Introduction to Sapelo Computing Resources at GACRC

Georgia Advanced Computing Resource Center (GACRC)
EITS/University of Georgia

Zhuofei Hou
2016-9-6

Outline

- GACRC
- Sapelo at GACRC
- Sapelo workflow
- Work on Sapelo

GACRC

Who Are We?

- **G**eorgia **A**dvanced **C**omputing **R**esource **C**enter
- Collaboration between the Office of Vice President for Research (**OVPR**) and the Office of the Vice President for Information Technology (**OVPIT**)
- Guided by a faculty advisory committee (GACRC-AC)

Why Are We Here?

- To provide computing hardware and network infrastructure in support of high-performance computing (**HPC**) at UGA

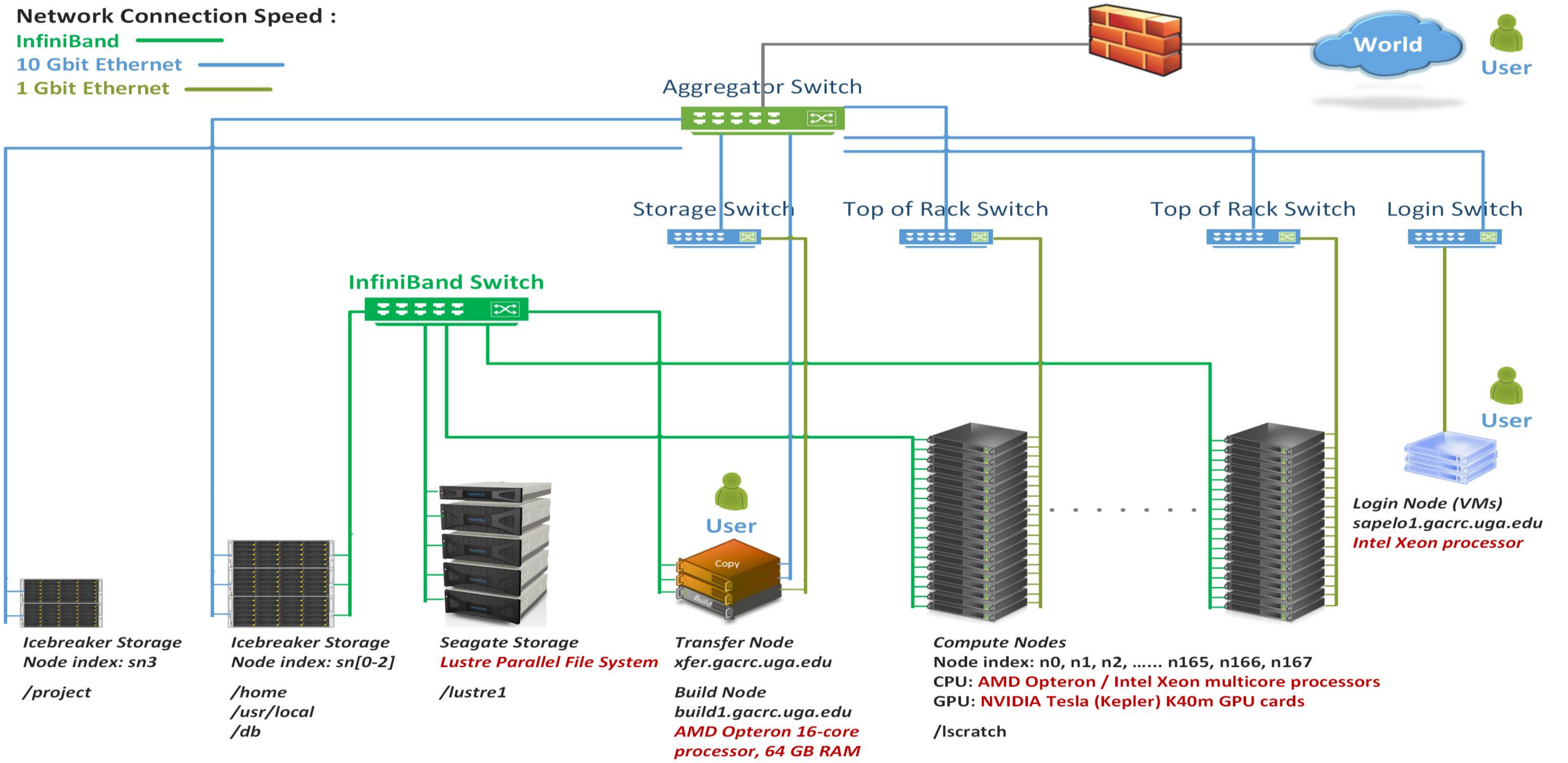
Where Are We?

- http://gacrc.uga.edu (Web)
- http://wiki.gacrc.uga.edu (Wiki)
- http://gacrc.uga.edu/help/ (Web Help)
- https://wiki.gacrc.uga.edu/wiki/Getting_Help (Wiki Help)

Sapelo at GACRC

- Cluster Structural Diagram
- Cluster Overview
- Computing Resources
- Storage Environment

The New GACRC Linux HPC Cluster Structural Diagram



Cluster Overview

Sapelo is a Linux high-performance computing (HPC) cluster:

- OS: 64-bit CentOS Linux 6.5
- You can log on to:

