GRID GUIDE

Miguel Lunet – up 202005556

Access the GRID

```
ssh up202005556@submit.grid.fe.up.pt
Password: abcdefgh
```

See ongoing jobs

```
# See all jobs
squeue
```

```
up202005556@submit:~$ squeue
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
            565902     batch fluent-r up202004  R 1-07:45:08      2 ava[15-16]
            565931     batch fluent-r up202004  R   12:08:32      2 ava[09,18]
            565920     batch fluent-r up202004  R   23:41:03      2 ava[08,10]
            565841     batch  gpr 280 up201503  R 1-07:44:52      1 ava15
            565842     batch  gpr 281 up201503  R 1-07:44:52      1 ava15
            565843     batch  gpr 282 up201503  R 1-07:44:52      1 ava15
            565844     batch  gpr 283 up201503  R 1-07:44:52      1 ava15
            565845     batch  gpr 284 up201503  R 1-07:44:52      1 ava15
            565846     batch  gpr 285 up201503  R 1-07:44:52      1 ava15
            565847     batch  gpr 286 up201503  R 1-07:44:52      1 ava15
            565848     batch  gpr 287 up201503  R 1-07:44:52      1 ava15
            565849     batch  gpr 288 up201503  R 1-07:44:52      1 ava16
            565850     batch  gpr 289 up201503  R 1-07:44:52      1 ava16
            565851     batch  gpr 290 up201503  R 1-07:44:52      1 ava16
            565852     batch  gpr 291 up201503  R 1-07:44:52      1 ava16
            565853     batch  gpr 292 up201503  R 1-07:44:52      1 ava16
            565854     batch  gpr 293 up201503  R 1-07:44:52      1 ava16
            565855     batch  gpr 294 up201503  R 1-07:44:52      1 ava16
            565856     batch  gpr 295 up201503  R 1-07:44:52      1 ava16
            565812     batch  gpr 251 up201503  R 1-22:18:52      1 ava14
            565675     batch  gpr 114 up201503  R 3-03:09:40      1 ava07
            565941     batch teste_il up202005  R       1:05      1 ava07
            537527       big batch-74 up202200 PD       0:00      1 (ReqNodeNotAvail, UnavailableNodes:avafat01,cfp08,ventos[01-02])
            537528       big batch-79 up202200 PD       0:00      1 (ReqNodeNotAvail, UnavailableNodes:avafat01,cfp08,ventos[01-02])
            537529       big batch-60 up202200 PD       0:00      1 (ReqNodeNotAvail, UnavailableNodes:avafat01,cfp08,ventos[01-02])
            537530       big batch-11 up202200 PD       0:00      1 (ReqNodeNotAvail, UnavailableNodes:avafat01,cfp08,ventos[01-02])
            537531       big batch-58 up202200 PD       0:00      1 (ReqNodeNotAvail, UnavailableNodes:avafat01,cfp08,ventos[01-02])
            537532       big batch-9. up202200 PD       0:00      1 (ReqNodeNotAvail, UnavailableNodes:avafat01,cfp08,ventos[01-02])
            537533       big batch-38 up202200 PD       0:00      1 (ReqNodeNotAvail, UnavailableNodes:avafat01,cfp08,ventos[01-02])
```

```
# See only my jobs
squeue -u up202005556
```

```
up202005556@submit:~$ squeue -u up202005556
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
            565941     batch teste_il up202005  R       0:25      1 ava07
```

