Introduction to Leonardo

Author: CINECA Production Team
superc@cineca.it

Presenter: Gabriella Bettonte
g.bettonte@cineca.it

20th April 2024

2023 OVERVIEW

HPC SYSTEMS

CINECA

CINECA enables world-class scientific research by operating and supporting leading-edge supercomputing technologies and by managing a state-of-the-art and effective environment for the different scientific communities.

LEONARDO | 2023

4992 nodes
Booster Module:
32 core per node
4 GPU NVIDIA Ampere custom
Data Centric Module:
56 cores per node
SOON IN PRODUCTION
110 PB Storage
250 PFlops

MARCONI | 2017

3188 nodes
48 cores per node
612 TB RAM
10 PFlops

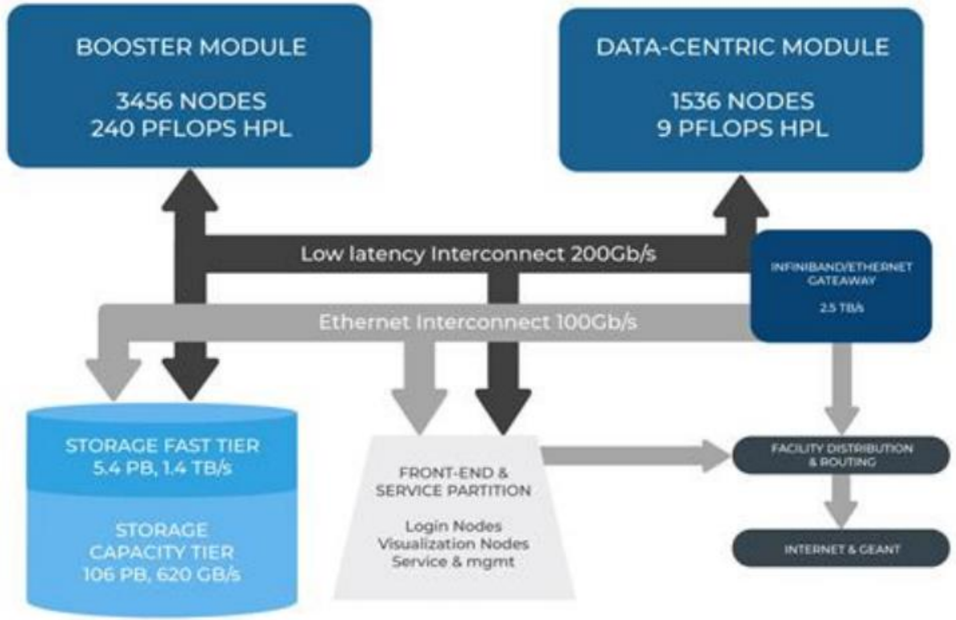
DGX | 2021

3 nodes
128 cores per node
8 GPU NVIDIA A100 per node
100 TB Storage
15 PFlops

GALILEO100 | 2021

564 nodes
48 cores per node
2 GPU NVIDIA V100 per node
~22 PB Storage
2 PFlops

Leonardo infrastructure and login nodes



Atos BullSequana X430-E6

- ➢ Processors: **2 x CPU Intel Whitley ICP06, 32 cores Intel Ice Lake, 2.4 GHz**
- ➢ Hyper Threading is enabled!
- ➢ RAM: 512 (16x32) GB RAM DDR4 3200MHz
- ➢ 6TB disk in RAID1 configuration
- ➢ **NO GPUs**

Booster (GPU) module

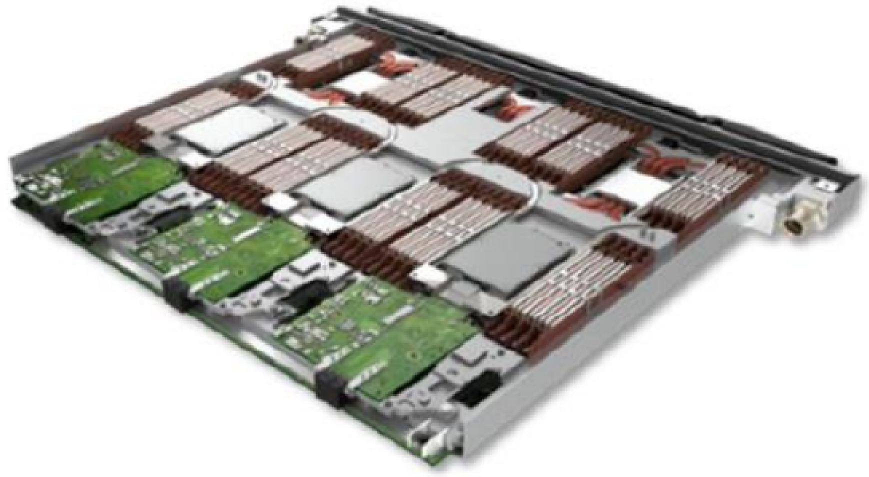
Atos BullSequana X2135 “Da Vinci” blade

- 3456 nodes (1 node per blase)
- Processors: **1 x CPU Intel Xeon 8358,**
  32 cores Intel Ice Lake, 2.6 GHz **(ONE SOCKET!)**
- RAM: 512 (8 x 64) GB DDR4 3200 MHz
- Accelerators: **4 x NVidia custom Ampere GPU A100 SXM4 64 GB,**
  NVLink 3.0
- Internal network: NVIDIA Mellanox HDR DragonFly+ 200Gb/s
- **DISKLESS!!!**
- Shared (via infiniband) storage space: 106 PB Capacity tier storage
  + 5.4 PB Fast tier storage

Peak performance per node: about 89,4 TFlops

Peak performance: about 309 PFlops

Data Centric & General Purpose (CPU) module



Peak performance per node: about 8.46 TFlops
Peak performance: about 13 PFlops

BullSequana X2140 three-node CPU Blade

- ➢ 1536 nodes (512 blades)
- ➢ Processors: 2 x CPU Intel Xeon 8480+, 56 cores Intel Sapphire Rapids, 2.0 GHz
- ➢ RAM: 256 (16x16) GB DDR5 4800MHz
  512 (16 x 32) GB DDR5 4800 MHz
- ➢ Infiniband: 1 x NVIDIA HDR cards 100 Gbps via PCIe Gen 5
- ➢ Disk: 1 x M.2 SSD 3,84 TB
- ➢ It will be in production soon.

Visualization (viz) module

16 Atos BullSequana X450-E6

- Processors: **2 x CPU Intel Whitley ICP06, 32 cores Intel Ice Lake, 2.4 GHz**
- RAM: 512 GB (16x32) GB 3200MHz DDR4
- 2x **Nvidia Quadro RTX8000 48GB** PCIe
- 1x 6.4TB NVMe disks
- 1 x NIC HDR100 via Connect-X 6 card
- 2x Nic Connect-X 5 Ethernet 50 Gb/s
- **Not yet in production**: these nodes will be probably dedicated to the **remote visualization** or to the **interactive computing service**

