PURDUE UNIVERSITY

# RESEARCH COMPUTING
# PBS → SLURM TRANSITION

January 31, 2020

# Queues

PURDUE UNIVERSITY

**There will be no significant changes to the scheduling policies on any systems for Slurm.**

**However, Slurm does not have “queues”.**
**Instead, Slurm uses “accounts” which serve the same function:**

```
$ qsub -q standby myjob.sub
123456.cluster-adm.rcac.purdue.edu
```

```
$ sbatch -A standby myjob.sub
Submitted batch job 123456
```

**You may still use “`qlist`” to see which accounts (queues) you can submit to:**

```
$ qlist
                        Current Number of Cores                          Node
Queue            Total     Queue      Run     Free     Max Walltime      Type
==============  ================================  ==============  ======
debug               96         0        0       96         0:30:00         A
standby         12,864   283,059    5,878    1,000         4:00:00         A
myqueue            120         0      120        0       336:00:00         A
```

```
$ qlist
                        Current Number of Cores                          Node
Account          Total     Queue      Run     Free     Max Walltime      Type
==============  ================================  ==============  ======
debug               96         0        0       96         0:30:00         A
standby         12,864   283,059    5,878    1,000         4:00:00         A
myqueue            120         0      120        0       336:00:00         A
```

# Submitting Jobs

PURDUE UNIVERSITY

**Use "`sbatch`" instead of "`qsub`" to submit batch jobs:**

```
$ qsub -q standby myjob.sub
123456.cluster-adm.rcac.purdue.edu
```

```
$ sbatch -A standby myjob.sub
Submitted batch job 123456
```

***If the submission script is not a file in the current directory, Slurm will search in your $PATH.***

**Any Slurm options may also be put inside your job script using "`#SBATCH`" instead of "`#PBS`":**

```
#/bin/bash
#PBS -q standby
```

```
#/bin/bash
#SBATCH -A standby
```

# Job Environment

PURDUE UNIVERSITY

**Jobs will start in your current working directory as when you submitted them.**
**Remove any "`cd $PBS_O_WORKDIR`":**

```
#/bin/bash
#PBS -q standby

cd $PBS_O_WORKDIR
```

```
#/bin/bash
#SBATCH -A standby

# job starts in current directory by default
```

**Jobs inherit all your current environment variables unless you specify "`--export=NONE`":**

```
$ sbatch --export=NONE myjob.sub
```

**You can also add a specific variable as part of your job submission:**

```
$ sbatch --export=NONE,myextravar=somevalue myjob.sub
$ sbatch --export=ALL,myextravar=somevalue  myjob.sub
```

# Slurm Environment Variables

PURDUE UNIVERSITY

**Within your job, Slurm provides many environment variables to help in your scripts:**

| PBS / Torque | Description | Example | Slurm |
|---|---|---|---|
| `$PBS_JOBID` | Job ID | 123456 | `$SLURM_JOB_ID` |
| `$PBS_JOBNAME` | Job Name | myjobname | `$SLURM_JOB_NAME` |
| `$PBS_QUEUE` | Queue / Account | standby | `$SLURM_JOB_ACCOUNT` |
| `$PBS_O_WORKDIR` | Submission Directory | /scratch/cluster/myusername | `$SLURM_SUBMIT_DIR` |
| `$PBS_NUM_NODES` | Total Number of Nodes | 5 | `$SLURM_JOB_NUM_NODES` |
| `$PBS_NP` | Total Number of Tasks | 80 | `$SLURM_NTASKS` |
| `$PBS_NUM_PPN` | Number of Tasks per Node | 16 | `$SLURM_NTASKS_PER_NODE` |
| - | Node List (Compact Form) | cluster-a[000-003,008] | `$SLURM_JOB_NODELIST` |
| `LIST=$(cat $PBS_NODEFILE)` | Node List (One Core per Line) | cluster-a000 | `LIST=$(srun hostname)` |
| `$PBS_ARRAYID` | Job Array Index Number | 43 | `$SLURM_ARRAY_TASK_ID` |

