Introduction to Discovery

http://discovery.dartmouth.edu
The Discovery Cluster
Agenda

• What is a cluster and why use it
• Overview of computer hardware in cluster
• Help Available to Discovery Users
• Logging on to the cluster with “ssh”
• Transferring files to and from the cluster
• The Environment
• Scheduler basics
• Requesting resources - PBS scripts
• Checking on submitted jobs
• Cluster Etiquette - running jobs & disk space
• Publishing
• Labs
Why Would You Need to Use Discovery?

- Your program runs for a LONG time
- Your program needs a lot of memory
- You need to run your program many times
- Your data files use up a lot of disk space
- You need to run your program in parallel
Cluster Nodes

Cell D: AMD 4284 3.0GHz Dual Octa-Core (16-cores)
Cell E: AMD 4286 3.1GHz Dual Octa-Core (16-cores)
Cell F: AMD 6348 2.8GHz Quad Dodeca-Core (48-cores)
Cell G: 2 NVidia K80 GPUs, Intel E5-2640 (16-cores) [coming soon]
Cell H: Intel Xeon E5-2470 2.3GHz Dual 8-Core (16-cores)
Cell J: Intel Xeon E5-2690 2.6GHz Dual 12-core (24 cores)
Cell K: Intel Xeon E5-2640V3 2.6GHz Dual 8-Core (16-cores)
Help Available for Discovery Users

- Build and install requested applications
- Help getting your applications running
- Specialized help from RC application specialists:
  - Bioinformatics
  - Debugging, optimizing and parallelizing code
  - GIS
  - Statistics
  - Python, R, Java, C/C++, Fortran, Matlab
- Help setting up shared data repositories for research groups
Logging On

- SSH (Secure Shell)
  - Linux: `ssh -X username@discovery.dartmouth.edu`
  - Mac: `ssh -Y username@discovery.dartmouth.edu`
  - Windows
    - MobaXterm built in Xserver and sftp (free and recommended)
    - Ssh secure shell or putty

- Changing your password
  - Use the `passwd` command to make the change.
Transferring Files To/From Discovery (CLI)

- **Linux or Mac (CLI): sftp & scp**
  - CLI secure file transfer program – “sftp”
  - `sftp username@discovery.dartmouth.edu`
  - Use `put`, `get`, `mput` & `mget`
    - `put filename (mput filenames*)`
    - `get filename (mget filenames*)`
  - To copy from outside machine to discovery
    - `scp file(s) username@discovery.dartmouth.edu:`
    - `scp -r dir username@discovery.dartmouth.edu:`
      - `dir` will be created in your HOME directory on the cluster.
Transferring Files To/From Discovery (GUI)
GUI SFTP clients

- Windows
  - MobaXterm
  - WinSCP
- Macintosh
  - Fetch
- Both
  - Filezilla
  - Cyberduck
Your Environment

BASH

- The bash shell is the default shell you will be using on Discovery. The environment is tailored to use this shell.

- If you change to some other shell then queuing jobs, compiling parallel code is not guaranteed to work.

- **Warning**: Do not replace your .bashrc or .bash_profile files. Only add to them.
Using Modules to Manage Software

- The Discovery cluster uses modules to manage the user environment for different third-party software versions.
- The advantage of the modules approach is that the user is no longer required to specify paths for different versions, and to try to keep the PATH, MANPATH and related variables coordinated.
- With the modules approach, users simply "load" and "unload" modules to control their environment.
Environment Modules II

- Module commands
  - To get a usage list of module options type the following (the listing has been abbreviated to only those commands discussed in this webpage):
    - `$ module help`

Available Commands and Usage:

```
add|load     modulefile [modulefile ...]
rmd|unload   modulefile [modulefile ...]
switch      modulefile1 modulefile2
display     modulefile [modulefile ...]
avail       path [path]
list        
initadd     modulefile [modulefile ...]
help        modulefile [modulefile ...]
```
Rstor Files

• If you have an AFS account...

• In order to have write access, to your AFS directory, you will need to use the **klog** command.

• The **klog** command will prompt you for your AFS account password.

