Using Large-Scale Computing Resources at CU

Shelley Knuth
shelley.knuth@colorado.edu

Peter Ruprecht
peter.ruprecht@colorado.edu

www.rc.colorado.edu Questions? #RC_Meetups

Link to survey on this topic: http://goo.gl/forms/8VidcwOhRT

Slides: https://github.com/ResearchComputing/Final_Tutorials

What does Research Computing do?

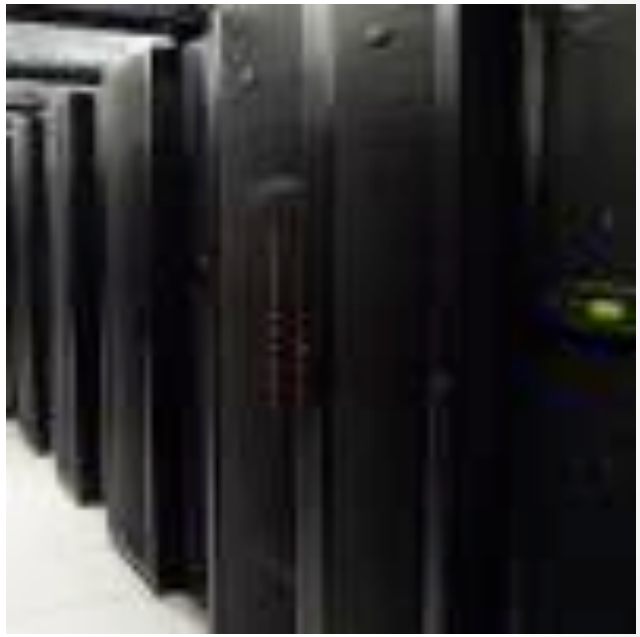
- We manage
  - Shared large scale compute resources
  - Large scale storage
  - High-speed network without firewalls – ScienceDMZ
  - Software and tools
- We provide
  - Consulting support for building scientific workflows on the RC platform
  - Training
  - Data management support in collaboration with the Libraries

What Is a Supercomputer?

- A supercomputer is one large computer made up of many smaller computers and processors
- Each different computer is called a node
- Each node has processors/cores
  - Carry out the instructions of the computer
- With a supercomputer, all these different computers talk to each other through a communications network
  - Example - InfiniBand

Hardware - Janus Supercomputer

- 1368 compute nodes (Dell C6100)
- 16,428 total cores
- No battery backup of the compute nodes
- Fully non-blocking QDR Infiniband network
- 960 TB of usable Lustre based scratch storage
  - 16-20 GB/s max throughput



Additional Compute Resources

- 2 Graphics Processing Unit (GPU) Nodes
  - Visualization of data
  - Exploring GPUs for computing
- 4 High Memory Nodes
  - 1 TB of memory, 60-80 cores per node
- 16 Blades for long running jobs
  - 2-week walltimes allowed
  - 96 GB of memory (4 times more compared to a Janus node)

Next-Generation Supercomputer

- Expected performance about 450 TFLOPS (compared to about 170 for Janus)
- Compute nodes
  - Expect 24 real cores and 128 GB RAM
- 10 GPU/visualization nodes
  - 2x NVIDIA K80 GPUs
- 5 High-memory nodes
- 20 Xeon Phi ("Knight's Landing") nodes
- "Omni-Path" high-performance interconnect
- 1 PB of high-performance scratch storage

Initial Steps to Use RC Systems

- Apply for an RC account
  - https://www.rc.colorado.edu/support/getting-started.html#account
- Get a One-Time Password device
- Apply for a computing allocation
  - Startup allocation of 50K SU granted immediately
  - Additional SU require a proposal
  - You may be able to use an existing allocation

Logging in

- From a command line:

```
ssh -X <username>@login.rc.colorado.edu
Password:  4-digit pin+6-digits from OTP
```

- Land at a login node

Different Node Types

- Login nodes
  - This is where you are when you log in
  - No heavy computation, interactive jobs, or long running processes
  - Script or code editing, minor compiling
  - Job submission
- Compute/batch nodes
  - This is where jobs that are submitted through the scheduler run
  - Intended for heavy computation
- Compile nodes
  - Compiling software (janus-compile1-4)

Storage Spaces

- System variations
- **Home Directories**
  - Store source code
  - Not for direct computation
  - Small quota (~5 GB)
  - Backed up
- **$WORK Space**
  - Mid level quota (~300 GB)
  - Large file storage
  - Not backed up

- **Scratch Directory**
  - Much larger – depends on system
  - Output from running jobs should go here
  - Files generally purged at some point

Job Scheduling

- On a supercomputer, jobs are scheduled rather than just run instantly at the command line
  - People “buy” time to use the resources
  - Shared system
  - Request the amount of resources needed and for how long
  - Jobs are put in a queue until resources are available
  - Once the job is run they are “charged” for the time they used

Job Schedulers - Slurm

- Jobs on supercomputers are managed and run by different software
- Simple Linux Utility for Resource Management (Slurm)
  - Open source software package
- Slurm is a resource manager
  - Keeps track of what nodes are busy/available, and what jobs are queued or running
- Slurm is a scheduler
  - Tells the resource manager when to run which job on the available resources

Modules Package

- Dynamic modification of environment settings that are required for using compilers, libraries, and applications
- Easily change environment settings through module files
- Software not loaded by default
  - Allows multiple versions of the software
- Hierarchical
  - Programs built with a compiler need to be connected to libraries built with the same compiler

Modules Package

- We have aliased the module commands
  - Module is now “ml”

Command	Purpose
ml	Displaying all loaded modulefiles.
ml av	Displaying all available modulefiles.
ml spider [module_file_name]	Search for software.
ml [module_file_name]	Loading the modulefiles.
ml -[module_file_name]	Unloading the modulefiles.

