



High Performance  
Computing &  
Big Data Services



[hpc.uni.lu](http://hpc.uni.lu)



[hpc@uni.lu](mailto:hpc@uni.lu)



[@ULHPC](https://twitter.com/ULHPC)

# HPC School - Beginner

S2 - Work on the ULHPC



# Important references!



University HPC web site: <https://hpc.uni.lu/>

Status of the HPC platforms: <https://hpc.uni.lu/live-status/motd>

Technical documentation: <https://hpc-docs.uni.lu/>

Github repository of the Advanced Tutorials: <https://github.com/ULHPC>

# 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 of 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)

- **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

```
=====
```

AION-Cluster

Open source AI framework for distributed training and inference

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/mbarborini/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  $7/28$ th ( $\frac{1}{4}$ ) 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

## Exercise 1

Request 8 cores for 60 minutes

Check the worker node name

Close your interactive session to deallocate the resources

## Exercise 2

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

## Exercise 3

Can you book the same amount of cores on an IRIS CPU node than the answer of exercise 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



## Exercise 1

Request 8 cores for 60 minutes

**Solution:** `si -c8 -t60 --reservation=school-interactive`

## Exercise 2

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

**Solution:** `si -c32 -t120 --reservation=school-interactive`

## Exercise 3

Can you book the same amount of cores on an IRIS CPU node than the answer of exercise 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
```

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

# Batch jobs - 🏗️ now it is your turn

## Exercise 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`

## Exercise 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

## Exercise 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 → 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.

Type	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



Type	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 → [check our tutorial](#)
- Containers → [check our tutorial](#)
- Use Jupyter Notebook, Abaqus CAE, Matlab or Stata via a GUI → [check our portal](#)
- Compile your own program → too advanced for this tutorial

## Note

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

# Modules



## 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.



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

# Modules

## 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/resif/iris-rhel8/2020b/broadwell/modules/all -----
bio/TopHat/2.1.2-GCC-10.2.0-Python-2.7.18
chem/spglib-python/1.16.0-foss-2020b
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)
lang/SciPy-bundle/2020.11-foss-2020b-Python-2.7.18
lib/Boost.Python/1.74.0-GCC-10.2.0

Where:
D: Default Module
```



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

# 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/broadwell/modules/all -----
bio/TopHat/2.1.2-GCC-10.2.0-Python-2.7.18
chem/spglib-python/1.16.0-foss-2020b
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)
lang/SciPy-bundle/2020.11-foss-2020b-Python-2.7.18
lib/Boost.Python/1.74.0-GCC-10.2.0

Where:
D: Default Module
```



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

# 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/1CN)$ module list
No modules loaded
0 [jschleich@iris-056 ~](3174323 1N/T/1CN)$ module load lang/Python
0 [jschleich@iris-056 ~](3174323 1N/T/1CN)$ module list

Currently Loaded Modules:
  1) compiler/GCCcore/10.2.0           7) lang/Tcl/8.6.10-GCCcore-10.2.0
  2) lib/zlib/1.2.11-GCCcore-10.2.0   8) devel/SQLite/3.33.0-GCCcore-10.2.0
  3) tools/binutils/2.35-GCCcore-10.2.0 9) tools/XZ/5.2.5-GCCcore-10.2.0
  4) tools/bzip2/1.0.8-GCCcore-10.2.0 10) math/GMP/6.2.0-GCCcore-10.2.0
  5) devel/ncurses/6.2-GCCcore-10.2.0 11) lib/libffi/3.3-GCCcore-10.2.0
  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/>

# 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
```

 Now it is your turn:

- 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



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



# Monitor your jobs



## 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

## 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 [jschleich@access1 ~]$ squeue -u djoubaud
```

JOBID	PARTIT	QOS	NAME	USER	NODE	CPUS	ST	TIME	TIME_LEFT	PRIORITY	NODELIST(REASON)
846027_1	batch	normal	DatasetsGeneration	djoubaud	1	4	PD	0:00	1-23:59:00	11410	(QOSMaxJobsPerUserLimit)
846037_1	batch	normal	DatasetsGeneration	djoubaud	1	4	PD	0:00	1-23:59:00	11410	(QOSMaxJobsPerUserLimit)
846036_1	batch	normal	DatasetsGeneration	djoubaud	1	4	PD	0:00	1-23:59:00	11410	(QOSMaxJobsPerUserLimit)
846035_1	batch	normal	DatasetsGeneration	djoubaud	1	4	PD	0:00	1-23:59:00	11410	(QOSMaxJobsPerUserLimit)
846034_1	batch	normal	DatasetsGeneration	djoubaud	1	4	PD	0:00	1-23:59:00	11410	(QOSMaxJobsPerUserLimit)
846033_1	batch	normal	DatasetsGeneration	djoubaud	1	4	PD	0:00	1-23:59:00	11410	(QOSMaxJobsPerUserLimit)
846027_7	batch	normal	DatasetsGeneration	djoubaud	1	4	R	9:06	1-23:49:54	11408	aion-0163
846027_6	batch	normal	DatasetsGeneration	djoubaud	1	4	R	12:12	1-23:46:48	11407	aion-0113
846027_5	batch	normal	DatasetsGeneration	djoubaud	1	4	R	32:42	1-23:26:18	11403	aion-0078
846027_1	batch	normal	DatasetsGeneration	djoubaud	1	4	R	57:38	1-23:01:22	11398	aion-0163
846027_0	batch	normal	DatasetsGeneration	djoubaud	1	4	R	59:36	1-22:59:24	11398	aion-0163
844973_7	batch	normal	DatasetsGeneration	djoubaud	1	4	R	1-08:57:50	15:01:10	11375	aion-0019

# Monitor your jobs



## 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



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



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

## Monitor your jobs - check the efficient usage of resources

```

0[||||100.0%] 0[||||100.0%] 16[||||100.0%] 24[||||100.0%] 32[||||100.0%] 40[||||100.0%] 48[||||100.0%] 56[||||99.4%] 64[||||100.0%] 72[||||100.0%] 80[||||100.0%] 88[||||100.0%] 96[||||100.0%] 104[||||100.0%] 112[||||100.0%] 120[||||100.0%]
1[||||100.0%] 1[||||100.0%] 17[||||100.0%] 25[||||100.0%] 33[||||100.0%] 41[||||100.0%] 49[||||100.0%] 57[||||100.0%] 65[||||100.0%] 73[||||100.0%] 81[||||100.0%] 89[||||100.0%] 97[||||100.0%] 105[||||100.0%] 113[||||100.0%] 121[||||100.0%]
2[||||100.0%] 18[||||100.0%] 18[||||100.0%] 26[||||100.0%] 34[||||100.0%] 42[||||100.0%] 50[||||100.0%] 58[||||100.0%] 66[||||100.0%] 74[||||100.0%] 82[||||100.0%] 90[||||100.0%] 98[||||100.0%] 106[||||100.0%] 114[||||100.0%] 122[||||100.0%]
3[||||100.0%] 11[||||100.0%] 19[||||100.0%] 27[||||100.0%] 35[||||100.0%] 43[||||100.0%] 51[||||100.0%] 59[||||100.0%] 67[||||100.0%] 75[||||100.0%] 83[||||100.0%] 91[||||100.0%] 99[||||100.0%] 107[||||100.0%] 115[||||100.0%] 123[||||100.0%]
4[||||100.0%] 12[||||100.0%] 20[||||100.0%] 28[||||100.0%] 36[||||100.0%] 44[||||100.0%] 52[||||100.0%] 60[||||100.0%] 68[||||100.0%] 76[||||100.0%] 84[||||100.0%] 92[||||100.0%] 100[||||100.0%] 108[||||100.0%] 116[||||100.0%] 124[||||100.0%]
5[||||100.0%] 13[||||100.0%] 21[||||100.0%] 29[||||100.0%] 37[||||100.0%] 45[||||100.0%] 53[||||100.0%] 61[||||100.0%] 69[||||100.0%] 77[||||100.0%] 85[||||100.0%] 93[||||100.0%] 101[||||100.0%] 109[||||100.0%] 117[||||100.0%] 125[||||100.0%]
6[||||100.0%] 14[||||100.0%] 22[||||100.0%] 30[||||100.0%] 38[||||100.0%] 46[||||100.0%] 54[||||100.0%] 62[||||100.0%] 70[||||100.0%] 78[||||100.0%] 86[||||100.0%] 94[||||100.0%] 102[||||100.0%] 110[||||100.0%] 118[||||100.0%] 126[||||100.0%]
7[||||100.0%] 15[||||100.0%] 23[||||100.0%] 31[||||100.0%] 39[||||100.0%] 47[||||100.0%] 55[||||100.0%] 63[||||100.0%] 71[||||100.0%] 79[||||100.0%] 87[||||100.0%] 95[||||100.0%] 103[||||100.0%] 111[||||100.0%] 119[||||100.0%] 127[||||100.0%]

