



# Introduction to Job Submission and Scheduling

Minnesota Supercomputing Institute

University of Minnesota

Ham C Lam, Ph.D.

Angel Mancebo, Ph.D.

MSI Research Informatics - Informatics and Computing

Fall 2023

# Job submission and scheduling


## Overview

- o Connect to MSI
- o HPC hardware
- o Job scheduling
- o Submit Jobs
- o Monitoring Jobs
- o Troubleshooting Tips
- o Hands-on demo

## Recommended background

- Basic UNIX commands and BASH scripting experience

## Training Level

- Beginner 

## Tutorial format

- Lecture combined with hands-on examples of submitting jobs

# Job submission and scheduling

## Why need access to more computers?

Frequently, research problems that use computing can outgrow the desktop or laptop computer

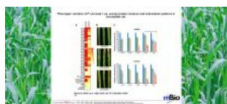
- A statistics student wants to cross-validate their model. This involves running the model 1000 times - but each run takes an hour. Running on their laptop will take over a month!
- A genomics researcher has been using small datasets of sequence data, but soon will be receiving a new type of sequencing data that is 10 times as large. It's already challenging to open the datasets on their computer - analyzing these larger datasets will probably crash it.
- An engineer is using a fluid dynamics package that has an option to run in parallel. So far, they haven't used this option on their desktop, but in going from 2D to 3D simulations, simulation time has more than tripled and it might be useful to take advantage of that feature.

In all these cases, what is needed is access to more computers that can be used at the same time.

# High Performance Computing (HPC) at MSI

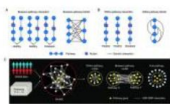
## What can I do with HPC?

- ❑ Solving large problems with little time
- ❑ Run simulation and analysis of large volume of data that would not be possible with standard computers



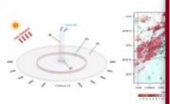
### Genomic Studies of Oat Crown Rust

Oat crown rust is a serious disease that affects oat crops worldwide.



### Genetic Interactions as Predictors of Breast Cancer Risk

Breast cancer is a leading cause of death for women. Researchers know that genetics play an important part in one's chance of getting the disease, but there is still much that they don't know.



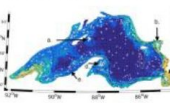
### Modeling an Air-Pollution Filtration System

Despite years of attention, air pollution continues to be a major problem in many large cities throughout the world.



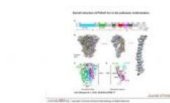
### Deep Learning at MSI

Over the past year, MSI has been increasing resources available to researchers working in machine learning fields.



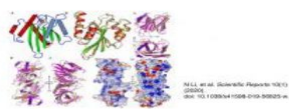
### Models of Water Mixing in Lake Superior

Lake Superior, the largest of the Great Lakes, is a major water source for humans as well as a habitat for a large amount of wildlife who depend on plants, plankton, fish, and other food sources for



### Pioneering Structural Study of Porcine Coronavirus

MSI PIs [Wei Zhang](#) (research associate professor, [D](#))



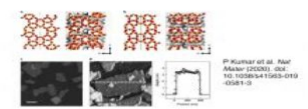
### Crystal Structure of Fowlpox Virus Resolvase

Fowlpox is a viral disease of chickens and turkeys that occurs worldwide.



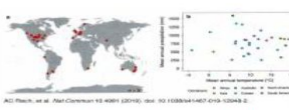
### 2020 MSI Research Exhibition

MSI held the eleventh annual Research Exhibition on April 28, 2020.



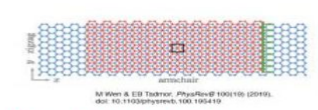
### Better Zeolite Nanosheets

Zeolites are a class of materials that are used by industry as adsorbents and catalysts. Very thin zeolite films - nanofilms - can be used as molecular filters.



### Improving Predictions of Soil Nitrogen Mineralization

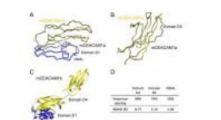
The Nutrient Network (<https://nutnet.org>) is a global collaborative study of the effects of two of



### A New Model for Multilayer

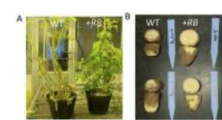


### Discovering the Origins of



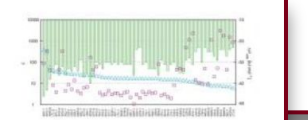
### Structural Comparisons of Two Receptors for Mouse Coronavirus

Coronaviruses are a large group of viruses that can cause illness in humans, other mammals, and birds. They use cell surface receptors to attach themselves to host cells.



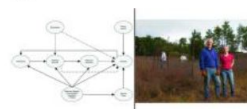
### Genetic Resistance Against Potato Blight

Late blight is a serious crop disease that affects potatoes and costs the potato industry billions of dollars annually.



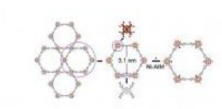
### Sweetening Natural Gas

Natural gas often contains contaminants. "Sour" gas is natural gas that contains a significant amount of hydrogen sulfide, H<sub>2</sub>S.



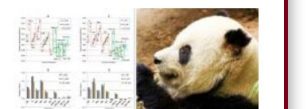
### Computer Model Links Plant Species Richness and Productivity

The relationship between vegetation productivity and species richness has been under investigation by ecologists for many years.



### A New Method for Creating a Catalyst on a Metal-Organic Framework

Effective and stable catalysts are important to a wide variety of processes important to research and industry.



### Panda Genomics

Conservationists seeking to increase in the panda population must understand issues related to genetic diversity.