**See "OUTPUT ENVIRONMENT VARIABLES" in the `sbatch` man page for more: "`man sbatch`"**

# Job Nodes & Tasks

PURDUE UNIVERSITY

**PBS required you to specify the number of nodes and the number of tasks on each node.**

**In Slurm, you may specify the number of nodes (`-N`), the total number of tasks (`-n`), and/or the number of tasks on each node (`--ntasks-per-node`):**

```
$ qsub -l nodes=2:ppn=16 myjob.sub
```

```
$ sbatch -N 2  -n 32                 myjob.sub
$ sbatch -N 2  --ntasks-per-node 16  myjob.sub
$ sbatch -n 32 --ntasks-per-node 16  myjob.sub
```

$$TotalTasks = Nodes \cdot TasksPerNode$$

# Job Time & Name

PURDUE UNIVERSITY

**Specify the total time your job needs (walltime) using "-t":**

```
$ qsub -l walltime=4:00:00 myjob.sub
```

```
$ sbatch -t    4:00:00 myjob.sub
$ sbatch -t 1-12:00:00 myjob.sub    # 1 day 12 hours
```

**You may give your job a custom name in job listings using "-J":**

```
$ qsub -N MyCustomName myjob.sub
```

```
$ sbatch -J MyCustomName myjob.sub
```

# Job Email

PURDUE UNIVERSITY

**Slurm can email your @purdue.edu address when your job reaches certain points.**

**To get an email when your job starts, completes, or fails:**

```
$ qsub -m bea myjob.sub
```

```
$ sbatch --mail-type=BEGIN,END,FAIL myjob.sub
```

**To get an email when your job reaches a certain percentage of its walltime limit:**

```
$ sbatch --mail-type=TIME_LIMIT_90 myjob.sub
$ sbatch --mail-type=TIME_LIMIT_80 myjob.sub
$ sbatch --mail-type=TIME_LIMIT_50 myjob.sub
```

# Interactive Jobs

PURDUE UNIVERSITY

Use "`sinteractive`" instead of "`qsub -I`" to submit interactive jobs:

```
cluster-fe00 $ qsub -I -X
qsub: waiting for job 123456.cluster-
adm.rcac.purdue.edu to start
qsub: job 123456.cluster-adm.rcac.purdue.edu ready
cluster-a000 $
```

```
cluster-fe00 $ sinteractive
salloc: Granted job allocation 123456
salloc: Waiting for resource configuration
salloc: Nodes cluster-a000 are ready for job
cluster-a000 $
```

`sinteractive` is a custom Purdue addition to Slurm to make it easier to start interactive jobs.

`sinteractive` will also take most of the same options as `sbatch`.

# Job Steps & Sub-jobs

PURDUE UNIVERSITY

**You can use Slurm to manage the *components* of a job as well as the whole job.**

**Use "`srun`" *within* your sbatch job script for fine-grained control over where commands in the script run. Here we see two separate commands, each running on 10 cores of the 20 total cores:**

```
#!/bin/bash
#SBATCH -N 2 -n 20

srun -n 10 myfirstcommand
srun -n 10 mysecondcommand
```

# MPI

PURDUE UNIVERSITY

**MPI programs can run under mpiexec / mpirun in the same manner as PBS:**

```
#!/bin/bash
#SBATCH -N 2 -n 32

module load intel
module load impi

mpiexec -n $SLURM_NTASKS ./mpi_hello
```

**You can also use `srun` for more control, but will need to tell Slurm which type of MPI to use:**

```
#!/bin/bash
#SBATCH -N 2 -n 32

module load intel
module load impi

srun -n $SLURM_NTASKS --mpi=pmi2 ./mpi_hello
```

```
#!/bin/bash
#SBATCH -N 2 -n 32

module load gcc
module load openmpi

srun -n $SLURM_NTASKS --mpi=openmpi ./mpi_hello
```

# Job Status

PURDUE UNIVERSITY

## Check the status of your job with "squeue" instead of "qstat":

```
$ qstat -a myqueue
$ qstat -a -U myusername
$ qstat -a

                                                                  Req'd
Job ID              Username    Queue    Jobname  NDS  TSK   Time      S
------------------  ----------  -------  -------  ---  ---  --------  -
123456.clustername  myusername  standby  testjob    2   48  00:30:00  Q
```

```
$ squeue -A myaccount
$ squeue -u myusername
$ squeue

JOBID   USER        ACCOUNT  NAME     NODES  CPUS  TIME_LIMIT  ST  TIME
123456  myusername  standby  testjob      2    48       30:00  PD  0:00
```

## Get an estimate for your job's starting time using "squeue":

```
$ showstart 123456

job 123456@3600 requires 24 procs for 1:00:00
Earliest start in       1:01:39 on Wed Mar 15 10:30:45
Earliest completion in  5:01:39 on Wed Mar 15 14:30:45
```

```
$ squeue -u myusername -O jobid,starttime

JOBID               START_TIME
123456              2020-02-05T18:00:00
```

