level of the multiprocessor function units that execute control-flow instructions on a scale of 0 to 10	Multi-context
cf_issued	Number of issued control-flow instructions	Multi-context
dram_read_throughput	Device memory read throughput. This is available for compute capability 3.0, 3.5 and 3.7.	Multi-context*
dram_read_transactions	Device memory read transactions. This is available for compute capability 3.0, 3.5 and 3.7.	Multi-context*

LSMS MPI+OpenMP+CUDA execution

```
module load gcc
export TAU_METRICS=TIME,achieved_occupancy
jsrun --smpiargs="-gpu" --nrs 2 --tasks_per_rs 1 --gpu_per_rs 1 --
rs_per_host 2 --cpu_per_rs 8 --bind rs tau_exec -T
mpi,pdt,papi,cupti,openmp -ompt -cupti $EXECUTABLE $INPUT
```

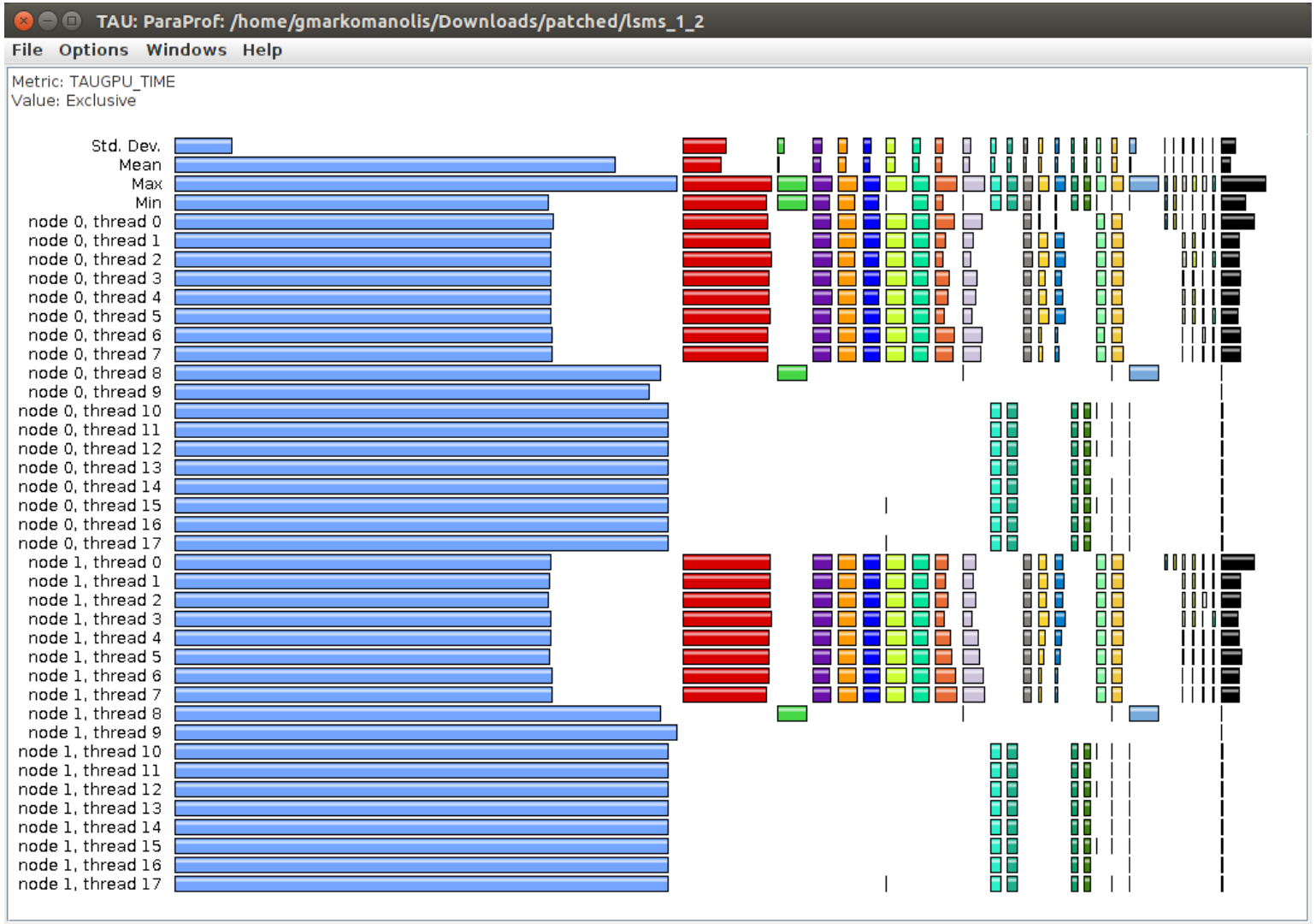
- Error: **Only counters for a single GPU device model can be collected at the same time**.
- Achieved_occupancy= CUDA.Tesla_V100-SXM2-16GB.domain_d.active_warps/CUDA.Tesla_V100-SXM2-16GB.domain_d.active_cycles

LSMS MPI+OpenMP+CUDA execution

- module load gcc
- jsrun --smpiargs="-gpu" --nrs 2 --tasks_per_rs 1 --gpu_per_rs 1 --rs_per_host 2 --cpu_per_rs 8 --bind rs **tau_exec -T mpi,pdt,papi,cupti,openmp -ompt -cupti** $EXECUTABLE $INPUT

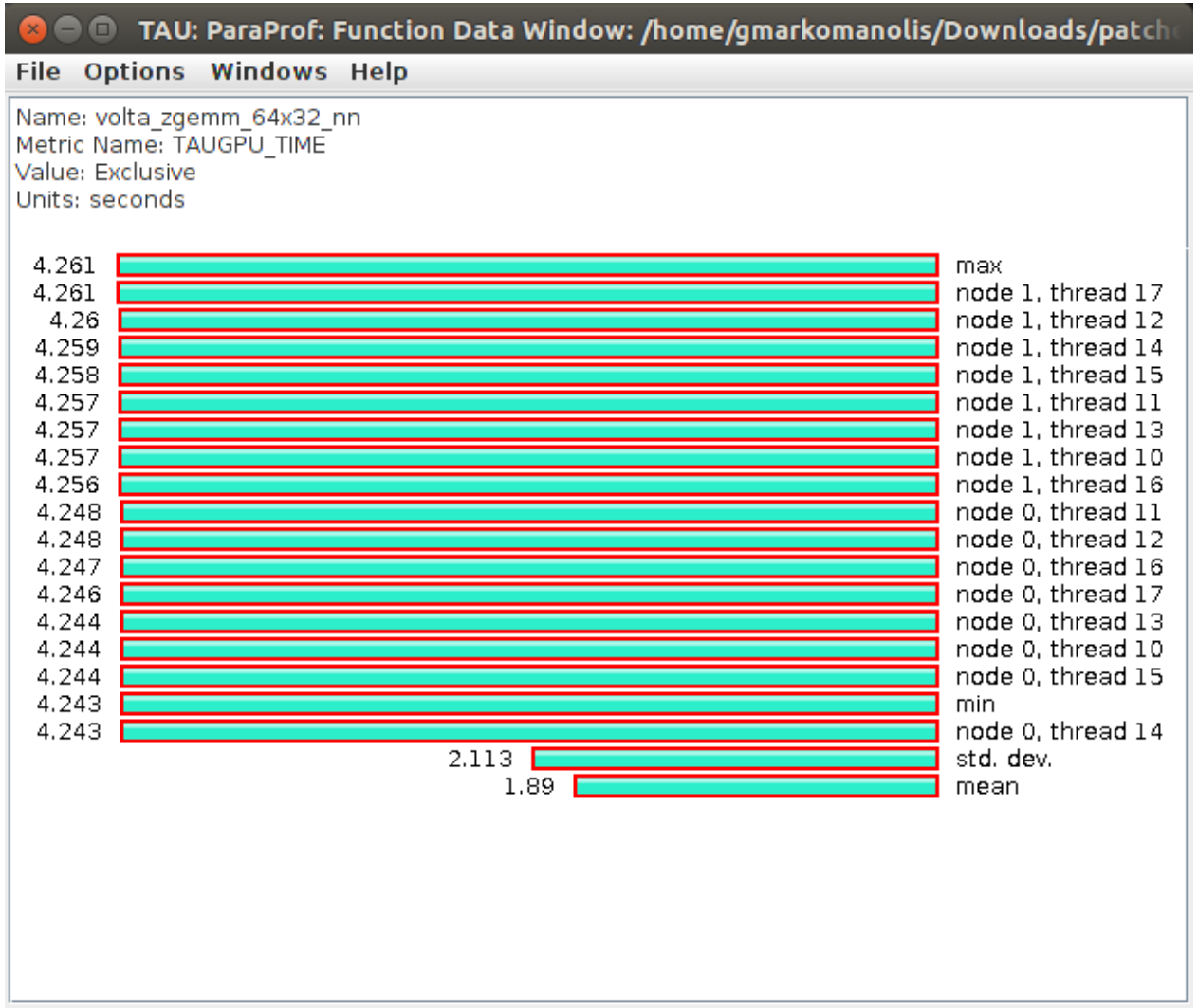
LSMS - Paraprof – MPI+OpenMP+CUDA

Options -> Uncheck Stack Bars Together



LSMS - Paraprof – MPI+OpenMP+CUDA

- Click on any color



LSMS - Paraprof – MPI+OpenMP+CUDA

- Statistics for thread on CPU

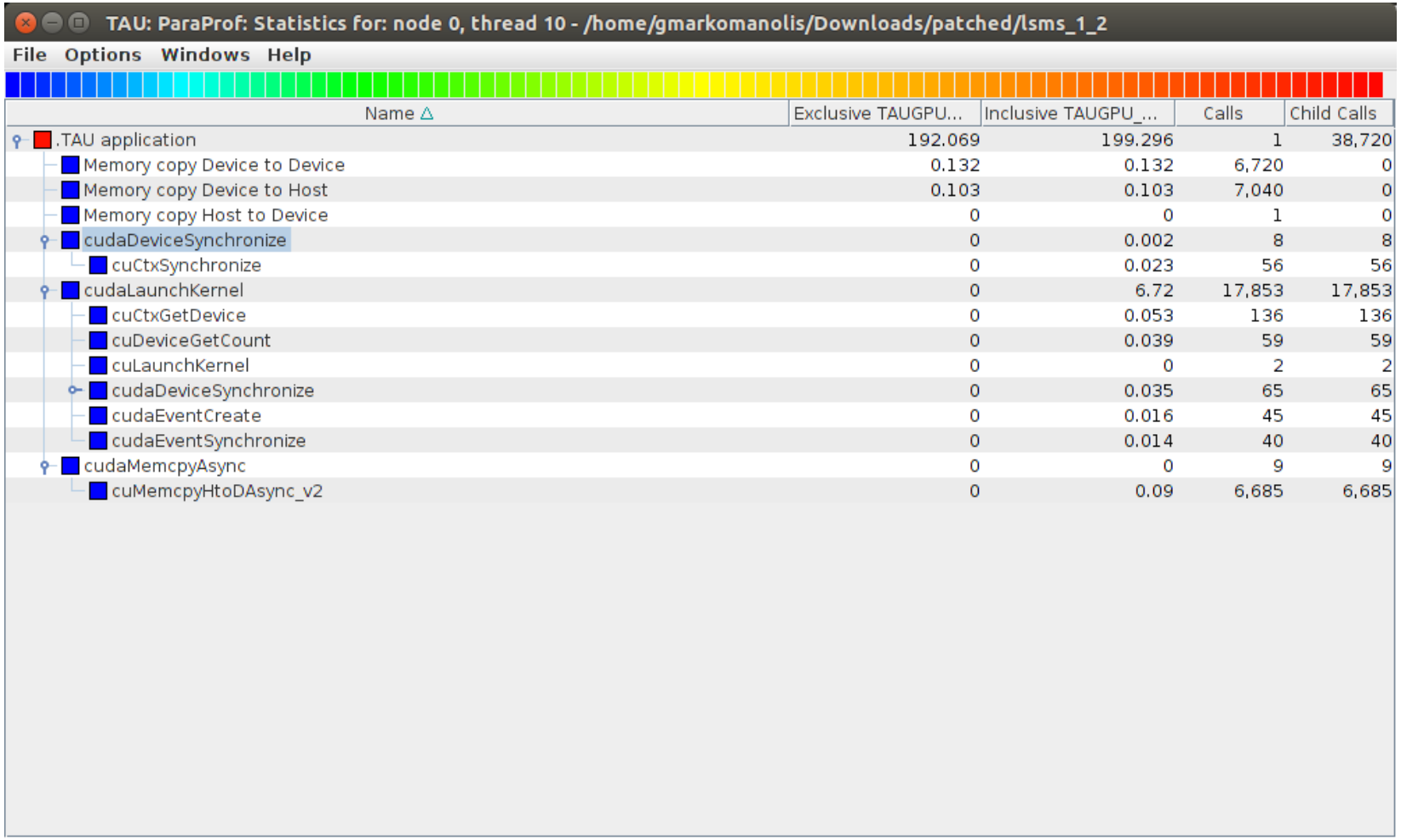
TAU: ParaProf: Statistics for: node 0, thread 1 - /home/gmarkomanolis/Downloads/patched/lsms_1_2

