

# Part 2

# Quick review

- What is: cluster / host?
- What is the smallest resource unit can we reserve in g5k? And by default?
- Which protocol do we use to get connected to g5k?

# Exercise (2)

- Transfer from local to remote a folder called `g5ktest/` with 2 empty files (`test1.txt`, `test2.txt`) inside with `rsync`, remote folder should contain the same folder
- Write “Hello from g5k” in `test1.txt` remotely, and transfer from back to local with `scp`

Tips: some useful bash commands

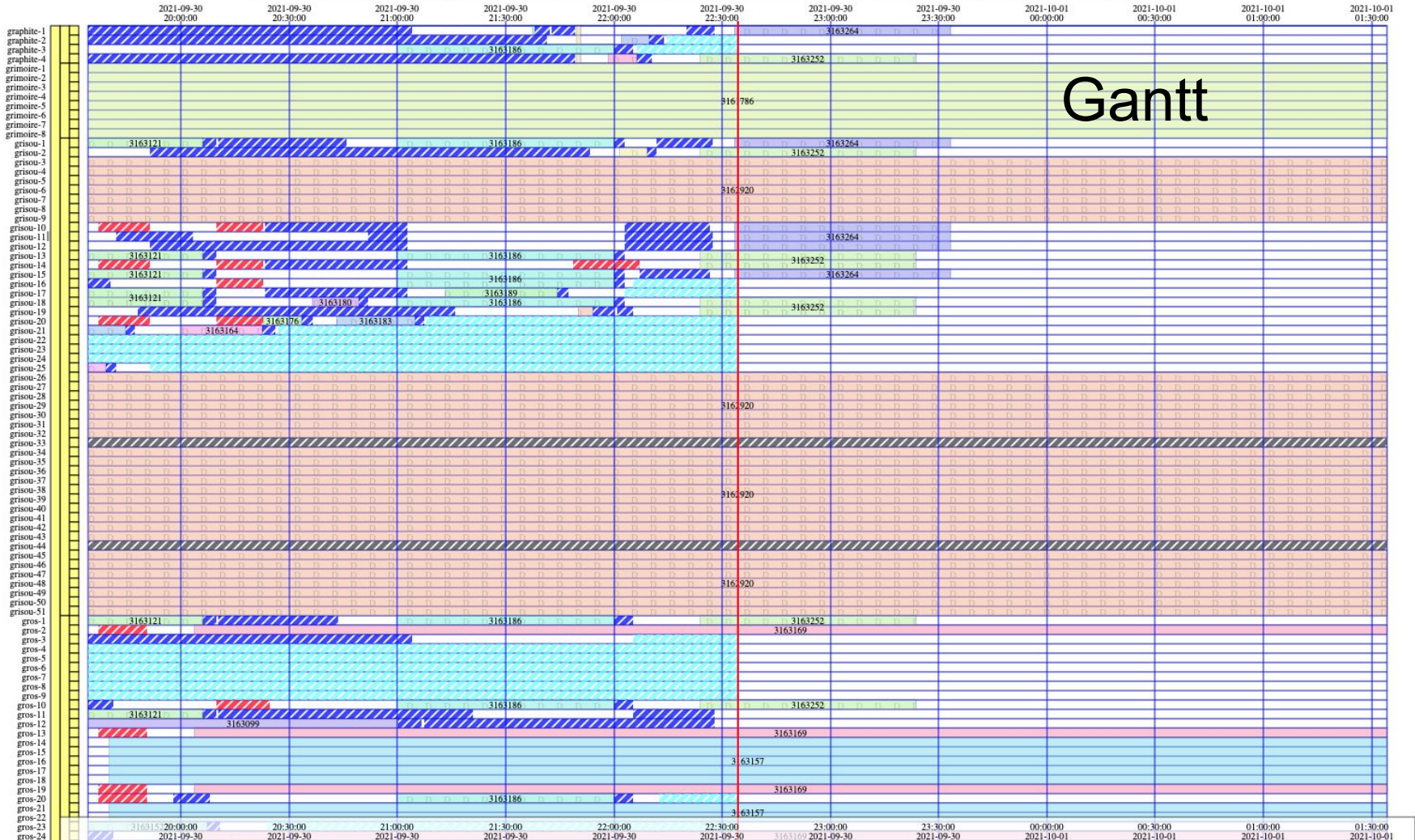
- Create a folder: ``$ mkdir folderName``
- Move into a folder: ``$ cd path2folder``
- List elements in a folder: ``$ ls``
- Create a file: ``$ touch myFile.txt``
- Write in a file: ``$ echo "hello g5k" > myFile.txt``
- Show content in a file: ``$ cat myFile.txt``
- Scp copy file from remote to local:
  - `scp remote_username@remote_ip:remote_file local_file`
- Rsync copy folder from local to remote:
  - `rsync -avzP local_folder remote_username@remote_ip:remote_folder`

