

HIGH PERFORMANCE RESEARCH COMPUTING

HPRC Primer

Using the Slurm Scheduler on the ACES Cluster

September 03, 2024



High Performance
Research Computing
DIVISION OF RESEARCH

Accessing the ACES Portal

- HPRC webpage: hprc.tamu.edu
- Aces portal shortcut: portal-aces.hprc.tamu.edu
- Requires ACCESS ID!

ATM TEXAS A&M HIGH PERFORMANCE RESEARCH COMPUTING

Home User Services Resources Research Policies Events Training About **Portal**

Quick Links

- New User Information
- Accounts
 - Apply for Accounts
 - Manage Accounts
- User Consulting
- Training
- Knowledge Base

LS-DYNA keyword deck by LS-PrePost
Time = 1.28

- Terra Portal
- Grace Portal
- FASTER Portal
- FASTER Portal (ACCESS)
- ACES Portal (ACCESS)**
- Launch Portal (ACCESS)

Accessing ACES via the HPRC Portal (ACCESS)

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

Log-in using your ACCESS credentials.

The screenshot shows the ACCESS portal interface. At the top, there is a 'Consent to Attribute Release' dialog box with a dropdown arrow. Below it, a list of attributes being requested is shown: 'Your CILogon user identifier', 'Your name', 'Your email address', and 'Your username and affiliation from your identity provider'. Below the consent dialog is a 'Select an Identity Provider' dropdown menu. The selected option is 'ACCESS CI (XSEDE)'. Below the dropdown is a 'Log On' button. At the bottom, there is a footer with links for 'For questions about this site, please see the FAQs or send email to help@cilogon.org', 'Know your responsibilities using the CILogon Service', and 'See acknowledgements of support for this site'.

The screenshot shows the 'Login to CILogon' form. It has two input fields: 'ACCESS Username' and 'ACCESS Password'. Below the password field is a checkbox labeled 'Don't Remember Login'. A 'Login' button is at the bottom. To the right of the form is the CILogon logo and text: 'CILogon facilitates secure access to CyberInfrastructure (CI)'. Below the logo are links: 'Register for an ACCESS Account', 'Forgot your password?', and 'Need Help?'. At the bottom, there is a link: 'Click Here for Assistance'.

This is a close-up of the 'Select an Identity Provider' dropdown menu. The selected option is 'ACCESS CI (XSEDE)' with a question mark icon to its right.

Select the Identity Provider appropriate for your account.

Shell Access via the HPRC Portal

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

- *shell* is also called
terminal or
command line

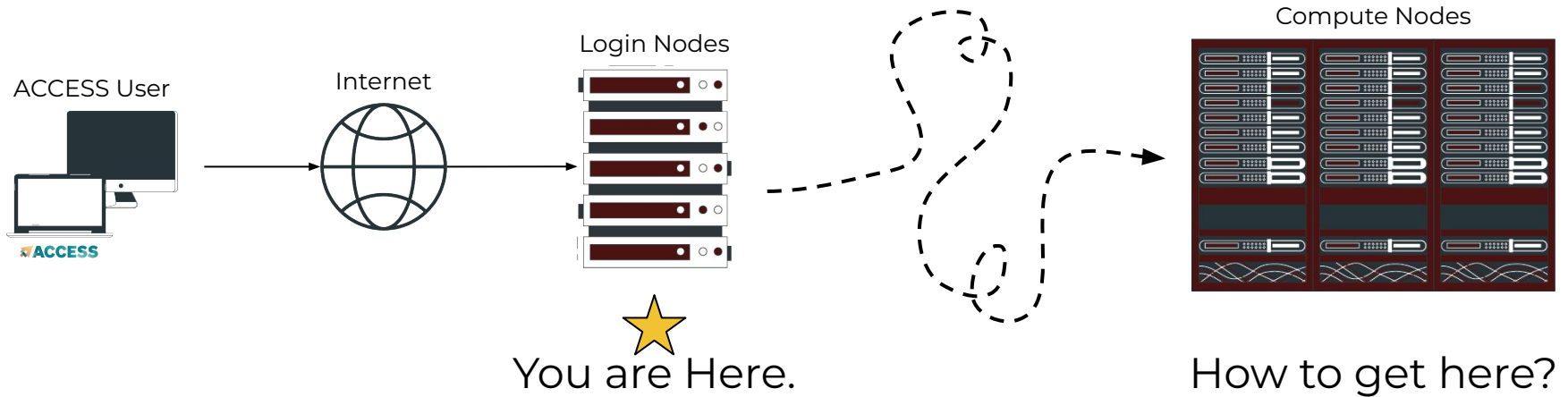


Hands-On Activity - 2 Minutes

- Connect to ACES now through the portal using portal-aces.hprc.tamu.edu
- Get a shell on the ACES cluster from the Clusters menu