File Options Windows Help

Name	Exclusive TAUGPU...	Inclusive TAUGPU_...	Calls	Child Calls
.TAU application	146.528	200.597	1	119,047
cudaDeviceSetSharedMemConfig	0	0	1	1
cudaDeviceSynchronize	0.063	4.655	14,080	14,080
cudaEventQuery	0.008	0.027	2,240	2,240
cudaEventRecord	0.096	1.559	22,720	22,720
cudaEventSynchronize	0.38	14.745	13,760	27,520
cudaGetLastError	0.05	0.05	18,240	0
cudaHostAlloc	0	0.01	2	2
cudaLaunchKernel	0.337	23.713	18,240	127,680
cuCtxGetDevice	0.081	0.081	36,480	0
cuDeviceGetCount	0.045	0.045	18,240	0
cuLaunchKernel	0.683	11.675	18,240	109,440
cudaDeviceSynchronize	0.082	3.534	18,240	18,240
cudaEventCreate	0.075	0.207	18,240	18,240
cudaEventSynchronize	0.128	7.834	18,240	36,480
cudaMalloc	0	0.007	3	18
cudaMallocHost	0	0.007	1	1
cudaMemcpy	0.005	5.155	320	640
cudaMemcpy2DAsync	0.062	3.769	13,760	13,760
cudaMemcpyAsync	0.059	0.274	6,720	6,720
cudaMemsetAsync	0.009	0.036	2,240	2,240
cudaStreamWaitEvent	0.03	0.063	6,720	6,720

LSMS - Paraprof – MPI+OpenMP+CUDA

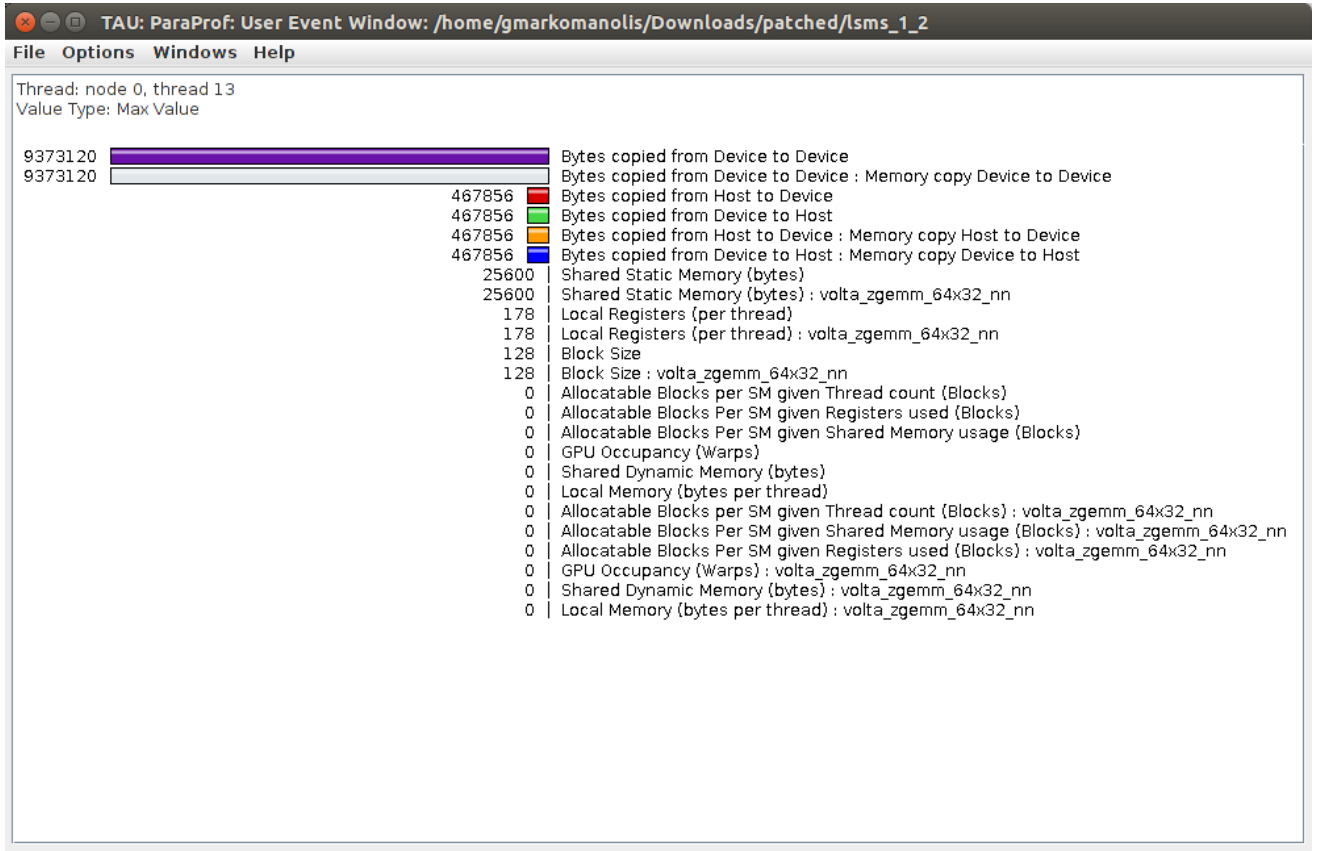
- Statistics for thread on GPU



Name △	Exclusive TAUGPU...	Inclusive TAUGPU_...	Calls	Child Calls
.TAU application	192.069	199.296	1	38,720
Memory copy Device to Device	0.132	0.132	6,720	0
Memory copy Device to Host	0.103	0.103	7,040	0
Memory copy Host to Device	0	0	1	0
cudaDeviceSynchronize	0	0.002	8	8
cuCtxSynchronize	0	0.023	56	56
cudaLaunchKernel	0	6.72	17,853	17,853
cuCtxGetDevice	0	0.053	136	136
cuDeviceGetCount	0	0.039	59	59
cuLaunchKernel	0	0	2	2
cudaDeviceSynchronize	0	0.035	65	65
cudaEventCreate	0	0.016	45	45
cudaEventSynchronize	0	0.014	40	40
cudaMemcpyAsync	0	0	9	9
cuMemcpyHtoDAsync_v2	0	0.09	6,685	6,685

LSMS - Paraprof – MPI+OpenMP+CUDA

- User event window



Benchmark for demonstration

```
export TAU_METRICS=TIME,achieved_occupancy
jsrun -n 2 -r 2 -g 1  tau_exec -T mpi,pdt,papi,cupti,openmp -ompt -cupti
./add
```

- Output folders:

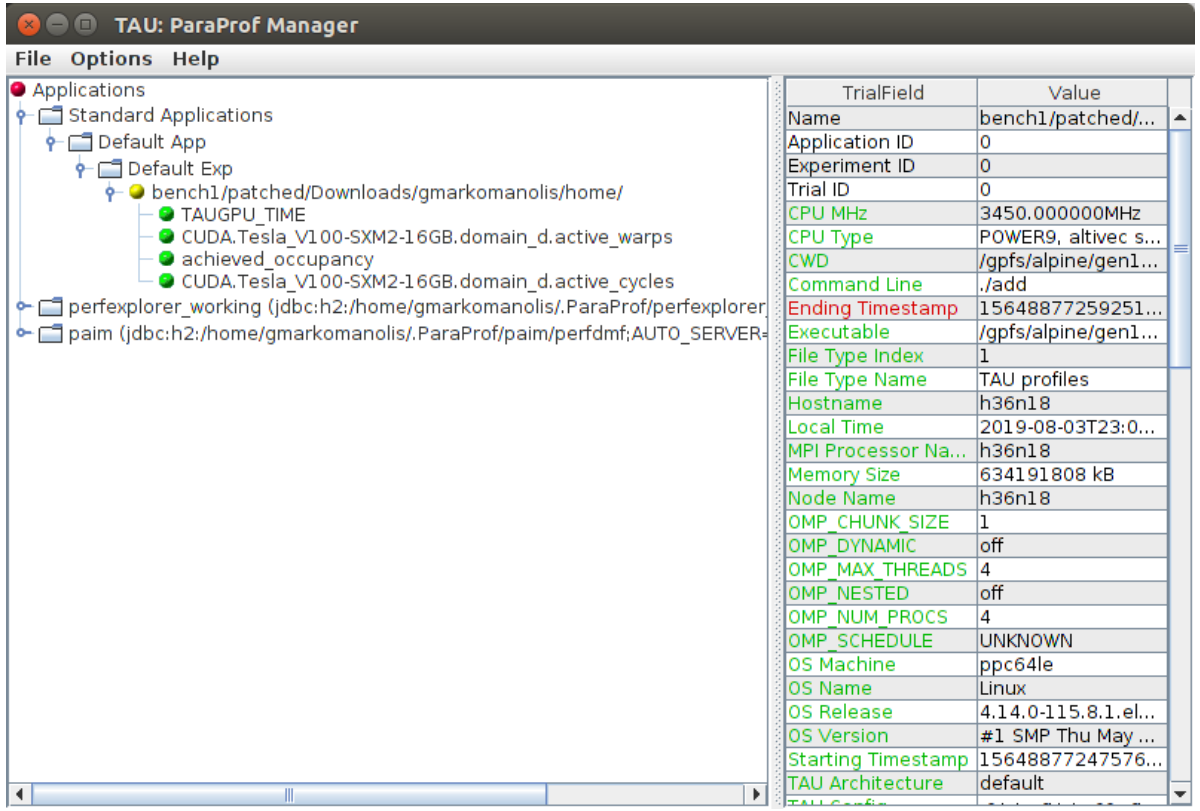
MULTI__TAUGPU_TIME

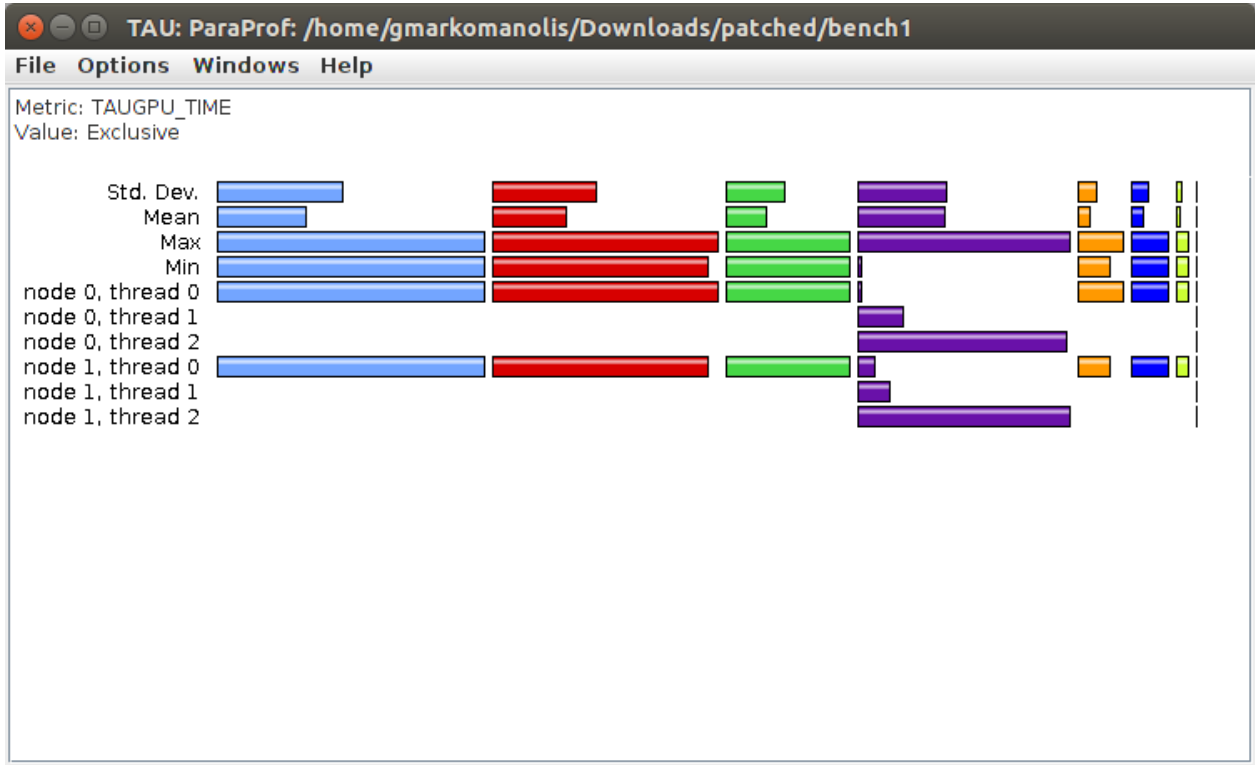
MULTI__CUDA.Tesla_V100-SXM2-16GB.domain_d.active_warps

MULTI__CUDA.Tesla_V100-SXM2-16GB.domain_d.active_cycles

MULTI__achieved_occupancy

Bechmark - Paraprof - MPI+OpenMP+CUDA

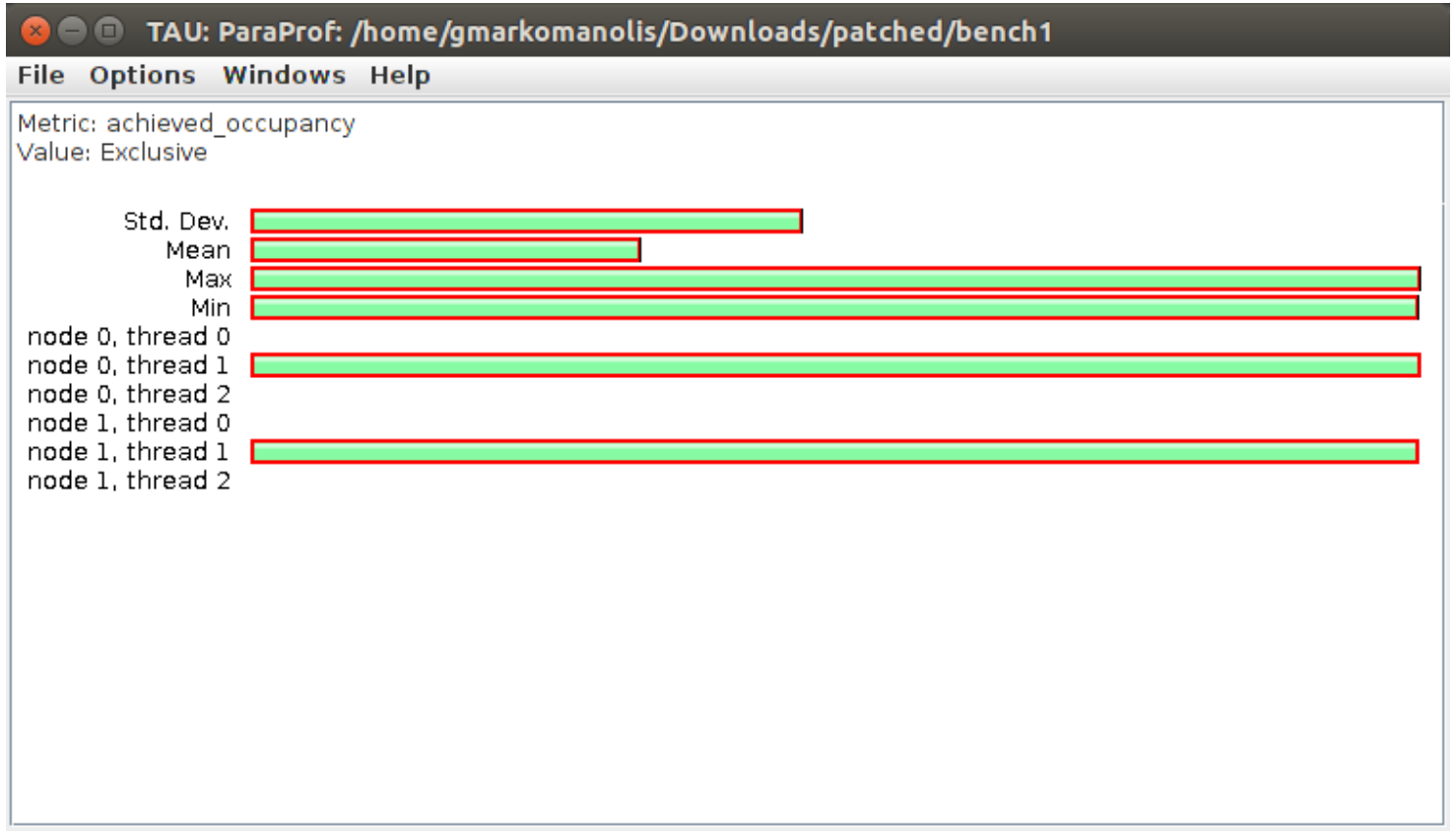




OAK RIDGE National Laboratory

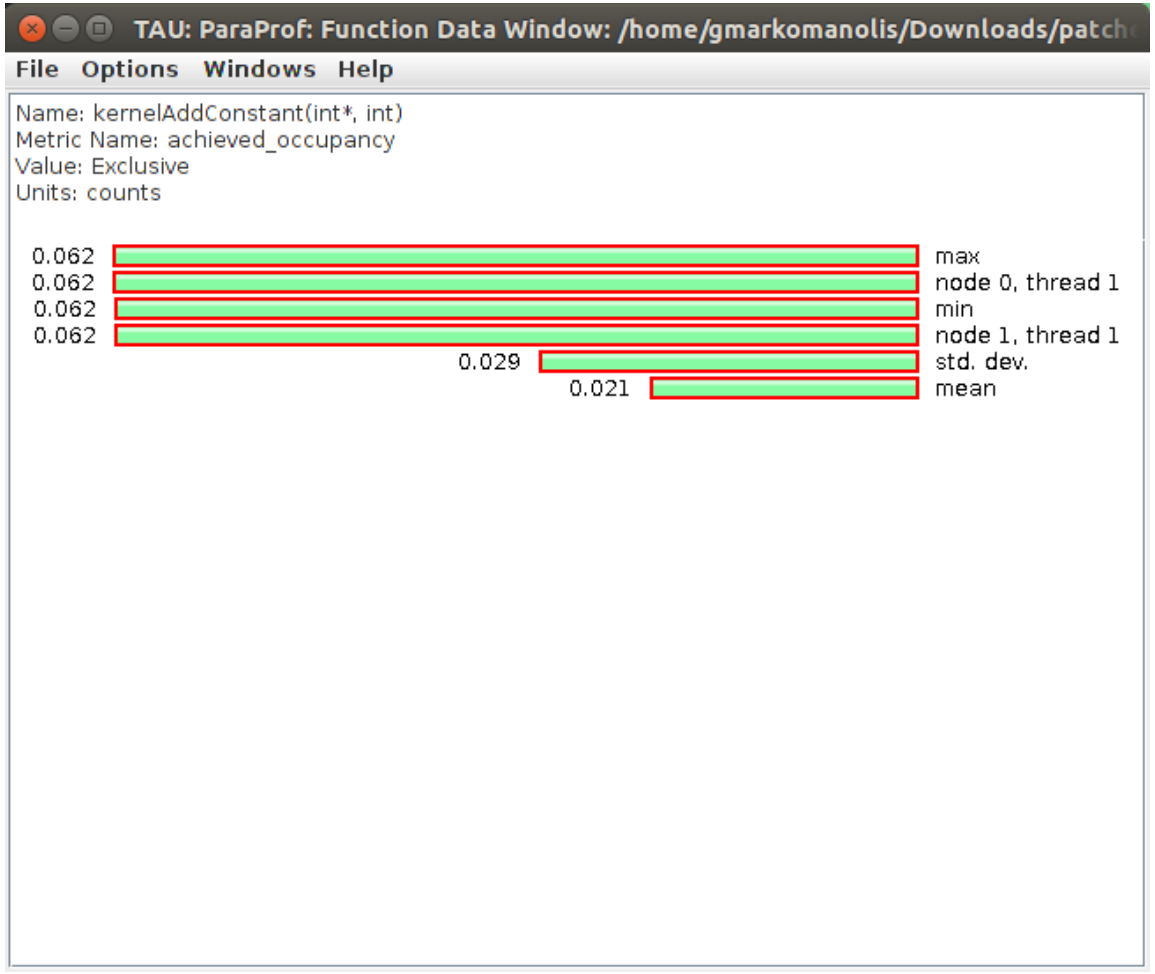
LSMS - Paraprof – MPI+OpenMP+CUDA

- Select the metric achieved occupancy



LSMS - Paraprof – MPI+OpenMP+CUDA

- Click on the colored bar
- The achieved occupancy for this simple benchmark is 6.2%



TAU and GPU

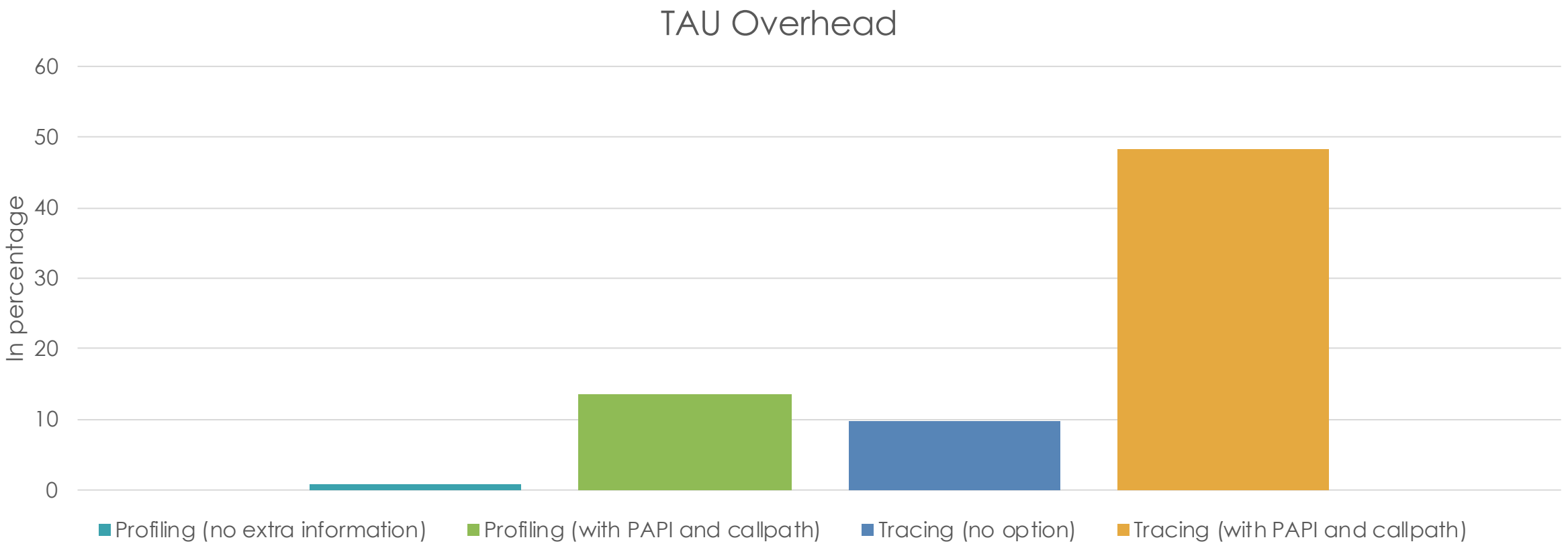
- Similar approach for other metrics, not all of them can be used.
- TAU provides a tool called **tau_cupti_avail** where we can see the list of available metrics, then we have to figured out which CUPTI metrics use these ones.

