Intro to Job Scheduling

Introduction: This workshop will introduce users to job management using the Slurm system - demonstrating how to create interactive jobs and submit jobs to the cluster queue that follow a basic workflow. After the workshop, participants will understand:

- How to create a script that defines their workflow (i.e. loading modules).
- Understand how to start interactive sessions to work within, as well as how to submit and track jobs on the cluster.
- Participants will require an intro level of experience of using Linux, as well as the ability to use a text editor from the command line.

Course Goals:

- What is Slurm?
- How to start an Interactive sessions, and perform job submission
- How to select appropriate resource allocations.
- How to monitor your jobs.
- What does a general workflow look like?
- Best practices in using HPC.
- How to be a good cluster citizen?

Sections:

1. What is Slurm: Introduction to Slurm and how to start interactive sessions, submit jobs and monitor.
2. Slurm: Workflows and Best practices: Discuss what workflows can look like, being a good cluster citizen, and some best practices.
3. Slurm: More Features: Introduce some further features, such as job efficiency and cluster utilization.
4. Slurm: Common Issues and How to Resolve: List some common issues and how to resolve.

What is Slurm

Goal: Introduction to Slurm and how to start interactive sessions, submit jobs and monitor.

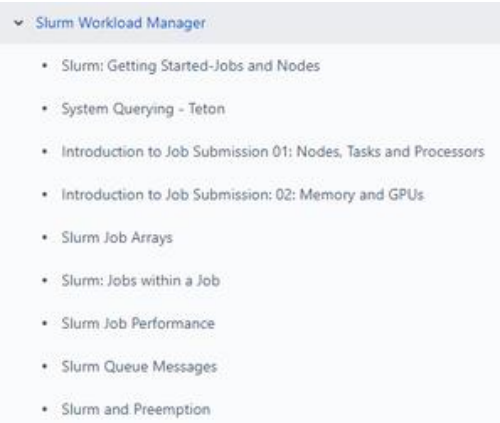
- Workload Managers
- Interactive Session: salloc

- Submit Jobs: sbatch
- Submit Jobs: squeue: What's happening?
- Submit Jobs: scancel: Cancel?
- Submit Jobs: sacct: What happened?
- Submit Jobs: sbatch: Options

Workload Managers

1. Allocates access to appropriate computer nodes specific to your requests.
2. Framework for starting, executing, monitoring, and even canceling your jobs.
3. Queue management and job state notification.

ARCC: Slurm: Wiki Pages



Slurm Related Commands

- Slurm commands:
  - Start an interactive session: salloc
  - Submit a job: sbatch
  - Monitor jobs:
    - squeue
    - sacct
    - sinfo

        - arccjobs
- Core hour usage: chu_user, chu_account

Interactive Session: `salloc`

- You're *there* doing the work.
- Suitable for developing and testing over a few hours.

```
[]$ salloc --help
# Lots of options.
# Notice short and long form options.
[]$ salloc –A <project-name> -t <wall-time>
# Format for: --time: Acceptable time formats include "minutes",
"minutes:seconds",
"hours:minutes:seconds", "days-hours", "days-hours:minutes" and "days-
hours:minutes:seconds".
```

Interactive Session: `salloc`: workshop

- You'll only use the `reservation` for this (and/or other) workshop.
- Once you have an account you typically do not need it.
- But there are use cases when we can create a specific reservation for you.

```
[]$ salloc –A arccanetrain –t 1:00 --reservation=<reservation-name>
```

Interactive Session: `squeue`: What's happening?

```
[]$ salloc -A arccanetrain -t 1:00 --reservation=<reservation-name>
salloc: Granted job allocation 13526337
salloc: Nodes m233 are ready for job
# Make a note of the job id.
# Notice the server/node name has changed.
[arcc-t05@m233 intro_to_hpc]$ squeue -u arcc-t05
             JOBID PARTITION     NAME     USER ST       TIME  NODES
NODELIST(REASON)
          13526337     moran interact arcc-t05  R       0:19      1 m233
# For an interactive session: Name = interact
# You have the command-line interactively available to you.
[]$
...
[]$ squeue -u arcc-t05
             JOBID PARTITION     NAME     USER ST       TIME  NODES
NODELIST(REASON)
          13526337     moran interact arcc-t05  R       1:03      1 m233
# Session will automatically time out
[]$ salloc: Job 13526337 has exceeded its time limit and its allocation has
been revoked.
slurmstepd: error: *** STEP 13526337.interactive ON m233 CANCELLED AT 2024-
03-22T09:36:53 DUE TO TIME LIMIT ***
```

```
exit
srun: Job step aborted: Waiting up to 32 seconds for job step to finish.
```

Interactive Session: `salloc`: Finished Early?

```
[]$ salloc -A arccanetrain -t 1:00 --reservation=<reservation-name>
salloc: Granted job allocation 13526338
salloc: Nodes m233 are ready for job
[arcc-t05@m233 ...]$ Do stuff…
[]$ exit
exit
salloc: Relinquishing job allocation 13526338
```

Submit Jobs: `sbatch`

- You submit a job to the queue and **walk away**.
- **Monitor its progress/state** using command-line and/or email notifications.
- Once complete, **come back** and analyze results.

