Running a job on HPC cluster using PBS and Slurm

Author: Md Mosharaf Hossain
Email: mhossain44@students.tntech.edu
Computer Science, Tennessee Tech University

Using Slurm:

Slurm is a resource manager and job scheduler, which is designed to allocate resources and to schedule jobs to run on worker nodes in an HPC cluster. However, you will require landing to a login node using your credentials before submitting jobs to the remote cluster. In Unix/Mac, you can use **ssh** command by opening bash shell/terminal. Consider the below example that shows landing in the login node.

ssh user_name@remote_server_address

e.g
ssh mhossain44@shell.hpc.tntech.edu

After you enter the above command, you will be asked to give your password for the login.

In Windows, you need to have Putty installed in your laptop. As you installed Putty you can setup easily to connect in a login node. In our case below is the configuration parameter for putty.

Host Name/Ip address: shell.hpc.tntech.edu
Port: 22
Connection Type: SSH

To transfer file from your laptop(Unix/Mac) to the remote cluster you can use below command.
scp file_location_in_your_pc user_name@remote_address:remote_file_name

e.g
scp /home/mosharaf/Desktop/softwares/myfiles2.txt
mhossain44@shell.hpc.tntech.edu:/home/tntech.edu/mhossain44/help/my_file.txt

In windows, you can transfer file using putty. Open Command prompt and and run below command to transfer a file.
pscp file_location_in_your_pc user_name@remote_address:remote_file_name

e.g
pscp E:\my_file.txt mhossain44@shell.hpc.tntech.edu:/home/tntech.edu/mhossain44/help/my_file.txt

Below is an example of a slurm file (sample_job.sh), that uses multiple CPU cores on a single compute node.

```
1 #!/bin/bash
2 #SBATCH --job-name=parallel_sum
3 #SBATCH --output=job_output.txt
4 #SBATCH --nodes=1
```

```
#SBATCH --ntasks=1
#SBATCH --cpus-per-task=28
#SBATCH --time=01:20:00
#SBATCH --mail-type=ALL
#SBATCH --mail-user=mhossain44@students.tntech.edu
#SBATCH --account=ipdc-2018
cd /home/CAE/mhossain44/ipdc
./parallel_sum
```

An explanation is given below for the above script.

1. The first line identifies shell used in this job. In this example, bash is used.
2. The second line states the name of the job.
3. The third line specifies an output file (in this case, **job_output.txt**), where all the outputs related to compiling and execution are saved.
4. Line four specifies the number of nodes requested for this job.
5. Line five specifies number of tasks.
6. In line six, the **--cpus-per-task** specifies number of cores being requested for a task (In this case, 28 cores).
7. Line seven states the time requirement for completing the job (In this case 1 hour and 20 minutes).
8. In line eight, **--mail-type=ALL** specifies that a mail alert should be sent at start, end and abortion of execution.
9. The line defines the users to whom the email will be sent.
10. The 11th line changes the current directory to the stated directory, where the binary file is located.
11. Finally, the command **./parallel_sum** runs the binary file generated from a program parallel_sum.cpp.

Now, to submit a job in the cluster, **sbatch** command is used. The above example can be submitted as follows.

sbatch sample_job.sh

A job id will be created after a successful submission of a slurm job. To see the status of the job below command is used.

squeue -u $USER

If the user wants to cancel a submitted job, he/she can do that by the following command.

scancel job_id

Slurm References:

1. https://ubccr.freshdesk.com/support/solutions/articles/5000688140-submitting-a-slurm-job-script
2. https://support.ceci-hpc.be/doc/_contents/QuickStart/SubmittingJobs/SlurmTutorial.html
3. https://its.tntech.edu/display/MON/HPC+Sample+Job%3A+Gaussian
4. https://wikis.nyu.edu/display/NYUHPC/Slurm+Tutorial

Using PBS:

To leverage the resources of an HPC cluster a portable batch system (PBS) needs to set up in that cluster. A PBS file (file extension is also pbs) contains all the commands that defines how and what resources a program wants to access.

Here is an example of a sample PBS file named as sample_job.pbs.

```
1 #!/bin/bash
2 #PBS -l nodes=1:ppn=8
3 #PBS -l walltime=01:30:00
4 #PBS -N parallel_sum
5 #PBS -m bea
6 #PBS -M mhossain44@students.tntech.edu
7
8 cd /home/CAE/mhossain44/ipdc
9 ./parallel_sum
```

The explanation of the script is as follows:

1. The first line identifies shell used in this job. In this example, bash is used.
2. The second line specifies number of nodes and number of processors/cores used per node. In the above example only 1 node is requested. The command also requests 8 cores from the requested node using the **ppn** command.
3. Third line specifies how much wall-clock time is being requested. In this example, the requested time is one and half hour.
4. The forth line states the name of the job.
5. The fifth and sixth line are used to set mailing options. The **PBS -m bea** option states that a mail will be sent when the job **b**egins, **e**nds or **a**borted.
6. In the sixth line, the `#PBS -M` defines the users to whom the email will be sent to. In case of multiple users, emails are separated by commas.
7. The 8th line changes the current directory to the stated directory, where the binary file is located.
8. Finally, the command **./parallel_sum** runs the binary file generated from a program parallel_sum.cpp.

The pbs file can be submitted using **qsub** command after entering into a login-node. After a successful submission, an ID is generated for that job. The submitted job is automatically placed in a PBS queue. The **qstat** command is used to check on the status of a submitted job. The most common commands are listed below:

```
qsub sample_job.pbs     # submits sample_job.pbs job in the PBS queue.
qstat -u user_name      #shows status of all jobs submitted by the user (user_name)
qdel job_id             # deletes the submitted job having id as job_id
```

PBS References:

1. https://wikis.nyu.edu/display/NYUHPC/Copy+of+Tutorial+-+Submitting+a+job+using+qsub
2. https://hpcc.usc.edu/support/documentation/running-a-job-on-the-hpcc-cluster-using-pbs/
3. http://www.uni-tuebingen.de/fileadmin/Uni_Tuebingen/Einrichtungen/ZDV/Bilder/Computing/batch_doc-1.pdf