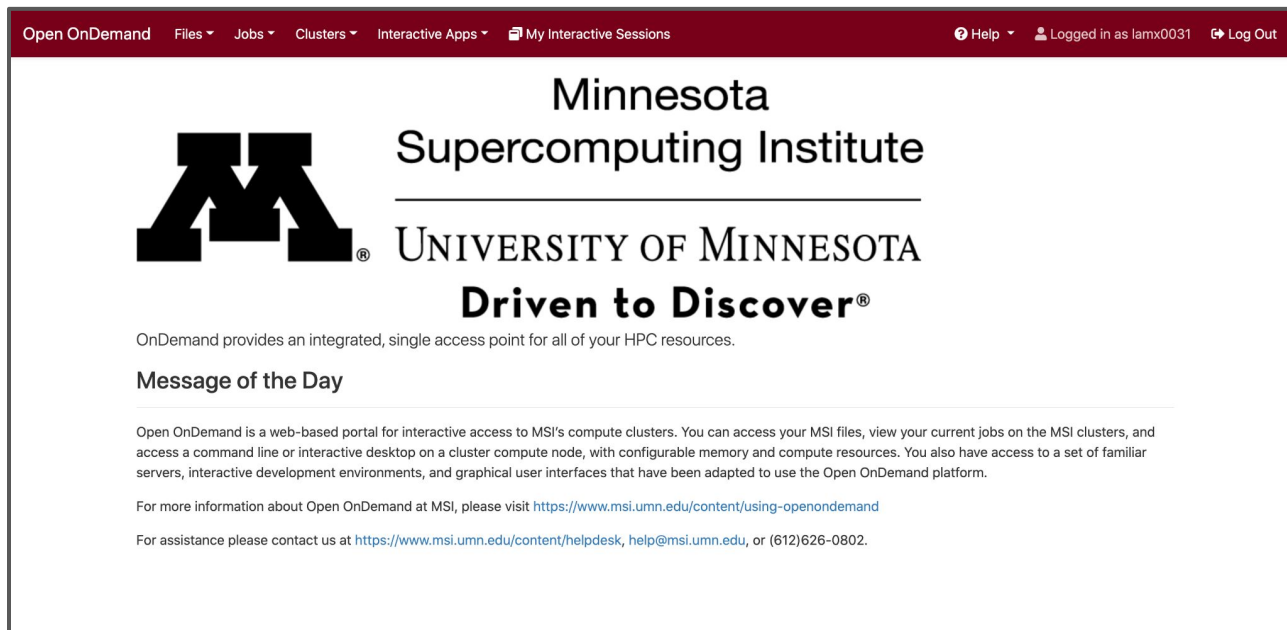
# Job submission and scheduling

 **Connecting to MSI**



## Open OnDemand

Point your browser to <https://ondemand.msi.umn.edu>



The screenshot shows the Open OnDemand web portal interface. The top navigation bar is dark red with white text for "Open OnDemand", "Files", "Jobs", "Clusters", "Interactive Apps", and "My Interactive Sessions". On the right side of the bar, there is a "Help" icon, "Logged in as lamx0031", and a "Log Out" button. The main content area is white and features the Minnesota Supercomputing Institute logo on the left, which consists of a stylized black 'M' with a registered trademark symbol. To the right of the logo, the text reads "Minnesota Supercomputing Institute" in a large serif font, followed by "UNIVERSITY OF MINNESOTA" in a smaller serif font, and "Driven to Discover®" in a bold sans-serif font. Below this, a paragraph states: "OnDemand provides an integrated, single access point for all of your HPC resources." Underneath is a section titled "Message of the Day" with a horizontal line. The text in this section describes the portal's capabilities and provides links for more information and assistance.



For more information: <https://www.msi.umn.edu/content/using-ondemand>

Whenever you log in to MSI, you are directed to a login node. A login node can be viewed as an interface to compute nodes.

### Login Nodes (**Not** for computation workload)

Agate	agate.msi.umn.edu (ah10<1,2,3,4>)
Mesabi	mesabi.msi.umn.edu (ln000<n>)
Mangi	mangi.msi.umn.edu (ln1001,ln1002)

```
ssh -Y <username>@mesabi.msi.umn.edu  
ssh -Y <username>@mangi.msi.umn.edu  
ssh -Y <username>@agate.msi.umn.edu
```

```
ssh -Y <username>@agate.msi.umn.edu  
ssh -Y <username>@mesabi.msi.umn.edu  
ssh -Y <username>@mangi.msi.umn.edu
```

```
hamlam~> [09:08:01 ~] ssh -Y lamx0031@mesabi.msi.umn.edu  
Last login: Thu Sep 30 08:18:03 2021 from ra-full-mfa-10-20-41-251.vpn.umn.edu  
-----  
University of Minnesota Supercomputing Institute  
Mesabi  
HP Haswell Linux Cluster  
-----  
For assistance please contact us at https://www.msi.umn.edu/support/help.html  
help@msi.umn.edu, or (612)626-0802.  
-----  
Home directories are snapshot protected. If you accidentally delete a file in  
your home directory, type "cd .snapshot" then "ls -lt" to list the snapshots  
available in order from most recent to oldest.  
-----  
January 6, 2021: Slurm is now the scheduler for all nodes.  
-----  
lamx0031@ln0006 [09:08:14 ~] |
```

```
ah104:lamx0031:~] ssh agate  
By using this system you agree to adhere to MSI and UMN Acceptable Use Policies -  
Last login: Wed Feb 8 10:24:34 2023 from 10.131.193.177  
-----  
University of Minnesota Supercomputing Institute  
Agate  
-----  
For assistance please contact us at https://www.msi.umn.edu/content/helpdesk,  
help@msi.umn.edu, or (612)626-0802.  
-----  
Home directories are snapshot protected. If you accidentally delete a file in  
your home directory, type "cd .snapshot" then "ls -lt" to list the snapshots  
available in order from most recent to oldest.  
-----  
The file you need can be copied from a snapshot back to its former place in  
your home directory.  
-----  
ah104:lamx0031:~] |
```

```
hamlam~> [09:11:59 ~] ssh -Y lamx0031@mangi.msi.umn.edu  
Last login: Wed Sep 22 10:18:57 2021 from in-ln0004.mesabi.msi.umn.edu  
-----  
University of Minnesota Supercomputing Institute  
Mangi  
AMD EPYC Linux Cluster  
-----  
For assistance please contact us at https://www.msi.umn.edu/support/help.html  
help@msi.umn.edu, or (612)626-0802.  
-----  
Home directories are snapshot protected. If you accidentally delete a file in  
your home directory, type "cd .snapshot" then "ls -lt" to list the snapshots  
available in order from most recent to oldest.  
-----  
January 6, 2021: Slurm is now the scheduler for all nodes.  
-----  
lamx0031@ln1002 [09:12:05 ~] |
```

