ISPL workstations

Welcome to the ISPL workstations cluster!
We are more than 60 users using the same set of workstations. The architecture of the system is such that you can seamlessly move your work from one workstation to another simply by logging into the new workstation.
This document is intended to guide you into our system, describing how it works and providing you with a set of best practices, suggestions, and useful tools.

Architecture Overview

Generally speaking, the system is composed by Linux computation workstations and a NAS (Network Attached Storage), i.e., a computer with 90TB of redundant storage. Every computation machine has its own home path, so that the user ends up with two different folders: one home per machine and single home on NAS, mounted on every machine. This said, the workstation machine is intended for the machine-only stuff (like, bash aliases, scripts, ecce cc). Indeed, a user wants to keep his/her projects on the NAS, so that it can be safe and agile.

REMEMBER! The system is only available inside the DEIB network,
so you have to either be physically at DEIB or have a VPN connection!
The VPN has to be activated by DEIB IT guys. Ask your advisor to open a ticket!

Best practice

It is extremely important to follow these rules:

- **Occam's razor**: use the machine that best fits your requirements without overestimating them.
- **Sharing is caring**: limit the number of multiprocessing cores to your need, bearing in mind others might want to use that machine alongside you.
- **A diamond is forever, a session is not**: log out from your permanent sessions (X2Go, tmux, screen, Jupyter notebook, Matlab GUI...) when you don't need them anymore.
- **Let the machines do the job**: python, matlab and jupyter can be launched from terminal. Run your intensive and long tests during nights and weekends, not on Monday at 9am just because "this machine is free".

Moreover, we strongly ask you to put some extra-effort during your usage:

- **Be nice**: if you are doing a time-consuming but not urgent job, please set a low process priority.
  `nice -n 10 python main.py`
- **Clean up after yourself**: our NAS is big but still limited. Compress old datasets and projects; keep active only on-going projects. And please, add documentation to your stuff!
- **Keep in touch**: if you need to intensively use a machine for a large amount of time it's fine. Just please warn your colleagues.
- **Don’t be lazy**: use resources carefully, don’t waste computational time and electrical power.

Workstations Specifications

- Hack 10.79.23.8
  - i7-6700K, @ 4 GHz, 4 Cores, 8 Thread
  - 32 GB RAM
  - 1 x NVIDIA GeForce GTX 1080 Ti, 12 GB DDR5, 3584 CUDA core, 1582 MHz
- Euler 10.79.0.8
  - E5-2630 v2 @ 2.60GHz, 6 core, 12 thread
  - 128 GB RAM
  - 2 x NVIDIA QUADRO P6000, 24GB DDR5, 3840 CUDA core, 1530 MHz
- Dirac 10.79.0.10
  - i7-2600 @ 3.40GHz, 4 core, 8 thread
  - 32GB RAM
  - NVIDIA TITAN X, 12GB DDR5, 3584 CUDA core, 1417 MHz
- Curie 10.79.0.13
  - E5-1607 v2 @ 3.00GHz, 4 core, 4 thread
  - 32GB RAM
  - NVIDIA GeForce GTX 1080, 8GB DDR5, 2560 CUDA core, 1607 MHz
- Bayes 10.79.0.9
  - 2 x X5570 @ 2.93GHz, 4C, HT, 8 core, 16 thread
  - 72GB RAM
- Ampere 10.79.0.7
  - Xeon E5506 @ 2.13GHz, 4 core, 4 thread
  - 12GB RAM

Dashboard

To monitor workstations usage and resources go to:
http://10.79.0.14/netdata/develop.php
user/password: ispg/paolo

User account

Name: Mario
Surname: Rossi
username: `mrossi`
default password: `MarioRossi` (you should be asked to change it at first login)

➔ `passwd` command on terminal (it changes the password on all the workstations)

User-available paths

- `/nas/home/mrossi:` NAS user home, shared among all the workstations. Only the user has write privileges; other users can only read.
- `/home/mrossi:` Workstation specific SSD storage. Use only for volatile/scratch data
- `/nas/public/dataset:` original datasets in read-only. Ask admins to add new ones. You are encouraged to use these datasets without replicating them into your project directory, to keep the disk usage low. If you need a modified version (e.g., with preprocessing), you cannot put it here.
- `/nas/public/exchange:` user writable folder, intended for sharing projects
- `/nas/public/exchange/archive:` intented for storing interesting completed projects (like challenges, break-through studies, ecc ecc)

Virtual desktop

- OSX Client: http://code.x2go.org/releases/X2GoClient_latest_macosx.dmg
- Ubuntu client: `sudo apt install -y x2goclient`
- Remember to set LXDE!

