



Ada

HPRC Clusters:

Ada Primer

Windows Users:

To follow along, please download Mobaxterm
(a free SSH client)

Google Search “Mobaxterm” or navigate to:
<https://mobaxterm.mobatek.net/download.html>

Download the “Installer edition” (green button)



MobaXterm Home Edition v20.2
(Installer edition)



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

Spring 2020

HPRC Clusters

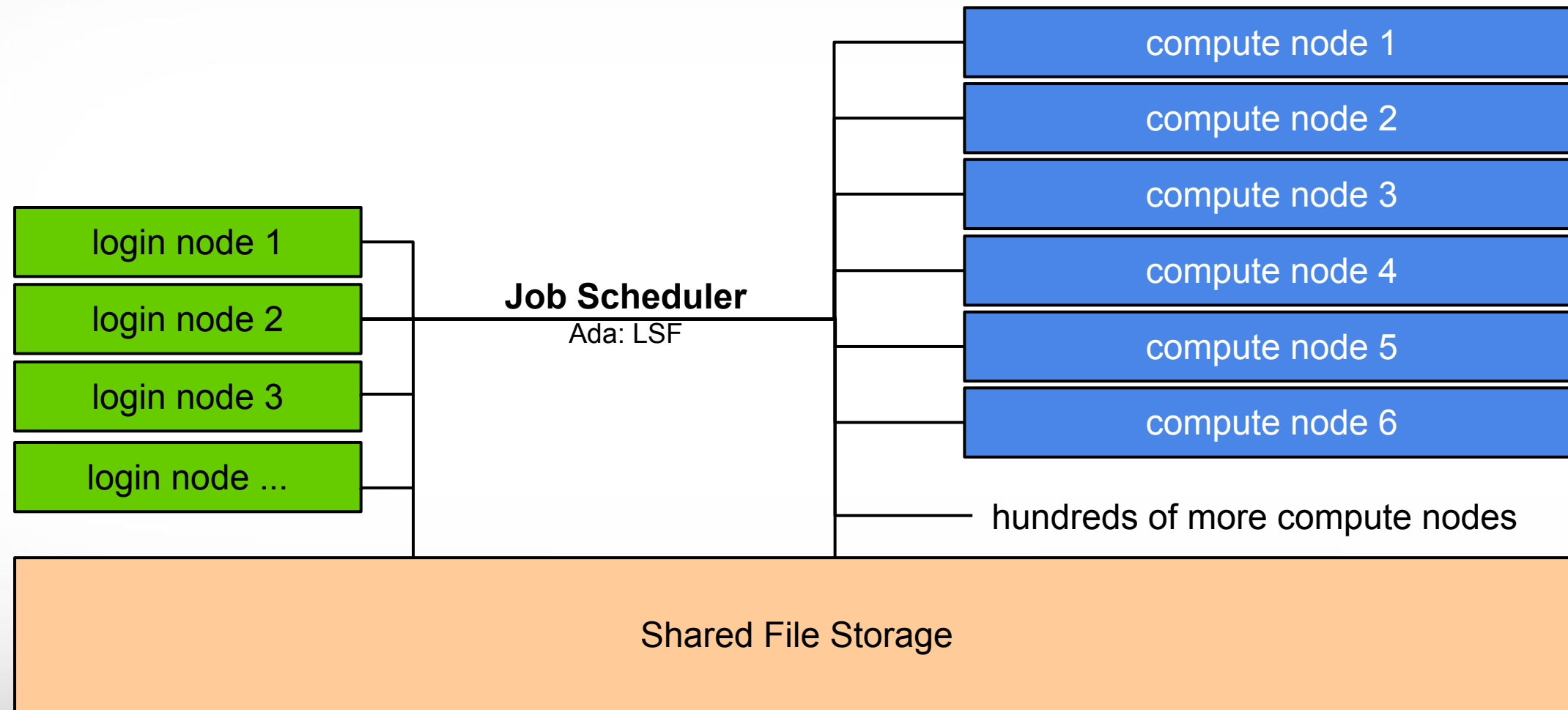
Ada
20 cores/node



Login Nodes	8
64 GB memory compute nodes	792
PHI 64 GB memory compute nodes	9
K20 GPU 64 GB memory compute nodes	10
K20 GPU 256 GB memory compute nodes	20
V100 192 GB memory compute nodes	4 (24 cores/node)
256 GB memory compute nodes	6
1 TB memory compute nodes	11 (40 cores/node)
2 TB memory compute nodes	4 (40 cores/node)