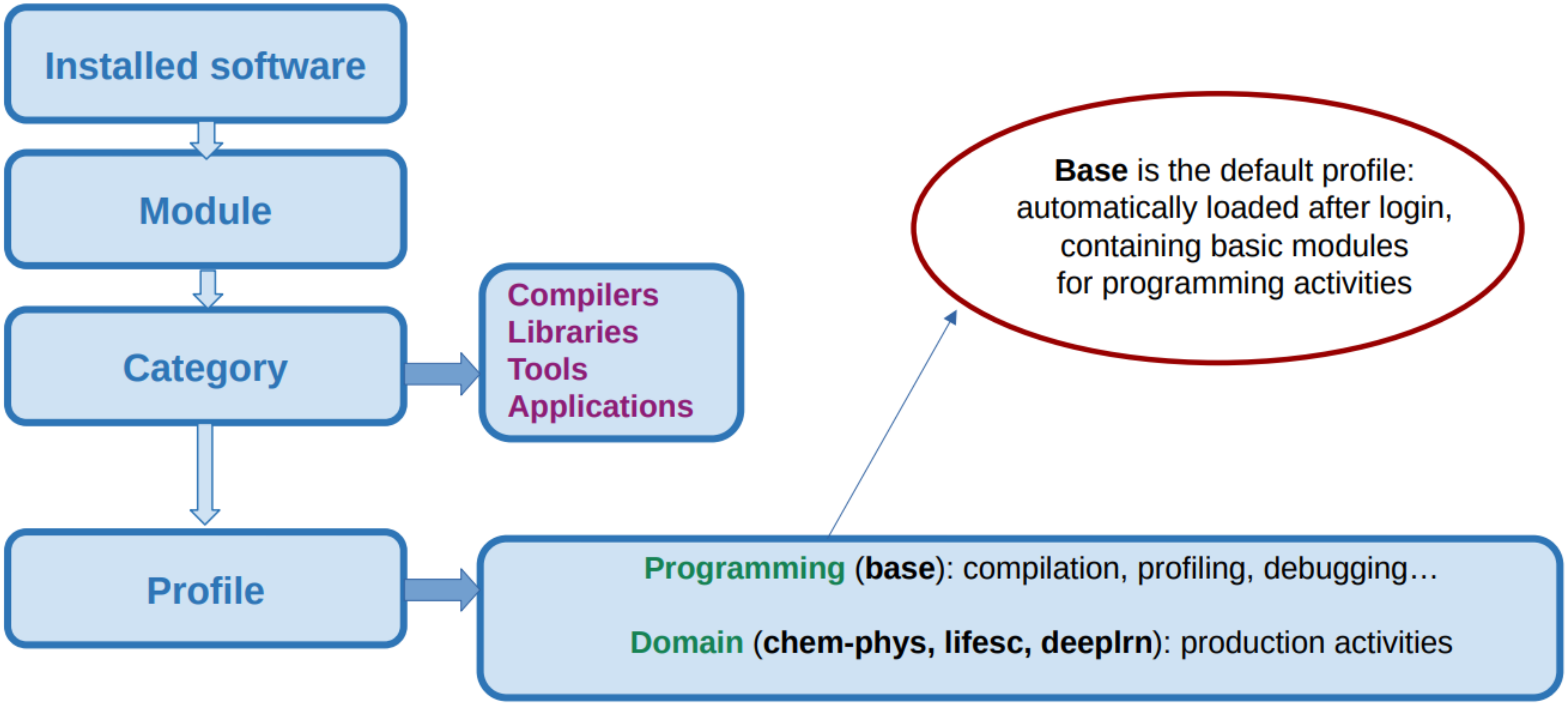
Access to Leonardo

ssh username@login.leonardo.cineca.it

For g100: ssh username@login.g100.cineca.it

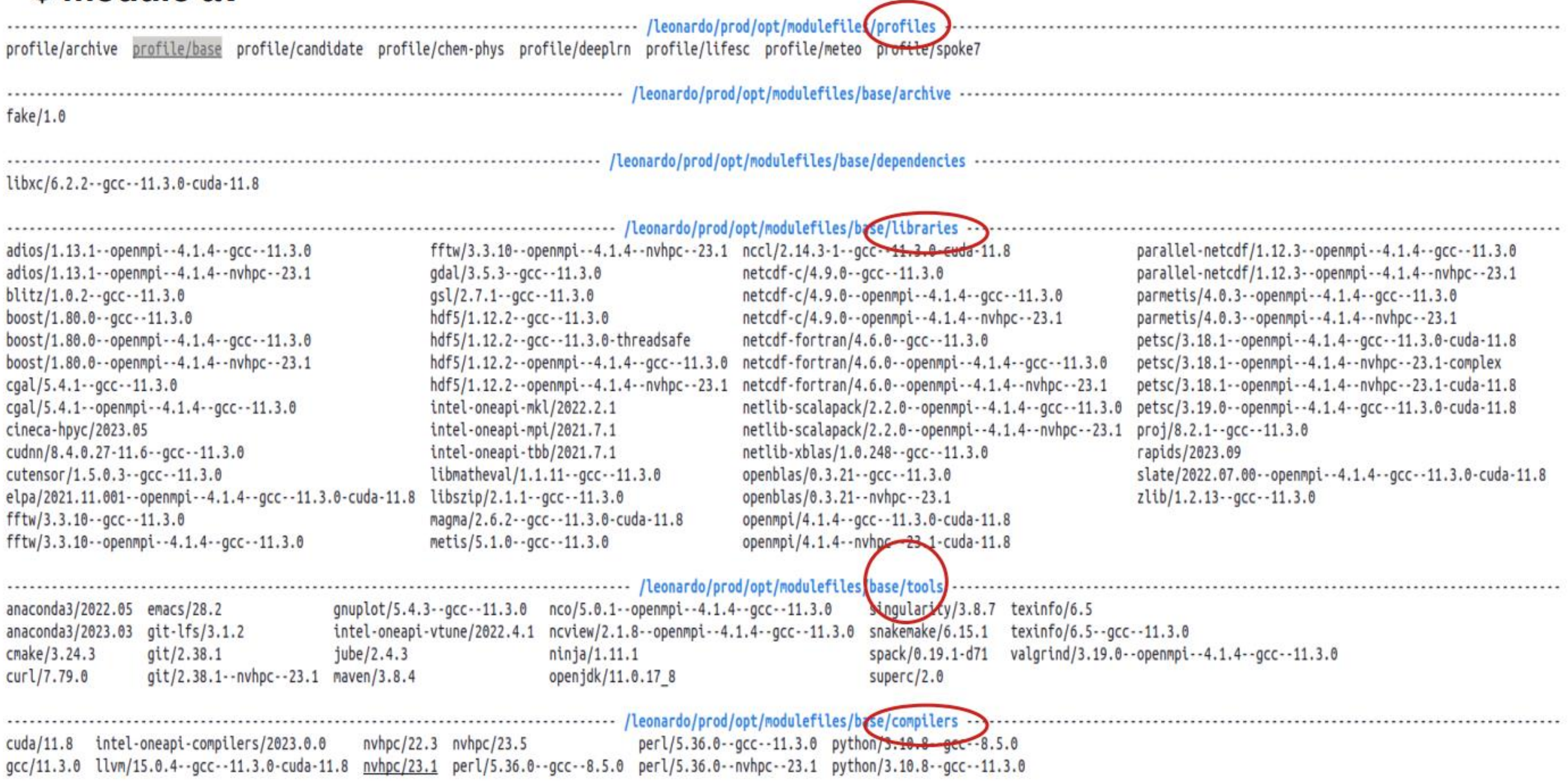
Module environment

Any available software is offered on Leonardo in a **module environment**.
The modules are organized in functional **categories** and collected in different **profiles**.


