# **Computing Power: HPC Tutorial Session**



Bocconi Students for Machine Learning (BSML)

Notes by Giacomo Cirò

April 16, 2025

# Introduction

In this tutorial, we go through an example use of the high performance computing (HPC) cluster. We will cover:

- 1. Login using ssh
- 2. VPN usage
- 3. Quality of life improvements (ssh keys, ssh configuration)
- 4. Create & submit a job
- 5. Misc commands

# 1 Login Using SSH

Connect to bocconi-students Wi-Fi. Open terminal and ssh into students HPC using password:

```
[local]$ ssh bocconi_id@slnode-da.sm.unibocconi.it
bocconi_id@slnode-da.sm.unibocconi.it's password:
```

Close connection:

[hpc]\$ logout

# 2 VPN Usage

Trying to access the cluster from a different Wi-Fi network causes an error:

```
[local]$ ssh bocconi_id@slnode-da.sm.unibocconi.it
ssh: Could not resolve hostname slnode-da.sm.unibocconi.it: No such host is
known.
```

You need to first setup and connect to the FortiClient VPN.

# 3 Quality of Life (SSH Keys, SSH Config)

### 3.1 SSH Keys

We want to setup ssh keys to avoid typing the password every time.

We need to have a public ssh key, call it id\_rsa.pub copied inside the file .ssh/authorized\_keys on the HPC. The corresponding private key, call it id\_rsa, must be stored inside the .ssh/ directory on the local machine.

Whenever we try to access the HPC, the ssh protocol will send a test message encrypted with the public key. Only if we have the correct private key we are able to decode it and send it back, allowing for secure access.

We create a pair of keys using the ssh-keygen command from inside the local directory .ssh/ (add a signature to recognize the key):

```
[local]$ cd .ssh
[local]$ ssh-keygen -C "my-new-key"
```

You can either change the file name or keep the default id\_rsa.

Now print the content of the public key and copy it:



```
[local]$ type id_rsa.pub
ssh-ed25519 AAAAC3NzaC11ZDI1NTE5AAAAIIOkb/JST6a//zakM6CMT1UKbVk7FwXWu75pzH/YfP9B
my-new-key
```

Login to the HPC and paste it in .ssh/authorized\_keys (create the file if it does not exists):

```
[local]$ ssh bocconi_id@slnode-da.sm.unibocconi.it
bocconi_id@slnode-da.sm.unibocconi.it's password:
[hpc]$ cd .ssh
[hpc]$ nano authorized_keys
```

Paste the public key and save. Logout and try connecting again (it should work without password).

```
[hpc]$ logout
[local]$ ssh bocconi_id@slnode-da.sm.unibocconi.it
```

### 3.2 SSH Config

To avoid typing the entire user and host name at every ssh login, open C:/Users/name/.ssh/config and add aliases:

```
Host hpc_stud
HostName slnode-da.sm.unibocconi.it
User bocconi_id
```

Connect to hpc again:

[local]\$ ssh hpc\_stud
[hpc]\$

At this point, you should be able to login without using the password and without typing the entire user and host name. We are now ready to create and submit our first SLURM job.

# 4 Create & Submit a SLURM Job

In this section, we will walk you through all the steps needed to submit the first job from inside the HPC. First, we create an anaconda environment. Then, we create a SLURM job file to execute a demo python script. Finally, we submit the job and monitor the runtime.

#### 4.1 Create Conda Environment

Create a new anaconda environment:

```
[hpc]$ conda create -n bsml python==3.12
```

If needed, install the required dependencies:

```
[hpc]$ conda activate bsml
[hpc]$ python -m pip install torch
...
```

### 4.2 Create a Python Script

Create a new .py file using the Linux command nano:

[hpc]\$ nano bsml.py



An interactive text editor will pop-up. Add your code inside:

```
import time
for i in range(10):
    print(f"{i}) Connecting Minds, Not Only Layers")
    time.sleep(2)
```

### 4.3 Create a SLURM Job Script

Again, use nano to create a .sh file:

```
[hpc]$ nano bsml.sh
```

Fill it using the SLURM syntax:

```
#!/bin/bash
#SBATCH --job-name="bsml"
#SBATCH --account=bocconi_id
#SBATCH --partition=stud
#SBATCH --cpus-per-task=1
#SBATCH --gpus=1
#SBATCH --mem=10GB
#SBATCH --chdir=.
#SBATCH --output=/home/bocconi_id/out.out
#SBATCH --error=/home/bocconi_id/err.err
#SBATCH --mail-type=ALL
#SBATCH --mail-user=giacomo.ciro@unibocconi.it
module load modules/miniconda3
eval "$(conda shell.bash hook)"
conda activate bsml
python bsml.py
```

module unload modules/miniconda3
echo "The end"

#### 4.4 Submit & Monitor the Job

Submit the job: [hpc]\$ sbatch bsml.sh Monitor the status of the job: [hpc]\$ squeue Cancel the job:

[hpc]\$ scancel <job\_id>

# 5 Misc Commands

We list some SLURM and Linux commands which might come handy when working on the HPC.



### 5.1 SLURM

List of past jobs

[hpc]\$ sacct

Nodes info:

[hpc]\$ sinfo

More node-specific info:

[hpc]\$ scontrol show node <node\_name>

Partition-specific info:

[hpc]\$ scontrol show partition <partition\_name>

## 5.2 Linux

List content of a directory (omit path to list current dir):

[hpc]\$ ls path/to/dir

Print content of file:

[hpc]\$ cat path/to/file

Remove file:

[hpc]\$ rm path/to/file

Remove entire directory recursively:

[hpc]\$ rm -r path/to/dir

Copy file:

[hpc]\$ cp path/to/file\_to\_copy path/to/new\_file\_name

Print end of file and refresh when new changes occur:

[hpc]\$ tail -f path/to/file

Run a command every 1 second:

[hpc]\$ watch -n 1 <cmd>

Memory info:

[hpc]\$ free -h

CPU info:

[hpc]\$ lscpu

GPU info:

[hpc]\$ nvidia-smi



### 5.3 Transfer Files

From local machine to HPC:

```
[local]$ scp /path/to/local/file bocconi_id@node-da.sm.unibocconi.it:/path/to/
    remote/directory
```

From HPC to local machine

```
[local]$ scp bocconi_id@node-da.sm.unibocconi.it:/path/to/remote/file /path/to/
local/directory
```

Use the flag -r to copy a directory recursively.

