CNAG Basics
Cluster introduction

Rubén Ramos, HPC Support
cnag_support@bsc.es

Barcelona, March 16th
**Aim**

- Understand CNAG's cluster design
  - HPC Cluster: Big and complex machine unlike your desktop

- Not all filesystems are born equal
  - Parts of the filesystem have specific purposes, strengths and weaknesses

- Learn how to use the cluster adequately
  - Best practices and rules of thumb

- Learn how software is made available
  - Module Environment, /apps

- Who to contact?
  - [www.bsc.es/user-support/cnag.php](http://www.bsc.es/user-support/cnag.php) / cnag_support@bsc.es
Cluster Overview
The cluster consists of:
- 2 login nodes
- 2 data transfer nodes
- 108 GenC nodes (24 cores, 256 GB RAM)
- 12 GPU nodes (16 cores, 128 GB RAM)
- 20 GenB nodes (16 cores, 128 GB RAM)
- 1 SMP node (48 cores, 1TB RAM)

Data is stored in:
- 1 NFS exported /home on all nodes
- 1 NFS exported /apps on all nodes
- 2 High Performance Distributed Filesystems (Lustre)

Jobs are executed with:
- Slurm Resource Manager (Batch system)
Login usage
Connecting

2 logins:
- login1: 172.16.10.20
- login2: 172.16.10.21

Use SSH:
- Windows: PuTTY
- Linux / Mac: Openssh
  
  ssh -l user 172.16.10.20
- SSH Keys

You have internet access from the logins, but cannot access logins from the internet (without VPN)
- Ask for a VPN at rrhh@crg.eu
Intended usage

**YES**
- Edit scripts
- Read results
- Process text
- Arrange data on filesystems
- Launch jobs
- Download/upload data (careful)

**NO**
- Test executions
- Perform big statistical computation

Otherwise, **interactive jobs**
Data Transfer

- **scp**
  - Slow, limited

- **lftp**
  - Faster
  - Parallel downloads

- **Aspera**
  - Fastest
  - Limited to certain origins

Transferences are limited and shared with all users. E.g.: limit your bandwith and use data transfer nodes.
Environment
Software

**Cluster wide installations:**
- System software: utils, default compiler, system libs
- Scientific/specific software: /apps

**Several versions of scientific software**
- Collisions
- Dependencies

**Managed by:**
- Environment Modules

**Default module (latest)**

**Modules optimized per CPU arch**
- Load modules inside job!
Environment modules

Used with command
module

Examples
module load python/2.7.3
module purge
module load python/2.7.6
module unload intel
module list
module avail
Lustre Filesystem
What is Lustre

A High Performance Parallel Distributed Filesystem

- High Performance → Fast access
- Parallel → Many different processes in different computers can access the same file
- Distributed → Many different servers holding different parts of the same file, working together

Meaning: Formula 1, not a 4WD
Lustre recommended usage

- No more than 1000 files per directory (directories count as files). Otherwise, access performance will be affected.

- Avoid many small files (< 4 MB)

- Do not create and delete lots of small files during execution (use $TMPDIR)

- Do not use find → use lfs find

- Do not 'rm -r *' → use 'lrm <path>' (module load bsc)

- Specify $TMPDIR whenever possible (java -io.tmpdir=; sort -T; vcftools --temp; ...)

15
Lustre Commands

Finding files
  lfs find /scratch/devel/… -name ".fastq.gz"

Listing files
  lfs ls /scratch/devel/…/*.gz

Deleting files
  rm -f /path/to/file
  rm -rf /scratch/devel/.../*/*.* #NEVER!

