A Quest for Computing at Northwestern

Matt Selensky, PhD
Computational Specialist
Northwestern IT
Research Computing Services

Applied Statistics Seminar
January 19th, 2024
Computing environments

A typical desktop/laptop
- Local access
- A single "node"
- Windows, MacOS, maybe Linux
- Limited to the hardware you can afford
- Lower RAM (4-16GB), lower core count (4-12), likely an integrated GPU
- Easy to use!

A "cluster"
- Remote access
- Many "nodes"
- Almost certainly Linux
- Limited to the hardware a large institution can afford ($$$$)
- Higher RAM (up to 1.5TB), higher core count (52+), dedicated GPU resources
- Takes time to learn!
What is Quest?
Quest: Northwestern's High-Performance Computing Cluster
Quest consists of ~1000 nodes

“Node” = computer
Each node consists of 28-64 cores

“Node” = computer

“Core” = CPU/processor
Technical specifications of nodes vary

There are several generations of nodes!

<table>
<thead>
<tr>
<th>Generation</th>
<th># cores per node</th>
<th>Total memory</th>
<th>Memory per core</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q9</td>
<td>40</td>
<td>192 GB</td>
<td>~4.8 GB</td>
</tr>
<tr>
<td>Q10</td>
<td>52</td>
<td>192 GB</td>
<td>~3.7 GB</td>
</tr>
<tr>
<td>Q11</td>
<td>64</td>
<td>256 GB</td>
<td>~4 GB</td>
</tr>
<tr>
<td>Q12</td>
<td>64</td>
<td>256 GB</td>
<td>~4 GB</td>
</tr>
</tbody>
</table>
Other types of nodes available for “General Access” (i.e., free) use

- “General Access”
  - 15 A100 GPU nodes
    - 40GB RAM (PCIE)
    - 80GB RAM (SXM)
  - 1 high memory node
    - 1.5TB RAM

Extra resources for bioinformaticians at NU

- The "Genomics Compute Cluster" (GCC)
  - A special account funded by Feinberg & Weinberg to support bioinformatics/genomics research at NU
- 8 additional A100 GPU cards on 2 nodes
- 3 additional high memory nodes with RAM up to 1.5TB
- ~350TB of shared scratch space
- Dedicated support specialist: Haley Carter

Learn more about the GCC!
Quest Analytics Nodes

The Quest Analytics Nodes provide…
- On-demand access to Rstudio Server, Jupyter and SAS Studio through the web browser.
- Access to Quest file system.
- More computational resources than available on a personal computer.

The Quest Analytics Nodes are…
- Available to all Quest users with an active allocation
- Accessible while on GlobalProtect VPN outside the campus

Learn more about Quest Analytics Nodes
How can I start using Quest?
Apply for a general access allocation

Research I Allocation
- 1 paragraph statement of purpose
- Approval process within 5 business days
- Renew for allocation yearly

Research II Allocation
- Research I allocations can be upgraded to Research II allocations via Research II renewal form
- Requires a research proposal

Join the Genomics Compute Cluster!
- Join allocation "b1042"

Request a Research Allocation
Please click on the link for the type of allocation you are requesting and complete the online form. If you are requesting a Research II Allocation, you will also need to fill out and attach the Research II Allocation Proposal to your application form.

Research Allocation I Original - Request to renew an existing R-I allocation or downgrade an existing R-II to R-I allocation for research purposes.
Research Allocation I Renewal - Request to renew an existing R-I allocation or upgrade an existing R-II to R-I allocation for research purposes.
Research Allocation II Original - Request to create a new R-II allocation for research purposes.
Research Allocation II Renewal - Request to renew an existing R-II allocation or upgrade and existing R-I to R-II allocation for research purposes.
Join an Existing Allocation - Request to join an existing allocation.

Apply now!

General access allocations are totally free to use
Purchasing resources on Quest

- “Buy-in” allocations are a great option for faculty wanting dedicated compute/storage on Quest
- Resources purchased for a period of 5 years

Contact us if you are interested!