  Login (UGAMyID@sapelo1.gacrc.uga.edu) : edit script, submit jobs

  Transfer (UGAMyID@xfer.gacrc.uga.edu) : transfer, compress, package data

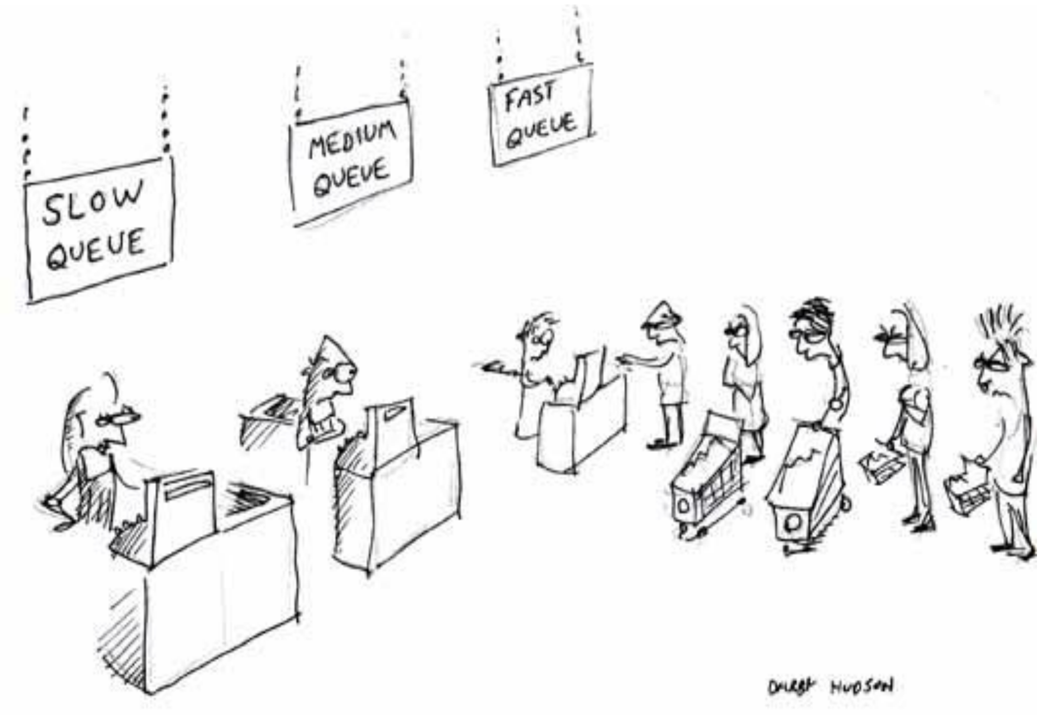
  Build (UGAMyID@build1.gacrc.uga.edu) : compile, test

- Login $\xrightarrow{\texttt{qlogin}}$ Interactive Node : run job interactively, edit script, submit jobs

Cluster Overview

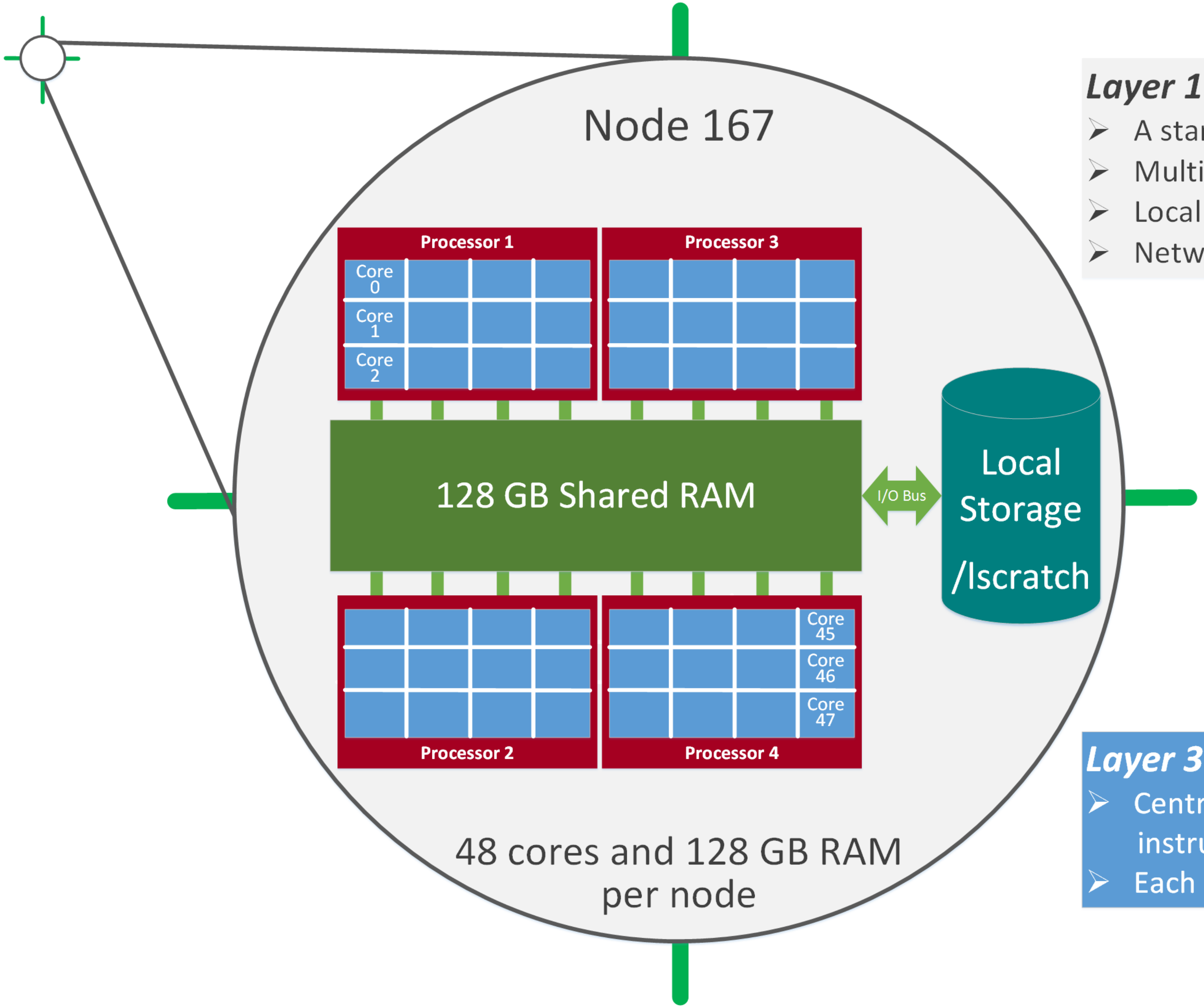
- Batch-queueing System:
  - ➢ Jobs can be started (submitted), monitored, and controlled
  - ➢ Determine which compute node is the best place to run a job
  - ➢ Determine appropriate execution priority for a job to run

