

# Introduction to SLURM & SLURM batch scripts

Anita Orendt

Assistant Director

Research Consulting & Faculty Engagement

[anita.orendt@utah.edu](mailto:anita.orendt@utah.edu)

30 August 2018

# Overview of Talk

- Basic SLURM commands
- SLURM batch directives
- Accounts and Partitions
- SLURM Environment Variables
- SLURM Batch scripts
- Running an Interactive Batch job
- Where to get more Information

# Basic SLURM commands

- **sinfo** - shows partition/node state
- **sbatch <scriptname>** - launches a batch script
- **squeue** - shows all jobs in the queue
  - **squeue -u <username>** - shows only your jobs
- **scancel <jobid>** - cancels a job

Notes:

For **sinfo**, **squeue** – can add **-M all** to see all clusters using given slurm installation (np, kp, lp, ember, ash)

Can also use full path

/uufs/<cluster>/sys/pkg/slurm/std/bin/<command> to look at other clusters, their queue, or submit or cancel jobs

# Some Useful Aliases

- Bash to add to .aliases file:

```
alias si="sinfo -o \" %20P %5D %14F %8z %10m %10d %11l %16f %N\""
```

```
alias si2="sinfo -o \" %20P %5D %6t %8z %10m %10d %11l %16f %N\""
```

```
alias sq="squeue -o \" %8i %12j %4t %10u %20q %20a %10g %20P %10Q %5D %11l %11L %R\""
```

- Tcsh to add to .aliases file:

```
alias si 'sinfo -o \" %20P %5D %14F %8z %10m %11l %16f %N\"'
```

```
alias si2 'sinfo -o \" %20P %5D %6t %8z %10m %10d %11l %N\"'
```

```
alias sq 'squeue -o \" %8i %12j %4t %10u %20q %20a %10g %20P %10Q %5D %11l %11L %R\"'
```

Can add **-M all** to **si** and **sq** also

# SLURM Batch Directives

- #SBATCH --time 1:00:00 ← wall time of a job (or -t)
- #SBATCH --partition=name ← partition to use (or -p)
- #SBATCH --account=name ← account to use (or -A)
- #SBATCH --nodes=2 ← number of nodes (or -N)
- #SBATCH --ntasks 12 ← total number of tasks (or -n)
- #SBATCH --mail-type=FAIL,BEGIN,END ← events on which to send email
- #SBATCH --mail-user=name@example.com ← email address to use
- #SBATCH -o slurm-%j.out-%N ← name for stdout; %j is job#, %N node
- #SBATCH -e slurm-%j.err-%N ← name for stderr; %j is job#, %N node
- #SBATCH --constraint "C20" ← can use features given for nodes (or -C)

# Accounts and Partitions

- You need to specify an account and partition to run jobs
- You can see a list of partitions using the `sinfo` command
- For general allocation usage the partition is the cluster name
- If no allocation (or out of allocation) use *clustername*-freecycle for partition
- Your account is typically your PI's name (e.g., if your PI is Baggins, use the "baggins" account) – there are a few exceptions!
- Private node accounts and partition have the same name – PI last name with cluster abbreviation, e.g., baggins-kp, baggins-em, etc
- Private nodes can be used as a guest using the "owner-guest" account and the *cluster*-guest partition

# More on Accounts

Allocations and node ownership status	What resource(s) are available
No general allocation, no owner nodes	<p><u>Unallocated general nodes</u>  <u>Allocated general nodes in freecycle mode</u> - not recommended  <u>Guest access on owner nodes</u></p>
General allocation, no owner nodes	<p><u>Unallocated general nodes</u>  <u>Allocated general nodes</u>  <u>Guest access on owner nodes</u></p>
Group owner nodes, no general allocation	<p><u>Unallocated general nodes</u>  <u>Allocated general nodes in freecycle mode</u> - not recommended  <u>Group owned nodes</u>  <u>Guest access on owner nodes of other groups</u></p>
Group owner node, general allocation	<p><u>Unallocated general nodes</u>  <u>Allocated general nodes</u>  <u>Group owned nodes</u>  <u>Guest access on owner nodes of other groups</u></p>