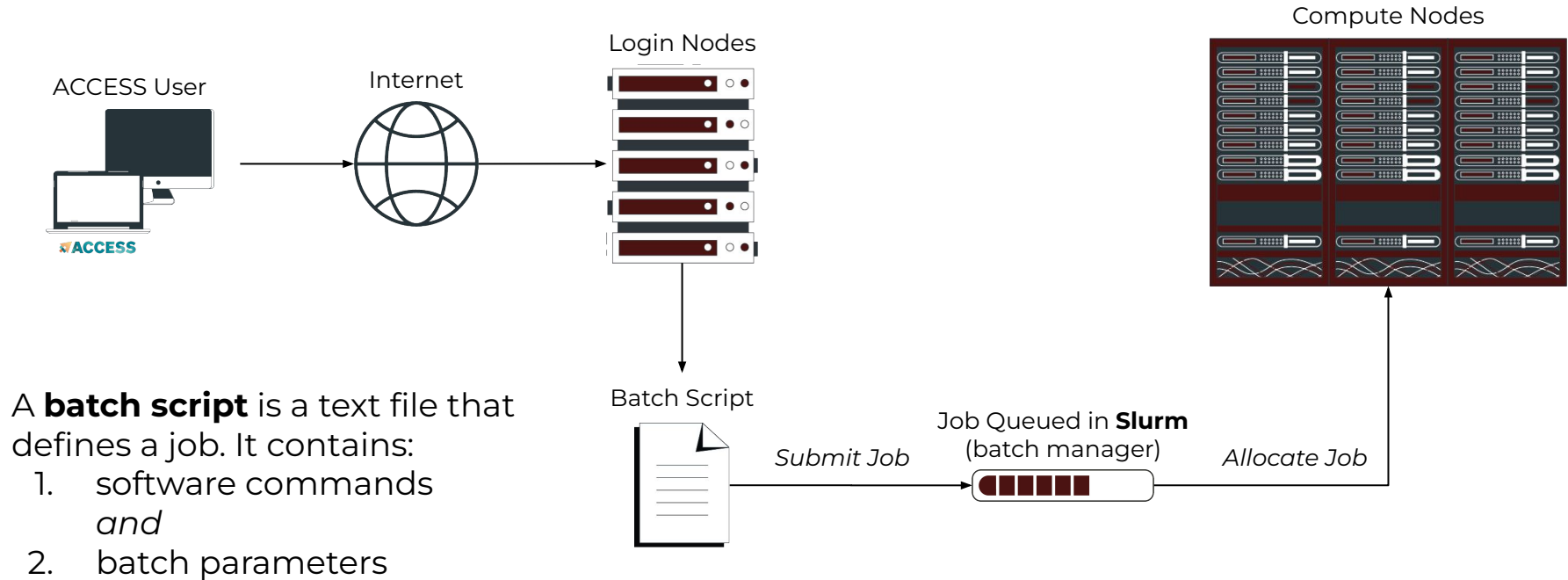
What is the hostname of the machine you connected to?

Batch Computing on the Clusters



- Types of nodes:
 - Login node - a shared machine for light editing
 - Compute node - an allocated machine for heavy computation

Batch Computing on ACES cluster



A **batch script** is a text file that defines a job. It contains:

1. software commands
and
2. batch parameters

The Drona Composer

- A simple app to assist you with composing jobs

Once in the Portal,
select at the top:
“Jobs” → “Drona
Composer”



DRONA COMPOSER - JOB SUBMISSION MADE EASY

DRONA COMPOSER (ACES)

Job Composer

Job Name

Location Change /scratch/user/u.as190723/job_composer

Environments Generic ▼

Upload files (optional) Select an option ▼ Add

Add modules Default (intel/2023a) ▼ Add

Number of cores (if not sure, keep at 1)

Number of nodes (if not sure, leave as 0)

Use Accelerator -- Choose an option -- ▼

TOTAL Memory GB ▼

Expected run time Days Hours Minutes

Account (leave blank if using default?)

Additional Slurm parameters?

Preview

⚠ Cautions: Job files will overwrite existing files with the same name. The same principle applies for your executable scripts.

You can select environments.

You can add modules.

You can choose your choice of accelerator.

You can set the expected runtime.

Sample Job Script Structure

```
#!/bin/bash

##NECESSARY JOB SPECIFICATIONS
#SBATCH --job-name=hello_world
#SBATCH --time=00:15:00
#SBATCH --ntasks=2
#SBATCH --ntasks-per-node=2
#SBATCH --nodes=1
#SBATCH --mem=3G
#SBATCH --output=hello_world_log.%j

# load required module(s)
module purge
module load GCCcore/11.3.0
module load Python/3.10.4
python hello_world.py

# Job Environment variables
echo $SLURM_JOBID
echo $SLURM_SUBMIT_DIR
echo $TMPDIR
echo $SCRATCH
```

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

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

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

Hands on Activity - 5 minutes

1. Create a directory in **\$SCRATCH** called **drona_composer**:

```
cd $SCRATCH  
mkdir drona_composer/environments
```

2. Copy **MyGeneric** in **/scratch/training/slurm_scheduler**:

```
cd $SCRATCH/drona_composer/environments/  
cp -r /scratch/training/slurm_scheduler/MyGeneric
```

3. Customize your environment by editing the **schema.json** file inside **MyGeneric**:

```
cd MyGeneric  
vi schema.json
```

