Introduction to GACRC Sapelo2 Cluster

jlmlab group

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

- What You Need to Know about Sapelo2 Cluster
- Work on Sapelo2 Cluster
- Appendix
- Guideline Tips
What You Need to Know about Sapelo2 Cluster

1. Cluster Overview

2. Storage Environment

3. Computing Resources

4. Software Environment
INTRODUCTION TO GACRC SAPELO2 CLUSTER

**Sapelo2 Cluster**

1. **Login Node**
   - **Login Node** (sapelo2.gacrc.uga.edu)
   - **Command**: `ssh
   - **Home**
     - `/home/MyID`
     - **100 GB**
     - **No quota limit**
   - **Global Scratch**
     - `/lustre1/MyID`
     - **1 TB**

2. **Transfer Node**
   - **Transfer Node** (xfer.gacrc.uga.edu)

3. **Compute Nodes**
   - **Computational Queues**
     - **batch**
       - For regular jobs
     - **highmem_q**
       - For high memory jobs
     - **gpu_q**
       - For GPU-enabled jobs
     - **jim_q**
       - For jmlab

4. **Node**: Computer for a specific function on cluster, e.g., login node
5. **Queue**: Collection of **compute nodes** for specific computing need
6. **Cluster**: **Nodes + Drives**, all connected by network
Cluster Overview

1. Sapelo2 cluster is a Linux (64-bit CentOS 7) high performance computing (HPC) cluster

2. You can log on to 2 nodes: Login node (sapelo2.gacrc.uga.edu) and Transfer node (xfer.gacrc.uga.edu)

3. From Login node, you can open Interactive node using `qlogin` command

4. You have 4 directories: Home, Global Scratch, Storage and Local Scratch

5. You can submit jobs to 4 computational queues: batch, highmem_q, gpu_q, and jlm_q

6. You can use more than 400 modules installed on cluster (as of 05/17/2018)
# Storage Environment – 4 Directories

<table>
<thead>
<tr>
<th>4 Directories</th>
<th>Role</th>
<th>Quota</th>
<th>Accessible from</th>
<th>Intended Use</th>
<th>Backuped</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>/home/MyID</td>
<td>Home</td>
<td>100GB</td>
<td>Login</td>
<td>Static data: 1. Scripts, source codes 2. Local software</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Transfer Interactive</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/lustre1/MyID</td>
<td>Global Scratch</td>
<td>No Limit</td>
<td>Interactive</td>
<td>Current job data: data being read/written by running jobs</td>
<td>No</td>
<td>User to clean up! *Subject to deletion in 30 days</td>
</tr>
<tr>
<td>/project/abclab</td>
<td>Storage</td>
<td>1TB (initial)</td>
<td>Transfer</td>
<td>Temporary data parking: non-current active data</td>
<td>Yes</td>
<td>Group sharing possible</td>
</tr>
<tr>
<td>/scratch</td>
<td>Local Scratch</td>
<td>200GB (regular)</td>
<td>Compute</td>
<td>Jobs with heavy disk I/O</td>
<td>No</td>
<td>1. User to clean up when job exists from the node! 2. Persistent data</td>
</tr>
</tbody>
</table>
Storage Environment (Cont.) – Accessing Directories from Nodes

- Login
  - Log on using ssh
  - qlogin
  - exit

- Interactive
  - /home/MyID
  - /lustre1/MyID

- Transfer
  - /project/abclab

User

log on using ssh
Storage Environment (Cont.) - Intended Use of Directories

I/O speed

Fast

/lustre1/MyID  \(\leftarrow\) **Current Job Data** being used by **current running jobs** on cluster

/home/MyID  \(\leftarrow\) **Static Data** being used frequently and **not being modified often**, e.g., scripts or local software

Slow

/project/abclab  \(\leftarrow\) **Non-current Active Data** to be analyzed in the future, e.g., 2 months

