NREL|HPC

Getting Started on Kestrel

April 2024

Haley Yandt & Olivia Hull

Introduction

Background

- NREL provides HPC in support of the DOE Office of Energy Efficiency and Renewable Energy (EERE)
    - NREL runs yearly open call (June) on behalf of EERE
    - NREL HPC is the principal provider of high performance and data-intensive computing services to the DOE EERE Offices
    - additional call this year (student applications), allocating 5.6M AU (560k node hours)

- **EERE Mission: …**accelerate the research, development, demonstration, and deployment of technologies and solutions to equitably transition America to net-zero greenhouse gas emissions…(https://www.energy.gov/eere/mission)

- **NREL Mission:** NREL strives to achieve our vision of a clean energy future for the world through our mission: leading research, innovation, and strategic partnerships to deliver solutions for a clean energy economy...(https://www.nrel.gov/about/vision.html)

Introduction

Allocation Questions: HPC-Requests@nrel.gov
General HPC Questions: HPC-Help@nrel.gov

Allocations & Accounts

- 'some' allocations activated Apr 22
  - defaulted to 100k AU (10k node hours) for CPU requests
  - contact **HPC-Requests** if additional AUs needed (up to 400k max)
  - remainder to be activated when account process complete
- accounts activated or in progress
  - additional paperwork required in some cases, please respond promptly to requests
  - reach out to **HPC-Help** if you need assistance with your account
- software licenses
  - unable to provide licenses in most cases
  - follow documentation for requesting access on webpage for listed software
  - reach out to HPC-Help for all other licensing issues
  - software docs: https://nrel.github.io/HPC/Documentation/Applications/

Connect to Kestrel

- Connect directly to Kestrel login node

```
$ ssh <username>@kestrel.nrel.gov # Login externally
   …
Password+OTPToken:
```

- Alternatively, you can use the HPC VPN or the SSH gateway.
- Logins nodes serve as a gateway to the rest of the system and are shared resources.

Data Analysis and Visualization

- Kestrel has 8 DAV nodes, intended for HPC applications that require a graphical user interface.
  - One DAV node accessible directly to non-NREL users at **kestrel-dav.nrel.gov**
  - The other 7 are accessible via the HPC VPN
- FastX is available for a graphical remote desktop
- Can connect to DAV nodes using FastX through a web or desktop client
- DAV nodes are not general-purpose remote desktops. They are intended for HPC or visualization software that requires Kestrel.
- DAV nodes are shared resources. Computationally intensive work needs to be done on compute nodes.

System Configuration

- **44** Pflops peak
- Interconnect: HPE Slingshot
- Compute nodes:
    - 2144 Dual socket Intel Sapphire Rapids 52-core processors (**104 cores** total per node)
    - Standard nodes have **256GB RAM**
    - 256 nodes have **1.75 NVMe** local disk
    - **10 bigmem nodes** with **2TB RAM** and **5.5TB NVMe** local disk
- GPU nodes:
    - Coming soon – in final stages of our acceptance testing
    - 132 GPU nodes with 4 NVIDIA H100 GPUs per node
- Additional information: https://www.nrel.gov/hpc/kestrel-system-configuration.html

Kestrel Filesystem

- **95 PB** ClusterStor Lustre Filesystem
- Parallel File System (PFS)
  - Intended for high performance I/O
  - /projects/<allocation handle>
  - /scratch/$USER
  - Run jobs out of /scratch, use /projects for longer term storage and sharing data with allocation team members
  - No quota on /scratch, but subject to purging after 28 days of inactivity
  - There are no backups of PFS data
- Home File System: 1.2PB, part of the ClusterStor system
  - User /home directories have a 50GB quota
  - Shouldn't run jobs out of /home - not tuned for parallel I/O
- Globus is available for large data transfers

