Introduction to the iPOP-UP HPC cluster

Alix Silvert Magali Hennion

November 2022

Table of Contents

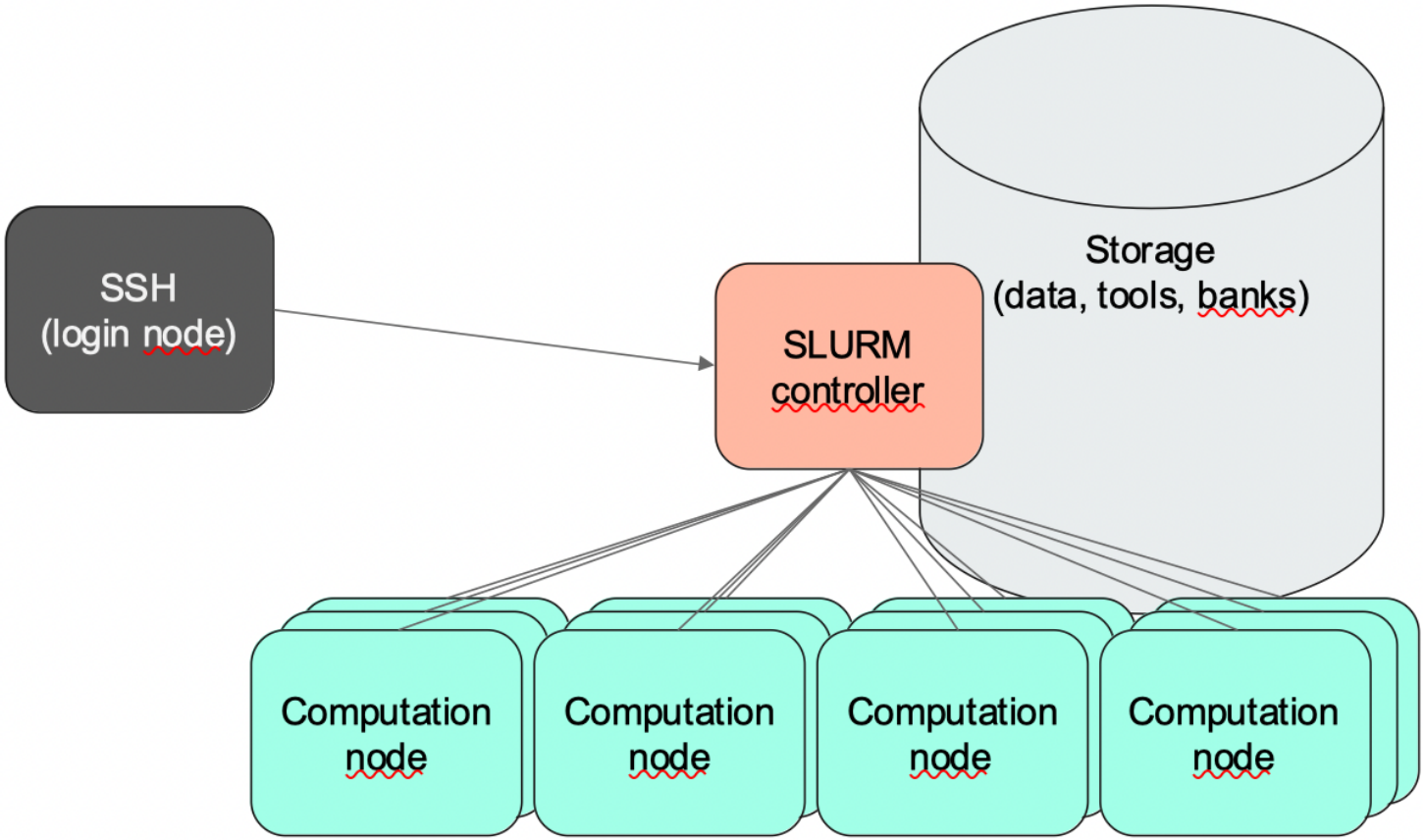
Who is this training for

- You are familiar with Bash
- You need (or might need) more computational power than you currently have
- You already have an account on the cluster
- You know how to use `vi` or `nano`

What is a cluster for ?

- High hardware resources needs
- Long running analyses
- A lot of similar analyses
- Shared work between users
- Free your desktop from the task

What is a cluster ?



