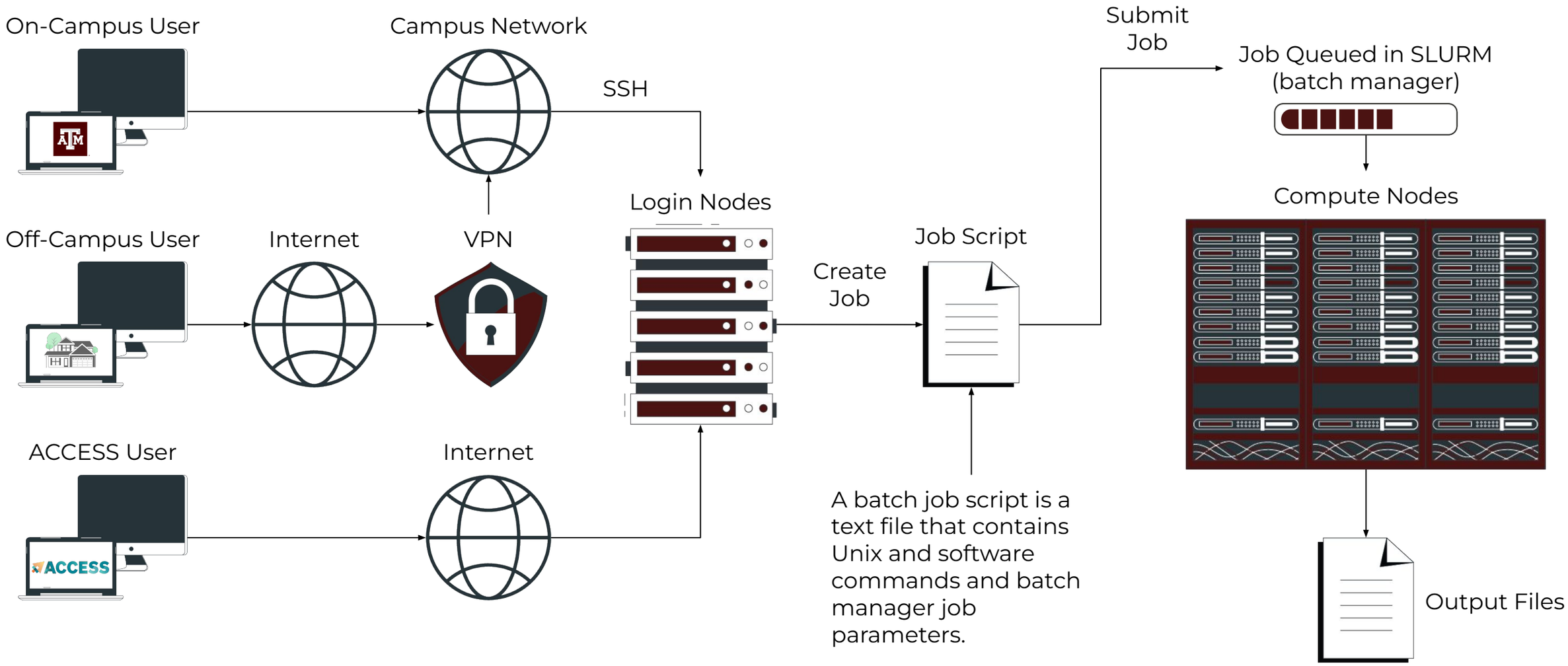
HPRC Primer

Using the SLURM Scheduler

Presented by Ashwin Kundeti
January 31, 2023

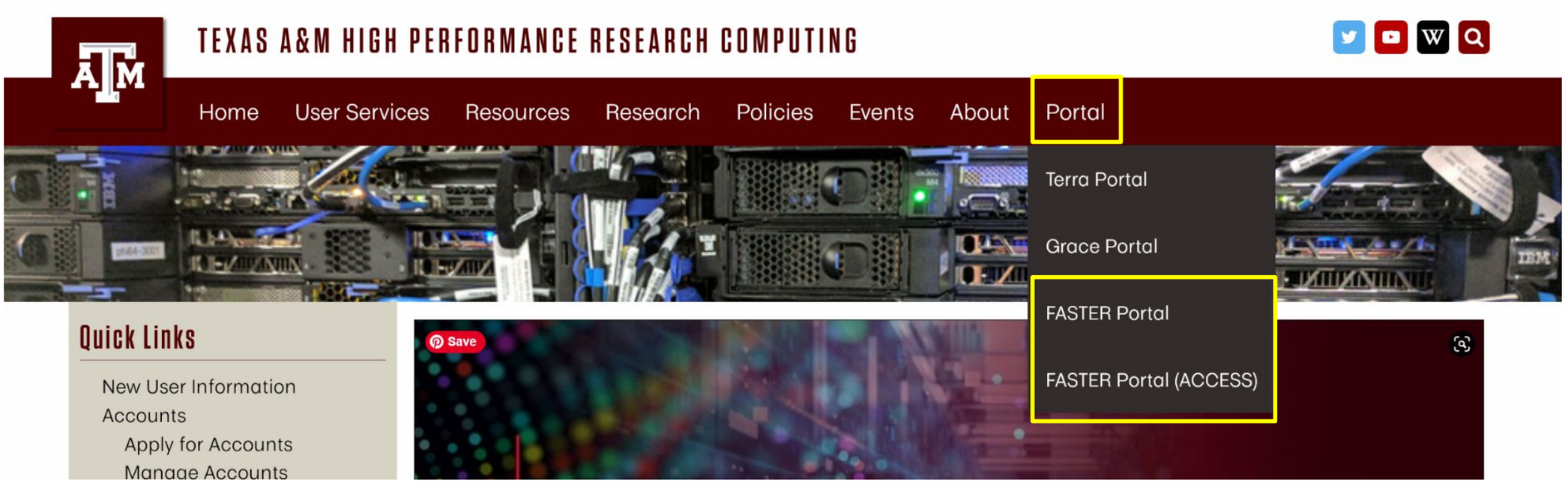
High Performance
Research Computing
DIVISION OF RESEARCH

Batch Computing on HPRC Clusters



