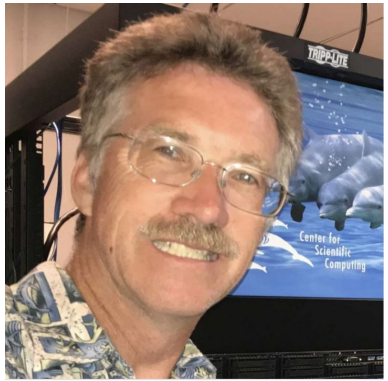
UC **SANTA BARBARA**


# Introduction to Center for Scientific Computing at UCSB

Paul Weakliem and Jay Chi

April 02, 2026

# Ye Olde People Introductions




Paul Weakliem, PhD
Co-Director
Center for Scientific Computing &
California Nanosystems Institute
Elings 3231
weakliem@cnsi.ucsb.edu

Fuzzy Rogers
That guy in the MRL
Center for Scientific Computing &
Materials Research Laboratory
MRL 2066B
fuz@ucsb.edu


Yu-Chieh “Jay” Chi, PhD
Research Computing Consultant
Center for Scientific Computing &
Information Technology Services
Elings 3213
jaychi@ucsb.edu

CSC Center for Scientific Computing

CNSI

UC SANTA BARBARA MRL Materials Research Laboratory
NSF Materials Research Science and Engineering Center [DMR 1720256]

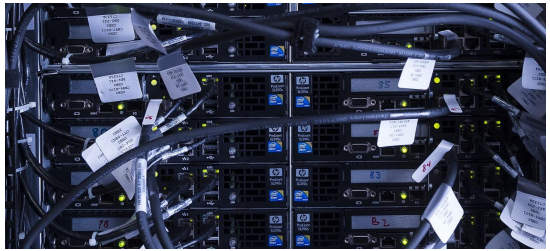
UC SANTA BARBARA Information Technology

# Center for Scientific Computing (CSC)

The tool for fairly large calculations is the High-Performance Computing (HPC) clusters at CSC.

Generally used for very large scale calculations, or many medium scale calculations. ***Primarily accessed via command line Interface(CLI)***

Center for Scientific Computing (CSC) Facilities



- High-Performance Computing (HPC) Cluster
  - Many Linux servers with high speed network
  - Specialize node - GPUs, Large Memory
  - Resource management done automatically via SLURM
  - Long running calculation - upwards of 30+ days
- Workshops, training, consultation, & knowledge bases to assist researchers
- More information: https://csc.cnsi.ucsb.edu