Seeing your quota

lquota

$ lquota

Disk quotas for user user_support (uid 40171):

<table>
<thead>
<tr>
<th>Filesystem</th>
<th>used</th>
<th>quota</th>
<th>limit</th>
<th>grace</th>
<th>files</th>
<th>quota</th>
<th>limit</th>
<th>grace</th>
</tr>
</thead>
<tbody>
<tr>
<td>/project</td>
<td>10.1T*</td>
<td>10T</td>
<td>12T</td>
<td>6d5h10m</td>
<td>7509</td>
<td>207508</td>
<td>407508</td>
<td></td>
</tr>
<tr>
<td>/scratch</td>
<td>1.254G</td>
<td>20T</td>
<td>24T</td>
<td>-</td>
<td>26</td>
<td>200026</td>
<td>400026</td>
<td></td>
</tr>
</tbody>
</table>
### Access times per storage type

**Applicable to CNAG cluster**

<table>
<thead>
<tr>
<th>Access</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lustre /scratch</td>
<td>1 u</td>
</tr>
<tr>
<td>/scratch_tmp</td>
<td>4 u</td>
</tr>
<tr>
<td>Lustre /project</td>
<td>~5 u</td>
</tr>
<tr>
<td>NFS (/home)</td>
<td>192 u</td>
</tr>
</tbody>
</table>

*Numbers approximated/estimated*
Batch System (Slurm)
What is Slurm

### Batch Scheduler
- Manages resources (compute nodes)
- Defines set of resources associated to execution
- Enforce limits
- Schedule execution once resources are available

### Cluster divided in partitions
- Main → Default partition
- GenB → GenB nodes
- Gpu → GPU nodes
- Interactive → interactive jobs
- SMP → SMP node
Slurm Concepts

Partition

- Group of resources. Usually a common set of limits or capabilities.

QOS

- An indication on the priority for a job. Usually has a time or amount limit.

Job

- A script with a defined execution purpose that requests a determined amount of resources for a determined amount of time in a specific partition and queue.

JobArray

- A set of jobs sharing logic but differing in some parameter or input.
CNAG's partitions
Queue tools

- List jobs in queue
  - `mnq / squeue`

- Submit job
  - `Mnsubmit / sbatch`
  - `MPI → srun`

- Cancel job
  - `mncancel / scancel`

- Interactive node
  - `mnsh`

- See older jobs
  - `sacct`
Jobscript examples

- **Sequential job**

```bash
#!/bin/bash
#SBATCH --job-name=test_serial
#SBATCH -D .
#SBATCH --output=serial_%j.out
#SBATCH --error=serial_%j.err
#SBATCH --ntasks=1
#SBATCH --time=00:02:00

module purge
module load python
module load perl

./serial_binary > serial.out
```
Extracting past execution data

**User jobs’ details**
```
sacct -u <user> --long
```

**Show memory stats**
```
sacct -u <user> -o "jobid,maxrss,alloccpus,elapsed,state"
```

**Show jobs in certain timeframe**
```
sacct -S2016-03-16 -E2016-03-20
```

**Show jobs run in certain hardware**
```
sacct -N cnc1
```

**Show jobs that timed out**
```
sacct -u <user> --state TIMEOUT -S2016-04-04 -X
```
Data Management Best practices
Data management tips

Avoid intermediate steps
- Use bash pipelines
  - ./do_stuff | ./manage_stuff_1 | ... | ./manage_stuff_N > out

Avoid replicating data

Compress as much data as possible
- Standard compression
  - tar -cvf name.tar.gz <path_to_file>
- Parallel compression → ++faster (Not in login nodes!)
  - $ module load pigz
  - $ tar -I pigz -cf name.tar.gz folder/

No need to untar every time
- zcat, zless, zgrep, zdiff
Rules of thumb for getting in contact with us
Need some fresh software? Be specific

Need a new package?

- Specify link
  - Python
    - https://pypi.python.org/pypi
    - https://www.bioconductor.org/
  - Perl
    - https://cran.r-project.org/

- Specify Python version ($module av)

Need a new bleeding edge software that some random bioinformatician locked at his own house created?

- Specify the link
Panic, something is not working

Did you Google it? Kidding... But no kidding
Panic, something is not working

Did you check StackOverflow?
Panic, something is not working with my job

Specify
- Job id
- Working dir
- The code you are trying to execute
  - Exact steps
  - Modules loaded
  - Any information about the use case is helpful
Panic, something is not working in the cluster

Unexpected cluster behaviour?

- Where? /scratch, /home, /project
- When? As specific as possible with the timing
  - Working late at night running to meet the paper deadline?
  - Morning bustle? Maybe somebody is being naughty
Little Hands-On
Why do it need to load modules?

