SDSC Webinar: Running Jobs on Comet

By: Mary Thomas

Outline

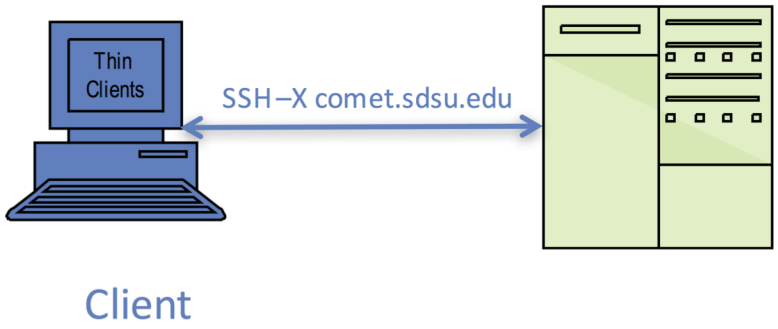
- **Preliminaries**
- **Comet Overview**
- **Getting Started/Comet System Environment**
- **Compiling and Linking Code**
- **Running Parallel Jobs**
  - Running OpenMP Jobs
  - Running MPI Jobs
  - Running Hybrid MPI-OpenMP Jobs
  - Running GPU/CUDA Jobs
- **Final Comments**

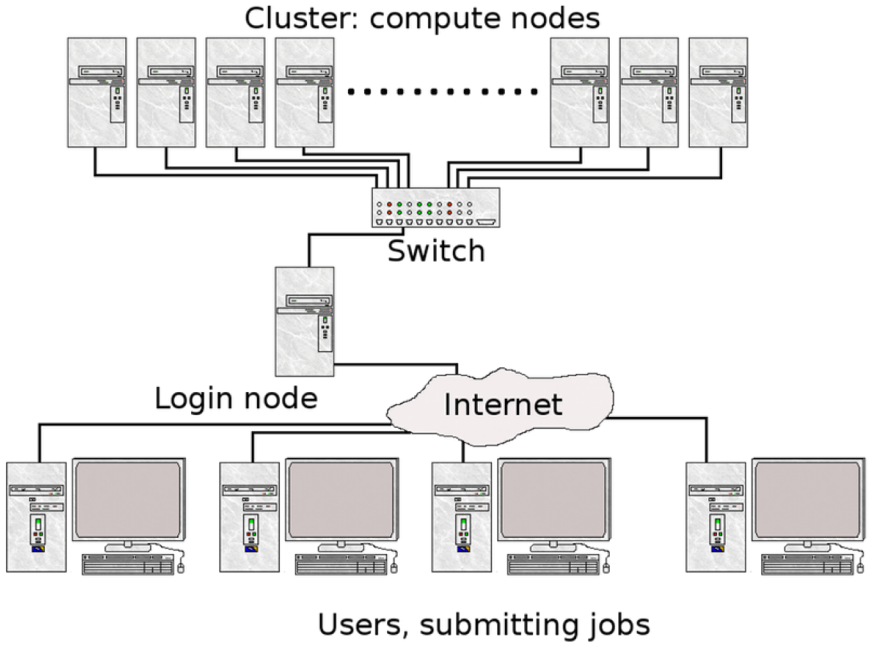
Getting Started

Basic Information

- Comet User Guide:
  - https://www.sdsc.edu/support/user_guides/comet.html
- Online repo for companion tutorial/webinar information:
  - https://github.com/sdsc-training/webinars
  - You must be familiar with running basic Unix commands: see the basic_skills and getting_started links in the webinar directory cited above.
- You must have a comet account in order to access the system. To obtain a trial account:
  - http://www.sdsc.edu/support/user_guides/comet.html#trial_accounts
- More training events listed at SDSC:
  - https://www.sdsc.edu/education_and_training/training.html

Logging On





- **System Access – Logging in**
  - Linux/Mac – Use available ssh clients.
  - ssh clients for windows – Putty, Cygwin
    - http://www.chiark.greenend.org.uk/~sgtatham/putty/
  - Login hosts for the SDSC Comet: comet.sdsc.edu

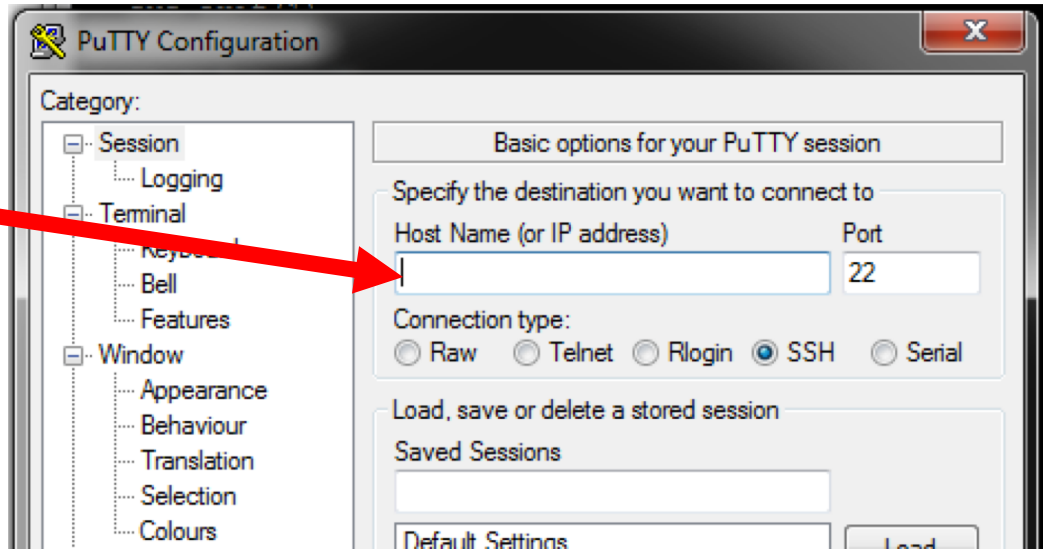
Logging into Comet

Mac/Linux:

```
ssh username@comet.sdsc.edu
```

Windows (PuTTY):

comet.sdsc.edu



Example of a terminal connection:

```
[$USER@wireless-169-228-105-171:~] ssh  comet.sdsc.edu
Warning: No xauth data; using fake authentication data for X11 forwarding.
Last login: Mon Jan  7 15:01:50 2019 from wireless-169-228-105-171.ucsd.edu
Rocks 6.2 (SideWinder)
Profile built 16:45 08-Feb-2016

Kickstarted 17:27 08-Feb-2016

                      WELCOME TO
      ______________  __  _______________
        -----/ ____/ __ \/  |/  / ____/_  __/
          --/ /   / / / / /|_/ / __/   / /
           / /___/ /_/ / /  / / /___  / /
           \____/\____/_/  /_/_____/ /_/

*******************************************************************************

[1] Example Scripts: /share/apps/examples

[2] Filesystems:

     (a) Lustre scratch filesystem : /oasis/scratch/comet/$USER/temp_project
         (Preferred: Scalable large block I/O)

     (b) Compute/GPU node local SSD storage: /scratch/$USER/$SLURM_JOBID
         (Meta-data intensive jobs, high IOPs)

     (c) Lustre projects filesystem: /oasis/projects/nsf

     (d) /home/$USER : Only for source files, libraries, binaries.
         *Do not* use for I/O intensive jobs.

[3] Comet User Guide: http://www.sdsc.edu/support/user_guides/comet.html
*******************************************************************************
[$USER@comet-ln3:~]
```

Obtaining Tutorial Example Code

- Create a test directory hold the comet example files (e.g. comet-examples)
- Copy the **PHYS244 directory** from the/share/apps/examples directory to your 'comet-examples' directory
- This tutorial will focus on examples in bold.

```
[$USER@comet-ln2 ~]$ mkdir comet-examples
[username@comet-ln3 ~]$ cp -r /share/apps/examples/PHYS244/ comet-examples/
[$USER@comet-ln3:~/comet-examples] ls -al PHYS244/
total 230
drwxr-xr-x 16 user use300 16 Aug  5 19:02 .
drwxr-xr-x  5 user use300  6 Aug  5 19:02 ..
drwxr-xr-x  2 user use300  5 Aug  5 19:02 COMPILER_EXAMPLES
drwxr-xr-x  2 user use300 14 Aug  6 00:56 CUDA
drwxr-xr-x  2 user use300 11 Aug  5 19:02 debug
drwxr-xr-x  3 user use300  3 Aug  5 19:02 HADOOP
drwxr-xr-x  2 user use300  6 Aug  6 00:12 HYBRID
drwxr-xr-x  2 user use300  6 Aug  5 19:02 LOCALSCRATCH
drwxr-xr-x  2 user use300  5 Aug  5 19:02 LOCALSCRATCH2
drwxr-xr-x  2 user use300  9 Nov 25 17:29 MKL
drwxr-xr-x  4 user use300  7 Aug  6 09:55 MPI
drwxr-xr-x  2 user use300  8 Aug  5 19:02 OpenACC
drwxr-xr-x  2 user use300  8 Aug  5 23:25 OPENMP
drwxr-xr-x  3 user use300  5 Aug  5 19:02 pytorch
drwxr-xr-x  4 user use300  4 Aug  5 19:02 SPARK
drwxr-xr-x  4 user use300  5 Aug  5 19:02 TensorFlow
```

Comet Overview

Comet
“HPC for the long tail of science”

iPhone panorama photograph of 1 of 2 server rows

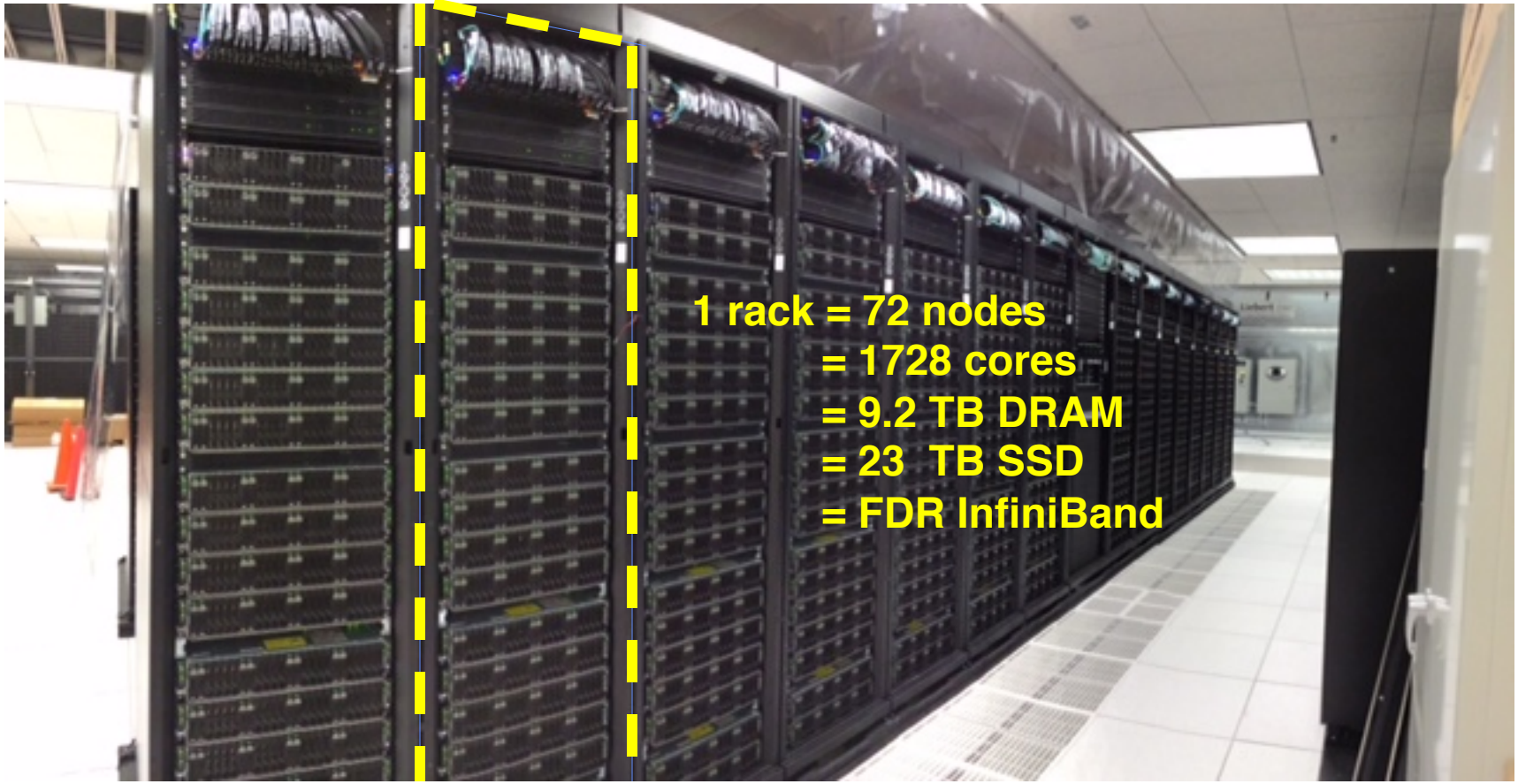
Comet: HPC for the “long tail of science:"

