Debugging Tools on Perlmutter

NERSC New User Training
Sept 08, 2023

Justin Cook
User Engagement Group

Outline

- Debug programs graphically with DDT and TotalView - full-fledged debuggers
- CUDA-gdb and Compute Sanitizer (aka CUDA-memcheck) are non-MPI CUDA debuggers provided by NVIDIA
- Can debug parallel programs with gdb4hpc, another text-based GDB-like tool
- Find memory-related bugs with valgrind4hpc and sanitizers4hpc
- Debug crashed or deadlocked programs with STAT and ATP
- https://docs.nersc.gov/tools/debug/

Before we start debugging

- Setup a remote desktop connection
- Compile your program
- Setup your environment
- Allocate compute resources

Setup a remote desktop connection

- NoMachine (https://docs.nersc.gov/connect/nx/)
- Better performance than traditional x11 forwarding over ssh
- DDT and TotalView have their own remote connections that can also be used

Compile your program

- Generate debugging data and disable compiler optimizations
- C
  - cc -g -O0 -o program program.c
- Fortran
  - ftn -g -O0 -o program program.f90
- CUDA
  - nvcc -g -O0 -G -o program program.cu

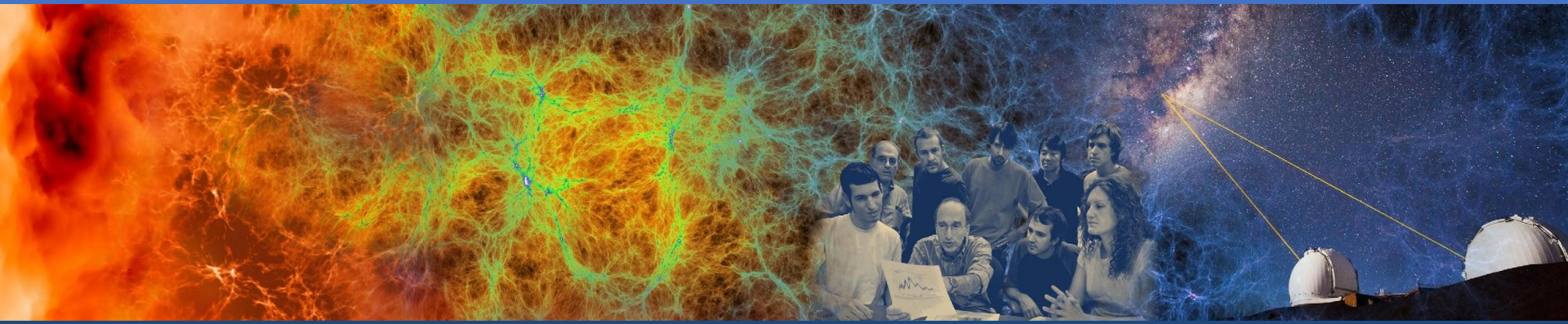
Setup your environment

- Allow creation of core files
  - ulimit -c unlimited
- Abort and create core file on error
  - export MPICH_ABORT_ON_ERROR=1
  - export CUDA_ENABLE_COREDUMP_ON_EXCEPTION=1
- Use cray-cti module for HPE / Cray tools
  - module load cray-cti
  - export CTI_WLM_IMPL=slurm
  - https://github.com/common-tools-interface/cti

Allocating nodes for debugging

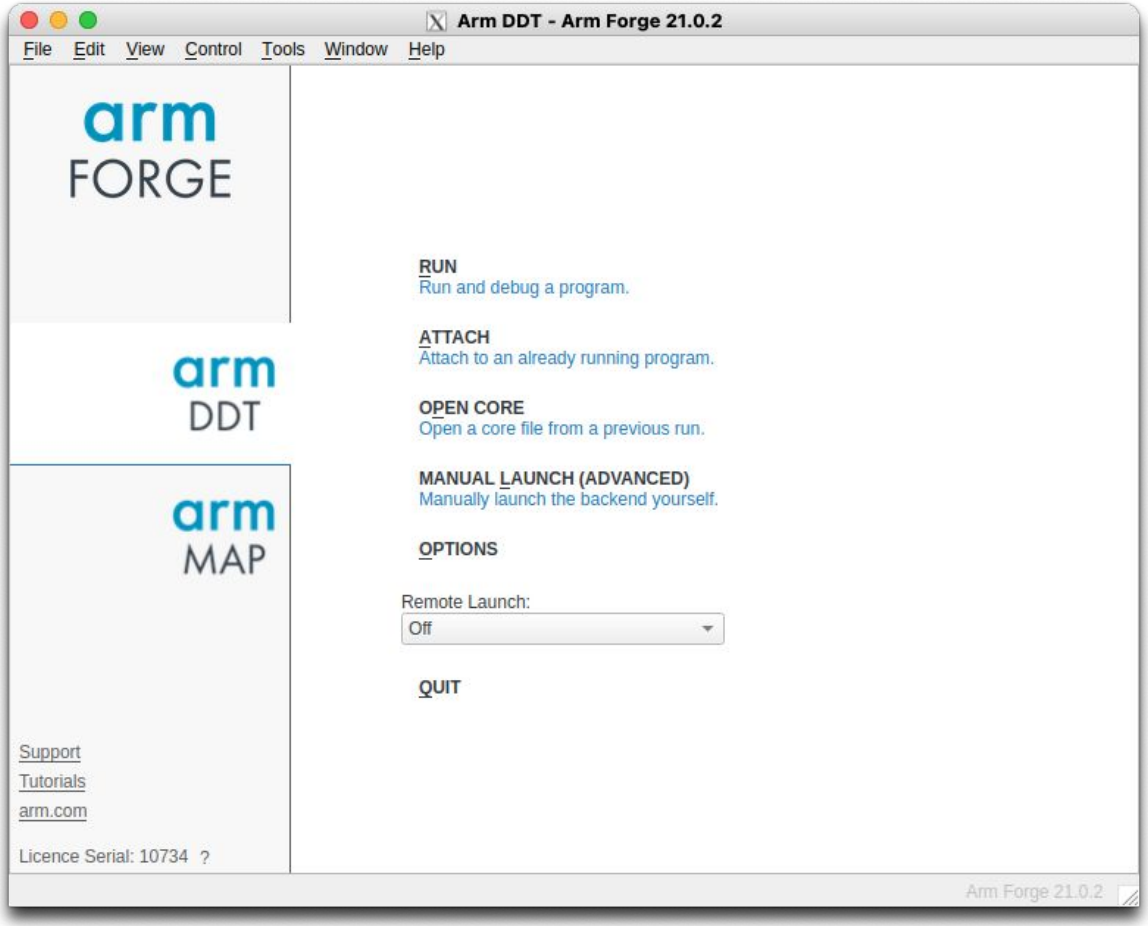
- CPU
  - salloc [options] -q interactive -C cpu
- GPU
  - salloc [options] -q interactive -C gpu
- https://docs.nersc.gov/jobs/policy/#qos-limits-and-charges

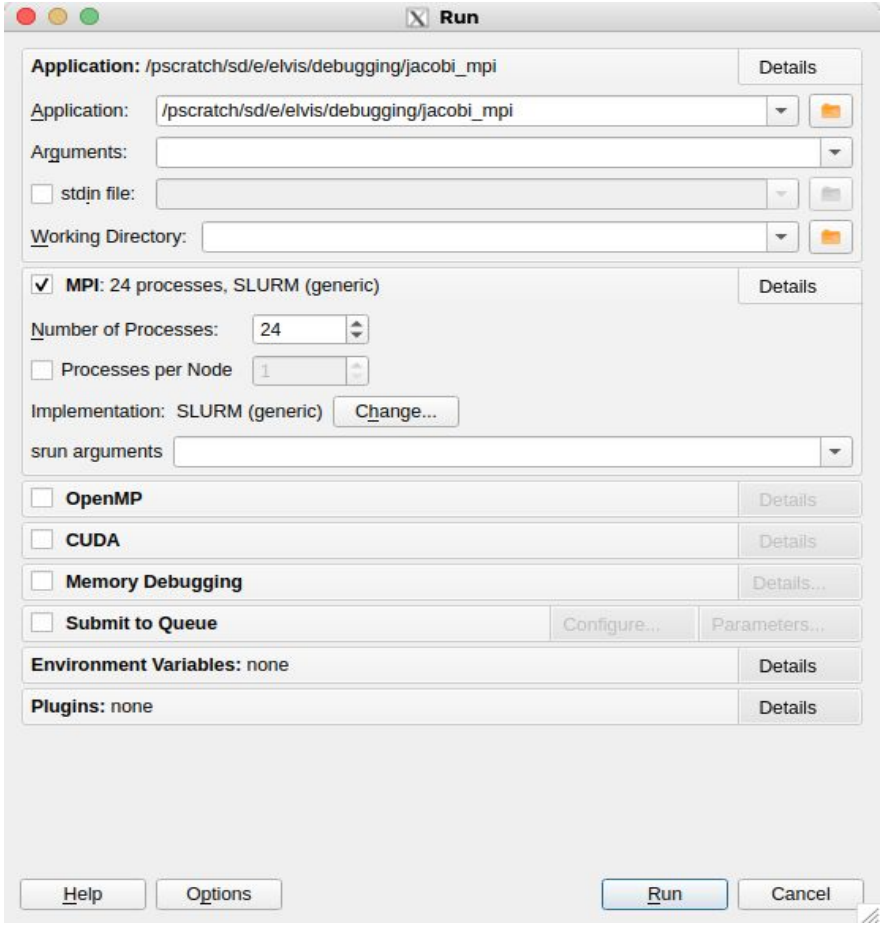
Distributed Debugging Tool (DDT)

