iACS INSTITUTE FOR ADVANCED COMPUTATIONAL SCIENCE

Introduction to SeaWulf for CSE 416 students

Dave Carlson

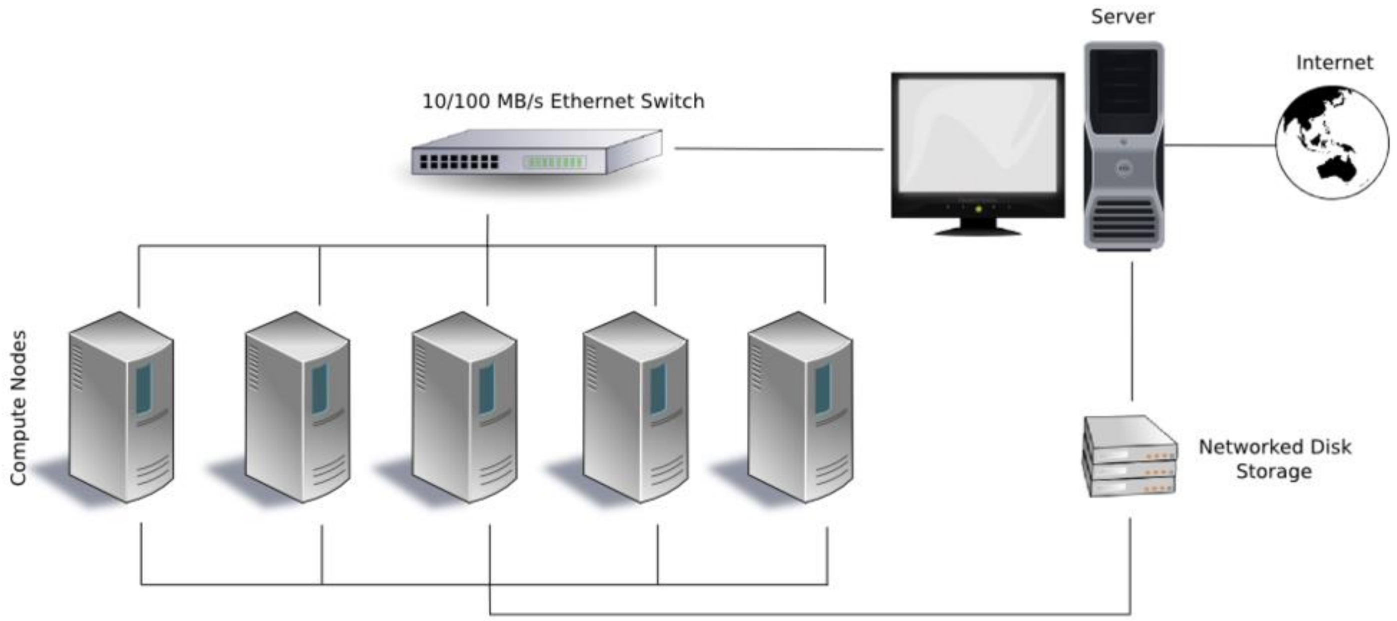
March 4th, 2021

SeaWulf

What's an HPC cluster, anyway?

A **High Performance Computing** (HPC) cluster contains multiple physically distinct computers ("nodes") that are connected over a network

A shared, parallel file system allows efficient access to the same data across all nodes



SeaWulf is …

- An HPC cluster dedicated to research applications for Stony Brook faculty, staff, and students

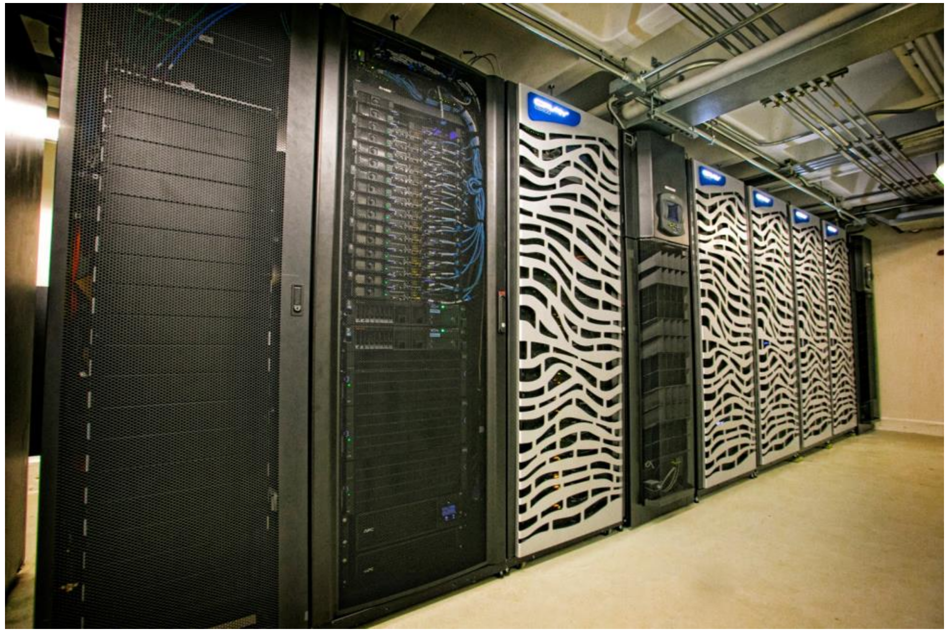
Available hardware:

- 2 **login nodes** = the entry points to the cluster

- 324 **compute nodes** = where the work is done
    - 24 – 40 CPUS each
    - 128 – 192 GB RAM each

- 8 **GPU nodes** each with 4 Nvidia Tesla K80 GPUs
- 1 **GPU node** with 2 Tesla P100 GPUs

- One **large memory node** with 3 TB of RAM

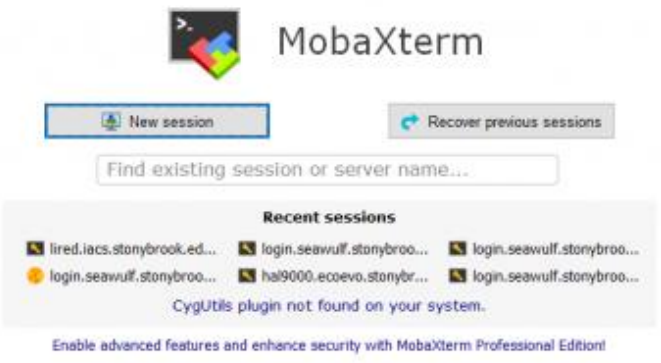


How do I connect to SeaWulf?

Mac & Linux users via the terminal:

```
ssh -X netid@login.seawulf.stonybrook.edu
```

Windows users:

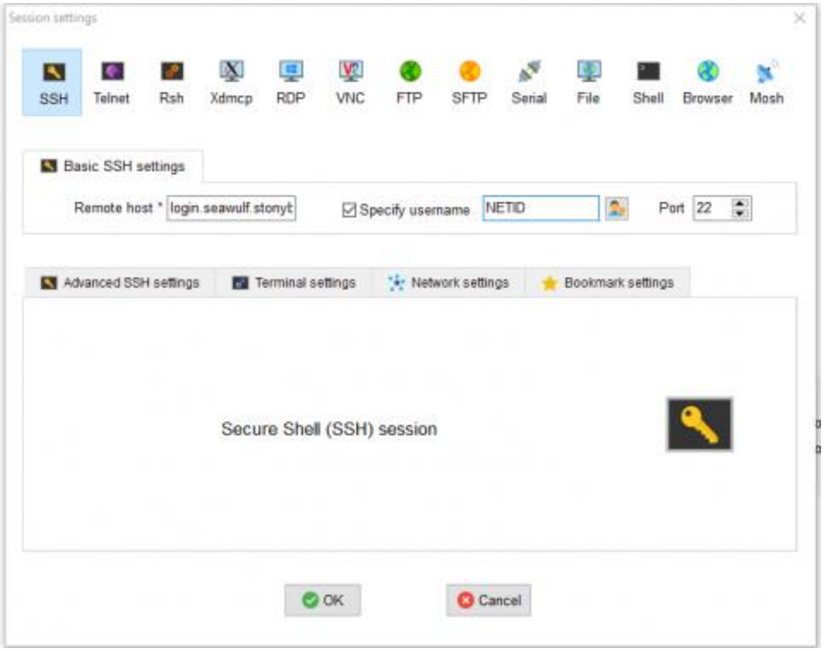

MobaXterm
New session
Recover previous sessions
Find existing session or server name...

Recent sessions

lired.iacs.stonybrook.ed... login.seawulf.stonybroo... login.seawulf.stonybroo...
login.seawulf.stonybroo... hal9000.ecoevo.stonybr... login.seawulf.stonybroo...

CygUtils plugin not found on your system.

Enable advanced features and enhance security with MobaXterm Professional Edition!



