INSTITUT FRANÇAIS
DE BIOINFORMATIQUE

The IFB Core Cluster Infrastructure

FAIR Bioinfo 2020

Gildas Le Corguillé & Julien Seiler
IFB Core Cluster taskforce

The **IFB C**ore **C**luster **I**nfrastructure

5042 cores (hyperthreaded)

26 To RAM

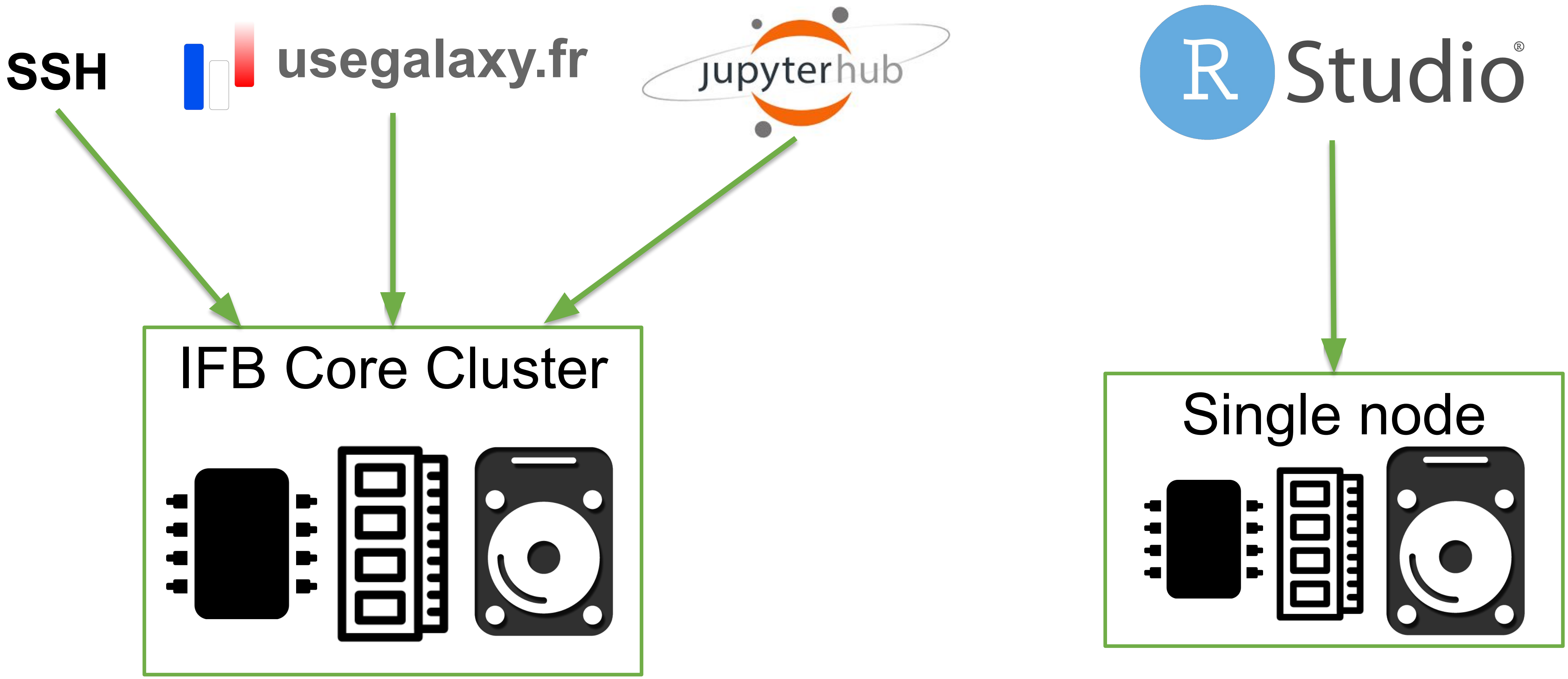
400 To

2.5PB in a few days

A support community made of users, administrators and experts

More than 300 tools

The IFB Core Cluster Infrastructure



SSH

usegalaxy.fr

jupyterhub

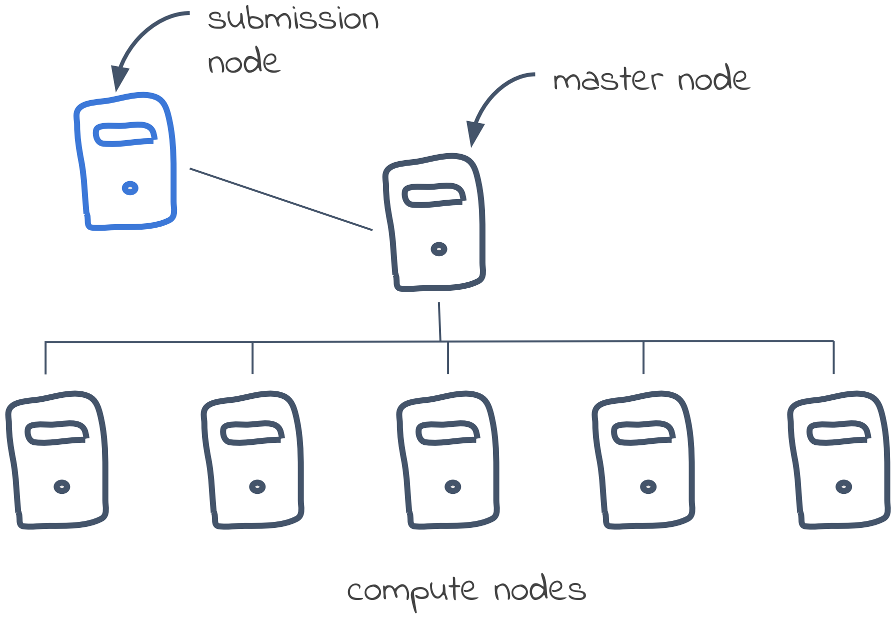
RStudio®

IFB Core Cluster

Single node

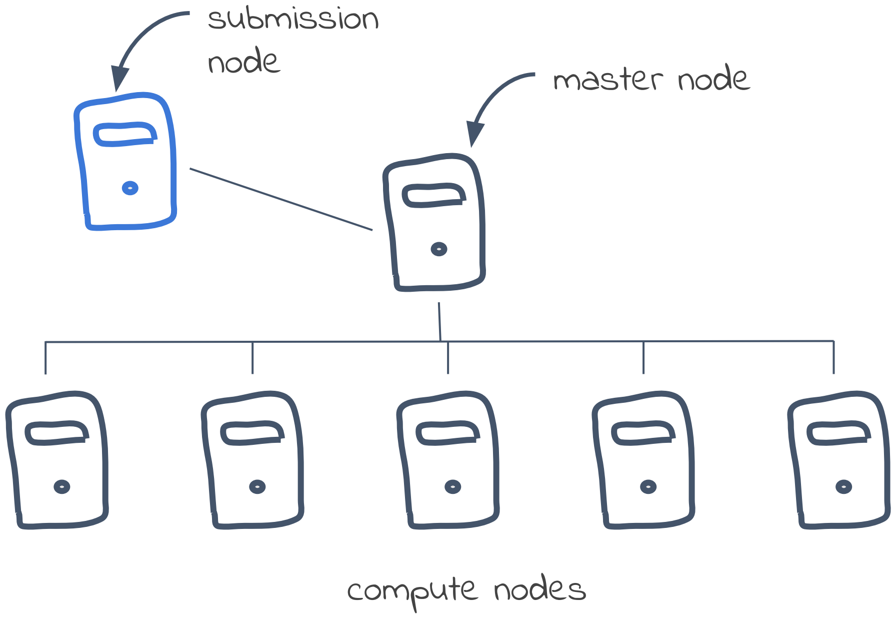
What is a **HPC** **C**luster ?

Basically, it is a bunch of computers working together



What is a **HPC Cluster** ?

These are the computers you want to use



What is a **HPC C**luster ?

How does a computer work ?

one or more chips

A chip (or microprocessor) is responsible for executing elementary instructions requested by the software

RAM (Random access memory)

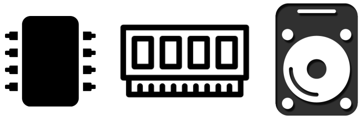
RAM is used by the chip to process data (a personal computer has between 4 to 8 GB of RAM)

storage space

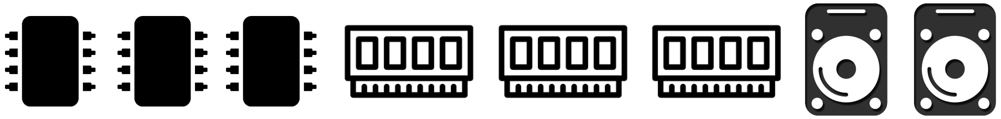
