Grappe de calcul *PlaFRIM*

Luca Cirrottola (INRIA)

luca.cirrottola@inria.fr

Bordeaux INP ENSEIRB-MATMECA, Université de Bordeaux

Automne 2024

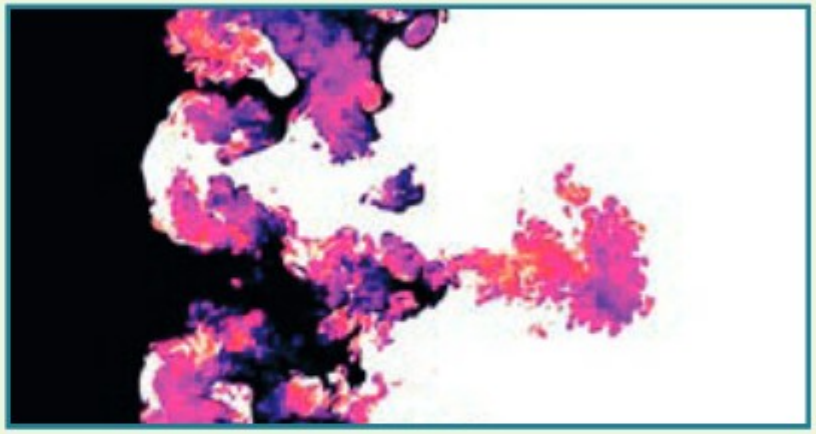
- Calcul haute performance (rappel):
  - Performances, architectures, écosystème
- Grappe de calcul *PlaFRIM*
- Utilisation de la grappe de calcul:
  - Accéder aux ressources
  - Gérer son environnement
  - Partager les ressources

Calcul haute performance

- Pourquoi ?

TERA 1

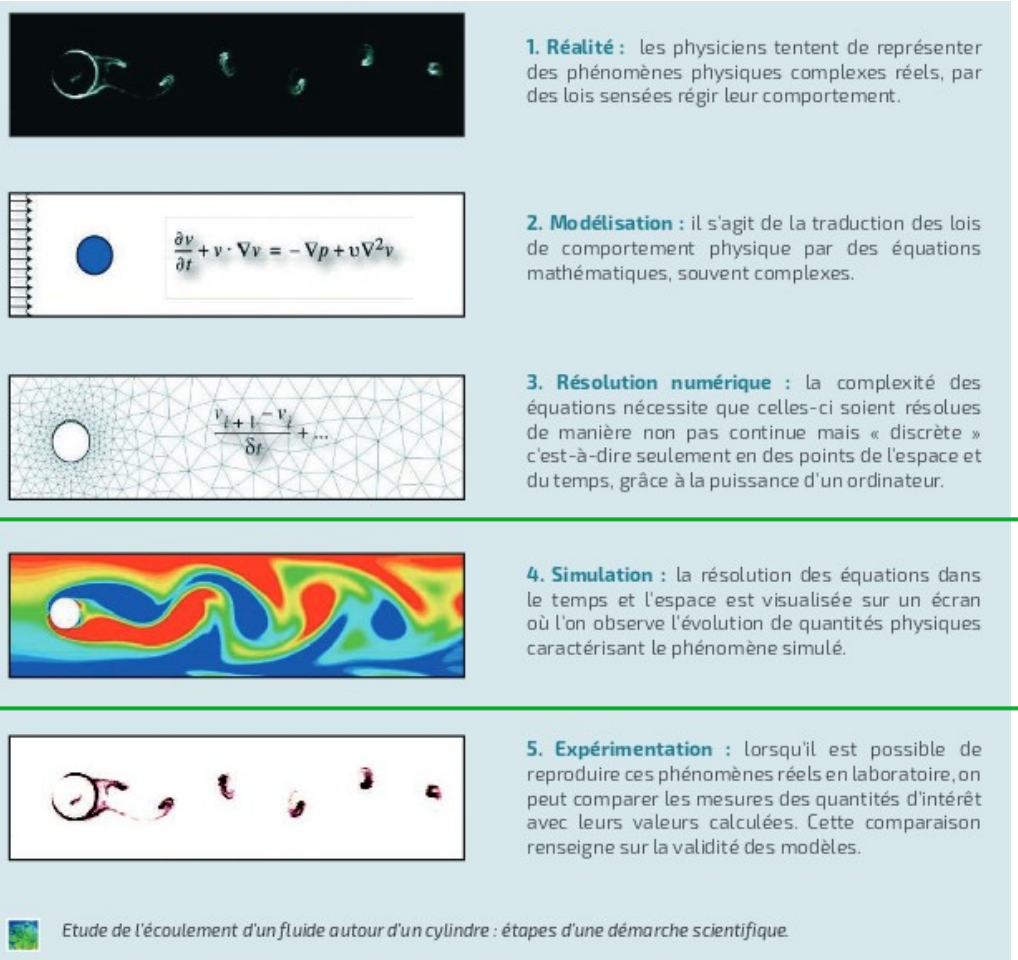
TERA 100



Evolution de la précision de simulation 2D d'un phénomène d'instabilités hydrodynamiques en fonction de la puissance des calculateurs TERA 1 à TERA 100 pour une même durée de calcul. La finesse de description des structures tourbillonnaires accessible par la simulation s'approche progressivement de la réalité physique.

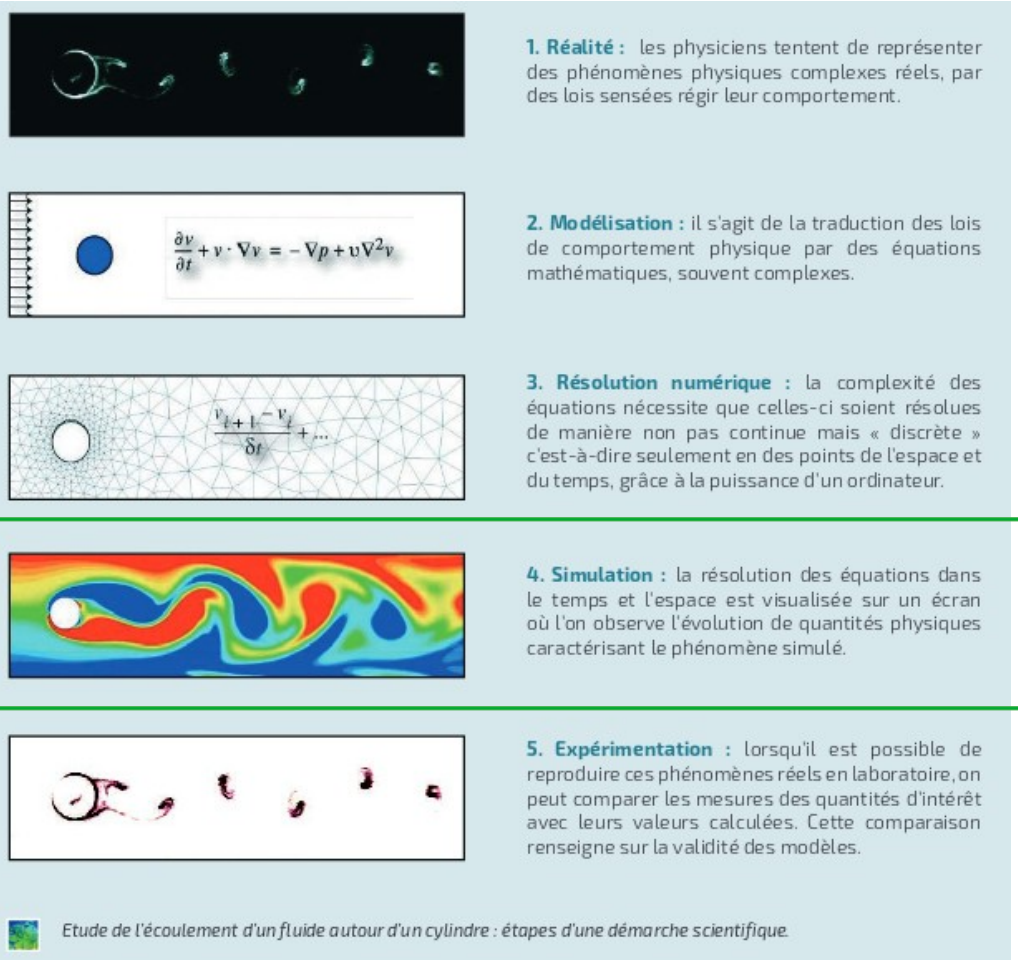
http://www.cea.fr/presse/Documents/actualites/20-ans-programme-simulation.pdf

Calcul haute performance



http://www.cea.fr/presse/Documents/actualites/20-ans-programme-simulation.pdf

Calcul haute performance



1. Réalité : les physiciens tentent de représenter des phénomènes physiques complexes réels, par des lois sensées régir leur comportement.

2. Modélisation : il s'agit de la traduction des lois de comportement physique par des équations mathématiques, souvent complexes.

3. Résolution numérique : la complexité des équations nécessite que celles-ci soient résolues de manière non pas continue mais « discrète » c'est-à-dire seulement en des points de l'espace et du temps, grâce à la puissance d'un ordinateur.

4. Simulation : la résolution des équations dans le temps et l'espace est visualisée sur un écran où l'on observe l'évolution de quantités physiques caractérisant le phénomène simulé.

5. Expérimentation : lorsqu'il est possible de reproduire ces phénomènes réels en laboratoire, on peut comparer les mesures des quantités d'intérêt avec leurs valeurs calculées. Cette comparaison renseigne sur la validité des modèles.

Etude de l'écoulement d'un fluide autour d'un cylindre : étapes d'une démarche scientifique.