- On Sapelo:
  - ➢ Torque Resource Manager + Moab Workload Manager
  - ➢ Queueing commands: `qsub`, `qstat`, `qdel` `showjobs`, `showq`



Computing Resources

Queue	Node Feature	Total	Processor	Cores /Node	RAM (GB) /Node	Max RAM (GB) /Single-node Job	GPU	GPU Cards /Node	InfiniBand
batch	AMD	112	AMD Opteron	48	128	126	N/A	N/A	Yes
		4			256	252			
	HIGHMEM	7	AMD Opteron	48	512 (6)	504	N/A	N/A	Yes
					1024 (1)	997			
	GPU	2	Intel Xeon	16	128	126	NVIDIA K40m	8	Yes
	abcnode (buy-in)	2	AMD Opteron	48	256	252	N/A	N/A	Yes



Layer 1: Node

- A standalone “computer in a box”
- Multiple processors, e.g. 4, sharing memory
- Local disk storage, network interface, etc.
- Networked into a cluster

Layer 2: Processor

- A single computing component
- Multicore processor, e.g. 12 cores

Layer 3: Core

- Central processing unit (CPU) reading and executing instructions independently
- Each core is assigned to a software thread

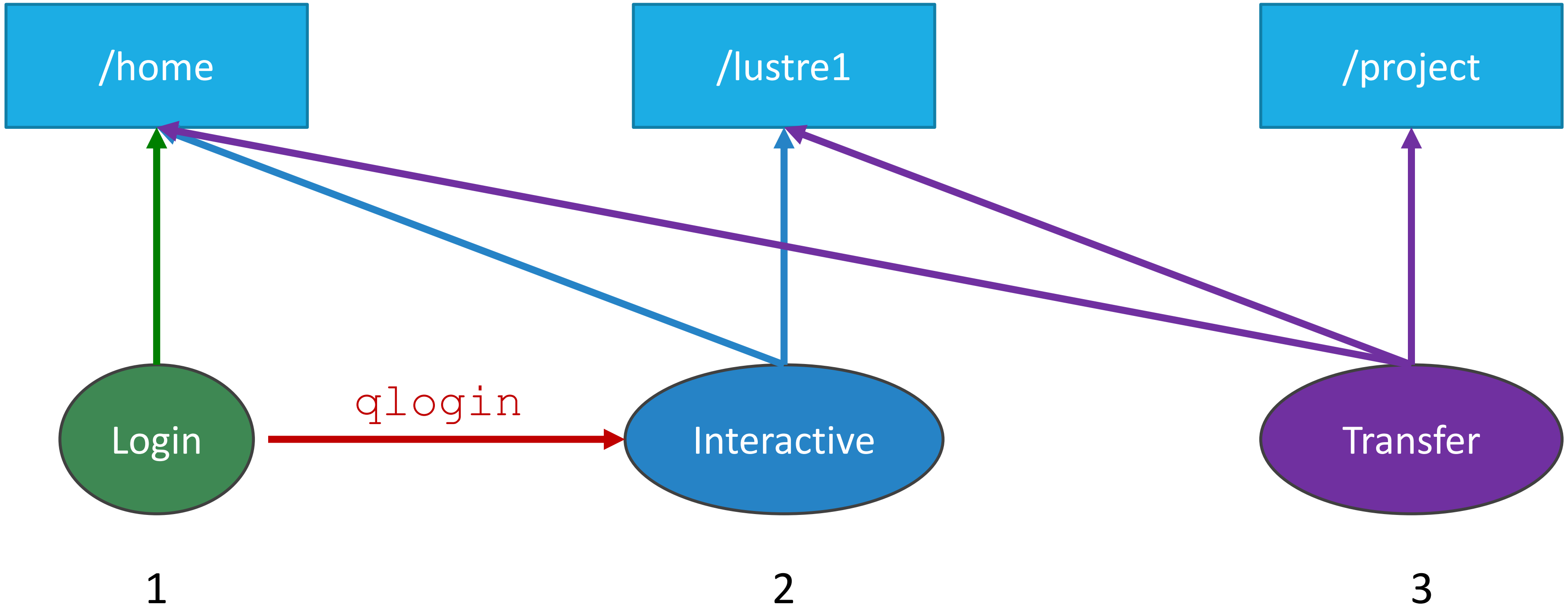
Storage Environment

4 Filesystems	Role	Quota	Accessible from	Intended Use	Notes
/home/username	Home	100GB	sapelo1.gacrc.uga.edu (Login) Interactive nodes (Interactive) xfer.gacrc.uga.edu (Transfer) build1.gacrc.uga.edu (Build) compute nodes (Compute)	Highly static data being used frequently	Snapshots
