Grace Hardware

Grace is a 925-node Intel cluster from Dell with an InfiniBand HDR-100 interconnect, A100 GPUs, RTX 6000 GPUs and T4 GPUs. The 925 nodes are based on the Intel Cascade Lake processor.

For more information:
https://hprc.tamu.edu/kb/User-Guides/Grace/

<table>
<thead>
<tr>
<th>Resource</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>Login Nodes</td>
<td>5</td>
</tr>
<tr>
<td>384GB memory general compute nodes</td>
<td>800</td>
</tr>
<tr>
<td>GPU - A100 nodes with 384GB memory</td>
<td>100</td>
</tr>
<tr>
<td>GPU - RTX 6000 nodes with 384GB memory</td>
<td>9</td>
</tr>
<tr>
<td>GPU - T4 nodes with 384GB memory</td>
<td>8</td>
</tr>
<tr>
<td>3TB Large Memory</td>
<td>8</td>
</tr>
</tbody>
</table>
Accessing Grace: Setup

- If off-campus:
  Set up and start VPN (Virtual Private Network): u.tamu.edu/VPnetwork
- **Two-Factor Authentication** required

- Today we’ll access Grace via the online Portal, but you can also use ssh.
Accessing Grace via the Portal

Access the HPRC portals through most web browsers:
1. Go to portal.hprc.tamu.edu or use the Portal dropdown menu on the HPRC homepage: https://hprc.tamu.edu/
2. Choose Grace Portal

https://hprc.tamu.edu/kb/User-Guides/Grace/Access/
Accessing Grace via the Portal

Once in the Portal, select at the top: “Clusters” → “Grace Shell Access”

- `shell` is also called `terminal` or `command line`

https://hprc.tamu.edu/kb/User-Guides/Grace/Access/
Hands-On Activity - 2 Minutes

Try to access a *shell* on Grace now, either through portal.hprc.tamu.edu or hprc.tamu.edu

*(also called *terminal* or *command line)*

What message do you see when you login?

Remember Grace has 5 login nodes. Which one does your command prompt say you got?
# File Systems and User Directories

<table>
<thead>
<tr>
<th>Directory</th>
<th>Environment Variable</th>
<th>Space Limit</th>
<th>File Limit</th>
<th>Intended Use</th>
</tr>
</thead>
<tbody>
<tr>
<td>/home/$USER</td>
<td>$HOME</td>
<td>10 GB</td>
<td>10,000</td>
<td>Small to modest amounts of processing.</td>
</tr>
<tr>
<td>/scratch/user/$USER</td>
<td>$SCRATCH</td>
<td>1 TB</td>
<td>250,000</td>
<td>Temporary storage of large files for on-going computations. Not intended to be a long-term storage area.</td>
</tr>
</tbody>
</table>

$SCRATCH is shared between the FASTER and Grace clusters.

View file usage and quota limits using the command: `showquota`

Do NOT share your home or scratch directories.
Request a group directory for sharing files.

[https://hprc.tamu.edu/kb/User-Guides/Grace/Filesystems_and_Files/](https://hprc.tamu.edu/kb/User-Guides/Grace/Filesystems_and_Files/)
Preferred way to request Quota Increases

Easily view Cluster utilization, Storage Quotas, & Allocation Balances

Quota and file limit increases will only be considered for scratch directories

Request help or software
Hands-On Activity - 2 Minutes

1. Please try to access dashboard now through the portal.

2. Check your quotas both on the command line and on the dashboard.

   showquota
Software

- See the Software Knowledge Base page [https://hprc.tamu.edu/kb/Software/](https://hprc.tamu.edu/kb/Software/) for instructions and examples

- Search for software modules on [https://hprc.tamu.edu/software/grace/](https://hprc.tamu.edu/software/grace/)

- License-restricted software
  - Contact help@hprc.tamu.edu

- Contact HPRC (can use the dashboard) for software installation help/request
  - User can install software in their home/scratch directory
  - Do NOT run the sudo command when installing software
Software: Application Modules

- Installed applications are made available with the module system.
- Grace uses a *software hierarchy* inside the module system.
- In this hierarchy, the user loads a compiler which then makes available software built with the currently-loaded compiler.

```
module avail
```
← shows which software is available

```
module load GCC/10.3.0 OpenMPI/4.1.1
```
← load GCC compiler version 10.3.0 and OpenMPI version 4.1.1

```
module avail
```
← show which software is available to use with the loaded modules

```
module load PyTorch/1.12.1
```
← load PyTorch version 1.12.1

```
module list
```
← see what software you've loaded already
Software: Modules and Toolchains