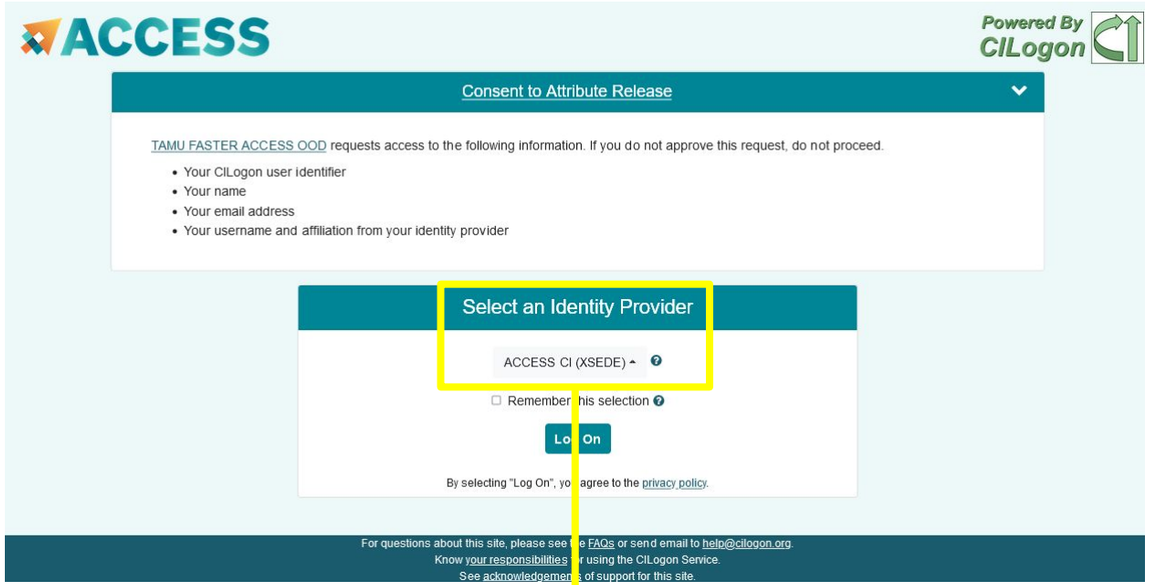
Accessing the HPRC Portal

- HPRC webpage: hprc.tamu.edu, Portal dropdown menu
- Shortcuts:
  - for TAMU NetID: portal-faster.hprc.tamu.edu
  - for ACCESS ID: portal-faster-access.hprc.tamu.edu

