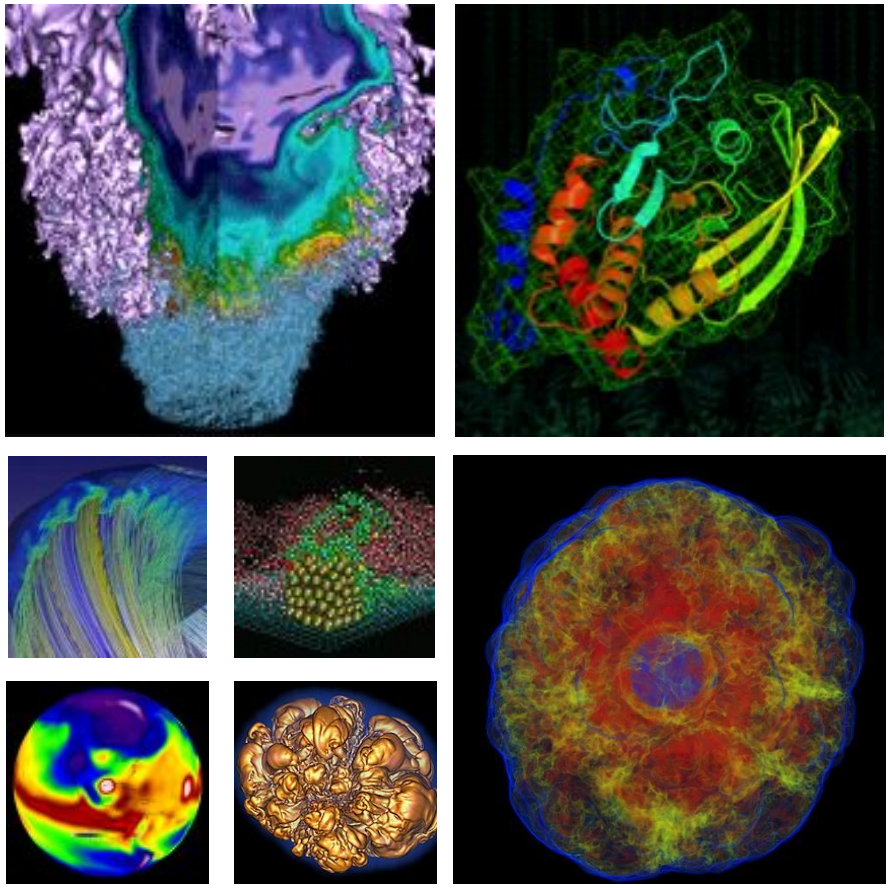
Debugging Tools

New User Training



Woo-Sun Yang

User Engagement Group, NERSC

January 25, 2019

Debuggers

- **Program errors**
  - Program crashes
  - Program hangs
  - Wrong results
- **How to find and fix them?**
  - Print statements
    - Difficult to guess where to put them and what to print
    - Recompile whenever you change them
    - Tedious and exhausting, especially for parallel codes
  - Debuggers
    - Compile only once (generally)
    - Control execution of your program
    - Check variables
    - Identify where the code fails or hangs and why

Parallel debuggers on Cori and Edison

- **Parallel debuggers with a graphical user interface**
  - DDT (Distributed Debugging Tool)
  - TotalView
- **Specialized debuggers on Cori and Edison**
  - STAT (Stack Trace Analysis Tool)
    - Collect stack backtraces from all (MPI) tasks
  - ATP (Abnormal Termination Processing)
    - Collect stack backtraces from all (MPI) tasks when an application fails
- **Valgrind**
  - Suite of debugging and profiling tools
  - Best known for its detailed memory debugging (memcheck)
  - https://docs.nersc.gov/development/performance-debugging-tools/valgrind/
- **Intel Inspector**
  - Thread and memory debugging
  - http://www.nersc.gov/users/software/performance-and-debugging-tools/inspector/
- **Cray debuggers for comparative debugging**
  - CCDB
  - lgdb