- Toolchains are what we call groups of compilers & libraries
- There’s a variety of toolchains available on the clusters:
  - intel/2022a
  - iomkl/2020a
  - foss/2022a
  - GCCcore/11.3.0
- Module management:
  - module spider
  - module purge
    - search for modules and their dependencies
    - removes all loaded modules
  - (more than just these versions)
Hands-On Activity - 5 Minutes

Remember: [module load]  [module purge]

1. Please search for and load the following module:

   [OpenMPI/4.1.4]

   (Tip) Type this to show which compiler needs to be loaded:

   module spider OpenMPI/4.1.4

   (Tip) And check that it’s been loaded with:

   module list

2. Next remove (unload) all your current modules.
Two sets of nodes on our clusters:

- **Login nodes**: log in, perform basic commands, write job scripts, and send job scripts to...
- **Compute nodes**: executes jobs, sends results back to you
Batch Jobs on HPRC Clusters

You start on login nodes, but it’s the compute nodes that do the real heavy-lifting.

You are charged service units (SUs) based on how much time and memory your jobs spend on the compute nodes.

Contains batch manager job parameters along with Unix and software commands.
Check your Service Unit (SU) Balance

- 1 SU = 1 core-hour (GPUs are more expensive per-hour!)
- List the SU Balance of your Account(s) with:
  - Run `myproject -d <Account#>` to change default project account
    (replace `<Account#>` with your number!)
  - Run `myproject -h` to see more options

https://hprc.tamu.edu/kb/User-Guides/AMS/Service_Unit/
https://hprc.tamu.edu/kb/User-Guides/AMS/UI/
SUs in the Grace Dashboard

The same functionality can be found in the Dashboard:
Hands-On Activity - 2 Minutes

1. Use myproject to check the SU balance of your accounts.

2. Use the dashboard to check the same information.
Sample Job Script Structure

```bash
#!/bin/bash

##NECESSARY JOB SPECIFICATIONS
#SBATCH --export=NONE
#SBATCH --get-user-env=L
#SBATCH --job-name=JobExample1
#SBATCH --time=01:30:00
#SBATCH --ntasks=1
#SBATCH --mem=2G
#SBATCH --output=stdout.%j

##OPTIONAL JOB SPECIFICATIONS
#SBATCH --account=123456
#SBATCH --mail-type=ALL
#SBATCH --mail-user=email_address

# load required module(s)
module purge
module load GCCcore/11.3.0 Python/3.10.4

# Run your program
python my_program.py
```

These parameters describe your job to the Slurm job scheduler. The lines starting with #SBATCH are NOT comments! See the **Knowledge Base** for more info.

Account number to be charged

Whatever commands or scripts you want to run. Here, we set up the modules we need for our environment and run a python program.

(We will practice with job files in a few slides)!
Submit a Job and Check Job Status

Submit job

```
sbatch example01.job
```

Submitted batch job 6853258
(from job_submit) your job is charged as below

- Project Account: 122792016265
- Account Balance: 1687.066160
- Requested SUs: 3

Check status

```
squeue -u netID
```

<table>
<thead>
<tr>
<th>JOBID</th>
<th>NAME</th>
<th>USER</th>
<th>PARTITION</th>
<th>NODES</th>
<th>CPUS</th>
<th>STATE</th>
<th>TIME</th>
<th>TIME_LEFT</th>
<th>START_TIME</th>
<th>REASON</th>
<th>NODELIST</th>
</tr>
</thead>
<tbody>
<tr>
<td>6853258</td>
<td>jobname</td>
<td>someuser</td>
<td>xlong</td>
<td>2</td>
<td>96</td>
<td>RUNNING</td>
<td>3-07:36:50</td>
<td>16:23:10</td>
<td>2023-01-23T17:27:3</td>
<td>None</td>
<td>c[180,202]</td>
</tr>
<tr>
<td>6853257</td>
<td>jobname</td>
<td>someuser</td>
<td>xlong</td>
<td>2</td>
<td>96</td>
<td>RUNNING</td>
<td>3-07:36:56</td>
<td>16:23:04</td>
<td>2023-01-23T17:27:2</td>
<td>None</td>
<td>c[523-524]</td>
</tr>
</tbody>
</table>
Hands-On Activity