User’s Local Computer  \(\leftarrow\) **Final Data**, e.g., final outputs and results
## Computing Resources – 4 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>
<th>InfiniBand</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>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>AMD</td>
<td>40</td>
<td>128</td>
<td>125</td>
<td>48</td>
<td>AMD Opteron</td>
<td></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>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>AMD</td>
<td>2</td>
<td>512</td>
<td>503</td>
<td>48</td>
<td>AMD Opteron</td>
<td></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>Intel Xeon</td>
<td>7 NVIDIA K20Xm</td>
<td></td>
</tr>
<tr>
<td>jlm_q</td>
<td>AMD</td>
<td>2 (n513/514)</td>
<td>128</td>
<td>125</td>
<td>48</td>
<td>AMD Opteron</td>
<td>N/A</td>
<td></td>
</tr>
</tbody>
</table>

**INTRODUCTION TO GACRC SAPELO2 CLUSTER**

5/18/2018
Software Environment

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)
Work on Sapelo2 Cluster

1. Job Submission Workflow

2. How to Know Job Details

3. How to Know Node Details

4. qlogin Commands: Open Interactive Node for Running Interactive Tasks
Job Submission Workflow

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

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

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

4. Change directory to `workDir`:
   `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`:
   `nano ./sub.sh`

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

8. Check job status:
   `qstat_me` or Cancel a job:
   `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)
Use Terminal utility on Mac or Linux!

1. Enter a passcode or select one of the following options:
   - Duo Push to XXX-XXX-5758
   - Phone call to XXX-XXX-1925
   - Phone call to XXX-XXX-5758
   - SMS passcodes to XXX-XXX-5758

2. Passcode or option (1-4): 1

   Pushed a login request to your device...

   Success. Logging you in...

   zhuofei@apelo2-sub2 ~$

   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/)
Step 1 (Cont.) - Windows

You will be prompted for your MyID password after step 4.

5. Select Duo login option!
Step 2: 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
  ➜ this is my home directory!
  ```

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

  ```
  zhuofei@sapelo2-sub2 /$ cd /lustre1/zhuofei
  zhuofei@sapelo2-sub2 zhuofei$ pwd
  /lustre1/zhuofei
  ➜ this is my global scratch folder!
  ```

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

  ```
  zhuofei@sapelo2-sub2 zhuofei$ ls
  user_test  workDir_Alex  workDir_bk
  ```
Step 3 - 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:/lustrel/zhuofei/workDir/
scp -r ./folder/ zhuofei@xfer.gacrc.uga.edu:/lustrel/zhuofei/workDir/
```

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

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

You will be prompted for your MyID password after step 4.
Step5 (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
```

Ctrl-x to save file and quit from nano

nano is a small and friendly text editor on Linux.
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:

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

```bash
#PBS -S /bin/bash
#PBS -q batch
#PBS -N bowtie2_test
#PBS -l nodes=1:ppn=1
#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
```

- node feature (AMD or Intel) is NOT needed!
- memory requested is enforced on cluster!
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`

