



Ada

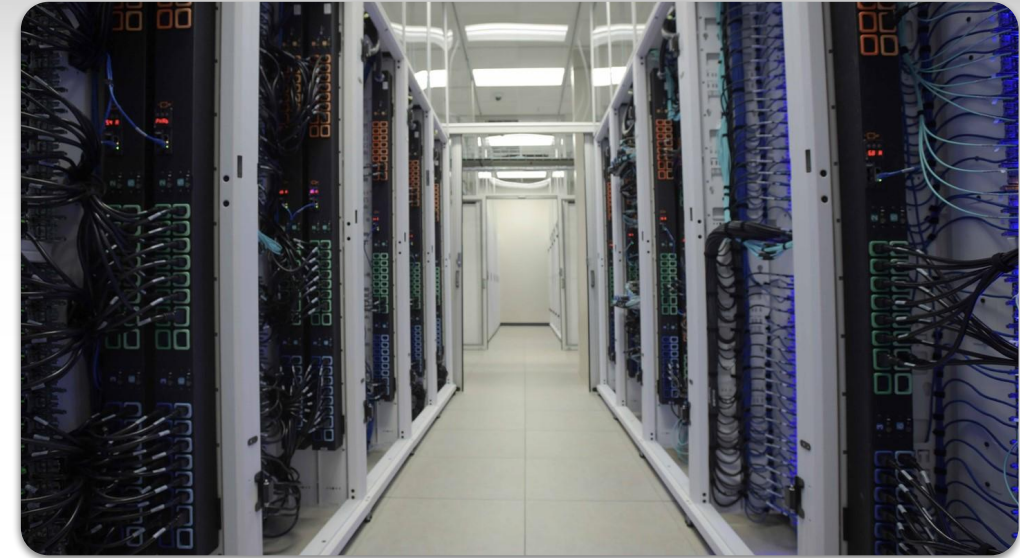
Using the LSF Job Scheduler on Ada

HPRC's Newest Cluster

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

Grace Status: Testing and Early user onboarding

Grace
3TB Large Memory-80 cores/nodes
Other Login Nodes-48 cores/node

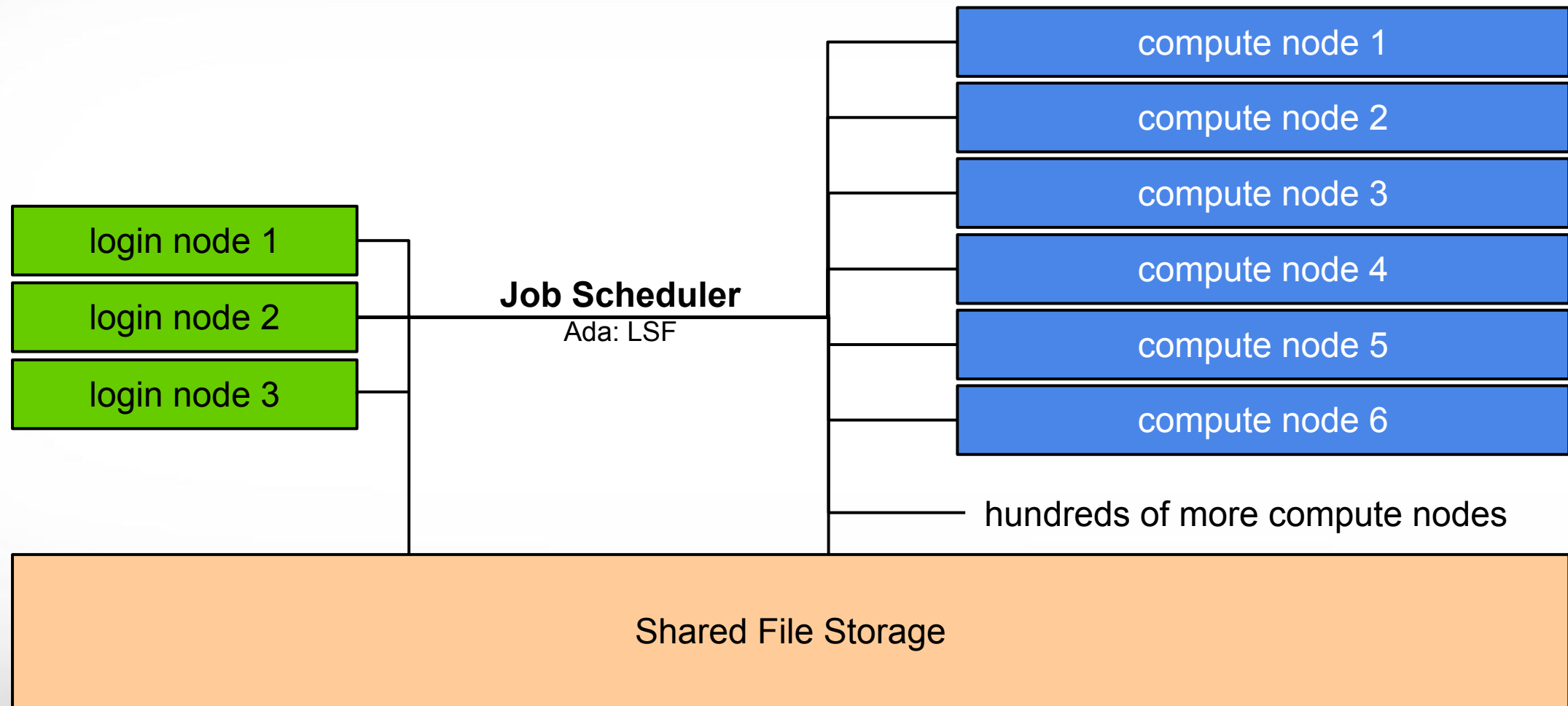


Login Nodes	5
384GB memory general compute nodes	800
GPU - A100 nodes with 384GB memory	100
GPU - RTX 6000 nodes with 384GB memory	9
GPU - T4 nodes with 384GB memory	8
3TB Large Memory	8

Available late Spring 2021

For more information:
<https://hprc.tamu.edu/wiki/Grace:Intro>

HPC Diagram

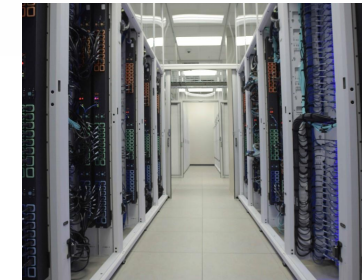
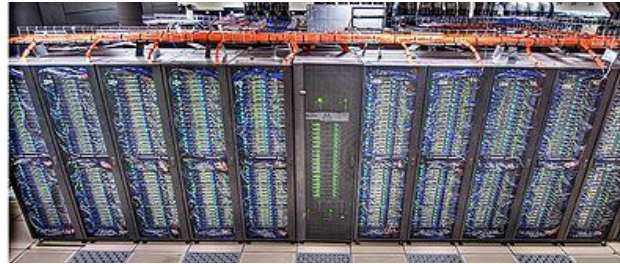


File Systems and User Directories

Directory	Environment Variable	Space Limit	File Limit	Intended Use
/home/\$USER	\$HOME	10 GB	10,000	Small to modest amounts of processing.
/scratch/user/\$USER	\$SCRATCH	1 TB	50,000	Temporary storage of large files for on-going computations. Not intended to be a long-term storage area.
/tiered/user/\$USER	\$ARCHIVE	10 TB	50,000	Intended to hold valuable data files that are not frequently used (on Ada/Curie only)

- `$HOME` and `$SCRATCH` directories are not shared between Ada and Terra clusters.
- View usage and quota limits using the command: `showquota`

Pop Quiz



Which one of the following is not an HPRC cluster?

