

Introduction to Composable Computing on NSF ACES

Jan 20, 2026
Josh Winchell



High Performance
Research Computing
DIVISION OF RESEARCH



Outline

- (Brief Disambiguation)
- ACES and Composability Overview
- Documentation and Training
- Getting Started on the ACES Cluster
- OOD Portal and Cluster Basics
- Software Infrastructure
- Batch Computing
- Composability, Accelerators, and AI/ML
- Bonus: ACES Configuration

Disambiguation

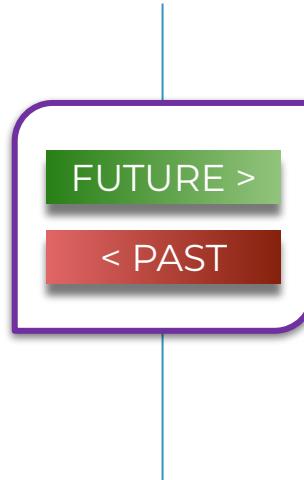
- **HPC** = “high performance computing”
(a general field)
- **HPRC** = “High Performance Research Computing”
(us – Texas A&M University’s HPC center)
- **TACC** = “Texas Advanced Computing Center”
(unrelated to us – UT Austin’s HPC center)
- **ACES** = “Advancing Computing for Emerging Sciences”
(one of HPRC’s clusters)
- **ACCESS** = “Advanced Cyberinfrastructure Coordination Ecosystem: Services & Support”
(nation-wide NSF program for HPC clusters)

ACES and Composability Overview

Composable HPC Architectures for AI

Common HPC

- Built on Converged Hardware
- Static Hardware Design
- Fixed GPU/Accelerator
- Fixed Memory
- Storage: SATA and SAS
- Vendor Lock

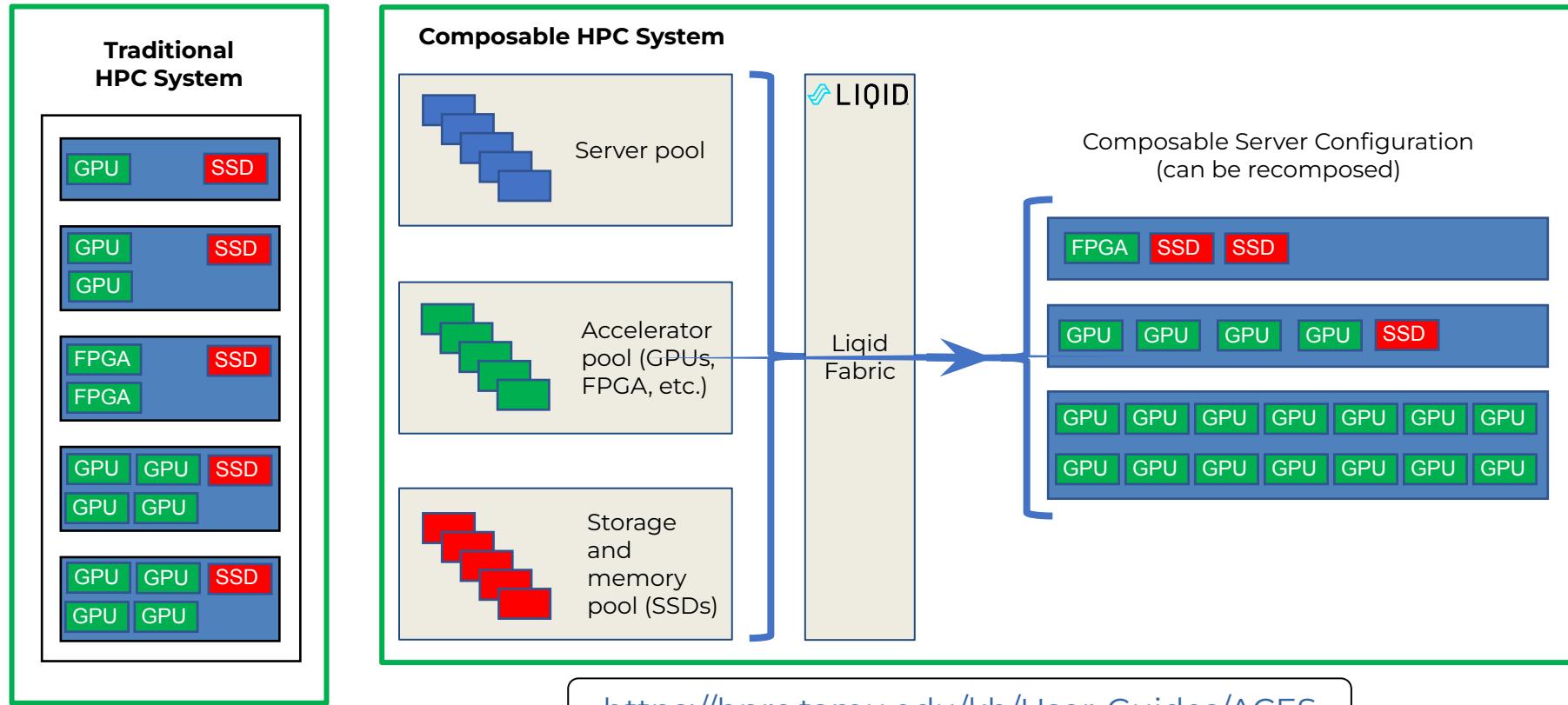


HPC for AI

- Built on Disaggregated Hardware
- Composable Hardware Platform
- Composable GPU/Accelerator
- Composable Memory - Optane
- Modern Storage: NVMe-oF
- Open Platform

Next Generation HPC/AI Platform Supports Composable Accelerators and Memory

Composability

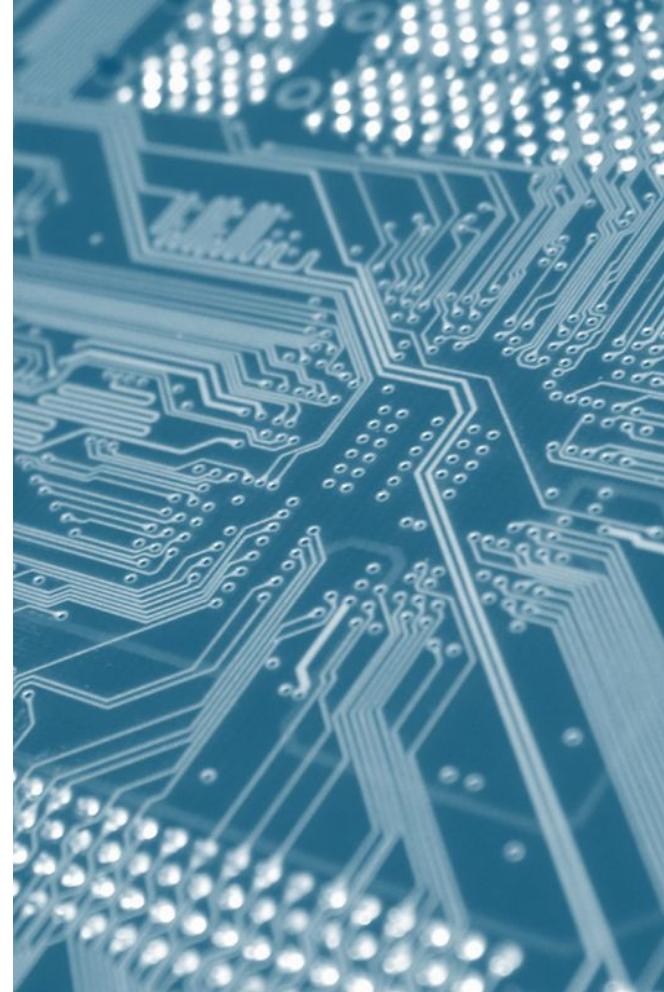


<https://hprc.tamu.edu/kb/User-Guides/ACES>

HPRC's Composable Clusters

- **FASTER** – First large-scale composable CPU/GPU system
- **ACES** – Composability for mixed-resource workflows

*FASTER is not available through
ACCESS, so we'll focus on ACES today*



NSF ACES

Accelerating Computing for Emerging Sciences

Our Mission:

- NSF ACSS CI testbed
- Offer an accelerator testbed for numerical simulations and **AI/ML workloads**
- Provide consulting, technical guidance, and training to researchers
- Collaborate on computational and data-enabled research.



ACES

ACCELERATING COMPUTING
FOR EMERGING SCIENCES

Login to ACES Portal

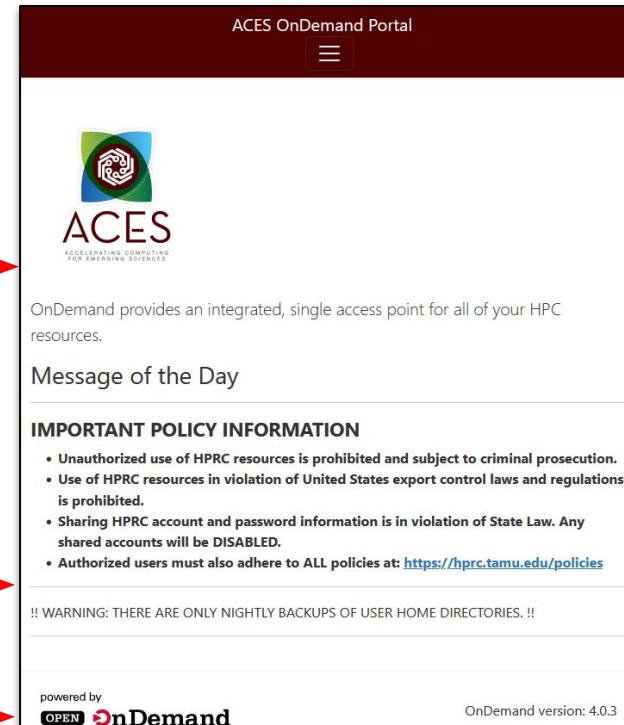


The screenshot shows the HPRC website with a banner for "H PERFORMANCE RESEARCH COMPUTING". The top navigation bar includes links for Services, Resources, Research, Policies, Events, Training, About, and Portal. The "Portal" link is highlighted with a yellow box. A dropdown menu appears, listing "Grace Portal", "FASTER Portal", "ACES Portal (ACCESS)" (which is also highlighted with a yellow box and has a red arrow pointing to the ACES OnDemand Portal screenshot), and "Launch Portal (ACCESS)". Below the menu is a background image of a server rack and a car.

(You will have to log in with ACCESS if you haven't already)

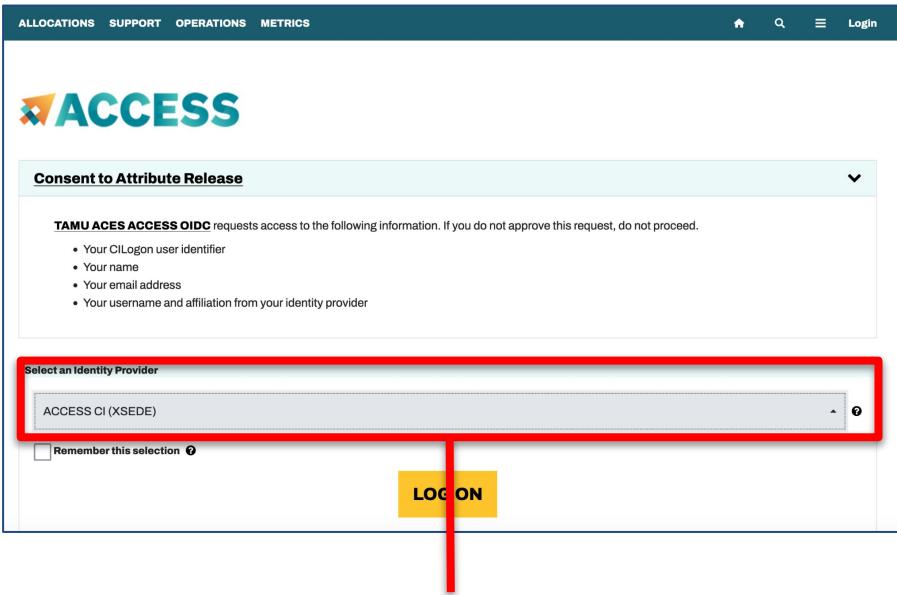
ACES Portal portal-aces.hprc.tamu.edu
is the web-based user interface for the ACES cluster

Open OnDemand (OOD) is an advanced web-based
graphical interface framework for HPC users



The screenshot shows the ACES OnDemand Portal. The header includes the ACES logo and the text "ACES OnDemand Portal". The main content area features the ACES logo and the text "ACCELERATING COMPUTING FOR EMERGING SCIENCES". Below this is a message: "OnDemand provides an integrated, single access point for all of your HPC resources." A "Message of the Day" is present. A section titled "IMPORTANT POLICY INFORMATION" contains a bulleted list of rules. At the bottom, a warning states "!! WARNING: THERE ARE ONLY DAILY BACKUPS OF USER HOME DIRECTORIES. !!". The footer includes the text "powered by OPEN OnDemand" and "OnDemand version: 4.0.3".