Running Jobs

- What is a “job”?
- Interactive jobs
  - Work interactively at the command line of a compute node
- Batch jobs
  - Submit job that will be executed when resources are available
  - Create a text file containing information about the job
  - Submit the job file to a queue
- Load the Slurm module!

Running an Interactive Job

- Open a login shell on first allocated compute node, supporting X11 forwarding
- Can use a GUI

```
sinteractive --qos janus-debug
```

- Don’t need a GUI?

```
salloc --qos janus-debug
```

Example – Interactive Job

- Let's practice running an interactive job

```
ml slurm
sinteractive --reservation=balch
```

Once the prompt comes up, type

```
ml intel
ml R
R
```

Interactive Jobs

- NEVER run for a long time
- Your job will start when resources become available (2 am?)
- Good for the janus-debug queue
- Once resources become available you are granted a job allocation on a node

  salloc: Granted job allocation 602576 (output)
- Once this comes on the screen your compute allocation is being used
- If you type hostname you will be told which node you are on

  bash-4.1$ hostname

  node0211
- Then type matlab, etc to run program

Queues

- There are several ways to define a “queue”
- Clusters may have different queues set up to run different types of jobs
  - Certain queues might exist on certain clusters/resources
  - Other queues might be limited by maximum wall time
- Slurm can use a “quality of service” for each queue
  - aka “QOS”
- Also can use a “partition” (or set of nodes) that corresponds to a queue
- Janus queues: https://www.rc.colorado.edu/support/user-guide/batch-queueing.html#qos

Submit Batch Job example

```
#!/bin/bash
#SBATCH –N 2                          #No. nodes
#SBATCH --ntasks-per-node=12          #No. cores
#SBATCH --time=1:00:00                #Max walltime
#SBATCH --job-name=SLURMDemo          #Job name
#SBATCH --output=SLURMDemo.out        #Output file name
###SBATCH -A <account>                #Allocation
###SBATCH --mail-type=end             #Send Email completion
###SBATCH --mail-user=<your@email>    #Email address

ml intel
ml openmpi/1.8.5

mpirun ./hello
```

Submit Batch Job example

- Have to make sure the slurm module is loaded!
- Submit the job normally:

```
Sbatch slurmSub.sh
```

- For us:

```
sbatch --reservation=balch slurmSub.sh
```

- Check job status in the janus-debug queue:

```
squeue –q janus-debug
```

- Check output:

```
cat SLURMDemo.out
```

Your Turn

- Submit a slurm job with the following instructions:

1. The job should run the Unix “hostname” command
   - Hint – the command “srun” will run commands in slurm
2. The job should be submitted from a bash script named practice.sh
   - Don’t forget to make it executable!
3. The job should run for 5 minutes in the default queue
4. The job should be run on 1 node
5. The output should be put in a file called hostname.txt

Your Turn - Solution

Bash Script practice.sh:

```
#!/bin/bash
#SBATCH -N 1                          # No. of nodes
#SBATCH --time=0:05:00                # Walltime
#SBATCH --output=hostname.txt         # Output file name

srun hostname
```

Submit the job:

```
sbatch practice.sh
```

Research Data Storage: PetaLibrary

- NSF Major Research Instrumentation grant
- Long term storage option
- Keep data on spinning disk or tape
- Provide expertise and services around this storage
  - Data management
  - Consulting
- No HIPAA, FERPA data
- Infrastructure guaranteed for 3 more years

Globus

- Globus is our preferred method of data transfer
- Designed with researchers in mind
- End points between computers make for easy data transfer with an easy to use interface
  - Endpoints are different locations that data can be moved to/from
  - Personal or multi-user
- Scripting in use also if don't want to use GUI

www.globus.org

Globus

- Preserves the integrity of data
  - Compares checksums
  - Resumes data transfer if interrupted
- Fast transfer of large data sets
- Globus can be set up to easily share data among collaborators

Using Globus to Transfer Between Laptop and Janus

- Create an endpoint on your laptop
  https://www.globus.org/xfer/ManageEndpoints#

- Start transfer between your laptop and Janus
  https://www.globus.org/xfer/StartTransfer
  - Janus' endpoint is colorado#gridftp

Questions?

- Email rc-help@colorado.edu
- Twitter: CUBoulderRC

- Link to survey on this topic:
  http://goo.gl/forms/8VidcwOhRT

- Slides:
  https://github.com/ResearchComputing/Final_Tutorials