Remote host * login.seawulf.stonyt

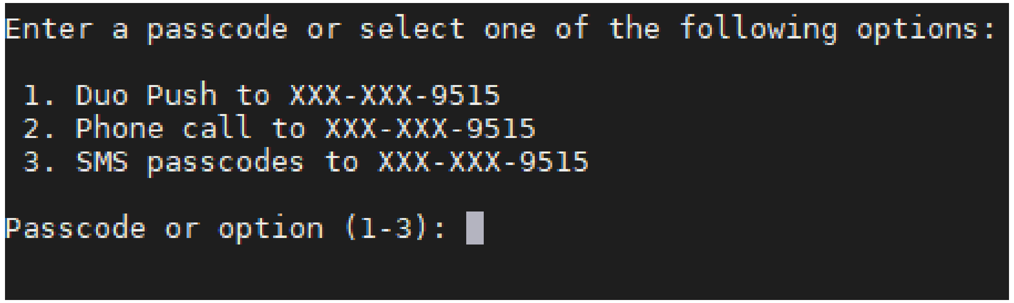
☑ Specify username NETID

Port 22

Secure Shell (SSH) session

2-factor authentication with DUO

- Upon login, you will be prompted to receive and respond to a push notification, sms, or phone call:



```
Enter a passcode or select one of the following options:

 1. Duo Push to XXX-XXX-9515
 2. Phone call to XXX-XXX-9515
 3. SMS passcodes to XXX-XXX-9515

Passcode or option (1-3): 
```

- Multiple failures to respond can lead to temporary lockout of your account

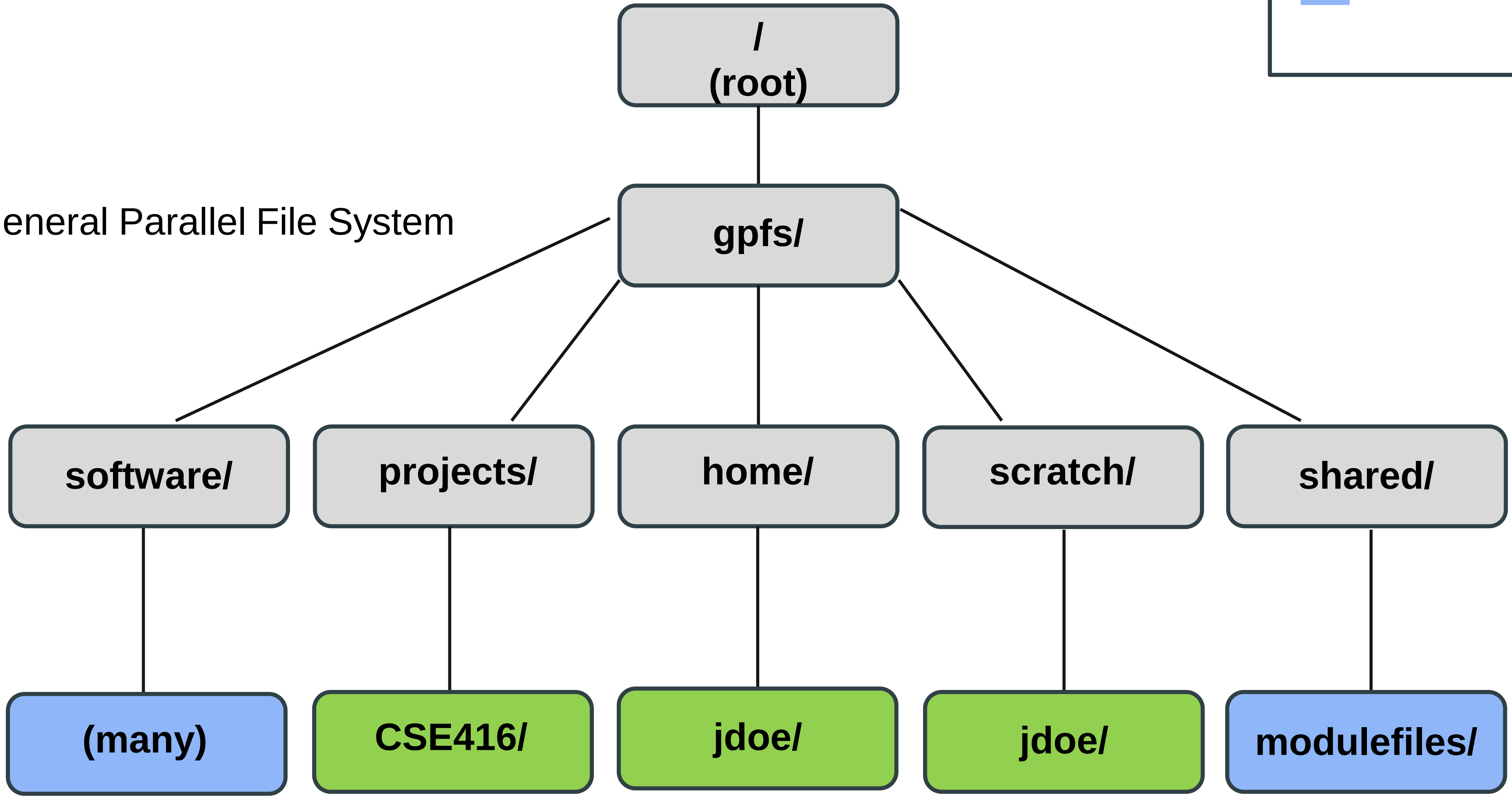
- *DUO 2FA can be bypassed if connected to SBU's VPN (GlobalProtect)*

The SeaWulf filesystem

🟩 = user *writeable*

🟦 = user *readable*

gpfs = General Parallel File System



Important paths to remember

- /gpfs/home/netid = **your home directory (20 GB)**

These environment variables also point to your home directory

$HOME

~

- /gpfs/scratch/netid = **your scratch directory (20 TB for housing temporary and intermediate files)**
- /gpfs/projects/CSE416 = **your project directory** (**5 TB shared space accessible to all class members)**

How do I transfer files onto SeaWulf?

Mac & Linux users should use scp (secure copy) to move files to and from SeaWulf

To transfer files from your computer to SeaWulf

1. Open terminal
2. scp /path/to/my/file netid@login.seawulf.stonybrook.edu:/path/to/destination/

To transfer files from SeaWulf to your computer

1. Open terminal
2. scp netid@login.seawulf.stonybrook.edu:/path/to/my/file /path/to/destination

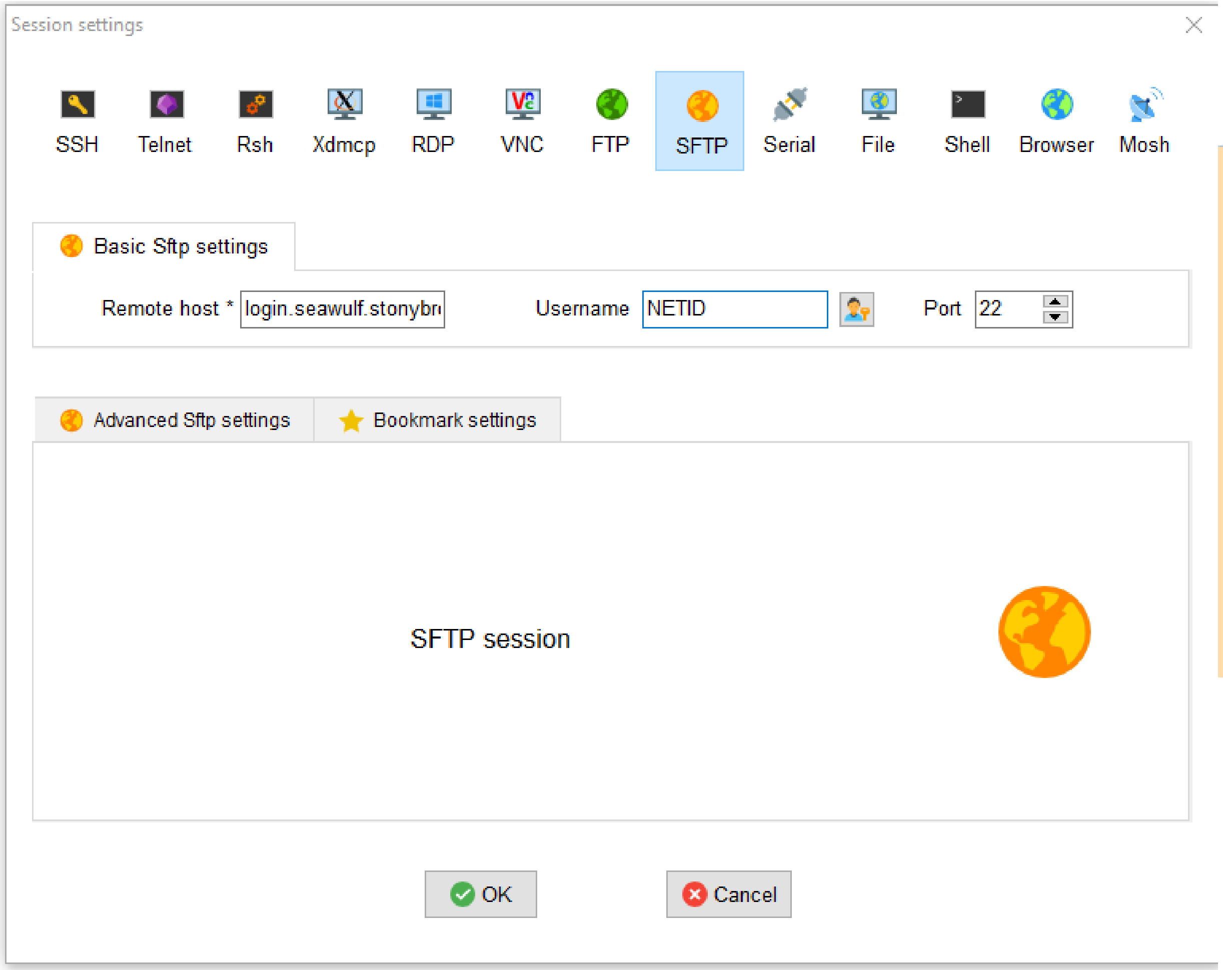
When possible, xfer archives (e.g. tarballs) or directories because:

1. It’s faster to transfer one large file than many small files

2. Unless connected to the SBU VPN, you may receive 1 DUO prompt for every scp command you run!

How do I transfer files onto SeaWulf?

