## Submitting a job - Resource limits

<table>
<thead>
<tr>
<th>Quick Links</th>
</tr>
</thead>
<tbody>
<tr>
<td>HPC at NYU</td>
</tr>
<tr>
<td>HPC Newsletter</td>
</tr>
<tr>
<td>Getting started on Prince</td>
</tr>
<tr>
<td>Getting an account</td>
</tr>
<tr>
<td>Prince How-to Articles</td>
</tr>
<tr>
<td>Logging in Windows</td>
</tr>
<tr>
<td>Mac / Linux Clusters and Storage July 2017</td>
</tr>
<tr>
<td>Mercer (HPC, retired)</td>
</tr>
<tr>
<td>Prince (HPC)</td>
</tr>
<tr>
<td>Dumbo (Hadoop)</td>
</tr>
</tbody>
</table>
Dalmatia (NYU Abu Dhabi)

Transferring data to/from the clusters
Transferring data to/from Prince cluster using Globus
Submitting jobs with sbatch
Available software
Building Software Tutorials
Programming for Biologists
Research Gallery
FAQs
NYU HPC Cheat Sheet
Wiki Map
HPC Policies
Running jobs on the NYU HPC clusters

Job Scheduling Overview

Login and compute nodes

Queues

Writing a job script

Job stdout and stderr

Which filesystems should I use?

Submitting a Job

Basic qsub options and directives

Setting resource limits

Running MPI jobs

Running GPU jobs

Running R jobs

Running Matlab, Gaussian, etc jobs

Working interactively
Jobs are submitted with the \texttt{qsub} command:

\begin{verbatim}
$ qsub options job-script
\end{verbatim}

The options tell Torque information about the job, such as what resources will be needed. These can be specified in the job-script as PBS directives, or on the command line as options, or both (in which case the command line options take precedence should the two contradict each other). For each option there is a corresponding PBS directive with the syntax:

\begin{verbatim}
#PBS option
\end{verbatim}

For example, you can specify that a job needs 2 nodes and 8 cores on each node by adding to the script the directive:
#!/bin/bash
#PBS -l nodes=2:ppn=8

or as a command-line option to qsub when you submit the job:

```
$ qsub -l nodes=2:ppn=8 my_script.q
```

What resources should I request?

At a minimum, the scheduler needs to know:

- How many CPU cores you need, and whether they must be on the same node
  - If you don't know, the answer is probably “1 core”. If the program supports parallel, it probably supports multithreading - multiple cores on a single node. To use multiple nodes, a program generally needs MPI, so you should only request multiple nodes if you are sure the program can use them.
- How much memory you need
  - NYU has nodes with 23GB, 46GB, 62GB, 90GB, 120GB and 189GB available to jobs. (The remaining memory is needed by the operating system)
- How long the job is expected to take
  - NYU HPC users can request up to 168 hours (1 week) for a single job. But priority is given to jobs requesting less time

CPUs - nodes and cores

HPC is associated with parallel processes - but it's not magic! To use multiple CPUs, the program must have been written with either threading (eg OpenMP) or message passing (MPI).

If in doubt:
- try with 1 core first
How much do I need?

The HPC cluster is not magic, its CPUs are only as fast as any other contemporary CPU. In fact, some nodes on Mercer are a few years old, and may be slower than your desktop (see Clusters July 2017 for a table of node types and when we installed them).

The performance of the HPC cluster comes from its scale. Most nodes in the cluster have 12 or 20 cores, 48GB up to 192GB of RAM, access to a large fast parallel filesystem and there is a 40Gb/s dedicated network link between any two nodes in each of the main groups. And there are thousands of nodes.

So although the resources your job needs depend very much on your job, and there is no simple rule for estimating requirements, you can make some initial guesses based on why you need the HPC cluster:

- **My desktop has not enough RAM**
  You should request at least as much RAM as your desktop possesses. The time required will probably be similar to the time required on your desktop. Be aware though that many problems scale with $O(n^2)$ or more, so doubling the number of data points might require 4x the RAM and 8x the compute time.