ssh



# Job submission and scheduling

 **Overview of HPC and job scheduling**



## Agate

- 412 nodes
- AMD processors with 64-128 CPU cores per node
- 344 CPU compute node
  - 244 have 512G mem
  - 100 have 2TB mem
- 58 GPU compute nodes
  - 50 A100 512G mem
  - 8 A100 1TB mem
- 10 GPU interactive nodes
  - 8 A40 GPUs 512G mem each



**Agate is ranked 497 out of 500 on Top500 list!**



## Mesabi

- Over 700 nodes
  - Memory configuration
    - 616 nodes have 64GB RAM
    - 24 nodes have 256GB RAM
    - 16 nodes have 1TB RAM
    - 40 k40 GPU nodes with 128GB RAM
- 17,784 cores provided by Intel Haswell Processors
- 480GB SSD available on 32 nodes



## Mangi

- 164 nodes
- AMD ROME processors
- 20,992 compute cores
- 12 nodes with 4-way v100 GPU
- 1 node with 8-way v100 GPU
  - Memory configuration
    - 144 nodes with 256GB RAM
    - 10 nodes with 512GB RAM
    - 10 nodes with 2TB RAM




# Job submission and scheduling

 **Q: What is job scheduling?**

**A:** It is the process of *arranging*, *controlling* and *optimizing* work and workloads in a *shared* HPC environment.

 **Q: Why do we need job scheduling?**

**A:**

-  MSI serves over **900** groups and over **4500** users with limited HPC resources so we all must share the computing hardware.
-  Automated allocation of limited resources
-  Tracking and monitoring jobs

# Job submission and scheduling

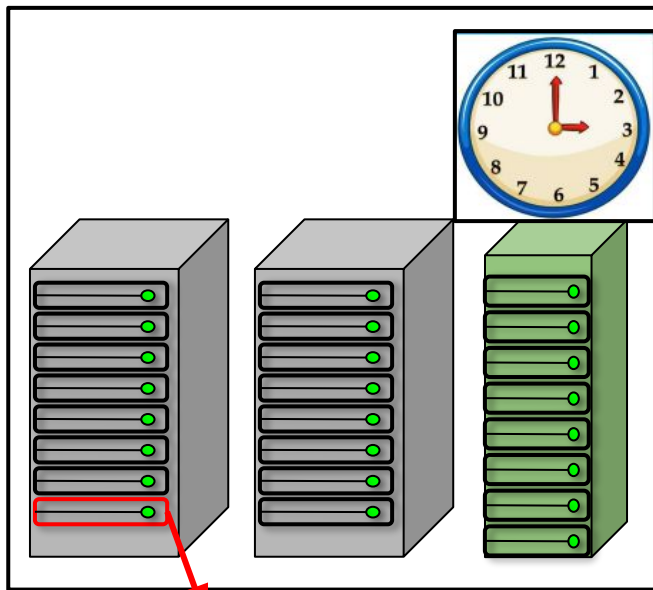


**Simple Linux Utility for Resource Management (SLURM)** is an open source, scalable cluster management and job scheduling system. It was created at Livermore Computing Center and has since been installed in many of the Top 500 supercomputers around the world.

## Quick Slurm highlights

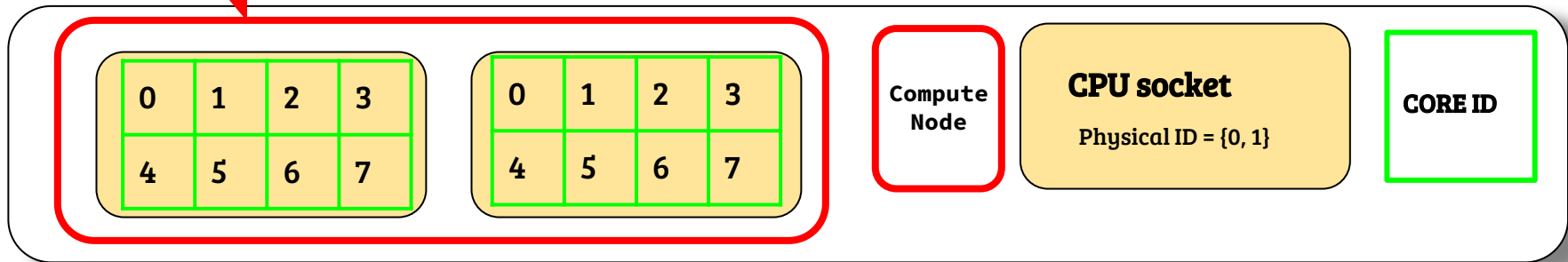
- It schedules jobs to be executed on a cluster of machines based on priorities
- Pending jobs are “queued” jobs waiting to be executed
- Jobs are submitted by users via shell commands
- It takes care of Input/Output (I/Os)
- It launches jobs on assigned compute node(s) and clean up after each job finishes

*Slurm decides who gets what and when!*







## High Performance Computing (HPC)

- A network of computers form the high performance computing system called a cluster.
- Each computer in a cluster is called a *node*.
- Each node can talk to each other through a high speed network.
- Each node has multiple processors with multiple cores and large memory.



