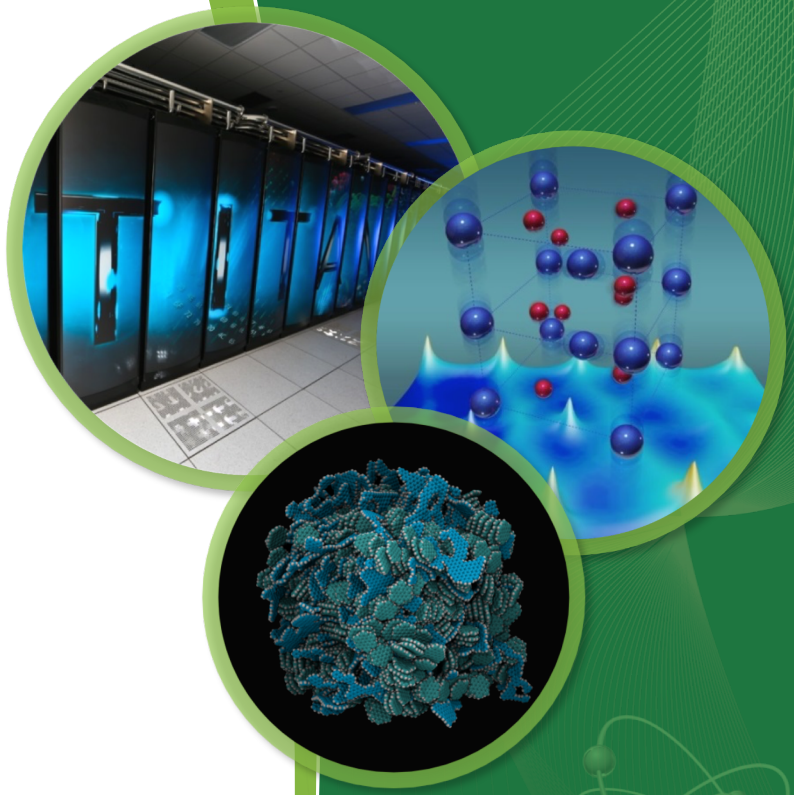
Advanced Score-P

Summit Profiling Tools Workshop

Oak Ridge National Laboratory

August 8, 2019

Mike Brim

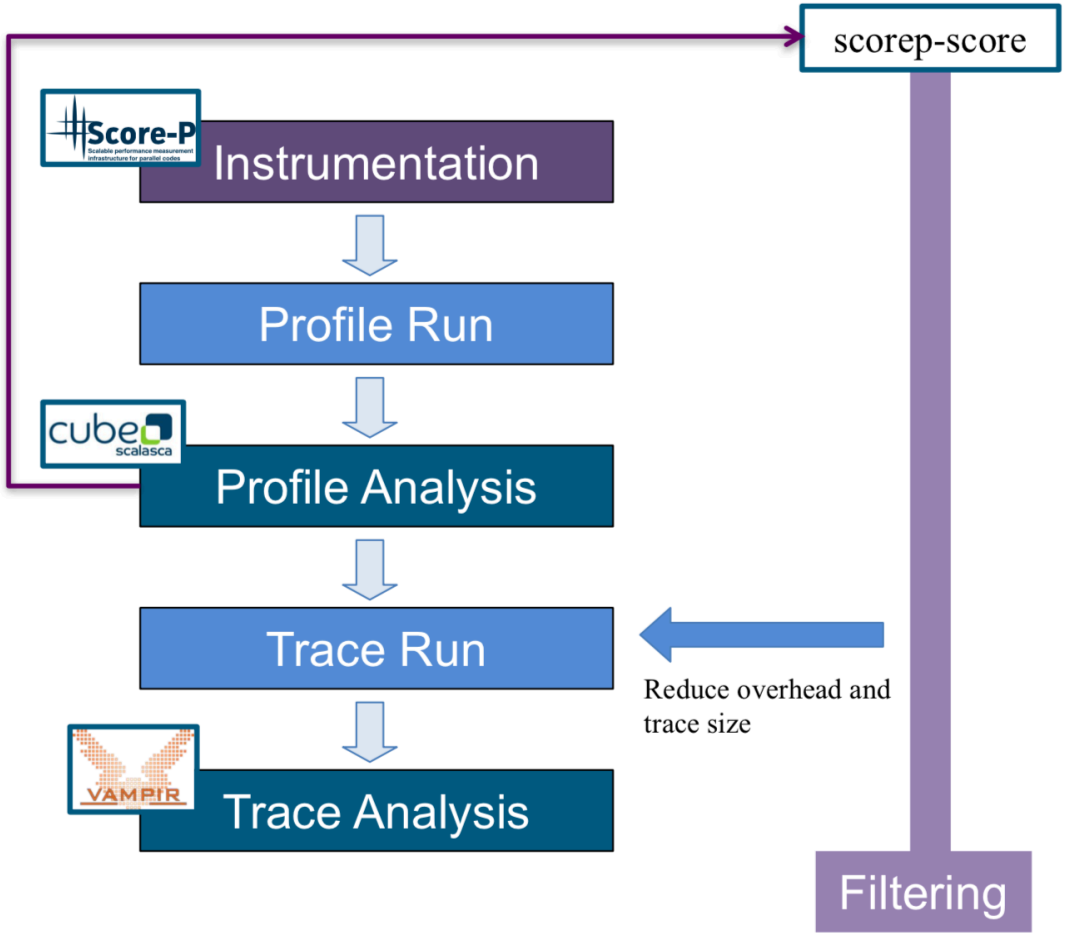


Score-P: *Scalable Performance Measurement Infrastructure for Parallel Codes*

- Project Home Page, Support List
  - https://www.vi-hps.org/projects/score-p/
  - support@score-p.org
- User Manual (v6.0)
  - http://scorepci.pages.jsc.fz-juelich.de/scorep-pipelines/docs/scorep-6.0/html/
  - On Summit: $SCOREP_DIR/share/doc/scorep/pdf/scorep.pdf
- OLCF Software Page
  - https://www.olcf.ornl.gov/software_package/score-p/

Score-P Workflow

1. Source code instrumentation
   - automatic via compiler wrapper
   - manual instrumentation of interesting code regions
2. Profiling runs
   - run in profiling mode
   - analyze profile results
   - optional filtering
3. Tracing runs
   - set tracing configuration environment
   - run in tracing mode
   - analyze traces
   - repeat as desired for alternative configs



Outline

- Case Study: Production application - LSMS
- Advanced manual code instrumentation
- Using hardware counters
- Analysis of large traces

Case Study: LSMS

Case Study: LSMS

- Production application
  - used for Summit acceptance testing
    - had input decks across a wide range of node counts
- C++ : always a challenge for instrumentation-based tools
- MPI + OpenMP + CUDA
  - designed to fully exploit Summit node hardware
    - 1 MPI process per GPU (6 per node)
    - 7 OpenMP threads per process (using 7 cores)
    - CUDA streams for every OpenMP thread

LSMS – Step 1: Code Instrumentation

- Used prefix method in Makefile

- 1 problem encountered
  - GPU code included <omp.h>, but OPARI instrumentation produced modified source files that nvcc could not compile due to missing OpenMP methods
  - Solution: use '`--noopenmp --thread=none`' for nvcc compilations

LSMS – Step 2: Profiling Runs (2 node Fe case)

```
> scorep-score profile.cubex
Estimated aggregate size of event trace:                   71GB
Estimated requirements for largest trace buffer (max_buf): 36GB
Estimated memory requirements (SCOREP_TOTAL_MEMORY):       36GB
(warning: The memory requirements cannot be satisfied by Score-P to avoid
 intermediate flushes when tracing. Set SCOREP_TOTAL_MEMORY=4G to get the
 maximum supported memory or reduce requirements using USR regions filters.)

flt     type     max_buf[B]        visits time[s] time[%] time/visit[us]  region
         ALL 37,758,243,845 2,906,934,514 3724.57   100.0          1.28  ALL
         USR 37,757,439,432 2,906,899,498 3689.63    99.1          1.27  USR
         OMP        757,272        32,688   31.91     0.9        976.24  OMP
         MPI         29,220           936    1.67     0.0       1781.03  MPI
         COM         17,880         1,390    1.36     0.0        979.62  COM
      SCOREP             41             2    0.00     0.0        232.46  SCOREP
```

C++ strikes again!!