- “Long Tail:” majority of computational research is performed at modest scale: large number jobs that run for less than 48 hours, but can be computationally intensive and generate large amounts of data.
- Comet is an NSF-funded system available through the eXtreme Science and Engineering Discovery Environment (XSEDE) program.
- Supports science gateways.

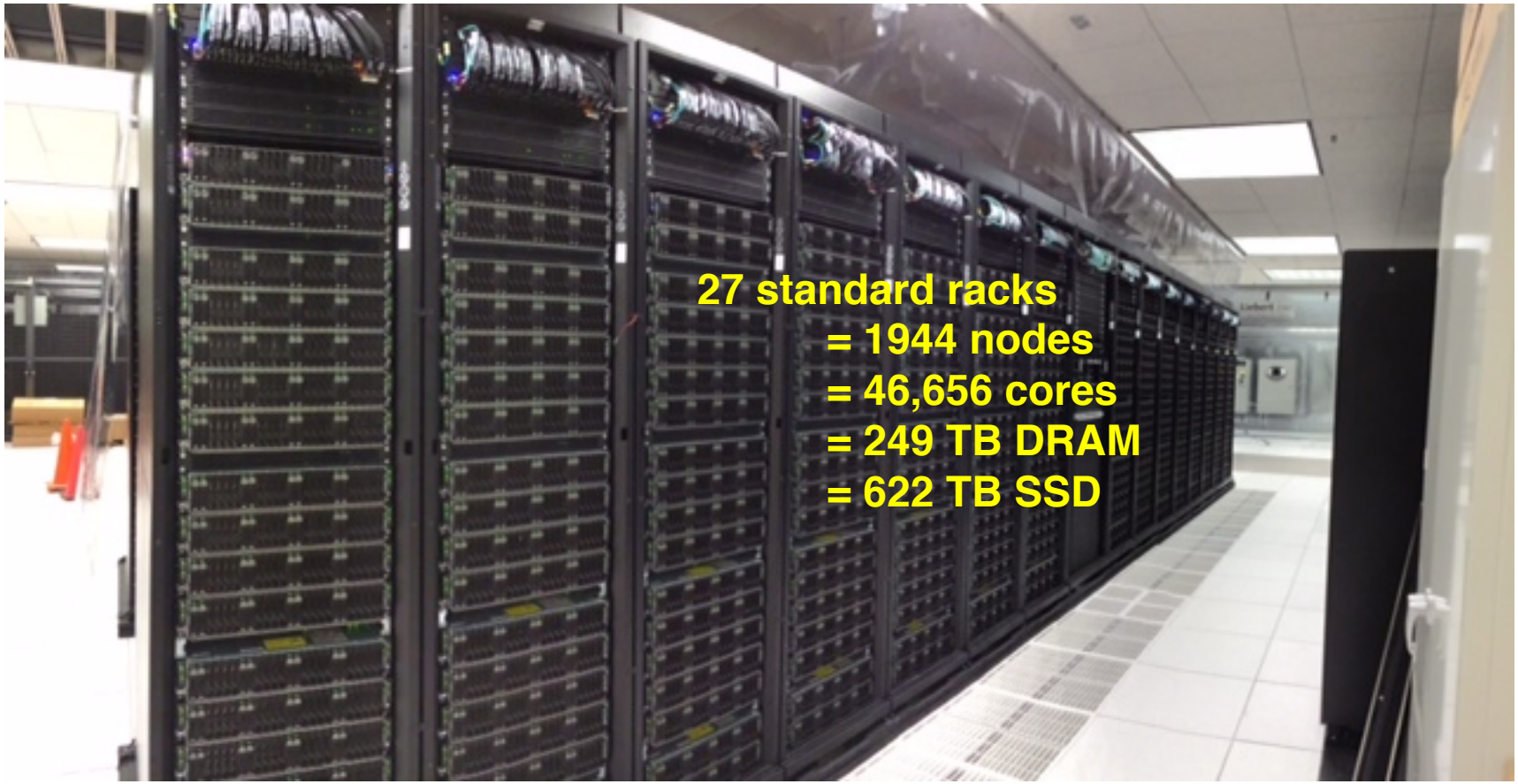
Comet: System Characteristics

- **Total peak flops ~2.6 PF**
- **Dell primary integrator**
  - Intel Haswell processors w/ AVX2
  - Mellanox FDR InfiniBand
- **1944 Standard compute nodes (46,656 cores)**
  - Dual CPUs, each 12-core, 2.5 GHz
  - 128 GB DDR4 2133 MHz DRAM
  - 2*160GB GB SSDs (local disk)
- **72 GPU nodes**
  - 36 nodes same as standard nodes *plus* Two NVIDIA K80 cards, each with dual Kepler3 GPUs
  - 36 nodes with 2 14-core Intel Broadwell CPUs plus 4 NVIDIA P100 GPUs
- **4 large-memory nodes**
  - 1.5 TB DDR4 1866 MHz DRAM
  - Four Haswell processors/node
  - 64 cores/node
- **Hybrid fat-tree topology**
  - FDR (56 Gbps) InfiniBand
  - Rack-level (72 nodes, 1,728 cores) full bisection bandwidth
  - 4:1 oversubscription cross-rack
- **Performance Storage (Aeon)**
  - 7.6 PB, 200 GB/s; Lustre
  - Scratch & Persistent Storage segments
- **Durable Storage (Aeon)**
  - 6 PB, 100 GB/s; Lustre
  - Automatic backups of critical data
- **Home directory storage**
- **Gateway hosting nodes**
- **Virtual image repository**
- **100 Gbps external connectivity to Internet2 & ESNet**

~67 TF supercomputer in a rack

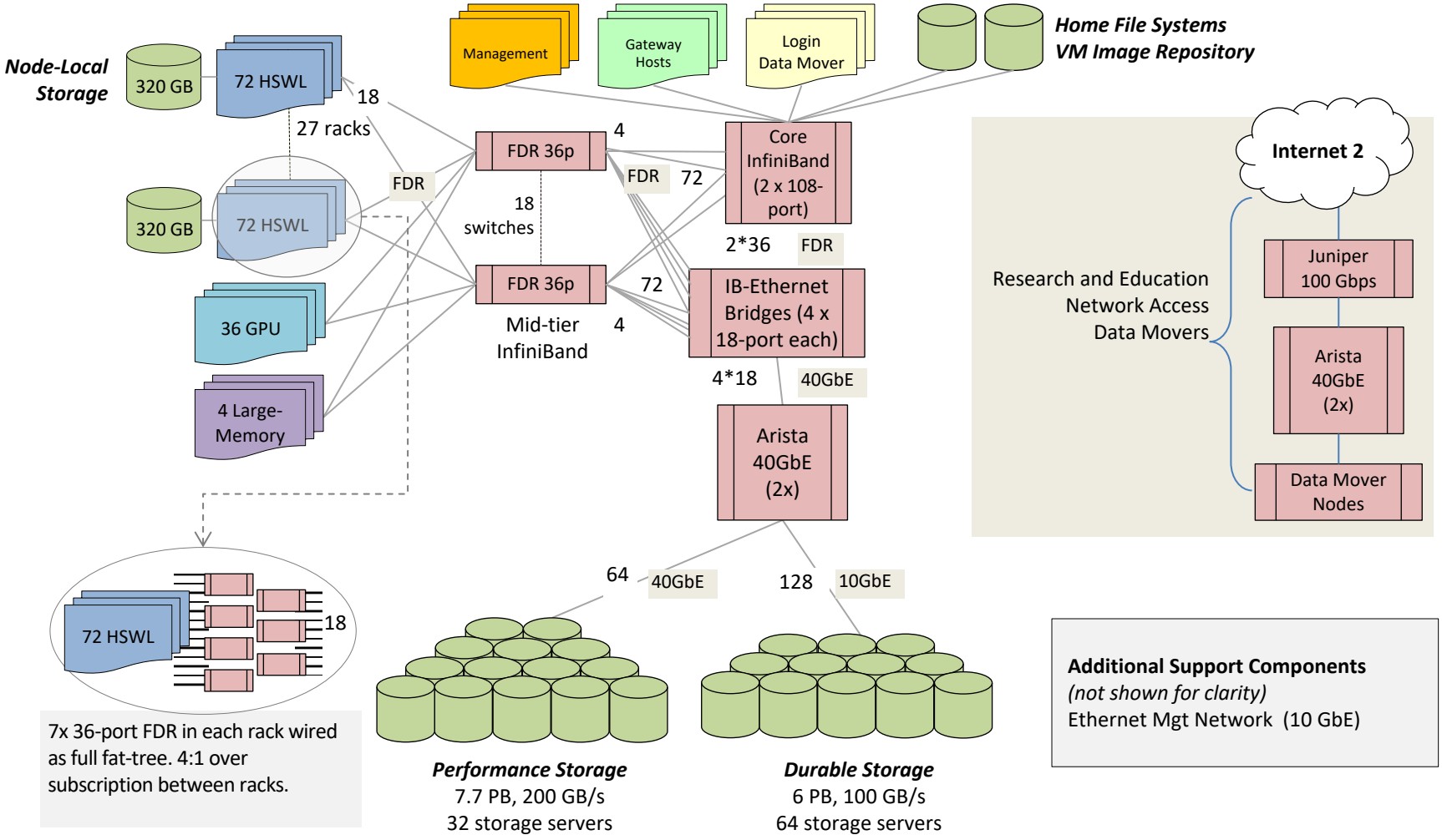


And 27 single-rack supercomputers



Comet Network Architecture

InfiniBand compute, Ethernet Storage



Comet File Systems

Path	Purpose	User Access Limits	Lifetime
$HOME	NFS storage; Source code, important files	100 GB	Backed-up
/oasis/scratch/comet/ $USER/temp_project	Global/Parallel Lustre FS; temp storage for distributed access	500 GB	No backup
/oasis/projects/nsf	Global/Parallel Lustre FS; project storage	~2.5 PB total	Backed-up
/scratch/$USER/$SL URM_JOB_ID	Local SSD on batch job node fast per-node access	210 GB per compute node, 286GB on GPU, Large memory nodes	Purged after job ends