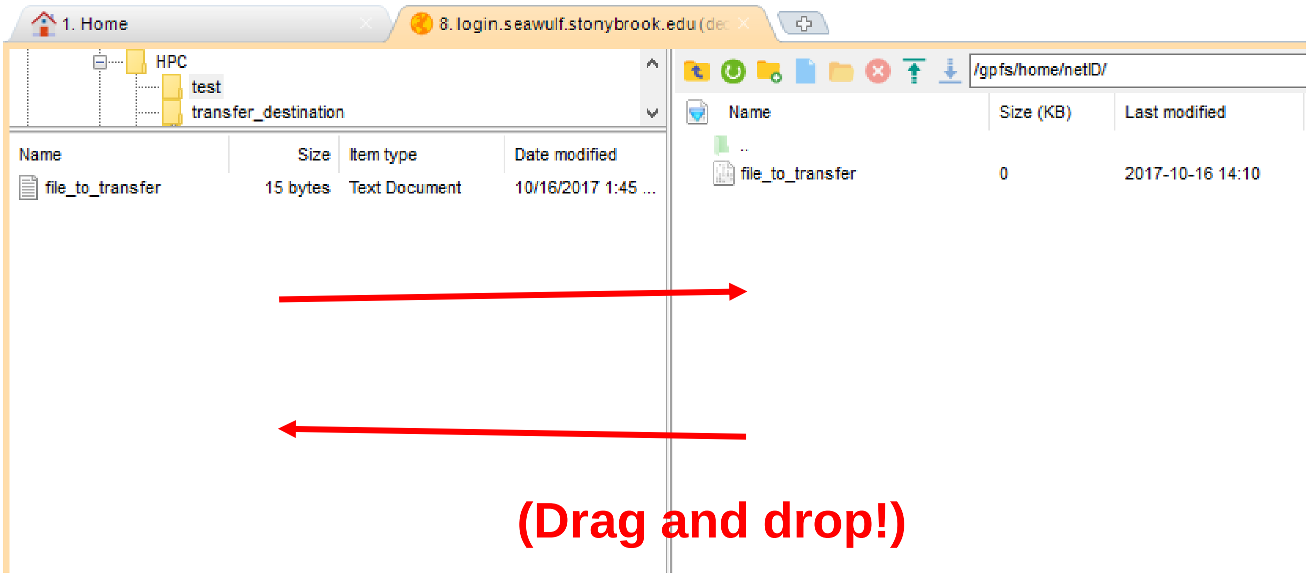
Windows users:



Remote host * login.seawulf.stonybr

Username NETID

Port 22



Name	Size	Item type	Date modified
file_to_transfer	15 bytes	Text Document	10/16/2017 1:45 ...

/gpfs/home/netID/

Name	Size (KB)	Last modified
..		
file_to_transfer	0	2017-10-16 14:10

(Drag and drop!)

Using the module system to access software

Useful module commands:

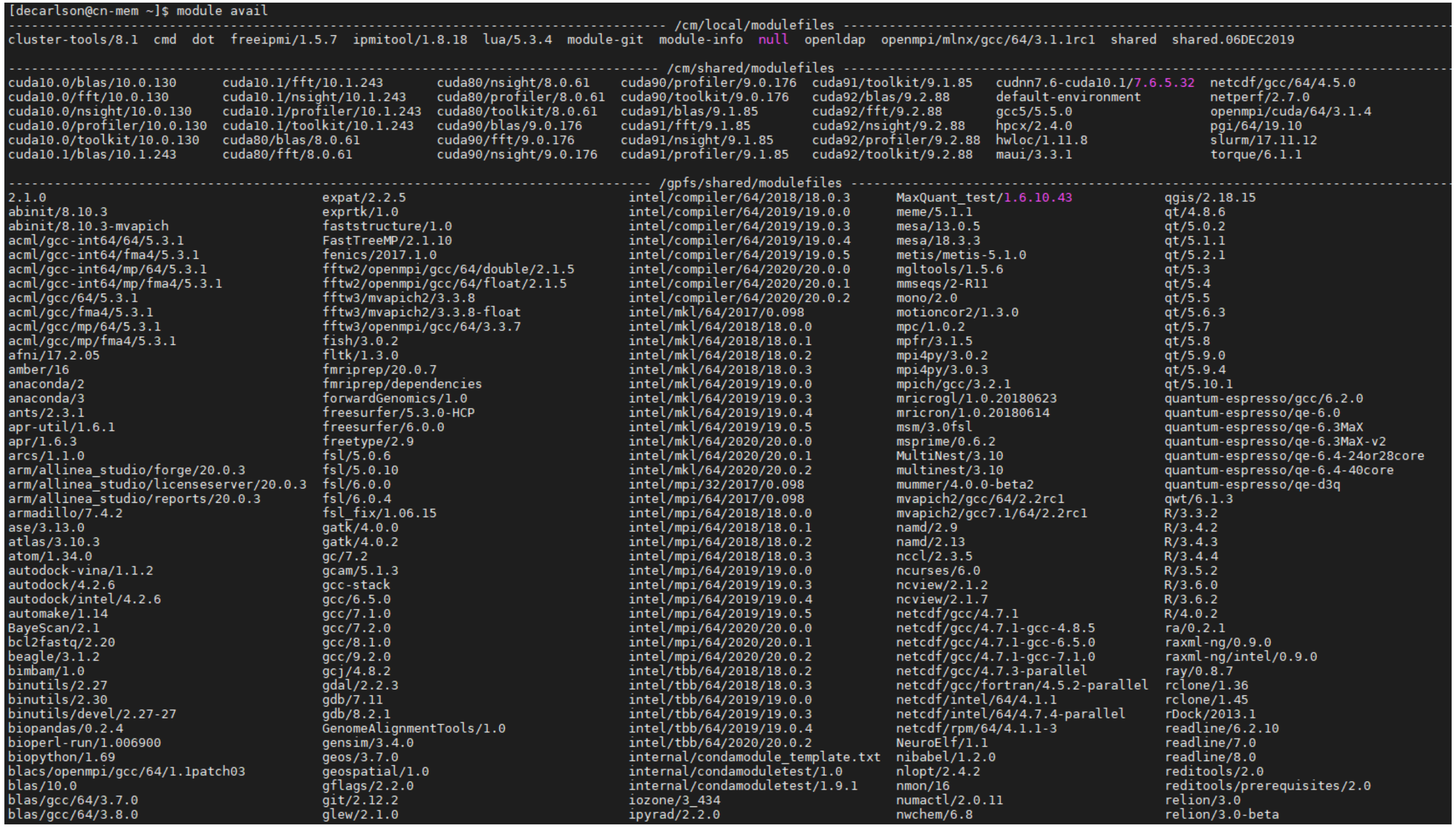
module avail

module load

module list

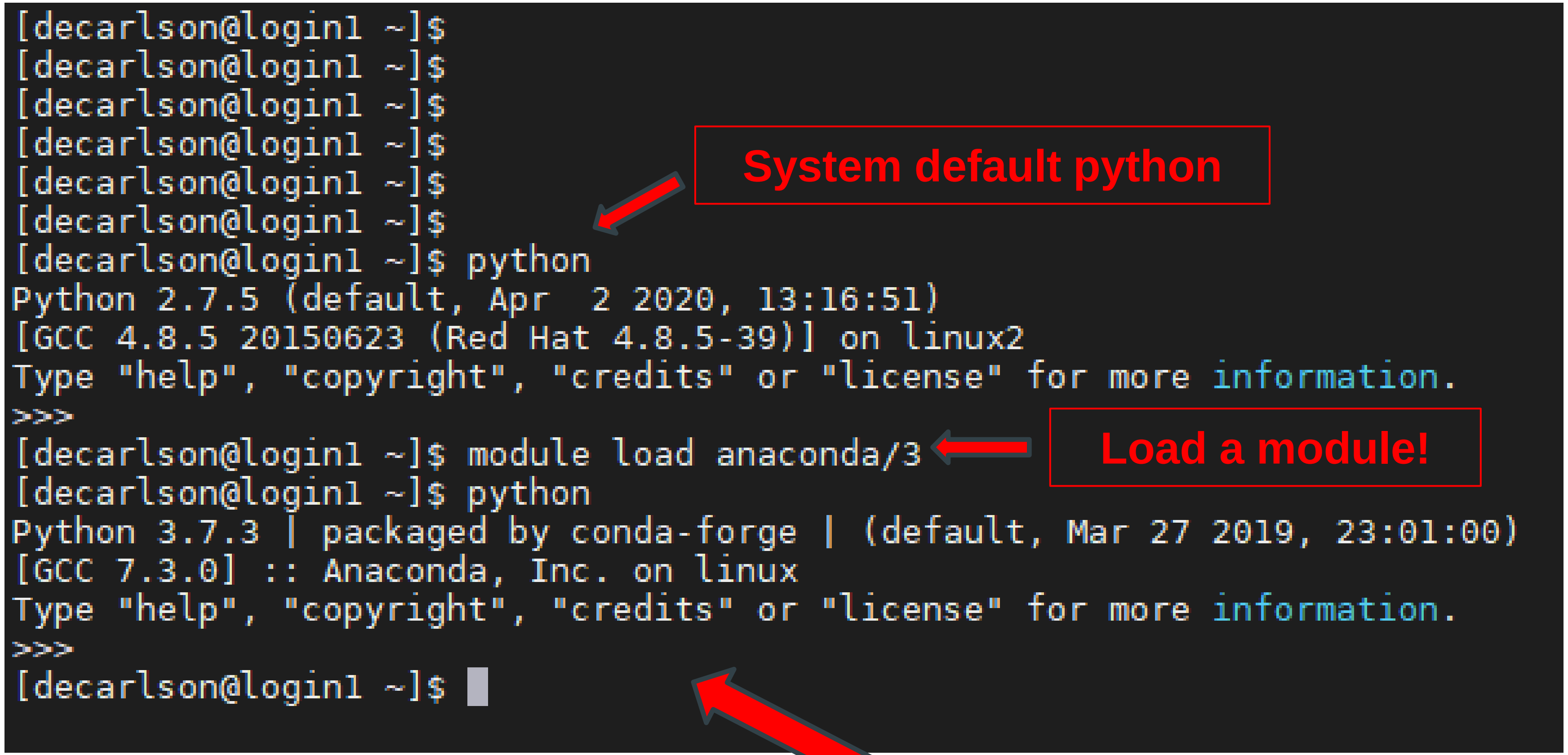
module unload

module purge



```
[decarlson@cn-mem ~]$ module avail
-------------------------------------------------------------------------------------- /cm/local/modulefiles -------------------------------------------------------------------------------
cluster-tools/8.1  cmd  dot  freeipmi/1.5.7  ipmitool/1.8.18  lua/5.3.4  module-git  module-info  null  openldap  openmpi/mlnx/gcc/64/3.1.1rc1  shared  shared.06DEC2019

------------------------------------------------------------------------------------- /cm/shared/modulefiles -------------------------------------------------------------------------------
cuda10.0/blas/10.0.130      cuda10.1/fft/10.1.243       cuda80/nsight/8.0.61    cuda90/profiler/9.0.176  cuda91/toolkit/9.1.85   cudnn7.6-cuda10.1/7.6.5.32  netcdf/gcc/64/4.5.0
cuda10.0/fft/10.0.130       cuda10.1/nsight/10.1.243    cuda80/profiler/8.0.61  cuda90/toolkit/9.0.176   cuda92/blas/9.2.88      default-environment         netperf/2.7.0
cuda10.0/nsight/10.0.130    cuda10.1/profiler/10.1.243  cuda80/toolkit/8.0.61   cuda91/blas/9.1.85       cuda92/fft/9.2.88       gcc5/5.5.0                  openmpi/cuda/64/3.1.4
cuda10.0/profiler/10.0.130  cuda10.1/toolkit/10.1.243   cuda90/blas/9.0.176     cuda91/fft/9.1.85        cuda92/nsight/9.2.88    hpcx/2.4.0                  pgi/64/19.10
cuda10.0/toolkit/10.0.130   cuda80/blas/8.0.61          cuda90/fft/9.0.176      cuda91/nsight/9.1.85     cuda92/profiler/9.2.88  hwloc/1.11.8                slurm/17.11.12
cuda10.1/blas/10.1.243      cuda80/fft/8.0.61           cuda90/nsight/9.0.176   cuda91/profiler/9.1.85   cuda92/toolkit/9.2.88   maui/3.3.1                  torque/6.1.1

------------------------------------------------------------------------------------ /gpfs/shared/modulefiles ------------------------------------------------------------------------------
2.1.0                                    expat/2.2.5                             intel/compiler/64/2018/18.0.3      MaxQuant_test/1.6.10.43            qgis/2.18.15
abinit/8.10.3                            exprtk/1.0                              intel/compiler/64/2019/19.0.0      meme/5.1.1                         qt/4.8.6
abinit/8.10.3-mvapich                    faststructure/1.0                       intel/compiler/64/2019/19.0.3      mesa/13.0.5                        qt/5.0.2
acml/gcc-int64/64/5.3.1                  FastTreeMP/2.1.10                       intel/compiler/64/2019/19.0.4      mesa/18.3.3                        qt/5.1.1
acml/gcc-int64/fma4/5.3.1                fenics/2017.1.0                         intel/compiler/64/2019/19.0.5      metis/metis-5.1.0                  qt/5.2.1
acml/gcc-int64/mp/64/5.3.1               fftw2/openmpi/gcc/64/double/2.1.5       intel/compiler/64/2020/20.0.0      mgltools/1.5.6                     qt/5.3
acml/gcc-int64/mp/fma4/5.3.1             fftw2/openmpi/gcc/64/float/2.1.5        intel/compiler/64/2020/20.0.1      mmseqs/2-R11                       qt/5.4
acml/gcc/64/5.3.1                        fftw3/mvapich2/3.3.8                    intel/compiler/64/2020/20.0.2      mono/2.0                           qt/5.5
acml/gcc/fma4/5.3.1                      fftw3/mvapich2/3.3.8-float              intel/mkl/64/2017/0.098            motioncor2/1.3.0                   qt/5.6.3
acml/gcc/mp/64/5.3.1                     fftw3/openmpi/gcc/64/3.3.7              intel/mkl/64/2018/18.0.0           mpc/1.0.2                          qt/5.7
acml/gcc/mp/fma4/5.3.1                   fish/3.0.2                              intel/mkl/64/2018/18.0.1           mpfr/3.1.5                         qt/5.8
afni/17.2.05                             fltk/1.3.0                              intel/mkl/64/2018/18.0.2           mpi4py/3.0.2                       qt/5.9.0
amber/16                                 fmriprep/20.0.7                         intel/mkl/64/2018/18.0.3           mpi4py/3.0.3                       qt/5.9.4
anaconda/2                               fmriprep/dependencies                   intel/mkl/64/2019/19.0.0           mpich/gcc/3.2.1                    qt/5.10.1
anaconda/3                               forwardGenomics/1.0                     intel/mkl/64/2019/19.0.3           mricrogl/1.0.20180623              quantum-espresso/gcc/6.2.0
ants/2.3.1                               freesurfer/5.3.0-HCP                    intel/mkl/64/2019/19.0.4           mricron/1.0.20180614               quantum-espresso/qe-6.0
apr-util/1.6.1                           freesurfer/6.0.0                        intel/mkl/64/2019/19.0.5           msm/3.0fsl                         quantum-espresso/qe-6.3MaX
apr/1.6.3                                freetype/2.9                            intel/mkl/64/2020/20.0.0           msprime/0.6.2                      quantum-espresso/qe-6.3MaX-v2
arcs/1.1.0                               fsl/5.0.6                               intel/mkl/64/2020/20.0.1           MultiNest/3.10                     quantum-espresso/qe-6.4-24or28core
arm/allinea_studio/forge/20.0.3          fsl/5.0.10                              intel/mkl/64/2020/20.0.2           multinest/3.10                     quantum-espresso/qe-6.4-40core
arm/allinea_studio/licenseserver/20.0.3  fsl/6.0.0                               intel/mpi/32/2017/0.098            mummer/4.0.0-beta2                 quantum-espresso/qe-d3q
arm/allinea_studio/reports/20.0.3        fsl/6.0.4                               intel/mpi/64/2017/0.098            mvapich2/gcc/64/2.2rc1             qwt/6.1.3
armadillo/7.4.2                          fsl_fix/1.06.15                         intel/mpi/64/2018/18.0.0           mvapich2/gcc7.1/64/2.2rc1          R/3.3.2
ase/3.13.0                               gatk/4.0.0                              intel/mpi/64/2018/18.0.1           namd/2.9                           R/3.4.2
atlas/3.10.3                             gatk/4.0.2                              intel/mpi/64/2018/18.0.2           namd/2.13                          R/3.4.3
atom/1.34.0                              gc/7.2                                  intel/mpi/64/2018/18.0.3           nccl/2.3.5                         R/3.4.4
autodock-vina/1.1.2                      gcam/5.1.3                              intel/mpi/64/2019/19.0.0           ncurses/6.0                        R/3.5.2
autodock/4.2.6                           gcc-stack                               intel/mpi/64/2019/19.0.3           ncview/2.1.2                       R/3.6.0
autodock/intel/4.2.6                     gcc/6.5.0                               intel/mpi/64/2019/19.0.4           ncview/2.1.7                       R/3.6.2
automake/1.14                            gcc/7.1.0                               intel/mpi/64/2019/19.0.5           netcdf/gcc/4.7.1                   R/4.0.2
BayeScan/2.1                             gcc/7.2.0                               intel/mpi/64/2020/20.0.0           netcdf/gcc/4.7.1-gcc-4.8.5         ra/0.2.1
bcl2fastq/2.20                           gcc/8.1.0                               intel/mpi/64/2020/20.0.1           netcdf/gcc/4.7.1-gcc-6.5.0         raxml-ng/0.9.0
beagle/3.1.2                             gcc/9.2.0                               intel/mpi/64/2020/20.0.2           netcdf/gcc/4.7.1-gcc-7.1.0         raxml-ng/intel/0.9.0
bimbam/1.0                               gcj/4.8.2                               intel/tbb/64/2018/18.0.2           netcdf/gcc/4.7.3-parallel          ray/0.8.7
binutils/2.27                            gdal/2.2.3                              intel/tbb/64/2018/18.0.3           netcdf/gcc/fortran/4.5.2-parallel  rclone/1.36
binutils/2.30                            gdb/7.11                                intel/tbb/64/2019/19.0.0           netcdf/intel/64/4.1.1              rclone/1.45
binutils/devel/2.27-27                   gdb/8.2.1                               intel/tbb/64/2019/19.0.3           netcdf/intel/64/4.7.4-parallel     rDock/2013.1
biopandas/0.2.4                          GenomeAlignmentTools/1.0                intel/tbb/64/2019/19.0.4           netcdf/rpm/64/4.1.1-3              readline/6.2.10
bioperl-run/1.006900                     gensim/3.4.0                            intel/tbb/64/2020/20.0.2           NeuroElf/1.1                       readline/7.0
biopython/1.69                           geos/3.7.0                              internal/condamodule_template.txt  nibabel/1.2.0                      readline/8.0
blacs/openmpi/gcc/64/1.1patch03          geospatial/1.0                          internal/condamoduletest/1.0       nlopt/2.4.2                        reditools/2.0
blas/10.0                                gflags/2.2.0                            internal/condamoduletest/1.9.1     nmon/16                            reditools/prerequisites/2.0
blas/gcc/64/3.7.0                        git/2.12.2                              iozone/3_434                       numactl/2.0.11                     relion/3.0
blas/gcc/64/3.8.0                        glew/2.1.0                              ipyrad/2.2.0                       nwchem/6.8                         relion/3.0-beta
```