LSMS – Step 2: Profiling Runs (2 node Fe case)

```
> scorep-score -r profile.cubex | fgrep -v ALL | awk '$5 > 5.0 {print $0}'
Estimated aggregate size of event trace:                   71GB
Estimated requirements for largest trace buffer (max_buf): 36GB
(warning: The memory requirements cannot be satisfied by Score-P to avoid
 intermediate flushes when tracing. Set SCOREP_TOTAL_MEMORY=4G to get the
 maximum supported memory or reduce requirements using USR regions filters.)
flt     type     max_buf[B]        visits time[s] time[%] time/visit[us]  region
         USR 37,757,439,432 2,906,899,498 3689.63    99.1           1.27  USR
         USR 22,486,328,384 1,729,717,541  262.36     7.0           0.15  dfv_m
         USR  1,064,960,000    81,920,000  344.64     9.3           4.21  void bulirsch_stoer_integrator_(double*,
double*, double*, double*, int*, double*, std::complex<double>*, double*, double*, double*, double*, int*)

         USR         66,560         5,120 1008.38    27.1      196948.52  void
zblock_lu_cuda_c_(std::complex<double>*, int*, int*, int*, int*, int*, int*, int*)
         USR         66,560         5,120 1376.11    36.9      268770.57  void buildKKRMatrix(LSMSSystemParameters&,
LocalTypeInfo&, AtomData&, Complex, Complex, int, Matrix<std::complex<double> >&)
```

Most time spent in these two functions. Let's create a focused filter to eliminate extra 36GB of trace data per process.

LSMS – Step 2: Profiling Runs (2 node Fe case)

```
> cat lsms-scorep-region-filter.txt
SCOREP_REGION_NAMES_BEGIN
  EXCLUDE *
  INCLUDE main
          calculateTau*
          buildKKRMatrix*
          *cuda*
          *$omp*
          MPI_*
SCOREP_REGION_NAMES_END
```

Eliminated 37GB of potential trace data!!

```
> scorep-score profile.cubex -f lsms-scorep-region-filter.txt

Estimated aggregate size of event trace:                   1537kB
Estimated requirements for largest trace buffer (max_buf): 769kB
Estimated memory requirements (SCOREP_TOTAL_MEMORY):       19MB
(hint: When tracing set SCOREP_TOTAL_MEMORY=19MB to avoid intermediate flushes
 or reduce requirements using USR regions filters.)

flt     type     max_buf[B]        visits time[s] time[%] time/visit[us]  region
 -       ALL 37,758,243,845 2,906,934,514 3724.57   100.0           1.28  ALL
 -       USR 37,757,439,432 2,906,899,498 3689.63    99.1           1.27  USR
 -       OMP        757,272        32,688   31.91     0.9         976.24  OMP
 -       MPI         29,220           936    1.67     0.0        1781.03  MPI
 -       COM         17,880         1,390    1.36     0.0         979.62  COM
 -    SCOREP             41             2    0.00     0.0         232.46  SCOREP

 *       ALL        786,533        33,626   33.58     0.9         998.60  ALL-FLT
 +       FLT 37,757,457,312 2,906,900,888 3690.99    99.1           1.27  FLT
 -       OMP        757,272        32,688   31.91     0.9         976.24  OMP-FLT
 -       MPI         29,220           936    1.67     0.0        1781.03  MPI-FLT
 -    SCOREP             41             2    0.00     0.0         232.46  SCOREP-FLT
```

LSMS – Step 2: Profiling Runs (2 node Fe case)

- Let's try CUDA profiling!
  - 💣
  - Cause: TBD

CUDA Profiling Error (Region Exit Mismatch)

```
> cat stderr.txt
[Score-P] src/measurement/profiling/scorep_profile_event_base.c:188: Error: Inconsistent profile. Stop profiling:
Exit event for other than current region occurred at location 6: Expected exit for region 'cudaLaunchKernel'. Exited
region 'cudaLaunchKernel'
[Score-P] src/measurement/profiling/scorep_profile_debug.c:223: Fatal: Cannot continue profiling. Activating core
files (export SCOREP_PROFILING_ENABLE_CORE_FILES=1) might provide more insight.
[Score-P] Please report this to support@score-p.org. Thank you.
[Score-P] Try also to preserve any generated core dumps.
[a28n16:24662] *** Process received signal ***
[a28n16:24662] Signal: Aborted (6)
[a28n16:24662] Signal code:  (-6)
[a28n16:24662] [ 0] [0x2000000504d8]
[a28n16:24662] [ 1] /lib64/libc.so.6(gsignal+0x60)[0x20000a42fbf0]
[a28n16:24662] [ 2] /lib64/libc.so.6(abort+0x18c)[0x20000a431f6c]
[a28n16:24662] [ 3]
/sw/summit/scorep/6.0/gcc-6.4.0/lib/libscorep_measurement.so.0(SCOREP_UTILS_Error_Abort+0x34)[0x20000803db54]
[a28n16:24662] [ 4]
/sw/summit/scorep/6.0/gcc-6.4.0/lib/libscorep_measurement.so.0(scorep_profile_on_error+0x284)[0x2000080151d4]
[a28n16:24662] [ 5]
/sw/summit/scorep/6.0/gcc-6.4.0/lib/libscorep_measurement.so.0(scorep_profile_exit+0x1e4)[0x200008013784]
[a28n16:24662] [ 6]
/sw/summit/scorep/6.0/gcc-6.4.0/lib/libscorep_measurement.so.0(SCOREP_Profile_Exit+0xf0)[0x20000800a530]
[a28n16:24662] [ 7] /sw/summit/scorep/6.0/gcc-6.4.0/lib/libscorep_measurement.so.0(+0x7ae04)[0x20000800ae04]
[a28n16:24662] [ 8]
/sw/summit/scorep/6.0/gcc-6.4.0/lib/libscorep_measurement.so.0(SCOREP_Location_ExitRegion+0xd0)[0x200007fe20d0]
[a28n16:24662] [ 9] /sw/summit/scorep/6.0/gcc-6.4.0/lib/libscorep_adapter_cuda_mgmt.so.0(+0xc9dc)[0x20000855c9dc]
[a28n16:24662] [10] /sw/summit/cuda/10.1.105/extras/CUPTI/lib64/libcupti.so.10.1(+0xef198)[0x20000989f198]
```

LSMS – Step 3: Tracing Runs (2 node Fe case)

- Using filter file
- Problem #1

  Exhausted Score-P memory buffer
- Solution

```
export SCOREP_TOTAL_MEMORY=64M
```

```
> cat stderr.txt
[Score-P] src/adapters/cuda/scorep_cupti4_activity.c:616: Warning: [CUPTI Activity] Reached maximum CUDA buffer size
for context 0x3ae04c10
[Score-P] src/adapters/cuda/scorep_cupti4_activity.c:616: Warning: [CUPTI Activity] Reached maximum CUDA buffer size
for context 0x1c33e840
[Score-P] src/measurement/SCOREP_Memory.c:190: Error: No free memory page available: Out of memory. Please increase
SCOREP_TOTAL_MEMORY=16384000 and try again.
[Score-P] src/measurement/SCOREP_Memory.c:194: Error: No free memory page available: Please ensure that there are at
least 2MB available for each intended location.
[Score-P] src/measurement/SCOREP_Memory.c:198: Error: No free memory page available: Where there are currently 16
locations in use in this failing process.
[Score-P] Memory usage of rank 1
[Score-P] Memory used so far:
[Score-P] Score-P runtime-management memory tracking:
[Score-P] Memory: Requested:
[Score-P] SCOREP_TOTAL_MEMORY [bytes]                          16384000
[Score-P] SCOREP_PAGE_SIZE [bytes]                             8192
[Score-P] Number of pages of size SCOREP_PAGE_SIZE             2000
...
```