Comet: Filesystems

- **Lustre filesystems – Good for scalable large block I/O**
  - Accessible from all compute and GPU nodes.
  - /oasis/scratch/comet - 2.5PB, peak performance: 100GB/s. Good location for storing large scale scratch data during a job.
  - /oasis/projects/nsf - 2.5PB, peak performance: 100 GB/s. Long term storage.
  - ***Not good for lots of small files or small block I/O***.
- **SSD filesystems**
  - /scratch local to each native compute node – 210GB on regular compute nodes, 285GB on GPU, large memory nodes, 1.4TB on selected compute nodes.
  - SSD location is good for writing small files and temporary scratch files. Purged at the end of a job.
- **Home directories (/home/$USER)**
  - Source trees, binaries, and small input files.
  - ***Not good for large scale I/O.***

Managing the Environment with Modules

Comet: System Environment

- **Modules used to manage environment for users.**
- **Default environment:**

```
$ module li
Currently Loaded Module files:
  1) intel/2013_sp1.2.144   2) mvapich2_ib/2.1        3) gnutools/2.69
```

- **Listing available modules:**

```
$ module av
------------------------ /opt/modulefiles/mpi/.intel ------------------------
intelmpi/2016.3.210(default) mvapich2_ib/2.1(default)
mvapich2_gdr/2.1(default)    openmpi_ib/1.8.4(default)
mvapich2_gdr/2.2
-------------------- /opt/modulefiles/applications/.intel --------------------
atlas/3.10.2(default)     lapack/3.6.0(default)     scalapack/2.0.2(default)
boost/1.55.0(default)     mxml/2.9(default)         slepc/3.6.2(default)
…
…
```

Modules: Managing the User Environment

Command	Description
module list	List the modules that are currently loaded
module avail	List the modules that are available
module display <module_name>	Show the environment variables used by and how they are affected
module show <module_name>	Same as display
module unload	Remove from the environment
module load	Load into the environment
module swap	Replace with in the environment

Module Command Examples

- Default environment: list, li

```
[$USER@comet-ln3:~/comet-examples]  module li
Currently Loaded Module files:   1) intel/2013_sp1.2.144   2) mvapich2_ib/2.1       3) gnutools/2.69
```

- List available modules: available, avail, av

```
[$USER@comet-ln3:~/comet-examples] module av

---------------------------------- /opt/modulefiles/applications/.intel ------------------------------
-----
atlas/3.10.2(default)     hdf5/1.8.14(default)      papi/5.4.1(default)       sundials/2.6.2(default)
boost/1.55.0(default)     ipm/2.0.3(default)        parmetis/4.0.3(default)   superlu/4.2(default)
fftw/2.1.5                lapack/3.6.0(default)     pdt/3.20(default)         tau/2.23(default)
fftw/3.3.4(default)       mxml/2.9(default)         petsc/3.6.3(default)
trilinos/11.12.1(default)
gsl/1.16                  netcdf/3.6.2              scalapack/2.0.2(default)
gsl/2.1(default)          netcdf/4.3.2(default)     slepc/3.6.2(default)
hdf4/2.11(default)        p3dfft/2.7.4(default)     sprng/2.0b(default)

$ module av ------------------------ /opt/modulefiles/mpi/.intel -------------------------
intelmpi/2016.3.210(default) mvapich2_ib/2.1(default)
mvapich2_gdr/2.1(default)    openmpi_ib/1.8.4(default) mvapich2_gdr/2.2 --------------------
/opt/modulefiles/applications/.intel ---------------------
atlas/3.10.2(default)     lapack/3.6.0(default)     scalapack/2.0.2(default)
boost/1.55.0(default)     mxml/2.9(default)         slepc/3.6.2(default)

… MORE….
```

Module Command Examples

- Load a module, and show what it does

```
[$USER@comet-ln3:~/comet-examples] env
HOSTNAME=comet-ln3.sdsc.edu
IPPROOT=/opt/intel/composer_xe_2013_sp1.2.144/ipp
INTEL_LICENSE_FILE=/opt/intel/composer_xe_2013_sp1.2.144/licenses:/opt/intel/licenses:/root/
intel/licenses
TERM=xterm-256color
SHELL=/bin/bash
HISTSIZE=5000
GDBSERVER_MIC=/opt/intel/composer_xe_2013_sp1.2.144/debugger/gdb/target/mic/bin/gdbserver
SSH_CLIENT=169.228.105.171 58704 22
[SNIP]
HOME=/home/user
ROLLSROOT=/opt/rocks/share/devel/src/roll
MPIHOME=/opt/mvapich2/intel/ib
FFTWHOME=/opt/fftw/3.3.4/intel/mvapich2_ib
SDSCHOME=/opt/sdsc
PYTHONPATH=/opt/sdsc/lib
LOGNAME=user
QTLIB=/usr/lib64/qt-3.3/lib
CVS_RSH=ssh
SSH_CONNECTION=169.228.105.171 58704 198.202.113.252 22
MODULESHOME=/usr/share/Modules
MKL_ROOT=/opt/intel/composer_xe_2013_sp1.2.144/mkl
LESSOPEN=||/usr/bin/lesspipe.sh %s
INFOPATH=/opt/intel/composer_xe_2013_sp1.2.144/debugger/gdb/intel64/share/info/:/opt/intel/c
omposer_xe_2013_sp1.2.144/debugger/gdb/intel64_mic/share/info/
DISPLAY=localhost:42.0
INCLUDE=/opt/intel/composer_xe_2013_sp1.2.144/mkl/include
INTELHOME=/opt/intel/composer_xe_2013_sp1.2.144
G_BROKEN_FILENAMES=1
BASH_FUNC_module()=() {  eval `/usr/bin/modulecmd bash $*`
}
_=/bin/env
```

Module: check Environment

- Once you have loaded the modules, you can check the system variables that are available for you to use.
-

```
[$USER@comet-ln3:~/comet-examples] module load fftw/3.3.4
[$USER@comet-ln3:~/comet-examples]
[$USER@comet-ln3:~/comet-examples] module li
Currently Loaded Modulefiles:
  1) intel/2013_sp1.2.144   2) mvapich2_ib/2.1        3) gnutools/2.69          4)
fftw/3.3.4
[$USER@comet-ln3:~/comet-examples] module show fftw/3.3.4
-------------------------------------------------------------------
/opt/modulefiles/applications/.intel/fftw/3.3.4:

module-whatis            fftw
module-whatis            Version: 3.3.4
module-whatis            Description: fftw
module-whatis            Compiler: intel
module-whatis            MPI Flavors: mvapich2_ib openmpi_ib
setenv                   FFTWHOME /opt/fftw/3.3.4/intel/mvapich2_ib
prepend-path             PATH /opt/fftw/3.3.4/intel/mvapich2_ib/bin
prepend-path             LD_LIBRARY_PATH /opt/fftw/3.3.4/intel/mvapich2_ib/lib
prepend-path             LIBPATH /opt/fftw/3.3.4/intel/mvapich2_ib/lib
-------------------------------------------------------------------
```

Suggestion:
Create module/env loading scripts

```
[comet-ln2:~] cat loadintelenv.sh
# Using the Intel Compilers (Default/Suggested)
module purge
module load gnutools
module load intel mvapich2_ib
```

Using module/env loading scripts

Control and guarantee the current working environment. In order for the commands run inside a script (child shell) to change the parent shell, you must use the *source* command.

```
[comet-ln3:~] source ./loadgpuenv.sh
[comet-ln3:~] module list
Currently Loaded Modulefiles:
  1) gnutools/2.69   2) cuda/7.0
[comet-ln3:~] which nvcc
/usr/local/cuda-7.0/bin/nvcc
[mthomas@comet-ln3:~] which mpirun
/usr/bin/which: no mpirun in (/opt/gnu/gcc/bin:usr/local/bin……..)

[comet-ln3:~] source loadintelenv.sh
[comet-ln3:~] module list
Currently Loaded Modulefiles:
  1) gnutools/2.69  2) intel/2013_sp1.2.144 3) mvapich2_ib/2.1
[mthomas@comet-ln3:~] which mpirun
/opt/mvapich2/intel/ib/bin/mpirun
[mthomas@comet-ln3:~] which nvcc
/usr/bin/which: no nvcc in (/opt/gnu/gcc/bin:usr/local/bin……..)
```

```
comet-ln3:~] cat loadgpuenv.sh
#!/bin/bash
module purge
module load gnutools
module load cuda
```

```
[comet-ln3:~] cat loadintelenv.sh
module purge
module load gnutools
module load intel mvapich2_ib
```

Compiling & Linking

Compiling & Linking: Topics

- Supported Compiler Types
- Intel Compiling
- PGI Compiling
- GNU Compiling

Supported Compiler Types

- **Comet compute nodes support several parallel programming models:**
  - **MPI**: Default Intel Compiler**: intel/2013_sp1.2.144**;
    - Versions 2015.2.164 and 2016.3.210 available.
    - Other options: openmpi_ib/1.8.4 (and 1.10.2), Intel MPI, mvapich2_ib/2.1
    - mvapich2_gdr: GPU direct enabled version
  - **OpenMP & Pthreads**:
    - All compilers (GNU, Intel, PGI) have OpenMP flags.
  - **GPU** nodes: support CUDA, OpenACC.
  - **Hybrid** modes are possible (see examples below).

Suggested Compilers

- Default/Suggested Compilers to used based on programming model and languages:

	Serial	MPI	OpenMP	MPI + OpenMP
Fortran	ifort	mpif90	ifort -openmp	mpif90 -openmp
C	icc	mpicc	icc -openmp	mpicc -openmp
C++	icpc	mpicxx	icpc -openmp	mpicxx -openmp

- In this tutorial, we include hands-on examples that cover many of the cases in the table:
  (1) MPI
  (2) OpenMP
  (3) HYBRID
  (4) Local scratch

Using the Intel Compilers

- The Intel compilers and the MVAPICH2 MPI implementation will be loaded by default.
- If you have modified your environment, you can reload by executing the module purge & load commands at the Linux prompt, or placing the load command in your startup file (`~/.cshrc or ~/.bashrc`)

```
[$USER@comet-ln2:~]
[$USER@comet-ln2:~] module purge
[$USER@comet-ln2:~] module list
No Modulefiles Currently Loaded.
[$USER@comet-ln2:~] module load gnutools
[$USER@comet-ln2:~] module list
Currently Loaded Modulefiles:
  1) gnutools/2.69
[$USER@comet-ln2:~] module load intel mvapich2_ib
[$USER@comet-ln2:~] module list
Currently Loaded Modulefiles:
  1) gnutools/2.69          2) intel/2013_sp1.2.144   3) mvapich2_ib/2.1

[$USER@comet-ln2:~/comet-examples/PHYS244/MKL] which mpicc
/opt/mvapich2/intel/ib/bin/mpicc
```

Using the Intel Compilers

- For Intel Advanced Vector Extensions (AVX2) support, compile with the -xH0ST option.
  - Note that -xHOST alone does not enable aggressive optimization, so compilation with -O3 is also suggested.
  - The -fast flag invokes -xHOST, but should be avoided since it also turns on interprocedural optimization (-ipo), which may cause problems in some instances.
- Intel Math Kernal Lib (MKL) libraries are available as part of the "intel" modules on Comet.
  - Once this module is loaded, the environment variable MKL_ROOT points to the location of the mkl libraries.
  - The MKL link advisor can be used to ascertain the link line (change the MKL_ROOT aspect appropriately).