# Visualisation & Reservation

# Visualizing Grid'5000 resources

- Several ways to learn about resources and their status
  - [Monika](#): reservation state
  - [Gantt](#): reservation history and forecast, very useful
  - [Ganglia](#): resources usage (load, memory, CPU, network usage in last hours)
  - [Platform events](#): show maintenance news
  - More info: ref [nancy home](#) site





Gantt

# Ganglia

Main Search Views Aggregate Graphs Compare Hosts Events Automatic Rotation Live Dashboard Mobile

## Nancy Cluster Report at Thu, 30 Sep 2021 22:42:21 +0200

Get Fresh Data

Last         or from  to

Physical View

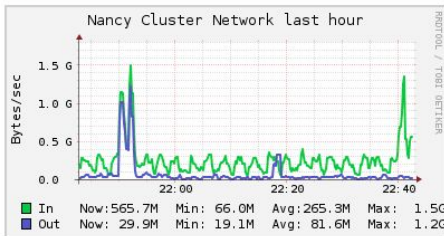
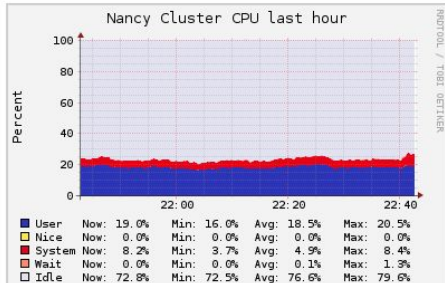
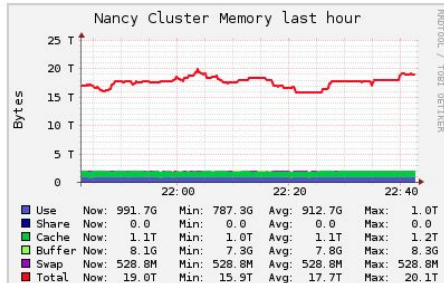
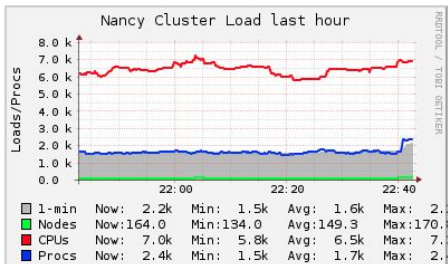
Grid5000 Grid > Nancy >

### Overview of Nancy @ 2021-09-30 22:42

CPUs Total: **16069**  
Hosts up: **164**  
Hosts down: **345**

Current Load Avg (15, 5, 1m):  
**11%, 12%, 14%**  
Avg Utilization (last hour):  
**25%**

#### Server Load Distribution





<https://intranet.grid5000.fr/oar/Nancy/monika.cgi>

<https://intranet.grid5000.fr/oar/Nancy/drawgantt-svg/>

<https://intranet.grid5000.fr/ganglia/>

# Reserving resources with OAR

- OAR: resources and jobs management system (batch manager) in g5k
- Smallest unit of resource: **core** (cpu core)
  - E.g.: graffiti have 2 CPU with 8 cores/CPU, maximum reserved for 16 tasks
  - By default a OAR job reserves a **host** (=nodes, physical computer with all cpu/cores)
- Reservation syntax, spot your machine in the Gantt chart

To reserve one host (one node), in interactive mode, do:

```
fnancy : oarsub -I
```

To reserve three hosts (three nodes), in interactive mode, do:

```
fnancy : oarsub -l host=3 -I
```


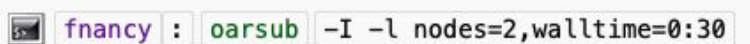
or equivalently:

```
fnancy : oarsub -l nodes=3 -I
```

To reserve only one core in interactive mode, run:

```
fnancy : oarsub -l core=1 -I
```

# Reserving resources with OAR: interactive mode

- Interactive mode
  - Use option `-I`
    - As soon as a resource is available, directly connected to that resource with an interactive shell. By default *walltime* = 1 hour
  - If you want to reserve GPU
    -  `fnancy : oarsub -l gpu=1 -I -q production`
    - This means reserve 1 GPU with the associated cores in the [queue production](#)
    - Nodes with GPU are **exclusively** in the production queue in Nancy
  - Terminate reservation and return to frontend
    - `exit` or `CTRL+d`
  - Need more than 1 node or longer time ([walltime](#)):
    -  `fnancy : oarsub -I -l nodes=2,walltime=0:30`