Adapted from a slide by Julien Seiler

Computational hardware - iPOP-UP partition

One computational node has

- 128 CPUs
- 256 GB of RAM

And the iPOP-UP partition has sixteen of those

In practice

Go to your terminal and connect to the cluster using the following line, don't forget to replace username with your personal username.

```
ssh username@ipop-up.rpbs.univ-paris-diderot.fr
```

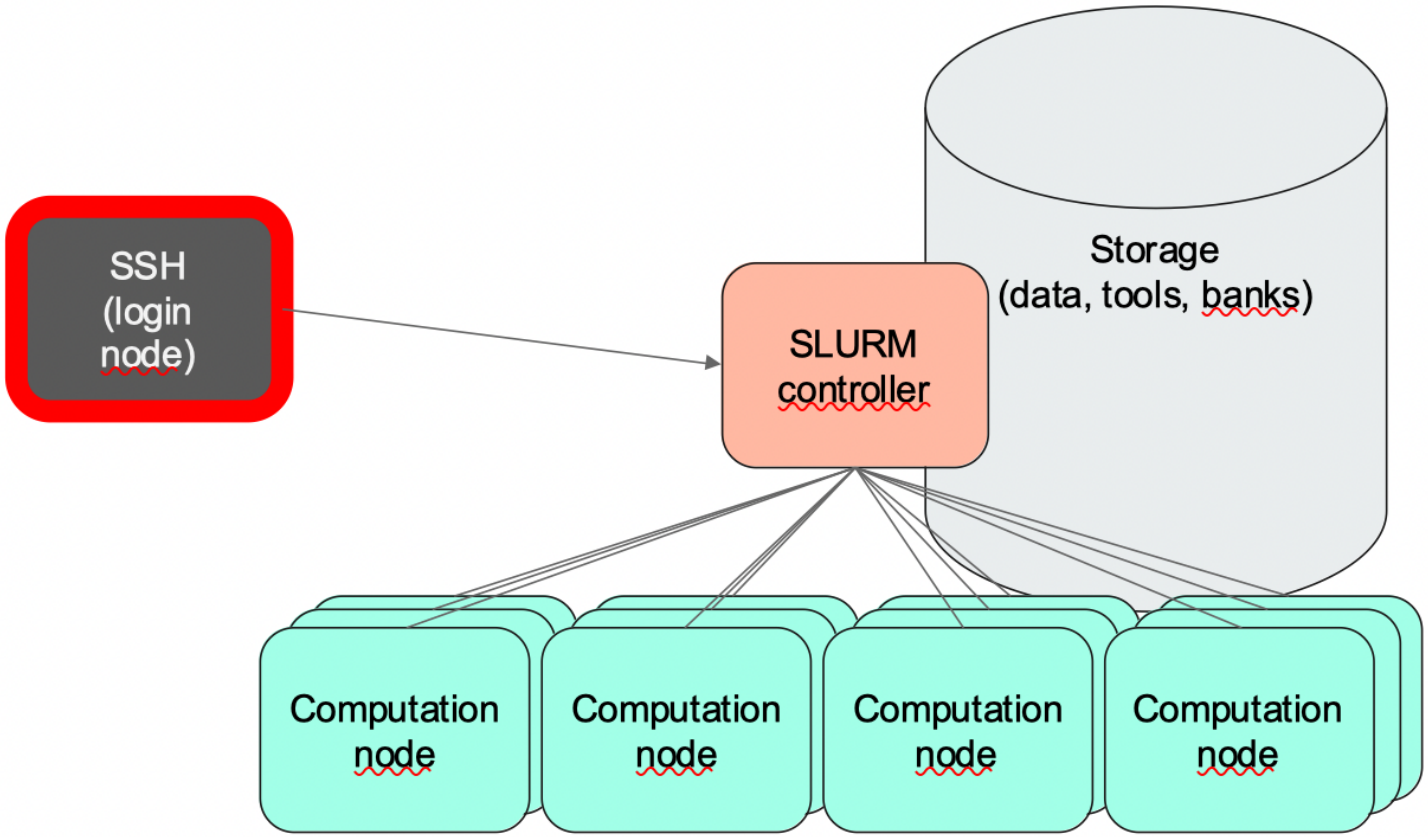
Type in your password and enter.

```
You@YourComputer:~/PathTo/RNAseqProject$ ssh username@ipop-up.rpbs.univ-paris-diderot.fr
################################################################
#          _ _____   ____  _____        _    _ _____          #
#         (_)  __ \ / __ \|  __ \      | |  | |  __ \         #
#          _| |__) | |  | | |__) |_____| |  | | |__) |        #
#         | |  ___/| |  | |  ___/______| |  | |  ___/         #
#         | | |    | |__| | |          | |__| | |             #
#         |_|_|     \____/|_|           \____/|_|             #
#                                                              #
#      Hosted by:                                              #
#      Ressource Parisienne en Bioinformatique Structurale     #
#              ---------------------------                     #
#          All connections are monitored and recorded.         #
#   Disconnect IMMEDIATELY if you are not an authorized user!  #
#                                                              #
################################################################
username@ipop-up.rpbs.univ-paris-diderot.fr's password:
Last login: Tue Jan 25 15:21:45 2022 from 172.28.18.162
Bienvenue sur le cluster iPOP-UP.

Pour toute question ou demande de support, rejoignez-nous sur le forum de RPBS : https://discourse.rpbs.univ-paris-diderot.fr

Pour changer le compte projet par défaut : sacctmgr update user $USER set defaultaccount=<project-name>
project1 [--------------------]       3 /    1024 GB
project2 [#-------------------]     913 /   10240 GB
Update: 2022-01-21 16:00 - default account in bold - More info: status_bars --help
[username@ipop-up ~]$
```