Installed software
Module
Category
Compilers
Libraries
Tools
Applications
Profile
Base is the default profile: automatically loaded after login, containing basic modules for programming activities
Programming (**base**): compilation, profiling, debugging…
Domain (**chem-phys, lifesc, deeplrn**): production activities

Module environment

$ module av



```
--------------------------------------------------------------------- /leonardo/prod/opt/modulefiles/profiles ---------------------------------------------------------------------
profile/archive  profile/base  profile/candidate  profile/chem-phys  profile/deeplrn  profile/lifesc  profile/meteo  profile/spoke7

------------------------------------------------------------------ /leonardo/prod/opt/modulefiles/base/archive ------------------------------------------------------------------
fake/1.0

---------------------------------------------------------------- /leonardo/prod/opt/modulefiles/base/dependencies ----------------------------------------------------------------
libxc/6.2.2--gcc--11.3.0-cuda-11.8

----------------------------------------------------------------- /leonardo/prod/opt/modulefiles/base/libraries -----------------------------------------------------------------
adios/1.13.1--openmpi--4.1.4--gcc--11.3.0               fftw/3.3.10--openmpi--4.1.4--nvhpc--23.1  nccl/2.14.3-1--gcc--11.3.0-cuda-11.8                   parallel-netcdf/1.12.3--openmpi--4.1.4--gcc--11.3.0
adios/1.13.1--openmpi--4.1.4--nvhpc--23.1               gdal/3.5.3--gcc--11.3.0                   netcdf-c/4.9.0--gcc--11.3.0                            parallel-netcdf/1.12.3--openmpi--4.1.4--nvhpc--23.1
blitz/1.0.2--gcc--11.3.0                                gsl/2.7.1--gcc--11.3.0                    netcdf-c/4.9.0--openmpi--4.1.4--gcc--11.3.0            parmetis/4.0.3--openmpi--4.1.4--gcc--11.3.0
boost/1.80.0--gcc--11.3.0                               hdf5/1.12.2--gcc--11.3.0                  netcdf-c/4.9.0--openmpi--4.1.4--nvhpc--23.1            parmetis/4.0.3--openmpi--4.1.4--nvhpc--23.1
boost/1.80.0--openmpi--4.1.4--gcc--11.3.0               hdf5/1.12.2--gcc--11.3.0-threadsafe       netcdf-fortran/4.6.0--gcc--11.3.0                      petsc/3.18.1--openmpi--4.1.4--gcc--11.3.0-cuda-11.8
boost/1.80.0--openmpi--4.1.4--nvhpc--23.1               hdf5/1.12.2--openmpi--4.1.4--gcc--11.3.0  netcdf-fortran/4.6.0--openmpi--4.1.4--gcc--11.3.0      petsc/3.18.1--openmpi--4.1.4--nvhpc--23.1-complex
cgal/5.4.1--gcc--11.3.0                                 hdf5/1.12.2--openmpi--4.1.4--nvhpc--23.1  netcdf-fortran/4.6.0--openmpi--4.1.4--nvhpc--23.1      petsc/3.18.1--openmpi--4.1.4--nvhpc--23.1-cuda-11.8
cgal/5.4.1--openmpi--4.1.4--gcc--11.3.0                 intel-oneapi-mkl/2022.2.1                 netlib-scalapack/2.2.0--openmpi--4.1.4--gcc--11.3.0    petsc/3.19.0--openmpi--4.1.4--gcc--11.3.0-cuda-11.8
cineca-hpyc/2023.05                                     intel-oneapi-mpi/2021.7.1                 netlib-scalapack/2.2.0--openmpi--4.1.4--nvhpc--23.1    proj/8.2.1--gcc--11.3.0
cudnn/8.4.0.27-11.6--gcc--11.3.0                        intel-oneapi-tbb/2021.7.1                 netlib-xblas/1.0.248--gcc--11.3.0                      rapids/2023.09
cutensor/1.5.0.3--gcc--11.3.0                           libmatheval/1.1.11--gcc--11.3.0           openblas/0.3.21--gcc--11.3.0                           slate/2022.07.00--openmpi--4.1.4--gcc--11.3.0-cuda-11.8
elpa/2021.11.001--openmpi--4.1.4--gcc--11.3.0-cuda-11.8 libszip/2.1.1--gcc--11.3.0                openblas/0.3.21--nvhpc--23.1                           zlib/1.2.13--gcc--11.3.0
fftw/3.3.10--gcc--11.3.0                                magma/2.6.2--gcc--11.3.0-cuda-11.8        openmpi/4.1.4--gcc--11.3.0-cuda-11.8
fftw/3.3.10--openmpi--4.1.4--gcc--11.3.0                metis/5.1.0--gcc--11.3.0                  openmpi/4.1.4--nvhpc--23.1-cuda-11.8

------------------------------------------------------------------- /leonardo/prod/opt/modulefiles/base/tools -------------------------------------------------------------------
anaconda3/2022.05  emacs/28.2                gnuplot/5.4.3--gcc--11.3.0  nco/5.0.1--openmpi--4.1.4--gcc--11.3.0     singularity/3.8.7  texinfo/6.5
anaconda3/2023.03  git-lfs/3.1.2             intel-oneapi-vtune/2022.4.1 ncview/2.1.8--openmpi--4.1.4--gcc--11.3.0  snakemake/6.15.1   texinfo/6.5--gcc--11.3.0
cmake/3.24.3       git/2.38.1                jube/2.4.3                  ninja/1.11.1                               spack/0.19.1-d71   valgrind/3.19.0--openmpi--4.1.4--gcc--11.3.0
curl/7.79.0        git/2.38.1--nvhpc--23.1   maven/3.8.4                 openjdk/11.0.17_8                          superc/2.0

----------------------------------------------------------------- /leonardo/prod/opt/modulefiles/base/compilers -----------------------------------------------------------------
cuda/11.8   intel-oneapi-compilers/2023.0.0       nvhpc/22.3  nvhpc/23.5              perl/5.36.0--gcc--11.3.0    python/3.10.8--gcc--8.5.0
gcc/11.3.0  llvm/15.0.4--gcc--11.3.0-cuda-11.8    nvhpc/23.1  perl/5.36.0--gcc--8.5.0 perl/5.36.0--nvhpc--23.1    python/3.10.8--gcc--11.3.0
```

Module environment

Profile base $\longrightarrow$ Automatically loaded

How to **load** additional profiles/modules?

```
$ module load profile/chem-phys
```

$\longrightarrow$ You can use tab for auto completion

Which modules did **I already load**?

```
$ module list
Currently Loaded Modulefiles:
 1) profile/base   2) profile/chem-phys
```

profile/base cannot be unloaded

How to **unload** a profile/module?

Specific profile/module

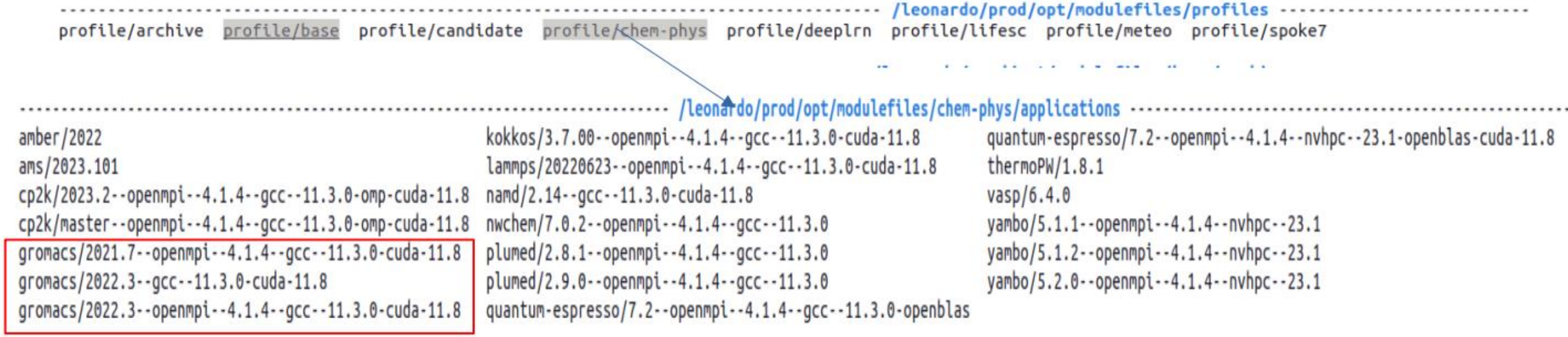
```
$ module unload profile/chem-phys
```

