Running Jobs on CARC Systems

Center for Advanced Research Computing
University of Southern California
Tomek Osinski
Research Facilitator in Life Science

USC | Advanced Research Computing
Enabling scientific breakthroughs at scale

Slurm

https://slurm.schedmd.com/

- **S**imple **L**inux **U**tility for **R**esource **M**anagement
- Development started in 2002 at Lawrence Livermore National Laboratory
- **Overview** **-** open source, fault-tolerant, and highly scalable cluster management and job scheduling system
- **Main functions**
    - allocates exclusive and/or non-exclusive access to resources (compute nodes) to users for some duration of time so they can perform work
    - provides a framework for starting, executing, and monitoring work (normally a parallel job) on the set of allocated nodes
    - arbitrates contention for resources by managing a queue of pending work
- Configuration specific to an HPC center; **CARC** has its own setup

Some terms

- **Head Node** – The system that controls the cluster
- **Worker (Compute) Node** – Systems that perform the computations in a cluster
- **Login Node** – System that users log into to use a cluster
- **Scheduler** – Software that controls when jobs are run and the node they are run on
- **Shell** – A program that users employ to type commands
- **Script** – A file that contains a series of commands that are executed
- **Job** – A chunk of work that has been submitted to the cluster

How does it work?

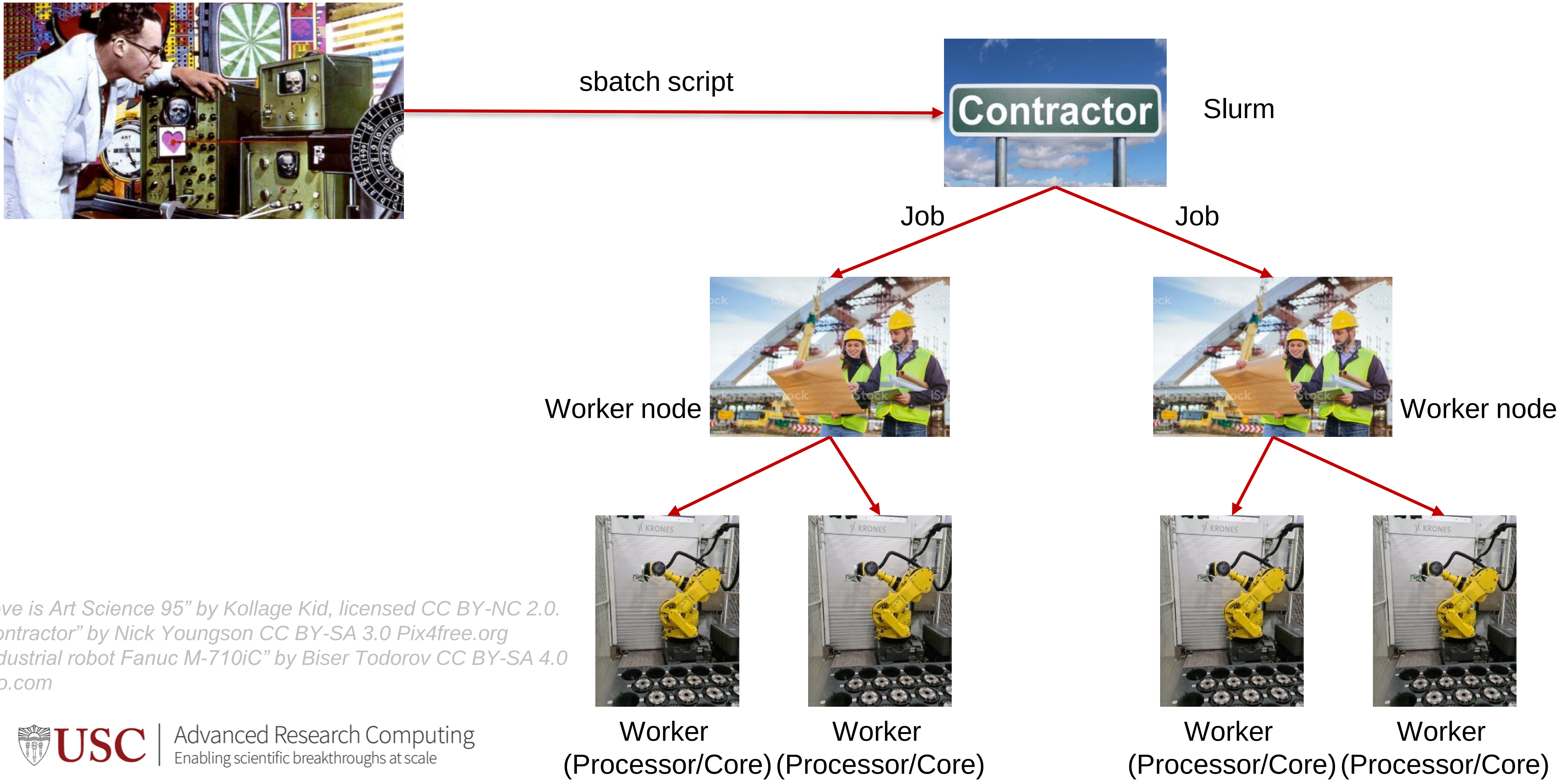
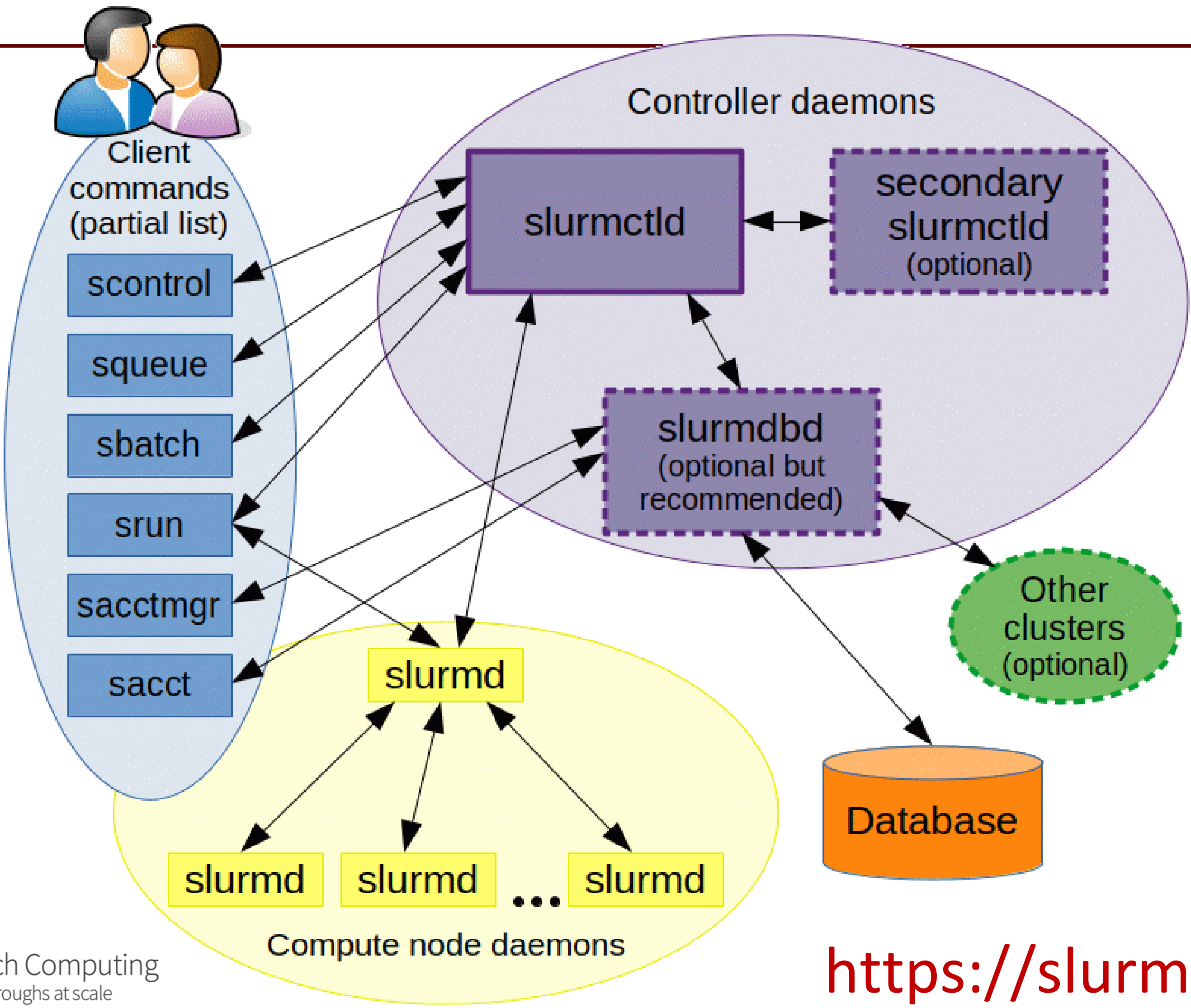


