Lab02 – Cluster environment

09.10.2024

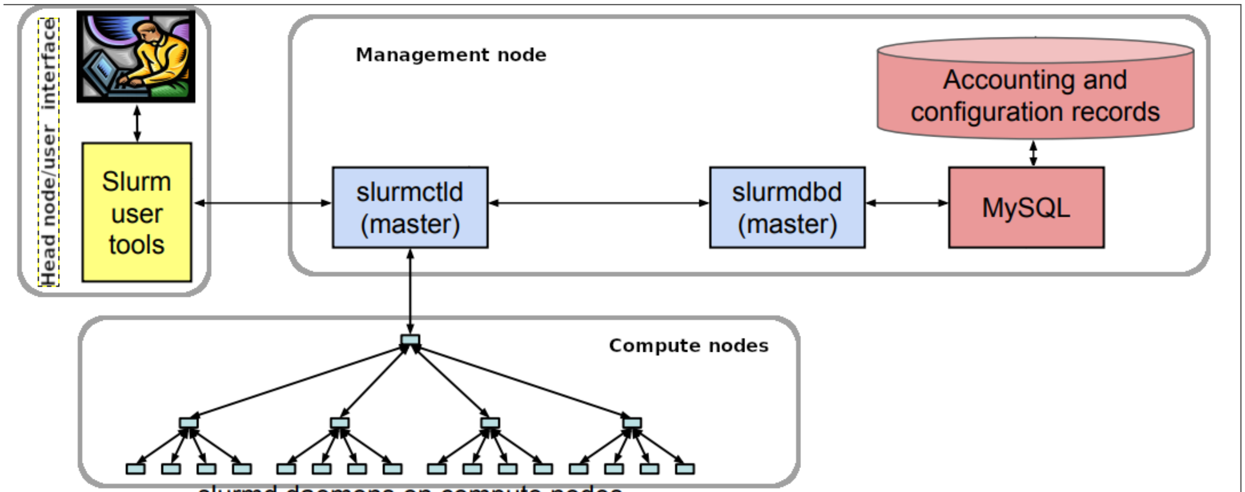
What is a computing cluster?

- Discussion
    - What are the main features of a cluster?
    - Why bother with a dedicated machine? - compared to e.g. cloud?
    - Nondirect access to resources
        - Cluster manager is responsible for executing jobs specified by the user
    - Emphasis on automation and non-interactive work
    - Linux enviroment – proficiency is a big plus!

Slurm resource manager/scheduler

- Was an abbreviation of: Simple Linux Utility for Resource Management
- Slurm is responsible for:
    - Resource management
    - Scheduling
    - Interaction with the cluster
        - Not a policing tool
        - No in-depth knowledge of what is being computed
- Links:
    - Information specific for Ares cluster https://docs.cyfronet.pl/display/~plgpawlik/Ares
    - Information specific for Athena cluster https://docs.cyfronet.pl/display/~plgpawlik/Athena
    - Quick start: https://kdm.cyfronet.pl/portal/Podstawy:SLURM
    - Rules: https://kdm.cyfronet.pl/portal/Zeus:Podstawy#Zasady_obowi.C4.85zuj.C4.85ce_na_klastrze_Zeus (rules apply to all Cyfronet's clusters!)
    - Modules: https://docs.cyfronet.pl/pages/viewpage.action?pageId=17629700 (note that "plgrid" suffix is not used on Ares!)
    - Slurm intro: https://slurm.schedmd.com/quickstart.html

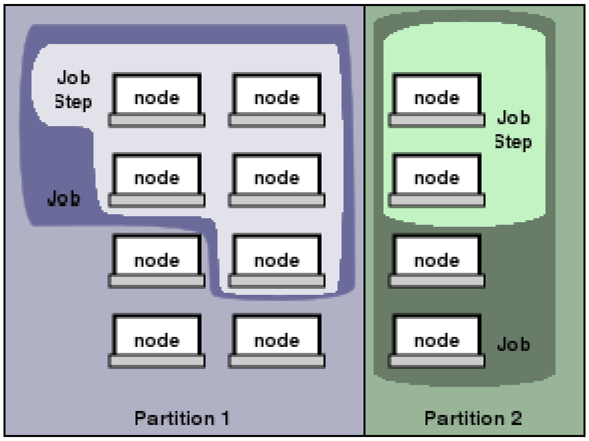
- SLURM architecture:



slurmd daemons on compute nodes
(Note hierarchical communications with configurable fanout)

Copyright 2017 SchedMD LLC
http://www.schedmd.com

- Cluster/job naming convention:



Basic commands

- sinfo – info about nodes
- squeue – queue status
- sbatch – submit a job
- srun – run an interactive job/job step
- scancel – cancel job
- sacct – accounting information
- Examples:

```
$ sinfo -p plgrid
$ squeue
$ sbatch job.sh
$ srun -N 1 --ntasks-per-node=1 -p plgrid-now --pty /bin/bash -l
$ scancel <jobid>
$ sacct -X
```