Unload all of them

```
$ module purge
```

Module environment

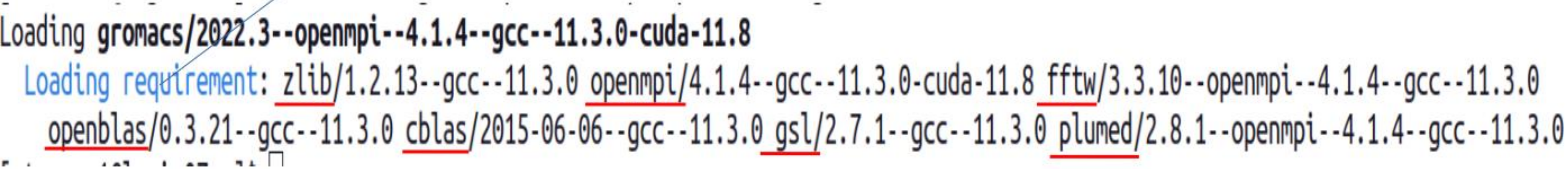
profile/base + profile/chem-phys $\longrightarrow$ $ module avail



```
------------------------------------------------------------------ /leonardo/prod/opt/modulefiles/profiles ---------------------------
profile/archive  profile/base  profile/candidate  profile/chem-phys  profile/deeplrn  profile/lifesc  profile/meteo  profile/spoke7

--------------------------------------------------------------- /leonardo/prod/opt/modulefiles/chem-phys/applications -----------------------------------------------------------------
amber/2022                                          kokkos/3.7.00--openmpi--4.1.4--gcc--11.3.0-cuda-11.8     quantum-espresso/7.2--openmpi--4.1.4--nvhpc--23.1-openblas-cuda-11.8
ams/2023.101                                        lammps/20220623--openmpi--4.1.4--gcc--11.3.0-cuda-11.8   thermoPW/1.8.1
cp2k/2023.2--openmpi--4.1.4--gcc--11.3.0-omp-cuda-11.8   namd/2.14--gcc--11.3.0-cuda-11.8                    vasp/6.4.0
cp2k/master--openmpi--4.1.4--gcc--11.3.0-omp-cuda-11.8   nwchem/7.0.2--openmpi--4.1.4--gcc--11.3.0           yambo/5.1.1--openmpi--4.1.4--nvhpc--23.1
gromacs/2021.7--openmpi--4.1.4--gcc--11.3.0-cuda-11.8    plumed/2.8.1--openmpi--4.1.4--gcc--11.3.0           yambo/5.1.2--openmpi--4.1.4--nvhpc--23.1
gromacs/2022.3--gcc--11.3.0-cuda-11.8                    plumed/2.9.0--openmpi--4.1.4--gcc--11.3.0           yambo/5.2.0--openmpi--4.1.4--nvhpc--23.1
gromacs/2022.3--openmpi--4.1.4--gcc--11.3.0-cuda-11.8    quantum-espresso/7.2--openmpi--4.1.4--gcc--11.3.0-openblas
```

$ module load gromacs/2022.3--openmpi--4.1.4--gcc--11.3.0-cuda-11.8

Autoload is not necessary → Different from other Cineca clusters



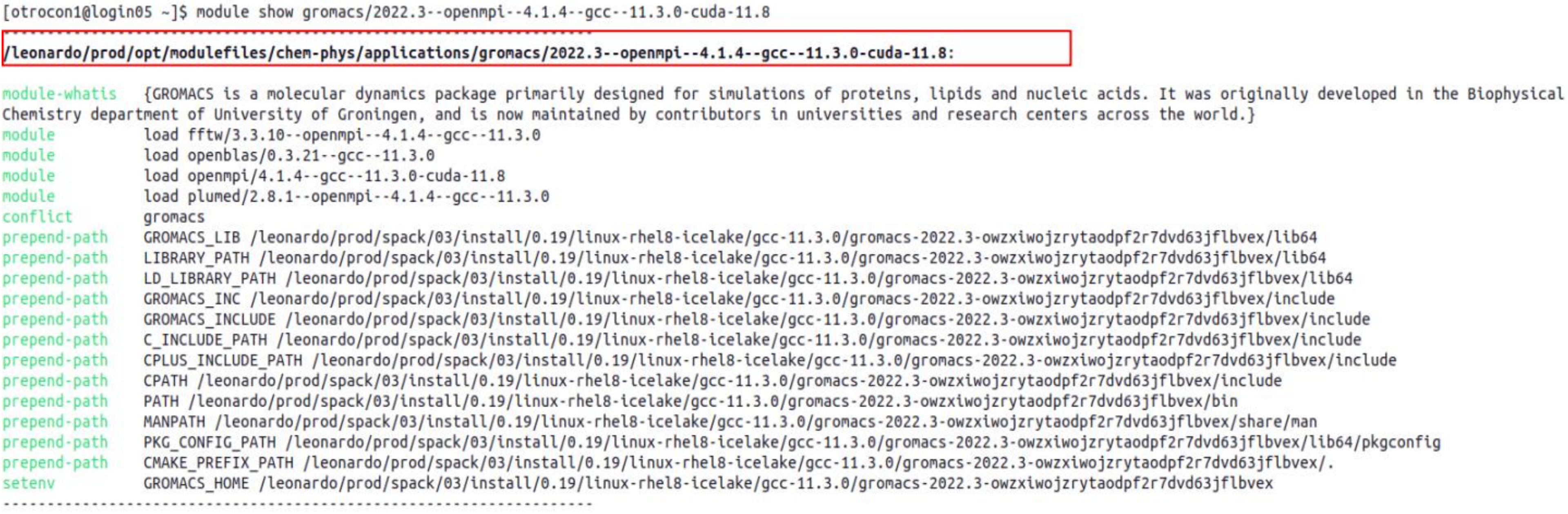
```
Loading gromacs/2022.3--openmpi--4.1.4--gcc--11.3.0-cuda-11.8
  Loading requirement: zlib/1.2.13--gcc--11.3.0 openmpi/4.1.4--gcc--11.3.0-cuda-11.8 fftw/3.3.10--openmpi--4.1.4--gcc--11.3.0
    openblas/0.3.21--gcc--11.3.0 cblas/2015-06-06--gcc--11.3.0 gsl/2.7.1--gcc--11.3.0 plumed/2.8.1--openmpi--4.1.4--gcc--11.3.0
```

Module environment

Loading a module $\longrightarrow$ Define or modify the environment variables, allowing to use the executable or libraries

```
$ module load gromacs/2022.3--openmpi--4.1.4--gcc--11.3.0-cuda-11.8
```