Submit Jobs: `sbatch`: Template

```
#!/bin/bash                                 # Shebang indicating this is a bash
script.
#SBATCH --account=arccanetrain              # Use #SBATCH to define Slurm
related values.
#SBATCH --time=10:00                        # Must define an account and wall-
time.
#SBATCH --reservation=<reservation-name>
echo "SLURM_JOB_ID:" $SLURM_JOB_ID          # Can access Slurm related
Environment variables.
start=$(date +'%D %T')                      # Can call bash commands.
echo "Start:" $start
module load gcc/12.2.0 python/3.10.6        # Load the modules you require for
your environment.
python python01.py                          # Call your scripts/commands.
sleep 1m
end=$(date +'%D %T')
echo "End:" $end
```

Submit Jobs: `squeue`: What's happening?

```
[]$ sbatch run.sh
Submitted batch job 13526340
[]$ squeue -u arcc-t05
             JOBID PARTITION     NAME     USER ST       TIME  NODES
NODELIST(REASON)
          13526340     moran   run.sh arcc-t05  R       0:05      1 m233
[]$ ls
python01.py  run.sh  slurm-13526340.out
```

```
# You can view this file while the job is still running.
[]$ cat slurm-13526340.out
SLURM_JOB_ID: 13526340
Start: 03/22/24 09:38:36
Python version: 3.10.6 (main, Oct 17 2022, 16:47:32) [GCC 12.2.0]
Version info: sys.version_info(major=3, minor=10, micro=6,
releaselevel='final', serial=0)
[]$ squeue -u arcc-t05
             JOBID PARTITION     NAME     USER ST       TIME  NODES
NODELIST(REASON)
          13526340     moran   run.sh arcc-t05  R       0:17      1 m233
```

Submit Jobs: `squeue`: What's happening Continued?

```
[]$ squeue -u arcc-t05
             JOBID PARTITION     NAME     USER ST       TIME  NODES
NODELIST(REASON)
          13526340     moran   run.sh arcc-t05  R       0:29      1 m233
[]$ squeue -u arcc-t05
             JOBID PARTITION     NAME     USER ST       TIME  NODES
NODELIST(REASON)
# squeue only shows pending and running jobs.
# If a job is no longer in the queue then it has finished.
# Finished can mean success, failure, timeout... It's just no longer running.
[]$ cat slurm-13526340.out
SLURM_JOB_ID: 13526340
Start: 03/22/24 09:38:36
Python version: 3.10.6 (main, Oct 17 2022, 16:47:32) [GCC 12.2.0]
Version info: sys.version_info(major=3, minor=10, micro=6,
releaselevel='final', serial=0)
End: 03/22/24 09:39:36
```

Submit Jobs: `scancel`: Cancel?

```
[]$ sbatch run.sh
Submitted batch job 13526341
[]$ squeue -u arcc-t05
             JOBID PARTITION     NAME     USER ST       TIME  NODES
NODELIST(REASON)
          13526341     moran   run.sh arcc-t05  R       0:03      1 m233
[]$ scancel 13526341
[]$ squeue -u arcc-t05
             JOBID PARTITION     NAME     USER ST       TIME  NODES
NODELIST(REASON)
[]$ cat slurm-13526341.out
SLURM_JOB_ID: 13526341
Start: 03/22/24 09:40:09
Python version: 3.10.6 (main, Oct 17 2022, 16:47:32) [GCC 12.2.0]
Version info: sys.version_info(major=3, minor=10, micro=6,
releaselevel='final', serial=0)
slurmstepd: error: *** JOB 13526341 ON m233 CANCELLED AT 2024-03-22T09:40:17
***
```

Submit Jobs: sacct: What happened?

```
[]$ sacct -u arcc-t05 -X
JobID           JobName  Partition    Account  AllocCPUS      State ExitCode
------------ ---------- ---------- ---------- ---------- ---------- --------
13526337     interacti+      moran arccanetr+          1    TIMEOUT      0:0
13526338     interacti+      moran arccanetr+          1  COMPLETED      0:0
13526340         run.sh      moran arccanetr+          1  COMPLETED      0:0
13526341         run.sh      moran arccanetr+          1 CANCELLED+      0:0
# Lots more information
[]$ sacct --help
[]$ sacct -u arcc-t05 --
format="JobID,Partition,nnodes,NodeList,NCPUS,ReqMem,State,Start,Elapsed" -X
JobID         Partition   NNodes        NodeList      NCPUS     ReqMem
State               Start    Elapsed
------------ ---------- -------- --------------- ---------- ---------- ------
---- ------------------- ----------
13526337          moran        1            m233          1      1000M
TIMEOUT 2024-03-22T09:35:25   00:01:28
13526338          moran        1            m233          1      1000M
COMPLETED 2024-03-22T09:37:41   00:00:06
13526340          moran        1            m233          1      1000M
COMPLETED 2024-03-22T09:38:35   00:01:01
13526341          moran        1            m233          1      1000M
CANCELLED+ 2024-03-22T09:40:08   00:00:09
```

Submit Jobs: sbatch: Options

```
[]$ sbatch --help
#SBATCH --account=arccanetrain          # Required: account/time
#SBATCH --time=72:00:00
#SBATCH --job-name=workshop             # Job name: Help to identify when
using squeue.
#SBATCH --nodes=1                       # Options will typically have
defaults.
#SBATCH --tasks-per-node=1              # Request resources in accordance to
how you want
#SBATCH --cpus-per-task=1               # to parallelize your job, type of
hardware partition
#SBATCH --partition=teton-gpu           # and if you require a GPU.
#SBATCH --gres=gpu:1
#SBATCH --mem=100G                      # Request specific memory needs.
#SBATCH --mem-per-cpu=10G
#SBATCH --mail-type=ALL                 # Get email notifications of the
state of the job.
#SBATCH --mail-user=<email-address>
#SBATCH --output=<prefix>_%A.out        # Define a named output file
postfixed with the job id.
```

Slurm: Workflows and Best Practices

Goal: Discuss what workflows can look like, being a good cluster citizen, and some best practices.