Mem[|||||||||]
Swp[|||||]

Tasks: 56, 1115 thr
Load average: 127.62 125.97 100.38
Uptime: 6 days, 06:10:58

Main I/O
PID USER PRI NI VIRT RES SHR S CPU%-MEM% TIME+ Command
338533 wpineros 20 0 9186M 244M 3708 R 12668.1 0.1 49h03:03 ./cloning_def_icc.exe
338592 20 9186M 244M 3708 R 100.7 0.1 22:57.60 ./cloning_def_icc.exe
338543 20 9186M 244M 3708 R 100.1 0.1 23:05.23 ./cloning_def_icc.exe
338545 20 9186M 244M 3708 R 100.1 0.1 23:02.74 ./cloning_def_icc.exe
338549 20 9186M 244M 3708 R 100.1 0.1 22:59.61 ./cloning_def_icc.exe
338550 20 9186M 244M 3708 R 100.1 0.1 23:03.20 ./cloning_def_icc.exe
338552 20 9186M 244M 3708 R 100.1 0.1 23:03.50 ./cloning_def_icc.exe
338556 20 9186M 244M 3708 R 100.1 0.1 22:54.79 ./cloning_def_icc.exe
338562 20 9186M 244M 3708 R 100.1 0.1 23:02.49 ./cloning_def_icc.exe
338569 20 9186M 244M 3708 R 100.1 0.1 23:02.13 ./cloning_def_icc.exe
338582 20 9186M 244M 3708 R 100.1 0.1 22:59.12 ./cloning_def_icc.exe
338585 20 9186M 244M 3708 R 100.1 0.1 22:59.91 ./cloning_def_icc.exe
338587 20 9186M 244M 3708 R 100.1 0.1 23:07.28 ./cloning_def_icc.exe
338591 20 9186M 244M 3708 R 100.1 0.1 23:00.32 ./cloning_def_icc.exe
338593 20 9186M 244M 3708 R 100.1 0.1 22:57.50 ./cloning_def_icc.exe
338595 20 9186M 244M 3708 R 100.1 0.1 22:59.48 ./cloning_def_icc.exe
338597 20 9186M 244M 3708 R 100.1 0.1 22:53.76 ./cloning_def_icc.exe
338598 20 9186M 244M 3708 R 100.1 0.1 22:58.77 ./cloning_def_icc.exe
338601 20 9186M 244M 3708 R 100.1 0.1 22:53.43 ./cloning_def_icc.exe
338602 20 9186M 244M 3708 R 100.1 0.1 22:52.93 ./cloning_def_icc.exe
338604 20 9186M 244M 3708 R 100.1 0.1 23:04.83 ./cloning_def_icc.exe
338605 20 9186M 244M 3708 R 100.1 0.1 22:56.83 ./cloning_def_icc.exe
338609 20 9186M 244M 3708 R 100.1 0.1 23:03.82 ./cloning_def_icc.exe
338610 20 9186M 244M 3708 R 100.1 0.1 23:00.40 ./cloning_def_icc.exe
338611 20 9186M 244M 3708 R 100.1 0.1 23:05.09 ./cloning_def_icc.exe
338613 20 9186M 244M 3708 R 100.1 0.1 22:59.17 ./cloning_def_icc.exe
  
```

# Monitor your jobs



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



## 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



## 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

# Monitor your jobs

## 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[|||||100.0%] 8[|||||100.0%] 16[|||||100.0%] 24[|||||100.0%] 32[|||||100.0%] 40[|||||100.0%] 48[|||||100.0%] 56[|||||100.0%] 64[|||||100.0%] 72[|||||100.0%] 80[|||||100.0%] 88[|||||100.0%] 96[|||||100.0%] 104[|||||100.0%] 112[|||||100.0%] 120[|||||100.0%]
1[|||||100.0%] 9[|||||100.0%] 17[|||||100.0%] 25[|||||100.0%] 33[|||||100.0%] 41[|||||100.0%] 49[|||||100.0%] 57[|||||100.0%] 65[|||||100.0%] 73[|||||100.0%] 81[|||||100.0%] 89[|||||100.0%] 97[|||||100.0%] 105[|||||100.0%] 113[|||||100.0%] 121[|||||100.0%]
2[|||||100.0%] 10[|||||100.0%] 18[|||||100.0%] 26[|||||100.0%] 34[|||||100.0%] 42[|||||100.0%] 50[|||||100.0%] 58[|||||100.0%] 66[|||||100.0%] 74[|||||100.0%] 82[|||||100.0%] 90[|||||100.0%] 98[|||||100.0%] 106[|||||100.0%] 114[|||||100.0%] 122[|||||100.0%]
3[|||||100.0%] 11[|||||100.0%] 19[|||||100.0%] 27[|||||100.0%] 35[|||||100.0%] 43[|||||100.0%] 51[|||||100.0%] 59[|||||100.0%] 67[|||||100.0%] 75[|||||100.0%] 83[|||||100.0%] 91[|||||100.0%] 99[|||||100.0%] 107[|||||100.0%] 115[|||||100.0%] 123[|||||100.0%]