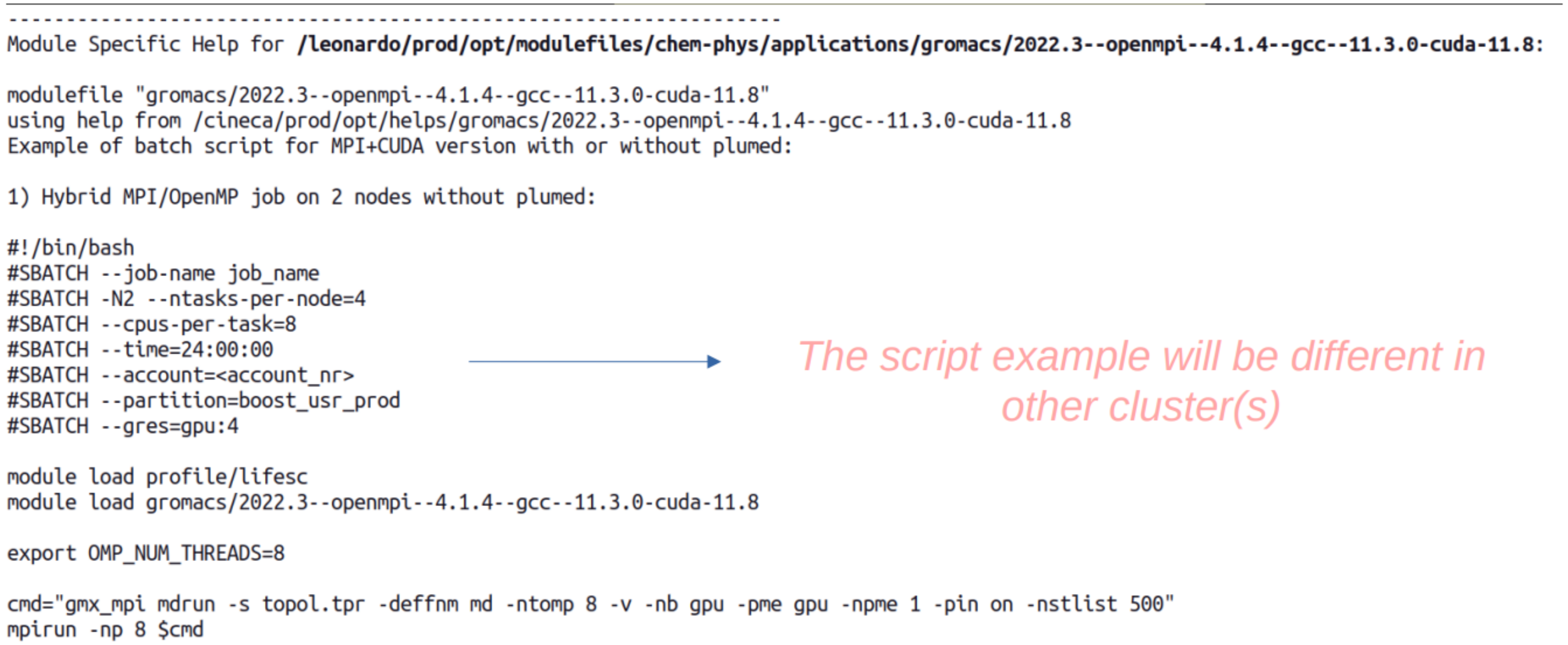
$ module show <module_name>/<version> $\longrightarrow$ Prints information about the module: dependencies, paths



```
[otrocon1@login05 ~]$ module show gromacs/2022.3--openmpi--4.1.4--gcc--11.3.0-cuda-11.8
-------------------------------------------------------------------
/leonardo/prod/opt/modulefiles/chem-phys/applications/gromacs/2022.3--openmpi--4.1.4--gcc--11.3.0-cuda-11.8:

module-whatis   {GROMACS is a molecular dynamics package primarily designed for simulations of proteins, lipids and nucleic acids. It was originally developed in the Biophysical
Chemistry department of University of Groningen, and is now maintained by contributors in universities and research centers across the world.}
module          load fftw/3.3.10--openmpi--4.1.4--gcc--11.3.0
module          load openblas/0.3.21--gcc--11.3.0
module          load openmpi/4.1.4--gcc--11.3.0-cuda-11.8
module          load plumed/2.8.1--openmpi--4.1.4--gcc--11.3.0
conflict        gromacs
prepend-path    GROMACS_LIB /leonardo/prod/spack/03/install/0.19/linux-rhel8-icelake/gcc-11.3.0/gromacs-2022.3-owzxiwojzrytaodpf2r7dvd63jflbvex/lib64
prepend-path    LIBRARY_PATH /leonardo/prod/spack/03/install/0.19/linux-rhel8-icelake/gcc-11.3.0/gromacs-2022.3-owzxiwojzrytaodpf2r7dvd63jflbvex/lib64
prepend-path    LD_LIBRARY_PATH /leonardo/prod/spack/03/install/0.19/linux-rhel8-icelake/gcc-11.3.0/gromacs-2022.3-owzxiwojzrytaodpf2r7dvd63jflbvex/lib64
prepend-path    GROMACS_INC /leonardo/prod/spack/03/install/0.19/linux-rhel8-icelake/gcc-11.3.0/gromacs-2022.3-owzxiwojzrytaodpf2r7dvd63jflbvex/include
prepend-path    GROMACS_INCLUDE /leonardo/prod/spack/03/install/0.19/linux-rhel8-icelake/gcc-11.3.0/gromacs-2022.3-owzxiwojzrytaodpf2r7dvd63jflbvex/include
prepend-path    C_INCLUDE_PATH /leonardo/prod/spack/03/install/0.19/linux-rhel8-icelake/gcc-11.3.0/gromacs-2022.3-owzxiwojzrytaodpf2r7dvd63jflbvex/include
prepend-path    CPLUS_INCLUDE_PATH /leonardo/prod/spack/03/install/0.19/linux-rhel8-icelake/gcc-11.3.0/gromacs-2022.3-owzxiwojzrytaodpf2r7dvd63jflbvex/include
prepend-path    CPATH /leonardo/prod/spack/03/install/0.19/linux-rhel8-icelake/gcc-11.3.0/gromacs-2022.3-owzxiwojzrytaodpf2r7dvd63jflbvex/include
prepend-path    PATH /leonardo/prod/spack/03/install/0.19/linux-rhel8-icelake/gcc-11.3.0/gromacs-2022.3-owzxiwojzrytaodpf2r7dvd63jflbvex/bin
prepend-path    MANPATH /leonardo/prod/spack/03/install/0.19/linux-rhel8-icelake/gcc-11.3.0/gromacs-2022.3-owzxiwojzrytaodpf2r7dvd63jflbvex/share/man
prepend-path    PKG_CONFIG_PATH /leonardo/prod/spack/03/install/0.19/linux-rhel8-icelake/gcc-11.3.0/gromacs-2022.3-owzxiwojzrytaodpf2r7dvd63jflbvex/lib64/pkgconfig
prepend-path    CMAKE_PREFIX_PATH /leonardo/prod/spack/03/install/0.19/linux-rhel8-icelake/gcc-11.3.0/gromacs-2022.3-owzxiwojzrytaodpf2r7dvd63jflbvex/.
setenv          GROMACS_HOME /leonardo/prod/spack/03/install/0.19/linux-rhel8-icelake/gcc-11.3.0/gromacs-2022.3-owzxiwojzrytaodpf2r7dvd63jflbvex
-------------------------------------------------------------------
```

Module environment

```
$ module load profile/lifesc
$ module load gromacs/2022.3—openmpi--4.1.4--gcc--11.3.0-cuda-11.8
$ module help gromacs/2022.3—openmpi--4.1.4--gcc--11.3.0-cuda-11.8
```