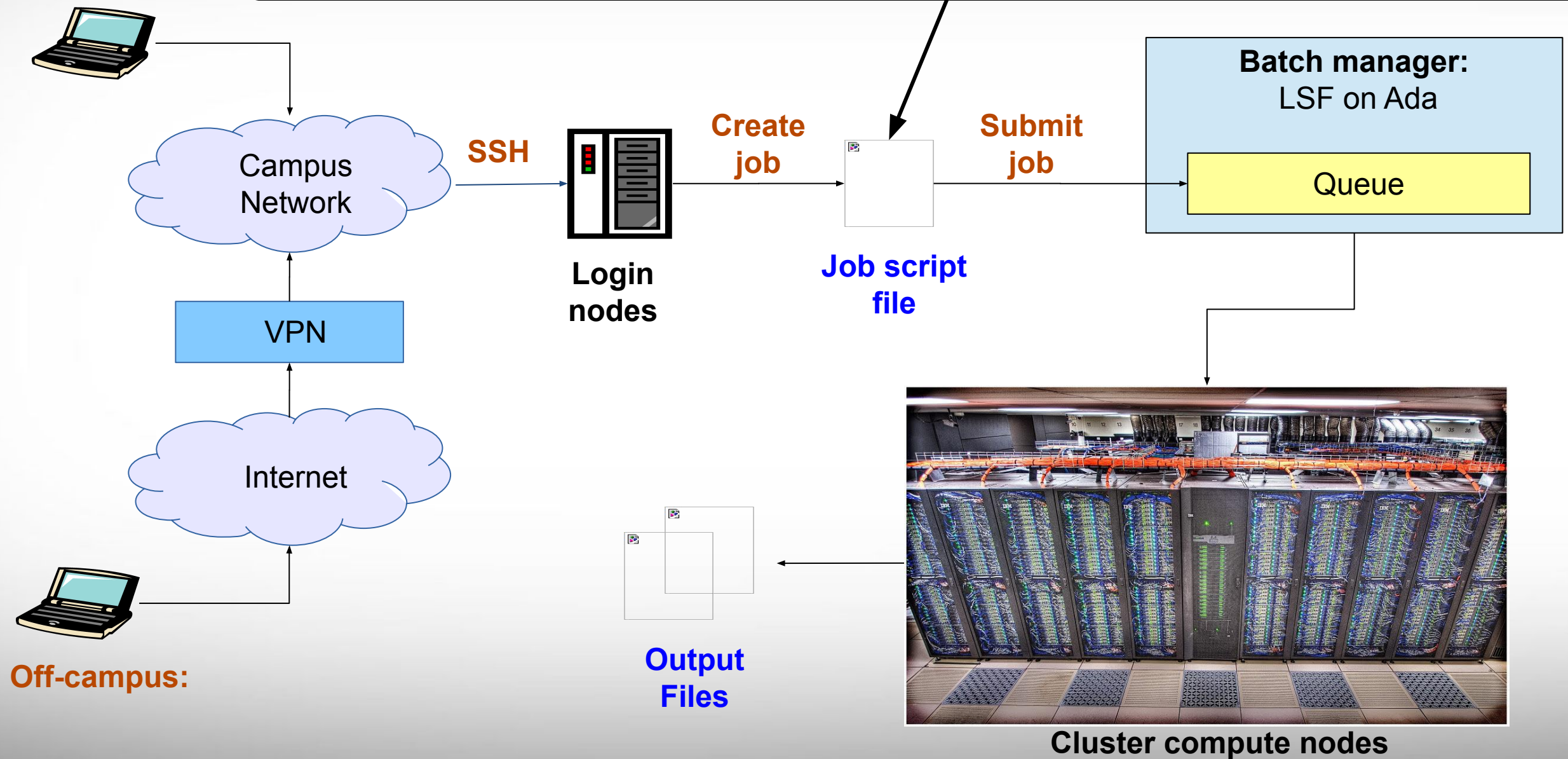
- A. Ada
- B. Bozo

- C. Grace
- D. Terra

Batch Computing on HPRC Clusters

On-campus:

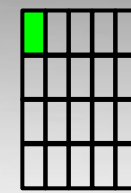
A batch job script is a text file that contains Unix and software commands and Batch manager job parameters



Off-campus:

Cluster compute nodes

Ada Job File (Serial Example)