- HPC environment different from your local computer
- Different versions of the same program
- Currently, around 352 different tools, apps, compilers... And growing
What modules can I use?

**module av python**

```
[rramos@login1 ~]$ module av python
```

```
--------------------- /apps/modules/modulefiles/tools ---------------------
PYTHON/2.7.11 (default) PYTHON/2.7.3 PYTHON/3.4.3 PYTHON/3.6.0
PYTHON/2.7.14 PYTHON/2.7.6 PYTHON/3.5.1 PYTHON/3.6.3
```

```
[rramos@login1 ~]$ module av atac-dnase-pipeline
[rramos@login1 ~]$ module av ATAC-DNASE-PIPELINE
[rramos@login1 ~]$ module av ATAC

--------------------- /apps/modules/modulefiles/applications ---------------------
ATAC_DNASE_PIPELINE/2016.11.14
```

**Still can’t find the software? Maybe you didn’t get the spelling right. Take a ride into /apps ($ ls -la /apps/ ) :D**
What modules can I use?

```
module av python
```

```
[rramos@login1 ~]$ module av python

------------------------------------------------------------------------ /apps/modules/modulefiles/tools
PYTHON/2.7.11 (default) PYTHON/2.7.3   PYTHON/3.4.3   PYTHON/3.6.0
PYTHON/2.7.14           PYTHON/2.7.6   PYTHON/3.5.1   PYTHON/3.6.3

[rramos@login1 ~]$ module av atac-dnase-pipeline

[rramos@login1 ~]$ module av ATAC-DNASE-PIPELINE

[rramos@login1 ~]$ module av ATAC

------------------------------------------------------------------------ /apps/modules/modulefiles/applications
ATAC_DNASE_PIPELINE/2016.11.14
```

Still can’t find the software? Maybe you didn’t get the spelling right. Take a ride into /apps (`$ ls -la /apps/`) :D
What is my jobs status

$squeue my_job.cmd

```
[rramos@login1 pruebas]$ sbatch parallel_job.cmd
Submitted batch job 13004652
[rramos@login1 pruebas]$ squeue -u $USER

<table>
<thead>
<tr>
<th>JOBID</th>
<th>PARTITION</th>
<th>NAME</th>
<th>USER</th>
<th>ST</th>
<th>TIME</th>
<th>NODES</th>
<th>NODELIST(REASON)</th>
</tr>
</thead>
<tbody>
<tr>
<td>13004652</td>
<td>main</td>
<td>parallel</td>
<td>rramos</td>
<td>PD</td>
<td>0:00</td>
<td>2</td>
<td>(Priority)</td>
</tr>
</tbody>
</table>
```

What is this QOSMaxWallDurationPerJobLimit?

```
[rramos@login1 pruebas]$ squeue -u $USER

<table>
<thead>
<tr>
<th>JOBID</th>
<th>PARTITION</th>
<th>NAME</th>
<th>USER</th>
<th>ST</th>
<th>TIME</th>
<th>NODES</th>
<th>NODELIST(REASON)</th>
</tr>
</thead>
<tbody>
<tr>
<td>13004653</td>
<td>main</td>
<td>parallel</td>
<td>rramos</td>
<td>PD</td>
<td>0:00</td>
<td>2</td>
<td>(QOSMaxWallDurationPerJobLimit)</td>
</tr>
</tbody>
</table>
```

Always check the QoS you are using.

- $ sacctmgr show qos
- Check your user available QoS
  - sacctmgr show assoc format=cluster,user,qos | grep $USER
- Default normal
Let’s launch some jobs together

1) `cp /tmp/basic_user_t_1.cmd $HOME`
   (1) `> sbatch $HOME/basic_user_t_1.cmd`
   (2) `> squeue -u $USER`
   (3) `> cat $HOME/test_%j.out`

2) `cp /tmp/basic_user_t_2.cmd $HOME`
3) `cp /tmp/basic_user_t_3.cmd $HOME`
Thank you!

For further information please contact
cnag_support@bsc.es
http://www.bsc.es/user-support/cnag.php