Introduction to Running Computations on the High Performance Cluster at the Center for Computational Research
CCR Seminar: MATLAB Example

Cynthia Cornelius

Center for Computational Research
University at Buffalo, SUNY

cdc at buffalo.edu

December 2015

CCR Resources

The Center for Computational Research provides high performance computing resources to the University at Buffalo.

- Supporting faculty research and classroom education, as well as local business and University collaborations.
- High performance and high through-put cluster.
- High performance remote visualization.

What is a cluster?

- A cluster is a collection of individual computers connected by a network that can act as a single more powerful machine.
  - Each individual computer has its own CPUs, memory and disks. It runs an operating system and it connected to a network that is usually private to the cluster. An individual computer is referred to as a compute node.

What is a cluster?

- A High Performance Cluster is a cluster that can run computations that require large number of compute nodes each. There are software packages and a high speed network that make this possible.
- A High Throughput Cluster is a cluster that can run a great many computations that require a single compute node or a single CPU on a compute node.

CCR Cluster

- The CCR cluster is collection of linux computers, private networks, a shared file system, a login computer, and job scheduler.
- Resources must be requested from the **job scheduler**.
- The user must decide what resources are required for the job.
- The user must control and optimize the job.

CCR Cluster

- The CCR cluster is **NOT a cloud**.
- Resources are NOT on demand.
- There is NO rapid elasticity.
- There is NO measured service to optimize the use of resources.

Access to the CCR Cluster

- The front-end machine is **rush.ccr.buffalo.edu**
- 32-core node with 256GB of memory.
- Accessible from **UB network only**.
- On campus your machine must be on the LAN or connected to UBsecure wireless.
- Off campus use the UB provided VPN client.

How to login

- Login from Linux or Mac
- ssh -X rush.ccr.buffalo.edu
- **ssh -X UBITusername@rush.ccr.buffalo.edu**
  - The -X flag enables a graphical display. This is optional.
  - Note: MAC users should in the XQuartz package.
- Windows users must install X-Win32, PuTTY, or OpenSSH for a secure login.
- The **X-Win32** program allows for a graphical display.
  - more on how to login

How to transfer files

- **Filezilla** is graphical file transfer program, which is available for Windows, Linux and MAC computers.
- **WinSCP** is available on Windows.
- **Cyberduck** is available on MACs.
- more on how to transfer files
- Get the software from the UBit webpage.
  - UBit software

Basic UNIX Commands

- The cluster compute nodes and front-end machine run Linux (CentOS).
- It is a command line **UNIX environment**.
- Here are two references for learning the basic commands:
  - more on basic UNIX commands
  - CCR UNIX Reference Card

Basic UNIX Commands

- Users should start with the basic UNIX commands, such as **ls** (list), **cd** (change directory), **cp** (copy), and **mkdir** (make directory).
- There are several editors available: emacs, nano and vi.
- Users can edit file on their laptops and transfer the files to the CCR cluster.
- The dos2unix command will remove any hidden characters in text files transferred from Windows machines

Where are my files?

- GPFS (IBM General Parallel File System) provides storage space of user home directories, projects directories and global scratch space.
- All compute node access GPFS.
- The path for a home directory is **/user/<username>**.
- Faculty can request projects space for the research group. The path is **/projects/<faculty-username>**
- Home and project directories are backed up daily.

Where are my files?

- **/gpfs/scratch** is global scratch space.
  - This space is for temporary use.
  - Use scratch space for running jobs only.
  - Files older than 3 weeks are automatically removed.
  - There is no backup of files in /gpfs/scratch.
- Every compute node has a **/scratch** directory on the local disk. Jobs can use this local scratch directory.

Class Exercise

- When you login are will be in your home directory.
- Login to **rush.ccr.buffalo.edu**
- List the contents of you home directory:
  - **TYPE:** ls
  - **TYPE:** ls -latr
    - (long, all, time, reverse)
- Show the path of the directory:
  - **TYPE:** pwd

Class Exercise

- Show my group membership:
  - **TYPE:** id username
- Show the quota on my home directory:
  - **TYPE:** gquota
    - the default quota is 2GB
- Show the quota on the group's projects space:
  - **TYPE:** gquota -g group
    - hint: use the gid name from the id command.
- What is the quota for your projects space?

Class Exercise

- Many application are already installed on the CCR cluster.
- LMOD module files are used to set environment variables and paths of application software.
- Show what modules you have loaded.
  - **TYPE:** module list
- List all available modules.
  - **TYPE:** module avail
- List all matlab modules.
  - **TYPE:** module avail matlab

Class Exercise

- Show the module for matlab/R2014b.
  - **TYPE:** module show matlab/R2014b
  - hint: this shows what the module files does when you load it.
- Load the matlab/R2014b module.
  - **TYPE:** module load matlab/R2014b
  - hint: now you can run matlab
- Unload the matlab module.
  - **TYPE:** module unload matlab/R2014b

Class Exercise

- Create a directory in your home directory.
  - **TYPE:** mkdir class
- List the current directory.
  - **TYPE:** ls -l
- Note the permissions on the class directory.
  - drwxr-xr-x
  - from left to right: d for directory, rwx (read, write, execute) for the user, r-x (read and execute) for members of the group, r-x for world.