TAU and tracing

- export TAU_TRACE=1
- export TAU_TRACE_FORMAT=otf2

- Currently, supported MPI and OpenSHMEM applications
- Use Vampir for visualization

Overhead



Should use PDT to exclude files/routines that cause overhead

TAU mechanisms

- TAU_THROTTLE
  - TAU by default excludes from the instrumentation routines that could cause overhead
  - Rule: If a routine is called more than 100,000 times and it spends up to 10 usecs/call, then exclude it.
  - Adjustable: TAU_THROTTLE_NUMCALLS, TAU_THROTTLE_PERCALL

Selective Instrumentation

- Do not instrument routine sort*(int *)

File select.tau:

BEGIN_EXCLUDE_LIST

void sort_#(int *)

END_EXCLUDE_LIST

TAU_OPTIONS="-optTauSelectFile=select.tau"

- Dynamic phase

BEGIN_INSTRUMENT_SECTION

dynamic phase name="phase1" file="miniWeather_mpi.cpp" line=300 to line=327

END_INSTRUMENT_SECTION

Static Phase

File phases.tau:

BEGIN_INSTRUMENT_SECTION

static phase name="phase1" file="miniWeather_mpi.cpp" line=300 to line=327

static phase name="phase2" file="miniWeather_mpi.cpp" line=333 to line=346

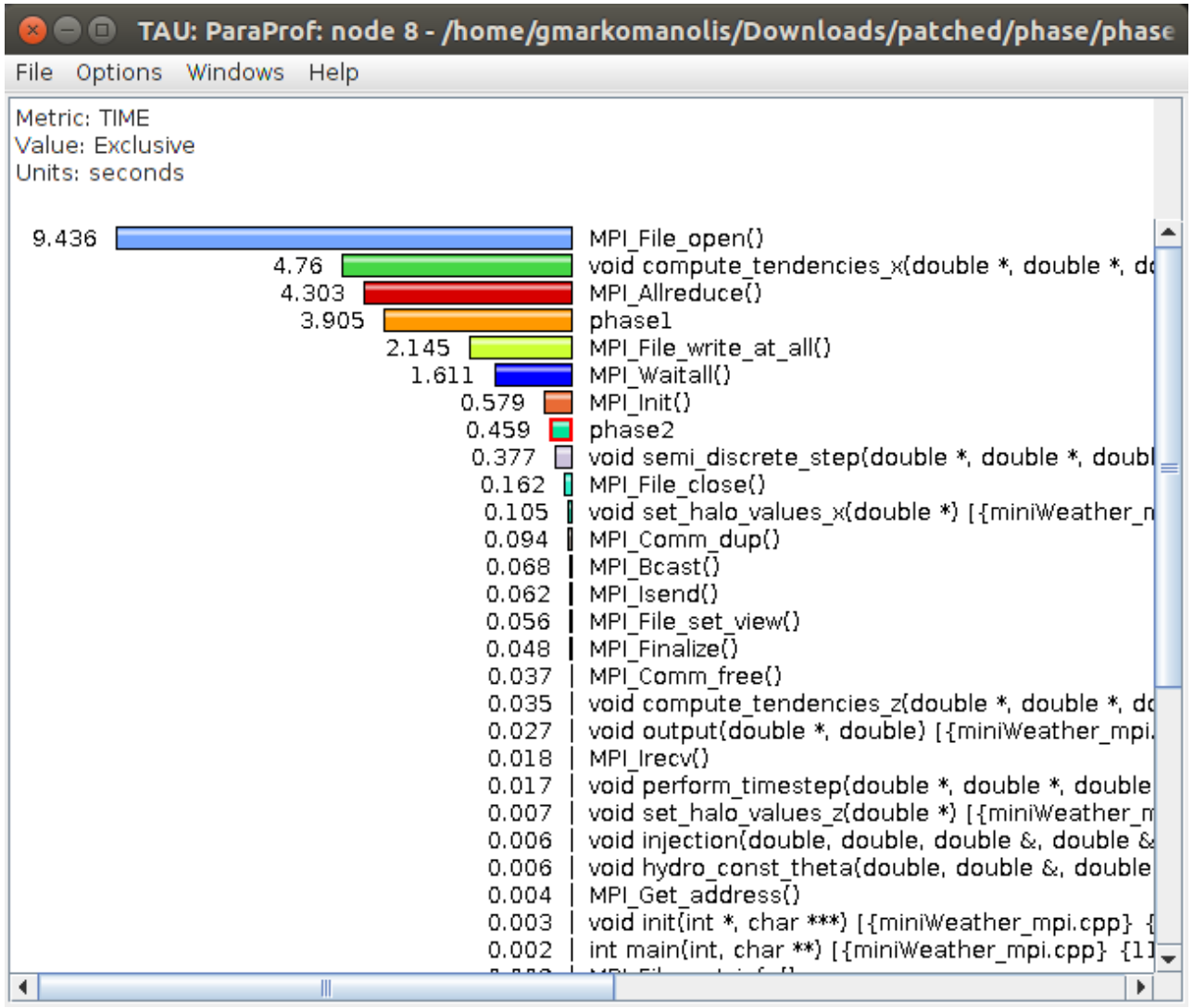
END_INSTRUMENT_SECTION

TAU_OPTIONS="-optTauSelectFile=phases.tau"