```
zhufei@sapelo2-sub2 workDir$ qstat_me
Job ID       Name     User   Time Use S  Queue
------------- ------------- ------- -------- --- ----
11943.sapelo2 bowtie2_test zhuofei 00:06:40 C  batch
11944.sapelo2 bowtie2_test zhuofei 00:05:17 R  batch
11946.sapelo2 bowtie2_test zhuofei 00:12:51 R  batch
11947.sapelo2 bowtie2_test zhuofei 0  R  batch
11948.sapelo2 bowtie2_test zhuofei 0  Q  batch
```

**R**: job is running  
**C**: job completed (or canceled or crashed) and is not longer running. Jobs stay in this state for 24 hour  
**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!
Step 8 (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>batch</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>00:00:09</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></td>
<td></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></td>
<td></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`

<table>
<thead>
<tr>
<th>Job ID</th>
<th>Name</th>
<th>User</th>
<th>Time Use S Queue</th>
</tr>
</thead>
<tbody>
<tr>
<td>11267.sapelo2</td>
<td>L80-500</td>
<td>jx57780</td>
<td>164:32:5 R batch</td>
</tr>
<tr>
<td>11269.sapelo2</td>
<td>L80-502</td>
<td>jx57780</td>
<td>164:55:5 C batch</td>
</tr>
<tr>
<td>11270.sapelo2</td>
<td>L80-503</td>
<td>jx57780</td>
<td>165:38:5 C batch</td>
</tr>
<tr>
<td>11607.sapelo2</td>
<td>canu</td>
<td>gd98309</td>
<td>3414:46: R bergman_q</td>
</tr>
<tr>
<td>11726.sapelo2</td>
<td>..._3_consoptTS</td>
<td>sm39091</td>
<td>3157:30: R wheeler_q</td>
</tr>
<tr>
<td>11729.sapelo2</td>
<td>..._2_consoptTS</td>
<td>sm39091</td>
<td>2731:29: R wheeler_q</td>
</tr>
<tr>
<td>11790.sapelo2</td>
<td>sp2_run19b_dye</td>
<td>castelao</td>
<td>4412:52: C batch</td>
</tr>
<tr>
<td>11804.sapelo2</td>
<td>...e-4_Nnoise=64</td>
<td>cotter</td>
<td>98:26:20 R batch</td>
</tr>
<tr>
<td>11806.sapelo2</td>
<td>...e-4_Nnoise=64</td>
<td>cotter</td>
<td>98:14:22 R batch</td>
</tr>
<tr>
<td>11987.sapelo2</td>
<td>..._th_W18-T5-L4</td>
<td>qbcg</td>
<td>08:02:40 C batch</td>
</tr>
<tr>
<td>11989.sapelo2</td>
<td>matlabjob</td>
<td>zhyw86</td>
<td>0 C batch</td>
</tr>
<tr>
<td>11990.sapelo2</td>
<td>..._1_consoptTS</td>
<td>sm39091</td>
<td>445:42:1 R wheeler_q</td>
</tr>
<tr>
<td>11991.sapelo2</td>
<td>..._1_consoptTS</td>
<td>sm39091</td>
<td>444:51:4 R wheeler_q</td>
</tr>
<tr>
<td>11992.sapelo2</td>
<td>..._cl_W18-T3-D1</td>
<td>qbcg</td>
<td>03:04:21 C batch</td>
</tr>
</tbody>
</table>

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`

   ```bash
   #PBS -S /bin/bash
   #PBS -q batch
   #PBS -N bowtie2_test
   #PBS -l nodes=1:ppn=1
   #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
   cd $PBS_O_WORKDIR
   ```

7. `qsub sub.sh`

8. `$ qstat_me or qdel JobID`
How to Know Job Details

Option 1: `qstat -f JobID` for running jobs or finished jobs in 24 hours

Option 2: Email notification from finished jobs (completed, canceled, or crashed), if using:

```
#PBS -M MyID@uga.edu
#PBS -m ae
```
Option 1: qstat -f JobID (running jobs or finished jobs in 24 hour)

$ qstat -f 12222
Job Id: 12222.sapelo2
  Job_Name = testBlast
  Job_Owner = zhuofei@10.56.200.51
    resources_used.cput = 00:00:00
    resources_used.vmem = 316864kb
    resources_used.walltime = 00:15:01
    resources_used.mem = 26780kb
    resources_used.energy_used = 0
  job_state = C
  queue = batch
  Error_Path = sapelo2-sub2.ecompute:/lustre1/zhuofei/examples/testBlast.e12222
  exec_host = n236/0-3
  Output_Path = sapelo2-sub2.ecompute:/lustre1/zhuofei/examples/testBlast.o12222

  Resource_List.nodes = 1:ppn=4:Intel
  Resource_List.mem = 20gb
  Resource_List.walltime = 02:00:00
  Resource_List.nodect = 1

  Variable_List = PBS_O_QUEUE=batch,PBS_O_HOME=/home/zhuofei,......
                   PBS_O_WORKDIR=/lustre1/zhuofei/workDir,
Option 2: Email notification from finished jobs

BS Job Id: 12332.sapelo2       Job Name: bowtie2_test
Queue: batch
Exechost: n232/0

Message: Execution terminated
Details:
Exit_status=0
resources_used.cput=00:09:26
resources_used.vmem=755024kb
resources_used.walltime=00:09:51
resources_used.mem=1468676kb
resources_used.energy_used=0

Short reason:
Execution terminated

PBS Job Id: 12331.sapelo2       Job Name: bowtie2_test
Queue: batch
Exechost: n235/0

Message: Execution terminated
Details:
Exit_status=271
resources_used.cput=00:02:58
resources_used.vmem=755024kb
resources_used.walltime=00:03:24
resources_used.mem=420712kb
resources_used.energy_used=0

Short reason:
Execution terminated

Sender: dispatch_root
How to Know Node Details

**Option 1:** `mdiag -v -n | grep [pattern] | ...
mdiag -v -n | grep batch | grep AMD
mdiag -v -n | grep batch | grep Intel
mdiag -v -n | grep highmem_q
mdiag -v -n | grep jlm_q`

**Option 2:** from login node, you can ssh to a compute node and run a command there!