LSMS – Step 3: Tracing Runs (2 node Fe case)

- Problem #2

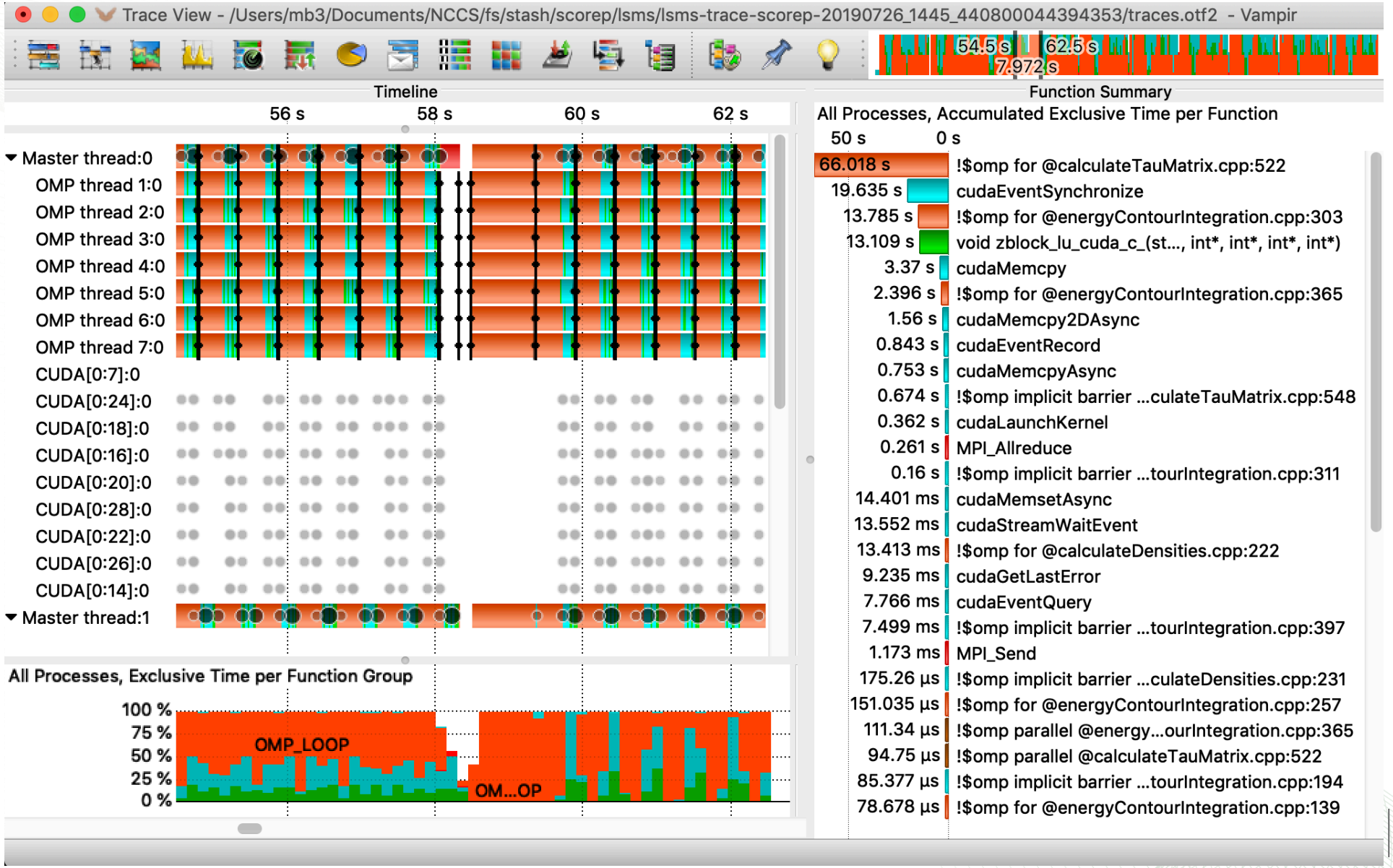
  Exhausted CUDA buffer

- Solution

```
export SCOREP_CUDA_BUFFER=32M
```

```
> cat stderr.txt
[Score-P] src/adapters/cuda/scorep_cupti4_activity.c:616: Warning: [CUPTI Activity] Reached maximum CUDA buffer size
for context 0x19864c30
[Score-P] src/adapters/cuda/scorep_cupti4_activity.c:616: Warning: [CUPTI Activity] Reached maximum CUDA buffer size
for context 0x4adee860
[Score-P] src/adapters/cuda/scorep_cupti_activity.c:562: Warning: [CUPTI Activity] Memcpy: start time < last written
timestamp! (CUDA device:stream [0:18])
[Score-P] src/adapters/cuda/scorep_cupti_activity.c:569: Warning: [CUPTI Activity] Set memcpy start time to
sync-point time (truncate 0.0170%)
[Score-P] src/adapters/cuda/scorep_cupti_activity.c:396: Warning: [CUPTI Activity] Kernel: start time
(438503093463291) < (438503093463306) last written timestamp!
[Score-P] src/adapters/cuda/scorep_cupti_activity.c:399: Warning: [CUPTI Activity] Kernel: 'volta_zgemm_32x32_nn',
CUdevice: 0, CUDA stream ID: 26
[Score-P] src/adapters/cuda/scorep_cupti_activity.c:405: Warning: [CUPTI Activity] Set kernel start time to
sync-point time (truncate 0.0001%)
[Score-P] src/adapters/cuda/scorep_cupti4_activity.c:214: Warning: [CUPTI Activity] Dropped 302124 records. Current
buffer size: 1048576 bytes
To avoid dropping of records increase the buffer size!
Proposed minimum SCOREP_CUDA_BUFFER=28843984
```

LSMS – Step 3: Tracing Runs (2 node Fe case)



Advanced Manual Code Instrumentation

Step 1: Manual Instrumentation - Source Code Region

C, C++

```
#include <scorep/SCOREP_User.h>

void foo() {

  SCOREP_USER_REGION_DEFINE( my_region )

  // more declarations

  SCOREP_USER_REGION_BEGIN( my_region, "foo",
SCOREP_USER_REGION_TYPE_COMMON )

  // do something

  SCOREP_USER_REGION_END( my_region )

}
```

Fortran

```
#include "scorep/SCOREP_User.inc"

subroutine foo

  SCOREP_USER_REGION_DEFINE( my_region )

  ! more declarations

  SCOREP_USER_REGION_BEGIN( my_region, "foo",
SCOREP_USER_REGION_TYPE_COMMON )

  ! do something

  SCOREP_USER_REGION_END( my_region )

end subroutine foo
```

Step 1: Manual Instrumentation – Other Region Types

Phases

```
#include <scorep/SCOREP_User.h>

while (!done) {

  SCOREP_USER_REGION_DEFINE( my_phase )

  // more declarations

  SCOREP_USER_REGION_BEGIN( my_phase,
”timestep", SCOREP_USER_REGION_TYPE_PHASE )

  do_timestep(…);

  SCOREP_USER_REGION_END( my_phase )

}
```

Loops

```
#include <scorep/SCOREP_User.h>

SCOREP_USER_REGION_DEFINE( my_loop )

SCOREP_USER_REGION_BEGIN( my_loop, ”omp_loop_X",
SCOREP_USER_REGION_TYPE_LOOP )

#pragma omp parallel for …
for (i=0; i < n_iter; i++) {
    do_iteration(…);
}

SCOREP_USER_REGION_END( my_loop )
```