- **Each run takes 4 hours on my 32GB desktop, and I have 1000 experiments to run**
  Each experiment will probably take 32GB of memory and 4 hours on the HPC cluster too - but you can submit 1000 of them at once and a few hundred might run simultaneously

For a few one-off jobs, you can safely request much more than you need. But use these qsub/PBS options:

```
#PBS -mae               # send an email if the job aborts (a) and when it ends (e)
#PBS -M $USER@nyu.edu   # send the email to this address
```

In the email you will see something like:

```
PBS Job Id: 3205348
Job Name:   testme.q
Exec host:  compute-1-10.local/7
Execution terminated
Exit_status=0
resources_used.cput=00:32:01
resources_used.mem=3052mb
resources_used.vmem=25644mb
resources_used.walltime=00:42:03
Error_Path: login-0-2.local:/home/ab123/testing/testme.q.e3205348
Output_Path: login-0-2.local:/home/ab123/testing/testme.q.o3205348
```

From which you can deduce that the job took 42 minutes of wallclock time and about 3GB of memory. So a sensible resource request for the next job is (see RESOURCES for more about the options):

```
qsub -l walltime=1:00:00 -l mem=4GB testme.q
```

Requesting exclusive use of nodes

There are two ways you can ensure you will not share a node with other jobs:

- check the documentation of the software, and next try with multiple cores on 1 node
- when using multiple nodes, check whether the job is actually running on all of the nodes. Contact us for help with this.
• Use the "-n" qsub option
• Request nodes with "-l nodes=num:ppn=all"
  Note that with the second option, you do not know in advance how many cores will be available on the nodes your job is allocated to.
  More on this below.

Until recently, requesting nodes but not ppn gave whole nodes. However this led to users inadvertently requesting full nodes, so now to request full node you use "ppn=all" instead

### PBS directives and qsub options for setting resource limits

#### Options to request compute resources:

- `-l walltime=walltime`
  Maximum wallclock time the job will need. Default depends on queue, mostly 1 hour. Walltime is specified in seconds or as hh:mm:ss or mm:ss.
- `-l mem=memory`
  Maximum memory per node the job will need. Default depends on queue, normally 2GB for serial jobs and the full node for parallel jobs. Memory should be specified with units, eg 500MB or 8GB. The available memory per node for different nodes of Mercer is described here.
- `-l procs=num`
  Total number of CPUs required. Use this if it does not matter how CPUs are grouped onto nodes - eg, for a purely-MPI job. Don’t combine this with `-l nodes=num` or odd behavior will ensue.
- `-l nodes=num:ppn=num`
  Number of nodes and number of processors per node required. Use this if you need processes to be grouped onto nodes - eg, for an MPI/OpenMP hybrid job with 4 MPI processes and 8 OpenMP threads each, use `-l nodes=4:ppn=8`. Don’t combine this with `-l procs=num` or odd behavior will ensue. Default is 1 node and 1 processor per node. When using multiple nodes the job script will be executed on the first allocated node. Torque will set the environment variables `PBS_NUM_NODES` to the number of nodes requested, `PBS_NUM_PPN` to the value of `ppn` and `PBS_S_NP` to the total number of processes available to the job.
- `-l nodes=num:ppn=num:gpus=num`
  If your job needs GPUs, you should specify the number of GPUs per node in the `-l nodes` option. We have nodes with older Tesla GPUs and newer Titan GPUs - if you specifically need the newer GPUs you should add the requirement ":titan" to the gpus specification.
- `-l nodes=num:ppn=all`
  This is an NYU HPC extension: if you need whole nodes but do not mind how many cores, you can request full nodes this way. Specifying `-n` will have the same effect. The environment variable `$PBS_PPN` will be set in the job to the number of cores on each node the job is running on. Jobs are always allocated to sets of nodes with the same number of cores, so you will not get one node with 12 cores and another with 20. Note that this currently only works when used in directives, not on the command line.
- `-n` Request exclusive use of nodes. If this is specified, no other jobs will share a node with this job, and if you did not specify a memory limit, no memory limit will be enforced (Note however that if you do not specify a memory limit, you may land on a node with only 24GB of memory)
- `-q queue`
  Submit to a specific queue. If not specified, Torque will choose a queue based on the resources requested.

Resources can be requested with multiple `-l` options, or as a comma-separated list of options. Both of the following examples are correct:

```bash
$ qsub -l walltime=2:00:00 -l mem=4GB myjob.q

$ qsub -l walltime=2:00:00,mem=4GB myjob.q
```
The serial queues on NYU HPC clusters are limited to a single node, but allow multiple processors on that node to be used. Therefore, parallel jobs using only one node, such as OpenMP or multithreaded jobs can be submitted to a serial queue.

When using more than one node, the job script is executed only on the first node. To make use of the other nodes you must use MPI or pbsdsh.

The MPI modules on Bowery are not compiled with support for Torque, so the CPU time reported by Torque will include only the CPU time for the first MPI process. For an estimate of the true total CPU time, use the elapsed (wallclock) time multiplied by the number of nodes. This issue has been solved on Mercer.