# CSC Cluster Resource

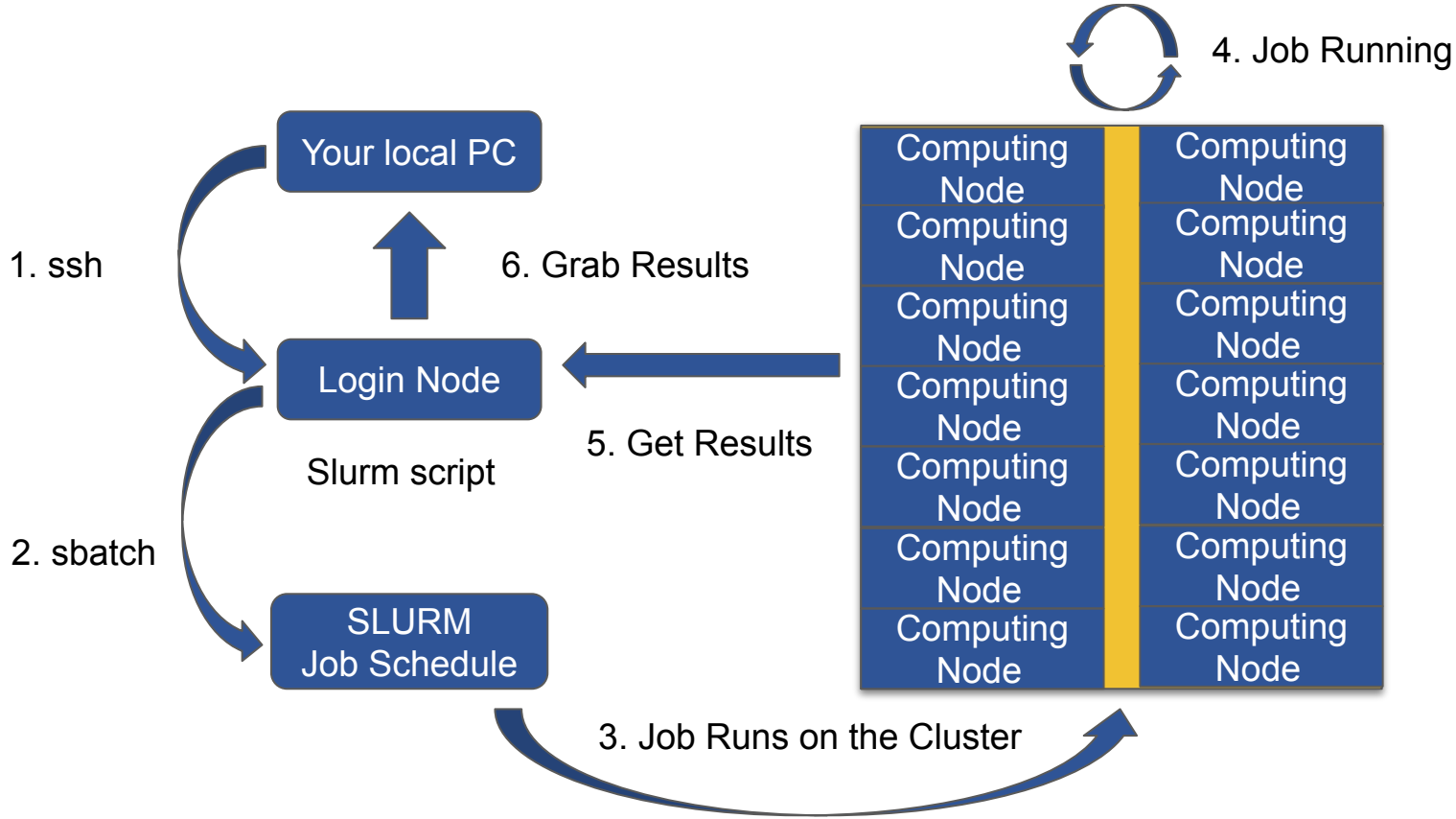
- CSC - POD (Pod of Dolphins) - pod.cnsi.ucsb.edu
  - FREE … but Campus available, usually pretty busy
    - Regular CPU nodes
    - GPU nodes (NVIDIA V100's)
    - Large memory nodes.
- CSC - Condo Clusters - Braid & Braid2 - braid.cnsi.ucsb.edu & braid2.cnsi.ucsb.edu
  - Individual Faculty Buy-In for access
    - Have regular, largemem, GPU nodes.
- ACCESS - National Supercomputing Centers
  - National Supercomputer Center - ACCESS-CI https://access-ci.org
  - PSC, SDSC, TACC, JetStream2, etc.
  - When hardware needs or calculations grow beyond CSC resources

