Using the Sapelo2 Cluster at the GACRC

Cluster New User Training Workshop

Georgia Advanced Computing Resource Center (GACRC)
EITS/University of Georgia
Zhuofei Hou  zhuofei@uga.edu
Outline

• GACRC

• Sapelo2 Cluster
  ➢ Diagram
  ➢ Overview
  ➢ Three Directories
  ➢ Three Computational Queues
  ➢ Software on Cluster

• Job Submission Workflow

• Work on Sapleo2
GACRC

- We are a high-performance-computing (HPC) center at UGA
- We provide to the UGA research and education community an advanced computing environment:
  - HPC computing and networking infrastructure located at the Boyd Data Center
  - Comprehensive collection of scientific, engineering and business applications
  - Consulting and training services

- [http://wiki.gacrc.uga.edu](http://wiki.gacrc.uga.edu) (GACRC Wiki)
- [https://wiki.gacrc.uga.edu/wiki/Getting_Help](https://wiki.gacrc.uga.edu/wiki/Getting_Help) (GACRC Support)
- [http://gacrc.uga.edu](http://gacrc.uga.edu) (GACRC Web)
Sapelo2 Cluster

1. ssh with MyID and password
2. Verify with Archpass Duo two-factor authentication

- Node: **Computer** for a specific function on cluster, e.g., login node
- Queue: Collection of **compute nodes** for specific computing need
- Cluster: **Nodes + Drives**, all connected by network
Sapelo2: A Linux HPC cluster (64-bit Centos 7)

- Two Nodes:
  1. Login node for batch job workflow: MyID@sapelo2.gacrc.uga.edu
  2. Transfer node for data transferring: MyID@xfer.gacrc.uga.edu

- Three Directories:
  1. Home: Landing spot; 100GB quota; Backuped
  2. Global Scratch: High performance job working space; NO quota; NOT backuped
  3. Storage: Temporary data parking; 1TB quota (for group); Backuped (ONLY accessible from Transfer node!)

- Three Computational Queues: batch, highmem_q, gpu_q
# Three Directories

<table>
<thead>
<tr>
<th>Role</th>
<th>Directory</th>
<th>Intended Use</th>
<th>Quota</th>
<th>Accessible from</th>
<th>Backuped</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Home</td>
<td>/home/MyID</td>
<td><strong>Static data:</strong> 1. Scripts, source codes 2. Local software</td>
<td>100GB</td>
<td>Login</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Global Scratch</td>
<td>/lustre1/MyID</td>
<td><strong>Current job data:</strong> data being read/written by running jobs</td>
<td>No Limit</td>
<td>Transfer</td>
<td>No</td>
<td>User to clean up! Subject to deletion in 30 days</td>
</tr>
<tr>
<td>Storage</td>
<td>/project/abclab</td>
<td><strong>Temporary data parking:</strong> non-current active data</td>
<td>1TB (Initial)</td>
<td>Transfer</td>
<td>Yes</td>
<td>Group sharing possible</td>
</tr>
</tbody>
</table>
Three Directories (Cont.) – Accessing Directories from Nodes

- Login
  - /home/MyID
  - /lustre1/MyID

- Interactive

- Transfer
  - /project/abclab

User:
- log on using ssh
- qlogin
- exit

Diagram:
- Log on using ssh
- Interactive
- Transfer
## Three Computational Queues

