Introduction to Running Computations on the High Performance Clusters at the Center for Computational Research

Cynthia Cornelius

Center for Computational Research
University at Buffalo, SUNY

cdc at buffalo.edu

Spring 2015

CENTER FOR COMPUTATIONAL RESEARCH
University at Buffalo *The State University of New York*

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.

CCR Cluster

- Users run programs on the cluster as **jobs** submitted to queues.
  - A **job** is a request for computers, cores, memory, networking, queue, and time.
- The CCR cluster is a high performance.
  - Large parallel jobs run on the cluster.
- The CCR cluster is high through-put.
  - A large number of jobs run on the cluster at one time.

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.

CCR Cluster

- The CCR cluster provides over 8,000 cores.
- There is a mix of 8-core, 12-core , 16-core and 32-core compute nodes.
  - A **compute node** is a linux machine with memory, disks, and cores (cpus), as well as both ethernet and Infiniband network connections.
- The GPFS (IBM General Parallel File System) is used for storage.
- There is a login front-end server.
- SLURM (Simple Linux Utility Resource Manager) is the job scheduler.

CCR Cluster

- **IMPORTANT!**
- Recent and planned changes to the CCR cluster.
  - All users should look at the 2015 What's New guide.

Where are my files?

- GPFS provides storage space of user home directories, projects directories and global scratch space.
- All compute node access GPFS.
- A home directory has a 2GB quota by default. The path 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.

CCR Cluster Compute Nodes

- There are over 700 compute nodes providing ~8000 cores.
- 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.
- The maximum number of compute nodes that users can request in a job is the total for that group of nodes.

Number of Compute Nodes

- **12-core** compute nodes — **372**
- **8-core** compute nodes — **256**
- **16-core** compute nodes — **32**
- **32-core** compute nodes — **18**
- **12-core** nodes with GPUs — **32**

- [more on compute nodes]

How to choose a compute node

- Most users choose compute nodes based on number of nodes, number of cores, and memory per node required for the job.
- 8-core compute node 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**

How to choose a partition

- **viz** - partition for jobs running on the remote visualization nodes.
  - Users do not submit directly to the viz partition.
  - **per user limit of 32** running or pending jobs at a time.
  - maximum time limit of 24 hours
  - how to use the Remote Visualization compute nodes

How to choose a partition

- **supporters** - higher priority partition for users belonging to research groups that have contributed funds to CCR.
  - almost all compute nodes
  - **per user limit of 1000** running or pending jobs at a time
  - maximum time limit of 72 hours
- more on CCR SLURM partitions

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**.
- Only secure protocols, such as **ssh** and **sftp**, are permitted to access the front-end.
- Small test computations can run on the front-end machine.
    - CPU time limit of 15 minutes on the front-end. Processes that exceed the time limit are automatically terminated.

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.
- Windows users must install X-Win32 or PuTTY for the 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.
- **UBVPN** must be used to access the rush login machine from off campus.
- Users should setup UB-Secure on their laptops for connecting through wireless.
- more on how to transfer files
- Get the software from the UBit webpage.
  - UBit software

Command Line Environment

- The compute nodes and front-end machine run Linux. It is a command line **UNIX environment**.
- Users should have know basic UNIX commands, such as ls, cd and mkdir.
- There are several editors available: emacs, nano and vi.
- Use the dos2unix command to remove any hidden characters in text files transferred from Windows machines.
- more on basic UNIX commands
- CCR UNIX Reference Card

Using Software and Compilers

- Compilers, JAVA, Python, MPI and application software are available to use on cluster.
- Lmod is used to set paths and variables for the software.
- "**module avail**" shows all installed software packages.
- "**module load <package-name>**" puts the software in the user's path.
- more on using Lmod and modules

Compilers, MPI and more

- The Intel, PGI and GNU compilers are available on the cluster. GNU compilers are already in the user's default path.
- MPI (Message Passing Interface) implementations are installed on the cluster.
- More recent versions of JAVA are available.
- Anaconda Python is available.
  - more on using Anaconda Python