**Q: Where do I find the hardware? A: *Slurm partitions***

-  **A** logical set(s) of compute nodes grouped together depending on their hardware characteristics or function.
-  **A** slurm job partition can be seen as an automated waiting list for use of a particular set of computational hardware.
-  **D**ifferent job partitions have different resources and limitations.
-  **M**ake sure to choose a job partition which has resources and limitations suitable to your jobs

**Q: Where do I find the hardware?**

**A: Slurm partitions**



**Federated Partitions** (msismall, msilarge, msibigmem, msigpu, interactive, interactive-gpu, preempt, preempt-gpu)

*Jobs that can run on either Agate or Mesabi/Mangi*

*Job arrays will only run on the cluster you submit the job on*



**Agate Partitions** (agsmall, aglarge, ag2tb, a100-4, a100-8, interactive, interactive-gpu, preempt, preempt-gpu)

*MSI newest HPC cluster with the best hardware, and should be your first choice for submitting jobs!*



**Mesabi/mangi Partitions** (small, large, max, ram256g, ram1t, k40, v100, amdsmall, amdlarge, amd512, amd2tb, interactive, interactive-gpu, preempt, preempt-gpu)





<https://www.msi.umn.edu/queues>


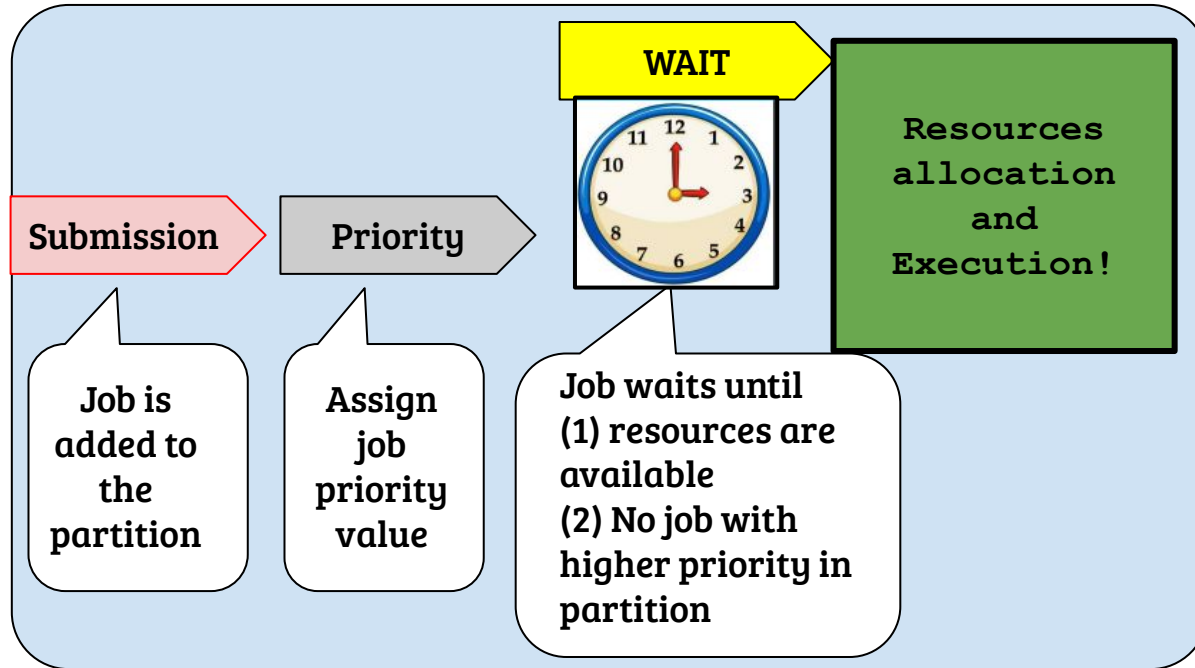
<https://www.msi.umn.edu/content/choosing-job-partition#slurm>

\*Users are limited to 2 jobs in the **interactive** and **interactive-gpu** partitions.

\*Jobs in the **preempt** and **preempt-gpu** partitions may be killed at any time to make room for jobs in the interactive or interactive-gpu partitions.



-  A small text file must be prepared by an user that says what program to run, where to get the input, and where to put the output.
-  The user then submit this job script to the SLURM *scheduler* which decides when and where it will run.
-  Once the job has finished, the user can retrieve the results of the calculation.
-  There is no interaction between the user and the program while the job is running.



Slurm will kill your job at runtime if your job exceeds the requested amount of resources.

## My job's priority



### **AGE**

The longer the job waits in queue, its age factor gradually increases.

### **JOB SIZE**

Jobs requesting more CPUs are favored (large jobs).

### **FAIRSHARE**

Prioritizes jobs belonging to underserved users/accounts. 1) is adjusted based on the recent usage of group members, 2) is proportional to the compute resources used by the group, and 3) impact on priority decay over time.



More information: <https://www.msi.umn.edu/content/hpc>

 **Accessing software**

 **Software at MSI**

- ❑ MSI has hundreds of software modules. Software environment modules are used to make software available to you!
- ❑ A **module** is a self-contained description of a software package - it contains the settings required to run a software package and, usually, encodes required dependencies on other software packages.
- ❑ On a high-performance computing system, it is often the case that no software is loaded by default. If we want to use a software package, we will need to “load” the module ourselves.



MSI also maintains a searchable directory of available software at <https://www.msi.umn.edu/software>.



## Working with software modules

The *module* command is used to interact with environment modules. An additional subcommand is usually added to the command to specify what you want to do.