Image: "Love is Art Science 95" by Kollage Kid, licensed CC BY-NC 2.0.
Image: "Contractor" by Nick Youngson CC BY-SA 3.0 Pix4free.org
Image: "Industrial robot Fanuc M-710iC" by Biser Todorov CC BY-SA 4.0
Istockphoto.com

How does it work? – the details


Controller daemons
Client commands (partial list)
scontrol
squeue
sbatch
srun
sacctmgr
sacct
slurmctld
secondary slurmctld (optional)
slurmdbd (optional but recommended)
Other clusters (optional)
Database
slurmd
slurmd
slurmd
...
slurmd
Compute node daemons

https://slurm.schedmd.com/

Commands

https://slurm.schedmd.com/quickstart.html

- **sinfo** – reports state of the partitions and nodes
- **squeue** – reports state of jobs or job steps
- **salloc** – allocates resources for a job in real time
- **sbatch** – submits a job script to queue for a later execution
- **srun** – submits a job for execution or initiate job steps in real time
- **scancel** – is used to cancel a job or job step
- **sprio** – displays a detailed view of the components affecting job’s priority
- **sstat** – is sed to get information about the resources utilized by a job or job step
- **sacct** – is used to report job accounting information about active or completed jobs
- **seff** – is used to display job efficiency for past jobs
- **scontrol** – is used to display or modify slurm configuration and state

Lets get going

- Detailed policies and directions
  - https://carc.usc.edu/user-information/getting-started
- Do not install software yourself, contact us
  - https://carc.usc.edu/education-and-outreach/office-hours:
    - Virtual (Tue, 2:30-5:00)
    - In-person: Leavey Library, room 3L (Tue, 2:30-5:00)
  - Submit a ticket! (https://carc.usc.edu/user-support/)
  - When we install software, it is available to everyone
- Program running slow? *Submit a ticket!*
- Don’t know what resources to use? *Submit a ticket!*
- Any other questions? *Submit a ticket* or *visit our forum*