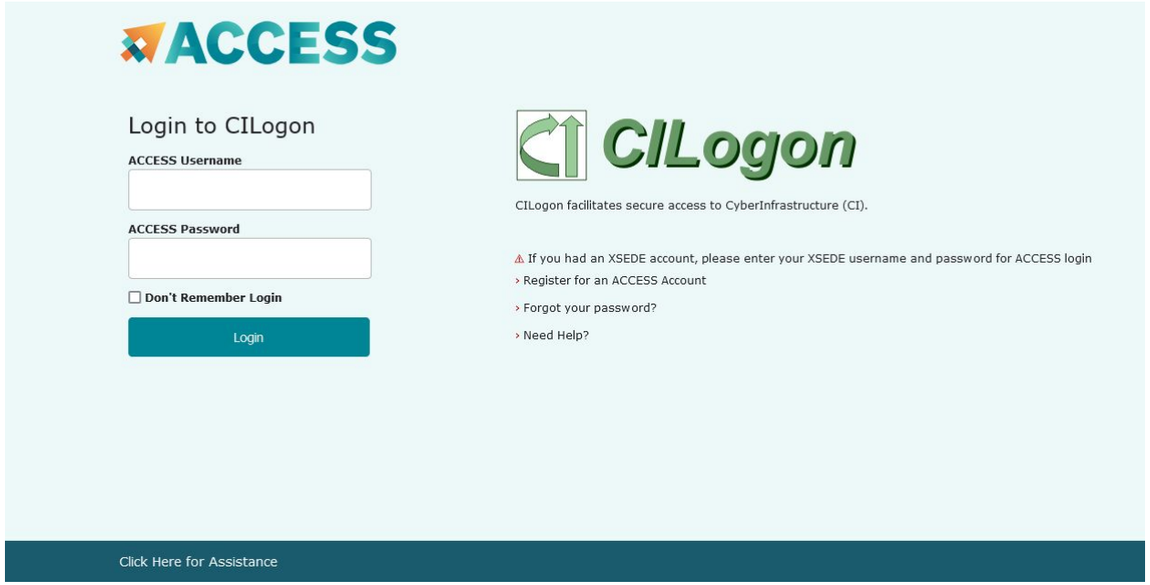


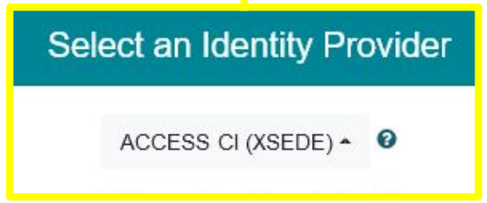
Accessing FASTER via the HPRC Portal (ACCESS)

Navigate to portal-faster-access.hprc.tamu.edu to get to the ACCESS CILogon OpenID Connect page.



Log-in using your ACCESS credentials. Create an account if you do not already have one.

