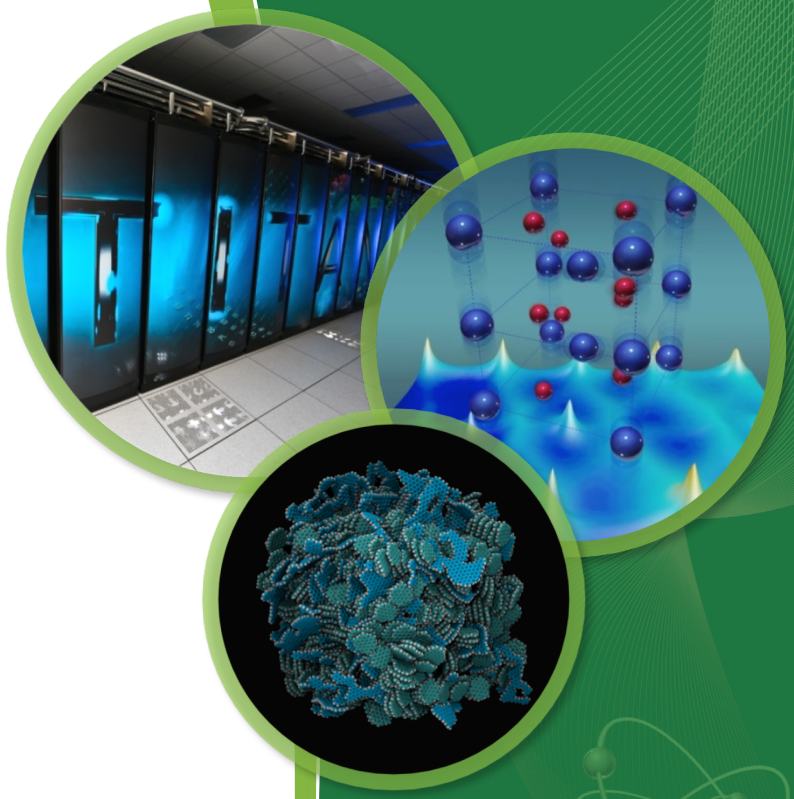
Score-P Introduction

Summit Profiling Tools Workshop

Oak Ridge National Laboratory

August 7, 2019



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 Overview

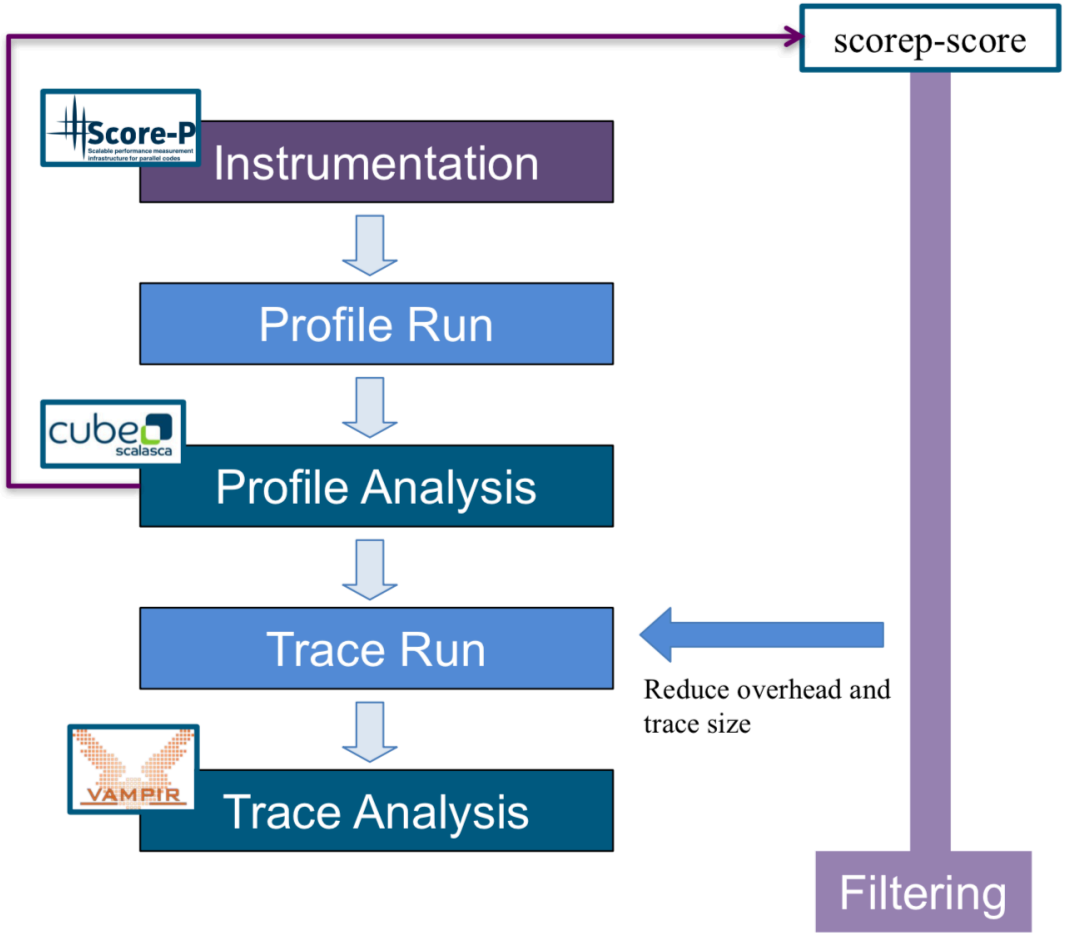
- Goal: support common HPC performance tool requirements
  - profiling, tracing
  - code instrumentation, sampling, online analysis
  - C, C++, Fortran
  - wide coverage of parallel programming models
- Provides data collection for many other HPC tools
  - event/API traces (OTF2): Vampir, Scalasca
  - call-path profiles (CUBE4): Cube, TAU
  - online analysis: Periscope
- Open source - community development started in 2009

Capability Matrix – Score-P

Capability	Profiling	Tracing	Notes/Limitations
MPI, MPI-IO			MPI3 one-sided and MPI-IO added in v6.0
OpenMP CPU			
OpenMP GPU			instrumented code failed to link
OpenACC			
CUDA			Score-P runtime error during profiling
POSIX I/O			Added in v6.0 (functionality not verified)
POSIX threads			
Memory – app-level			getrusage()
Memory – function-level			
Hotspot Detection			Cube profile viewer
Variance Detection			Vampir *Process Summary*
Hardware Counters			PAPI for CPU, fixed set for NVIDIA GPU

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



Step 1a: Automatic Source Code Instrumentation

- Prefix method
  - insert `scorep` before compiler
  - use in simple makefiles and build scripts

```
CC  = scorep <options> gcc
CXX = scorep <options> g++
F90 = scorep <options> gfortran
```

```
CC = gcc
…
target: target.c
        scorep <options> $(CC) -o $@ $^
```