• Once you have done this, you can use your AFS account to archive data and files from discovery.

• AFS is only available from the discovery head node. It is not available from the compute nodes.
Disk Space

- You have write access to
  - $HOME – your home directory (shortcut: ~)
  - /scratch (local to nodes)
    - /scratch should be used for intermediate storage of the job data, if possible.
  - /global/scratch (central scratch)
    - Data in /scratch and /global/scratch cleaned by the system after 7 days.
  - /global/data (members data space)
    - If you are part of a member's Discovery account (qr command)

- Home directories backed up daily offsite
  - Snapshots taken daily, weekly & monthly and are available in your
    - $HOME/.zfs/snapshot directory (if home path is /home or /cgl/home)
    - $HOME/.snapshot (if home path is /ihome)
Disk Space II

Disk quotas

- $HOME (20GB)
  - Email sent if quota usage reaches 95%
  - Use `quota` command to view your usage
- `/global/data`
  - Quota dependent on members investment
- `/scratch (no quota enforced)`
  - Please have job cleanup
- `/global/scratch (no quota enforced)`
  - Please have job cleanup
Disk Space III

- If you need to store large quantities of data, we will work with you to arrange alternatives most suited to your needs.
- When over quota you can't write any files and sometimes can't login
- Don't go over your quota
Publishing your work

- Discovery provides you a website to publish your work.
- The contents of your website is kept in a subdirectory below your HOME directory called `public_html`.
- The directory should be created as follows:
  ```
  $ mkdir -m 711 ~/public_html
  ```
- URL: `http://discovery.dartmouth.edu/~username/`
How to Get Started Running on Discovery

- Install your program(s) and copy any data to Discovery
- Run your program interactively on test nodes
- Debug your program if necessary
- Monitor and time your application
- Write a submit script and submit a sample job
- Look at job output and debug submit script
- Submit and monitor your job(s)
Scheduler Basics

- Scheduling jobs
- PBS scripts
- Resources available
- Using the scheduler
How The Scheduler Works

- Submit jobs to the scheduler - PBS scripts
- Torque – resource manager
  - Controls when and where jobs will run.
  - Does the work of putting the jobs on the nodes.
- Moab – job scheduler
  - Controls who can run on what resources for up to some period of time.
  - Determines Policies and Limits
- Priority, core count and walltime is based on your status
  - Part of a Membership Account (Buy-in)
  - Part of a Grant Account (3-months)
  - Part of a Free Access Account
Example PBS Script

#!/bin/bash
#
# declare a name for this job to be sample_job
#PBS -N my_serial_job
# request the queue (enter the possible names, if omitted, default is the default)
# if more then 600 jobs use the largeq
#PBS -q default
# request 1 core on 1 node
# ensure you reserve enough cores for the projected memory usage
# figuring 4G/core
#PBS -l nodes=1:ppn=1
# request 4 hours and 30 minutes of wall time
#PBS -l walltime=04:30:00
# mail is sent to you when the job begins and when it exits or aborts
# you can use all or some or none. If you don't want email leave this
# and the following (#PBS -M) out of the script.
#PBS -m bea
# specify your email address
#PBS -M John.Smith@dartmouth.edu
# By default, PBS scripts execute in your home directory, not the
# directory from which they were submitted. The following line
# places you in the directory from which the job was submitted.
cd $PBS_O_WORKDIR
# run the program
./program_name arg1 arg2 ...

07/26/16
Using The Scheduler

- `qsub pbs_script_filename` submit job
- `myjobs [-rn]` view job(s) status
- `qshow [-r]` view queue status
- `pbsmon` view nodes & status
- `checkjob [-v] jobID` view job(s) status
- `qr` view your resources
- `qdel jobID` remove job
- `qnotify` notify near run end
Things to Check Before Job Submission

● Have I saved all results (data and graphics)?