Log into CARC

- Setup Cisco Anyconnect VPN client and connect to connect.usc.edu
- Open the terminal:
    - Mac: Applications>Utilities>Terminal or open Spotlight and start typing "terminal"
    - Windows: Start menu>cmd (or use MobaXterm, tmux, PuTTY or Cygwin)
    - Linux: System tools>Terminal or Accessories>Terminal or search for Terminal (or any other terminal)
- Type `ssh ttrojan@discovery.usc.edu`
- Enter your password (characters are not visible when typing)
- If successful, your prompt should look something like:
  `[ttrojan@discovery1 ~]`

sinfo

https://slurm.schedmd.com/sinfo.html

- reports state of the partitions and nodes

```
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
debug        up    1:00:00      7   idle a02-26,e05-[42,76,78,80],e09-18,e23-02
epyc-64      up 2-00:00:00     10    mix b22-[10,12-13,15,21-24,29-30]
epyc-64      up 2-00:00:00     22  alloc b22-[01-09,11,14,16-20,25-28,31-32]
main*        up 2-00:00:00    178    mix d05-[05-42],d06-[15-18,20,22-26,28],d18-[01,05,08,12-13,16,23-24,27-30,32-
38],d22-[51-52],e06-[01-04,09-10,12-13,16-19,22],e07-[02,05,08-09,14-16,18],e11-[26,29,45,47],e13-[11,28-29,32,38-47]
main*        up 2-00:00:00     59  alloc d06-[19,21,27],d17-[03-05,22],d18-[17,22,31],e06-[06-08,20-21,24],e07-[01,03]
main*        up 2-00:00:00     68   idle d11-[09-41],d17-[12,18,31-37],d18-[02-26],e06-[05,11,14-15
gpu          up 2-00:00:00     10   resv e17-[10-19]
gpu          up 2-00:00:00     21    mix d11-[02-04],d13-[02,04-07,09],d14-[03-04,07-10],d23-[10,13-14,16],e22-[01-02]
gpu          up 2-00:00:00     46   idle d13-[03,08,10-11],d14-[05-06,11-18],d23-15,e21-[01-16],e22-[03-16],e23-01
oneweek      up 7-00:00:00      2  alloc e02-[45,72]
oneweek      up 7-00:00:00     45   idle e01-[46,48,52,60,62,64,76],e02-[40-44,46,48-71,73-80]
largemem     up 7-00:00:00      2    mix a16-[02-03]
largemem     up 7-00:00:00      1   idle a16-04
```

sinfo (continued)

https://slurm.schedmd.com/sinfo.html

- Useful options `–Node`, `--partition`, and `--states`

```
$ sinfo --partition largemem
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
largemem     up 7-00:00:00      2    mix a16-[02-03]
largemem     up 7-00:00:00      1   idle a16-04
```

```
$ sinfo -lNp largemem
Thu Sep 16 08:12:37 2021
NODELIST   NODES PARTITION       STATE CPUS    S:C:T MEMORY TMP_DISK WEIGHT AVAIL_FE REASON
a16-02         1  largemem       mixed 40     4:10:1 103160        0      1 xeon-485 none
a16-03         1  largemem       mixed 40     4:10:1 103160        0      1 xeon-485 none
a16-04         1  largemem        idle 40     4:10:1 103160        0      1 xeon-485 none
```

- Formatting is manageable through `–format`
- `SINFO_FORMAT` environment variable can be used (`export SINFO_FORMAT="…"`)
- sinfo2 is an alias to `sinfo -o "%60N %10P %8t %8D %10X %10Y %10m %25G %b "`

Codes for common node states

https://slurm.schedmd.com/sinfo.html

- **ALLOCATED –** the node has been allocated to one or more jobs
- **DOWN –** the node is unavailable for use
- **DRAINING –** the node is currently executing a job, but will not be allocated additional jobs
- **IDLE –** the node is available for use
- **MAINT –** the node is currently in a reservation with a flag value of "maintenance"
- **MIXED –** the node has some of its CPUs ALLOCATED while others are IDLE
- **RESERVED –** the node is in advanced reservation and not generally available

sinfo (continued)

https://slurm.schedmd.com/sinfo.html

- Use `--states=idle` to help in choosing a partition for your job to run

```
$ sinfo --states=idle
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
debug        up    1:00:00      7   idle a02-26,e05-[42,76,78,80],e09-18,e23-02
epyc-64      up 2-00:00:00      0    n/a
main*        up 2-00:00:00     48   idle d11-[11,16,18,20,27,41],d17-[11-12,18,31-38],d18-[02-04,06-07,09-11,14-15],e07-
[06-07,10-12],e13-[30-31,33,35-37],e16-[08-12,16-17],e17-[02,04,06]
gpu          up 2-00:00:00     10   resv e17-[10-19]
gpu          up 2-00:00:00     46   idle d13-[03,08,10-11],d14-[05-06,11-18],d23-15,e21-[01-16],e22-[03-16],e23-01
oneweek      up 7-00:00:00     45   idle e01-[46,48,52,60,62,64,76],e02-[40-44,46,48-71,73-80]
largemem     up 7-00:00:00      1   idle a16-04
```

What partition should I use?

https://carc.usc.edu/user-information/user-guides/hpc-basics/discovery-resources

- **debug –** small, short or test jobs; short queue
- **main** (default) – most jobs (serial and small-to-medium), can utilize older K40 gpus
- **epyc-64 –** medium-to-large parallel jobs
- **gpu –** jobs that require GPUs (P100, V100, A100, A40)
- **largemem –** jobs requiring lots of memory (up to 1TB)
- **oneweek –** long-running jobs

What partition should I use? (limits)

https://carc.usc.edu/user-information/user-guides/hpc-basics/discovery-resources

Queue (or partition)	Maximum run time	Maximum concurrent CPUs	Maximum concurrent GPUs	Maximum concurrent memory	Maximum concurrent jobs running	Maximum number of jobs queued
main	48 hours	1,200	36	---	500	5,000
epyc-64	48 hours	1,200	N/A	---	500	5,000
gpu	48 hours	400	36	---	36	100
oneweek	168 hours	208	N/A	---	50	50
largemem	168 hours	120	N/A	1000GB	3	10
debug	1 hour	48	4	---	5	5

sbatch

https://slurm.schedmd.com/sbatch.html

- Submit a job script for remote execution
- Use `module purge` to clear automatically loaded modules
- Use `--mem=0` to request all available memory on a node
- Pack short-running jobs together as job steps
- By default, output log files are named slurm-<jobid>.out and saved to the submit directory with both standard output and standard error messages
- Use `--output` and/or `--error` options to customize them
- Formatting options can be used (e.g., %x = job name -> %x.out)

sbatch (continued)