4. See the changes after reloading the Drona Composer page.

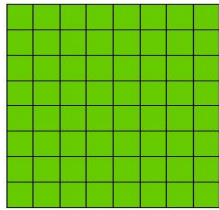
Important Batch Job Parameters

Slurm	Comment
<code>#SBATCH --time=HH:MM:SS</code>	Specifies the time limit for the job. Must specify seconds SS on ACES
<code>#SBATCH --ntasks=x</code>	Total number of tasks (cores) for the job.
<code>#SBATCH --ntasks-per-node=xx</code>	Specifies the maximum number of tasks (cores) to allocate per node
<code>#SBATCH --mem=xxxxM</code> or <code>#SBATCH --mem=xG</code>	Sets the maximum amount of memory (MB) per <i>node</i> . G for GB is supported on ACES
<code>#SBATCH --nodes=x</code>	Specifies the number of nodes to use

(These usually go in your job script file)

Mapping Jobs to Cores per Node on ACES

A.



■ = 1.5 cores

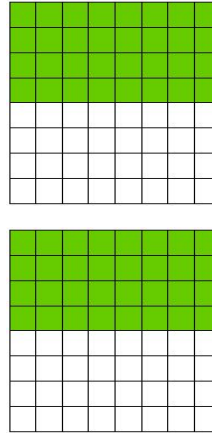
96 cores on
1 compute node

#SBATCH --ntasks=96

#SBATCH --ntasks-per-node=96

Preferred Mapping
(if applicable)

B.

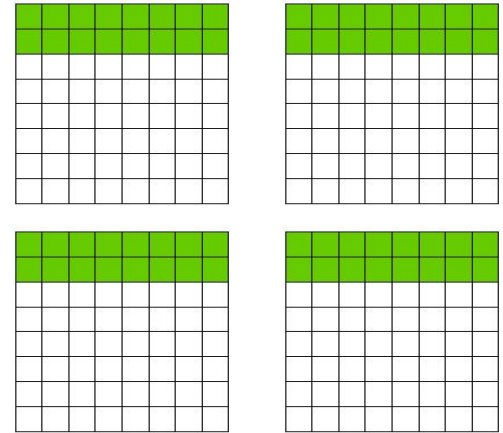


96 cores on
2 compute nodes

#SBATCH --ntasks=96

#SBATCH --ntasks-per-node=48

C.



96 cores on
4 compute nodes

#SBATCH --ntasks=96

#SBATCH --ntasks-per-node=24

Pop Quiz

```
#SBATCH --job-name=JobExample2
#SBATCH --time=48:00:00
#SBATCH --ntasks=96
#SBATCH --ntasks-per-node=24
#SBATCH --nodes=4
#SBATCH --mem=48G
#SBATCH --output=stdout.%J
#SBATCH --error=stderr.%J
```

How many cores is this job requesting?

A. 1024

C. 960

B. 96

D. 4

Job Submission and Tracking

Slurm	Description
<code>sbatch jobfile</code>	Submit jobfile to batch system
<code>squeue [-u user_name] [-j job_id]</code>	List jobs
<code>scancel job_id</code>	Kill a job
<code>sacct -X -j job_id</code>	Show information for a job (can be when job is running or recently finished)
<code>sacct -X -S YYYY-MM-DD</code>	Show information for all of your jobs since YYYY-MM-DD

(These are command-line commands,
not parts of your job script)

Job Environment Variables

Each job has access to several self-referential variables:

- **\$SLURM_JOBID** = job id
- **\$SLURM_SUBMIT_DIR** = directory where job was submitted from
- **\$TMPDIR** = /work/job.\$SLURM_JOBID

You can also still use non-Slurm variables like:

- **\$SCRATCH** = /scratch/user/username

<https://hprc.tamu.edu/kb/Helpful-Pages/Batch-Translation/#environment-variables>

Submit a Job and Check Job Status

Submit job

```
sbatch example01.slurm
```

```
Submitted batch job 6853258
(from job_submit) your job is charged as below
    Project Account: 122792016265
    Account Balance: 1687.066160
    Requested SUs:   3
```

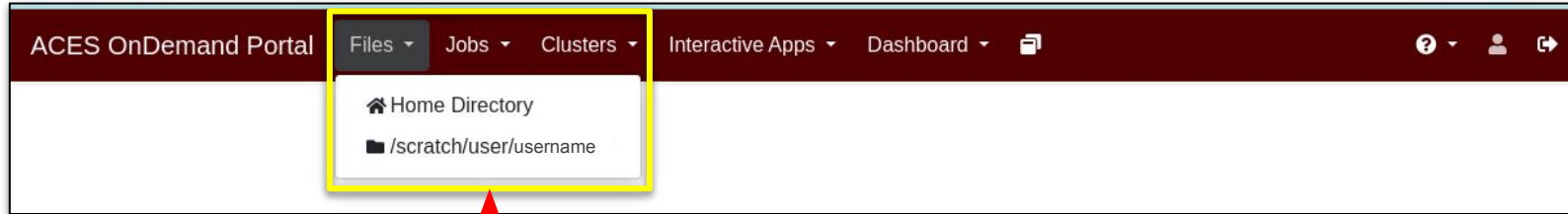
Check status

matching JOBID

```
squeue -u <username>
```

JOBID	NAME	USER	PARTITION	NODES	CPUS	STATE	TIME	TIME_LEFT	START_TIME	REASON	NODELIST
6853258	jobname	someuser	xlong	2	96	RUNNING	3-07:36:50	16:23:10	2023-01-23T17:27:3	None	c[180,202]