Login to ACES Portal - ACCESS



ALLOCATIONS SUPPORT OPERATIONS METRICS Home Search Menu Login

ACCESS

Consent to Attribute Release

TAMU ACES ACCESS OIDC requests access to the following information. If you do not approve this request, do not proceed.

- Your CILogon user identifier
- Your name
- Your email address
- Your username and affiliation from your identity provider

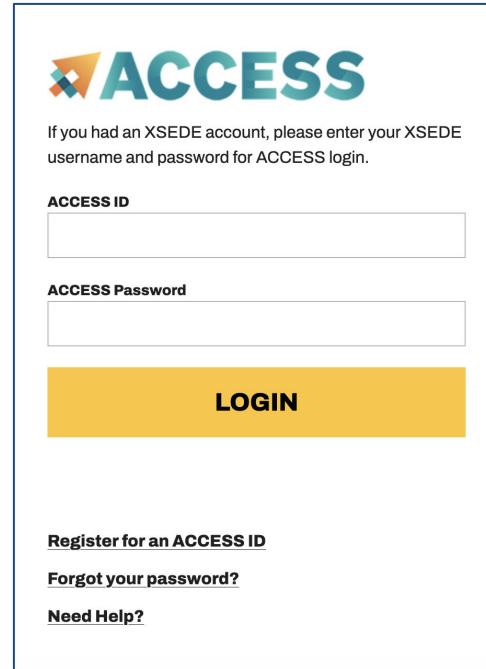
Select an Identity Provider

ACCESS CI (XSEDE)

Remember this selection ?

LOG ON

Select the Identity Provider appropriate for your account.



ACCESS

If you had an XSEDE account, please enter your XSEDE username and password for ACCESS login.

ACCESS ID

ACCESS Password

LOGIN

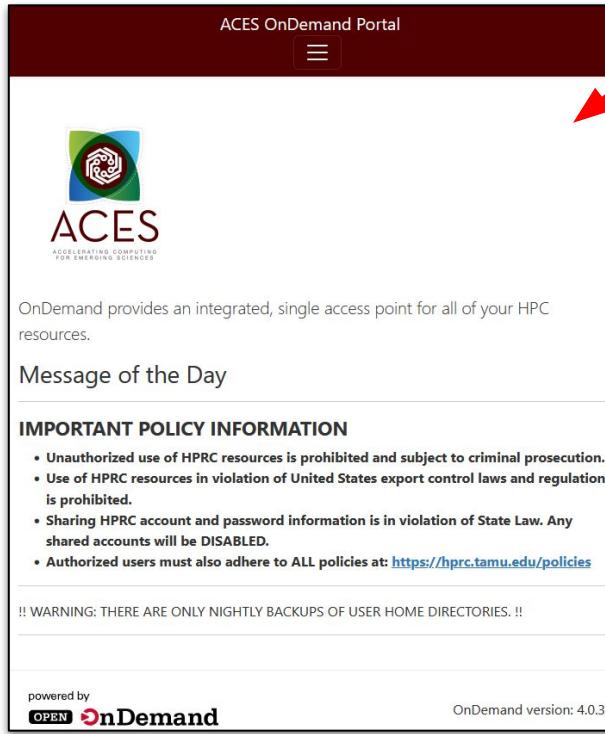
[Register for an ACCESS ID](#)

[Forgot your password?](#)

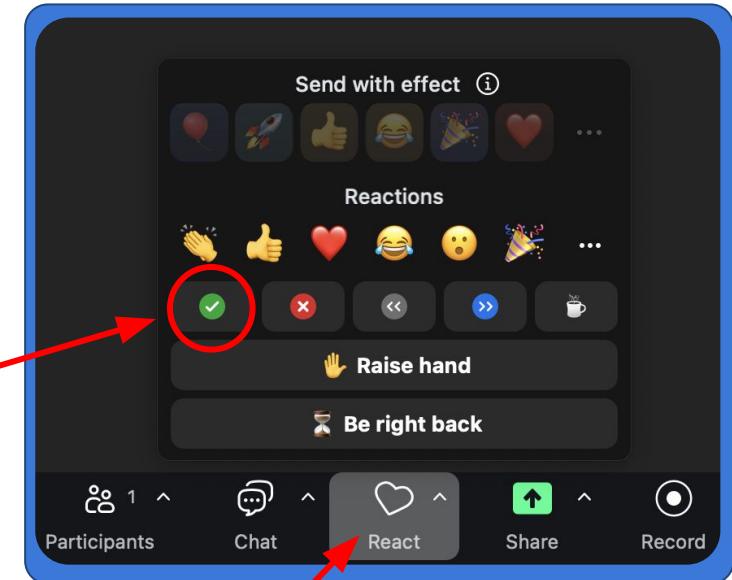
[Need Help?](#)

Log-in using your ACCESS or institutional credentials.

Login to ACES Portal



Once you're on this screen...

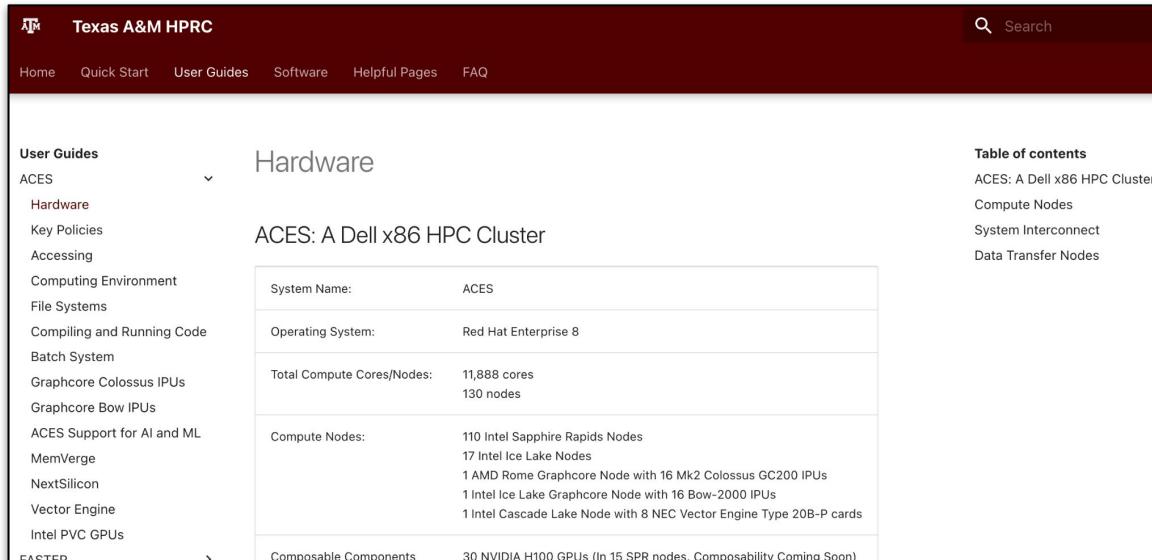


...Click the
green
checkmark
reaction in
Zoom

(find it here)

Documentation and Training

HPRC Knowledge Base



The screenshot shows a website for the Texas A&M HPRC. The top navigation bar includes links for Home, Quick Start, User Guides, Software, Helpful Pages, and FAQ. A search bar is located in the top right corner. The main content area has a sidebar on the left with a 'User Guides' section, currently expanded to show 'ACES' and its sub-sections: Hardware, Key Policies, Accessing, Computing Environment, File Systems, Compiling and Running Code, Batch System, Graphcore Colossus IPUs, Graphcore Bow IPUs, ACES Support for AI and ML, MemVerge, NextSilicon, Vector Engine, and Intel PVC GPUs. The main content area displays the 'Hardware' page for the 'ACES: A Dell x86 HPC Cluster'. The page title is 'ACES: A Dell x86 HPC Cluster'. Below the title is a table with the following data:

System Name:	ACES
Operating System:	Red Hat Enterprise 8
Total Compute Cores/Nodes:	11,888 cores 130 nodes
Compute Nodes:	110 Intel Sapphire Rapids Nodes 17 Intel Ice Lake Nodes 1 AMD Rome Graphcore Node with 16 Mk2 Colossus GC200 IPUs 1 Intel Ice Lake Graphcore Node with 16 Bow-2000 IPUs 1 Intel Cascade Lake Node with 8 NEC Vector Engine Type 20B-P cards
Composable Components:	30 NVIDIA H100 GPUs (In 15 SPR nodes, Composability Coming Soon)

On the right side of the content area, there is a 'Table of contents' sidebar with links to ACES: A Dell x86 HPC Cluster, Compute Nodes, System Interconnect, and Data Transfer Nodes.

Knowledge Base for
announcements, more hardware
details, and more about the
subjects we cover today

hprc.tamu.edu/kb
hprc.tamu.edu/kb/User-Guides/ACES

Training on YouTube



HOME VIDEOS PLAYLISTS COMMUNITY CHANNELS ABOUT 

Created playlists



ACES: Getting Started

[View full playlist](#)

ACES

[View full playlist](#)

Project Account Management (AMS)

[View full playlist](#)

Previous Primers and Short Courses:

[View full playlist](#)

Interview with LEARN President and CEO, Akbar Kara

[View full playlist](#)

HPRC OpenOnDemand Portal

[View full playlist](#)

Accelerator Training

IPU Labs

Lab I. Intro to IPU (30 mins)

We will introduce ACES, Graphcore IPU architecture, and the IPU system on TAMU ACES platform.

04

Lab IV. PyTorch on IPU (30 minutes)

We will learn to convert a PyTorch Fashion-MNIST classification model to run on IPU

Lab II. Demo on ACES (30 mins)

We will demonstrate how to run models of different frameworks on ACES IPU system.

Q&A (5 mins/lab)

03

Lab III TensorFlow on IPU (30 minutes)

We will learn to convert a Keras MNIST classification model to run on IPU

02

Figure 1. Structure of the IPU Training Laboratories.



Texas A&M HPRC

@TexasAMHPRC

825 subscribers

HOME

VIDEOS

PLAYLISTS

COMMUNITY

CHAN

Subscribed

Short Course: Graphcore Intelligence Processing Units (IPUs) on ACES (Fall...)

Texas A&M HPRC · 96 views · 2 months ago

Instructor: Zhenhua He Description: This short course includes introduction to Graphcore IPU, demonstration to run models of differ...



This semester we will have training sessions for several accelerators:

- PVCs - https://hprc.tamu.edu/training/aces_intel_pvc.html
- IPUs - https://hprc.tamu.edu/training/aces_ipus.html
- Vector Engines - TBA

See also our “ACES Training” YouTube Playlist for previous courses:

- <https://www.youtube.com/@TexasAMHPRC/playlists>

Getting Started on the ACES Cluster

Usage Policies (Be a good compute citizen)

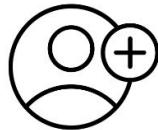
- It is illegal to share computer passwords and accounts
- Clusters must not be used in any manner that violates the United States export control laws and regulations, EAR & ITAR
- Abide by the license restrictions when using commercial software

hprc.tamu.edu/policies

Allocations Management



You can get allocations on ACES through both ACCESS and the NAIRR Pilot.



CREATE
ACCOUNT



SELECT
OPPORTUNITY



REQUEST
ALLOCATION



RECEIVE
CREDITS

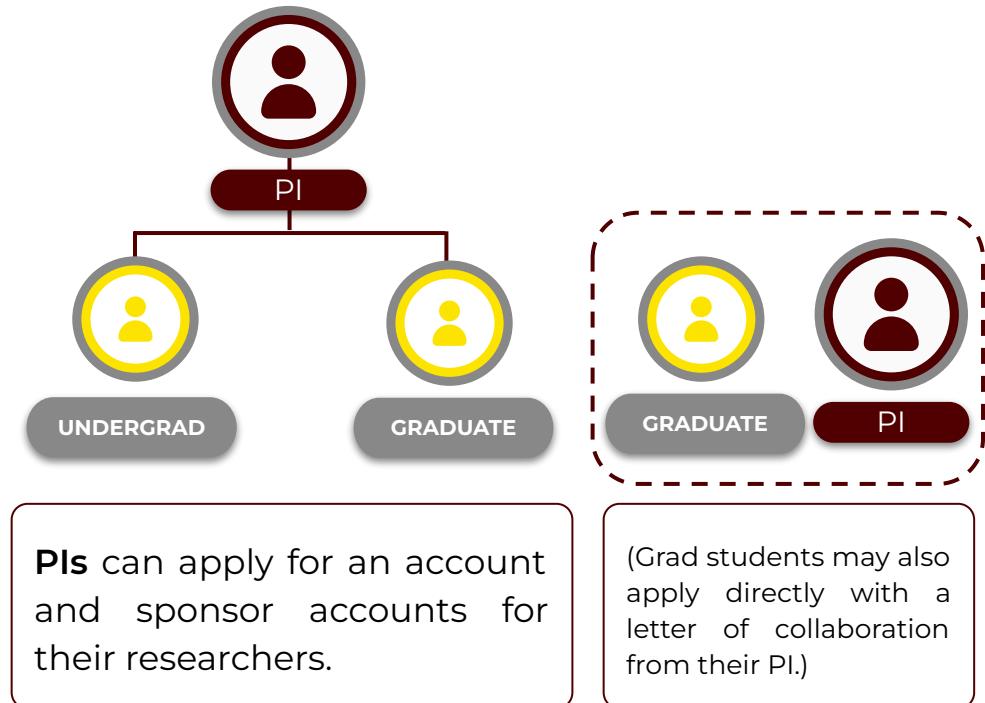


EXCHANGE
CREDITS

This is all free!