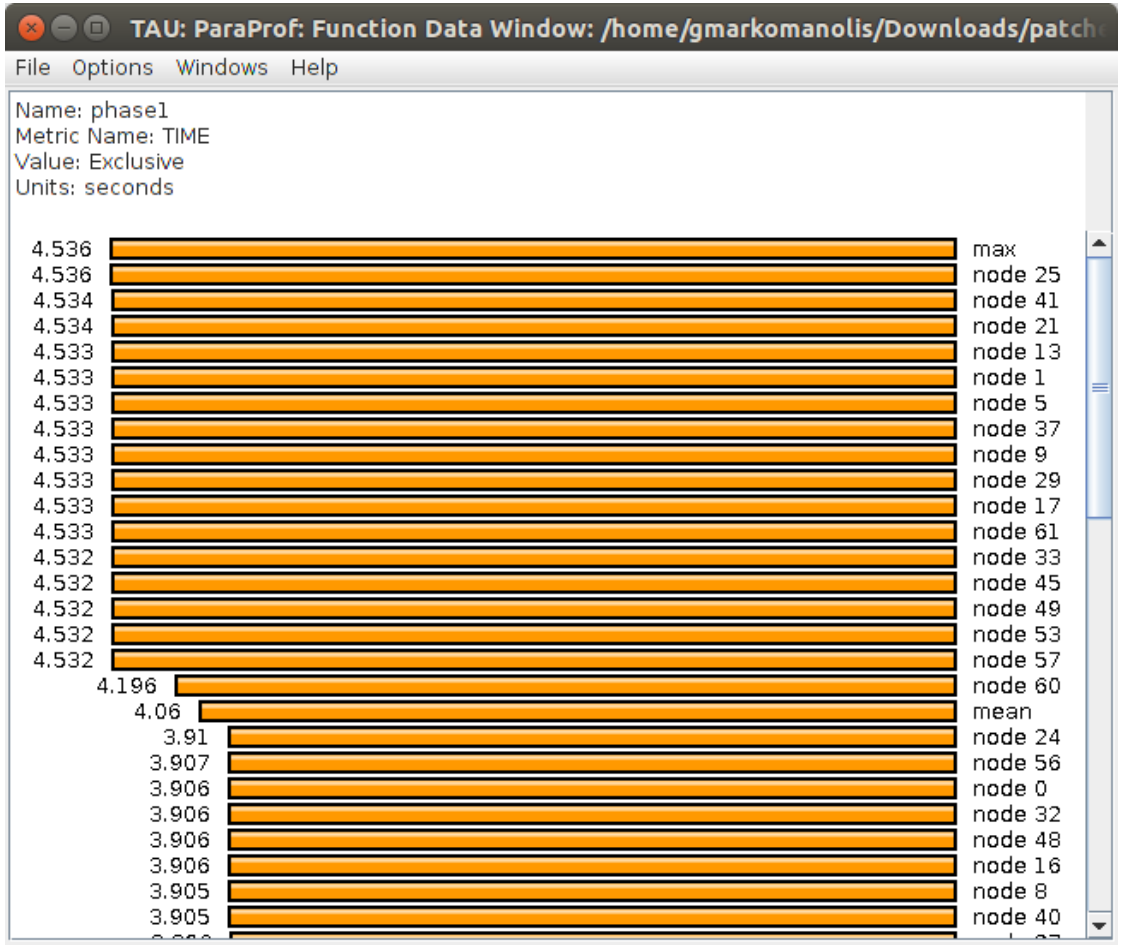
Creating static Phases

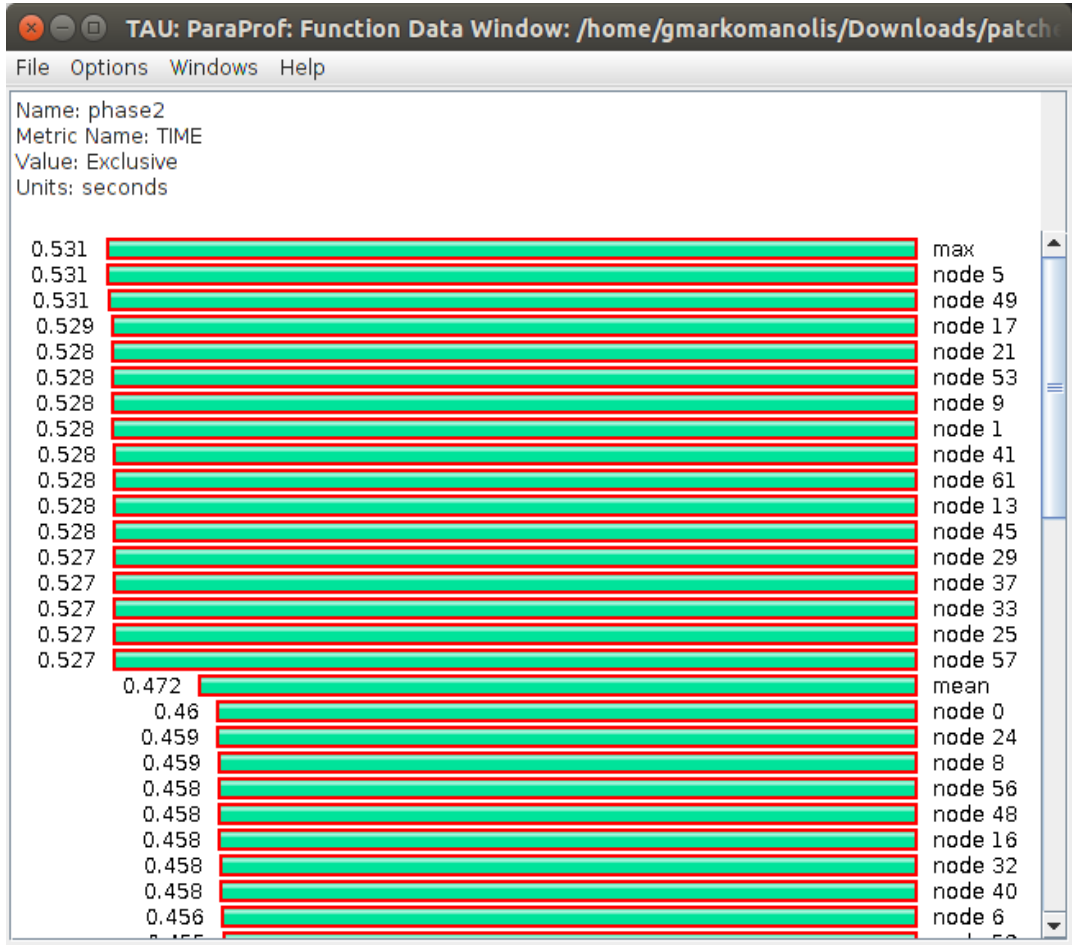
```
for (k=0; k<nz+1; k++) {
  for (i=0; i<nx; i++) {
    //Use fourth-order interpolation from four cell averages to compute the value at the interface in question
    for (ll=0; ll<NUM_VARS; ll++) {
      for (s=0; s<sten_size; s++) {
        inds = ll*(nz+2*hs)*(nx+2*hs) + (k+s)*(nx+2*hs) + i+hs;
        stencil[s] = state[inds];
      }
      //Fourth-order-accurate interpolation of the state
      vals[ll] = -stencil[0]/12 + 7*stencil[1]/12 + 7*stencil[2]/12 - stencil[3]/12;
      //First-order-accurate interpolation of the third spatial derivative of the state
      d3_vals[ll] = -stencil[0] + 3*stencil[1] - 3*stencil[2] + stencil[3];
    }

    //Compute density, u-wind, w-wind, potential temperature, and pressure (r,u,w,t,p respectively)
    r = vals[ID_DENS] + hy_dens_int[k];
    u = vals[ID_UMOM] / r;
    w = vals[ID_WMOM] / r;
    t = ( vals[ID_RHOT] + hy_dens_theta_int[k] ) / r;
    p = C0*pow((r*t),gamm) - hy_pressure_int[k];

    //Compute the flux vector with hyperviscosity
    flux[ID_DENS*(nz+1)*(nx+1) + k*(nx+1) + i] = r*w     - hv_coef*d3_vals[ID_DENS];
    flux[ID_UMOM*(nz+1)*(nx+1) + k*(nx+1) + i] = r*w*u   - hv_coef*d3_vals[ID_UMOM];
    flux[ID_WMOM*(nz+1)*(nx+1) + k*(nx+1) + i] = r*w*w+p - hv_coef*d3_vals[ID_WMOM];
    flux[ID_RHOT*(nz+1)*(nx+1) + k*(nx+1) + i] = r*w*t   - hv_coef*d3_vals[ID_RHOT];
  }
}

////////////////////////////////////////////////
// TODO: THREAD ME
////////////////////////////////////////////////
//Use the fluxes to compute tendencies for each cell
for (ll=0; ll<NUM_VARS; ll++) {
  for (k=0; k<nz; k++) {
    for (i=0; i<nx; i++) {
      indt  = ll* nz   * nx    + k* nx    + i  ;
      indf1 = ll*(nz+1)*(nx+1) + (k  )*(nx+1) + i;
      indf2 = ll*(nz+1)*(nx+1) + (k+1)*(nx+1) + i;
      tend[indt] = -( flux[indf2] - flux[indf1] ) / dz;
      if (ll == ID_WMOM) {
        inds = ID_DENS*(nz+2*hs)*(nx+2*hs) + (k+hs)*(nx+2*hs) + i+hs;
        tend[indt] = tend[indt] - state[inds]*grav;
      }
    }
  }
}
```

Creating static Phases



Creating static Phases

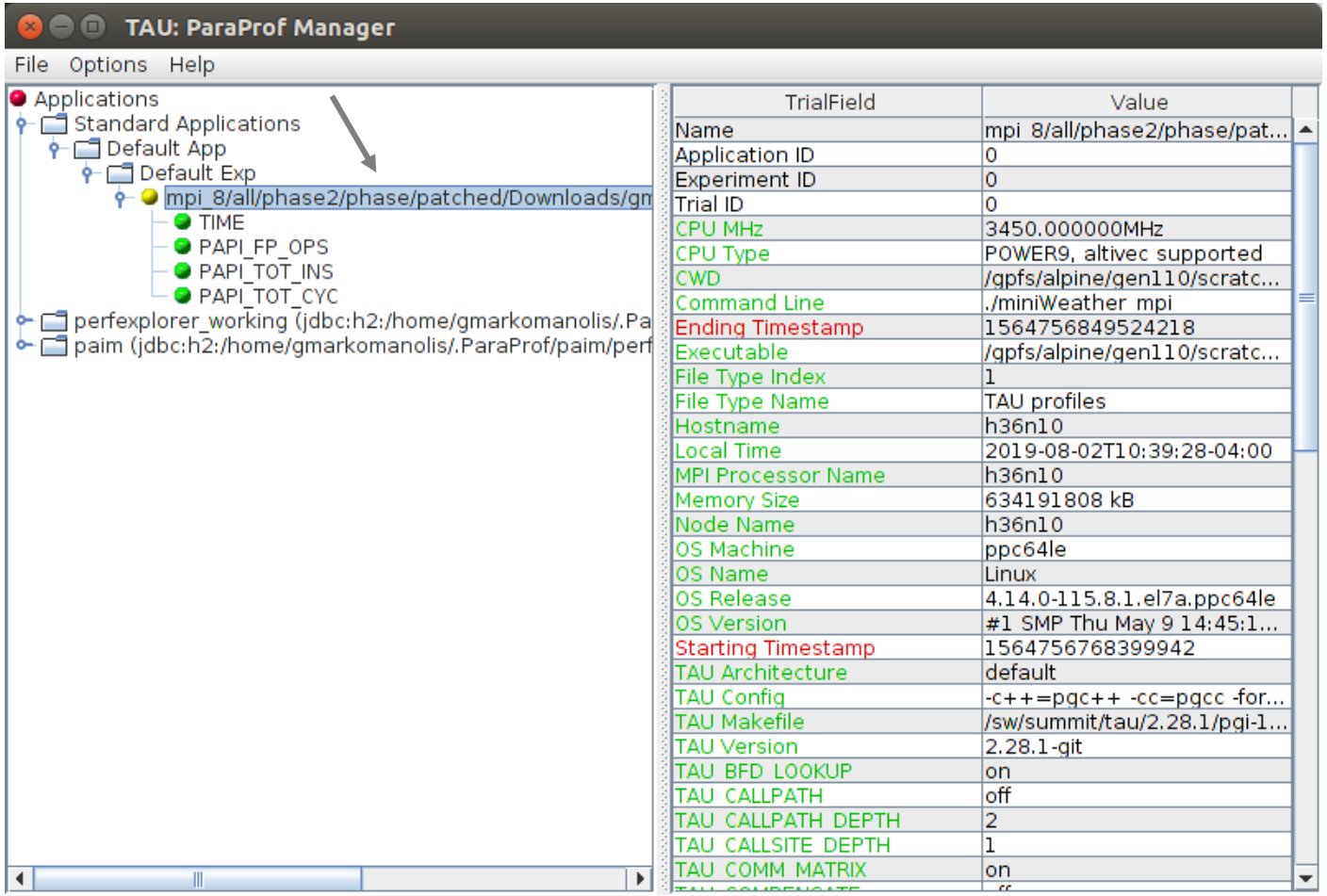




PerfExplorer

- PerfExplorer is a framework for parallel performance data mining and knowledge discovery
- Unfortunately not working efficient on Summit for now
- Perfexplorer should be installed, if it is not configured, it will propose the next steps
- In this example we execute MiniWeather with MPI for 8,16,32,64 processes, we load them to paraprof, select the name of the experiment, right click and select “Upload Trial to DB”

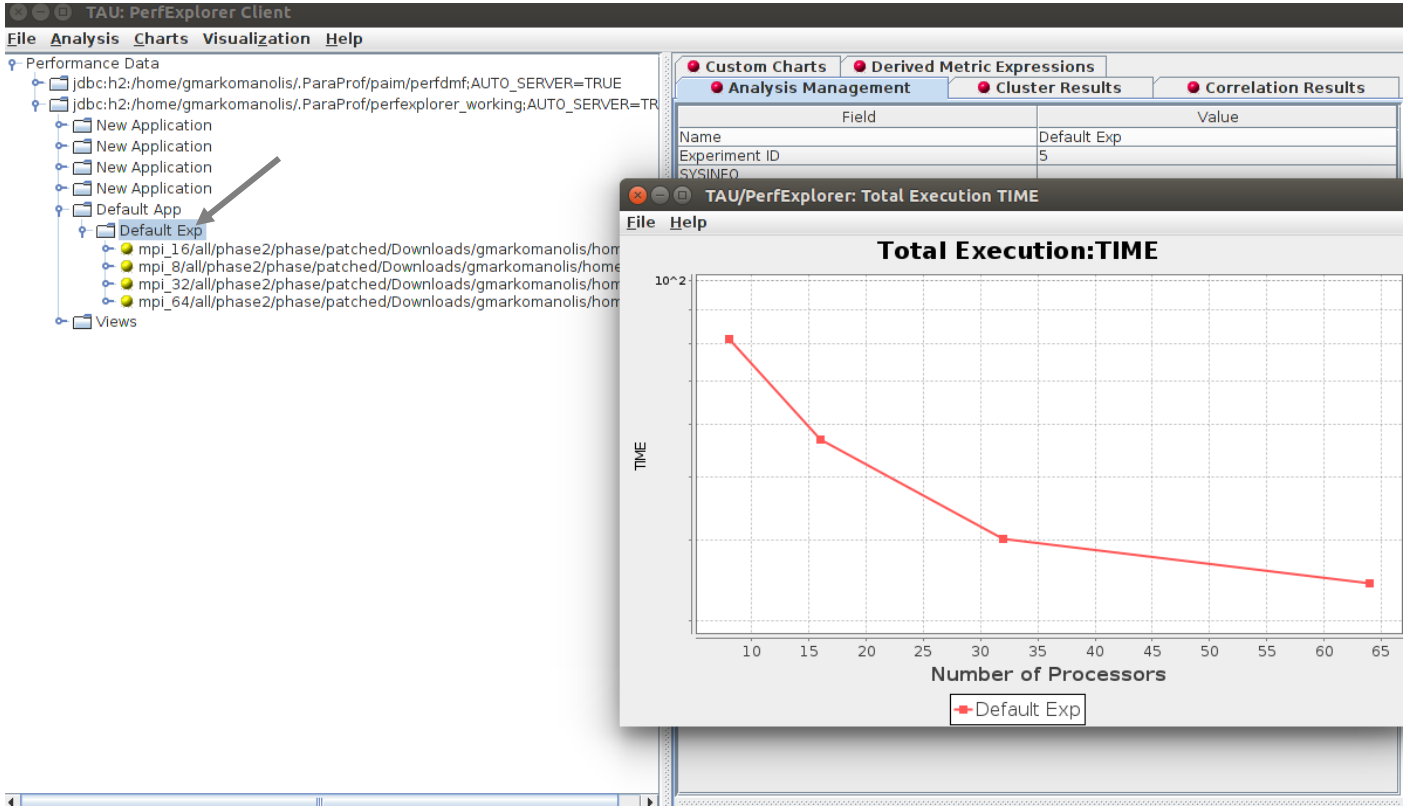
From Paraprof to the DB



TrialField	Value
Name	mpi_8/all/phase2/phase/pat...
Application ID	0
Experiment ID	0
Trial ID	0
CPU MHz	3450.000000MHz
CPU Type	POWER9, altivec supported
CWD	/gpfs/alpine/gen110/scratc...
Command Line	./miniWeather_mpi
Ending Timestamp	1564756849524218
Executable	/gpfs/alpine/gen110/scratc...
File Type Index	1
File Type Name	TAU profiles
Hostname	h36n10
Local Time	2019-08-02T10:39:28-04:00
MPI Processor Name	h36n10
Memory Size	634191808 kB
Node Name	h36n10
OS Machine	ppc64le
OS Name	Linux
OS Release	4.14.0-115.8.1.el7a.ppc64le
OS Version	#1 SMP Thu May 9 14:45:1...
Starting Timestamp	1564756768399942
TAU Architecture	default
TAU Config	-c++=pgc++ -cc=pgcc -for...
TAU Makefile	/sw/summit/tau/2.28.1/pgi-1...
TAU Version	2.28.1-git
TAU_BFD_LOOKUP	on
TAU_CALLPATH	off
TAU_CALLPATH_DEPTH	2
TAU_CALLSITE_DEPTH	1
TAU_COMM_MATRIX	on