/lustre1/username	Global Scratch	No Limit	Interactive nodes (Interactive) xfer.gacrc.uga.edu (Transfer) compute nodes (Compute)	Temporarily storing large data being used by jobs	Auto-moved to /project if 30 days no modification
/lscratch/username	Local Scratch	250GB	Individual compute node	Jobs with heavy disk I/O	User to clean up
/project/abclab	Storage	Variable	xfer.gacrc.uga.edu (Transfer)	Long-term data storage	Group sharing possible

Note: /usr/local/apps : Software installation directory
/db : Bioinformatics database installation directory

Storage Environment

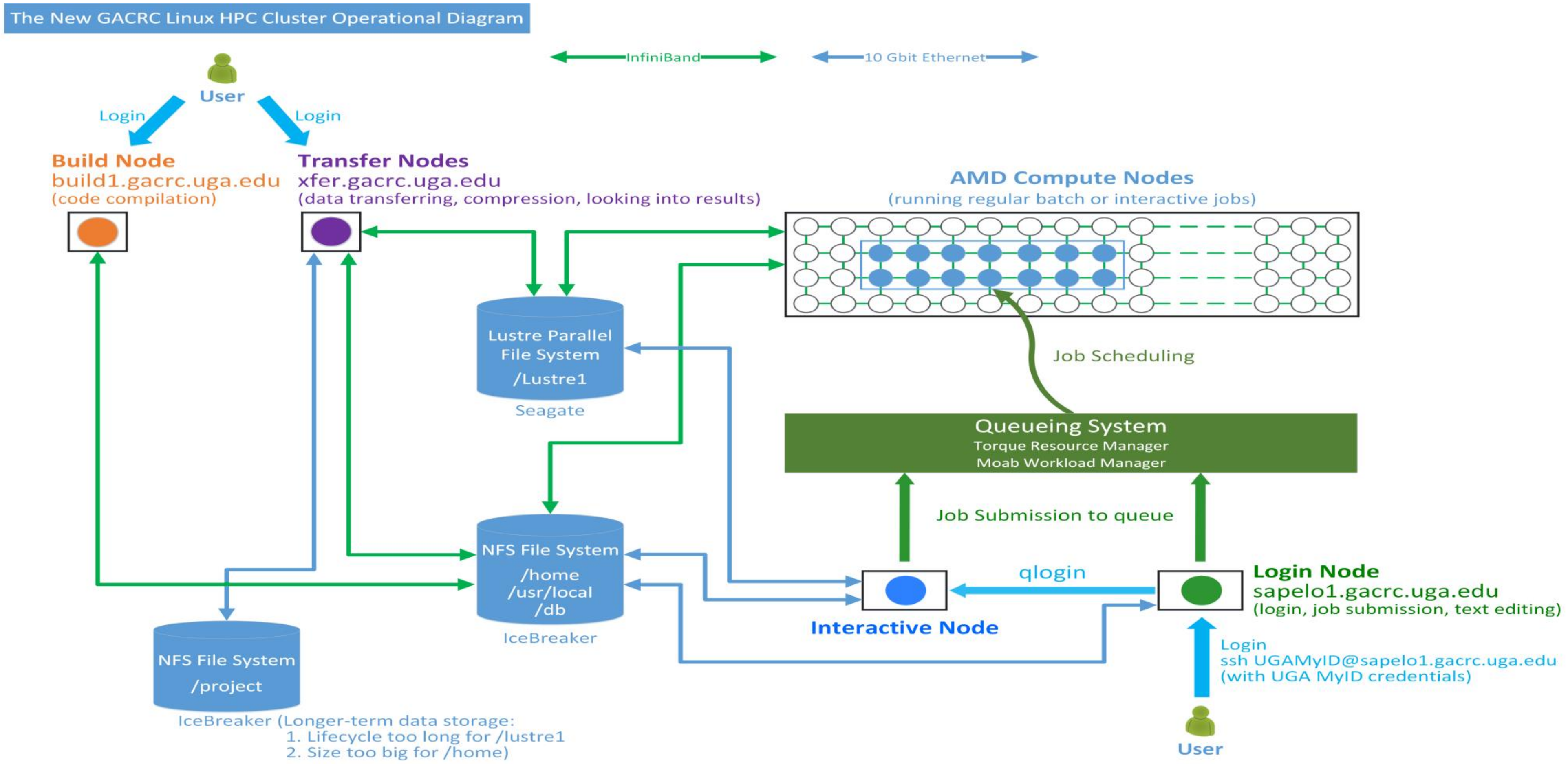
Access Rule of 123:



What is the new cluster – Storage Environment

7 Main Functions	On/From-Node	Related Filesystem
Login Landing	Login or Transfer or Build	/home/username (Home) (Always!)
Batch Job Submitting	Login	/home/username (Home)
	Interactive	/lustre1/username (Scratch) (Suggested!) /home/username (Home)
Interactive Job Running	Interactive	/lustre1/username (Scratch) /home/username (Home)
Data Transferring, Archiving , and Compressing	Transfer	/lustre1/username (Scratch) /home/username (Home)
Job Data Temporarily Storing	Compute	/lscratch/username (Local Scratch) /lustre1/username (Scratch)
Long-term Data Storing	Transfer	/project/abclab
Code Compilation	Build	/home/username (Home)