DDT and TotalView

- **GUI-based traditional parallel debuggers**
- **C, C++, Fortran codes with MPI, OpenMP, pthreads**
- **Licenses**
  - DDT: up to 8192 MPI tasks on Cori and Edison
  - TotalView: up to 512 MPI tasks on Cori and Edison
  - Shared among users and machines
- **For info**
  - https://developer.arm.com/products/software-development-tools/hpc/arm-forge
  - https://docs.nersc.gov/development/performance-debugging-tools/ddt/
  - https://www.roguewave.com/products-services/totalview
  - https://docs.nersc.gov/development/performance-debugging-tools/totalview/

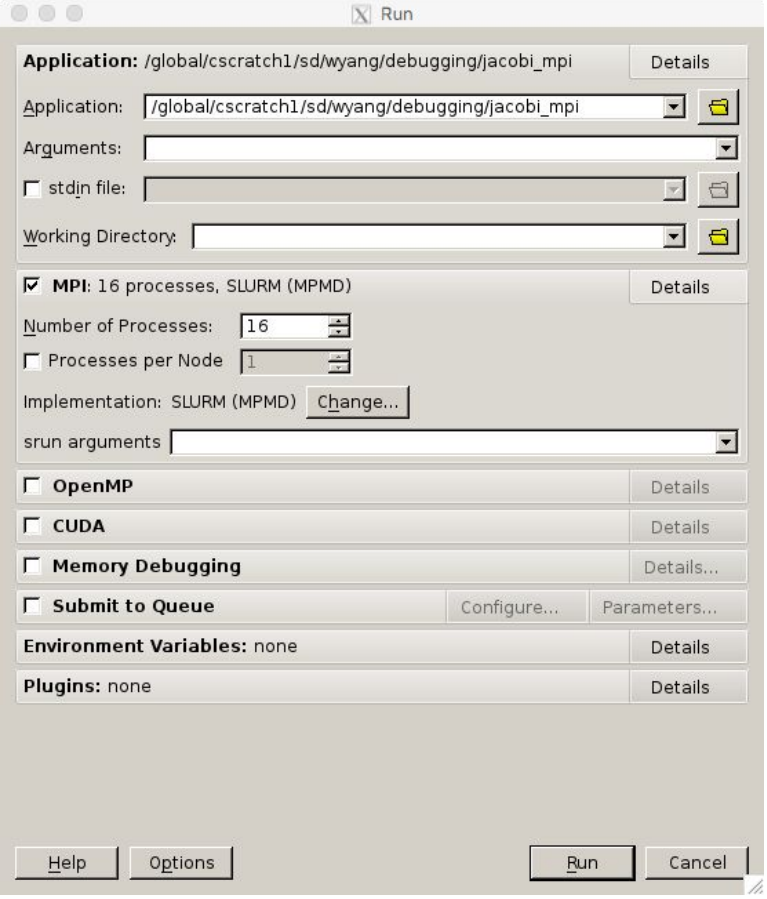
How to build and run with DDT

```
$ ftn -g -O0 -o jacobi_mpi jacobi_mpi.f90
```

-g for debugging symbols;
-O0 for the Intel compiler

```
$ salloc -N 1 -t 30:00 -q debug -C knl
$ module load allinea-forge
$ ddt ./jacobi_mpi
```

Start an interactive batch session
Load the allinea-forge module to use DDT
Start DDT