Right click on the highlighted name of the experiment with the path and select "Upload Trial to DB", repeat for all the experiments

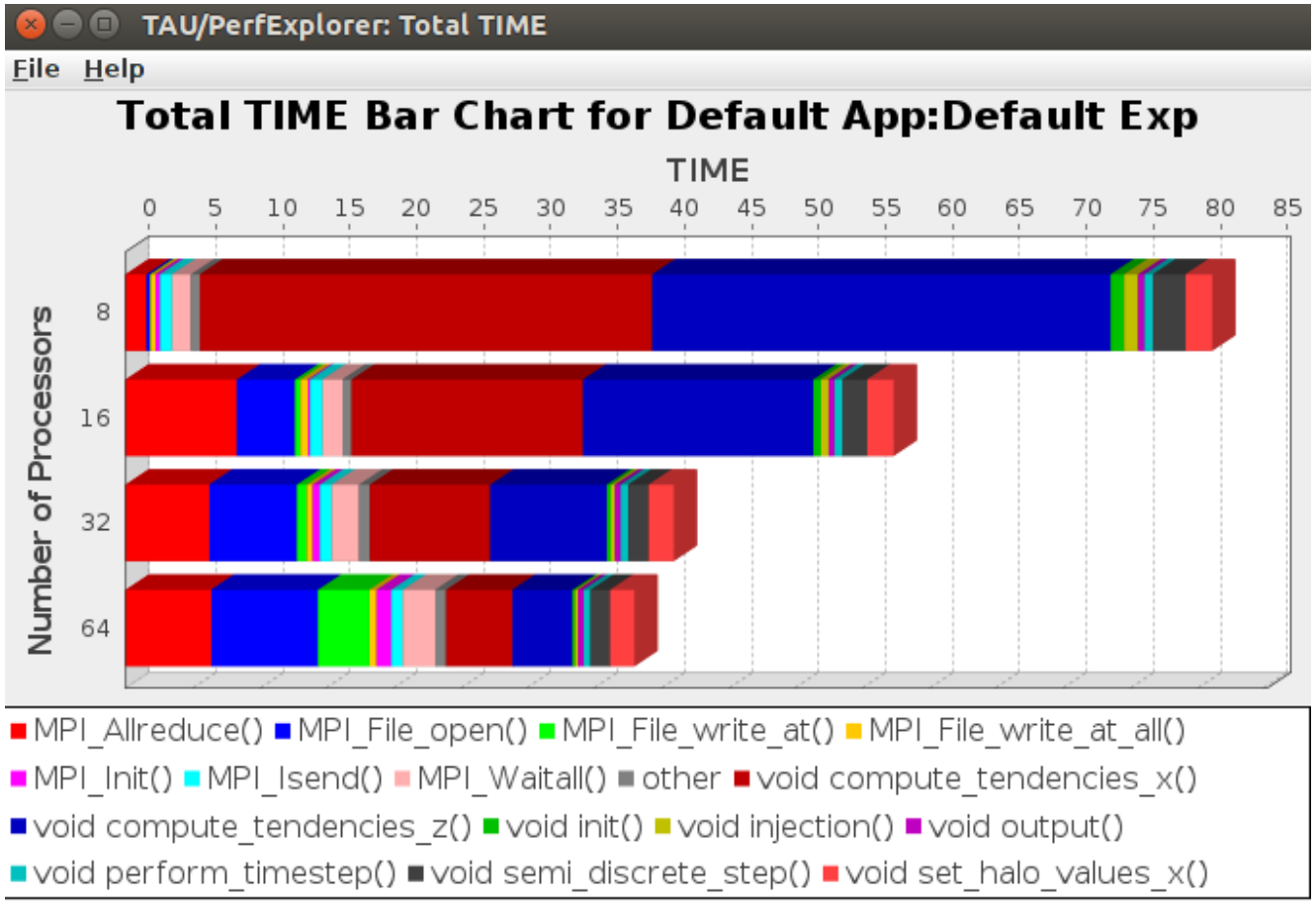
PerfExplorer – Total Execution



All the menus start from the main window of the PerfExplorer

Select the name of the experiment (arrow) and then select Charts -> Total Execution Time -> Select the metric TIME-> Click OK

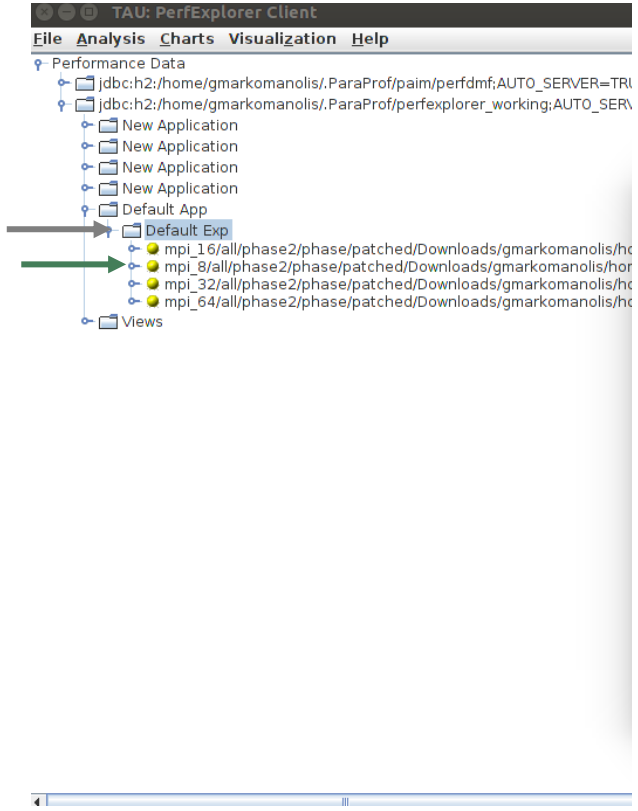
PerfExplorer – Total Execution



Although the duration of computation decreases as we increase the number of the processes, some MPI calls remain similar duration and the MPI_File_write_at is increasing.

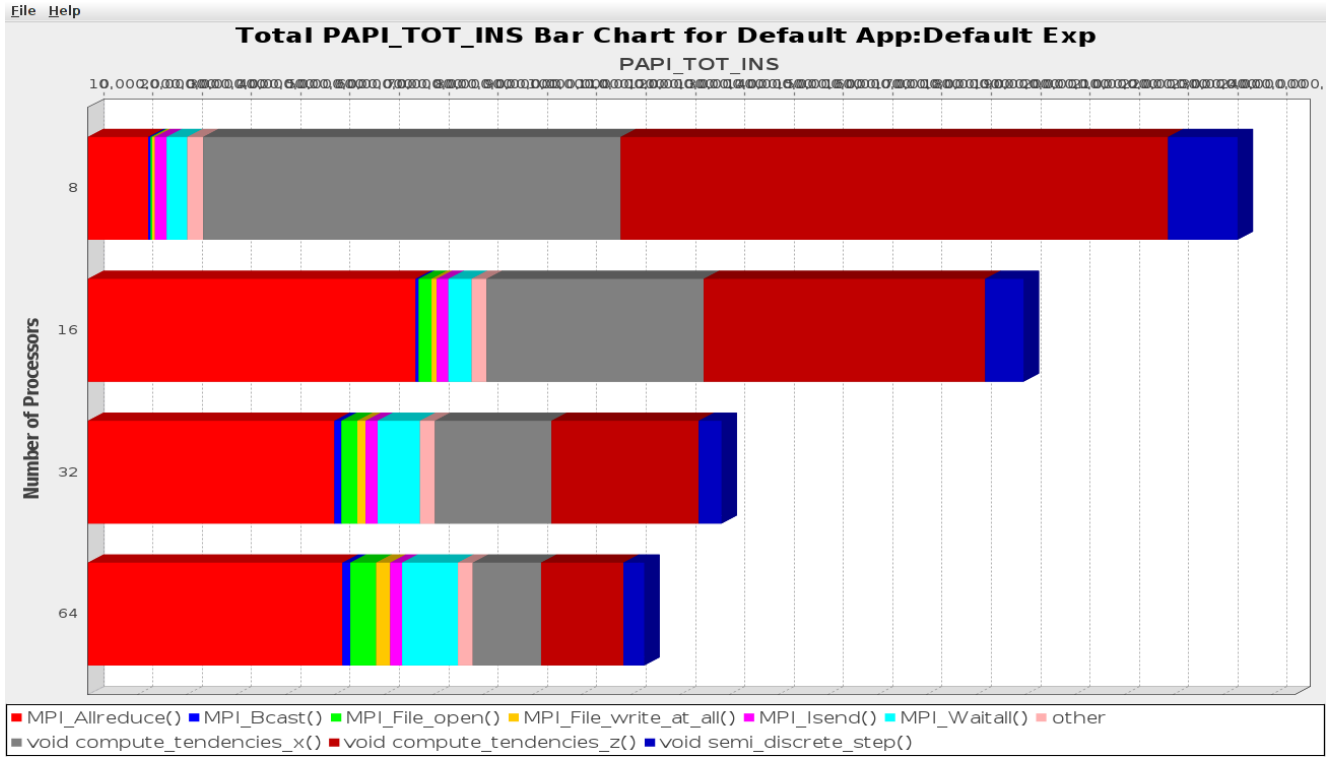
Click Charts -> Stacked Bar Chart -> Select the metric TIME -> Click OK

PerfExplorer – Tip



If the data include more than one metric, and the user wants to visualize another metric, then click in any of the experiments in the green arrow and then back to the name of the experiments (grey arrow). Then in the next visualization, PerfExplorer will require to choose which metric to visualize.

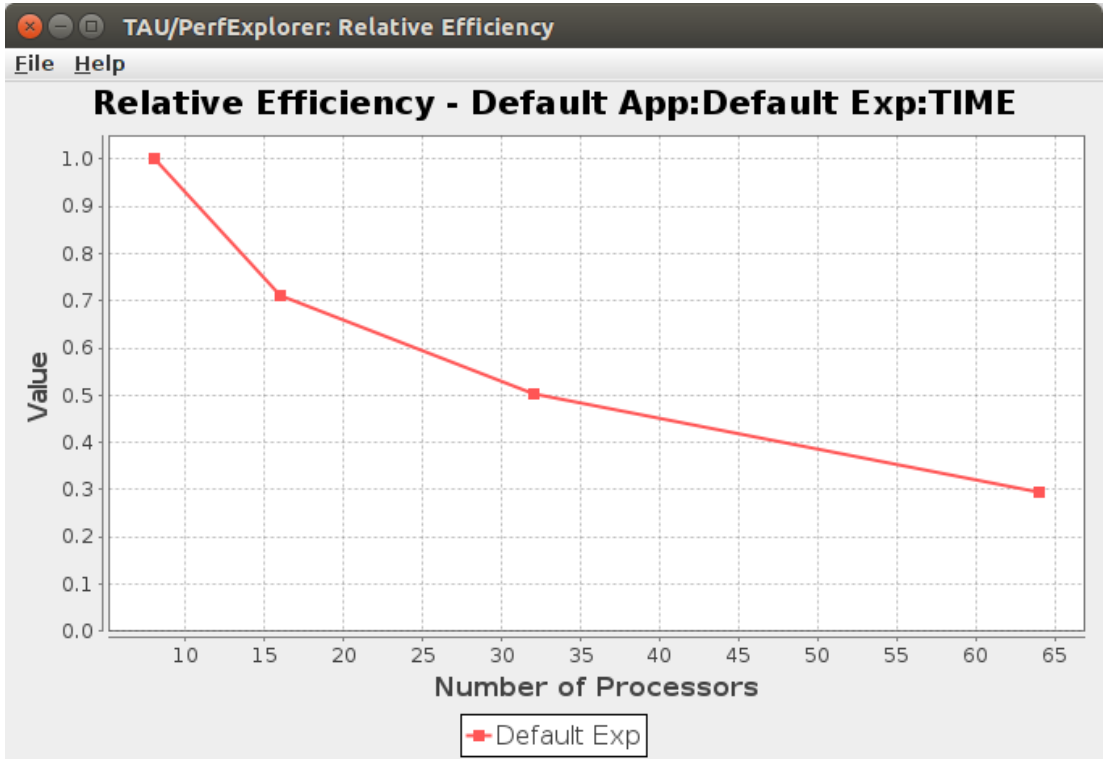
PerfExplorer – PAPI_TOT_INS



MPI calls are increasing instructions as we increase the number of MPI processes.

