# Quick guide for using SLURM in Cluster

Imad Eddine TIBERMACINE

## 1. Abstract:

SLURM (Simple Linux Utility for Resource Management) is an open source tool for cluster management and job scheduling in linux clusters. In my work (MP and MPI), I used slurm for compiling and running, and in this sheet here is a quick explanation for it.

## 2. SLURM Basics:

Here is some essential instructions we need to use:

**sinfo**: provides information on the state of resources on the cluster, and the important results are the following:

- **Idle**: it means that this noeud is free
- **Alloc**:  it means that this noeud is allocated
- **Down**: noeud under maintenance

```
[atibermacine@ibnbadis0 ~]$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
visu         up   infinite      1  alloc ibnbadis10
r424*        up   infinite      6 drain* ibnbadis[12,16,19,22,37,40]
r424*        up   infinite      7  down* ibnbadis[13,18,24-25,31,35,42]
r424*        up   infinite     17  alloc ibnbadis[11,14-15,17,20-21,23,26-30,32-34,36,41]
r424*        up   infinite      2   idle ibnbadis[38-39]
[atibermacine@ibnbadis0 ~]$
```

**squeue**: allows you to see the jobs in the slurm queue (running or waiting for resources).

**sbatch**: allows you to launch a parallel script on IBN-BADIS. The script will contain all the information regarding the number of nodes to use and the parallel program to run on the scripts.

Here is an example of a script that you can use for your uses on IBN-BADIS:

```
#!/bin/bash

#SBATCH –nodes=3

#SBATCH –ntasks-per-node=8

> machines .txt

for i in $( nodeset -e ${SLURM_NODELIST} ) ;

do

echo -e « ${i} » >> machines .txt

done

mpirun -machinefile machines.txt myprogram
```

--nodes=3 : indicates the number of nodes to use in parallel

– –ntasks-per-node=8 : indicates the number of tasks to launch per node

> machines.txt : to empty the contents of the machines.txt file. The machines.txt file contains the machines on which the parallel program will run.

The for loop is used to fill the machines.txt file by the machines reserved by SLURM.

mpirun: command that executes a parallel program written in mpi

–Machinefile machines.txt: indicates where are the machines on which the program is executed

myprogram: indicates the program that will be executed in parallel.

## 3. Sbatch on MPI:

MPI is an important and efficace mechanism used in HPC, and the following instruction defines the steps of running an MPI script on cluster:

- Write our MPI program and save it in .c file
- Compile MPI program by the next instruction:

  mpicc myprogram.c -o myprogram

- Run the MPI script by the sbatch instruction as follows:

  **sbatch mpijob.sh**

  The script is:

```
[atibermacine@ibnbadis0 ~]$ cat mpijob.sh
#!/bin/bash
#SBATCH --ntasks=10
mpirun ./myprogram
[atibermacine@ibnbadis0 ~]$
```

  --ntasks=10 : defines the number of processors

  - After launching your script, a job is created with an identifier ID (in the previous example ID = 18582). The output stream of your program is

redirected to the slurm- ”ID” .out file that you will find in the current directory (in our example slurm-18582.out).

```
[atibermacine@ibnbadis0 ~]$ sbatch mpijob.sh
Submitted batch job 18582
[atibermacine@ibnbadis0 ~]$
```

```
[atibermacine@ibnbadis0 ~]$ ls
a.out         mpijob.sh   myprogram        slurm-18571.out  slurm-18574.out  slurm-18577.out
job1.sh       mpprog      simplematrix.c   slurm-18572.out  slurm-18575.out  test1
matrixmult.c  mpsimple.c  slurm-18570.out  slurm-18573.out  slurm-18576.out
[atibermacine@ibnbadis0 ~]$
```

- The last step is visiting the file slurm-18582.out the heck our MPI program’s result, and here is the result of simple Hello World program:

```
[atibermacine@ibnbadis0 ~]$ cat slurm-18571.out
Hello from process: 6
Hello from process: 7
Hello from process: 0
Hello from process: 4
Hello from process: 2
Hello from process: 3
Hello from process: 1
Hello from process: 5
[atibermacine@ibnbadis0 ~]$
```

Finally, the previous steps are enough to run any MPI program you want, and personally I used it for matrix multiplication script.

## 4. Sbatch in MP:

To run an MP program on the cluster, we need to follow the next steps:

- Write our MP program and save it in .c file
- Compile the MP program using the next command:

**gcc mpprogram.c -o moprogram -fopenmp**

- Run the MP program using the following SBATCH script:

```
[atibermacine@ibnbadis0 ~]$ cat job1.sh
#!/bin/bash
#SBATCH --nodes=1 --cpus-per-task=1
./mpprog ${SLURM_CPUS_PER_TASK}
[atibermacine@ibnbadis0 ~]$
```

**--cpus-per-task=1 : defines the number of Threads to be used**

```
[atibermacine@ibnbadis0 ~]$ sbatch job1.sh
Submitted batch job 18583
[atibermacine@ibnbadis0 ~]$
```

- Finally the results are in the next file (the results are of matrix multiplication)

```
[atibermacine@ibnbadis0 ~]$ ls
a.out        mpijob.sh   myprogram       slurm-18571.out  slurm-18574.out  slurm-18577.out
job1.sh      mpprog      simplematrix.c  slurm-18572.out  slurm-18575.out  slurm-18583.out
matrixmult.c  mpsimple.c  slurm-18570.out  slurm-18573.out  slurm-18576.out  test1
[atibermacine@ibnbadis0 ~]$ cat slurm-18583.out
TIME = 11.811260

[atibermacine@ibnbadis0 ~]$
```

## 5. Conclusion:

In the first steps, I faced a lot of problems understanding SLURM commands and especially sbatch scripts. Later, after some practice I found out that it's easy for use and manipulating.