"I do not fear computers. I fear lack of them."
- Isaac Asimov

Eigen Letters

Location: Mumbai, India

VOL.I. . . No.4 *JUNE 12, 2023* Written & edited by: Puneet Panwar

Hey friends, I hope you are doing great. This is the fourth edition of the Eigen Letters which will cover the basics of SLURM (Simple Linux Utility for Resource Management). SLURM is a job scheduler and resource manager widely used in High Performance Computing (HPC) environments for managing and scheduling jobs across distributed clusters.

As machine learning models grow in size and complexity, so does the need for computational power. Supercomputing clusters address this need by enabling remote training of neural networks, leveraging High Performance Computing (HPC) for efficient and fast model training. *Cluster computing* refers to the use of interconnected computers, known as nodes, that work together as a unified system to solve complex computational problems or perform high-performance tasks. In a cluster, nodes can be dedicated servers, workstations, or even individual computers connected via a network. Figure 1 presents the architectural structure of a cluster computing system. Users authenticate and gain access to the Head or Frontend node using the Secure Shell (SSH) protocol. Upon gaining access, jobs are delegated to compute nodes managed by a job scheduler or manager.

Let's delve into the SLURM job manager, which serves as an excellent starting point for new users. This article serves as a comprehensive guide to understanding Slurm, its core concepts, and practical examples of using Slurm commands for job submission, monitoring, and resource management. We will explore its features, capabilities, and best practices for efficient job scheduling and resource management.

SLURM Job Scheduler

A key component of cluster computing is a job manager, also known as a job scheduler or resource manager. The job manager is responsible for coordinating and allocating resources within the cluster to execute submitted jobs efficiently. It ensures that jobs are executed on available resources, taking into consideration factors such as job priority, resource availability, and scheduling policies. Popular job managers used in cluster computing include *Slurm*, *PBS Pro*, *Torque*, *LSF*, and *Grid Engine*. These job managers provide a command-line interface, as well as web-based or graphical user interfaces, to interact with the cluster, submit jobs, monitor their progress, and manage resources.

I am using a Supercomputing cluster named 'Pragya' which uses SLURM, which is an open-source job scheduler and resource management system widely used in high-performance computing (HPC) environments. Vocabulary used in this article is as follows:

• **Partition:** A partition, also known as a queue, represents a logical grouping of computing resources. It allows the cluster to be divided based on policies, user groups, or resource types.

• **Job:** A job is a unit of work that is submitted to the cluster for execution. It can be a single task or a parallel job consisting of multiple tasks running on multiple nodes.

• **Node:** A node refers to an individual machine in the cluster that can execute jobs. It typically consists of one or more processors (CPUs) and memory.

• **Head Node / Front-end Node:** The node that users interact with directly. This node is often responsible for distributing tasks among the Compute Nodes.

• **Compute Node:** Nodes that perform the computational work assigned by the Head Node.

• **Job Script:** A job script is a shell script that specifies the commands and parameters required to execute a job. It includes directives for resource allocation, job dependencies, and execution settings.

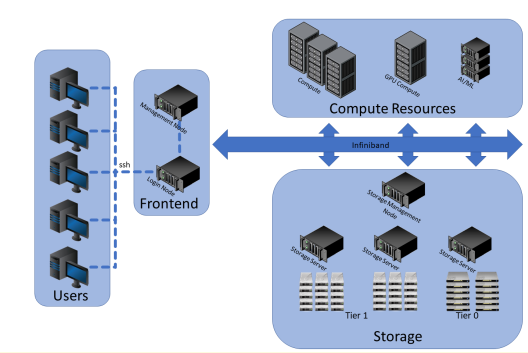

Figure 1: Cluster Computing Architecture

SLURM Commands

• **sinfo -a:** Displays information about nodes and partitions in the cluster.

```
(base) [panwarp@baaz1110 ~]$ sinfo -a
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
pragya*      up   infinite      1  down* pragya02
pragya*      up   infinite      7  drain pragya[06-12]
pragya*      up   infinite      3    mix pragya[01,04-05]
pragya*      up   infinite      1  alloc pragya03
(base) [panwarp@baaz1110 ~]$
```

sinfo -N -o "%N %G" can be used to show the detailed node information.

• **squeue:** Displays the status of jobs in the queue.

```
(base) [panwarp@baaz1110 ~]$ squeue
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
              1730    pragya     test  mayankg PD       0:00      1 (Resources)
              1731    pragya     test  mayankg PD       0:00      1 (Priority)
              1699    pragya      job  ppshete  R 1-01:43:05      1 pragya03
              1717    pragya     test  mayankg  R    1:08:17      2 pragya[04-05]
              1728    pragya     test  mayankg  R    1:22:16      1 pragya01
              1729    pragya     test  mayankg  R    1:22:16      1 pragya01
(base) [panwarp@baaz1110 ~]$
```

squeue -u <userid> is used to list jobs for a particular user. To see the expected start times of your queued jobs: *squeue -u <userid> - -start*.

```
(base) [panwarp@baaz1110 ~]$ squeue -u panwarp
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
              1866    pragya train_gp  panwarp PD       0:00      1 (Priority)
(base) [panwarp@baaz1110 ~]$
```