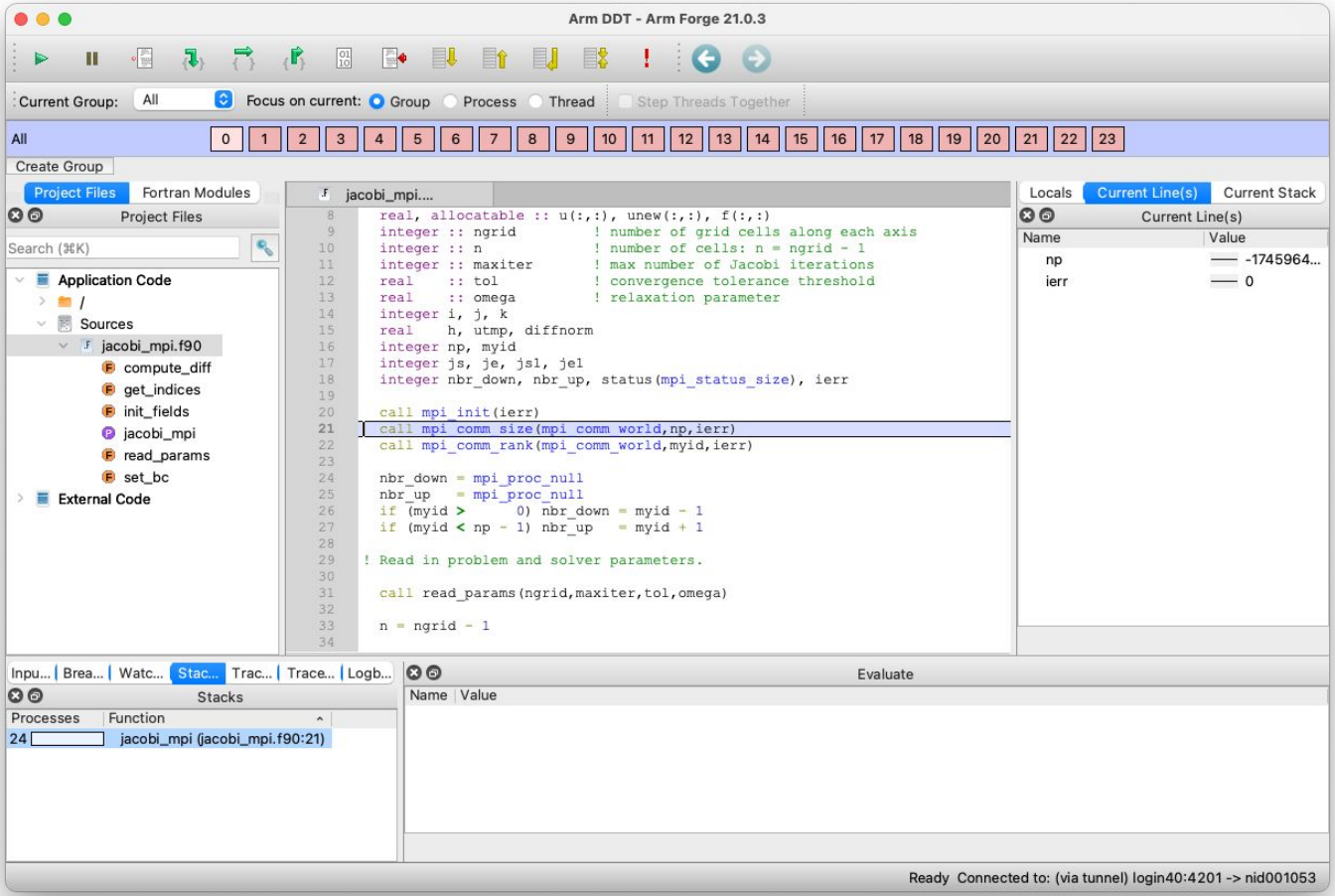


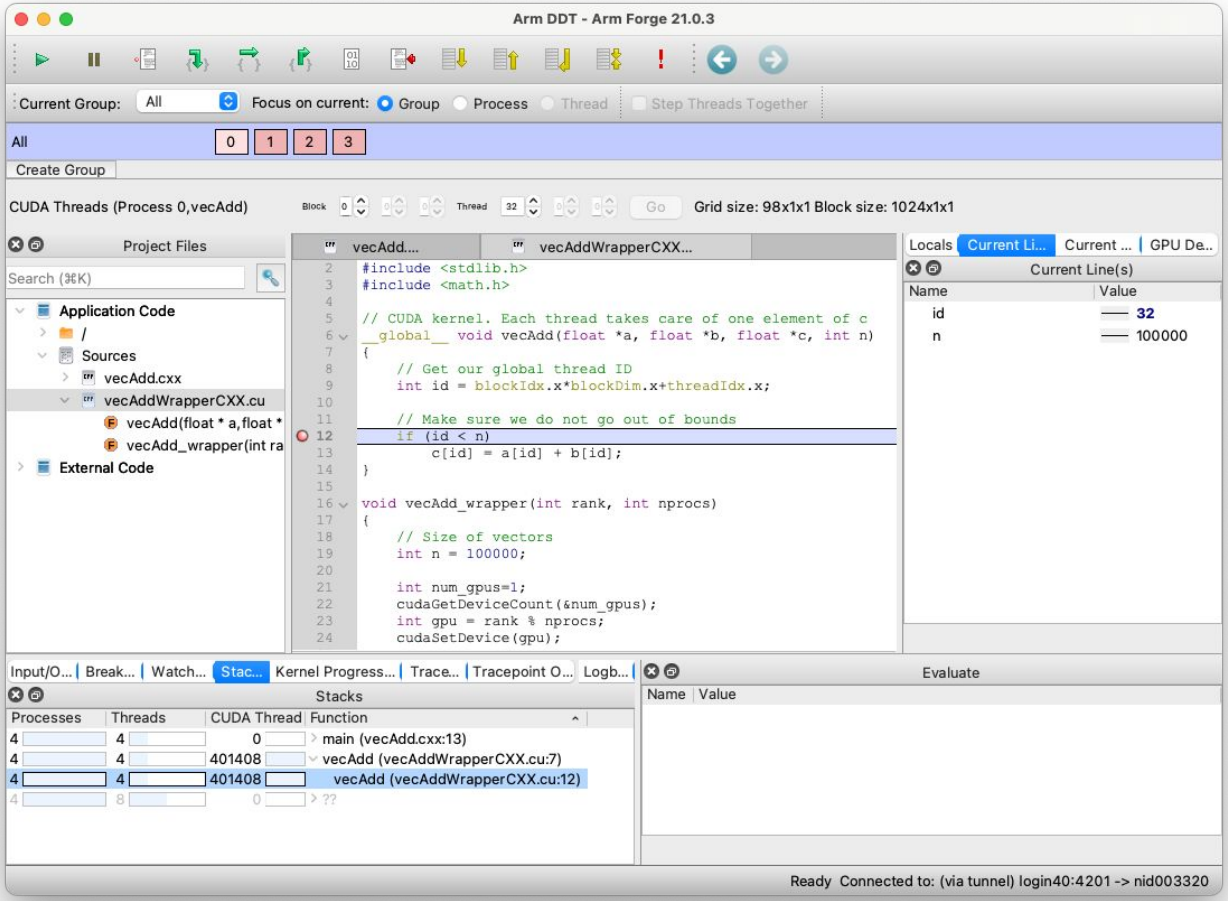
Debugging programs with DDT

- Supports MPI, OpenMP, OpenACC, CUDA, Python
- Developed by Linaro (previously Allinea, ARM)
- Remote client
  - https://docs.nersc.gov/tools/debug/ddt/#reverse-connect-using-remote-client
- module load forge
- ddt [options] ./program
- https://docs.nersc.gov/tools/debug/ddt/