Workflow on Sapelo



Workflow – Global Scratch as Job Working Space

1. Linux/Mac user:

`ssh UGAMyID@sapelo1.gacrc.uga.edu`

Windows user:

Login

2. `qlogin`

Interactive

3. `cd /lustre1/userID`

4. `mkdir ./workDir`

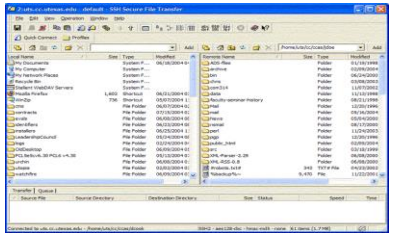
5. `cd ./workDir`

lustre1

6. Linux/Mac user:

`scp input UGAMyID@xfer.gacrc.uga.edu:/lustre1/userID/workDir`

Windows user:



Note: inputDate could be files or data folder (`scp -r`)

7. `nano ./sub.sh`

```
#PBS -S /bin/bash
#PBS -q batch
#PBS -N testBlast
#PBS -l nodes=1:ppn=1:AMD
#PBS -l mem=20gb
#PBS -l walltime=48:00:00
cd $PBS_O_WORKDIR
module load ncbiblast+/2.2.29
blastn [options] > outputfile
```

8.

```
$ qsub sub.sh
```

Useful PBS headers:

`#PBS mem=200gb : use 256GB AMD nodes)`

`ppn=24 : request 24 cores for 24 threads`

Workflow – Global Scratch as Job Working Space

1. Log on to Sapelo Login node: `ssh UGAMyID@sapelo1.gacrc.uga.edu`
2. From Sapelo Login node, transfer to Interactive node by issuing the command: `qlogin`
3. From Interactive node: Change directory to global scratch: `cd /lustre1/userID`
4. Create a working subdirectory on global scratch: `mkdir ./workDir`
5. Change directory to `workDir`: `cd ./workDir`
6. Transfer data to `workDir` using `scp` or SSH Client File Transfer (with `tar` or `gzip`)
7. Make a Sapelo job submission script: `nano ./sub.sh`
8. Submit job: `qsub ./sub.sh`

Useful PBS headers: `#PBS -l mem=200gb` : use 256GB high-RAM AMD nodes

`#PBS -l nodes=1:ppn=24:AMD` : request 24 cores for 24 threads, max 48!

Work on Sapelo

- Connect and Login
- Transfer Files Using Transfer Node
- Software Packages
- Run Interactive Jobs
- Run Batch Jobs
    - ✓ Binary Code Compilation
    - ✓ Submit Batch Jobs
    - ✓ Serial, Threaded, MPI, and GPU-enabled Job Scripts
    - ✓ Check Computing Resources Used by a job

Connect and Login

- On Linux/Mac: use Terminal utility and `ssh` to your account:

```
ssh zhuofei@sapelo1.gacrc.uga.edu
```

or

```
ssh -X zhuofei@sapelo1.gacrc.uga.edu
```

([1] `-X` is for *X windows application* running on the cluster with its UGI to be forwarded to local
[2] On Windows, use a *SSH client* to open the connection (next page))

- Logging in: You will be prompted for your UGA MyID password:

```
zhuofei@sapelo1.gacrc.uga.edu's password:
```

([3] On Linux/Mac, when you type in the password, the prompt blinks and does not move)

- Logging out: `exit` to leave the system:

```
[zhuofei@75-104 ~]$ exit
```

Transfer Files Using Transfer Node xfer.gacrc.uga.edu

- ✓ `ssh UGAMyID@xfer.gacrc.uga.edu` with your UGA MyID password
- ✓ Landing directory: `/home/username` (Sapelo home)
- ✓ Move data into/out of Sapelo (`scp`, `sftp`, `rsync`, SSH Secure Shell File Transfer, FileZilla)
- ✓ Compress or package data on Sapelo (`tar`, `gzip`)
- ✓ Transfer data between Sapelo and zcluster (`cp`, `mv`)
- ✓ Filesystems you can access:
  - /home/username/ : Sapelo home (landing spot)
  - /lustre1/username/ : Sapelo global scratch
  - /panfs/pstor.storage/home/abclab/username/ : zcluster home
  - /escratch4/username/ : zcluster scratch
  - /project/abclab/ : long-term active data storage
- ✓ Most file systems on Transfer are *auto-mounted* upon *the first time full-path access*, e.g., `cd /project/abclab/`

Transfer Files Using Transfer Node xfer.gacrc.uga.edu

User's local 

Transfer (xfer.gacrc.uga.edu)

- On Linux, Mac or cygwin on Windows : `scp (-r) [Source] [Target]`

  E.g. 1: working on local machine, from Local ➔ Sapelo global scratch

```
scp ./file zhuofei@xfer.gacrc.uga.edu:/lustre1/zhuofei/work_Jul_1/
```

```
scp -r ./folder/ zhuofei@xfer.gacrc.uga.edu:/lustre1/zhuofei/work_Jul_1/
```

  E.g. 2: working on local machine, from Sapelo global scratch ➔ Local

```
scp zhuofei@xfer.gacrc.uga.edu:/lustre1/zhuofei/work_Jul_1/file ./
```

```
scp -r zhuofei@xfer.gacrc.uga.edu:/lustre1/zhuofei/work_Jul_1/folder/ ./
```