The storage space is used to keep huge amount of data in a more permanent way (a personal computer has an average of one TB of storage space)

What is a **HPC C**luster ?

How does a computer work ?



A personal computer has enough resources to let you run a lot of tasks like **browsing the Internet**, **work with spreadsheet** or **text processing software**. Some personal computers have even enough resources to let **process videos** or **play 3D video games**.



However, personal computer are not powerful enough to run **massive data analysis programs**. Indeed, these programs need a huge number of processing units (10 to 100 CPUs), huge amounts of RAM (100 GB for some programs) and large data storage capabilities (several TB for a single research project).

What is a **HPC C**luster ?

A set of big computers connected together that can be considered as a single system.

A HPC cluster is usually located in a **data center**, *i.e.* a dedicated room providing all conditions required by HPC in terms of temperature, humidity, power supply and physical security.

Typical cluster admin outfit

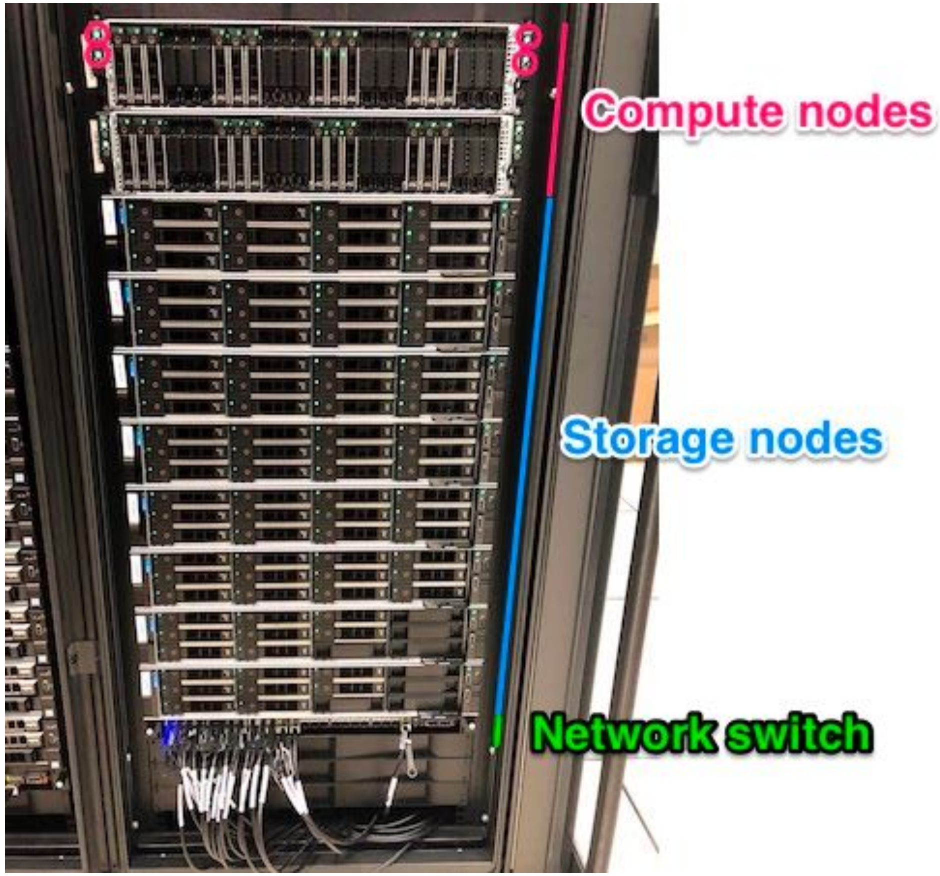
What is a **HPC Cluster** ?