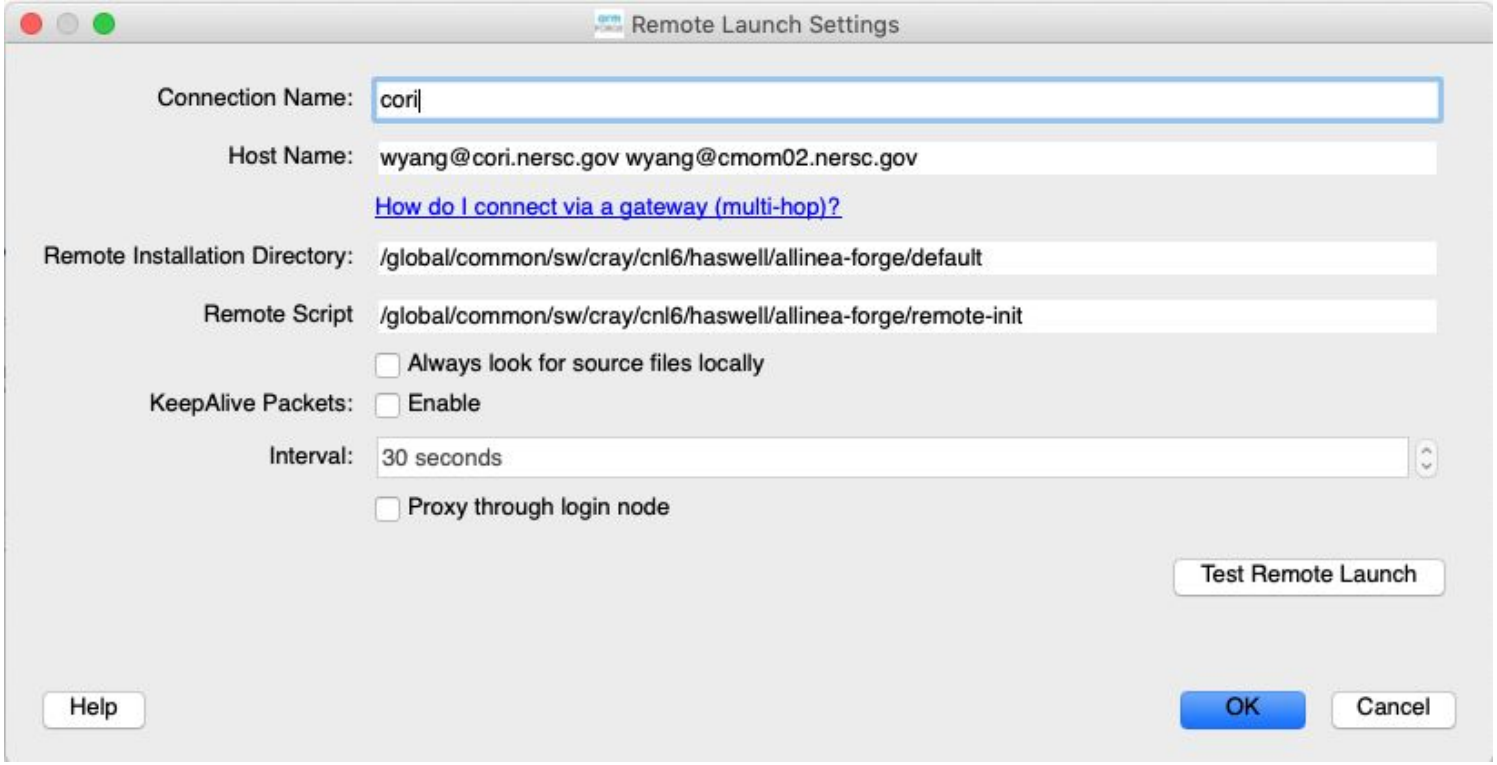


If you work far away from NERSC

- **Remote X11 window GUI application over network: painfully slow response**
- **Two solutions**
  - Use NX (NoMachine) to improve the speed
    - Works for X window applications
    - https://docs.nersc.gov/connect/nx/
  - Use Arm Forge remote client
    - Run on your desktop/laptop
    - Submit a debugging batch job on a NERSC machine and make the job connect to the client ("**reverse connect**")
    - Displays results in real time
    - https://docs.nersc.gov/development/performance-debugging-tools/ddt/#reverse-connect-using-remote-client (for setup)
    - https://developer.arm.com/products/software-development-tools/hpc/downloads/download-arm-forge (for downloading remote clients)

