Institut Pierre Simon Laplace
Sciences de l'environnement

ESPRI

MÉSOCENTRE IPSL

MODULE 1

Infrastructure

CICLAD cluster at UPMC (Jussieu)

ClimServ at Polytechnique (Palaiseau)

MESOCENTRE DESCRIPTION

Mesocentre ESPRI / IPSL

- Computing resources closest to the data storage
- Distributed on several sites (UPMC and Ecole Polytechnique)

https://mesocentre.ipsl.fr

Description Items

- Computing Units
- Storage and File System
- Networks
- Disk spaces
- Data distribution services / Projects hosting

MESOCENTRE DESCRIPTION

Computing units

Interactive / Head Nodes:

- SSH connection to the Mesocentre
- Dedicated to calculations and short or interactive processes:
  - Program developments
  - Visualisation of results
  - Short processing requiring interaction:
    - ➔ Entering input values
    - ➔ Follow-up of results
- Shared resources (CPU, Memory)
- Limited resources
  - 1 h CPU max — 4Go RAM/user
- Registered on national computing centres (access allowed from these nodes)

```
# Login on ClimServ

# camelot
# AMD 16-core 2.5GHz | 64Gb
> ssh login@camelot.ipsl.polytechnique.fr

# loholt1/2
# Intel 16-core 2.4GHz | 64Gb
> ssh login@loholt1.ipsl.polytechnique.fr
> ssh login@loholt2.ipsl.polytechnique.fr
```

```
# Login on CICLAD

# ciclad
# AMD 32-core 3.0GHz | 132Gb
> ssh login@ciclad.ipsl.jussieu.fr

# ciclad2
# Intel 16-core 2.4GHz | 64Gb
> ssh login@login@ciclad2.ipsl.jussieu.fr
```

MESOCENTRE DESCRIPTION

Computing units

High Performance Computing Clusters

- Accessible via the Head Nodes: `qsub` command
- 50 computing nodes (2400 AMD cores)
- 32 or 64 core per node (4 sockets)
- Memory: 4Gb/core (up to 256 GB per node)
- Dedicated to **<u>any</u>** calculations or processes (short, long, interactive, non-interactive, etc.)
- Managed by a Queueing / Resource manager system (Torque / Maui)
- Per job/process reserved resources (dedicated CPUs)

PBS Queues on ClimServ

```
[ root@camelot:~ ]--
-$ qstat -q

server: merlinvirt-c.climserv

Queue            Memory CPU Time Walltime Node  Run Que Lm  State
---------------- ------ -------- -------- ----  --- --- --  -----
infini             --      --    999:59:5   --    3   0 --   E R
wrf                --      --    999:59:5   --    6   1 --   E R
std                --      --    06:00:00   --    1   0 --   E R
day                --      --    24:00:00   --    6   0 --   E R
default            --      --       --      --    0   0 --   E R
week               --      --    168:00:0   --   10   0 --   E R
threedays          --      --    72:00:00   --    0   0 --   E R
                                                ----- -----
                                                   26     1
```

PBS Queues on Ciclad

```
[root@ciclad2 ~]# qstat -q

server: cicladpbs6.private.ipsl.fr

Queue            Memory CPU Time Walltime Node  Run Que Lm  State
---------------- ------ -------- -------- ----  --- --- --  -----
short              --      --    02:00:00   --    3   0 --   E R
day                --      --    24:00:00   --    9   2 --   E R
weeks2            32gb     --    360:00:0   --   10   0 --   E R
infini            16gb     --    860:00:0   --    4   0 --   E R
iago2rt            --      --    24:00:00   --    0   0 --   E R
week               --      --    168:00:0   --   13   0 --   E R
default            --      --       --      --    0   0 --   E R
std                --      --    06:00:00   --   13  12 --   E R
h12                --      --    12:00:00   --    1   0 --   E R
days3              --      --    72:00:00   --    1   0 --   E R
                                                ----- -----
                                                   54    14
```

Computing units

Processors Architecture AMD / Intel

- What Wikipedia teaches us on “Processor architecture”:

“All computers run using very low-level commands which do some very basic functions, such as reading data, writing data, jumping to addresses, and calculating basic arithmetic. […] Instruction sets are relatively small; most higher-order programming languages, such as C++, Ada, Fortran, or Visual Basic, must be compiled (or translated, or interpreted) into these low level commands in order for a program to run.”

- Instruction sets can be extended depending on the CPU generation and version (MMX, 3DNow, SSE, PAE, etc.) List the CPU properties with: `cat /proc/cpu`
- The extended instructions are used by the compiler when optimisation parameters are used (-fast, -O3, -O4, etc.) for the compilation

So what ?