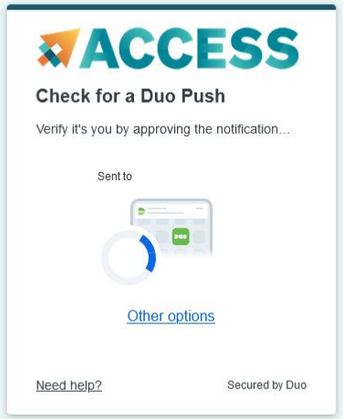




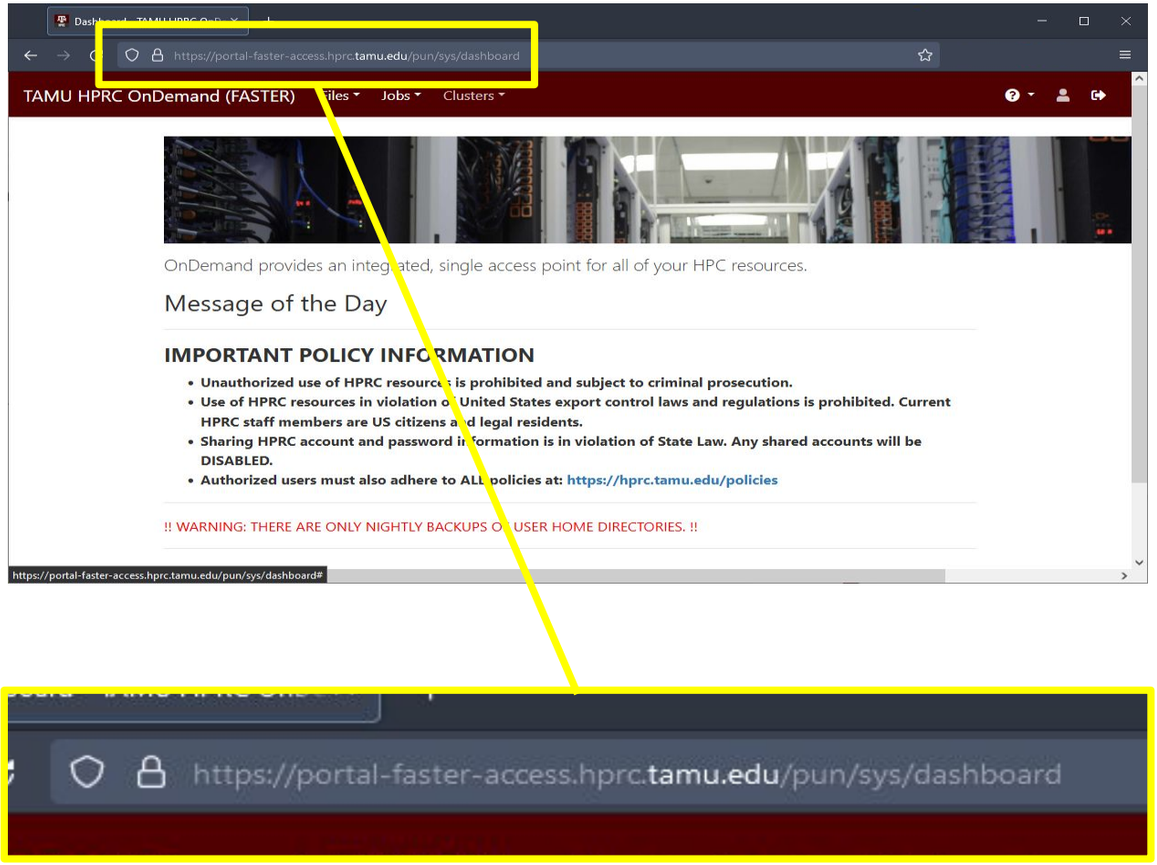
Select the Identity Provider appropriate for your account.

Accessing FASTER via the HPRC Portal (ACCESS)

Complete the Duo Two-Factor Authentication prompt. Set-up the two-factor authentication if you have not already done so.



You will now have access to the FASTER portal.



Hands-On Activity - 2 Minutes

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

2. What message do you see when you log on to the shell?

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	250,000	Temporary storage of large files for on-going computations. Not intended to be a long-term storage area.

`$SCRATCH` is shared between FASTER (TAMU/ACCESS) and Grace (TAMU) 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.

hprc.tamu.edu/wiki/FASTER:Filesystems_and_Files

Sample Job Script Structure

```
#!/bin/bash
#
##NECESSARY JOB SPECIFICATIONS
#SBATCH --job-name=JobExample1
#SBATCH --time=01:30:00
#SBATCH --ntasks=1
#SBATCH --mem=3G
#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
module load Python/3.10.4

python my_program.py
```

These parameters describe your job to the job scheduler.

Account number to be charged.

This is a single line comment and not run as part of the script.