Using the module system to access software



```
[decarlson@login1 ~]$
[decarlson@login1 ~]$
[decarlson@login1 ~]$
[decarlson@login1 ~]$
[decarlson@login1 ~]$
[decarlson@login1 ~]$
[decarlson@login1 ~]$ python
Python 2.7.5 (default, Apr  2 2020, 13:16:51)
[GCC 4.8.5 20150623 (Red Hat 4.8.5-39)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>>
[decarlson@login1 ~]$ module load anaconda/3
[decarlson@login1 ~]$ python
Python 3.7.3 | packaged by conda-forge | (default, Mar 27 2019, 23:01:00)
[GCC 7.3.0] :: Anaconda, Inc. on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
[decarlson@login1 ~]$
```

System default python

Load a module!

Newer python version
now available!

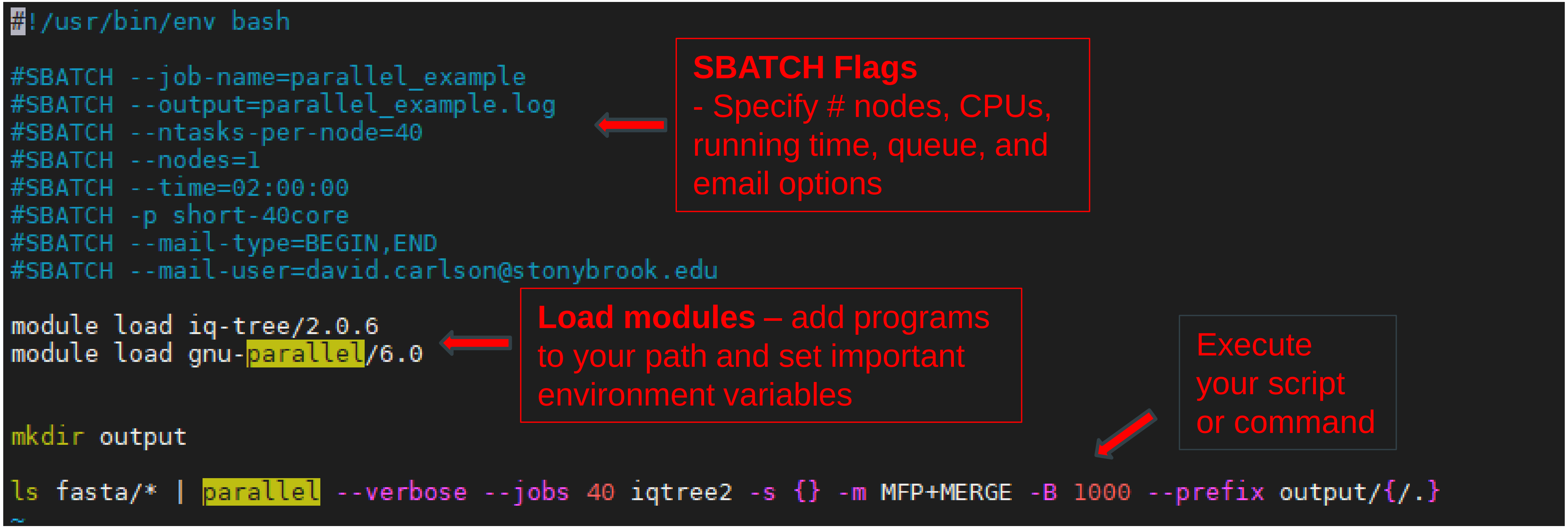
How do I do work on SeaWulf?

Computationally intensive jobs should *not* be done on the login node!

- To run a job, you must submit a batch script to the Slurm Workload Manager
- A batch script is a collection of bash commands issued to the scheduler, which which distributes your job across one or more compute nodes
- The user requests specific resources (nodes, cpus, job time, etc.), while the scheduler places the job into the queue until the resources are available

Example Slurm Job Script

All jobs submitted through a job scheduling system using scripts



```
#!/usr/bin/env bash

#SBATCH --job-name=parallel_example
#SBATCH --output=parallel_example.log
#SBATCH --ntasks-per-node=40
#SBATCH --nodes=1
#SBATCH --time=02:00:00
#SBATCH -p short-40core
#SBATCH --mail-type=BEGIN,END
#SBATCH --mail-user=david.carlson@stonybrook.edu

module load iq-tree/2.0.6
module load gnu-parallel/6.0


mkdir output

ls fasta/* | parallel --verbose --jobs 40 iqtree2 -s {} -m MFP+MERGE -B 1000 --prefix output/{/.}
```

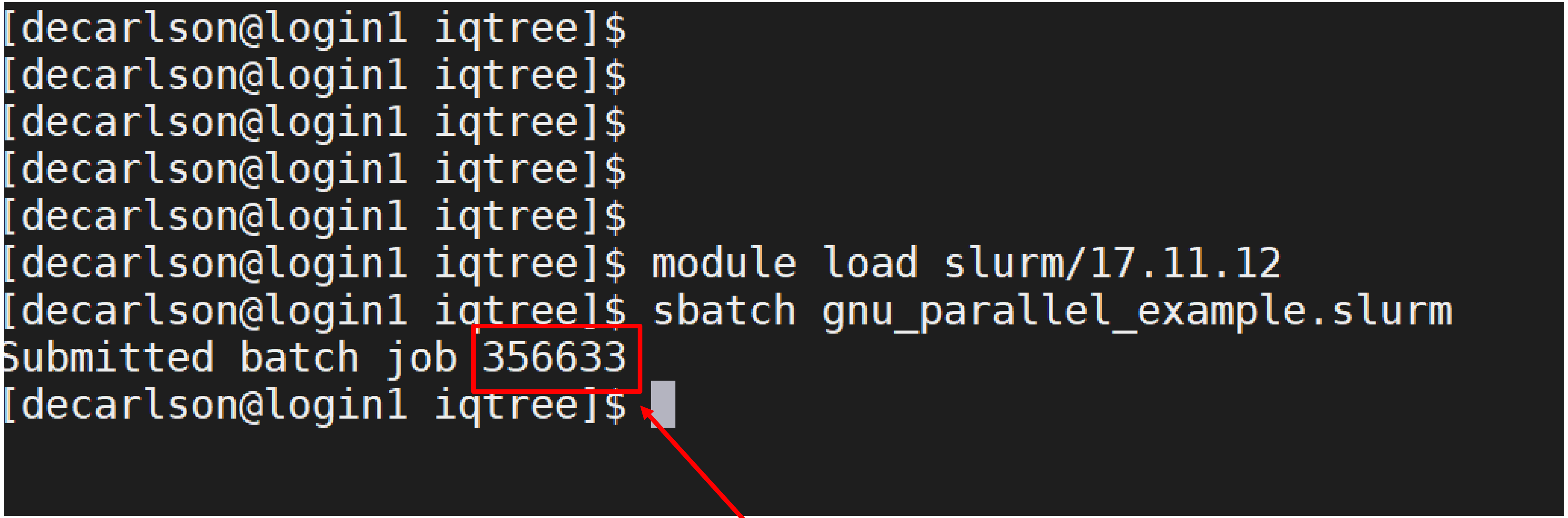
SBATCH Flags
- Specify # nodes, CPUs, running time, queue, and email options

Load modules – add programs to your path and set important environment variables

Execute your script or command

How do I execute my Slurm script?

Jobs are submitted via the Slurm Workload Manager using the “sbatch” command



```
[decarlson@login1 iqtree]$
[decarlson@login1 iqtree]$
[decarlson@login1 iqtree]$
[decarlson@login1 iqtree]$
[decarlson@login1 iqtree]$
[decarlson@login1 iqtree]$ module load slurm/17.11.12
[decarlson@login1 iqtree]$ sbatch gnu_parallel_example.slurm
Submitted batch job 356633
[decarlson@login1 iqtree]$
```

This is your job ID.

Useful Slurm commands

sbatch <script> = submit a job

scancel <job id> = cancel a job

squeue = get job status

scontrol show job <job id> = get detailed job info

sinfo = get info on node/queue status and utilization

What queue should I submit to?

