

HPC School - Beginner

 $\ensuremath{\mathsf{S2}}\xspace$ - Work on the ULHPC



1



Outline

- When can the ULHPC help me?
- How a cluster works?
- Types of workers
- Types of jobs
- Partitions and QoS
- Modules
- Monitoring your jobs
- Storage
- Learn more by yourself
- What can I do to help the ULHPC?





When can the ULHPC help me?

Embarrassingly parallel jobs

This is when you have a lot of similar jobs to run. Maybe running one job on your laptop is fine but 10000 jobs would take too long.

Multi threaded applications

Laptop / work machines usually have 2 to 16 cores. If what you run can take advantage (compute multiple things at the same time by distributing computation on the available cores), then you could benefit from our nodes, ranging from 28 to 128 cores.





When can the ULHPC help me?

Not enough memory on my machine

Laptop / work machines usually have between 8G to 32G or RAM. This may be too small for your experiments. We have nodes from 128G to 3000G of RAM.

Multi node (computer) application

Sometime, even one big node is not enough. Our cluster allow you to run jobs up to 64 nodes per job. On AION this means 8192 cores and 16T of RAM.

Not enough storage

The ULHPC benefits from several storage services for a total of 10PB (10 000 TB)





How a cluster works?

- You first access a cluster via its access node
- You then use worker nodes to compute your jobs
- Access nodes
 - Servers on which you "land" when you connect on the cluster
 - Can be used to request resources
 - Should not be used to compute things
 - Application programs via module are not available
- Worker nodes
 - Servers on which computation should be run
 - When you request some resources from the access nodes, the resources are from the worker nodes
 - Several types of worker nodes at ULHPC (discussed in detail later)





How a cluster works?

Example:

- You connect on AION, you are now on an access node
- For your work, you need 64 cores / 128G of RAM
- You request those resources from the access node
- When available, you land on the machine on which the resources you received are located

- ssh aion-cluster

Welcome to access1.aion-cluster.uni.lux



Last login: Thu Jun 22 14:53:25 2023 from 83.194.117.49 (base) 0 [jschleich@access1 ~]\$





How a cluster works?

Example:

- You connect on AION, you are now on an access node
- For your work, you need 64 cores / 128G of RAM
- You request those resources from the access node
- When available, you land on the machine on which the resources you received are located

0 [jschleich@access1 ~]\$ si -c64

salloc -p interactive --qos debug -C batch -c64
salloc: Granted job allocation 803494
salloc: Waiting for resource configuration
salloc: Nodes aion-0041 are ready for job
(base) 0 [jschleich@aion-0041 ~](803494 1N/T/1CN)\$





Lëtz build ourselves a little playground

Go to the home directory
\$ cd

If you have not done it yet - clone the repository containing the files on the ULHPC
\$ git clone https://github.com/ULHPC/hpc-school-for-beginners.git

If you already have cloned it - please update it
\$ cd hpc-school-for-beginners; git pull

If you encounter an issue delete the folder and clone again
\$ rm -Rf hpc-school-for-beginners





Types of worker nodes at ULHPC

Currently the ULHPC offers the following types of resources:

CPU nodes

- Recommended for most usages
- Large number of nodes

GPU nodes

- Nodes with graphic card accelerators
- More and more tools take advantage of GPUs
- Limited number of nodes

Bigmem nodes

- Recommended when a tool has huge memory requirements which cannot be distributed over multiple nodes
- Very limited number of nodes





Types of worker nodes at ULHPC

Currently the ULHPC offers the following types of resources:

CPU nodes

- AION: 354 nodes, each node has 128 cores and 256G of RAM
- IRIS: 168 nodes, each node has 28 cores and 128G of RAM

GPU nodes

- IRIS: **18 nodes**, each node has 28 cores and 768G of RAM and 4 NVIDIA V100 with 16G
- IRIS: 6 nodes, each node has 28 cores and 768G of RAM and 4 NVIDIA V100 with 32G

Bigmem nodes

- IRIS: 4 nodes, each node has 112 cores and 3T RAM





Types of jobs

Two types of jobs:

- interactive jobs
- batch jobs

Interactive: when you receive the resources you can type commands in an interactive fashion and see the results. This is adapted to debugging / trial and errors.

Batch: you submit the commands you wish to be executed and you specify the resources. When the resources are available, your commands are executed automatically. This type of job is adapted to run campaigns of experiments.





Interactive jobs

Request an interactive job

- si for CPU nodes
- si-gpu for GPU nodes (on the IRIS cluster only)
- si-bigmem for bigmem nodes (on the IRIS cluster only)

Important parameters

- -t to specify the duration. 30 min is the default, 120 min is the max
- - c to specify the number of cores. 1 by default.
- add --reservation=school-interactive to use the HPC School reservation

Example: si -c8 -t120 --reservation=school-interactive request a 2 hours interactive session with 8 cores on a CPU node





Interactive jobs

Multiple jobs can run on each node, from multiple users. How are the resources shared?

Example 1

- A user wants 64 cores on an AION node
- Reminder: each AION node has 128 cores and 256G of RAM
- If the user enters si -c64, that user will have half the available cores and will automatically receive half the RAM:
 64 cores and 128G of RAM.
- It means 1 AION core \rightarrow 2G of RAM

Example 2

- A user wants 1 core on an IRIS node:
- Reminder: each IRIS CPU node has 28 cores and 128G of RAM
- The user will receive 1/28th of 128G of RAM, roughly 4G

Note: IRIS and AION CPU nodes have a different RAM per core ratio





Interactive jobs

Multiple jobs can run on each node, from multiple users. How are the resources shared?

Example 3

- A user wants 1 GPU to run some experiment
- On a GPU node, you also have CPU cores and RAM
- All of those resources are linked together
- Reminder: each GPU node has
 - 28 cores
 - 768G of RAM
 - 4 NVIDIA V100
- si-gpu -c7 will lead to: 1 GPU, 7 CPU cores and 7/28th (¼) of the 768G of RAM

Note: requesting more than 7 CPU cores could lead to some GPUs to not be allocable for other users by Slurm. Please think about this when using GPU nodes. In case of doubt, contact us via service now.



