Cheese Cluster Training

The Biostatistics Computer Committee (BCC)
Anjishnu Banerjee
Dan Eastwood
Chris Edwards
Michael Martens
Rodney Sparapani
Sergey Tarima
and The Research Computing Center (RCC)
Matthew Flister
Greg McQuestion

May 17, 2017

Review of Cluster Management Software

- Portable Batch System (PBS): early job scheduler started at NASA in 1991
- OpenPBS: open source version of PBS released in 1998 with the OpenPBS license
- Terascale Open-source Resource and QUEue manager (TORQUE): a cross-platform fork of OpenPBS originally released in 2003 with the OpenPBS license
- TORQUE: a distributed resource manager for the cluster providing control over queued jobs and distributed computers
- Maui: an open source scheduler which integrates with PBS/OpenPBS/TORQUE to improve overall utilization, scheduling and administration of the cluster (started mid-90s)
- Moab: a commercial scheduler originally based on Maui
- Adaptive Computing of Provo, UT: develops, maintains and supports TORQUE, Maui and Moab
- LANL Roadrunner runs TORQUE/Moab: ranked as the world's fastest supercomputer in 2008

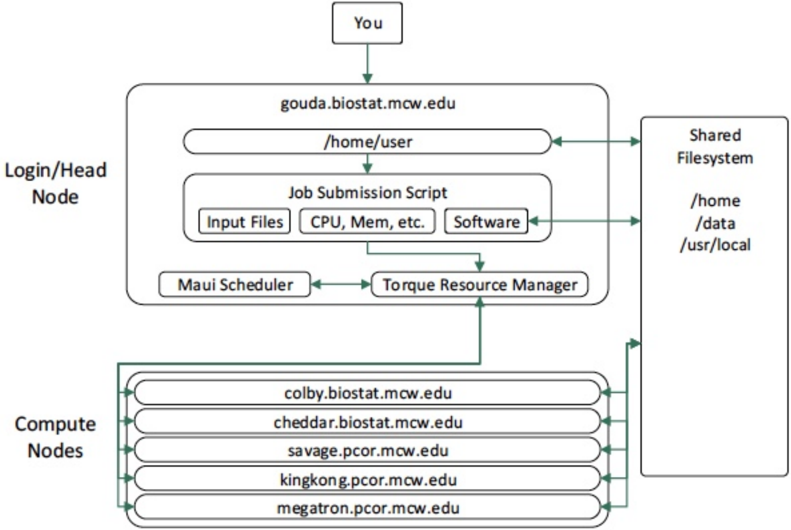
TORQUE Enhancements to OpenPBS

- More fault tolerant
- Better scheduling
- More user friendly
- More scalable

Cheese Cluster Timeline

- 2010, Jan: Oracle buys Sun Microsystems
- 2010, Mar: AMD releases AMD64 Opteron CPU with 12 cores
- 2010, Dec: PCOR switches to Linux with Dell 4 CPU servers
- 2014, Nov: BCC “discovers” TORQUE/Maui
- 2014, Nov: RCC forms and adopts TORQUE/Moab
- 2015, Mar: BCC adopts TORQUE/Maui
- 2015, May: PCOR acquires, godzilla, a large storage array
- 2015, Jun 4: Biostat moves to Linux with Dell
- 2015, Jul: Biostat and PCOR unite to create Cheese Cluster
- 2017, Jan-May: /home, /data, /usr/local moved to godzilla
- 2017, May 4: Cheese Cluster moves into Test phase
- 2017, Jun: Production phase

The Biostatistics/PCOR Cheese Cluster and You



Cheese Cluster Nodes: AMD64 Architecture

Node	Role	CPU	Core	Thread	Total
gouda	head/master	2 Intel Xeon	8	16	32
cheddar	compute/slave	4 Intel Xeon	8	16	64
colby	compute/slave	4 Intel Xeon	8	16	64
kingkong	compute/slave	4 AMD Opteron	12	12	48
megatron	compute/slave	4 AMD Opteron	12	12	48
savage	compute/slave	4 AMD Opteron	12	12	48

TORQUE/PBS Commands Batch: gouda$ qsub dm20

```
#!/bin/sh
####################################################
#this is the contents of script file dm20
#PBS -N dm20                # Set the job name
#PBS -l nodes=1:ppn=31      # N nodes with M threads
#PBS -l mem=2gb             # Xb/kb/mb/gb RAM (integer)
#PBS -l walltime=5:00:00    # H:00:00hrs elapsed time
cd $PBS_O_WORKDIR          # move to your current dir
time R --no-save < dm20.R >& dm20.Rout
#this is the end of script file dm20
####################################################
```

Take care with `-l mem` and `-l walltime`!

In R, use liberally: `object.size()`, `saveRDS()`, `rm()`, `gc()`

Each job generates 1 file for stderr: `NAME.eNNN` , e.g., `dm20.e97`

And 1 file for stdout: `NAME.oNNN` , e.g., `dm20.o97`

The output of `time` ends up in stderr

Notice that we don't need `nohup` nor `nice`

TORQUE/Maui Commands

TORQUE	submit a batch job	`gouda$ qsub` *script_file*
	submit an interactive job	`gouda$ qsub -I` *script_file*
		`gouda$ qsub -I`
	with X windows	`gouda$ qsub -I -X`
	check job status	`gouda$ qstat`
		`gouda$ qstat -u` *user_id*
	delete a job	`gouda$ qdel` *job_id*
	debug a job	`gouda$ tracejob` *job_id*
Maui	show queue	`gouda$ showq`
	show usage stats	`gouda$ showstats -u` *user_id*

TORQUE/Maui and PBS Documentation

- TORQUE/Maui: `man` *command_name*
- PBS environment variables and options: `man qsub`
- `#PBS -l` *resource_list*: `man pbs_resources_sunos4`

Cheese Cluster Queues

Name	Walltime Limit	Threads Limit	Jobs/User Limit	Description
small	72:00:00	8	3	Single node only
medium	96:00:00	32	2	Multiple nodes allowed
large	168:00:00	128	1	Multiple nodes allowed

- Multiple node jobs: the Message Passing Interface (MPI) for multiple nodes, i.e., suppose you need to run your job on 2 or more nodes simultaneously with TORQUE

Cheese Cluster Etiquette

- Do not start CPU intensive work on head/master gouda
  if gouda becomes unresponsive: all queued jobs may crash
  (GPU computing on gouda is OK, but please be careful)
- All CPU intensive jobs must be run through TORQUE
- Please refrain from logins to the compute/slave nodes unless you are debugging a failed job
  N.B. you can only login to a compute/slave node from gouda

Cheese Cluster Training Wrap-up

- This presentation and the final documentation will be put on our web page ASAP
- Keep in mind that we are sharing this precious resource
- Don't unnecessarily inflate `-l mem` or `-l walltime`
  your job may be repeatedly placed at the end of the queue
- Please look over the documentation
- If you have any problems, then contact Chris
  and thank him for all of his hard work!
- CRAN R packages: 8951 installed out of 10629
  Also, working on Bioconductor 1383 packages
- To "turn on" all of the nodes, we need to schedule a system shutdown. What is a good time for everyone?
- TORQUE demo
- Any questions?