More information about buy-in allocations
Quest's filesystem

• File system is shared across all computers on Quest
  • You can access your files & folders from anywhere – you do not have to be on the same node

• You will be working with 4 main folders:

<table>
<thead>
<tr>
<th>Folder</th>
<th>Description</th>
<th>Space</th>
</tr>
</thead>
<tbody>
<tr>
<td>/home/&lt;netID&gt;</td>
<td>Personal home directory</td>
<td>80GB</td>
</tr>
<tr>
<td>/projects/&lt;allocID&gt;</td>
<td>Allocation directory</td>
<td>1TB (R1) or 2TB (R2)</td>
</tr>
<tr>
<td>/scratch/&lt;netID&gt;</td>
<td>Scratch space (opt-in)</td>
<td>5TB</td>
</tr>
<tr>
<td>/hpc/software</td>
<td>System-level software</td>
<td>N/A</td>
</tr>
</tbody>
</table>
Quest's filesystem: extra resources

- **Quest Storage Knowledge Base**
- **Quest Storage and Data Policy**

<table>
<thead>
<tr>
<th></th>
<th>Home Space</th>
<th>Allocation or Project Space</th>
<th>Scratch Space</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Storage Name</strong></td>
<td>Home Directories</td>
<td>Allocation or Project</td>
<td>Scratch Space Directories</td>
</tr>
<tr>
<td>Where is this located on</td>
<td>/home/&lt;netid&gt;</td>
<td>Directories</td>
<td></td>
</tr>
<tr>
<td>Quest?</td>
<td></td>
<td>/projects/&lt;allocation_id&gt;</td>
<td>/scratch/&lt;netid&gt;</td>
</tr>
<tr>
<td>What should go here?</td>
<td>Submission scripts, job log</td>
<td>High-speed storage which</td>
<td>High-speed storage which</td>
</tr>
<tr>
<td></td>
<td>files, and local package and</td>
<td>should be used for</td>
<td>should be used for</td>
</tr>
<tr>
<td></td>
<td>software installations</td>
<td>computation Input/Output</td>
<td>storing temporary files</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(IQ) and/or data analyses.</td>
<td>from running jobs, downloading</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>data for processing, and</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>short-term storage for large</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>datasets.</td>
</tr>
<tr>
<td>What are the default</td>
<td>Only the user has read, write</td>
<td>All members of the</td>
<td>Only the user has read,</td>
</tr>
<tr>
<td>permissions?</td>
<td>and execute permissions.</td>
<td>allocation will have read,</td>
<td>write and execute permissions.</td>
</tr>
<tr>
<td>How much disk space?</td>
<td>80GB</td>
<td>write and/or execute</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>permissions on files and</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>directories created in the</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>allocation directory.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>buyin allocations can</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>purchase storage by the TB.</td>
<td></td>
</tr>
</tbody>
</table>
Transferring data to and from Quest: Globus

- Globus transfers have faster rates compared to `scp`, `rsync` or `sftp`.
- Globus enables collaboration and data sharing on local or remote systems
- “Fire and forget” transfers, retries any failed attempts, checks for corruption
- Learn more about Globus!
Submitting jobs to the Quest compute nodes

User → Submit jobs → Compute nodes
Log into the **login nodes** & submit jobs to the **scheduler**
Jobs: Requesting resources from SLURM

• Batch job
  o Submit your job as a pre-written bash script
  o Benefit – submit & forget about it

• Interactive job
  • Run interactive session on the compute nodes
  • Benefit – exploratory work, troubleshooting etc.
Batch jobs – a special bash script

#!/bin/bash
#SBATCH --account=p12345
#SBATCH --partition=short
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=1
#SBATCH --time=00:10:00
#SBATCH --mem-per-cpu=1G
#SBATCH --job-name=sample_job
#SBATCH --output=outlog
#SBATCH --mail-type=END,FAIL
#SBATCH --mail-user=email@northwestern.edu

module purge all
module load python/3.10.1

python hello_world.py

Learn more about batch jobs!
Interactive jobs

Learn more about interactive jobs!

[netID@quser31 ~]$ srun -A p12345 -p short -t 04:00:00 --mem=18G --pty bash -l
srun: job 4465087 queued and waiting for resources
srun: job 4465087 has been allocated resources
----------------------------------------
...  
central
----------------------------------------
[netID@qnode9030 ~]$ [netID@qnode9030 ~]$ $ echo "Hello, Applied Statistics Seminar"
Hello, Applied Statistics Seminar
New service coming soon: Quest OnDemand!