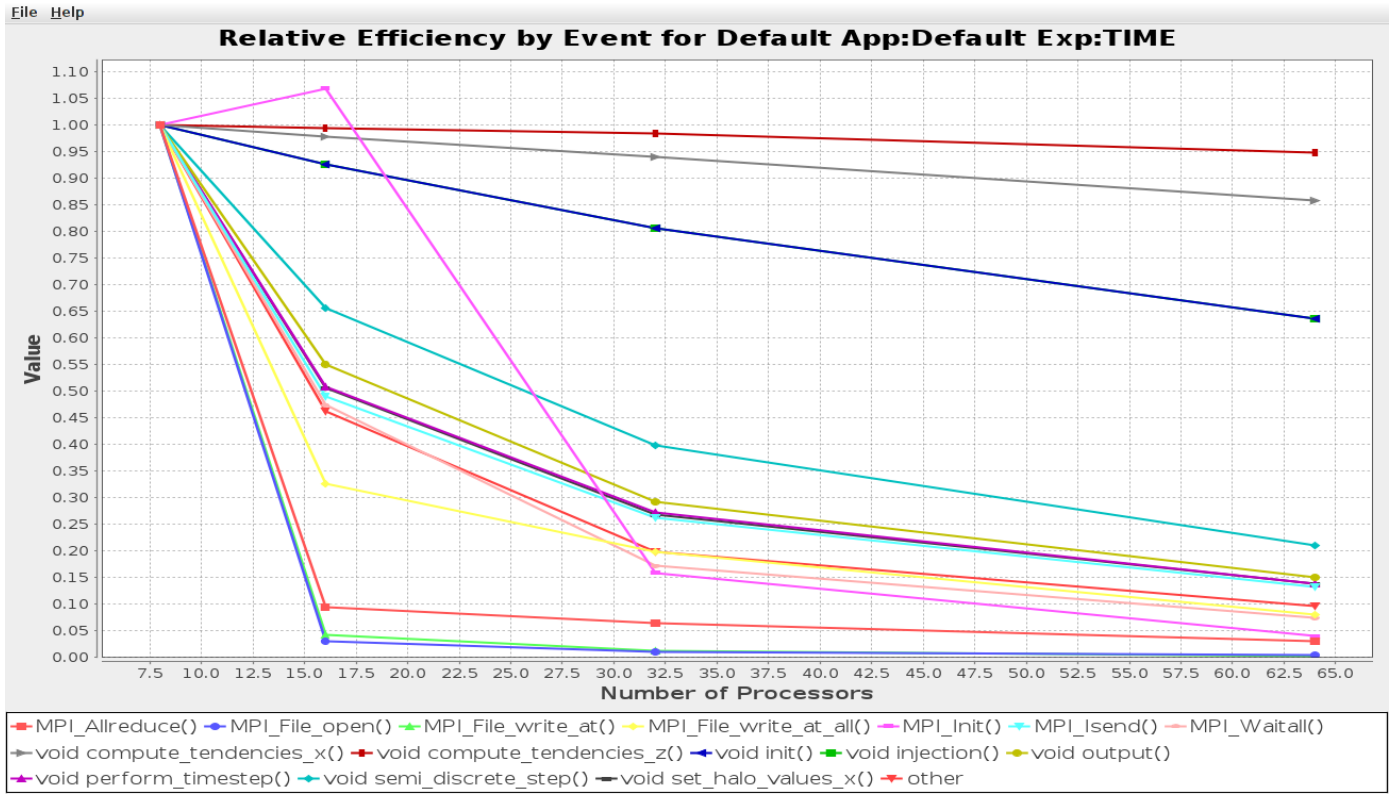
Click Charts -> Stacked Bar Chart -> Select the metric TIME -> Click OK

PerfExplorer – Relative efficiency



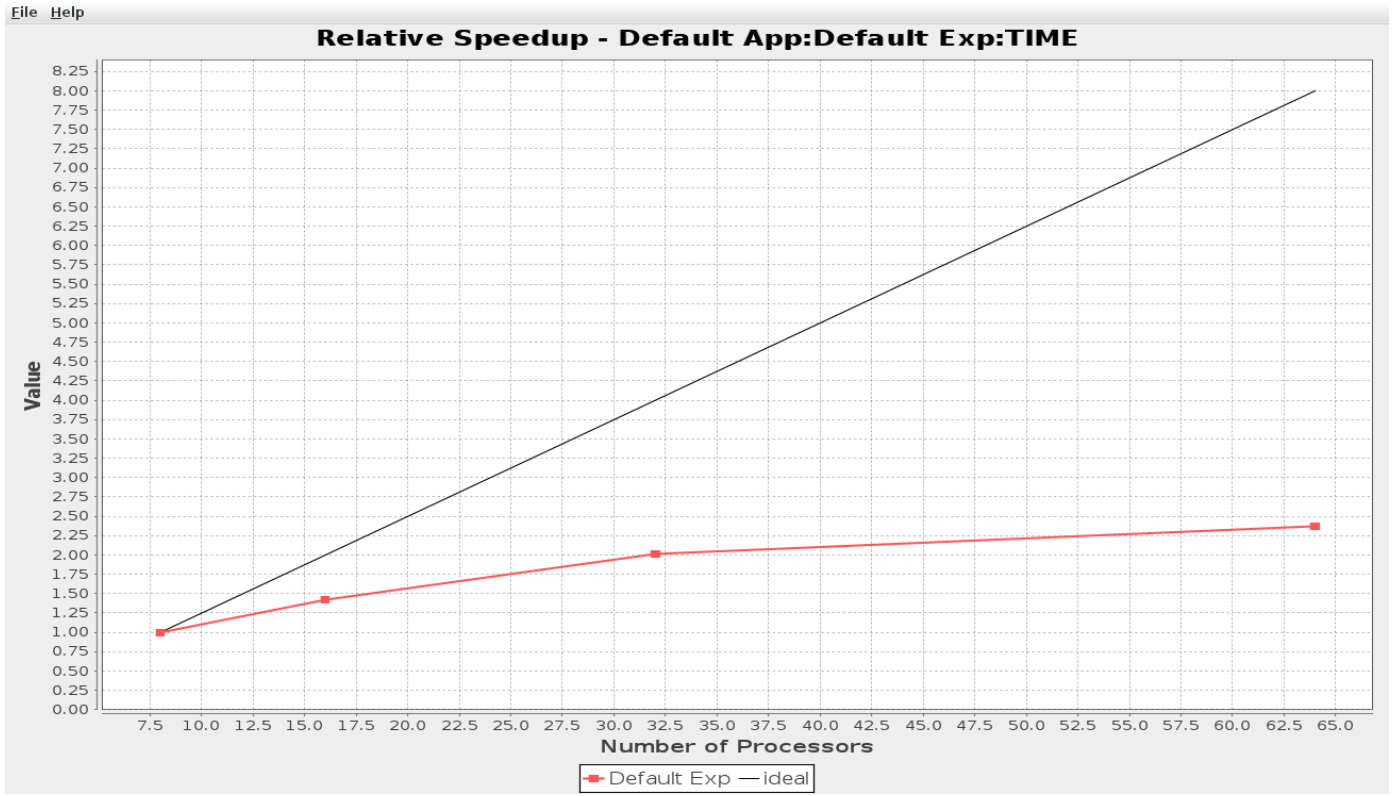
Click Charts -> Relative Efficiency -> “The problem size remains constant” -> Click OK

PerfExplorer – Relative Efficiency by event



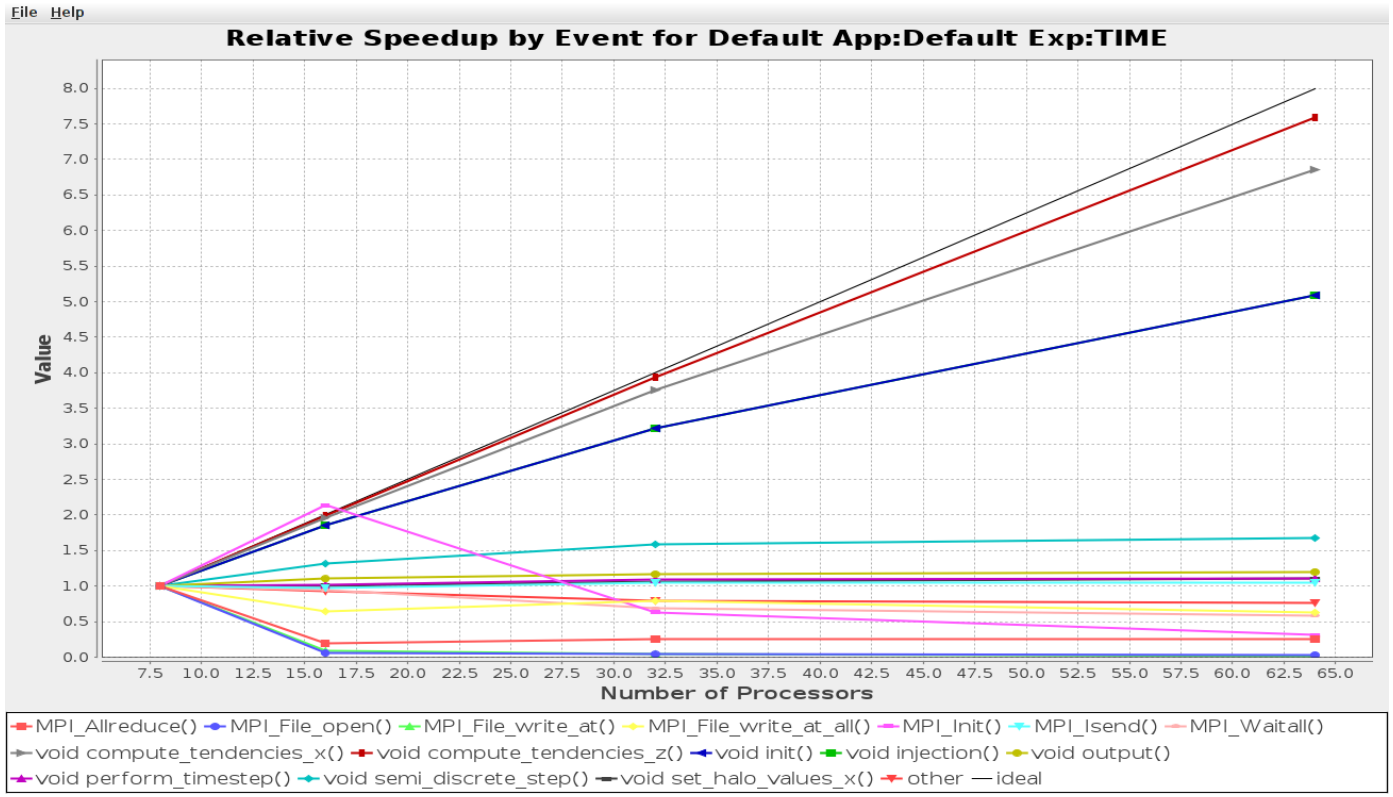
Click Charts -> Relative Efficiency by Event -> Metric TIME -> “The problem size remains constant” -> Click OK

