Using CARC Systems

Jim Prewett

2014-03-28 Fri

Outline

What are Environment Modules?

- Environment Modules set up your shell's environment to enable you to use a specific software package.
  - They set the shell's search so that you can invoke the executables.
  - They can also set other environment variables used by the software.
- At CARC, they are the preferred way to set up your environment!

What sorts of Modules are available?

- Modules should be available for any software that is installed into a "non-standard" location.
  - Software installed by the operating system's package manager will usually not have environment modules as the default environment should be able to use the software without modifications.
- Modules are available for Compilers (Intel & PGI), Math libraries (FFTW, BLAS, etc.), MPI libraries (OpenMPI, MPICH-MX, MVAPICH, etc.) software packages (Matlab, Gaussian, etc.).

Basic Module commands

Subcommand	Meaning
--help	Display help for the module command
avail	List all available environment modules
list	List all *currently loaded* environment modules
add/load	Load the named environment module
rm/unload	Unload the named environment module
display/show	Show the contents of the named modulefile
help	Display information about the named modulefile

Listing Available Modules

- To list the environment modules available on a system, run `module avail`.
- Let's take a look at the output of that now!

Example usage:

```
# Load the OpenMPI Environment Module for the
# Intel 14.0.0 Compilers
module load openmpi/1.6.5/intel/14.0.0

# Use "mpicc" to compile the source code
mpicc my_program.c -o my_program

# Run the program on 4 cores

mpirun -np 4 my_program
```

PBS (Torque)

- Torque is the batch manager for our supercomputer and Beowulf cluster systems.
- Torque allocates compute nodes to jobs so that they are allocated the resources that they need in order to run.
- Torque is derived from the Portable Batch System (PBS) originally from NASA.
- As such, it is common to refer to Torque as "PBS".

PBS (Torque) Usage

Command	Use
`qsub`	Submit a job to the batch scheduler
`qstat`	List the jobs in the queue(s)
`qdel`	Delete a job from the batch scheduler

Submit a job using qsub

- The `qsub` command submits a job to the batch scheduler
  - Most people submit shell scripts to the batch scheduler like this:

```
qsub myjob.pbs
```

  - Sometimes it is useful to submit "interactive" jobs to the scheduler like this:

```
qsub -I -lnodes=1:ppn=4,walltime=1:00:00
```

Check the job/queue status using qstat

- The `qstat` command checks the status of the jobs currently in the queue(s) on the system.
  - To check the status of a given job, run this command:

    `qstat <JOB ID>`
  - To see all of the jobs in the queue simply run `qstat` without arguments.

qstat Output

- When you run the `qstat` command, the output will look something like this:

```
Job id                    Name             User            Time Use S Queue
------------------------- ---------------- --------------- -------- - -----
26567.nano                 ...ramid_test6_1 brh5103          25:04:27 R one_node
26877.nano                 ErbB23_Main      mmmccabe         24:41:15 R one_node
26878.nano                 ErbB23_Main      mmmccabe         24:38:48 R one_node
26879.nano                 ErbB23_Main      mmmccabe         24:38:56 R one_node
26880.nano                 ErbB23_Main      mmmccabe         24:40:14 R one_node
26881.nano                 ErbB23_Main      mmmccabe         23:13:26 R one_node
26882.nano                 ErbB23_Main      mmmccabe         10:28:28 R one_node
26883.nano                 ErbB23_Main      mmmccabe         09:13:26 R one_node
26884.nano                 ErbB23_Main      mmmccabe         02:03:21 R one_node
26885.nano                 ErbB23_Main      mmmccabe                0 Q one_node
26886.nano                 ErbB23_Main      mmmccabe                0 Q one_node
26887.nano                 ErbB23_Main      mmmccabe                0 Q one_node
26892.nano                 rr               lgilkey          02:39:33 R defaultq
```

Using qgrok to Grok the queue status

Here is the output of the qgrok command on Galles:

queue:	free	busy	offline	Running jobs	Nodes	CPUs
DEFAULT	0	0	4	0	4	8
PD-2.80	98	0	3	0	101	202
PD-3.00	6	0	0	0	6	12
Core2	74	0	1	0	75	150
Hadoop	17	0	0	0	17	34
totals:	195	0	8	0	203	406

Queue names

- Here are the names of the queues on our machines:

System	Queue
Nano	defaultq[1]
Nano	debug
Nano	one_node
Nano	one_long
Metropolis	defaultq[1]
Pequena	workq[1]
Pequena	debug
Pequena	one_node
Pequena	one_long
Gibbs	defaultq[1]
Galles	default[1]
Galles	PD-2.80
Galles	PD-3.00

Submitting jobs to a specific queue

- *Usually* you don't need to specify a specific queue. The default queue on the machine should do what you need!
- To submit a job to a specific queue, pass the `-q <queue name>` argument to qsub
  - eg.
    - `qsub -q debug -lnodes=1:ppn=4,walltime=30:00`

Queue Limits

- Each of our systems has limits on the queues.
- These limit the:
  1. Maximum nodes used by a single user.
  2. The maximum amount of wallclock time.
  3. The maximum number of jobs a single user may have in the queue.
- Other limitations are possible, but are not (currently) used here at CARC.

Queue Limit specifics

- Here are the limits on the queues on our machines:

System	Queue	Max Nodes	Max Walltime
Nano	defaultq[1]	8	80:00:00
Nano	debug	1	00:30:00
Nano	one_node	1	48:00:00
Nano	one_long	1	160:00:00
Metropolis	defaultq[1]	35	48:00:00
Pequena	workq[1]	8	48:00:00
Pequena	debug	1	00:30:00
Pequena	one_node	1	48:00:00
Pequena	one_long	1	120:00:00
Gibbs	defaultq[1]	6	—
Galles	default[1]	72	—
Galles	PD-2.80	72	—
Galles	PD-3.00	4	—

PBS Environment Variables

- These environment variables are set in the shell's environment when you are *running a job*:

Variable Name	Meaning
`PBS_NODEFILE`	The name of the file containing the list of machines.
`PBS_O_WORKDIR`	The directory from which the job was submitted.
`PBS_NUM_NODES`	The number of nodes allocated to this job.
`PBS_NUM_PPN`	The number of processors per node.
`PBS_NP`	The number of processors allocated to this job.
`PBS_JOBID`	The PBS Job ID of this job.
`PBS_JOBNAME`	The name of this PBS job.

PBS Directives

These directives are interpreted by the batch scheduler:

Directive	Meaning
-lwalltime	The amount of wallclock time to request.
-lnodes=<X>:ppn=<Y>	The number of nodes/processors to request.
-m <mail options>	The point(s) at which to send email. e = mail at exit b = mail at beginning a = mail at abort n = no mail
-M <email addresses>	The address to send email when a job has started, finished, or aborted.
-q <queue name>	The queue to submit the job to.
-S <shell>	The command shell to use for this job.
-j oe	Combine STDOUT and STDERR files

Example TCSH Job

- Here we will write and submit an example job to the batch scheduler:
- Here is the script:

```
## Introductory Example
## Copyright (c) 2010 The Center for Advanced Research
##              Computing at The University of New Mexico
#PBS -lnodes=1:ppn=4
#PBS -lwalltime=1:00
## Specify the shell to be tcsh
#PBS -S /bin/tcsh

# print out a hello message
# indicating the host this is running on
setenv THIS_HOST `hostname`
echo Hello World from host $THIS_HOST
```

Example BASH Job

- Here is the same script specifying the BASH shell:

```
## Introductory Example
## Copyright (c) 2010 The Center for Advanced Research
##                Computing at The University of New Mexico
#PBS -lnodes=1:ppn=4
#PBS -lwalltime=1:00
## Specify the shell to be bash
#PBS -S /bin/bash

# print out a hello message
# indicating the host this is running on
export THIS_HOST=‘hostname‘
echo Hello World from host $THIS_HOST
```

Submitting the Job & Checking It's Status

- First, we submit the job to the queue using `qsub`:

  `qsub myjob.pbs`
- Now, check it's status using `qstat`:

  `qstat <JOBID>`

  or

  `qstat`

Viewing the Output and Error Files

- Once the job has completed, it is important to review the output and error files created by the scheduler.
- There should be a file named <Job Name>.o<Job ID> which contains all of the output printed to STDOUT.
  - eg. If the job is "myjob" and the ID is 1234, then the name of the output file would be myjob.o1234.
- There should also be a file named <Job Name>.e<Job ID> which contains all of the output printed to STDERR.
  - eg. With the above example names, the name of the error file would be myjob.e1234.