Running Jobs

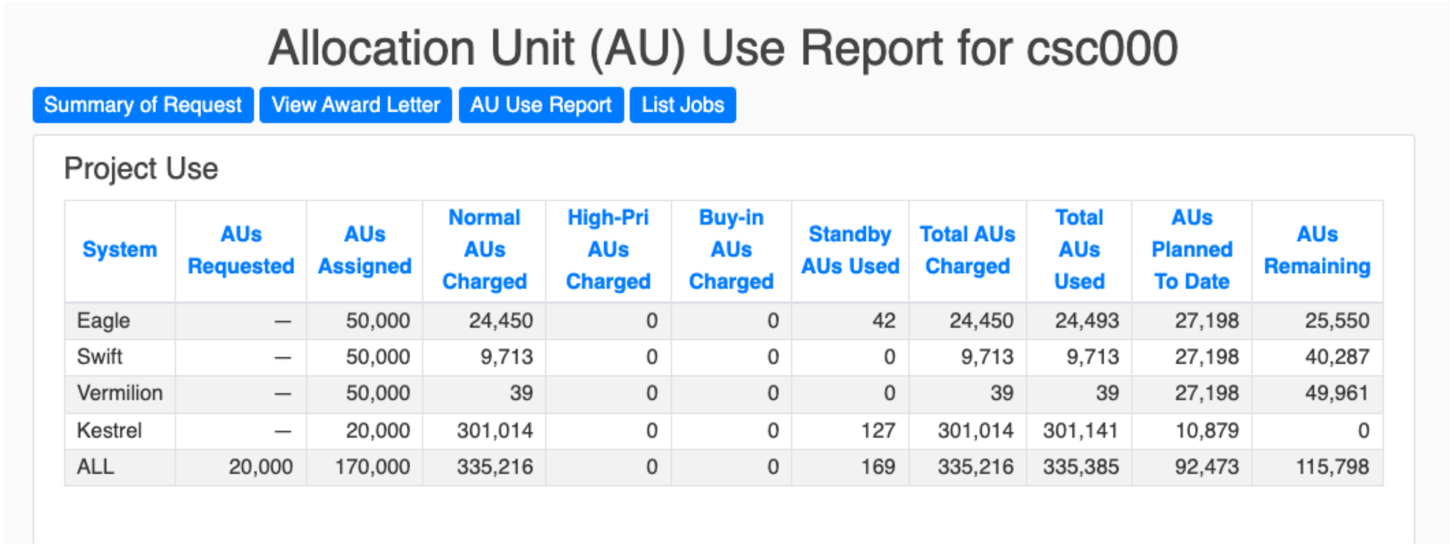
- Kestrel uses the Slurm job scheduler
- Submitting jobs:
  - sbatch: Use to submit a batch job
  - salloc: Used to allocate resources for an interactive job
- Slurm includes a suite of command-line tools to monitor jobs in the queue
- Job priority is calculated as a sum of multiple factors including age, jobsize, QOS, and fairshare
- Quality of service (QOS) :
  - standby: jobs with standby qos will only run if there are idle nodes that non-standby jobs can't use. No AU cost. Default for allocations with no AUs remaining, or can also specify in your job script.
  - --qos=high: gives small priority bump. Allocation will be charged at twice the standard rate of AUs
- Fairshare: calculated based on quarter AU assignment/total system AUs
  - Those running more than their fair share in the past two weeks will have decreased priority
  - Large impact on priority calculation
- Required flags for your jobs
  - --account=<project handle>: The allocation to use for the job
  - --time=<walltime>: Maximum Job Duration (walltime)

Partitions

- Compute nodes belong to one or more partitions. A partition is a collection of compute nodes which often share similar characteristics (memory, gpu, etc.).
- Kestrel partition specifications
- In general, we recommend not specifying a partition. Slurm automatically routes jobs to the appropriate partition based on requested resources.
  - Exceptions are debug and shared partitions
- --partition=debug: Nodes dedicated to developing and troubleshooting jobs
- --partition=shared
  - Nodes in the shared partition can be shared by multiple users or jobs. This partition is intended for jobs that do not require a whole node.
  - These nodes have 250GB of usable RAM and 104 cores. By default, your job will be allocated 1.024GB of RAM per core requested. You can use --mem or --mem-per-cpu to change this.
  - AU cost is calculated based on either the amount of cores or the amount of memory requested, whichever is a greater percentage of the total of that resource available on the node.