- Wrapper method
  - substitute `scorep-compiler` wrapper scripts for compilers
  - use for autotools or cmake builds

```
> SCOREP_WRAPPER=off cmake ..
    -DCMAKE_C_COMPILER=scorep-gcc
    -DCMAKE_CXX_COMPILER=scorep-g++
```

```
> SCOREP_WRAPPER=off ../configure
    CC=scorep-gcc CXX=scorep-g++
    --disable-dependency-tracking
```

```
> make
SCOREP_WRAPPER_INSTRUMENTER_FLAGS=<options>
```

Step 1a: Automatic Instrumentation Options

- Common programming models enabled by default
  - MPI, SHMEM, OpenMP, POSIX threads
  - can disable using `--no<model>` (e.g., `--noopenmp`)
- Other programming models must be enabled in `<options>`
  - `CUDA      : --cuda`
  - `OpenACC   : --openacc`
  - `POSIX IO : --io=posix`
- See '*Table 3.1: Score-P instrumenter option overview*' in user manual for full list of options and associated configuration variables

Step 1a: Automatic Instrumentation Options

- Compiler-based instrumentation used by default
  - instruments every single function
    - for C++, this can cause extreme overhead

- Alternative: Program Database Toolkit (pdtoolkit)
  - more efficient source instrumentation
    - also more tunable - see pdtoolkit documentation
  - comes from TAU project
  - use option '`--pdt`'

Step 1b: Manual Source Code Region Instrumentation

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 2: Profiling Runs

1. set profiling environment

2. run a small problem size
   - analyze `profile.cubex`
     - CLI: `scorep-score -r`
     - GUI: `cube`

3. run a few larger problem sizes
   - view profiles to observe any scaling trends
   - `scorep-score` will estimate tracing requirements

```
> module load scorep/<version>
> export SCOREP_ENABLE_PROFILING=true
> jsrun <jsrun-opts> my_app <app-opts>
> ls ./scorep-<date>-<time>-<runid>/
MANIFEST.md  profile.cubex  scorep.cfg
> cd scorep-*/
> scorep-score -r profile.cubex > profile.txt
> less profile.txt
> cube profile.cubex # requires X forwarding
```

Step 2: Profi ile

Command Line:

```
> scorep-score profile
Estimated aggregate si
Estimated requirements
Estimated memory requi
(hint: When tracing se                                    es
 or reduce requirements using USR region

flt     type max_buf[B]     visits time[
        ALL 11,192,447 14,768,958 3053.
        MPI 10,976,358 14,012,706 2400.
        COM    216,048    756,168  653.
     SCOREP         41         84    0.
```

Score-P Region Type Key

COM : user functions found on callstack to other regions
CUDA : CUDA API & kernels
MEMORY : memory alloc/dealloc
MPI : All MPI functions
OMP : OpenMP constructs
OPENACC : OpenACC API & kernels
PTHREAD : all pthread functions
SCOREP : Score-P instrumentation
SHMEM : All shmem functions
USR : user functions not found in COM

time[%]	time/visit[us]	region
100.0	206.74	ALL
78.6	171.28	MPI
21.4	863.76	COM
0.0	87.36	SCOREP

Step 2: Profiling Runs – Viewing Region Profile

Command Line: scorep-score –r

```
> scorep-score -r profile.cubex
Estimated aggregate size of event trace:                   892MB
Estimated requirements for largest trace buffer (max_buf): 11MB
Estimated memory requirements (SCOREP_TOTAL_MEMORY):       13MB
(hint: When tracing set SCOREP_TOTAL_MEMORY=13MB 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 11,192,447 14,768,958 3053.31   100.0         206.74  ALL
         MPI 10,976,358 14,012,706 2400.16    78.6         171.28  MPI
         COM    216,048    756,168  653.14    21.4         863.76  COM
      SCOREP         41         84    0.01     0.0          87.36  SCOREP

         MPI  4,698,000  4,536,000    8.08     0.3           1.78  MPI_Irecv
         MPI  4,698,000  4,536,000    9.50     0.3           2.09  MPI_Isend
         MPI  1,296,000  4,536,000   88.63     2.9          19.54  MPI_Waitall
         COM    216,000    756,000  649.84    21.3         859.58  perform_timestep(double*,
double*, double*, double*, double)
         MPI     58,800     50,400   63.76     2.1        1265.17  MPI_File_write_at_all
         MPI     49,500     63,000 1133.58    37.1       17993.25  MPI_Allreduce
         ...
```

Step 2: Profiling Runs – Find Hot Regions (> 10% of time)

Command Line: scorep-score –r | awk

```
> scorep-score -r profile.cubex | awk 'NF >= 7 && $5 >= 10.0 {print $0}'
Estimated aggregate size of event trace:                   892MB
Estimated requirements for largest trace buffer (max_buf): 11MB
(hint: When tracing set SCOREP_TOTAL_MEMORY=13MB 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 11,192,447 14,768,958 3053.31   100.0         206.74  ALL
         MPI 10,976,358 14,012,706 2400.16    78.6         171.28  MPI
         COM    216,048    756,168  653.14    21.4         863.76  COM
         COM    216,000    756,000  649.84    21.3         859.58  perform_timestep(double*,
double*, double*, double*, double)
         MPI     49,500     63,000 1133.58    37.1       17993.25  MPI_Allreduce
         MPI     20,100     25,200  530.14    17.4       21037.15  MPI_File_open
         MPI      3,600     12,600  461.78    15.1       36648.92  MPI_Comm_dup
```

Step 2: Profiling Runs – Filtering

- Filtering reduces overhead of both profiling and tracing
- Simple region/function filtering
  - eliminate functions called many times and having short duration
  - helps manage trace sizes
- Focused region/function filtering
  - exclude all functions, then include only those you want to focus on

```
> cat my_app_filter.flt
SCOREP_REGION_NAMES_BEGIN
  EXCLUDE small
          short
          useless
SCOREP_REGION_NAMES_END


> cat my_app_filter.exclusive.flt
SCOREP_REGION_NAMES_BEGIN
  EXCLUDE *
  INCLUDE bar foo
          baz main
SCOREP_REGION_NAMES_END
```

Step 2: Profiling Runs – Viewing Filtering Effects

- Pass filter file option '`-f`' to `scorep-score`
- Output shows '+' in `flt` column if region will be filtered
- Note: some APIs cannot be excluded (e.g., MPI and OpenACC)

e.g., only COM regions end up filtered

```
> cat all.filter
SCOREP_REGION_NAMES_BEGIN
  EXCLUDE *
SCOREP_REGION_NAMES_END

> scorep-score -f all.filter profile.cubex

Estimated aggregate size of event trace:                   559MB
Estimated requirements for largest trace buffer (max_buf): 50MB
Estimated memory requirements (SCOREP_TOTAL_MEMORY):       52MB
(hint: When tracing set SCOREP_TOTAL_MEMORY=52MB 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 56,251,529 21,259,014  443.14   100.0          20.84  ALL
 -   OPENACC 40,727,960 17,312,520   79.20    17.9           4.57  OPENACC
 -       MPI 11,311,476  2,002,458  333.18    75.2         166.38  MPI
 -       COM  4,212,052  1,944,024   30.76     6.9          15.82  COM
 -    SCOREP         41         12    0.00     0.0          87.02  SCOREP

 *       ALL 52,039,477 19,314,990  412.38    93.1          21.35  ALL-FLT
 -   OPENACC 40,727,960 17,312,520   79.20    17.9           4.57  OPENACC-FLT
 -       MPI 11,311,476  2,002,458  333.18    75.2         166.38  MPI-FLT
 +       FLT  4,212,052  1,944,024   30.76     6.9          15.82  FLT
 -    SCOREP         41         12    0.00     0.0          87.02  SCOREP-FLT
```