Load the required module(s).

This is a command that is executed by the job.

Important Batch Job Parameters

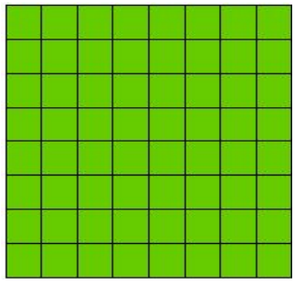
Slurm	Comment
`#SBATCH --time=HH:MM:SS`	Specifies the time limit for the job. Must specify seconds SS on FASTER
`#SBATCH --ntasks=NNN`	Total number of tasks (cores) for the job.
`#SBATCH --ntasks-per-node=XX`	Specifies the maximum number of tasks (cores) to allocate per node

`#SBATCH --mem=xxxxM`
`or`
`#SBATCH --mem=xG`
 	
`(memory per NODE)`
	Sets the maximum amount of memory (MB).
	
	G for GB is supported on FASTER

hprc.tamu.edu/wiki/HPRC:Batch_Translation

Mapping Jobs to Cores per Node on FASTER

A.

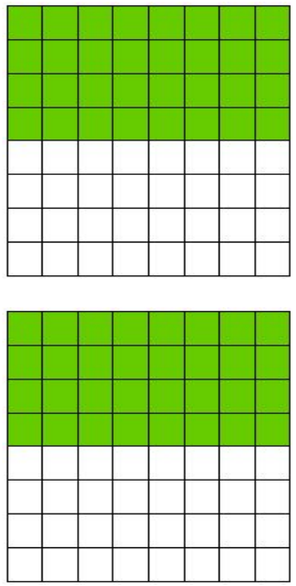


64 cores on
1 compute node

```
#SBATCH --ntasks=64
#SBATCH --ntasks-per-node=64
```

Preferred Mapping
(if applicable)

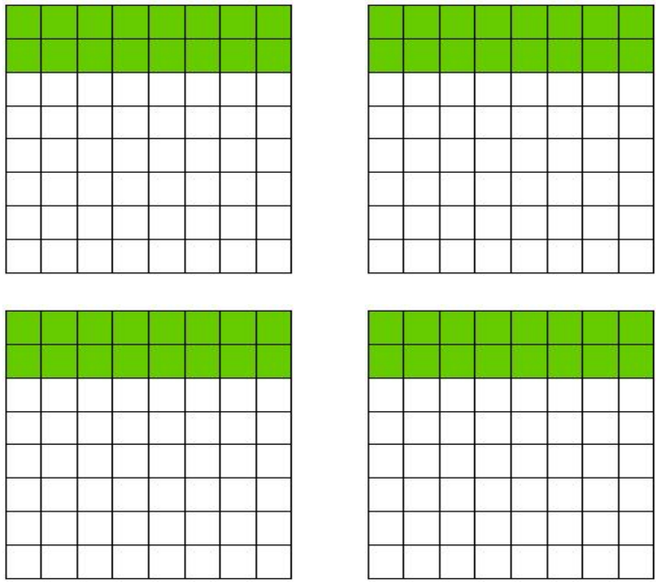
B.



64 cores on
2 compute nodes

```
#SBATCH --ntasks=64
#SBATCH --ntasks-per-node=32
```

C.



64 cores on
4 compute nodes

```
#SBATCH --ntasks=64
#SBATCH --ntasks-per-node=16
```

Job Memory Requests on FASTER

- Specify memory request based on memory per node:

```
#SBATCH --mem=xxxxM          # memory per node in MB
```

or

```
#SBATCH --mem=xG             # memory per node in GB
```

- On 256GB nodes, usable memory is at most 250 GB. The per-process memory limit should not exceed 3900 MB for a 64-core job.
- On the 1TB node, usable memory is at most 990 GB. The per-process memory limit should not exceed 15460 MB for a 64-core job.

Pop Quiz

```
#SBATCH --job-name=stacks_S2
#SBATCH --ntasks=64
#SBATCH --ntasks-per-node=16
#SBATCH --mem=48G
#SBATCH --time=48:00:00
#SBATCH --output=stdout.%J
#SBATCH --error=stderr.%J
```

How many cores is this job requesting?