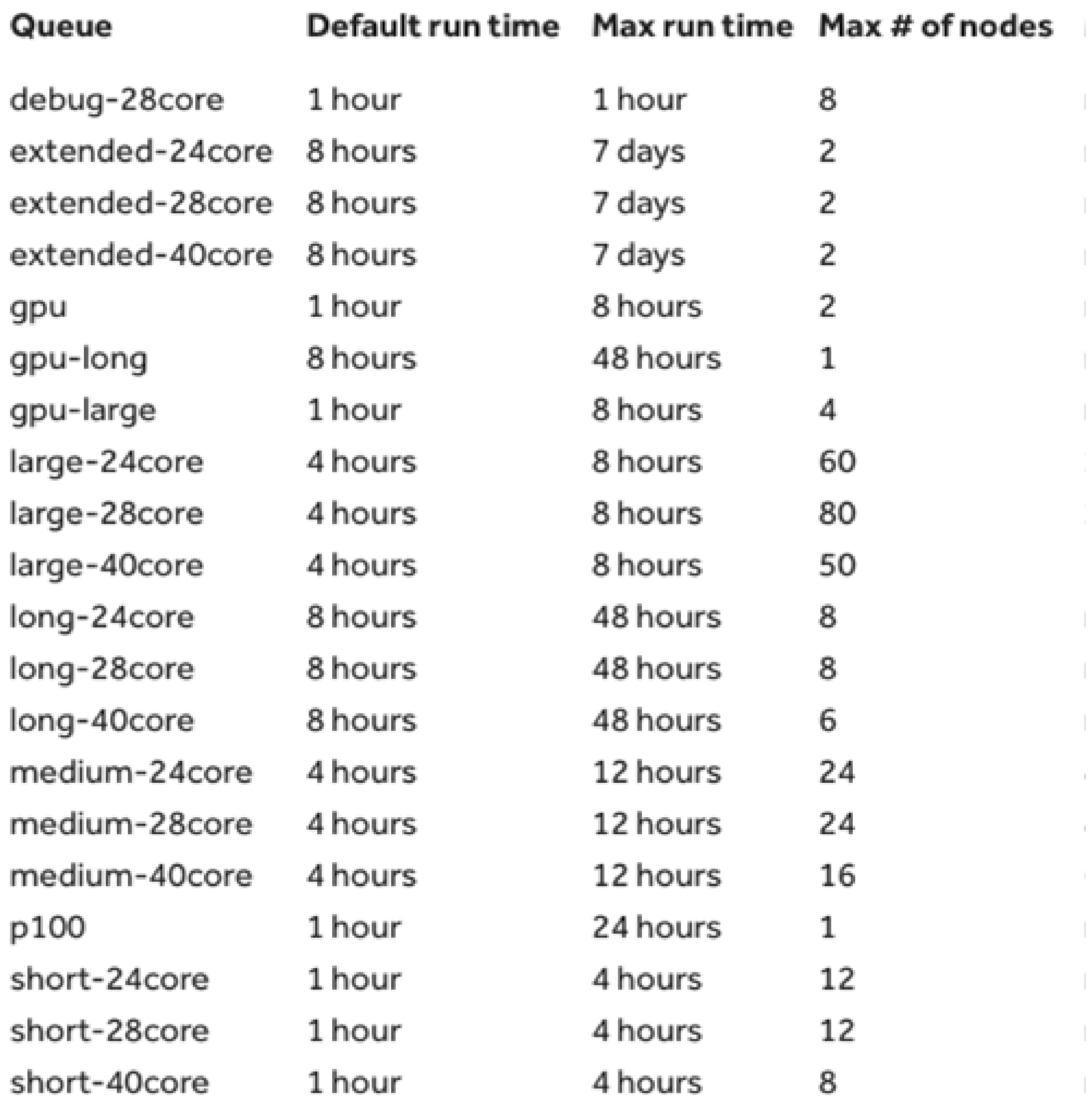
How many nodes do you need?

How many cores per node?

How much time do you need?

There is often a tradeoff between resource usage and wait time!!

Don’t wait until the last minute to submit jobs when you have a deadline!



Queue	Default run time	Max run time	Max # of nodes
debug-28core	1 hour	1 hour	8
extended-24core	8 hours	7 days	2
extended-28core	8 hours	7 days	2
extended-40core	8 hours	7 days	2
gpu	1 hour	8 hours	2
gpu-long	8 hours	48 hours	1
gpu-large	1 hour	8 hours	4
large-24core	4 hours	8 hours	60
large-28core	4 hours	8 hours	80
large-40core	4 hours	8 hours	50
long-24core	8 hours	48 hours	8
long-28core	8 hours	48 hours	8
long-40core	8 hours	48 hours	6
medium-24core	4 hours	12 hours	24
medium-28core	4 hours	12 hours	24
medium-40core	4 hours	12 hours	16
p100	1 hour	24 hours	1
short-24core	1 hour	4 hours	12
short-28core	1 hour	4 hours	12
short-40core	1 hour	4 hours	8

Parallel processing on the cluster

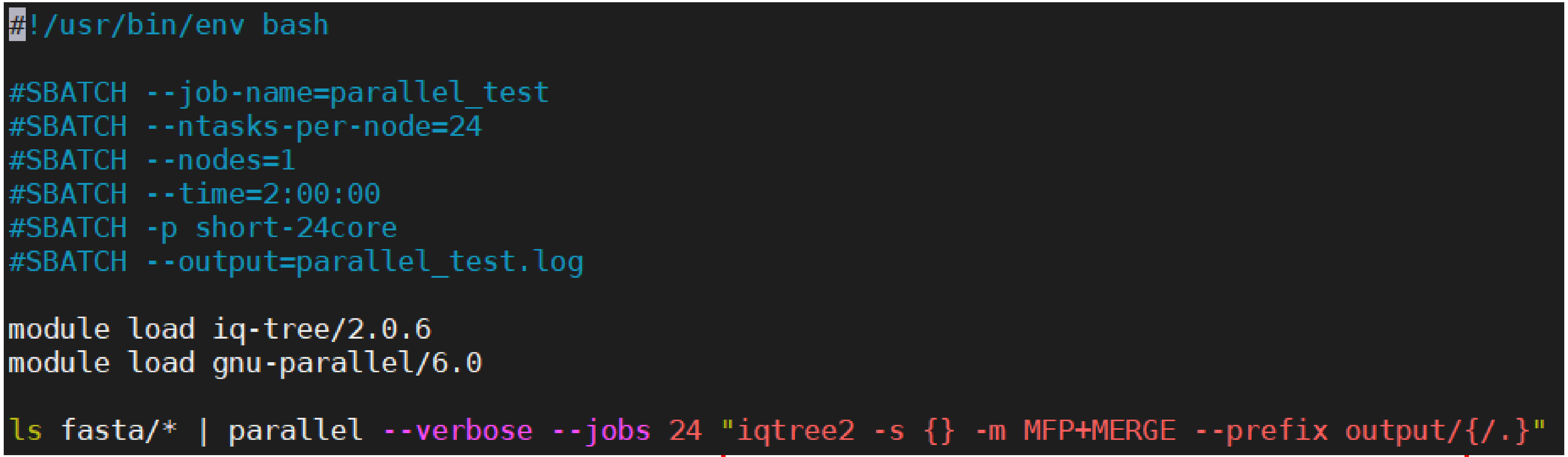
- **Parallelization *within a single compute node***
  - Lots of ways of doing this
  - Some software written to run on multiple cores/threads
  - Some tasks easily parallelized with scripting (e.g, “Embarrassingly Parallel” tasks)
- **Parallelization across *multiple nodes***
  - Requires the use of MPI
- **Parallelization with GPUs** - only available on specific (“sn-nvda”) GPU nodes

Parallel processing on a single node with GNU Parallel

- Perfect for “embarrassingly parallel” situations
- Available as a module: gnu-parallel/6.0
- Can easily take in a series of inputs (e.g., files) and run a command on each input simultaneously
- Lots of tutorials and resources available on the web!

https://www.gnu.org/software/parallel/parallel_tutorial.html (thorough!!)
https://www.msi.umn.edu/support/faq/how-can-i-use-gnu-parallel-run-lot-commands-parallel (many practical examples)

Parallel processing on a single node with GNU Parallel



```
#!/usr/bin/env bash

#SBATCH --job-name=parallel_test
#SBATCH --ntasks-per-node=24
#SBATCH --nodes=1
#SBATCH --time=2:00:00
#SBATCH -p short-24core
#SBATCH --output=parallel_test.log

module load iq-tree/2.0.6
module load gnu-parallel/6.0

ls fasta/* | parallel --verbose --jobs 24 "iqtree2 -s {} -m MFP+MERGE --prefix output/{/.}"
```