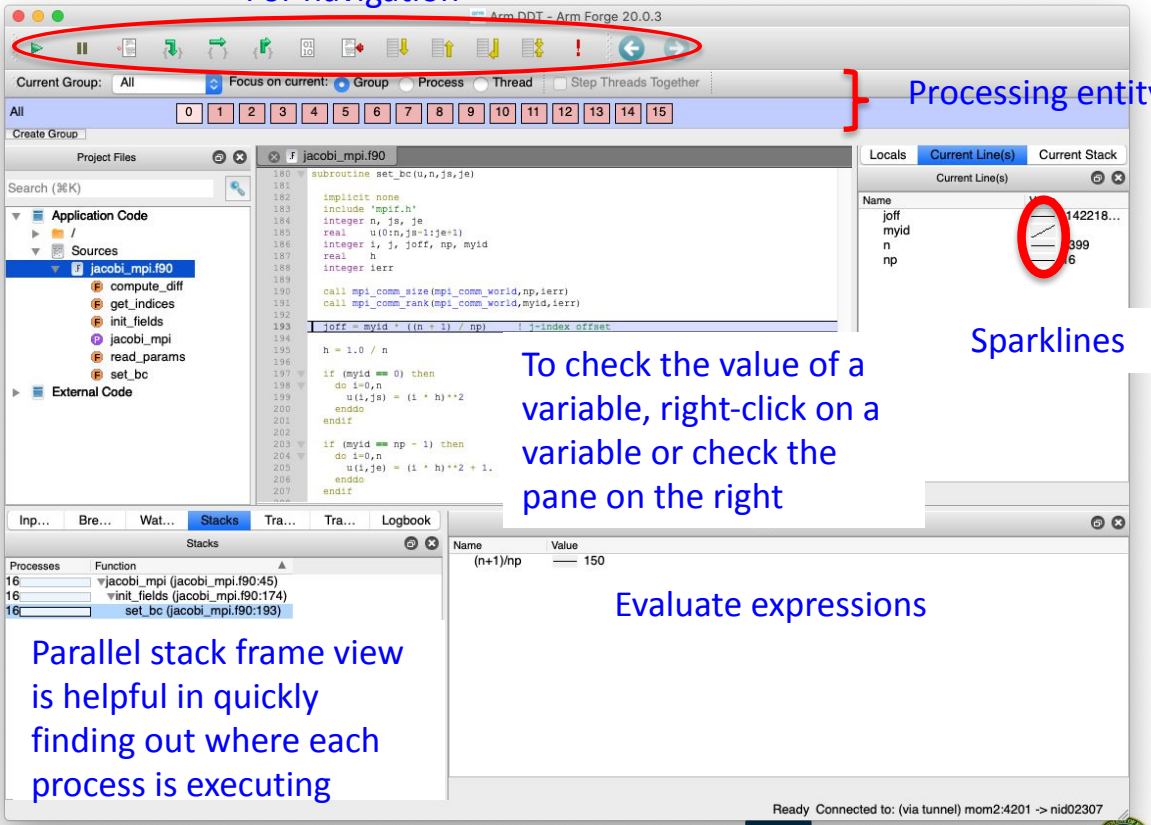

Arm DDT - Arm Forge 21.0.2
File Edit View Control Tools Window Help
arm FORGE
arm DDT
arm MAP
RUN
Run and debug a program.
ATTACH
Attach to an already running program.
OPEN CORE
Open a core file from a previous run.
MANUAL LAUNCH (ADVANCED)
Manually launch the backend yourself.
OPTIONS
Remote Launch:
Off
QUIT
Support
Tutorials
arm.com
Licence Serial: 10734 ?
Arm Forge 21.0.2


Run
Application: /pscratch/sd/e/elvis/debugging/jacobi_mpi
Details
Application:
/pscratch/sd/e/elvis/debugging/jacobi_mpi
Arguments:
stdin file:
Working Directory:
MPI: 24 processes, SLURM (generic)
Details
Number of Processes:
24
Processes per Node
1
Implementation: SLURM (generic)
Change...
srun arguments
OpenMP
Details
CUDA
Details
Memory Debugging
Details...
Submit to Queue
Configure...
Parameters...
Environment Variables: none
Details
Plugins: none
Details
Help
Options
Run
Cancel




Arm DDT - Arm Forge 21.0.3
Current Group: All
Focus on current: Group Process Thread
Step Threads Together
All 0 1 2 3
Create Group
CUDA Threads (Process 0,vecAdd)
Block 0
Thread 32
Go
Grid size: 98x1x1 Block size: 1024x1x1
Project Files
Search (⌘K)
Application Code
/
Sources
vecAdd.cxx
vecAddWrapperCXX.cu
vecAdd(float * a,float *
vecAdd_wrapper(int ra
External Code
vecAdd....
vecAddWrapperCXX...
#include <stdlib.h>
#include <math.h>
// CUDA kernel. Each thread takes care of one element of c
__global__ void vecAdd(float *a, float *b, float *c, int n)
{
// Get our global thread ID
int id = blockIdx.x*blockDim.x+threadIdx.x;
// Make sure we do not go out of bounds
if (id < n)
c[id] = a[id] + b[id];
}
void vecAdd_wrapper(int rank, int nprocs)
{
// Size of vectors
int n = 100000;
int num_gpus=1;
cudaGetDeviceCount(&num_gpus);
int gpu = rank % nprocs;
cudaSetDevice(gpu);
Locals Current Li... Current ... GPU De...
Current Line(s)
Name Value
id 32
n 100000
Input/O... Break... Watch... Stac... Kernel Progress... Trace... Tracepoint O... Logb...
Stacks
Processes Threads CUDA Thread Function
4 4 0 main (vecAdd.cxx:13)
4 4 401408 vecAdd (vecAddWrapperCXX.cu:7)
4 4 401408 vecAdd (vecAddWrapperCXX.cu:12)
4 8 0 ??
Evaluate
Name Value
Ready Connected to: (via tunnel) login40:4201 -> nid003320