Option	Default value	Description
--nodes=<number>	1	Number of nodes to use
--ntasks=<number>	1	Number of processes to use
--cpus-per-taks=<number>	1	Number of cores per task
--mem=<number>	2GB	Total memory (single node)
--mem-per-cpu=<number>	2GB	Memory per processor core
--constraint=<attribute>		Node property to request (e.g., xeon-2640v4)
--partition=<partition_name>	main	Request nodes on specified partition
--time=<D-HH:MM:SS>	1:00:00	Maximum run time
--account=<account_id>	Default project account	Account to charge resources to
--mail-type=<value>		Email notifications type; can be: begin, end, fail, all
--mail-user=<address>		Email address
--output=<filename>		File for standard output redirection
--error=<filename>		File for standard error redirection

Create and submit a simple job script

- Use a text editor to create a file sample_job.sh that contains as follows:

```
#!/bin/bash
#SBATCH --nodes 1
#SBATCH --ntasks 1
#SBATCH --partition debug
#SBATCH --time=00:05:00
#SBATCH --chdir /home1/ttrojan/running-jobs-on-CARC-2022
#SBATCH --account=<account_id>
module purge
module load gcc/9.2.0
echo “Example start”
echo `date`
sleep 30
echo “Example end”
```

- Then submit it

```
$ sbatch sample_job.sh
Submitted batch job 154837654
```

sbatch (continued)

Variable	Description
SLURM_JOB_ID	The ID of the job allocation
SLURM_JOB_NODELIST	List of nodfes allocated to the job
SLURM_JOB_NUM_NODES	Total number of nodes in the job’s resource allocation
SLURM_NTASKS	Number of tasks requested
SLURM_CPUS_PER_TASK	Number of CPUs requested per task
SLURM_SUBMIT_DIR	The directory from which sbatch was invoked
SLURM_ARRAY_TASK_ID	Job array ID (index) number

Variables example for a job script

- Use a text editor to create a file sample_var_job.sh that contains as follows:

```
#!/bin/bash
#SBATCH --nodes 1
#SBATCH --ntasks 1
#SBATCH --partition debug
#SBATCH --time=00:05:00
#SBATCH --chdir /home1/ttrojan/running-jobs-on-CARC-2022
#SBATCH --account=<account_id>
module purge
module load gcc/9.2.0
echo “Job ID: $SLURM_JOB_ID”
echo “Nodelist: $SLURM_JOB_NODELIST”
cd $SLURM_SUBMIT_DIR
echo `pwd`
```

- Then submit it

```
$ sbatch sample_var_job.sh
Submitted batch job 154837654
```

srun

https://slurm.schedmd.com/srun.html

- Launch parallel tasks or job steps for MPI jobs
- More details on using MPI:
  https://carc.usc.edu/user-information/user-guides/software-and-programming/mpi
- Use `srun --help` for more options

Variables example for a job script

- Use a text editor to create a file sample_mpi_job.sh that contains as follows:

```
#!/bin/bash
#SBATCH --nodes 3
#SBATCH --ntasks 30
#SBATCH --partition main
#SBATCH --time=00:10:00
#SBATCH --chdir /home1/ttrojan/running-jobs-on-CARC-2022
#SBATCH --account=<account_id>
#SBATCH --exclusive
module purge
module load gcc/8.3.0
module load openmpi/4.0.2
module load pmix/3.1.3
ulimit -s unlimited
srun --mpi=pmix_2 --ntasks $SLURM_NTASKS ./mpi_app
```

- Then submit it

```
$ sbatch sample_mpi_job.sh
Submitted batch job 154837654
```

salloc

https://slurm.schedmd.com/salloc.html

- Allocates resources for an interactive job
- Shares most options with `sbatch`
- Example interactive session with the use of K40 GPUs:

```
[ttrojan@discovery1 ~]$ salloc --time=2:00:00 --cpus-per-task=8 --gres=gpu:k40:2 --partition=main
salloc: Granted job allocation 5919107
salloc: Waiting for resource configuration
salloc: Nodes e16-03 are ready for job
[osinski@e16-03 ~]$ hostname
e16-03.hpc.usc.edu
[ttrojan@e16-03 ~]$ nvidia-smi -L
GPU 0: Tesla K40m (UUID: GPU-1f625725-19f5-b4f7-ad27-1901ee9b12f5)
GPU 1: Tesla K40m (UUID: GPU-3ed86dc4-3046-74e0-4983-9b8bd01a0671)
[ttrojan@e16-03 ~]$ exit
exit
salloc: Relinquishing job allocation 5919108
[ttrojan@discovery1 ~]$
```

squeue

https://slurm.schedmd.com/squeue.html

- Displays status of jobs and job steps
- `squeue --help`
- All jobs:

```
[ttrojan@discovery1 ~]$ squeue | head
           4679566      main discover    userX  R    2:19:33      8 d23-[13,15-16],e21-14,e22-[08-09,12],e23-01
           4680126      main discover    userY  R      39:11      8 d23-[13-16],e22-[05-06,08-09]
           4678655      main job.slur    userZ  R   11:09:20      1 d14-08
           4679445      main  1086-7B  ttrojan  R    4:18:00      1 d11-46
           4679444      main  1086-7B  ttrojan  R    4:19:31      1 d05-40
```

- Just your jobs:

```
[ttrojan@discovery1 ~]$ squeue -u ttrojan
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
           3678639   epyc-64   test_1  ttrojan PD       0:00      4 (Resources)
           3678721   epyc-64   test_2  ttrojan PD       0:00      4 (Priority)
           3675759   epyc-64   test_3  ttrojan  R 1-01:48:12      2 b22-[29-30]
```