# Cluster Use Cases

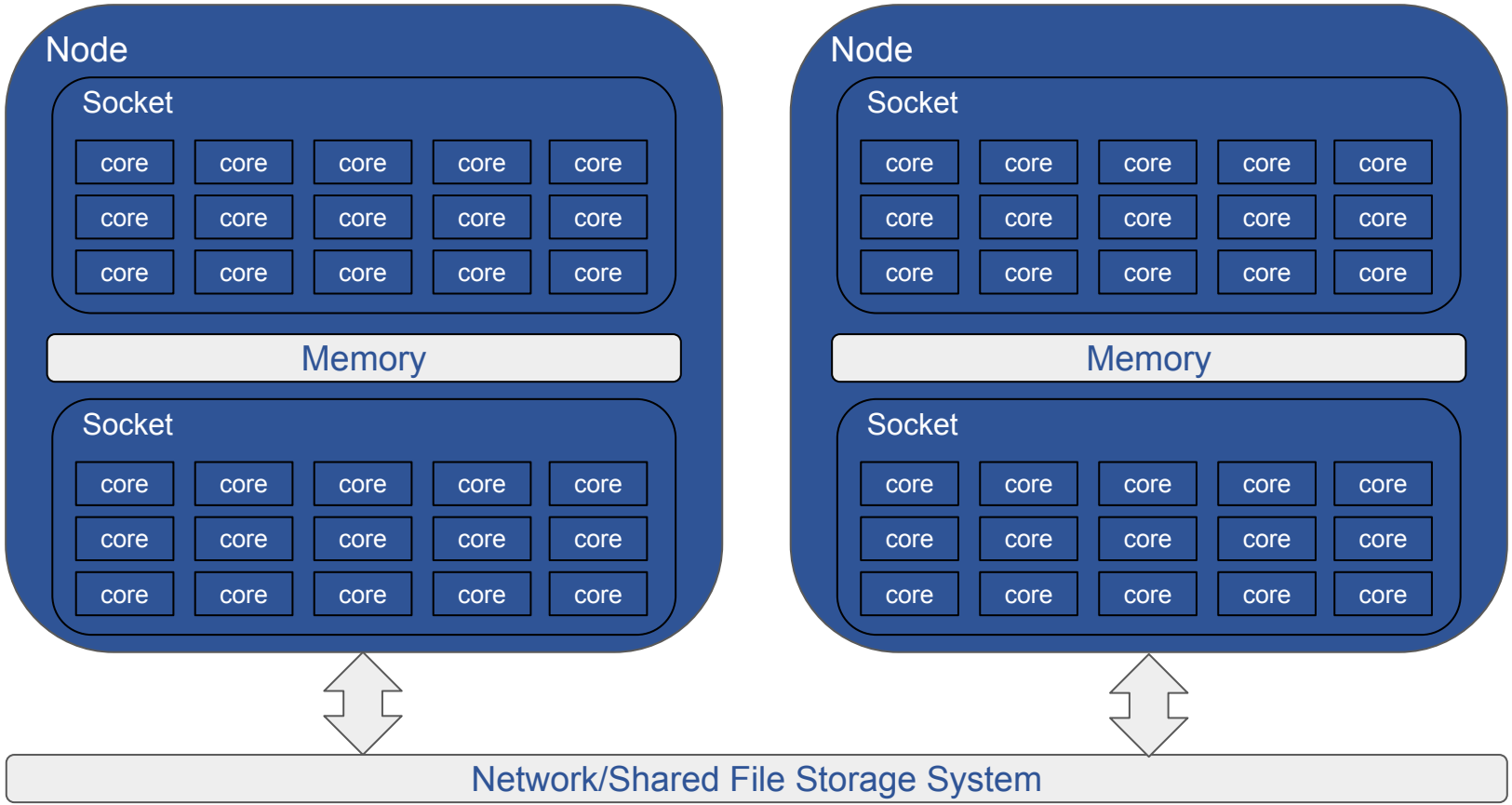
- Multiple calculation at once (parameter sweep), able to let resource manager (SLURM) handle the in-and-out of your calculations
- Request multiple cores and nodes (MPI) for your large-scale computing (e.g. need 100 cores for 5 days)

- Utilize standard software - Abaqus, Matlab, Mathematica, R, VASP, etc. (CSC can help with research group licensed software)
- Compile / Build software - GNU, Intel, CUDA, MPI … help install your software
- Conda/virtual Environment - add your own packages in your own environment
- Specialized hardware - ML/AI/NLP/LLM on GPUs which are typically programmed in …
- Compute on dataset in the TB size