##NECESSARY JOB SPECIFICATIONS

```
#BSUB -L /bin/bash           # Uses bash to initialize the job's execution environment.
#BSUB -J ExampleJob1        # Set the job name to "ExampleJob1"
#BSUB -W 2:00                # Set the wall clock limit to 2hr
#BSUB -n 1                    # Request 1 core
#BSUB -R "span[ptile=1]"     # Request 1 core per node.
#BSUB -R "rusage[mem=2500]"  # Request 2500MB per process (CPU) for the job
#BSUB -M 2500                # Set the per process enforceable memory limit to 2500MB.
#BSUB -o stdout.%J          # Send stdout to "stdout.[jobID]"
#BSUB -e stderr.%J          # Send stderr to "stderr.[jobID]"
```

SUs = 2

##OPTIONAL JOB SPECIFICATIONS

```
#BSUB -P 123456              # Set billing account to 123456
#BSUB -u email_address       # Send all emails to email_address
#BSUB -B -N                  # Send email on job begin (-B) and end (-N)
```

```
# load required module(s)
```

```
module load Python/3.5.2-intel-2017A
```

```
# run your program
```

```
./my_program.py
```

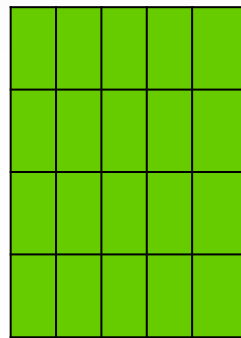
Important Batch Job Parameters

Ada	Comment
<code>#BSUB -L /bin/bash</code>	Initialize job environment.
<code>#BSUB -W HH:MM</code> or <code>#BSUB -W MM</code>	Specifies the time limit for the job. Must specify seconds SS on Terra
<code>#BSUB -n NNN</code>	Total number of tasks (cores) for the job.
<code>#BSUB -R "span[ptile=XX]"</code>	Specifies the maximum number of tasks (cores) to allocate per node
<code>#BSUB -R "rusage[mem=nnnn]"</code> <code>#BSUB -M nnnn</code> (memory per CORE)	Sets the maximum amount of memory (MB). G for GB is supported on Terra

hprc.tamu.edu/wiki/HPRC:Batch_Translation

Mapping Jobs to Cores per Node on **Ada**

A.

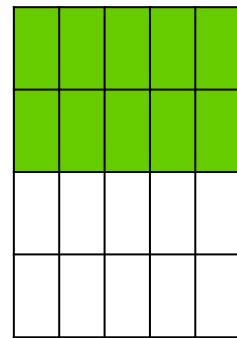


20 cores on
1 compute node

#BSUB -n 20
#BSUB -R "span[ptile=20]"

Preferred Mapping
(if applicable)

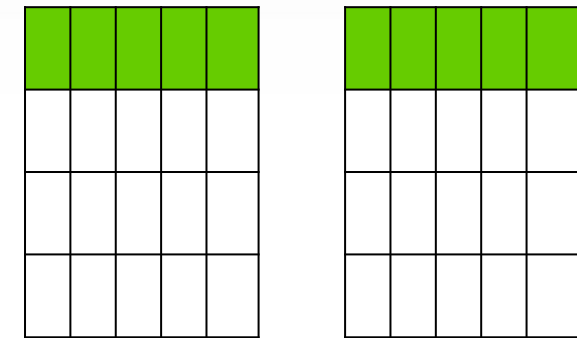
B.



20 cores on
2 compute nodes

#BSUB -n 20
#BSUB -R "span[ptile=10]"

C.



20 cores on
4 compute nodes

#BSUB -n 20
#BSUB -R "span[ptile=5]"

Job Memory Requests

- Must specify both parameters for requesting memory:
 - **#BSUB -R "rusage[mem=process_alloc_size]"**
 - **#BSUB -M process_size_limit**
- Default value of 2500 MB (2.5 GB) per job slot if -R/-M not specified.
- On 64GB nodes, usable memory is < **54 GB** (10 GB used by system).
 - Per-process memory limit < **2700 MB** for a 20-core job.
- For more memory, request a large memory nodes:
 - If under 256 GB (up to 20 cores per node):
 - **#BSUB -R "select[mem256gb]"**
 - For 1 or 2 TB of memory (up to 40 cores):
 - use the **-q xlarge** option with either **-R "select[mem1tb]"** or **-R "select[mem2tb]"**
 - The mem1tb and mem2tb nodes are accessible only via the *xlarge* queue.