A data center contains **racks**

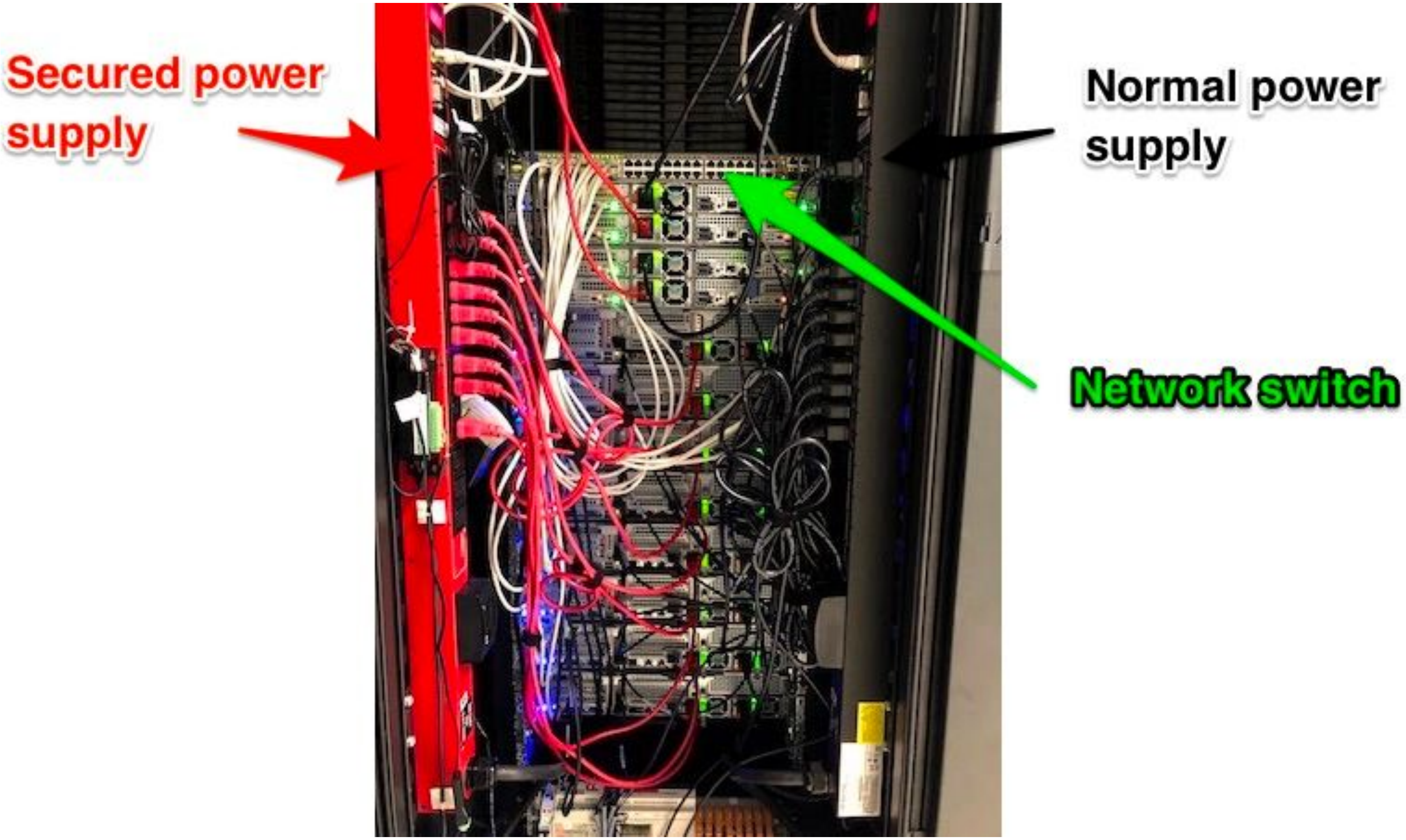
What is a **HPC Cluster** ?