Using the Intel Compilers

- In the example below, we are working with the HPC examples that can be found in PHYS244/MKL :

```
[$USER@comet-14-01:~/comet-examples/PHYS244/MKL] pwd
/home/user/comet-examples/PHYS244/MKL
[$USER@comet-14-01:~/comet-examples/PHYS244/MKL] ls -al
total 25991
drwxr-xr-x  2 user use300        9 Nov 25 17:20 .
drwxr-xr-x 16 user use300       16 Aug  5 19:02 ..
-rw-r--r--  1 user use300      325 Aug  5 19:02 compile.txt
-rw-r--r--  1 user use300     6380 Aug  5 19:02 pdpttr.c
-rwxr-xr-x  1 user use300 44825440 Nov 25 16:55 pdpttr.exe
-rw-r--r--  1 user use300      188 Nov 25 16:57 scalapack.20294236.comet-07-27.out
-rw-r--r--  1 user use300      376 Aug  5 19:02 scalapack.sb
```

Using the Intel Compilers

The file `compile.txt` contains the full command to compile the `pdpttr.c` program statically linking 64 bit scalapack libraries on Comet:

```
[$USER@comet-14-01:~/comet-examples/PHYS244/MKL] cat compile.txt
mpicc -o pdpttr.exe pdpttr.c  -I$MKL_ROOT/include
${MKL_ROOT}/lib/intel64/libmkl_scalapack_lp64.a -Wl,--start-group
${MKL_ROOT}/lib/intel64/libmkl_intel_lp64.a
${MKL_ROOT}/lib/intel64/libmkl_core.a
${MKL_ROOT}/lib/intel64/libmkl_sequential.a -Wl,--end-group
${MKL_ROOT}/lib/intel64/libmkl_blacs_intelmpi_lp64.a -lpthread -lm```
```

compile the command:

```
[$USER@comet-14-01:~/comet-examples/PHYS244/MKL] mpicc -o pdpttr.exe pdpttr.c  -I$MKL_ROOT/include
${MKL_ROOT}/lib/intel64/libmkl_scalapack_lp64.a -Wl,--start-group
${MKL_ROOT}/lib/intel64/libmkl_intel_lp64.a ${MKL_ROOT}/lib/intel64/libmkl_core.a
${MKL_ROOT}/lib/intel64/libmkl_sequential.a -Wl,--end-group
${MKL_ROOT}/lib/intel64/libmkl_blacs_intelmpi_lp64.a -lpthread -lm
```

For more information on the Intel compilers run: [ifort | icc | icpc] -help

Using the PGI Compilers

- PGI (formerly The Portland Group, Inc.), was a company that produced a set of commercially available Fortran, C and C++ compilers for high-performance computing systems.
- It is now owned by NVIDIA.
- PGI compilers can be loaded by executing the following commands at the Linux prompt or placing in your startup file (~/.cshrc or ~/.bashrc).
- For AVX support, compile with -fast

```
[$USER@comet-ln2:~/comet-examples/PHYS244/MKL] module purge
[$USER@comet-ln2:~/comet-examples/PHYS244/MKL] module load gnutools
[$USER@comet-ln2:~/comet-examples/PHYS244/MKL] module load pgi
mvapich2_ib
[$USER@comet-ln2:~/comet-examples/PHYS244/MKL] module list
Currently Loaded Modulefiles:
  1) gnutools/2.69     2) pgi/17.5          3) mvapich2_ib/2.1

[$USER@comet-ln2:~/comet-examples/PHYS244/MKL] which mpicc
/opt/mvapich2/pgi/ib/bin/mpicc
```

- For more information on the PGI compilers run: man [pgf90 | pgcc | pgCC]

Recommended PGI Compilers

	Serial	MPI	OpenMP	MPI+OpenMP
Fortran	pgf90	mpif90	pgf90 -mp	mpif90 -mp
C	pgcc	mpicc	pgcc -mp	mpicc -mp
C++	pgCC	mpicxx	pgCC -mp	mpicxx -mp

- PGI supports the following high-level languages:
  - Fortran 77, 90/95/2003, 2008 (partial)
  - High Performance Fortran (HPF)
  - ANSI C99 with K&R extensions
  - ANSI/ISO C++
  - CUDA Fortran
  - OpenCL
  - OpenACC
  - OpenMP

Using the GNU Compilers

- The GNU compilers can be loaded by executing the following commands at the Linux prompt or placing in your startup files (~/.cshrc or ~/.bashrc)

```
[$USER@comet-ln2:~/comet-examples/PHYS244/MKL] module purge
Unloading compiler-dependent module gnutools/2.69
[$USER@comet-ln2:~/comet-examples/PHYS244/MKL] module load gnutools
[$USER@comet-ln2:~/comet-examples/PHYS244/MKL] module load gnu openmpi_ib
[$USER@comet-ln2:~/comet-examples/PHYS244/MKL]
[$USER@comet-ln2:~/comet-examples/PHYS244/MKL] which mpicc
/opt/openmpi/gnu/ib/bin/mpicc
[$USER@comet-ln2:~/comet-examples/PHYS244/MKL]
```

- For AVX support, compile with -mavx.
- Note that AVX support is only available in version 4.7 or later, so it is necessary to explicitly load the gnu/4.9.2 module until such time that it becomes the default.

Using the GNU Compilers

Table of recommended GNU compilers:

	Serial	MPI	OpenMP	MPI+OpenMP
Fortran	gfortran	mpif90	gfortran -fopenmp	mpif90 -fopenmp
C	gcc	mpicc	gcc -fopenmp	mpicc -fopenmp
C++	g++	mpicxx	g++ -fopenmp	mpicxx -fopenmp

Running Jobs On Comet

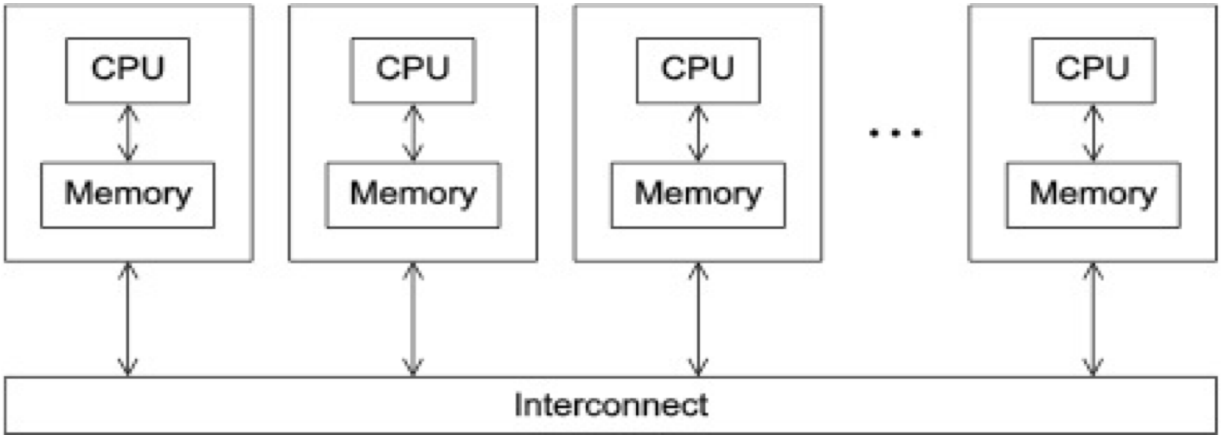
Factors Impacting Job Execution

- **Parallel Models:**
  - Impacts language used, libraries, performance.
- **How you choose to run the job:**
  - Command line execution
  - Batch/queuing System -- Comet uses the Simple Linux Utility for Resource Management (SLURM):
    - Batch Queue
    - Interactive jobs
- **Data I/O choices (topic of upcoming Webinar):**
  - https://www.sdsc.edu/education_and_training/training.html

Parallel Models: Memory

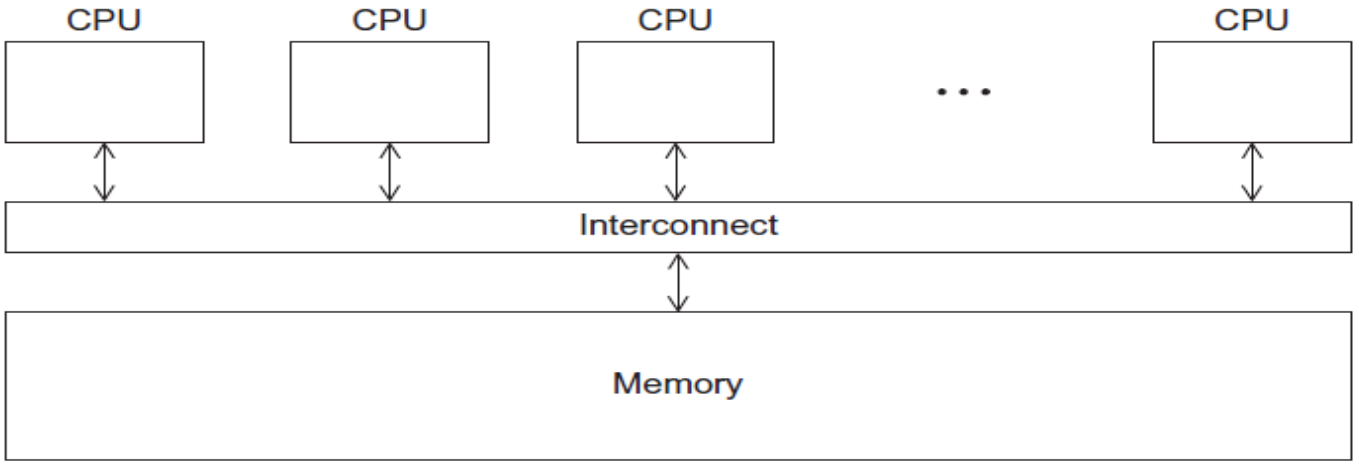
- **Distributed Memory**
- **Shared Memory**
- **Implemented in several languages:**
  - FORTRAN, C, Python, OOPs (sort-of)
- **Large number of libraries and API's**
- **Adds to compilation/linking complexity**

Distributed Memory



- Programs that run asynchronously, pass messages for communication and coordination between resources.
- Examples include: SOA-based systems, massively multiplayer online games, peer-to-peer apps.
- Different types of implementations for the message passing mechanism: HTTP, RPC-like connectors, message queues
- HPC historically uses the **Message Passing Interface (MPI)**

Parallel Models: Shared Memory



- **CPUs all share same localized memory (SHMEM);**
  - Coordination and communication between tasks via interprocessor communication (IPC) or virtual memory mappings.
- **May use: uniform or non-uniform memory access (UMA or NUMA); cache-only memory architecture (COMA).**
- **Most common HPC API's for using SHMEM:**
  - Portable Operating System Interface (POSIX); Open Multi-Processing (OpenMP) designed for parallel computing – best for multi-core computing.