# General HPC Workflow



# HPC Infrastructure

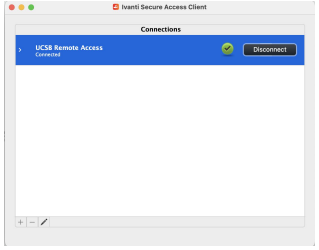


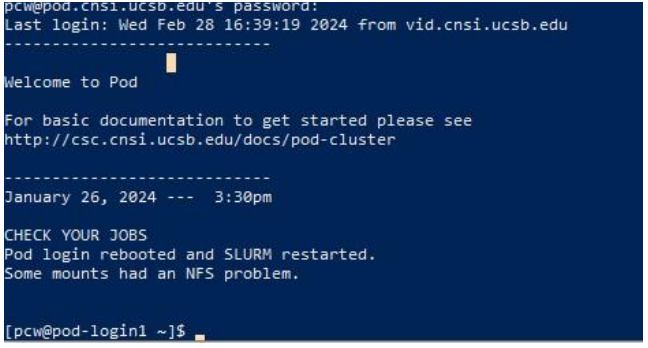
# Basic Linux Commands

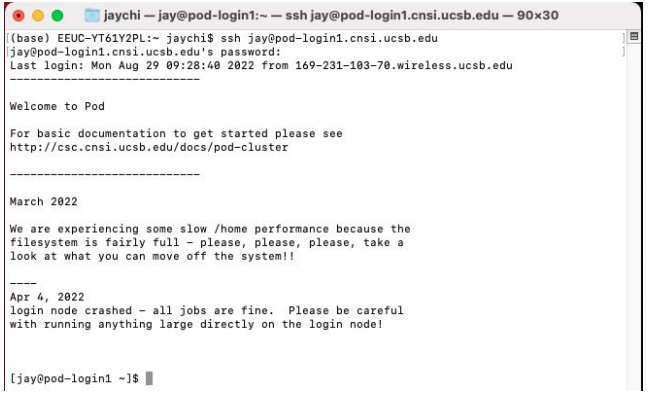
- Listing files (ls)
- Print Working Directory (pwd)
- Change Directory (cd)
- Make Directory (mkdir)
- Copy (cp)
- Moving Files (mv)
- Remove Files (rm)
- Secure Copy (scp)
- Display beginning/end of file (head/tail)
- View file (cat)
- Display manual for a command (man)
- nano, vim, or emacs to edit your file.

# Connecting to POD

- For the Windows system, open a ‘powershell’ window (type powershell in the search bar)
- For the Mac or Linux system, you can open the terminal (in Applications->Utilities)
- From either - use the ssh command
  - $ ssh your_user_name@pod.cnsi.ucsb.edu
- ***UCSB VPN is required to connect to the campus cluster from off-campus***
  - https://www.ets.ucsb.edu/pulse-secure-campus-vpn/get-connected-campus-vpn