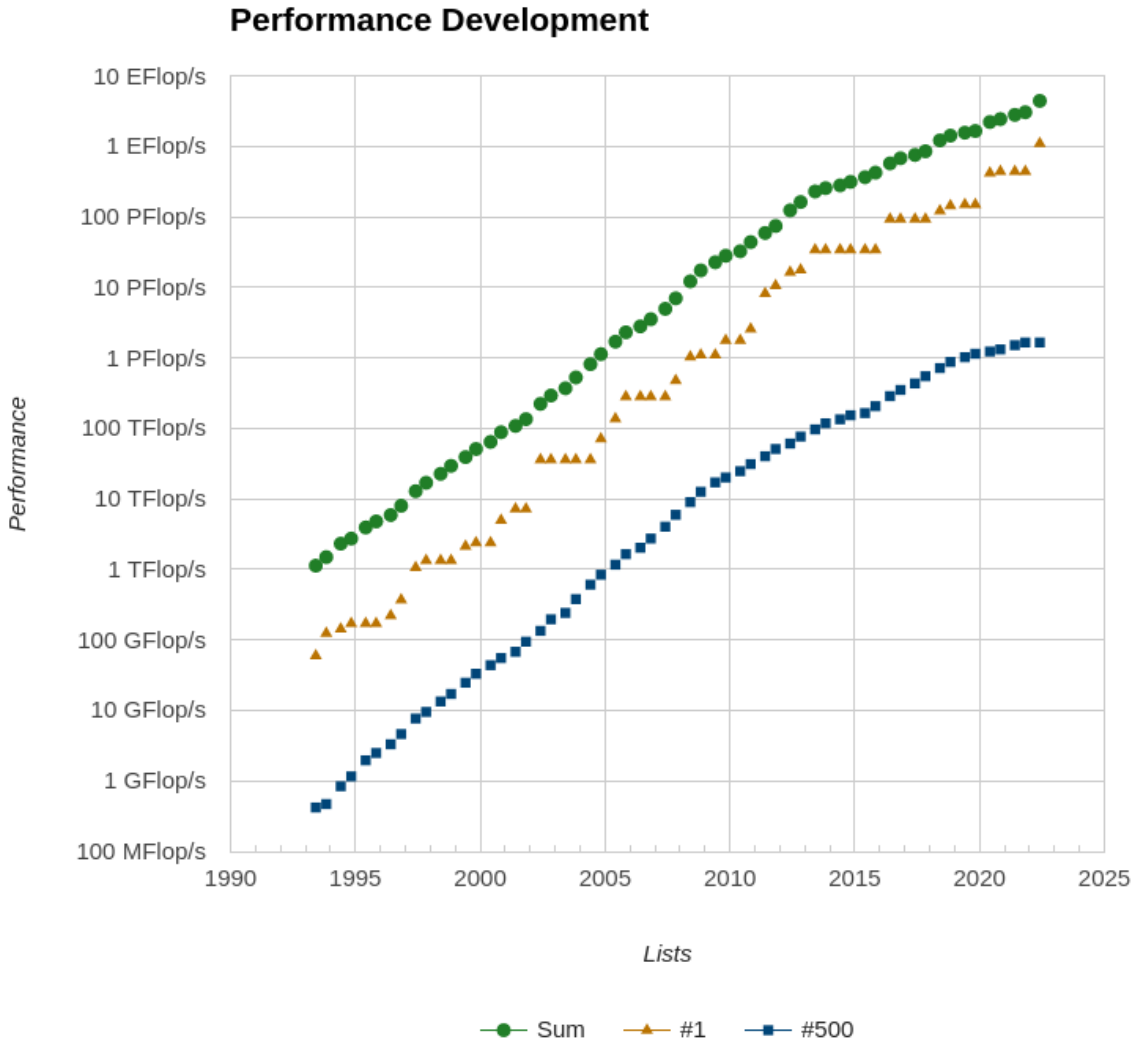
Moyens de calcul

http://www.cea.fr/presse/Documents/actualites/20-ans-programme-simulation.pdf

Superordinateurs

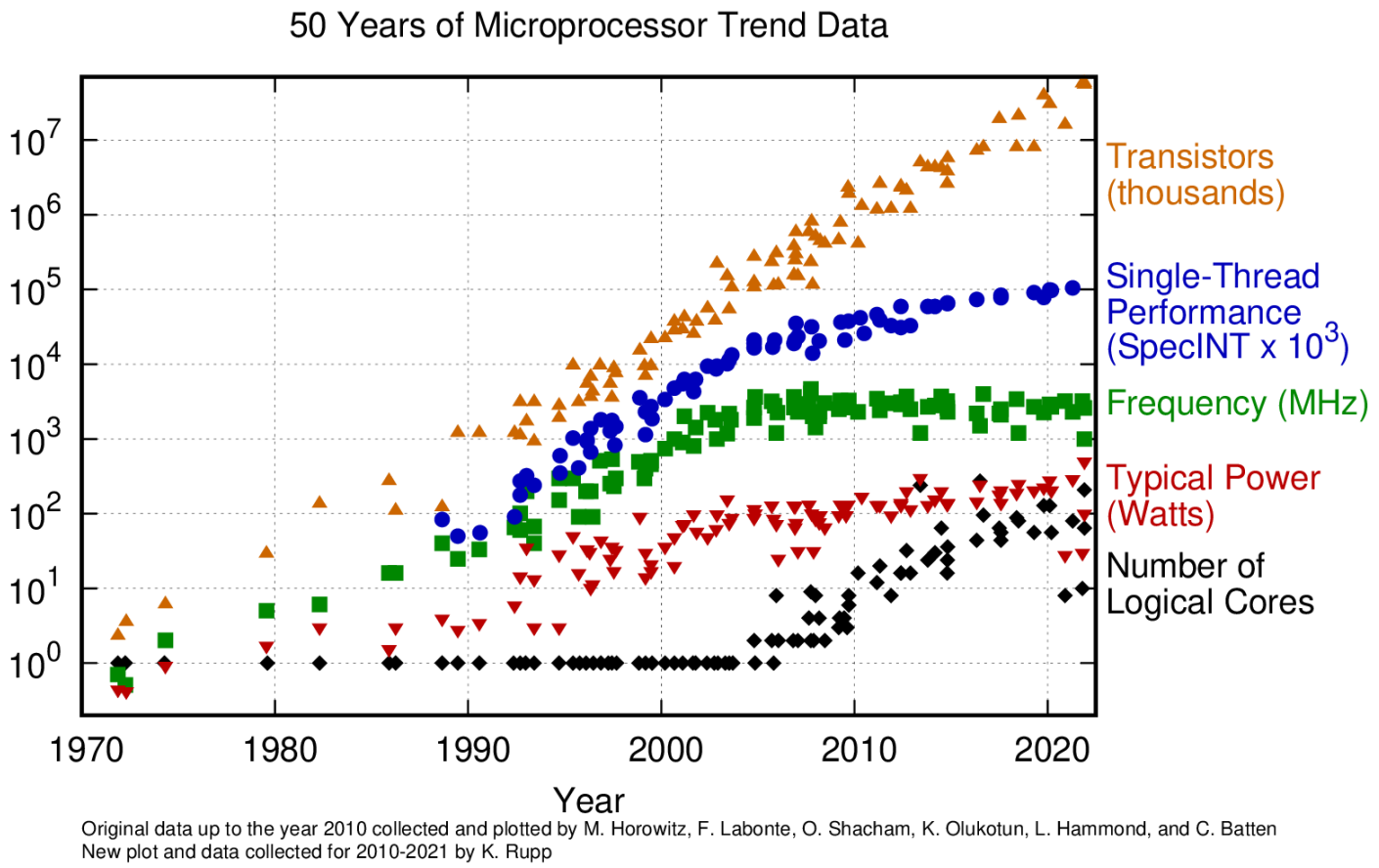
https://upload.wikimedia.org/wikipedia/commons/d/d8/2017_BSC_Superordenador_MareNostrum-4_Barcelona-Supercomputing-Center.jpg

Superordinateurs



https://www.top500.org/statistics/

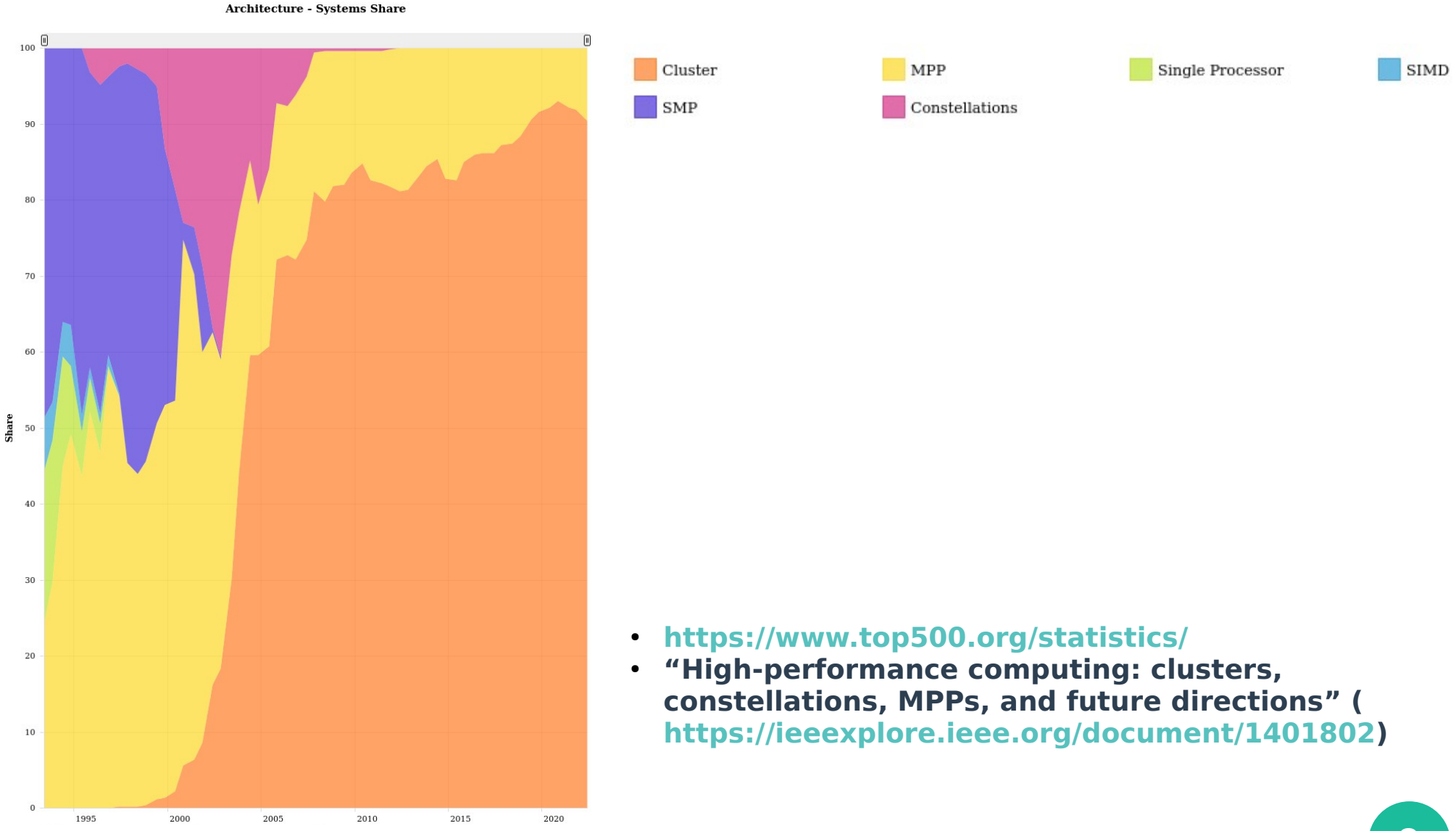
CPUs (multi-cœurs)



https://github.com/karlrupp/microprocessor-trend-data/blob/master/50yrs/50-years-processor-trend.png