# Query your accounts/partitions

```
~]$ sacctmgr -p show assoc user=u0028729 format=cluster,account,partition,qos
```

```
Cluster|Account|Partition|QOS|
```

```
notchpeak|chpc||notchpeak|
```

```
kingspeak|kingspeak-gpu||kingspeak-gpu|
```

```
ember|ember-gpu||ember-gpu|
```

```
ash|smithp-guest||ash-guest,ash-guest-res|
```

```
lonepeak|chpc||lonepeak|
```

```
kingspeak|chpc||kingspeak|
```

```
lonepeak|owner-guest||lonepeak-guest|
```

```
ember|owner-guest||ember-guest|
```

```
kingspeak|owner-guest||kingspeak-guest|
```

```
kingspeak|owner-gpu-guest||kingspeak-gpu-guest|
```

```
ember|chpc||ember|
```

***Note that partition field is empty – for the most part partition and qos are paired***



# SLURM Environment Variables

- Depends on SLURM Batch Directives used
- Can get them for a given set of directives by using the `env` command inside a script (or in a `srun` session).
- Some useful environment variables:
  - `SLURM_JOBID`
  - `SLURM_SUBMIT_DIR`
  - `SLURM_NNODES`
  - `SLURM_NTASKS`

# Basic SLURM script flow

1. Set up the #SBATCH directives for the scheduler to request resources for job
2. Set up the working environment, by loading appropriate modules
3. If necessary, add any additional libraries or programs to \$PATH and \$LD\_LIBRARY\_PATH, or set other environment needs
4. Set up temporary/scratch directories if needed
5. Switch to the working directory (often group/scratch)
6. Run the program
7. Copy over any results files needed
8. Clean up any temporary files or directories

# Basic SLURM script - bash

```
#!/bin/bash
#SBATCH --time=02:00:00
#SBATCH --nodes=1
#SBATCH -o slurmjob-%j.out-%N
#SBATCH -e slurmjob-%j.err-%N
#SBATCH --account=owner-guest
#SBATCH --partition=kingspeak-guest

#Set up whatever package we need to run with
module load somemodule

#set up the temporary directory
SCRDIR=/scratch/general/lustre/$USER/$SLURM_JOBID
mkdir -p $SCRDIR

#Set up the path to the working directory
cp file.input $SCRDIR/
cd $SCRDIR

#Run the program with our input
myprogram < file.input > file.output

cp file.output $HOME/
cd $HOME

rm -rf $SCRDIR
```

# Basic SLURM script - tcsh

```
#!/bin/tcsh
#SBATCH --time=02:00:00
#SBATCH --nodes=1
#SBATCH -o slurmjob-%j.out-%N
#SBATCH -e slurmjob-%j.err-%N
#SBATCH --account=owner-guest
#SBATCH --partition=kingspeak-guest

#Set up whatever package we need to run with
module load somemodule

#set up the scratch directory
set SCRDIR /scratch/local/$USER/$SLURM_JOBID
mkdir -p $SCRDIR

#move input files into scratch directory
cp file.input $SCRDIR/
cd $SCRDIR

#Run the program with our input
myprogram < file.input > file.output
cp file.output $HOME/
cd $HOME

rm -rf $SCRDIR
```

# Parallel Execution

- MPI installations at CHPC are SLURM aware, so mpirun will usually work without a machinefile (unless you are manipulating the machinefile in your scripts)
- If machinefile or host list needed, create the node list:
  - `srun hostname | sort -u > nodefile.$SLURM_JOBID`
  - `srun hostname | sort > nodefile.$SLURM_JOBID`
- Alternatively, you can use the srun command instead, but you need to compile with a more recently compiled MPI
- Mileage may vary, and for different MPI distributions, srun or mpirun may be preferred (check our slurm page on the CHPC website for more info or email us)

# Running interactive batch jobs

- An interactive command is launched through the `srun` command

```
srun --time=1:00:00 --nodes=1 --account=chpc  
      --partition=ember --pty /bin/tcsh -l
```