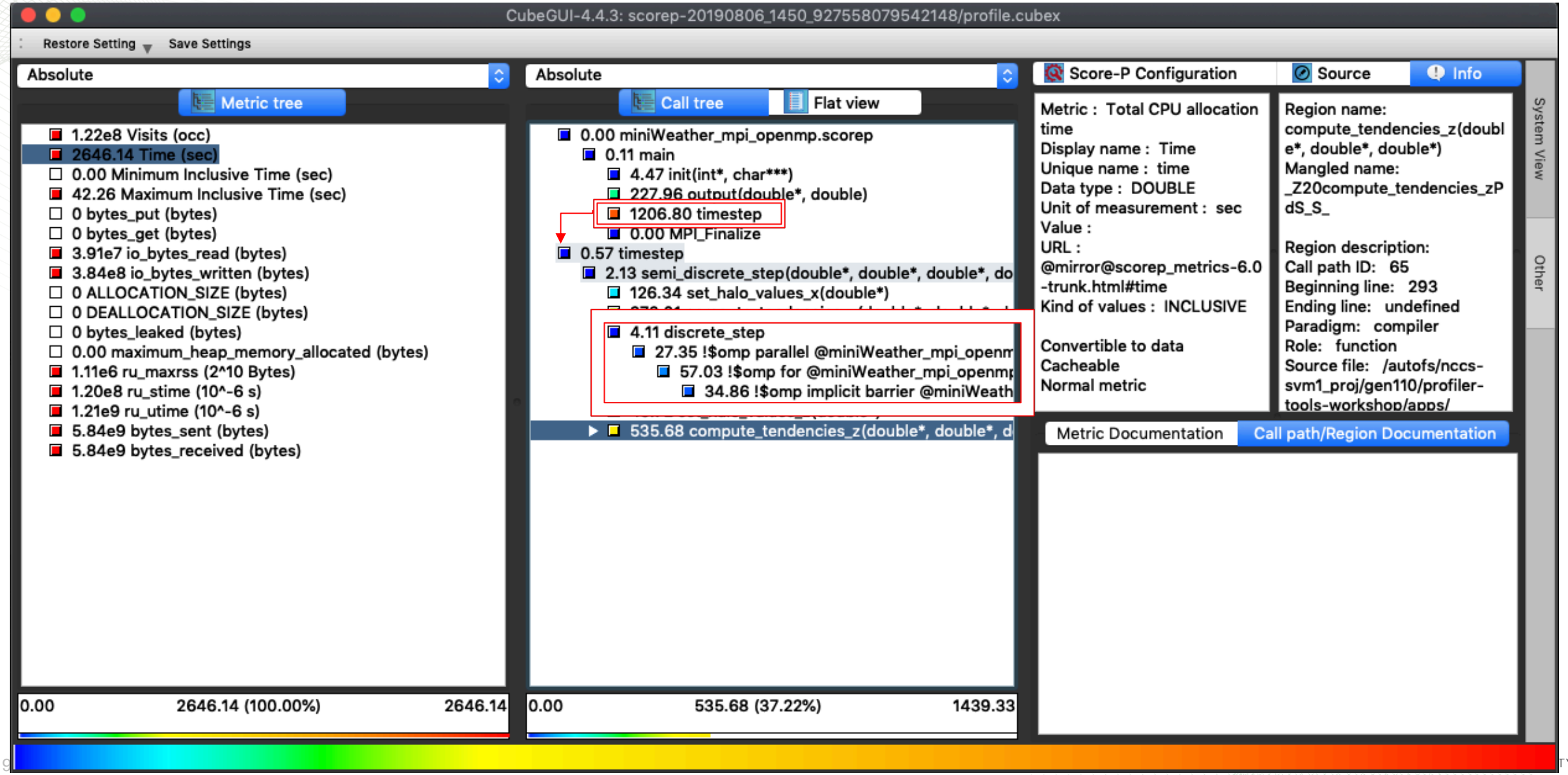
Step 1: Manual Instrumentation – MiniWeather Example

Define “timestep” phase

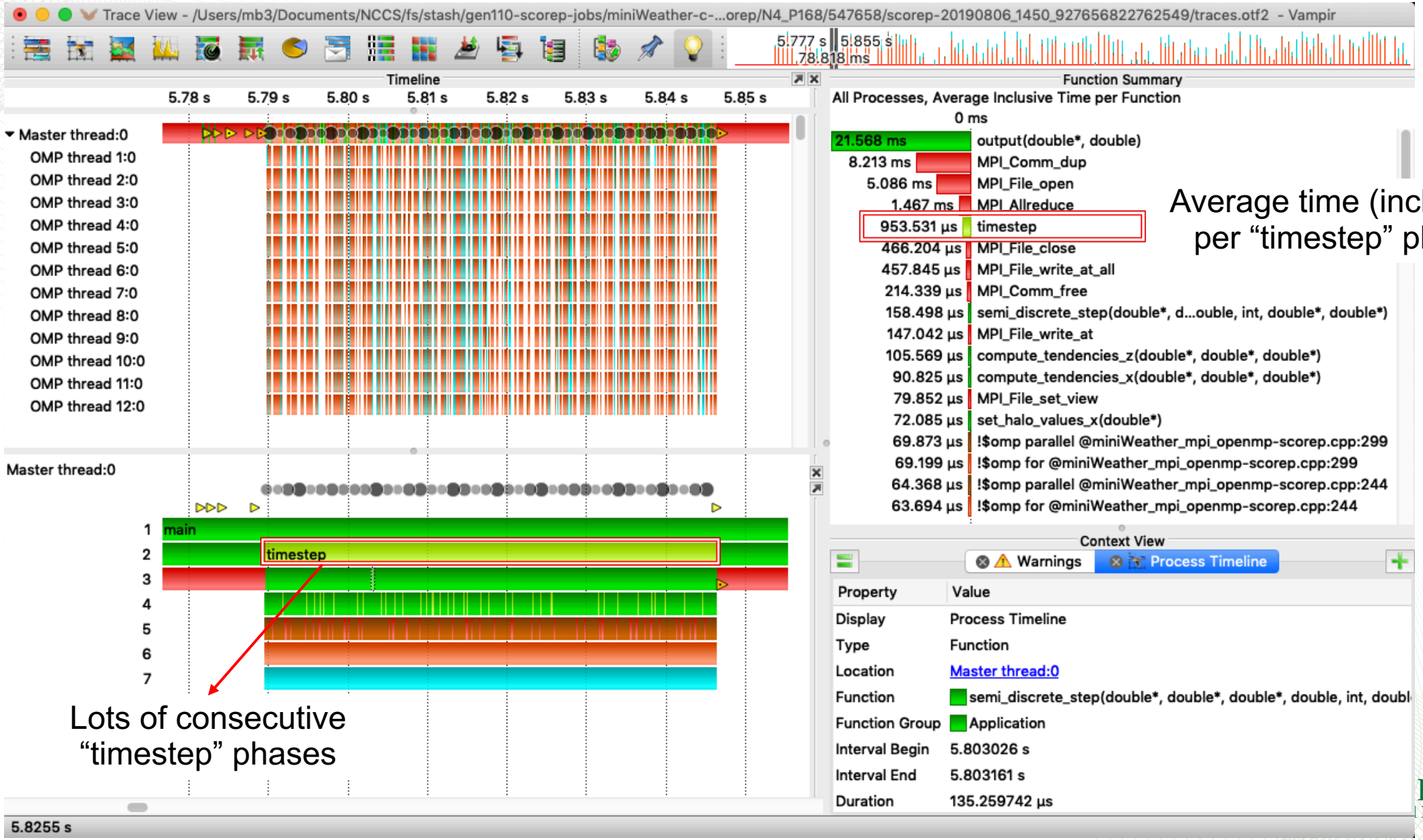
Define “discrete_step” loop

```
> udiff miniWeather_mpi_openmp.cpp miniWeather_mpi_openmp-scorep.cpp
--- miniWeather_mpi_openmp.cpp  2019-07-15 14:00:34.716710000 -0400
+++ miniWeather_mpi_openmp-scorep.cpp   2019-08-06 12:49:59.510846000 -0400
@@ -13,6 +13,8 @@
 #include <mpi.h>
 #include "pnetcdf.h"

+#include <scorep/SCOREP_User.h>
+
 const double pi        = 3.14159265358979323846264338327;   //Pi
 const double grav      = 9.8;                               //Gravitational acceleration (m / s^2)
 const double cp        = 1004.;                             //Specific heat of dry air at constant pressure
@@ -169,6 +171,8 @@
 // q**    = q[n] + dt/2 * rhs(q*  )
 // q[n+1] = q[n] + dt/1 * rhs(q** )
 void perform_timestep( double *state , double *state_tmp , double *flux , double *tend , double dt ) {
+  SCOREP_USER_REGION_DEFINE(timestep_hdl)
+  SCOREP_USER_REGION_BEGIN(timestep_hdl, "timestep",SCOREP_USER_REGION_TYPE_PHASE)
   if (direction_switch) {
     //x-direction first
     semi_discrete_step( state , state     , state_tmp , dt / 3 , DIR_X , flux , tend );
@@ -188,6 +192,7 @@
     semi_discrete_step( state , state_tmp , state_tmp , dt / 2 , DIR_X , flux , tend );
     semi_discrete_step( state , state_tmp , state     , dt / 1 , DIR_X , flux , tend );
   }
+  SCOREP_USER_REGION_END(timestep_hdl)
   if (direction_switch) { direction_switch = 0; } else { direction_switch = 1; }
 }

@@ -210,6 +215,8 @@
   }

   //Apply the tendencies to the fluid state
+  SCOREP_USER_REGION_DEFINE(step_hdl)
+  SCOREP_USER_REGION_BEGIN(step_hdl, "discrete_step", SCOREP_USER_REGION_TYPE_LOOP)
 #pragma omp parallel for private(inds,indt,i) collapse(2)
   for (ll=0; ll<NUM_VARS; ll++) {
     for (k=0; k<nz; k++) {
@@ -220,6 +227,7 @@
       }
     }
   }
+  SCOREP_USER_REGION_END(step_hdl)
 }
```