If you don’t ask, you don’t get: GPU Example

```
#!/bin/bash
#SBATCH --account=arccanetrain
#SBATCH --time=1:00
#SBATCH --reservation=HPC_workshop
#SBATCH --partition=teton-gpu
#SBATCH --gres=gpu:1
echo "SLURM_JOB_ID:" $SLURM_JOB_ID
echo "SLURM_GPUS_ON_NODE:" $SLURM_GPUS_ON_NODE
echo "SLURM_JOB_GPUS:" $SLURM_JOB_GPUS
echo "CUDA_VISIBLE_DEVICES:" $CUDA_VISIBLE_DEVICES
nvidia-smi -L
# Output:
SLURM_JOB_ID: 13517905
SLURM_GPUS_ON_NODE: 1
SLURM_JOB_GPUS: 0
CUDA_VISIBLE_DEVICES: 0
GPU 0: Tesla P100-PCIE-16GB (UUID: GPU-c1859587-9722-77f3-1b3a-63e9d4fe9d4f)
```

If you don’t ask, you don’t get: No GPU device requested

```
# Comment out the gres option.
##SBATCH --gres=gpu:1
# Output:
SLURM_JOB_ID: 13517906
SLURM_GPUS_ON_NODE:
SLURM_JOB_GPUS:
CUDA_VISIBLE_DEVICES:
No devices found.
```

Just because a partition/compute node has *something*,

you still need to explicitly request it.

Modules and using salloc and sbatch

Typically: Modules loaded, and environment variables that have been set on the login nodes will be inherited when you create an interactive `salloc` session and or call an `sbatch`.

```
[salexan5@mblog1 ~]$ module load gcc/13.2.0 r/4.4.0
[salexan5@mblog1 ~]$ ml
Currently Loaded Modules:
  1) slurm/latest           (S)  42) libxau/1.0.8
...
 41) xproto/7.0.31

[salexan5@mblog1 ~]$ salloc -A arcc -t 10:00
salloc: Granted job allocation 1243593
salloc: Nodes mbcpu-025 are ready for job
[salexan5@mbcpu-025 ~]$ ml
Currently Loaded Modules:
  1) slurm/latest    (S)  15) libxml2/2.10.3          29) perl/5.38.0
43) libxdmcp/1.1.4      57) curl/8.4.0                   71)
openjdk/11.0.20.1_1
 ...
 14) xz/5.4.1             28) gdbm/1.23                42) libxau/1.0.8
56) nghttp2/1.57.0      70) openblas/0.3.24
```

Modules and using salloc and sbatch: Best Practice

Although modules and environment variables are typically inherited, this is not good practice since we have observed cases where not everything has been inherited.

Also, when ARCC is asked to assist, typically we have no idea, and users forget, how an environment has been setup on a login node.

Best Practice: After performing an `salloc`, or within the script you `sbatch`-ed, perform a `module purge` and then only `module load` (including versions) what you explicitly know you need to use.

When requesting ARCC help, this is then documented within your scripts that are `sbatch`-ed helping us to exactly replicate what you've done.

This is good reproducibility practice.

What does a general workflow look like?

Getting Started:

- Understand your application / programming language.
- What are its capabilities / functionality.
- Read the documentation, find examples, online forums – community.

Develop/Try/Test:

- Typically use an ***interactive session (salloc)*** where you're typing/trying/testing.
- Are modules available? If not submit a HPC Software Consultation request to start the discussion.
- Develop code/scripts.
- Understand how the command-line works – what commands/scripts to call with options.
- Understand if parallelization is available – can you optimize your code/application?
- Test against a subset of data. Something that runs *quick* – maybe a couple of minutes/hours.
- Do the results look correct?

What does a general workflow look like? Continued.

Production:

- Put it all together within a bash Slurm script:
  - Request appropriate resources using `#SBATCH`
  - Request appropriate wall time – hours, days…
  - Load modules: `module load …`
  - Run scripts/command-line.
- Finally, ***submit your job to the cluster (sbatch)*** using a complete set of data.
  - Use: `sbatch <script-name.sh>`
  - Monitor job(s) progress.

What does it mean for an application to be parallel?

Read the documentation and look at the command's help: Does it mention:

- **Threads** - multiple cpus/cores: Single node, single task, multiple cores.
  - Example: Chime
- **OpenMP**: Single task, multiple cores. Set environment variable.
  - an application programming interface (API) that supports multi-platform shared-memory multiprocessing programming in C, C++, and Fortran.
  - Example: ImageMagick
- **MPI**: Message Passing Interface: Multiple nodes, multiple tasks
  - OpenMPI: ARCC Wiki: OpenMPI and oneAPI Compiling,
- Hybrid: MPI / OpenMP and/or threads.
  - Examples: DFTB and Quantum Espresso

What does it mean for an application to be GPU enabled?

Read the documentation and look at the command's help: Does it mention:

- GPU / Nvidia / Cuda?
- Examples:
  - Applications: AlphaFold and GPU Blast
    - Via conda based environments built with GPU libraries - and converted to Jupyter kernels:
    - Examples: TensorFlow and PyTorch
    - Jupyter Kernels: PyTorch 1.13.1

How can I be a good cluster citizen?