Description	Command	Example
See all available modules	<code>module avail</code>	<code>module avail</code>
Load a module	<code>module load</code>	<code>module load matlab/2021a</code>
Unload a module	<code>module unload</code>	<code>module unload matlab/2021a</code>
Unload all modules	<code>module purge</code>	<code>module purge</code>
See what a module does	<code>module show</code>	<code>module show matlab/2021a</code>
List currently loaded modules	<code>module list</code>	<code>module list</code>

- The module system handles software versioning and package conflicts for you automatically.

# Job submission and scheduling

## Module command hands-on examples

```
hamlam~> [11:55:14 ~] mesabi
Last login: Wed Feb  9 10:23:49 2022 from 10.21.28.209
-----
                University of Minnesota Supercomputing Institute
                    Mesabi
                HP Haswell Linux Cluster
-----
For assistance please contact us at https://www.msi.umn.edu/support/help.html
help@msi.umn.edu, or (612)626-0802.
-----
Home directories are snapshot protected. If you accidentally delete a file in
your home directory, type "cd .snapshot" then "ls -lt" to list the snapshots
available in order from most recent to oldest.

January 6, 2021: Slurm is now the scheduler for all nodes.
-----
lamx0031@ln0006 [11:55:17 ~] module avail gcc
----- /panfs/roc/soft/modulefiles.hpc -----
gcc/8.2.0(default) gcc/9.2.0
----- /panfs/roc/soft/modulefiles.common -----
gcc/4.9.2 gcc/5.1.0 gcc/5.4.0 gcc/6.1.0 gcc/6.3.0 gcc/7.2.0(default) gcc/8.1.0
lamx0031@ln0006 [11:55:21 ~] █
```

module avail

module load

module unload

module purge

module show

module list

MSI also maintains a searchable directory of available software at <https://www.msi.umn.edu/software>.



## Run scripted and interactive jobs

To access compute nodes (CPUs and GPUs), you must either submit a job script or initiate an interactive session.



**sbatch** - Submit a job script to Slurm for remote execution

Request resource allocation through a job script or at the command lines using the **sbatch** command. If invoked at command lines with options, these options take precedence over the **#SBATCH** options in the job script.

```
#!/bin/bash
#SBATCH --job-name=demo1

echo "I ran on node: "
hostname; sleep 120
```

Simple job script

```
%sbatch demo1.sh
%sbatch: Setting account: support
%Submitted batch job 9704508
```

submit a job

The **#SBATCH** directives must appear at the top of the submission file, before any other line except for the very first line which should be the **#!/bin/bash**.

*p=<Name>*  
*--exclusive?*  
*--ntasks=N*  
*--time=hrs:min:sec*  
*--mem=N*  
*--mem-per-cpu=N*  
*--tmp=N*  
*--nodes=N*

Partition name	Node sharing?	Cores per node	Walltime limit	Total node memory	Advised memory per core	Local scratch per node	Maximum Nodes per Job
<b>amdsmall</b> <sup>(1)</sup>	Yes	128	96:00:00	248 GB	1900 MB	415 GB	1
<b>amdlarge</b>	No	128	24:00:00	248 GB	1900 MB	415 GB	32
<b>amd512</b>	Yes	128	96:00:00	499 GB	4000 MB	415 GB	1
<b>amd2tb</b>	Yes	128	96:00:00	1995 GB	15 GB	415 GB	1
<b>v100</b> <sup>(1)</sup>							

sbatch will stop processing further #SBATCH directives once the first non-comment non-whitespace line has been reached in the script.

Use command `squeue --me` (or `-u <username>`) to view jobs status.

```
%sbatch demo1.sh
sbatch: Setting account: support
Submitted batch job 9704508
%squeue -u lamx0031
      JOBID PARTITION      NAME      USER  ST      TIME  NODES NODELIST(REASON)
      9704508      small demo1.sb lamx0031 PD      0:00      1 (None)
```

## (ST) JOB STATE CODES

**PD** PENDING Job is awaiting resource allocation    **CA** CANCELLED Job was explicitly cancelled by the user/admin  
**CG** COMPLETING Job is in the process of completing    **R** RUNNING Job currently has an allocation

## NODELIST (REASON)

**Priority** One or more higher priority jobs exist for this partition

**Resources** The job is waiting for resources to become available

**ReqNodeNotAvail** Some node specifically required by the job is not currently available.

By default, the output of a job is saved in a file called "slurm-<job id>.out". The '#SBATCH --output' can be used to specify a different name and location of the output file. The '#SBATCH --error <filename>' is used to capture stderr of the job to a file.

```
#!/bin/bash
## My first job script
#SBATCH --job-name=demo1

#SBATCH --output demo1_%j.output    # -o (shorthand)
#SBATCH --error demo1_%j.error     # -e (shorthand)

echo "I ran on node: "
hostname
sleep 120
```

## Simple job script

```
%cat demo1_9704508.output
I ran on node:
cn1140
```

Each job has an associated account name that corresponds to a PI group. If you belong to multiple groups, you can specify an account that will be used by the job.

```
#!/bin/bash

#SBATCH --job-name=demo1
#SBATCH --account=support

echo "I ran on node: "
hostname
sleep 120
```

Your MSI user account is associated with the PI who created the account, known as your primary group. If you are working with another PI on a project, you can use the ‘--account=<groupname>’ option to have jobs run under a different group that allows you to have access to compute resources (quota space, files, etc) of that group.

You can use the “id” command to find out if you belong to more than one group:

```
%id <username>
```

Your job begins in the directory that it was submitted from.

**#SBATCH -J**: short for `--jobname`, name of the job.

**#SBATCH -N** : short for `--nodes`, *number of nodes* on which to run.

**#SBATCH -n** : short for `--ntasks`, number of tasks (CPU cores) to run job on.

**#SBATCH -c** : short for `--ncpus-per-task`, *number of cpu* per process.

**#SBATCH -p *partition***: short for `--partition`, submit job to the *partition* queue.

- Partitions can be found via the ***sinfo*** command.