Running Jobs on Comet

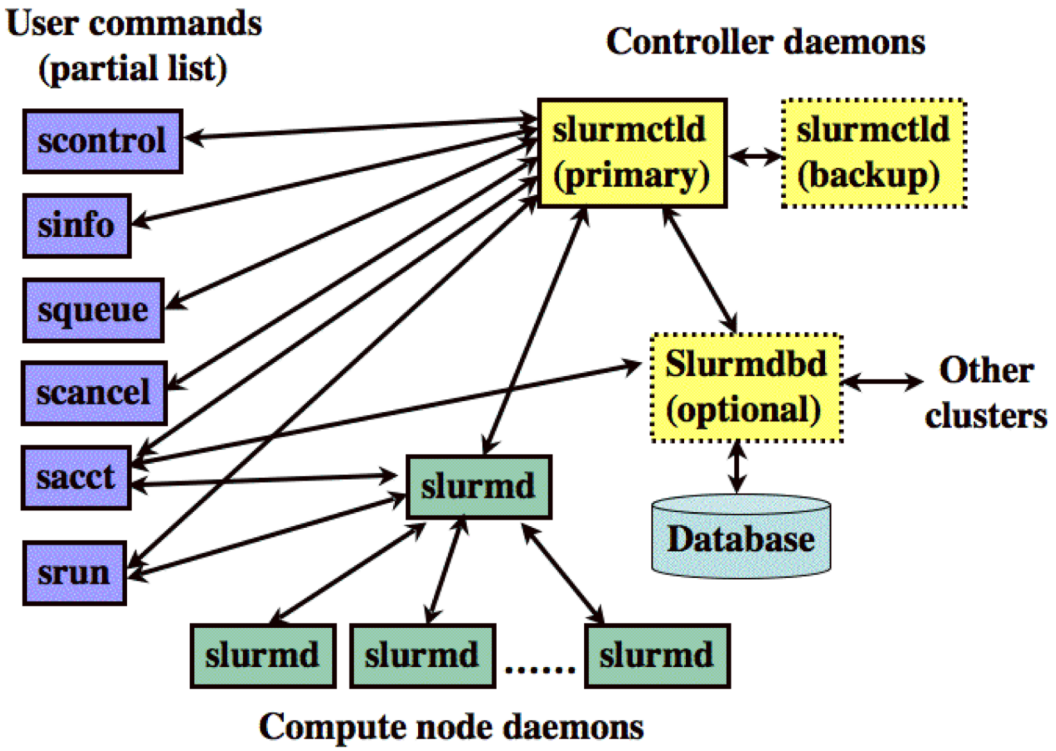
- **Important note: Do not run on the login nodes - even for simple tests.**
- **All job runs must be via the Slurm scheduling infrastructure.**
  - **Interactive Jobs**: Use **srun** command to obtain nodes for 'live' interactive access:

    srun --pty --nodes=1 --ntasks-per-node=24 -p debug -t 00:30:00 --wait 0 /bin/bash
  - **Batch Jobs**: Submit batch scripts from the login nodes. Can choose:
    - Partition (details on upcoming slide)
    - Time limit for the run (maximum of 48 hours)
    - Number of nodes, tasks per node
    - Memory requirements (if any)
    - Job name, output file location
    - Email info, configuration

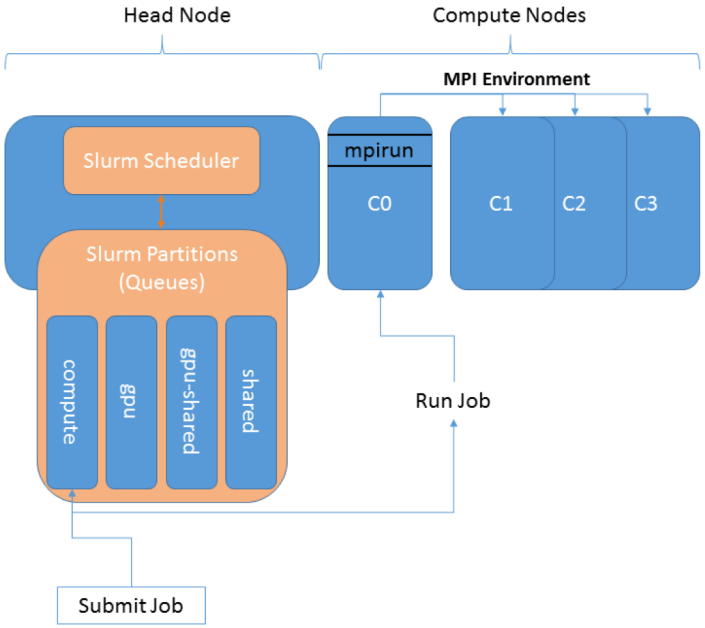
Slurm Resource Manager

Simple Linux Utility for Resource Management

- "Glue" for parallel computer to schedule and execute jobs
- Role: Allocate resources within a cluster
  - Nodes (unique IP address)
  - Interconnect/switces
  - Generic resources (e.g. GPUs)
  - Launch and otherwise manage jobs
- Functionality:
  - Prioritize queue(s) of jobs;
  - decide when and where to start jobs;
  - terminate job when done;
  - Appropriate resources;
  - manage accounts for jobs



Slurm Partitions on Comet



Specified using -p option in b
script. For example:
#SBATCH -p gpu

Queue Name	Max Walltime	Max Nodes	Comments
compute	48 hrs	72	Used for access to regular compute nodes
gpu	48 hrs	4	Used for access to the GPU nodes
gpu-shared	48 hrs	1	Used for shared access to a partial GPU node
shared	48 hrs	1	Single-node jobs using fewer than 24 cores
large-shared	48 hrs	1	Single-node jobs using large memory up to 1.45 TB
debug	30 mins	2	Used for access to debug nodes

Common Slurm Commands

- **Submit jobs using the sbatch command:**

  $ sbatch mycode-slurm.sb

  Submitted batch job 8718049

- **Check job status using the squeue command:**

  $ squeue -u $USER

```
             JOBID PARTITION     NAME     USER      ST       TIME  NODES  NODELIST(REASON)
           8718049   compute      mycode user     PD       0:00      1              (Priority)
```

- **Once the job is running:**

  $ squeue -u $USER

```
             JOBID PARTITION     NAME     USER    ST       TIME  NODES  NODELIST(REASON)
           8718064     debug      mycode user     R       0:02      1         comet-14-01
```

Hands-on Examples

General Steps: Compiling/Running Jobs

- **Change to working directory**
  cd /home/$USER/comet-examples/MPI

- **Verify modules loaded:**
  module list
  Currently Loaded Modulefiles:
  1) intel/2013_sp1.2.144 2) mvapich2_ib/2.1 3) gnutools/2.69

- **Compile the MPI hello world code:**
  mpif90 -o hello_mpi hello_mpi.f90

- **Verify executable has been created (check that date):**
  ls -lt hello_mpi
  -rwxr-xr-x 1 user sdsc 721912 Mar 25 14:53 **hello_mpi**

- **Submit job from IBRUN directory (not required but helps with organization):**
  cd /home/$USER/comet-examples/MPI/IBRUN
  sbatch --res=comet-examplesDAY1 hellompi-slurm.sb

Hands On Examples

- **Examples for :**
  - MPI
  - OpenMP
  - HYBRID
  - Local scratch

- **Running on Comet Compute Nodes**
  - 2-Socket (Total 24 cores)
  - Intel Haswell Processors

Getting Set up

- Create a test directory (`e.g. comet-examples`)
- Copy the `/shared/apps/PHYS244` codebase to your test directory.
- Change to the test examples directory:

```
[comet-ln2:~] mkdir comet-examples
[comet-ln2:~/comet-examples/PHYS244] cd MPI
[comet-ln2:~/comet-examples/PHYS244/MPI] ll
total 872
drwxr-xr-x  4 user use300      7 Aug  6 09:55 .
drwxr-xr-x 16 user use300     16 Aug  5 19:02 ..
-rwxr-xr-x  1 user use300 721944 Aug  6 09:55 hello_mpi
-rwxr-xr-x  1 user use300 721912 Aug  5 19:11 hello_mpi.bak
-rw-r--r--  1 user use300    357 Aug  5 19:22 hello_mpi.f90
drwxr-xr-x  2 user use300      6 Aug  6 10:04 IBRUN
drwxr-xr-x  2 user use300      3 Aug  5 19:02 MPIRUN_RSH
[comet-ln2:~/comet-examples/PHYS244/MPI] cat hello_mpi.f90
!  Fortran example
   program hello
   include 'mpif.h'
   integer rank, size, ierror, tag, status(MPI_STATUS_SIZE)

   call MPI_INIT(ierror)
   call MPI_COMM_SIZE(MPI_COMM_WORLD, size, ierror)
   call MPI_COMM_RANK(MPI_COMM_WORLD, rank, ierror)
   print*, 'node', rank, ': Hello and Welcome to Webinar Participants!'
   call MPI_FINALIZE(ierror)
   end
```

Running MPI Jobs

MPI Hello World

- Change to the MPI examples directory:

```
[comet-ln2:~/comet-examples/PHYS244] cd MPI
[comet-ln2:~/comet-examples/PHYS244/MPI] ll
total 872
drwxr-xr-x  4 user use300      7 Aug  6 09:55 .
drwxr-xr-x 16 user use300     16 Aug  5 19:02 ..
-rwxr-xr-x  1 user use300 721944 Aug  6 09:55 hello_mpi
-rwxr-xr-x  1 user use300 721912 Aug  5 19:11 hello_mpi.bak
-rw-r--r--  1 user use300    357 Aug  5 19:22 hello_mpi.f90
drwxr-xr-x  2 user use300      6 Aug  6 10:04 IBRUN
drwxr-xr-x  2 user use300      3 Aug  5 19:02 MPIRUN_RSH
[comet-ln2:~/comet-examples/PHYS244/MPI] cat hello_mpi.f90
!  Fortran example
   program hello
   include 'mpif.h'
   integer rank, size, ierror, tag, status(MPI_STATUS_SIZE)

   call MPI_INIT(ierror)
   call MPI_COMM_SIZE(MPI_COMM_WORLD, size, ierror)
   call MPI_COMM_RANK(MPI_COMM_WORLD, rank, ierror)
   print*, 'node', rank, ': Hello and Welcome to Webinar Participants!'
   call MPI_FINALIZE(ierror)
   end
```

MPI Hello World: Compile

Set the environment and then compile the code

```
[comet-ln2:~/comet-examples/PHYS244/MPI] module purge
[comet-ln2:~/comet-examples/PHYS244/MPI] module load gnutools
[comet-ln2:~/comet-examples/PHYS244/MPI] module load intel mvapich2_ib
[comet-ln2:~/comet-examples/PHYS244/MPI] module list
Currently Loaded Modulefiles:
  1) gnutools/2.69          2) intel/2013_sp1.2.144   3) mvapich2_ib/2.1

[comet-ln2:~/comet-examples/PHYS244/MPI] which mpif90
/opt/mvapich2/intel/ib/bin/mpif90

[comet-ln2:~/comet-examples/PHYS244/MPI] mpif90 -o hello_mpi hello_mpi.f90
[comet-ln2:~/comet-examples/PHYS244/MPI]
```