<table>
<thead>
<tr>
<th>Queue</th>
<th>Node Feature</th>
<th>Total Nodes</th>
<th>RAM(GB) /Node</th>
<th>Max RAM(GB) /Single-node Job</th>
<th>Cores /Node</th>
<th>Processor Type</th>
<th>GPU Cards /Node</th>
</tr>
</thead>
<tbody>
<tr>
<td>batch</td>
<td>Intel</td>
<td>30</td>
<td>64</td>
<td>62</td>
<td>28</td>
<td>Intel Xeon</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td>AMD</td>
<td>3</td>
<td>128</td>
<td>126</td>
<td>48</td>
<td>AMD Opteron</td>
<td></td>
</tr>
<tr>
<td>highmem_q</td>
<td>Intel</td>
<td>1</td>
<td>1024</td>
<td>997</td>
<td>28</td>
<td>Intel Xeon</td>
<td></td>
</tr>
<tr>
<td>gpu_q</td>
<td>GPU</td>
<td>2</td>
<td>96</td>
<td>94</td>
<td>12</td>
<td>7 NVIDIA K20Xm</td>
<td></td>
</tr>
<tr>
<td>PIMyID_q</td>
<td>variable</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Software on Cluster

1. Software names are long and have a Easybuild toolchain name associated to it, e.g., foss-2016b
2. Software names are case-sensitive!

- `ml avail`: List all available software modules installed on cluster
- `ml moduleName`: Load a module into your working environment
- `ml`: List modules currently loaded
- `ml unload moduleName`: Remove a module from your working environment
- `ml spider pattern`: Search module names matching a pattern (case-insensitive)
Job Submission Workflow

1. Log on to Login node using MyID and password, and two-factor authentication with Archpass Duo:
   
   ```bash
   ssh MyID@sapelo2.gacrc.uga.edu
   ```

2. On Login node, change directory to global scratch:
   ```bash
   cd /lustre1/MyID
   ```

3. Create a working subdirectory for a job:
   ```bash
   mkdir ./workDir
   ```

4. Change directory to `workDir`:
   ```bash
   cd ./workDir
   ```

5. Transfer data from local computer to `workDir`:
   - Use `scp` or `SSH File Transfer` to connect Transfer node
   - Transfer data on cluster to `workDir`:
     - Log on to Transfer node and then use `cp` or `mv`

6. Make a job submission script in `workDir`:
   ```bash
   nano ./sub.sh
   ```

7. Submit a job from `workDir`:
   ```bash
   qsub ./sub.sh
   ```

8. Check job status or Cancel a job:
   ```bash
   qstat_me or qdel JobID
   ```
Step 1: Log on to Login node - Mac/Linux using ssh

1. Open Terminal utility

2. Type command line: `ssh MyID@sapelo2.gacrc.uga.edu`

3. You will be prompted for your MyID password

4. Sapelo2 access requires ID verification using two-factor authentication with Archpass Duo.

   If you are not enrolled in Archpass Duo, please refer to

   [https://eits.uga.edu/access_and_security/infosec/tools/archpass_duo/](https://eits.uga.edu/access_and_security/infosec/tools/archpass_duo/) on how to enroll

   [https://wiki.gacrc.uga.edu/wiki/Connecting](https://wiki.gacrc.uga.edu/wiki/Connecting)
Step 1 (Cont.) - Mac/Linux

Use Terminal utility on Mac or Linux!

Log on

Input MyID password!

Select Duo login option!

I am on sapelo2-sub2!
Step1 (Cont.) - Windows

1. Download and install SSH Secure Utilities: [http://eits.uga.edu/hardware_and_software/software/](http://eits.uga.edu/hardware_and_software/software/)

2. You can use PuTTY as an alternative: [https://www.putty.org/](https://www.putty.org/)
You will be prompted for your MyID password after step 4.

5. Select Duo login option!
Step2: On Login node change directory to global scratch

- Once you logged on, your current directory will be your **home directory**

```
zhuofei@sapelo2-sub2 ~$ pwd
/home/zhuofei
```

- Use `cd` command to change your current directory to `/lustre1/MyID`

```
zhuofei@sapelo2-sub2 /$ cd /lustre1/zhuofei
zhuofei@sapelo2-sub2 zhuofei$ pwd
/lustre1/zhuofei
```

- Use `ls` command to take a look in `/lustre1/MyID`

```
zhuofei@sapelo2-sub2 zhuofei$ ls
user_test  workDir_Alex  workDir_bk
```
Step3 - 4: Create and cd to a working subdirectory