https://www.karlrupp.net/2018/02/42-years-of-microprocessor-trend-data/

Superordinateurs

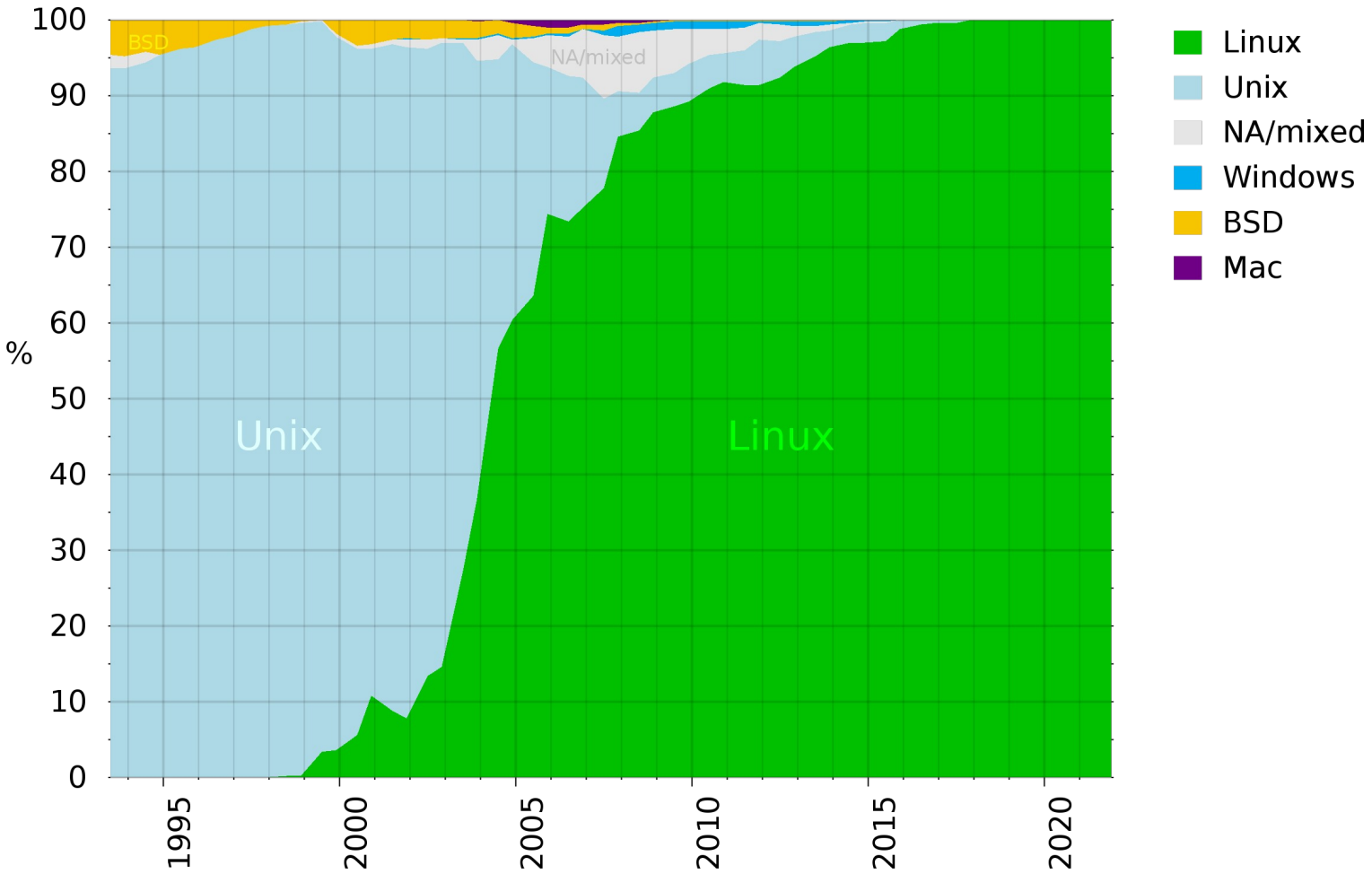


- https://www.top500.org/statistics/
- “High-performance computing: clusters, constellations, MPPs, and future directions” (https://ieeexplore.ieee.org/document/1401802)

Linux dans le CHP

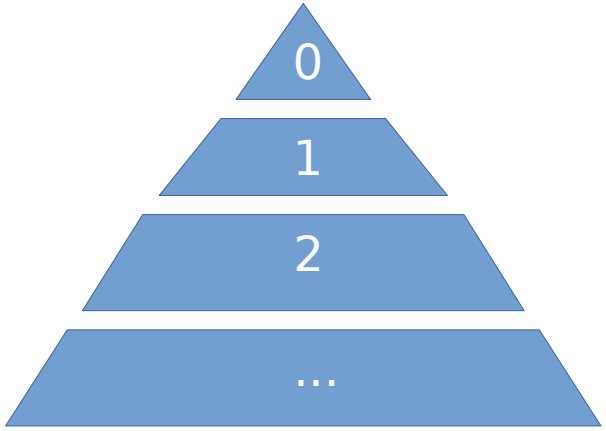
- **C’est libre:**
  - à utiliser,
  - à étudier et à adapter,
  - à copier et rédistribuer dans sa version originale,
  - à modifier et rédistribuer dans des versions modifiées.
- **C’est la norme (et pas l’exception) dans le monde du CHP.**

Linux dans le CHP



https://en.wikipedia.org/wiki/TOP500#/media/File:Operating_systems_used_on_top_500_supercomputers.svg

Centres de calcul en Europe



- **Tier 0: centres européens**
- **Tier 1: centres nationaux**
- **Tier 2: centres régionaux (mésocentres)**
- **(...): Centres de recherche locaux**

PRACE – Partnership for Advanced Computing in Europe (https://prace-ri.eu/)

GENCI – Grand Équipement National de Calcul Intensif (https://www.genci.fr/)

MCIA – Mésocentre de Calcul Intensif Aquitain (https://www.mcia.fr/)

PlaFRIM

-

Plateforme Fédérative pour la Recherche en Informatique et Mathématiques

https://www.plafrim.fr/

PlaFRIM

- Grappe

PlaFRIM

- Nœuds

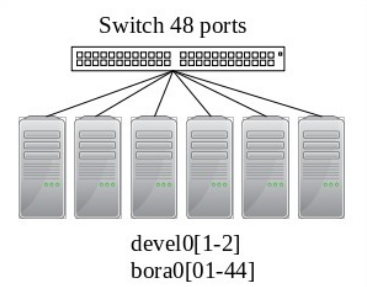
	CPU	Memory	GPU	Storage
bora001-044	2x 18-core Intel CascadeLake	192GB		/tmp of 1 To
miriel001-088	2x 12-core Intel Haswell	128 GB		/tmp of 300 GB
diablo01-04	2x 32-core AMD Zen2	256 GB		/tmp of 1 TB
diablo05	2x 64-core AMD Zen2	1 TB		/tmp of 1 TB
diablo06-09	2x 64-core AMD Zen3	1 TB		/scratch of 4 TB
zonda01-21	2x 32-core AMD Zen2	256 GB		
arm01	2x 28-core ARM TX2	256 GB		/tmp of 128 GB
sirocco01-02,05	2x 12-core Intel Haswell	128 GB	4 NVIDIA K40M	/tmp of 1 TB
sirocco03-04	2x 12-core Intel Haswell	128 GB	3 NVIDIA K40M	/tmp of 1 TB
sirocco06	2x 10-core Intel IvyBridge	128 GB	2 NVIDIA K40M	/tmp of 1 TB
sirocco07-13	2x 16-core Intel Broadwell	256 GB	2 NVIDIA P100	/tmp of 300 GB
sirocco14-16	2x 16-core Intel Skylake	384 GB	2 NVIDIA V100	/scratch of 750 GB
sirocco17	2x 20-core Intel Skylake	1 TB	2 NVIDIA V100	/tmp of 1 TB
sirocco18-20	2x 20-core Intel CascadeLake	192 GB	2 NVIDIA Quadro	
sirocco21	2x 24-core AMD Zen2	512 GB	2 NVIDIA A100	/scratch of 3.5 TB
sirocco22-25	2x 32-core AMD Zen3	512 GB	2 NVIDIA A100	/scratch of 4 TB
kona01-04	64-core Intel Xeon Phi	96GB + 16GB		/scratch of 800 GB
brise	4x 24-core Intel Broadwell	1TB		/tmp of 280 GB
souris	12x 8-core Intel IvyBridge	3TB		

PlaFRIM

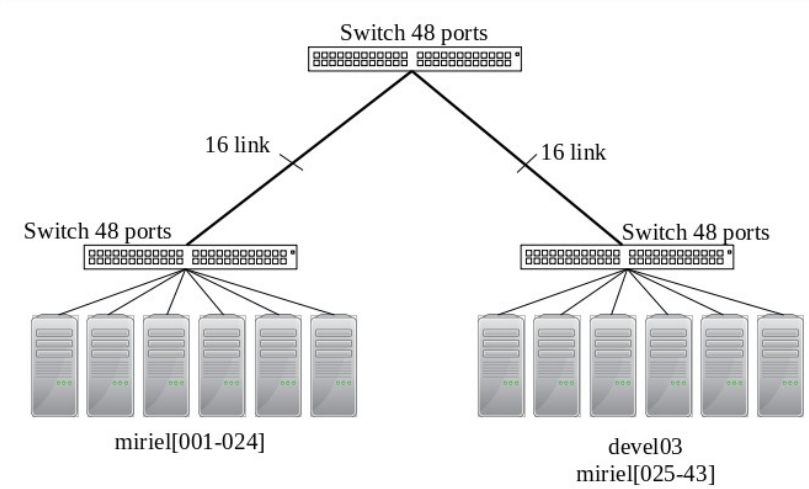
- **Interconnexion(s)**

OmniPath 100Gbit/s (3 separate networks):

- **bora nodes** (and devel[01-02]), also used for BeeGFS storage.

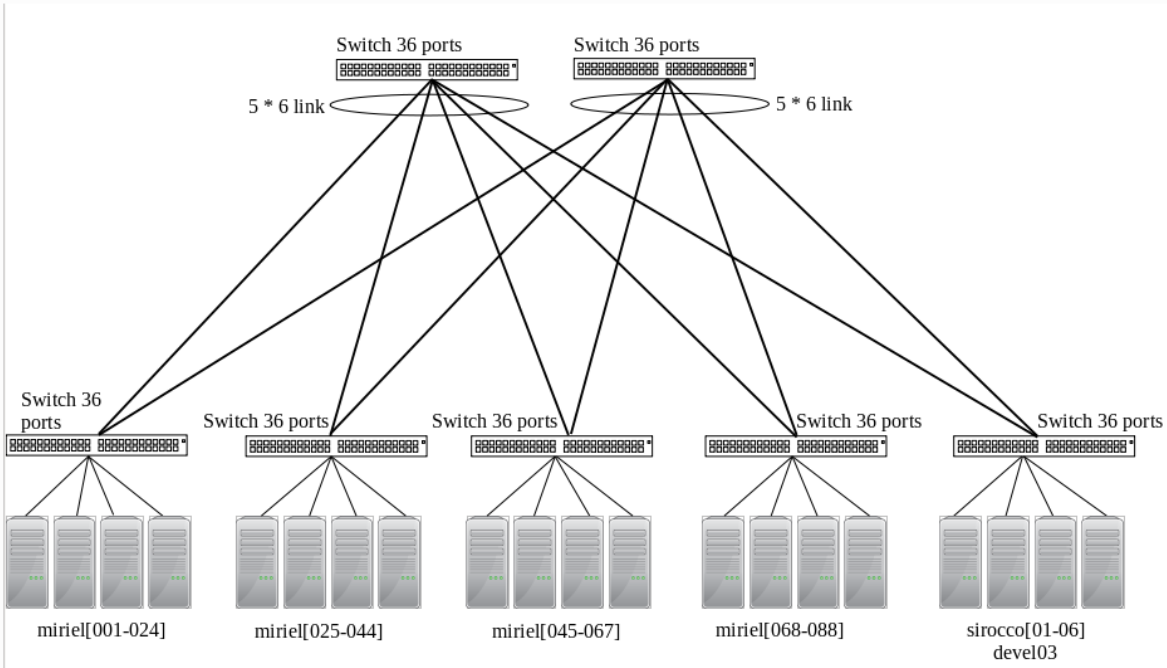


- **miriel[01-43]** (and devel03).