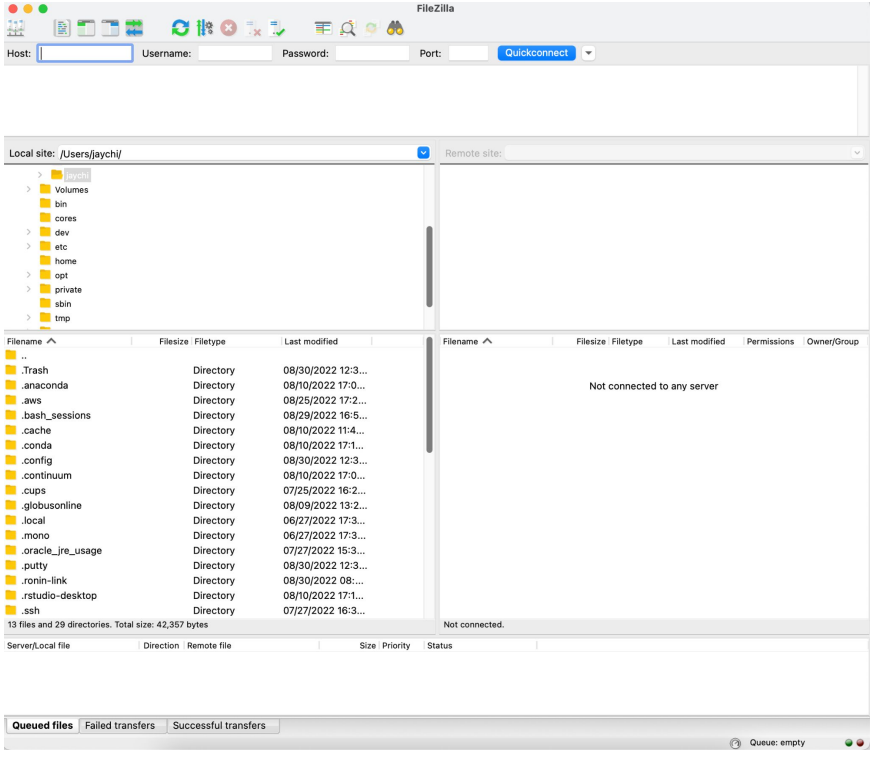



# File Transfer

- How do I uploaded data & download my files?
  - Graphical User Interface (GUI)
    - Filezilla: https://filezilla-project.org/
    - Cyberduck: https://cyberduck.io/
  - Command-Line Interface (CLI)
    - “scp” command
- FileZilla
  - Host: pod.cnsi.ucsb.edu
  - Username: your_user_name
  - Password: your_password
  - Port: 22
- Globus (for larger files transfers)
  - https://csc.cnsi.ucsb.edu/docs/globus-v5-new

**Filezilla**

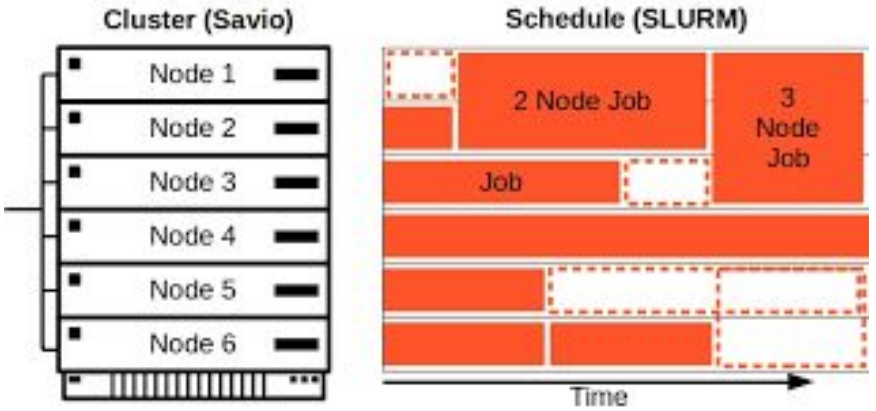


# Job Submission Script

- All jobs must be submitted to the queue - it just allocate nodes.
- Submission to the queue requires a job script to be written.
- Job script need to specify the resource that you need. There are three basic units:
  - Number of Nodes
  - Number of Cores
  - Time (Optional)
- Other resource you might need to add such as: job name, memory, reminder email, etc.

# Simple Scheduling Algorithms

- Backfilling
  - The scheduler maintains the "First Come, First Serve" concept without preventing long-running jobs from executing.
  - The scheduler checks whether the first job in the queue can be executed:
    - If true, the job is executed without further delay.
    - If false, the scheduler looks for the next job that can be executed without extending the waiting time of the first job in the queue and runs it.
  - Jobs that only need a few computing resources are easily “backfillable.”
    - Small jobs will usually encounter shorter queue times.




Ref: https://docs-research-it.berkeley.edu/services/high-performance-computing/user-guide/running-your-jobs/why-job-not-run/

# Simple Slurm Job Submission script

- There are three simple Slurm script files in your directory: slurm-mpi2.job, slurm-mpi.job, and slurm-serial.job.

Slurm job script file: slurm-serial.job

```
#!/bin/bash -l

#SBATCH —nodes=1 –ntasks-per-node 1

module load MatLab/R2021b

cd $SLURM_SUBMIT_DIR

/bin/hostname

matlab -nodisplay -nodesktop -nosplash <
my-inputfile.m
```

Slurm job script file: slurm-mpi.job

```
#!/bin/bash -l

#SBATCH —nodes=1 –ntasks-per-node 12

module load intel/18

cd $SLURM_SUBMIT_DIR

/bin/hostname

mpirun -np $SLURM_NPROCS ./a.out >& logfile
```