- Policies
- **Don't** run intensive applications on the login nodes.
- Understand your software/application.
- Shared resource - multi-tenancy.
  - Other jobs running on the same node do not affect each other.
- Don't ask for *everything*. Don't use:
  - mem=0
  - exclusive tag.
  - Only ask for a GPU if you know it'll be used.
- Use `/lscratch` for I/O intensive tasks rather than accessing `/gscratch` over the network.
  - You will need to copy files back before the job ends.
- Track usage and job performance: `seff <jobid>`

Being a good Cluster Citizen: Requesting Resources

Good Cluster Citizen:

- Only request what you need.
- Unless you know your application:
  - can utilize multiple nodes/tasks/cores, request a single node/task/core (default).
  - can utilize multiple nodes/tasks/cores, requesting them will not make your code *magically* run faster.
  - is GPU enabled, having a GPU will not make your code *magically* run faster.
- Within your application/code check that resources are actually being detected and utilized.
  - Look at the job efficiency: job performance: `seff <jobid>`
  - This is emailed out if you have Slurm email notifications turned on.
- Slurm cheatsheet

Slurm: More Features

Goal: Introduce some further features, such as job efficiency and cluster utilization.

- Job Efficiency
- What's the Current Cluster Utilization?

Job Efficiency

```
[]$ seff 13515489
Job ID: 13515489
Cluster: beartooth
User/Group: salexan5/salexan5
State: COMPLETED (exit code 0)
Cores: 1
CPU Utilized: 00:00:05
CPU Efficiency: 27.78% of 00:00:18 core-walltime
Job Wall-clock time: 00:00:18
Memory Utilized: 0.00 MB (estimated maximum)
Memory Efficiency: 0.00% of 8.00 GB (8.00 GB/node)
```

Note:

- Only accurate is the job is successful.
- If the job fails with say an OOM: Out-Of-Memory the details will be inaccurate.
- This is emailed out if you have Slurm email notifications turned on.

What's the Current Cluster Utilization?

There are a number of ways to see the current status of the cluster:

- `arccjobs`: Prints a table showing active projects and jobs.
- `pestat`: Prints a nodes list with allocated jobs - can query individual nodes.
- `sinfo`: View the status of the Slurm partitions or nodes. Status of nodes that are drained can be seen using the -R flag.
- OnDemand's MedicineBow System Status page.

arccjobs example

```
[]$ arccjobs
================================================================================
Account                       Running                       Pending
  User                  jobs     cpus          cpuh    jobs     cpus
cpuh
================================================================================
eap-amadson              500      500         30.42       3        3
2.00
  amadson                500      500         30.42       3        3
2.00

eap-larsko                 1       32       2262.31       0        0
0.00
  fghorban                 1       32       2262.31       0        0
0.00

pcg-llps                   2       64       1794.41       0        0
0.00
  hbalantr                 1       32        587.68       0        0
0.00
  vvarenth                 1       32       1206.73       0        0
0.00

================================================================================
TOTALS:                  503      596       4087.14       3        3
2.00
================================================================================
Nodes                      9/51       (17.65%)
```

```
Cores                       596/4632    (12.87%)
Memory (GB)               2626/46952    ( 5.59%)
CPU Load                  803.43        (17.35%)
================================================================================
==
```

pestat example

```
[salexan5@mblog1 ~]$ pestat
Hostname          Partition     Node Num_CPU  CPUload  Memsize  Freemem
Joblist
                                State Use/Tot  (15min)     (MB)     (MB)
JobID(JobArrayID) User ...
mba30-001            mb-a30    idle    0  96    0.00    765525   749441
mba30-002            mb-a30    idle    0  96    0.00    765525   761311
mba30-003            mb-a30    idle    0  96    0.00    765525   761189
mba30-004            mb-a30    idle    0  96    0.00    765525   761093
mba30-005            mb-a30    idle    0  96    0.00    765525   761357
mba30-006            mb-a30    idle    0  96    0.00    765525   761307
mba30-007            mb-a30    idle    0  96    0.00    765525   761264
mba30-008            mb-a30    idle    0  96    0.00    765525   761184
mba6000-001        mb-a6000    idle    0  64    0.00   1023575  1020075
mbcpu-001               mb*    idle    0  96    0.00   1023575  1019621
mbcpu-002               mb*    idle    0  96    3.18*  1023575   479265
mbcpu-003               mb*    idle    0  96    2.56*  1023575   609488
mbcpu-004               mb*    idle    0  96    0.00   1023575  1019530
mbcpu-005               mb*    idle    0  96    0.38   1023575   280581
mbcpu-006               mb*    idle    0  96    2.06*  1023575   106400*
mbcpu-007               mb*    idle    0  96    1.60*  1023575   369035
mbcpu-008               mb*    idle    0  96    2.07*  1023575    38900*
mbcpu-009               mb*    idle    0  96    0.00   1023575  1019394
mbcpu-010               mb*    idle    0  96    0.00   1023575  1019942
mbcpu-011               mb*    idle    0  96    0.00   1023575   315295
mbcpu-012               mb*    idle    0  96   12.05*  1023575   397291
mbcpu-013               mb*    idle    0  96    2.47*  1023575   538938
mbcpu-014               mb*    idle    0  96   12.70*  1023575   953417
mbcpu-015               mb*     mix   22  96   35.56*  1023575  1002195
1544825 amadson 1544826 amadson 1544824 amadson 1544772 amadson 1544773
amadson 1544708 amadson 1544709 amadson 1544691 amadson 1544692 amadson
1544686 amadson 1544656 amadson 1544646 amadson 1544644 amadson 1544624
amadson 1544622 amadson 1544609 amadson 1544610 amadson 1544592 amadson
1544588 amadson 1544584 amadson 1544573 amadson 1544530 amadson
mbcpu-016               mb*    idle    0  96    0.17   1023575  1019397
mbcpu-017               mb*    idle    0  96    0.00   1023575  1019767
mbcpu-018               mb*    idle    0  96    0.00   1023575  1019609
mbcpu-019               mb*    idle    0  96    0.00   1023575  1019667
mbcpu-020               mb*   maint*   0  96    0.00   1023575  1019991
mbcpu-021               mb*     mix   94  96  143.19*  1023575    69205*
1541097 amadson 1541072 amadson 1541044 amadson 1541031 amadson 1541016
amadson 1544276 amadson 1544221 amadson 1543933 amadson 1544830 amadson
1544829 amadson 1544827 amadson 1544821 amadson 1544815 amadson 1544811
amadson 1544812 amadson 1544802 amadson 1544804 amadson 1544800 amadson
1544792 amadson 1544786 amadson 1544787 amadson 1544788 amadson 1544775
amadson 1544764 amadson 1544748 amadson 1544749 amadson 1544745 amadson
1544741 amadson 1544743 amadson 1544736 amadson 1544727 amadson 1544724
amadson 1544721 amadson 1544717 amadson 1544719 amadson 1544700 amadson
1544701 amadson 1544689 amadson 1544681 amadson 1544682 amadson 1544683
amadson 1544680 amadson 1544677 amadson 1544668 amadson 1544666 amadson
```