For navigation



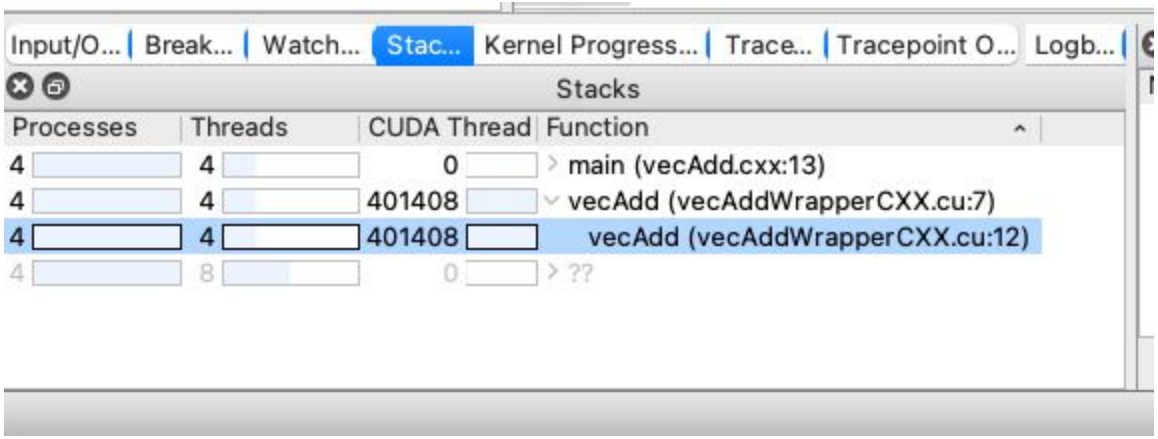
Processing entity to control

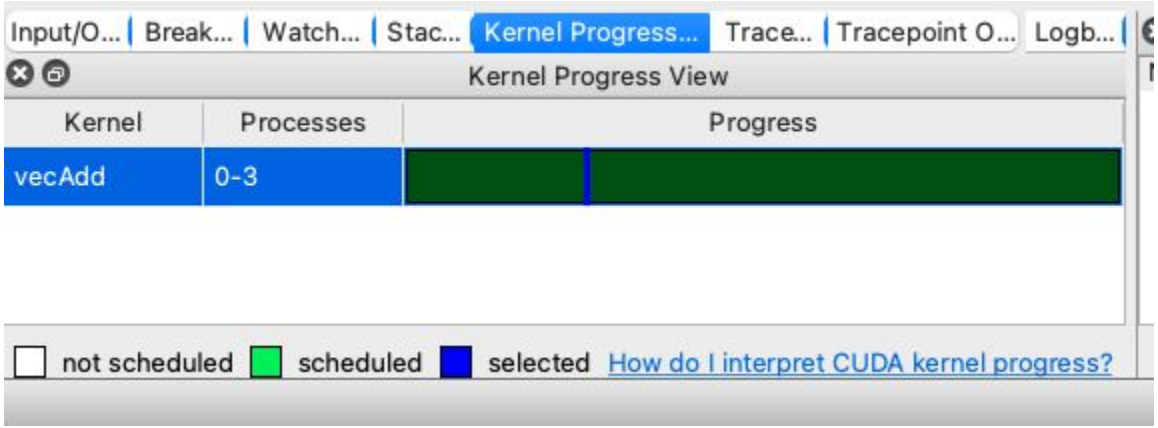
Sparklines

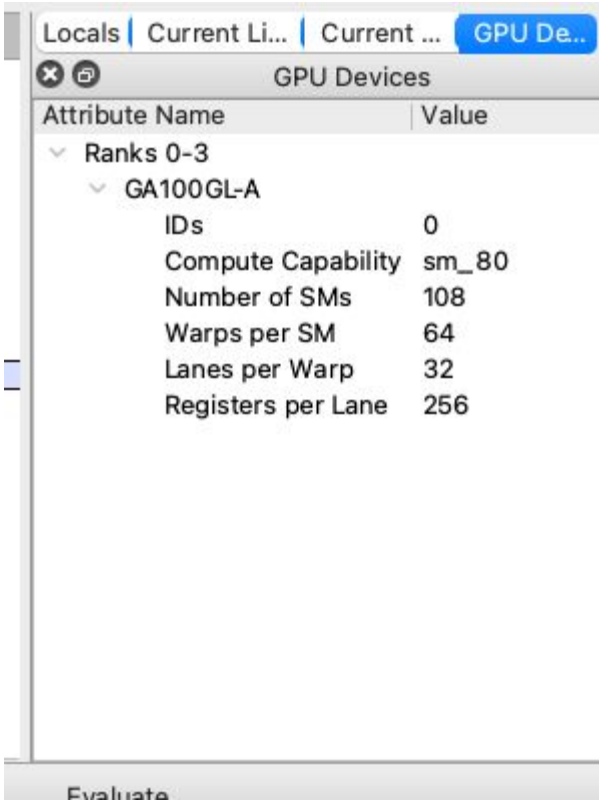
To check the value of a variable, right-click on a variable or check the pane on the right

Evaluate expressions

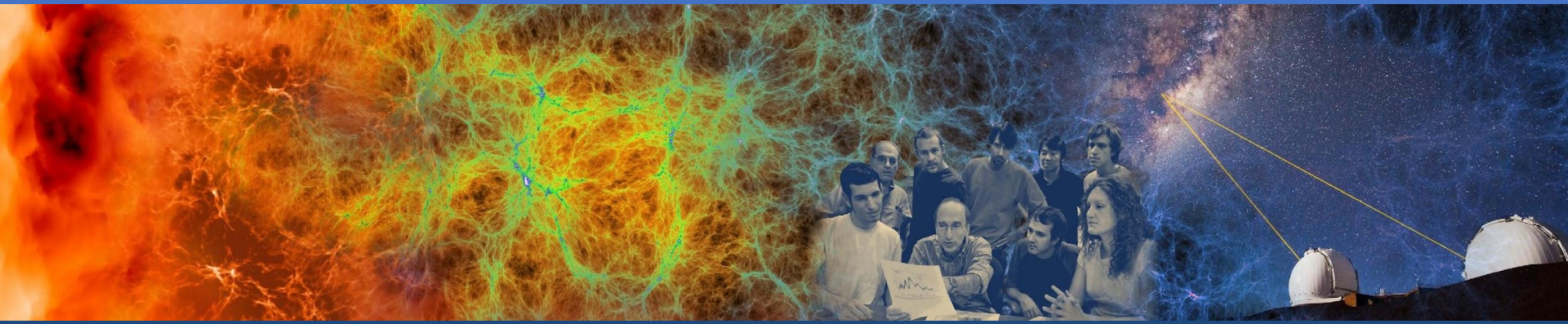
Parallel stack frame view is helpful in quickly finding out where each process is executing


Input/O... Break... Watch... Stac... Kernel Progress... Trace... Tracepoint O... Logb...
Stacks
Processes Threads CUDA Thread Function
4 4 0 main (vecAdd.cxx:13)
4 4 401408 vecAdd (vecAddWrapperCXX.cu:7)
4 4 401408 vecAdd (vecAddWrapperCXX.cu:12)
4 8 0 ??


Input/O... Break... Watch... Stac... Kernel Progress... Trace... Tracepoint O... Logb...
Kernel Progress View
Kernel Processes Progress
vecAdd 0-3
not scheduled scheduled selected How do I interpret CUDA kernel progress?

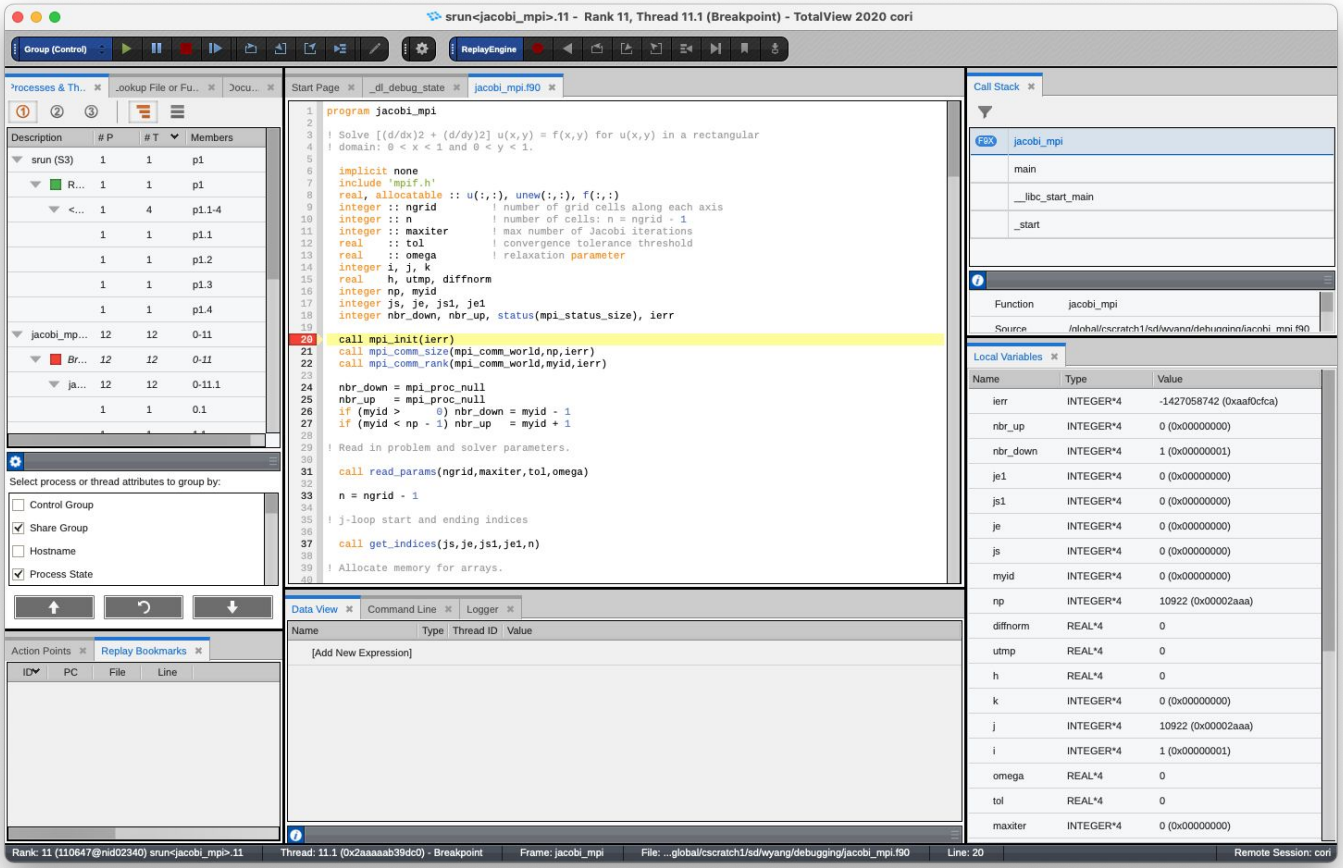

Locals Current Li... Current ... GPU De...
GPU Devices
Attribute Name Value
Ranks 0-3
GA100GL-A
IDs 0
Compute Capability sm_80
Number of SMs 108
Warps per SM 64
Lanes per Warp 32
Registers per Lane 256
Evaluate