$ sbatch slurm-serial.job ‘or’ $ sbatch -p short slurm-serial.job

# Example Slurm Job Submission script (ORCA)

Slurm job script file: job.s

```
#!/bin/bash                                  ### Set linux shell: Telling the shell to run the script using the batch
#SBATCH -J 'testJob'                         ### Job Name
#SBATCH --nodes=1                            ### No. of Nodes
#SBATCH --ntasks=4                           ### No. of Tasks
# SBATCH -p batch                            ### Running on the CPU Node
#SBATCH -o outLog                            ### Output Log File (Optional)
#SBATCH -e errLog                            ### Error Log File (Optional but suggest to have it)
#SBATCH -t 00:10:00                          ### Job Execution Time
#SBATCH --mail-user=usernam@ucsb.edu         ### Mail to you (Optional)
#SBATCH --mail-type ALL                      ### Mail send you when the job starts and end (Optional)

module purge all
module load chem/orca/6.1.0
cd $SLURM_SUBMIT_DIR/                        ### Absolute path of the current working directory when you submit the job

ulimit -s unlimited                          ###  Needed for many ORCA runs (if you see segfault, etc. errors, try this)

/sw/chem/orca/orca_6.1.0/bin/orca YOUR_ORCA_INPUT.inp > log
```

# Example Slurm Job Submission script

Slurm job script file: job.s

```
#!/bin/bash                          ### Set linux shell: Telling the shell to run the script using the batch
#SBATCH -J 'testJob'                 ### Job Name
#SBATCH --nodes=1                    ### No. of Nodes
#SBATCH --ntasks=1                   ### No. of Tasks
#SBATCH -p gpu                       ### Submit the job to Partition
#SBATCH –gres=gpu:1                  ### Request 1 GPU
#SBATCH -o outLog                    ### Output Log File (Optional)
#SBATCH -e errLog                    ### Error Log File (Optional but suggest to have it)
#SBATCH -t 00:10:00                  ### Job Execution Time
#SBATCH --mail-user=usernam@ucsb.edu ### Mail to you (Optional)
#SBATCH --mail-type ALL              ### Mail send you when the job starts and end (Optional)

module purge all
module load anaconda                 ### Load softwares that the job depends on to execute
source activate my_env_1

cd $SLURM_SUBMIT_DIR/                ### Absolute path of the current working directory when you submit the job

python my_python.py
```

# How to Submit and Monitor Your Job

- Once you have a job script, you may submit this script to SLURM using the sbatch command. SLURM will find an available compute node or set of compute nodes and run your job there, or leave your job in a queue until some resources become available.

```
$ sbatch job.s
Submitted batch job 1234567
```

- List all current jobs from the user.

```
$ squeue -u your_user_name
$ squeue -u $USER
$ showq your_user_name
```

- Stop and delete the Job

```
$ scancel 1234567
```

# How to Submit and Monitor Your Job

- List all partitions on the cluster

  $ sinfo

```
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
batch*       up 32-00:00:0      1  down* node1
batch*       up 32-00:00:0      1   drng node20
batch*       up 32-00:00:0      1  drain node4
batch*       up 32-00:00:0     30    mix node[6,8-9,11,13-16,19,22-23,26-28,32,36-39,42-44,49,52-56,58,60]
batch*       up 32-00:00:0     29  alloc node[3,5,7,10,12,17-18,21,24-25,29-31,33-35,40-41,45-48,50-51,57,59,61-63]
batch*       up 32-00:00:0      1   idle node2
short        up    2:00:00      1    mix node64
largemem     up 37-12:00:0      4    mix node[101-104]
gpu          up 7-00:00:00      1  down* node117
gpu          up 7-00:00:00     13    mix node[111-113,115-116,118-125]
```

- List the partition who are using

  $ squeue -p short

- Report the job expected start time

  $ squeue --start -j job_ID

# Running Jobs on Pod (Slurm)

- Start/submit a job: $ sbatch job.s
- Check status of the running jobs: $ squeue -u user_name

  $ showq user_name
- Delete a running job: $ scancel job_id

- Available partition:
  - Short partition: running under 2 hrs
    - #SBATCH -p short
  - Large memory partition: running the longest 37 days
    - #SBATCH -p largemem
  - GPU partition: running the longest 10 days
    - #SBATCH -p gpu

# Questions and Thought

- More Information:

  https://csc.cnsi.ucsb.edu/