**#SBATCH -t *hh:mm:ss***: short for `--time`, request resources to run job for *hh* hours, *mm* minutes and *ss* seconds.

List of common useful SLURM environmental variables and their meaning:

- **SLURM\_JOBID:** Job ID number given to this job
- **SLURM\_JOB\_NODELIST:** List of nodes allocated to the job
- **SLURM\_SUBMIT\_DIR:** Directory where the sbatch command was executed
- **SLURM\_NNODES:** Total number of nodes in the job's resource allocation.
- **SLURM\_NTASKS:** Total number of CPU cores requested in a job.

# Job submission and scheduling

 **Interactive jobs**



- The most common use of `srun` is to launch an interactive session on a compute node with the “`--pty`” option.

e.g. `srun -t 100 -N1 -n1 -c1 -p interactive --pty bash` #request a ‘shell’

```
ah103:lamx0031:~] srun -N 1 -n1 -t 100 -p interactive --pty bash
srun: Setting account: support
srun: job 71059264 queued and waiting for resources
srun: job 71059264 has been allocated resources
acn01:lamx0031:~] █
```

Dedicated partitions for interactive workflow: `interactive` and `interactive-gpu`  
(Users are limited to 2 jobs in the `interactive` and 2 jobs in `interactive-gpu` partitions)



- *User-provided command* can be “/bin/bash” (to launch an interactive shell) or a script with arguments.

```
lamx0031@ln0003 [~] srun --nodes 1 --ntasks 1 --cpus-per-task 4 -p small ./omp_hw
srun: Setting account: support
srun: job 7000821 queued and waiting for resources
srun: job 7000821 has been allocated resources
Hello World from thread = 0
Number of threads = 4
Hello World from thread = 2
Hello World from thread = 3
Hello World from thread = 1
lamx0031@ln0003 [~]
```

**1 node, 1 task, 4 cores  
with 1 thread per core**

Dedicated partitions for interactive workflow: interactive and interactive-gpu  
(Users are limited to 2 jobs in the interactive and 2 jobs in interactive-gpu partitions).

# Job submission and scheduling

 **Scripted job examples**

This is the simplest type of job which uses one core on a single compute node. It is also our default setting when no "#SBATCH" statement is provided in a job script.

```
#!/bin/bash

#SBATCH --job-name=demo1
#SBATCH --nodes=1 # specify one node
#SBATCH --ntasks=1 # specify one task per cpu-core
#SBATCH --mem=4g # request 4 gb (default is 1 gb)
#SBATCH -o demo1_%j.output
#SBATCH -e demo1_%j.error

echo "I ran on node: "
hostname
sleep 120
```

A multithreaded job launches one slurm task (process) which uses several CPUs.

This example script launches a single process with 4 CPU cores

```
#!/bin/bash
#SBATCH --job-name=multithreaded      #
#SBATCH --nodes=1                    # A single node count
#SBATCH --ntasks=1                   # One task
#SBATCH --cpus-per-task=4           # Request 4 cores

export OMP_NUM_THREADS=$SLURM_CPUS_PER_TASK
./omp_helloworld
```

Creating a job array provides an easy way to group related jobs together. A job array is a collection of jobs that differ from each other by only a single index parameter. All jobs in a job array must have the same resource requirements.

```
#!/bin/bash
#SBATCH --job-name=my_array_job
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=1
#SBATCH --time=00:10:00
#SSBATCH --array=1-5 # 5 jobs
```

```
python arraytest.py file-${SLURM_ARRAY_TASK_ID}.txt >output_${SLURM_ARRAY_TASK_ID}
```

JOBID	PARTITION	NAME	USER	STATE	TIME	TIME_LIMI	NODES	NODELIST(REASON)
6506483	interacti	bash	lamx0031	RUNNING	4:10:40	12:00:00	1	cn2001
6511654_2	small	job_arra	lamx0031	COMPLETI	0:11	10:00	1	cn0367
6511654_3	small	job_arra	lamx0031	COMPLETI	0:11	10:00	1	cn0445
6511654_1	small	job_arra	lamx0031	RUNNING	0:11	10:00	1	cn0250
6511654_4	small	job_arra	lamx0031	RUNNING	0:11	10:00	1	cn0579
6511654_5	small	job_arra	lamx0031	RUNNING	0:11	10:00	1	cn0602

Job arrays will only run on the cluster you submit the job on.

# Job submission and scheduling

 **More advanced scripted job examples**

# GPU jobs

## Job submission and scheduling

Partition	Node sharing	Cores per node	Walltime limit	Total node memory	Advised memory per core	Local scratch per node	Maximum Nodes per job
k40	Yes	24	24:00:00	124GB	5GB	429GB	40
v100	Yes	24	24:00:00	374GB	15GB	859GB	1
A100-4	Yes	64	24:00:00	499GB	4GB	850GB	4
A100-8	Yes	128	24:00:00	1002GB	7.5GB	850GB	1
Interactive-gpu	Yes	24	24:00:00	60GB	2GB	228GB	2
preempt-gpu	Yes	24	24:00:00	60GB	2GB	228GB	2

Specify the type of GPU using ‘--gres=gpu:<type>:count

```
#!/bin/bash
```

You need to specify the **GRES Generic Resource Scheduling** parameter in your job script

```
#SBATCH -p v100
```

```
#SBATCH --gres=gpu:v100:1
```

```
python script.py
```

In addition to selecting a GPU partition, GPUs need to be requested for all GPU jobs.