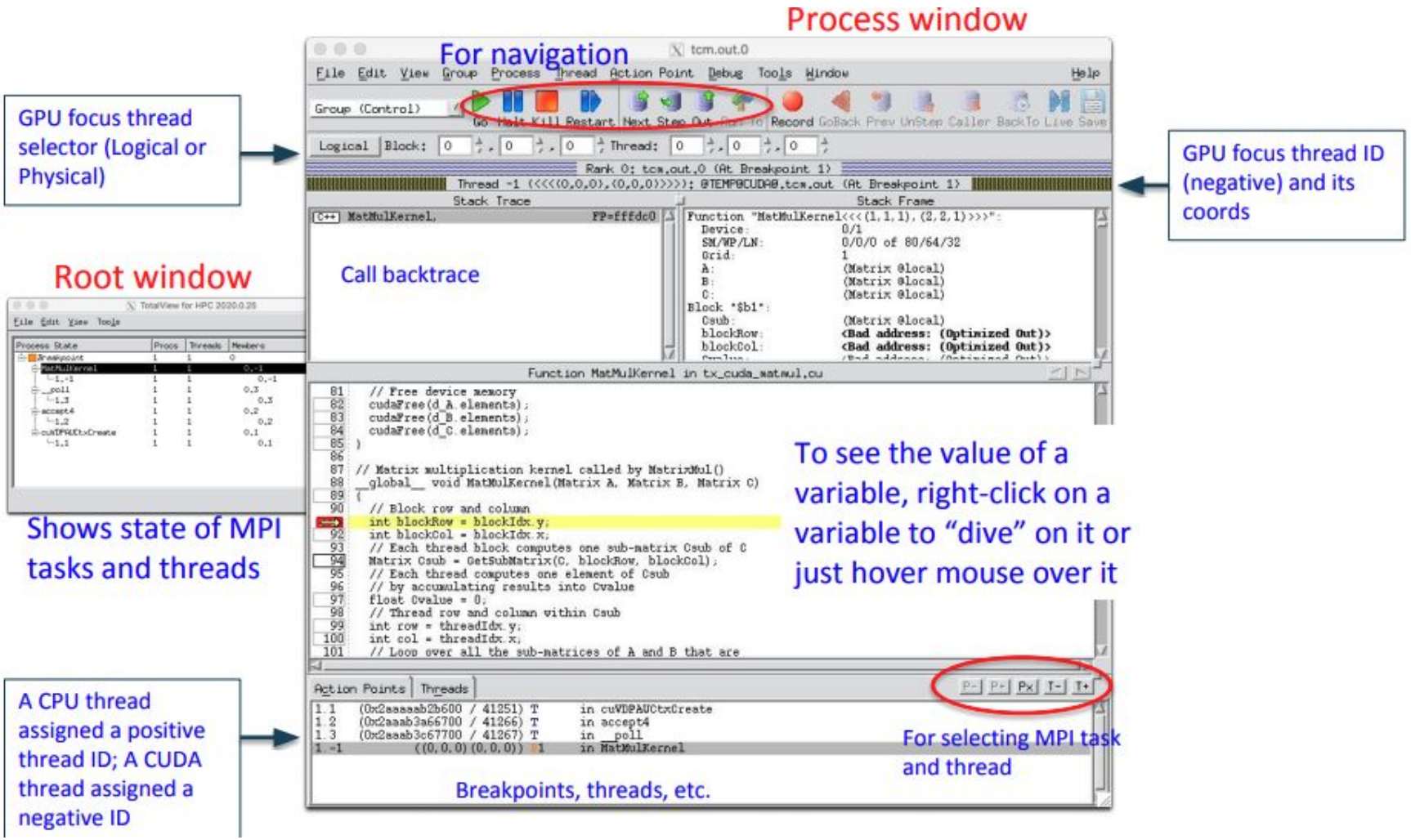
TotalView



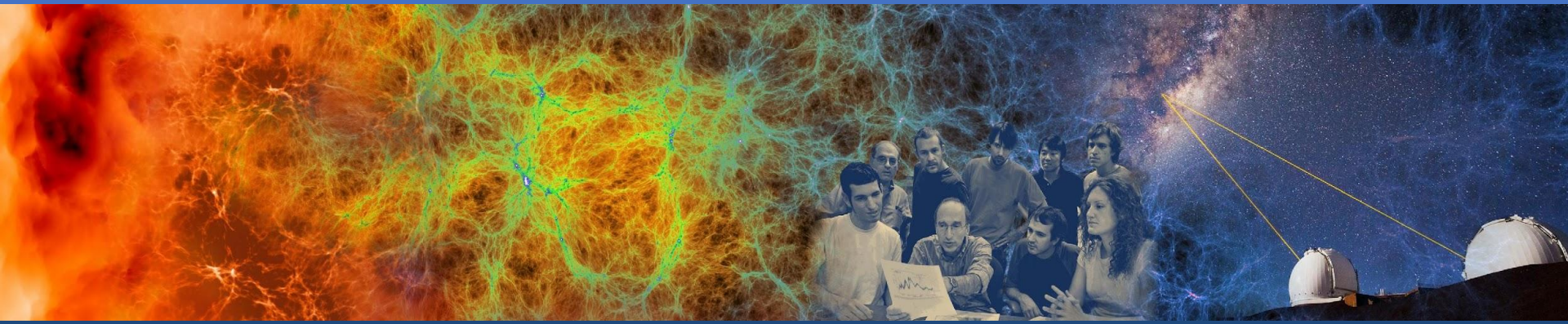
Debugging programs with TotalView

- Supports MPI, OpenMP, OpenACC, CUDA
- Developed by Perforce
- Remote client
  - https://docs.nersc.gov/tools/debug/totalview/#remote-display-client
- Remote connection
  - https://docs.nersc.gov/tools/debug/totalview/#remote-connections
- module load totalview
- totalview srun -a [options] ./program
- man totalview
- https://docs.nersc.gov/tools/debug/totalview/
- Review a previous training session
  - https://www.nersc.gov/users/training/events/totalview-tutorial-september-29-2022/


srun<jacobi_mpi>.11 - Rank 11, Thread 11.1 (Breakpoint) - TotalView 2020 cori
Group (Control)
ReplayEngine
Processes & Th..
Lookup File or Fu..
Docu..
Description
P
T
Members
srun (S3) 1 1 p1
R... 1 1 p1
<... 1 4 p1.1-4
1 1 p1.1
1 1 p1.2
1 1 p1.3
1 1 p1.4
jacobi_mp... 12 12 0-11
Br... 12 12 0-11
ja... 12 12 0-11.1
1 1 0.1
Select process or thread attributes to group by:
Control Group
Share Group
Hostname
Process State
Action Points
Replay Bookmarks
ID
PC
File
Line
Start Page
_dl_debug_state
jacobi_mpi.f90
1 program jacobi_mpi
2
3 ! Solve [(d/dx)2 + (d/dy)2] u(x,y) = f(x,y) for u(x,y) in a rectangular
4 ! domain: 0 < x < 1 and 0 < y < 1.
5
6 implicit none
7 include 'mpif.h'
8 real, allocatable :: u(:,:), unew(:,:), f(:,:)
9 integer :: ngrid ! number of grid cells along each axis
10 integer :: n ! number of cells: n = ngrid - 1
11 integer :: maxiter ! max number of Jacobi iterations
12 real :: tol ! convergence tolerance threshold
13 real :: omega ! relaxation parameter
14 integer i, j, k
15 real h, utmp, diffnorm
16 integer np, myid
17 integer js, je, js1, je1
18 integer nbr_down, nbr_up, status(mpi_status_size), ierr
19
20 call mpi_init(ierr)
21 call mpi_comm_size(mpi_comm_world,np,ierr)
22 call mpi_comm_rank(mpi_comm_world,myid,ierr)
23
24 nbr_down = mpi_proc_null
25 nbr_up = mpi_proc_null
26 if (myid > 0) nbr_down = myid - 1
27 if (myid < np - 1) nbr_up = myid + 1
28
29 ! Read in problem and solver parameters.
30
31 call read_params(ngrid,maxiter,tol,omega)
32
33 n = ngrid - 1
34
35 ! j-loop start and ending indices
36
37 call get_indices(js,je,js1,je1,n)
38
39 ! Allocate memory for arrays.
40
Data View
Command Line
Logger
Name
Type
Thread ID
Value
[Add New Expression]
Call Stack
F90 jacobi_mpi
main
__libc_start_main
_start
Function jacobi_mpi
Source /global/cscratch1/sd/wyang/debugging/jacobi_mpi.f90
Local Variables
Name Type Value
ierr INTEGER*4 -1427058742 (0xaaf0cfca)
nbr_up INTEGER*4 0 (0x00000000)
nbr_down INTEGER*4 1 (0x00000001)
je1 INTEGER*4 0 (0x00000000)
js1 INTEGER*4 0 (0x00000000)
je INTEGER*4 0 (0x00000000)
js INTEGER*4 0 (0x00000000)
myid INTEGER*4 0 (0x00000000)
np INTEGER*4 10922 (0x00002aaa)
diffnorm REAL*4 0
utmp REAL*4 0
h REAL*4 0
k INTEGER*4 0 (0x00000000)
j INTEGER*4 10922 (0x00002aaa)
i INTEGER*4 1 (0x00000001)
omega REAL*4 0
tol REAL*4 0
maxiter INTEGER*4 0 (0x00000000)
Rank: 11 (110647@nid02340) srun<jacobi_mpi>.11
Thread: 11.1 (0x2aaaaab39dc0) - Breakpoint
Frame: jacobi_mpi
File: ...global/cscratch1/sd/wyang/debugging/jacobi_mpi.f90
Line: 20
Remote Session: cori


Process window
For navigation
GPU focus thread selector (Logical or Physical)
GPU focus thread ID (negative) and its coords
Root window
Call backtrace
Shows state of MPI tasks and threads
To see the value of a variable, right-click on a variable to "dive" on it or just hover mouse over it
A CPU thread assigned a positive thread ID; A CUDA thread assigned a negative ID
For selecting MPI task and thread
Breakpoints, threads, etc.