Open the File Navigator



In the portal, choose one of the options under “Files”

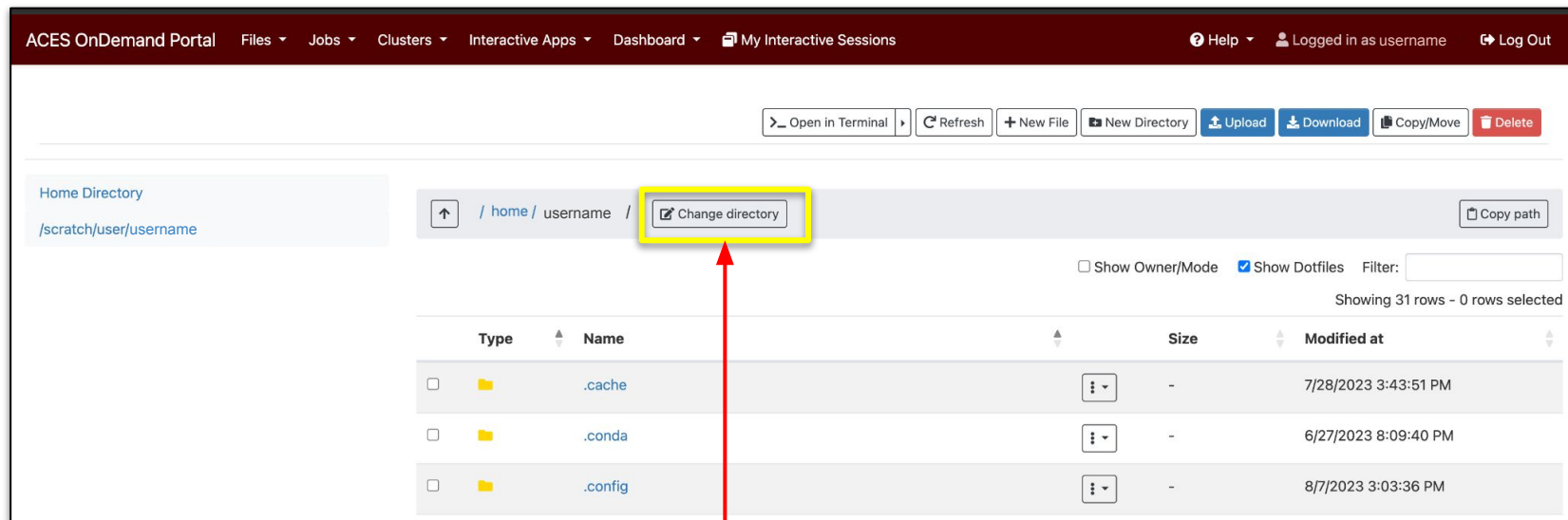
Hands-On Activity- 10 mins

There are example Files located at `/scratch/training/slurm_scheduler`

1. Copy these files to your home directory
2. Edit a batch file.
3. Submit a batch file using `sbatch`.
4. Check that the job is running in a Slurm queue with `squeue`.
5. Check the contents of the output file.

We are going to see above steps one-by-one in action in following slides.

Navigating to Training Directory



ACES OnDemand Portal Files Jobs Clusters Interactive Apps Dashboard My Interactive Sessions Help Logged in as username Log Out

> Open in Terminal Refresh + New File New Directory Upload Download Copy/Move Delete

Home Directory
/scratch/user/username

↑ / home / username / Change directory Copy path

☐ Show Owner/Mode ☒ Show Dotfiles Filter: Showing 31 rows - 0 rows selected

Type	Name	Size	Modified at
☐ Folder	.cache	-	7/28/2023 3:43:51 PM
☐ Folder	.conda	-	6/27/2023 8:09:40 PM
☐ Folder	.config	-	8/7/2023 3:03:36 PM

Click on “change directory” and type
/scratch/training/slurm_scheduler

Copy the Example Files

ACES OnDemand Portal Files Jobs Clusters Interactive Apps Affinity Groups Dashboard My Interactive Sessions Help Logged in as username Log Out

>_ Open in Terminal Refresh + New File + New Directory Upload Download Copy/Move Delete

Home Directory
/scratch/user/username

/ scratch / training / slurm_scheduler / Change directory Copy path

☐ Show Owner/Mode ☐ Show Dotfiles Filter: Showing 2 rows - 2 rows selected

Type	Name	Size	Modified at
☒	hello_world.py	73 Bytes	9/8/2023 3:54:45 PM
☒	hello_world.slurm	432 Bytes	9/8/2023 2:39:03 PM

Check box the two files and click Copy/Move

Return Home

The screenshot shows the ACES OnDemand Portal interface. At the top is a navigation bar with links: ACES OnDemand Portal, Files, Jobs, Clusters, Interactive Apps, Affinity Groups, Dashboard, and My Interactive Sessions. On the right of the navigation bar are links for Help, Logged in as username, and Log Out.