hprc.tamu.edu/resources

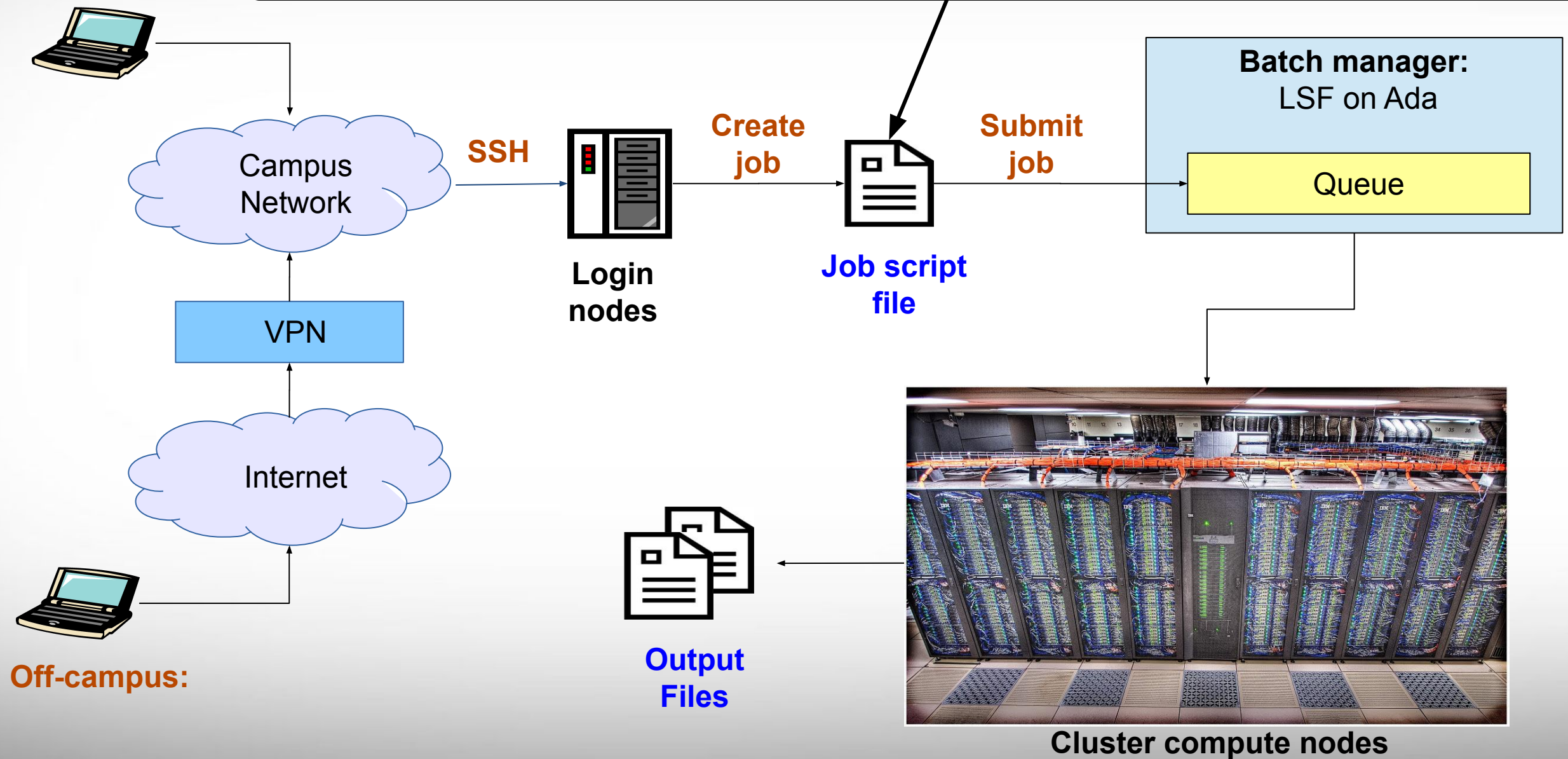
Ada Cluster Diagram



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

Accessing Ada

Access

- On campus: `ssh NetID@ada.tamu.edu`
 - Off campus:
 - Set up and start VPN (Virtual Private Network): u.tamu.edu/VPnetwork
 - Then: `ssh NetID@ada.tamu.edu`
 - *Two-Factor Authentication* enabled
- SSH programs for Windows:
 - MobaXTerm (preferred, includes SSH and X11) or PuTTY SSH
- Access through portal.hprc.tamu.edu (Menu “Clusters” => “Ada Shell Access”)