Getting on ACES

- Using an [ACCESS](#) account
- Application for ACES is available through ACCESS:
<https://allocations.access-ci.org>
- Email us at help@hprc.tamu.edu for questions, comments, and concerns.





Allocations Opportunities

See also:

<https://hprc.tamu.edu/policies/allocations.html>

Preparing Your Explore ACCESS Request

To request an Explore ACCESS allocation, submit:

- An overview of the research questions you intend to explore along with any details on how you intend to integrate ACCESS resources into your investigations.
- CVs for the PI and any co-PIs, in PDF format.
- Letter of collaboration if a Graduate Student, in PDF format.
- The following key data fields:
 - Title of the project
 - Keywords pertaining to the research
 - Field of science

We welcome requests from **graduate students** to help them complete a thesis or dissertation. Graduate students listed as PI should include a letter of collaboration from their advisor on institutional letterhead stating that the proposed work is being performed primarily by the graduate student and is separate from other funded grants or the advisor's own research. In addition, the advisor must be added to the allocation as a co-PI.

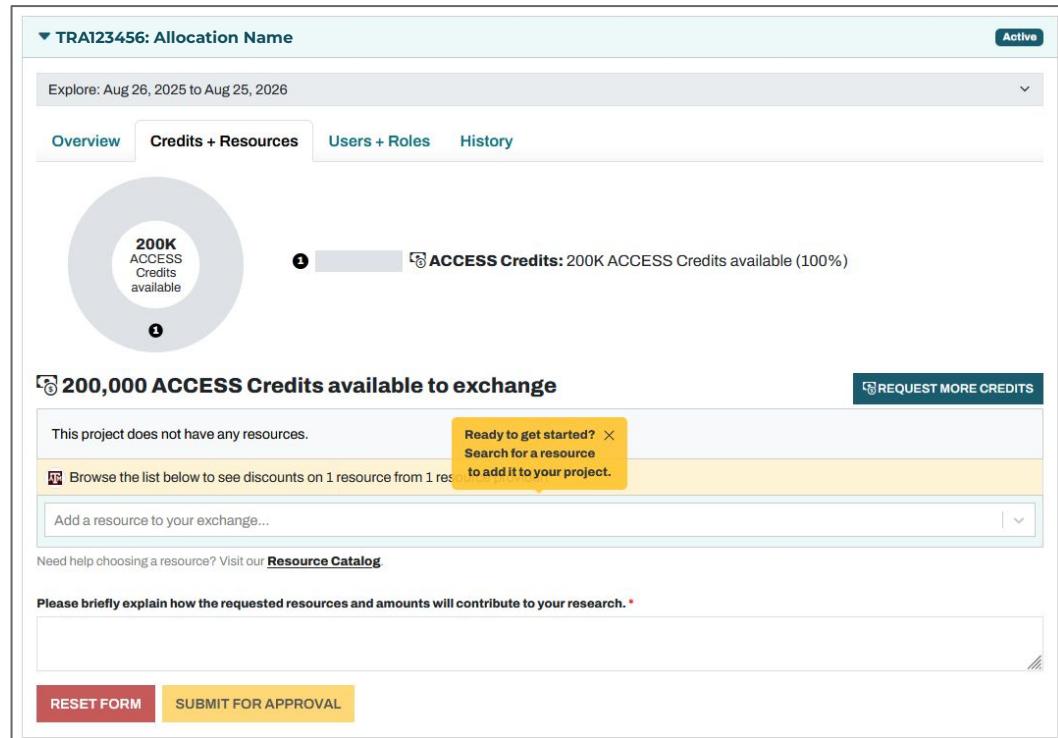
	Explore	Discover	Accelerate	Maximize
Purpose	Resource Evaluations, grad student projects, small classes, etc.	Large classes, benchmarking at-scale, Campus Champions	Multi-grant programs, Collaborations, Growing gateways	Large-scale research requiring more resources
Requests Accepted	Continuously; multiple requests allowed			Every 6 months; usually only 1 allowed
Review requirements	Overview	1-page proposal	3-page (max) proposal	10-page (max) proposal

ACCESS to ACES

Once your ACCESS allocation is approved, you request credits on ACES.

Once the resource provider (us) approves, we will set up your accounts on the cluster.

From then on, you can log into the cluster using your ACCESS credentials.



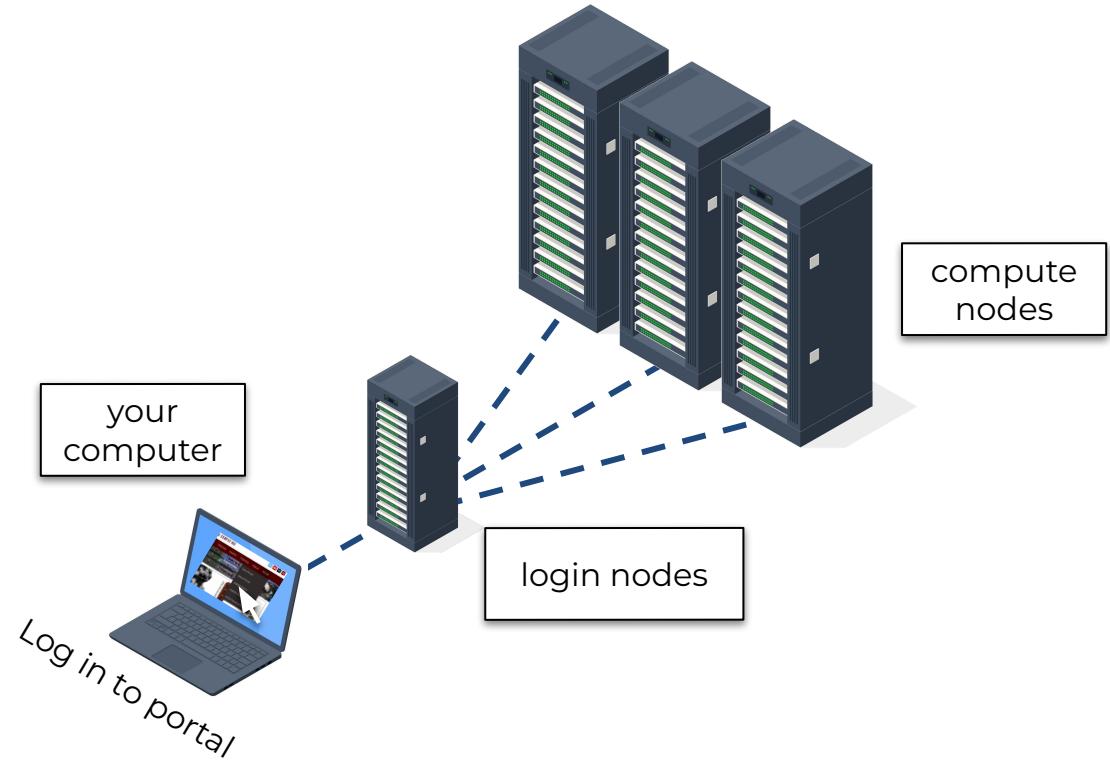
The screenshot shows the ACES Allocation Management interface for allocation TRA123456. The interface includes a navigation bar with tabs for Overview, Credits + Resources (selected), Users + Roles, and History. A large circular icon indicates 200K ACCESS Credits available. A message box states "200,000 ACCESS Credits available to exchange". A yellow callout box says "Ready to get started? Search for a resource to add it to your project." Buttons for "REQUEST MORE CREDITS" and "SUBMIT FOR APPROVAL" are visible at the bottom.

Portal and Cluster Basics

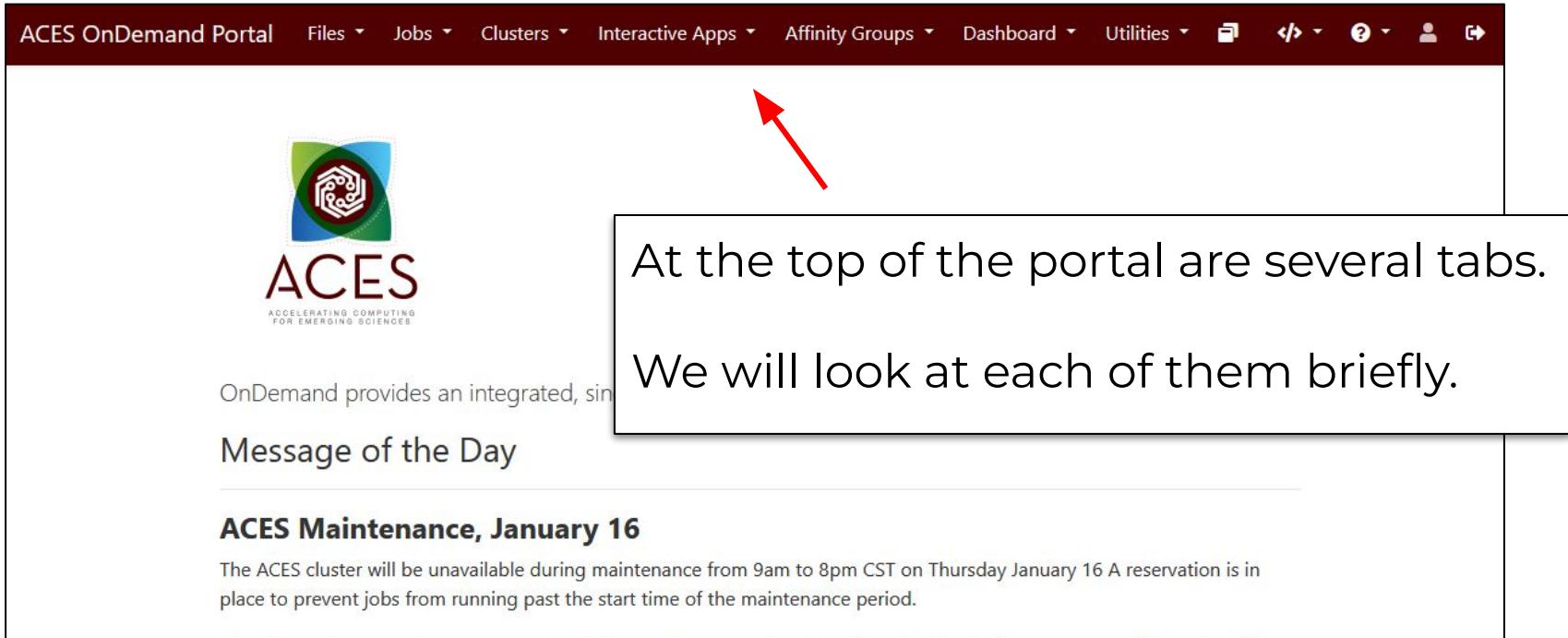
- Portal Walkthrough
- Quotas and SUs

Batch Computing on Clusters

- Interact via **your own machine**.
- Log in to the cluster's **OOD portal** (on the **login nodes**) and write instructions.
- Send instructions to **compute nodes** to do the heavy-lifting (spending credits/SUs).



HPRC Portal Overview



ACES OnDemand Portal Files ▾ Jobs ▾ Clusters ▾ Interactive Apps ▾ Affinity Groups ▾ Dashboard ▾ Utilities ▾          


ACES
ACCELERATING COMPUTING
FOR EMERGING SCIENCES

OnDemand provides an integrated, sin

Message of the Day

ACES Maintenance, January 16

The ACES cluster will be unavailable during maintenance from 9am to 8pm CST on Thursday January 16 A reservation is in place to prevent jobs from running past the start time of the maintenance period.