1544648 amadson 1544640 amadson 1544634 amadson 1544635 amadson 1544631
amadson 1544625 amadson 1544615 amadson 1544616 amadson 1544617 amadson
1544613 amadson 1544614 amadson 1544595 amadson 1544594 amadson 1544591
amadson 1544581 amadson 1544578 amadson 1544567 amadson 1544564 amadson
1544566 amadson 1544555 amadson 1544556 amadson 1544557 amadson 1544551
amadson 1544546 amadson 1544548 amadson 1544543 amadson 1544541 amadson
1544542 amadson 1544532 amadson 1544505 amadson 1544501 amadson 1544502
amadson 1544499 amadson 1544490 amadson 1544492 amadson 1544484 amadson
1544487 amadson 1544473 amadson 1544467 amadson 1544468 amadson 1544461
amadson 1544454 amadson 1544433 amadson 1544424 amadson 1544426 amadson
1544409 amadson 1544381 amadson 1544378 amadson 1544368 amadson
mbcpu-022 mb* alloc 96 96 143.42* 1023575 100423*
1541136 amadson 1541134 amadson 1541104 amadson 1544275 amadson 1544271
amadson 1544817 amadson 1544809 amadson 1544810 amadson 1544798 amadson
1544791 amadson 1544789 amadson 1544782 amadson 1544779 amadson 1544776
amadson 1544766 amadson 1544768 amadson 1544760 amadson 1544746 amadson
1544744 amadson 1544742 amadson 1544740 amadson 1544731 amadson 1544726
amadson 1544716 amadson 1544718 amadson 1544713 amadson 1544710 amadson
1544704 amadson 1544705 amadson 1544706 amadson 1544699 amadson 1544703
amadson 1544696 amadson 1544698 amadson 1544688 amadson 1544684 amadson
1544678 amadson 1544675 amadson 1544673 amadson 1544671 amadson 1544670
amadson 1544669 amadson 1544665 amadson 1544664 amadson 1544652 amadson
1544650 amadson 1544647 amadson 1544649 amadson 1544643 amadson 1544645
amadson 1544641 amadson 1544626 amadson 1544619 amadson 1544611 amadson
1544608 amadson 1544604 amadson 1544602 amadson 1544596 amadson 1544582
amadson 1544575 amadson 1544576 amadson 1544574 amadson 1544568 amadson
1544565 amadson 1544561 amadson 1544558 amadson 1544552 amadson 1544547
amadson 1544540 amadson 1544536 amadson 1544538 amadson 1544535 amadson
1544519 amadson 1544521 amadson 1544510 amadson 1544512 amadson 1544503
amadson 1544506 amadson 1544493 amadson 1544489 amadson 1544485 amadson
1544478 amadson 1544479 amadson 1544483 amadson 1544472 amadson 1544470
amadson 1544458 amadson 1544453 amadson 1544449 amadson 1544440 amadson
1544441 amadson 1544415 amadson 1544402 amadson 1544398 amadson 1544370
amadson 1544314 amadson
mbcpu-023 mb* alloc 96 96 134.43* 1023575 43715*
1544292 amadson 1544289 amadson 1544267 amadson 1544820 amadson 1544822
amadson 1544823 amadson 1544819 amadson 1544813 amadson 1544801 amadson
1544803 amadson 1544805 amadson 1544796 amadson 1544784 amadson 1544785
amadson 1544781 amadson 1544783 amadson 1544780 amadson 1544778 amadson
1544777 amadson 1544762 amadson 1544763 amadson 1544758 amadson 1544759
amadson 1544756 amadson 1544752 amadson 1544753 amadson 1544754 amadson
1544750 amadson 1544751 amadson 1544747 amadson 1544737 amadson 1544738
amadson 1544735 amadson 1544734 amadson 1544732 amadson 1544728 amadson
1544723 amadson 1544725 amadson 1544720 amadson 1544722 amadson 1544711
amadson 1544712 amadson 1544707 amadson 1544702 amadson 1544697 amadson
1544694 amadson 1544685 amadson 1544687 amadson 1544679 amadson 1544676
amadson 1544674 amadson 1544667 amadson 1544662 amadson 1544660 amadson
1544661 amadson 1544639 amadson 1544636 amadson 1544638 amadson 1544629
amadson 1544630 amadson 1544627 amadson 1544618 amadson 1544620 amadson
1544612 amadson 1544607 amadson 1544605 amadson 1544598 amadson 1544597
amadson 1544587 amadson 1544583 amadson 1544586 amadson 1544579 amadson
1544577 amadson 1544569 amadson 1544571 amadson 1544553 amadson 1544550
amadson 1544537 amadson 1544534 amadson 1544522 amadson 1544516 amadson
1544517 amadson 1544507 amadson 1544500 amadson 1544488 amadson 1544463
amadson 1544457 amadson 1544456 amadson 1544448 amadson 1544439 amadson
1544435 amadson 1544431 amadson 1544412 amadson 1544374 amadson 1544355
amadson 1544319 amadson

```
mbcpu-024              mb*   alloc   96  96   85.79*  1023575   799871
1541230 amadson 1541228 amadson 1541216 amadson 1541181 amadson 1541147
amadson 1541120 amadson 1541075 amadson 1541049 amadson 1541048 amadson
1541005 amadson 1541427 amadson 1541399 amadson 1541400 amadson 1541366
amadson 1541301 amadson 1541283 amadson 1541281 amadson 1541257 amadson
1542376 amadson 1542257 amadson 1542256 amadson 1542254 amadson 1542129
amadson 1542126 amadson 1542123 amadson 1542053 amadson 1542054 amadson
1542056 amadson 1542005 amadson 1542004 amadson 1542002 amadson 1541891
amadson 1541819 amadson 1541820 amadson 1541764 amadson 1541765 amadson
1541763 amadson 1541676 amadson 1541620 amadson 1541539 amadson 1544203
amadson 1544198 amadson 1544199 amadson 1544200 amadson 1544197 amadson
1544192 amadson 1544193 amadson 1544190 amadson 1544191 amadson 1544189
amadson 1544186 amadson 1544181 amadson 1544179 amadson 1544174 amadson
1544175 amadson 1544171 amadson 1544165 amadson 1544166 amadson 1544162
amadson 1544163 amadson 1544158 amadson 1544159 amadson 1544153 amadson
1544148 amadson 1544147 amadson 1544139 amadson 1544140 amadson 1544137
amadson 1544127 amadson 1544116 amadson 1544118 amadson 1544121 amadson
1544112 amadson 1544110 amadson 1544107 amadson 1544097 amadson 1544098
amadson 1544099 amadson 1544091 amadson 1544094 amadson 1544089 amadson
1544085 amadson 1544086 amadson 1544081 amadson 1544080 amadson 1544077
amadson 1544079 amadson 1544073 amadson 1544037 amadson 1544038 amadson
1543934 amadson 1543935 amadson 1543909 amadson 1543910 amadson 1543884
amadson 1543885 amadson
mbcpu-025               mb   alloc   96  96  133.55*  1023575    13523*
1544064 amadson 1544832 amadson 1544831 amadson 1544828 amadson 1544818
amadson 1544816 amadson 1544814 amadson 1544807 amadson 1544808 amadson
1544806 amadson 1544799 amadson 1544793 amadson 1544794 amadson 1544795
amadson 1544797 amadson 1544790 amadson 1544771 amadson 1544774 amadson
1544767 amadson 1544769 amadson 1544770 amadson 1544765 amadson 1544761
amadson 1544757 amadson 1544755 amadson 1544739 amadson 1544733 amadson
1544730 amadson 1544729 amadson 1544715 amadson 1544714 amadson 1544695
amadson 1544693 amadson 1544690 amadson 1544672 amadson 1544663 amadson
1544659 amadson 1544655 amadson 1544658 amadson 1544651 amadson 1544653
amadson 1544654 amadson 1544642 amadson 1544637 amadson 1544632 amadson
1544633 amadson 1544628 amadson 1544623 amadson 1544621 amadson 1544601
amadson 1544603 amadson 1544599 amadson 1544600 amadson 1544590 amadson
1544589 amadson 1544585 amadson 1544580 amadson 1544570 amadson 1544563
amadson 1544562 amadson 1544559 amadson 1544560 amadson 1544554 amadson
1544549 amadson 1544544 amadson 1544545 amadson 1544529 amadson 1544533
amadson 1544527 amadson 1544525 amadson 1544518 amadson 1544509 amadson
1544511 amadson 1544513 amadson 1544514 amadson 1544497 amadson 1544498
amadson 1544481 amadson 1544482 amadson 1544474 amadson 1544471 amadson
1544466 amadson 1544459 amadson 1544455 amadson 1544451 amadson 1544452
amadson 1544436 amadson 1544428 amadson 1544429 amadson 1544419 amadson
1544417 amadson 1544403 amadson 1544388 amadson 1544379 amadson 1544358
amadson 1544335 amadson
mbh100-001          mb-h100  drain$    0  96    0.00   1281554        0
mbh100-002          mb-h100    idle    0  96    0.01   1281554  1275170
mbh100-003          mb-h100    idle    0  96    0.00   1281554   961682
mbh100-004          mb-h100    idle    0  96    0.00   1281554   984596
mbh100-005          mb-h100     mix   32  96    8.00*  1281554   107608*
1243592 fghorban
mbh100-006          mb-h100   down$    0  96    0.00   1281554  1275587
mbl40s-001          mb-l40s     mix   64  96   55.83*   765525   310548
1480439 hbalantr 1440260 vvarenth
mbl40s-002          mb-l40s    idle    0  96    0.00    765525   760896
mbl40s-003          mb-l40s    idle    0  96    0.00    765525   761192
```

```
mbl40s-004          mb-l40s    idle    0  96    0.00    765525   761030
mbl40s-005          mb-l40s    idle    0  96    0.00    765525   760728
mbl40s-007          mb-l40s    idle    0  96    0.00    765525   761452
wi001          inv-wildiris    idle    0  48    0.00    506997   505745
wi002          inv-wildiris    idle    0  48    0.00    506997   505726
wi003          inv-wildiris    idle    0  48    0.00    506997   505746
wi004          inv-wildiris    idle    0  48    0.00    506997   505729
wi005          inv-wildiris    idle    0  56    0.00   1031000  1020610
```