Try to run from command line: it works, but it is not recommended.

```
[comet-ln2:~/comet-examples/PHYS244/MPI] mpirun -np 4 ./hello_mpi
 node           0 : Hello and Welcome Webinar Participants!
 node           1 : Hello and Welcome Webinar Participants!
 node           2 : Hello and Welcome Webinar Participants!
 node           3 : Hello and Welcome Webinar Participants!
```

Using Interactive mode

Move to the IBRUN directory, and request nodes:

```
[comet-ln2:~/comet-examples/PHYS244/MPI/IBRUN] date
Tue Jan  8 00:22:42 PST 2019
[comet-ln2:~] hostname
comet-ln2.sdsc.edu
[comet-ln2:~/comet-examples/PHYS244/MPI/IBRUN] srun --pty --nodes=1 --ntasks-per-node=24 -p debug -t
00:30:00 --wait 0 /bin/bash
srun: job 20912306 queued and waiting for resources
srun: job 20912306 has been allocated resources
[comet-14-01:~/comet-examples/PHYS244/MPI/IBRUN] hostname
comet-14-01.sdsc.edu
[comet-14-01:~/comet-examples/PHYS244/MPI/IBRUN] mpirun -np 4 ../hello_mpi
 node         0 : Hello and Welcome Webinar Participants!
 node         1 : Hello and Welcome Webinar Participants!
 node         2 : Hello and Welcome Webinar Participants!
 node         3 : Hello and Welcome Webinar Participants!
[comet-14-01:~/comet-examples/PHYS244/MPI/IBRUN] exit
exit
[comet-ln2:~/comet-examples/PHYS244/MPI/IBRUN]
```

- Exit interactive session when work is done or you will be charged CPU time.
- Beware of oversubscribing your job: asking for more cores than you have. Intel compiler allows this, but your performance will be degraded.

MPI Hello World: Batch Script

Move to the IBRUN directory, where the SLURM batch script is located:

```
[comet-ln2:~/comet-examples/PHYS244/MPI] cd IBRUN/
[comet-ln2:~/comet-examples/PHYS244/MPI/IBRUN] cat hellompi-slurm.sb
#!/bin/bash
#SBATCH --job-name="hellompi"
#SBATCH --output="hellompi.%j.%N.out"
#SBATCH --partition=compute
#SBATCH --nodes=2
#SBATCH --ntasks-per-node=24
#SBATCH --export=ALL
#SBATCH -t 01:30:00

#This job runs with 2 nodes, 24 cores per node for a total of 48 cores.
#ibrun in verbose mode will give binding detail

ibrun -v ../hello_mpi
```

MPI Hello World: submit job & monitor

- To run the job, use the batch script submission command.
- Monitor the job until it is finished using the squeue command.

```
[comet-ln3:~/comet-examples/PHYS244/MPI/IBRUN] sbatch hellompi-slurm.sb
Submitted batch job 20918244
[comet-ln3:~/comet-examples/PHYS244/MPI/IBRUN] squeue -u user
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
          20918244   compute hellompi  user PD       0:00      2 (None)
[comet-ln3:~/comet-examples/PHYS244/MPI/IBRUN] squeue -u user JOBID PARTITION     NAME
USER ST       TIME  NODES NODELIST(REASON)
          20918244   compute hellompi  user  R       0:01      2 comet-11-[01,58]
[comet-ln3:~/comet-examples/PHYS244/MPI/IBRUN] squeue -u user
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
          20918244   compute hellompi  user CG       0:02      1 comet-11-01
[comet-ln3:~/comet-examples/PHYS244/MPI/IBRUN] ll
total 67
drwxr-xr-x 2 user use300    5 Jan  8 13:25 .
drwxr-xr-x 4 user use300    8 Jan  8 13:12 ..
-rw-r--r-- 1 user use300 9218 Jan  8 13:25 hellompi.20918244.comet-11-01.out
-rw-r--r-- 1 user use300  342 Aug  5 19:34 hellompi-slurm.sb
```

MPI Hello World: Output

Monitor the job until it is finished

```
[comet-ln2:~/comet-examples/PHYS244/MPI/IBRUN] cat hellompi.20912353.comet-20-06.out
IBRUN: Command is ../hello_mpi
IBRUN: Command is /home/user/comet-examples/PHYS244/MPI/hello_mpi
IBRUN: no hostfile mod needed
IBRUN: Nodefile is /tmp/AaTm2VFWKx
IBRUN: MPI binding policy: compact/core for 1 threads per rank (12 cores per socket)
IBRUN: Adding MV2_USE_OLD_BCAST=1 to the environment
IBRUN: Adding MV2_CPU_BINDING_LEVEL=core to the environment
IBRUN: Adding MV2_ENABLE_AFFINITY=1 to the environment
IBRUN: Adding MV2_DEFAULT_TIME_OUT=23 to the environment
IBRUN: Adding MV2_CPU_BINDING_POLICY=bunch to the environment
IBRUN: Adding MV2_USE_HUGEPAGES=0 to the environment
IBRUN: Adding MV2_HOMOGENEOUS_CLUSTER=0 to the environment
IBRUN: Adding MV2_USE_UD_HYBRID=0 to the environment
IBRUN: Added 8 new environment variables to the execution environment
IBRUN: Command string is [mpirun_rsh -np 48 -hostfile /tmp/AaTm2VFWKx -export-all
/home/user/comet-examples/PHYS244/MPI/hello_mpi]
 node           15 : Hello and Welcome Webinar Participants!
 node           16 : Hello and Welcome Webinar Participants!
 node           19 : Hello and Welcome Webinar Participants!
 node            9 : Hello and Welcome Webinar Participants!
……..
node           25 : Hello and Welcome Webinar Participants!
 node           30 : Hello and Welcome Webinar Participants!
 node           29 : Hello and Welcome Webinar Participants!
 node           33 : Hello and Welcome Webinar Participants!
 node           31 : Hello and Welcome Webinar Participants!
IBRUN: Job ended with value 0
```

Running OpenMP Jobs

OpenMP Hello World

Change to the OPENMP examples directory:

```
[comet-ln2:~/comet-examples/PHYS244] cd OPENMP
[comet-ln2:~/comet-examples/PHYS244/OPENMP] ls -al
total 498
drwxr-xr-x  2 user use300      8 Aug  5 23:25 .
drwxr-xr-x 16 user use300     16 Aug  5 19:02 ..
-rw-r--r--  1 user use300    267 Aug  5 22:19 hello_openmp.f90
-rw-r--r--  1 user use300    311 Aug  5 23:25 openmp-slurm.sb
-rw-r--r--  1 user use300    347 Aug  5 19:02 openmp-slurm-shared.sb

[comet-ln2:~/comet-examples/PHYS244/OPENMP] cat hello_openmp.f90
      PROGRAM OMPHELLO
      INTEGER TNUMBER
      INTEGER OMP_GET_THREAD_NUM

!$OMP PARALLEL DEFAULT(PRIVATE)
      TNUMBER = OMP_GET_THREAD_NUM()
      PRINT *, 'Hello from Thread Number[',TNUMBER,'] and Welcome Webinar!'
!$OMP END PARALLEL

      STOP
      END
```

MPI Hello World: Compile

Check the environment and then compile the code

```
[comet-ln2:~/comet-examples/PHYS244/OPENMP] module list
Currently Loaded Modulefiles:
  1) gnutools/2.69          2) intel/2013_sp1.2.144   3) mvapich2_ib/2.1
[comet-ln2:~/comet-examples/PHYS244/OPENMP] ifort -o hello_openmp -openmp hello_openmp.f90
```

Compile using the ifort command

```
[comet-ln2:~/comet-examples/PHYS244/OPENMP] ifort -o hello_openmp -openmp hello_openmp.f90
[comet-ln2:~/comet-examples/PHYS244/OPENMP]
```

OpenMP Hello World: Controlling #Threads

A key issue when running OpenMP code is controlling thread behavior.
If you run from command line, it will work, but it is not recommended because you will be using Pthreads, which automatically picks the number of threads - in this case 24.

```
[comet-ln2:~/comet-examples/PHYS244/OPENMP]./hello_openmp
 Hello from Thread Number[           0 ] and Welcome Webinar!
 Hello from Thread Number[           2 ] and Welcome Webinar!
  .   .   .   .
 Hello from Thread Number[          22 ] and Welcome Webinar!
 Hello from Thread Number[          11 ] and Welcome Webinar!
 Hello from Thread Number[          23 ] and Welcome Webinar!
```

To control thread behavior, there are several key environment variables:
OMP_NUM_THREADS controls the number of threads allowed, and OMP_PROC_BIND binds threads to “places” (e.g. cores) and keeps them from moving around (between cores).

```
[comet-ln2:~/comet-examples/PHYS244/OPENMP] export OMP_NUM_THREADS=4; ./hello_openmp
HELLO FROM THREAD NUMBER = 3
HELLO FROM THREAD NUMBER = 1
HELLO FROM THREAD NUMBER = 2
HELLO FROM THREAD NUMBER = 0
```

See: https://www.ibm.com/support/knowledgecenter/SSGH2K_13.1.3/com.ibm.xlc1313.aix.doc/compiler_ref/ruomprun.html

OpenMP Hello World: Batch Script

```
[comet-ln2:~/comet-examples/PHYS244/OPENMP] cat openmp-slurm.sb
#!/bin/bash
#SBATCH --job-name="hello_openmp"
#SBATCH --output="hello_openmp.%j.%N.out"
#SBATCH --partition=compute
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=24
#SBATCH --export=ALL
#SBATCH -t 01:30:00

#SET the number of openmp threads
export OMP_NUM_THREADS=24

#Run the job using mpirun_rsh
./hello_openmp
```

- Comet supports shared-node jobs (more than one job on a single node).
- Many applications are serial or can only scale to a few cores.
- Shared nodes improve job throughput, provide higher overall system utilization, and allow more users to run on jobs.

```
[comet-ln2:~/comet-examples/PHYS244/OPENMP] cat openmp-slurm-shared.sb
#!/bin/bash
#SBATCH --job-name="hell_openmp_shared"
#SBATCH --output="hello_openmp_shared.%j.%N.out"
#SBATCH --partition=shared
#SBATCH --share
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=16
#SBATCH --mem=80G
#SBATCH --export=ALL
#SBATCH -t 01:30:00

#SET the number of openmp threads
export OMP_NUM_THREADS=16

#Run the openmp job
./hello_openmp
```

OpenMP Hello World: submit job & monitor

To run the job, type the batch script submission command:

```
[comet-ln2:~/comet-examples/PHYS244/OPENMP] sbatch openmp-slurm.sb
Submitted batch job 20912556
[comet-ln2:~/comet-examples/PHYS244/OPENMP] squeue -u user
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
          20912556   compute hello_op  user PD       0:00      1 (None)
[comet-ln2:~/comet-examples/PHYS244/OPENMP] squeue -u user
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
          20912556   compute hello_op  user  R       0:00      1 comet-10-45
[comet-ln2:~/comet-examples/PHYS244/OPENMP] squeue -u user
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
          20912556   compute hello_op  user CG       0:03      1 comet-10-45
[comet-ln2:~/comet-examples/PHYS244/OPENMP] squeue -u user
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
[comet-ln2:~/comet-examples/PHYS244/OPENMP] cat  hello_openmp.20912556.comet-10-45.out
 Hello from Thread Number[           0 ] and Welcome Webinar Participants!
 Hello from Thread Number[          18 ] and Welcome Webinar Participants!
 Hello from Thread Number[           4 ] and Welcome Webinar Participants!
 Hello from Thread Number[          15 ] and Welcome Webinar Participants!
 Hello from Thread Number[          21 ] and Welcome Webinar Participants!
 Hello from Thread Number[          11 ] and Welcome Webinar Participants!
 Hello from Thread Number[          16 ] and Welcome Webinar Participants!
….
```

Running Hybrid MPI-OpenMP Jobs

Hybrid MPI + OpenMP Jobs

- Several HPC codes use a hybrid MPI, OpenMP approach.
- **ibrun** wrapper developed to handle hybrid use cases.
  - Automatically senses the MPI build (*mvapich2, openmpi*) and binds tasks correctly.
- **ibrun** -help gives detailed usage info.