Interactive jobs - 🤣 now it is your turn

Exercice 1 Request 8 cores for 60 minutes Check the worker node name Close your interactive session to deallocate the ressources

Exercice 2 Request enough cores to have 64G of RAM on an AION node for 2 hours

Exercice 3

Can you book the same amount of cores on an IRIS CPU node than the answer of exercice 2? How much cores would you have to request on a IRIS CPU node to have 64G of RAM?

Note: do not forget to add --reservation=school-interactive or to use the HPC School reservation





Interactive jobs - solutions

Exercice 1
Request 8 cores for 60 minutes
Solution: si -c8 -t60 --reservation=school-interactive

Exercice 2 Request enough cores to have 64G of RAM on an AION node for 2 hours **Solution:** si -c32 -t120 --reservation=school-interactive

Exercice 3

Can you book the same amount of cores on an IRIS CPU node than the answer of exercice 2? Solution: no, IRIS CPU nodes have 28 cores How much cores would you have to request on a IRIS CPU node to have 64G of RAM? Solution: 14 cores, each cores receives 4G on an IRIS CPU node





Batch jobs

Submit a batch job

- Use the sbatch command, usually, sbatch some-script.sh
- The script contains:
 - A first section containing Slurm parameters (what resources you want, for how long...).
 - A second section containing what your job should do with those resources
- This script is usually referred as the launcher script
- We maintain launcher script templates for various use cases, see documentation





Batch jobs

Submit a batch job

- First line is mandatory for scripts
- #SBATCH parameters specify your job characteristics. Here we request 16 cores for 5 minutes on the batch partition (CPU)
- Anything after #SBATCH is what should be executed on the allocated resources. Here, we execute a Python script.

```
1 #!/bin/bash -l
2 #SBATCH -c 16
3 #SBATCH --time=0-00:05:00
4 #SBATCH -p batch
5
6 module load lang/Python/3
7
8 python my-script.py
9
```

Socumentation about SBATCH options: https://hpc-docs.uni.lu/slurm/#job-submission-options





Batch jobs - 🤣 now it is your turn

Exercice 1

Execute your first batch job, use the one in batch-job/batch-job-launcher.sh.Check the slurm output file and ensure it contains the "It works" message. It should be in a file named like this slurm-JOBID.out

Exercice 2

Execute the same launcher multiple times but:

- The job names should be different, e.g. job1, job2...
- The output and error files should contain the job names, e.g. job1.out, job2.err

Exercice 3

Add email notification to your launcher to receive an email with your jobs are done



Number of tasks and core per task

Slurm tasks?

- In our documentation you will come across the notion of Slurm task
- In our launcher templates you will see -n or --n-tasks-per-node
- For most use case, do not use it or set it to 1
- If your application does not support multi-node computation \rightarrow 1 task
- There are exceptions, in case of doubt, contact us via service now

Note: if your app is not fast enough, do not increase -n as an attempt to speed up the computation: it will allocate more resources but they will likely not be used





What are partitions?

Partitions

In Slurm multiple nodes can be grouped into partitions which are sets of nodes aggregated by shared characteristics.

You will find on ULHPC resources the following partitions:

- **batch** is intended for running parallel scientific applications as passive jobs on CPU nodes
- gpu is intended for running GPU-accelerated scientific applications as passive jobs on "gpu" nodes
- **bigmem** is dedicated for memory intensive data processing jobs on "bigmem" nodes
- **interactive**: a floating partition intended for interactive jobs





Partitions

Partitions

In Slurm multiple nodes can be grouped into partitions which are sets of nodes aggregated by shared characteristics.

Туре	Default/MaxTime	MaxNodes (per job)
interactive	30min - 2h	2
batch (cpu)	2h-48h	64
gpu	2h-48h	4
bigmem	2h-48h	1

Question:

- What is the maximum amount of GPUs you can use for one single job?
- Can you use the interactive partition to test a program over 10 nodes?





QoS

QoS (Quality of Service)

Quality of Service or QOS is used to constrain or modify the characteristics that a job can have. For example: longer run time or a high priority queue for a given job

Interesting QoS

- long: for longer jobs, max 4 (running) jobs per user (simplification), up to 14 days
- besteffort: a preemptible (your jobs can be killed when the cluster is too busy with other normal jobs and restarted when resources are available again), max 300 (running) jobs per user (simplification), up to 50 days

You can type sqos to learn about all existing QoS and their restrictions





QoS

Туре	Max # of running jobs	Max duration
normal	100	2 days
long	4 per users, 6 per user group	14 days
best effort	300	50 days





QoS

Example: submit a long job

sbatch --qos long my-script.sh

Example: submit a besteffort job

sbatch --qos besteffort my-script.sh



Documentation: https://hpc-docs.uni.lu/jobs/long/#long-jobs





Software on ULHPC

There are plenty of way to run software on the ULHPC:

- Modules (see next slides)
- Conda \rightarrow <u>check our tutorial</u>
- Containers \rightarrow <u>check our tutorial</u>
- Use Jupyter Notebook, Abaqus CAE, Matlab or Stata via a GUI \rightarrow <u>check our portal</u>
- Compile your own program \rightarrow too advanced for this tutorial

Note

The portal is only accessible from the UL network (or via the UL VPN)





Modules

- The ULHPC proposes and maintain software via modules.
- Pre-installed software, multiple version of the same software can co-exist
- Workflow: search modules, load them, use them
- **Only available on worker nodes**: you will see an error if you try to use the module command on an access node.





Module search

module av the-program-you-want

On the right, we search with the keyword "Python". The list of results contains various elements which are sorted by category (e.g. chem = Chemistry, lang = Programming languages, ...)