Request for all regions to be excluded

Step 3: Tracing Runs

1. set tracing environment

2. run a small problem size
   - analyze `traces.otf2`
     - GUI: `vampir`

3. run a few larger problem sizes
   - view traces to observe any scaling trends

4. targeted filtering or manual region instrumentation

```
> module load scorep/<version>
> export SCOREP_ENABLE_PROFILING=false
> export SCOREP_ENABLE_TRACING=true
> jsrun <jsrun-opts> my_app <app-opts>
> ls ./scorep-<date>-<time>-<runid>/
MANIFEST.md  scorep.cfg  traces/  traces.def  traces.otf2
> cd scorep-*/
> ls -l traces/
(SEE NEXT SLIDE)
```

Step 3: Tracing Runs – Traces Grow with Process Count

2 node trace – 12 processes

```
login5 /gpfs/alpine/gen110/proj-shared/summit/jobs/scorep/miniWeather-c-mpi_openacc/N2_P84/533189/scorep-20190731_0015_635273122182436
> ll
total 546K
-rw-r--r-- 1 mjbrim gen110  953 Jul 31 00:14 MANIFEST.md
-rw-r--r-- 1 mjbrim gen110 1.9K Jul 31 00:14 scorep.cfg
drwxr-sr-x 2 mjbrim gen110 256K Jul 31 00:15 traces/
-rw-r--r-- 1 mjbrim gen110  21K Jul 31 00:15 traces.def
-rw-r--r-- 1 mjbrim gen110  289 Jul 31 00:15 traces.otf2
login5 /gpfs/alpine/gen110/proj-shared/summit/jobs/scorep/miniWeather-c-mpi_openacc/N2_P84/533189/scorep-20190731_0015_635273122182436
> ll traces | more
total 597M
-rw-r--r-- 1 mjbrim gen110  405 Jul 31 00:15 0.def
-rw-r--r-- 1 mjbrim gen110  54M Jul 31 00:15 0.evt
-rw-r--r-- 1 mjbrim gen110 1017 Jul 31 00:15 1.def
-rw-r--r-- 1 mjbrim gen110  50M Jul 31 00:15 1.evt
-rw-r--r-- 1 mjbrim gen110 1.1K Jul 31 00:15 10.def
-rw-r--r-- 1 mjbrim gen110  50M Jul 31 00:15 10.evt
-rw-r--r-- 1 mjbrim gen110 1.1K Jul 31 00:15 11.def
-rw-r--r-- 1 mjbrim gen110  50M Jul 31 00:15 11.evt
-rw-r--r-- 1 mjbrim gen110 1017 Jul 31 00:15 2.def
-rw-r--r-- 1 mjbrim gen110  50M Jul 31 00:15 2.evt
-rw-r--r-- 1 mjbrim gen110 1017 Jul 31 00:15 3.def
-rw-r--r-- 1 mjbrim gen110  50M Jul 31 00:15 3.evt
-rw-r--r-- 1 mjbrim gen110 1017 Jul 31 00:15 4.def
-rw-r--r-- 1 mjbrim gen110  50M Jul 31 00:15 4.evt
-rw-r--r-- 1 mjbrim gen110 1016 Jul 31 00:15 5.def
-rw-r--r-- 1 mjbrim gen110  50M Jul 31 00:15 5.evt
-rw-r--r-- 1 mjbrim gen110 1.1K Jul 31 00:15 6.def
-rw-r--r-- 1 mjbrim gen110  50M Jul 31 00:15 6.evt
-rw-r--r-- 1 mjbrim gen110 1.1K Jul 31 00:15 7.def
-rw-r--r-- 1 mjbrim gen110  50M Jul 31 00:15 7.evt
-rw-r--r-- 1 mjbrim gen110 1.1K Jul 31 00:15 8.def
-rw-r--r-- 1 mjbrim gen110  50M Jul 31 00:15 8.evt
-rw-r--r-- 1 mjbrim gen110 1.1K Jul 31 00:15 9.def
-rw-r--r-- 1 mjbrim gen110  50M Jul 31 00:15 9.evt
```

16 node trace – 96 processes

```
login5 /gpfs/alpine/gen110/proj-shared/summit/jobs/scorep/miniWeather-c-mpi_openacc/N16_P672/533191/scorep-20190731_0017_634355032010606
> ll
total 546K
-rw-r--r-- 1 mjbrim gen110  953 Jul 31 00:16 MANIFEST.md
-rw-r--r-- 1 mjbrim gen110 1.9K Jul 31 00:16 scorep.cfg
drwxr-sr-x 2 mjbrim gen110 256K Jul 31 00:17 traces/
-rw-r--r-- 1 mjbrim gen110  29K Jul 31 00:17 traces.def
-rw-r--r-- 1 mjbrim gen110  289 Jul 31 00:17 traces.otf2
login5 /gpfs/alpine/gen110/proj-shared/summit/jobs/scorep/miniWeather-c-mpi_openacc/N16_P672/533191/scorep-20190731_0017_634355032010606
> ll traces | more
total 4.7G
-rw-r--r-- 1 mjbrim gen110  405 Jul 31 00:17 0.def
-rw-r--r-- 1 mjbrim gen110  54M Jul 31 00:17 0.evt
-rw-r--r-- 1 mjbrim gen110 1.0K Jul 31 00:17 1.def
-rw-r--r-- 1 mjbrim gen110  50M Jul 31 00:17 1.evt
-rw-r--r-- 1 mjbrim gen110 1.1K Jul 31 00:17 10.def
-rw-r--r-- 1 mjbrim gen110  50M Jul 31 00:17 10.evt
-rw-r--r-- 1 mjbrim gen110 1.1K Jul 31 00:17 11.def
-rw-r--r-- 1 mjbrim gen110  50M Jul 31 00:17 11.evt
-rw-r--r-- 1 mjbrim gen110 1.1K Jul 31 00:17 12.def
-rw-r--r-- 1 mjbrim gen110  50M Jul 31 00:17 12.evt
-rw-r--r-- 1 mjbrim gen110 1.1K Jul 31 00:17 13.def
-rw-r--r-- 1 mjbrim gen110  50M Jul 31 00:17 13.evt
-rw-r--r-- 1 mjbrim gen110 1.1K Jul 31 00:17 14.def
-rw-r--r-- 1 mjbrim gen110  50M Jul 31 00:17 14.evt
-rw-r--r-- 1 mjbrim gen110 1.1K Jul 31 00:17 15.def
-rw-r--r-- 1 mjbrim gen110  50M Jul 31 00:17 15.evt
-rw-r--r-- 1 mjbrim gen110 1.1K Jul 31 00:17 16.def
-rw-r--r-- 1 mjbrim gen110  50M Jul 31 00:17 16.evt
```

