PBS Pro – Documentation

Introduction

Most jobs will require greater resources than are available on individual nodes. All jobs must be scheduled via the batch job system. The batch job system in use is PBS Pro. Jobs are submitted to PBS specifying required resources, including the queue, number of CPUs, the amount of memory, and the length of time needed. PBS will then run a job or jobs when the resources are available, subject to constraints on maximum resource usage.

Basic PBS commands

Some basic PBS commands are:

Command	Description

qsub *commandfile*	Submit jobs to the queues. The simplest use of the qsub command is typified by the following example:
	
	`[username@hpc-login-prd-t1 ~]$ qsub commandfile`
	or
	`[username@hpc-login-prd-t1 ~]$ qsub -q default -l \`
	`walltime=20:00:00,vmem=3000MB commandfile`
	
	where *commandfile* is an ascii file containing PBS commands (**not** the compiled executable which is a binary file).
	
	See qsub Options below for more options.

qstat –*u username*	Displays the status of PBS jobs and queues for the user *username*. See *man qstat* for details of options
qdel *jobid*	Delete your job from a queue. The *jobid* is returned by qsub at job submission time, and is also displayed in the qstat output.
qhold *jobid*	Place a hold on your job in the queue and stops it from running.
qrls –h u *jobid*	Release a user hold on your job and allows it to be run.
qrerun *jobid*	Terminate an executing job and return it to a queue.

qmove *jobid*	Move a job to a different queue or server.

qsub Options

There are two methods of specifying qsub options:

1. Within a PBS commandfile, and
2. On the qsub command-line.

Below is a simply example showing both methods.

```
[username@hpc-login-prd-t1 ~]$ qsub commandfile
```

Where *commandfile* contains:

```
#!/bin/bash -l
#
#PBS -l select=1:ncpus=16
#PBS -l walltime=1:00:00
#PBS -q default

### Specify the executable...
./an_exectuable
```

or

```
[username@hpc-login-prd-t1 ~]$ qsub -q default –l select=1:ncpus=16,walltime=1:00:00 an_executable
```

Below are some commonly used qsub options.

qsub Option	Description
#PBS -A *acct*	Causes the job time to be charged to "acct".
#PBS -N *myJob*	Assigns a job name. The default is the name of PBS job script.
#PBS -l nodes=4:ppn=2	The number of nodes and processors per node. (depreciated)
#PBS -l select=1:ncpus=2	The number of chunks or nodes and processors per.
#PBS -l ngpus=2	The number of gpus required.
#PBS -l walltime=01:00:00	Sets the maximum wall-clock time during which this job can run. (walltime=*hh:mm:ss*)
#PBS -l mem=*n*{mb|gb}	Sets the maximum amount of memory allocated to the job.
#PBS -l vmem=*n*{mb|gb}	Sets the maximum amount of virtual memory allocated to the job. (depreciated)
#PBS -q *queuename*	Assigns your job to a specific queue.
#PBS -o *mypath/my.out*	The path and file name for standard output.
#PBS -e *mypath/my.err*	The path and file name for standard error.
#PBS -j oe	Join option that merges the standard error stream with the standard output stream of the job.
#PBS -M *email-address*	Sends email notifications to a specific user email address.

#PBS -m {a|b|e}	Causes email to be sent to the user when: • a - the job aborts • b - the job begins • e - the job ends
#PBS –P *project*	Specifies what project the job belongs to.
#PBS -r n	Indicates that a job should not rerun if it fails.
#PBS -S *shell*	Sets the shell to use. Make sure the full path to the shell is correct.
#PBS -V	Exports all environment variables to the job.
#PBS -W	Used to set job dependencies between two or more jobs.
NOTE	PBS directives are all at the start of a script, that there are no blank lines between them, and there are no other non-PBS commands until after all the PBS directives.

A Job Script Example

A working job submission script takes the following form:

```
#!/bin/bash -l
#PBS -N Example_Job
#PBS -q default
#PBS -l select=2:ncpus=16
#PBS -l walltime=<hh:mm:ss>
#PBS -o <output-file>
#PBS -e <error-file>

module load matlab/r2016b

matlab –nodisplay –nosplash –r example_job.m
```

Where the line "**-l select=2:ncpus=16** " is the number of processors required for the job. **select** specifies the number of nodes (or chunks of resource) required; **ncpus** indicates the number of CPUs per chunk required.