- On Window: SSH Secure Shell File Transfer, FileZilla, WinSCP

Software Packages

- The cluster uses environment modules to define the various paths for software packages
- Current number of modules installed is ~300 and expanding daily!
- `module avail` to list all available modules (centralized installation):

```
[zhuofei@75-104 ~]$ module avail

----------------------------------------------------------------------- /usr/local/modulefiles -----------------------------------------------------------------------
   Core/StdEnv                        exabayes/1.4.1                 java/jdk1.8.0_20                openmpi/1.6.5/gcc/4.4.7             rsem/latest
   Data/cache/moduleT.new             examl/3.0.11                   java/latest               (D)   openmpi/1.6.5/pgi/14.9              rsem/1.2.20         (D)
   Data/cache/moduleT           (D)   expat/latest                   lammps/5Sep14                   openmpi/1.8.3/gcc/4.4.7             samtools/latest
   Data/system.txt                    expat/2.0.1              (D)   lammps/16Aug13            (D)   openmpi/1.8.3/gcc/4.7.4             samtools/0.1.19
   R/3.1.2                            fastqc/latest                  moab/7.2.10                     openmpi/1.8.3/gcc/4.8.0       (D)   samtools/1.1
   bedops/latest                      fastqc/0.11.3            (D)   moab/8.1.1                (D)   openmpi/1.8.3/intel/14.0            samtools/1.2        (D)
   bedops/2.4.14                (D)   gcc/4.7.4                      moabs/1.3.2                     openmpi/1.8.3/intel/15.0.2 (D)      scripture/latest
   boost/1.47.0/gcc447                gcc/4.8.0                (D)   mvapich2/2.0.0/gcc/4.4.7        openmpi/1.8.3/pgi/14.9              scripture/03202015 (D)
   boost/1.57.0/gcc447                gmap-gsnap/latest              mvapich2/2.0.0/pgi/14.9         orca/3.0.3                          sparsehash/latest
   boost/1.57.0_thread/gcc447         gmap-gsnap/2014-12-24    (D)   ncbiblast+/2.2.29               perl/latest                         sparsehash/2.0.2    (D)
   bowtie/latest                      gnuplot/5.0.0                  netcdf/3.6.3/gcc/4.4.7          perl/5.20.1                         tophat/latest
   bowtie/1.1.1                 (D)   gsl/1.16/gcc/4.4.7             netcdf/3.6.3/intel/14.0         perl/5.20.2                   (D)   tophat/2.0.13       (D)
   bowtie2/latest                     hdf5/1.8.14/gcc/4.4.7          netcdf/3.6.3/intel/15.0.2 (D)   pgi/14.9                            trinity/latest
   bowtie2/2.2.4                (D)   hdf5/1.8.14/intel/15.0.2       netcdf/4.1.3/gcc/4.4.7          pgi/14.10                     (D)   trinity/r20140717
   cuda/5.0.35/gcc/4.4.7              hdf5/1.8.14/pgi/14.9           netcdf/4.1.3/intel/15.0.2       python/2.7.8-ucs4                   trinity/2.0.6       (D)
   cuda/6.5.14/gcc/4.4.7              imb/3.2                        netcdf/4.1.3/pgi/14.10          python/2.7.8                        zlib/gcc447/1.2.8
   cufflinks/latest                   intel/14.0                     netcdf/4.3.2/gcc/4.4.7          python/3.4.3                  (D)
   cufflinks/2.2.1              (D)   intel/15.0.2             (D)   netcdf/4.3.2/pgi/14.9           raxml/8.1.20
```

Software Packages

- `module list` to list which modules currently loaded:

```
[zhuofei@75-104 ~]$ module list

Currently Loaded Modules:
  1) StdEnv   2) moab/7.2.10
```

- `module load` to load the needed modules:

```
[zhuofei@75-104 ~]$ module load ncbiblast+/2.2.29
[zhuofei@75-104 ~]$ module load python/2.7.8
[zhuofei@75-104 ~]$ module load R/3.1.2
[zhuofei@75-104 ~]$ module list

Currently Loaded Modules:
  1) StdEnv   2) moab/7.2.10   3) ncbiblast+/2.2.29   4) python/2.7.8   5) R/3.1.2
```

- `module unload` to remove the specific module:

```
[zhuofei@75-104 ~]$ module unload R/3.1.2
[zhuofei@75-104 ~]$ module list

Currently Loaded Modules:
  1) StdEnv   2) moab/7.2.10   3) ncbiblast+/2.2.29   4) python/2.7.8
```

Run Interactive Jobs

- To run an interactive job, using `qlogin` command from Login node:

```
[zhuofei@75-104 ~]$ qlogin
qsub: waiting for job 1058157.pbs.scm to start
qsub: job 1058157.pbs.scm ready
[zhuofei@n14 ~]                          ← Now I am on n14, which is an interactive node
[zhuofei@n14 ~]$ module load R/3.2.3
[zhuofei@n14 ~]$ R

R version 3.2.3 (2015-12-10) -- "Wooden Christmas-Tree"
Copyright (C) 2015 The R Foundation for Statistical Computing
Platform: x86_64-pc-linux-gnu (64-bit)

...
[Previously saved workspace restored]

>
```

- When you are done, remember to `exit` the session

Run Batch Jobs

- Components you need to run a batch job:
  - binary compiled on build1 or interactive node (AMD Opteron)
  - Job submission script to run the binary, specifying computing resources:
    - ✓ Number of nodes and cores
    - ✓ Amount of memory
    - ✓ Type of nodes
    - ✓ Maximum wallclock time, etc.
- Common queueing commands:
  - qsub, qdel
  - qstat -f, showjobs, showq