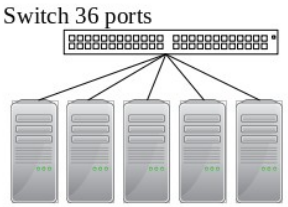


- **sirocco[07-17]** and all **kona** only

InfiniBand QDR 40Gbit/s between all **miriel** nodes and **sirocco[01-06]** (and devel03).

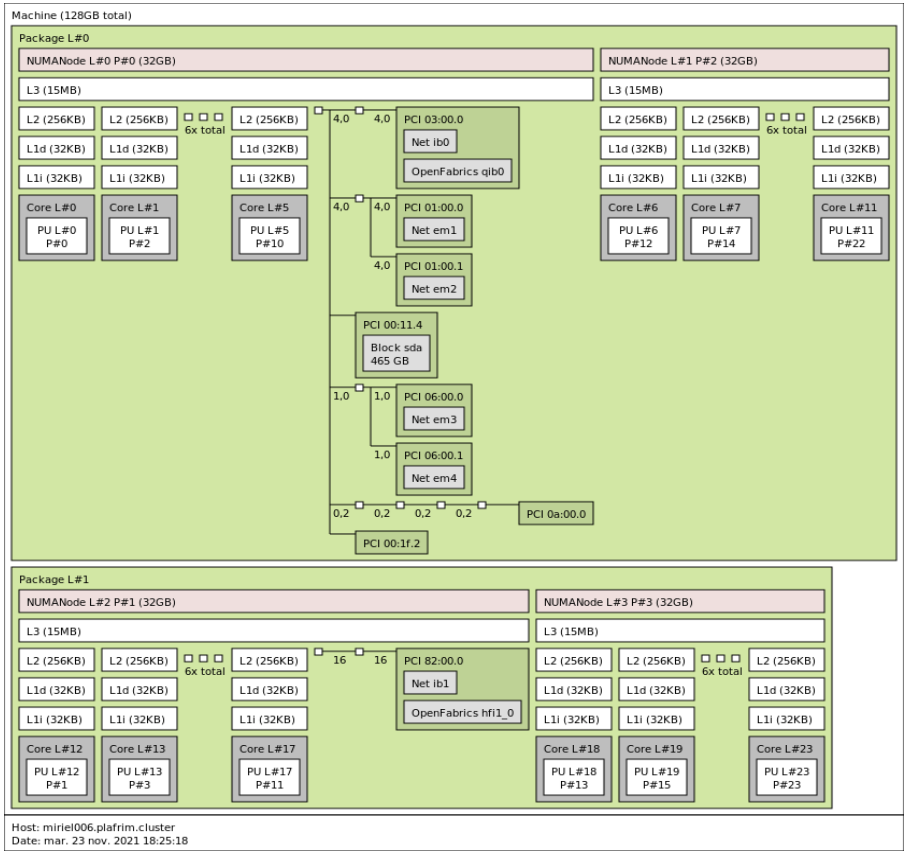


Mellanox InfiniBand HDR 200Gbit/s between **diablo[01-09]**.



PlaFRIM

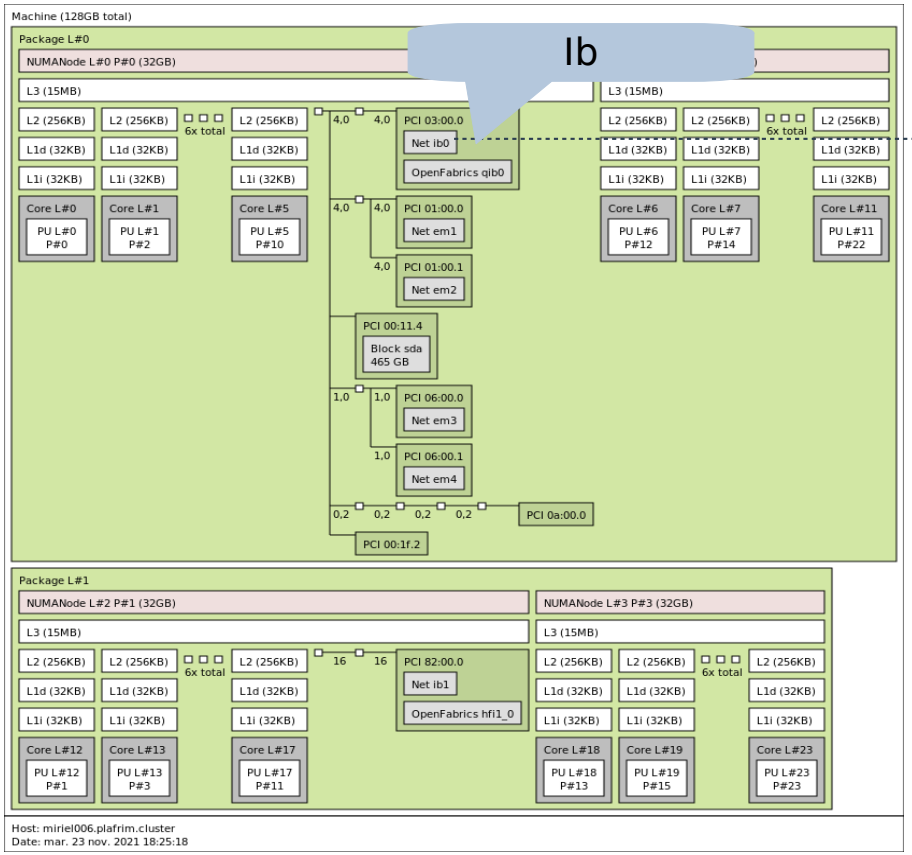
- Nœud (miriel)



https://www.open-mpi.org/projects/hwloc/

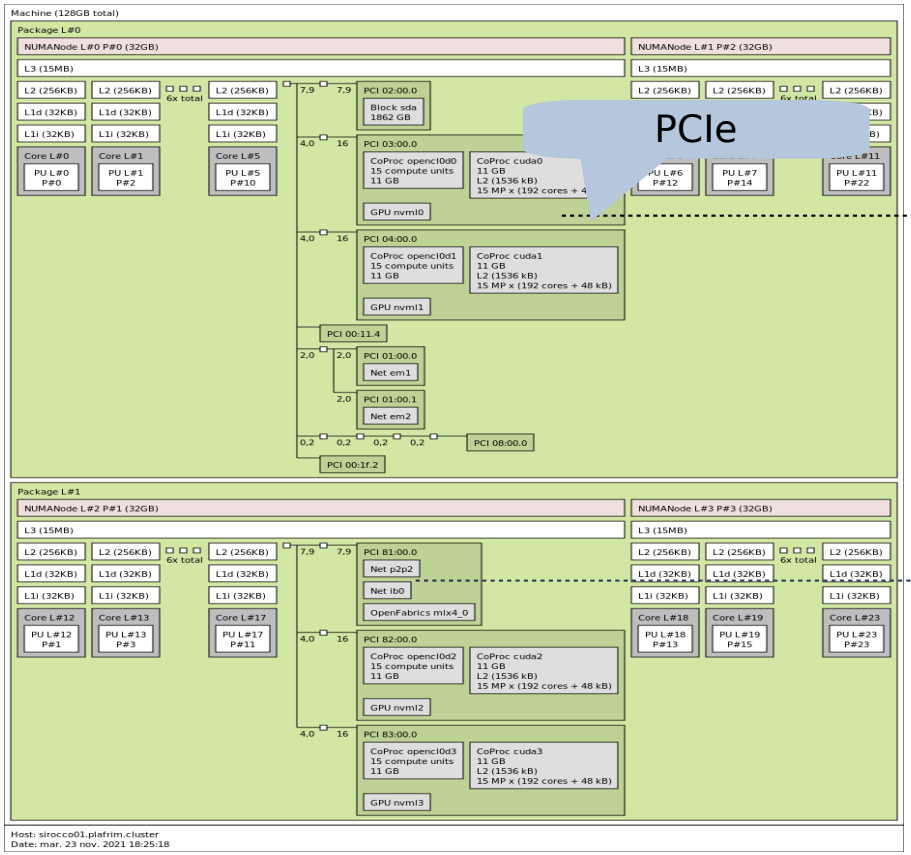
PlaFRIM

- Nœud (miriel)

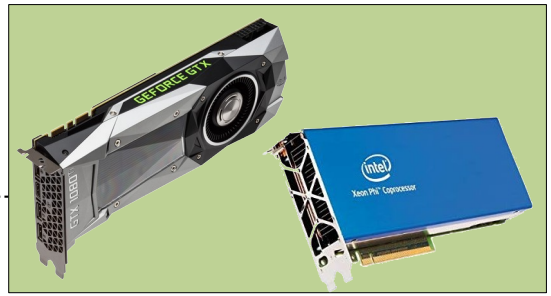


PlaFRIM

- Nœud (sirocco)