• Use `mkdir` command to make a subdirectory in `/lustre1/MyID`

```
zhuofei@sapelo2-sub2 zhuofei$ mkdir workDir
zhuofei@sapelo2-sub2 zhuofei$ ls
user_test workDir workDir_Alex workDir_bk
```

• Use `cd` command to change your current directory to `/lustre1/MyID/workDir`

```
zhuofei@sapelo2-sub2 zhuofei$ cd workDir
zhuofei@sapelo2-sub2 workDir$ ls
```

← it is empty!
Step 5: Transfer data from local computer to workDir - Mac/Linux

1. You need to connect to cluster’s Transfer node (xfer.gacrc.uga.edu)

2. Open Terminal utility on local computer to use scp (-r) [Source] [Target]

E.g. 1: working on local computer, from Local ➔ workDir on cluster

```
scp ./file zhuofei@xfer.gacrc.uga.edu:/lustre1/zhuofei/workDir/
scp -r ./folder/ zhuofei@xfer.gacrc.uga.edu:/lustre1/zhuofei/workDir/
```

E.g. 2: working on local computer, from workDir on cluster ➔ Local

```
scp zhuofei@xfer.gacrc.uga.edu:/lustre1/zhuofei/workDir/file .
scp -r zhuofei@xfer.gacrc.uga.edu:/lustre1/zhuofei/workDir/folder/ .
```
Step 5 (Cont.) - Windows

1. You need to connect to cluster’s **Transfer node** (xfer.gacrc.uga.edu)

2. Use **SSH File Transfer** or FileZilla or WinSCP on local computer.

   ![Diagram showing steps 5 and 6 of the process.]

   **You will be prompted for your MyID password after step 4.**
Step 5 (Cont.): Transfer data on cluster to workDir

- Log on to Transfer node (xfer.gacrc.uga.edu)
  - Mac/Linux: ssh MyID@xfer.gacrc.uga.edu (page 12, 13)
  - Windows: use SSH Secure Client app (page 14, 15)
- Landing folder: /home/MyID (Home)
- Transfer data between folders on cluster using cp, mv
- Directories you can access using full path:
  1. /home/MyID
  2. /lustre1/MyID
  3. /project/abclab
- Most file systems on Transfer are auto-mounted upon the first time full-path access, e.g.,
  cd /project/abclab/
Step 6: Make a job submission script in workDir using nano

$ nano sub.sh

nano is a small and friendly text editor on Linux.

Ctrl-x to save file and quit from nano
Step6 (Cont.)

1. Copy a sample script from GACRC Wiki Software:
   https://wiki.gacrc.uga.edu/wiki/Bowtie2-Sapelo2
2. Modify it as needed for your computing

To run this example, you need to copy input data to your current working folder:

```bash
cp -r /usr/local/training/index .
cp /usr/local/training/myreads.fq .
```

```
#PBS -S /bin/bash
#PBS -q batch
#PBS -N bowtie2_test
#PBS -l nodes=1:ppn=1:Intel
#PBS -l mem=2gb
#PBS -l walltime=1:00:00
#PBS -M zhuofei@uga.edu
#PBS -m ae

cd $PBS_O_WORKDIR

module load Bowtie2/2.3.3-foss-2016b

time bowtie2 -p 1 -x ./index/lambda_virus -U ./myreads.fq -S output.sam
```
Step 7: Submit a job from workDir using qsub

```
zhuofei@sapelo2-sub2 workDir$ pwd
/lustre1/zhuofei/workDir
zhuofei@sapelo2-sub2 workDir$ ls
index myreads.fq sub.sh
zhuofei@sapelo2-sub2 workDir$ qsub sub.sh
11943.sapelo2
```

**sub.sh** is job submission script to

1. specify computing resources:
2. load software using `ml load`
3. run any Linux commands you want to run
4. run the software
Step 8: Check job status using qstat_me