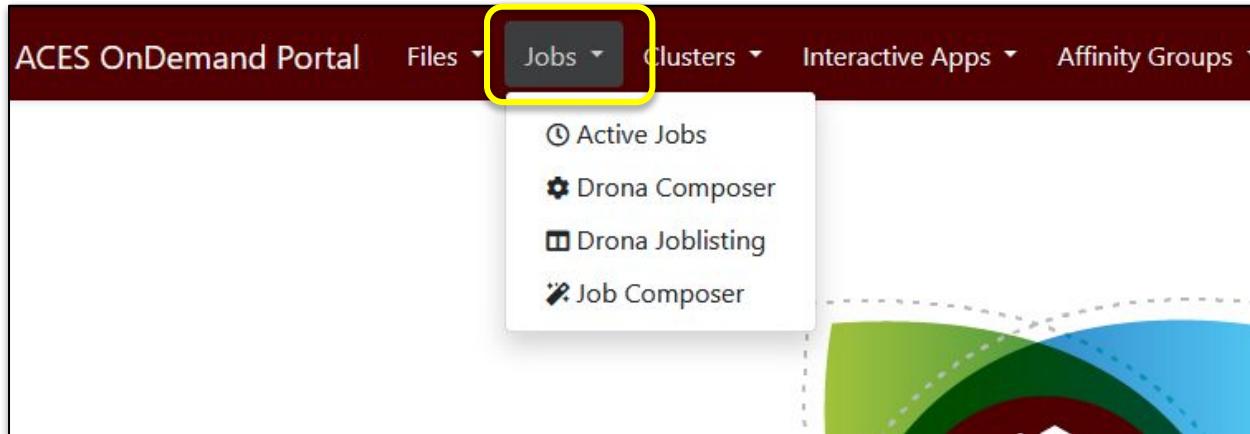
At the top of the portal are several tabs.
We will look at each of them briefly.

File Browsing, Viewing, and Editing

The screenshot shows the ACES OnDemand Portal interface. A yellow box highlights the 'Files' dropdown menu in the top navigation bar. A red arrow points from a text box on the left to this 'Files' menu. The main area displays a file list in a table format. A yellow box highlights the three-dot menu icon for the file 'hello_world.slurm'. A context menu is open for this file, showing options: View, Edit, Rename, Download, and Delete. The table headers are Type, Name, Size, and Modified at. The table data includes:

Type	Name	Size	Modified at
Folder	ACES_FundamentalsOfRProgramming	-	9/26/2023 10:56:01 AM
File	hello_world.py	73 Bytes	2/2/2024 11:45:32 AM
File	hello_world.slurm	432 Bytes	2/2/2024 11:45:32 AM
File	module.avail.aces		9/18/2023 11:20:41 AM

Portal Job Composers



Manage jobs without using the command line (will talk more about these later)

- “Active Jobs” is like squeue.
- “Job Composer” lets you:
 - Create job scripts
 - Save job templates
 - Monitor your own jobs

We will focus on the new “Drona” options:

- “Drona Joblisting” shows current and past jobs.
- “Drona Composer” helps you build workflows and manage job scripts.

Shell Access via the Portal

ACES OnDemand Portal Files ▾ Jobs ▾ Clusters ▾ **Interactive Apps ▾** Affinity Groups ▾ Dashboard

>_ aces Shell Access



ACES
ACCELERATING COMPUTING
FOR EMERGING SCIENCES

Get a shell terminal
right in your browser

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 subject to criminal prosecution.
- Use of HPRC resources in violation of United States export control laws and regulations
- Sharing HPRC account and password information is in violation of State Law. Any sharing is **DISABLED**.
- Authorized users must also adhere to ALL policies at: <https://hprc.tamu.edu/policies>

!! WARNING: THERE ARE ONLY NIGHTLY BACKUPS OF USER HOME DIRECTORIES. !!

Host: logitchens Themes: Default

Warning: Permanently added 'login.aces,10.71.1.13' (ECDSA) to the list of known hosts.

This computer system and the data herein are available only for authorized purposes by authorized users. Use for any other purpose is prohibited and may result in disciplinary actions or criminal prosecution against the user. Usage may be subject to security testing and monitoring. There is no expectation of privacy on this system except as otherwise provided by applicable privacy laws. Refer to University SAP 29.01.03.M0.02 Acceptable Use for more information.

Last login: Mon Feb 12 13:11:13 2024 from 10.71.1.6

Texas A&M University High Performance Research Computing

Website:	https://hprc.tamu.edu
Consulting:	https://hprc.tamu.edu (preferred) or (979) 845-0219
ACES Documentation:	https://hprc.tamu.edu/User-Guides/ACES
FASTER Documentation:	https://hprc.tamu.edu/kb/User-Guides/FASTER
Grace Documentation:	https://hprc.tamu.edu/kb/User-Guides/Grace
Terra Documentation:	https://hprc.tamu.edu/kb/User-Guides/Terra
YouTube Channel:	https://www.youtube.com/texassamhprc

===== IMPORTANT POLICY INFORMATION =====
* - Unauthorized use of HPRC resources is prohibited and subject to criminal prosecution.
* - Use of HPRC resources in violation of United States export control laws and regulations is prohibited. Current HPRC staff members are US citizens and legal residents.
* - Sharing HPRC account and password information is in violation of Texas State Law. Any shared accounts will be DISABLED.
* - Authorized users must always adhere to ALL policies at: <https://hprc.tamu.edu/policies/>

**** ACES Partial Availability, February 12 ***

We are still troubleshooting issues for various compute nodes that were reconfigured for PCIe fabric connectivity to the H100 and PVCs.

!! WARNING: THERE ARE ONLY NIGHTLY BACKUPS OF USER HOME DIRECTORIES. !!

Please restrict usage to 8 CORES across ALL login nodes.
Users found in violation of this policy will be SUSPENDED.

To see these messages again, run the **motd** command.

Your current disk quotas are:

Disk	Disk Usage	Limit	File Usage	Limit
/home/u.jw123527	169M	10.0G	499	10000
/scratch/user/u.jw123527	28.1G	1.0T	102472	250000

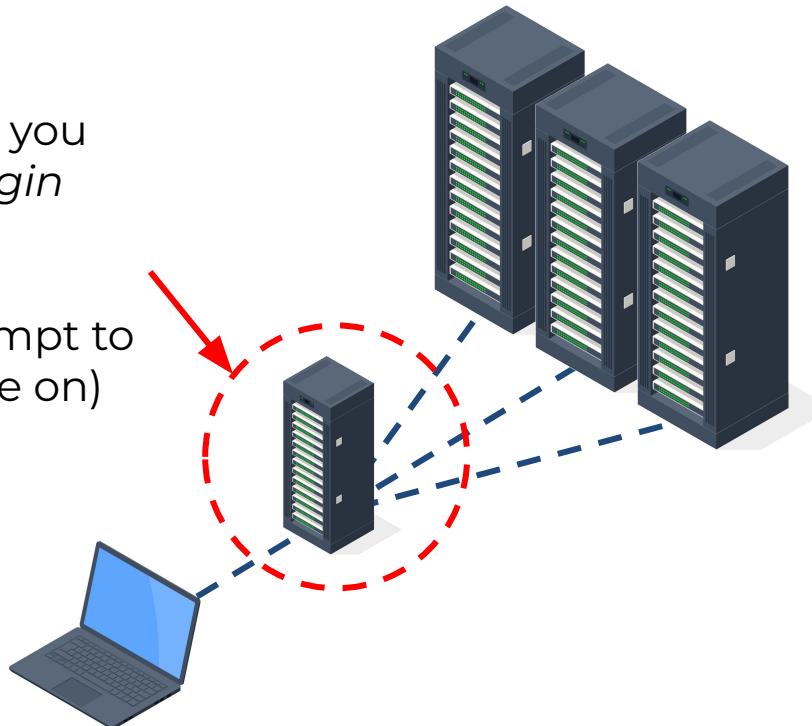
Type 'showquota' to view these quotas again.

[u.jw123527@aces-login3 ~]\$

Shell Access via the Portal: Logging In

When you first log in, you are on a dedicated *login node*.

(check your shell prompt to see which one you are on)



Login nodes are not for running big processes!

There are rules:

- No processes longer than 1 hr
- Sessions idle for 1 hr will be killed
- Do not use more than 8 cores.
- Do not use “sudo”

Interactive Apps

Join Affinity Groups

ACES OnDemand Portal Files ▾ Jobs ▾ Clusters ▾ Interactive Apps ▾ **Affinity Groups ▾** Dashboard ▾ Util

ACES
R HPC

Links to support.access-ci.org

OnDemand provides an integrated, single access point for all of your computing needs.

Message of the Day

IMPORTANT POLICY INFORMATION

- Unauthorized use of HPRC resources is prohibited and subject to criminal prosecution.
- Use of HPRC resources in violation of United States export control laws is prohibited and subject to criminal prosecution.

ACES

ACCELERATING COMPUTING FOR EMERGING SCIENCES

Quick Links Community CCEP Knowledge Base MATC

SUPPORT / AFFINITY GROUPS / ACES

ACES TAMU novel-accelerators

Accelerating Computing for Emerging Sciences (ACES) is a NSF-Category II-funded test bed (award number 2112356) that offers state of the art GPUs and other novel accelerators in a composable environment. The ACES innovative composable hardware platform helps accelerate transformative changes in research areas that can leverage novel accelerators for analytics and computing. The test bed enables researchers to creatively develop new programming models and workflows that utilize these architectures while simultaneously advancing HPC (High Performance Computing) and data science projects.

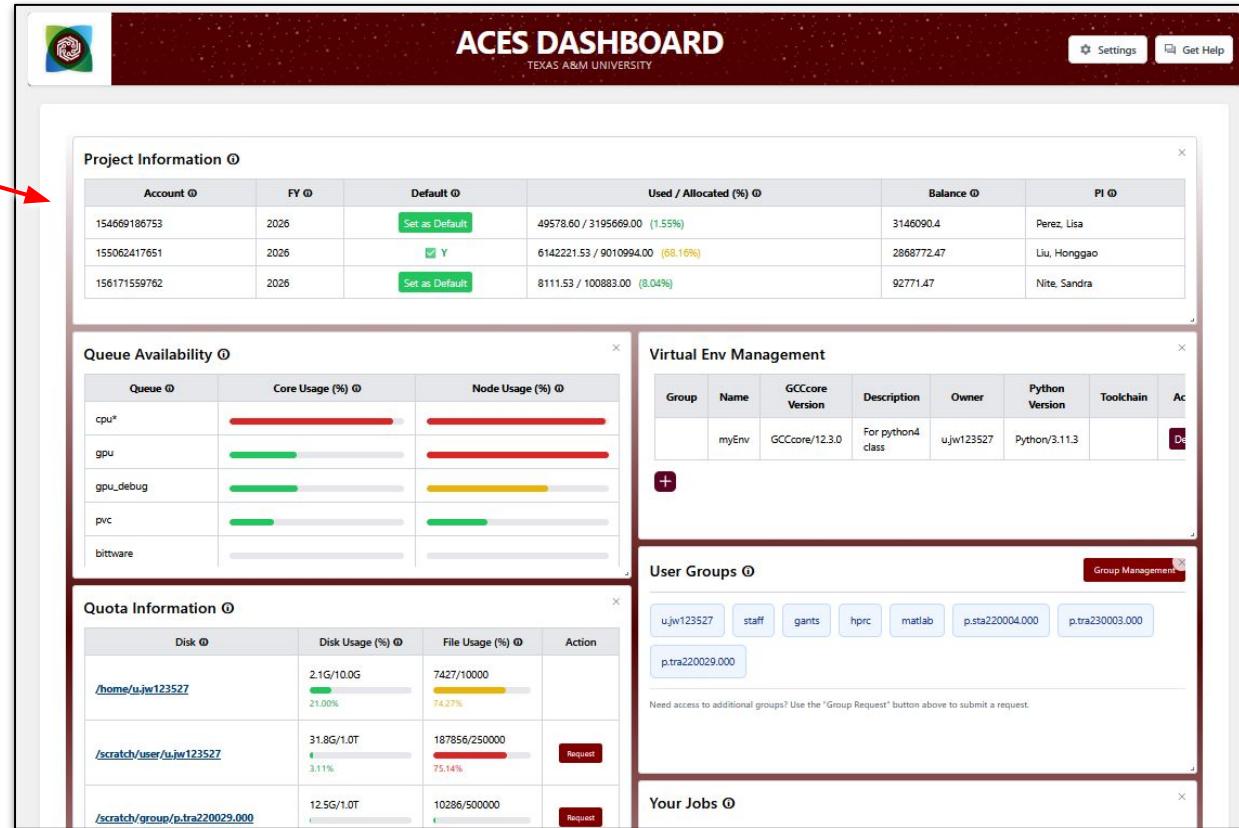
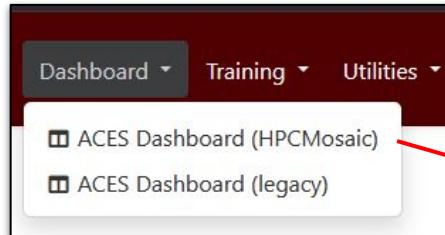
Members get updates about announcements, events, and outages.