You are here



Adapted from a slide by Julien Seiler

A good thing to do

Change your password

```
passwd
```

Where you can go, write, or execute

Your home

```
cd ~
cd /shared/home/username
```

Your projects

```
cd /shared/projects/projectName
```

The data banks

```
cd /shared/banks/
```

About the data banks

```
[hennion @ ipop-up 12:45]$ ~ : tree -L 2 /shared/banks/
/shared/banks/
...
├── homo_sapiens
│   └── hg38
│       ├── fasta
│       ├── gtf
│       ├── hisat2
│       ├── star-2.7.5a
│       └── transcriptome
...
├── mus_musculus
│   ├── mm10
│   │   └── BS
│   └── mm39
│       ├── fasta
│       ├── gtf
│       ├── hisat2
│       └── repeat_masker
...
```

To ask for some resources to be added, please contact bibs@parisepigenetics.com or ask directly on https://discourse.rpbs.univ-paris-diderot.fr/

Getting your data on the cluster

- `scp`
- `rsync`
- FileZilla
- File Manager
- `git` (for your scripts)
- and others ...

Slurm

Slurm is the cluster management and job scheduling system.

It is what will take your code and distribute it on one of the computing nodes, while ensuring it has the CPU(s) and RAM that you asked for.

And it requires specific commands to run.

Hands-on example

`sinfo`

sbatch

`sbatch` allows you to send an executable file to be ran on a computation node.

Exercise : create the document flatter.sh (using `vi` or `nano`) and type the following

```
#!/bin/bash

#SBATCH --partition=ipop-up

echo "What a nice training !"
```

and run

```
sbatch flatter.sh
```

sbatch

```
(base) [silvert@ipop-up training]$ ls
(base) [silvert@ipop-up training]$ vi flatter.sh
(base) [silvert@ipop-up training]$ sbatch flatter.sh
Submitted batch job 202186
(base) [silvert@ipop-up training]$ ls
flatter.sh  slurm-202186.out
(base) [silvert@ipop-up training]$ cat slurm-202186.out
What a nice training !
(base) [silvert@ipop-up training]$
```

The output that should have appeared on your screen has been diverted to slurm-xxxxx.out

but this name can be changed using SBATCH options.

SBATCH options

Modify flatter.sh to add this line, then run it

```
#!/bin/bash

#SBATCH --partition=ipop-up
#SBATCH -o flatter.out

echo "What a nice training !"
```

Anything different ?

Exercise

Exercise

Run using sbatch the command `hostname` in a way that the sbatch outfile is called hostname.out.

Results

What is the output ? How does it differ from typing directly `hostname` in the terminal and why ?

Useful options 1/2

Options	Flag	Function
−−partition	-p	Partition to run the job (mandatory)
−−job-name	-J	Give a job a name
−−output	-o	output file name
−−error	-e	error file name
−−chdir	-D	Sets the working directory before the script is run
−−time	-t	limit on the total run time (default : no limit)
−−mem		Asks memory that your job will have access to (per node)

To find out more, the Slurm manual `man sbatch` or https://slurm.schedmd.com/sbatch.html

Modules

A lot of tools are installed on the cluster.