## Why run preemptable jobs?

*Same priority value as non-preempt jobs but with a smaller fairshare impact!*

## What happen to my job when it is preempted?

*Cancel or Requeue options: When re-queue, your job must be able to pick up where it left off or can deal with interruption effectively when it restarts.*

```
#!/bin/bash
#SBATCH --nodes=1
#SBATCH --ntasks=10
#SBATCH --mem-per-cpu=1G
#SBATCH --time=01:00:00

#SBATCH --partition preempt
#SBATCH --requeue

module load parallel

# use --resume to deal with interruption
parallel -v --delay .3 -j $SLURM_NTASKS --joblog logs/testjob.log --resume
```

Partitions available: preempt and preempt-gpu

Note: Jobs submitted to preempt queue can be killed at any time to make room for interactive jobs.

Jobs with dependency can be submitted to the scheduler and queued in the system. The execution time of dependent jobs are varied depending on the 'dependency' option specified.

```
#!/bin/bash
job1=$(sbatch --parsable dataprep.sbatch)
job2=$(sbatch --parsable --dependency=afterok:$job1 --kill-on-invalid-dep=yes analyze_data1.sbatch)
job3=$(sbatch --parsable --dependency=afterok:$job2 --kill-on-invalid-dep=yes analyze_data2.sbatch)
sbatch --dependency=afterok:$job3 --kill-on-invalid-dep=yes summary.sbatch
```

Sequential multi step jobs

```
#!/bin/bash
job1=$(sbatch --parsable dataprep.sbatch)
job2=$(sbatch --parsable --dependency=afterok:$job1 --kill-on-invalid-dep=yes analyze_data1.sbatch)
job3=$(sbatch --parsable --dependency=afterok:$job1 --kill-on-invalid-dep=yes analyze_data2.sbatch)
sbatch --dependency=afterok:$job2:$job3 summary.sbatch
# if only depends on either job2 or job3
sbatch --dependency=afterok:$job2?$job3 summary.sbatch
```

Fan-out jobs

Your job begins in the directory that it was submitted from.

### **/home/groupname/<MSI\_Login\_ID> directory (\$HOME)**

- Most of the time you will stage and run your jobs here
- Backed up in ~/.snapshot directory nightly

### **/scratch.local**

- **Temporary storage space** (/tmp or \$TMPDIR) from a compute node.
- The job script must copy the result files back to home directory at the end of execution.

### **/scratch.global/\$USER**

- My job produces many large intermediate files but I only need to keep a few.
- My group quota is not large enough for all of the intermediate and persistent data.
- It's a shared space that is visible to all nodes (including login nodes)
- Data in /scratch.global is not backed up and is deleted after 30 days.

# Job submission and scheduling

 **Monitoring scripted jobs**

## Current jobs

## Monitoring jobs

**queue** command gives all the jobs the scheduler is managing at the moment, use the “-u <your username>” option to list your jobs only.

```
%queue -u lamx0031 -l #-l gives more information about your job
```

```
lamx0031@cn1001 [14:03:43 ~] queue -u lamx0031 -l
Thu Jan 06 14:03:47 2022
```

JOBID	PARTITION	NAME	USER	STATE	TIME	TIME_LIMI	NODES	NODELIST(REASON)
9780263	interacti	bash	lamx0031	RUNNING	40:38	4:00:00	1	cn1001



Executing **queue** sends a remote procedure call to slurmctld. If enough calls from scontrol or other Slurm client commands that send remote procedure calls to the slurmctld daemon come in at once, it can result in a degradation of performance of the slurmctld daemon, possibly resulting in a denial of service.

## Insert slurm email notification statements in the job script

```
#!/bin/bash

#SBATCH --mail-type=begin          # send email when job begins
#SBATCH --mail-type=end            # send email when job ends
#SBATCH --mail-user=<id>@umn.edu

echo "I ran on node: "
hostname
sleep 120
```

```
--mail-type=begin
```

```
Sender: MSI Slurm <msi_slurm@msi.umn.edu>
```

```
Subject: Slurm Job_id=8343769 Name=jobscript.sbatch Began, Queued time 00:05:00
```

```
Body: <empty>
```

Log into the compute node directly using **'ssh'**.


```
%ssh <node id> #log into the compute node (node id returned by squeue command)
```

```
ahl01:~/TMP> srun -N 1 -n10 --mem=24g -t 240 --tmp=50g -p interactive --pty bash
srun: Setting account: support
srun: job 76450054 queued and waiting for resources
srun: job 76450054 has been allocated resources
acn02:~/TMP> █
```

← → ↻ [ondemand.msi.umn.edu/pun/sys/shell/ssh/agate.msi.umn.edu](https://ondemand.msi.umn.edu/pun/sys/shell/ssh/agate.msi.umn.edu)

Host: agate.msi.umn.edu

```
ahl04:~> ssh acn02
!! MSI Cluster Resource
!! Disconnect IMMEDIATELY if you are not an authorized user!
Last login: Mon Oct 9 21:24:01 2023 from acn02.agate.msi.umn.edu
acn02:~> █
```

 The **seff <jobid>** command will output a short summary of the CPU and Memory efficiency of a job.

```
%seff 8457631
Job ID: 8457631
Cluster: mesabi
User/Group: lamx0031/support
State: COMPLETED (exit code 0)
Nodes: 1
Cores per node: 24
CPU Utilized: 06:33:21
CPU Efficiency: 95.29% of 06:52:48 core-walltime
Job Wall-clock time: 00:17:12
Memory Utilized: 726.06 MB
Memory Efficiency: 1.48% of 48.00 GB
```





The `sacct -j <jobid>` command will output a more detailed information about a completed job. You can control what it print using the `--format` option.

```
%sacct -j 8457631 --format=JobID,JobName,MaxRSS,MaxRSSTask,MaxRSSNode,Elapsed
```

JobID	JobName	MaxRSS	MaxRSSTask	MaxRSSNode	Elapsed
8457631	Run_MD_st+				00:17:12
8457631.bat+	batch	743484K	0	cn3007	00:17:12
8457631.ext+	extern	928K	0	cn3007	00:17:12

The (.bat+) is the job submission script where the compute resources are usually consumed and the (.ext+) normally does not consume large resources. The MaxRSS returns the largest resident set size which indicates the memory the job needed for any tasks. The MaxRSSTask gives you where which job step is consumed the largest memory and MaxRSSNode gives you the compute node that carries out such task.