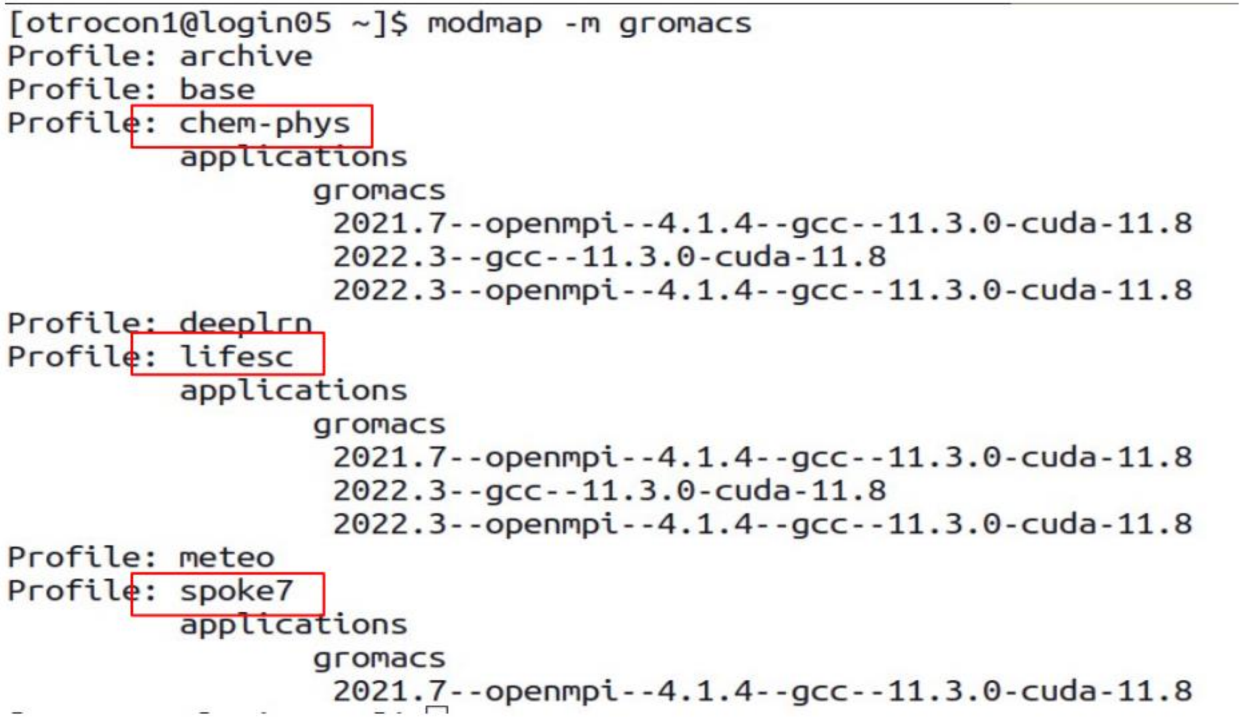
```
-------------------------------------------------------------------
Module Specific Help for /leonardo/prod/opt/modulefiles/chem-phys/applications/gromacs/2022.3--openmpi--4.1.4--gcc--11.3.0-cuda-11.8:

modulefile "gromacs/2022.3--openmpi--4.1.4--gcc--11.3.0-cuda-11.8"
using help from /cineca/prod/opt/helps/gromacs/2022.3--openmpi--4.1.4--gcc--11.3.0-cuda-11.8
Example of batch script for MPI+CUDA version with or without plumed:

1) Hybrid MPI/OpenMP job on 2 nodes without plumed:

#!/bin/bash
#SBATCH --job-name job_name
#SBATCH -N2 --ntasks-per-node=4
#SBATCH --cpus-per-task=8
#SBATCH --time=24:00:00
#SBATCH --account=<account_nr>
#SBATCH --partition=boost_usr_prod
#SBATCH --gres=gpu:4

module load profile/lifesc
module load gromacs/2022.3--openmpi--4.1.4--gcc--11.3.0-cuda-11.8

export OMP_NUM_THREADS=8

cmd="gmx_mpi mdrun -s topol.tpr -deffnm md -ntomp 8 -v -nb gpu -pme gpu -npme 1 -pin on -nstlist 500"
mpirun -np 8 $cmd
```

$\longrightarrow$ *The script example will be different in other cluster(s)*

Module environment

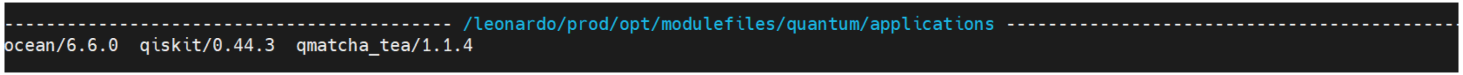
How to find a module that I do not know in which profile is it?

$ modmap -m <module_name> $\longrightarrow$ a command that looks for a module in all profiles



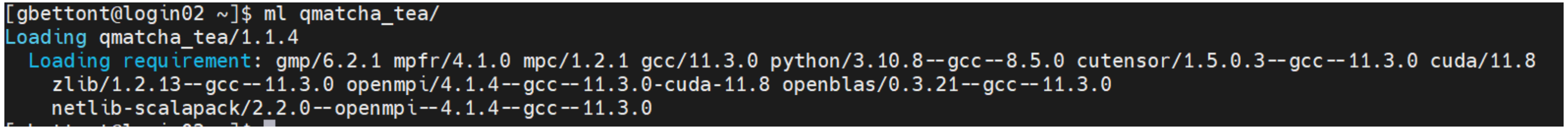
```
[otrocon1@login05 ~]$ modmap -m gromacs
Profile: archive
Profile: base
Profile: chem-phys
         applications
                gromacs
                 2021.7--openmpi--4.1.4--gcc--11.3.0-cuda-11.8
                 2022.3--gcc--11.3.0-cuda-11.8
                 2022.3--openmpi--4.1.4--gcc--11.3.0-cuda-11.8
Profile: deeplrn
Profile: lifesc
         applications
                gromacs
                 2021.7--openmpi--4.1.4--gcc--11.3.0-cuda-11.8
                 2022.3--gcc--11.3.0-cuda-11.8
                 2022.3--openmpi--4.1.4--gcc--11.3.0-cuda-11.8
Profile: meteo
Profile: spoke7
         applications
                gromacs
                 2021.7--openmpi--4.1.4--gcc--11.3.0-cuda-11.8
```

module load profile/quantum



```
------------------------------------------ /leonardo/prod/opt/modulefiles/quantum/applications ------------------------------------------
ocean/6.6.0  qiskit/0.44.3  qmatcha_tea/1.1.4
```

ml qmatcha_tea/



```
[gbettont@login02 ~]$ ml qmatcha_tea/
Loading qmatcha_tea/1.1.4
  Loading requirement: gmp/6.2.1 mpfr/4.1.0 mpc/1.2.1 gcc/11.3.0 python/3.10.8--gcc--8.5.0 cutensor/1.5.0.3--gcc--11.3.0 cuda/11.8
    zlib/1.2.13--gcc--11.3.0 openmpi/4.1.4--gcc--11.3.0-cuda-11.8 openblas/0.3.21--gcc--11.3.0
    netlib-scalapack/2.2.0--openmpi--4.1.4--gcc--11.3.0
```

Please remember that on G100 you need also this command --> module load autoload

Production environment: how to submit my simulations

You have compiled your code! Congratulations!!!
…now, what to do with it?

The **production environment** takes care of all the aspects related to the actual execution of the program you want to run. Time to "produce" some results!!