- The Mesocentre’s nodes are built upon several generation of processors
- Need to use an instruction set appropriate to the computing node CPU

```
-$ pbsnodes merlin11-c.climserv
merlin11-c.climserv
     state = free
     np = 64
     properties = Piledriver,merlin11
```

```
[root@ciclad-ng ~]#  pbsnodes ciclad25.private.ipsl.fr
ciclad25.private.ipsl.fr
     state = free
     np = 64
     properties = bulldozer,jago2rt
```

- By default, Intel compilers does not generate processor-specific code (option `-march`)
- By default, PGI compilers generates processor-specific code. To force code / CPU compatibility :
  - Instructs PGI compiler to generate a code with common instruction sets (option `-tp=x64`)
  - Instructs PGI compiler to generate a processor-specific code (e.g. `-tp=piledriver`) and submit the job on the appropriate node: `qsub -l nodes=1:piledriver […]`

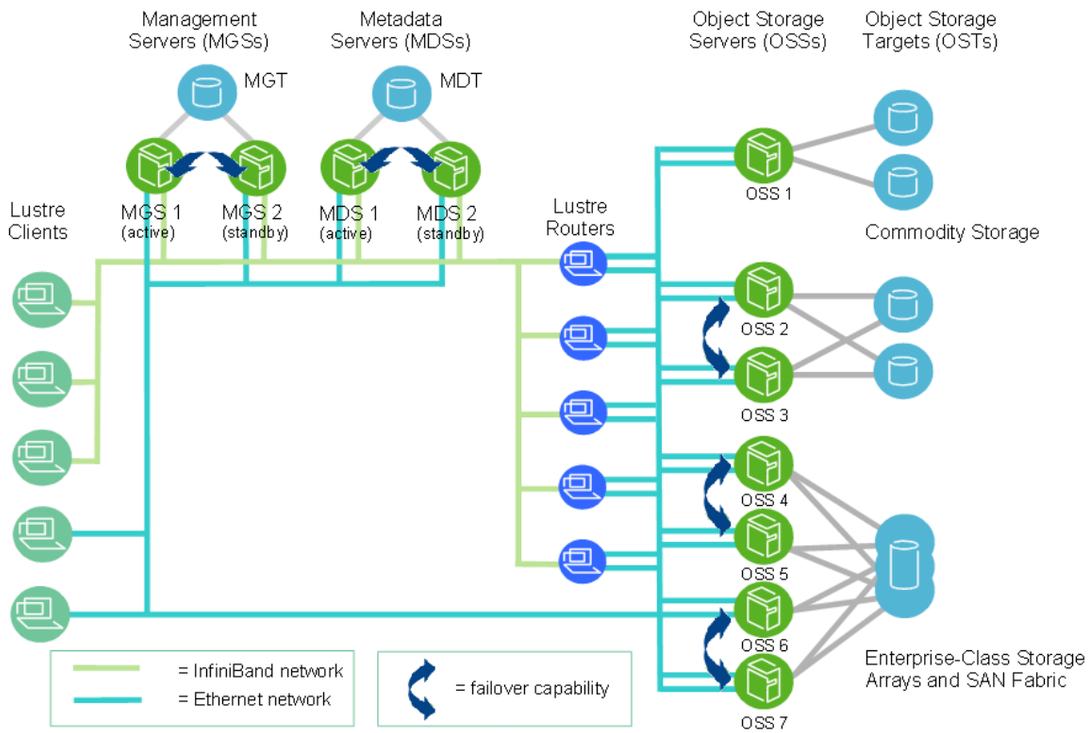
MESOCENTRE DESCRIPTION

Storage and filesystem

Storage Hardware

- 60-Disks Redundant (RAID) Storage Arrays
- 8 or 16 Gbps Fibre Channel server connection
- 25 physical servers
- More than 1300 Hard Disk
- 4.8 Pb of raw hard disk = **3 Pb of effective storage capacity**

Storage File System



Lustre Parallel Filesystem

- MetaData Servers (MDS) with MetaData Targets (MDT) store namespace metadata (filenames, directories, access permissions, and file layout)
- Object Storage Server (OSS) with Object Storage Targets (OST) store file data
- Clients that access and use the data.

The capacity of a Lustre FS is the sum of the capacities provided by the OSTs.

```
[ramage@camelot ~]$ lfs df -h /bdd/ERA20C/
UUID                       bytes        Used   Available Use% Mounted on
climfs-MDT0000_UUID       150.0G        6.4G       141.5G   4% /mnt/climfs-2.5/lov[MDT:0]
climfs-OST0000_UUID        55.8T       36.3T        18.9T  66% /mnt/climfs-2.5/lov[OST:0]
climfs-OST0001_UUID        55.8T       36.4T        18.9T  66% /mnt/climfs-2.5/lov[OST:1]
climfs-OST0002_UUID        57.3T       36.1T        20.6T  64% /mnt/climfs-2.5/lov[OST:2]
climfs-OST0003_UUID        57.3T       36.2T        20.5T  64% /mnt/climfs-2.5/lov[OST:3]

filesystem summary:       226.2T      145.0T        78.9T  65% /mnt/climfs-2.5/lov
```