We can see that two version of the Python language are available: 2.7.18 and 3.8.6. If no version is specified, the default choice (D) will be assumed, here 3.8.6.

jschleich@iris-056 ~](3174323 1N/T/1CN)\$ module av Python

/opt/apps/re bio/TopHat/2.1.2-GCC-10.2	esif/iris-rhel8/2020b/broad 2.0-Python-2.7.18	well/modules/all -
chem/spglip-python/1.16.0	0-TOSS-2020D	(D)
devel/flatbuffers-python/	/1.12-GCCcore-10.2.0	(0)
devel/pkgconfig/1.5.1-GCC	Ccore-10.2.0-python	
devel/protobuf-python/3.1	14.0-GCCcore-10.2.0	
lang/Python/2.7.18-GCCco	re-10.2.0	
lang/Python/3.8.6-GCCcore	e-10.2.0	(D)
lang/SciPy-bundle/2020.11	1-foss-2020b-Python-2.7.18	
lib/Boost.Python/1.74.0-0	GCC-10.2.0	

Where: D: Default Module



Documentation: https://hpc-docs.uni.lu/environment/modules/



Module search

module av the-program-you-want

Now it is your turn:

- Look for a program that may interest you, e.g. Matlab

jschleich@iris-056 ~](3174323 1N/T/1CN)\$ module av Python

/opt/apps/resif/iris-rhel8/2020b/broad bio/TopHat/2.1.2-GCC-10.2.0-Python-2.7.18 chem/spulib-python/1.16.0-foss-2020b	well/modules/all -
chem/spglib-python/1.16.0-intel-2020b	(D)
devel/flatbuffers-python/1.12-GCCcore-10.2.0	
devel/pkgconfig/1.5.1-GCCcore-10.2.0-python	
devel/protobuf-python/3.14.0-GCCcore-10.2.0	
lang/Python/2.7.18-GCCcore-10.2.0	
lang/Python/3.8.6-GCCcore-10.2.0	(D)
<pre>lang/SciPy-bundle/2020.11-foss-2020b-Python-2.7.18</pre>	
lib/Boost.Python/1.74.0-GCC-10.2.0	

Where:

D: Default Module



Documentation: https://hpc-docs.uni.lu/environment/modules/



Module list

List the currently loaded modules

module list

Module load

module load the-program-you-want

Module purge

Unload all loaded modules

module purge

0 [jschleich@iris-056 ~](3174323 1N/T/10	N)\$ module list
No modules loaded	
0 [jschleich@iris-056 ~](3174323 1N/T/10	N)\$ module load lang/Python
0 [jschleich@iris-056 ~](3174323 1N/T/10	<code>XN)\$ module list</code>
Currently Loaded Modules:	
 compiler/GCCcore/10.2.0 	7) lang/Tcl/8.6.10-GCCcore-10.2.0
<pre>2) lib/zlib/1.2.11-GCCcore-10.2.0</pre>	8) devel/SQLite/3.33.0-GCCcore-10.2.
3) tools/binutils/2.35-GCCcore-10.2.0	9) tools/XZ/5.2.5-GCCcore-10.2.0
<pre>4) tools/bzip2/1.0.8-GCCcore-10.2.0</pre>	10) math/GMP/6.2.0-GCCcore-10.2.0
5) devel/ncurses/6.2-GCCcore-10.2.0	<pre>11) lib/libffi/3.3-GCCcore-10.2.0</pre>
6) lib/libreadline/8.0-GCCcore-10.2.0	12) lang/Python/3.8.6-GCCcore-10.2.0

0 [jschleich@iris-056 ~](3174323 1N/T/1CN)\$ module purge 0 [jschleich@iris-056 ~](3174323 1N/T/1CN)\$ module list No modules loaded

Documentation: https://hpc-docs.uni.lu/environment/modules/





Module list

List the currently loaded modules

module list

Module load

module load the-program-you-want

Module purge

Unload all loaded modules

module purge



- Ensure you have no loaded module
- Look for Python and load the 3.8 Python module
- Ensure Python 3.8 is loaded via python --version
- Purge your environment





Sticky modules

Modules that change the set of available modules

Default set of modules:

module load env/development/2023b

/opt/a env/deprecated/2019b (\$) env/development/2 env/development/2023b (\$) env/legacy/2019b	apps/easybuild/environment/modules	<pre>/opt/apps/easybuild/environment/modules</pre>
	fs/software.eessi.io/init/modules	/cvmfs/software.eessi.io/init/modules
/opt/app bio/ABySS/2.2.5-foss-2020b bio/BEDTools/2.30.0-GCC-10.2.0 bio/BLAST+/2.11.0-gompi-2020b bio/BMATools/2.5.1-GCC-10.2.0 bio/BomTools/2.5.1-GCC-10.2.0 bio/FastQC/0.11.9-Java-11 bio/GROMACS/2021-foss-2020b bio/HTSlib/1.12-GCC-10.2.0 bio/SAMtools/1.12-GCC-10.2.0 bio/SAMtools/1.12-GCC-10.2.0 bio/SAMtools/1.12-GCC-10.2.0 bio/SAMtools/1.12-GCC-10.2.0 bio/SAMtools/1.12-GCC-10.2.0 bio/SAMtools/1.12-GCC-10.2.0 bio/SAMtools/1.12-GCC-10.2.0 bio/SAMtools/1.12-GCC-10.2.0 bio/SAMtools/2021-hotfix-2207 cae/ABAQUS/2021-hotfix-2207 cae/ABAQUS/2022	<pre>ps/resif/aion/2020b/epyc/modules/all</pre>	ai/PyTorch/2.3.0-foss-2023b math/GMP-ECM/7.0.5-GCCcore-13.2.0 bio/GROMACS/2024.4-foss-2023b-PLUMED-3.9.2 math/GMP-ECM/7.0.5-GCCcore-13.2.0 bio/GROMACS/2024.4-foss-2023b (D) cae/OpenFOAM/V2312-foss-2023b math/GMP/6.3.0-GCCcore-13.2.0 chem/CP2K/2023.1-foss-2023b math/GMP/6.3.0-GCCcore-13.2.0 chem/LAMMPS/29Aug2024-foss-2023b math/GL/0.6-GCCcore-13.2.0 chem/LDMMPS/29Aug2024-foss-2023b math/INL/1.0.5-gfbf-2023b chem/LDMMPS/29Aug2024-foss-2023b-kokkos math/ISL/0.26-GCCcore-13.2.0 chem/MDI/1.4.29-gompi-2023b math/KAHIP/3.16-gompi-2023b chem/PLUMED/2.9.2-foss-2023b math/ME/1.3.1-6-GCcore-13.2.0 chem/PLUMED/2.9.2-foss-2023b math/METI/5.1.0-GCCcore-13.2.0 chem/PLUMED/2.9.2-foss-2023b math/MPC/1.3.1-6CCcore-13.2.0 chem/PLUMED/2.9.2-foss-2023b math/MPTI/1.5.1.0-GCCcore-13.2.0 chem/PLUMED/2.9.2-GCC-13.2.0 math/MPTI/1.5.1.0-GCCcore-13.2.0 chem/libxc/6.2.2-6CC-13.2.0 math/MPTI/1.5.4-GCCcore-13.2.0 compiler/GCC/13.2.0 math/MPTFI/1.5.4-GCCcore-13.2.0 compiler/GCC/13.2.0 math/MPTFI/1.5.4-GCCcore-13.2.0 compiler/GO/1.22.1 math/NUMPS/5.6.1-foss-2023b-metis compiler/GO/1.22.1 math/NUMPS/5.1.0-GCCcore-13.2.0