sinfo examples:

```
# View overall cluster:
[]$ sinfo -eO "CPUs:8,Memory:9,Gres:14,NodeAIOT:16,NodeList:50"
CPUS    MEMORY   GRES          NODES(A/I/O/T)  NODELIST
96      1023575  (null)        6/19/0/25       mbcpu-[001-025]
96      765525   gpu:a30:8     0/8/0/8         mba30-[001-008]
96      765525   gpu:l40s:8    1/4/0/5         mbl40s-[001-005]
96      765525   gpu:l40s:4    0/1/0/1         mbl40s-007
64      1023575  gpu:a6000:4   0/1/0/1         mba6000-001
48      506997   (null)        0/4/0/4         wi[001-004]
56      1031000  gpu:a30:2     0/1/0/1         wi005
96      1281554  gpu:h100:8    1/3/2/6         mbh100-[001-006]

# View a particular (investment) partition:
[]$ sinfo -p inv-wildiris
PARTITION    AVAIL  TIMELIMIT  NODES  STATE NODELIST
inv-wildiris    up   infinite      5   idle wi[001-005]

# View compute nodes currently drained:
[]$ sinfo -R
REASON               USER      TIMESTAMP           NODELIST
HW Status: Unknown - slurm     2024-07-19T12:02:04 mbh100-001
Not responding       slurm     2024-07-30T13:49:06 mbh100-006
```