Usage Policies

- Login sessions idle for **60** minutes, will be closed automatically
- Processes running longer than **60** minutes on login nodes will be killed automatically.
- **Please do not use more than 8 cores on the login nodes!**

hprc.tamu.edu/wiki/HPRC:Access

Two-Factor Authentication

- Duo NetID two-factor authentication to enhance security (it.tamu.edu/duo/)
 - All web login (howdy, portal.hprc.tamu.edu, Globus) through CAS
 - VPN to TAMU campus (since Oct 1st, 2018)
 - SSH/SFTP to HPRC clusters (since Nov 4th, 2019)
- See instructions in two-factor wiki page (hprc.tamu.edu/wiki/Two_Factor)
- SSH clients work with Duo
 - ssh command from Linux, macOS Terminal, Windows cmd
 - MobaXterm for Windows (click on “Session” icon or via local session: hit “enter” 3 times and wait for “Password:” prompt)
- SFTP clients work with Duo
 - scp/sftp command from Linux, macOS Terminal, Windows cmd
 - WinSCP for Windows
 - Cyberduck for macOS
 - FileZilla for Linux/macOS/Windows (but one file per authentication)
- Not all software supports SSH+Duo: SFTP in Matlab

Example: SSH login with Duo

```
$ ssh ada.tamu.edu
```

```
*****
```

```
.... warning message (snipped) .....
```

```
*****
```

Password:

Duo two-factor login for UserNetID

Enter a passcode or select one of the following options:

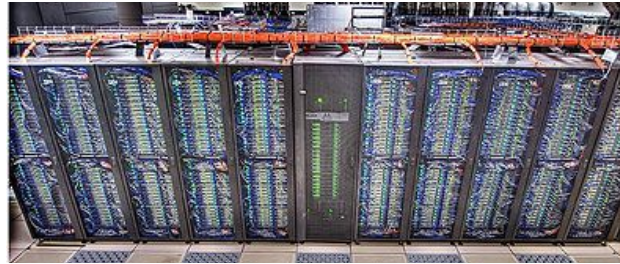
1. Duo Push to XXX-XXX-1234
2. Phone call to XXX-XXX-1234
3. SMS passcodes to XXX-XXX-1234 (next code starts with: 9)

Passcode or option (1-3): 1

Success. Logging you in...

hprc.tamu.edu/wiki/Two_Factor

Pop Quiz



Which one of the following is not an HPRC cluster?

- A. Ada
- B. Bozo

- C. Curie
- D. Terra

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`
- Quota and file limit increases will only be considered for scratch and tiered directories
- Request a group directory for sharing files.
- **Do not share your home, scratch, tiered directories.**

hprc.tamu.edu/wiki/Terra:Filesystems_and_Files
hprc.tamu.edu/wiki/Ada:Filesystems_and_Files

Software

- See the Software wiki page for instructions and examples
 - hprc.tamu.edu/wiki/SW
- Contact license owner to use restricted software
- Contact us for help with software installations
 - Users can install software in their home/scratch dir
 - Do not run the “*sudo*” command when installing software

Application Modules

- Installed applications are available as modules which are available to all users
 - (except for restricted modules)
- **Avoid loading modules in your `~/ .bashrc`**

```
module list
```

```
# list all loaded modules
```

```
module spider matlab
```

```
# search for matlab
```

```
module load Matlab/R2019b
```

```
# load matlab R2019b
```

```
module list
```

```
# list all loaded modules
```

```
module purge
```

```
# remove all loaded modules
```

hprc.tamu.edu/wiki/Ada:Computing_Environment#Modules

Modules and Toolchains

- Load modules with the same toolchains in your job scripts
- The **2018b** and **GCCcore-7.3.0** toolchain versions are recommended
 - `intel/2018b`
 - `iomkl/2018b`
 - `foss/2018b`
 - `GCCcore/7.3.0`
- Avoid loading modules in your `.bashrc` and `.bash_profile` files
- Avoid mixing toolchains if loading multiple modules in the same job script

```
module load HISAT2/2.0.4-foss-2016b  
module load TopHat/2.1.1-intel-2017A-Python-2.7.12  
module load Cufflinks/2.2.1-intel-2015B
```



- Same rule applies to compilers and libraries.