Codes for common job states

https://slurm.schedmd.com/squeue.html

- **PD PENDING –** Job is awaiting resource allocation
- **R RUNNING –** Job currently has an allocation
- **CD COMPLETED –** Job has terminated on all nodes with an exit code of zero
- **CG COMPLETING –** Job is in the process of completing. Some processes on some nodes may still be active
- **CA CANCELLED –** Job was explicitly cancelled by the user or system administrator. The job may or may not have been initiated

Codes for common pending reason

https://slurm.schedmd.com/squeue.html

- **Resources –** Job is waiting for resources to become available
- **Priority –** One or more higher priority jobs exist for this partition or advanced reservation
- **ReqNodeNotAvail –** Some node specifically required by the job is not currently available
- **QOSMaxCpuPerUserLimit –** The job has reached the maximum CPU per user limit
- **QOSMaxGresPerUser –** The job has reached the maximum GPU per user limit
- **AssocGrpCPUMinutesLimit –** The project account has run out of CPU time
- **InvalidAccount –** the job's account is invalid

squeue (continued)

https://slurm.schedmd.com/squeue.html

- Useful options: `--start` and `--partition`
- Formatting options with `--format` or `--Format`
- Can use environment variable (`export SQUEUE_FORMAT="..."`)
- Create an alias `alias myq="squeue –u $USER"`
- And add it to your .bashrc file

Job priorities

https://slurm.schedmd.com/fair_tree.html

- Based on fairshare algorithm and job age
- Fairshare values depend on a number of factors:
  - Number of jobs submitted
  - Resources used
  - Resources requested
  - project account activity

sprio

https://slurm.schedmd.com/sprio.html

- Display job priority information
- Can be difficult to interpret
- After normalizing, a priority value closer to 1 means a higher priority

```
[ttrojan@discovery1 ~]$ sprio -j 5918718
          JOBID PARTITION   PRIORITY       SITE        AGE  FAIRSHARE    JOBSIZE  PARTITION        QOS                 TRES
        5918718 main            1142          0          5        136          1       1000          0          cpu=0,mem=1
[ttrojan@discovery1 ~]$ sprio -j 5918718 -n
          JOBID PARTITION PRIORITY   AGE        FAIRSHARE  JOBSIZE    PARTITION  QOS        TRES
        5918718 main      0.00000026 0.0048570  0.0135671  0.0010957  1.0000000  0.0000000  cpu=0.00,mem=0.00
```

USC | Advanced Research Computing
Enabling scientific breakthroughs at scale

scancel

https://slurm.schedmd.com/scancel.html

- Cancel pending or running jobs
- `scancel --help`

```
[ttrojan@discovery1 ~]$ scancel 2918718

[ttrojan@discovery1 ~]$ scancel –u ttrojan
```

sstat

https://slurm.schedmd.com/sstat.html

- Display status information for running jobs
- `sstat --help`

```
[ttrojan@discovery1 ~]$ sstat -j <jobid>

[ttrojan@discovery1 ~]$ sstat -j <jobid> --format=JobID,MaxRSS,AveCPUFreq,MaxDiskRead,MaxDiskWrite
```

USC | Advanced Research Computing
Enabling scientific breakthroughs at scale

sacct

https://slurm.schedmd.com/sacct.html

- Display accounting information for past jobs
- `sacct --help`
- By default only jobs from past day
- Useful options `--starttime`, `--endtime`, --brief, and `--state`

```
[ttrojan@discovery1 ~]$ sacct -j <jobid>

[ttrojan@discovery1 ~]$ sacct -j <jobid> --format=JobID,MaxRSS,AveCPUFreq,MaxDiskRead,MaxDiskWrite,State,ExitCode
```

Job exit codes

https://slurm.schedmd.com/job_exit_code.html

- Exit status, 0-255
- 0 -> success, completed
- Non-zero -> failure
- Codes 1-127 indicate error in job
- Exit codes 129-255 indicate jobs terminated by Unix signals
- `man signal`

seff

- Display job efficiency information for past jobs (CPU and memory use)
- Is used to optimize resource requests
- `sacct --help`
- By default only jobs from past day
- Check also our wrapper `jobinfo <JOB_ID>`

```
$ seff 5919108
Job ID: 5919108
Cluster: discovery
User/Group: ttrojan/ttrojan
State: COMPLETED (exit code 0)
Nodes: 1
Cores per node: 8
CPU Utilized: 00:00:01
CPU Efficiency: 0.13% of 00:12:24 core-walltime
Job Wall-clock time: 00:01:33
Memory Utilized: 2.53 MB
Memory Efficiency: 0.02% of 16.00 GB
```

USC | Advanced Research Computing
Enabling scientific breakthroughs at scale

scontrol

https://slurm.schedmd.com/scontrol.html

- Display or modify slurm configuration and state
- Mostly for admins, some commands for users
- `scontrol --help`
- Examples:

```
scontrol show partition <partition>

scontrol show node <nodeid>

scontrol show job <jobid>

scontrol hold <jobid>

scontrol release <jobid>
```

Job dependencies

- Are allowing to submit at once a set of jobs from a larger pipeline
- Defer the start of a job until the specified dependencies have been satisfied
- Examples:
  - `-d depend=afterok:jobid[:jobid...]`

    Starts after jobid has finished without errors.
  - `-d depend=afternotok:jobid[:jobid...]`

    Starts after jobid has finished with errors.
  - `-d depend=afterok:jobid,before:jobid2`

    Starts after jobid is finished, but not until jobid2 has started.
  - `-d depend:afterok:jobid -d depend:afterok:jobid2`

    Starts after both jobid and jobid2 have finished.
  - `-d depend=afterokarray:jobid`

    Starts after the job array jobid has finished without errors.
  - `-d depend=after:jobid[+time][:jobid[+time]...]`

    Starts after the job array jobid in minutes specified in ‘time’ or without delay if no ‘time’ is given
  - `-d depend=afterany:jobid`

    Starts after the job jobid has finished regardless of exit code.