Slurm: Common Issues and How to Resolve

Goal: List some common issues and how to resolve.

Common Questions

- How do I know what number of nodes, cores, memory etc to ask for my jobs?
- How do I find out whether a cluster/partition supports these resources?
- How do I find out whether these resources are available on the cluster?
- How long will I have to wait in the queue before my job starts? How busy is the cluster?
- How do I monitor the progress of my job?

Common Questions: Suggestions

- How do I know what number of nodes, cores, memory etc to ask for my jobs?
    - **Understand your software and application.**
        - **Read the docs – look at the help for commands/options.**
        - **Can it run multiple threads - use multi cores (OpenMP) / nodes (MPI)?**
        - **Can it use a GPU? Nvidia cuda.**
        - **Are their suggestions on data and memory requirements?**
- How do I find out whether a cluster/partition supports these resources?
- How do I find out whether these resources are available on the cluster?
    - **Consult the wiki**: Medicine Hardware Summary Table
- How long will I have to wait in the queue before my job starts?
    - **How busy is the cluster?**
    - **Current Cluster utilization**: Commands `sinfo` / `arccjobs` / `pestat` and OnDemand's MedicineBow System Status page.
- How do I monitor the progress of my job?
    - **Slurm commands**: `squeue`

Common Issues

- Not defining the `account` and `time` options.
- The `account` is the name of the ***project you are associated with***. It is not your username.
- Requesting combinations of ***resources that can not be satisfied***: Medicine Hardware Summary Table
    - For example, you can not request 40 cores on a compute node with a max of 32.
    - Requesting too much memory, or too many GPU devices with respect to a partition.
- My job is pending? Why?
    - Because the ***resources are currently not available***.
    - Have you unnecessarily defined a specific partition (*restricted yourself*) that is *busy*?
    - We only have a small number of GPUs.

    - This is a ***shared resource*** - sometimes you just have to be patient…
    - Check ***current cluster utilization***.
- Preemption: Users of an investment get priority on their hardware.
    - We have the `non-investor` partition.

Required: Account and Walltime

Remember: By default you must define the project (*account*) you're using and a walltime.

```
[salexan5@mblog2 ~]$ salloc
salloc: error: You didn't specify a project account (-A,--account). Please
open a ticket at arcc-help@uwyo.edu for help.
salloc: error: Job submit/allocate failed: Invalid account or
account/partition combination specified

[salexan5@mblog2 ~]$ salloc -A arcc
salloc: error: You didn't specify a walltime (-t, --time=) for the job.
Please open a ticket at arcc-help@uwyo.edu for help.
salloc: error: Job submit/allocate failed: Requested time limit is invalid
(missing or exceeds some limit)

[salexan5@mblog2 ~]$ salloc -t 10:00
salloc: error: You didn't specify a project account (-A,--account). Please
open a ticket at arcc-help@uwyo.edu for help.
salloc: error: Job submit/allocate failed: Invalid account or
account/partition combination specified

# The bare minimum:
[salexan5@mblog2 ~]$ salloc -A arcc -t 10:00
salloc: Granted job allocation 1250349
salloc: Nodes mbcpu-025 are ready for job
```

Correct Partitions

If you need to explicitly request a partition, the name must be correct:

```
[salexan5@mblog2 ~]$ salloc -A arcc -t 10:00 --partition=mb-140
salloc: error: invalid partition specified: mb-140
salloc: error: Job submit/allocate failed: Invalid partition name specified
```

Use the `sinfo` command to get a list of know partitions, as well as detailing their current use:

Example: sinfo

```
[salexan5@mblog2 ~]$ sinfo
PARTITION    AVAIL  TIMELIMIT  NODES  STATE NODELIST
mb*             up 7-00:00:00      4    mix mbcpu-[008,010-011,025]
```

```
mb*             up 7-00:00:00      8  alloc mbcpu-[001-007,009]
mb*             up 7-00:00:00     13   idle mbcpu-[012-024]
mb-a30          up 7-00:00:00      8   idle mba30-[001-008]
mb-l40s         up 7-00:00:00      1    mix mbl40s-001
mb-l40s         up 7-00:00:00      5   idle mbl40s-[002-005,007]
mb-h100         up 7-00:00:00      1 drain$ mbh100-001
mb-h100         up 7-00:00:00      1  down$ mbh100-006
mb-h100         up 7-00:00:00      1  drain mbh100-002
mb-h100         up 7-00:00:00      3    mix mbh100-[003-005]
mb-a6000        up 7-00:00:00      1   idle mba6000-001
inv-arcc        up   infinite      1    mix mbcpu-025
inv-inbre       up   infinite      1   idle mbl40s-007
inv-sshesha     up   infinite      1   idle mba6000-001
inv-wysbc       up   infinite      1  alloc mbcpu-001
inv-wysbc       up   infinite      1   idle mba30-001
inv-soc         up   infinite      1    mix mbl40s-001
inv-wildiris    up   infinite      5   idle wi[001-005]
non-investor    up 7-00:00:00      1 drain$ mbh100-001
non-investor    up 7-00:00:00      1  down$ mbh100-006
non-investor    up 7-00:00:00      1  drain mbh100-002
non-investor    up 7-00:00:00      6    mix mbcpu-[008,010-011],mbh100-[003-
005]
non-investor    up 7-00:00:00      7  alloc mbcpu-[002-007,009]
non-investor    up 7-00:00:00     24   idle mba30-[002-008],mbcpu-[012-
024],mbl40s-[002-005]
# Corrected:
[salexan5@mblog2 ~]$ salloc -A arcc -t 10:00 --partition=mb-l40s
salloc: Pending job allocation 1250907
salloc: job 1250907 queued and waiting for resources
salloc: job 1250907 has been allocated resources
salloc: Granted job allocation 1250907
salloc: Nodes mbl40s-001 are ready for job
```

Timeouts

Timeouts aren’t errors as such, just that the time you requested was not long enough to compete the computation.

The maximum allowed wall time is 7 days:

```
[arcc-t01@mblog2 ~]$ salloc -A arccanetrain -t 7-00:00:01
salloc: error: Job submit/allocate failed: Requested time limit is invalid
(missing or exceeds some limit)

[arcc-t01@mblog2 ~]$ salloc -A arccanetrain -t 7-00:00:00
salloc: Granted job allocation 1251651
salloc: Nodes mbcpu-010 are ready for job
```

Do not request 7 days just because you can!

Wall time is considered when Slurm tries to allocate your job. A job is more likely to be back filled (slotted onto the cluster) in busy times than pending jobs will longer wall times.

My Jobs Need to Run Longer than 7 Days

ARCC can provide users with wall times longer than 7 days.

Please contact use, but we require that you can demonstrate that you job can not be optimized, for example:

- Can it run faster by using more cores, or even multiple nodes?
- Can it utilize GPUs?
- Can the job actually be divided up into sections that can be run concurrently across multiple jobs?

ARCC can provide assistance with trying to understand if a job can be *optimized*.

Requested node configuration is not available

This is caused because you're trying to request a configuration that isn't available, or requires more details: For example:

Too many cores on a node:

```
[salexan5@mblog2 ~]$ salloc -A arcc -t 10:00 -c 100
salloc: error: CPU count per node can not be satisfied
salloc: error: Job submit/allocate failed: Requested node configuration is
not available
```

Must define a GPU enabled partition:

```
[salexan5@mblog2 ~]$ salloc -A arcc -t 10:00 --gres=gpu:1
salloc: error: Job submit/allocate failed: Requested node configuration is
not available
salloc: Job allocation 1253677 has been revoked.

[salexan5@mblog2 ~]$ salloc -A arcc -t 10:00 --gres=gpu:1 --partition=mb-a30
salloc: Granted job allocation 1253691
salloc: Nodes mba30-001 are ready for job
```

OUT-OF-MEMORY: Segmentation Fault

Segmentation faults are typically caused by an application trying to access memory outside what has been allocated to the job.

Basically, you job is out of memory of what it requested.

Resolved: Request more memory using either the `mem` or `mem-per-cpu` .

My Job Stopped and Re-Started: Preemption

As discussed in the Intro to HPC workshop, we have a Condominium Model where if your job is running an a compute node that is part of another project's hardware investment, your job can be preempted.

Your job will be stopped and automatically re-queued and when resources come available on the cluster, it will be restarted.

Further details can be found on our Slurm and Preemption page and how to use the `non-investor` partition to prevent this from happening.

Why Is My Job One-of-Many on a Compute Node?

When I run `pestat`, it appears that my job is one of many on a particular compute node.

```
[]$ pestat -n mbl40s-001
Select only nodes in hostlist=mbl40s-001
Hostname         Partition     Node Num_CPU  CPUload  Memsize  Freemem
Joblist
                              State Use/Tot  (15min)     (MB)     (MB)
JobID(JobArrayID) User ...
mbl40s-001         mb-l40s     mix   64  96   55.62*   765525   310618
1480439 hbalantr 1440260 vvarenth
```

As discussed in the Intro to HPC workshop when talking about Compute Nodes this is perfectly acceptable and one of the tasks that Slurm manages.

Remember: All jobs are independent and do not affect anyone elses.

Use the following link to provide feedback on this training: https://forms.gle/bxhKoVaPns51Qhb99 or use the QR code below.