Sticky modules

Modules that change the set of available modules

Default set of modules:

	/opt/apps	/easybuil	d/environment/module	s
env/deprecated/2019b (\$)	env/development/2023	(S ,D)	env/release/default	(S,L)
env/development/2023b (S)	env/legacy/2019b	(S)	env/release/2020b	(S,D)
	/cvmfs/s	oftware.e	essi.io/init/modules	
EESSI/2023.06				
	/opt/apps/r	esif/aion	/2020b/epyc/modules/	all
bio/ABySS/2.2.5-foss-2020b			lib/libtirpc/1.3.	1-GCCcore-10.2.0
bio/BEDTools/2.30.0-GCC-10.2	2.0		lib/libtool/2.4.6	-GCCcore-10.2.0
bio/BLAST+/2.11.0-gompi-2020	b		lib/libunwind/1.4	.0-GCCcore-10.2.0
bio/BWA/0.7.17-GCC-10.2.0			lib/libvorbis/1.3	.7-GCCcore-10.2.0
bio/BamTools/2.5.1-GCC-10.2.	0		lib/libwebp/1.1.0	-GCCcore-10.2.0
bio/BioPerl/1.7.8-GCCcore-10	.2.0		lib/libxml2/2.9.1	0-GCCcore-10.2.0
bio/Bowtie2/2.4.2-GCC-10.2.0			lib/libyaml/0.2.5	-GCCcore-10.2.0
bio/FastQC/0.11.9-Java-11			lib/lz4/1.9.2-GCC	core-10.2.0
bio/GROMACS/2021-foss-2020b			lib/nettle/3.6-GC	Ccore-10.2.0
bio/HTSlib/1.12-GCC-10.2.0			lib/pybind11/2.6.	0-GCCcore-10.2.0
bio/SAMtools/1.12-GCC-10.2.0			lib/scikit-build/	0.11.1-foss-2020b
bio/TopHat/2.1.2-GCC-10.2.0-	Python-2.7.18		lib/snappy/1.1.8-	GCCcore-10.2.0
cae/ABAQUS/2021-hotfix-2207			lib/tbb/2020.3-GC	Ccore-10.2.0
cae/ABAQUS/2022		(D)	lib/tqdm/4.56.2-G	CCcore-10.2.0
cao/Nonor/4 6 0-foss-2020b			lib/zlib/1 2 11_G	CCcore 10.20

module load EESSI/2023.06

/cvmfs/software.eessi.	io/versions/20	023.06/software/linux/x86_64/amd/zen2/mo	dules/all
ALL/0.9.2-foss-2023a		Score-P/8.4-gompi-2023b	
AOFlagger/3.4.0-foss-2023b		Seaborn/0.13.2-gfbf-2023a	
ASE/3.22.1-gfbf-2022b		Shapely/2.0.1-gfbf-2023a	
ATK/2.38.0-GCCcore-12.2.0		SlurmViewer/1.0.1-GCCcore-13.2.0	
ATK/2.38.0-GCCcore-12.3.0		Solids4foam/2.1-foss-2023a	l
ATK/2.38.0-GCCcore-13.2.0	(D)	SuiteSparse/7.1.0-foss-2023a	
Abseil/20230125.2-GCCcore-12.2.0		SuperLU_DIST/8.1.2-foss-2023a	l
Abseil/20230125.3-GCCcore-12.3.0		Szip/2.1.1-GCCcore-12.2.0	
Abseil/20240116.1-GCCcore-13.2.0	(D)	Szip/2.1.1-GCCcore-12.3.0	
Archive-Zip/1.68-GCCcore-12.2.0		Szip/2.1.1-GCCcore-13.2.0	(D)
Armadillo/11.4.3-foss-2022b		Tcl/8.6.12-GCCcore-12.2.0	
Armadillo/12.6.2-foss-2023a		Tcl/8.6.13-GCCcore-12.3.0	
Armadillo/12.8.0-foss-2023b	(D)	Tcl/8.6.13-GCCcore-13.2.0	(D)
Arrow/11.0.0-gfbf-2022b		TensorFlow/2.13.0-foss-2023a	
Arrow/14.0.1-gfbf-2023a		Tk/8.6.12-GCCcore-12.2.0	
Arrow/16.1.0-gfbf-2023b	(D)	Tk/8.6.13-GCCcore-12.3.0	
BCFtools/1.17-GCC-12.2.0		Tk/8.6.13-GCCcore-13.2.0	(D)
BCFtools/1.18-GCC-12.3.0	(D)	Tkinter/3.10.8-GCCcore-12.2.0	
BLAST+/2.14.0-gompi-2022b		Tkinter/3.11.3-GCCcore-12.3.0	
BLAST+/2.14.1-gompi-2023a	(D)	Tkinter/3.11.5-GCCcore-13.2.0	(D)
BLIS/0.9.0-GCC-12.2.0		Tombo/1.5.1-foss-2023a	
BLIS/0.9.0-GCC-12.3.0		Transrate/1.0.3-GCC-12.3.0	







Sticky modules

Modules that change the set of available modules

• Local modules:

module load env/X/Y
where X = development, release, deprecated

• EESSI modules:

module load EESSI/2023.06

- Local modules more optimized, EESSI more standardized across systems
- Purge sticky modules with --force flag:

module --force purge







Why monitor your jobs?