SLURM Commands

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

How to submit an interactive job

- Submit an interactive job using the **fisbatch** wrapper.
- Useful for debugging.
- Specify partition, nodes, cores or tasks, and time.
- Once the job starts the user is logged into the compute node.
- more on submitting an interactive job

SLURM script example

```
#!/bin/sh
#SBATCH --partition=general-compute
#SBATCH --time=00:15:00
#SBATCH --nodes=2
#SBATCH --ntasks-per-node=8
#SBATCH --mem=24000
# Memory per node specification is in
MB. It is optional.
# The default limit is 3000MB per core.
```

SLURM script example

```
#SBATCH --job-name="hello_test"
#SBATCH --output=test-srun.out
#SBATCH --mail-
user=username@buffalo.edu
#SBATCH —mail-type=ALL
echo "SLURM_JOBID="$SLURM_JOBID
echo "SLURM_JOB_NODELIST"=
$SLURM_JOB_NODELIST
```

SLURM script example

```
echo "SLURM_NNODES"=$SLURM_NNODES
echo “SLURMTMPDIR="$SLURMTMPDIR

echo  "working directory =
"$SLURM_SUBMIT_DIR

module load intel/13.1
module load intel-mpi/4.1.3
module list
ulimit -s unlimited
```

SLURM script example

```
NPROCS=`srun --nodes=${SLURM_NNODES}
bash -c 'hostname' |wc -l`
echo NPROCS=$NPROCS
echo "Launch helloworld with srun"
#The PMI library is necessary for srun
export I_MPI_PMI_LIBRARY=/usr/lib64/
libpmi.so
srun ./helloworld
#
echo "All Done!"
```

Commonly used SLURM variables

- **$SLURM_JOBID**
- **$SLURM_JOB_NODELIST**
  - Node list in SLURM format; for example f16n[04,06].
- **$SLURM_NNODES**
  - Number of nodes
- **$SLURMTMPDIR**
  - /scratch/jobid
  - local to the compute node
- **$SLURM_SUBMIT_DIR**
  - Directory from which the job was submitted
- **NOTE!** Jobs start in the $SLURM_SUBMIT_DIR.

Task Launching

- The number of cores/processes can be computed.
  - Use srun to get the total number of cores.
  - NPROCS=`srun --nodes=${SLURM_NNODES} bash -c 'hostname' |wc -l`
- Intel-MPI mpirun and mpiexec are SLURM aware.
- srun will execute a command across nodes.
  - Typically, this is the best choice for launching a parallel computation.

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

Check the status of the job

```
squeue -u <username>
squeue -j <job_id>
```

```
 JOBID PARTITION NAME USER ST TIME NODES
NODELIST(REASON)
4832 general-c hello_te cdc R 0:20 2 f16n[10-11]
```

- Job status:
  - **R** – job is running.
  - **PD** – job is waiting for resource.
    - Reasons are usually (Resources) or (Priority).
  - Others commons reasons are CA (cancelled) and CD (completed).

Partition and node status

sinfo -p general-compute

```
PARTITION      AVAIL  TIMELIMIT  NODES  STATE
NODELIST
general-comput*    up 3-00:00:00    264   idle
d07n07s[01-02],d07n08s[01-02], …
```

- Node states:
  - **alloc** – all cores are in use.
  - **mix** – some cores are available.
  - **idle** – node is free. All cores are available.
  - **down**- node is down.
  - **drained** – node is offline.