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

```
license_status -a
```

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
```

Computing Environment

- Paths:
 - \$PATH: for commands (eg. /bin:/usr/bin:/usr/local/sbin:/usr/sbin:/home/netid/bin)
 - \$LD_LIBRARY_PATH: for libraries
 - See your \$PATH variable with the command `echo $PATH`
- There is a lot of software, many versions, and many paths to manage
..... How do you manage all these software versions?
- The solution (lmod) which uses the command: `module`
- Each version of a software, application, library, etc. is available as a module.
 - Module names have the format:

software_name / version toolchain [Python-version]
`TopHat/2.1.1-intel-2017A-Python-2.7.12`

hprc.tamu.edu/wiki/Terra:Computing_Environment#Modules
hprc.tamu.edu/wiki/Ada:Computing_Environment#Modules

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

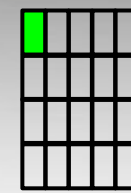
bqueues : Current Queues on **Ada**

```
[ 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

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]"

##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
```

SUs = 2

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 |
-----
```

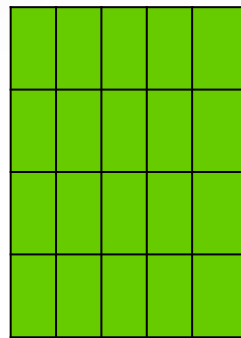
- To specify a project ID to charge in the job file
 - **Ada: #BSUB -P Account#**
- Run "myproject -d Account#" to change default project account
- Run "myproject -h" to see more options

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

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

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

A.

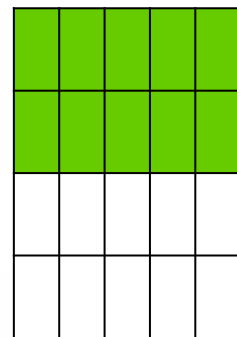
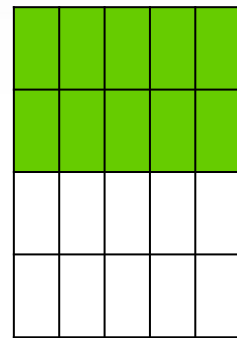


20 cores on
1 compute node

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

Preferred Mapping
(if applicable)

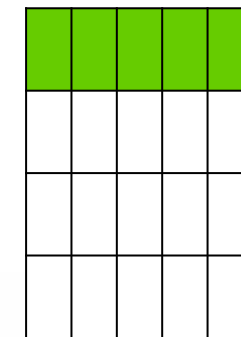
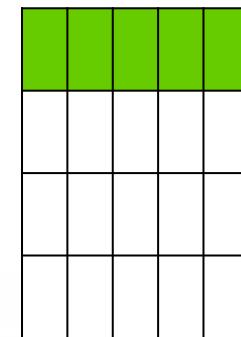
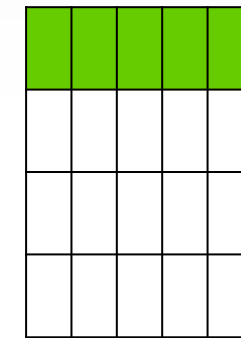
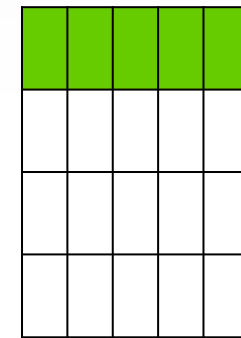
B.



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]"

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> <code>(memory per CORE)</code>	Sets the maximum amount of memory (MB). G for GB is supported on Terra

hprc.tamu.edu/wiki/HPRC:Batch_Translation

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
-	Show resource usage for a running job
-	Check CPU/memory efficiency for a job

hprc.tamu.edu/wiki/HPRC:Batch_Translation

Job submission issue: insufficient SUs

```
$ bsub < myjob
Verifying job submission parameters...
Verifying project account...
  Account to charge: 082792010838
  Balance (SUs):    342.5322
  SUs to charge:   480.0000
-----
|ERROR! Your project account does not have sufficient balance to submit your job! |
-----
Request aborted by esub. Job not submitted.
```

- What to do if you need more SUs
 - Ask your PI to transfer SUs to your account
 - Apply for more SUs (if you are eligible, as a PI or permanent researcher)

hprc.tamu.edu/wiki/HPRC:CommonProblems#Q:_How_do_I_get_more_SUs.3F

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

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

Pop Quiz

```
#BSUB -L /bin/bash
#BSUB -J stacks_S2
#BSUB -n 10
#BSUB -R "span [ptile=10] "
#BSUB -R "rusage [mem=2500] "
#BSUB -M 2500
#BSUB -W 36:00
#BSUB -o stdout.%J
#BSUB -e stderr.%J
```

How much total memory is requested for this job?