Consumable Computing Resources

- Resources specified in a job file:
 - Processor cores
 - Memory
 - Wall time
 - GPU
- Service Unit (SU) - Billing Account
 - Use "myproject" to query
hprc.tamu.edu/wiki/HPRC:AMS:Service_Unit

```
myproject
```

```
=====
List of YourNetID's Project Accounts
=====
```

Account	FY	Default	Allocation	Used & Pending SUs	Balance	PI
1228000223136	2019	N	10000.00	0.00	10000.00	Doe, John
1428000243716	2019	Y	5000.00	-71.06	4928.94	Doe, Jane

- Other resources:
 - Software license/token
 - Use "license_status" to query
 - hprc.tamu.edu/wiki/SW:License_Checker

Find available license for "ansys":

```
license_status -s ansys
```

```
License status for ANSYS:
```

```
-----
| License Name           | # Issued | # In Use | # Available |
|-----|-----|-----|-----|
| aa_mcad                | 50       | 0        | 50         |
| aa_r                   | 50       | 32       | 18         |
| aim_mp1                | 50       | 0        | 50         |
| .....                 |          |          |           |
|-----|-----|-----|-----|
```

Find detail options:

```
license_status -h
```

Batch Queues

- Job submissions are auto-assigned to batch queues based on the resources requested (number of cores/nodes and walltime limit)
- Some jobs can be directly submitted to a queue:
 - On **Ada**, if the 1TB or 2TB nodes are needed, use the xlarge queue
#BSUB -q xlarge

hprc.tamu.edu/wiki/Terra:Batch#Queues
hprc.tamu.edu/wiki/Ada:Batch_Queuees

Queue Limits on **Ada**

Queue	Min/Default/Max Cores	Default/Max Walltime	Compute Node Types	Pre-Queue Limits	Aggregate Limits Across Queues	Per-User Limits Across Queues	Notes
sn_short	1 / 1 / 20	10 min / 1 hr	64 GB nodes (811) 256 GB nodes (26)		Maximum of 7000 cores for all running jobs in the single-node (sn_*) queues.	Maximum of 1000 cores and 100 jobs per user for all running jobs in the single node (sn_*) queues.	For jobs needing only one compute node .
sn_regular		1 hr / 1 day					
sn_long		24 hr / 4 days					
sn_xlong		4 days / 30 days					
mn_short	2 / 2 / 200	10 min / 1 hr		Maximum of 2000 cores for all running jobs in this queue.	Maximum of 12000 cores for all running jobs in the multi-node (mn_*) queues.	Maximum of 3000 cores and 150 jobs per user for all running jobs in the multi-node (mn_*) queues.	For jobs needing more than one compute node .
mn_small	2 / 2 / 120	1 hr / 10 days		Maximum of 7000 cores for all running jobs in this queue.			
mn_medium	121 / 121 / 600	1 hr / 7 days		Maximum of 6000 cores for all running jobs in this queue.			
mn_large	600 / 601 / 2000	1 hr / 5 days		Maximum of 8000 cores for all running jobs in this queue.			
xlarge	1 / 1 / 280	1 hr / 10 days		1 TB nodes (11) 2 TB nodes (4)			
vnc	1 / 1 / 20	1 hr / 6 hr	GPU nodes (30)				For remote visualization jobs.
special	None	1 hr / 7 days	64 GB nodes (811) 256 GB nodes (26)				Requires permission to access this queue.
v100	1 / 1 / 96	1 hr / 2 days	192 GB nodes dual 32 GB v100 (4)				

Run "*blimits -w*" to show how policies are applied to users and queues.