When checking the status of a job, you may want to repeatedly call the squeue command to check for updates. We can accomplish this by adding the - - *iterate* flag to squence command:

squeue - -user=username - -start - -iterate=n_seconds

```
(base) [panwarp@baaz1110 RNN_MSD]$ squeue --user=panwarp --start --iterate=5_seconds
Tue Jun 06 11:26:14 2023
             JOBID PARTITION     NAME     USER ST          START_TIME  NODES SCHEDNODES           NODELIST(REASON)
              1866    pragya train_gp  panwarp PD 2024-05-31T09:34:48      1 (null)              (Priority)

Tue Jun 06 11:26:19 2023
             JOBID PARTITION     NAME     USER ST          START_TIME  NODES SCHEDNODES           NODELIST(REASON)
              1866    pragya train_gp  panwarp PD 2024-05-31T09:34:48      1 (null)              (Priority)

Tue Jun 06 11:26:24 2023
             JOBID PARTITION     NAME     USER ST          START_TIME  NODES SCHEDNODES           NODELIST(REASON)
              1866    pragya train_gp  panwarp PD 2024-05-31T09:34:48      1 (null)              (Priority)
```

• **sbatch:** A batch job in the context of Slurm refers to a job that is submitted to the cluster for execution without requiring immediate user interaction. It is a way to automate the execution of tasks or programs on a cluster without the need for manual intervention. To submit a batch job in Slurm, you need to create a job script, which is a shell script containing the necessary commands and directives to define the job requirements and actions. Here is an example of a batch job script:

```
#!/bin/bash
#SBATCH --job-name=test_script
#SBATCH -N 1                # nodes requested
#SBATCH -n 1                # tasks requested
#SBATCH -c 4                # cores requested
#SBATCH --mem=10G           # memory requested
#SBATCH -t 3:00:00      # time requested (HH:MM:SS)
#SBATCH --mail-type=begin   # email when job begins
#SBATCH --mail-type=end     # email when job ends
#SBATCH --mail-type=fail    # email if job fails
#SBATCH --mail-user=<user's email id>

# commands to run your job
module purge
python3 test_script.py
echo "Test script run complete."
```

The **#!** at the beginning of a shell script file is called a **shebang** or **hashbang**. It is a special construct used in Unix-like operating systems to specify the interpreter that should be used to execute the script. The purpose of the shebang line is to ensure that the script is run by the appropriate interpreter, regardless of how it is invoked (e.g., from the command line or as part of a larger script). For example, in a shell script, the shebang line #!/bin/bash indicates that the script should be executed using the Bash interpreter.

In SLURM, the hyphen (-) and double hyphen (- -) are used to specify short-form and long-form command-line options, respectively. Here are some commonly used double hyphen (- -) options in SLURM and their corresponding single hyphen (-) equivalents:

- - -nodes (-N): number of nodes requested for a job
- - -cpus-per-task (-c): Specify the number of CPUs per task.
- - -exclusive (-x): Request exclusive node access.
- - -export (-E): Export all environment variables to the job.
- - -job-name (-J): Set the job name.
- - -mem (-M): Specify the memory requirement for the job.
- - -ntasks (-n): Specify the number of tasks.
- - -output (-o): Set the job's standard output file.
- - -partition (-p): Specify the desired partition.
- - -time (-t): Specify the maximum runtime for the job.

The **#SBATCH** directive is a special comment used in SLURM job scripts to provide instructions and settings to the SLURM job scheduler. It is important to note that, #SBATCH directives are specific to SLURM and are not recognized or interpreted by the shell itself. The shell treats them as comments, but the SLURM scheduler recognizes them and uses them to configure the job execution.

```
(base) [panwarp@baaz1110 RNN_MSD]$ sbatch testjobscript.sh
Submitted batch job 1867
```

Note: It is important to strike a balance between providing sufficient resources to complete your job in a reasonable time and efficiently utilizing the cluster's resources.

'module purge' command is used to remove all currently loaded modules from the environment. It is often used at the beginning of a job script or when you want to start with a clean environment. The *'echo'* command is used in shell scripting and the command-line interface to display text or variables as output.

• **scancel:** Cancel a running or pending job.

```
(base) [panwarp@baaz1110 RNN_MSD]$ squeue -u panwarp
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
              1866    pragya train_gp  panwarp PD       0:00      1 (Priority)
              1867    pragya train_gp  panwarp PD       0:00      1 (Priority)
(base) [panwarp@baaz1110 RNN_MSD]$ scancel 1867
(base) [panwarp@baaz1110 RNN_MSD]$ squeue -u panwarp
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
              1866    pragya train_gp  panwarp PD       0:00      1 (Priority)
(base) [panwarp@baaz1110 RNN_MSD]$
```

• **scontrol:** this command provides users extended control of their jobs run through Slurm. This includes actions like suspending a job, holding a job from running, or pulling extensive status information on jobs. To suspend a job that is currently running on the system, we can use *scontrol* with the *suspend* command. This will stop a running job on its current step that can be resumed at a later time.

```
(base) [panwarp@baaz1110 RNN_MSD]$ scontrol suspend 1870
```

To resume a paused job, we use *scontrol* with the *resume* command:

```
(base) [panwarp@baaz1110 RNN_MSD]$ scontrol resume 1870
```

Slurm also provides a utility to hold jobs that are queued in the system. Holding a job will place the job in the lowest priority, effectively holding the job from being run. A job can only be held if its waiting on the system to be run.

```
(base) [panwarp@baaz1110 RNN_MSD]$ scontrol hold 1879
```

We can release the hold job using *release* command:

```
(base) [panwarp@baaz1110 RNN_MSD]$ scontrol release 1879
```

In summary, cluster computing relies on job managers to orchestrate the execution of jobs across a cluster of interconnected computers. Content of this newsletter will provide a good starting point for users to understand the basics of cluster computing, with a specific focus on the utilization of systems with Slurm job scheduler. For more information on SLURM please visit this *link*.

I hope that you liked the content of this edition and would request you to provide your valuable feedback and suggestions to improve future editions. To subscribe to *Eigen Letters*, visit *here*. Thanks for reading.

Best: Puneet Panwar