Storage and filesystem

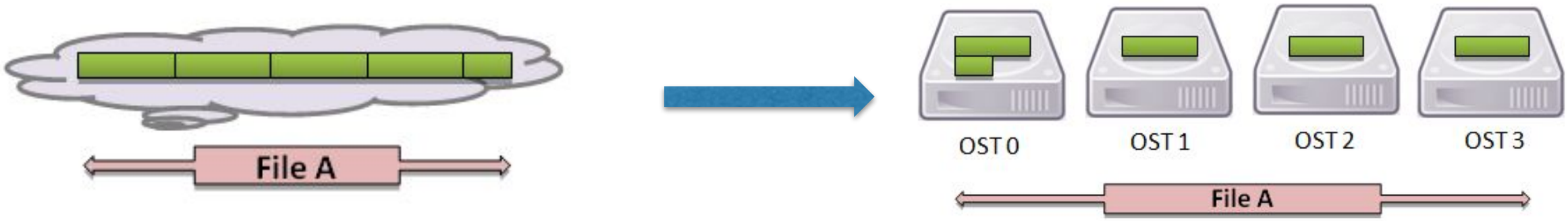
Lustre Best practices

- Reading, writing, and removing many small files burdens the file system and can slow it down for all users.
- Placing too many files in a single directory negatively impacts performance.
- Avoid using `ls -l` (or any ls command that uses the -l flag, or any command that must stat many files in quick succession).
- Do not use wildcards (`*`) in directories containing thousands of files.
- Open files read-only, specifying O_NOATIME if there is no reason to update the access time.
- If many processes need the stat information from a single file, it is most efficient to have a single process perform the stat call and broadcast the results.
- Avoid frequently opening files in append mode, writing small amounts of data, and closing the file.
- Instead of reading a small file from every task, read the entire file from one task and broadcast the contents to all other tasks.

(https://www.rc.colorado.edu/support/examples-and-tutorials/parallel-io-on-janus-lustre.html)

File Striping

Lustre FS can distribute the segments of a single file across multiple OSTs to improve I/O performances



- By default file striping is not activated system-wide: If an OST is failed the whole FS is corrupted.
- File striping can be activated for a specific file or directory

(www.nics.tennessee.edu/computing-resources/file-systems/io-lustre-tips)

Storage and filesystem

Lustre User commands

Lustre's **lfs** utility provides several options for monitoring and configuring your Lustre environment

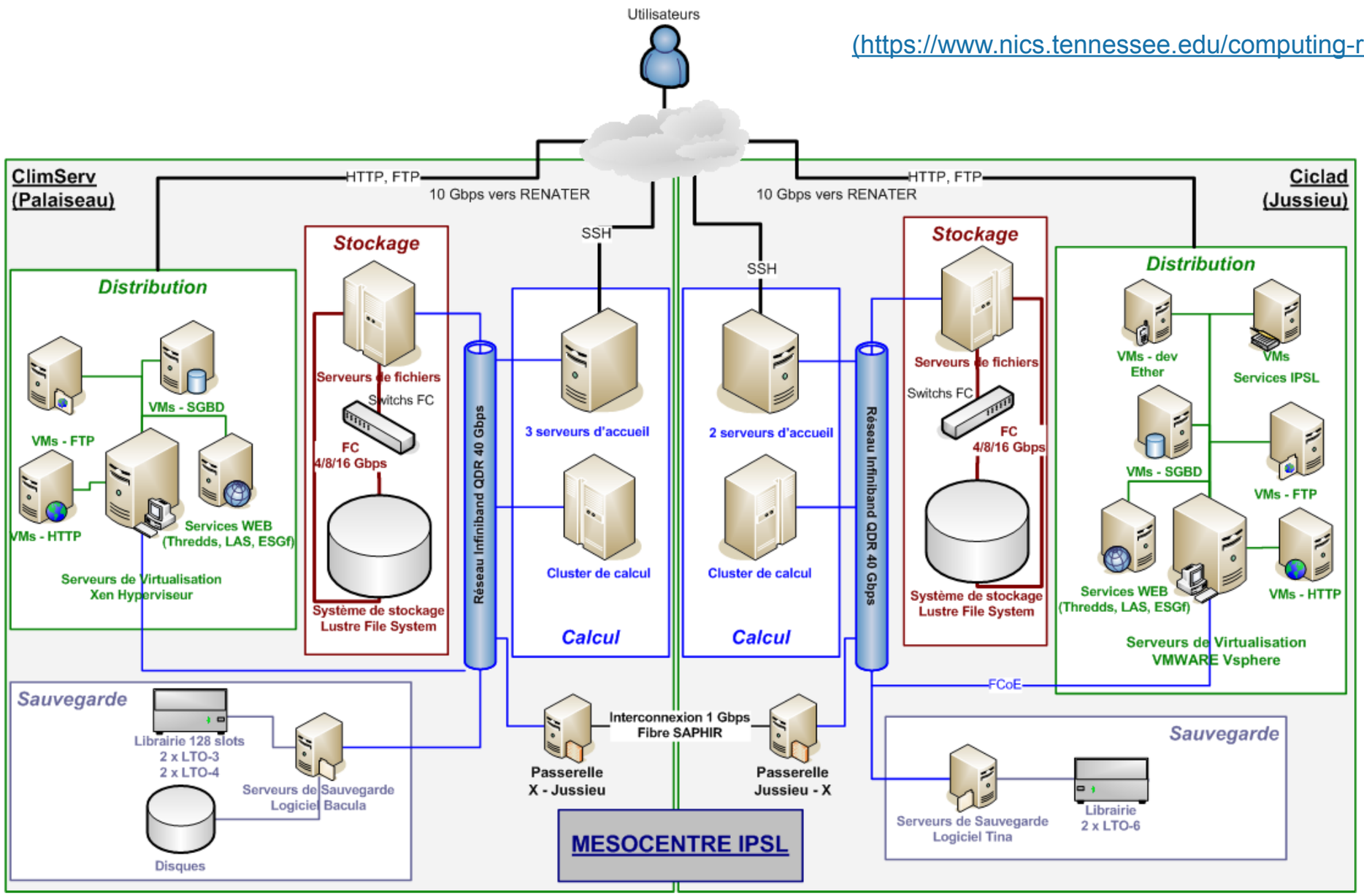
```
# Options of LFS command
> lfs help
> lfs help option-name

# Check disk space usage
> lfs df [-i] [-h] [--pool|-p <fsname>[.<pool>] [path]

# Check disk quotas
> lfs quota -u username /home

# Search the directory tree (more efficient than GNU find command)
> lfs find [[!] --atime|-A [-+]N] [[!] --mtime|-M [-+]N] [[!] --ctime|-C [-+]N] [--
maxdepth|-D N] [--name|-n ] [—print|-p] [--print0|-P] [[!] --obd|-O ] [[!] --size|-S
[+-]N[kMGTPE]] --type |-t {bcdflpsD}] [[!] --gid|-g|--group|-G |][[!] --uid|-u|--user|-
U |] <dirname|filename>

# Get striping information
> lfs getstripe [--obd|-O ] [--quiet|-q] [—verbose|-v] [--count|-c] [--index|-i | --
offset|-o] [--size|-s] [--pool|-p] [--directory|-d] [--recursive|-r]

# Set striping patterns
> lfs setstripe [--size|-s stripe_size] [--count|-c stripe_cnt] [--index|-i|--offset|-o
start_ost_index] [--pool|-p <pool>] <dirname|filename>
```

