Monitoring batch jobs

Tutorial 2:
HPC at NYU

Back to the NYU HPC Wiki
Back to Tutorials index
Back to the Introduction

Prequel: preparing your workstation for the HPC access (and this tutorial)

Accessing software with Environment Modules
Job scripts and how to reserve resources

Introduction to job scheduling
Submitting a job with qsub
Requesting resources

Requesting GPUs
Using compute nodes interactively
Where is my job in the queue - or on the system - and why?

The simplest queuing algorithm is "first come, first served". The queuing of jobs on the HPC cluster is a little more sophisticated as we pursue several goals:

- Minimal queuing times, especially for short jobs. Nobody wants to spend 4 hours in the queue for a 1-hour job.
- Efficient use of the available resources. If there is a job ready which can use hardware that would otherwise be idle, run it, even if it's not next in the queue.
- Fair use of resources. If you've made heavy use of the cluster recently, jobs belonging to a user who has had less CPU time will get higher priority.

At NYU "recently" means "the last 24 hours", so users with large workloads are not excessively penalized.
Special consideration for HPC Stakeholders. NYU HPC uses a “condo” model in which we manage HPC resources owned by specific schools and departments, in exchange for allowing the rest of the NYU HPC community to use those resources when the owners are not.

Moab supports these goals by calculating a priority for each submitted job and placing the job in the queue according to its priority. The schedule of which job will run where and when is built from the job queues. When a job finishes earlier than scheduled (due to an overestimated walltime request), Moab attempts to fill the newly-available space by scanning the queue for the first job which will fit without delaying an already-scheduled, higher priority job. In this way low-priority jobs with smaller resource requirements can jump ahead and be run early.

Monitoring jobs with qstat

To see the status of a single job - or a list of specific jobs - pass the Job IDs to `qstat`, as in the following example:

```
$ qstat 3593014 3593016
```

```
Job id       Name             User            Time Use S Queue
------------- ---------------- --------------- -------- - ----- 
3593014      model_scen_1     ab123            7:23:47 R s48
3593016      model_scen_1     ab123            7:23:26 R s48
```

Most of the fields in the output are self-explanatory. The second-last column "S" is the job status, which can be:

- Q meaning "Queued"
- H meaning "Held" - this may be the result of a manual hold or of a job dependency
- R meaning "Running"
- C meaning "Completed". After the job finishes, it will remain with "completed" status for a short time before being removed from the batch system.

Other, less common job status flags are described in the manual (`man qsub`).

The `qstat` command is described in more detail here.

Exercise

Request an interactive batch session on Mercer. When it starts (or immediately, in another window), use `qstat -u` to see it in the queue. Try `qstat -f <job_id>` too.

Next: Interpreting pbsnodes