Step 2: Manual Instrumentation – MiniWeather Profile



Step 3: Manual Instrumentation – MiniWeather Trace



Average time (inclusive) per "timestep" phase

Step 1: Automatic Library Wrapping

- HPC applications often use community/vendor libraries
  - only have headers and precompiled libraries on HPC systems
  - Q: How to profile/trace their usage?

- Score-P makes it easy to wrap any C/C++ library
  - automatically processes library header files
  - requires configuration of Score-P with libclang support

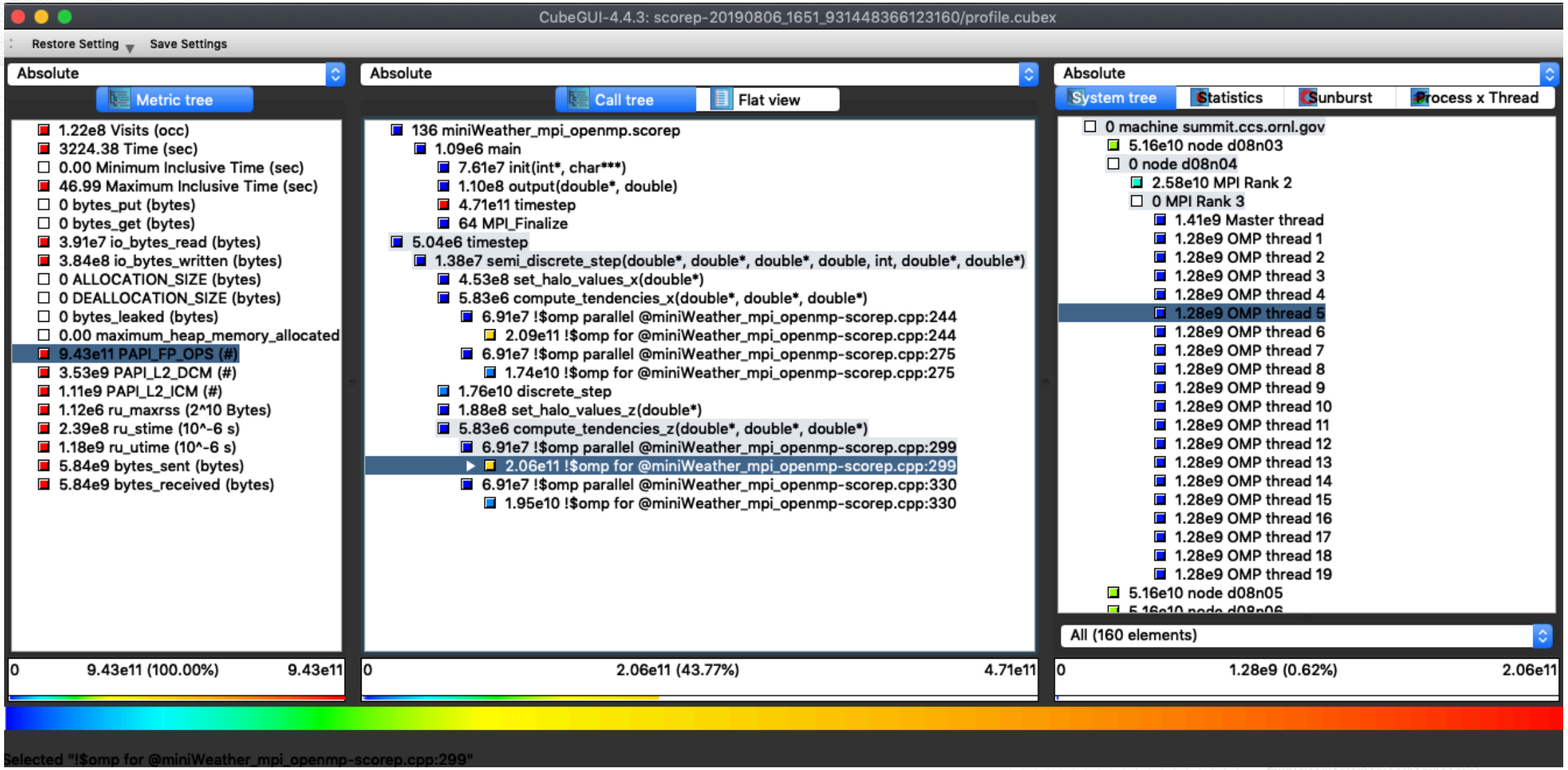
- See User Manual - Appendix I for detailed steps