Use **scancel** with the job ID to cancel a job:

```
$ scancel <jobid>
```

You can cancel all your jobs, or all your pending jobs with scancel:

```
$ scancel -u $USER
```

```
$ scancel -t PENDING -u $USER
```

# Job submission and scheduling

 **Troubleshooting tips**

## A typical batch job workflow:

1. You create a job script
  - o Several key resource requests:
    - i. Node (-N), core(-c) How many nodes and cores does your job need?
    - ii. --time, How long does your job need to run?
    - iii. --mem, How much total memory does your job need?
    - iv. -p <partition>, which partition fits your job best?
2. You submit job script with command: **sbatch**
3. You check job status with command: **squeue**
4. When job completes, check output or log file(s)
5. If job failed, modify job script and resubmit (back to step 1)
  - o Check job information with command “**sacct**” or “**seff**”
6. If job *succeeded*, check job information with “**sacct**” or “**seff**”

sbatch messages: **incorrect resource configuration**

```
lamx0031@ln0006 [~] sbatch myscript  
sbatch: error: Setting account: support  
sbatch: error: Memory specification can not be satisfied  
sbatch: error: Batch job submission failed: Requested node configuration is not available
```

For example, if you requested more memory than what a compute node actually has.  
\*Check our Slurm partitions specification webpage for proper configuration settings.

**Do a dry-run using the “--test-only” option with sbatch**

```
%sbatch --test-only myscript.sh  
sbatch: Setting account: support  
sbatch: Job 1433090 to start at 2021-03-10T18:42:53 using 4 processors on nodes cn0166 in partition small
```

A good way to validate the slurm script and return an estimate of when a job would be scheduled to run. No job is actually submitted.

## Out of Memory Issues

```
#!/bin/bash
#SBATCH --j OOM
#SBATCH --ntasks=1
#SBATCH --mem-per-cpu=10M
#SBATCH --time=00:05:00

stress --cpu 1 --io 1 --vm 1 --vm-bytes 128M --timeout 120s
```

After the job exits, run the below **sacct** command to check the status of the job.

```
%sacct -j 10662930 --format=JOBID,JOBNAME,averSS,maxRSS,START,END,NCPUS,NTASK,STATE --unit=M
```

JobID	JobName	AveRSS	MaxRSS	Start	End	NCPUS	NTasks	State
10662930	OOM			2022-01-28T15:07:29	2022-01-28T15:09:34	1		<b>OUT_OF_ME+</b>
10662930.ba+	batch	2.62M	2.62M	2022-01-28T15:07:29	2022-01-28T15:09:34	1	1	<b>OUT_OF_ME+</b>
10662930.ex+	extern	0.89M	0.89M	2022-01-28T15:07:29	2022-01-28T15:09:35	1	1	COMPLETED

## Timeout issues

```
#!/bin/bash
#SBATCH --j OOM
#SBATCH --ntasks=1
#SBATCH --mem-per-cpu=200M
#SBATCH --time=00:01:00
```

```
stress --cpu 1 --io 1 --vm 1 --vm-bytes 128M --timeout 200s
```

After the job exits, run the below **sacct** command to check the status of the job.

```
sacct -j 10663966 --format=JOBID,JOBNAME,averSS,maxRSS,START,END,NCPUS,NTASK,STATE --unit=M
```

JobID	JobName	AveRSS	MaxRSS	Start	End	NCPUS	NTasks	State
10663966	OOM			2022-01-28T15:33:00	2022-01-28T15:34:32	1		<b>TIMEOUT</b>
10663966.ba+	batch	101.92M	101.92M	2022-01-28T15:33:00	2022-01-28T15:34:33	1	1	CANCELLED
10663966.ex+	extern	0.89M	0.89M	2022-01-28T15:33:00	2022-01-28T15:34:33	1	1	COMPLETED



## Find exit code

```
#!/bin/bash
#SBATCH --j OOM
#SBATCH --ntasks=1
#SBATCH --mem-per-cpu=200M
#SBATCH --time=00:010:00
```

```
stress --cpu 1 --io 1 --vm 1 --vm-bytes 128M --timeout 200s #command not found
```

After the job exits, run the below **sacct** command to check the status of the job and check its exit code.

```
sacct -j 10664308 --format=JOBID,JOBNAME,averSS,maxRSS,NCPUS,NTASK,STATE,exitcode --unit=M
```

JobID	JobName	AveRSS	MaxRSS	NCPUS	NTasks	State	ExitCode
10664308	OOM			1		FAILED	<b>127:0</b>
10664308.ba+	batch	1.21M	1.21M	1	1	FAILED	<b>127:0</b>
10664308.ex+	extern	0.90M	0.90M	1	1	COMPLETED	0:0

Code	Meaning	Note
0	Success	Check output
1	General error	Check log files
2	Incorrect use of shell builtins	Check log files
3-124	Job error	check exit code of software
125	Out of memory	
126	Command not executed	
127	Command not found	
128	Invalid argument	
129 -192	Terminated via signal	Subtract 128 from the number and match to signal code



**Check our SLURM webpage**

<https://www.msi.umn.edu/slurm>



**Check our Job FAQ webpage**

<https://www.msi.umn.edu/support/faq/jobs>



**Submit a ticket to the helpdesk**

**Email:** [help@msi.umn.edu](mailto:help@msi.umn.edu)

<https://www.msi.umn.edu/content/helpdesk>



**SLURM cheat sheet**

<https://slurm.schedmd.com/pdfs/summary.pdf>

# Job submission and scheduling

THANK YOU!