- Check the status of your jobs
- For each job, check its progression
- Ensure ULHPC resources are used efficiently





Monitor your jobs - check the status of your jobs

To see the full list of your jobs and their current status, you can use: sq

In this example you see jobs of random user. The ST column means status and you can see jobs which are PD (pending, i.e. not yet started) and jobs which are R (running).

(base) 0	[jschlei	.ch@access1	~]\$ squeue -u djouba	aud							
	JOBID	PARTIT	QOS	NAME		NODE	CPUS	ST	TIME	TIME_LEFT	PRIORITY	NODELIST(REASON)
8	46027_[batch	normal	DatasetsGeneration	djoubaud			PD	0:00	1-23:59:00	11410	(QOSMaxJobsPerUserLimit)
8	46037_[batch	normal	DatasetsGeneration	djoubaud			PD	0:00	1-23:59:00	11410	(QOSMaxJobsPerUserLimit)
8	46036_[batch	normal	DatasetsGeneration	djoubaud			PD	0:00	1-23:59:00	11410	(QOSMaxJobsPerUserLimit)
8	46035_[batch	normal	DatasetsGeneration	djoubaud			PD	0:00	1-23:59:00	11410	(QOSMaxJobsPerUserLimit)
8	46034_[batch	normal	DatasetsGeneration	djoubaud			PD	0:00	1-23:59:00	11410	(QOSMaxJobsPerUserLimit)
8	46033_[batch	normal	DatasetsGeneration	djoubaud			PD	0:00	1-23:59:00	11410	(QOSMaxJobsPerUserLimit)
8	46027_7	batch	normal	DatasetsGeneration	djoubaud				9:06	1-23:49:54	11408	aion-0163
8	846027_6	batch	normal	DatasetsGeneration	djoubaud				12:12	1-23:46:48	11407	aion-0113
8	46027_5	batch	normal	DatasetsGeneration	djoubaud				32:42	1-23:26:18	11403	aion-0078
8	46027_1	batch	normal	DatasetsGeneration	djoubaud				57:38	1-23:01:22	11398	aion-0163
8	46027_0	batch	normal	DatasetsGeneration	djoubaud				59:36	1-22:59:24	11398	aion-0163
8	844973_7	batch	normal	DatasetsGeneration	djoubaud				1-08:57:50	15:01:10	11375	aion-0019





Monitor your jobs - check the progression of a job

By default, for a running job, there will be two files:

- An output file, containing the log of your job
- An error file, containing the errors of your job

By default, the files will be named slurm-JOBID.out and slurm-JOBID.err

You can check the content of those files with a variety of commands, from an access node:

- cat filename, less filename will display the current full content of the file
- tail -f filename will display the end of the file and keep waiting for new content until you close it via CTRL+C





Monitor your jobs - check the progression of a job

Now it is your turn:

- Go to the monitor folder
- Submit the launcher monitor.sh script inside it
- Follow the progression of the execution using tail -f command

Reminders: By default, the files will be named slurm-JOBID.out and slurm-JOBID.err

tail -f filename will display the end of the file and keep waiting for new content until you close it via CTRL+C





Monitor your jobs - check the efficient usage of resources

- 1. Use the following command: sjoin JOB-ID to connect to your worker's job
- 2. Use the htop command, press u and select your user to see what is happening
- 3. Exit by pressing q or CTRL+C