Arm Forge remote client settings

- **Uncheck the 'Proxy through login node' box -- for MFA authentication**



DDT window

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

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

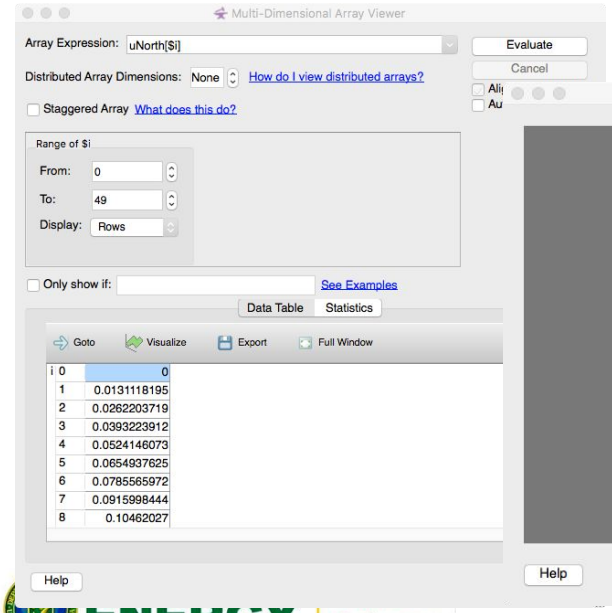
To evaluate expressions

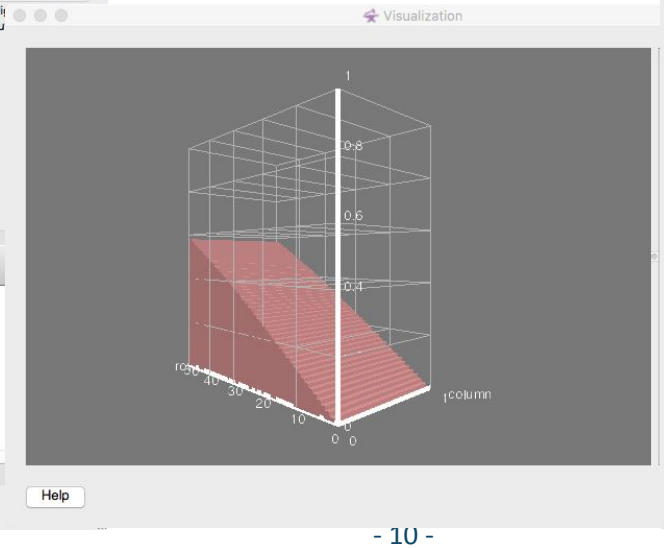
Breakpoints, watchpoints and tracepoints

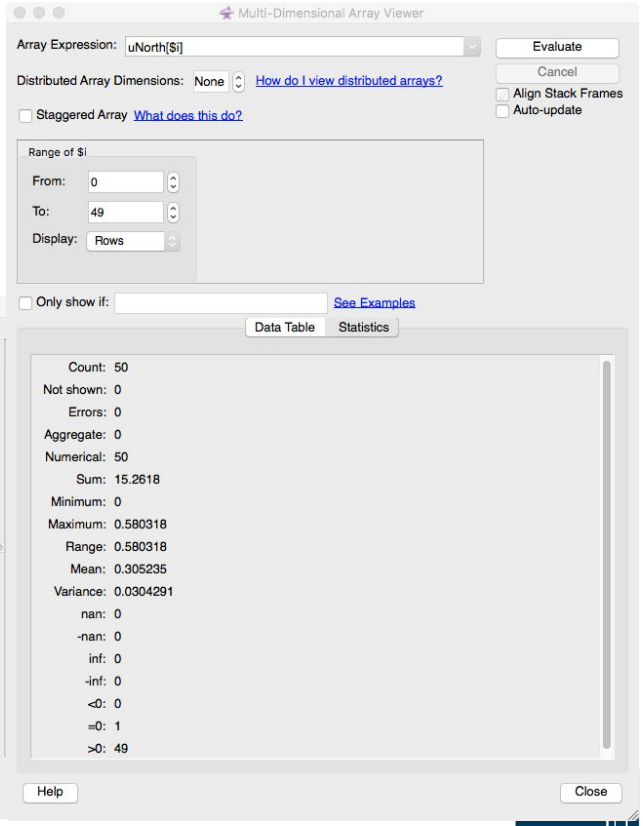
- **Breakpoint**
  - Stops execution when a selected line (breakpoint) is reached
  - Double click on a line to create one; there are other ways, too
- **Watchpoints for variables or expressions**
  - Stops when a variable or an expression changes its value
- **Tracepoints**
  - When reached, prints what lines of codes is being executed and the listed variables
- **Can add a condition for an action point**
  - Useful inside a loop
- **Can be made active or inactive**