MESOCENTRE DESCRIPTION

Network

- **High bandwith / Low latency Network:** 40 Gbps Infiniband Network within each cluster (cluster intranet)
- **10 Gbps shared connection to Renater Network (Mesocentre Internet)**
- **1 Gbps dedicated fiber channel connection between ClimServ and Ciclad**

(https://www.nics.tennessee.edu/computing-resources/file-systems/io-lustre-tips)



MESOCENTRE DESCRIPTION

Storage and filesystem

Mesocentre disk spaces:

- All disk spaces are visible from the clusters (ClimServ/Ciclad) of the mesocentre
  - Data stored on ClimServ are mounted read-only via NFS on CICLAD
  - Data stored on CICLAD are mounted read-only via NFS on ClimServ
- Because of the network topology of the mesocentre, it is advisable to work on the cluster closest to the processed data
- **User directories:**

ClimServ				CICLAD		
Directory	*Limits*	*Backup*		*Directory*	*Limits*	*Backup*
`/home`	10Gb | 300 000 files	Daily incremental		`/home`	20Gb | 300 000 files	Daily sync `/backupfs`
`/homedata`	200Gb | 300 000 files	15-day differential		`/data`	1To | 300 000 files	No
`/ciclad-home`	Read-Only	N/A	⇔	`/climserv-home`	Read-Only	N/A
`/data`	Read-Only	N/A	⇔	`/homedata`	Read-Only	N/A

- **Data directories:**
  - ClimServ: `/bdd`
  - CICLAD: `/etherfs, /prodigfs, /loceanfs, /ccc`

MESOCENTRE DESCRIPTION

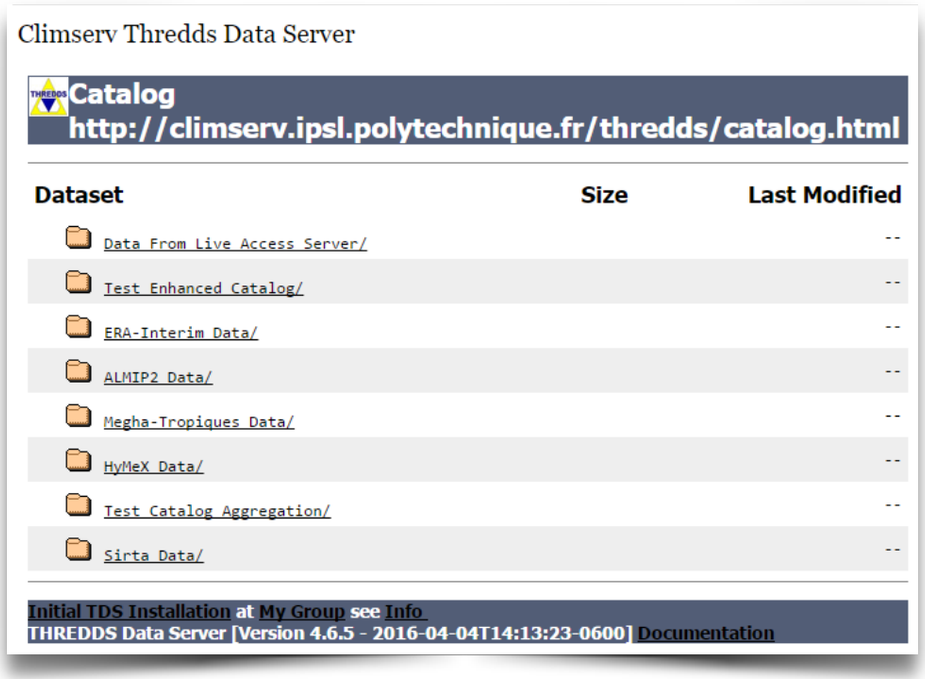
Project hosting and data distribution

Virtualisation system

- Two virtualisation systems based on VMWARE and Xen are used to host applications using data from the mesocentre, while optimally adjusting the resources required.