Below the navigation bar is a toolbar with buttons: Open in Terminal, Refresh, New File, New Directory, Upload, Download, Copy/Move, and Delete.

On the left is a sidebar with a dialog box titled "Copy or move the files below from /scratch/training/slurm_scheduler to the current directory:". The dialog lists two files: hello_world.py and hello_world.slurm. Below the list are "Copy" and "Move" buttons. At the bottom of the sidebar is a button labeled "Home Directory" which is highlighted with a yellow box. Below this button is the text "/scratch/user/username". A red arrow points from the "Home Directory" button to the text below it.

The main area of the interface shows a directory view for "/scratch/training/slurm_scheduler /". It includes a "Change directory" button and a "Copy path" button. Below this is a table with columns: Type, Name, Size, and Modified at. The table shows two files: hello_world.py (73 Bytes, Modified 9/8/2023 3:54:45 PM) and hello_world.slurm (432 Bytes, Modified 9/8/2023 2:39:03 PM). The table also includes checkboxes for "Show Owner/Mode" and "Show Dotfiles", and a "Filter:" input field. The text "Showing 2 rows - 2 rows selected" is displayed below the table.

Select the directory to copy to

Paste the Example Files

The screenshot shows the ACES OnDemand Portal interface. At the top, there is a navigation bar with links: Files, Jobs, Clusters, Interactive Apps, Affinity Groups, Dashboard, and My Interactive Sessions. On the right, there are links for Help, Logged in as username, and Log Out.

Below the navigation bar, there is a toolbar with buttons: Open in Terminal, Refresh, New File, New Directory, Upload, Download, Copy/Move, and Delete.

The main content area is divided into two panels. The left panel shows a dialog box with the text: "Copy or move the files below from /scratch/training/slurm_scheduler to the current directory:". Below this text, there are two files listed: hello_world.py and hello_world.slurm. At the bottom of the dialog, there are two buttons: Copy (highlighted with a yellow box and a red arrow) and Move.

The right panel shows a directory listing for the path /home/username/. The listing includes a table with columns: Type, Name, Size, and Modified at. The table contains two rows of data:

Type	Name	Size	Modified at
Folder	ACES_FundamentalsOfRProgramming	-	9/26/2023 10:56:01 AM
File	module.avail.aces	16 KB	9/18/2023 11:20:41 AM

Below the table, there is a status bar that says "Showing 2 of 28 rows - 0 rows selected".

The files on the right will change to show the directory you chose.
Hit "Copy" to actually copy the files to that directory.

View and Edit the Example Files

The screenshot displays the ACES OnDemand Portal file manager. The top navigation bar includes links for Files, Jobs, Clusters, Interactive Apps, and Dashboard. Below this, a toolbar contains buttons for 'Open in Terminal', 'Refresh', 'New File', 'New Directory', 'Upload', 'Download', 'Copy/Move', and 'Delete'. The main area shows the current directory as '/home/username/' with a 'Change directory' button. A table lists files, with 'hello_world.py' selected. A context menu is open over this file, showing options: View, Edit, Rename, Download, and Delete. The footer indicates the portal is powered by OPEN OnDemand and shows the version 3.0.0.

ACES OnDemand Portal Files Jobs Clusters Interactive Apps Dashboard

Open in Terminal Refresh New File New Directory Upload Download Copy/Move Delete

Home Directory
/scratch/user/username

/ home / username / Change directory Copy path

Show Owner/Mode Show Dotfiles Filter: Showing 2 of 21 rows - 0 rows selected

Type	Name	Size	Modified at
File	hello_world.py	73 Bytes	9/11/2023 10:49:24 AM
File	hello_world.slurm		9/11/2023 10:49:24 AM

View Edit Rename Download Delete

powered by OPEN OnDemand OnDemand version: 3.0.0

Open a Terminal (another option)

ACES OnDemand Portal Files Jobs Clusters Interactive Apps Affinity Groups Dashboard My Interactive Sessions Help Logged in as username Log Out

>_ Open in Terminal Refresh + New File New Directory Upload Download Copy/Move Delete

Home Directory
/scratch/user/username

↑ / scratch / training / slurm_scheduler / Change directory Copy path

☐ Show Owner/Mode ☐ Show Dotfiles Filter:

Showing 2 rows - 0 rows selected

	Type	Name		Size	Modified at
☐		hello_world.py		73 Bytes	9/8/2023 3:54:45 PM
☐		hello_world.slurm		432 Bytes	9/8/2023 2:39:03 PM

Hands on Activity- Cont.