● Have I requested enough time?
  – #PBS -l walltime=2:00:00 (hr:min:sec)

● Have I requested enough cores?
  – Specify 1 core per 4GB of memory usage

● Have I specified any other needed features?
  – #PBS -l feature='cellk'
Diagnosing Problems

Blocked jobs

- Use `checkjob -v` see the reason
- Try changing parameters and resubmitting

Jobs that do not return results

- Contact research.computing@dartmouth.edu

Out of disk space (quota)

- The `quota` command will show your usage
- `/scratch` can also fill up (have job clean up)
- This condition can cause errors that are very hard to diagnose
Scheduler Etiquette

Our goal is to provide fair use of the resources

Stage large quantity job submissions

- If more than 600 jobs, use the largeq (routing queue)

To maximize your use of the available resources

- Start modestly - test new or unfamiliar code
- Use test nodes x01, x02 or x03 for testing and timing
- Use top or htop on Test nodes to check performance
- Use pmap to test memory usage.
  - pmap <process-id>
Scheduler Etiquette II

- To maximize your use of the available resources (cont’d)
  - Know your code and what your cluster resources are
    - The `qr` (queue resources) command can help
  - Know cluster policies on runtime and resource limitations
    - available on the Discovery website
    - http://discovery.dartmouth.edu
  - Plan ahead for long jobs
    - Are the resources available?
  - If possible, compile code on the cluster
  - Ask us (research.computing@dartmouth.edu)
    - if you must run in an unusual way
Discovery: Helpful Commands

- myjobs [-rbi]
- tnode-load
- quota
- pbsmon
- features [-h][-a] <feature>
- qr [-h]
- qshow [-r]
- qnotify job-id hour(s)
### myjobs

- **myjobs [-rn]**

```bash
$ myjobs

#### active jobs

<table>
<thead>
<tr>
<th>JOBID</th>
<th>USERNAME</th>
<th>STATE</th>
<th>PROCS</th>
<th>REMAINING</th>
<th>STARTTIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>3810851</td>
<td>ryanu</td>
<td>Running</td>
<td>1</td>
<td>14:09:05</td>
<td>Mon Mar 22 02:55:08</td>
</tr>
<tr>
<td>3810867</td>
<td>ryanu</td>
<td>Running</td>
<td>1</td>
<td>14:38:28</td>
<td>Mon Mar 22 03:24:31</td>
</tr>
<tr>
<td>3810873</td>
<td>ryanu</td>
<td>Running</td>
<td>1</td>
<td>14:52:15</td>
<td>Mon Mar 22 03:38:18</td>
</tr>
</tbody>
</table>

3 active jobs

3 of 1548 processors in use by local jobs (0.33%)

88 of 114 nodes active (77.19%)

#### eligible jobs

<table>
<thead>
<tr>
<th>JOBID</th>
<th>USERNAME</th>
<th>STATE</th>
<th>PROCS</th>
<th>WCLIMIT</th>
<th>QUEUETIME</th>
</tr>
</thead>
</table>

0 eligible jobs

#### blocked jobs

<table>
<thead>
<tr>
<th>JOBID</th>
<th>USERNAME</th>
<th>STATE</th>
<th>PROCS</th>
<th>WCLIMIT</th>
<th>QUEUETIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>3811629</td>
<td>ryanu</td>
<td>Idle</td>
<td>1</td>
<td>1:00:00:00</td>
<td>Mon Mar 22 09:59:23</td>
</tr>
<tr>
<td>3811630</td>
<td>ryanu</td>
<td>Idle</td>
<td>1</td>
<td>1:00:00:00</td>
<td>Mon Mar 22 10:00:23</td>
</tr>
<tr>
<td>3811633</td>
<td>ryanu</td>
<td>Idle</td>
<td>1</td>
<td>1:00:00:00</td>
<td>Mon Mar 22 10:07:53</td>
</tr>
</tbody>
</table>

3 blocked jobs

Total jobs: 6
tnodeload

$ tnodeload

<table>
<thead>
<tr>
<th>Node</th>
<th>Users</th>
<th>Load</th>
<th>Memory</th>
<th>Scratch</th>
<th>Speed</th>
<th>Max</th>
<th>Chip Set</th>
</tr>
</thead>
<tbody>
<tr>
<td>x01</td>
<td>0</td>
<td>0.04</td>
<td>64.5G</td>
<td>779G</td>
<td>2.4GHz</td>
<td>2.4GHz</td>
<td>AMD Opteron(tm) Processor 6136</td>
</tr>
<tr>
<td>x02</td>
<td>0</td>
<td>0.00</td>
<td>64.5G</td>
<td>779G</td>
<td>2.4GHz</td>
<td>2.4GHz</td>
<td>AMD Opteron(tm) Processor 6136</td>
</tr>
<tr>
<td>x03</td>
<td>1</td>
<td>0.00</td>
<td>64.6G</td>
<td>779G</td>
<td>2.4GHz</td>
<td>2.4GHz</td>
<td>AMD Opteron(tm) Processor 6136</td>
</tr>
</tbody>
</table>
$ quota

User: pete

----- -----
Quota: 20G
Used: 12G
Available: 8.7G
Use: 57%
$ quota

User: pete
----- -----
Quota: 20G
Used: 19G
Available: 2.0G
Use: 95%
pbsmon

------------------------
nodes free : 54
<= 50% cores in use : 12
> 50% cores in use : 12
nodes down : 9
100% cores in use : 64
Total cores in use : 1134
```bash
[pete@discovery ~]$ features -h
Syntax: features [-a] [-h] [-f feature]