Using Hardware Counters

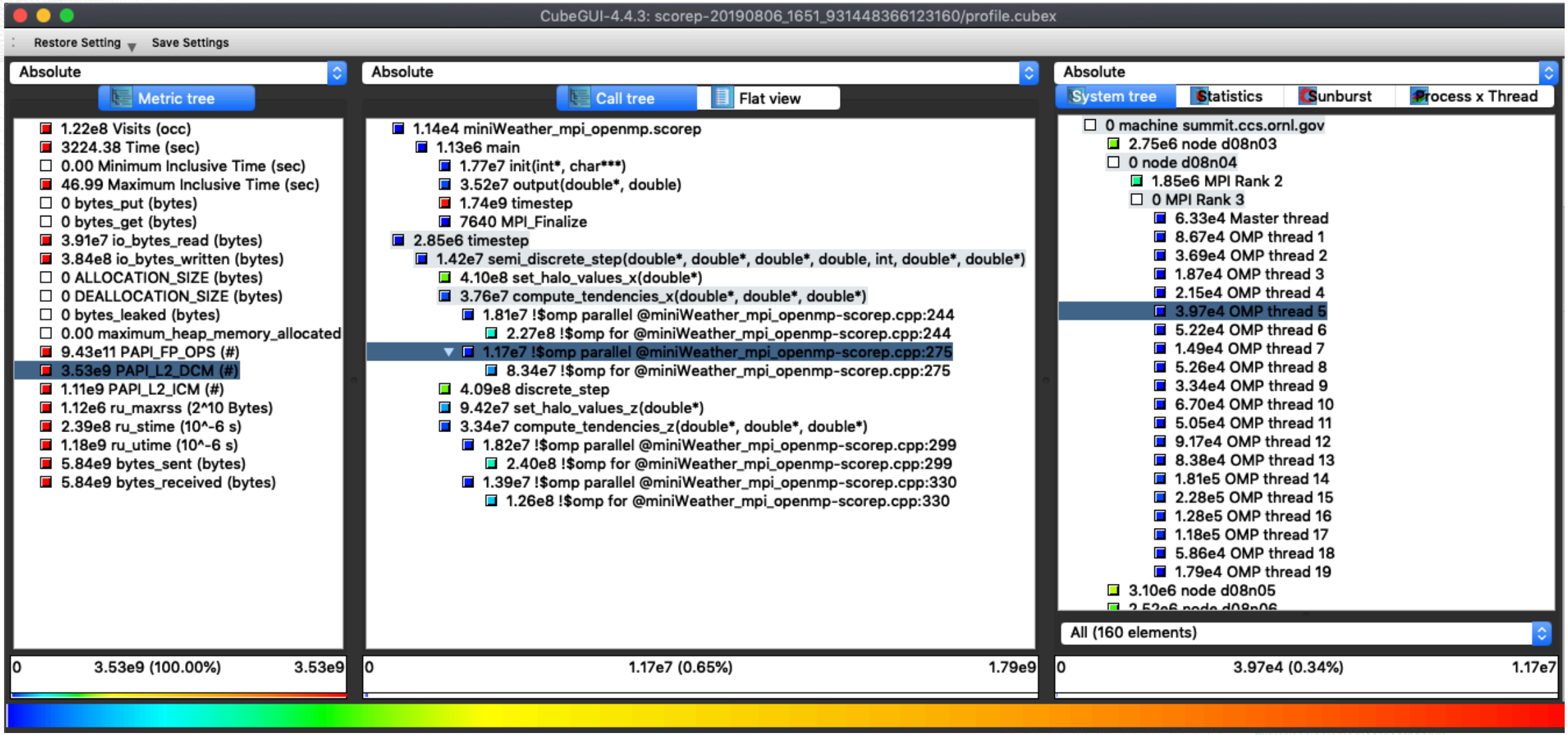
Using CPU Hardware Counters

- `export SCOREP_METRIC_PAPI=<ctr_name>,<ctr_name>,…`
  - works in both profiling and tracing modes
- `papi_avail` and `papi_native_avail` list the counters
  - only those marked “YES” are available on platform
- Used: PAPI_FP_OPS, PAPI_L2_DCM, PAPI_L2_ICM
  - failed: PAPI_DP_OPS (even though reported as available)

Using CPU Hardware Counters – Profile (FP_OPS)



Using CPU Hardware Counters – Profile (L2_DCM)



Using NVIDIA GPU Hardware Counters

- `export SCOREP_CUDA_ENABLE=kernel,kernel_counter,…`
  - enables fixed set of kernel counters

- Other interesting CUDA_ENABLE options

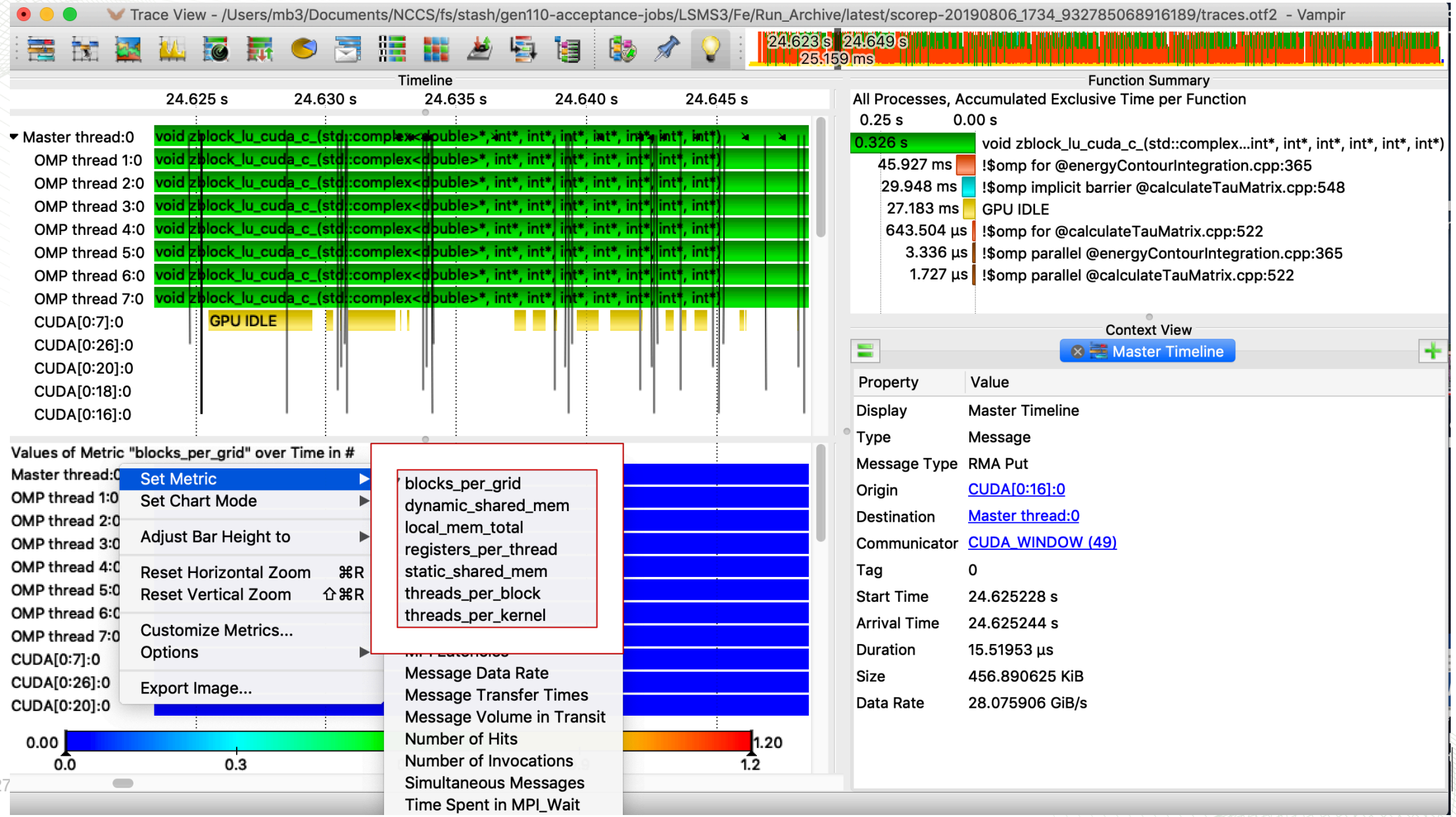
sync Record implicit and explicit CUDA synchronization

idle GPU compute idle time

pure_idle GPU idle time (memory copies are not idle)