# Reserving resources with OAR: passive mode

- Passive mode

- By default, no need to add an option

```
fnancy : oarsub -l host=1/core=1 "my_mono_threaded_script.py --in $HOME/data --out $HOME/results"
```

- Reservation in 2 steps

- First reserve a node and ask it to sleep for a long time
    - Allocate a job\_ID quickly
    - Then use this command to enter the host

```
fnancy : oarsub "sleep 10d"
```

```
fnancy : oarsub -C job_id
```

- Advantage: no worry about accidentally terminate your task (terminal closed or network disconnection)
  - More parameters:
    - **-r**: reserve a specific time in the future


```
fnancy : oarsub -l nodes=3,walltime=3 -r '2020-12-23 16:30:00'
```

- More options to reserve a resource check `oarsub --help`

# Job management

- View your list of jobs with `oarstat`
  - Option `-u` see only your jobs: `oarstat -u`
  - Option `-j job_id` see the state for this particular job
  - Stats: W=waiting, L=launching, R=running, F=finish

Job id	Name	User	Submission Date	S	Queue
3158594		cli	2021-09-27 22:19:44	W	default
3159821		cli	2021-09-29 10:30:41	R	production
3163328		cli	2021-09-30 23:22:17	F	default
3163330		cli	2021-09-30 23:24:24	W	default


- Delete a job with `oardel`
  - 
- Passive mode jobs, **stdout** and **stderr** streams are created automatically
  - check out stream (or error stream) with `cat` at any time
  - `cat OAR.2758674.stdout`

# Job management

- Specify the properties of host with option ``-p``

- exemples :

-  fluxembourg : oarsub -p "cluster='granduc'" -l nodes=5,walltime=2 -I

-  flyon : oarsub -p "wattmeter='YES' and gpu\_count > 0" -l nodes=2,walltime=2 -I

- oarsub also accepts SQL

- All properties: [https://www.grid5000.fr/w/OAR\\_Properties](https://www.grid5000.fr/w/OAR_Properties)

- Extend the duration with ``+time``:

-  fnancy : oarwalltime 12345 +1:30

- Not whenever you want, check rules in Usage Policy

## More exercises (3)

- (1) Reserve a host in interactive mode
- (2) Reserve 1 core and launch a bash command 'sleep 10d' in non-interactive mode
- (3) Reserve 2 GPUs in host 'graffiti-4' (site Nancy), in queue production for 1 hour, interactive mode
- (4) Reserve 2 cores in 'grvingt' in production queue and sleep 10 days
- (5) Reserve 1 node in cluster 'grvingt' for 20 minutes, and launch script 'run.sh'
- (6) Check your reservations, delete (4)

# More exercises (3)

- (1) Reserve a host in interactive mode
- (2) Reserve 1 core and launch a bash command 'sleep 10d' in non-interactive mode
- (3) Reserve 2 GPUs in host 'graffiti-4' (site Nancy), in queue production for 1 hour, interactive mode
- (4) Reserve 2 cores in 'grvingt' in production queue and sleep 10 days
- (5) Reserve 1 node in cluster 'grvingt' for 20 minutes, and launch script 'run.sh'
- (6) Check your reservations, delete (4)

- `oarsub -l`
- `oarsub -l core=1 "sleep 10d"`
- `oarsub -p "host IN ('graffiti-4.nancy.grid5000.fr')" -l host=1/gpu=3,walltime=1 -q production -l`
- `oarsub -p "cluster='grvingt'" -l core=2 "sleep 10d" -q production`
- `oarsub -p "cluster='grvingt'" -l nodes=1,walltime=0:20 "bash run.sh" -q production`
- `oarstat -u`
- `oardel jobID(4)`



# Useful links for reservation

Basics about reservation:

[https://www.grid5000.fr/w/Getting\\_Started#Reserving\\_resources\\_with\\_OAR:the\\_basics](https://www.grid5000.fr/w/Getting_Started#Reserving_resources_with_OAR:the_basics)

Advanced OAR: [https://www.grid5000.fr/w/Advanced\\_OAR#Passive\\_mode](https://www.grid5000.fr/w/Advanced_OAR#Passive_mode)