- Large traces take longer to visualize, and are less portable
- Filtering is the suggested method to avoid large traces

Step 3: Tracing Runs – Viewing OTF2 Traces

- Viewing small traces
  1. use Vampir on your desktop
     - best interactivity
     - requires copying trace data to desktop (or something like sshfs)

Install Vampir locally

Vampir's installers are in directory `/sw/sources/vampir/client` on any of the OLCF systems. Download the installer for your operating system from there. Follow the instructions of the installer. When running Vampir for the first time, you are asked to activate your copy. For this use the license `/sw/sources/vampir/vampir.license` and make sure to use your ORNL email address for the process.

Without an ORNL email address, you can use the license `/sw/sources/vampir/vampir-remote.license`. With this license the Vampir client can only connect to Vampir server instances, for example launched on OLCF systems.

  2. use X forwarding and run Vampir on Summit login node
     - slow unless on ORNL network
       – VNC helps if available

```
desktop> ssh -X user@summit.olcf.ornl.gov

summit> cd /path/to/scorep-*/

summit> module load vampir

summit> cd scorep-*/

summit> vampir &
```

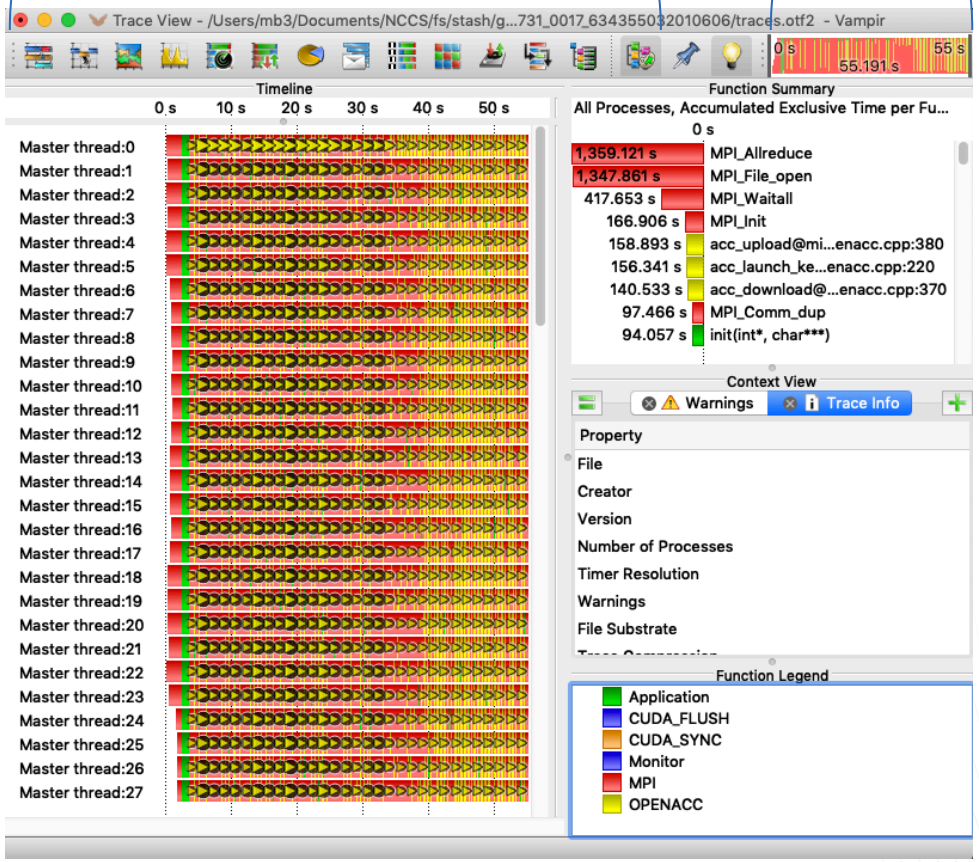
Step 3: Tracing Runs – Navigating Vampir

Use to add new content frames to the display window.

Toolbar

Trace Timeline

Use to select timeframe of focus. Click-drag with mouse, or right-click to enter specific start and end times.



Master Timeline

View behavior over time for processes/threads/streams. Can also select focus timeframe. Right-click 'Group Processes' menu allows for various forms of grouping.

Function Summary

Summarizes time spent per function/region across all processes for selected timeframe. Right-click to use pie chart format, or to use inclusive time as metric.

Context View

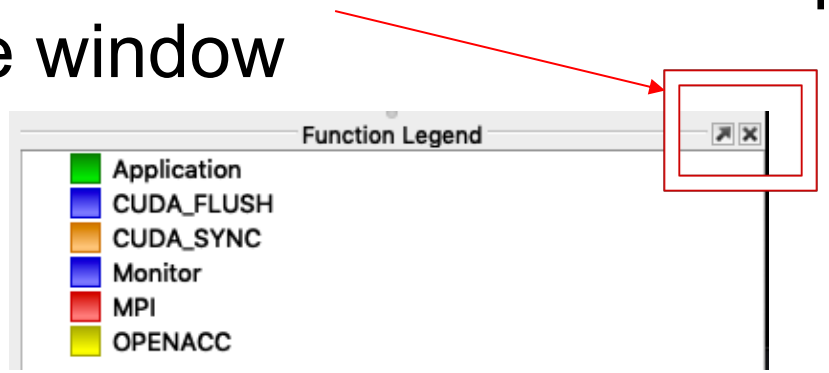
Provides context for item selected in Master/Process Timeline. Properties change according to item type.

Color Legend

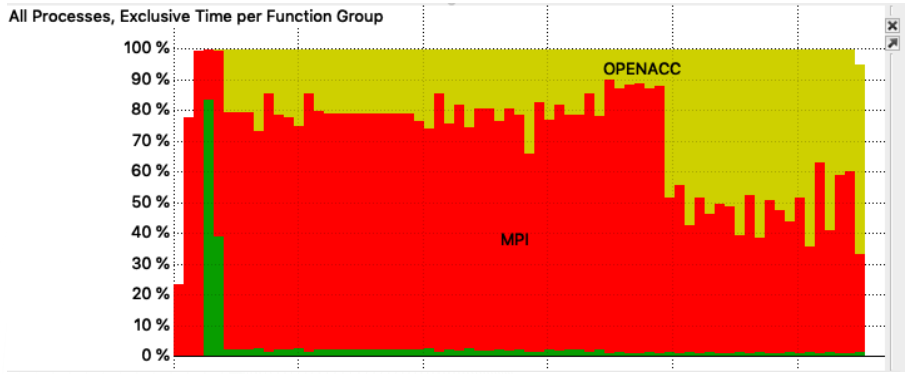
Shows colors associated with various region types. Right-click to modify the color assignments.

