Blueshark Cluster

How Do I Run Code On Blueshark?

To run a job using Blueshark, you must submit it through Blueshark's scheduler. This differs from how you'd normally run a command, as you need to prepare a submission script and optionally make your program MPI capable.

Blueshark uses SLURM (Simple Linux Utility for Resource Management) to manage available resources and to distribute jobs to free nodes. Slurm also provides a queueing system; if not enough resources are available, it will hold your job until it can run it.

Slurm Submission Script

In order to submit a job to slurm, a job submission script must be created.

A sample submission script is provided below.

```bash
#!/bin/bash
#SBATCH --job-name TestJob
#SBATCH --nodes 2
#SBATCH --ntasks 2
#SBATCH --mem=50MB
#SBATCH --time=00:15:00
#SBATCH --partition=short
#SBATCH --error=testjob.%J.err
#SBATCH --output=testjob.%J.out
module load mpich
echo "Starting at `date`"
echo "Running on hosts: $SLURM_NODELIST"
echo "Running on $SLURM_NNODES nodes."
echo "Running on $SLURM_NPROCS processors."
echo "Current working directory is `pwd`"
```

The only options you absolutely need are

- `--job-name` Will give a unique name to your job. This can be set to anything.
- `--nodes` number of nodes to request.
- `--ntasks` number of tasks in total accross all nodes. Note that this differs from torque's `ppn`, which assigns this number of tasks to every node.
- `--mem` amount of memory to request on each node. This is a hard limit and you will run into out-of-memory errors if you fail to provide the...
Blueshark Cluster

correct amount.
- --partition Sets the partition for your job. Valid partitions can be found by using sinfo.

If you need MPICH to run your jobs, set it to load at login using:

module initadd mpich

or by placing

module load mpich

in your submission script like above. Many more options are available for Slurm's submission scripts which can be found here.

Submitting a Job

Like Torque, Slurm has its own set of commands for job management. To submit your submission script, use

sbatch script.sh

Some other commands you may want to use are listed below.

- squeue similar to showq for torque. list the jobs that are currently running for everyone.
- sinfo Show node status.
- scancel cancel a currently running job.
- sstat show statistics for a job.

For a more in-depth look into Slurm and its respective commands, check out their quick start guide.

Partitions

Partitions are very similar to Torque's queue system. Jobs that only need a short amount of time to run, but a large amount of processors will have their jobs categorized differently than jobs that may need to run for days and need less processors. In addition, partitions can be used to group together nodes that have general hardware that others don't (ex, gpu partition has GPUs in its nodes). Currently, these partitions exist:

<table>
<thead>
<tr>
<th>Partition Name</th>
<th>Max Compute Time</th>
<th>Max Nodes</th>
<th>Allowed Groups</th>
</tr>
</thead>
<tbody>
<tr>
<td>short</td>
<td>45 minutes</td>
<td>N/A</td>
<td>blueshark users</td>
</tr>
</tbody>
</table>
Blueshark Cluster

<table>
<thead>
<tr>
<th></th>
<th>med</th>
<th>N/A</th>
<th>blueshark users</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>long</td>
<td>7 days</td>
<td>blueshark users</td>
</tr>
<tr>
<td></td>
<td>eternity</td>
<td>infinite</td>
<td>blueshark users</td>
</tr>
<tr>
<td></td>
<td>class</td>
<td>10 minutes</td>
<td>Parallel Programming class</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>blueshark users</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>users</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>infinite</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

To set a partition, use:

`#SBATCH --partition=[partition]`

in your submission script, or specify it on the command line using --partition.

**Running GPU Jobs**

Running GPU jobs is very similar to running regular jobs. An extra parameter has to be passed (--gres) and the partition must be set to gpu.

`#SBATCH --partition=gpu`

will set your partition.

`#SBATCH --gres=gpu:[#]`

will set the number of GPUs you want **per node**. Note that this differs from ntask, specified earlier. --gres will request n number of GPUs from each node. Thus, if you request 4 nodes with --gres=gpu:2, you will have [4 nodes] * [2 gpu/node] = 8 GPUs in total. This option will not exceed 4, as we only have 4 GPUs per node.

GPU's can also be selected based on whether or not they support GPUDirect technology. Each GPU node has 2 standard GPUs and 2 GPUDirect enabled GPUs. To select between the two, you can use:

`#SBATCH --gres=gpu:[type]:[#]`

where [type] is either gpudirect or standard.

**Optimizing your Submission Script**

Slurm will attempt to run your job wherever it can place it. This is hugely dependent on how your submission script specifies its resources. Thus, if you can reduce your
Blueshark Cluster

Membrane requirements, your job has a much higher chance of being scheduled faster.

**Memory Requirements**

`--mem` is most often used to specify the amount of memory your job will take per node. However, this is largely dependent on how many tasks you can fit in a node, or the number of nodes you'll require. If you don't specify the number of nodes you need, slurm won't balance out the tasks, often leading to out-of-memory errors on nodes where more jobs were placed than expected. Another issue arises when the cluster is under heavy use. Small pockets of resources are scattered through the cluster, and won't be easy to acquire when your job needs a fixed amount of memory per node.

To prevent this, we can use `--mem-per-cpu` instead. If each task only requires a certain amount of memory, you can specify this amount instead. This way, the scheduler can better allocate resources — if tasks require more memory than what's available on a node, they'll be split, and if there are pockets of resources a single task can fit in, it will allocate that spot.

**Common Errors and Solutions**

`slurmstepd: error: Exceeded job memory limit at some point.`

The job you ran tried using more memory than what was defined in your submission script. As a result, slurm automatically killed your job.

A simple fix is to increase the amount of memory dedicated to your job, using `--mem` at the command line or `"#SBATCH --mem"` in your submission script.

`error: Batch job submission failed: Requested time limit is invalid (missing or exceeds some limit)`

You attempted to submit a job to a partition that didn't support your `--time` option.

The solution is to move your job to a partition with a longer execution time (med, long, etc.)

By default, jobs are sent to the short queue, which only permits at most 45 minutes. Specify a partition in your submission script, or reduce your `--time` option.

`SSH: Access denied: user [username] (uid=[uid]) has no active jobs.`

This error comes up when you attempt to ssh into a node that you're not currently
Blueshark Cluster

running a job on. Under normal circumstances, you should not run jobs directly on the nodes as this can confuse the scheduler and prevent other users from submitting jobs. If you're unable to use the scheduler to submit your job, and you absolutely need to ssh in (for example, X11 forwarding), see this section of Slurm's FAQ.

SFTP: Received message too long [random number]

This error comes up when you attempt to use sftp and you have a message printed via a *profile config in your directory (.bash_profile, etc.) To correct the issue, remove the printing message and try again.

sbatch: error: Batch script contains DOS[MAC] line breaks (\n)
sbatch: error: instead of expected UNIX line breaks (\n).

Sometimes, if you download a SLURM submission script to a Windows or Mac computer and re-upload to Blueshark, you may get this error when attempting to submit the script using sbatch. The solution is to run "dos2unix" on the file.

Unique solution ID: #1368
Author: Daniel Campos
Last update: 2016-07-19 14:50