# Community

- Report the problems to the community
  - [users@lists.grid5000.fr](mailto:users@lists.grid5000.fr)
- (if you want) join the technical committee
  - Subscribe to [devel@lists.grid5000.fr](mailto:devel@lists.grid5000.fr)
  - Discussions and bugs
- If you want to apply for a new account
  - [https://www.grid5000.fr/w/Grid5000:Get\\_an\\_account](https://www.grid5000.fr/w/Grid5000:Get_an_account)
  -

Towards deep learning

# Deep learning

- Creation of a virtual environment for python
- Installation of deep learning software
- Configuration of software (such as cudnn library, config file)
- Running DL software on Grid'5000
  - Reservation with oarsub
  - monitoring (log files, kill)
  - Use several GPU cards
- Tips and tricks, for detailed info follow [this link](#)

# Deep learning - virtual env.

- Creation of a virtual environment for python

- Go to Nancy g5k site

-  `inside` : `virtualenv /home/ login /venv`

- Can precise interpreter with `-p`` such as `--python=python3.7``

- Activate virtual environment

-  `inside` : `source /home/ login /venv/bin/activate`

- Otherwise, can do with anaconda

# Deep learning - pytorch installation

- Pytorch
  - Reserve a cluster with GPU (graffiti, graphique, grimani, etc.)
  - In the host, [install torch](#) with pip or anaconda
  - Load module cuda and cudnn in current shell
    - `$ module av`
    - `$ module load cuda/11.0.1_gcc-8.3.0`
    - `$ module load cudnn/7.6.5.32-10.1-linux-x64_gcc-8.3.0`
  - Check if pytorch is correctly installed to work with GPU
    - `$ python3 -c "import torch; print(torch.cuda.is_available())"`
- Similar for Tensorflow

# Deep learning - nancy site

- Available nodes
  - grimani: 6 nodes, each node has 2 Nvidia K40m GPU cards
  - graphique: 6 nodes, 2 x Nvidia Titan Black ( graphique-1 ), 2 x Nvidia GTX 980 GPU ( other nodes )
  - grele: 14 nodes, each node has 2 Nvidia Geforce 1080 Ti GPU cards
  - [graffiti](#): 13 nodes, each node has 4 Nvidia Geforce RTX2080 GPU cards
- Each gpu cluster has 2 GPU cards
  - Script can use already the 2 cards
  - If want to use multiple GPU cards of one machine in parallel, ref [this tuto](#)

# Deep learning - reservation

## - Reserve one GPU

- Interactive mode: `inside : oarsub -q production -l "nodes=1/gpu=1,walltime=0:20:00" -I`
- Passive mode:
  - `inside : oarsub -q production -l "nodes=1/gpu=1,walltime=0:20:00" <path to a bash script>`
  - Move into the host: `site:~$ oarsub -C job_id``
- Check GPU usage: `host:~$ nvidia-smi -l 2``

```
+-----+
| NVIDIA-SMI 450.51.05      Driver Version: 450.51.05      CUDA Version: 11.0      |
+-----+-----+-----+-----+-----+-----+
| GPU  Name          Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC | | | |
| Fan  Temp   Perf    Pwr:Usage/Cap|      Memory-Usage | GPU-Util  Compute M. |
|====|=====|====|=====|=====|=====|
|  0   Tesla K40m      Off          | 00000000:03:00.0 Off  |          0          |
| N/A   22C    P8      21W / 235W | 0MiB / 11441MiB |      0%      Default |
|====|=====|====|=====|=====|=====|
+-----+-----+-----+-----+-----+-----+

+-----+
| Processes: |
| GPU  GI  CI       PID   Type   Process name                      GPU Memory |
|  ID   ID             |          |          |                               | Usage     |
|====|=====|====|=====|=====|=====|
| No running processes found |
+-----+
```



# Conclusion

# Wrap up

## We have seen

- Connecting to Grid'5000
- Infrastructure map, with some basic concepts
- Visualizing resources
- Transferring files
- Reserving resources with 2 modes
- Job management
- A deep learning framework

# Wrap up

## We have seen

- Connecting to Grid'5000
- Infrastructure map, with some basic concepts
- Visualizing resources
- Transferring files
- Reserving resources with 2 modes
- Job management
- A deep learning framework

## We have used

- ssh
- site, cluster, nodes, core...
- Gantt, Monika...
- scp, rsync
- oarsub
- oarstat, oardel, oarwalltime
- Pytorch installation

# Wrap up

- Grid'5000 is a fantastic tool for your research
- Mastering it is challenging
- Be positive, find a problem, ask and share =)
- Questions?