Step 3: Tracing Runs – Customizing Vampir

- Frames shown can be easily customized
- Add new frames using toolbar buttons
  - Hint: hover over buttons for text description
- Use frame controls to remove or place in a separate window

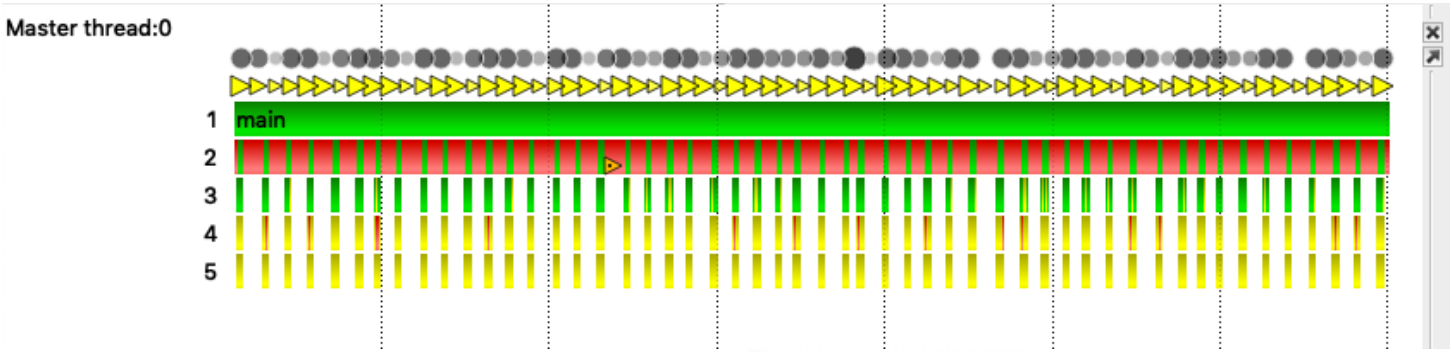


Summary Timeline



Step 3: Tracing Runs – Navigating Vampir

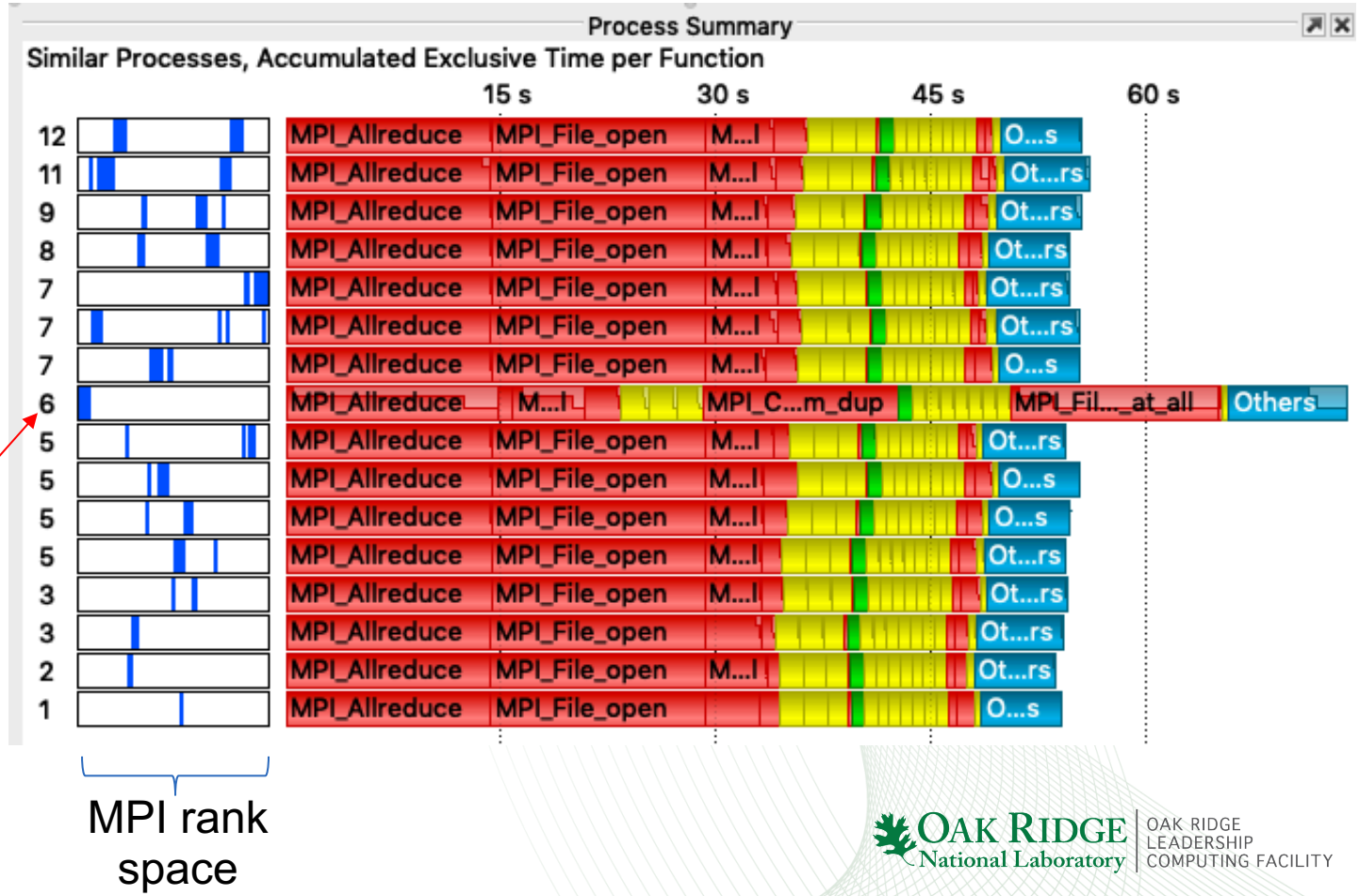
- Viewing a Single Process
  - Toolbar `Add Process Timeline`
- Shows callstack timeline for a single process
  - inclusive compute time
    - for exclusive time: right-click ‘Set Mode → Exclusive’
  - choose any process
    - right-click ‘Set Process’
- Other information shown
  - message bursts: gray circles
  - I/O operations: triangles