JOIN SLACK

Upcoming Events
2/13/2024 11:00 AM EST



ACES Dashboard



The ACES Dashboard is a web-based interface for managing high-performance computing resources. It includes the following sections:

- Project Information**: Shows account details, usage, and balance for three accounts: 154669186753, 155062417651, and 156171559762.
- Queue Availability**: Displays usage percentages for various queues: cpu*, gpu, gpu_debug, pvc, and bittware.
- Virtual Env Management**: Lists virtual environments with details like Group, Name, and Description.
- Quota Information**: Shows disk and file usage for paths like /home/u.jw123527, /scratch/user/u.jw123527, and /scratch/group/p.tra220029.000.
- User Groups**: Lists user groups and provides a "Group Management" button for requesting additional access.
- Your Jobs**: A section for managing and monitoring user jobs.

Utilities

The screenshot shows the ACES OnDemand Portal interface. At the top, there is a navigation bar with 'Affinity Groups', 'Dashboard', and 'Utilities' (which is highlighted with a yellow box). Below the navigation bar, there is a sidebar with links: 'cpuavail', 'envsavail', 'gpuavail', 'license_status', and 'venvavail'. The main content area has a title 'ACES GPU Availability' and a sub-section 'ACES License Status'. The 'ACES License Status' section displays the output of the command '\$ license_status -a', which lists various software licenses and their status. The table shows the following data:

License Name	# Issued	# In Use	# Available
abaqus_teaching	320	0	320
abaqus_extended_teaching	40	0	40
cse_teaching	1	0	1
euler_lagrange_teaching	1	0	1

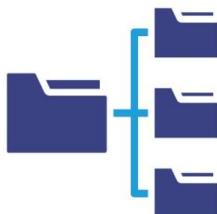
- Convenient collection of command-line tools
- Check availability of gpus, licenses, and other resources

Portal and Cluster Basics

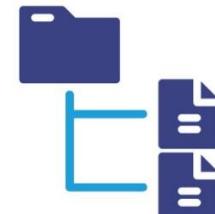
- Portal Walkthrough
- Quotas and SUs

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. Backed up.
/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. Not backed up.
/scratch/group/PROJECTID	\$PROJECT	5 TB	500,000	High performance storage for specific storage allocation requests. Not purged while allocation is active.



Do not share your Home or Scratch areas!
Use Project to collaborate and share files!



Command-line Tools: Quota Usage

Check your file and storage use with the “showquota” command:

```
[username@aces ~] $ showquota

Your current disk quotas are:
Disk           Disk Usage     Limit   File Usage   Limit
/home/username      1.4G      10.0G      3661      10000
/scratch/user/username  117.6G     1.0T      24226    250000
/scratch/group/projectid  510.5G     5.0T     128523    500000
```

Service Units

- Service Units (SUs) are part of our account management system (AMS).
 - **You spend SUs to run jobs on compute nodes**
- 1 SU ~ 1 core-hour on a CPU ... more if using accelerators or extra memory
- Number of SUs you start with depends on which ACCESS allocation you got
- SUs are charged as your jobs spend time on the compute nodes.
(we will see how later).
(Work on login nodes is not charged, but you cannot do big computing there!)
- You can check your SU balance on both:
 - The command line
 - The HPRC Portal

<https://hprc.tamu.edu/kb/User-Guides/AMS/#service-unit>

<https://hprc.tamu.edu/kb/User-Guides/AMS/#ams-user-interfaces>

<https://hprc.tamu.edu/policies/allocations.html>

Command-line Tools : Allocation Usage

Check your allocation balances with the “myproject” command:

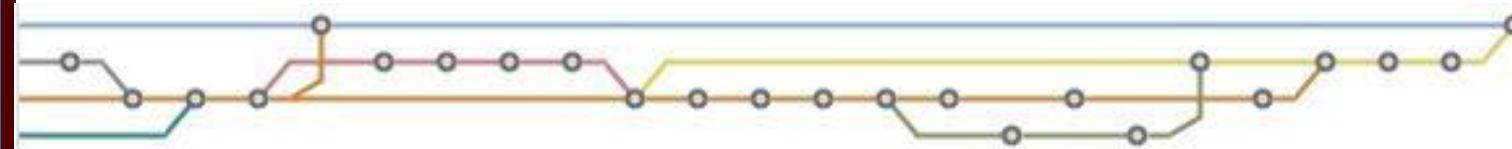
```
[username@aces ~]$ myproject
=====
List of user's Project Accounts
-----
| Account      | Default | Allocation | Used & Pending SUs | Balance | PI |
-----
| 1228000223136 | N | 150000.00 | 0.00 | 10000.00 | Last, First |
-----
| 1428000243716 | Y | 20000.00 | 0.00 | 20000.00 | Last, First |
-----
| 1258000247058 | N | 5000.00 | 0.00 | 5000.00 | Last, First |
```

(Run with the `-h` flag for more information.)

ACES Charging Scheme

Resource(s)	SUs (per hour)	ACCESS Credits (per hour)
Intel SPR / Icelake	1	0.1
Intel PVC GPUs	30	3.75
Bittware Agilex FPGA	30	3.75
Intel Optane Memory	30	3.75
Graphcore IPU Classic	45	5.625
NextSilicon Co-processor	50	6.25
Graphcore IPU Bow	60	7.5
NVIDIA A30	64	8
NEC Vector Engine	75	9.375
NVIDIA H100	128	16

Software Infrastructure



Getting Software

You need software! How to get it?

1. Check to see if we have it installed as a 'module' already.
2. If not, see about getting it installed:
 - a. Install for yourself in your \$SCRATCH.
 - b. Ask us to install system-wide as a module.



Software Modules

SOFTWARE MODULES ON THE ACES CLUSTER

[ACES Software Modules](#) [FASTER Software Modules](#) [Grace Software Modules](#)

(Last Updated: Jul 2, 2025)

Name	Versions available	Description
ABYSS	ABYSS/2.3.7	Assembly By Short Sequences - a de novo, parallel, paired-end sequence assembler
ACTC	ACTC/1.1	ACTC converts independent triangles into triangle strips or fans.
ADIOS2	ADIOS2/2.10.2	The Adaptable Input/Output (I/O) System transports data as groups of self-describing variables and attributes across different media types (such as files, wide-area-networks, and remote direct memory access) using a common application programming interface for all transport modes.
AI-TOOLS-GPU	AI-Tools-GPU/20240816	AI Tools from Intel (formerly referred to as the Intel AI Analytics Toolkit) give data scientists, AI developers, and researchers familiar Python tools and frameworks to accelerate end-to-end data science and analytics pipelines on Intel architecture. The components are built using oneAPI libraries for low-level compute optimizations. The AI Tools maximize performance from preprocessing through machine learning, and provides interoperability for efficient model development.
ANICALCULATOR	ANICalculator/1.0	This tool will calculate the bidirectional average nucleotide identity (gANI) and Alignment Fraction (AF) between two genomes. Required input is the full path to the fna file (nucleotide sequence of genes in fasta format) of each query genome. Either the rRNA and tRNA genes can be excluded, or provided in a list with the -ignoreList option. This is necessary as the presence of tRNA and/or rRNA genes in the fna will artificially inflate the ANI.
ANTLR	ANTLR/4.7.7-Linux-14	ANTLR, ANOther Tool for Language Recognition, (formerly PCCTS) is a language tool that provides a framework for constructing recognizers, compilers, and translators from context-free grammars, including support for standard semantics. It is a Java-based tool, but can be used with other languages.
AMD OPT	AMD OPT/2.0.0	AMD Optimized C/C++ & Fortran compilers (AMD OPT) are compilers that are optimized for AMD processors. They are designed to take advantage of the unique features of AMD processors, such as their SIMD architecture, to produce highly optimized code.
APBS	APBS/2.2.0	APBS (Adaptive Poisson-Boltzmann Solver) is a software package for calculating the electrostatic potential and energy of molecules in aqueous solution. It is based on the Poisson-Boltzmann equation and uses a fast multipole method to solve the resulting system of equations.

940 unique software packages –
most with multiple version
modules (over 3,040 total)

hprc.tamu.edu/software/aces

See also:

- hprc.tamu.edu/kb/Software
- hprc.tamu.edu/software

Computing Environment

Managing software versions using Lmod and Easybuild

- Uses the command: **module**
- Each version of a software, application, library, etc. is available as a module.
 - Module names have the format:

toolchain-name / version toolchain-name / version

GCC/14.2.0

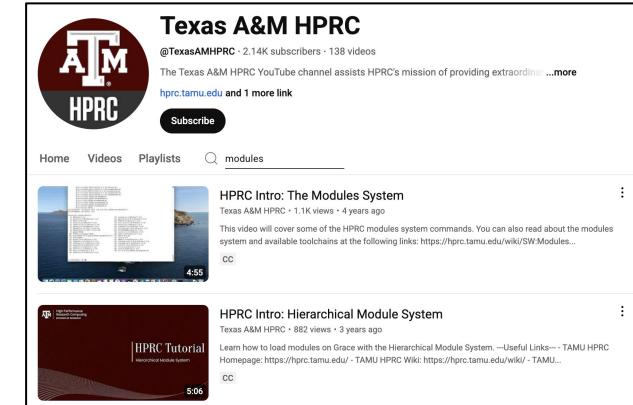
OpenMPI/5.0.7

software-name / version-spec

PyTorch/2.7.0-CUDA-12.6.0

- **module** sets the correct environment variables for you

hprc.tamu.edu/kb/Software/useful-tools/Modules/



Module Usage Basics

module avail

- Lists all available modules (may be slow).
- Navigation:
 - spacebar, arrows, or `j` and `k`
 - quit with `q`
- Case-sensitive search: `/`
- Use `m1a` instead to save results to a file as well (will be named `module.avail.aces` or similar)

module spider <word>

- Case-sensitive search for modules with “word” in name.
- Provide an exact name to see dependencies.

```
[u.jw123527@aces-login2 ~]$ module spider Python
```

```
Python:
```

```
Description:
  Python is a programming language that lets you work more quickly and integrate your systems more effectively.

Versions:
  Python/2.7.18-bare
  Python/2.7.18
  Python/3.8.6
  Python/3.9.5-bare
  Python/3.9.5
  Python/3.9.6-bare
  Python/3.9.6
  Python/3.10.4-bare
  Python/3.10.4
  Python/3.10.8-bare
  Python/3.10.8
  Python/3.11.3
  Python/3.11.5

Other possible module matches:
  Biopython  Boost.Python  Brotli-python  IPython  LASSO-Python  Python-bundle-PyPI  flatbuffers-python  graphviz
```

```
To find other possible module matches execute:
$ module -r spider '.*Python.*'
```

Module Usage Basics

module list

- See what modules are loaded in your current session

module load <module>

- add <module> paths to the current environment variables

module purge

- Unload all modules

```
[u.jw123527@aces-login2 ~]$ module load GCCcore/13.2.0
[u.jw123527@aces-login2 ~]$ module list
```

```
Currently Loaded Modules:
 1) GCCcore/13.2.0
```

```
[u.jw123527@aces-login2 ~]$ module purge
[u.jw123527@aces-login2 ~]$ module list
No modules loaded
[u.jw123527@aces-login2 ~]$
```

There is also a shorthand:

```
ml
ml <module>
ml purge
```

Hands-on Exercise: Module Loading

1. `mla blast+`
-See which versions of BLAST+ are available.
2. `ml BLAST+/2.14.1`
-error! You cannot do that yet.
3. `module spider BLAST+/2.14.1`
-Learn how to load this module.
4. `ml [REDACTED] BLAST+/2.14.1`
-Fill in the blank (with the correct toolchains) to load this module.
5. `ml list`
-List all loaded modules.
6. `ml GCC/10.2.0`
-Change version of a loaded Toolchain module (GCC); notice the message about reloaded modules.
7. `ml list`
-List all loaded modules.
8. `ml purge`
-Remove all loaded modules.

Installing Software

- Researchers can install software in their own directories.
 - Exact steps depend on the software.
 - You **CANNOT** run the "sudo" command when installing software.
 - Watch your file quotas! Install in **\$SCRATCH**!
- Contact us if you need help
 - We can install software *cluster-wide*.
 - Requests can be sent from the Dashboard.
- License-restricted software
 - Check on command line with `license_status`
 - Contact help@hprc.tamu.edu

Python and Virtual Environments

Python is a language which supports many external libraries in the form of extensions. (called Python Packages).

Some commonly used packages:

- SciPy & NumPy
- Jupyter notebook
- Scikit-learn

Once you have loaded the appropriate Python module, you can install these additional packages yourself using the *Virtual Environment* feature.

Instructions are on our KB:

- <https://hprc.tamu.edu/kb/Software/Python/#hprc-venv-management-tools>
- <https://hprc.tamu.edu/kb/Software/ModuLair/> (special HPRC tool!)