1. Navigate to `/scratch/training/Intro-to-Grace`
2. Copy `hello_world.job` to your home directory
3. Return to your home directory and submit the job file using `sbatch`.
4. Check that the job is running in a Slurm queue with `squeue`.
5. When your job completes, check the contents of the output file.
Hands-On Activity

Linux command line tools

```
cd /scratch/training/Intro-to-Grace

cp example01.job $HOME

cd ~

vi example01.job
```

Portal file navigator

Then submit and check on the command line with `sbatch` and `squeue`.
Batch Queues

- Job submissions are auto-assigned to batch queues based on the resources requested (e.g. number of cores/nodes and walltime limit)
- Use `sinfo` to check their status:

```
[kromero2001@grace5 ~]$ sinfo
PARTITION     AVAIL TIMELIMIT JOB_SIZE NODES(A/I/O/T) CPUS(A/I/O/T)
short*        up   2:00:00    1-32   431/259/110/800 18789/14283/5328/384
medium        up   1:00:00:00 1-128   431/259/110/800 18789/14283/5328/384
long          up   7:00:00:00 1-64   431/259/110/800 18789/14283/5328/384
xlong         up   21:00:00:00 1-32   431/259/110/800 18789/14283/5328/384
vnc           up   12:00:00    1-32   98/3/16/117    849/3999/768/5616
gpu           up   4:00:00:00    1-32   98/3/16/117    849/3999/768/5616
bigmem        up   2:00:00:00    1-4    1/6/1/8        18/542/80/640
staff         up   infinite    1-infinite 529/262/126/917 19638/18282/6096/440
special       up   7:00:00:00    1-infinite 529/262/126/917 19638/18282/6096/440
gpu-a40       up   10:00:00:00    1-15    3/12/0/15     12/708/0/720
```

For the NODES and CPUS columns:
- **A** = Active (in use by running jobs)
- **I** = Idle (available for jobs)
- **O** = Offline (unavailable for jobs)
- **T** = Total

[https://hprc.tamu.edu/kb/User-Guides/Grace/Batch/#batch-queues](https://hprc.tamu.edu/kb/User-Guides/Grace/Batch/#batch-queues)
## Job Submission and Tracking

<table>
<thead>
<tr>
<th>Slurm queue command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>sbatch</code> <code>jobfile1</code></td>
<td>Submit <code>jobfile1</code> to batch system</td>
</tr>
<tr>
<td><code>squeue</code> [-u <code>user_name</code>] [-j <code>job_id</code>]</td>
<td>List jobs</td>
</tr>
<tr>
<td><code>scancel</code> <code>job_id</code></td>
<td>Kill a job</td>
</tr>
<tr>
<td><code>sacct -X -j </code> <code>job_id</code></td>
<td>Show information for a job (can be when job is running or recently finished)</td>
</tr>
<tr>
<td><code>sacct -X -S </code> <code>YYYY-HH-MM</code></td>
<td>Show information for all of your jobs since <code>YYYY-HH-MM</code></td>
</tr>
<tr>
<td><code>lmu </code> <code>job_id</code></td>
<td>Show resource usage for a job</td>
</tr>
<tr>
<td><code>pestat -u $USER</code></td>
<td>Show resource usage for a running job</td>
</tr>
<tr>
<td><code>seff</code> <code>job_id</code></td>
<td>Check CPU/memory efficiency for a job</td>
</tr>
</tbody>
</table>

[https://hprc.tamu.edu/kb/Helpful-Pages/Batch-Translation/](https://hprc.tamu.edu/kb/Helpful-Pages/Batch-Translation/)
Need Help?

First check the **FAQ**
- [Grace User Guide](#)
- Email your questions to help@hprc.tamu.edu

Help us help you -- when you contact us, tell us:
- Which Cluster you’re using
- Your username
- Job id(s) if any
- Location of your jobfile, input/output files
- Application used if any
- Module(s) loaded if any
- Error messages
- Steps you have taken, so we can reproduce the problem
Continued Learning

Intro to HPRC Video Tutorial Series

HPRC’s Knowledge Base
Thank you.

Any questions?