Hybrid MPI + OpenMP Hello World

```
[comet-ln2:~/comet-examples/PHYS244] cd HYBRID
[comet-ln2:~/comet-examples/PHYS244/HYBRID] ls -al
total 94
drwxr-xr-x  2 user use300      5 Jan  8 01:53 .
drwxr-xr-x 16 user use300     16 Aug  5 19:02 ..
-rw-r--r--  1 user use300    636 Aug  5 19:02 hello_hybrid.c
-rw-r--r--  1 user use300    390 Aug  5 19:02 hybrid-slurm.sb
[comet-ln2:~/comet-examples/PHYS244/HYBRID] cat hello_hybrid.c
#include <stdio.h>
#include "mpi.h"
#include <omp.h>

int main(int argc, char *argv[]) {
  int numprocs, rank, namelen;
  char processor_name[MPI_MAX_PROCESSOR_NAME];
  int iam = 0, np = 1;

  MPI_Init(&argc, &argv);
  MPI_Comm_size(MPI_COMM_WORLD, &numprocs);
  MPI_Comm_rank(MPI_COMM_WORLD, &rank);
  MPI_Get_processor_name(processor_name, &namelen);

  #pragma omp parallel default(shared) private(iam, np)
  {
    np = omp_get_num_threads();
    iam = omp_get_thread_num();
    printf("Hello Webinar particpants from thread %d out of %d from process %d out of %d on %s\n",
           iam, np, rank, numprocs, processor_name);
  }

  MPI_Finalize();
}
```

Hybrid Hello World: Compile, batch script

- To compile the hybrid MPI + OpenMPI code, we need to refer to the table of compilers listed above (and listed in the user guide).
- We will use the command **mpicc -openmp**

```
[comet-ln2:~/comet-examples/PHYS244/HYBRID] mpicc -openmp -o hello_hybrid hello_hybrid.c
[comet-ln2:~/comet-examples/PHYS244/HYBRID] ls -al
total 94
drwxr-xr-x  2 user use300      5 Jan  8 02:00 .
drwxr-xr-x 16 user use300     16 Aug  5 19:02 ..
-rwxr-xr-x  1 user use300 103032 Jan  8 02:00 hello_hybrid
-rw-r--r--  1 user use300    636 Aug  5 19:02 hello_hybrid.c
-rw-r--r--  1 user use300    390 Aug  5 19:02 hybrid-slurm.sb
[comet-ln2:~/comet-examples/PHYS244/HYBRID]
```

```
[comet-ln2:~/comet-examples/PHYS244/HYBRID] cat hybrid-slurm.sb
#!/bin/bash
#SBATCH --job-name="hellohybrid"
#SBATCH --output="hellohybrid.%j.%N.out"
#SBATCH --partition=compute
#SBATCH --nodes=2
#SBATCH --ntasks-per-node=24
#SBATCH --export=ALL
#SBATCH -t 01:30:00

#This job runs with 2 nodes, 24 cores per node for a total of 48 cores.
# We use 8 MPI tasks and 6 OpenMP threads per MPI task

export OMP_NUM_THREADS=6
ibrun --npernode 4 ./hello_hybrid
```

Hybrid Hello World: submit job & monitor

To run the job, type the batch script submission command:

```
[comet-ln2:~/comet-examples/PHYS244/HYBRID] sbatch hybrid-slurm.sb
Submitted batch job 20912643
[comet-ln2:~/comet-examples/PHYS244/HYBRID] squeue -u user
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
          20912643   compute hellohyb  user PD       0:00      2 (None)
[comet-ln2:~/comet-examples/PHYS244/HYBRID] squeue -u user
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
          20912643   compute hellohyb  user  R       0:01      2 comet-06-[48,64]
[comet-ln2:~/comet-examples/PHYS244/HYBRID] squeue -u user
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
          20912643   compute hellohyb  user CG       0:06      2 comet-06-[48,64]
[comet-ln2:~/comet-examples/PHYS244/HYBRID] squeue -u user
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
[comet-ln2:~/comet-examples/PHYS244/HYBRID] ll
total 132
drwxr-xr-x  2 user use300      7 Jan  8 02:12 .
drwxr-xr-x 16 user use300     16 Aug  5 19:02 ..
-rwxr-xr-x  1 user use300 103032 Jan  8 02:00 hello_hybrid
-rw-r--r--  1 user use300   3771 Jan  8 02:12 hellohybrid.20912643.comet-06-48.out
-rw-r--r--  1 user use300    636 Aug  5 19:02 hello_hybrid.c
-rw-r--r--  1 user use300    390 Aug  5 19:02 hybrid-slurm.sb ….
```

Hybrid Hello World: Output

Code ran on:

- 2 nodes,
- 4 cores per node,
- 6 threads per core

```
[comet-ln2:~/comet-examples/PHYS244/HYBRID] cat hellohybrid.20912643.comet-06-48.out | sort
Hello from thread 0 out of 6 from process 0 out of 8 on comet-06-48.sdsc.edu
Hello from thread 0 out of 6 from process 1 out of 8 on comet-06-48.sdsc.edu
Hello from thread 0 out of 6 from process 2 out of 8 on comet-06-48.sdsc.edu
Hello from thread 0 out of 6 from process 3 out of 8 on comet-06-48.sdsc.edu
Hello from thread 0 out of 6 from process 4 out of 8 on comet-06-64.sdsc.edu
Hello from thread 0 out of 6 from process 5 out of 8 on comet-06-64.sdsc.edu
Hello from thread 0 out of 6 from process 6 out of 8 on comet-06-64.sdsc.edu
Hello from thread 0 out of 6 from process 7 out of 8 on comet-06-64.sdsc.edu
Hello from thread 1 out of 6 from process 0 out of 8 on comet-06-48.sdsc.edu
Hello from thread 1 out of 6 from process 1 out of 8 on comet-06-48.sdsc.edu
Hello from thread 1 out of 6 from process 2 out of 8 on comet-06-48.sdsc.edu
Hello from thread 1 out of 6 from process 3 out of 8 on comet-06-48.sdsc.edu
Hello from thread 1 out of 6 from process 4 out of 8 on comet-06-64.sdsc.edu
Hello from thread 1 out of 6 from process 5 out of 8 on comet-06-64.sdsc.edu
Hello from thread 1 out of 6 from process 6 out of 8 on comet-06-64.sdsc.edu
Hello from thread 1 out of 6 from process 7 out of 8 on comet-06-64.sdsc.edu
Hello from thread 2 out of 6 from process 0 out of 8 on comet-06-48.sdsc.edu
Hello from thread 2 out of 6 from process 1 out of 8 on comet-06-48.sdsc.edu
Hello from thread 2 out of 6 from process 2 out of 8 on comet-06-48.sdsc.edu
Hello from thread 2 out of 6 from process 3 out of 8 on comet-06-48.sdsc.edu
Hello from thread 2 out of 6 from process 4 out of 8 on comet-06-64.sdsc.edu
Hello from thread 2 out of 6 from process 5 out of 8 on comet-06-64.sdsc.edu
Hello from thread 2 out of 6 from process 6 out of 8 on comet-06-64.sdsc.edu
Hello from thread 2 out of 6 from process 7 out of 8 on comet-06-64.sdsc.edu
Hello from thread 3 out of 6 from process 0 out of 8 on comet-06-48.sdsc.edu
Hello from thread 3 out of 6 from process 1 out of 8 on comet-06-48.sdsc.edu
Hello from thread 3 out of 6 from process 2 out of 8 on comet-06-48.sdsc.edu
Hello from thread 3 out of 6 from process 3 out of 8 on comet-06-48.sdsc.edu
Hello from thread 3 out of 6 from process 4 out of 8 on comet-06-64.sdsc.edu
Hello from thread 3 out of 6 from process 5 out of 8 on comet-06-64.sdsc.edu
Hello from thread 3 out of 6 from process 6 out of 8 on comet-06-64.sdsc.edu
Hello from thread 3 out of 6 from process 7 out of 8 on comet-06-64.sdsc.edu
Hello from thread 4 out of 6 from process 0 out of 8 on comet-06-48.sdsc.edu
Hello from thread 4 out of 6 from process 1 out of 8 on comet-06-48.sdsc.edu
Hello from thread 4 out of 6 from process 2 out of 8 on comet-06-48.sdsc.edu
Hello from thread 4 out of 6 from process 3 out of 8 on comet-06-48.sdsc.edu
Hello from thread 4 out of 6 from process 4 out of 8 on comet-06-64.sdsc.edu
Hello from thread 4 out of 6 from process 5 out of 8 on comet-06-64.sdsc.edu
Hello from thread 4 out of 6 from process 6 out of 8 on comet-06-64.sdsc.edu
Hello from thread 4 out of 6 from process 7 out of 8 on comet-06-64.sdsc.edu
Hello from thread 5 out of 6 from process 0 out of 8 on comet-06-48.sdsc.edu
Hello from thread 5 out of 6 from process 1 out of 8 on comet-06-48.sdsc.edu
Hello from thread 5 out of 6 from process 2 out of 8 on comet-06-48.sdsc.edu
Hello from thread 5 out of 6 from process 3 out of 8 on comet-06-48.sdsc.edu
Hello from thread 5 out of 6 from process 4 out of 8 on comet-06-64.sdsc.edu
Hello from thread 5 out of 6 from process 5 out of 8 on comet-06-64.sdsc.edu
Hello from thread 5 out of 6 from process 6 out of 8 on comet-06-64.sdsc.edu
Hello from thread 5 out of 6 from process 7 out of 8 on comet-06-64.sdsc.edu
```

Running GPU/CUDA Jobs

Comet GPU Nodes

NVIDIA Kepler K80 GPU Nodes	
Node count	36
CPU cores:GPUs/node	24:4
CPU:GPU DRAM/node	128 GB:48 GB
NVIDIA Pascal P100 GPU Nodes	
Node count	36
CPU cores:GPUs/node	28:4
CPU:GPU DRAM/node	128 GB:64 GB