Hands-on Exercise: Python Software Install

1.

```
ml purge
create_venv myEnv -d "Example Env" -t "GCCcore/14.3.0 Python/3.13.5"
```

Start from a clean environment and use 'create_venv'
The **-d** arg is optional Use the **-t** arg if you haven't loaded modules already
2.

```
source activate_venv myEnv
```

-Activate virtual environment.
3.

```
python -c "import pytime"
```

-Check if python-time is installed (it is not).
4.

```
pip install python-time
```

-Install python-time.
5.

```
python -c "import pytime; print(pytime)"
```

-Where is python-time installed?
6.

```
deactivate
```

-Close virtual environment.

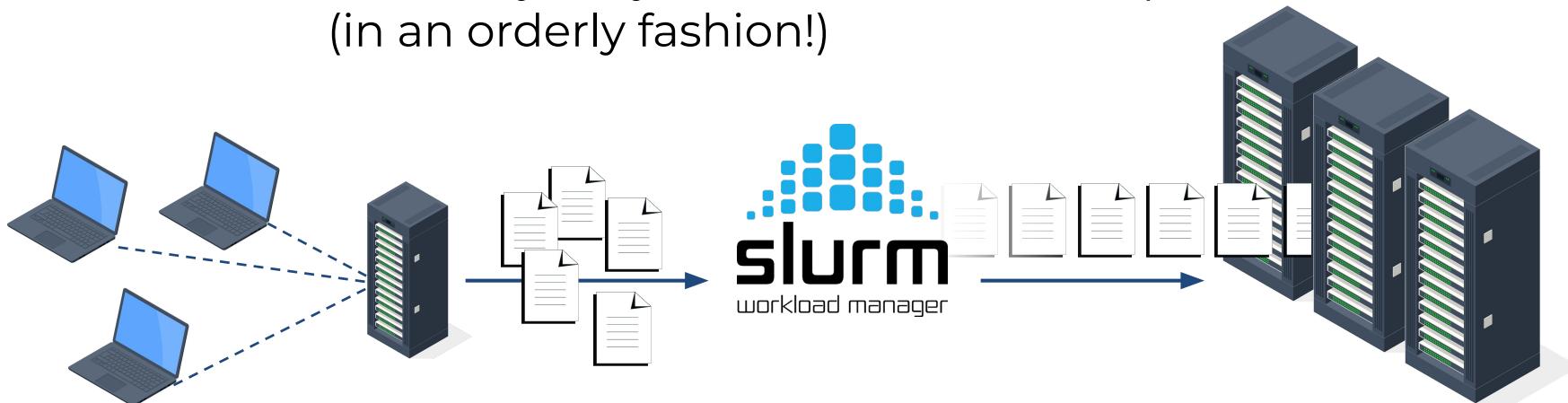
Batch Computing

- Command Line
- Job Management Tools

Batch Computing on HPRC Clusters: Overview

Since the cluster is a shared resource, jobs have to be moderated somehow so everyone has a turn. In short:

1. Write instructions on the login node
2. Give the instructions to Slurm
3. Slurm sends everybody's instructions to compute nodes
(in an orderly fashion!)



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/13.3.0
module load Python/3.12.3
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 comments to the shell, but NOT to Slurm!
- 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.

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 $USER
```

or

```
squeue --me
```

JOBID	NAME	USER	PARTITION	NODES	CPUS	STATE	TIME	TIME_LEFT	START_TIME	REASON	NODELIST
6853258	jobname	someuser	cpu	2	96	RUNNING	3-07:36:50	16:23:10	2025-01-23T17:27:3	None	ac[180,202]
6853257	jobname	someuser	cpu	2	96	RUNNING	3-07:36:56	16:23:04	2025-01-23T17:27:2	None	ac[523-524]

Hands-on Exercise: Job Submission

1. `cp -r /scratch/training/slurm_example/ $SCRATCH` -Copy example files to \$SCRATCH
2. `cd $SCRATCH/slurm_example` -Go to the example files
3. `vi hello_world.job` -View/edit job file (optional)
4. `sbatch hello_world.job` -Submit job
5. `squeue --me` -Check job
6. `cat <output file name>` -Check output when done

Checking Resources for Jobs

`sinfo`

`pestat`

`maxconfig`

There are several command-line tools available to check what you can use in your jobs...

Checking Resources for Jobs: Queues

sinfo

pestat

maxconfig

PARTITION	AVAIL	TIMELIMIT	JOB_SIZE	NODES (A/I/O/T)	CPUS (A/I/O/T)
cpu*	up	3-00:00:00	1-64	69/1/3/73	5960/760/288/7008
gpu	up	2-00:00:00	1-8	6/0/1/7	211/365/96/672
gpu_debug	up	2:00:00	1	1/1/1/3	16/176/96/288
pvc	up	2-00:00:00	1-30	15/13/2/30	892/1796/192/2880
bittware	up	2-00:00:00	1	0/2/0/2	0/192/0/192
nextsilicon	up	2-00:00:00	1	1/1/0/2	51/141/0/192
nec	up	2-00:00:00	1	0/1/0/1	0/48/0/48
staff	up	2-00:00:00	1-110	86/18/6/110	6696/3288/576/10560

Shows job queue status

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

Checking Resources for Jobs: Nodes

sinfo

pestat

maxconfig

```
[u.jw123527@aces-login2 ~]$ pestat -p gpu -G
Print only nodes in partition gpu
GPU GRES (Generic Resource) is printed after each JobID
Hostname      Partition      Node  Num_CPU  CPUload (15min)  Memsize  Freemem  GRES/node      Joblist
                           State Use/Tot      (MB)          (MB)          JobID User GRES/job ...
ac041          gpu           mix   64   96  29.40*    500000  193421  gpu:h100:8(S:0)  220099 u.xw127610 gpu:h100=8 220026 u.xw127610 gpu:h100=8 *
ac045          gpu           mix   64   96  14.21*    500000  256566  gpu:h100:8(S:0)  220099 u.xw127610 gpu:h100=8 239377 u.ch204012 gpu:h100=2 239432 u.ch204012 gpu:h100=2 *
ac049          gpu           mix   32   96  14.81*    500000  260781  gpu:h100:4(S:0)  220001 u.xw127610 gpu:h100=8 *
ac055          gpu           mix   32   96  14.78*    500000  260220  gpu:h100:4(S:0)  220001 u.xw127610 gpu:h100=8 *
ac064          gpu           down* 0   96   0.00    500000  511159  gpu:a30:2(S:0)
ac065          gpu           down* 0   96   0.00    500000  511187  gpu:a30:2(S:0)
ac096          gpu           mix   32   96  14.58*    500000  256768  gpu:h100:4(S:0)  220026 u.xw127610 gpu:h100=8 *
[u.jw123527@aces-login2 ~]$
```

Shows status of nodes

Notable options:

- -p <partition name> show only a specific partition
- -u \$USER show only specified user's jobs
- -G show "generic resources" (e.g., the gpus used)

Checking Resources for Jobs: SUs

`sinfo`

`pestat`

`maxconfig`

If you ask for too many resources, your job will not run. You can check beforehand:

```
[u.ab12345@aces-login2 ~]$ maxconfig
```

```
ACES partitions:  cpu  gpu  gpu_debug  pvc  bittware  memverge  nextsilicon
ACES GPUs in gpu partition:  a30:2  h100:2  h100:4  h100:8  pvc:2  pvc:4  pvc:6  pvc:8
```

```
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      # max 64 nodes for partition c
#SBATCH --ntasks-per-node=1
#SBATCH --cpus-per-task=96
#SBATCH --mem=488G
#SBATCH --output=stdout.%x.%j
#SBATCH --error=stderr.%x.%j
```

Check specific partitions with:

```
maxconfig -p <partitionName>
```

Estimate the SUs a job will require with:

```
maxconfig -f <jobScriptName>
```

Job Summary: myjob

Command-line tool to show you details of a specific job:

myjob

Shows, e.g.:

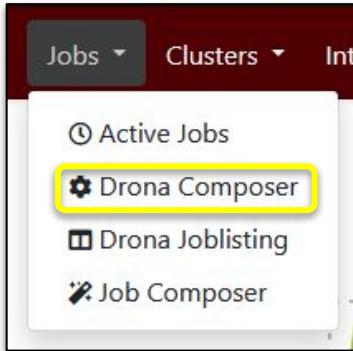
- Submission details
- Resource usage of finished jobs
- Status of pending job