Each rack can hold several computers

Front view



Rear view

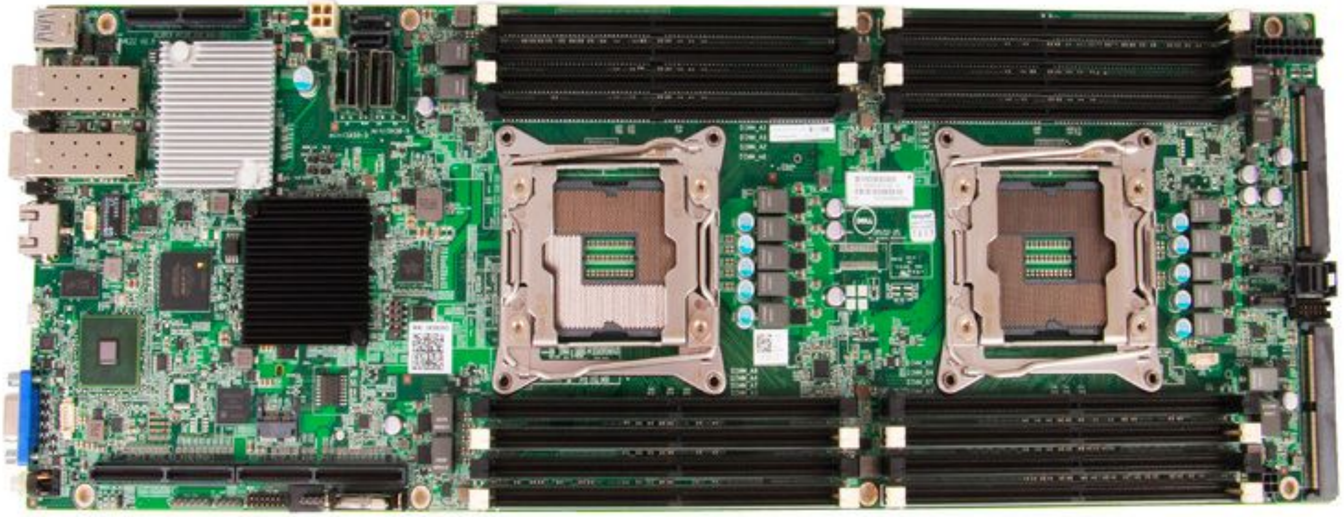


What is a **HPC C**luster ?

Inside a **computer** : a node = a physical machine

Each physical machine has one **motherboard**

Rear view



Front view

This motherboard has 2 **sockets** to plug **microprocessors**.
A microprocessor is a **multicore** technology.

What is a **HPC C**luster ?

Do not get confused between Microprocessor and Core

A microprocessor is a physical chip

Nowadays, one microprocessor contains **several cores**.

Each core behaves like a real separated microprocessor.

What is a **HPC** Cluster ?

The IFB cluster federation (NNCR)

Cluster	Data center location	Cores	RAM (GB)	Storage (TB)	Access modality
IFB Core	IDRIS - Orsay	5 042	26 542	400*	Open to all academic biologists and bioinformaticians
Genotoul	Toulouse	6 128	34 304	3 000	Open to all academics with priority to INRA/Occitane region (currently overloaded)
ABiMS	Roscoff	2608	10 600	2 500	Open to all academic biologists and bioinformaticians
GenOuest	Rennes	1 824	7 500	2 300	Open to all academic biologists and bioinformaticians
Migale	Jouy en Josas	1 084	7 000	350	Open to all academic biologists and bioinformaticians
BiRD	Nantes	560	4 000	500	Open to all academic biologists and bioinformaticians

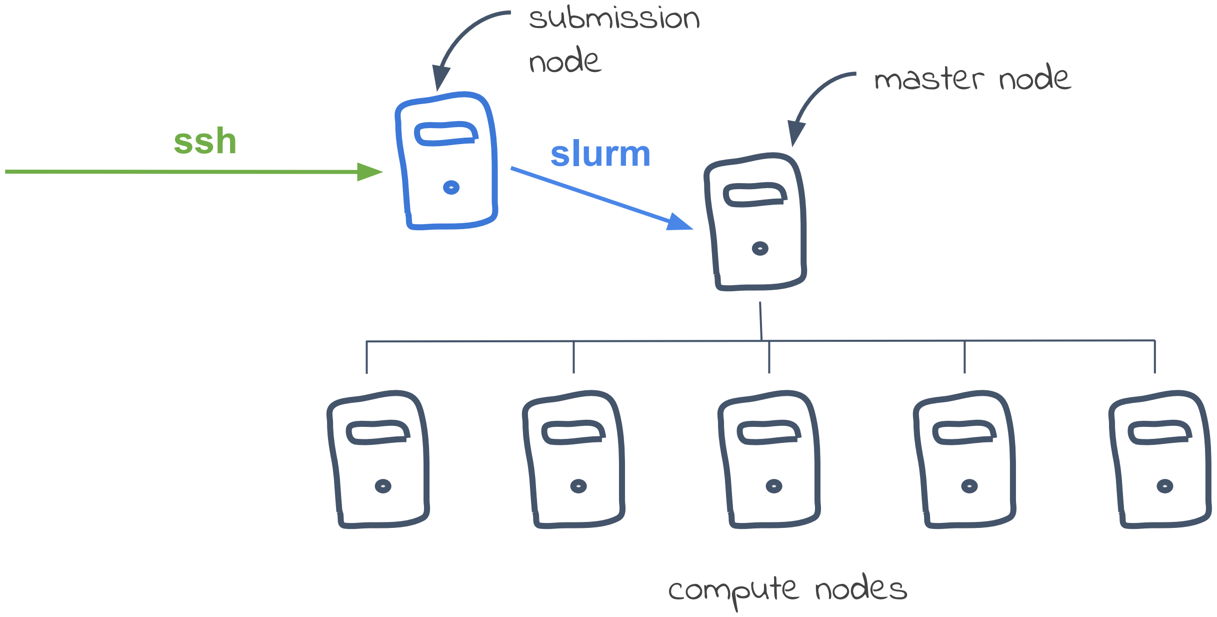
Introduction to **SLURM**

Common terms

- **Job** : a reservation of resources to run some analysis. A job is composed of one or more job steps that consume the reserved resources (eventually in parallel).
- **Job step** : part of a job that consist in the execution of a program. One job step can use multiple tasks. By default a job step uses one task.
- **Tasks** : a single process. One task can use multiple CPU (multi-threaded process).
- **CPU** : smallest computer processor unit (generally a single processor core).
- **RAM** : memory used by a processor to store data being computed