1. Investigate the two example files
 - Make a small edit to personalize.
(e.g., “Hello MyName”)
2. Submit the batch file using:
`sbatch hello_world.slurm`
3. Check the job status using:
`squeue -u $USER`
4. Once the job is completed, inspect the output file:
`hello_world_log.<job_id>`

Consumable Computing Resources

- Resources which we can specify in a job/slurm file:
 - Processor cores
 - Memory
 - Wall time
 - GPU
- Other resources:
 - SUs
 - Software license/token
 - Use `license status` to query
 - License Checker:
httprc.tamu.edu/kb/Software/useful-tools/License_Checker/

Find available license for "FEMZIP":

```
license_status -s FEMZIP
```

License status for FEMZIP:

License Name	# Issued	# In Use	# Available
komp	100	0	100

More information on this command:

```
license_status -h
```

Key:

Your inputs in green

Terminal output in blue

How Does Slurm Assign Jobs?

- Job submissions are auto-assigned to batch *queues* (also called *partitions*) based on the resources requested
 - number of cores/nodes and walltime limit
 - specific resources requested
- Some jobs can be directly submitted to a queue:
 - If gpu nodes are needed, use the gpu partition/queue:
`#SBATCH --partition=gpu`

<https://hprc.tamu.edu/kb/User-Guides/Common/BatchProcessing/#batch-queues>

sinfo: Info for Node/Partition

To check the status of the nodes/partitions:

sinfo

PARTITION	AVAIL	TIMELIMIT	JOB_SIZE	NODES (A/I/O/T)	CPUS (A/I/O/T)
cpu*	up	7-00:00:00	1-64	17/48/13/78	1537/4703/1248/7488
gpu	up	2-00:00:00	1-15	14/1/2/17	1025/415/192/1632
atasp	up	2-00:00:00	1	0/2/2/4	0/192/192/384
bittware	up	2-00:00:00	1	0/0/2/2	0/0/192/192
d5005	up	2-00:00:00	1	0/2/0/2	0/192/0/192
memverge	up	2-00:00:00	1	0/5/3/8	0/480/288/768
nextsilicon	up	2-00:00:00	1	0/1/1/2	0/96/96/192
staff	up	2-00:00:00	1-110	31/59/20/110	2562/6078/1920/10560

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 ACES
- -p allows you to show a specific partition
- Example:

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

GPU GRES (Generic Resource) is printed after each jobid

Print only nodes in partition gpu

Hostname	Partition	Node State	Num CPU	CPUload	Memsize	Freemem	GRES/node	Joblist
			Use/Tot	(15min)	(MB)	(MB)		
ac036	gpu	alloc	72 96	2.95	500000	492233*	gpu:h100:2	<jobid>
ac037	gpu	idle	0 96	0.00	500000	479355	gpu:h100:2	
ac038	gpu	drain*	0 96	0.00	500000	489374	gpu:h100:2	
ac039	gpu	idle	0 96	0.00	500000	511650	gpu:h100:2	
ac040	gpu	idle	0 96	0.00	500000	463374	gpu:h100:2	
ac046	gpu	mix	0 96	0.00	500000	442789	gpu:h100:2	<jobid>
ac047	gpu	alloc	96 96	1.97*	500000	489464	gpu:h100:2	
ac064	gpu	idle	0 96	0.00	500000	509287	gpu:a30:6	

Job Memory Requests on ACES

- If you request more resources than is allowed, Slurm will reject the job
- To check the maximum requestable amount, use:

maxconfig

```
ACES partitions:  cpu  gpu  atsp  bittware  d5005  memverge  nextsilicon
ACES GPUs in gpu partition:  a30:6  h100:2
Showing max parameters (cores, mem, time) for partition cpu
```

```
#!/bin/bash
#SBATCH --job-name=my_job
#SBATCH --time=7-00:00:00
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=1
#SBATCH --cpus-per-task=96
#SBATCH --mem=488G
#SBATCH --output=stdout.%x.%j
#SBATCH --error=stderr.%x.%j
```