Class Exercise

- Note: the write permission means write, modify, and delete.
  - **Exception:** projects spaces do not allow one user to modify or delete another user's files.
- Use the chmod command to remove access to work for the class directory.
  - **TYPE:** chmod o-rwx class
  - hint: the syntax is chmod ugo+-rwx name
- View the manual page for chmod.
  - **TYPE:** man chmod

Class Exercise

- Change directory to the class directory.
  - **TYPE:** cd class
- Copy the following files from /gpfs/scratch/cdc/matlab-examples directory.
  - **TYPE:** cp /gpfs/scratch/cdc/ccr-class/slurmMATLAB slurmMATLAB
  - **TYPE:** cp /gpfs/scratch/cdc/ccr-class/test.m test.m
- List the contents of the class directory.
  - **TYPE:** ls -l

CCR Cluster Resources

- You must select resources for each job.
  - partition (job queue)
    - default is general-compute
  - compute nodes
    - default is 1 node
  - number of cores (CPUs) per node
    - default is 1 core
  - memory (per compute node)
    - default is 2800MB per core

CCR Cluster Resources

- time limit
  - default for the general-compute partition is 72 hours
  - default for debug partition is 1 hour
- type of compute node (constraint)
  - default is to run on any node that fulfills the core and memory requirements of the job
- Note: compute nodes are shared by default

CCR Cluster Compute Nodes

- There are over **700 compute nodes** providing ~8000 cores.
  - Hint: think of a core as a CPU.
- Compute nodes are grouped according to number of cores.
- A job will always be assigned nodes with the same number of cores.
  - No job would ever have a mix of 8-core and 12-core compute nodes.

Number of Compute Nodes

- 372 **12-core** compute nodes
- 256 **8-core** compute nodes
- 32 **16-core** compute nodes
- 18 **32-core** compute nodes
- 32 **12-core** compute nodes with 2 GPUs each
- more on compute nodes

How to choose a compute node

- Most users choose compute nodes based on **number of nodes**, **number of cores per node**, and **memory per node** required for the job.
- 8-core compute nodes have 24GB of memory.
- 12-core compute nodes have 48GB of memory.
- 16-core compute nodes have 128GB of memory.
- 16 of the 32-core compute nodes have 256GB, while 2 have 512GB of memory.
- GPU Compute nodes have 48GB of memory and 2 Nvidia Fermi GPUs.

How to choose a partition

- SLURM **job queues** are referred to as partitions. Here are the **partitions** for the CCR cluster.
- **general-compute** - default partition if no partition is specified for a job.
  - almost all compute nodes
  - **per user limit of 1000** running and pending jobs at a time
  - maximum time limit of 72 hours

How to choose a partition

- **gpu** - higher priority for only the compute nodes with GPUs.
  - **per user limit of 32** running or pending jobs at a time.
  - maximum time limit of 72 hours
- **largemem** - higher priority for only the 32-core compute nodes.
  - **per user limit of 32** running or pending jobs at a time.
  - maximum time limit of 72 hours

How to choose a partition

- **debug** - small partition for quick debugging.
  - 8 dedicated compute nodes
    - 4 8-core compute nodes,
    - 2 12-core compute nodes
    - 1 16-core node with 2 GPUs
    - 1 16-core node with a XEON PHI coprocessor
  - **per user limit of 4** running or pending jobs at a time.
  - maximum time limit of **1 hour**

SLURM Commands

- **squeue** — shows the status of jobs.
- **sbatch** —submits a script job.
- **scancel** —cancels a running or pending job.
- **snodes** — shows details of the compute nodes.
- **slurmjobvis** – graphical job monitoring tool.
- more on SLURM commands

Class Exercise

- List the jobs in the queues.
  - **TYPE:** squeue
- List the jobs in the debug partition.
  - **TYPE:** squeue -p debug
- List the compute nodes in the queues
  - **TYPE:** snodes
- List the compute nodes in the debug partition.
  - **TYPE:** snodes all debug

SLURM Script Example

- The #SBATCH lines are directives to the scheduler. The directives are the resource requests, job output file, and email preferences.

```
#!/bin/sh
#SBATCH --partition=debug
#SBATCH --time=00:15:00
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=8
#SBATCH --mem=23000
#SBATCH --constraint=CPU-L5630|CPU-L5520
#SBATCH --job-name="test"
#SBATCH --output=test.out
##SBATCH --mail-user=username@buffalo.edu
##SBATCH --mail-type=END
```

SLURM Script Example

- The following lines print information to the job output file.

```
echo "SLURM_JOB_ID="$SLURM_JOB_ID
echo "SLURM_JOB_NODELIST"=$SLURM_JOB_NODELIST
echo "SLURMTMPDIR="$SLURMTMPDIR

echo "working directory = "$SLURM_SUBMIT_DIR
```

- The job starts in the directory from which you submitted it. This is the working directory.

SLURM Script Example

- Load the module file for MATLAB.
- List the loaded modules.

```
module load matlab/R2014b
module list
ulimit -s unlimited
#
```

- The ulimit command removes the size limit on the instruction stack. This helps large programs to run.