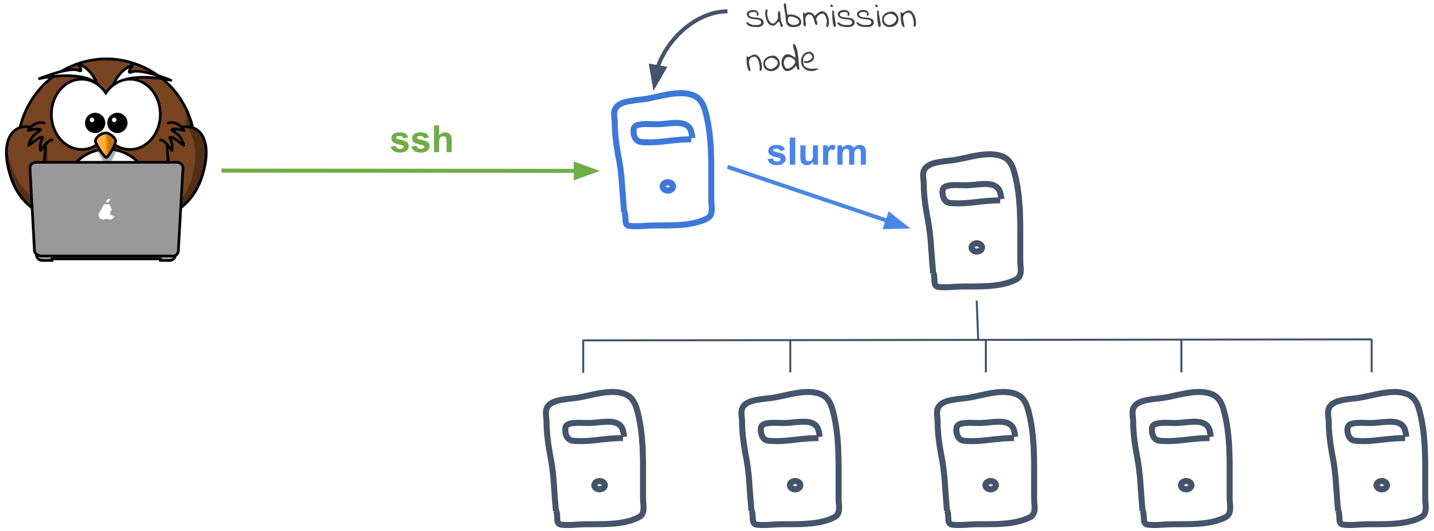
Introduction to **SLURM** - **How to submit a job**

```
ssh <login>@core.cluster.france-bioinformatique.fr
```



You are connected to the “submission node” of the cluster !
or “login node”

Introduction to **SLURM** - **How to submit a job**



Don’t run any computing software on the submission node

It is very weak and not designed for computing !

Introduction to **SLURM** - How to submit a job

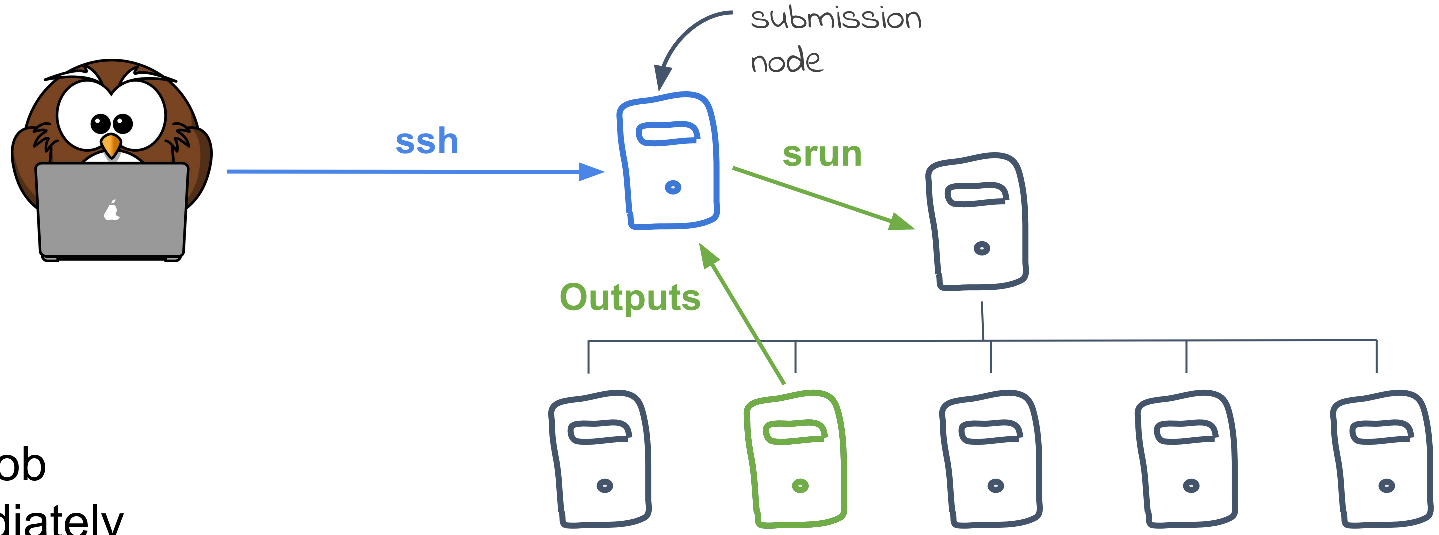
To run a simple command on the cluster, use **srun**

Example : `srun tar -xvzf my_big_data.tar.gz`

srun will :

- Reserve CPU and memory for your job (by default 1 CPU and 2GB of RAM on a single node)
- Wait for these resources to become available
- Run the given command on the compute node selected by SLURM
- Send back the command outputs to the user terminal

Introduction to **SLURM** - **How to submit a job**



```
tar -xvzf my_big_data.tar.gz
```

srun : simple interactive job

- Starts or waits immediately
- Outputs are returned to the terminal
- You have to wait until the job has terminated before starting a new job
- Works with **ANY command**

```
$ srun tar -xvzf my_big_data.tar.gz
```

Introduction to **SLURM** - How to submit a job

srun in brief

- `srun <command>`
- Default settings :
  - 1 CPU core
  - 2 GB RAM
- Common parameters :
  - `--cpus=`
  - `--mem=` (Warning : SLURM is enforcing memory usage !)
  - `--nodes=`
- Outputs comes in your console directly
- The console is blocked while your job is running

Introduction to **SLURM** - **How to submit a job**

Most of the time, you don't want to run a single command and don't want to wait for each command to end to start the next one.