Monitor your jobs - check the efficient usage of resources

0[1[2[3[4[5[6[100.0%] 100.0%] 100.0%] 100.0%] 100.0%] 100.0%]	8[10 9[10 10[10 11[10 12[10 13[10 14[10	0.0%] 16 0.0%] 17 0.0%] 18 0.0%] 19 0.0%] 20 0.0%] 20 0.0%] 21 0.0%] 22	[10 [10 [10 [10 [10 [10 [10	0.0%] 0.0%] 0.0%] 0.0%] 0.0%] 0.0%] 0.0%]	24[25[26[27[28[28[30[100.0 100.0 100.0 100.0 100.0 100.0	 8] 32[] 8] 33[] 8] 34[] 8] 35[] 8] 36[] 8] 37[] 8] 38[] 8] 38[] 	10 10 10 10 10 10	0.0%] 0.0%] 0.0%] 0.0%] 0.0%] 0.0%] 0.0%]	40[41[42[43[44[45[46[100.09 100.09 100.09 100.09 100.09 100.09	<pre>48[49[50[50[51[52[53[53[54[</pre>	100.0 100.0 100.0 100.0 100.0 100.0	%] 56 %] 57 %] 58 %] 59 %] 60 %] 61 %] 62	[99. [100. [100. [100. [100. [100. [100.	4%] 0%] 0%] 0%] 0%] 0%] 0%]	64[65[66[67[68[69[70[100.0%] 100.0%] 100.0%] 100.0%] 100.0%] 100.0%] 100.0%]	72[73[74[75[76[77[78[100 100 100 100 100	0.0%] 80[0.0%] 81[0.0%] 82[0.0%] 83[0.0%] 84[0.0%] 85[0.0%] 86[100 100 100 100 100 100	.0%] 88 .0%] 89 .0%] 90 .0%] 91 .0%] 92 .0%] 93 .0%] 94	.00.0%] .00.0%] .00.0%] .00.0%] .00.0%] .00.0%]1 .00.0%]1	96[97[98[99[00[01[02[100.0%] 100.0%] 100.0%] 100.0%] 100.0%] 100.0%] 100.0%]	104[105[106[107[108[109[110[100 100 100 100 100 100	.0%]112 .0%]113 .0%]114 .0%]115 .0%]116 .0%]117 .0%]118	[10 [10 [10 [10 [10 [10 [10	8.0%]12 0.0%]12 0.0%]12 0.0%]12 0.0%]12 0.0%]12 0.0%]12 0.0%]12 0.0%]12 0.0%]12	0[1 1[1 2[1 3[1 4[1 5[1 6[1	L00.0%] L00.0%] L00.0%] L00.0%] L00.0%] L00.0%] L00.0%]
MemE			01001 201					J 351										Tasks:	56, 111	15 thr		; 1	28 run	ning										
Swp[127.6	2 125.													
																			6 days	, 06:1	10:58													
Main	I/0																																	
PIE	USER	PRI N	I VIRT	RES	SHR	S CI	PU%-	EM% 7	TIME+	Com	and																							
338533	wpineros	5 20 (0 9186M	244M	3708	R 1266	8.1	0.1 49	h03:0	3 ./c	oning_	def_ic	c.exe																					
338592		20	9186M	244M	3708	R 10	0.7	0.1 22:	: 57.6	0 ./c	oning	_def_ic	c.exe																					
338545		20	9180M	244M	3708	R 10	0.1 0.1	0.1 23:	·03.7	·/c	oning	def_ic	c.exe																					
338549		20	9186M	244M	3708	R 10	0.1	0.1 22	:59.6	1 ./c	oning	def ic	c.exe																					
338556			9186M		3708				:03.2	0 ./c	oning																							
338552																																		
338556																																		
338562			9186M	244M	3708	R 10	0.1	0.1 23	:02.4	9 ./c																								
338569		20	9186M	244M	3708	R 10	0.1	0.1 23:	:03.1	3 ./c			c.exe																					
338584		20	9186M	244M	3708	R 10	0.1	0.1 22:	:59.1	2 ./C	oning	def_10	c.exe																					
330505		20	0196M	244M	3700	P 10	0.1 A 1	0.1 22	· 07 7	R /C	oning	det in	C. exe																					
338591		20	9186M	244M	3708	R 10	0.1	0.1 23	:00.3	2./c	oning	def ic	c.exe																					
338593					3708		0.1	0.1 22		0 ./c																								
338595																																		
338597																																		
338598				244M			0.1	0.1 22	:58.7																									
338601		20	9186M	244M	3708	R 10	0.1	0.1 22:	:53.4	3 ./c																								
338602		20	9186M	244M	3708	R 10	0.1	0.1 22:	. 52.9	5 ./c	oning	der_10	c.exe																					
338604		20	9186M	244M	3708	R 10	0.1	0.1 23:	-56 8	3 ./ 6	oning	def ic	c.exe																					
338609		20	9186M	244M	3708	R 10	0.1	0.1 23	:03.8	2 ./c	oning	def ic	C. exe																					
338610		20	9186M	244M	3708	R 10	0.1	0.1 23	:00.4	a ./c	oning	def ic																						
338611											oning																							
338613		20	9186M	244M	3708	R 10	0.1	0.1 22	:59.1	7 ./c	onina	def ic	c.exe																					





Monitor your jobs - check the efficient usage of resources

Now it is your turn:

- 1. Go to the monitor folder
- 2. Submit the launcher stress.sh script inside it
- 3. Find out what is your job id: sq
- 4. Use sjoin JOB-ID to go on the worker node of your job
- 5. Use the htop command (optional: press u and select your user to see what is happening for your user)
- 6. Exit by pressing q or CTRL+C
- 7. Exit the worker node and go back to the access node via CTRL+D





Monitor your jobs - cancel a job

- 1. Use the following command: scancel JOB-ID to cancel a specific job
- 2. Use the following command: scancel -u username to cancel all your jobs

Please cancel all your jobs with scancel -u username before next exercise





Monitor your jobs - cancel a job

Now it is your turn:

- 1. Go to the monitor folder
- 2. Submit the launcher stress-toolong.sh script inside it
- 3. Find out what is your job id
- 4. Cancel the job via the scancel command
- 5. Ensure your job is no longer running with sq

Reminder: scancel JOB-ID to cancel a specific job





Example 1

Here we can see that all 128 cores look very busy (100%) and we can see the load average is high. We can also see that the memory usage is quite low. Good usage of ULHPC resource for a CPU bound job.