# Job Information

PURDUE UNIVERSITY

**`jobinfo` will show you the full history of a job, for both completed jobs and jobs in progress:**

```
$ jobinfo 123456
Name                : myjobname
User                : myusername
Account             : myqueue
Partition           : cluster-a
Nodes               : cluster-a[000,005,100,101]
Cores               : 64
GPUs                : 0
State               : COMPLETED
ExitCode            : 0:0
Submit              : 2020-01-15T20:55:00
Start               : 2020-01-15T21:25:00
End                 : 2020-01-16T10:55:00
Waited              :     0:30:00
Reserved walltime   : 14-00:00:00
Used walltime       :    13:30:00
Max Mem used        : 2.1G (cluster-a005)
[…]
```

***You can see this job only used 13.5 hours of the 14 days requested, and started in 30 minutes.***

***This job also peaked at 2.1G of memory used, and this was on the second node in the set.***

# Job History

PURDUE UNIVERSITY

**You can get a list of *all* your jobs, both completed and in progress, using `sacct`:**

```
$ sacct -X -u myusername

       JobID    JobName  Partition    Account  AllocCPUS      State ExitCode
------------ ---------- ---------- ---------- ---------- ---------- --------
123456         myjobname  cluster-a    myqueue         64    RUNNING      0:0
789012        interacti+ cluster-s+    standby          2  COMPLETED      0:0
```

***If needed, you can then get more details on each job using `jobinfo`.***

# Deleting Jobs

PURDUE UNIVERSITY

**Use “`scancel`” instead of “`qdel`” to cancel or delete jobs:**

```
$ qdel 123456
```

```
$ scancel 123456
```

# Job Output

PURDUE UNIVERSITY

**It is no longer necessary to look for output on the node with "`qpeek`".**
**Slurm will write output and error messages as they happen in real time.**

***Slurm will put output and error into the same file in the directory from which you submitted.***
***You can change this with the "`-e`" (error file) and "`-o`" (output file) options.***

**You can use "`%j`" (and other variables) to customize these filenames:**

```
$ sbatch -J myjobname -o %x-%u-%j.out myjob.sub
Submitted batch job 123456

$ ls
myjob.sub
myjobname-myusername-123456.out
```

# Node Sharing

**All kernel security patches are being applied, including Meltdown & Spectre.**
**This removes the primary reason node sharing was not able to be offered previously.**

**Therefore, Slurm *will allow node sharing* on Research Computing clusters.**

**You can still request exclusive nodes using the "`--exclusive`" option to `sbatch` or `srun`:**

```
$ qsub myjob.sub
```

```
$ sbatch --exclusive myjob.sub
```

***An equivalent to "`naccesspolicy=singleuser`" can still be achieved by using `srun` in a script.***

# Memory Use

**Provide a memory estimate for your jobs!**
**Slurm controls memory just like nodes and cores.**

***Slurm will assign a default value if you do not.***
***This default will be proportional to the number of cores you requested on the node.***
***If you exceed the memory requested per node—even the default, your job will be killed!***

**This job asked for the default one core on one node and the default memory slice.**
**On a 24-core node with 96GB nodes, if this job exceeds 4GB, *it will be killed*:**

```
$ sbatch myjob.sub
```

**This job will be able to use up to 90GB of memory *per node*:**

```
$ sbatch --mem=90G myjob.sub
```

# Requesting GPUs

PURDUE UNIVERSITY

**If you use GPUs, such as on Gilbreth, you will need to request these in your job.**

**You can do this by giving the total number of GPUs needed (`--gpus`), the number per node (`--gpus-per-node`), or the number per task (`--gpus-per-task`):**

```
$ qsub -l nodes=2:gpus=2 myjob.sub
```

```
$ sbatch -N 2 --gpus=4           myjob.sub
$ sbatch -N 2 --gpus-per-node=2  myjob.sub
$ sbatch -N 2 --gpus-per-task=2  myjob.sub
```

# Questions?

PURDUE UNIVERSITY

| | |
|---|---|
| **Email Help:** | rcac-help@purdue.edu |
| **Drop-in Coffee Hours:** | **Monday–Thursday, 2:00pm**<br>**Various Locations**<br>rcac.purdue.edu/coffee |
| **Friday Talks:** | **Fridays, 2:00pm**<br>**Envision Center**<br>rcac.purdue.edu/news/events |

**Acknowledgements to Purdue Research Computing staff:**
**Stephen Kelley, Lev Gorenstein, Kevin D. Colby**