What you want is running a batch script !

A batch script can be any shell script (Bash, R, Python etc.) but most of the time we use **Bash**.

Here is a simple example : my_script.sh

shebang is mandatory !

```
#!/bin/bash
srun tar -xvzf my_data.tar.gz
srun analyse my_data
```

each srun is a job step

Introduction to **SLURM** - **How to submit a job**

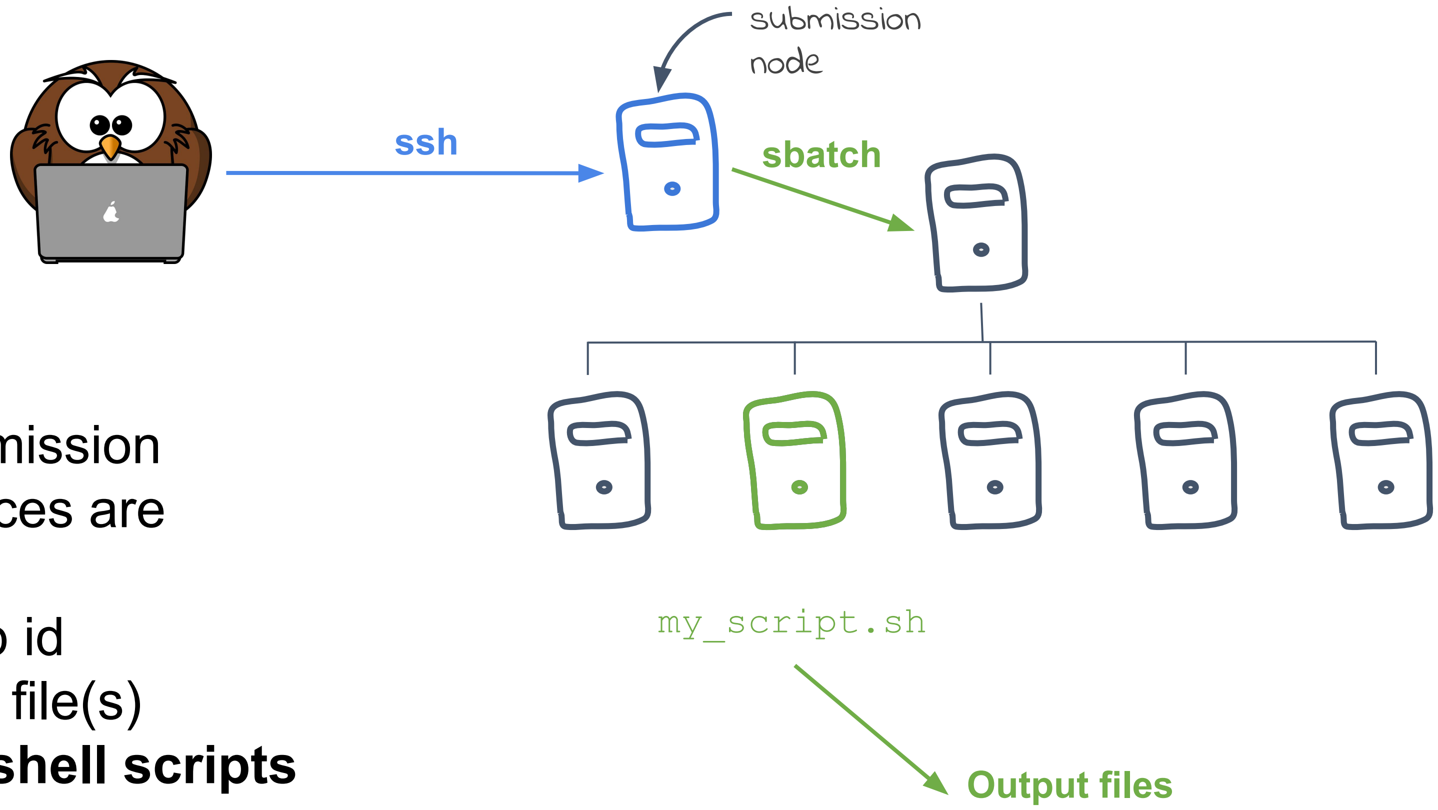
To run a batch script on the cluster, use **sbatch**

Example : `sbatch my_script.sh`

sbatch will :

- Reserve CPU and memory for your job (by default 1 CPU and 2GB of RAM on a single node)
- Place the job in the waiting queue and return
- When the resources are available start the job script step by step.
- Outputs are written to files

Introduction to **SLURM** - **How to submit a job**



sbatch : batch job submission

- Starts when resources are available
- Only returns the job id
- Outputs are sent to file(s)
- Works **ONLY with shell scripts**

```
$ sbatch my_script.sh
```

Introduction to **SLURM** - **How to submit a job**

sbatch in brief

- `sbatch <command>`
- Default settings :
  - 1 CPU core
  - 2 GB RAM
- Common parameters :
  - `--cpus=`
  - `--mem=` (Warning : SLURM is enforcing memory usage !)
  - `--nodes=`
- sbatch is NOT blocking the console
- Outputs comes in files (slurm-<jobid>.out and slurm-<jobid>.err)
- Works ONLY with Shell script