Accounting

- Track AU usage at https://hpcprojects.nrel.gov
  - Login with HPC account credentials
- AU cost = (Walltime in hours * Number of Nodes * QoS Factor * Charge Factor)
- Charge factor on Kestrel is 10 for CPU nodes, and 25 per GPU (100 per GPU node)



Allocation Unit (AU) Use Report for csc000

Project Use

System	AUs Requested	AUs Assigned	Normal AUs Charged	High-Pri AUs Charged	Buy-in AUs Charged	Standby AUs Used	Total AUs Charged	Total AUs Used	AUs Planned To Date	AUs Remaining
Eagle	—	50,000	24,450	0	0	42	24,450	24,493	27,198	25,550
Swift	—	50,000	9,713	0	0	0	9,713	9,713	27,198	40,287
Vermilion	—	50,000	39	0	0	0	39	39	27,198	49,961
Kestrel	—	20,000	301,014	0	0	127	301,014	301,141	10,879	0
ALL	20,000	170,000	335,216	0	0	169	335,216	335,385	92,473	115,798

Example submit script

```
#!/bin/bash
#SBATCH --nodes=2
#SBATCH --tasks-per-node=104
#SBATCH --time=2:00:00
#SBATCH --account=<your-account-name>
#SBATCH --job-name=<your-job-name>

module load vasp/<version>

srun vasp_std |& tee out
```

Non-shared partition

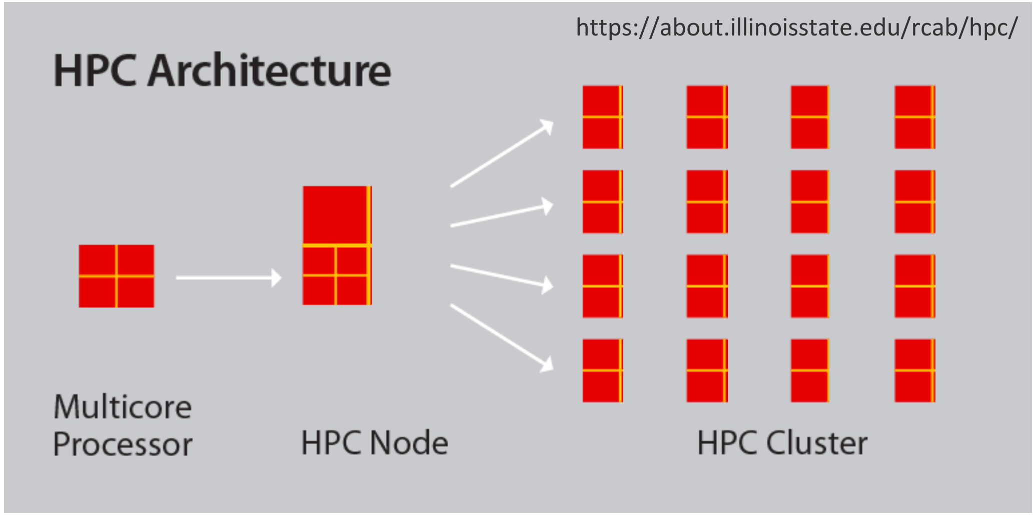
```
#!/bin/bash
#SBATCH --nodes=1
#SBATCH --tasks-per-node=26
#SBATCH --time=2:00:00
#SBATCH --partition=shared
#SBATCH --mem-per-cpu=2G
#SBATCH --account=<your-account-name>
#SBATCH --job-name=<your-job-name>

module load vasp/<version>

srun vasp_std |& tee out
```