Step 3: Tracing Runs – Navigating Vampir

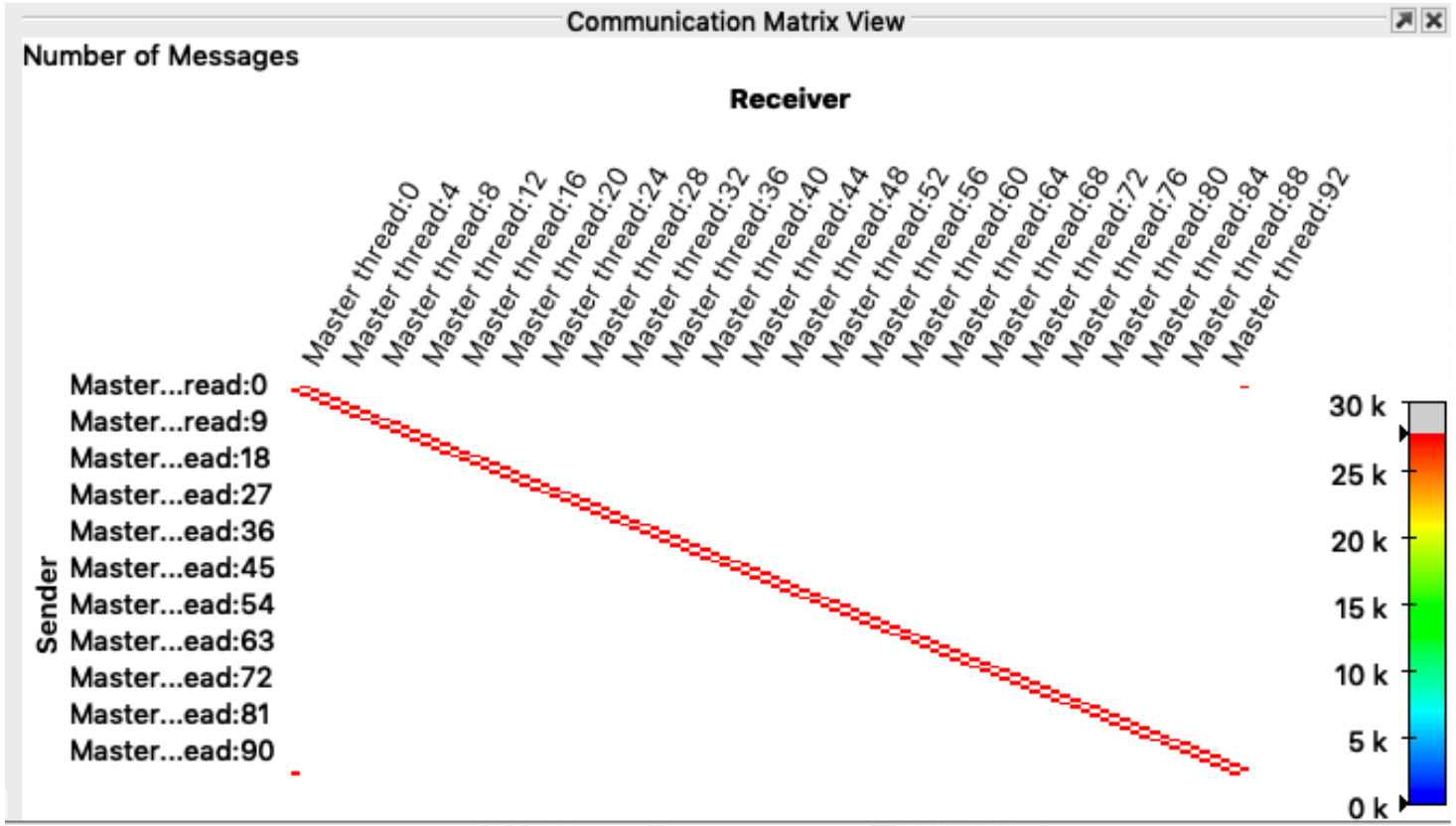
- Finding Process Variance
  - Toolbar `Add Process Summary`
- Groups similarly-behaving processes
  - quick way to identify outliers or subgroups

what's going on here? first six ranks take much longer



Step 3: Tracing Runs – Navigating Vampir

- Viewing Communication
  - Toolbar `Add Communication Matrix View`
- Shows point-to-point messages between MPI ranks
  - heat chart color denotes total bytes sent or received



Case Study: MiniWeather

- “A parallel programming training mini app simulating weather-like flows, focusing on MPI, OpenMP, and OpenACC”
  - https://github.com/mrnorman/miniWeather

- Breadth of functionality makes it a good test case for exploring what works and what does not

Case Study: MiniWeather – Code Instrumentation

```
> diff ../../c/Makefile Makefile
21c21
<       ${CC} ${INCLUDE} ${CFLAGS} -o miniWeather_mpi miniWeather_mpi.cpp ${LDFLAGS}
---
>       scorep ${CC} ${INCLUDE} ${CFLAGS} -o miniWeather_mpi miniWeather_mpi.cpp ${LDFLAGS}
24c24
<       ${CC} ${INCLUDE} ${CFLAGS} ${OMPFLAGS} -o miniWeather_mpi_openmp miniWeather_mpi_openmp.cpp ${LDFLAGS}
---
>       scorep ${CC} ${INCLUDE} ${CFLAGS} ${OMPFLAGS} -o miniWeather_mpi_openmp miniWeather_mpi_openmp.cpp ${LDFLAGS}
27c27
<       ${CC} ${INCLUDE} ${CFLAGS} ${ACCFLAGS} -o miniWeather_mpi_openacc miniWeather_mpi_openacc.cpp ${LDFLAGS}
---
>       scorep --openacc --cuda ${CC} ${INCLUDE} ${CFLAGS} ${ACCFLAGS}
```

```
> diff ../../fortran/Makefile Makefile
18c18
<       ${FC} ${INCLUDE} ${FFLAGS} -o miniWeather_serial miniWeather_seria
---
>       scorep ${FC} ${INCLUDE} ${FFLAGS} -o miniWeather_serial miniWeathe
21c21
<       ${FC} ${INCLUDE} ${FFLAGS} -o miniWeather_mpi miniWeather_mpi.F90 ${LDFLAGS}
---
>       scorep ${FC} ${INCLUDE} ${FFLAGS} -o miniWeather_mpi miniWeather_mpi.F90 ${LDFLAGS}
24c24
<       ${FC} ${INCLUDE} ${FFLAGS} ${OMPFLAGS} -o miniWeather_mpi_openmp miniWeather_mpi_openmp.F90 ${LDFLAGS}
---
>       scorep --pdt ${FC} ${INCLUDE} ${FFLAGS} ${OMPFLAGS} -o miniWeather_mpi_openmp miniWeather_mpi_openmp.F90 ${LDFLAGS}
27c27
<       ${FC} ${INCLUDE} ${FFLAGS} ${ACCFLAGS} -o miniWeather_mpi_openacc miniWeather_mpi_openacc.F90 ${LDFLAGS}
---
>       scorep --openacc --cuda ${FC} ${INCLUDE} ${FFLAGS} ${ACCFLAGS} -o miniWeather_mpi_openacc miniWeather_mpi_openacc.F90 ${LDFLAGS}
```

```
[Score-P] src/measurement/thread/fork_join/scorep_thread_fork_join_omp.c:402: Fatal: Bug
'TPD == 0': Invalid OpenMP thread specific data object. Please ensure that all omp parallel
regions are instrumented. [Score-P] Please report this to support@score-p.org. Thank you.
[Score-P] Try also to preserve any generated core dumps.
[e25n13:149147] * Process received signal *
[e25n13:149147] Signal: Aborted (6)
[e25n13:149147] Signal code: (-6)
```