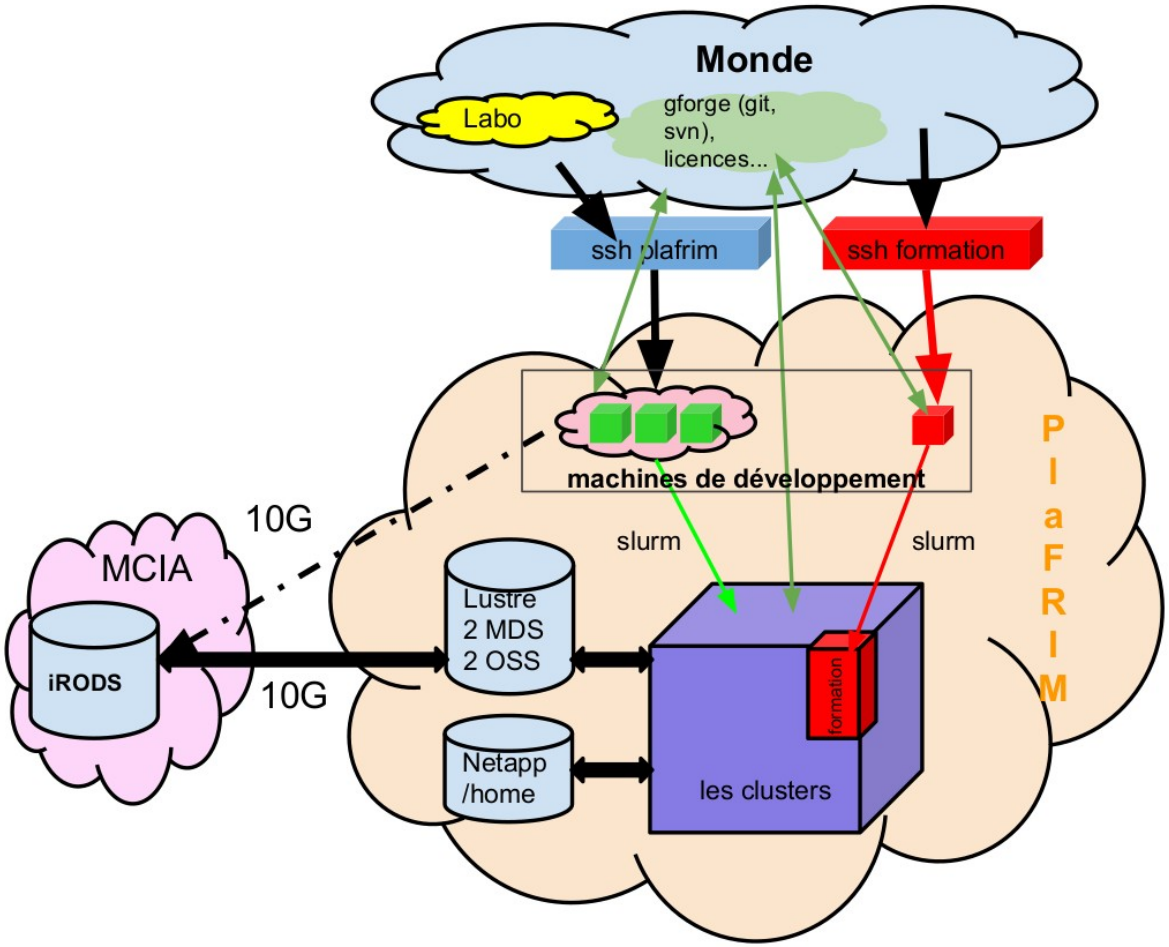
PCIe



Accéder aux ressources

Usage

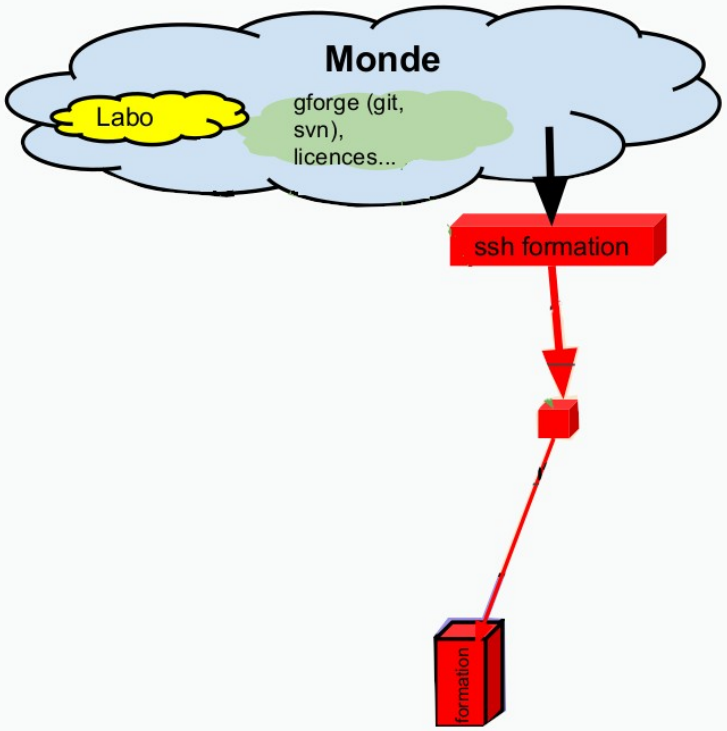
- **Grappe**



Usage

- **Grappe**

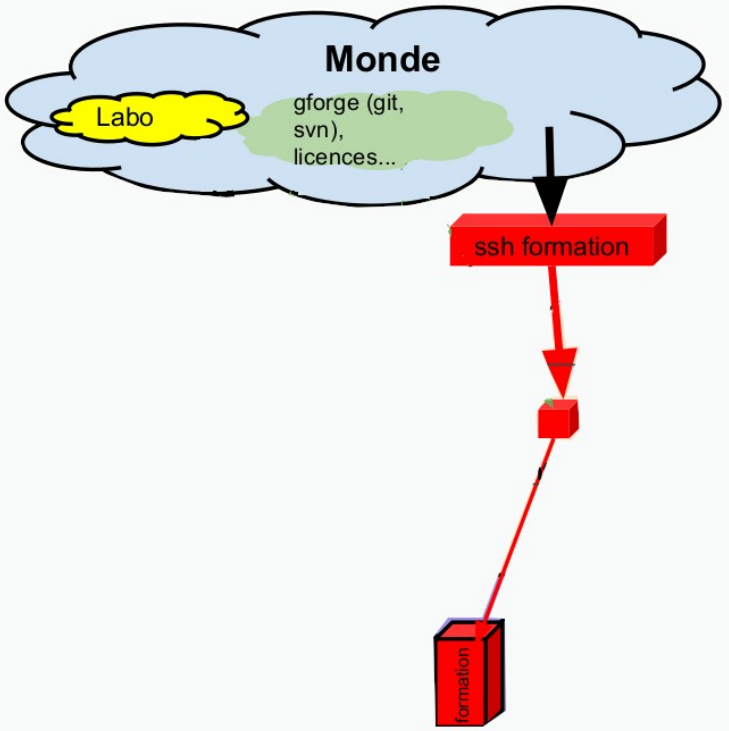
  - Passerelle
    - miriel045



Usage

- **Grappe**

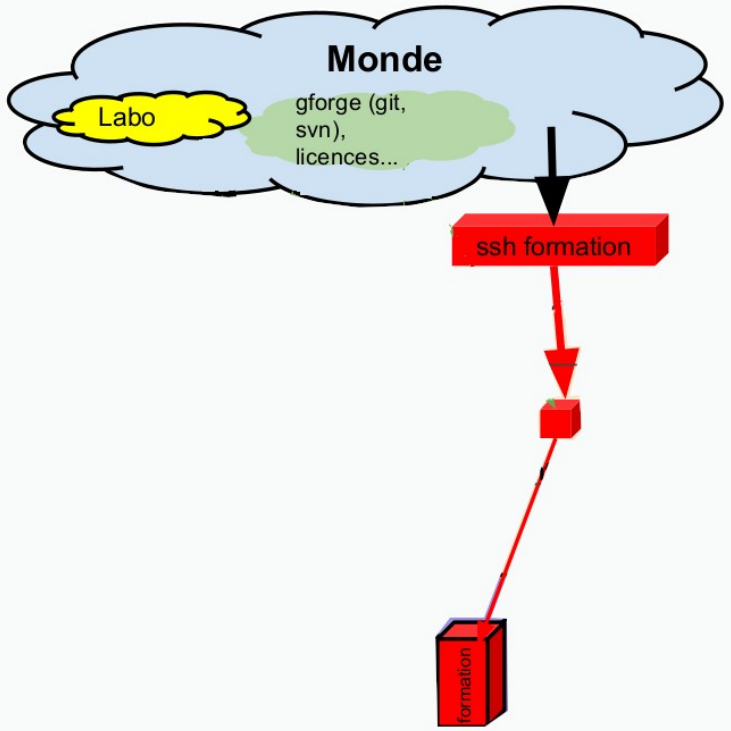
  - Passerelle
    - miriel045
  - Generalistes
    - miriel048-088



Usage

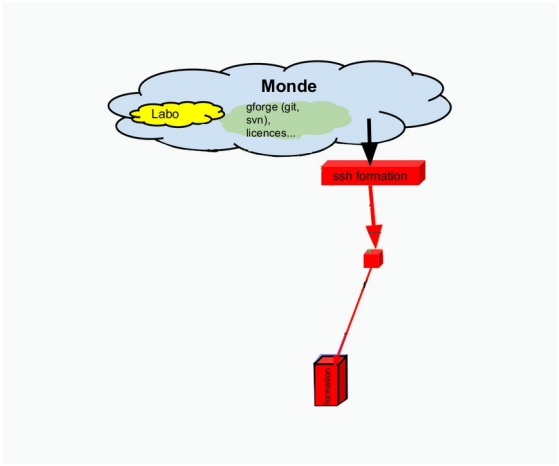
- **Grappe**

  - Passerelle
    - miriel045
  - Generalistes
    - miriel048-088
  - [Accelerateurs
    - sirocco06
    - mistral13]