Shared partition
Note the --mem-per-cpu directive

Environments



- Kestrel multiprocessor: Intel Sapphire Rapids, 52 CPU cores
- Kestrel node: two multiprocessors, **104 CPU cores total per node**
- Kestrel Cluster: 2100+ CPU nodes
- Hardware informs compiler/MPI choices

The "optimal environment" depends on your needs, but for CPU nodes:

- Intel processors → Intel compilers
- Cray Slingshot network → Cray MPICH or Cray MPICH ABI (multinode)

Using Anaconda

- Load the anaconda module:

  ```
  module load anaconda3
  ```

- Mamba is also available (install packages much faster)
- Install conda environments somewhere other than home directory
  - For example, project directory: /projects/your_project_name
- Mpi4py in anaconda utilizes OpenMPI, which is unstable for multinode jobs on our machine. If you need to use mpi4py multinode, please reach out to hpc-help@nrel.gov
- Documentation link: https://nrel.github.io/HPC/Documentation/Environment/Customization/conda/

Pre-made modules

- Use the command `module avail [name]` to see available modules
- To see more information on a module, use the command `module show [name]`.
- For example, if you are using a code like vasp, all you will need in order to set up your environment is `module load vasp/<version>` (and you can see which versions are available with `module avail vasp`
- Some applications have NREL documentation pages with sample submit scripts: https://nrel.github.io/HPC/Documentation/Applications/
  - Not an exhaustive list

PrgEnvs

- If you have not worked on a Cray machine before, environments and modules may work differently than you expect.
- Default environment is 'PrgEnv-cray'
- PrgEnvs are "modules of modules" - when you load a PrgEnv, you load a bundle of modules. For PrgEnv-cray:
  - Cray compiler environment "CCE" compilers (C, C++, Fortran)
  - Cray-mpich: Cray's MPI package, works better than other MPIs on our network
  - Some underlying communication/compilation libraries
  - Can see list of modules associated with PrgEnv-cray with `module show PrgEnv-cray`
- We recommend PrgEnv-cray (or PrgEnv-intel) for CPU codes.
- Tutorial on PrgEnvs: https://nrel.github.io/HPC/Documentation/Systems/Kestrel/Environments/tutorial/

Compiling code within PrgEnvs

- The PrgEnvs work differently than "typical" module environments
  - use the Cray wrappers cc, CC, and ftn (for c, c++, fortran, respectively)
- Wrapper on top of both the compiler and the MPI
- But flags you pass should be for underlying compiler
- May need to load additional modules like cray-fftw, cray-hdf5-parallel, etc.

Env	Wrappers	Underlying compiler	MPI	MPI location
PrgEnv-cray	cc, CC, ftn	CCE suite	Cray MPICH	/opt/cray/pe/mpich/8.1.23/ofi/crayclang/10.0/bin/
PrgEnv-intel	cc, CC, ftn	Intel suite (ifort, etc)	Cray MPICH	/opt/cray/pe/mpich/8.1.23/ofi/intel/19.0/bin/
PrgEnv-gnu	cc, CC, ftn	GNU (gcc, gfortran, etc)	Cray MPICH	/opt/cray/pe/mpich/8.1.23/ofi/gnu/9.1/bin/

```
cc --version
cc –o hello_world_mpi.exe hello_world_mpi.c
```

Environments

- The default environments are PrgEnvs
  - PrgEnv-cray, PrgEnv-intel, PrgEnv-gnu...
  - Compilers given in name
    - e.g. PrgEnv-intel uses intel compilers
  - MPI used is **Cray MPICH**
- Why use PrgEnv-*?
  - Cray MPICH better utilizes the Cray Slingshot network, so multi-node jobs run faster than with other MPIs
- NREL-built environments are also available
  - These behave similar to those on Eagle
- Refer to the environments tutorial for instructions on swapping between "realms" https://nrel.github.io/HPC/Documentation/Systems/Kestrel/Environments/tutorial/

PrgEnv "Realm"

Compilers:	MPI:	Scientific Libraries:
Cray CCE	Cray MPICH	cray-libsci
Intel		cray-fftw
GCC		cray-hdf5-parallel

/opt/cray/

Example:
module load PrgEnv-gnu
- Loads gcc compilers, Cray MPICH, cray-libsci, a bundle of communication libraries

NREL-built "Realm"

Compilers:	MPI:	Scientific Libraries:
Intel	Intel MPI	MKL, GSL
GCC	Open MPI	Open BLAS, LaPACK,
	MPICH	HDF5, etc.

