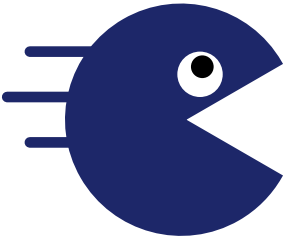


Batch Usage on Dalek – Introduction to SLURM

Sorbonne Université – EI5-SE – Calcul haute performance (EPU-F9-IHP)

Adrien Cassagne

September 29, 2024

Source of Inspiration

Acknowledgment

These document is inspired and adapted from the the excellent slides of the Jülich SuperComputing center (JSC) in Germany.

Please visit:

- https://www.fz-juelich.de/en/ias/jsc/news/events/training-courses/2023/supercomputing-2/07-batch/@@download/file

Table of Contents

Batch System Concepts

- **Resource Manager** is the software responsible for managing the resources of a cluster, usually controlled by a scheduler.
  - It manages resources like tasks, nodes, CPUs, memory, network, etc.
  - It handles the execution of the jobs on the compute nodes.
  - It makes sure that jobs are not overlapping on the resources.
- **Scheduler** is the software that controls user's jobs on a cluster according to policies. It receives and handles jobs from the users and controls the resource manager. It offers many features like:
  - Partitions, queues and QoS to control jobs according to policies/limits.
  - Scheduling mechanisms (backfill, fifo, etc).
  - Interfaces for defining workflows (jobscripts) or job dependencies and commands for managing the jobs (submit, cancel, etc).
- **Batch-System/Workload-Manager** is the combination of a scheduler and a resource manager. It combines all the features of these two parts in an efficient way.

Dalek Batch Model

1 Concepts

- Job **scheduling according to priorities**. The jobs with the highest priorities will be scheduled next.
- **Backfilling scheduling algorithm**. The scheduler checks the queue and may schedule jobs with lower priorities that can fit in the gap created by freeing resources for the next highest priority jobs.
- **No core-sharing**. The smallest allocation for jobs is one core. It is not possible to have two different jobs on a single core (hardware threads). Multiple users can share the same node.
- Each **real person has a unique Unix-user** account that can be member of multiple projects/Unix-groups. For PACC, you will be in the `ei5-se-25` group.
- **CPU-Quota**. In the `pacc_qos` QoS, **max job duration is one hour**, **max CPU resources is 4 hardware threads** and **max 2 nodes** (may change depending on the current lab).

Table of Contents

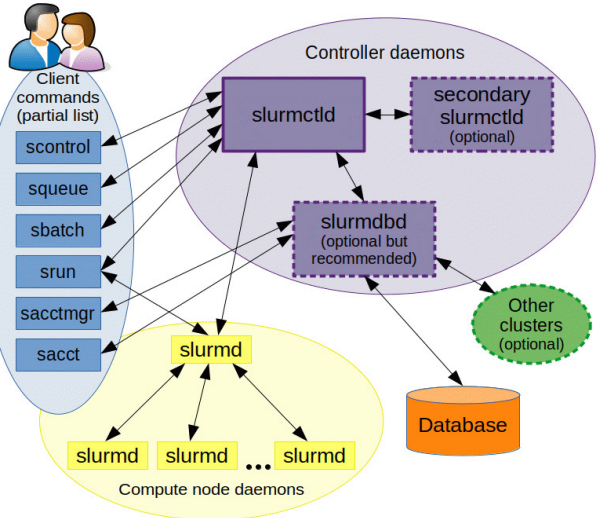
Slurm– Introduction (1)

- **SLURM** (Simple Linux Utility for Resource Management) is the chosen Batch System (Workload Manager) that is used on Dalek. Slurm is an open-source project developed by SchedMD. For Dalek, the standard `slurmd` daemon, is ran on the compute nodes.
- Slurm's configuration on Dalek:
  - High-availability for the main daemons `slurmctld` and `slurmdbd` on the frontend node.
  - Backfilling scheduling algorithm.
  - Accounting mechanism: slurmdbd with MySQL/MariaDB database.
  - User and job limits configured by QoS and Partitions.
  - No preemption configured. Running jobs cannot be preempted.
  - Generic resources (GRES) for different types of resources on the nodes.

Slurm– Introduction (2)

- Slurm groups the compute nodes into **Partitions**. Some limits and policies can be configured for each Partition:
  - allowed users, groups or accounts (for now everything can be accessed)
  - max. nodes and max. wall-time limit per job (for now infinite)
- Other limits are enforced also by the Quality-of-Services (QoS), according to the contingent of user's group, e.g. max. wall-time limit, max number or queued or running jobs per user, etc...
  - In `ei5-se-25-qos`: wall-time limit = 1 hour, max 4 CPUs (= hardware threads) and max 2 nodes for now
- Default limits/settings are used when not given by the users, like: number of nodes, number of tasks per node, wall-time limit, etc.

Slurm– Architecture



Slurm– User Commands (1)

- `salloc`: is used to request interactive jobs/allocations.
- `sbatch`: is used to submit a batch script (which can be a bash, Perl or Python script).
- `scancel`: is used to cancel a pending or running job or job step.
- `scontrol`: provides also some functionality for the users to manage jobs or query and get some information about the system configuration.
- `sinfo`: is used to retrieve information about the partitions, reservations and node states.
- `squeue`: allows to query the list of pending and running jobs.