Check variables

- **Right click on a variable for a quick summary**
- **Variable pane**
- **Evaluate pane**
- **Display variable values over processes (Compare across processes) or threads (Compare across threads)**
- **MDA (Multi-dimensional Array) Viewer**
  - Visualization
  - Statistics

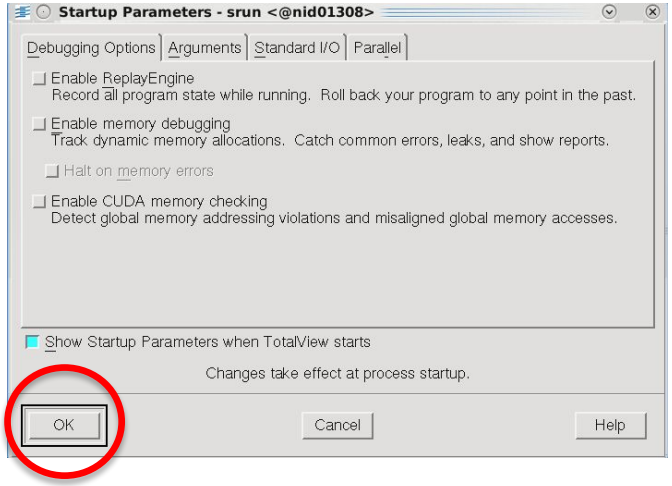






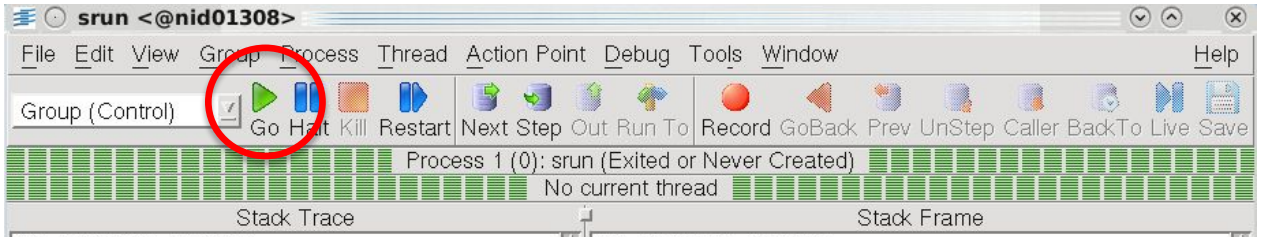
TotalView

```
$ salloc -N 1 -t 30:00 -q debug
$ module load totalview
$ export OMP_NUM_THREADS=6
$ totalview srun -a -n 4 ./jacobi_mpiomp
```

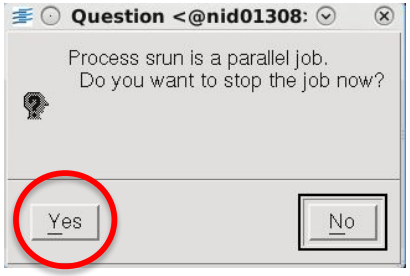


Then,

- Click OK in the 'Startup Parameters - srun' window
- Click 'Go' button in the main window