List input files & pipe to GNU parallel command

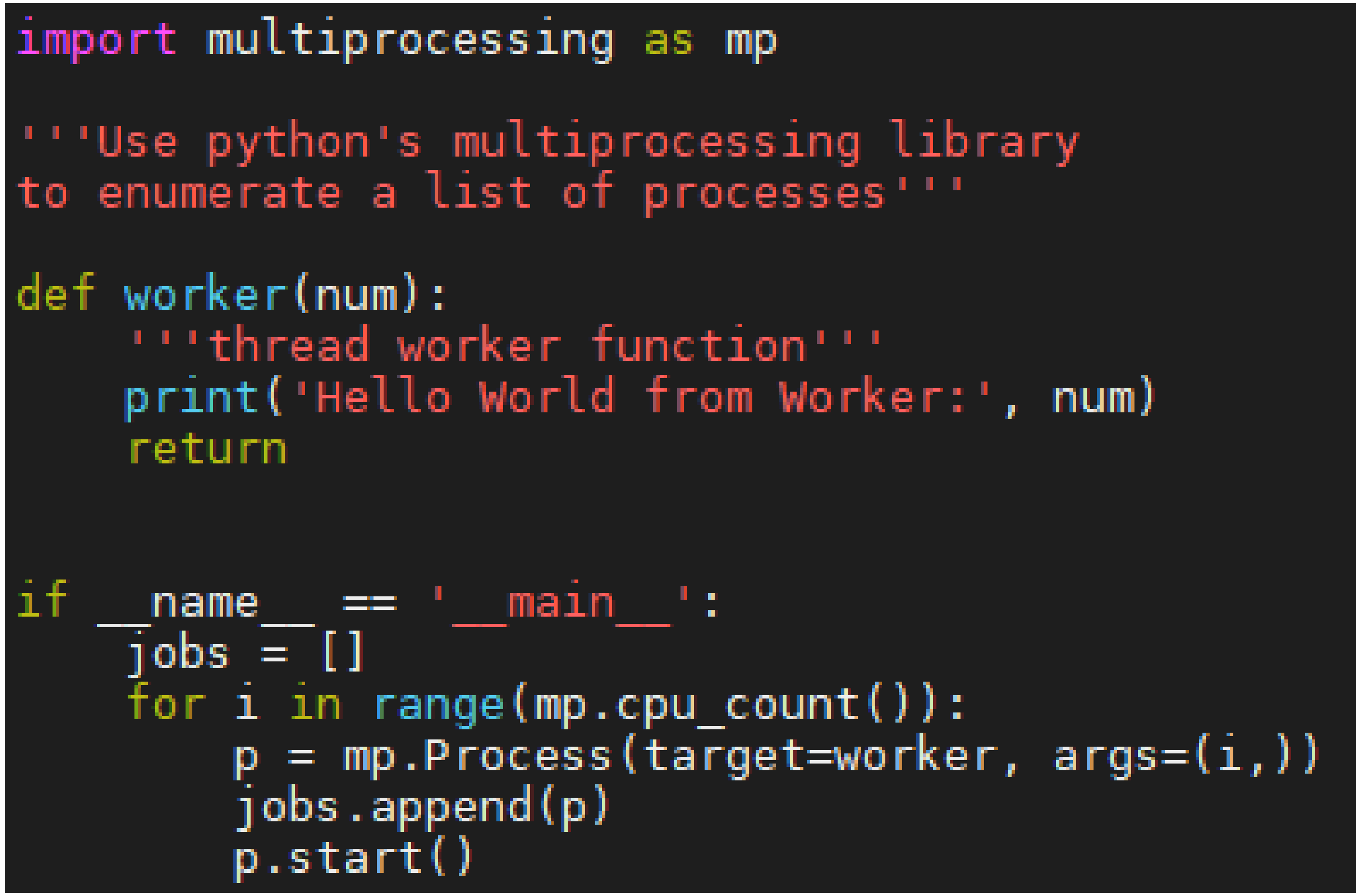
Optional flags to control behavior

This command will be run on each input file

{} = each file in fasta/

{/.} = each file with path and extension truncated

Parallel processing on a single node with Python's multiprocessing library



```
import multiprocessing as mp

'''Use python's multiprocessing library
to enumerate a list of processes'''

def worker(num):
    '''thread worker function'''
    print('Hello World from Worker:', num)
    return


if __name__ == '__main__':
    jobs = []
    for i in range(mp.cpu_count()):
        p = mp.Process(target=worker, args=(i,))
        jobs.append(p)
        p.start()
```

Basic premise:
1. Write a function to do something
2. Parallelize the execution of the function using mp (e.g., in a loop)

(modified from https://pymotw.com/2/multiprocessing/basics.html)

Can my job use multiple nodes?

Yes! (...well...maybe)

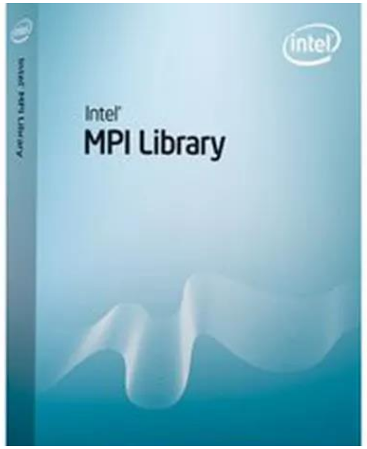
Message Passing Interface (MPI) facilitates communication between processes within or among nodes

Not all software is compatible with MPI

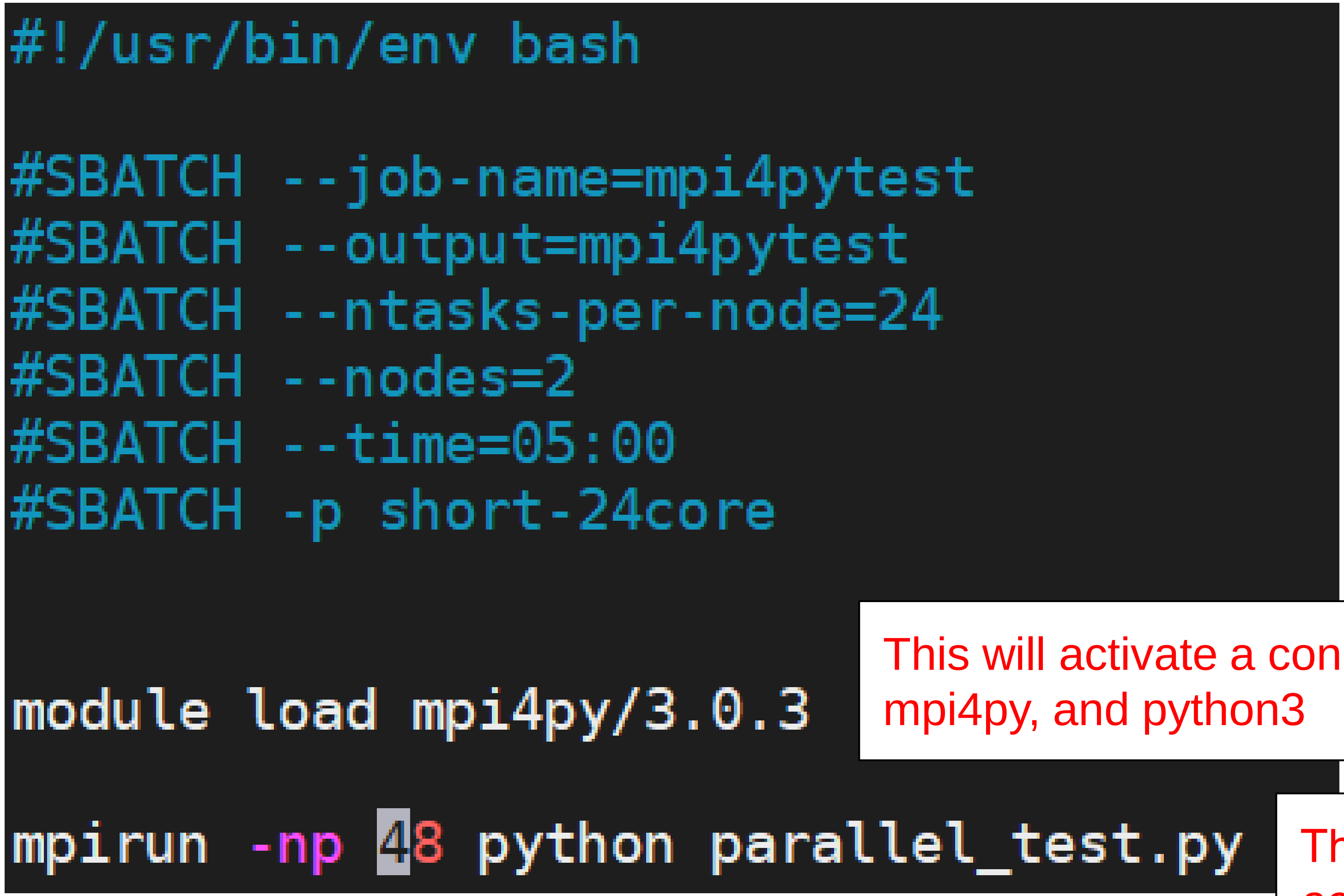
Multiple “flavors” of MPI are available on SeaWulf

- Opensource: **mvapich***, **mpich, OpenMPI**
- Licensed: **Intel***

*=officially supported



Example MPI Job Submission Script



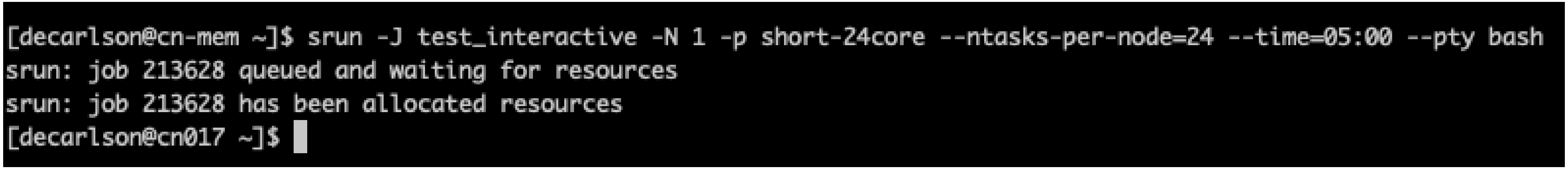
```
#!/usr/bin/env bash

#SBATCH --job-name=mpi4pytest
#SBATCH --output=mpi4pytest
#SBATCH --ntasks-per-node=24
#SBATCH --nodes=2
#SBATCH --time=05:00
#SBATCH -p short-24core


module load mpi4py/3.0.3

mpirun -np 48 python parallel_test.py
```

This will activate a conda env with OpenMPI, mpi4py, and python3

The “-np” flag specifies how many processors *across all nodes* will be used.

Need to troubleshoot? Use an interactive job!