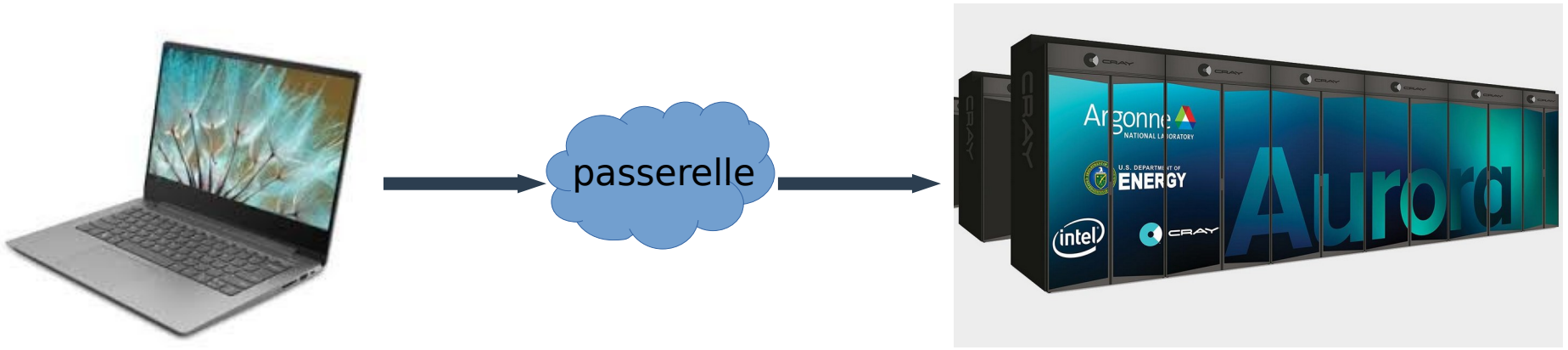


L’Utilisateur lambda

L’Utilisateur lambda



L’Utilisateur lambda

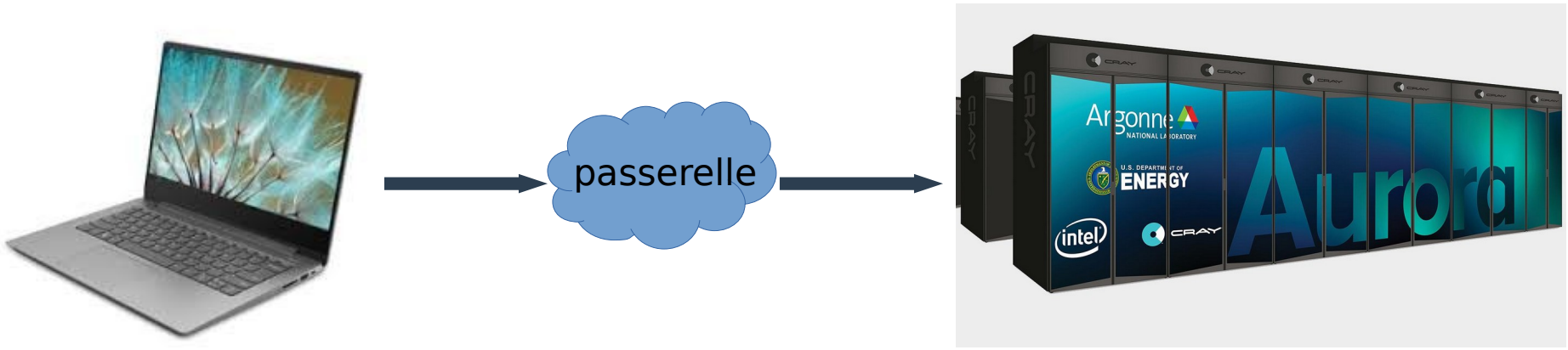


L’Utilisateur lambda

- **Identité de l’utilisateur**

1) Login

2) password



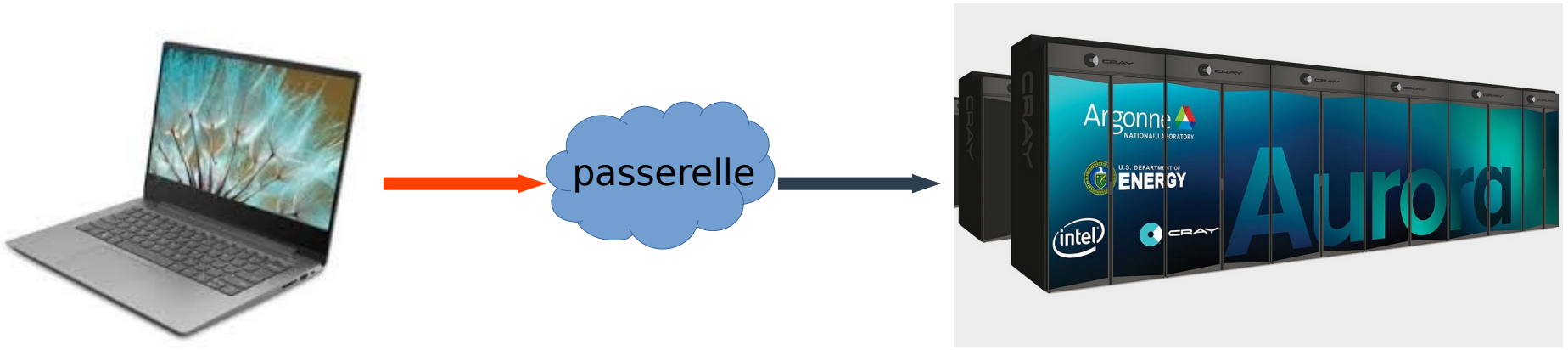
L'Utilisateur lambda

- **Identité de l'utilisateur**

1) Login

ssh -Y <LOGIN>@formation.plafrim.fr

2) password



L'Utilisateur lambda

- **Identité de l'utilisateur**

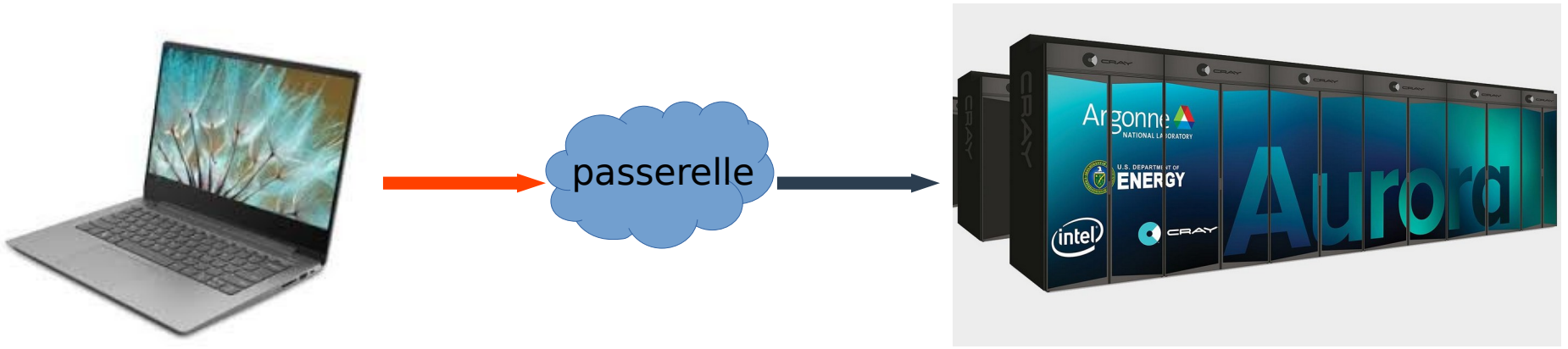
1) Login

ssh -Y <LOGIN>@formation.plafrim.fr

2) password

- **Accès à la plateforme**

1) clef



L'Utilisateur lambda

- **Identité de l'utilisateur**

1) Login

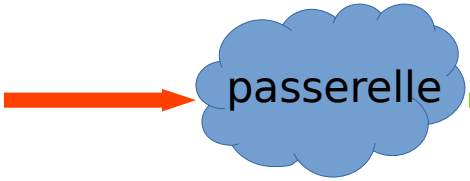
2) password

ssh -Y <LOGIN>@formation.plafrim.fr

- **Accès à la plateforme**

1) clef

ssh -Y plafrim



L'Utilisateur lambda

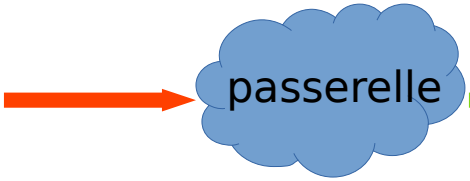
- **Identité de l'utilisateur**

1) Login ssh -Y <LOGIN>@formation.plafrim.fr

2) password

- **Accès à la plateforme**

1) clef ssh -Y plafrim



- **Pour simplifier cette double authentification**

à ajouter dans le fichier .ssh/config

```
Host formation
  ForwardAgent yes
  ForwardX11 yes
  User <LOGIN>
  ProxyCommand ssh -T -q -o "ForwardAgent Yes" -l <LOGIN> formation.plafrim.fr 'ssh-add -t 1 && nc plafrim 22'
```

L'Utilisateur lambda

- **Identité de l'utilisateur**

1) Login

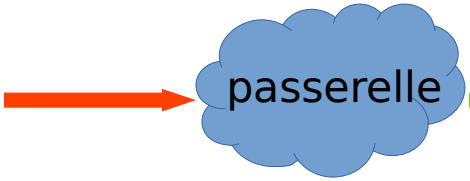
2) password

ssh -Y <LOGIN>@formation.plafrim.fr

- **Accès à la plateforme**

1) clef

ssh -Y plafrim



- **Pour simplifier cette double authentification**

à ajouter dans le fichier .ssh/config

```
Host formation
  ForwardAgent yes
  ForwardX11 yes
  User <LOGIN>
  ProxyCommand ssh -T -q -o "ForwardAgent Yes" -l <LOGIN> formation.plafrim.fr 'ssh-add -t 1 && nc plafrim 22'
```

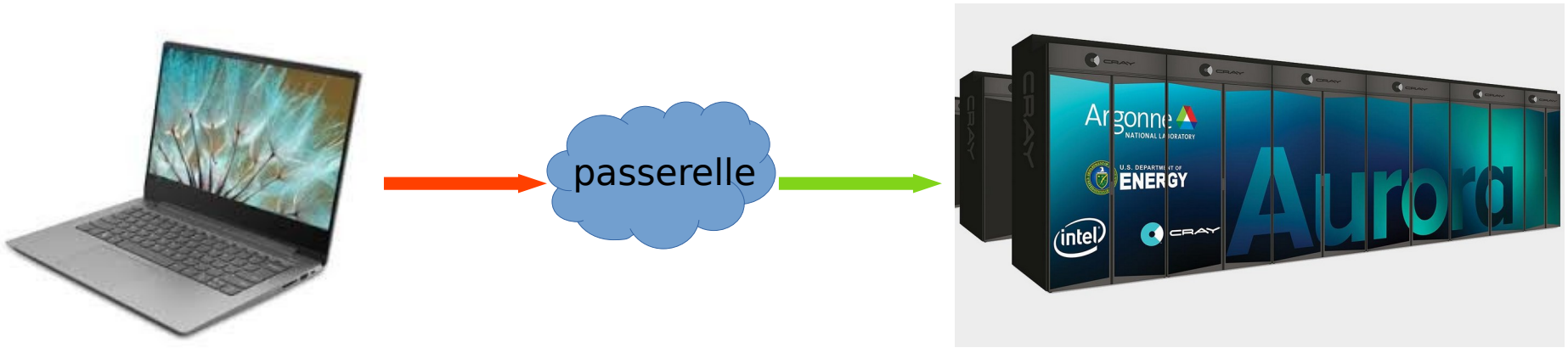
ssh formation

L'Utilisateur lambda

- Opportunités

  . sshfs

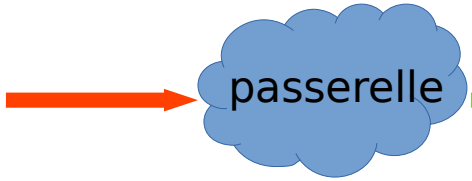
  . application en accès distant



L'Utilisateur lambda

- Opportunités

  . sshfs



. permet de voir localement les donnees distantes

```
localhost ~$ ls mirror/
localhost ~$ sshfs formation:/home/<LOGIN> mirror/
localhost ~$ ls mirror/
ACM-Python-Tutorials-KAUST-2014  intel   project_read_only.dflgadvixe  test    testCuda.py  vec.c
e000                             pong    project_read_only.infoadvixe  test.c  test.s       vec.o
frue.advixeproj                  pong.c  RAPL                          test.cu vec          waveeq
localhost ~$ fusermount -u mirror
```

Gérer son environnement

Environnement

- Objectif:

  Avoir les “bonnes” bibliothèques disponibles dans son propre environnement…
  Sans *polluer l’environnement des autres*.