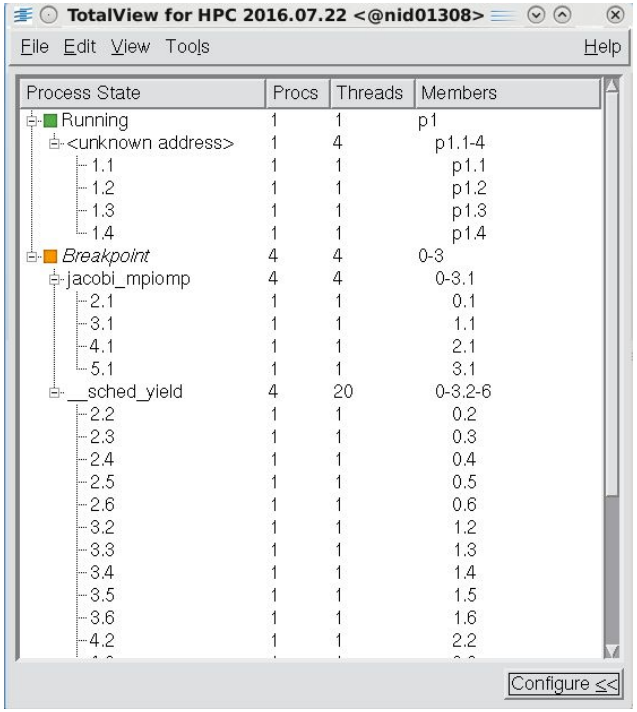


- Click 'Yes' to the question 'Process srun is a parallel job. Do you want to stop the job now?'



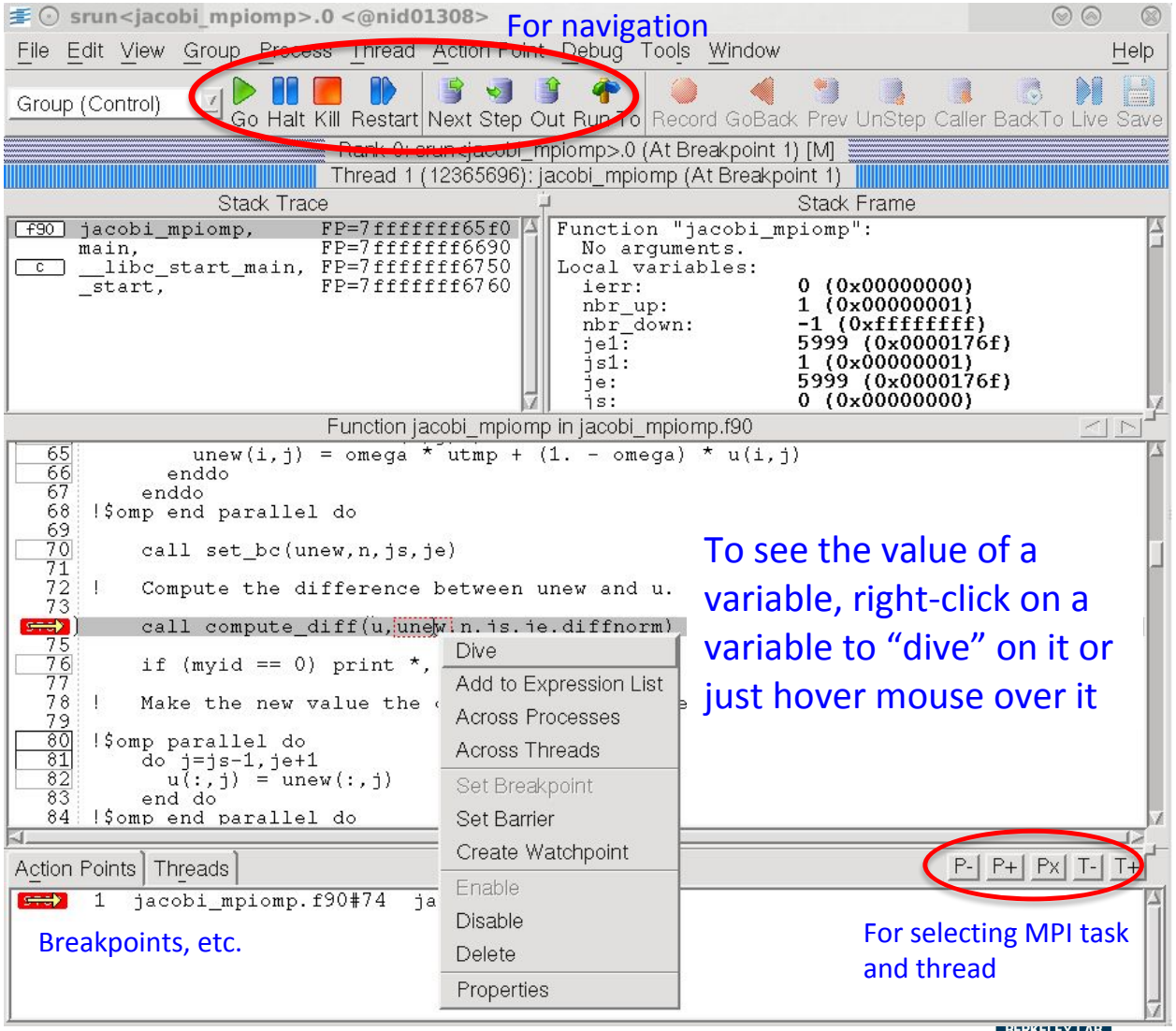
TotalView (cont'd)

