Seawulf

June 11, 2021

Seawulf introduction

The following link provide the introduction to Seawulf and slurm.

`https://it.stonybrook.edu/services/high-performance-computing#faq-section-related-9916`

Slurm is an open source workload manager and job scheduler that is now used for all SeaWulf queues.

On seawulf, you will be given two spaces: home (20 GB) and scratch (20TB). You can run jobs from your home directory, however due to the 20 GB quota, it is recommended that you run jobs from the scratch space, and save the results in your home directory. Files older than 1 month will be purged from the scratch directory, which is not backed up.

Following are some of command that might help you:

To use slurm, load the slurm module with the following command:
>>module load slurm

But to run a job on the cluster, we need to write bash shell script.

To make a bash shell script, please create a file by the following command:
>> emacs filename].sh

Once you have created a file, you can open it with your choice editor:
>> emacs [filename].sh

```
#!/bin/bash
#
#SBATCH --job-name= [job_name]
#SBATCH --output= [output filename]
#SBATCH --ntasks-per-node=1
#SBATCH --nodes=1
#SBATCH --time= [maximum_time_limit]
#SBATCH -p [queue name]
#SBATCH --mail-type=BEGIN,END
#SBATCH --mail-user=[email address]
```

```
module load shared
module load anaconda/3

source activate pytorch

mkdir /dev/shm/[your name]
scp -r /gpfs/scratch/[your name]/WCML /dev/shm/[your name]

cd [working directory address]

python [filename].py

rm -r /dev/shm/fkarankumar
```

#!/bin/bash - specifies the shell under which the script is to run. You can check which shell you are working by command 'echo $SHELL'. This script was written under bash shell. I don't know much about shell, so I don't know if this will work under other shell like tcsh.

#SBATCH –job-name= [job name] - defines the job name (you can put any name)

#SBATCH –output= [output filename] - this file only contains the print statement outputs, not the files.

#SBATCH –ntasks-per-node=1
#SBATCH –nodes = 1 - These above two command specifies, for a single-threaded script, to allocate 1 node and 1 processor-per-node. Leave these two command to 1 for right now as increasing these number will increase the time in queue.

#SBATCH –time= [maximum time limit] - specifies the maximum time your program needed to run. If this is not included, the code will run for default time and default time will depend on which queue you are running on. You can check information about the queue with the following link:
`https://it.stonybrook.edu/help/kb/how-to-use-the-gpu-nodes-on-seawulf`
With this script, the program took around 5 hours for 5 epoch for 75/25 division of training/test. For 75/25 division of data, every epoch takes approximately 1 hour. To be safe, you can put 7 hours maximum. Your jobs will be terminated after it reached the maximum time limit. Asking for more hours will increase your time in queue.

#SBATCH -p [queue name] - indicates to the batch scheduler that you want to use the GPU queue. You can check queue from the above link.

#SBATCH –mail-type=BEGIN,END
#SBATCH –mail-user=[email address] - these two commands notifies the user, when their jobs got out of queue and when it is ended by sending emails.

Modules

Software on the SeaWulf cluster is made available through the module system and Anaconda package manager. You may view all software available through loading module files by entering the following commands:
>>module load shared
To you check the list of modules with the following command:
>>module avail

To load module with the following command:
>> module load [module name]

You may view all software available through the loaded Anaconda environment by entering the following commands:
>> module load anaconda/3
>> conda list
You need to create environment and activate the environment to run the program. I don't have much knowledge about them. Here what needed to do to run this code, pip install matplotlib in local directory or other librabies if you need and copy the above the command about module. Please note that when you submit jobs with Slurm, all of your environment variables will by default be copied into the environment for your job. This includes all of the modules you have loaded on the login node at the time of submitting your job.

First, put the data into your scratch directory. You can copy data from Cris directory.
scp -r /gpfs/scratch/crfernandesv/WCML [your scratch directory address]

mkdir /dev/shm/[your name]
scp -r [directory where data is saved] /dev/shm/[your name]

We are importing data from scratch directory to /dev/shm/[your name] directory. It is easier to load data from /dev/shm/[your directory].

cd [working directory address]
Your working directory is a directory from where you submit your job. Everything that is needed should be placed in working directory. The python script should be in it and any external libraries should be in it (in our case, we use iotools). Place your iotools in your working directory. The output files will goes to working directory. In case your bash script is in other directory and all the other program is in different directory, you can cd the working directory otherwise you can remove that line.

python [filename].py
This is just to run your python script.

rm -r /dev/shm/[your name]
This will delete the data from /dev/shm directory.