Your work areas
(recap)

$HOME:

Permanent, backed-up, and local to LEONARDO. 50 Gb of quota. For source code or important input files.

$SCRATCH:

Large, parallel filesystem. No quota. Run your simulations and calculations here. A cleaning policy will delete all your files older than 40 days (not active yet).

$WORK:

Similar to $SCRATCH, but the content is shared among all the users of the same account.

1 Tb of quota (no cleaning policy).

$PUBLIC:

A small area to share installations, permanent but not backed-up. 50 Gb of quota.

use the command **cindata** to get info on your disk occupation

https://wiki.u-gov.it/confluence/display/SCAIUS/UG2.5%3A+Data+storage+and+FileSystems

Login nodes

If you have a serial program, the most intuitive thing to do is to just launch `./myprogram` wherever you are.

When you log into LEONARDO, you find yourself in one of the four login nodes, selcted in round robin fashion to balance out the load of users.

Interactive runs on login nodes are strongly discouraged and should be limited to **short test runs**
There are per user limits on cpu-time (10 minutes)
IMPORTANT: avoid running large parallel applications on the front-ends!!

LEONARDO is a general purpose system used by hundreds of users.

Compute nodes: jobs & scheduler

What we actually want most of the time is to gain access to the compute nodes to exploit their power

Like in any HPC cluster, LEONARDO allows you to run your simulations by submitting **"jobs"** to the compute nodes

Your job is then taken in consideration by a **scheduler**, that adds it to a queuing line and allows its execution when the resources required are available

The operative scheduler on LEONARDO is **SLURM**

SLURM stands for "Simple Linux Utility for Resource Management"

- Allocating access to resources
- Job starting, executing and monitoring
- Queue of pending jobs management

Compute nodes: jobs & scheduler

The scheme for a SLURM job script is as follows:

#!/bin/bash

#SLURM directives

variables environment

execution line

Jobscript example

```
#!/bin/bash
#SBATCH -t 1:00:00
#SBATCH -N 2
#SBATCH --ntasks-per-node=16
#SBATCH --cpus-per-task=2
#SBATCH --gres=gpu:4
#SBATCH --mem=10GB
#SBATCH -o job.out
#SBATCH -e job.err
#SBATCH -p boost_usr_prod
#SBATCH -A <my_account>
module load openmpi
export OMP_PROC_BIND=true
mpirun -n 32 ./myprogram
```

Slurm directives

#SBATCH --job-name=myname, -J myname

Defines the name of your job

#SBATCH --output=job.out, -o job.out

Specifies the file where the standard output is directed (default=slurm-<Pid>)

#SBATCH --error=job.err, -e job.err

Specifies the file where the standard error is directed (default=slurm-<Pid>)

Slurm directives: resource requirements

```
#SBATCH --nodes=1, -N 1
#SBATCH --ntasks-per-node=8
#SBATCH --cpus-per-task=4
#SBATCH --gres=gpu:4
#SBATCH --mem=10000        # mem=0 equals to full memory
```

nodes – number of compute nodes
ntasks-per-node – number of tasks per node (max. 32)
cpus-per-task – number of cpus to be assigned to each task
ntasks-per-node*cpus-per-task ≤ 32
gres=gpu:x – number of GPUs for each node (x=1..4)
mem – memory allocated for each node (max=494000 MB).

Slurm directives: walltime and partitions

#SBATCH --time=00:30:00, -t 00:30:00

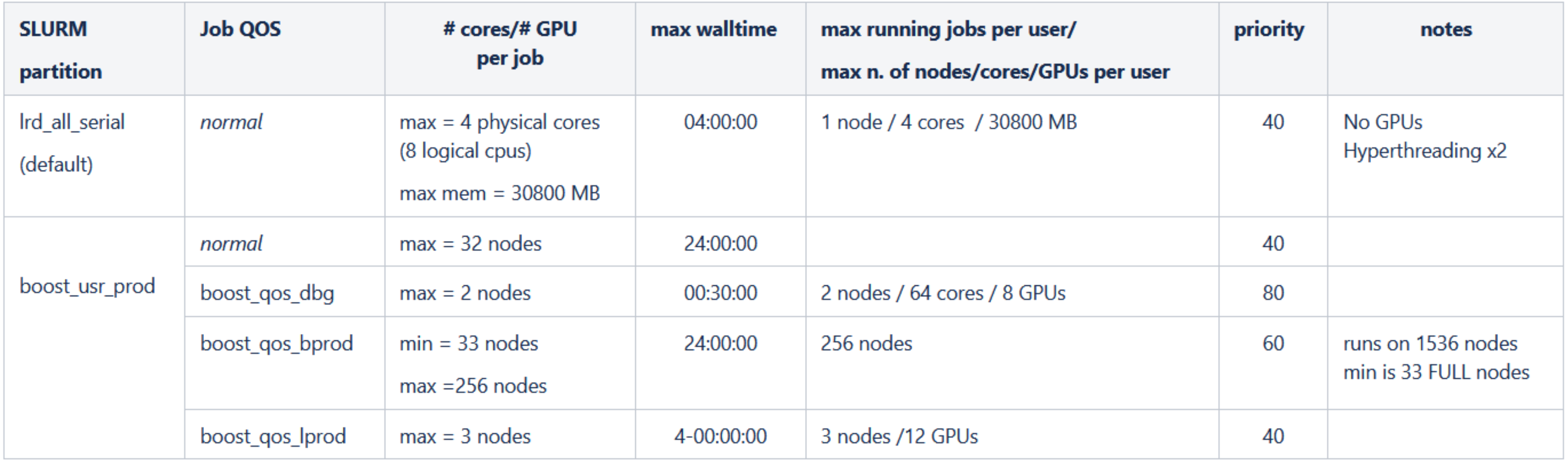
Specifies the maximum duration of the job. The maximum time allowed depends on the partition used

Pro-tip: the less walltime you ask, the faster your job will enter in execution. Think about it!

#SBATCH --partition=boost_usr_prod, -p boost_usr_prod
#SBATCH --qos=boost_qos_dbg, -q boost_qos_dbg (optional)

Specifies the “partition”, a.k.a. the specific set of nodes among which your job can search for resources. Optionally you can specify a QoS (Quality of Service) for jobs with particular purposes, like debugging or large production

Available partitions and QoS on LEONARDO



SLURM partition	Job QOS	# cores/# GPU per job	max walltime	max running jobs per user/ max n. of nodes/cores/GPUs per user	priority	notes
lrd_all_serial (default)	*normal*	max = 4 physical cores (8 logical cpus) max mem = 30800 MB	04:00:00	1 node / 4 cores / 30800 MB	40	No GPUs Hyperthreading x2
boost_usr_prod	*normal*	max = 32 nodes	24:00:00		40	
	boost_qos_dbg	max = 2 nodes	00:30:00	2 nodes / 64 cores / 8 GPUs	80	
	boost_qos_bprod	min = 33 nodes max =256 nodes	24:00:00	256 nodes	60	runs on 1536 nodes min is 33 FULL nodes
	boost_qos_lprod	max = 3 nodes	4-00:00:00	3 nodes /12 GPUs	40	

Notes:
- the partition **lrd_all_serial** run on front-end nodes, and as such it is not subject to accounting and can be used for free
- to use the QoS **boost_qos_bprod**, the minimum requirement is for cpus, gpus, nodes and mem to be over the regular limit…that’s why the nodes have to be asked in full!