Data distribution services: Data produced or hosted on the mesocentre can be made available through several distribution services according to the needs of the project

- FTP
- HTTP
- THREDDS Data Server



MESOCENTRE DESCRIPTION

http://meso-ganglia.ipsl.fr

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

MESOCENTRE-IPSL-ESPRI Grid Report at Wed, 17 May 2017 23:07:02 +0200 — Get Fresh Data

Last: hour | 2hr | 4hr | day | week | month | year | job — or from ____ to ____ Go Clear

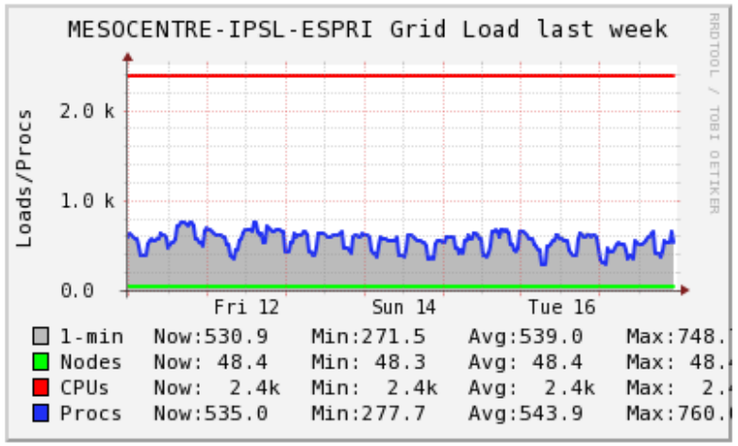
Sorted: ascending | descending | by name | by hosts up | by hosts down

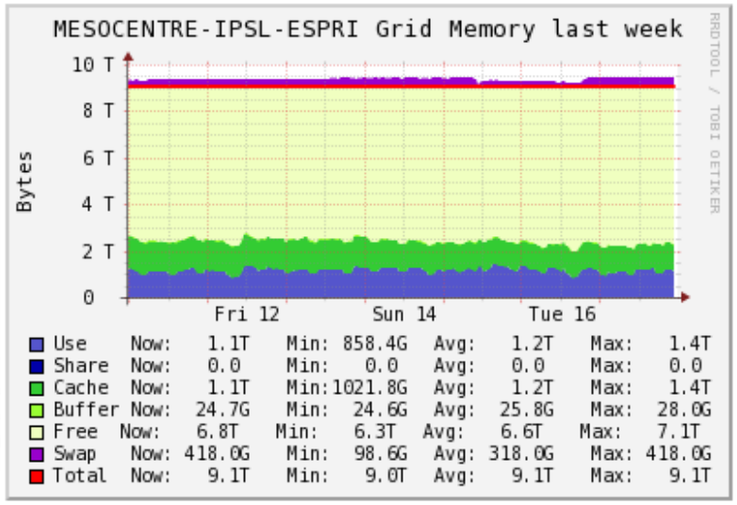
MESOCENTRE-IPSL-ESPRI Grid > --Choose a Source

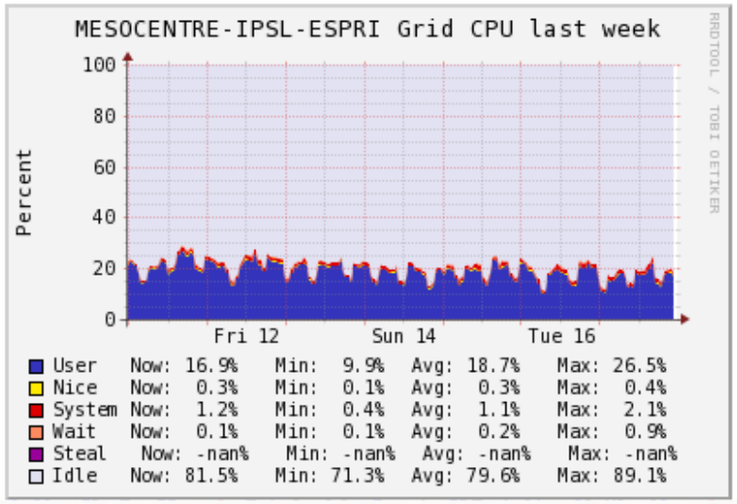
MESOCENTRE-IPSL-ESPRI Grid (2 sources) (tree view)

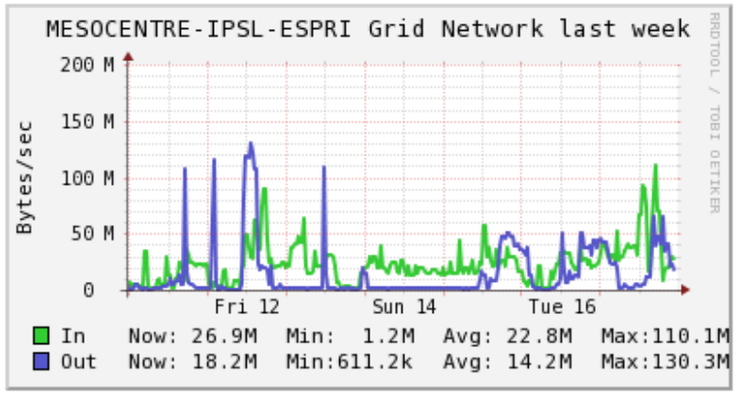
CPUs Total: **2348**
Hosts up: **48**
Hosts down: **2**