As this is not the most intuitive command, the following table is provided as guide to how this command works:

select	ncpus	Description
2	16	32 Processor job, using 2 nodes and 16 processors per node
4	8	32 Processor job, using 4 nodes and 8 processors per node
16	1	16 Processor job, using 16 nodes and 1 processor per node
8	16	128 Processor job, using 8 nodes and 16 processors per node

The line "**-l walltime=<hh:mm:ss>**" is the time limit for the job. If your job exceeds this time the scheduler will terminate the job. It is recommended to find a usual runtime for the job and add some more (say 20%) to it. For example, if a job took approximately 10 hours, the walltime limit could be set to 12 hours, e.g. "**-l walltime=12:00:00**". By setting the walltime the scheduler can perform job scheduling more efficiently and also reduces occasions where errors can leave the job stalled but still taking up resource for the default much longer walltime limit (for queue walltime defaults run "**qstat -q**" command).

Job management

The **qstat** command displays the status of the PBS scheduler and queues. Using the flags **-Qa** shows the queue partitions available. If no queue is defined, it will use the queue called *default*. The following table shows the commonly using queues:

express:

- all nodes available
- low priority
- 8 hours of run time available

serial:

- all nodes available
- high priority
- 168 hours of run time available

short:

- all nodes available
- standard priority
- 24 hours of run time available

medium:

- all nodes available
- standard priority
- 72 hours of run time available

long:

- all nodes available
- standard priority
- 168 hours of run time available

PBS Job States

The table below describes the different job states through the life cycle of a job. There are some attributes that are only applicable when submitting jobs to an Enterprise PBS Professional complex.

Job State	Description
B	Job arrays only: job array has Begun.
E	Job is Exiting after having run.
F	Job has Finished exiting and execution. The job was completed successfully and had no application errors.
	Job has Finished exiting and execution; however, the job experienced application errors.
H	Job is Held. A job is put into a held state by the server or by a user or administrator. A job stays in a held state until it is released by a user or administrator.
Q	Job is Queued, eligible to run or be routed.
R	Job is Running.
S	Job is Suspended by server. A job is put into the suspended state when a higher priority job needs the resources.
T	Job is in Transition (being moved to a new location).
U	Job is User-suspended.
W	Job is Waiting for its requested execution time to be reached or job specified a staging request which failed for some reason.
X	Sub jobs only; sub job is finished (expired).

Queue Limits

	express	serial	short	medium	long
Priority	161	140	160	160	160
Max CPU per job	500	1	200	100	40
Max Node	29	29	29	29	29
Min Walltime (hr)	1	1	8	24	72
Max Walltime (hr)	8	168	24	72	168
Default Walltime (hr)	1	24	24	24	72
Default Memory (gb)	2	2	2	2	2
Max Running Jobs	500	450	450	400	200
Max Queued Jobs	20000	10000	20000	5000	2000

Queue Scheduling Issues

The scheduling algorithm used on the HPC aims to:

- promote large scale parallel use of the HPC
- allow equal access to resources for all users
- provide good turnaround for all users
- minimize the impact of jobs on one another

Some of the scheduler features to achieve these aims are:

- resources are strictly allocated so jobs will not start unless there is sufficient free resources (e.g. cpus and memory).
- queued jobs are shuffled so that jobs from different users are "interleaved". This means your first job should appear near the top of the queue even if there are many jobs in the queue.

From a user's perspective, it is very important that you minimize your requests for resources (e.g. walltime, memory and cpus). Otherwise your job may be queued or suspended longer than necessary. Of course, make sure you ask for sufficient resources - a little experimentation might help.

PBS Variables

PBS sets multiple environment variables at submission time. The following PBS variables are commonly used in command files:

Variable Name	Description
PBS_ARRAYID	Array ID numbers for jobs submitted with the -t flag. For example a job submitted with #PBS -t 1-8 will run eight identical copies of the shell script. The value of the PBS_ARRAYID will be an integer between 1 and 8.
PBS_ENVIRONMENT	Set to PBS_BATCH to indicate that the job is a batch job; otherwise, set to PBS_INTERACTIVE to indicate that the job is a PBS interactive job.
PBS_JOBID	Full jobid assigned to this job. Often used to uniquely name output files for this job, for example: *mpirun -np 16 ./a.out >output.${PBS_JOBID}*
PBS_JOBNAME	Name of the job. This can be set using the -N option in the PBS script (or from the command line). The default job name is the name of the PBS script.
PBS_NODEFILE	Contains a list of the nodes assigned to the job. If multiple CPUs on a node have been assigned, the node will be listed in the file more than once. By default, *mpirun* assigns jobs to nodes in the order they are listed in this file
PBS_O_HOME	The value of the HOME variable in the environment in which qsub was executed.
PBS_O_HOST	The name of the host upon which the qsub command is running.
PBS_O_PATH	Original PBS path. Used with pbsdsh.
PBS_O_QUEUE	Queue job was submitted to.
PBS_O_WORKDIR	PBS sets the environment variable *PBS_O_WORKDIR* to the directory from which the batch job was submitted
PBS_QUEUE	Queue job is running in (typically this is the same as PBS_O_QUEUE).

Interactive PBS Jobs

Use of PBS is not limited to batch jobs only. It also allows users to use the compute nodes interactively, when needed. For example, users can work with the developer environments provided by Matlab or R on compute nodes, and run their jobs (until the walltime expires). Instead of preparing a submission script, users pass the job requirements directly to the qsub command. For instance, the following PBS script:

```
#PBS -l nodes=7:ppn=4
#PBS -l mem=2gb
#PBS -l walltime=15:00:00
#PBS -q default
```

This corresponds to:

```
qsub -I -X -q default -l select=7:ncpus=4,walltime=15:00:00,mem=2gb
```

Hence, the PBS scheduler will allocate 7*4=28 cores to the user as soon as nodes with given specifications become available, then automatically log the user into one of the compute nodes. From now on, the user can work interactively using these cores until the walltime expires. Note that there should be no space between the parameters being passed to -l (as in 'L'ima) flag, only commas!

Here, -I (as in 'I'ndia) stands for 'interactive' and -X allows for GUI applications.

PBS Job Dependencies

In some situations a job or jobs will be dependent on the output of another job in order to run. To add a job dependency, the option *-W [additional attributes]* is used when submitting a job. In the example below the afterok rule will be used, but there are several other rules that may be useful. In this example two PBS command files will be used:

- number.pbs - generates a list of numbers in the file number.list
- order.pbs - sorts the list of numbers generated by number.pbs

If both jobs were submitted as:

```
[username@hpc-login-prd-t1 ~]$ qsub number.pbs ; qsub order.pbs
```

the error output from order.pbs will be *order: open failed: number.list: No such file or directory* If order.pbs was submitted with a dependency on number.pbs as in:

```
[username@hpc-login-prd-t1 ~]$ qsub number.pbs
4674.hpc-admin-prd-t1
[username@hpc-login-prd-t1 ~]$ qsub -W depend=afterok:4674 order.pbs
4675.hpc-admin-prd-t1
[username@hpc-login-prd-t1 ~]$ qstat -u $USER



hpc-admin-prd-t1.usq.edu.au:

                                                                  Req'd  Req'd   Elap
Job ID               Username Queue    Jobname          SessID NDS   TSK Memory Time  S Time
-------------------- -------- -------- ---------------- ------ ----- --- ------ ----- - -----
4674.hpc-admi        username short number.pbs        18029     1   1    --  48:00 R 00:00
4675.hpc-admi        username short order.pbs                   1   1    --  48:00 H   --
```

Notice the order.pbs is in a hold state however once the dependent job completes the order job run as:

```
[username@hpc-login-prd-t1 ~]$ qstat -u $USER

hpc-admin-prd-t1.usq.edu.au:
                                                                  Req'd  Req'd   Elap
Job ID               Username Queue    Jobname          SessID NDS   TSK Memory Time  S Time
-------------------- -------- -------- ---------------- ------ ----- --- ------ ----- - -----
4675.hpc-admi        username short order.pbs                   1   1    --  48:00 R   --
```

Other options to -W include:

- afterany:jobid[:jobid...] implies that the job may be scheduled for execution after jobs jobid have terminated, with or without errors.
- afterok:jobid[:jobid...] implies that job may be scheduled for execution only after jobs jobid have terminated with no errors.
- afternotok:jobid[:jobid...] implies that job may be scheduled for execution only after jobs jobid have terminated with errors.

References:

1. PBS Professional 14 User Guide
2. PBS Professional 14 Administrator's Guide
3. PBS Professional - HPC Cluster Workload Manager