Check your Service Unit (SU) Balance

- List the SU Balance of your Account(s) with:

```
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

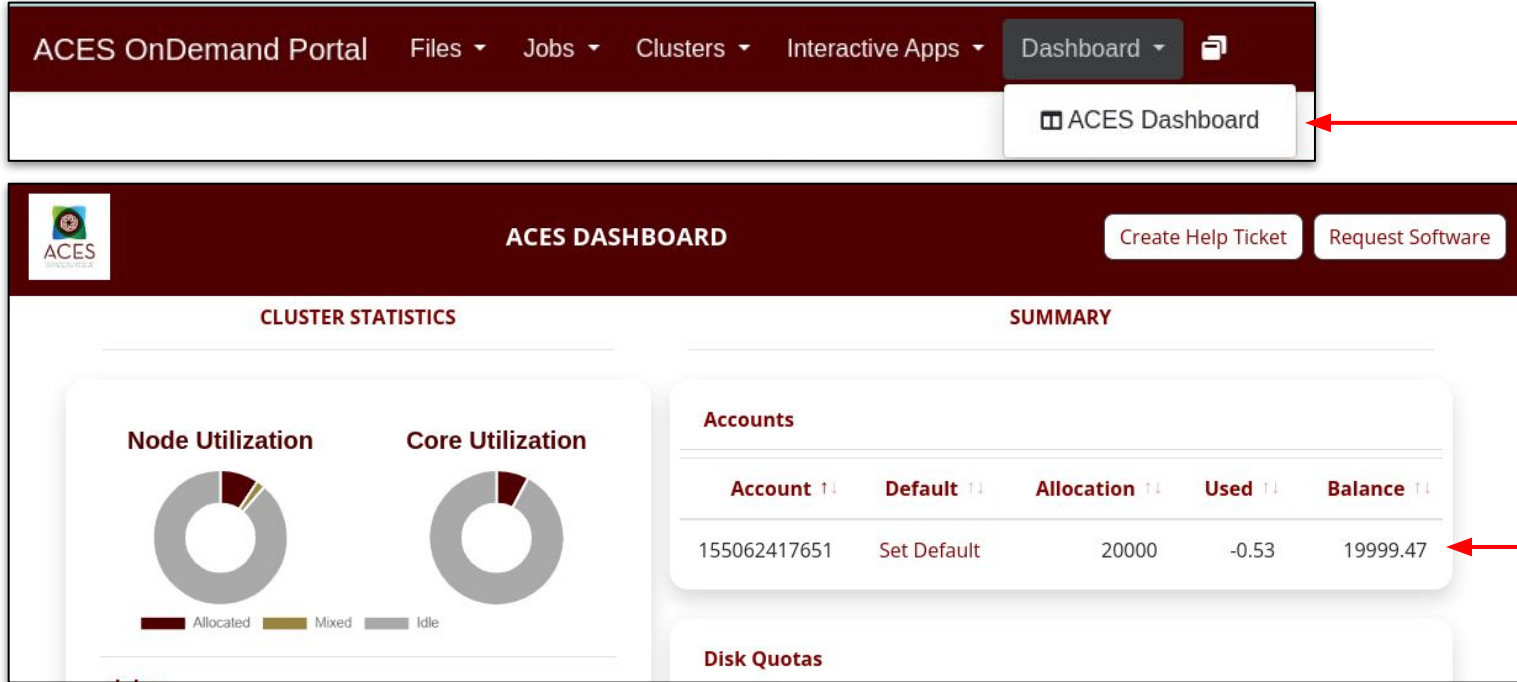
```
=====
```

- 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/>

Check your Service Unit (SU) Balance



The screenshot shows the ACES OnDemand Portal interface. The top navigation bar includes links for Files, Jobs, Clusters, Interactive Apps, and a Dashboard dropdown menu. A red arrow points to the 'ACES Dashboard' option in the dropdown. Below the navigation bar, the ACES Dashboard is displayed, featuring a logo, a 'Create Help Ticket' button, and a 'Request Software' button. The dashboard is divided into two main sections: 'CLUSTER STATISTICS' and 'SUMMARY'. The 'CLUSTER STATISTICS' section contains two donut charts for 'Node Utilization' and 'Core Utilization', with a legend indicating 'Allocated' (dark red), 'Mixed' (olive green), and 'Idle' (grey). The 'SUMMARY' section contains a table of account balances. A red arrow points to the 'Balance' column in the table.

ACES OnDemand Portal Files Jobs Clusters Interactive Apps Dashboard

ACES Dashboard

ACES DASHBOARD

Create Help Ticket Request Software

CLUSTER STATISTICS

SUMMARY

Node Utilization Core Utilization

Allocated Mixed Idle

Accounts

Account	Default	Allocation	Used	Balance
155062417651	Set Default	20000	-0.53	19999.47

Disk Quotas

Planned Charging Scheme

Resource	Service Units (per hour)	ACCESS Credits (per hour)
Intel SPR / Icelake	1	0.1
NVIDIA H100	240	30
NVIDIA A30 and Intel PVC GPUs	120	15
Bittware Agilex FPGA	200	25
Intel D5005 FPGA	50	6.2
NEC Vector Engine	150	18.7
NextSilicon coprocessor	100	12.5
Graphcore IPU Classic	120	15
Graphcore IPU Bow	150	18.7
Intel Optane Memory	150	18.7

Continued Learning

[HPRC YouTube](#)

[HPRC Homepage](#)

[ACES Quick Start Guide](#)

[ACES Portal \(ACCESS users\)](#)

help@hprc.tamu.edu

Need Help?

First check the [FAQ](#)

- [ACES 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?

u.tamu.edu/hprc_shortcourse_survey

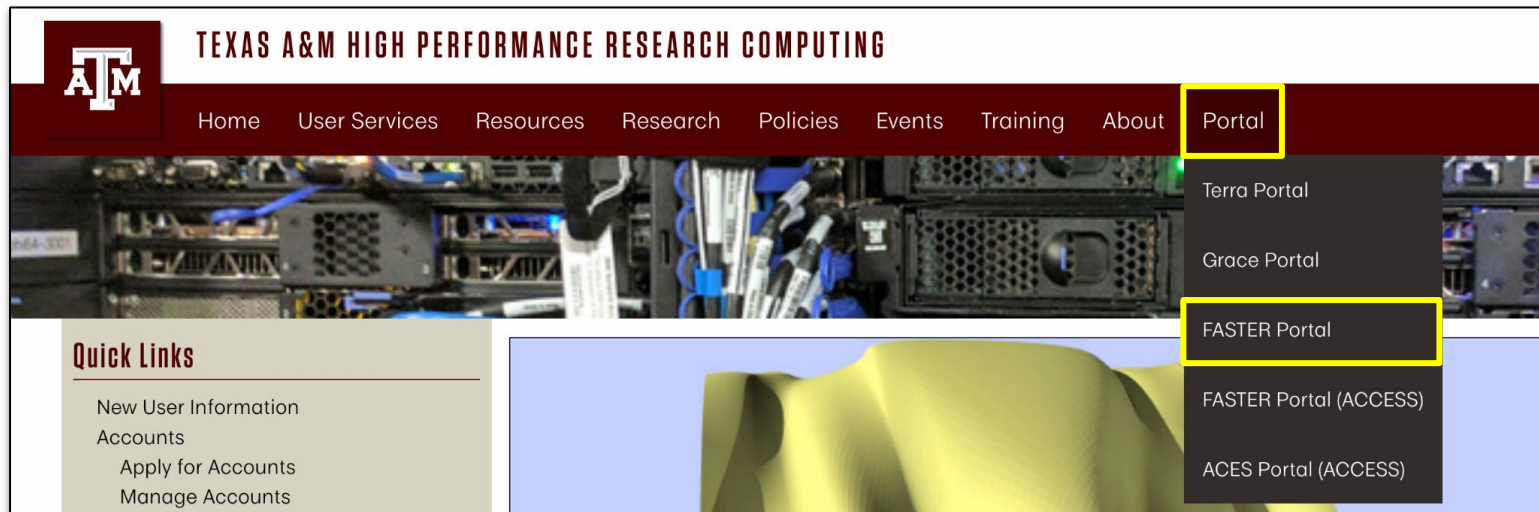
Accessing HPRC Clusters (TAMU Users Only)

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

<https://hprc.tamu.edu/kb/Helpful-Pages/#off-campus>
<https://hprc.tamu.edu/kb/Helpful-Pages/#two-factor-authentication>

Accessing the FASTER Portal (TAMU Users Only)

- HPRC webpage: hprc.tamu.edu
- Aces portal shortcut: portal-faster.hprc.tamu.edu
- Requires HPRC Account!



Slurm Job File (multi core, single node)

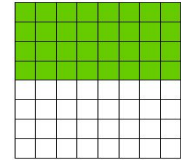
```
#!/bin/bash