<table>
<thead>
<tr>
<th>Job ID</th>
<th>Name</th>
<th>User</th>
<th>Time Use</th>
<th>S</th>
<th>Queue</th>
</tr>
</thead>
<tbody>
<tr>
<td>11943.sapelo2</td>
<td>bowtie2_test</td>
<td>zhuofei</td>
<td>00:06:40</td>
<td>C</td>
<td>batch</td>
</tr>
<tr>
<td>11944.sapelo2</td>
<td>bowtie2_test</td>
<td>zhuofei</td>
<td>00:05:17</td>
<td>R</td>
<td>batch</td>
</tr>
<tr>
<td>11946.sapelo2</td>
<td>bowtie2_test</td>
<td>zhuofei</td>
<td>00:12:51</td>
<td>R</td>
<td>batch</td>
</tr>
<tr>
<td>11947.sapelo2</td>
<td>bowtie2_test</td>
<td>zhuofei</td>
<td>0</td>
<td>R</td>
<td>batch</td>
</tr>
<tr>
<td>11948.sapelo2</td>
<td>bowtie2_test</td>
<td>zhuofei</td>
<td>0</td>
<td>Q</td>
<td>batch</td>
</tr>
</tbody>
</table>

- **R**: job is running
- **C**: job completed (or canceled or crashed) and is not longer running. Jobs stay in this state for 24 hours.
- **Q**: job is pending, waiting for resources to become available.

Note: “Time Use” is the CPU time, instead of the wall-clock time of your job staying on cluster!
Step8 (Cont.): Cancel job using `qdel`

```
zhuofei@sapelo2-sub2 workDir$ qdel 11947
zhuofei@sapelo2-sub2 workDir$ qstat_me
```

<table>
<thead>
<tr>
<th>Job ID</th>
<th>Name</th>
<th>User</th>
<th>Time Use</th>
<th>S</th>
<th>Queue</th>
</tr>
</thead>
<tbody>
<tr>
<td>11943.sapelo2</td>
<td>bowtie2_test</td>
<td>zhuofei</td>
<td>00:06:40</td>
<td>C</td>
<td>batch</td>
</tr>
<tr>
<td>11944.sapelo2</td>
<td>bowtie2_test</td>
<td>zhuofei</td>
<td>00:05:17</td>
<td>R</td>
<td>batch</td>
</tr>
<tr>
<td>11946.sapelo2</td>
<td>bowtie2_test</td>
<td>zhuofei</td>
<td>00:12:51</td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>11947.sapelo2</td>
<td>bowtie2_test</td>
<td>zhuofei</td>
<td>00:00:09</td>
<td>C</td>
<td>batch</td>
</tr>
<tr>
<td>11948.sapelo2</td>
<td>bowtie2_test</td>
<td>zhuofei</td>
<td>Q</td>
<td>Q</td>
<td>batch</td>
</tr>
</tbody>
</table>

job 11947 status is changed from R to C
C status will stay in list for 24 hour
Step8 (Cont.): Check Job using `qstat -n -u MyID`

```
zhuofei@sapelo2-sub2 workDir$ qstat -n -u zhuofei
```

```
dispatch.ecompute:

<table>
<thead>
<tr>
<th>Job ID</th>
<th>Username</th>
<th>Queue</th>
<th>Jobname</th>
<th>SessID</th>
<th>NDS</th>
<th>TSK</th>
<th>Req'd Memory</th>
<th>Req'd Time</th>
<th>Elap Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>12175.sapelo2 zhuofei batch bowtie2_test</td>
<td>132442</td>
<td>1</td>
<td>2gb</td>
<td>01:00:00</td>
<td>R</td>
<td>00:23:44</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>n238/0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>16</td>
<td>02:00:00</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12176.sapelo2 zhuofei batch bowtie2_test</td>
<td>67226</td>
<td>1</td>
<td>2gb</td>
<td>01:00:00</td>
<td>R</td>
<td>00:20:44</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>n237/0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>16</td>
<td>02:00:00</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12177.sapelo2 zhuofei batch bowtie2_test</td>
<td>119643</td>
<td>1</td>
<td>2gb</td>
<td>01:00:00</td>
<td>R</td>
<td>00:05:44</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

Note: “Elap Time” is the wall-clock time, instead of the CPU time, which `qstat_me` can give you!
Step 8 (Cont.): Check all Jobs on cluster using `qstat`.

