tutorial

December 19, 2018

1 Tutorial - The Grabow Group

Authors: Hari Thirumalai, Juan Manuel Arce-Ramos and Karun Kumar Rao

1.1 Introduction

The aim of this document is to help users who are new to the concepts and tools used in the group. By the end of this tutorial, the reader will should have a basic understanding of the various components of typical computational research workflows, starting from basic Linux commands to running calculations using VASP and other softwares. This document was not designed to be a thorough guide and the user is encourage to complement the information provided here with extensive Google searches.

We also recommend you to look at John Kitchin's DFT Book pdf or html versions since it contains many working examples that touch upon practical concepts of computational catalysis which can be easily (or relatively easy) followed and implemented.

1.2 High Performance Computing

A significant portion of the research conducted in the group pertains to computational modeling in which various properties of interest for a system are calculated by means of /ab initio/ density functional theory calculations. We refer the reader to an excellent book that delves on the nuts and bolts of DFT by David Sholl. These calculations are computationally intensive and are exclusively performed on computing clusters or supercomputers. These massively-parallel machines are accessed remotely through the Secure Shell Protocol and the user is able to access his or her account on that machine. Once logged in, the user sets up these jobs and submits them to the system's resource manager, or the queue. Once resources become available, the queue executes the job and the user is notified upon completion of the job.

1.2.1 Queues

The queue is a utility that accepts job submissions from users, implements a fair use policy, and allocates resources based on job requirements and other parameters. Most of the systems used by our group are managed by the SLURM Workload Manager=. Maxwell is managed by the Torque Resource Manager. The configuration keywords and parameters are different for different systems and every job submission script must contain these parameters for it to be accepted by the queue. The queue keywords are

For =SLURM=

```
#SBATCH -p <queue partition>
#SBATCH -o myMPI.o%j
#SBATCH -N <number of nodes> -n <number of processors per node>
#SBATCH -t <walltime in hhh:mm:ss>
#SBATCH --mail-type=END
#SBATCH --mail-user=<user email id>
```

For =Torque=

```
#PBS -e myMPI.e%j
#PBS -o myMPI.o%j
#PBS -m ae
#PBS -M <user email id>
#PBS -l <walltime in hhh:mm:ss>
#PBS -r n
#PBS -l nodes=<number of nodes>:ppn=<number of processors per node>
#PBS -l pmem=<Memory requested per node in mb>
#PBS -S /bin/tcsh <Specify type of Shell>
```

A more detailed explanation of these parameters follows: - queue partition: This specifies the partition to which you want to submit your job. - number of nodes: A node is a group of processors, which are designed to work together with maximum efficiency. A simple example of a node would be a computer with an Intel i5 processor, where the single node has 4 processors. - number of processors: This is the number of processors or threads in a node. Usually, the user is expected to request all processors in a node. This parameter is system configuration dependent. - walltime in hours: This specifies the time until which the job will execute on the system. Once runtime exceeds this value, the job execution is terminated.

1.2.2 Jobscripts

Jobscripts are executable files of a defined environment which consist of executable code. Jobscripts can be in a variety of file formats and the most commonly used ones are python, shell and cshell jobscripts. A jobscript and a simple file are differentiated by the file type identifier. This line tells the compiler, interpreter and any text-editor the type of the file. This removes the need for an extension to the file, which can also serve as an identifier. A properly identified file also enables source code formatting on text-editors.