hprc.tamu.edu/wiki/Ada:Batch_Queues

bqueues : Current Queues on **Ada**

```
[ user_netid@ada1 ~]$ bqueues
```

QUEUE_NAME	PRIO	STATUS	MAX	JL/U	JL/P	JL/H	NJOBS	PEND	RUN	SUSP
staff	450	Open:Active	-	-	-	-	0	0	0	0
special	400	Open:Active	-	-	-	-	2080	0	2080	0
xlarge	100	Open:Active	-	-	-	-	80	0	80	0
vnc	90	Open:Active	-	-	-	-	2	0	2	0
sn_short	80	Open:Active	-	-	-	-	0	0	0	0
mn_short	80	Open:Active	2000	-	-	-	0	0	0	0
mn_large	80	Open:Active	8000	-	-	-	2500	720	1780	0
v100	80	Open:Active	-	-	-	-	0	0	0	0
general	50	Closed:Inact	0	-	-	-	0	0	0	0
sn_regular	50	Open:Active	-	-	-	-	846	31	815	0
sn_long	50	Open:Active	-	-	-	-	897	0	897	0
sn_xlong	50	Open:Active	-	-	-	-	2466	60	2406	0
mn_small	50	Open:Active	7000	-	-	-	3550	30	3520	0
mn_medium	50	Open:Active	6000	-	-	-	3784	0	3784	0
curie_devel	40	Open:Active	-	-	-	-	0	0	0	0
curie_medium	35	Open:Active	-	-	-	-	0	0	0	0
curie_long	30	Open:Active	-	-	-	-	765	301	464	0
curie_general	25	Closed:Inact	0	-	-	-	0	0	0	0
low_priority	1	Open:Active	2500	500	-	-	0	0	0	0

hprc.tamu.edu/wiki/Ada:Batch_Queues

Submitting Your Job and Check Job Status

Submit job

```
bsub < example01.job
```

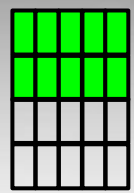
```
Verifying job submission parameters...
Verifying project account...
  Account to charge: 082792010838
  Balance (SUs): 4871.5983
  SUs to charge: 0.0333
Job <2470599> is submitted to default queue <sn_short>.
```

Check status

```
bjobs
```

JOBID	STAT	USER	QUEUE	JOB_NAME	NEXEC_HOST	SLOTS	RUN_TIME	TIME_LEFT
2470599	RUN	tmarkhuang	sn_short	sample01	1	1	0 second(s)	0:5 L

Ada Job File (multi core, single node)



##NECESSARY JOB SPECIFICATIONS

```
#BSUB -L /bin/bash           # Use bashto initialize the job's execution environment.
#BSUB -J ExampleJob2        # Set the job name to "ExampleJob2"
#BSUB -W 6:30                # Set the wall clock limit to 6hr and 30min
#BSUB -n 10                  # Request 10 cores total for the job
#BSUB -R "span[ptile=10] "   # Request 10 cores per node.
#BSUB -R "rusage[mem=2500] " # Request 2500MB per process (CPU) for the job
#BSUB -M 2500                # Set the per process enforceable memory limit to 2500MB.
#BSUB -o stdout.%J          # Send stdout to "stdout.[jobID]"
#BSUB -e stderr.%J          # Send stderr to "stderr.[jobID]"
```

SUs = 65

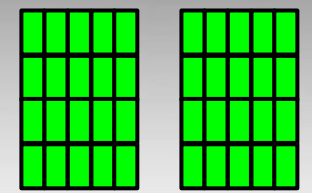
##OPTIONAL JOB SPECIFICATIONS

```
#BSUB -P 123456             # Set billing account to 123456 #find your account with "myproject"
#BSUB -u email_address      # Send all emails to email_address
#BSUB -B -N                  # Send email on job begin (-B) and end (-N)
```

```
# load required module(s)
module load intel/2017A
```

```
# run your program
./my_multicore_program
```


Ada Job File (multi core, multi node)