Binary Code Compilation

- Login to build1.gacrc.uga.edu or go to Interactive node

- Load compilers, e.g.:

```
module load pgi/14.9                 → PGI 14.9 compilers
module load intel/15.0.2             → Intel 15.0.2 compilers
module load mvapich2/2.0.0/gcc/4.4.7 → MVAPICH2 2.0.0 using GNU 4.4.7 compilers
module load openmpi/1.8.3/pgi/14.9   → OpenMPI 1.8.3, using PGI 14.9 compilers
```

1. *Default GNU 4.4.7 compilers*
2. *No MPICH2, which uses Ethernet only*

- Compile source codes into binary

https://wiki.gacrc.uga.edu/wiki/Code_Compilation_on_Sapelo
https://wiki.gacrc.uga.edu/wiki/MPI

Submit Batch Jobs

```
[zhuofei@n15 work_0906]$ pwd                ← n15: interactive node
/lustre1/zhuofei/work_0906                  ← /lustre1 global scratch
[zhuofei@n15 work_0906]$
[zhuofei@n15 work_0906]$ qsub sub.sh
1165617.pbs.scm
```

qsub is to submit a job

sub.sh is your **job submission script** specifying:

- ✓ Number of nodes and cores
- ✓ Amount of memory
- ✓ Type of nodes
- ✓ Maximum wallclock time, etc.

Example 1: sub.sh to run a serial binary

#PBS -S /bin/bash	→ Linux shell (bash)
#PBS -q batch	→ Queue name (batch)
#PBS -N testJob	→ Name of the job (testJob)
#PBS -l nodes=1:ppn=1:AMD	→ Number of nodes (1), number of cores/node (1), node type (AMD)
#PBS -l mem=10gb	→ Maximum amount of physical memory (10 GB) used by the job
#PBS -l walltime=48:00:00	→ Maximum wall clock time (48 hours) for the job, default 6 minutes
cd $PBS_O_WORKDIR	→ Use the directory from which the job is submitted as the working directory
time ./binary > outputfile 2>&1	→ Run a binary with 'time' command to measure the amount of time it takes to run the application

Example 1: sub.sh to run a serial binary

```
#PBS -S /bin/bash
#PBS -q batch
#PBS -N testJob
#PBS -l nodes=1:ppn=1:AMD
#PBS -l mem=200gb
#PBS -l walltime=48:00:00

cd $PBS_O_WORKDIR



time ./binary > outputfile 2>&1
```

➔ Job will be dispatched to run on AMD 256GB node

Example 2: sub.sh to run an OpenMP threaded binary

```
#PBS -S /bin/bash
#PBS -q batch
#PBS -N testJob
#PBS -l nodes=1:ppn=10:AMD
#PBS -l mem=200gb
#PBS -l walltime=48:00:00

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

cd $PBS_O_WORKDIR

export OMP_NUM_THREADS=10

time ./binary > outputfile 2>&1
```

- `#PBS -l nodes=1:ppn=10:AMD` → Number of nodes (1), number of cores/node (10), AMD node
  Number of threads (10) = Number of cores requested (10)
- `#PBS -M jSmith@uga.edu` → Email address to receive a notification for computing resources
- `#PBS -m ae` → Send email notification when job aborts (a) or terminates (e)
- `#PBS -j oe` → Standard error file (testJob.e1234) will be merged into standard out file (testJob.o1234)
- `time ./binary > outputfile 2>&1` → Run an OpenMP threaded binary with 10 threads

Example 3: sub.sh to run an OpenMPI binary

```
#PBS -S /bin/bash
#PBS -q batch
#PBS -N testJob
#PBS -l nodes=2:ppn=48:AMD          ➔ Number of nodes (2), number of cores/node (48)
#PBS -l walltime=48:00:00              Total cores requested = 2 × 48 = 96
#PBS -l mem=10gb

cd $PBS_O_WORKDIR

module load openmpi/1.8.3/pgi/14.9

mpirun ./binary > outputfile 2>&1   ➔ Run an OpenMPI binary with 96 MPI processes
```

```
#PBS -S /bin/bash
#PBS -q batch
#PBS -N testJob
#PBS -l nodes=2:ppn=24:AMD
#PBS -l walltime=48:00:00
#PBS -l mem=10gb

cd $PBS_O_WORKDIR

# Context Sharing
CONTEXTS=$(/usr/local/bin/set_contexts.sh $PBS_NUM_PPN)
if [[ "$?" -eq "0" ]] ; then
  export PSM_SHAREDCONTEXTS_MAX=$CONTEXTS
fi

module load openmpi/1.8.3/pgi/14.9

mpirun ./binary > outputfile 2>&1
```

➔ ppn number (24) fewer than 48 MUST be a multiplier of 3!
Total cores requested = 2 × 24 = 48

New lines copied from GACRC Wiki

➔ Run an OpenMPI binary with 48 MPI processes

```
#PBS -S /bin/bash
#PBS -q batch
#PBS -N testJob
#PBS -l nodes=2:ppn=27:AMD
#PBS -l walltime=48:00:00
#PBS -l mem=10gb

cd $PBS_O_WORKDIR

# Context Sharing
CONTEXTS=$(/usr/local/bin/set_contexts.sh $PBS_NUM_PPN)
if [[ "$?" -eq "0" ]] ; then
  export PSM_SHAREDCONTEXTS_MAX=$CONTEXTS
fi

module load openmpi/1.8.3/pgi/14.9

mpirun –np 50 ./binary > outputfile 2>&1
```