Example:



```
[decarlson@cn-mem ~]$ srun -J test_interactive -N 1 -p short-24core --ntasks-per-node=24 --time=05:00 --pty bash
srun: job 213628 queued and waiting for resources
srun: job 213628 has been allocated resources
[decarlson@cn017 ~]$ 
```

”srun” requests a compute node for interactive use

“--pty bash” configures the terminal on the compute node

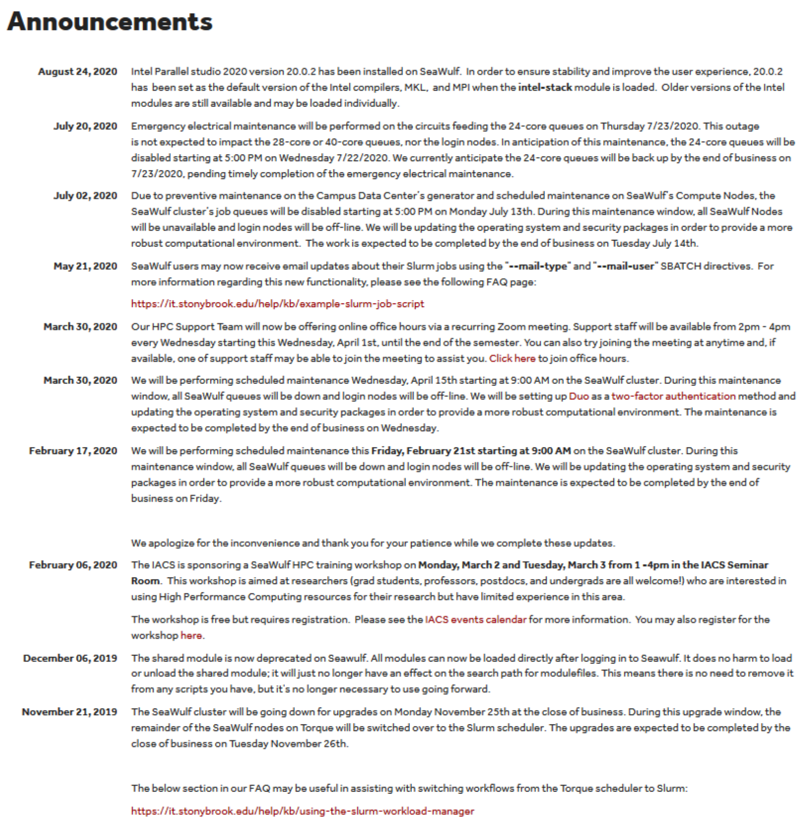
Once a node is available, you can issue commands on the command line

Good for troubleshooting,

Inefficient once your code is working

Need more help or information?

Check out our FAQ: https://it.stonybrook.edu/services/high-performance-computing



Announcements

August 24, 2020 Intel Parallel studio 2020 version 20.0.2 has been installed on SeaWulf. In order to ensure stability and improve the user experience, 20.0.2 has been set as the default version of the Intel compilers, MKL, and MPI when the **intel-stack** module is loaded. Older versions of the Intel modules are still available and may be loaded individually.

July 20, 2020 Emergency electrical maintenance will be performed on the circuits feeding the 24-core queues on Thursday 7/23/2020. This outage is not expected to impact the 28-core or 40-core queues, nor the login nodes. In anticipation of this maintenance, the 24-core queues will be disabled starting at 5:00 PM on Wednesday 7/22/2020. We currently anticipate the 24-core queues will be back up by the end of business on 7/23/2020, pending timely completion of the emergency electrical maintenance.

July 02, 2020 Due to preventive maintenance on the Campus Data Center's generator and scheduled maintenance on SeaWulf's Compute Nodes, the SeaWulf cluster's job queues will be disabled starting at 5:00 PM on Monday July 13th. During this maintenance window, all SeaWulf Nodes will be unavailable and login nodes will be off-line. We will be updating the operating system and security packages in order to provide a more robust computational environment. The work is expected to be completed by the end of business on Tuesday July 14th.

May 21, 2020 SeaWulf users may now receive email updates about their Slurm jobs using the "**--mail-type**" and "**--mail-user**" SBATCH directives. For more information regarding this new functionality, please see the following FAQ page:

https://it.stonybrook.edu/help/kb/example-slurm-job-script

March 30, 2020 Our HPC Support Team will now be offering online office hours via a recurring Zoom meeting. Support staff will be available from 2pm - 4pm every Wednesday starting this Wednesday, April 1st, until the end of the semester. You can also try joining the meeting at anytime and, if available, one of support staff may be able to join the meeting to assist you. Click here to join office hours.

March 30, 2020 We will be performing scheduled maintenance Wednesday, April 15th starting at 9:00 AM on the SeaWulf cluster. During this maintenance window, all SeaWulf queues will be down and login nodes will be off-line. We will be setting up Duo as a two-factor authentication method and updating the operating system and security packages in order to provide a more robust computational environment. The maintenance is expected to be completed by the end of business on Wednesday.

February 17, 2020 We will be performing scheduled maintenance this **Friday, February 21st starting at 9:00 AM** on the SeaWulf cluster. During this maintenance window, all SeaWulf queues will be down and login nodes will be off-line. We will be updating the operating system and security packages in order to provide a more robust computational environment. The maintenance is expected to be completed by the end of business on Friday.

We apologize for the inconvenience and thank you for your patience while we complete these updates.

February 06, 2020 The IACS is sponsoring a SeaWulf HPC training workshop on **Monday, March 2 and Tuesday, March 3 from 1 -4pm in the IACS Seminar Room**. This workshop is aimed at researchers (grad students, professors, postdocs, and undergrads are all welcome!) who are interested in using High Performance Computing resources for their research but have limited experience in this area.

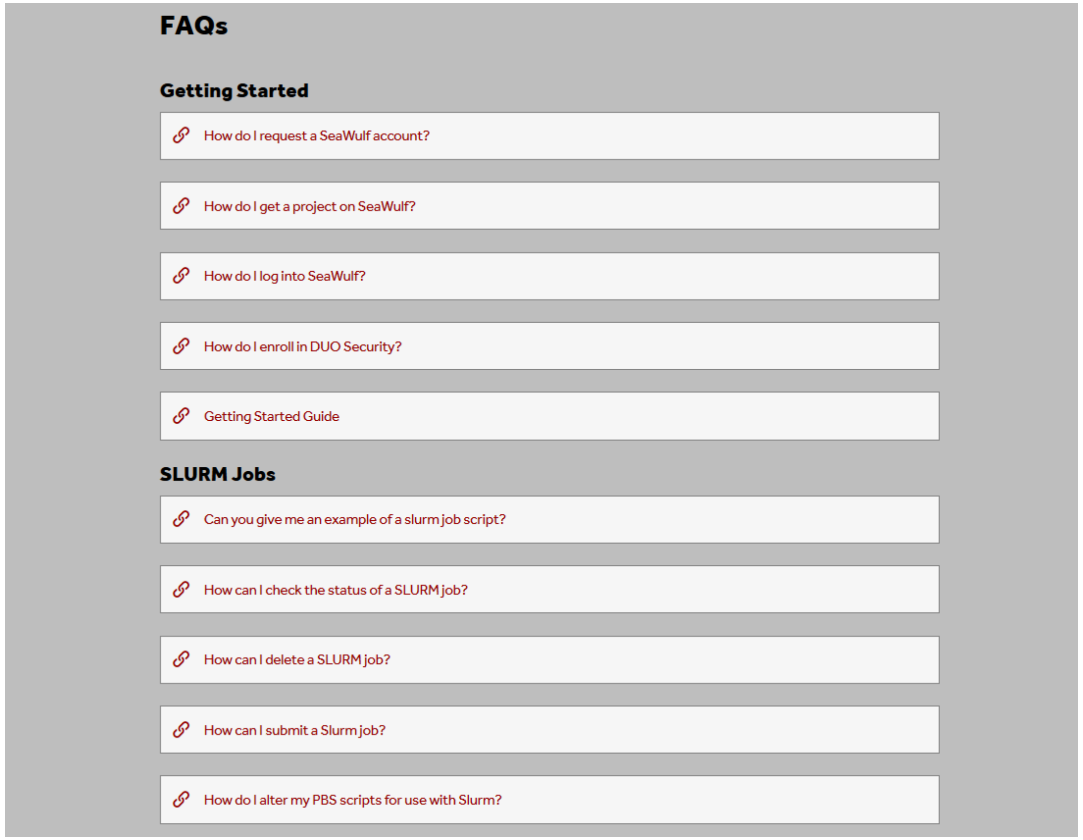
The workshop is free but requires registration. Please see the IACS events calendar for more information. You may also register for the workshop here.

December 06, 2019 The shared module is now deprecated on Seawulf. All modules can now be loaded directly after logging in to Seawulf. It does no harm to load or unload the shared module; it will just no longer have an effect on the search path for modulefiles. This means there is no need to remove it from any scripts you have, but it's no longer necessary to use going forward.

November 21, 2019 The SeaWulf cluster will be going down for upgrades on Monday November 25th at the close of business. During this upgrade window, the remainder of the SeaWulf nodes on Torque will be switched over to the Slurm scheduler. The upgrades are expected to be completed by the close of business on Tuesday November 26th.

The below section in our FAQ may be useful in assisting with switching workflows from the Torque scheduler to Slurm:

https://it.stonybrook.edu/help/kb/using-the-slurm-workload-manager



FAQs

Getting Started

- How do I request a SeaWulf account?
- How do I get a project on SeaWulf?
- How do I log into SeaWulf?
- How do I enroll in DUO Security?
- Getting Started Guide

SLURM Jobs

- Can you give me an example of a slurm job script?
- How can I check the status of a SLURM job?
- How can I delete a SLURM job?
- How can I submit a Slurm job?
- How do I alter my PBS scripts for use with Slurm?

Submit a ticket: https://iacs.supportsystem.com

QUESTIONS?