Job dependencies

Example:

- **Step 1:**

```
[ttrojan@discovery1 ~]$ sbatch preprocessing-step.sh
Submitted batch job 18866
```

- **Step 2:**

```
[ttrojan@discovery1 ~]$ sbatch -d after:18866 job-array-step.sh
Submitted batch job 18870
```

- **Step 3:**

```
[ttrojan@discovery1 ~]$ sbatch –d afterok:18870 postprocessing-step.sh
Submitted batch job 18867
```

Job Arrays

- A way to run the same commands on many (hundreds, thousands) of datasets/samples.
- A variable called `$SLURM_ARRAY_TASK_ID` is used to determine the element of the array being run.
- `#SBATCH --array=1-1000`
- `$SLURM_ARRAY_TASK_ID` becomes 1 in first job, 2 in second job, etc…
- Modify job or application to use index

Without Job Arrays – Numbered Files

```
#!/bin/bash
#SBATCH --nodes 1
#SBATCH --ntasks 1
#SBATCH --partition main
#SBATCH --time 00:05:00
#SBATCH --chdir /home1/ttrojan/running-jobs-on-CARC-2022
#SBATCH --account=<account_id>
module purge
module load gcc/9.2.0
module load fastqc
echo "Starting FastQC job‟
fastqc -o results/fastqc-rawseq-ordered raw-seq-ordered/yeast_1_50K.fastq
fastqc -o results/fastqc-rawseq-ordered raw-seq-ordered/yeast_2_50K.fastq
fastqc -o results/fastqc-rawseq-ordered raw-seq-ordered/yeast_3_50K.fastq
fastqc -o results/fastqc-rawseq-ordered raw-seq-ordered/yeast_4_50K.fastq
fastqc -o results/fastqc-rawseq-ordered raw-seq-ordered/yeast_5_50K.fastq
fastqc -o results/fastqc-rawseq-ordered raw-seq-ordered/yeast_6_50K.fastq
echo "Finish FastQC job"
```

Job Arrays – Numbered Files

- Here is an example SLURM script for a job array. Save as `fastqc_numbered_array.job`

```
#!/bin/bash
#SBATCH --nodes 1
#SBATCH --ntasks 1
#SBATCH --partition main
#SBATCH --time 00:05:00
#SBATCH --array=1-6
#SBATCH --chdir /home1/ttrojan/running-jobs-on-CARC-2022
#SBATCH --account=<account_id>
module purge
module load gcc/9.2.0
module load fastqc
echo "Starting FastQC job"
sleep 20
fastqc -o results/fastqc-rawseq-ordered-arr raw-seq-
ordered/yeast_${SLURM_ARRAY_TASK_ID}_50K.fastq
echo "Finish FastQC job"
```

View Job Array

squeue -u uscnetid

```
[ttrojan@disocvery1 running-jobs-on-CARC-2022]$ squeue -u ttrojan
JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
1152 main     bash ttrojan  R    2:17:32      1 d05-40
1153 main     bash ttrojan  R    2:17:12      1 d05-40
1207_1 main numbered ttrojan  R       0:02      1 d05-41
1207_2 main numbered ttrojan  R       0:02      1 d05-40
1207_3 main numbered ttrojan  R       0:02      1 d05-42
1207_4 main numbered ttrojan  R       0:02      1 d05-45
1207_5 main numbered ttrojan  R       0:02      1 d05-44
1207_6 main numbered ttrojan  R       0:02      1 d05-44
```

Job Arrays – Unnumbered Files

- Start by creating a list of all of the unnumbered filenames
- Then create slurm array script for fastqc jobs that have unnumbered filenames

```
#!/bin/bash
#SBATCH --nodes 1
#SBATCH --ntasks 1
#SBATCH --partition main
#SBATCH --time 00:05:00
#SBATCH --array=1-6
#SBATCH --chdir /home1/ttrojan/running-jobs-on-CARC-2022
#SBATCH --account=<account_id>
module purge
module load gcc/9.2.0
module load fastqc echo "Starting FastQC job‟
sleep 20
ls raw-seq/ > unnumbered-filenames.txt
line=$(sed -n -e "$SLURM_ARRAY_TASK_ID p" unnumbered-filenames.txt)
fastqc -o results/fastqc-rawseq-unordered raw-seq/${line}
echo "Finish FastQC job"
```

Important Things to Note

- Job length
  - If over 24 hours, can this be split up, can threads be increased?
- Many small files
  - To be avoided!
  - Group into larger files
- Data
  - Save space by removing temp files
  - Archive data as soon as reasonable
  - Let us know if you are adding several TB of data
  - Use /scratch1 or /scratch2 whenever possible for temporary files

Important Things to Note

- Make sure you are not on the login node when you launch an application
  - You can check the system you are on by typing `hostname`
- Make sure you reserve as many processors as you need
  - A mismatch here can increase your runtime or wait time
- Make sure you reserve as much RAM as needed
  - Overestimating increases wait time, underestimating crashes
- Know which resources work the best
  - Sometimes using a debug or epyc-64 is better

Resources

- CARC home page
  - https://carc.usc.edu
- CARC User Forum
  - https://hpc-discourse.usc.edu/categories
- SLURM tutorials
  - https://slurm.schedmd.com/tutorials.html
- SLURM quick reference
  - https://slurm.schedmd.com/pdfs/summary.pdf

Resources

- CARC home page
  - https://carc.usc.edu
- CARC User Forum ← the most value for the community!
  - https://hpc-discourse.usc.edu/categories
- SLURM tutorials
  - https://slurm.schedmd.com/tutorials.html
- SLURM quick reference
  - https://slurm.schedmd.com/pdfs/summary.pdf