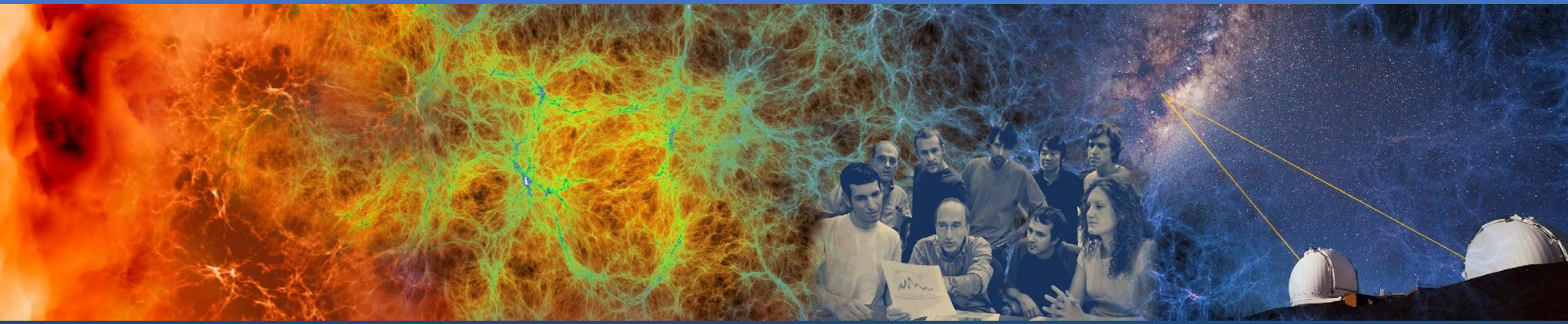
CUDA for the GNU Debugger (CUDA-gdb)



Debug cuda programs with cuda-gdb

- An extension to GDB that supports cuda programs
- Developed by NVIDIA
- cuda-gpu [options] ./program [core-file]
- (cuda-gdb) help
- (cuda-gdb) bt
- (cuda-gdb) list
- (cuda-gdb) help cuda
- cuda-gdb --help
- https://docs.nvidia.com/cuda/cuda-gdb/index.html

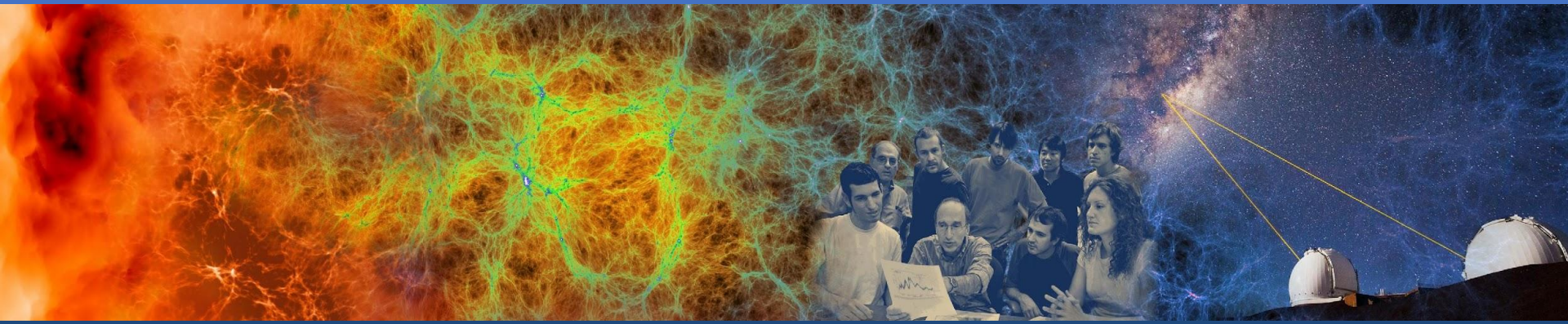
Compute Sanitizer (aka CUDA-memcheck)



Perform dynamic analysis of cuda programs with compute-sanitizer

- Use several tools to check program correctness at run-time
- Dynamic instrumentation at compile time
- Developed by NVIDIA
- srun compute-sanitizer --tool=memcheck [options] ./program
- memcheck, Detect memory errors
- racecheck, Detect race conditions
- initcheck, Detect use of uninitialized variables
- syncheck, Detect sync errors
- https://docs.nvidia.com/compute-sanitizer/ComputeSanitizer/index.html

GNU Debugger for HPC (gdb4hpc)



Debug parallel programs with gdb4hpc

- An extension to GDB that supports parallel programming models
- Does not support GPUs
- Developed by Cray / HPE
- module load gdb4hpc
- gdb4hpc
- (debug all) launch $p{N} ./program
- (debug all) help viewset
- (debug all) viewset $p
- man gdb4hpc
- https://docs.nersc.gov/tools/debug/gdb4hpc_ccdb/#parallel-debugging-with-gdb4hpc

```
dbg all> launch $p{8} ./pcm
Starting application, please wait...
Creating MRNet communication network...
sbcast: error: No compression library available, compression disabled.
sbcast: error: No compression library available, compression disabled.
Waiting for debug servers to attach to MRNet communications network...
Timeout in 400 seconds. Please wait for the attach to complete.
Number of dbgsrvs connected: [0];  Timeout Counter: [1]
Number of dbgsrvs connected: [1];  Timeout Counter: [0]
Number of dbgsrvs connected: [1];  Timeout Counter: [1]
Number of dbgsrvs connected: [1];  Timeout Counter: [2]
Number of dbgsrvs connected: [8];  Timeout Counter: [0]
Finalizing setup...
Launch complete.
p{0..7}: Initial breakpoint, main at /global/u1/j/jscook/hpe-tools/pi_calc_mpi.c:25
```

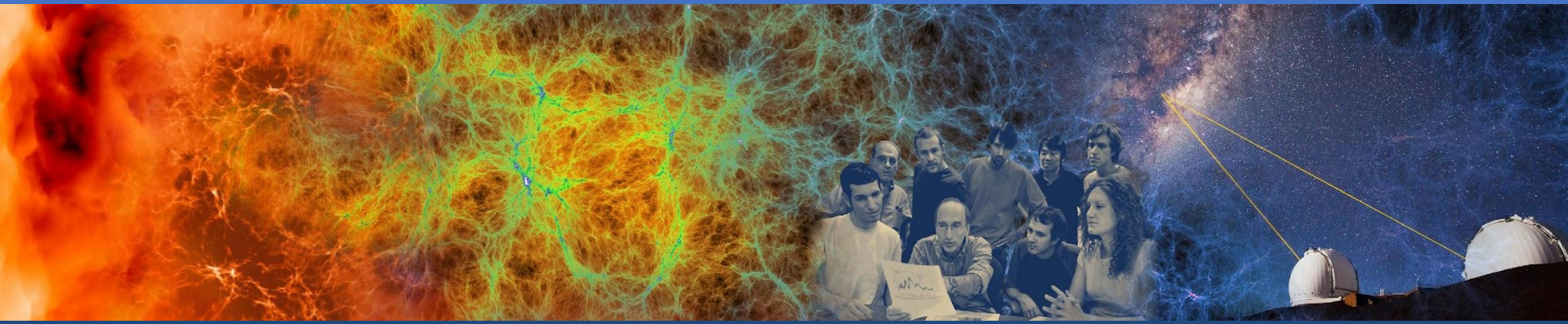
```
dbg all> list
p{0..7}: 25       reqSteps = 31415;          /* running about 31 thousand steps */
p{0..7}: 26       mySumBuf = 0;
p{0..7}: 27       mySum = 0;
p{0..7}: 28
p{0..7}: 29       MPI_Init(&argc, &argv);
p{0..7}: 30       MPI_Comm_rank(MPI_COMM_WORLD, &myRank);
p{0..7}: 31       MPI_Comm_size(MPI_COMM_WORLD, &mpiSize);
p{0..7}: 32
p{0..7}: 33       if (myRank == 0) {
p{0..7}: 34
```

```
dbg all> viewset $p
Name       Procs
p          p{0..7}
```

```
p{0..7}: Breakpoint 1: file /global/u1/j/jscook/hpe-tools/pi_calc_mpi.c, line 31.
dbg all> print $p::myRank
p{0..7}: 0
```

```
p{0..7}: Breakpoint 1, main at /global/u1/j/jscook/hpe-tools/pi_calc_mpi.c:31
dbg all> print $p::myRank
p{0}: 0
p{1}: 1
p{2}: 2
p{3}: 3
p{4}: 4
p{5}: 5
p{6}: 6
p{7}: 7
dbg all> list
p{0..7}: 31       MPI_Comm_size(MPI_COMM_WORLD, &mpiSize);
p{0..7}: 32
p{0..7}: 33       if (myRank == 0) {
p{0..7}: 34
p{0..7}: 35         /* sum my share of the series */
p{0..7}: 36         mySum = sumFractions(reqSteps,  mpiSize,  myRank);
p{0..7}: 37
p{0..7}: 38         /* add up sums from all nodes */
p{0..7}: 39         for (int srcRank = 1; srcRank < mpiSize; srcRank++){
p{0..7}: 40             MPI_Recv(&mySumBuf, 1, MPI_DOUBLE, srcRank, 0, MPI_COMM_WORLD, &myStatus);
```