4[|||||100.0%] 12[|||||100.0%] 20[|||||100.0%] 28[|||||100.0%] 36[|||||100.0%] 44[|||||100.0%] 52[|||||100.0%] 60[|||||100.0%] 68[|||||100.0%] 76[|||||100.0%] 84[|||||100.0%] 92[|||||100.0%] 100[|||||100.0%] 108[|||||100.0%] 116[|||||100.0%] 124[|||||100.0%]
5[|||||100.0%] 13[|||||100.0%] 21[|||||100.0%] 29[|||||100.0%] 37[|||||100.0%] 45[|||||100.0%] 53[|||||100.0%] 61[|||||100.0%] 69[|||||100.0%] 77[|||||100.0%] 85[|||||100.0%] 93[|||||100.0%] 101[|||||100.0%] 109[|||||100.0%] 117[|||||100.0%] 125[|||||100.0%]
6[|||||100.0%] 14[|||||100.0%] 22[|||||100.0%] 30[|||||100.0%] 38[|||||100.0%] 46[|||||100.0%] 54[|||||100.0%] 62[|||||100.0%] 70[|||||100.0%] 78[|||||100.0%] 86[|||||100.0%] 94[|||||100.0%] 102[|||||100.0%] 110[|||||100.0%] 118[|||||100.0%] 126[|||||100.0%]
7[|||||100.0%] 15[|||||100.0%] 23[|||||100.0%] 31[|||||100.0%] 39[|||||100.0%] 47[|||||100.0%] 55[|||||100.0%] 63[|||||100.0%] 71[|||||100.0%] 79[|||||100.0%] 87[|||||100.0%] 95[|||||100.0%] 103[|||||100.0%] 111[|||||100.0%] 119[|||||100.0%] 127[|||||100.0%]

Mem[|||||100.0%]
Swap[|||||100.0%]

Tasks: 56, 1115 thr          : 128 running
Load average: 127.62 125.97 100.38
Uptime: 6 days, 06:10:58

Main 1/0
PID USER      PRI NI  VIRT   RES   SHR  % CPU% MEM%  TIME+  Command
338533 wpineros    20  0 918M 244M 3708 R 12668.1 0.1 49h03:03 ./cloning_def_icc.exe
338592          20  0 918M 244M 3708 R 100.7 0.1 22:57.60 ./cloning_def_icc.exe
338543          20  0 918M 244M 3708 R 100.1 0.1 23:05.23 ./cloning_def_icc.exe
338545          20  0 918M 244M 3708 R 100.1 0.1 23:03.74 ./cloning_def_icc.exe
338549          20  0 918M 244M 3708 R 100.1 0.1 22:59.61 ./cloning_def_icc.exe
338550          20  0 918M 244M 3708 R 100.1 0.1 23:03.20 ./cloning_def_icc.exe
338552          20  0 918M 244M 3708 R 100.1 0.1 23:03.50 ./cloning_def_icc.exe
338556          20  0 918M 244M 3708 R 100.1 0.1 22:54.79 ./cloning_def_icc.exe
338562          20  0 918M 244M 3708 R 100.1 0.1 23:02.49 ./cloning_def_icc.exe
338569          20  0 918M 244M 3708 R 100.1 0.1 23:03.13 ./cloning_def_icc.exe
338582          20  0 918M 244M 3708 R 100.1 0.1 22:59.12 ./cloning_def_icc.exe
338585          20  0 918M 244M 3708 R 100.1 0.1 22:59.91 ./cloning_def_icc.exe
338587          20  0 918M 244M 3708 R 100.1 0.1 23:07.28 ./cloning_def_icc.exe
338591          20  0 918M 244M 3708 R 100.1 0.1 23:00.32 ./cloning_def_icc.exe
338593          20  0 918M 244M 3708 R 100.1 0.1 22:57.50 ./cloning_def_icc.exe
338595          20  0 918M 244M 3708 R 100.1 0.1 22:59.48 ./cloning_def_icc.exe
338597          20  0 918M 244M 3708 R 100.1 0.1 22:53.76 ./cloning_def_icc.exe
338598          20  0 918M 244M 3708 R 100.1 0.1 22:58.77 ./cloning_def_icc.exe
338601          20  0 918M 244M 3708 R 100.1 0.1 22:53.43 ./cloning_def_icc.exe
338602          20  0 918M 244M 3708 R 100.1 0.1 22:52.93 ./cloning_def_icc.exe
338604          20  0 918M 244M 3708 R 100.1 0.1 23:04.83 ./cloning_def_icc.exe
338605          20  0 918M 244M 3708 R 100.1 0.1 22:56.83 ./cloning_def_icc.exe
338609          20  0 918M 244M 3708 R 100.1 0.1 23:03.82 ./cloning_def_icc.exe
338610          20  0 918M 244M 3708 R 100.1 0.1 23:00.40 ./cloning_def_icc.exe
338611          20  0 918M 244M 3708 R 100.1 0.1 23:05.09 ./cloning_def_icc.exe
338613          20  0 918M 244M 3708 R 100.1 0.1 22:59.17 ./cloning_def_icc.exe