Review: Interactive Jobs

- When you need to provide unpredictable input

```
[ttrojan@discovery1 ~]$ hostname
discovery1.usc.edu
[ttrojan@discovery1 ~]$ salloc -p debug
[ttrojan@a02-26 ~]$ hostname
a02-26.hpc.usc.edu
[ttrojan@a02-26 ~]$ exit
exit
[ttrojan@discovery1 ~]$ hostname
discovery1.usc.edu

[ttrojan@discovery1 ~]$
```

Review: Bash Scripts

- Bash scripts are a series of commands that can be grouped together within files to accomplish a series of tasks
- This allows you to run one command instead of several successive commands

Exercise:

- Start an interactive job to the debug queue
- This program sleeps for 10 seconds and then prints out "Hello World"
- Make this file, give it execute permissions, and run

```
#!/bin/bash
# This program: sleeps for 10 seconds, then prints "Hello World"
sleep 10
echo "Hello World"
```

Prepare to Run Jobs

- Copy example data to your home directory

```
[ttrojan@discovery1 ~]$

[ttrojan@discovery1 ~]$ git clone https://github.com/uschpc/running-jobs-on-CARC-2022.git

[ttrojan@discovery1 ~]$ cd running-jobs-on-CARC-2022

[ttrojan@discovery1 ~]$ ls
```

Example: Create the FastQC Job Script

- Use a text editor to create a file name samplefastqc.sh that contains what follows:

```
#!/bin/bash
#SBATCH --nodes 1
#SBATCH --ntasks 1
#SBATCH --partition debug
#SBATCH --chdir /home1/ttrojan/running-jobs-on-CARC-2022
#SBATCH --account=<account_id>
module purge
module load gcc/9.2.0
module load fastqc
echo “Example FastQC start”
sleep 20
fastqc -o results/fastqc-rawseq raw-seq/yeast_1_50K.fastq
echo “Example FastQC end”
```

Example: Run the FastQC Job Script

- Submit the job

```
[ttrojan@discovery1 ~]$ sbatch fastqc1.sh

Submitted batch job 33723
```

- Check the status of the job

```
[ttrojan@discovery1 ~]$ squeue –u ttrojan
JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
33723 debug fastqc.s ttrojan  R       0:02      1 a02-26
```

Example: Check Output File for Errors

- Check Output File for Errors

```
[ttrojan@discovery1 ~]$ cat slurm-33723.out
Started analysis of yeast_1_50K.fastq
Approx 5% complete for yeast_1_50K.fastq
Approx 10% complete for yeast_1_50K.fastq
Approx 15% complete for yeast_1_50K.fastq
Approx 20% complete for yeast_1_50K.fastq
Approx 25% complete for yeast_1_50K.fastq
Approx 30% complete for yeast_1_50K.fastq
Approx 35% complete for yeast_1_50K.fastq
Approx 40% complete for yeast_1_50K.fastq
Approx 45% complete for yeast_1_50K.fastq
Approx 50% complete for yeast_1_50K.fastq
Approx 55% complete for yeast_1_50K.fastq
Approx 60% complete for yeast_1_50K.fastq
Approx 65% complete for yeast_1_50K.fastq
Approx 70% complete for yeast_1_50K.fastq
Approx 75% complete for yeast_1_50K.fastq
Approx 80% complete for yeast_1_50K.fastq
Approx 85% complete for yeast_1_50K.fastq
Approx 90% complete for yeast_1_50K.fastq
Approx 95% complete for yeast_1_50K.fastq
Approx 100% complete for yeast_1_50K.fastq
Analysis complete for yeast_1_50K.fastq
```

Example: Create the BLAST Job Script

- Replace swissprot with the path to the v5 of swissprot db obtained from

  https://carc.usc.edu/user-information/bio-resources/genbank

```
#!/bin/bash
#SBATCH --nodes 1
#SBATCH --ntasks 10
#SBATCH --partition debug
#SBATCH --chdir /home1/ttrojan/running-jobs-on-CARC-2022
#SBATCH --account=<account_id>
#SBATCH –time 00:05:00
module purge
module load gcc/9.2.0
module load blast-plus
echo "Start BLAST Job"
blastp -db swissprot -query blast/query.txt -out results/blast/results.txt -num_threads
$SLURM_NTASKS
echo "Finish BLAST Job"
```

Example: Run the BLAST Job Script

- Submit the job

```
[ttrojan@discovery1 ~]$ sbatch blast1.sh

Submitted batch job 4773117
```

- Check the status of the job

```
[ttrojan@discovery1 ~]$ squeue -u ttrojan
JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
4773117      Main   blast1.s  ttrojan  R       0:02      1 a02-d11
```

Example: Check BLAST Job Stats with sacct

- sacct can get stats for a job after its completed

https://slurm.schedmd.com/sacct.html

```
[ttrojan@discovery1 ~]$ sacct -j 4773117 --format=JobID,State,Elapsed,NCPUS,MaxRSS
```

```
[ttrojan@discovery1 ~]$ sacct -j 4773117 --format=JobID,State,Elapsed,NCPUS,MaxRSS
       JobID      State    Elapsed      NCPUS     MaxRSS
------------ ---------- ---------- ---------- ----------
4773117       COMPLETED   00:00:09         10
4773117.bat+  COMPLETED   00:00:09         10      1228K
4773117.ext+  COMPLETED   00:00:09         10       832K
```

What is Wrong

```
#!/bin/bash
# ----------------SLURM Parameters----------------
#SBATCH --partition main
#SBATCH --ntasks 1
#SBATCH --mem=1g
#SBATCH --nodes 1
#SBATCH --chdir /home1/ttrojan/running-jobs-on-CARC-2022
#SBATCH --account=<account_id>
# ----------------Load Modules--------------------
# ----------------Commands------------------------
python3 /home1/ttrojan/script.py
```

