Monitoring Jobs

Careful examination of running times, memory usage and output files will allow you to ensure the job completed correctly and give you a good idea of what memory and time limits to request in the future.

Monitoring Completed Jobs:

To see the runtime and memory usage of a job that has completed, use the `sacct` command:

```
sacct
```

Lists all jobs by the current user and displays information such as JobID, JobName, State, and ExitCode.

```
[cathrine98@login.crane erik]$ sacct
    JobID  JobName  Partition  Account  AllocCPUS  State  ExitCode
7957748  bash  devel  swanson   1  COMPLETED  0:0
7957751  gamedes  devel  swanson   1  FAILED  1:0
7957751.bat+  batch  devel  swanson   1  FAILED  1:0
7957779  gamedes  devel  swanson  16  COMPLETED  0:0
7957779.bat+  batch  devel  swanson   8  COMPLETED  0:0
```

Coupling this command with the `--format` flag will allow you to see more than the default information about a job. Fields to display should be listed as a comma separated list after the `--format` flag (without spaces). For example, to see the Elapsed time and Maximum used memory by a job, this command can be used:

```
sacct --format JobID,JobName,Elapsed,MaxRSS
```

```
[cathrine98@login.crane erik]$ sacct --format JobID,JobName,Elapsed,MaxRSS
    JobID  JobName  Elapsed  MaxRSS
7957748  bash  00:22:22  2448K
7957751  gamedes  00:00:01  822K
7957751.bat+  batch  00:00:01  822K
7957779  gamedes  00:00:06  882K
7957779.bat+  batch  00:00:06  882K
```

Additional arguments and format field information can be found in the SLURM documentation.

Monitoring Running Jobs:

There are two ways to monitor running jobs, the `top` command and monitoring the cgroup files. `Top` is helpful when monitoring multi-process jobs, whereas the cgroup files provide information on memory usage. Both of these tools require the use of an interactive job on the same node as the job to be monitored.

```
If the job to be monitored is using all available resources for a node, the user will not be able to obtain a simultaneous interactive job.
```

After the job to be monitored is submitted and has begun to run, request an interactive job on the same node using the `srun` command:
srun --jobid=<JOB_ID> --pty bash

Where `<JOB_ID>` is replaced by the job id for the monitored job as assigned by SLURM.

Alternately, you can request the interactive job by nodename as follows:

srun --nodelist=<NODE_ID> --pty bash

Where `<NODE_ID>` is replaced by the node name that the monitored job is running. This information can be found out by looking at the `squeue` output under the Nodelist column.

Once the interactive job begins, you can run `top` to view the processes on the node you are on:

```
Top - 12:22:52 up 20 days, 1:08, 2 users, load average: 1.11, 0.29, 0.14
Tasks: 577 total, 1 running, 576 sleeping, 0 stopped, 0 zombie
Cpu(s): 47.6%us, 5.8%sy, 0.0%ni, 46.4%id, 0.1%wa, 0.0%hi, 0.0%si, 0.0%st
Mem: 65804336k total, 5362168k used, 60442168k free, 342228k buffers
Swap: 20 total, 0k used, 0k free, 2209352k cached
```

Output for `top` displays each running process on the node. From the above image, we can see the various MATLAB processes being run by user cathrine98. To filter the list of processes, you can type `u` followed by the username of the user who owns the processes. To exit this screen, press `q`.

During a running job, the cgroup folder is created which contains much of the information used by `sacct`. These files can provide a live overview of resources used for a running job. To access the cgroup files, you will need to be in an interactive job on the same node as the monitored job. To view specific files, and information, use one of the following commands:

To view current memory usage:

```
less
/cgroup/memory/slurm/uid_<UID>/job_<SLURM_JOB_ID>/memory.usage_in_bytes
```

Where `<UID>` is replaced by your UID and `<SLURM_JOB_ID>` is replaced by the monitored job's Job ID as assigned by Slurm.
To view maximum memory usage from start of job to current point:

```
less /cgroup/memory/slurm/uid_<UID>/job_<SLURM_JOB_ID>/memory.max_usage_in_bytes
```