/nopt/nrel/apps/modules/

Example:
module load hdf5/1.14.1-2-openmpi-gcc
- HDF5 built with gcc compilers and OpenMPI

Compiling

- Want to compile a code that needs MPI, scalapack, fftw, hdf5

PrgEnv-cray Example:

```
module restore
module load cray-libsci
module load cray-fftw
module load cray-hdf5-parallel
```

What this gets you:
Program built with Cray compilers, Cray MPICH, and Cray's implementations of fftw (auto-loads correct build), scalapack (libsci), and hdf5

NREL Envs Example (intel toolchain):

```
module restore
module unload PrgEnv-cray
module unload cce
module load intel-oneapi-compilers
module load intel-oneapi-mpi
module load intel-oneapi-mkl
module load fftw/3.3.10-intel-oneapi-mpi-intel
module load hdf5/1.14.1-2-intel-oneapi-mpi-intel
```

What this gets you:
Program built with intel compilers, Intel MPI, scalapack in MKL, fftw and hdf5 built with intel/intel MPI

Quick Reference Page

- To connect:

```
[user@mac ~]$ ssh <username>@kestrel.hpc.nrel.gov↲
```

- PrgEnv-cray loaded by default
- To set up for NREL-built environment modules:

```
[user@kestrel ~]$ module restore
[user@kestrel ~]$ module unload PrgEnv-cray
```

- If using a PrgEnv:
  - use the Cray wrappers (ftn/cc/CC) to compile
  - module load cray-[fftw, libsci, hdf5-parallel, etc] for scientific libraries

- "salloc -N 1 -n 104 --time=1:00:00 –account=[your account]" for an interactive session
- Build codes on compute nodes (via salloc, etc.)
- For applications that were compiled with OpenMP threading enabled, include in slurm script:
  export OMP_PROC_BIND=spread or export KMP_AFFINITY=balanced (intel)
- Getting started on Kestrel docs: https://nrel.github.io/HPC/Documentation/Systems/Kestrel/getting_started_kestrel/

Issues

- **MPI Scaling**
  - Poor scaling in multinode jobs is a known issue
  - We are working to get this resolved as fast as possible
  - See performance recommendations here: https://nrel.github.io/HPC/Documentation/Systems/eagle_to_kestrel_transition/#5-performance-recommendations

- We don't expect OpenMPI to perform multinode
- Reporting issues
  - Submit tickets to hpc-help@nrel.gov
  - Include details on the problem
    - Include job ID and any relevant outputs and inputs (log files, std-err, std-out, etc.)
    - Modules loaded, sbatch file, etc.

Resources

- Kestrel environments resources:
  - Overview of environments on Kestrel: https://nrel.github.io/HPC/Documentation/Systems/Kestrel/Environments/
  - Very simple tutorial: https://github.com/NREL/HPC/tree/master/kestrel/mpi_version_check
  - More complex environments tutorial: https://nrel.github.io/HPC/Documentation/Systems/Kestrel/Environments/tutorial/
- Code (examples, etc.) repo: https://github.com/NREL/HPC/tree/master/kestrel
- HPC office hours
  - https://www.nrel.gov/hpc/training.html
  - Stop by with questions
- Tickets: hpc-help@nrel.gov
- Documentation site: https://nrel.github.io/HPC/Documentation/

Feedback is Appreciated!

If you have any suggestions to improve this presentation we invite you to share with us at HPC-Help@nrel.gov

Thank You

www.nrel.gov

NREL
NATIONAL RENEWABLE ENERGY LABORATORY