Current Load Avg (15, 5, 1m):
23%, 24%, 24%
Avg Utilization (last week):
23%
Localtime:
2017-05-17 23:07









MESOCENTRE DESCRIPTION

History

- **Climserv** (Ecole Polytechnique) 1996
- **CICLAD** (UPMC) 2008
- Node interconnection Infiniband 40Gb/s (2012)
- Interconnection by dedicated 1Gb/s link 2015
- **ESPRI Mesocentre** (Climserv + Ciclad) 2017
- Website: http://mesocentre.ipsl.fr/

Support

Mail to: meso-support@ipsl.fr

- This is ticket system not mailing list
- thank you to answer only if problem persist
- or adding information (answering reopen tickets)
- Don't use old ticket to ask something new
- this is added to the old ticket

Try to be precise even in subject ;-) when asking

- For SSH: Your workstation OS? OS version?
- For jobs: jobs number? Script path? Script outputs?

MESOCENTRE DESCRIPTION

Hardware

Head nodes

- SSH Connection Access (5)

Computing nodes

- 50 nodes (2400 AMD core)
- 32 or 64 core per node (4 sockets)
- Memory 4GB/core

Filers nodes

- Lustre parallel Filesystem (3Po)
- 25 physical servers
- More than 1300 hard disk

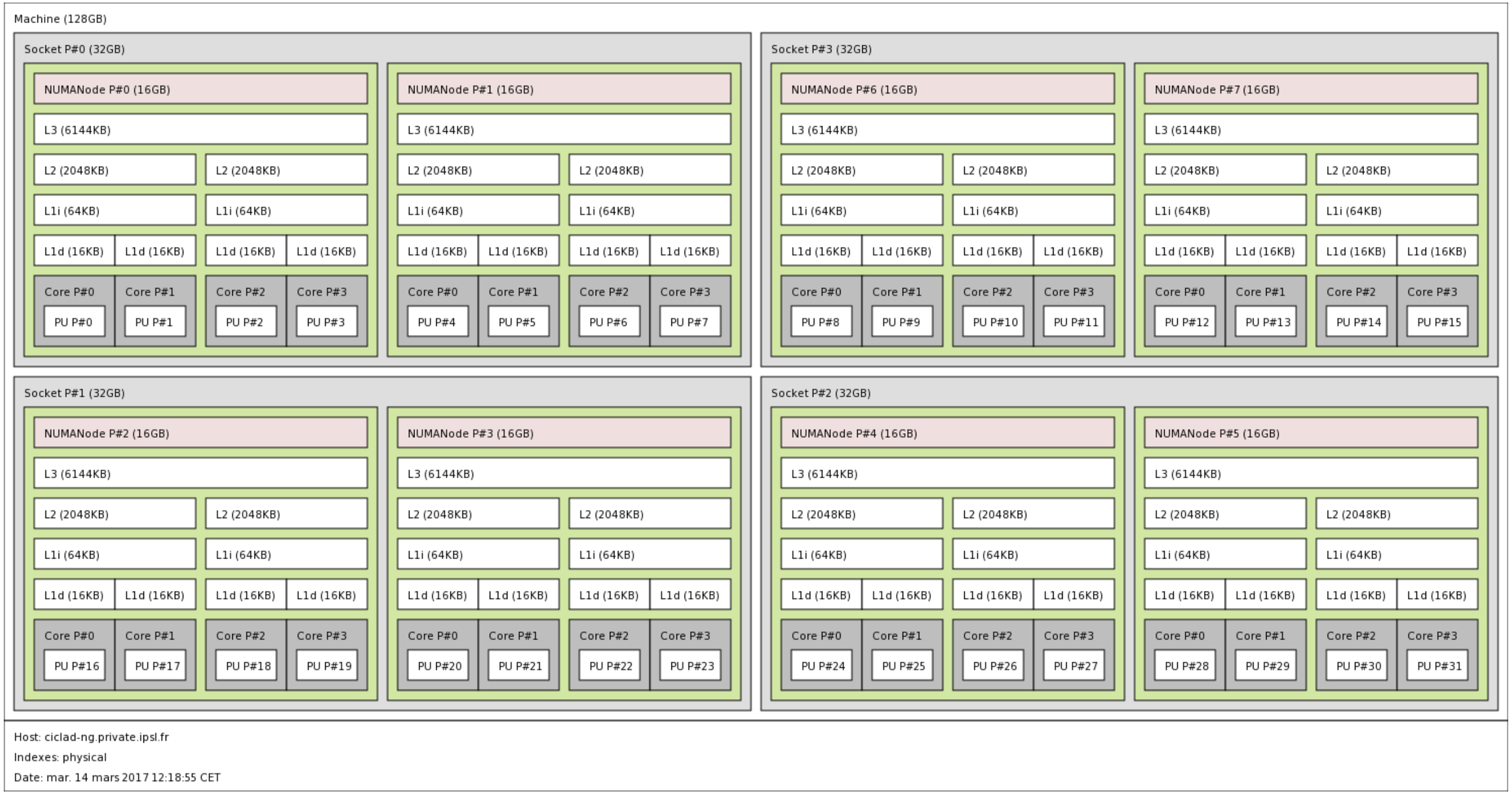
Virtualisation nodes

- VMWare and XEN

MESOCENTRE DESCRIPTION

Computing node architecture

NUMA (Non Uniform Memory Access)