- A. 2.5 GB
- B. 2500 MB
- C. 25 GB
- D. 250 GB

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.

Ada: Software for Large Memory Nodes

- The large memory nodes (1TB and 2TB) on Ada have their own separate modules

```
module load Westmere
module avail
```

- Look in the Westmere section:

```
----- /software/easybuild/Westmere/modules/all -----
ABINIT/8.0.8-intel-2016a                               Tcl/8.6.3-foss-2015a
ABYSS/1.9.0-foss-2016a                               Tcl/8.6.3-intel-2015B
ABYSS/1.9.0-intel-2015B-Python-2.7.10                (D) Tcl/8.6.4-foss-2016a
```

- Or type **module spider tool_name/version** to see if Westmere needs to be loaded

```
module spider Canu/1.7-intel-2017A-Perl-5.24.0
```

```
You will need to load all module(s) on any one of the lines below before the
"Canu/1.7-intel-2017A-Perl-5.24.0" module is available to load.
```

```
Westmere/0.devel
Westmere/1
```

- You can just load Westmere along with the other module(s)

```
module load Westmere
module load Canu/1.7-intel-2017A-Perl-5.24.0
```


Sample Job Script for 1TB Node on **Ada**

```
##NECESSARY JOB SPECIFICATIONS
#BSUB -L /bin/bash # Uses bash to initialize the job's execution environment.
#BSUB -J my_job_script # Set the job name to "my_job_script"
#BSUB -W 24:00 # Set the wall clock limit to 24hr
#BSUB -q xlarge # Request xlarge queue
#BSUB -R "select [mem1tb] " # Request 1TB memory node
#BSUB -n 40 # Request 40 core
#BSUB -R "span [ptile=40] " # Request 40 core per node.
#BSUB -R "rusage [mem=24500] " # Request 24500MB (24.5GB) per process (CPU) for the job
#BSUB -M 24500 # Set the per process enforceable memory limit to 24500MB.
#BSUB -o stdout.%J # Send stdout to "stdout.[jobID]"
#BSUB -e stderr.%J # Send stderr to "stderr.[jobID]"
#BSUB -u email_address # (optional) Send all emails to email_address
#BSUB -B -N # (optional) Send email on job begin (-B) and end (-N)

# load required module(s) for use on the large memory nodes
module load Westmere
module load Canu/1.7-intel-2017A-Perl-5.24.0

# run your commands
canu -assemble *fastq
```

Submitting Your Job and Check Job Status

Submit job

```
cd $SCRATCH/batch_examples
```

Ada: `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

Ada: `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
```



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

Thank you.

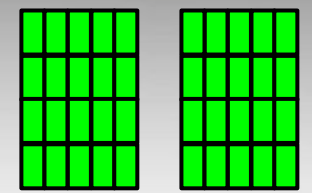
Any question?

Common Job Problems

- Control characters (^M) in job files or data files edited with Windows editor
 - remove the ^M characters with: `dos2unix my_job_file`
- Did not load the required module(s)
- Insufficient walltime specified in #SBATCH --time or #BSUB -W parameter
- Insufficient memory specified in , #SBATCH --mem or --mem-per-cpu or #BSUB -M and -R "rusage [mem=xxx] " parameters
- No matching resource (--mem or -R rusage [mem] too large)
- Running OpenMP jobs across nodes
- Insufficient SU: See your SU balance: `myproject`
- Insufficient disk or file quotas: check quota with `showquota`
- Using GUI-based software without setting up X11 forwarding
 - Enable X11 forwarding at login
 - Or use Portal (portal.hprc.tamu.edu) `ssh -X user@ada.tamu.edu`

FAQ: hprc.tamu.edu/wiki/HPRC:CommonProblems

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

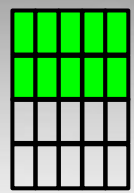
##OPTIONAL JOB SPECIFICATIONS

```
#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 (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
```