Resource specification

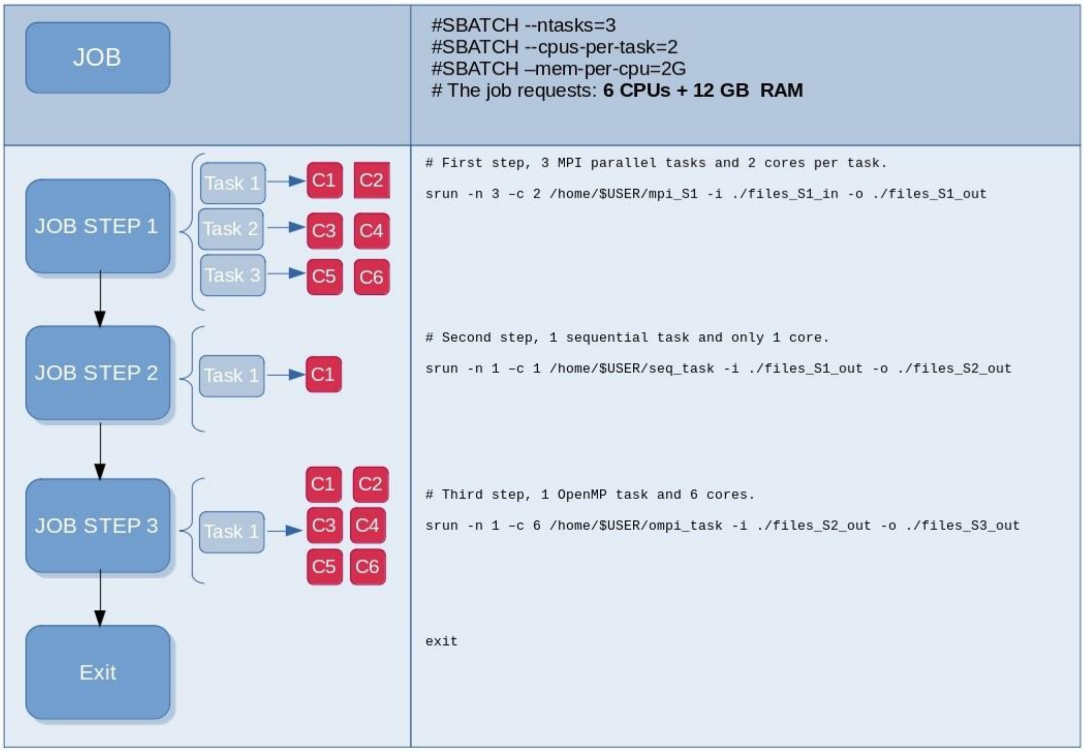
- How to specify resources? With -N, -n, --ntasks-per-node etc. man sbatch
- sbatch -N 2 – Node count
- sbatch -N 1 --ntasks-per-node=6 – Node count with 6 tasks per node = 6 cores
- sbatch -N 2 --ntasks-per-node=6 – Node count with 6 tasks per node = 12 cores on 2 nodes
- sbatch -N 2 --ntasks-per-node=6 -t 20:00 – declared duration of job (HH:MM:SS), defaults to 15 minutes
- **sbatch -A plglscclass24-cpu – declare using the plglscclass24-cpu grant for your computations, this is important if you have other computing grants and want to use a specific one!**
- The account name plglscclass24-cpu can be determined by inspecting the `hpc-grants` output, and looking for “allocation” name. Please note that there can be multiple grants, use the most recent one!
- Be cautious about available node configurations, you are responsible for “fitting in”
- In most cases jobs are specified with “job scripts”
- Example job script (*.sh file), parameters can be included in the job script or be supplied from the command line as sbatch arguments:

```
#!/bin/bash
#SBATCH --nodes=1
#SBATCH --time=00:05:00

#initialization
module load package/version

# data handling and work
cd /path/to/working/directory
binary [arguments]

# end of the script
```



```
#SBATCH --ntasks=3
#SBATCH --cpus-per-task=2
#SBATCH –mem-per-cpu=2G
# The job requests: 6 CPUs + 12 GB  RAM

# First step, 3 MPI parallel tasks and 2 cores per task.
srun -n 3 –c 2 /home/$USER/mpi_S1 -i ./files_S1_in -o ./files_S1_out

# Second step, 1 sequential task and only 1 core.
srun -n 1 –c 1 /home/$USER/seq_task -i ./files_S1_out -o ./files_S2_out

# Third step, 1 OpenMP task and 6 cores.
srun -n 1 –c 6 /home/$USER/ompi_task -i ./files_S2_out -o ./files_S3_out

exit
```