To list them

```
module available
module av
```

For example

Look for the different versions of multiqc on the cluster using

```
module av multiqc
```

```
[hennion @ ipop-up 16:48]$ ~ : module av multiqc

-------------------------- /shared/software/modulefiles --------------------------
multiqc/1.11 multiqc/1.12 multiqc/1.3  multiqc/1.6  multiqc/1.7  multiqc/1.9
```

Modules

To load a tool

```
module load tool/1.3
module load tool1 tool2 tool3
```

To list modules loaded

```
module list
```

To remove all loaded modules

```
module purge
```

Load your modules within your "sbatch" file for consistency

Long jobs

sleep

The `sleep` command asks the terminal to stop for the set number of seconds.

Exercise

Start a simple job that will launch `sleep 600`.

Job monitoring - squeue

On your terminal, type `squeue`

```
(base) [silvert@ipop-up ~]$ squeue
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
            202187     cmpli unb02_va domingue  R 2-20:14:02      1 gpu-node2
            202199     cmpli unb05_va domingue  R 2-03:12:09      1 gpu-node14
            202194     cmpli unb04_va domingue  R 2-03:45:25      1 gpu-node14
            202190     cmpli unb01_va domingue  R 2-04:08:32      1 gpu-node14
            202189     cmpli rank02_v domingue  R 2-04:09:18      1 gpu-node4
            202188     cmpli rank02_v domingue  R 2-04:09:39      1 gpu-node4
            202511     cmpli rank05_v domingue  R   20:59:22      1 gpu-node8
            202509     cmpli rank03_v domingue  R   21:09:23      1 gpu-node8
            202508     cmpli rank03_v domingue  R   21:09:26      1 gpu-node7
            202507     cmpli rank04_v domingue  R   21:10:26      1 gpu-node7
            199537      rpbs PP1domLR domingue  R 6-23:11:03      1 gpu-node6
            198465      rpbs Converge domingue  R 7-19:39:50      1 gpu-node6
            198464      rpbs Converge domingue  R 7-19:41:10      1 gpu-node5
            198457      rpbs rank01_v domingue  R 7-19:44:53      1 gpu-node13
            198456      rpbs rank01_v domingue  R 7-19:45:41      1 gpu-node5
            198452      rpbs unb01_va domingue  R 7-19:51:37      1 gpu-node13
```

ST : Status of the job. R means Running, PD means Pending

To see only iPOP-UP jobs
`squeue -p ipop-up`

To see only your jobs
`squeue -u username`

scancel

To cancel a job which you started, use the `scancel` command followed by the jobID (Number given by SLURM, visible in squeue)

```
scancel jobID
```

Monitoring your jobs, sacct

(Re-run sleep if needed and) type `sacct`

```
(base) [silvert@ipop-up ~]$ sacct
       JobID    JobName  Partition    Account  AllocCPUS      State ExitCode
------------ ---------- ---------- ---------- ---------- ---------- --------
202695          sleep.sh    ipop-up      cotech          1    RUNNING      0:0
202695.batch        batch                cotech          1    RUNNING      0:0
```

You can pass the option `--format` to list the information that you want to display, including memory usage, time of running, ... For instance:
`sacct --format=JobID,JobName,Start,Elapsed,CPUTime, NCPUS,NodeList,MaxRSS,ReqMeM,State`

To see every options, run `sacct --helpformat`

Job efficiency

After the run, the `seff` command allows you to access information about the efficiency of a job.
Try it now !
`seff <jobid>`

```
[(base) [silvert@ipop-up training]$ sbatch flatter.sh
Submitted batch job 239831
[(base) [silvert@ipop-up training]$ seff 239831
Job ID: 239831
Cluster: production
User/Group: silvert/umr7216
State: COMPLETED (exit code 0)
Cores: 1
CPU Utilized: 00:00:00
CPU Efficiency: 0.00% of 00:00:00 core-walltime
Job Wall-clock time: 00:00:00
Memory Utilized: 0.00 MB (estimated maximum)
Memory Efficiency: 0.00% of 1.95 GB (1.95 GB/core)
```

Bringing it all together

Exercise : Alignment

Run an alignment using STAR version 2.7.5a

Files

FASTQ files to align : `/shared/banks/mus_musculus/test_fastq`
aligner to use : `star-2.7.5a`
index : `/shared/banks/mus_musculus/mm39/star-2.7.5a`
memory needed : 25G

```
STAR --genomeDir $pathToIndex \
--readFilesIn $pathToFastq1 $pathToFastq2 \
--outFileNamePrefix $outputFileName \
--readFilesCommand zcat
```