PerfExplorer – Relative Speedup



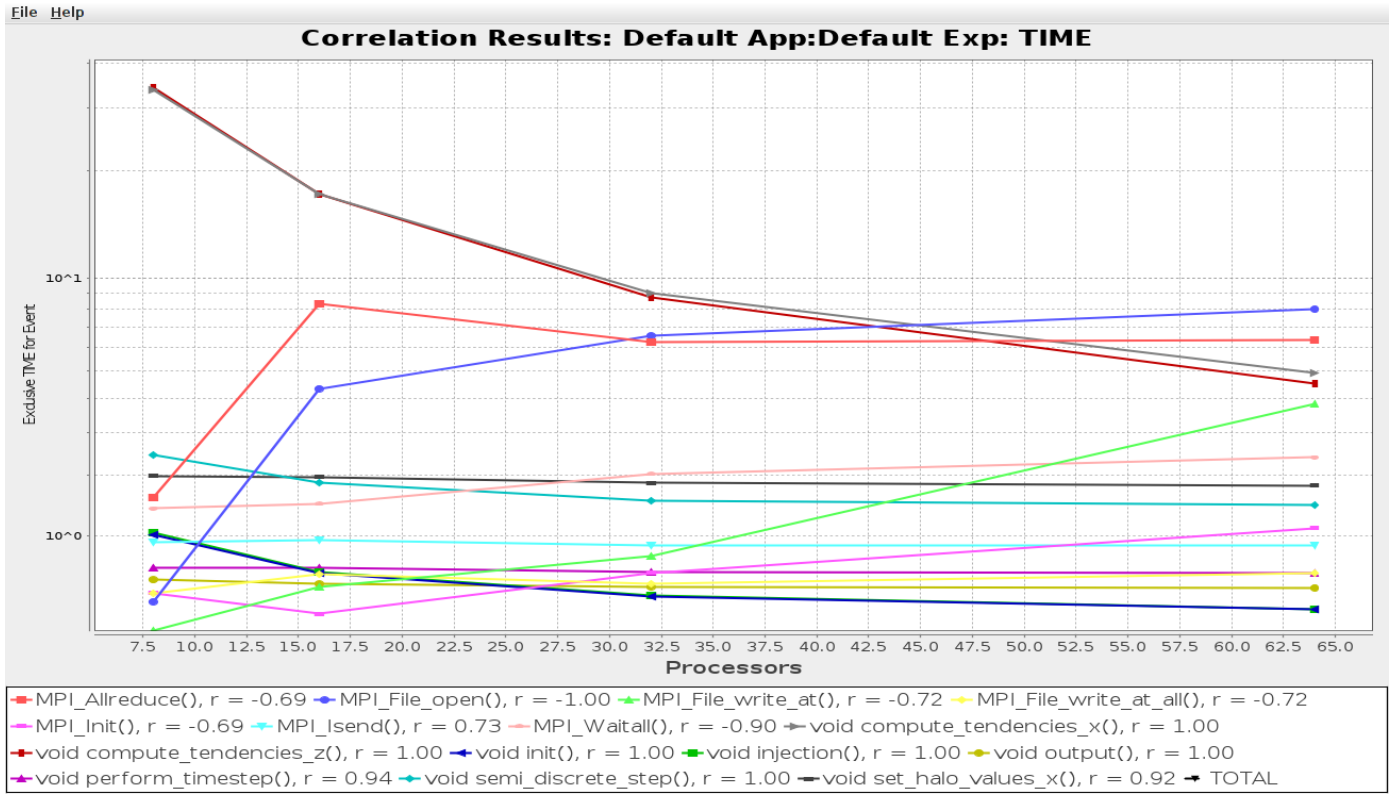
Click Charts -> Relative Speedup -> Metric TIME -> Click OK -> "The problem size remains constant" -> Click OK

PerfExplorer – Relative Speedup



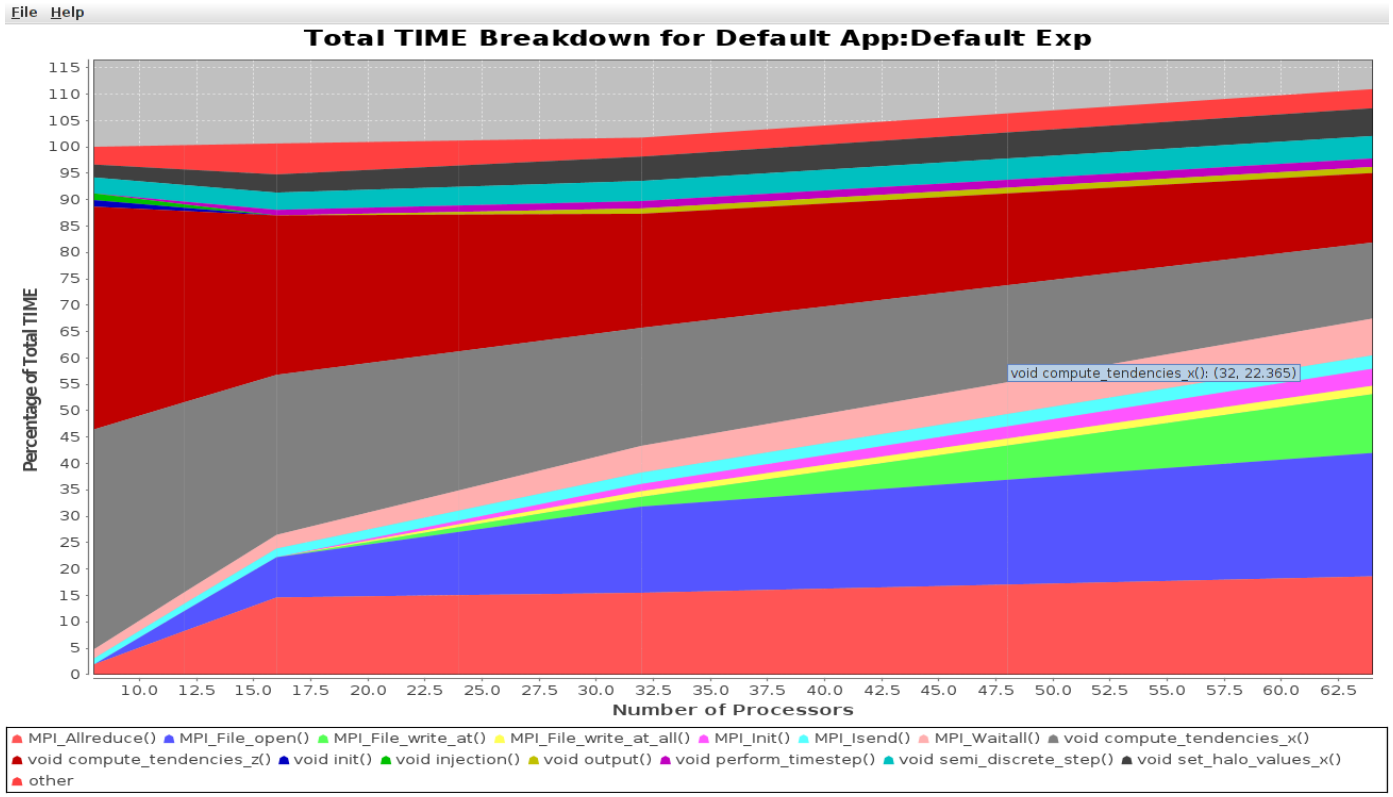
Click Charts -> Relative Speedup by Event -> Metric TIME -> Click OK -> "The problem size remains constant" -> Click OK

PerfExplorer – Relative Speedup



Click Charts -> Correlate Events with Total Runtime -> Metric TIME -> Click OK -> "The problem size remains constant" -> Click OK

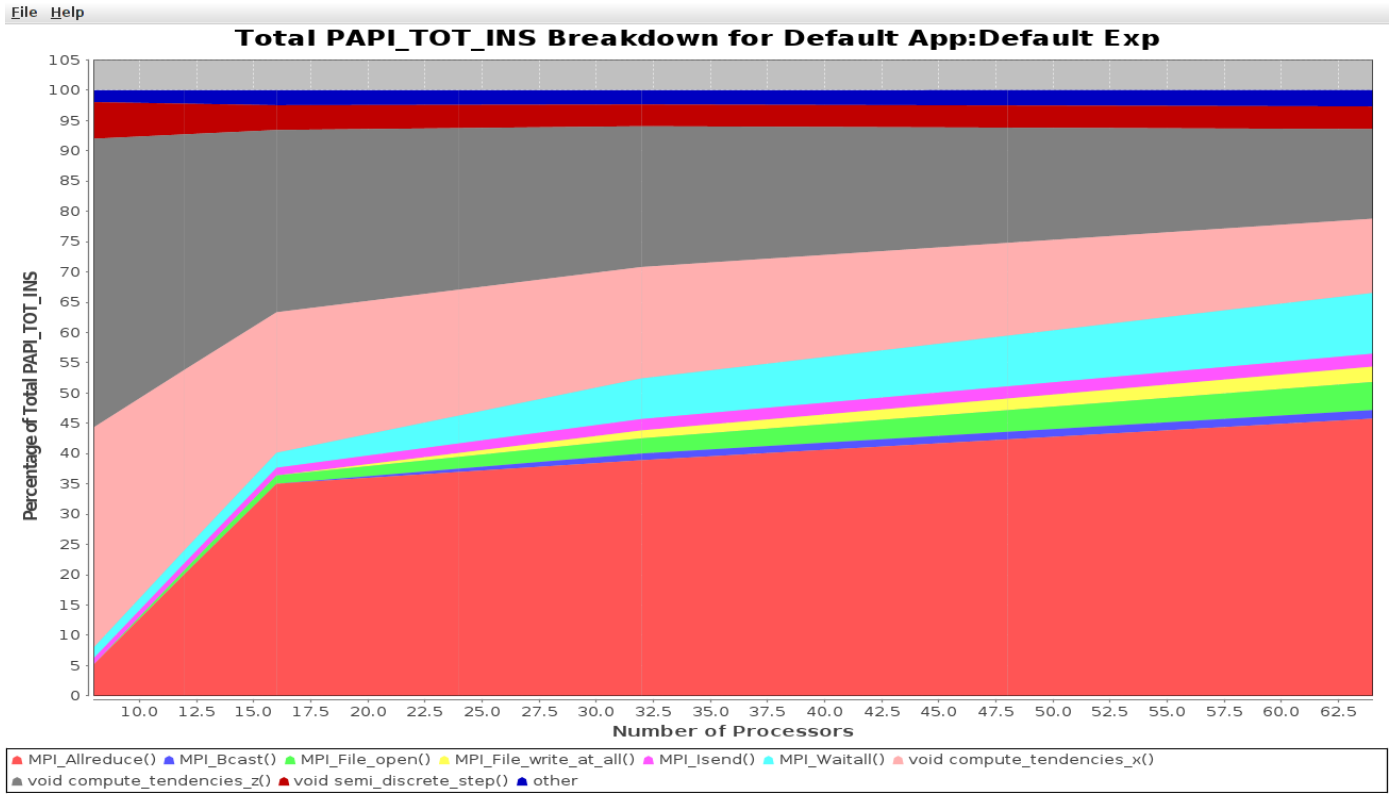
PerfExplorer – Relative Speedup



Click Charts -> Runtime Breakdown -> Metric TIME -> Click OK -> “The problem size remains constant” -> Click OK

OAK RIDGE National Laboratory

PerfExplorer – Relative Speedup



Click Charts -> Runtime Breakdown -> Metric PAPI_TOT_INS -> Click OK ->
"The problem size remains constant" -> Click OK

Conclusions

- TAU is a promising tool, with some good features to be improved related to GPU performance analysis
- Interesting functionalities to identify loop issues and create dynamic phases of an iterative analysis
- We would like OTF2 traces with CUDA/OpenACC
- OpenACC with support of CUPTI metrics is coming around to SC19
- It supports Python instrumentation but not activated during compilation