```
ssh n514 'lscpu'
ssh n514 'free -g'
ssh n514 "ps aux | grep '^jlm10'"
```
qlogin Commands

1. Type qlogin commands from Login node to open Interactive node:
   - `qlogin_intel`: Start an interactive session on an Intel node
   - `qlogin_amd`: Start an interactive session on an AMD node
   - `qlogin`: start an interactive job on either types of nodes

2. Type `exit` command to quit and back to Login node
qlogin Commands

Purpose1: Open interactive node for running interactive tasks of R, Python, Bash scripts, etc.

zhuofei@sapelo2-sub1 ~$ qlogin
qsub: waiting for job 12426.sapelo2 to start
qsub: job 12426.sapelo2 ready

zhuofei@n204 ~$ ml spider R
---------------------------------------------------------------
  R: R/3.4.1-foss-2016b-X11-20160819-GACRC
---------------------------------------------------------------
...

zhuofei@n204 ~$ ml R/3.4.1-foss-2016b-X11-20160819-GACRC
zhuofei@n204 ~$ R
R version 3.4.1 (2017-06-30) -- "Single Candle"
...

[Previously saved workspace restored]
> a<-1 ; b<-7
> a+b
[1] 8
>
qlogin Commands

Purpose: Open interactive node for compiling/testing source codes of Fortran, C/C++, Python, etc.

zhuofei@sapelo2-sub1 ~$ qlogin_intel
qsub: waiting for job 20912.sapelo2 to start
qsub: job 20912.sapelo2 ready

zhuofei@n206 ~$ ml spider iomkl
-----------------------------------------
iomkl:
----------------------------------------
Description:
Intel Cluster Toolchain Compiler Edition provides Intel C/C++ and Fortran compilers, Intel MKL & OpenMPI.

Versions:
iomkl/2018a
...

zhuofei@n206 ~$ ml iomkl/2018a
zhuofei@n206 ~$ icc mysource.c -o myexec.x
zhuofei@n206 ~$
Appendix: Examples of Running Batch Serial and Threaded 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
     ✓ 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
Example 1: **Serial job script running NCBI Blast+ using 1 CPU**

```
#PBS -S /bin/bash
#PBS -q jlm_q
#PBS -N testBlast
#PBS -l nodes=1:ppn=1
#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 (jlm_q)**
- **Job name (testBlast)**
- **Number of nodes (1), number of cores (1), node feature is Not needed!**
- **Maximum amount of RAM memory (20 GB) is enforced!**
- **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](https://wiki.gacrc.uga.edu/wiki/BLAST%2B-Sapelo2)
Example 2: **Threaded job script** running NCBI Blast+ using **4 CPUS**

```bash
#PBS -S /bin/bash
#PBS -q jlm_q
#PBS -N testBlast
#PBS -l nodes=1:ppn=4
#PBS -l mem=20gb
#PBS -l walltime=120:00:00
#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)
  
  \[\text{Number of cores requested (4)} = \text{Number of threads (4)}\]

- 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)
Guideline Tips

• Do NOT use Login node to run CPU/memory intensive jobs directly ➔ submit jobs to queue!
• Do NOT use Login Node to upload/download large data to/from cluster ➔ use Transfer node!
• Do NOT use home dir for storing large job data ➔ use global scratch /lustre1/MyID
• Do NOT park data on global or local scratch ➔ clean up when job finished or exits from node
• NO large memory job running on batch or jlm_q ➔ use highmem_q
• NO small memory job running on highmem_q ➔ use batch or jlm_q
• As a general rule, threads # = cores # requested