##NECESSARY JOB SPECIFICATIONS

```
#BSUB -J ExampleJob3           # Set the job name to "ExampleJob3"
#BSUB -L /bin/bash             # Use bash to initialize the job's execution environment.
#BSUB -W 24:00                 # Set the wall clock limit to 24hr
#BSUB -n 40                    # Request 40 cores total for the job
#BSUB -R "span[ptile=20] "     # Request 20 cores per node.
#BSUB -R "rusage[mem=2500] "   # Request 2500MB per process (CPU) for the job
#BSUB -M 2500                  # Set the per process enforceable memory limit to 2500MB.
#BSUB -o stdout.%J            # Send stdout to "stdout.[jobID]"
#BSUB -e stderr.%J           # Send stderr to "stderr.[jobID]"
```

SUs = 960

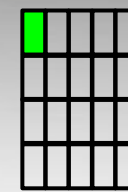
##OPTIONAL JOB SPECIFICATIONS

```
#BSUB -P 123456                # Set billing account to 123456 #find your account with " myproject"
#BSUB -u email_address         # Send all emails to email_address
#BSUB -B -N                    # Send email on job begin (-B) and end (-N)
```

```
# load required module(s)
module load intel/2017A
```

```
# run your program
./my_multicore_multinode_program
```

Ada Job File (serial GPU)



```
##NECESSARY JOB SPECIFICATIONS
#BSUB -J ExampleJob4           # Set the job name to "ExampleJob4"
#BSUB -L /bin/bash             # Use bash login shell to initialize the job's execution environment.
#BSUB -W 2:00                  # Set the wall clock limit to 2hr
#BSUB -n 1                     # Request 1 core total for the job
#BSUB -R "span[ptile=1]"      # Request 1 core per node.
#BSUB -R "rusage[mem=2500]"   # Request 2500MB per process (CPU) for the job
#BSUB -M 2500                  # Set the per process enforceable memory limit to 2500MB.
#BSUB -o stdout.%J            # Send stdout to "stdout.[jobID]"
#BSUB -e stderr.%J           # Send stderr to "stderr.[jobID]"

#BSUB -R "select[gpu]"        # Request a node with a GPU
##OPTIONAL JOB SPECIFICATIONS
#BSUB -P 123456                # Set billing account to 123456 #find your account with "myproject"
#BSUB -u email_address        # Send all emails to email_address
#BSUB -B -N                    # Send email on job begin (-B) and end (-N)

# load required module(s)
module load CUDA/9.2.148.1

# run your program
./my_gpu_program
```

SUs = 1

Other Type of Jobs

- MPI and OpenMP
- Visualization:
 - portal.hprc.tamu.edu (visualization jobs can be run on both Ada and Terra)
- Large number of concurrent single core jobs
 - Check out ***tamulauncher***
 - hprc.tamu.edu/wiki/SW:tamulauncher
 - Useful for running many single core commands concurrently across multiple nodes within a job
 - Can be used with serial or multi-threaded programs
 - Distributes a set of commands from an input file to run on the cores assigned to a job
 - Can only be used in batch jobs
 - If a tamulauncher job gets killed, you can resubmit the same job to complete the unfinished commands in the input file

Job Submission and Tracking

Ada	Description
<code><i>bsub</i> < jobfile1</code>	Submit jobfile1 to batch system
<code><i>bjobs</i> [-u all or user_name] [[-1] job_id]</code>	List jobs
<code><i>bkill</i> job_id</code>	Kill a job
<code><i>bhist</i> [-1] job_id</code>	Show information for a job (can be when job is running or recently finished)
-	Show information for all of your jobs since YYYY-HH-MM
<code><i>lnu</i> [-1] -j job_id</code>	Show resource usage for a job

hprc.tamu.edu/wiki/HPRC:Batch_Translation

List Node Utilization on **Ada**: *lnu*

lnu [-h] [-l] -j **jobid** # lists the node utilization across all nodes for a running job.
to see more options use: **lnu -h**