MESOCENTRE UTILITIES

`module` command

Now ESPRI Mesocentre (CICLAD and ClimServ) is using the `module` command for software initialisation

```
module (avail [product] | load product[/version] | list | switch product/version1 product/version2 | display
```

- `product[/version]`: Product could be a compiler, a library or a software
- `avail` — List all known product and versions
- `load` — Initialise a product in his default version if no version is specified
- `list` — List already loaded product and version
- `switch` — Changing the version of already loaded product
- `purge` — Unload product
- `display` — Show the module file

MESOCENTRE UTILITIES

module command

```
> module avail
------------------------------------------- /usr/share/Modules/modulefiles -------------------------------------------
dot         module-git  module-info modules     null        use.own
---------------------------------------------- /etc/modulefiles/Compilers ----------------------------------------------
gnu/4.4.7(default)        intel/12.1.3.293(default) nagfor/5.3                pgi/2013(default)
gnu/4.7.2                 intel/15.0.6.233          nagfor/6.0(default)       pgi/2016
[…]
---------------------------------------------- /etc/modulefiles/Libraries ----------------------------------------------
cmor/2.9.1-gfortran         hdf5/1.8.14-pgf2016         netcdf4/4.3.3.1-pgf2016     openmpi/1.4.5-pgfgcc
hdf5/1.8.10.patch1-gfortran netcdf4/4.2.1.1-gfortran    oasis3-mct/2.0-gfortran     openmpi/1.6.5-gfortran
[…]
---------------------------------------------- /etc/modulefiles/Products -----------------------------------------------
cdo/1.6.8(default)              matlab/2010b.sp2                nco/4.6.1
ferret/6.7.2(default)           matlab/2013b(default)           pyferret/7.0.0
ferret/6.9                      matlab/2015b                    python/2.7.3-epd7
[…]


> type pgf90
pgf90 est /opt/pgi-2013/linux86-64/2013/bin/pgf90

> module load pgi/2011

> type pgf90
pgf90 est /opt/pgi-2011/linux86-64/2011/bin/pgf90

> type matlab
matlab est /opt/matlab-2013b/matlab

> module load matlab/2016b
> type matlab
matlab est /opt/matlab-2016b/matlab
```

MESOCENTRE UTILITIES

module command

```
> module switch matlab/2010b.sp2
> type matlab
matlab est /opt/matlab-2010b.sp2/matlab

> module list
Currently Loaded Modulefiles:
  1) pgi/2011        2) matlab/2010b.sp2

> module purge
> type pgf90
-bash: type: pgf90 : non trouvé

> module list
No Modulefiles Currently Loaded.
```

Preference to use anaconda Python

```
# Load Anaconda system instance
> module load python/2.7anaconda

# Clone it
> conda create -n my_python --clone /opt/anaconda

# Activate the resulting Python virtual environment
source activate my_python

# After you could add any personal  package using:
> conda install pkg_name
> pip install pkg_name
> easy_install pkg_name

# Deactivate the environment
source deactivate my_python
```

Managing jobs

A job is a script (Shell, Perl, Python, etc. Not a binary)

To submit a job you need to ask some resources (memory, running time, number of cpu, etc.)

Not used asked resources is lost for everybody

You script run on another node , cpu and memory asked is dedicated .

Managing jobs

Batch system commands:

Function	Command
Launch a job	`qsub`
How many jobs in queue	`qstat / showq`
Kill a job	`qdel`
Nodes state	`check-cluster / pbsnodes`
Modify a job (not running)	`qalter`

Managing jobs

Execution queues

- `qstat -q` to show all queues definitions
- `qstat -Qf [queue-name]` to see full definiton
- 8 queues
- WALLTime are in Hour
- CPUTIME is unlimited

```
Queue              Memory CPU Time Walltime Node  Run Que Lm  State
---------------- ------ -------- -------- ----  --- --- --  -----
short              --      --    02:00:00   --    0   0 --   E R
default            --      --       --      --    0   0 --   E R
std                --      --    06:00:00   --    1   0 --   E R
h12                --      --    12:00:00   --    0   0 --   E R
day                --      --    24:00:00   --    0   0 --   E R
days3              --      --    72:00:00   --    0   0 --   E R
week               --      --    168:00:0   --    0   0 --   E R
weeks2             --      --    340:00:0   --    0   0 --   E R
infini             --      --    840:00:0   --    0   0 --   E R
```

Managing jobs

User limits (March 2017):

- No more than 24 actives Jobs or 24 cpu core
- No more than 3 active jobs in infini queue
- No more than 4 active jobs in weeks2 queue
- No more than 8 active jobs in week queue
- No more than 16 active jobs in days3 queue
- No more than 24 active jobs in other queue
- By default memory is limited to 4GB by process

Please don't stress interactive nodes you could do interactive work with batch system:

- `qsub -IV[X] [ -q short|std|h12 ]`

Managing jobs

Nodes status:

- **`pbsnodes -l`** — List out of order nodes or reserved node for sysadmin
- **`pbsnodes -a`** — List all nodes state
- **`check-cluster`** (local command)

```
> check-cluster
NODE          STATE          FREE-CPU     FREE-MEM     LOAD   PROPERTY    PARTITION
ciclad5       Running           1/8       10/31 Gb     7.08   [std]       std
ciclad10      Idle              8/8       31/31 Gb     0.24   [std]       std
ciclad18      Busy             0/32      26/126 Gb    32.13   [std]       std
ciclad19      Running         63/64     249/252 Gb     0.62   [heppi]     std
ciclad20      Idle            64/64     252/252 Gb     0.06   [heppi]     std
ciclad21      Idle            64/64     252/252 Gb     0.04   [heppi]     std
ciclad22      Running         32/64     152/252 Gb    32.05   [heppi]     std

 10 Active Jobs       72 of  304 Processors Active (23.68%)
                      4 of    7 Nodes Active      (57.14%)
Total Jobs: 12  Active Jobs: 10   Idle Jobs: 0   Blocked Jobs: 2
```

Managing jobs

`qstat` command:

- `qstat` — List all jobs (not only yours)
- `qstat -q` — Give queue definition information
- `qstat -a` — List all jobs with more info (should be a good default)
- `qstat -n` — List also node used by jobs
- `qstat -f num_job` — Give all info on one job

`showq` command:

- Could be complementary with `qstat`
- 3 states for a job
  - Running
  - Idle (no resources for running now but should start when free resources are sufficient)
  - Blocked (you are over your limits or asking impossible resources)
- `showq -i` — To see idle jobs
- `showq -b` — To see why jobs are blocked

Managing jobs

qdel command:

- `qdel job_num` — You can find job number by `qstat` or `showq`

qsub command:

- `qsub [-a date_time] [-A account_string] [-c interval] [-C directive_prefix] [-e path] [-h] [-I] [-j join] [-k keep] [-l resource_list][-m mail_options] [-M user_list] [-N name] [-o path] [-p priority] [-qdestination] [-r c] [-S path_list] [-u user_list] [-v variable_list][-V] [-W additional_attributes] [-z] [script]`
- `man qsub` — Display help
- `qsub -q queue_name my_shell_script`
- `qsub -IV` (`-VI` doesn't work anymore) — To launch interactive jobs
- `qsub -j oe` — Joining standard and error output in one file
- `qsub -k oe` — Keeping Standard and error output : could be see during the execution of your job but in this case those files are in the roots of your home (warn for quota if you have debug output in your job)

Managing jobs

```
# First thing to know is that only ShellScript (no binary files)
could be submited to batch system and by default script start in
your $HOME so you have to go in the good directory in your job
# Minimum shell script is:
#!/bin/bash
my_binary

# Shell script must be runable
> chmod +x my_shell_script
> qsub my_shell_script

# At the end of your jobs you have two files
# The standard output of your job and node computation
information ( node name running , cputime walltime and memory
used by your job
> cat my_shell_script.o$JOB_NUMBER

# The error output of your job in case of problems
> cat my_shell_script.e$JOB_NUMBER
```

Managing jobs

`qsub` command:

- `qsub -v variable_name=variable_value` — Permit to pass variables to shell script (argument on command line are not supported by torque)
- In the script you can use `$variable_name`
- You could put job directives directly in the script using line begining by `#PBS`:

```
#PBS -qweek
#PBS -k oe
#PBS -j oe
```

- `qsub -q queue_name my_shell_script`
- `qsub -j oe -k oe job.sh` — With this you could view in realtime output of your job in your home
- `tail -f job.sh.exxx`
- You can use IDL, Matlab, Scilab in batch jobs only if there is no X11 graphic in your job.
- Graphic should be produced directly in .ps, .eps, .pdf or image formats.

Managing jobs

qsub command:

- Default limit for jobs memory is :
  - 4GB of virtual memory per Job
  - 3GB of real memory per Job
- How to launch job with more memory ?
  - **mem** — Memory per job
  - **vmem** — Virtual memory per job

```
> qsub -l mem=10gb -l vmem=12gb
Mem:  32442980k total, 21798348k used, 10644632k free,    47112k buffers
Swap: 67111528k total,  2499232k used, 64612296k free, 14130248k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
12355 weill     25   0  9164m 8.1g 8.1g S   44  26.3   1:08.55
MATLAB
```

- Warning do not put those values to high without good reason

Managing jobs

`qsub` command:

- Starting a job at specific time:
  - `qsub -a time` :
  - `qsub -a 2010` — Lauch your job at 20:10
  - `qsub -a $(date -d '+2min' +%H%M)` — Launch this le job 2 minutes after submission. Use this when you have a job that submit another job at the end.
- Specifying `walltime` and `cputime` :
  - `qsub -l walltime=20:00:00,cput=80:00:00` — 20h of walltime with 80h of cputime
- Jobs dependancy: running a job after the successful run of another job[s]
  - `qsub -W depend=afterok:jobid[:jobid...]`

Managing jobs

`qsub` command:

- Submission of parallels jobs: `qsub -l nodes=n:ppn=m`
  - I want `n` nodes with `m` core by nodes
  - `qsub -l nodes=1:ppn=2+nodes=3:ppn=8` — Special request of one node with 2 core and 3 nodes with eight core

- Job MPI:
  - `mpirun` program
  - Batch system pass the number of CPU to MPI program
  - `-np` is not necessary except for special purpose.