```

# Monitor your jobs

## 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 [by opening a ticket](#).

```

0[|||||85.8 ] 8[          ] 16[|||||100.0%] 24[          ] 32[|||||100.0%] 40[          ] 48[|||||100.0%] 56[          ] 64[|||||100.0%] 72[          ] 80[|||||100.0%] 88[          ] 96[|||||100.0%] 104[          ] 112[|||||100.0%] 120[          ]
1[|||||100.0%] 9[          ] 17[|||||100.0%] 25[          ] 33[|||||100.0%] 41[          ] 49[|||||100.0%] 57[          ] 65[|||||100.0%] 73[          ] 81[|||||100.0%] 89[          ] 97[|||||100.0%] 105[          ] 113[|||||100.0%] 121[          ]
2[|||||100.0%] 10[          ] 18[|||||100.0%] 26[          ] 34[|||||100.0%] 42[          ] 50[|||||100.0%] 58[          ] 66[|||||100.0%] 74[          ] 82[|||||100.0%] 90[          ] 98[|||||100.0%] 106[          ] 114[|||||100.0%] 122[          ]
3[|||||100.0%] 11[          ] 19[|||||100.0%] 27[          ] 35[|||||100.0%] 43[          ] 51[|||||100.0%] 59[          ] 67[          ] 75[          ] 83[          ] 91[          ] 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[          ] 61[          ] 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[          ] 71[          ] 79[          ] 87[          ] 95[          ] 103[          ] 111[          ] 119[          ] 127[          ]
Mem[|||||          ]
Swp[|||||          ]
Tasks: 91, 1265 thr          ; 31 running
Load average: 32.71 31.91 31.83
Uptime: 5 days, 05:56:57
  