Remote access to remote filesystem

- SFTP (FileZIlla)
- scp
- rsync

Installed Software

- Google Chrome
- Sublime Text
- MATLAB 2020b

Suggested software and routines for your convenience

- JetBrains PyCharm (python IDE). This is a super-useful software that allows you to work on a local copy of your project and automatically deploy it to the workstations. To the best of our knowledge, it is the only IDE that supports remote debugging (the codes run on a remote server, but you can see the variables on your laptop). Read the guide!
- Compress a folder: `tar cvf MyFolder.tar MyFolder`
- Decompress a folder: `tar -xvf MtFolder.tar -C MyFolder`

First configuration and bash utilities

➔ **Windows users**: install Windows Subsystem for Linux (WLS) from the Microsoft Store. It allows you to use a Linux terminal inside Windows without virtualization and other nasty stuff. It is completely transparent for Windows, and allows you to use all Linux utilities listed in this guide.

The connections to our servers are managed by SSH. Open a terminal and simply do:
`ssh mrossi@10.79.X.X`
It will ask you for the password, and then you are logged to the linux server.

Advanced utilities

For your convenience, in the following you can configure your machine to speed up the connections.
Do it after you have logged in one of the machines.
First, we generate a ssh authentication key to avoid being ask for the password everytime.
Then, create ssh aliases for the connections, in order to avoid the ugly "mrossi@IP_ADDRESS" notation.

1. Generate a unique SSH key: `ssh-keygen`
   The authentication key avoids you writing the password at every connection.
2. Create a ssh config file: `touch ./.ssh/config`
   This file will contain information on how to connect to the hosts using aliases.
3. Fill the config file as follows with `nano ./.ssh/config`
   ```
   Host ampere
       HostName 10.79.0.7
       User mrossi
   ```
   Please, substitute "mrossi" with your username.
4. Copy the above three lines for all the machines, substituting the name and the IP address. Then save the file (ctrl-X) and close it.
5. Check if the alias works with `ssh ampere`
   It will ask for your password.
   With the next step, we will copy your authentication key to the machine and at the next login you will not be forced to type the password anymore.
6. Copy it to the server: `ssh-copy-id -i ./.ssh/id_rsa ampere`
7. Check if the SSH key works with `ssh ampere`
8. Copy your SSH key to all machines repeating points 6 and 7.
   BEWARE: you cannot log in to the NAS, only admins can! Skip this machine from the configuration procedure.

Other utilities:

9. If you want togo directly to your /nas/home folder every time you open a connection:
   `echo "cd /nas/home/mrossi" >> ~/.bashrc`
10. For your convenience, you can create aliases for shortcuts. For instance, using:
    `echo "alias activ='conda activate '" >> ~/.bashrc`
    Allows you to activate the environment with: `activ my_env`

➔ If you need a persistent session (i.e., it still runs even you disconnect from that machine), use screen or tmux

Note: the ssh configuration can be used also by PyCharm; if you use Linux or Mac, just locate the `id_rsa` key file and use it for the deployment connection.
If you use Windows, you have generated the key in the Windows Subsystem for Linux that is NOT visible by Windows. So you have to copy the key in a Windows-readable directory:
`cp ./.ssh/id_rsa /mnt/c/Users/YOUR_WINDOWS_USER/Documents`

Python

As any user has its own Python requirements, we do not maintain the packages at system level. You just have to create your own python environments. **Conda** is our preferred system. You can install the virtual environment on your NAS home folder, so that it will be visible by all the workstations. Check it out at https://conda.io/miniconda.html.

To install Conda on your NAS home:

1. Go to your nas home: `cd /nas/home/mrossi`
2. Download installer: `wget https://repo.continuum.io/miniconda/Miniconda3-latest-Linux-x86_64.sh`
3. `bash Miniconda3-latest-Linux-x86_64.sh` and follow the instructions on terminal (watch out for the path choice! Eventually initialize conda on that machine)
4. Now you can remove the installation file with: `rm Miniconda3-latest-Linux-x86_64.sh`
5. Log out and login back
6. On all servers:
   6.1. add conda to .bashrc:
   `echo export PATH=/nas/home/mrossi/miniconda3/bin:$PATH >> ~/.bashrc`
   6.2. Log out and login back
   6.3. Check that conda is the default with `which conda`: it should return
   /nas/home/mrossi/miniconda3/bin/conda
   6.4. Initialize: `conda init`
   6.5. Log out and login back

You can now create your own environent it two ways:

1. Through a yml file (strongly suggested, useful for reproducibility).
2. Via command line.

Run jupyter notebooks on servers

1. Run a jupyter server on the remote workstation, then use ssh port forwarding to load the page on your local browser. Instructions here.
2. Run and save a jupyter notebook via command line with runipy (`pip install runipy`):
   `runipy -o mynotebook.ipynb`

Other useful python resources

1. Have a look at this repo https://github.com/polimi-ispl/blank_project: we inserted some scripts, tricks, and suggestions to have a project running in the shortest time possible
2. Numpy is the basics of scientific calculus, learn it here https://numpy.org/learn/
3. Need a simple machine learning framework? Scikit-learn got you covered https://scikit-learn.org/stable/auto_examples/index.html
4. Want to learn Pytorch/Tensorflow? Have a look at the official tutorials https://pytorch.org/tutorials/ https://www.tensorflow.org/tutorials