Problem #1:
Fortran with OpenMP

Compiler instrumentation led to runtime failure

Solution: use `--pdt`

Case Study: MiniWeather – MPI+OpenMP Profile

MPI+OpenMP (2 nodes)

```
> scorep-score profile.cubex
Estimated aggregate size of event trace:                   2710MB
Estimated requirements for largest trace buffer (max_buf): 732MB
Estimated memory requirements (SCOREP_TOTAL_MEMORY):       772MB
(hint: When tracing set SCOREP_TOTAL_MEMORY=772MB 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 766,552,809 61,293,318  851.53   100.0          13.89  ALL
         OMP 751,025,340 59,976,720  731.87    85.9          12.20  OMP
         MPI  11,311,476    667,986  110.81    13.0         165.88  MPI
         COM   4,215,952    648,608    8.86     1.0          13.66  COM
     SCOREP          41          4    0.00     0.0          77.53  SCOREP


> scorep-score -r profile.cubex | fgrep -v ALL | awk '$5 > 10.0 {print $0}'
Estimated aggregate size of event trace:                   2710MB
Estimated requirements for largest trace buffer (max_buf): 732MB
Estimated memory requirements (SCOREP_TOTAL_MEMORY):       772MB
(hint: When tracing set SCOREP_TOTAL_MEMORY=772MB to avoid intermediate flushes
 or reduce requirements using USR regions filters.)
flt     type  max_buf[B]     visits time[s] time[%] time/visit[us]  region
         OMP 751,025,340 59,976,720  731.87    85.9          12.20  OMP
         MPI  11,311,476    667,986  110.81    13.0         165.88  MPI
         OMP  14,040,000  2,160,000  260.56    30.6         120.63  !$omp for @miniWeather_mpi_openmp.cpp:291
         OMP  14,040,000  2,160,000  257.48    30.2         119.20  !$omp for @miniWeather_mpi_openmp.cpp:236
```

MPI+OpenMP (32 nodes)

```
> scorep-score -r profile.cubex
Estimated aggregate size of event trace:                   42GB
Estimated requirements for largest trace buffer (max_buf): 732MB
Estimated memory requirements (SCOREP_TOTAL_MEMORY):       772MB
(hint: When tracing set SCOREP_TOTAL_MEMORY=772MB 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 766,552,809 956,381,838 4319.36   100.0          4.52  ALL
         OMP 751,025,340 935,327,520 2134.05    49.4          2.28  OMP
         MPI  11,311,476  10,676,526 2060.28    47.7        192.97  MPI
         COM   4,215,952  10,377,728  125.03     2.9         12.05  COM
     SCOREP          41          64    0.00     0.0         77.72  SCOREP

         OMP  93,960,000  69,120,000   45.83     1.1          0.66  !$omp parallel @miniWeather_mpi_openmp.cpp:213
         OMP  46,980,000  34,560,000   23.49     0.5          0.68  !$omp parallel @miniWeather_mpi_openmp.cpp:291
         OMP  46,980,000  34,560,000   22.72     0.5          0.66  !$omp parallel @miniWeather_mpi_openmp.cpp:322
         OMP  46,980,000  34,560,000   23.67     0.5          0.68  !$omp parallel @miniWeather_mpi_openmp.cpp:236
         OMP  46,980,000  34,560,000   22.76     0.5          0.66  !$omp parallel @miniWeather_mpi_openmp.cpp:267
         OMP  46,980,000  34,560,000   23.06     0.5          0.67  !$omp parallel @miniWeather_mpi_openmp.cpp:369
         OMP  46,980,000  34,560,000   22.71     0.5          0.66  !$omp parallel @miniWeather_mpi_openmp.cpp:408
         OMP  46,980,000  34,560,000   23.77     0.6          0.69  !$omp parallel @miniWeather_mpi_openmp.cpp:351
         OMP  46,980,000     540,000    0.36     0.0          0.66  !$omp parallel @miniWeather_mpi_openmp.cpp:384
         OMP  28,080,000  69,120,000  105.22     2.4          1.52  !$omp for @miniWeather_mpi_openmp.cpp:213
         OMP  28,080,000  69,120,000  170.13     3.9          2.46  !$omp implicit barrier
@miniWeather_mpi_openmp.cpp:222
         OMP  14,040,000  34,560,000   36.86     0.9          1.07  !$omp for @miniWeather_mpi_openmp.cpp:369
         OMP  14,040,000  34,560,000   24.05     0.6          0.70  !$omp for @miniWeather_mpi_openmp.cpp:408
         OMP  14,040,000  34,560,000   35.07     0.8          1.01  !$omp for @miniWeather_mpi_openmp.cpp:351
         OMP  14,040,000  34,560,000  194.79     4.5          5.64  !$omp implicit barrier
@miniWeather_mpi_openmp.cpp:336
         OMP  14,040,000  34,560,000  118.55     2.7          3.43  !$omp for @miniWeather_mpi_openmp.cpp:322
```

Case Study: MiniWeather – MPI+OpenACC Profile

MPI+OpenACC (2 nodes)

```
> scorep-score profile.cubex
Estimated aggregate size of event trace:                   607MB
Estimated requirements for largest trace buffer (max_buf): 54MB
Estimated memory requirements (SCOREP_TOTAL_MEMORY):       56MB
(hint: When tracing set SCOREP_TOTAL_MEMORY=56MB 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 56,251,529 21,259,014  443.14   100.0          20.84  ALL
    OPENACC 40,727,960 17,312,520   79.20    17.9           4.57  OPENACC
         MPI 11,311,476  2,002,458  333.18    75.2         166.38  MPI
         COM  4,212,052  1,944,024   30.76     6.9          15.82  COM
     SCOREP         41         12    0.00     0.0          87.02  SCOREP

> scorep-score -r profile.cubex | fgrep -v ALL | awk '$5 > 10.0 {print $0}'
Estimated aggregate size of event trace:                   607MB
Estimated requirements for largest trace buffer (max_buf): 54MB
Estimated memory requirements (SCOREP_TOTAL_MEMORY):       56MB
(hint: When tracing set SCOREP_TOTAL_MEMORY=56MB to avoid intermediate flushes
 or reduce requirements using USR regions filters.)
flt     type max_buf[B]     visits time[s] time[%] time/visit[us]  region
    OPENACC 40,727,960 17,312,520   79.20    17.9           4.57  OPENACC
         MPI 11,311,476  2,002,458  333.18    75.2         166.38  MPI
         MPI     51,000      9,000  122.72    27.7       13635.07  MPI_Allreduce
         MPI     20,700      3,600   88.93    20.1       24703.26  MPI_File_open
         MPI      3,900      1,800   81.15    18.3       45084.03  MPI_Comm_dup
```