Introduction to **SLURM** - How to submit a job

sbatch in brief

- You can pass sbatch parameters in your shell script directly

```
#!/bin/bash
#
#SBATCH -p fast                      # partition
#SBATCH --cpus 10                    # number of CPU cores required
#SBATCH --mem 100                    # memory pool for all cores
#SBATCH -t 0-2:00                    # time (D-HH:MM)
#SBATCH -o slurm.%N.%j.out           # STDOUT
#SBATCH -e slurm.%N.%j.err           # STDERR
#SBATCH --mail-type=ALL              # can be BEGIN, END, FAIL or REQUEUE
#SBATCH --mail-user=your@email.com

srun ––mem=50 bash -c "prepare_data > large.dataset"
srun big_computing_tool large.dataset
```

- `sbatch my_script.sh`

Introduction to **SLURM - Summary**

- Get connected to the login node and keep working on it

- For **basic command** (cd, ls, mv, mkdir…), run it directly on the "**submission node"**

- For all the rest, including **bioinformatics tools,** prepend all command lines by **srun** so that your job will be submitted to the cluster master and then run on a **node** of the cluster

- For batch treatment (like pipeline) use **sbatch**

Introduction to **SLURM – Job control**

squeue

View all job running on the cluster

```
$ squeue
```

View only my jobs

```
$ squeue -u <my_login>
```

View only my RUNNING jobs

```
$ squeue -t RUNNING -u <my_login>
```

Introduction to **SLURM – Job control**

Job resources

View resources used by a job

```
$ sacct --format=JobID,Submit,MaxVMSize,Start,NodeList,CPUTime,State -j <job_id>
```

View detailed information about one running job :

```
$ scontrol show jobid -dd <job_id>
```

Introduction to **SLURM – Job control**

sinfo

View available Slurm partitions

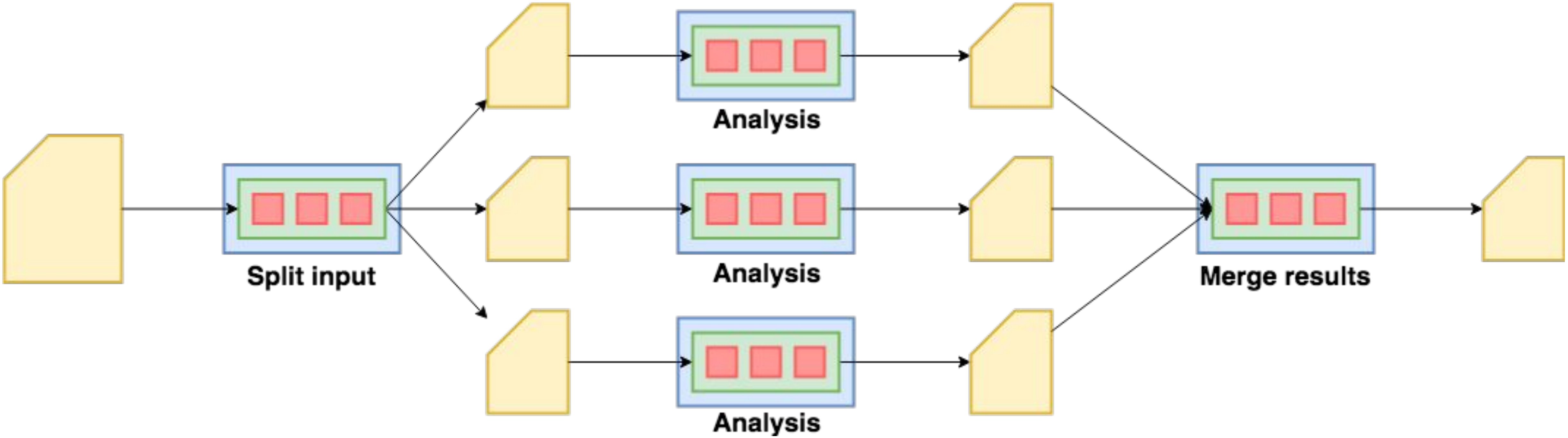
```
$ sinfo -l
```

View available Slurm nodes

```
$ sinfo -Nl
```

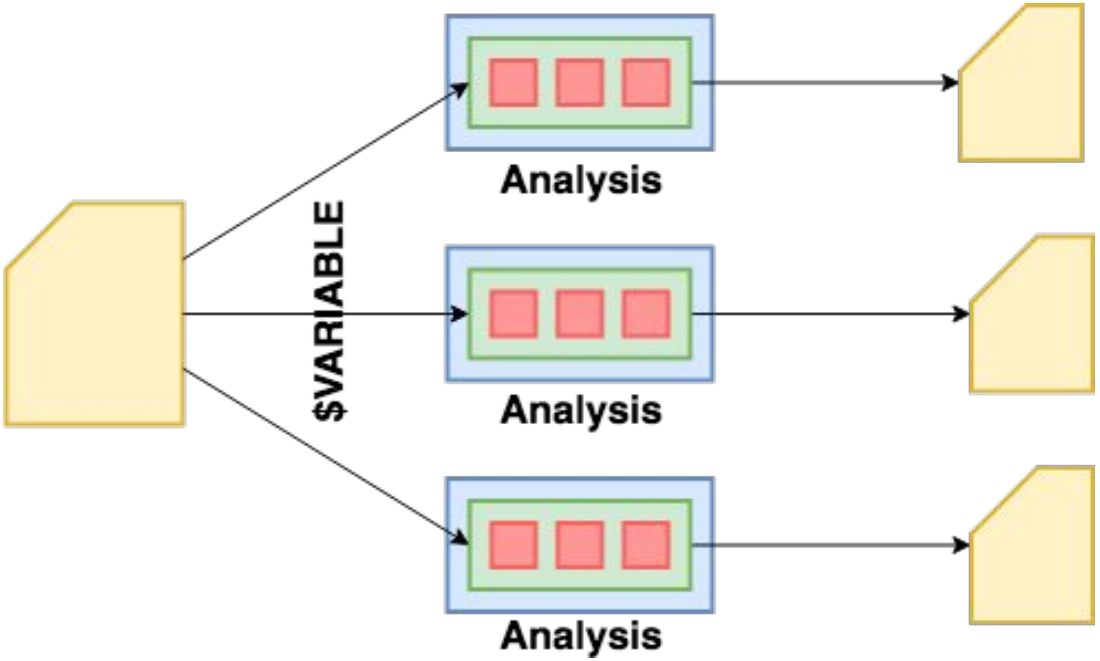
Introduction to **SLURM** – **Parallelization patterns**