gpumemusage Record CUDA memory (de)allocations as a counter

Using NVIDIA GPU Hardware Counters



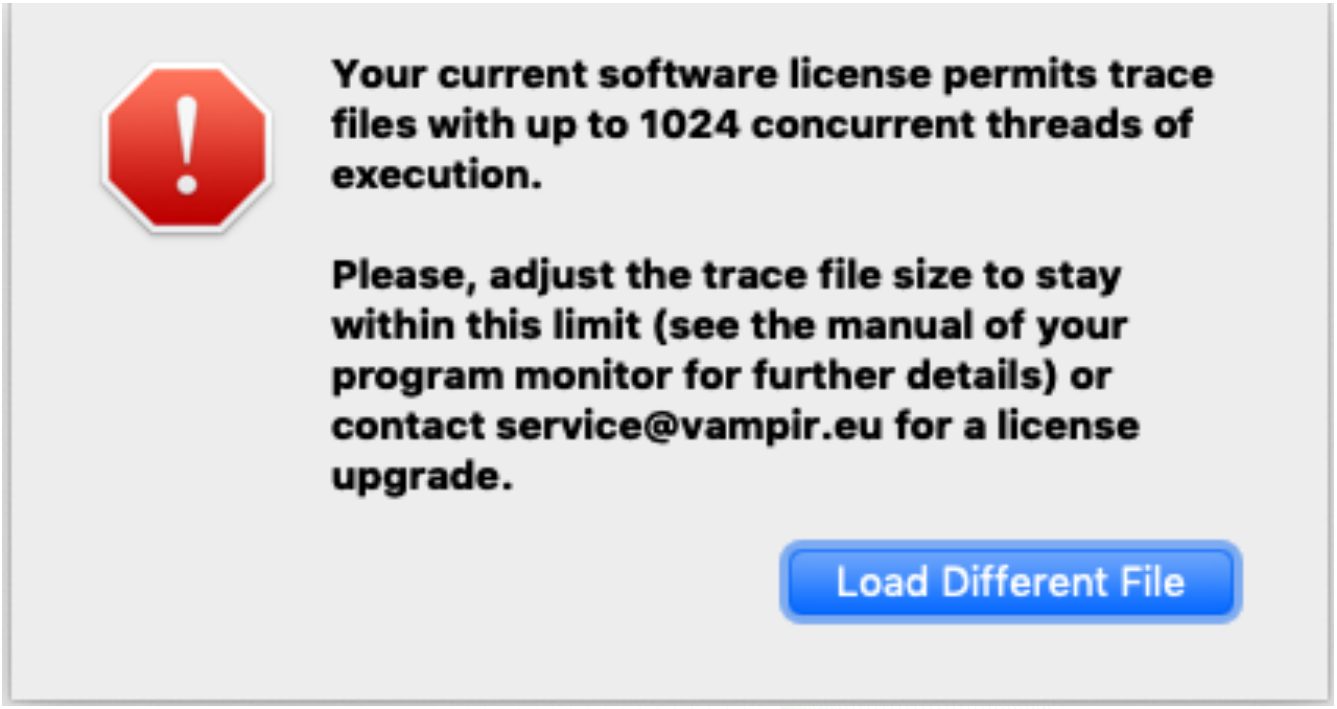
Large Trace Analysis

Analyzing Large Traces – Problem #1

- 128 node LSMS trace
  - ~11,000 processes/threads

- Problem:

Vampir client limits number of processes/threads to 1024



Analyzing Large Traces – VampirServer

- Solution: VampirServer (parallel trace analysis)
- On Summit, usage is documented when loading vampir module

Use Vampir Client + Server

If your local computer is too slow, runs out of memory, or you want to analyze large traces using an HPC system, Vampir's client+server approach is the right way. Loading the module `vampir` on any OLCF system prints instructions on how to use Vampir in client+server mode.

Titan, Eos, Rhea | Summit

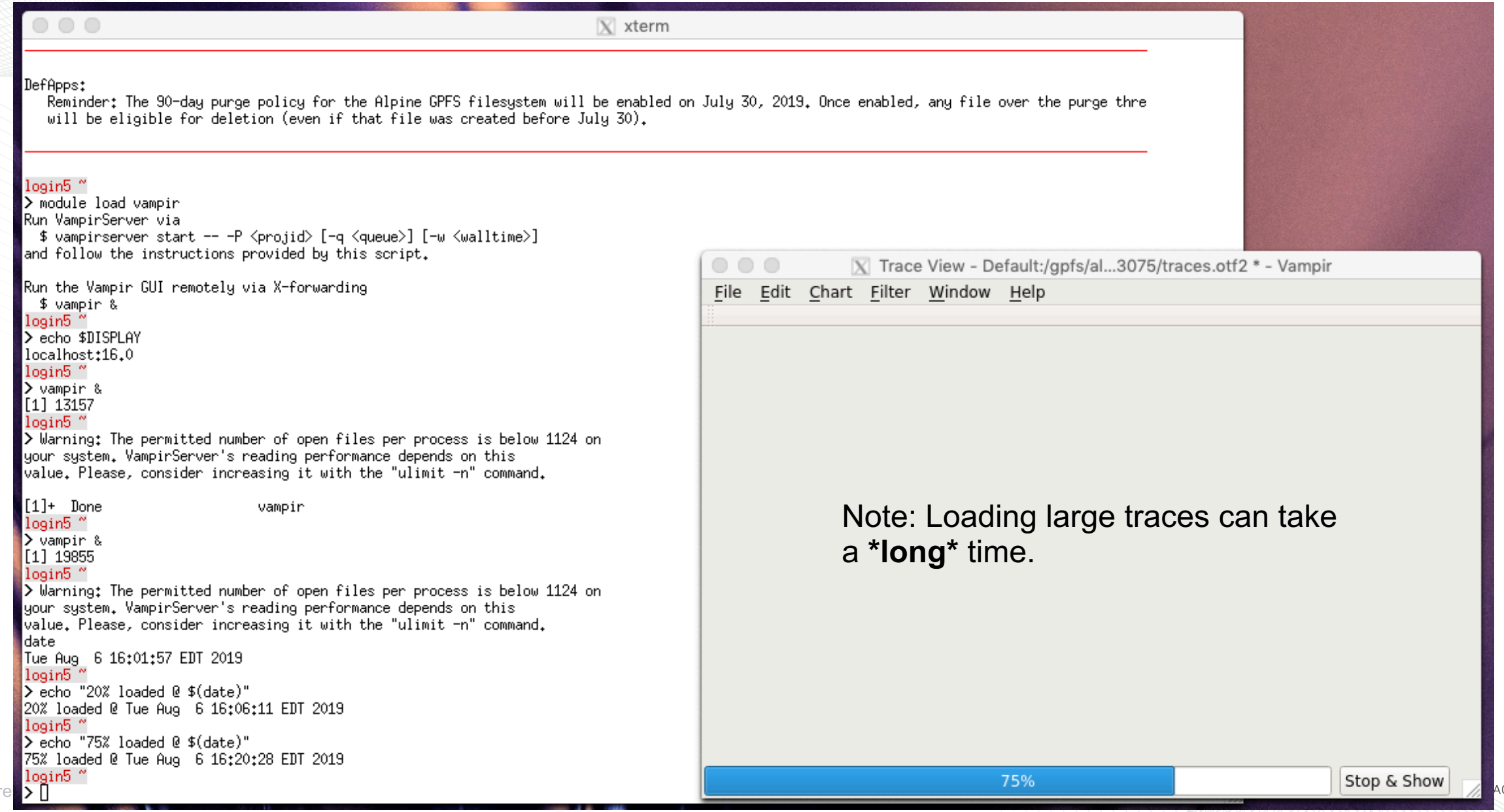
```
$ module load vampir
Run VampirServer via
  $ vampirserver start -- -P  [-q ] [-w ]
and follow the instructions provided by this script.

Run the Vampir GUI remotely via X-forwarding
  $ vampir &

$ vampirserver start -- -P your_project_id
Launching VampirServer...
[..]
User: your_user_name
Password: random_password
VampirServer <4193> listens on: h41n10:30081
```

At the moment, forwarding ports to Summit is not available. Therefore you have to start the Vampir client on the login node via `vampir`, and connect to the server via Open Other -> Remote File, where you put in the above connection details.

Analyzing Large Traces – VampirServer on Summit

```
login3 /gpfs/alpine/gen110/proj-shared/summit-acceptance-apps/summit/LSMS3/Fe_n128/1565104639.4189284/workdir
> modld vampir
Run VampirServer via
  $ vampirserver start -- -P <projid> [-q <queue>] [-w <walltime>]
and follow the instructions provided by this script.

Run the Vampir GUI remotely via X-forwarding
  $ vampir &
login3 /gpfs/alpine/gen110/proj-shared/summit-acceptance-apps/summit/LSMS3/Fe_n128/1565104639.4189284/workdir
> vampirserver start -- -P STF010
Launching VampirServer...
Submitting LSF batch job (this might take a while)...
Warning: more than 1 task/rank assigned to a core
VampirServer 9.5.0  (c263a57)
Licensed to ORNL
Running 4 analysis processes... (abort with vampirserver stop 3906)
User: mjbrim
Password: 8W+V9NoyMWb2
VampirServer <3906> listens on: g18n04:30070
```

Analyzing Large Traces – VampirServer



Note: Loading large traces can take a ***long*** time.