- Launching an interactive job automatically forwards environment information, including X11 forwarding
- "`--pty`" must be set to shell preferred for the session (either `/bin/tcsh` or `/bin/bash`)
- `-l` (lower case "L") at the end required

# Slurm for use of GPU Nodes

- Ember – 11 GPU nodes
  - All are general nodes, each with M2090 cards
- Kingspeak – 8 GPU nodes
  - 2 general nodes each with 4 Tesla K80 cards (8 GPUs)
  - 2 general nodes each with 8 GeForce TitanX cards
  - 4 owner nodes each with 2 Tesla P100 cards (owned by School of Computing)
- Notchpeak – 4 GPU nodes
  - 3 general nodes each with 3 Tesla V100 cards
  - 1 owner node with 4 TitanV cards
- Use partition and account set to **cluster-gpu**
- Must get added to account – request via [helpdesk@chpc.utah.edu](mailto:helpdesk@chpc.utah.edu)
- Use only if you are making use of the GPU for the calculation
- Most codes do not yet make efficient use of multiple GPUs so we have enabled node sharing
- See <https://www.chpc.utah.edu/documentation/guides/gpus-accelerators.php>

# Node Sharing on GPU nodes

- Need to specify number of CPU cores, amount of memory, and number of GPU
- Core hours used based on highest % requested among cores, memory and GPUs

Option	Explanation
<code>#SBATCH --gres=gpu:k80:1</code>	request one K80 GPU (others types names are titanx, m2090, p100, v100, titanv)
<code>#SBATCH --mem=4G</code>	request 4 GB of RAM (default is 2GB/core if not specified)
<code>#SBATCH --mem=0</code>	request all memory of the node; use this if you do not want to share the node as this will give you all the memory
<code>#SBATCH --tasks=1</code>	requests 1 core



# Strategies for Serial Applications

- <https://www.chpc.utah.edu/documentation/software/serial-jobs.php>
- When running serial applications (no MPI, no threads) unless memory constraint, you should look to options to bundle jobs together so using all cores on nodes
- There are multiple ways to do so
  - `srun --multi-prog`
  - submit script
- Also consider OSG as an option (especially if you have a large number of single core, short jobs)

# Strategies for Job Arrays

- <https://www.chpc.utah.edu/documentation/software/slurm.php#jobarr>
- Useful if you have many similar jobs when each use all cores on a node or multiple nodes to run where only difference is input file
- `sbatch --array=1-30%n myscript.sh` – where `n` is maximum number of jobs to run at same time
- In script: use `$SLURM_ARRAY_TASK_ID` to specify input file:
  - `./myprogram input$SLURM_ARRAY_TASK_ID.dat`

# Job Priorities

- <https://www.chpc.utah.edu/documentation/software/slurm.php#priority>
- **sprio** give job priority for all jobs
  - sprio -j JOBID for a given job
  - sprio -u UNID for all a given user's jobs
- Combination of three factors added to base priority
  - Time in queue
  - Fairshare
  - Job size

# Slurm Documentation at CHPC

<https://www.chpc.utah.edu/documentation/software/slurm.php>

<https://www.chpc.utah.edu/documentation/software/serial-jobs.php>

<https://www.chpc.utah.edu/documentation/software/slurm.php#jobarr>

<https://www.chpc.utah.edu/documentation/guides/index.php#GenSlurm>

## Other good documentation sources

<http://slurm.schedmd.com/documentation.html>

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

<http://www.schedmd.com/slurmdocs/rosetta.pdf>

# Getting Help

- CHPC website
  - [www.chpc.utah.edu](http://www.chpc.utah.edu)
    - Getting started guide, cluster usage guides, software manual pages, CHPC policies
- Service Now Issue/Incident Tracking System
  - Email: [helpdesk@chpc.utah.edu](mailto:helpdesk@chpc.utah.edu)
- Help Desk: 405 INSCC, 581-6440 (9-5 M-F)
- We use [chpc-hpc-users@lists.utah.edu](mailto:chpc-hpc-users@lists.utah.edu) for sending messages to users