```
>$ myjob 1234701

Job ID: 1234701
        Cluster: aces
        User/Group: u.ab123456/u.ab123456
        Account: 123455963789
        State: COMPLETED (exit code 0)
        Partition: gpu
        Node Count: 1
        NodeList: ac098
        Cores per node: 48
        CPU Utilized: 11:25:16
        CPU Efficiency: 2.68% of 17-18:31:12 core-walltime
        Submit time: 2025-07-20 21:37:13
        Start time: 2025-07-21 02:42:10
        End time: 2025-07-21 11:35:19
        Job Wall-clock time: 08:53:09
        Memory Utilized: 6.41 GB
        Memory Efficiency: 2.63% of 244.00 GB (244.00 GB/node)
        Job Name: GPU_Analysis
        Job Submit Directory: /scratch/user/u.ab123456/job_scripts
        Submit Line: sbatch
        ratpose_video_analysis_job.slurm
```

Batch Computing

- Command Line
- Job Management Tools



Portal: Drona Composer GUI

Import additional environments

Select customizable environments or workflows

History of previously submitted jobs

Rerun jobs

High Performance Research Computing
Division of Research

DRONA COMPOSER (ACES)

Job Name:
Location: /scratch/user/u.jw123527/drona_composer/runs
Environments:

Job Submission History
From To

ID	Name	Location	Environment	Date	Actions
462015...	tutorial-job	u.jw123527/drona_composer/run...	IPUTutorial	2025-08-04 10:53:16	<input type="button" value="Actions ▾"/>
616049...	tutorial-job	u.jw123527/drona_composer/run...	IPUTutorial	2025-08-04 10:52:41	<input type="button" value="Actions ▾"/>

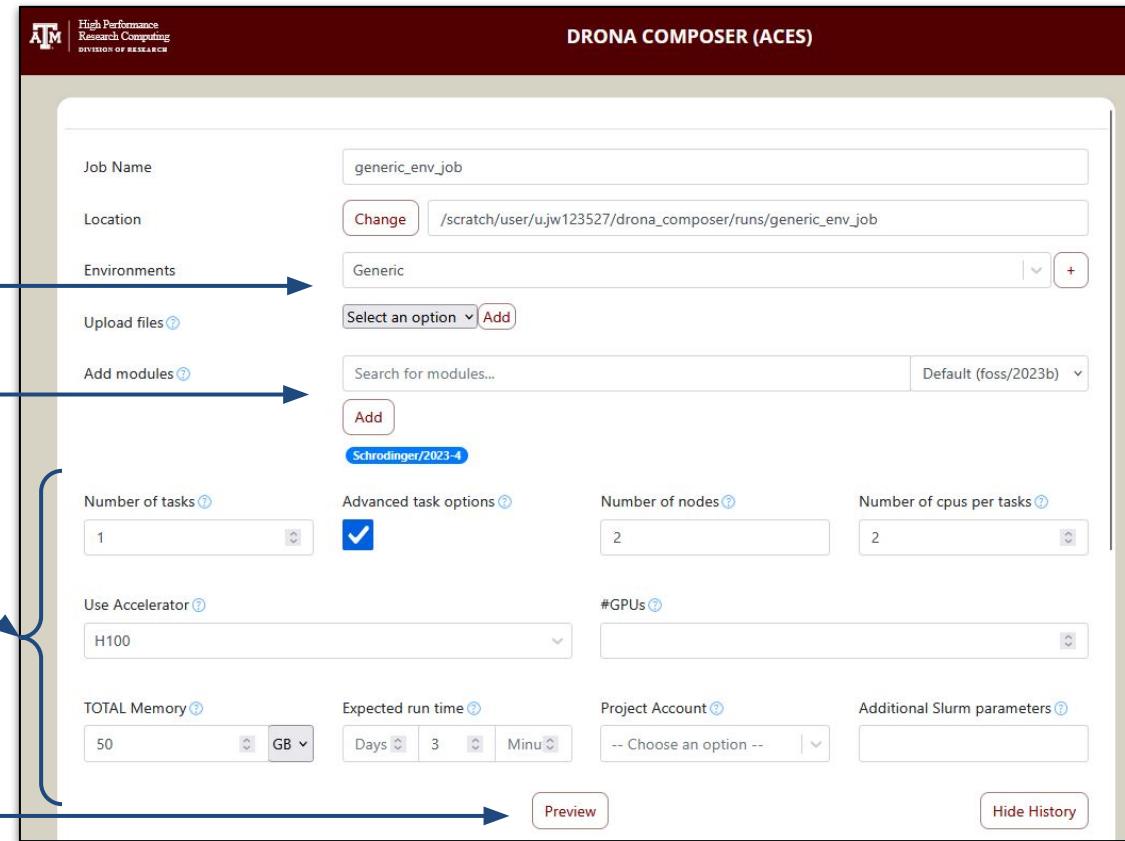
Portal: Drona Composer GUI

Generic environment to create custom Slurm batch job

Add modules for job

Fill out info that Slurm would need

Hit 'Preview'



DRONA COMPOSER (ACES)

Job Name: generic_env_job

Location: /scratch/user/u.jw123527/drona_composer/runs/generic_env_job

Environments: Generic

Upload files: Select an option

Add modules: Search for modules... Default (foss/2023b)

Number of tasks: 1 Advanced task options:

Number of nodes: 2 Number of cpus per tasks: 2

Use Accelerator: H100 #GPUs:

TOTAL Memory: 50 GB Expected run time: 3 Days 0 Minu 0

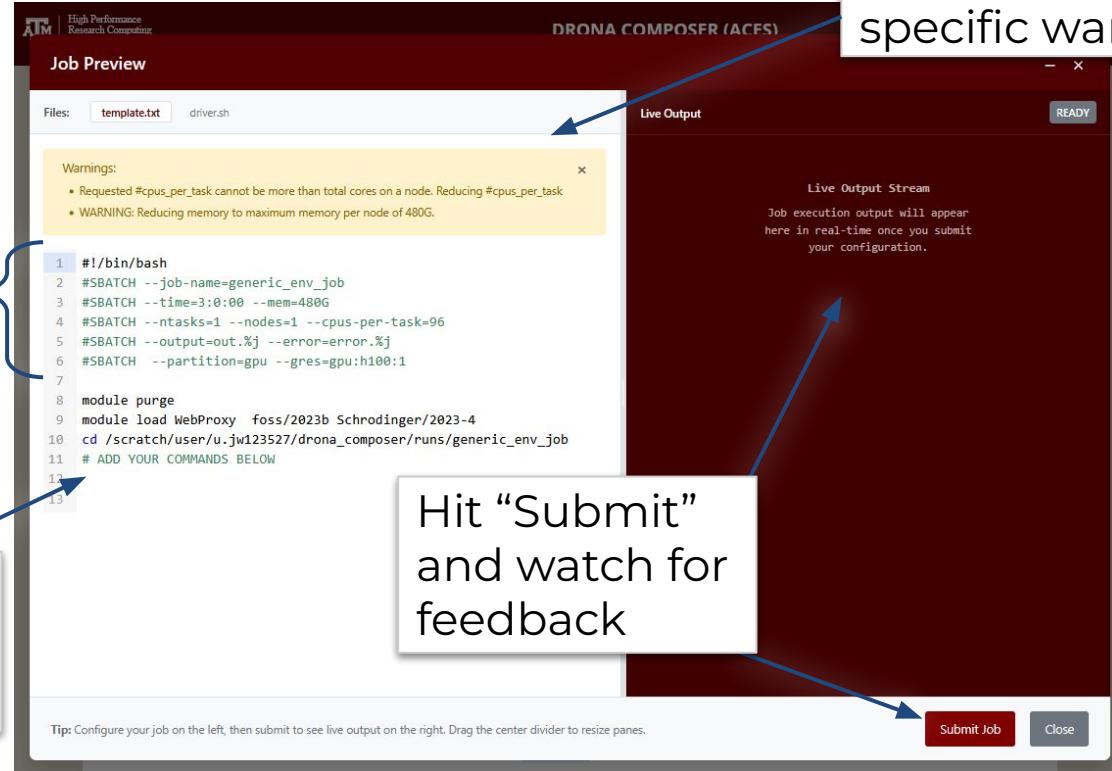
Project Account: -- Choose an option -- Additional Slurm parameters:

Hide History

Portal: Drona Composer GUI

Generated Slurm
directives and
module loads

Add commands
directly into editable
window



TAMULauncher

- A more robust and cluster-efficient way to submit “array” jobs

`tamulauncher commands_file.txt`

- Use when you have hundreds or thousands of commands to run, each utilizing a single-core or a few cores.
 - tamulauncher keeps track of which commands completed successfully and can pick up where it left off
 - Can be run in a batch script or interactively (for <8 login node cores)
 - You can check the --status on the command line from the working directory.
- More information: <https://hprc.tamu.edu/kb/Software/tamulauncher>
(See also our Slurm course this afternoon!)

TAMULauncher Multi-Node Single-Core Commands

[commands.txt](#)

(500 lines for example)

```
spades.py -1 s1_R1.fastq.gz -2 s1_R2.fastq.gz -o s1_out --threads 1
spades.py -1 s2_R1.fastq.gz -2 s2_R2.fastq.gz -o s2_out --threads 1
spades.py -1 s3_R1.fastq.gz -2 s3_R2.fastq.gz -o s3_out --threads 1
spades.py -1 s4_R1.fastq.gz -2 s4_R2.fastq.gz -o s4_out --threads 1
spades.py -1 s5_R1.fastq.gz -2 s5_R2.fastq.gz -o s5_out --threads 1
spades.py -1 s6_R1.fastq.gz -2 s6_R2.fastq.gz -o s6_out --threads 1
spades.py -1 s7_R1.fastq.gz -2 s7_R2.fastq.gz -o s7_out --threads 1
spades.py -1 s8_R1.fastq.gz -2 s8_R2.fastq.gz -o s8_out --threads 1
spades.py -1 s9_R1.fastq.gz -2 s9_R2.fastq.gz -o s9_out --threads 1
spades.py -1 s10_R1.fastq.gz -2 s10_R2.fastq.gz -o s10_out --threads 1
spades.py -1 s11_R1.fastq.gz -2 s11_R2.fastq.gz -o s11_out --threads 1
spades.py -1 s12_R1.fastq.gz -2 s12_R2.fastq.gz -o s12_out --threads 1
spades.py -1 s13_R1.fastq.gz -2 s13_R2.fastq.gz -o s13_out --threads 1
spades.py -1 s14_R1.fastq.gz -2 s14_R2.fastq.gz -o s14_out --threads 1
spades.py -1 s15_R1.fastq.gz -2 s15_R2.fastq.gz -o s15_out --threads 1
spades.py -1 s16_R1.fastq.gz -2 s16_R2.fastq.gz -o s16_out --threads 1
spades.py -1 s17_R1.fastq.gz -2 s17_R2.fastq.gz -o s17_out --threads 1
spades.py -1 s18_R1.fastq.gz -2 s18_R2.fastq.gz -o s18_out --threads 1
spades.py -1 s19_R1.fastq.gz -2 s19_R2.fastq.gz -o s19_out --threads 1
spades.py -1 s20_R1.fastq.gz -2 s20_R2.fastq.gz -o s20_out --threads 1
spades.py -1 s21_R1.fastq.gz -2 s21_R2.fastq.gz -o s21_out --threads 1
spades.py -1 s22_R1.fastq.gz -2 s22_R2.fastq.gz -o s22_out --threads 1
```

[run_spades_tamulauncher.sh](#)

```
#!/bin/bash
#SBATCH --job-name=spades
#SBATCH --time=1-00:00:00
#SBATCH --nodes=2
#SBATCH --ntasks-per-node=96
#SBATCH --cpus-per-task=1
#SBATCH --mem=488G
#SBATCH --output=stdout.%x.%j
#SBATCH --error=stderr.%x.%j

module purge
module load GCC/11.3.0  SPAdes/3.15.5

tamulauncher commands.txt
```

Run 96 spades.py commands per node with each command using 1 core.
Requesting all 96 cores on ACES reserves entire node for your job.

- Run 96 single-core commands per node; useful when each command requires <= 5GB memory.
- Create a commands file (named whatever you want) to go with the the job script.
- The commands.txt file will contain one command per line.
- Load the software module in the job script not the commands file.

Data Transfers Using Globus

<https://hprc.tamu.edu/kb/Software/Globus/>
<https://app.globus.org>

Transfer Settings	<ul style="list-style-type: none">• verify file integrity after transfer• transfer encrypted• overwriting all files on destination
Destination	/scratch/user/ username /supersim_transfer/

Composability, Accelerators, and AI/ML

Status of Composability

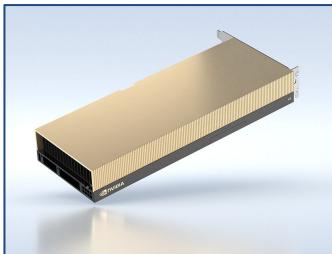
Composability is currently handled by sysadmins.

- Contact help@hprc.tamu.edu to request a given composable configuration for a cluster.
- After the composed node has been created, simply specify the relevant resources in your Slurm job file.
- In-progress: making Slurm and Liquid handle it automatically
- Note: The IPUs and NEC Vector Engines cannot be composed.

Accelerator Access

Component	Access	node or partition
NVIDIA A30 GPUs and H100 GPUs	Slurm	--partition=gpu --partition=gpu_debug
Intel GPU Max 1100 (PVC)	Slurm	--partition=pvc
BittWare IA-840F FPGA	Slurm	--partition=bittware
Intel Optane SSD	Slurm	--partition=memverge
NextSilicon Coprocessor	Slurm	--partition=nextsilicon
NEC Vector Engine	Slurm	--partition=nec
Graphcore Bow IPUs	Interactive	ssh poplar2
Graphcore Colossus IPUs	Interactive	ssh poplar1
Kubernetes & Development Cluster	Interactive	upon request (pilot)

NVIDIA GPUS on ACES



A30s: Support less intense workloads relying on numerical simulations and AI/ML methods.



H100s: Supports computationally intensive workloads employing numerical simulations and AI/ML methods.

Specify in Slurm file:

```
#SBATCH --partition=gpu  
#SBATCH --gres=gpu:<gpu type>:<number>
```

Intel Data Center Max GPU 1100 GPUs (PVC GPUs)



We are planning to have a PVC training event in March!

See also our YouTube channel for previous PVC training sessions!

Intel GPUs for HPC, DL Training, AI Inference

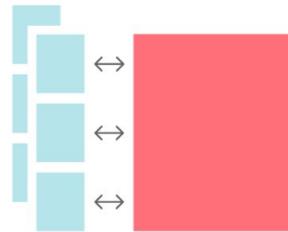
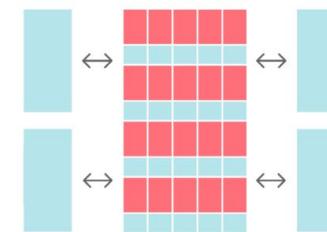
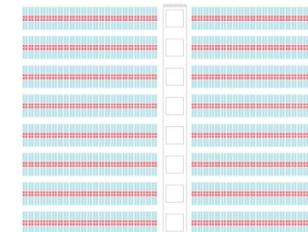
Specify in Slurm file:

```
#SBATCH --partition=pvc
#SBATCH --gres=gpu:pvc:<number>
```

See also our command-line tool to check configuration:

```
$ show_pvc_features
HOSTNAME  AVAIL_FEATURES          GRES      STATE
ac010      gen4_fabric           gpu:pvc:4  mixed
...
ac024      gen4_fabric           gpu:pvc:8  idle
ac025      gen4_fabric           gpu:pvc:4  mixed
ac026      gen5_fabric           gpu:pvc:6  reserved
ac030      gen5_fabric           gpu:pvc:8  reserved
ac081      gen5_fabric,xelink4  gpu:pvc:4  reserved
ac082      gen5_fabric,xelink2  gpu:pvc:2  reserved
...
```

Graphcore IPUs

	CPU	GPU	IPU
Parallelism	Designed for scalar processes	SIMD/SIMT architecture. Designed for large blocks of dense contiguous data	Massively parallel MIMD. Designed for fine-grained, high-performance computing
Processors			
Memory Access	Off-chip memory	Model and data spread across off-chip and small on-chip cache, and shared memory	Model and data tightly coupled, and large locally distributed SRAM

www.graphcore.ai/bow-processors

further documentation: docs.graphcore.ai/en/latest

Accessing Graphcore IPUs

- SSH into poplar1 or poplar2 from ACES
 - `[username@login ~]$ ssh poplar2`
- Enable the SDK environment. See our KB for details:
 - hprc.tamu.edu/kb/User-Guides/ACES/Graphcore_Colossus_IPUs/
 - hprc.tamu.edu/kb/User-Guides/ACES/Graphcore_Bow_IPUs/
- Type **gc-monitor** to view the status of the IPUs:

Contact us first to be given access to poplar.