##NECESSARY JOB SPECIFICATIONS
#SBATCH --job-name=JobExample2
#SBATCH --time=6:30:00
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=32
#SBATCH --mem=28G
#SBATCH --output=stdout.%j
#SBATCH --error=stderr.%j
##OPTIONAL JOB SPECIFICATIONS
#SBATCH --account=123456
#SBATCH --mail-type=ALL
#SBATCH --mail-user=email_address

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

# run your program
./my_multicore_program
```

```
# Set the job name to "JobExample2"
# Set the wall clock limit to 6hr and 30min
# Request 1 node
# Request 32 tasks(cores) per node
# Request 28GB per node
# Send stdout to "stdout.[jobID]"
# Send stderr to "stderr.[jobID]"
# Set billing account to 123456
# Send email on all job events
# Send all emails to email_address
```



SUs = 208

Slurm Job File (multi core, multi node)

```
#!/bin/bash
```

```
##NECESSARY JOB SPECIFICATIONS
```

```
#SBATCH --job-name=JobExample3
```

```
#SBATCH --time=1-12:00:00
```

```
#SBATCH --ntasks=8
```

```
#SBATCH --ntasks-per-node=2
```

```
#SBATCH --mem=2.5G
```

```
#SBATCH --output=stdout.%j
```

```
# Set the job name to "JobExample3"
```

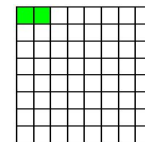
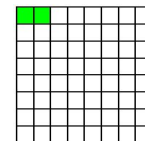
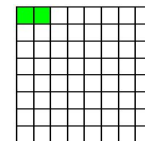
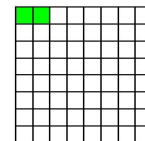
```
# Set the wall clock limit to 1 Day and 12hr
```

```
# Request 8 tasks (cores)
```

```
# Request 2 tasks(cores) per node
```

```
# Request 2.5 GB per node
```

```
# Send stdout and stderr to "stdout.[jobID]"
```



SUs = 96

```
##OPTIONAL JOB SPECIFICATIONS
```

```
#SBATCH --account=123456
```

```
"myproject")
```

```
#SBATCH --mail-type=ALL
```

```
#SBATCH --mail-user=email_address
```

```
# Set billing account to 123456 (find your account with
```

```
# Send email on all job events
```

```
# Send all emails to email_address
```

```
# purge and then load the modules required for your software
```

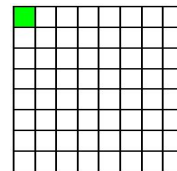
```
module purge
```

```
module load GCC/11.3.0 OpenMPI/4.1.4
```

```
# run program with MPI
```

```
mpirun my_multicore_multinode_program
```

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)
- Large number of concurrent single core jobs
 - Check out ***tamulauncher***
 - <https://hprc.tamu.edu/kb/Software/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