```
zhuofei@sapelo2-sub2 workDir$ qstat
Job ID                   Name             User            Time Use S Queue
-----------------------------------------------------------------------------------
11267.sapelo2            L80-500          jx57780         164:32:5 R batch
11269.sapelo2            L80-502          jx57780         164:55:5 C batch
11270.sapelo2            L80-503          jx57780         165:38:5 C batch
11607.sapelo2            canu              gd98309         3414:46: R bergman_q
11726.sapelo2            ..._3_constoptTS sm39091         3157:30: R wheeler_q
11729.sapelo2            ..._2_constoptTS sm39091         2731:29: R wheeler_q
11790.sapelo2            sp2_run19b_dye castelao         4412:52: C batch
11804.sapelo2            ...e-4_Nnoise=64 cotter             98:26:20 R batch
11806.sapelo2            ...e-4_Nnoise=64 cotter             98:14:22 R batch
11987.sapelo2            ..._th_W18-T5-L4 qbcg              08:02:40 C batch
11989.sapelo2            matlabjob         zhyw86               0 C batch
11990.sapelo2            ..._1_constoptTS sm39091         445:42:1 R wheeler_q
11991.sapelo2            ..._1_constoptTS sm39091         444:51:4 R wheeler_q
11992.sapelo2            ..._cl_W18-T3-D1 qbcg              03:04:21 C batch
```

Note: `qstat` command will give you a long list of all jobs from all users on cluster!
1. Linux/Mac user:
   ssh MyID@sapelo1.gacrc.uga.edu

Windows user:

5. Linux/Mac user:
   scp file MyID@xfer.gacrc.uga.edu:/lustre1/MyID/workDir

Windows user:

2. cd /lustre1/MyID

3. mkdir ./workDir

4. cd ./workDir

6. nano ./sub.sh

$ qsub sub.sh

#PBS -S /bin/bash
#PBS -q batch
#PBS -N bowtie2_test
#PBS -l nodes=1:ppn=1:Intel
#PBS -l mem=2gb
#PBS -l walltime=1:00:00
#PBS -M zhuofei@uga.edu
#PBS -m ae

cd $PBS_O_WORKDIR

module load Bowtie2/2.3.3

7. $ qsub sub.sh

8. $ qstat_me or qdel JobID

GACRC SAPELO2 CLUSTER NEW USER TRAINING

3/23/2018
Work on Sapelo2 - GACRC Wiki

Main Page: http://wiki.gacrc.uga.edu
Running Jobs: https://wiki.gacrc.uga.edu/wiki/Running_Jobs_on_Sapelo2
Software: https://wiki.gacrc.uga.edu/wiki/Software
Transfer File: https://wiki.gacrc.uga.edu/wiki/Transferring_Files
Linux Command: https://wiki.gacrc.uga.edu/wiki/Command_List
Training: https://wiki.gacrc.uga.edu/wiki/Training
User Account Request: https://wiki.gacrc.uga.edu/wiki/User_Accounts
Support: https://wiki.gacrc.uga.edu/wiki/Getting_Help
Get Help

➤ **Job Troubleshooting:**

Please tell us details of your question or problem, including but not limited to:

- Your user name
- Your job ID
- Your working directory
- The queue name and command you used to submit the job

➤ **Software Installation:**

- Specific name and version of the software
- Download website
- Supporting package information if have

Please note to make sure the correctness of datasets being used by your jobs!
Request Support

* indicates Required fields.

Your Name *  
John smith

MyID *  
jsmith

E-mail *  
jsmith@uga.edu

Phone Number  
411-555-1212

Brief Description  
my job can not be started

Request Details *

* For questions on cluster or software, please include the command/script used, working path and working nodes (interactive / queue name) if applicable.

* For software installation, please specify software name, version and include link to the software if applicable.

* Please review your message on the next page and then click the Submit button.
Appendix - Run Batch Jobs

- Components you need to run a job:
  - Software already installed (cluster software or the one installed by yourself)
  - Job submission script to
    1. specify computing resources:
       - number of nodes and cores
       - amount of memory
       - node’s feature
       - maximum wallclock time
    2. load software using `ml load` (for cluster software)
    3. run any Linux commands you want to run, e.g., `pwd`, `mkdir`, `cd`, `echo`, etc.
    4. run the software

- Common queueing commands you need:
  - `qsub`, `qstat_me`, `qstat`, `qdel`
  - `qstat -f`, `showq`
Example 1: **Serial job script running NCBI Blast+ using 1 CPU**