Environnement

- Gestionnaires de paquet:
  - modules
  - Guix
  - spack
  - easybuild
  - conda

Environnement

- Gestionnaires de paquet:
  - modules
  - Guix
  - spack
  - easybuild
  - conda

Environnement

- Gestionnaires de paquet:
  - modules

    les commandes utiles :

    - **module avail**
    - **module list**
    - **module load / unload <nom du module>**
    - **module switch <module 1> <module 2>**
    - **module show <nom du module>**
    - **….. module purge**

Environnement

- Gestionnaires de paquet:
  - modules

plafrim > module avail

```
------------------------------------ /cm/shared/modules/generic/modulefiles -------------------------------------
mpi/intel/2019.4.243       mpi/openmpi/4.0.1

---------------------------------- /cm/shared/modules/intel/ivybridge/modulefiles ----------------------------------
compiler/gcc/8.2.0          compiler/intel/2019_update4 linalg/mkl/2019_update4
compiler/gcc/9.1.0          intel-tbb/2019_update4
```

Environnement

- Gestionnaires de paquet:
  - modules

```
[lcirrott@miriel045 ~]$ module avail 2>&1 | grep -i "mpi/"
build/bison/3.3                 mpi/openmpi/3.1.4-all
build/cmake/3.15.3              mpi/openmpi/4.0.1
build/cmake/3.21.3              mpi/openmpi/4.0.1-intel
compiler/cuda/10.0              mpi/openmpi/4.0.2
compiler/cuda/10.2              mpi/openmpi/4.0.2-testing
compiler/cuda/11.2              mpi/openmpi/4.0.3
compiler/cuda/11.3              mpi/openmpi/4.0.3-mlx
compiler/cuda/11.4              mpi/openmpi/4.1.1
mpi/intel/2019.4.243            tools/trace/papi/5.0.1
mpi/openmpi/2.0.4               tools/trace/papi/5.0.1-amd
mpi/openmpi/3.1.4
io/hdf5/nompi/1.10.5
[lcirrott@miriel045 ~]$
```

Environnement

- Gestionnaires de paquet:
  - modules

```
plafrim  > module show compiler/gcc/9.1.0
-------------------------------------------------------------------
/cm/shared/modules/intel/ivybridge/modulefiles/compiler/gcc/9.1.0:

module-whatis    adds GNU Cross Compilers to your environment variables
prepend-path     PATH /cm/shared/modules/intel/ivybridge/compiler/gcc/9.1.0/bin
prepend-path     LD_LIBRARY_PATH /cm/shared/modules/intel/ivybridge/compiler/gcc/9.1.0/lib64
prepend-path     LD_LIBRARY_PATH /cm/shared/apps/gcc/dep/gmp/6.1.0/lib
prepend-path     LD_LIBRARY_PATH /cm/shared/apps/gcc/dep/mpc/1.0.3/lib
prepend-path     LD_LIBRARY_PATH /cm/shared/apps/gcc/dep/mpfr/3.1.3/lib
prepend-path     LIBRARY_PATH /cm/shared/modules/intel/ivybridge/compiler/gcc/9.1.0/lib64
prepend-path     LIBRARY_PATH /cm/shared/apps/gcc/dep/gmp/6.1.0/lib
prepend-path     LIBRARY_PATH /cm/shared/apps/gcc/dep/mpc/1.0.3/lib
prepend-path     LIBRARY_PATH /cm/shared/apps/gcc/dep/mpfr/3.1.3/lib
prepend-path     INCLUDE /cm/shared/modules/intel/ivybridge/compiler/gcc/9.1.0/include
...
setenv           CC gcc
setenv           CXX g++
setenv           F77 gfortran
setenv           F90 gfortran
setenv           FC gfortran
setenv           GCC_VER 9.1.0
....
```

Environnement

- Gestionnaires de paquet:
  - modules

```
plafrim  > module show compiler/gcc/9.1.0
-------------------------------------------------------------------
/cm/shared/modules/intel/ivybridge/modulefiles/compiler/gcc/9.1.0:

module-whatis    adds GNU Cross Compilers to your environment variables
prepend-path     PATH /cm/shared/modules/intel/ivybridge/compiler/gcc/9.1.0/bin
prepend-path     LD_LIBRARY_PATH /cm/shared/modules/intel/ivybridge/compiler/gcc/9.1.0/lib64
prepend-path     LD_LIBRARY_PATH /cm/shared/apps/gcc/dep/gmp/6.1.0/lib
prepend-path     LD_LIBRARY_PATH /cm/shared/apps/gcc/dep/mpc/1.0.3/lib
prepend-path     LD_LIBRARY_PATH /cm/shared/apps/gcc/dep/mpfr/3.1.3/lib
prepend-path     LIBRARY_PATH /cm/shared/modules/intel/ivybridge/compiler/gcc/9.1.0/lib64
prepend-path     LIBRARY_PATH /cm/shared/apps/gcc/dep/gmp/6.1.0/lib
prepend-path     LIBRARY_PATH /cm/shared/apps/gcc/dep/mpc/1.0.3/lib
prepend-path     LIBRARY_PATH /cm/shared/apps/gcc/dep/mpfr/3.1.3/lib
prepend-path     INCLUDE /cm/shared/modules/intel/ivybridge/compiler/gcc/9.1.0/include
...
setenv           CC gcc
setenv           CXX g++
setenv           F77 gfortran
setenv           F90 gfortran
setenv           FC gfortran
setenv           GCC_VER 9.1.0
....
```

Environnement

- Gestionnaires de paquet:
  - modules

```
plafrim  > module show compiler/gcc/9.1.0
-------------------------------------------------------------------
/cm/shared/modules/intel/ivybridge/modulefiles/compiler/gcc/9.1.0:

module-whatis    adds GNU Cross Compilers to your environment variables
prepend-path     PATH /cm/shared/modules/intel/ivybridge/compiler/gcc/9.1.0/bin
prepend-path     LD_LIBRARY_PATH /cm/shared/modules/intel/ivybridge/compiler/gcc/9.1.0/lib64
prepend-path     LD_LIBRARY_PATH /cm/shared/apps/gcc/dep/gmp/6.1.0/lib
prepend-path     LD_LIBRARY_PATH /cm/shared/apps/gcc/dep/mpc/1.0.3/lib
prepend-path     LD_LIBRARY_PATH /cm/shared/apps/gcc/dep/mpfr/3.1.3/lib
prepend-path     LIBRARY_PATH /cm/shared/modules/intel/ivybridge/compiler/gcc/9.1.0/lib64
prepend-path     LIBRARY_PATH /cm/shared/apps/gcc/dep/gmp/6.1.0/lib
prepend-path     LIBRARY_PATH /cm/shared/apps/gcc/dep/mpc/1.0.3/lib
prepend-path     LIBRARY_PATH /cm/shared/apps/gcc/dep/mpfr/3.1.3/lib
prepend-path     INCLUDE /cm/shared/modules/intel/ivybridge/compiler/gcc/9.1.0/include
...
setenv           CC gcc
setenv           CXX g++
setenv           F77 gfortran
setenv           F90 gfortran
setenv           FC gfortran
setenv           GCC_VER 9.1.0
....
```

Environnement

- Gestionnaires de paquet:
  - modules
  - Guix
  - spack
  - easybuild
  - conda

Environnement

- Gestionnaires de paquet:
  - guix

plafrim > guix pull

Les canaux guix-hpc et guix-hpc-non-free

distribution de systeme d'exploitation autonome pour plusieurs architectures
- i686,
- x86_64,
- ARMv7,
- Aarch64

≈15,000 paquets

Prédisposition à la *reproductibilité* des environnements logiciels.

https://guix.gnu.org/

Environnement

- Gestionnaires de paquet:
  - guix

concepts fondateurs de guix (nixos)
- Store : /gnu/store
- Profile. /run/current-system/profile/bin
- Notion de génération (system / package / profile)

Ce que cela permet d'abord
- plusieurs générations de configurations de votre système (y compris tout l'historique de vos paquets)
- Faire du rollback
- Faire du repositionnement de paquet
- Et plein d'autres choses ...

Environnement

- Gestionnaires de paquet:
  - guix

Ce que cela permet d'abord de:

- Chercher une "recette" de compilation d'un "paquet " logiciel (ici, Vim):
  > guix search vim
- Compiler (construire) le paquet sans l'installer dans l'espace utilisateur:
  > guix build vim
- Construire le paquet et l'installer dans l'espace utilisateur:
  > guix install vim

Environnement

- Gestionnaires de paquet:
  - guix

- Gérer plusieurs générations de configurations de votre système (y compris tout l'historique de vos paquets):
  > guix package –list-generations
- Faire du rollback vers une génération précedente:
  > guix package –roll-back
- Faire du repositionnement de paquet
  > guix pack --relocatable “mon paquet”
- Créer des “pack”
  > guix pack -f docker bash guile emacs geiser
  > guix pack -f squashfs bash guile emacs geiser

Environnement

- Gestionnaires de paquet:
  - guix

- Installer un paquet dans un environnement dédié:
  > guix shell vim
- Protéger l'environnement des autres variables:
  > guix shell --pure vim
- Isoler l'environnement du reste du système:
  > guix shell --container vim
- Et plein d'autres choses...

