Submitting SLURM Batch Jobs

SLURM

SLURM is the batch job scheduling system for the SLAC batch compute systems. Generic documentation about SLURM can be found in this Quick Start User Guide.

S3DF Computing Facility

LCLS specific information about the S3DF facility is here: Running at S3DF.

General S3DF documentation is here: https://s3df.slac.stanford.edu

Some hardware details about the S3DF can be found here, although note that 8 of the milano cluster node cores are reserved for the filesystem so only 120 (out of 128) can be used on each node for batch processing: https://s3df.slac.stanford.edu/public/doc/#/batch-compute?id=clusters-amp-repos

Partitions

The partition/queue information can be provided by the `sinfo` command.

sinfo

LCLS users typically use the "milano" queue at s3df:

sinfo

```
[cpo@sdfiana002 ~]$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
roma*        up 10-00:00:0      1   comp sdfrome004
roma*        up 10-00:00:0     16  drng@ sdfrome[006-018,041-043]
roma*        up 10-00:00:0      1  down$ sdfrome003
roma*        up 10-00:00:0      1 drain$ sdfrome037
roma*        up 10-00:00:0      1  drain sdfrome005
roma*        up 10-00:00:0     21    mix sdfrome[019-036,038-040]
milano       up 10-00:00:0      1  inval sdfmilan221
milano       up 10-00:00:0     14  drng@ sdfmilan[036-038,120-121,126,129,204-205,212,229-232]
milano       up 10-00:00:0      4 drain$ sdfmilan[009,041,049,112]
milano       up 10-00:00:0      1  drain sdfmilan032
milano       up 10-00:00:0     12   resv sdfmilan[001-005,029-030,052,057,117-119]
milano       up 10-00:00:0    102    mix sdfmilan[006-008,010-019,021-028,031,033-035,039-040,042-048,050-
051,053-056,058-072,101-111,113-116,122-125,127-128,130-131,201-203,206-211,213-220,222-228]
milano       up 10-00:00:0      1   idle sdfmilan020
ampere       up 10-00:00:0      1  drng@ sdfampere010
ampere       up 10-00:00:0      1   drng sdfampere011
ampere       up 10-00:00:0      3  drain sdfampere[005,008,023]
ampere       up 10-00:00:0     18    mix sdfampere[001-004,006-007,009,012-022]
[cpo@sdfiana002 ~]$
```

The "*" following the roma queue name indicates that it is a default queue for submission.

Job Submission

sbatch

The following is a simple submission script of a parallel psana batch job run with mpi. It can be submitted with the command "`sbatch submit.sh`". The commands specified in the script file will be ran on the first available compute node that fits the resources requested. There are two ideas: "nodes" and "tasks per node". A "node" is a physical computer box (with a host-name, for example) but each box/node typically has multiple-cpu-cores. Typically the tasks-per-node parameter is set to utilize all the cores on each node.

NOTE: when running interactively the "mpirun" command takes a "-n" argument with a number of cores. However, when running with slurm no "-n" is needed, as slurm infers it from the "--nodes" and "--ntasks-per-node" values.

```
> cat submit.sh #!/bin/bash

#SBATCH --partition=milano
#SBATCH --nodes=2
#SBATCH --ntasks-per-node=120
#SBATCH --output=%j.log

# -u flushes print statements which can otherwise be hidden if mpi hangs
mpirun python -u -m mpi4py.run my_psana_script.py
```

One can also do this same command from the command line using the "--wrap" option for sbatch:

```
sbatch -p milano --nodes 2 --ntasks-per-node 3 --wrap="mpirun python -u -m mpi4py.run my_psana_script.py"
```

srun

In principle the slurm "srun" command can also be used to launch parallel jobs, however the current S3DF "srun" version only supports an older "pmi2" protocol, which is incompatible the mpi packages from conda that LCLS uses which use the newer "pmix" protocol. srun should be avoided for parallel jobs at S3DF (see output of "srun --mpi=list").

Monitoring/Status

squeue

To check that jobs that exist on the system use the s*queue* command:

squeue

```
[cpo@sdfiana002 ~]$ squeue -u ytl
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
          30703603 ampere,ro      out      ytl PD       0:00      1 (launch failed requeued held)
          30703602 ampere,ro      out      ytl PD       0:00      1 (launch failed requeued held)
          30701730 ampere,ro      out      ytl PD       0:00      1 (launch failed requeued held)
          30700739 ampere,ro      out      ytl PD       0:00      1 (launch failed requeued held)
          30700738 ampere,ro      out      ytl PD       0:00      1 (launch failed requeued held)
          30699545 ampere,ro      out      ytl PD       0:00      1 (launch failed requeued held)
          30704838    milano      out      ytl CG       4:07      1 sdfmilan221
[cpo@sdfiana002 ~]$
```

The ST (job state) field shows that most jobs are pending (PD) and one is completing (CG).

sacct

Get information about status of finished jobs

sacct

```
[cpo@sdfiana002 ~]$ sacct -u ytl
JobID           JobName  Partition    Account  AllocCPUS      State ExitCode
------------ ---------- ---------- ---------- ---------- ---------- --------
30682524            out     milano shared:de+        112  PREEMPTED      0:0
30682524.ba+      batch            shared:de+        112  CANCELLED     0:15
30682524.ex+     extern            shared:de+        112  COMPLETED      0:0
30682525            out     milano shared:de+        112  PREEMPTED      0:0
30682525.ba+      batch            shared:de+        112  CANCELLED     0:15
30682525.ex+     extern            shared:de+        112  COMPLETED      0:0
```

Misc Slurm commands

scontrol is used to view or modify Slurm configuration including: job, job step, node, partition, reservation, and overall system configuration. Most of the commands can only be executed by user root or an Administrator.

- Detail job information: `scontrol show jobid -dd <jobID>`
- Show reservation: `scontrol show res`

sacctmgr is used to deal with accounts, associations and users.
Format can be modified at will or removed to see all (can be hard to read, especially on smaller windows):

- Show what account a user is associated with: `sacctmgr show associations Users=<user_names> format=cluster,account,partition,QOS`
- Show priorities for an account: `sacctmgr list associations -p accounts=<accounts>`
- Show priority coefficients: `sacctmgr show qos format=name,priority,usagefactor`

Others

- Show priority level for a job: `sprio -j <jobID>`