A. 1024
B. 64
C. 640
D. 4

Consumable Computing Resources

- Resources specified in a job file:
  - Processor cores
  - Memory
  - Wall time
  - GPU
- Service Unit (SU) - Billing Account
  - Use "myproject" to query
    https://hprc.tamu.edu/wiki/HPRC:AMS:Service_Unit
- Other resources:
  - Software license/token
    - Use "license_status" to query
    - License Checker:
      https://hprc.tamu.edu/wiki/SW:License_Checker

Find available license for "matlab":

```
license_status -s matlab
```

```
License status for Matlab:
--------------------------------------------------------------------
|License Name                  |   # Issued|   # In Use|# Available|
--------------------------------------------------------------------
|Matlab                        |         50|          0|         50|
--------------------------------------------------------------------
```

Find detail options:

```
license_status -h
```

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|  2023    |       N|    10000.00|              0.00|    10000.00|Doe, John |
--------------------------------------------------------------------------------
|1428000243716|  2023    |       Y|     5000.00|            -71.06|     4928.94|Doe, Jane |
--------------------------------------------------------------------------------
|1258000247058|  2023    |       N|     5000.00|             -0.91|     4999.09|Doe, Jane |
--------------------------------------------------------------------------------
```

- To specify a project ID to charge in the job file
  - Slurm: `#SBATCH --account Account#`
- Run "myproject -d Account#" to change default project account
- Run "myproject -h" to see more options

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

Examples of SUs charged based on Job Cores, Time and Memory Requested

A Service Unit (SU) on **FASTER** is equivalent to one core or **4** GB memory usage for one hour.

Number of Cores	GB of memory per core	Total Memory (GB)	Hours	SUs charged
1	3.9	3.9	1	1
1	7	7	1	2
1	250	250	1	64
64	3.9	250	1	64

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

Examples of SUs charged based on GPU and Time Requested for FASTER

The SUs for GPUs are added to the usual charge for effective cores/hour.

GPU	Number of GPUs	Hours	SUs charged
T4	1	1	64
A10	1	1	128
A30	1	1	128
A40	1	1	128
A100	1	1	128

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

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:
  - If gpu nodes are needed, use the gpu partition/queue:
    #SBATCH --partition=gpu
  - Jobs that have special resource requirements are scheduled in the special queue (must request access to use this queue)

https://hprc.tamu.edu/wiki/FASTER:Batch#Batch_Queues

sinfo : Current Queues

Enter command in green in Linux terminal

```
sinfo
```

Command output in blue

```
PARTITION       AVAIL  TIMELIMIT    JOB SIZE    NODES(A/I/O/T)    CPUS(A/I/O/T)
cpu*            up     7-00:00:00   1-32        52/33/27/112      3245/2195/1728/7168
gpu             up     7-00:00:00   1-32        11/30/11/52       294/2330/704/3328
```

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

pestat : Processor Status

- **pestat** allows you to check the status of the nodes on FASTER
- Type **pestat -G** to show the status of all nodes
- -p allows you to show a specific partition
- Example:

```
pestat -G -p gpu
```

```
GPU GRES (Generic Resource) is printed after each jobid
Print only nodes in partition gpu
Hostname        Partition     Node Num_CPU  CPUload  Memsize   Freemem  GRES/node
Joblist
                              State Use/Tot  (15min)     (MB)      (MB)
fc001                 gpu       mix    2   64     2.29     256000    50151* gpu:a100:8
fc002                 gpu      idle    0   64     0.00     256000   252853  gpu:a100:4
fc003                 gpu    drain*    0   64     0.00*    256000        0  gpu:t4:4
fc009                 gpu      idle    0   64     0.00     256000   252104  gpu:t4:4
fc010                 gpu      idle    0   64     0.00     256000   253323  gpu:t4:4
fc011                 gpu      idle    0   64     0.00     256000   253496  gpu:t4:4
fc012                 gpu     alloc   64   64     1.22*    256000   108178  gpu:t4:8
fc013                 gpu     drain*   0   64     0.00     256000   252515  gpu:t4:8
```

Submitting Your Job and Check Job Status

Submit job