Example solution

```
#!/bin/bash
###SBATCH OPTIONS###
#SBATCH --partition=ipop-up
#SBATCH --job-name=trainingAlignment
#SBATCH --output=star-alignment-%j.out
#SBATCH --error=star-alignment-%j.err
#SBATCH --mem=25G

###Script###
module purge
module load star/2.7.5a

...
```

Monitoring your jobs, seff

Check the resource that was used.

```
[hennion @ ipop-up 11:51]$ ~ : seff 239787
Job ID: 239787
Cluster: production
User/Group: hennion/umr7216
State: COMPLETED (exit code 0)
Cores: 1
CPU Utilized: 00:09:57
CPU Efficiency: 100.67% of 00:09:53 core-walltime
Job Wall-clock time: 00:09:53
Memory Utilized: 24.64 GB
Memory Efficiency: 98.56% of 25.00 GB
```

Some vocabulary

- job : A script, typically started with sbatch
- job step : A specific step in the big job, it can be a "srun" line within the script
- job task : A unit of resource allocation

We will not go into `srun` usage here, but we can talk about it later if you want.

Useful options 2/2

Options	Default	Function
−−nodes	1	Number of nodes required (or min-max)
−−nodelist		Select one or several nodes
−−ntasks-per-node	1	Number of tasks invoked on each node
−−mem	2GB	Memory required per node
−−cpus-per-task	1	Number of CPUs allocated to each task
−−mem-per-cpu	2GB	Memory required per allocated CPU
−−array		Submit multiple jobs to be executed with identical parameters

Ask for more CPUs for a tool

Some tools allow multi-threading, i.e. the use of several CPUs to accelerate one task.
It is the case of STAR with the `--runThreadN` option.

Exercise : Alignment, parallel

Modify the previous sbatch file to use 4 threads to align the FASTQ files on the reference. Run and check time and memory usage.

Ask for more CPUs for a tool

```
#!/bin/bash
###SBATCH OPTIONS###
#SBATCH --partition=ipop-up
#SBATCH --cpus-per-task=4
#SBATCH --mem=25G

###Script###
module purge
module load star/2.7.5a

STAR --runThreadN $SLURM_CPUS_PER_TASK
     ...
```

The cost of parallelization

- It may cost more in memory
- The gain in time is not linear

Job arrays

Job arrays allow to start the same job a lot of times (same executable, same resources)

```
#!/bin/bash
###SBATCH OPTIONS###
#SBATCH --partition=ipop-up
#SBATCH --array=0-3
#SBATCH --output=HelloArray_%A_%a.out

###Script###
echo "Hello I am the task number $SLURM_ARRAY_TASK_ID \
from the job array $SLURM_ARRAY_JOB_ID."

SAMPLE_LIST=(SRR11806587 SRR11806588 SRR11806589 SRR11806590)
SAMPLE=${SAMPLE_LIST[$SLURM_ARRAY_TASK_ID]}
echo "And I will process sample $SAMPLE."
```

Job arrays examples

Take all FASTQ files in a directory:

```
#SBATCH --array=0-3    # If 4 files
PATH2="/shared/banks/mus_musculus/test_fastq/"
cd $PATH2
FQ=(*fastq.gz)
echo ${FQ[@]}
INPUT=$(basename -s .fastq.gz "${FQ[$SLURM_ARRAY_TASK_ID]}")
echo $INPUT
```