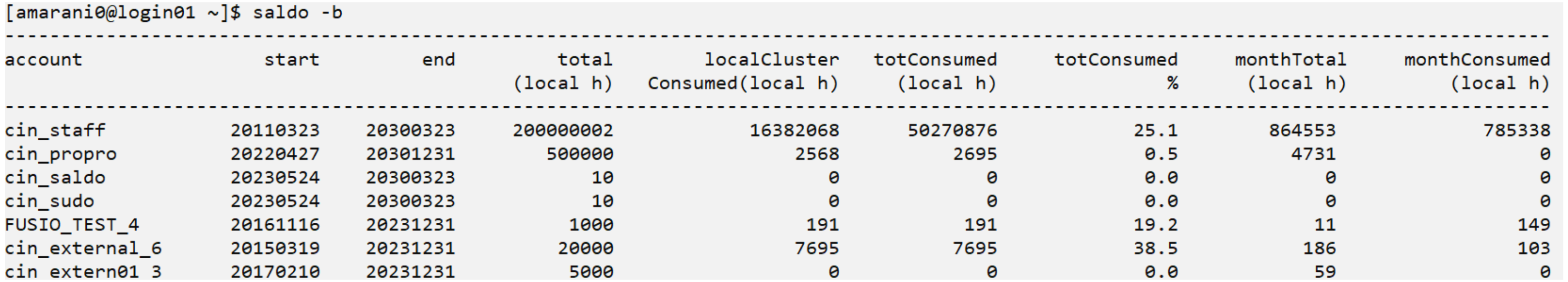
Slurm directives: accounting

#SBATCH --account=<my_account>, -A <my_account>

Specifies the account to use the CPU hours from.

As an user, you have access to a limited number of CPU hours to spend. They are not assigned to users, but to **projects** and are shared between the users who are working on the same project (i.e. your research partners). Such projects are called **accounts** and are a different concept from your username.

You can check the status of your account with the command "*saldo -b*", which tells you how many CPU hours you have already consumed for each account you're assigned at (a more detailed report is provided by "*saldo -r*").



```
[amarani0@login01 ~]$ saldo -b
-----------------------------------------------------------------------------------------------------------------------
account                 start         end         total        localCluster   totConsumed     totConsumed     monthTotal     monthConsumed
                                              (local h)   Consumed(local h)     (local h)               %      (local h)         (local h)
-----------------------------------------------------------------------------------------------------------------------
cin_staff            20110323    20300323     200000002            16382068      50270876            25.1         864553            785338
cin_propro           20220427    20301231        500000                2568          2695             0.5           4731                 0
cin_saldo            20230524    20300323            10                   0             0             0.0              0                 0
cin_sudo             20230524    20300323            10                   0             0             0.0              0                 0
FUSIO_TEST_4         20161116    20231231          1000                 191           191            19.2             11               149
cin_external_6       20150319    20231231         20000                7695          7695            38.5            186               103
cin_extern01_3       20170210    20231231          5000                   0             0             0.0             59                 0
```

Submitting a job

You have created your jobscript! Congratulations!!!
…now, what to do with it?

sbatch

```
sbatch <job_script>
```

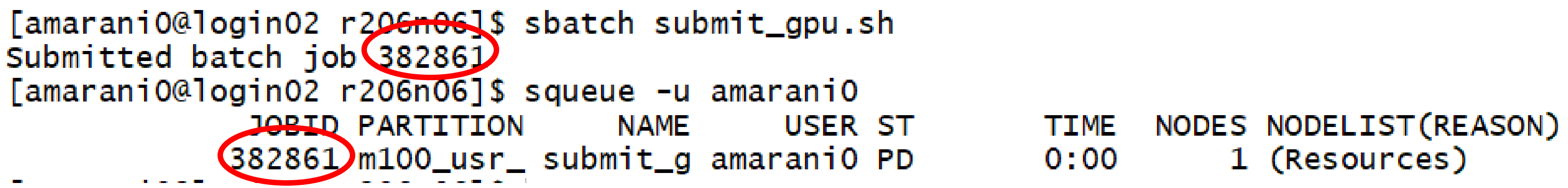
Your job will be submitted to the SLURM scheduler and executed when there will be nodes available (according to your priority and the partition you requested)

squeue -u

```
squeue -u <username>
```

Shows the list of all your scheduled jobs, along with their status (idle, running, closing, …)
It also shows you the job id required for other SLURM commands

Other Slurm commands - 1



```
[amarani0@login02 r206n06]$ sbatch submit_gpu.sh
Submitted batch job 382861
[amarani0@login02 r206n06]$ squeue -u amarani0
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
            382861 m100_usr_ submit_g amarani0 PD       0:00      1 (Resources)
```

scontrol show job

scontrol show job <job_id>

Provides a long list of informations for the job requested.

In particular, if your job isn’t running yet, you'll be notified about the reason it is not starting and, if it is scheduled with top priority, you will get an estimated start time

scancel

scancel <job_id>

Removes the job (queued or running) from the scheduled job list by killing it

Other Slurm commands - 2

sinfo

sinfo -p <partition name>

sinfo -l

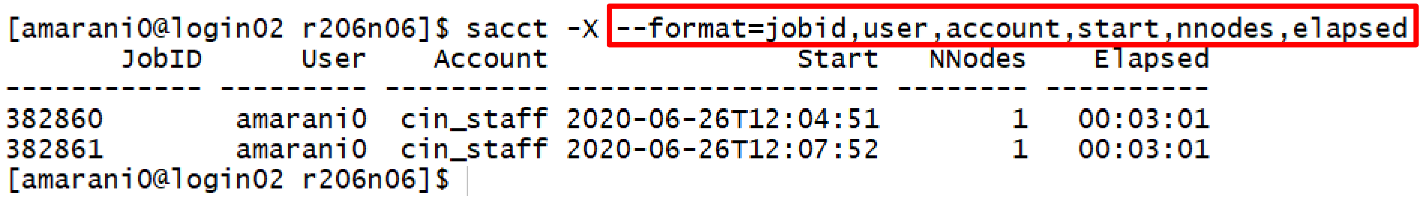
sinfo -N -l -p m100_usr_prod

Provides information about SLURM nodes and partitions

sacct

sacct OPTIONS <job_id>

displays accounting data for all jobs and job steps in the SLURM job accounting log or Slurm database. Can also see the jobs already completed or cancelled



```
[amarani0@login02 r206n06]$ sacct -X --format=jobid,user,account,start,nnodes,elapsed
       JobID      User    Account               Start   NNodes    Elapsed
------------ --------- ---------- ------------------- -------- ----------
382860        amarani0  cin_staff 2020-06-26T12:04:51        1   00:03:01
382861        amarani0  cin_staff 2020-06-26T12:07:52        1   00:03:01
[amarani0@login02 r206n06]$
```

Documentation

Our userguide goes into more depth about all the aspects described during this presentation. In particular, we suggest:

https://wiki.u-gov.it/confluence/display/SCAIUS/UG3.2%3A+LEONARDO+UserGuide#UG3.2:LEONARDOUserGuide-Productionenvironment
Production environment on LEONARDO

https://wiki.u-gov.it/confluence/display/SCAIUS/UG2.5%3A+Data+storage+and+FileSystems
Work areas and filesystem

https://wiki.u-gov.it/confluence/display/SCAIUS/UG2.4%3A+Accounting
Accounting and budget consumption

https://wiki.u-gov.it/confluence/display/SCAIUS/UG2.6.1%3A+How+to+submit+the+job+-+Batch+Scheduler+SLURM
Batch scheduler Slurm

coming soon…
Analysis on thread and task affinity on LEONARDO