Partager les ressources

Usage

I. On accède aux ressources

II. On gère l'environnement qui nous intéresse pour nos expériences

III. On lance nos expériences *sur des ressources partagées*

Usage

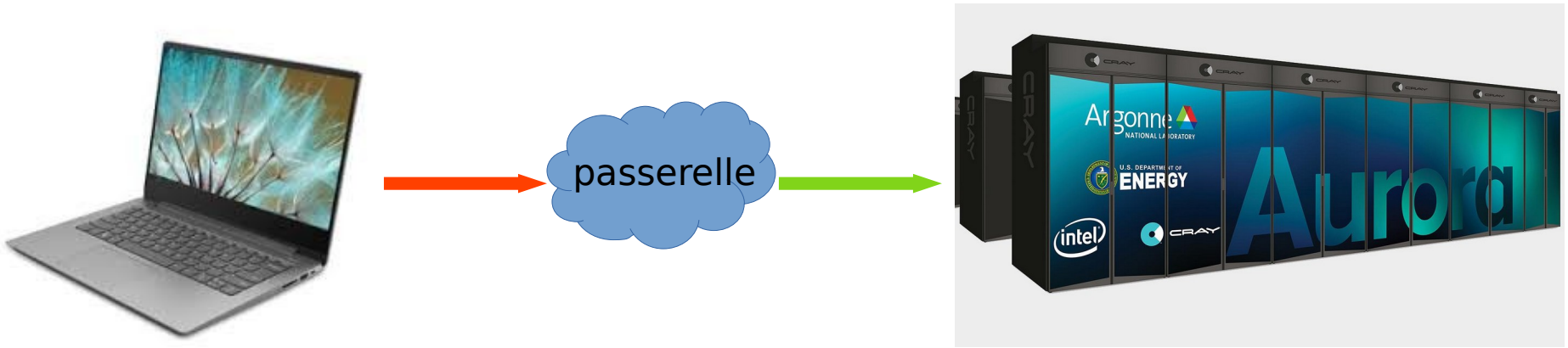
I. On accède aux ressources

II. On gère l'environnement qui nous intéresse pour nos expériences

III. On lance nos expériences *sur des ressources partagées*

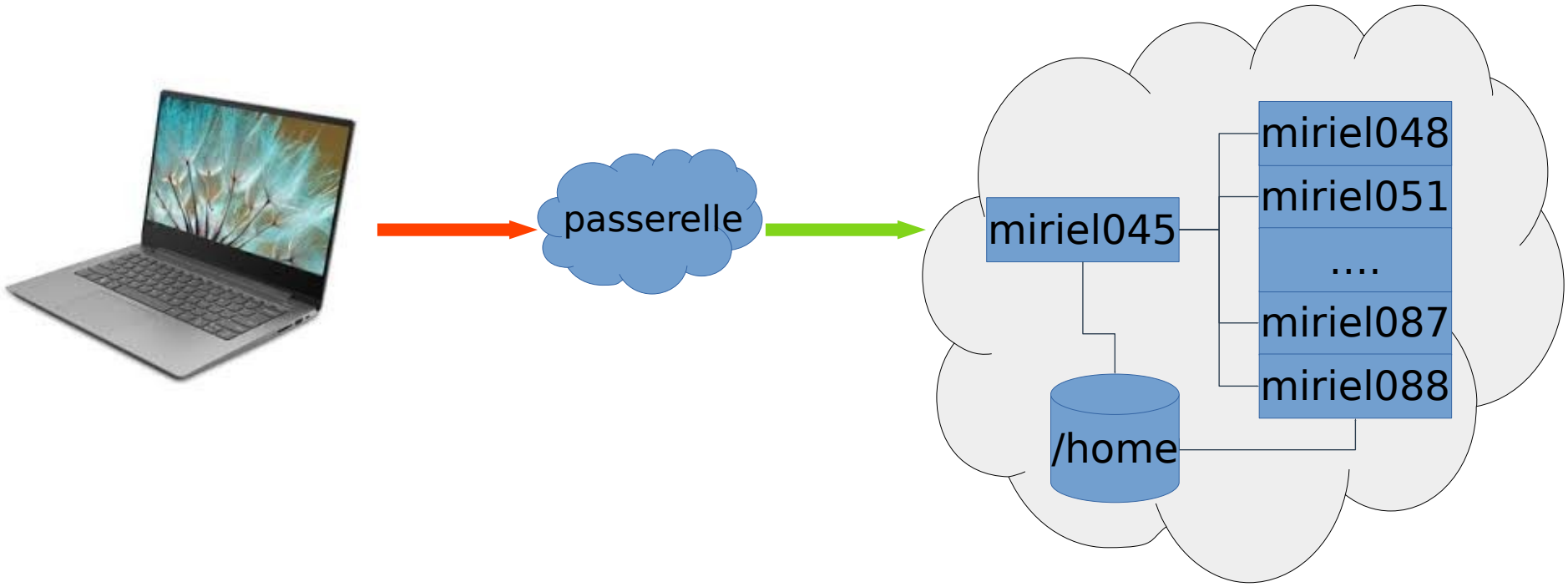
Usage

Un ordonnanceur de tâches (ex. *Slurm*) gère le partage des ressources



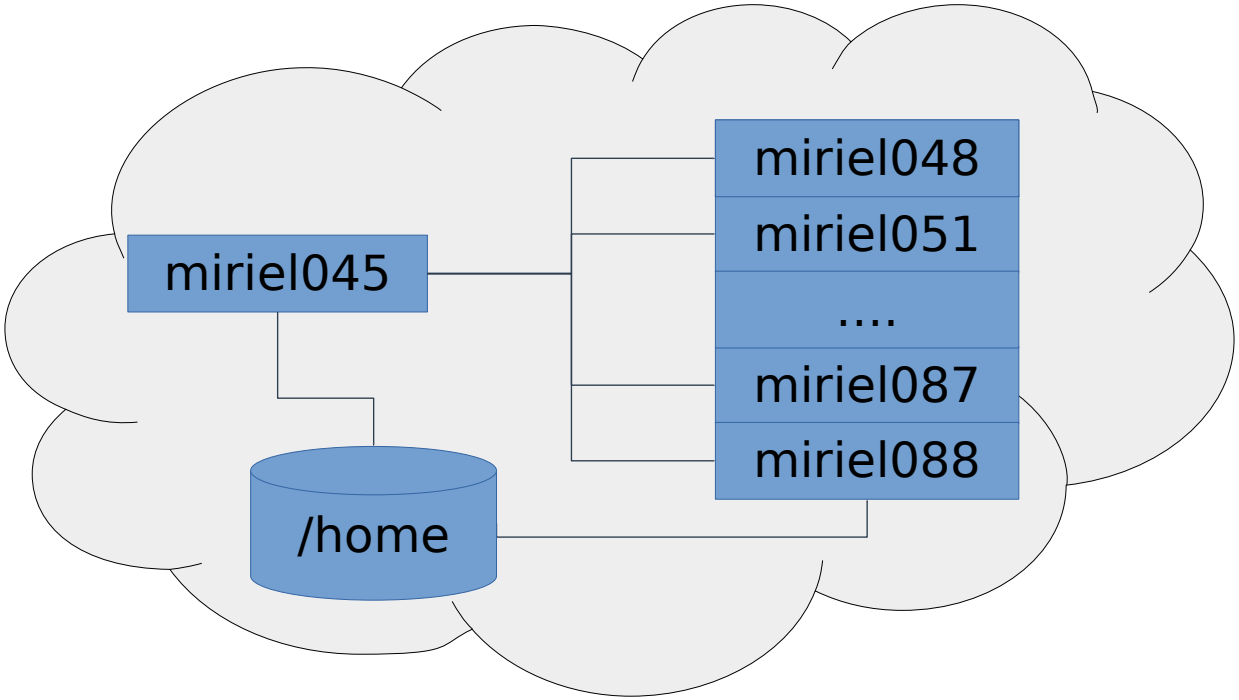
Usage

Un ordonnanceur de tâches (ex. *Slurm*) gère le partage des ressources



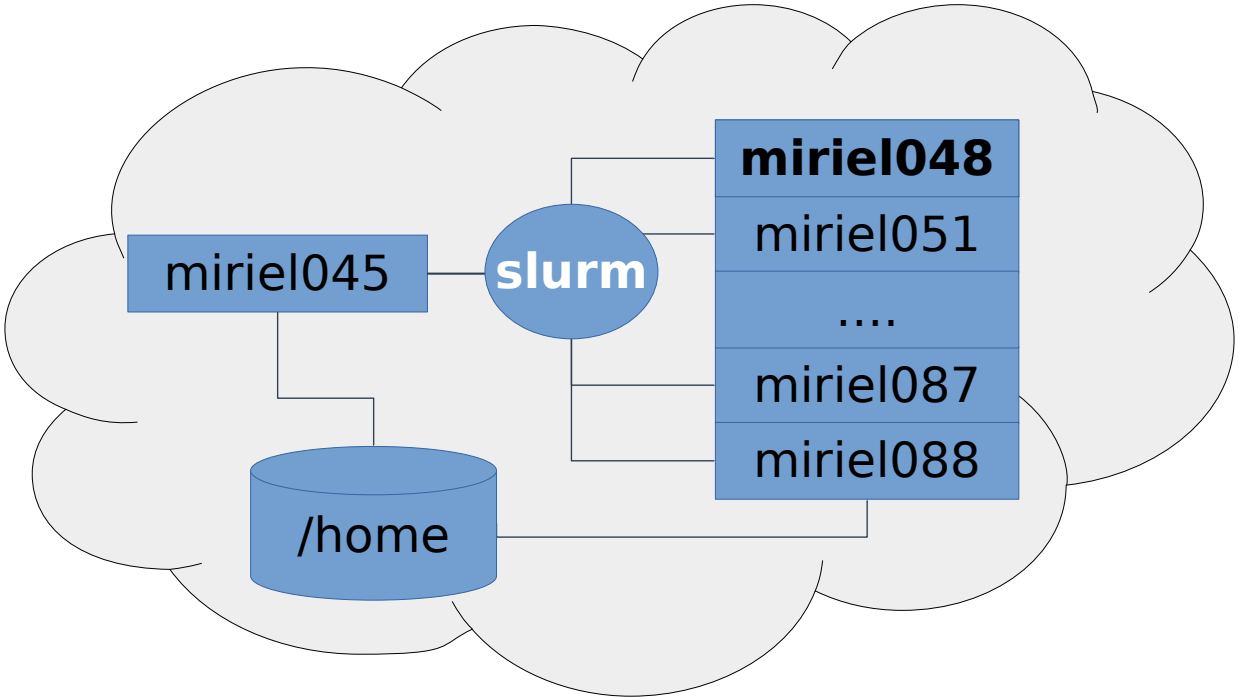
Usage

Un ordonnanceur de tâches (ex. *Slurm*) gère le partage des ressources



Usage

Un ordonnanceur de tâches (ex. *Slurm*) gère le partage des ressources



Usage

- Slurm

  les commandes utiles :

  - **sinfo**
  - **salloc -p <partition> -N <number of nodes>**
  - **sbatch <batch script>**
  - **squeue -u <login name>**
  - **scontrol show jobid <job id>**
  - **scancel <job id>**

Usage

- Slurm

  salloc – pour faire de l’interactif

  - **salloc -p <partition> -N <number of nodes>**
  - **ouvre un pseudo terminal**
  - **ssh possible sur les noeuds alloués**

Usage

- Slurm

  sbatch – pour faire de l'interactif

  - **sbatch <batch script>**

    #!/bin/bash

    #SBATCH –p miriel

    #SBATCH --job-name=myjob

    #SBATCH --mem=4G

    #SBATCH –-time=01:30:00

    #SBATCH -o myjob.slurm.out

    #SBATCH -e myjob.slurm.err

    module load <mon environnement>

    <commandes de lancement de travaux>

Usage

- Slurm

  sbatch – pour faire de l’interactif

  - **sbatch <batch script>**

    un petit test rapide avec le code hello

hello.c - code de test

I. On accède aux ressources

> ssh formation

II. On gère l'environnement

III. On lance nos expériences

hello.c - code de test

I. On accède aux ressources

II. On gère l'environnement

> module load compiler/gcc

> module list

> echo $PATH

> which gcc

III. On lance nos expériences

hello.c - code de test

I. On accède aux ressources

II. On gère l'environnement

> module load mpi/openmpi/3.1.4-all

> module list

> which mpicc

> mpicc hello.c -o program_hello

III. On lance nos expériences

hello.c – code de test

I. On accède aux ressources

II. On gère l'environnement

III. On lance nos expériences

> salloc -C miriel
et
> mpirun -np 4 ./program_hello

ou

> sbatch job_hello.slm

hello.c – code de test

I. On accède aux ressources

II. On gère l'environnement

III. On lance nos expériences

```
#!/bin/bash
#SBATCH –C miriel
#SBATCH --job-name=myjob
#SBATCH --mem=4G
#SBATCH –-time=00:10:00
#SBATCH --output myjob.slurm.out
#SBATCH --error myjob.slurm.err
module load compiler/gcc/9.3.0
module load mpi/openmpi/3.1.4-all
mpirun ./program_hello
```