```
sbatch example01.job
```

```
Submitted batch job 161997
(from job_submit) your job is charged as below
               Project Account: 122792016265
               Account Balance: 1687.066160
               Requested SUs:   3
```

Check status

```
squeue -u username
```

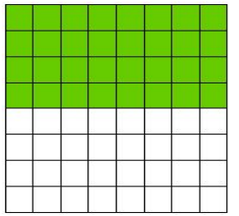
```
JOBID   NAME      USER      PARTITION  NODES  CPUS  STATE    TIME        TIME_LEFT  START_TIME          REASON      NODELIST
94669   somejob   someuser     gpu       6     24   RUNNING  1-17:27:19 2-00:37:41 2023-01-29T15:53:4 None        fc[100-105]
```

Hands-on Activity - 5 min.

```
#!/bin/bash
#
##NECESSARY JOB SPECIFICATIONS
#SBATCH --job-name=JobExample1
#SBATCH --time=01:30:00
#SBATCH --ntasks=1
#SBATCH --mem=3G
#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
module load Python/3.10.4

python --version
```

Slurm Job File (multi core, single node)

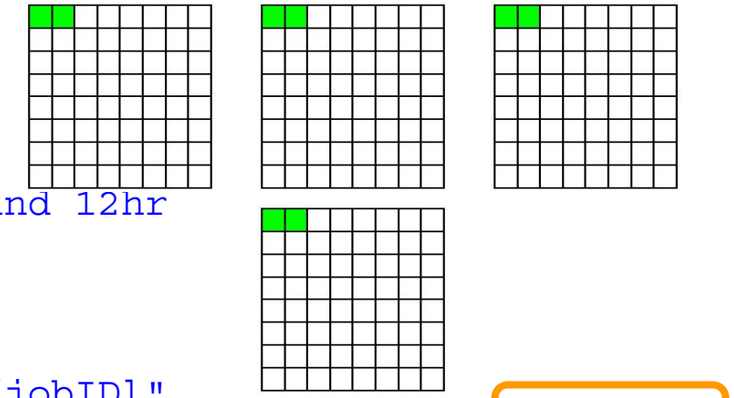
```
#!/bin/bash

##NECESSARY JOB SPECIFICATIONS
#SBATCH --job-name=JobExample2       # Set the job name to "JobExample2"
#SBATCH --time=6:30:00               # Set the wall clock limit to 6hr and 30min
#SBATCH --nodes=1                    # Request 1 node
#SBATCH --ntasks-per-node=32         # Request 32 tasks(cores) per node
#SBATCH --mem=28G                    # Request 28GB per node
#SBATCH --output=stdout.%j           # Send stdout to "stdout.[jobID]"
#SBATCH --error=stderr.%j            # Send stderr to "stderr.[jobID]"
##OPTIONAL JOB SPECIFICATIONS
#SBATCH --account=123456             # Set billing account to 123456
#SBATCH --mail-type=ALL              # Send email on all job events
#SBATCH --mail-user=email_address    # Send all emails to email_address

# purge module
module purge
# load required module(s)
module load GCC/12.1.0

# run your program
./my_multicore_program
```



SUs = 208

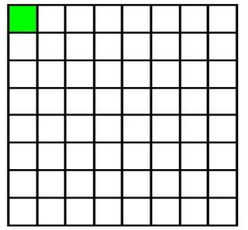
Slurm Job File (multi core, multi node)

```
#!/bin/bash

##NECESSARY JOB SPECIFICATIONS
#SBATCH --job-name=JobExample3        # Set the job name to "JobExample3"
#SBATCH --time=1-12:00:00             # Set the wall clock limit to 1 Day and 12hr
#SBATCH --ntasks=8                    # Request 8 tasks (cores)
#SBATCH --ntasks-per-node=2           # Request 2 tasks(cores) per node
#SBATCH --mem=2.5G                    # Request 2.5 GB  per node
#SBATCH --output=stdout.%j            # Send stdout and stderr to "stdout.[jobID]"