- Browser-based tool
- Launch a variety of GUIs on a compute node without having to know how to use Slurm or the command line!
- Jupyter, RStudio, MATLAB, and more!
- Requires GlobalProtect VPN if you are off-campus
Live demonstration
Questions?

quest-help@northwestern.edu

Schedule a Zoom consultation with us

In-person office hours:
Mondays 3-4pm at the Mudd Library GIS Lab
(2nd Floor across from the bridge to Tech)
EXTRA SLIDES: Special types of jobs and extra Slurm information
Job Arrays
#!/bin/bash
#SBATCH --account=w10001  ## YOUR ACCOUNT pXXXX or bXXXX
#SBATCH --partition=w10001  ### PARTITION (buyin, short, normal, etc)
#SBATCH --nodes=1  ## how many computers do you need
#SBATCH --ntasks-per-node=4  ## how many cpus or processors do you need on each computer
#SBATCH --time=00:10:00  ## how long does this need to run
#SBATCH --mem-per-cpu=1G  ## how much RAM do you need per CPU (affects your FairShare score!)
#SBATCH --job-name=sample_job  ## When you run squeue -u NETID this is how you can identify the job
#SBATCH --output=outlog  ## standard out and standard error goes to this file
#SBATCH --mail-type=ALL  ## you can receive e-mail alerts from SLURM when your job begins and when your job finishes (completed, failed, etc)
#SBATCH --mail-user=email@u.northwestern.edu  ## your email

module purge all
module load python-anaconda3
source activate /projects/intro/envs/slurm-py37-test

python slurm_test.py --nproc ${SLURM_NPROCS}
#!/bin/bash
#SBATCH --account=w10001  ## YOUR ACCOUNT pXXXX or bXXXX
#SBATCH --partition=w10001  ### PARTITION (buyin, short, normal, etc)
#SBATCH --nodes=2 ## how many computers do you need
#SBATCH --ntasks-per-node=4 ## how many cpus or processors do you need on each computer
#SBATCH --time=00:10:00 ## how long does this need to run
#SBATCH --mem-per-cpu=1G ## how much RAM do you need per CPU (this affects your FairShare score)
#SBATCH --job-name=sample_job  ## When you run squeue -u NETID this is how you can identify the job
#SBATCH --output=outlog ## standard out and standard error goes to this file
#SBATCH --mail-type=ALL ## e-mail alerts from SLURM about job status
#SBATCH --mail-user=email@u.northwestern.edu ## your email
#SBATCH --constraint="[quest7|quest8|quest9|quest10]" ### you want computers you have requested to be from
## either quest7 or quest8 or quest 9 or quest10 nodes, not a combination of nodes. Important for MPI

module purge all
module load python-anaconda3
source activate /projects/intro/envs/slurm-py37-test

mpiexec -n ${SLURM_NTASKS} python -m mpi4py.bench helloworld
SLURM: Job-Array Example

#SBATCH --account=w10001  ## YOUR ACCOUNT pXXXX or bXXXX
#SBATCH --partition=w10001  ### PARTITION (buyin, short, normal, etc)
#SBATCH --array=0-9 ## number of jobs to run "in parallel"
#SBATCH --nodes=1  ## how many computers do you need
#SBATCH --ntasks-per-node=1  ## how many cpus or processors do you need on each computer
#SBATCH --time=00:10:00  ## how long does this need to run (remember different partitions have restrictions on this param)
#SBATCH --mem-per-cpu=1G  ## how much RAM do you need per CPU (this effects your FairShare score so be careful to not ask for more than you need)
#SBATCH --job-name="sample_job_${SLURM_ARRAY_TASK_ID}"  ## use the task id in the name of the job
#SBATCH --output=sample_job.%A_%a.out  ## use the jobid (A) and the specific job index (a) to name your log file
#SBATCH --mail-type=ALL  ## you can receive e-mail alerts from SLURM when your job begins and when your job finishes (completed, failed, etc)
#SBATCH --mail-user=email@u.northwestern.edu  ## your email

module purge all
module load python-anaconda3
source activate /projects/intro/envs/slurm-py37-test