Example job scripts for SLURM and PBS (toruque) schedulers are given below

```
#!/usr/bin/env python --> File environment identifier

#SBATCH -p batch
#SBATCH -o myMPI.o%j
#SBATCH -N 5 -n 100                         [SLURM Parameters]
#SBATCH -t 168:00:00
#SBATCH --mail-type=END
#SBATCH --mail-user=hthirumalai@gmail.com

# Your executable python code begins here
from ase.io import read
from ase.calculators.vasp import Vasp

...
```

and

```
#!/usr/bin/env python

#PBS -e stderr
#PBS -o stdout
#PBS -m ae
#PBS -M hthirumalai@gmail.com
#PBS -l walltime=100:00:00
#PBS -r n                                        [PBS Parameters]
#PBS -l nodes=1:ppn=12
#PBS -l pmem=2500mb
#PBS -S /bin/tcsh
#PBS -V

from ase import *
from ase.calculators.vasp import Vasp

...

#!/bin/sh --> File environment identifier

#SBATCH -p batch
#SBATCH -o myMPI.o%j
#SBATCH -N 5 -n 100                            [SLURM Parameters]
#SBATCH -t 168:00:00
#SBATCH --mail-type=END
#SBATCH --mail-user=hthirumalai@gmail.com

# Your executable shell script begins here
echo 'VASP starting execution ..'

...
```

and

```
#!/bin/sh

#PBS -e stderr
#PBS -o stdout
#PBS -m ae
#PBS -M mayerzmytm@gmail.com
#PBS -l walltime=100:00:00
#PBS -r n                                        [PBS Parameters]
#PBS -l nodes=1:ppn=12
```

```
#PBS -l pmem=2500mb
#PBS -S /bin/tcsh
#PBS -V

# Your executable shell script begins here
echo 'VASP starting execution ..'
```

1.2.3 System Specific Settings

Our group has access to various clusters at any given time and job scripts must be modified such that they execute without errors when transferred from one cluster to another. This section consists of all cluster relevant information. All storage-intensive jobs must be executed on the group's project directories. These locations are backed-up on a daily basis. $SCRATCH directories on Cori and Stampede2 are short term, high I/O performance storage that are periodically purged. Therefore, the reader is advised to use these directories for running jobs only and transfer these files to permanent storage on the University of Houston clusters.

Opuntia

```
project directory: /project/grabow

#SBATCH -p grabow
#SBATCH -o myMPI.o%j
#SBATCH -N 1 -n 20
#SBATCH -t 24:00:00
#SBATCH --mail-type=END
#SBATCH --mail-user=@gmail.com
```

uHPC

```
project directory: /uhpc/grabow

#SBATCH -p batch
#SBATCH -o myMPI.o%j
#SBATCH -N 1 -n 20
#SBATCH -t 24:00:00
#SBATCH --mail-type=END
#SBATCH --mail-user=@gmail.com
```

Juniper

```
project directory: /project/grabow

#SBATCH -p batch
#SBATCH -o myMPI.o%j
#SBATCH -N 1 -n 24
#SBATCH -t 24:00:00
#SBATCH --mail-type=END
#SBATCH --mail-user=@gmail.com
```

Sabine

```
project directory: /brazos/grabow

#SBATCH -p batch
#SBATCH -o myMPI.o%j
#SBATCH -N 1 -n 24
#SBATCH -t 24:00:00
#SBATCH --mail-type=END
#SBATCH --mail-user=@gmail.com
```

Cori

```
scratch directory: $SCRATCH
project directory: /global/project/projectdirs/m2029/

#SBATCH -p regular
#SBATCH -C knl
#SBATCH -A m2029
#SBATCH -o myMPI.o%j
#SBATCH -N 1 -n 64
#SBATCH -t 24:00:00
#SBATCH --mail-type=END
#SBATCH --mail-user=@gmail.com
```

AND

```
#SBATCH -p regular
#SBATCH -C haswell
#SBATCH -A m2029
#SBATCH -o myMPI.o%j
#SBATCH -N 1 -n 32
#SBATCH -t 24:00:00
#SBATCH --mail-type=END
#SBATCH --mail-user=@gmail.com
```

Stampede2

```
scratch directory: $SCRATCH
project directory: $WORK

#SBATCH -p normal
#SBATCH -o myMPI.o%j
#SBATCH -N 1 -n 64
#SBATCH -t 24:00:00
#SBATCH --mail-type=END
#SBATCH --mail-user=@gmail.com
```

AND