Input data splitting



Introduction to **SLURM – Parallelization patterns**

Variables exploration



Introduction to **SLURM** – Parallelization patterns

Fastqc example

fastqc.sh

```
#!/bin/bash
#SBATCH --array=0-3 # 4 jobs
#SBATCH --cpus=1 # 1 cpu cores
module load fastqc/0.11.8
INPUTS=(../fastqc/*.fq.gz)
srun fastqc -t 1 ${INPUTS[$SLURM_ARRAY_TASK_ID]}
```

```
$ sbatch fastqc.sh
Submitted batch job 3161045
```

multiqc.sh

```
#!/bin/bash
srun multiqc .
```

```
$ sbatch --dependency=afterok:3161045 multiqc.sh
```

Approach	Total duration of the treatment
Sequential, single thread	13 minutes 53 seconds
Sequential, 16 threads	11 minutes 45 seconds
Parallel, 4+1 jobs, 1 thread	4 minutes 2 seconds

Using tool

Why do we need to "load" tools ?
Each tools need its environment (binaries, libraries, documentation, special variables)
Each tools has its own dependencies.
It is not possible to coexist all tools in the same environment.
Reproducibility does matter: some user might need different versions of the same tool
At the IFB, the cluster community is installing all tools required by the users.

All tool deplo

CONDA Singularity

To get access to a tool, you need to load it into your environment using a special tool called **`module`**.

Using tool

Which tools are available ?

```
$ module avail -l
abyss/2.2.1                                          2020/01/17 17:53:23
adxv/1.9.14                                          2020/08/28 17:25:16
anvio/6.1                                            2020/01/17 18:06:41
arcs/1.1.0                                           2020/01/17 18:09:18
ascatngs/2.1.0                                       2020/02/28 09:32:19
augustus/3.3.3                                       2020/05/25 18:55:47
bam2fastx/1.3.0                                      2020/06/04 14:07:17
bc/1.07.1                                            2020/01/17 18:10:51
bcftools/1.9                                         2020/01/17 18:12:17
bcftools/1.10.2                                      2020/07/30 09:33:18
bcl2fastq/2.20.0                                     2020/07/16 09:40:56
bedtools/2.26.0                                      2020/01/17 18:13:20
bedtools/2.27.1                                      2020/01/17 18:14:17
bedtools/2.29.2                                      2020/07/30 09:35:12
berokka/0.2.3                                        2020/04/15 17:14:05
bio-vcf/0.9.2                                        2020/07/13 10:58:48
...
```

Using tool

Loading, listing, switching, unloading

`module load blast`: Load latest version of blast available on the cluster

`module load blast/2.6.3`: Load version 2.6.3 of blast

`module list`: List tools currently loaded in your environment

`module switch blast/2.7.1`: Replace blast currently loaded by blast version 2.7.1

`module unload blast`: Unload blast from your environment

`module purge`: Unload all tools

Useful links

Request an account:
https://my.cluster.france-bioinformatique.fr

Community support:
https://community.france-bioinformatique.fr/

Learn SLURM in 5 minutes:
https://asciinema.org/a/275233

IFB Core Cluster Documentation
https://ifb-elixirfr.gitlab.io/cluster/doc/

BONUS

The **IFB** **C**ore **C**luster **I**nfrastructure

- Infrastructure administration is automated using Continuous Integration technologies :

ANSIBLE git GitLab IFBot

- Most IFB Core Cluster repositories are **open to contribution**
  - Help us manage the cluster infrastructure
  - Deploy bioinformatics software (conda, singularity, etc.)
  - Deploy new services

What’s new on the IFB NNCR Cluster(s) ?

David BENABEN [1,2] , Nicole CHARRIÈRE [3] , David CHRISTIANY [3] , François GERBES [3,6] , Jean-Christophe HAESSIG [4] ,
Didier LABORIE [5] , Gildas LE CORGUILLÉ [6*] , Olivier SALLOU [7] , Julien SEILER [4*] and Guillaume SEITH [4]

[1] CBiB, Université de Bordeaux, 142 rue Léo Saignat, 33076 Bordeaux, France
[2] INRAE, UMR 1332, Biologie du Fruit et Pathologie, CS20032 Villenave d'Ornon, France
[3] IFB/Institut Français de Bioinformatique, CNRS UMS 3601, IFB-Core, Génoscope, 91057, Évry, France
[4] CNRS, INSERM, IGBMC, 1 rue Laurent Fries, 67404, Illkirch, France
[5] GenoToul-Bioinfo, INRAE, 24 chemin de Borde-Rouge, Auzeville, 31326 Castenet-Tolosan, France
[6] Sorbonne Université/CNRS, FR2424, ABiMS, Station Biologique, 29680, Roscoff, France
[7] IRISA/Université Rennes 1, 263 Avenue Général Leclerc, 35000 Rennes, France

* Corresponding Authors: lecorguille@sb-roscoff.fr, julien.seiler@igbmc.fr

Montpellier
JOBIM
2020

Link