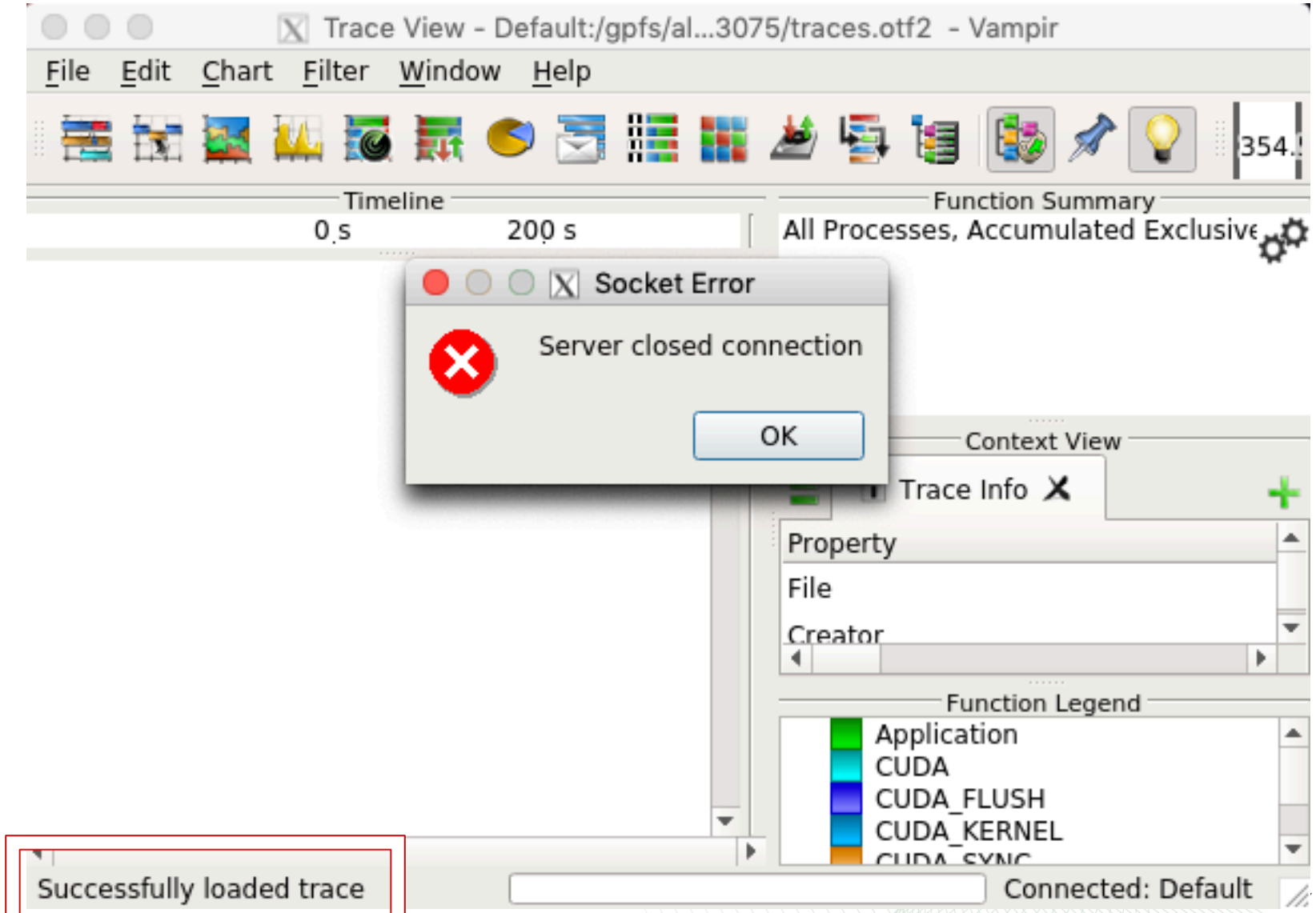
Analyzing Large Traces – VampirServer – Problem #2

- Problem:

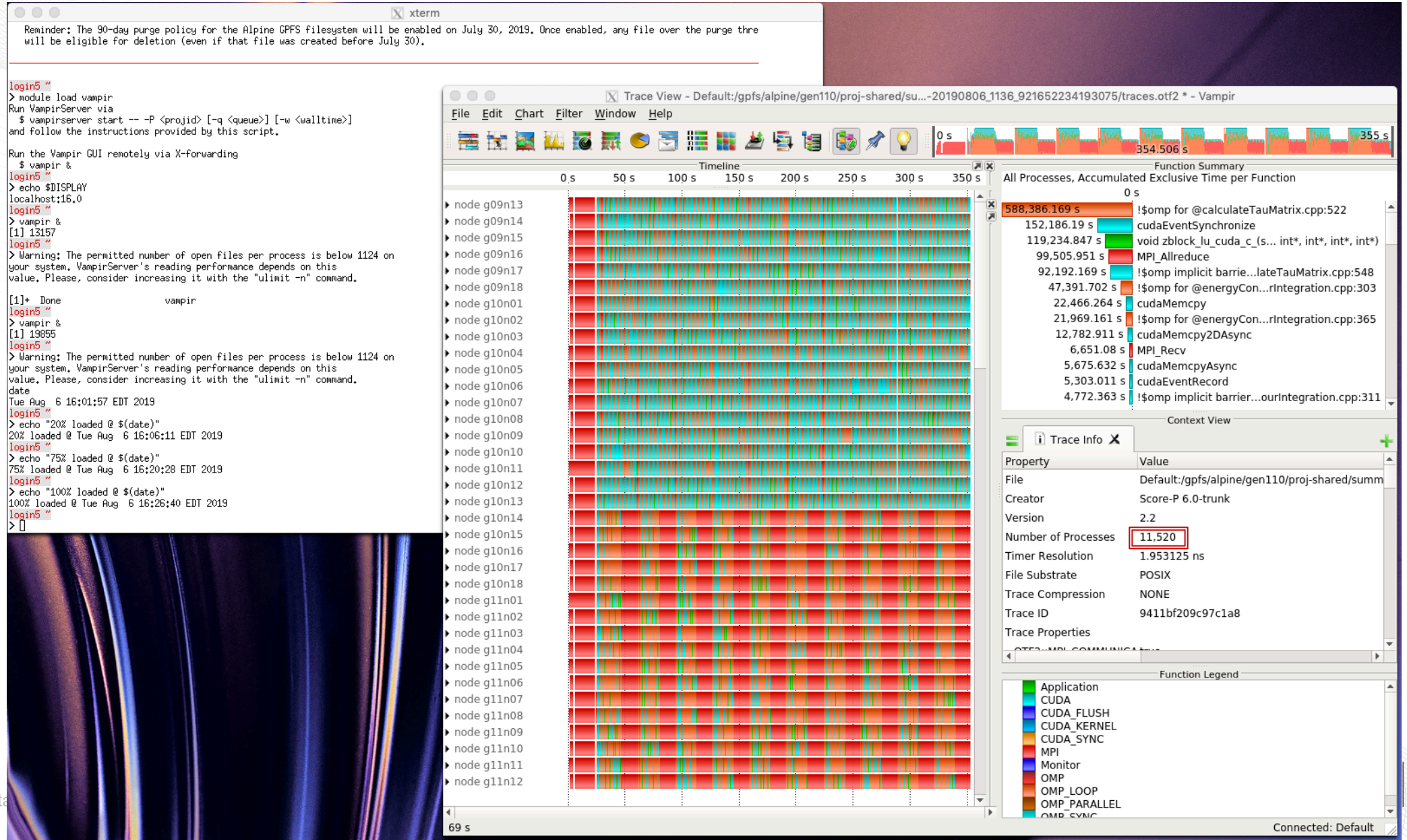
VampirServer job timed-out soon after loading completed (default job length is 30 minutes)

- Solution:

use "`-w <minutes>`" to specify longer walltime



Step 3: Analyzing Large Traces – VampirServer – Success



Questions?