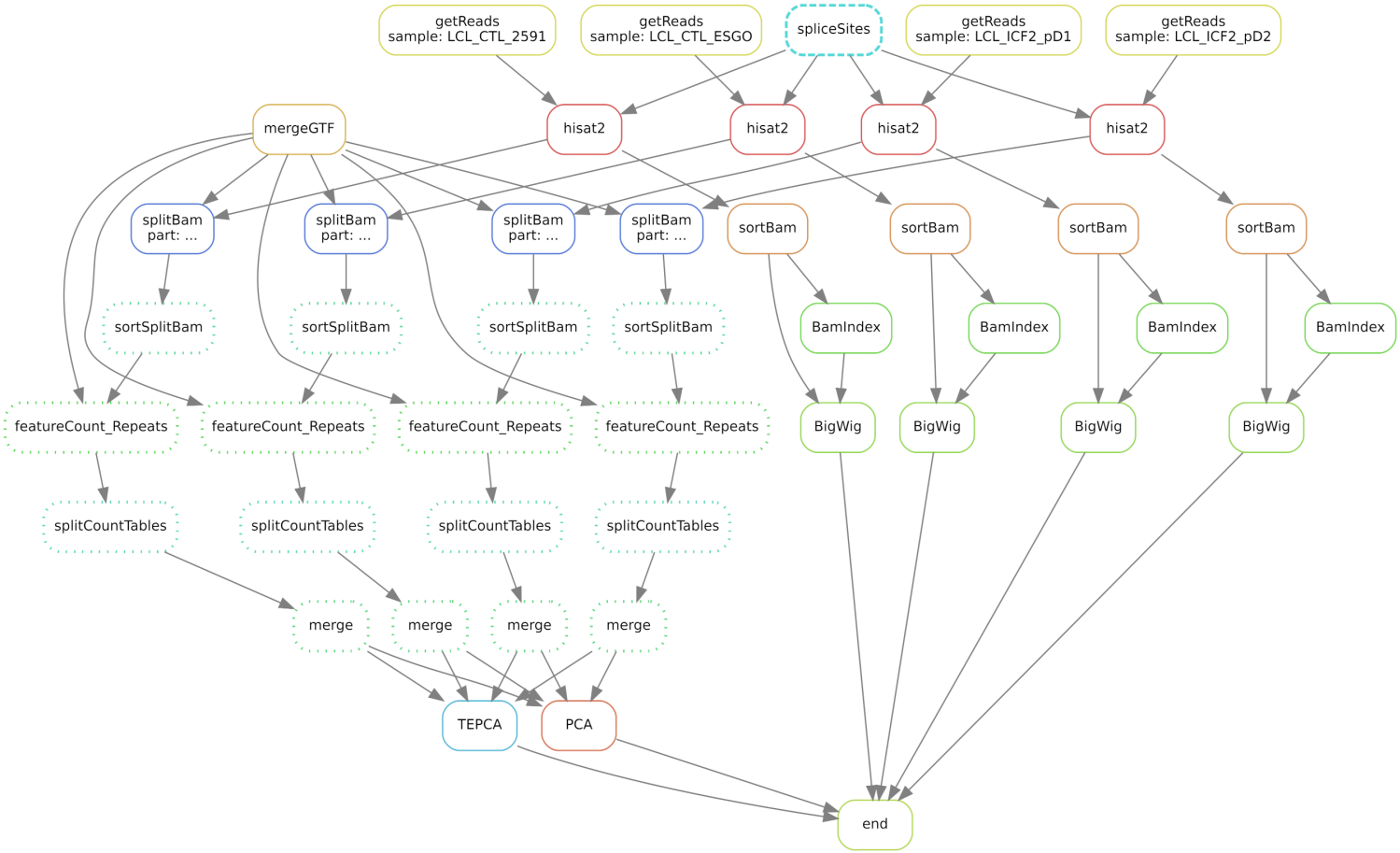
List or find files to process (`ls` or `find`) and get the nth with `sed` (or `awk`)

```
#SBATCH --array=1-4    # If 4 files, as sed index start at 1
INPUT=$(ls $PATH2/*.fq.gz | sed -n ${SLURM_ARRAY_TASK_ID}p)
echo $INPUT
```

Job Array Common Mistakes

- The index of bash lists starts at 0
- Don't forget to have different output files for each task of the array
- Same with your log names (%a or %J in the name will do the trick)
- Do not overload the cluster! Please use %50 (for example) at the end of your indexes to limit the number of tasks (here to 50) running at the same time. The 51st will start as soon as one finishes!
- The RAM defined using `#SBATCH --mem=25G` is for each task

Complex workflows



Use workflow managers such as Snakemake or Nextflow.

Useful resources

To find out more, the SLURM manual : `man sbatch` or https://slurm.schedmd.com/sbatch.html

Ask for help or signal problems on the cluster : https://discourse.rpbs.univ-paris-diderot.fr/

iPOP-UP cluster documentation: https://ipop-up.docs.rpbs.univ-paris-diderot.fr/documentation/

Thanks

- Julien Rey
- Olivier Kirsh

iPOP-UP's technical and steering comitees

iPOP-UP