0[1[2[3[4[5[7[7[Swp[100.0%]] 100.0%] 100.0%] 100.0%] 100.0%] 100.0%] 100.0%]	8[10 9[10 10[10 11[10 11[10 12[10 12[10 13[10 14[10	0.0%] 16 0.0%] 17 0.0%] 18 0.0%] 19 0.0%] 20 0.0%] 21 0.0%] 22 0.0%] 23 0.0%] 23	[10 [10 [10 [10 [10 [10 [10	0.0%] 0.0%] 0.0%] 0.0%] 0.0%] 0.0%] 0.0%]	24[25[26[27[28[30[31[100.0 100.0 100.0 100.0 100.0 100.0 100.0 100.0	6] 32[6] 33[6] 34[6] 35[7] 36[7] 36[7] 38[7] 39[10 10 10 10 10 10	30.0% 30.0% 30.0% 30.0% 30.0% 30.0%	40[41[42[43[44[45[46[47[100.0 100.0 100.0 100.0 100.0 100.0 100.0	 *] 48[*] 49[*] 50[*] 51[*] 52[*] 53[*] 54[*] 55[100. 100. 100. 100. 100. 100. 100.	8%] 56 8%] 57 8%] 58 8%] 59 8%] 69 8%] 69 8%] 61 8%] 63	6[7[1 8[1 9[1 0[1 2[1 2[1 3[1	99.4%] 00.0%] 00.0%] 00.0%] 00.0%] 00.0%] 00.0%] 00.0%]]	<pre>64[64[65[66[67[68[69[70[70[70[71[Task Load Upti</pre>	()))10 ()10 (0.0%] 0.0%] 0.0%] 0.0%] 0.0%] 0.0%] 0.0%] 0.0%] 0.0%] 0.0%] 0.0%] 0.0%] 0.0%]	72[73[74[75[77[78[79[5 thr 127.6 , 06:	[1 [1 [1 [1 [1 [1 [1 [100.0% 100.0% 100.0% 100.0% 100.0% 100.0% 100.0% 100.0% 25.97 58	80[81[82[83[84[85[86[87[; 12 100.38	100 100 100 100 100 100 100 8 run	.0%] .0%] .0%] .0%] .0%] .0%] .0%] ning	88[89[90[91[92[93[94[95[100 100 100 100 100 100 100	0%] 9 0%] 9 0%] 9 0%] 9 0%]10 0%]10 0%]10 0%]10	6[7[8[9[0[1[2[3[100.0 100.0 100.0 100.0 100.0 100.0 100.0	%]104[%]105[%]106[%]107[%]108[%]108[%]110[%]111[1 1 1 1 1 1	00.0% 00.0% 00.0% 00.0% 00.0% 00.0% 00.0%	3]112[3]113[3]114[3]114[3]115[3]116[3]117[4]118[3]119[100 100 100 100 100 100	0.0%]1 0.0%]1 0.0%]1 0.0%]1 0.0%]1 0.0%]1 0.0%]1 0.0%]1 0.0%]1 0.0%]1	20[22[23[23[24[25[26[27[100.0%] 100.0%] 100.0%] 100.0%] 100.0%] 100.0%] 100.0%]	
Main	I/0																																							1
PID	USER	PRI N	I VIRT	RES	SHR	S (CPU%-M	EM%	TIME+	- Cor	nmand																													
338533	wpineros	5 20	0 9186M	244M	3708	R 126	68.1	0.1 49	h03:0	3 ./	loning		cc.exe																											
338592					3708		00.7	0.1 22	:57.6																															
338543																																								1
338545				244M			00.1	0.1 23	:03.7																															1
338549				244M	3708		00.1	9.1 22	:59.6																															1
338550				244M			00.1	0.1 23	:03.2																															1
338552				244M	3708		00.1	0.1 23	:03.5																															1
338556			9186M	244M	3708	R 10	00.1	0.1 22	:54.7	/9 ./0																														1
338562			9186M	244M	3708	R 10	00.1	0.1 23	:02.4	19 ./(i	cc.exe																											1
338569		20	9186M	244M	3708	R 10	00.1	0.1 23	:03.1			11	cc.exe																											1
338582		20	9186M	244M	3708	R 10	00.1	0.1 22	:59.1	12 ./		11	cc.exe																											1
338585		20	9186M	244M	3708	R 10	00.1	0.1 22	59.9	1 ./		1																												1
338587		20	9180M	24411	3708	K II	00.1	0.1 23	107.2		cioning	1	cc.exe																											1
338593		20	9180M	244M	3708	R 11	00.1	0.1 23 0 1 22	.00.3	0 /	loning		cc.exe																											1
338505		20	0186M	244P	3708	D 10	00.1	a 1 22	.50 /	18 /	loning		CC. eve																											1
338507		20	0186M	244M	3708	D 10	99.1	a 1 22	-53 7	16	loning	def i	CC AVA																											1
338598		20	9186M	244M	3708	R 10	00.1	0.1 22	158.7		loning	def i	CC. exe																											1
338601		20	9186M	244M	3708	R 10	00.1	0.1 22	:53.4		lonin	i def i	CC. exe																											1
338602			9186M	244M	3708		00.1	0.1 22	: 52.9			def i																												1
338604			9186M		3708	R 10	00.1	0.1 23	:04.8	33 ./		def i																												1
338605					3708		00.1	0.1 22	:56.8			def i																												Γ.
338609					3708		00.1		:03.8			defi																												Γ.
338610							00.1																																	Γ.
338611							00.1																																	Γ.
338613	6	20	9186M	244M	3708	R 10	00.1	0.1 22	:59.1	17 ./	loning	i def i	cc.exe																											Γ.





Example 2

Here we can see that not all the cores are used and that the memory is not used much. It is likely that this job could be optimized. In case of doubt, please contact us <u>by opening a ticket</u>.

0[85.8] 8] 16[100.0%] 24[] 32[10	00.0%] 40[] 48[1	00.0%] 56[] 64[10	30 .0%] 72[] 80[10	00. 0%] 88[] 96[1	00 .0%]104[]112[1	00.0%]120[
1[100.0%] 9	[] 17[]	100.0%] 25[] 33[10	30.0%] 41[] 49[1	00.0%] 57[] 65[10	30 .0%] 73[] 81[10	00. 0%] 89[] 97[1	00. <mark>0%]105[</mark>]113[1	00.0%]121[
2[100.0%] 10	[] 18[100.0%] 26[] 34[10	00.0%] 42[] 50[1	00.0%] 58[] 66[10	30 .0%] 74[] 82[10	00. 0%] 90[] 98[1	00. 0%]106[]114[1	00.0%]122[
3[100.0%] 11	[] 19[]	100.0%] 27[] 35[10	30.0%] 43[] 51[1	00.0%] 59[] 67[] 75[] 83[] 99[]107[]115[]123[
4[] 12	[] 20[] 28[] 36[] 44[] 52[] 60[] 68[] 76[] 84[] 92[]100[]108[]116[]124[
5[] 13	[] 21[] 29[] 37[] 45[] 53[] 69[] 77[] 85[] 93[]101[]109[]117[]125[
6[] 14	[] 22[] 30[] 38[] 46[] 54[] 62[] 70[] 78[] 86[] 94[]102[]110[]118[]126[
7[] 15	[]] 23[] 31[] 39[] 47[] 55[] 63[] 79[] 87[] 95[]103[]111[]119[]127[
Mem[] Tasks: 91	L, 1265 thr		nning					
Swp[rage: 32.71 31.9							
							Uptime: 5	5 days, 05:56:57							





Example 3

Don't be that person :)

0[] 8[]] 16[] 24[] 32[] 40[] 48[] 56[] 64[] 72[] 80[] 88[] 96[]104[]112[]120[]	1
] 9[] 17[] 25[] 33[] 41[] 49[] 57[] 65[] 73[] 81[] 89[] 97[]105[]113[]121[1
] 10[] 18[] 26[] 34[] 50[] 58[] 66[] 74[] 82[] 90[] 98[]106[]114[]122[1
] 27[] 67[] 91[] 99[]107[
4[] 20[] 28[] 36[] 44[88.2%] 52[] 60[] 68[] 76[] 84[] 92[]100[]108[]116[]124[1
] 21[] 29[] 37[] 69[] 85[] 93[]101[]109[]117[]125[1
6[] 14[] 22[] 30[] 38[] 46[] 54[] 62[] 70[] 78[] 86[] 94[]102[]110[]118[]126[1
] 15[] 23[] 31[] 39[] 55[] 63[] 71[] 79[] 87[] 95[]103[]111[]119[]127[1
1em[] Tasks: 54, 1541 thr ; 2 running									
Swp[11111]] Load average: 0.94 1.01 3.57																
	Uptime: 12 days, 02:55:06															