Providing the -h option prints this help message.

If given the option "-a" then all features and their available resources are displayed. The Totals are the count of both the amd and intel features.

If given one of the following features as an argument to "-f", then that feature's available resources will be displayed.

Features Available: cella cellb cellc celld celle cellf cellh ib2 amd intel

Current Feature Assignments:

<table>
<thead>
<tr>
<th>Feature</th>
<th>Resources</th>
</tr>
</thead>
<tbody>
<tr>
<td>a01-a04,a13-a21:</td>
<td>cella,amd (Opteron 2.7Ghz 32G RAM 8-cores)</td>
</tr>
<tr>
<td>b01-b16:</td>
<td>cellb,intel (Xeon Nahalem 2.3Ghz 32G RAM 8-cores)</td>
</tr>
<tr>
<td>c01-c27:</td>
<td>cellc,amd (Opteron 2.4Ghz 64G RAM 16-cores)</td>
</tr>
<tr>
<td>d01-d39:</td>
<td>celld,amd (Opteron 3.0Ghz 64G RAM 16-cores)</td>
</tr>
<tr>
<td>d01-d24:</td>
<td>ib2 (Infiniband)</td>
</tr>
<tr>
<td>e01-e34:</td>
<td>celle,amd (Opteron 3.1Ghz 64G RAM 16-cores)</td>
</tr>
<tr>
<td>f01-f08:</td>
<td>cellf,amd (Opteron 2.8Ghz 192G RAM 48-cores)</td>
</tr>
<tr>
<td>h01-h08:</td>
<td>cellh,intel (Xeon 2.5Ghz 64G RAM 16-cores)</td>
</tr>
</tbody>
</table>
```
qr  (queue resources)

<table>
<thead>
<tr>
<th>Account</th>
<th>Owned CPUs</th>
<th>MAX CPUs</th>
<th>UserMAX CPUs</th>
<th>MAX Wall</th>
<th>UserMAX Wall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Moore</td>
<td>796</td>
<td>1811</td>
<td>400</td>
<td>204480</td>
<td>102240</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Account</th>
<th>Wall</th>
<th>Jobs</th>
<th>CPUs</th>
<th>FS %</th>
</tr>
</thead>
<tbody>
<tr>
<td>Moore</td>
<td>44109</td>
<td>895</td>
<td>1270</td>
<td>35.0</td>
</tr>
</tbody>
</table>

pete's Usage/Availability

<table>
<thead>
<tr>
<th>Rem Wall</th>
<th>Running Jobs</th>
<th>Used CPUs</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

pete's Blocked Jobs

<table>
<thead>
<tr>
<th>Wall</th>
<th>Jobs</th>
<th>CPUs</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
qhist  (review past usage)

The purpose of qhist is to provide statistics on past cluster usage. It can report a summary or line-by-line report for a time period. It can also provide a more detailed report on a single job.

$ qhist -h

qhist -h     Syntax
qhist -j <jobnum>    Single Job Report
qhist -r    Job records
qhist -s     Summary
### qshow

<table>
<thead>
<tr>
<th>User</th>
<th>Running Jobs</th>
<th>Running CPUs</th>
<th>Blocked Jobs</th>
<th>Blocked CPUs</th>
<th>Eligible Jobs</th>
<th>Eligible CPUs</th>
</tr>
</thead>
<tbody>
<tr>
<td>aglaser</td>
<td>5</td>
<td>80</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>bzhuy</td>
<td>3</td>
<td>48</td>
<td>1</td>
<td>16</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>ccheng</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>chandana</td>
<td>2</td>
<td>48</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>denton</td>
<td>1</td>
<td>48</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>dfisher</td>
<td>1</td>
<td>64</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>ebrahimi</td>
<td>1</td>
<td>16</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>pandrews</td>
<td>75</td>
<td>75</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>piotr</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>qpan</td>
<td>400</td>
<td>400</td>
<td>198</td>
<td>198</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>rhughes</td>
<td>6</td>
<td>24</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>roberdt</td>
<td>18</td>
<td>72</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>ryanu</td>
<td>400</td>
<td>400</td>
<td>200</td>
<td>200</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>rzhong</td>
<td>11</td>
<td>110</td>
<td>2</td>
<td>20</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>tingh</td>
<td>25</td>
<td>400</td>
<td>68</td>
<td>1088</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>952</strong></td>
<td><strong>1789</strong></td>
<td><strong>469</strong></td>
<td><strong>1522</strong></td>
<td><strong>0</strong></td>
<td><strong>0</strong></td>
</tr>
</tbody>
</table>
$ qnotify

Syntax: qnotify job-id hours
qnotify -l (list notifications)

$ qnotify 3872942 1

QNotify will notify you when there are about 1 hours of walltime remaining on job 3872942.

$ qnotify -l

<table>
<thead>
<tr>
<th>JobID</th>
<th>Remaining</th>
<th>Notify</th>
</tr>
</thead>
<tbody>
<tr>
<td>3872942</td>
<td>1:59:20</td>
<td>1</td>
</tr>
</tbody>
</table>
```
qshow -r