##OPTIONAL JOB SPECIFICATIONS
#SBATCH --account=123456              # Set billing account to 123456 (find your account with "myproject")
#SBATCH --mail-type=ALL               # Send email on all job events
#SBATCH --mail-user=email_address     # Send all emails to email_address

# purge and then load the modules required for your software
module purge
module load GCC/11.3.0 OpenMPI/4.1.4

# run program with MPI
mpirun my_multicore_multinode_program
```



SUs = 96

Slurm Job File (serial GPU)



```
#!/bin/bash

##NECESSARY JOB SPECIFICATIONS
#SBATCH --job-name=JobExample4       # Set the job name to "JobExample4"
#SBATCH --time=01:00:00                 # Set the wall clock limit to 1hr
#SBATCH --ntasks=1                      # Request 1 task (core)
#SBATCH --mem=2G                        # Request 2GB per node
#SBATCH --output=stdout.%j           # Send stdout and stderr to "stdout.[jobID]"
#SBATCH --gres=gpu:1                    # Request 1 GPU
#SBATCH --partition=gpu                 # Request the GPU partition/queue

##OPTIONAL JOB SPECIFICATIONS
#SBATCH --account=123456                # Set billing account to 123456
#SBATCH --mail-type=ALL                 # Send email on all job events
#SBATCH --mail-user=email_address       # Send all emails to email_address

# purge and then load required module(s)
module purge
module load CUDA/11.7.0

# run your program
./my_gpu_program
```

SUs = 65

Slurm Job File (specific GPU)

```
#!/bin/bash

##NECESSARY JOB SPECIFICATIONS
#SBATCH --job-name=T4-GPU-Job          #Set the job name to "T4-GPU-Job"
#SBATCH --time=01:30:00                 #Set the wall clock limit to 1hr and 30min
#SBATCH --ntasks=8                      #Request 8 tasks
#SBATCH --mem=250G                      #Request 250GB (250GB) per node
#SBATCH --output=T4-GPU-Job-out.%j      #Send stdout/err to "Example4Out.[jobID]"
#SBATCH --gres=gpu:t4:8                 #Request 8 "t4" GPU per node
#SBATCH --partition=gpu                 #Request the GPU partition/queue

##OPTIONAL JOB SPECIFICATIONS
##SBATCH --account=123456              #Set billing account to 123456
##SBATCH --mail-type=ALL               #Send email on all job events
##SBATCH --mail-user=email_address     #Send all emails to email_address

# purge and then load required module(s)
module purge
module load CUDA/11.7.0

# run your program
./my_gpu_program
```

SUs = 608

Other Type of Jobs

- MPI and OpenMP
- Visualization:
  - https://portal.hprc.tamu.edu/ (visualization jobs can be run on both FASTER and Grace; (more details in later slide)
- Large number of concurrent single core jobs
  - Check out ***tamulauncher***
    - https://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

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

hprc.tamu.edu/wiki/HPRC:Batch_Translation

Job submission issue: insufficient SUs

```
sbatch myjob
```

```
sbatch: error: (from job_submit) your account's balance is not sufficient to submit your job
               Project Account: 123940134739
               Account Balance: 382.803877
               Requested SUs:   18218.666666667
```

- 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

Job Environment Variables

- **`$SLURM_JOBID`** = job id
- **`$SLURM_SUBMIT_DIR`** = directory where job was submitted from
- **`$SCRATCH`** = /scratch/user/NetID
- **`$TMPDIR`** = /work/job.$SLURM_JOBID

https://hprc.tamu.edu/wiki/FASTER:Batch#Environment_Variables

Continued Learning

HPRC YouTube

HPRC Homepage

FASTER Quick Start Guide

FASTER Portal (NetID users)

FASTER Portal (ACCESS users)

help@hprc.tamu.edu

Need Help?

First check the FAQ

- Grace User Guide
- Email your questions to help@hprc.tamu.edu

Help us, help you -- we need more info

- Which Cluster
- 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

High Performance
Research Computing
DIVISION OF RESEARCH

Thank you.

Any questions?