GPU/CUDA MatMul

- Change to the CUDA examples directory:

```
[comet-ln2:~/comet-examples/PHYS244] cd CUDA
[comet-ln3:~/comet-examples/PHYS244/CUDA] ll -al
total 474
drwxr-xr-x  2 user use300     16 Jan  8 09:47 .
drwxr-xr-x 16 user use300     16 Aug  5 19:02 ..
-rw-r--r--  1 user use300    503 Jan  8 09:31 CUDA.20915480.comet-31-11.out
-rw-r--r--  1 user use300    253 Aug  5 19:02 cuda.sb
-rw-r--r--  1 user use300   5106 Aug  5 19:02 exception.h
-rw-r--r--  1 user use300   1168 Aug  5 19:02 helper_functions.h
-rw-r--r--  1 user use300  29011 Aug  5 19:02 helper_image.h
-rw-r--r--  1 user use300  23960 Aug  5 19:02 helper_string.h
-rw-r--r--  1 user use300  15414 Aug  5 19:02 helper_timer.h
-rwxr-xr-x  1 user use300 535634 Jan  8 09:28 matmul
-rw-r--r--  1 user use300  13556 Aug  6 00:54 matrixMul.cu
```

GPU/CUDA: Compile

- Set the environment
- Then compile the code

```
[comet-ln2:~/comet-examples/PHYS244/CUDA] module purge
[comet-ln2:~/comet-examples/PHYS244/CUDA] which nvcc
/usr/bin/which: no nvcc in (/usr/lib64/qt-
3.3/bin:/usr/local/bin:/bin:/usr/bin:/usr/local/sbin:/usr/sbin:/sbin:/opt/sdsc/bin:/o
pt/sdsc/sbin:/opt/ibutils/bin:/usr/java/latest/bin:/opt/pdsh/bin:/opt/rocks/bin:/opt/
rocks/sbin:/home/user/bin)
[comet-ln2:~/comet-examples/PHYS244/CUDA] module load cuda
[comet-ln2:~/comet-examples/PHYS244/CUDA] which nvcc
/usr/local/cuda-7.0/bin/nvcc
[comet-ln2:~/comet-examples/PHYS244/CUDA] nvcc -o matmul -I.  matrixMul.cu
[comet-ln2:~/comet-examples/PHYS244/CUDA] ll matmul
-rwxr-xr-x 1 user use300 535634 Jan  8 09:28 matmul
[comet-ln2:~/comet-examples/PHYS244/CUDA]
```

GPU/CUDA: check node for GPU card

Note: you will be able to compile GPU code on the login nodes, but they will not run. To see if your node has GPU hardware, run *lspci*. Comet login nodes do not have GPU.

```
[comet-ln2:~/comet-examples/PHYS244/CUDA] lspci | grep VGA
09:00.0 VGA compatible controller: ASPEED Technology, Inc. ASPEED Graphics Family
(rev 30)
```

If the node does have a GPU card, you will see output similar to the following:

```
[user@host.sdsu.edu]$ ssh node9 "/sbin/lspci | grep VGA"
01:00.0 VGA compatible controller: NVIDIA Corp.. NV44 [GeForce 6200 LE] (rev a1)
02:00.0 VGA compatible controller: NVIDIA Corp.. GF100 [GeForce GTX 480] (rev a3)
03:00.0 VGA compatible controller: NVIDIA Corp.. GF100 [GeForce GTX 480] (rev a3)
```

GPU/CUDA: Batch Script Config

- GPU nodes can be accessed via either the "gpu" or the "gpu-shared" partitions.

```
#SBATCH -p gpu
   or
#SBATCH -p gpu-shared
```

- In addition to the partition name(required), the type of gpu(optional) and the individual GPUs are scheduled as a resource.

```
#SBATCH --gres=gpu[:type]:n
```

- GPUs will be allocated on a first available, first schedule basis, unless specified with the [type] option, where type can be k80 or p100 (type is case sensitive)

```
#SBATCH --gres=gpu:4      #first available gpu node
#SBATCH --gres=gpu:k80:4 #only k80 nodes
#SBATCH --gres=gpu:p100:4 #only p100 nodes
```

GPU/CUDA: Batch Script

SLURM batch script contents:

```
[comet-ln2:~/comet-examples/PHYS244/CUDA] cat cuda.sb
#!/bin/bash
#SBATCH --job-name="CUDA"
#SBATCH --output="CUDA.%j.%N.out"
#SBATCH --partition=gpu-shared          # define GPU partition
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=6
#SBATCH --gres=gpu:1          # define type of GPU
#SBATCH -t 01:00:00

#Load the cuda module
module load cuda

#Run the job
./matmul
[comet-ln2:~/comet-examples/PHYS244/CUDA]
```

GPU/CUDA: submit job & monitor

- To run the job, type the batch script submission command:

```
[comet-ln2:~/comet-examples/PHYS244/CUDA] sbatch cuda.sb
Submitted batch job 20915480
```

- Monitor the job until it is finished

```
[comet-ln2:~/comet-examples/PHYS244/CUDA] squeue -u user
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
          20915480 gpu-share     CUDA  user PD       0:00      1 (None)
[comet-ln2:~/comet-examples/PHYS244/CUDA] squeue -u user
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
[comet-ln2:~/comet-examples/PHYS244/CUDA] ll CUDA.20915480.comet-31-11.out
-rw-r--r-- 1 user use300 503 Jan  8 09:31 CUDA.20915480.comet-31-11.out
```

GPU/CUDA: Check Environment!

obtaining device information: enum_gpu.cu (1)

```
#include "../common/book.h"
int main( void ) {
    cudaDeviceProp  prop;
    int count;
    HANDLE_ERROR( cudaGetDeviceCount( &count ) );
    for (int i=0; i< count; i++) {
        HANDLE_ERROR( cudaGetDeviceProperties( &prop, i ) );
        printf( "   --- General Information for device %d ---\n", i );
        printf( "Name:  %s\n", prop.name );
        printf( "Compute capability:  %d.%d\n", prop.major, prop.minor );
        printf( "Clock rate:  %d\n", prop.clockRate );
        printf( "Device copy overlap:  " );
        if (prop.deviceOverlap)
            printf( "Enabled\n" );
        else
            printf( "Disabled\n");
        printf( "Kernel execution timeout :  " );
        if (prop.kernelExecTimeoutEnabled)
            printf( "Enabled\n" );
        else
            printf( "Disabled\n" );
```

GPU/CUDA: Check Environment!

obtaining device information: enum_gpu.cu (2)

```
        printf( "   --- Memory Information for device %d ---\n", i );
        printf( "Total global mem:  %ld\n", prop.totalGlobalMem );
        printf( "Total constant Mem:  %ld\n", prop.totalConstMem );
        printf( "Max mem pitch:  %ld\n", prop.memPitch );
        printf( "Texture Alignment:  %ld\n", prop.textureAlignment );
        printf( "   --- MP Information for device %d ---\n", i );
        printf( "Multiprocessor count:  %d\n",
                    prop.multiProcessorCount );
        printf( "Shared mem per mp:  %ld\n", prop.sharedMemPerBlock );
        printf( "Registers per mp:  %d\n", prop.regsPerBlock );
        printf( "Threads in warp:  %d\n", prop.warpSize );

        printf( "Max threads per block:  %d\n",
                    prop.maxThreadsPerBlock );
        printf( "Max thread dimensions:  (%d, %d, %d)\n",
                    prop.maxThreadsDim[0], prop.maxThreadsDim[1],
                    prop.maxThreadsDim[2] );
        printf( "Max grid dimensions:  (%d, %d, %d)\n",
                    prop.maxGridSize[0], prop.maxGridSize[1],
                    prop.maxGridSize[2] );
        printf( "\n" );
        }
}
```

GPU/CUDA: Check Environment

```
------------------------------------------------
--- General Information for device 0 ---
Name:  Tesla C1060
Compute capability:  1.3
Clock rate:  1296000
Device copy overlap:  Enabled
Kernel execution timeout :  Disabled
--- Memory Information for device 0 ---
Total global mem:  4294770688
Total constant Mem:  65536
Max mem pitch:  2147483647
Texture Alignment:  256
--- MP Information for device 0 ---
Multiprocessor count:  30
Shared mem per mp:  16384
Registers per mp:  16384
Threads in warp:  32
Max threads per block:  512
Max thread dimensions:  (512, 512, 64)
Max grid dimensions:  (65535, 65535, 1)
------------------------------------------------
--- General Information for device 2 ---
Name:  GeForce GT 240
Compute capability:  1.2
Clock rate:  1340000
Device copy overlap:  Enabled
Kernel execution timeout :  Disabled
--- Memory Information for device 2 ---
Total global mem:  1073020928
Total constant Mem:  65536
Max mem pitch:  2147483647
Texture Alignment:  256
```

```
------------------------------------------------
--- General Information for device 1 ---
Name:  Tesla C1060
Compute capability:  1.3
Clock rate:  1296000
Device copy overlap:  Enabled
Kernel execution timeout :  Disabled
   --- Memory Information for device 1 ---
Total global mem:  4294770688
Total constant Mem:  65536
Max mem pitch:  2147483647
Texture Alignment:  256
   --- MP Information for device 1 ---
Multiprocessor count:  30
Shared mem per mp:  16384
Registers per mp:  16384
Threads in warp:  32
Max threads per block:  512
Max thread dimensions:  (512, 512, 64)
Max grid dimensions:  (65535, 65535, 1)
------------------------------------------------
   --- MP Information for device 2 ---
Multiprocessor count:  12
Shared mem per mp:  16384
Registers per mp:  16384
Threads in warp:  32
Max threads per block:  512
Max thread dimensions:  (512, 512, 64)
Max grid dimensions:  (65535, 65535, 1)
```

Wrapping it up

Yes, You are Correct: Running jobs on HPC Systems is Complex

- **Multiple layers of hardware and software affect job performance**
- **Learn to develop and test in a modular fashion**
- **Build up a suite of test cases:**
  - When things go wrong, make sure you can run simple test cases (HelloWorld).
  - This can eliminate questions about your environment.
- **Consider using a code repository**
  - When things go wrong, you can get back to a working version
- **If you need help/have questions, contact XSEDE help desk:**
  - **They are very helpful and respond quickly**
  - Support users around the world, so they are truly a 7/24 service
  - Avoid wasting your time.

When Things Go Wrong, Check Your User Environment

- **Do you have the right modules loaded?**
- **What software versions do you need?**
- **Is your code compiled and updated (or did you compile it last year?)**
- **Are you running your job from the right location?**
  - $HOME versus $WORK?

Run jobs from the right location

- **Lustre scratch filesystem:**
  - /oasis/scratch/comet/$USER/temp_project
  - Preferred: Scalable large block I/O)
- **Compute/GPU node local SSD storage:**
  - /scratch/$USER/$SLURM_JOBID
  - Meta-data intensive jobs, high IOPs)
- **Lustre projects filesystem:**
  - /oasis/projects/nsf
- **/home/$USER:**
  - Only for source files, libraries, binaries.
  - *Do not* use for I/O intensive jobs.

References

- **Comet User Guide**
  - https://www.sdsc.edu/support/user_guides/comet.html#compiling
- **SDSC Training Resources**
  - https://www.sdsc.edu/education_and_training/training.html
  - https://github.com/sdsc-training/webinars
  - Comet shared apps/examples; can be found in
    - /share/apps
- **XSEDE Training Resources**
  - https://www.xsede.org/for-users/training
  - https://cvw.cac.cornell.edu/comet/