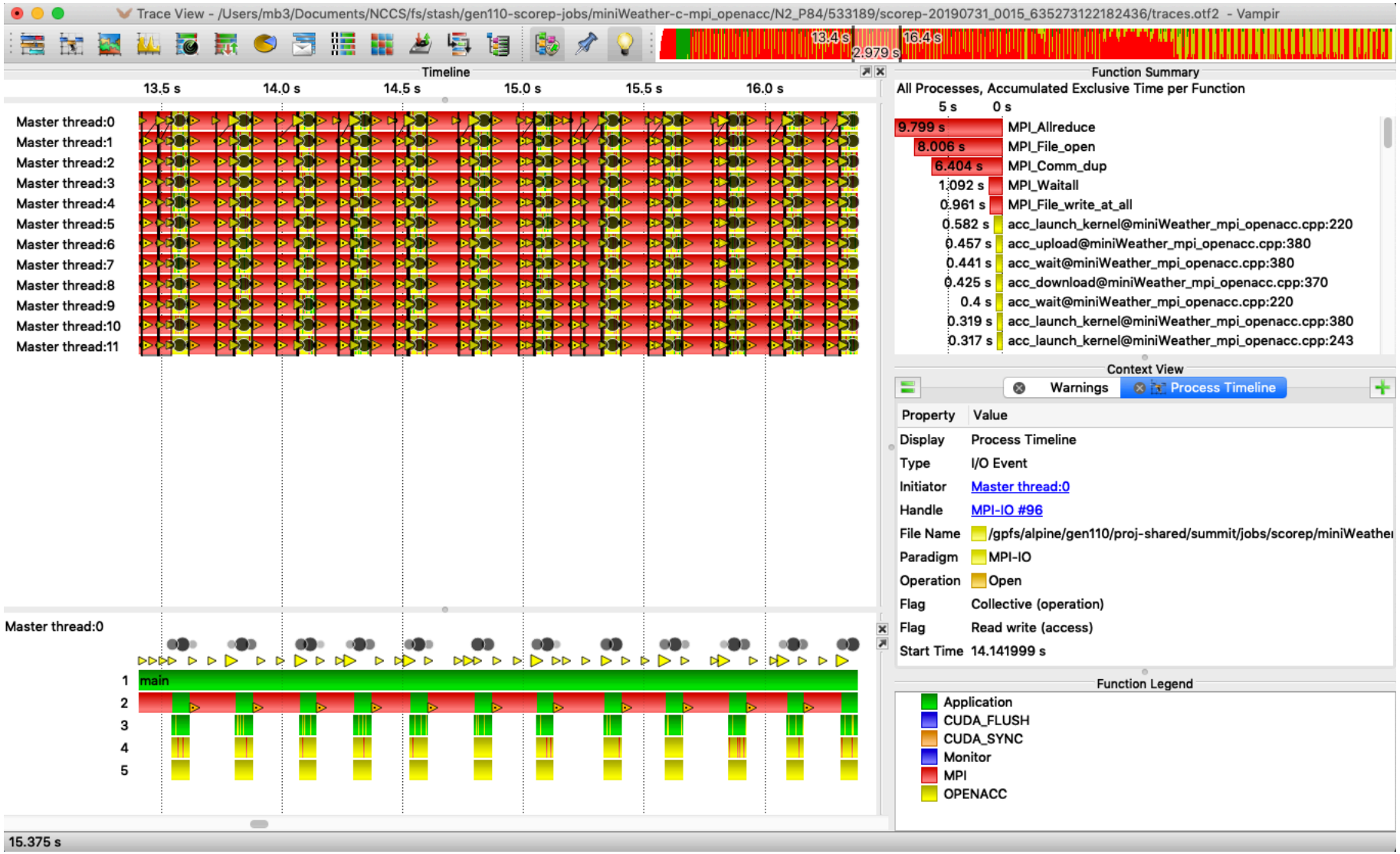
MPI+OpenACC (32 nodes)

```
> scorep-score -r profile.cubex
Estimated aggregate size of event trace:                   10GB
Estimated requirements for largest trace buffer (max_buf): 54MB
Estimated memory requirements (SCOREP_TOTAL_MEMORY):       56MB
(hint: When tracing set SCOREP_TOTAL_MEMORY=56MB 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 56,251,529 338,107,974 8776.19   100.0          25.96  ALL
    OPENACC 40,727,960 274,975,320 1250.30    14.2           4.55  OPENACC
         MPI 11,311,476  32,028,078 7063.12    80.5         220.53  MPI
         COM  4,212,052  31,104,384  462.75     5.3          14.88  COM
     SCOREP         41         192    0.02     0.0          89.76  SCOREP

         MPI  4,806,000  10,368,000   20.95     0.2           2.02  MPI_Irecv
         MPI  4,806,000  10,368,000   28.45     0.3           2.74  MPI_Isend
         MPI  1,404,000  10,368,000  194.51     2.2          18.76  MPI_Waitall
         COM  1,404,000  10,368,000   29.48     0.3           2.84  semi_discrete_step(double*, double*, double*,
double, int, double*, double*)
    OPENACC  1,404,000  10,368,000   53.82     0.6           5.19  acc_download@miniWeather_mpi_openacc.cpp:370
    OPENACC  1,404,000  10,368,000   64.17     0.7           6.19  acc_upload@miniWeather_mpi_openacc.cpp:380
    OPENACC  1,404,000  10,368,000   82.34     0.9           7.94  acc_wait@miniWeather_mpi_openacc.cpp:380
    OPENACC  1,404,000  10,368,000   20.92     0.2           2.02  acc_data_enter@miniWeather_mpi_openacc.cpp:220
    OPENACC  1,404,000  10,368,000   22.53     0.3           2.17  acc_compute@miniWeather_mpi_openacc.cpp:220
    OPENACC  1,404,000  10,368,000   59.78     0.7           5.77  acc_launch_kernel@miniWeather_mpi_openacc.cpp:220
    OPENACC  1,404,000  10,368,000   81.63     0.9           7.87  acc_wait@miniWeather_mpi_openacc.cpp:220
    OPENACC  1,404,000  10,368,000   22.81     0.3           2.20  acc_data_exit@miniWeather_mpi_openacc.cpp:220
    OPENACC    702,000   5,184,000   32.72     0.4           6.31  acc_launch_kernel@miniWeather_mpi_openacc.cpp:243
    OPENACC    702,000   5,184,000   11.06     0.1           2.13  acc_compute@miniWeather_mpi_openacc.cpp:329
    OPENACC    702,000   5,184,000    9.91     0.1           1.91  acc_data_enter@miniWeather_mpi_openacc.cpp:329
    OPENACC    702,000   5,184,000   15.22     0.2           2.94  acc_data_exit@miniWeather_mpi_openacc.cpp:298
    OPENACC    702,000   5,184,000   67.64     0.8          13.05  acc_wait@miniWeather_mpi_openacc.cpp:298
```

Case Study: MiniWeather – MPI+OpenACC Trace (N=2)



Case Study: MiniWeather – MPI+OpenACC Trace (N=16)