```
#PBS -S /bin/bash
#PBS -q batch
#PBS -N testBlast
#PBS -l nodes=1:ppn=1:Intel
#PBS -l mem=20gb
#PBS -l walltime=48:00:00

cd $PBS_O_WORKDIR

ml load BLAST+/2.6.0-foss-2016b-Python-2.7.14

time blastn [options] ...
```

- Linux default shell (bash)
- Queue name (batch)
- Job name (testBlast)
- Number of nodes (1), number of cores (1), node type (Intel)
- Maximum amount of RAM memory (20 GB) used by the job
- Maximum wall-clock time (48 hours) for the job, default 6 minutes

- Compute node will use the directory from which the job is submitted as the working directory, i.e., /lustre1/MyID/workDir

- Load the module of ncbiblast+, version 2.6.0

- Run blastn with ‘time’ command to measure the amount of time it takes to run the application

https://wiki.gacrc.uga.edu/wiki/BLAST%2B-Sapelo2
*Example 2: Threaded job script running NCBI Blast+ using 4 CPUS

```
#PBS -S /bin/bash
#PBS -q batch
#PBS -N testBlast
#PBS -l nodes=1:ppn=4:Intel
#PBS -l mem=20gb
#PBS -l walltime=480:00:00

#PBS -M jsmith@uga.edu
#PBS -m ae
#PBS -j oe

cd $PBS_O_WORKDIR

ml load BLAST+/2.6.0-foss-2016b-Python-2.7.14

time blastn -num_threads 4 [options] ...
```

- Number of nodes (1), number of cores (4), node type (AMD)
  \[Number of cores requested (4) = Number of threads (4)\]

- Email address to receive a notification for computing resources
- Send email notification when job aborts (a) or terminates (e)
- Standard error file (testBlast.e12345) will be merged into standard out file (testBlast.o12345)

- Run blastn with 4 threads (-num_threads 4)
**Example 3: MPI job script running RAxML using 2 full nodes**

```bash
#PBS -S /bin/bash
#PBS -q batch
#PBS -N testRAxML
#PBS -l nodes=2:ppn=28:Intel
#PBS -l walltime=120:00:00
#PBS -l mem=100gb

# Number of nodes (2), number of cores (28), node type (Intel)
# Total cores requested = 2 × 28 = 56
# We suggest, Number of MPI Processes (50) ≤ Number of cores requested (56)

cd $PBS_O_WORKDIR

ml load RAxML/8.2.11-foss-2016b-mpi-avx

# To run raxmlHPC-MPI-AVX, MPI version using OpenMPI
mpirun --np 50 raxmlHPC-MPI-AVX [options]

# Run raxmlHPC-MPI-AVX with 50 MPI processes (--np 50), default 56
```
Thank You!