<table>
<thead>
<tr>
<th>User</th>
<th>Running Jobs</th>
<th>Running CPUs</th>
<th>Blocked Jobs</th>
<th>Blocked CPUs</th>
<th>Eligible Jobs</th>
<th>Eligible CPUs</th>
<th>Routing Jobs</th>
</tr>
</thead>
<tbody>
<tr>
<td>aglaser</td>
<td>5</td>
<td>80</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>bzhui</td>
<td>3</td>
<td>48</td>
<td>1</td>
<td>16</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>ccheng</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>chandana</td>
<td>2</td>
<td>48</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>denton</td>
<td>1</td>
<td>48</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>dfisher</td>
<td>1</td>
<td>64</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>ebrahimi</td>
<td>1</td>
<td>16</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>pandrews</td>
<td>75</td>
<td>75</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>piotr</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>qpan</td>
<td>400</td>
<td>400</td>
<td>199</td>
<td>199</td>
<td>0</td>
<td>0</td>
<td>11351</td>
</tr>
<tr>
<td>rhughes</td>
<td>6</td>
<td>24</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>robertd</td>
<td>18</td>
<td>72</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>ryanu</td>
<td>400</td>
<td>400</td>
<td>200</td>
<td>200</td>
<td>0</td>
<td>0</td>
<td>633</td>
</tr>
<tr>
<td>rzhang</td>
<td>11</td>
<td>110</td>
<td>2</td>
<td>20</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>tingh</td>
<td>25</td>
<td>400</td>
<td>68</td>
<td>1088</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Total: 952 Jobs, 1789 CPUs, 470 Jobs, 1523 CPUs, 0 Jobs, 0 CPUs, 11984 Jobs
```
Summary

- Cluster introduction
- Connecting/Transferring data
- Environment settings
- Submitting jobs (PBS script, qsub)
- Checking jobs
- Usage policies and etiquette overview
  - submitting jobs etiquette
  - monitoring disk usage