Slurm– User Commands (2)

- `srun`: is used to initiate job-steps mainly within a job or start an interactive jobs. A job can contain multiple job steps executing sequentially or in parallel on independent or shared nodes within the job's node allocation.
- `sstat`: allows to query status information about a running job.
- `sacct`: is used to retrieve accounting information about jobs and job steps in Slurm's database.
- `sacctmgr`: allows also the users to query some information about their accounts and other accounting information in Slurm's database.

Check the online documentation or the man pages for more detailed info.

Slurm– Job Submission

- There are 2 commands for job allocation: `sbatch` is used for batch jobs and `salloc` is used to allocate resource for interactive jobs. The format of these commands:
- List of the most important submission/allocation options:

```
-c|--cpus-per-task  Number of logical CPUs (hardware threads) per task
-e|--error          Path to the job's standard error
-i|--input          Connect the jobscript's standard input directly to a file
-J|--job-name       Set the name of the job
--mail-user         Define the mail address for notifications
--mail-type         When to send mail notifications. Options: BEGIN,END,FAIL,ALL,..
-N|--nodes          Number of compute nodes used by the job
-n|--ntasks         Number of tasks (MPI processes)
--ntasks-per-node   Number of tasks per compute node
-o|--output         Path to the job's standard output
-p|--partition      Partition to be used from the job
-t|--time           Maximum wall-clock time of the job
```

Slurm– Submission Filter

- Slurm is using a submission filter with the following functionality:
  - Deny jobs requesting multiple partitions, we allow only one.
  - Disable the `--requeue` option. We do not allow users to re-queue their jobs.
  - Deny submission if budget account was not defined (with `--account` or `-A`). By default, is no `-A` specified, the `ei5-se-25` account is selected.

- Examples:
  - Submit a job in the `az4-n4090` partition requesting 2 nodes:

```
sbatch -N 2 -p az4-n4090 <job-script>
```

  - Submit a job in the `az5-a890m` partition requesting 4 nodes:

```
sbatch -N 4 -p az5-a890m <job-script>
```

Slurm– Spawning Command

- With `srun` the users can spawn any kind of application, process or task inside a job allocation. `srun` should be used either:
  - Inside a job script submitted by `sbatch` (starts a job-step).
  - After calling `salloc` (execute programs interactively).
- Command format:
  - `srun [options..] <executable> [args..]`
- `srun` accepts almost all allocation options of `sbatch` and `salloc`. There are however some other unique options:

```
--pty      Execute a task in pseudo terminal mode.
--exact    Allow a step access to only the resources requested for the step.
--overlap  Allow steps to overlap each other on the CPUs.
```

Slurm– Serial Jobscript

- Instead of passing options to `sbatch` from the command-line, it is better to specify these options using the `#SBATCH` directives inside the job scripts which must be positioned in the very beginning of the jobscript!
- Here is a simple example where some system commands are executed inside the jobscript. This job will have the name "TestJob". One compute node will be allocated for 30 minutes.The job will run in the default partition (`dalek`) and on the default account (`ei5-se-25`).

```
#!/bin/bash

#SBATCH --job-name=TestJob      # Name of the job
#SBATCH --output=TestJob_%j.out # Standard output (stdout)
#SBATCH --error=TestJob_%j.err  # Error output (stderr)
#SBATCH --nodes=1               # Number of nodes to reserve
#SBATCH --time=00:30:00         # Maximum duration of the job (hh:mm:ss)
date; hostname; sleep 30; date # commands to execute on the node
```

Slurm– Parallel Jobscript

- Here is a simple example of a jobscript where we allocate 2 compute nodes for 10 minutes. Inside the jobscript with the `srun` command we request to execute on 2 nodes with 4 processes per node the system command `hostname`. In order to start a parallel job, users have to use the `srun` command that will spawn processes on the allocated compute nodes of the job.

```
1 #!/bin/bash
2
3 #SBATCH -J TestJobPar          # Name of the job
4 #SBATCH -N 2                  # Number of nodes to reserve (--nodes)
5 #SBATCH -o TestJobPar-%j.out  # Standard output (--output)
6 #SBATCH -e TestJobPar-%j.out  # Error output (--error)
7 #SBATCH --time=00:10:00       # Maximum duration of the job (hh:mm:ss)
8
9 srun --ntasks-per-node=4 hostname
```

Slurm– Interactive Jobs

- Interactive sessions can be allocated using the `salloc` command. The following command will allocate 1 node for 30 minutes:

```
@login   $ salloc -p iml-ia770 --nodes=1 -t 00:30:00
```

- After a successful allocation, `salloc` will start a new shell on the login node where the submission happened. After the allocation the users can execute `srun` in order to spawn interactively their applications on the compute nodes.

```
@login   $ srun -N 1 --ntasks-per-node=2 -t 00:10:00 hostname
```

- It is possible to obtain a remote shell on the a node, after `salloc`, by running `srun` with the following arguments:

```
@login   $ hostname # should display "front"
@login   $ srun -N 1 -n 1 --interactive -t 00:05:00 -w iml-ia770-1 --pty /bin/bash
@compute $ hostname # should display "iml-ia770-1"
```

Further Information

- Updated status of the systems:
  - Read “Message of the day” on login nodes.
- Check the Dalek online documentation
  - `https://dalek.proj.lip6.fr`
- User support for Dalek
  - dalek-support@listes.lip6.fr
  - Please be polite!

Q&A

Thank you for listening!
Do you have any questions?