SLURM Script Example

- Run the MATLAB computation using the command line.

```
#
echo "run matlab computation"
matlab -nodisplay < test.m
date
echo "All Done!"
```

- Once all the lines of the SLURM have been executed, then the job completes and exits the compute node.

How to submit a SLURM script job

- Submit an interactive job using the **sbatch** <your_slurm_script>
- The job will be submitted to the SLURM scheduler.
- The job will wait in the queue until the scheduler assigns resources to it. This is a pending state.
- more on submitting a job script

Class Exercise

- View the slurmMATLAB script file.
  - **TYPE:** more slurmMATLAB
  - hint: press space bar to page
  - Note: a sleep was added so that you will see the job in the queue. The computation runs very quickly.
- What resources are requested by the SLURM job script?
- View the test.m file.
  - **TYPE:** more test.m

Class Exercise

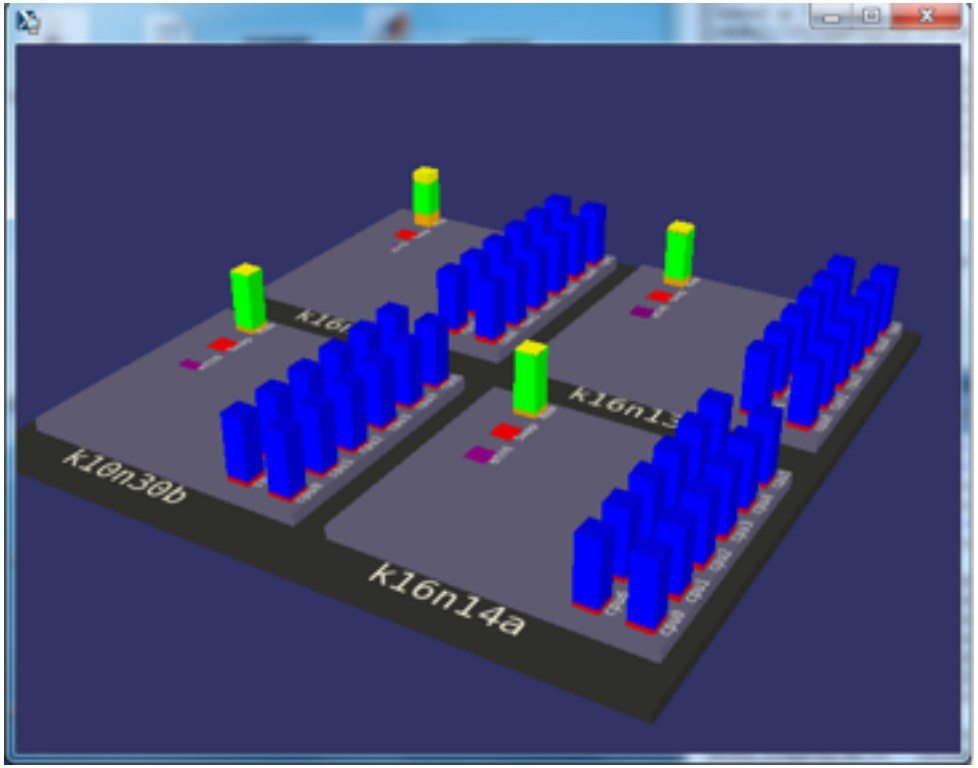
- Submit the job.
  - **TYPE:** sbatch slurmMATLAB
- Check the status of the job.
  - **TYPE:** squeue -u username
  - **TYPE:** squeue -j jobid
- Check the status of the debug partition.
  - **TYPE:** squeue -p debug
- Note: The PD state is pending. The job is waiting in the queue.

Class Exercise

- Monitor the job using slurmjobvis.
  - **TYPE:** slurmjobvis jobid
- Monitor the using ssh and top:
  - **TYPE:** ssh cpn-XXX-xx
  - hint: get the compute node running the job using the squeue command
  - **TYPE**: top
  - The top command show the CPU and memory utilization.
- Leave the compute node.
  - **TYPE:** exit

Job monitoring

- The **slurmjobvis** is a graphical display of the activity on the node.
- CPU, memory, network, as well as GPU utilization, are displayed.



How to submit an interactive job

- Submit an interactive job using the **fisbatch** wrapper.
- Although the job is interactive it still waits in the queue.
- Useful for debugging.
- Specify partition, nodes, cores or tasks, and time.
- Once the job starts the user is logged into the compute node.

How to submit an interactive job

- **TYPE:** fisbatch --partition=debug --nodes=1 --ntasks-per-node=8 —time=01:00:00
- When the job starts you will be logged into the compute node.
- Now load the matlab module and type matlab. The matlab GUI will launch.
  - This can be a bit slow.
- more on submitting an interactive job

More Information and Help

- CCR SLURM web pages
- More sample SLURM scripts can be found in the /util/academic/slurm-scripts directory on rush.
- Compute Cluster web page
- Remote Visualization web page
- Users can get assistance by sending an email to ccr-help@ccr.buffalo.edu.