➔ Total cores requested = 2 × 27 = 54

➔ Run an OpenMPI binary with 50 MPI processes

Example 4: sub.sh to run a GPU-enabled binary

```
#PBS -S /bin/bash
#PBS -q batch
#PBS -N testJob
#PBS -l nodes=1:ppn=10:gpus=1:GPU        ➔ Number of GPU card (1), node type (GPU)
#PBS -l walltime=48:00:00
#PBS -l mem=10gb

cd $PBS_O_WORKDIR

module load cuda/6.5/gcc/4.4.7           ➔ Load CUDA 6.5 module

Time ./binary >outputfile 2>&1           ➔ Run GPU-enabled binary with 10 cores and 1 GPU card
```

```
#PBS -S /bin/bash
#PBS -q batch
#PBS -N testJob
#PBS -l nodes=1:ppn=10:gpus=1:GPU                    ➔ Number of GPU card (1), node type (GPU)
#PBS -l walltime=48:00:00
#PBS -l mem=10gb

cd $PBS_O_WORKDIR

module load cuda/6.5/gcc/4.4.7

#To get a list of the numbers of the GPU devices allocated to the job, separated by a blank space:
CUDADEV=$(cat $PBS_GPUFILE | rev | cut -d"u" -f1)

#To remove the blank space between two device numbers in the CUDADEV variable above:
GPULIST=$(echo $CUDADEV | sed 's/ //')

Time ./binary $CUDALIST >outputfile 2>&1
```

Check Computing Resources Used by a Job

Option 1: **`qstat -f JobID`** for ***running jobs*** or ***finished jobs in 1 hour***

Option 2: **`showjobs JobID`** for ***finished jobs over 1 hour, but ≤ 7 days***

Option 3: Email notification from ***finished jobs (completed, canceled, or crashed)***,

if using:

```
#PBS -M jSmith@uga.edu
#PBS -m ae
```

How to work with it – Run Jobs

- **qstat -f JobID** for ***running jobs*** or ***finished jobs in 1 hour***

```
[zhuofei@75-104 MPIs]$ qstat -f 699847
Job Id: 699847.pbs.scm
    Job_Name = testJob
    Job_Owner = zhuofei@uga-2f0f976.scm
    resources_used.cput = 00:11:55
    resources_used.energy_used = 0
    resources_used.mem = 411572kb
    resources_used.vmem = 6548528kb
    resources_used.walltime = 07:01:36
    job_state = C
    queue = batch
     .
    Error_Path = uga-2f0f976.scm:/home/zhuofei/MPIs/testJob.e699847
    exec_host = n165/0-23
    Output_Path = uga-2f0f976.scm:/home/zhuofei/MPIs/testJob.o699847
     .
    Resource_List.mem = 5gb
    Resource_List.nodect = 1
    Resource_List.nodes = 1:ppn=24:AMD
    Resource_List.walltime = 10:00:00
     .
    Variable_List = PBS_O_QUEUE=batch,PBS_O_HOME=/home/zhuofei, …… ,
                    PBS_O_WORKDIR=/home/zhuofei/MPIs,
```

How to work with it – Run Jobs

- **`showjobs JobID`** for ***finished jobs over 1 hour, but ≤ 7 days***

```
[zhuofei@75-104 MPIs]$ showjobs 699847
Job Id              : 699847.pbs.scm
Job Name            : testJob
Output File         : uga-2f0f976.scm:/home/zhuofei/MPIs/testJob.o699847
Error File          : uga-2f0f976.scm:/home/zhuofei/MPIs/testJob.e699847
Working Directory : /home/zhuofei/MPIs
Home Directory      : /home/zhuofei
Submit Arguments    : sub.sh
User Name           : zhuofei
Group Name          : rccstaff
Queue Name          : batch
Wallclock Limit     : 10:00:00
Wallclock Duration: 07:01:36
CPUTime             : 00:11:55
Memory Used         : 401.9Mb
Memory Limit        : 5gb
vmem Used           : 6.2Gb
Submit Time         : Wed Nov  4 12:02:22 2015
Start Time          : Wed Nov  4 12:03:41 2015
End Time            : Wed Nov  4 12:04:45 2015
Exit Code           : 0
Master Host         : n165
```

How to work with it – Run Jobs

- Email notification from ***finished jobs (completed, canceled, or crashed)***

```
PBS Job Id: 700009.pbs.scm
Job Name:   testJob
Exec host:  n1/4-27
Execution terminated
Exit_status=0
resources_used.cput=00:05:12
resources_used.energy_used=0
resources_used.mem=410984kb
resources_used.vmem=6548516kb
resources_used.walltime=04:00:59
Error_Path: uga-
2f0f976.scm:/home/zhuofei/MPIs/testJob.o700009
Output_Path: uga-
2f0f976.scm:/home/zhuofei/MPIs/testJob.o700009
```

```
PBS Job Id: 700097.pbs.scm
Job Name:   testJob
Exec host:  n1/4-27
Execution terminated
Exit_status=271
resources_used.cput=00:11:22
resources_used.energy_used=0
resources_used.mem=412304kb
resources_used.vmem=6548524kb
resources_used.walltime=05:00:41
Error_Path: uga-
2f0f976.scm:/home/zhuofei/MPIs/testJob.o700097
Output_Path: uga-
2f0f976.scm:/home/zhuofei/MPIs/testJob.o700097
```

GACRC

Thank You!