Example with a multi-node job (4 nodes):

```
lnu -l -j 795375
```

```
Job          User          Queue          Status Node  Cpus
795375      jomber23      medium          R      4     80
HOST_NAME   status  r15s  r1m  r15m  ut  pg  ls  it  tmp  swp  mem  Assigned Cores
nxt1417      ok      20.0  21.0  21.0  97%  0.0  0  94976  366M  3.7G  41.6G  20
nxt1764 (L)  ok      19.7  20.0  20.0  95%  0.0  0  95040  366M  3.7G  41.5G  20
nxt2111      ok      20.0  20.0  20.0  98%  0.0  0  91712  370M  4.2G  41.5G  20
nxt2112      ok      20.0  21.1  21.0  97%  0.0  0  91712  370M  4.2G  41.6G  20
=====
```

The % utilization (**ut**) in conjunction with Assigned Cores is the most useful. Note that the **tmp**, **swp**, and **mem** refer to available amounts respectively and not usage. See "*man lsload*" for explanations on labels.

hprc.tamu.edu/wiki/Ada:Batch_Processing_LSF#Job_Tracking

Check your Service Unit (SU) Balance

- List the SU Balance of your Account(s)

```
myproject
```

```
=====
List of YourNetID's Project Accounts
-----
| Account | FY | Default | Allocation | Used & Pending SUs | Balance | PI |
-----
| 1228000223136 | 2019 | N | 10000.00 | 0.00 | 10000.00 | Doe, John |
-----
| 1428000243716 | 2019 | Y | 5000.00 | -71.06 | 4928.94 | Doe, Jane |
-----
| 1258000247058 | 2019 | N | 5000.00 | -0.91 | 4999.09 | Doe, Jane |
-----
```

- To specify a project ID to charge in the job file
 - **Ada: #BSUB -P Account#**
- Run "**myproject -d Account#**" to change default project account
- Run "**myproject -h**" to see more options

hprc.tamu.edu/wiki/HPRC:AMS:Service_Unit

hprc.tamu.edu/wiki/HPRC:AMS:UI

Job Environment Variables

Ada:

- **\$LSB_JOBID** = job id
- **\$LS_SUBCWD** = directory where job was submitted from
- **\$SCRATCH** = /scratch/user/NetID
- **\$TMPDIR** = /work/\$LSB_JOBID.tmpdir
 - \$TMPDIR is local to each assigned compute node for the job and is about 750 GB
 - Use of \$TMPDIR is recommended for jobs that use many small temporary files

hprc.tamu.edu/wiki/Ada:Batch_Processing_LSF#Environment_Variables

Jobs using tamubatch

Automatic batch job script that submits jobs for the user without the need of writing a full batch script on the cluster

Access help with: `tamubatch --help`

portal.hprc.tamu.edu



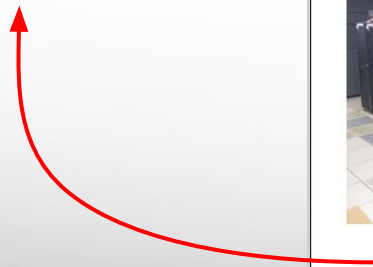
OnDemand provides an integrated, single access point for all of your HPC resources.

Message of the Day

IMPORTANT POLICY INFORMATION

- Unauthorized use of HPRC resources is prohibited and
- Use of HPRC resources in violation of United States export regulations is prohibited for non-US citizens and legal residents.
- Sharing HPRC account and password information is in violation of policy.
- Authorized users must also adhere to ALL policies at: [https://hprc.tamu.edu/wiki/SW:Portal](#)

!! WARNING: There are NO active backups of user data. !!



High Performance Research Computing
A Resource for Research and Discovery

TAMU HPRC OnDemand Homepage

[Ada OnDemand Portal](#)

[Terra OnDemand Portal](#)

[User Guide](#)

The HPRC portal allows users to do the following

- Browse files on the filesystem
- Access the Ada, Terra, Curie Unix command line
- Launch jobs
- Compose job scripts
- Launch interactive GUI apps (SUs charged)

<https://hprc.tamu.edu/wiki/SW:Portal>

Continued Learning

[Intro to HPRC Video Tutorial Series](#)

[HPRC's Wiki Page](#)



**HIGH PERFORMANCE
RESEARCH COMPUTING**
TEXAS A&M UNIVERSITY

Thank you.

Any questions?