Root window



State of MPI tasks and threads; members denoted roughly as '*rank.thread*'

Process window



To see the value of a variable, right-click on a variable to "dive" on it or just hover mouse over it

Breakpoints, etc.

For selecting MPI task and thread

STAT (Stack Trace Analysis Tool)

- **Gathers stack backtraces (sequence of function calls leading up to the current function) for all (MPI) processes**
  - Merge them into a single file (*.dot)
  - Results displayed as a single call tree for all processes
  - Can be useful for debugging a hanging application
  - With the info learned from STAT, can investigate further with DDT or TotalView
- **Works for MPI, CAF and UPC, OpenMP**

STAT (Cont'd)

- **STAT commands (after loading the 'stat' module)**
  - stat-cl: invokes STAT to gather stack backtraces
  - STATview: a GUI to view the results
  - STATGUI: a GUI to run STAT or view results
- **For more info:**
  - 'intro_stat', 'STAT', 'STATview' and 'STATGUI' man pages
  - /opt/cray/pe/stat/default/doc/stat_userguide.pdf
  - https://docs.nersc.gov/development/performance-debugging-tools/stat_atp/

Debug a hanging application with STAT

- **If your code hangs in a consistent manner, you can use STAT to see whether some MPI ranks got stuck.**

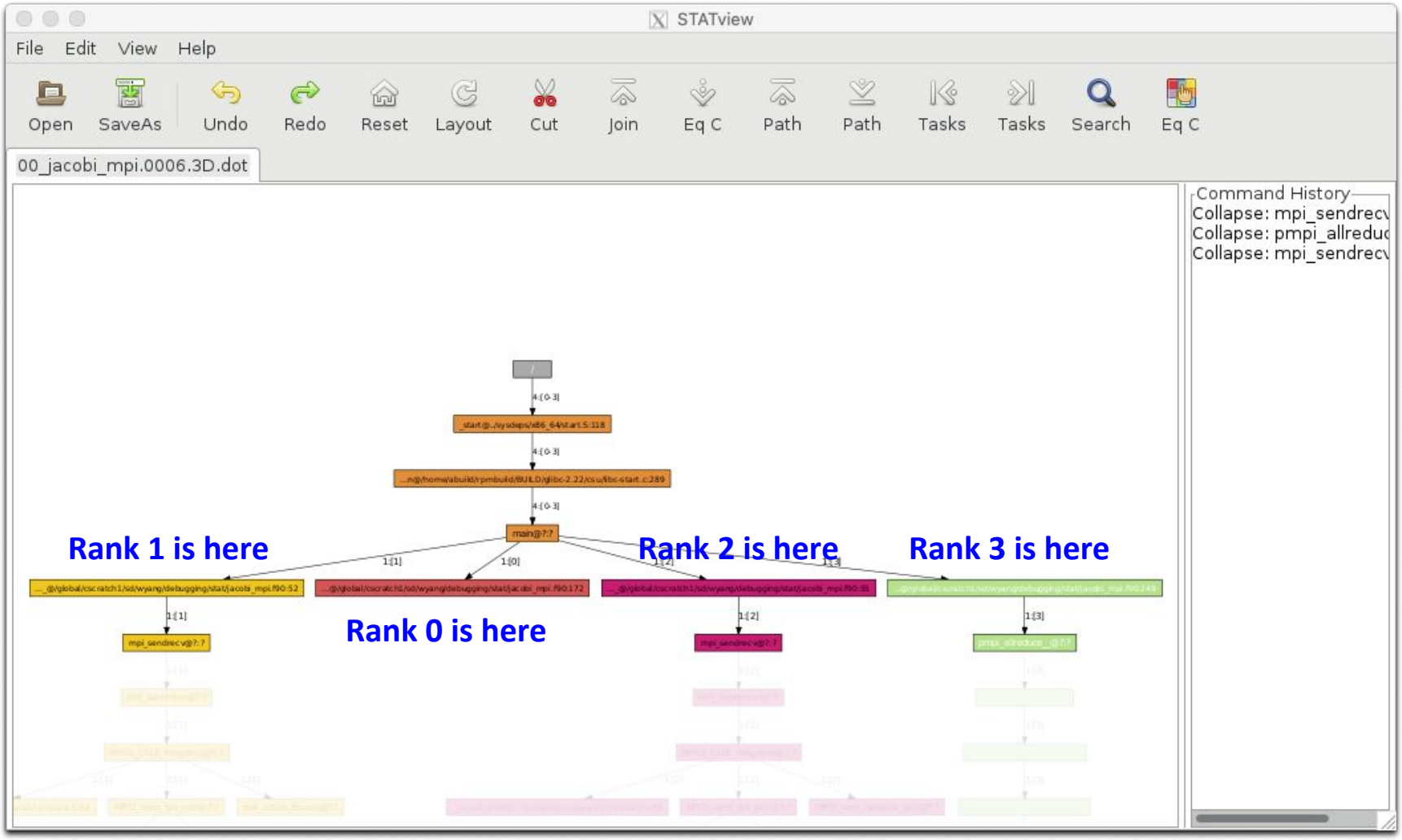
```
$ ftn -g -o jacobi_mpi jacobi_mpi.f90
$ salloc -N 2 -t 30:00 -q debug -C knl
...
$ srun -n 4 ... ./jacobi_mpi &