Monitor your jobs - check the efficient usage of resources - GPU case

- Use the following command: sjoin JOB-ID to connect to your worker's job
- 2. Type nvidia-smi to check the GPU usage (computing and memory)

Solution States Now it's your turn:

GPU nodes are rare and in high demand, and we are too many so no practical session, sorry

+ Proce GPU 	esses: GI ID	CI ID	PID T	уре	Process name		GPU Memory Usage				
0 2 3	N/A N/A N/A	N/A N/A N/A	4040853 3063370 3246580	C C C	python /opt/conda/bin/pyt python	non3	22662MiB 476MiB 8526MiB				
(@iris-191 ~](3182672 N/T/CN)\$ nvidia-smi Tue Jul 18 16:04:49 2023											
NVID]	A-SMI	525.8	5.12 Drive	er Ve	ersion: 525.85.12	CUDA Versio	on: 12.0				
GPU Fan 	Name Temp	Perf	Persistence- Pwr:Usage/Ca	-M E Ip I	Bus-Id Disp.A Memory-Usage	Volatile GPU-Util 	Uncorr. ECC Compute M. MIG M.				
====== 0 N/A 	Tesla 61C	V100- P0	SXM2 On 127W / 300W	=+== 	22673MiB / 32768MiB	 99% 	0 Default N/A				
1 N/A	Tesla 35C	V100- P0	-SXM2 On 43W / 300W	e 	00000000:1C:00.0 Off 3MiB / 32768MiB	 0%	0 Default N/A				
2 N/A	Tesla 38C	V100- P0	-SXM2 On 58W / 300W	e 	00000000:1D:00.0 Off 479MiB / 32768MiB	 7% 	0 Default N/A				
3 N/A 	Tesla 66C	V100- P0	-SXM2 On 291W / 300W	e 	00000000:1E:00.0 Off 8535MiB / 32768MiB	 99% 	0 Default N/A				
+ Proce GPU 	sses: GI ID	CI ID	PID T	 уре	Process name		+ GPU Memory Usage				
= 0 2 3	N/A N/A N/A	N/A N/A N/A N/A	4040853 3063370 3246580		python /opt/conda/bin/pyt python	non3	======================================				





Storage

- Types of storage
- Different storage quotas
- Pricing





Storage

We offer different storage services:

- Home: this storage is personal to each user. When connecting to the ULHPC, you land in your home storage. The location should look like this: /home/users/your-username
- Project: project storage are meant to store / share files for a specific project. Multiple users can have access to a project space. The location starts with /work/projects/project-name
- Scratch: special storage for temporary files. The location starts with /scratch/users/your-username.

Important note on storage

ULHPC storage is shared and costly. It is meant for running computation only and should not be used as a long term solution. We cannot backup everything and we do not guarantee the long term safety of your storage.



Storage - quota - price

Storage quota and pricing

- Home: free, 500G quota, no possible extension
- Scratch: free, 10T quota, no possible extension
- Project: 1T free, 0.02€ (excl. VAT) / GB / Month above the free 1T

Note

You can check your current quota usage with the following command df-ulhpc

Note 2

Additionally to the storage size quota, the is a number of files quota (referred as inodes quota), e.g., you cannot have as many files as you want. you can check this quota usage with the following command df-ulhpc -i





Storage - quota

Let's see an example of df-ulhpc

	df-u <mark>~></mark> df-ulhpc Directory 	Used	Soft quota	Hard quota	Grace period
Your home	/home/users/	339.4G	500G	550G	none
and scratch	/mnt/lscratch/	40.56G	10T	11T	none
	/work/projects/adhoc	0	1000G	1.074T	none
Your projects	/work/projects/cplex	0	16M	16M	none
	/work/projects/hpcbenchs	6.011G	10T	10T	none





Storage - quota - price

Let's see an example of df-ulhpc -i



Now it's your turn: try df-ulhpc and df-ulhpc -i





Storage - quota

<pre> df-ulhpc −i Directory</pre>	Used	Soft quota	Hard quota	Grace period
/home/users/	891633	1000000	1100000	none
/mnt/lscratch/	301644	1000000	1100000	none
/work/projects/adhoc	3	1000000	1100000	none
/work/projects/cplex	1	0	0	none
/work/projects/hpcbenchs	30283	30000000	31000000	none

Soft quota is the quota you should respect

Hard quota is slightly above the soft quota, the system will prevent you to go above

Grace period is the remaining duration you have when you are between the soft and the hard quota.

Example: if the grace period states "1 day" you can still create / modify files while being above the soft quota. After the grace period is expired, you will be blocked until you fix the situation.



Storage - transfer

To transfer data from and to the ULHPC you can:

- Use MobaXterm file transfer feature, see our documentation
- Use rsync to synchronise a source directory with a destination directory, see our documentation





I want to know more

- Use virtual environments (R / Python / Conda)
 - Why? Compartmentalize your experimental setups, promotes reproducibility
 - $R \rightarrow try packrat$
 - Python \rightarrow try <u>venv</u>
 - Python things but also other non-Python stuffs \rightarrow try conda
- Even more reproducibility? Containers
- If you use interactive job, use <u>tmux</u> to prevent losing your current terminal state
- Use GNU parallel to efficiently run embarrassingly parallel jobs, see tutorial
- Check our <u>tutorials</u>, maybe there is something that you need





What can I do to help the ULHPC?

When you submit a FNR project, include a budget for HPC resources

- It helps us to buy new hardware
- Link to our estimators
- If you need help with our estimators, contact us via service now

When you submit jobs for your **paid** project, do not forget to link your jobs to your project as follows

- See our documentation about this