```
mouse@poplar1:~$ gc-monitor
```

Partition: p16 [active] has 16 reconfigurable IPUs								
IPU-M	Serial	IPU-M SW	Server version	ICU FW	Type	ID	IPU#	Routing
10.1.5.1	0010.0002.8213921		1.9.0	2.4.4	M2000	0	3	DNC
10.1.5.1	0010.0002.8213921		1.9.0	2.4.4	M2000	1	2	DNC
10.1.5.1	0010.0001.8213921		1.9.0	2.4.4	M2000	2	1	DNC
10.1.5.1	0010.0001.8213921		1.9.0	2.4.4	M2000	3	0	DNC
10.1.5.2	0030.0002.8213921		1.9.0	2.4.4	M2000	4	3	DNC
10.1.5.2	0030.0002.8213921		1.9.0	2.4.4	M2000	5	2	DNC
10.1.5.2	0030.0001.8213921		1.9.0	2.4.4	M2000	6	1	DNC

We are planning to have an IPU training event in March!

See also our Youtube channel for previous IPU training sessions!

NEC Vector Engine

- 300W PCIe Gen3 x16 card
- 8 cores, 48 GB HBM2 with 1.5 TB/s memory bandwidth
- 2.45 TFLOPS FP64 peak performance
- Eight (8) NEC Vector Engine Type 20B-P cards hosted in a Dell DSS8440 server
- Links to further documentation at hprc.tamu.edu/kb/User-Guides/ACES/vectorengine/
- Slurm access: `--partition=nec`



VE training event is currently planned for April!

See also our Youtube channel for previous training sessions!

NextSilicon Coprocessors



- 300W PCIe Gen5 x8 card (i.e. 300W per card)
- 64 GB HBM2e memory,
1.6 TB/s memory bandwidth
- Maverick-1 coprocessor
- Hardware integer compute units
- 2 TFLOPS FP64 peak performance
- Slurm Access: **--partition=nextsilicon**

BittWare FPGAs



BittWare IA-840F FPGA

Supported Tools:

- Intel FPGA OneAPI
- Intel Quartus Prime

Slurm access: **--partition=bittware**

Composing Memory

MemVerge Memory Machine installed on 8 nodes:

- Base nodes:
 - 96 Intel Xeon 8468 processors
 - ~488 Gb DRAM
- Intel Optane SSDs provide additional memory for large jobs
 - up to 18 TB
- Best for jobs that infrequently access data in memory

Slurm access:

```
#SBATCH --partition=memverge
```

hprc.tamu.edu/kb/User-Guides/ACES/memverge

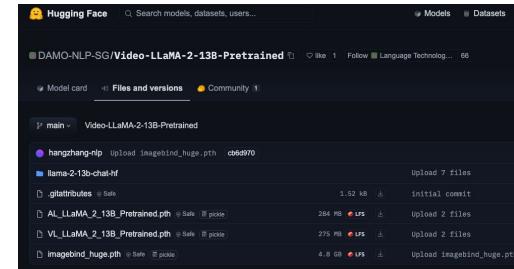


AI Workflows: Hugging Face Hub on ACES

Hugging Face Hub

- huggingface_hub (models, datasets, etc)
- git-lfs (Git Large File Storage)

Using Git LFS (Large File Storage) with the Hugging Face Hub is crucial for handling large files like machine learning models and datasets.



```
[u.zhang108696@aces-login3 ~]$ ml spider git-lfs
-----
git-lfs:
-----
  Description:
    Git Large File Storage (LFS) replaces large files
    while storing the file contents on a remote server

  Versions:
    git-lfs/3.2.0
    git-lfs/3.3.0
    git-lfs/3.5.1
```



huggingface_hub

AI Workflows: Hugging Face Hub on ACES

Hugging Face Libraries

- Transformers
- Datasets
- Tokenizers
- Accelerate
- Diffusers
- ...



```
(hgf-env) [u.zh108696@aces-login3 huggingface]$ pip install transformers datasets tokenizers
Looking in indexes: https://pypi.org/simple, https://pypi.ngc.nvidia.com
Collecting transformers
  Downloading transformers-4.48.0-py3-none-any.whl.metadata (44 kB)
  44.4/44.4 kB 5.7 MB/s eta 0:00:00
Collecting datasets
  Downloading datasets-3.2.0-py3-none-any.whl.metadata (20 kB)
Collecting tokenizers
```

```
from transformers import AutoTokenizer, AutoModelForSequenceClassification

# Choose a model, e.g., BERT for sentiment analysis
model_name = "distilbert-base-uncased-finetuned-sst-2-english"

# Load tokenizer and model
tokenizer = AutoTokenizer.from_pretrained(model_name)
model = AutoModelForSequenceClassification.from_pretrained(model_name)

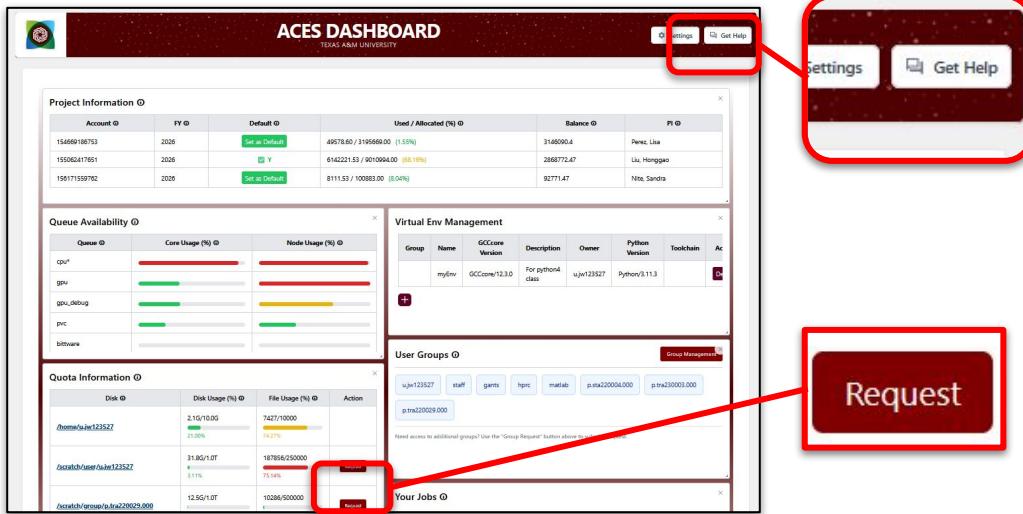
# Tokenize input
inputs = tokenizer("I love ACES!", return_tensors="pt")

# Perform inference
outputs = model(**inputs)
predictions = outputs.logits.argmax(dim=-1)
print(predictions) # 1 for positive sentiment
```

Need Help?

First check the FAQ: <https://hprc.tamu.edu/kb/FAQ/Accounts>

- ACES user Guide: <https://hprc.tamu.edu/kb/User-Guides/ACES>
- FASTER user Guide: <https://hprc.tamu.edu/kb/User-Guides/FASTER>



Remember the
Dashboard!

These buttons have
automations that help
us address your issue
faster!

Need Help? - Email

You can also contact our helpdesk at help@hprc.tamu.edu

Help us help you -- tell us:

- 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!

Questions?

Give us feedback on the class with this survey:
https://u.tamu.edu/hprc_shortcourse_survey

https://u.tamu.edu/hprc_shortcourse_survey



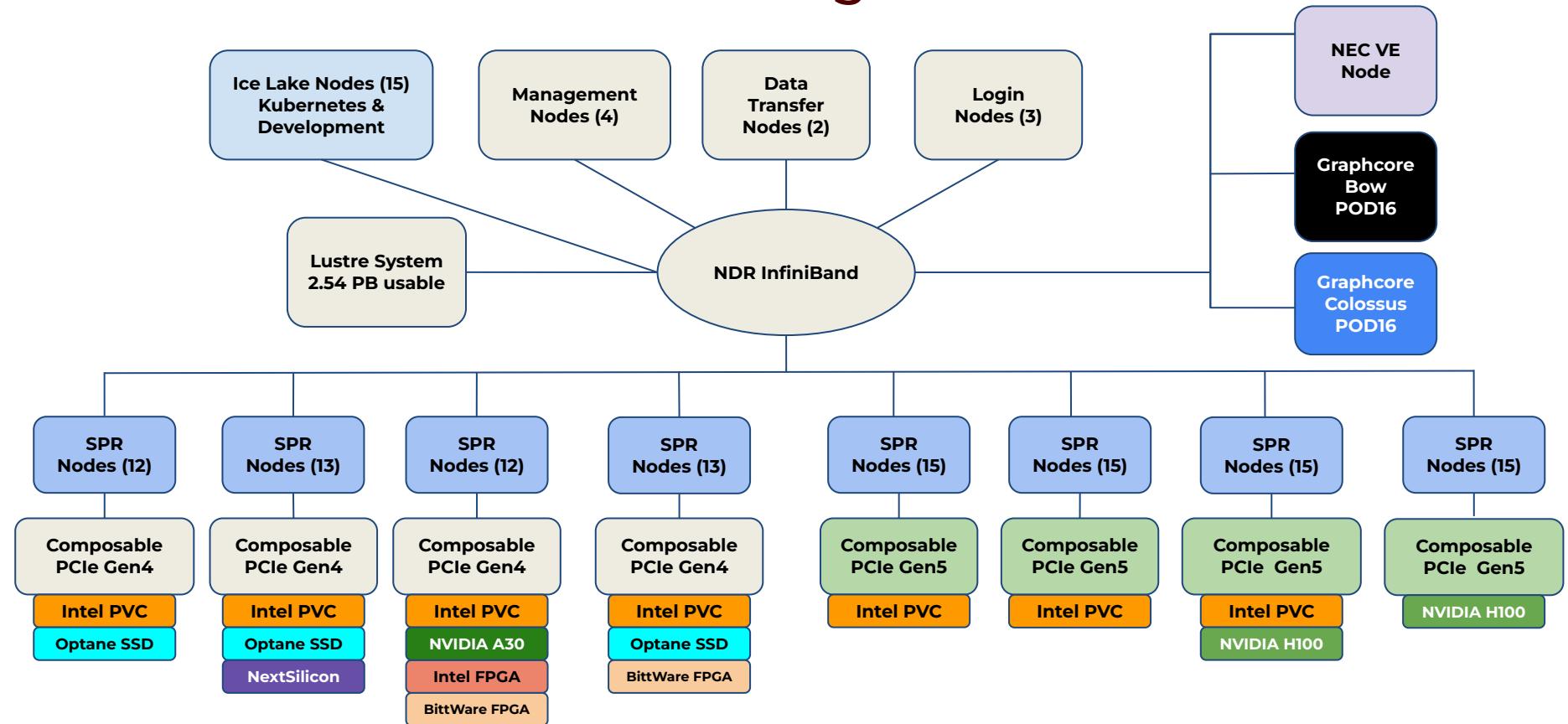
HPRC Survey



High Performance Research Computing | hprc.tamu.edu | NSF Award #2112356

ACES Configuration

ACES Configuration



ACES System Description

Component	Quantity	Description
Sapphire Rapids Nodes: Compute Nodes Data Transfer Nodes Login & Management Nodes	110 nodes 2 nodes 5 nodes	96 cores per node, dual Intel Xeon 8468 processors 512 GB DDR5 memory 1.6 TB NVMe local storage Compute: NVIDIA Mellanox NDR 200 Gbps InfiniBand adapter DTNs & Login & Management nodes: 100 Gbps Ethernet adapter
Ice Lake Login & Management Nodes	2 nodes	64 cores per node, dual Intel Xeon 8352Y processors 512 GB DDR4 memory 1.6 TB NVMe local storage NVIDIA Mellanox NDR 200 Gbps InfiniBand adapter
PCIe Gen4 Composable Infrastructure	50 SPR nodes	Dynamically reconfigurable infrastructure that allows up to 20 PCIe cards (GPU, FPGA, etc.) per compute node
PCIe Gen5 Composable Infrastructure	60 SPR nodes	Dynamically reconfigurable infrastructure that allows up to 16 H100s or 14 PVCs per compute node
NVIDIA InfiniBand (IB) Interconnect	110 nodes	Two leaf and two spine switches in a 2:1 fat tree topology
DDN Lustre Storage	2.5 PB usable	HDR IB connected flash and disk storage for Lustre file systems

ACES Accelerators

Component	Quantity	Description
Graphcore IPU: Colossus GC200 Bow	16 16	Each IPU group hosted with a CPU server as a POD16 on a 100 GbE RoCE fabric
NVIDIA GPUs: H100 A30	30 4	For HPC, DL Training, AI Inference For AI Inference and Mainstream Compute
BittWare IA-840F FPGA	3	Accelerator with Agilex AGF027 FPGA and 64 GB of DDR4
NextSilicon Coprocessor	2	Reconfigurable accelerator with an optimizer continuously evaluating application behavior.
NEC Vector Engine	8	Vector computing card (8 cores and HBM2 memory)
Intel Optane SSD	48	18 TB of SSDs addressable as memory w/ MemVerge Memory Machine.
Intel PVC GPUs	120	Intel GPUs for HPC, DL Training, AI Inference