See all nodes and their specifications

```
#Node list and current state
sinfo
```

```
up202005556@submit:~$ sinfo
 PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
 batch*       up 5-00:00:00      1 drain* cfp01
 batch*       up 5-00:00:00      2    mix ava[07,14]
 batch*       up 5-00:00:00      6  alloc ava[08-10,15-16,18]
 batch*       up 5-00:00:00     17   idle ava[01,04,13,17,19-22],cfp[02-03,05-06]
 ,demec[02-06]
 big          up 7-00:00:00      1 drain* avafat01
 big          up 7-00:00:00      4  alloc cfp08,inegi01,ventos[01-02]
 big          up 7-00:00:00      1   idle cristalflow
 cfp          up 28-00:00:0      1    mix cfp10
 cfp          up 28-00:00:0      1  alloc cfp09
 cfp          up 28-00:00:0      1   idle cfp11
 ceft         up 28-00:00:0      2   idle ceft[01-02]
 lsrelcm      up 28-00:00:0      2  alloc lsrelcm[01,05]
 lsrelcm      up 28-00:00:0      3   idle lsrelcm[02-04]
```

```
# Check each node’s feature set
sinfo -N -o "%N %f %t"
```

```
 up202005556@submit:~$ sinfo -N -o "%N %f %t"
NODELIST AVAIL_FEATURES STATE
 ava01 AVX,IB idle
 ava04 AVX,IB idle
ava07 AVX,IB mix
 ava08 AVX,IB alloc
 ava09 AVX,IB alloc
 ava10 AVX,IB alloc
 ava13 AVX,IB idle
 ava14 AVX,IB mix
 ava15 AVX,IB alloc
 ava16 AVX,IB alloc
 ava17 AVX,IB idle
 ava18 AVX,IB alloc
 ava19 AVX,IB idle
 ava20 AVX,IB idle
 ava21 AVX,IB idle
 ava22 AVX,IB idle
 avafat01 AVX,GPU,IB drain*
 ceft01 AVX,AVX2 idle
ceft02 AVX,AVX2 idle
```

Request a node

```
# Request node ava17
srun -p batch -w ava17 --pty bash
```

```
up202005556@submit:~$ srun -p batch -w ava17 --pty bash
srun: job 565939 queued and waiting for resources
srun: job 565939 has been allocated resources
up202005556@ava17 ~ $
```

```
# Request any node that satisfies the mentioned constraint (AVX
feature set)
srun -p batch --constraint=AVX --pty bash

# Leave the requested node
exit
```

Delete folder/file

```
# Delete folder
rm -r folder_name

# Delete file
rm ~/folder_name/file_name
```

Submit a job

```
sbatch file_name.slurm
```

```
up202005556@submit:~$ sbatch il_instruction.slurm
Submitted batch job 565941
```

Virtual environment

```
# Create venv
pip install --user virtualenv
~/.local/bin/virtualenv ~/myvenv

# Load venv
source ~/myvenv/bin/activate

# Leave venv
deactivate
```

Writing a slurm file

```
#!/bin/bash
#Submit this script with: sbatch thefilename
#SBATCH --time=24:00:00   # walltime
#SBATCH --ntasks=1   # number of processor cores (i.e. tasks)
#SBATCH --nodes=1   # number of nodes
#SBATCH -p batch   # partition(s)
#SBATCH --mem-per-cpu=2G   # memory per CPU core
#SBATCH -J "benders_final"   # job name
#SBATCH --mail-user=up202005556@edu.fe.up.pt   # email address
#SBATCH --mail-type=BEGIN
#SBATCH --mail-type=END
#SBATCH --mail-type=FAIL
```

Additionally, one may specify the node or add a constraint to restrict the possible nodes.

```
#SBATCH --nodelist=ava21
#SBATCH --constraint=AVX
```

How to install Gurobi?

Select the x64 Linux Installer from the link below:

https://www.gurobi.com/downloads/gurobi-software/?_gl=1*3ir72x*_up*MQ..?_gl=1%2A10wy3xn%2A_gcl_aw%2AR0NMLjE3NjcwMjM2ODguQ2p3S0NBaUE5YVBLQmhCaEVpd0F5ejgySnk5OU83Tk0tTFhFQ08waFVRQVFkNXE0Wm1JaTlxSUR2SmUxQ3JfWGkzQlpCdUNGR09pb01Cb0NzMWdRQXZEX0J3RQ..%2A_gcl_au%2ANTMyNDYwODc2LjE3NjcwMjM2MjI.%2A_up%2AMQ..%2A_ga%2AMzc4NzU5ODU5LjE3NjcwMjM2MjM.%2A_ga_RTTPP25C8N%2AczE3NjcwMjM2MjIkbzEkZzEkdDE3NjcwMjM2ODckajYwJGwwJGgxMjMxMTE4NDE.&gclid=CjwKCAiA9aPKBhBhEiwAyz82Jy99O7NM-LXECO0hUQAQd5q4Zmli9qIDvJe1Cr_Xi3BZBuCFGOioMBoCs1gQAvD_BwE