Figure 1: Example SLURM job structure and script. (Source: https://garnatxadoc.uv.es/slurm/slurm_info.html)

Interactive work

- Typical use cases?

```
$ srun -p plgrid-now --time 1:00:00 --pty /bin/bash -l
```

- Hint: using the plgrid-now partition usualy leads to very short queue time with the cost of imposing limits on the job like shorter runtime

Environment

- SLURM communicates a lot of information through environment variables
- $SLURM_SUBMIT_HOST - host, what is the host from which the job originated?
- $SLURM_SUBMIT_DIR – directory, from where the job was submitted
- Other variables: man sbatch, or check with ‘env’ command in an interactive job

Array/parametric jobs

- We can create multiple instances of the same job by using the –array parameter.

```
$ sbatch -N 1 --ntasks-per-node=1 -p plgrid --array=1-4 test.sh
```

- This will create an array job composed of 4 jobs. This can be verified by using the squeue command. It’s possible to “know” which job is which by reading the SLURM_ARRAY_TASK_ID environment variable:

```
$ echo $SLURM_ARRAY_TASK_ID
```

Modules

- Managing available software is done through the system called “modules”.
- The software includes: scientific software, libraries (e.g. sqlite, cuda), core utilities (compiles, math libraries)
- The “module” command is used to work with the system, where subcommands include:
  - avail – see the list of availble modules
  - **spider – search for a module and provide information about modules found**
  - load – to load a module
  - list – to list loaded modules
  - purge – clean your environment from any laoded modules
  - help – consult the manual
- Some examples:

```
$ module avail
( ..This will list all modules.. )
$ module load plumed
( ..This will load the plumed module.. )
$ module purge
( ..This will list the unloaded modules.. )
```

- What does the module command actually do? It doesn’t install the software, it adjusts the environment, so the application becomes available as it would be installed in a usual way. You can think of a module as a preset of PATH, LD_LIBRARY_PATH and other important environment variables.
- Load modules in the job script!

Tasks

1. Answer the questions:
   a. Explain the purpose of the $SCRATCH filesystem accoriding to the documentation. What filesystem is used for $SCRATCH and can you explain in a simple way why ext4/xfs/nfs cannot be used here?

   b. What is the main feature (and benefit) of an RDMA network transfer? Why is it important for highly parallel jobs? What network types offer RDMA?

2. Create a “hello world” batch job, which will:
   a. Get information about CPU (hint: use lscpu or similar command)
   b. Report how many cores are available for the job (there are multiple ways to do this, think of a convenient way of checking how many CPUs should be used)
   c. Please remember to specify the account (–A) parameter for jobs, which should be set to the allocation name reported by “hpc-grants” command. In most cases it will be something like that:
      –A plglscclass24-cpu
3. Use an array job to render a an animation from the blender demo-files. Here are some tips:
   a. Warning! Rendering on a cluster is pretty fast, but queue times are, in some cases, unpredictable. Please account for queue times from minutes to, in some cases, hours!
   b. You can use the sample animation from blender demo page: https://mirrors.dotsrc.org/blender/demo/geometry-nodes/repeat_zone_flower_by_MiRA.blend (frame range for this animation is from 1 to 100)
      demo site: https://www.blender.org/download/demo-files/ (Repeat Zone – Flower is linked above). Unfortunately, not all demo scenes work in an environment without display :(
   c. Please use the “plgrid” partition to submit jobs.
   d. Assuming you chose to use the linked demo scene: Using a batch job configuration of 1 node with 4 CPUs per task seems to be a good choice, as rendering one frame may take up to 20 minutes. Requesting more CPUs will shorten render time, but queue times might increase. You can declare that each job will use up to 1GB of memory instead of the default 4GB per CPU – this might help with queue times.
      i. (hint) In a real-world scenario, choosing job configuration, accounting for application performance, and queue times is one of the main challenges of using a cluster-real world scenario, choosing job configuration, accounting for application performance and queue times is one of the main challenges of using a cluster.
   e. Blender is available through the modules system described above.
   f. Each job should render one frame, this can be achieved with blender this way: https://docs.blender.org/manual/en/latest/advanced/command_line/render.html
      i. (hint) Ensure that the ‘-f’ parameter is at the end of a command line! Blender tends to ignore it otherwise.
   g. Please verify if the images were rendered and if the animation looks OK.
   h. Please provide a part of hpc-jobs-history command output with information about your jobs. What efficiency was achieved?
   i. Answer the question: Can you estimate how many CPU-hours were used for the whole animation?
      i. (hint) This can be estimated on job parameters and/or read from the hpc-jobs-history ourput.
   j. Answer the question: How many CPUs/threads blender uses?

i. (hint) The answer doesn’t have to include a number.