```

# Monitor your jobs

## 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[ ]
2[      ] 10[ ] 18[ ] 26[ ] 34[ ] 42[ ] 50[ ] 58[ ] 66[ ] 74[ ] 82[ ] 90[ ] 98[ ] 106[ ] 114[ ] 122[ ]
3[      ] 11[ ] 19[ ] 27[ ] 35[ ] 43[ ] 51[ ] 59[ ] 67[ ] 75[ ] 83[ ] 91[ ] 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[ ] 61[ ] 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[ ] 71[ ] 79[ ] 87[ ] 95[ ] 103[ ] 111[ ] 119[ ] 127[ ]

Mem[|||||]
Swp[|||||]

] Tasks: 54, 1541 thr ; 2 running
] Load average: 0.94 1.01 3.57
] Uptime: 12 days, 02:55:06
  
```

# Monitor your jobs

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

1. 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)

👉 Now it's your turn:

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

```
Processes:
GPU  GI  CI      PID  Type  Process name      GPU Memory
   ID ID      ID                Usage
=====
  0   N/A N/A    4040853  C    python             22662MiB
  2   N/A N/A    3063370  C    /opt/conda/bin/python3  476MiB
  3   N/A N/A    3246580  C    python             8526MiB

@iris-191 ~](3182672 N/T/CN)$ nvidia-smi
Tue Jul 18 16:04:49 2023

NVIDIA-SMI 525.85.12      Driver Version: 525.85.12      CUDA Version: 12.0

GPU  Name                Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC
 Fan  Temp    Perf   Pwr:Usage/Cap|      Memory-Usage | GPU-Util  Compute M.
                                MIG M.
=====+=====
  0   Tesla V100-SXM2...  On          | 00000000:1A:00:0 Off |    0          0
 N/A   61C    P0   127W / 300W | 22673MiB / 32768MiB |    99%    Default
                                N/A
  1   Tesla V100-SXM2...  On          | 00000000:1C:00:0 Off |    0          0
 N/A   35C    P0    43W / 300W |   3MiB / 32768MiB |     0%    Default
                                N/A
  2   Tesla V100-SXM2...  On          | 00000000:1D:00:0 Off |    0          0
 N/A   38C    P0    58W / 300W | 479MiB / 32768MiB |     7%    Default
                                N/A
  3   Tesla V100-SXM2...  On          | 00000000:1E:00:0 Off |    0          0
 N/A   66C    P0   291W / 300W | 8535MiB / 32768MiB |    99%    Default
                                N/A

Processes:
GPU  GI  CI      PID  Type  Process name      GPU Memory
   ID ID      ID                Usage
=====
  0   N/A N/A    4040853  C    python             22662MiB
  2   N/A N/A    3063370  C    /opt/conda/bin/python3  476MiB
  3   N/A N/A    3246580  C    python             8526MiB
```

# 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, there 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-ulhpc		Used	Soft quota	Hard quota	Grace period
Directory		-----	-----	-----	-----
Your home	/home/users/ [redacted]	339.4G	500G	550G	none
and scratch	/mnt/lscratch/	40.56G	10T	11T	none
Your projects	/work/projects/adhoc	0	1000G	1.074T	none
	/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`

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

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

# Storage - quota

```
~ df-ulhpc -i
```

Directory	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 → try [packrat](#)
  - Python → try [venv](#)
  - Python things but also other non-Python stuffs → try [conda](#)
- Even more reproducibility? Containers
- If you use interactive job, use [tmux](#) to prevent losing your current terminal state
- Use GNU parallel to efficiently run embarrassingly parallel jobs, see [tutorial](#)
- Check our [tutorials](#), 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](#)

# Thank you