Gurobi Optimizer v13.0.0 Release Notes md5sums

v13.0.0	Installer	md5 Checksum
x64 Windows	Gurobi-13.0.0-win64.msi	Gurobi-13.0.0-win64.msi.md5
x64 Linux	gurobi13.0.0_linux64.tar.gz	gurobi13.0.0_linux64.tar.gz.md5
macOS Universal2	gurobi13.0.0_macos_universal2.pkg	gurobi13.0.0_macos_universal2.pkg.md5
arm64 Linux	gurobi13.0.0_armlinux64.tar.gz	gurobi13.0.0_armlinux64.tar.gz.md5

cd ~tar -xvzf gurobi13.0.0_linux64.tar.gz #install gurobi

```
#Activate key (must be connected to FEUP VPN)
export GUROBI_HOME=gurobi1300/linux64
export PATH=$GUROBI_HOME/bin:$PATH
export LD_LIBRARY_PATH=$GUROBI_HOME/lib:$LD_LIBRARY_PATH
grbgetkey 84cbd980-4840-4e35-afbe-b7268f3fb5b2
grbgetkey 99006d14-b24a-4080-9eb8-0a2f72246acd
```

How to dynamically change the Gurobi license from node to node?

```
# Set the Gurobi license path dynamically
export
GRB_LICENSE_FILE="/homes/up202005556/gurobi_lic/${HOSTNAME}/gurob
i.lic"

# Call the Python script to assign the Gurobi license
python3 gurobi_licenses_assignment.py
```

The gurobi_licenses_assignment.py file has the following structure:

```python
import os
import subprocess

gurobi_keys = {"ava01": "",
"ava04": "",
"ava07": "",
"ava14": "",
"ava15": "",
"ava16": "",
"ava17": "",
"ava18": "",
"ava19": "99006d14-b24a-4080-9eb8-0a2f72246acd",
"ava20": "9396f859-c6ef-4819-95ad-09b35b261404",
"ava21": "c533f38d-1df0-4ce0-affe-e39568851572",
"ava22": "",
"cfp01": "",
"cfp02": "",
"cfp03": "",
"cfp05": "",
"cfp06": "",
"cfpsmall02": "",
"cfpsmall03": "",
"cfpsmall05": "",
"cfpsmall06": "",
"cfpsmall07": "",
"cfpsmall08": "",
"cfpsmall09": "",
"cfpsmall10": "",
"demec02": "",
"demec03": "",
"demec04": "",
"demec06": ""}
```

```
curr_machine = os.getenv('HOSTNAME')

if curr_machine not in gurobi_keys.keys():
    exit(1)
else:
    if os.path.join(os.path.expanduser("~"), "gurobi_lic", curr_machine):
        pass
    else:
        subprocess.run(["bash", "~/gurobi1300/linux64/bin/grbgetkey",
gurobi_keys[curr_machine]])
        os.mkdir(os.path.join("gurobi_lic", curr_machine))
        os.rename("gurobi.lic", os.path.join("gurobi_lic", curr_machine,
"gurobi.lic"))
```

Parser

The Python library Parser enables users to configure instance parameters, specify output files, manage additional options, and execute directly from the terminal.

```
# LOAD MODULES, INSERT CODE, AND RUN YOUR PROGRAMS HERE
module load python
cd teste_parser

# Define letters and numbers
letters=("A" "B" "C")
numbers=("1" "2")

# Loop over all combinations
for letter in "${letters[@]}"; do
  for number in "${numbers[@]}"; do
    instance_file="instances/instance_${letter}.json"
    params_file="parameters/params_${number}.json"
    output_file="outputs/output_${letter}${number}.csv"

    # Run Python script for each combination
    python3 main.py -instance "$instance_file" -params "$params_file" -output "$output_file"
  done
done
```

Jupyter notebook

```
jupyter nbconvert --to notebook --execute train_rl.ipynb --output
train_rl_executed.ipynb
```

Final comments

- When using GUROBI, it is essential to update the license being used from node to node. Each node must have its own license.
- When using TensorFlow or related libraries, the node must have the AVX structure.
- `Ctrl + C` allows to stop the code run manually.