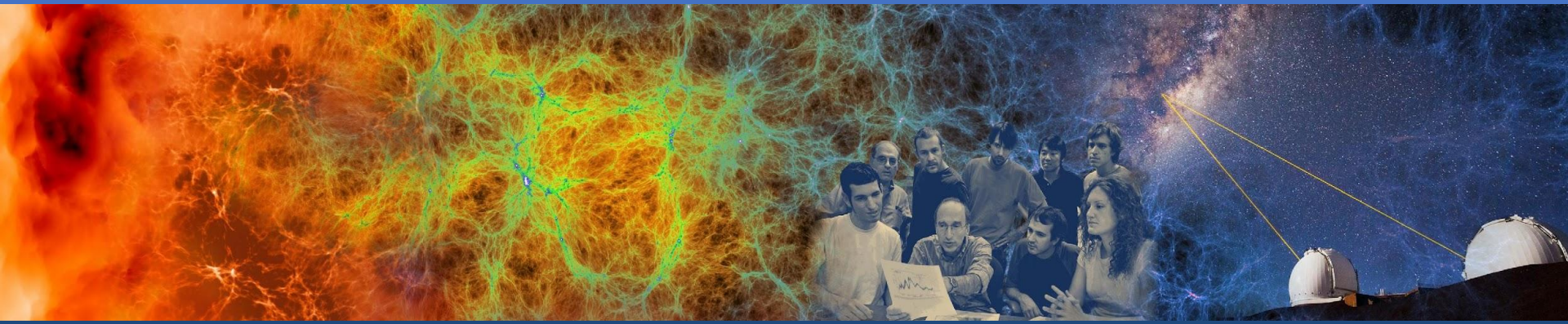
Valgrind for HPC (valgrind4hpc)



Perform dynamic analysis of parallel programs with valgrind4hpc

- Use several tools to check program correctness at run-time
- Dynamic instrumentation at compile time
- Does not support GPUs
- Based on valgrind
- Developed by HPE / Cray
- Aggregated messages/results from all MPI ranks
- Less spurious error messages than valgrind
- module load valgrind4hpc
- valgrind4hpc -n4 --tool=memcheck [launcher-args] [valgrind-args] ./program
- Tools: memcheck, helgrind, exp-sgcheck, drd
- man valgrind4hpc
- man valgrind
- https://docs.nersc.gov/tools/debug/valgrind/

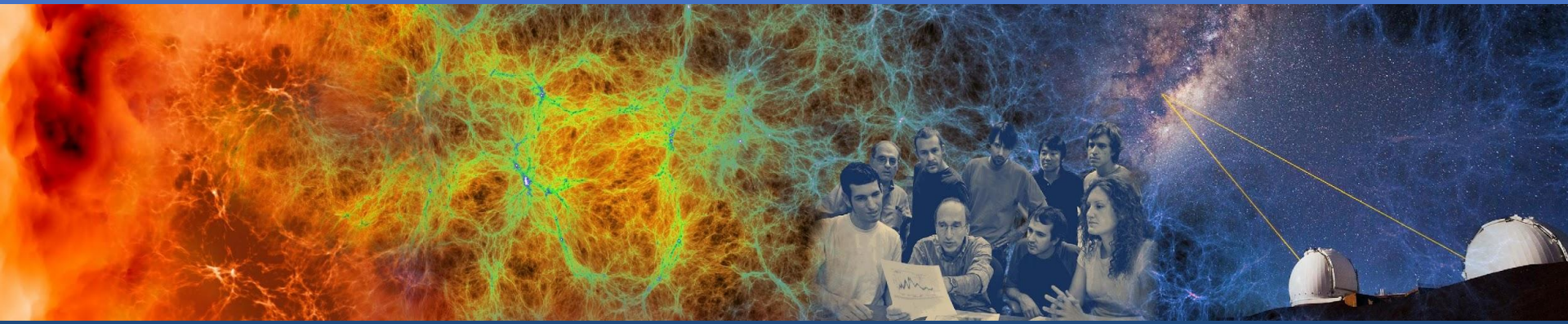
Sanitizers for HPC (sanitizers4hpc)



Perform dynamic analysis of parallel programs with sanitizers4hpc

- Use several tools to check program correctness at run-time
- Static instrumentation at compile time
- Aggregates report across all processes
- Based on LLVM Sanitizers
- Supports CCE, GCC
- Supports GPUs with cuda-memcheck
- Developed by HPE / Cray
- module swap PrgEnv-gnu PrgEnv-cray
- module load sanitizers4hpc
- -fsanitize=<sanitizer>
- Sanitizers: Address, Leak, Thread
- sanitizers4hpc -l “-n4” -- ./program
- man sanitizers4hpc
- https://github.com/google/sanitizers

Stack Trace Analysis Tool (STAT)