What is Wrong

- The module is not loaded

```
#!/bin/bash
# ----------------SLURM Parameters----------------
#SBATCH --partition main
#SBATCH --ntasks 1
#SBATCH --mem=1g
#SBATCH --nodes 1
#SBATCH --chdir /home1/ttrojan/running-jobs-on-CARC-2022
#SBATCH --account=<account_id>
# ----------------Load Modules--------------------
module purge
module load gcc/9.2.0
module load python/3.7.6
# ----------------Commands------------------------
python3 /home1/ttrojan/script.py
```

What is Wrong II

```
#!/bin/bash
# ----------------SLURM Parameters----------------
#SBATCH --partition main
#SBATCH --ntasks 20
#SBATCH --mem=10g
#SBATCH --nodes 1
# ----------------Load Modules--------------------
module purge
module load gcc/9.2.0
module load blast-plus
# ----------------Commands------------------------
blastn -query fasta.file -db database_name -outfmt 6 \
-num_alignments 1 -num_descriptions 1 -out output_file
```

What is Wrong II

- Number of processors and no working directory

```
#!/bin/bash
# ----------------SLURM Parameters----------------
#SBATCH --partition main
#SBATCH --ntasks 20
#SBATCH --mem=10g
#SBATCH --nodes 1
#SBATCH --chdir /home1/ttrojan/running-jobs-on-CARC-2022
#SBATCH --account=<account_id>
# ----------------Load Modules--------------------
module purge
module load gcc/9.2.0
module load blast-plus
# ----------------Commands------------------------
blastn -query fasta.file -db database_name -outfmt 6 num_alignments 1 \ -num_descriptions 1 -out
output_file -num_threads 20
```

What is Wrong II

- Number of processors and no working directory
- Better to use $SLURM_NTASKS

```
#!/bin/bash
# ----------------SLURM Parameters----------------
#SBATCH --partition main
#SBATCH --ntasks 20
#SBATCH --mem=10g
#SBATCH --nodes 1
#SBATCH --chdir /home1/ttrojan/running-jobs-on-CARC-2022
#SBATCH --account=<account_id>
# ----------------Load Modules--------------------
module purge
module load gcc/9.2.0
module load blast-plus
# ----------------Commands------------------------
blastn -query fasta.file -db database_name -outfmt 6 num_alignments 1 \ -num_descriptions 1 -out
output_file -num_threads $SLURM_NTASKS
```

What is Wrong III

```
#!/bin/bash
# ----------------SLURM Parameters----------------
#SBATCH --partition main
#SBATCH --ntasks 1
#SBATCH --mem=200g
#SBATCH --nodes 1
#SBATCH --chdir /home1/ttrojan/running-jobs-on-CARC-2022
#SBATCH --account=<account_id>
# ----------------Load Modules--------------------
module purge
module load gcc/8.3.0
module load R
# ----------------Commands------------------------
Rscript /home1/ttrojan/R_example.R
```

What is Wrong III

- Wrong partition/mem requirements too high

```
#!/bin/bash
# ----------------SLURM Parameters----------------
#SBATCH --partition epyc-64
#SBATCH --ntasks 1
#SBATCH --mem=200g
#SBATCH --nodes 1
#SBATCH --chdir /home1/ttrojan/running-jobs-on-CARC-2022
#SBATCH --account=<account_id>
# ----------------Load Modules--------------------
module purge
module load gcc/8.3.0
module load R
# ----------------Commands------------------------
Rscript /home1/ttrojan/R_example.R
```

What is Wrong IV

```
#!/bin/bash
# ----------------SLURM Parameters----------------
#SBATCH --partition gpu
#SBATCH --nodes 1
#SBATCH --mem=4g
#SBATCH --ntasks 1
#SBATCH --chdir /home1/ttrojan/running-jobs-on-CARC-2022
# ----------------Load Modules--------------------
module purge
module load gcc/8.3.0
Module load cuda/10.0.130
# ----------------Commands------------------------
python3 /home1/ttrojan/my_gpu_accelerated_script.py
```

What is Wrong IV

- GPU resources not specified

```
#!/bin/bash
# ----------------SLURM Parameters----------------
#SBATCH --partition gpu
#SBATCH --nodes 1
#SBATCH --mem=4g
#SBATCH --ntasks 1
#SBATCH --gres=gpu:p100:1
#SBATCH --chdir /home1/ttrojan/running-jobs-on-CARC-2022
#SBATCH --account=<account_id>
# ----------------Load Modules--------------------
module purge
module load gcc/8.3.0
Module load cuda/10.0.130
# ----------------Commands------------------------
python3 /home1/ttrojan/my_gpu_accelerated_script.py
```

What is Wrong V

```
# ----------------SLURM Parameters----------------
#SBATCH --partition main
#SBATCH --ntasks 1
#SBATCH --mem=15g
#SBATCH --nodes 1
#SBATCH --chdir /home1/ttrojan/running-jobs-on-CARC-2022
#SBATCH --account=<account_id>
# ----------------Load Modules--------------------
module purge
module load gcc/9.2.0
module load samtools
# ----------------Commands------------------------
samtools stats example.bam
```

What is Wrong V

- No bash shebang line, #!/bin/bash
- Can use long names for SBATCH parameters

```
#!/bin/bash
# ----------------SLURM Parameters----------------
#SBATCH --partition main
#SBATCH --ntasks 1
#SBATCH --mem=15g
#SBATCH --nodes 1
#SBATCH --chdir /home1/ttrojan/running-jobs-on-CARC-2022
#SBATCH --account=<account_id>
# ----------------Load Modules--------------------
module purge
module load gcc/9.2.0
module load samtools
# ----------------Commands------------------------
samtools stats example.bam
```