Node status and details

```
[cdc@rush:~]$ snodes k14n16s01
HOSTNAMES  STATE    CPUS S:C:T    CPUS(A/I/O/T)   CPU_LOAD
MEMORY   GRES     PARTITION          FEATURES
k14n16s01  alloc    12   2:6:1    12/0/0/12      12.06    48000
(null)   general-compute*   IB,CPU-E5645

cdc@rush:~]$ snodes k05n22
HOSTNAMES  STATE    CPUS S:C:T    CPUS(A/I/O/T)   CPU_LOAD
MEMORY   GRES     PARTITION          FEATURES
k05n22     idle     16   2:8:1    0/16/0/16      0.04     128000   mic:1
debug            CPU-E5-2660,MIC
[cdc@rush:~]$
```

Node Sharing

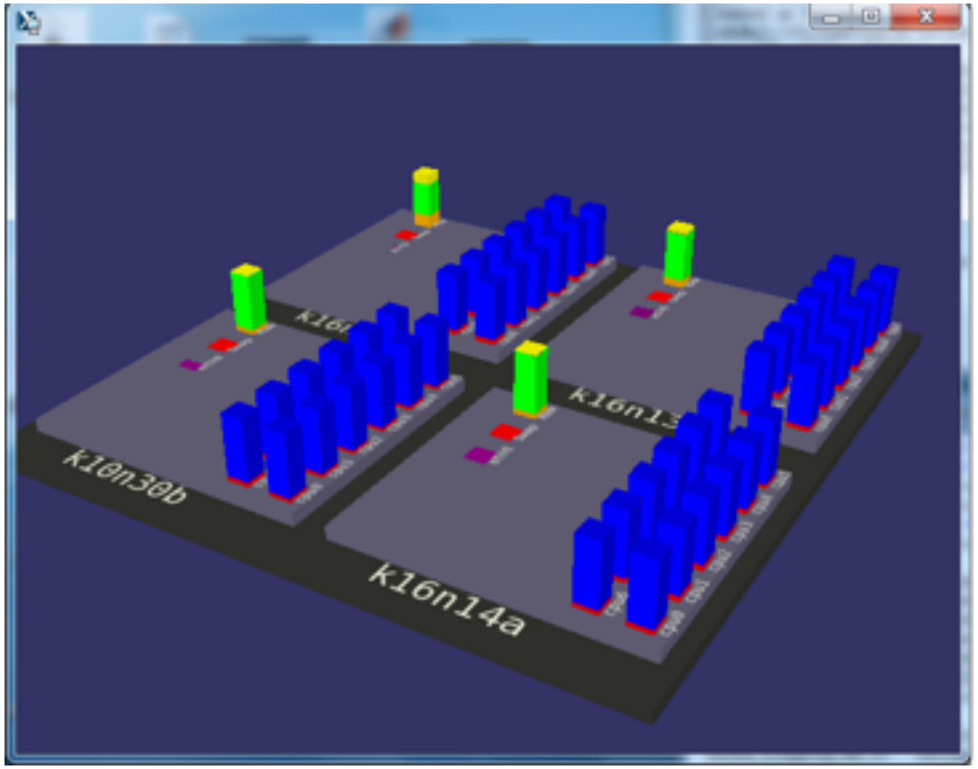
- Compute nodes are **shared** among different jobs and users.
- Tasks are limited to the number of cores and memory specified.
- The integration of CPUSETS and SLURM makes this possible.
  - **CPUSET** is a Linux kernel level mechanism that can be used to control access to individual cores.
- The default memory limit per core is 3000MB.

Node Sharing

- --mem=24000
  - Requests 24GB per node.
- --mem-per-core=16000
  - Requests 16GB on a core; use for serial job.
- Jobs exceeding the memory limit will be terminated by the resource manager.
- Check the General Compute Cluster webpage for node memory and core details.
- The --exclusive flag will request the nodes as dedicated. The nodes will not be shared.

Job monitoring

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



Putting it all together

- Login to the cluster login front-end machine.
- Transfer files to home directory.
- Locate necessary software with “module avail”
- Load modules and compile the code.
- Create a job script.
- Submit a script.
- Monitor job with squeue and slurmjobvis.

More Information and Help

- CCR SLURM web pages
- **/gpfs/courses/class-notes** for sample jobs.
- 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.