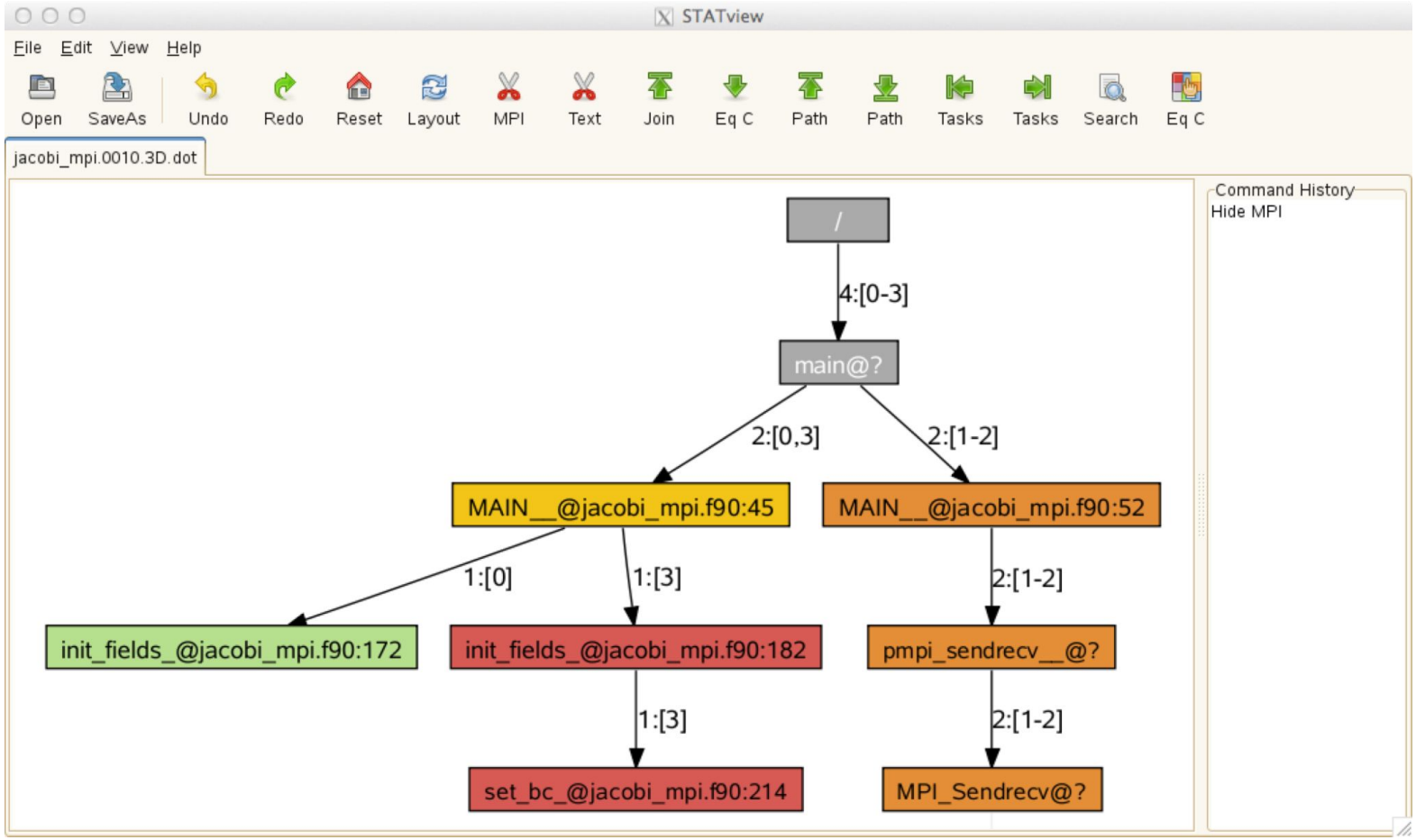


Debug deadlocked programs with STAT

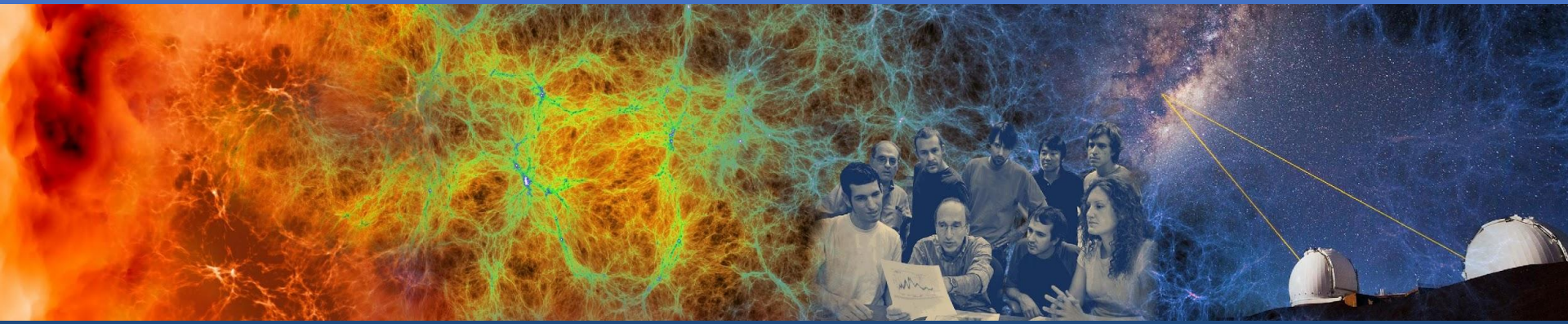
- Attach to a job launcher process
- Gathers and merges stack traces for all processes
- Supports MPI, threads, and cuda (using cuda-gdb)
- module load cray-stat
- srun [options] program &
- <program-pid will output>
- stat-cl [options] program-pid
- stat-view stat-output-file
- man stat-cl
- man stat-view
- https://docs.nersc.gov/tools/debug/stat_atp/#stat

```
$ ftn -g -o jacobi_mpi jacobi_mpi.f90
$ salloc -N 1 -t 30:00 -q debug -C knl
...
$ srun -n 4 -c 64 --cpu-bind=cores ./jacobi_mpi &
[1] 135543
$ module load stat
$ stat-cl -i 135543
...
Attaching to application...
Attached!
Application already paused... ignoring request to pause
Sampling traces...
Traces sampled!
...
Resuming the application...
Resumed!
Merging traces...
Traces merged!
Detaching from application...
Detached!

Results written to /global/cscratch1/sd/wyang/debugging/stat/stat_results/jacobi_mpi.0003
$ ls -l stat_results/jacobi_mpi.0003/*.dot
-rw-rw---- 1 wyang wyang 5201 Jun  7 14:55 stat_results/jacobi_mpi.0003/00_jacobi_mpi.0003.3D.dot
$ STATview stat_results/jacobi_mpi.0003/00_jacobi_mpi.0003.3D.dot
```

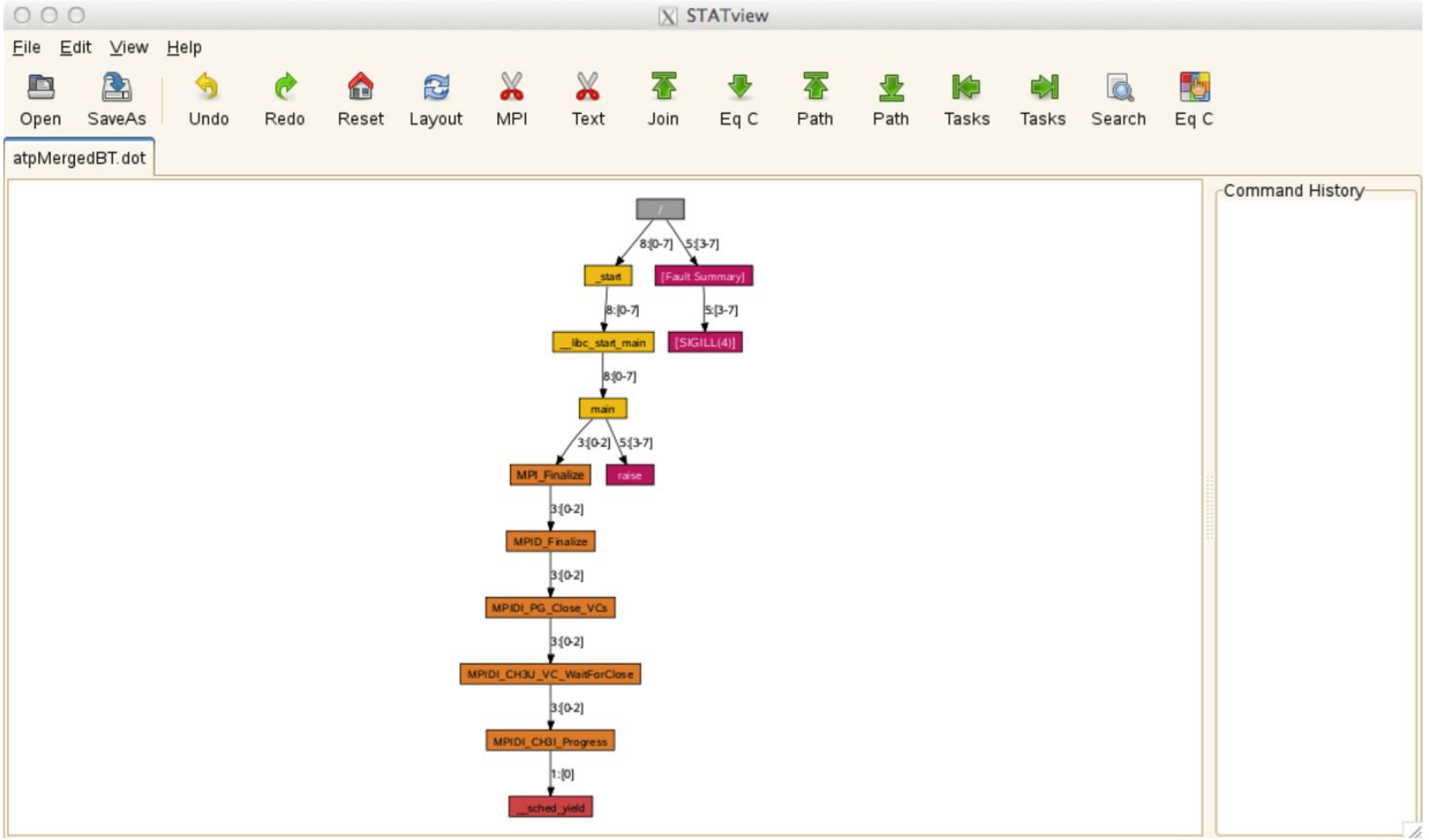

STATview
File Edit View Help
Open SaveAs Undo Redo Reset Layout MPI Text Join Eq C Path Path Tasks Tasks Search Eq C
jacobi_mpi.0010.3D.dot
Command History
Hide MPI
/
4:[0-3]
main@?
2:[0,3]
2:[1-2]
MAIN__@jacobi_mpi.f90:45
MAIN__@jacobi_mpi.f90:52
1:[0]
1:[3]
2:[1-2]
init_fields_@jacobi_mpi.f90:172
init_fields_@jacobi_mpi.f90:182
pmpi_sendrecv__@?
1:[3]
2:[1-2]
set_bc_@jacobi_mpi.f90:214
MPI_Sendrecv@?

Abnormal Termination Processing (ATP)



Debugging crashed programs with ATP

- Signal handler processes termination signals
- Gathers and merges stack traces for all processes
- Selectively produces core files
- Support MPI, threads, and cuda (using cuda-gdb)
- module load cray-stat
- module load atp
- export ATP_ENABLED=1
- export ATP_GDB_BINARY=$(which gdb) # optional
- -fno-backtrace # GNU Fortran
- srun [options] program
- <termination signal> or <app crashes>
- stat-view dot-file
- man atp
- https://docs.nersc.gov/tools/debug/stat_atp/#atp


STATview
File Edit View Help
Open SaveAs Undo Redo Reset Layout MPI Text Join Eq C Path Path Tasks Tasks Search Eq C
atpMergedBT.dot
Command History
/
8:[0-7] 5:[3-7]
_start [Fault Summary]
8:[0-7] 5:[3-7]
__libc_start_main [SIGILL(4)]
8:[0-7]
main
3:[0-2] 5:[3-7]
MPI_Finalize raise
3:[0-2]
MPID_Finalize
3:[0-2]
MPIDI_PG_Close_VCs
3:[0-2]
MPIDI_CH3U_VC_WaitForClose
3:[0-2]
MPIDI_CH3I_Progress
1:[0]
__sched_yield

Thank You and Welcome to NERSC!