[1] 158190
$ module load stat
$ stat-cl -i 158190
…
Attaching to application...
Attached!
Application already paused... ignoring request to pause
Sampling traces...
Traces sampled!
…
Resuming the application...
Resumed!
Merging traces...
Traces merged!
Detaching from application...
Detached!

Results written to /global/cscratch1/sd/wyang/debugging/stat/stat_results/jacobi_mpi.0006
$ ls -l stat_results/jacobi_mpi.0006/*.dot
-rw-r--r-- 1 wyang wyang 4855 Nov  6 00:58 stat_results/jacobi_mpi.0006/00_jacobi_mpi.0006.3D.dot
$ STATview stat_results/jacobi_mpi.0006/00_jacobi_mpi.0006.3D.dot
```

with usual optimization flags, if any

-i to get source line numbers

STAT samples stack backtraces a few times

Debug a hanging application with STAT (Cont'd)



Cray ATP (Abnormal Termination Processing)

- **ATP gathers stack backtraces from all processes *when an application fails***
  - Invokes STAT underneath
  - Output in atpMergedBT.dot and atpMergedBT_line.dot (which shows source code line numbers), which are to be viewed with STATview
- **The atp module is loaded on Cori and Edison by default, but ATP is *not* enabled; to enable:**

```
export ATP_ENABLED=1    # sh/bash/ksh
setenv ATP_ENABLED 1    # csh/tcsh
```

- **For more info**
  - 'intro_atp' man page
  - https://docs.nersc.gov/development/performance-debugging-tools/stat_atp/

National Energy Research Scientific Computing Center