IFS="'\n' read -d '' -r -a lines < list_of_files.txt

python slurm_test.py --job-id ${SLURM_ARRAY_TASK_ID} --filename ${lines[${SLURM_ARRAY_TASK_ID}]}
SLURM: Job-Array Example (cont)

```bash
#SBATCH --account=w10001  ## YOUR ACCOUNT pXXXX or bXXXX
#SBATCH --partition=w10001  ### PARTITION (buyin, short, normal, etc)
#SBATCH --array=0-9%2  ## number of jobs to run "in parallel" the %2 restricts so that only 2 jobs max can be running simultaneously
#SBATCH --nodes=1  ## how many computers do you need
#SBATCH --ntasks-per-node=1  ## how many cpus or processors do you need on each computer
#SBATCH --time=00:10:00  ## how long does this need to run (remember different partitions have restrictions on this param)
#SBATCH --mem-per-cpu=1G  ## how much RAM do you need per CPU (this effects your FairShare score so be careful to not ask for more than you need))
#SBATCH --job-name="sample_job_${SLURM_ARRAY_TASK_ID}"  ## use the task id in the name of the job
#SBATCH --output=sample_job.%A_%a.out  ## use the jobid (A) and the specific job index (a) to name your log file
#SBATCH --mail-type=ALL  ## you can receive e-mail alerts from SLURM when your job begins and when your job finishes (completed, failed, etc)
#SBATCH --mail-user=email@u.northwestern.edu  ## your email

module purge all
module load python-anaconda3
source activate /projects/intro/envs/slurm-py37-test
IFS='\n' read -d '' -r -a lines < list_of_files.txt
python slurm_test.py --job-id $SLURM_ARRAY_TASK_ID --filename ${lines[$SLURM_ARRAY_TASK_ID]}
Dependent Jobs
SLURM: Dependency Example

#!/bin/bash

jid0=($(sbatch --time=00:10:00 --account=w10001 --partition=w10001 --nodes=1 --ntasks-per-node=1 --mem=8G --job-name=example --output=job_%A.out example_submit.sh))

echo "jid0 ${jid0[-1]}" >> slurm_ids

jid1=($(sbatch --dependency=afterok:${jid0[-1]} --time=00:10:00 --account=w10001 --partition=w10001 --nodes=1 --ntasks-per-node=1 --mem=8G --job-name=example --output=job_%A.out --export=DEPENDENTJOB=${jid0[-1]} example_submit.sh))

echo "jid1 ${jid1[-1]}" >> slurm_ids

jid2=($(sbatch --dependency=afterok:${jid1[-1]} --time=00:10:00 --account=w10001 --partition=w10001 --nodes=1 --ntasks-per-node=1 --mem=8G --job-name=example --output=job_%A.out --export=DEPENDENTJOB=${jid1[-1]} example_submit.sh))

echo "jid2 ${jid2[-1]}" >> slurm_ids
SLURM: Dependency Example

#!/bin/bash
#SBATCH --mail-type=ALL ## you can receive e-mail alerts from SLURM when your job begins and
when your job finishes (completed, failed, etc)
#SBATCH --mail-user=email@u.northwestern.edu ## your email

if [[ -z "${DEPENDENTJOB}" ]]; then
  echo "First job in workflow"
else
  echo "Job started after " $DEPENDENTJOB
fi

module purge all
module load python-anaconda3
source activate /projects/intro/envs/slurm-py37-test

python --version
python slurm_test.py
Architecture Constraints
SLURM: Constraints Example

```
#!/bin/bash
#SBATCH --account=w10001  ## YOUR ACCOUNT pXXXX or bXXXX
#SBATCH --partition=w10001  ### PARTITION (buyin, short, normal, etc)
#SBATCH --nodes=2 ## how many computers do you need
#SBATCH --ntasks-per-node=4 ## how many cpus or processors do you need on each computer
#SBATCH --time=00:10:00 ## how long does this need to run (remember different partitions have restrictions on this param)
#SBATCH --mem-per-cpu=1G ## how much RAM do you need per CPU (this effects your FairShare score so be careful to not ask for more than you need))
#SBATCH --job-name=sample_job  ## When you run squeue -u NETID this is how you can identify the job
#SBATCH --output=outlog ## standard out and standard error goes to this file
#SBATCH --mail-type=ALL ## you can receive e-mail alerts from SLURM when your job begins and when your job finishes (completed, failed, etc)
#SBATCH --mail-user=email@u.northwestern.edu ## your email
#SBATCH --constraint="[quest7|quest8|quest9|quest10]" ### you want computers you have requested to be from either quest7 or quest8 or quest9 or quest10 nodes, not a combination of nodes. Useful for MPI applications.

module purge all
module load python-anaconda3
source activate /projects/intro/envs/slurm-py37-test

mpiexec -n ${SLURM_NTASKS} python -m mpi4py.bench helloworld
```
SLURM Environmental Variables
SLURM: Environmental Variables

[quser21 ~]$ srun -N 2 --ntasks-per-node=5 --mem=10G --account=a9009 --partition=all --time=00:10:00 --pty bash -l
...
...
[qnode0156 ~]$ printenv | grep SLURM
SLURM_NODELIST=qnode[0156-0157]
SLURM_NTASKS_PER_NODE=5
SLURM_NNODES=2
SLURM_JOBID=8177560
SLURM_NTASKS=10
SLURM_SUBMIT_DIR=/home/tempuser03
SLURM_NPROCS=10