CENTRALE NANTES SUPER COMPUTING

Hands-on Liger

Advanced session - SLURM

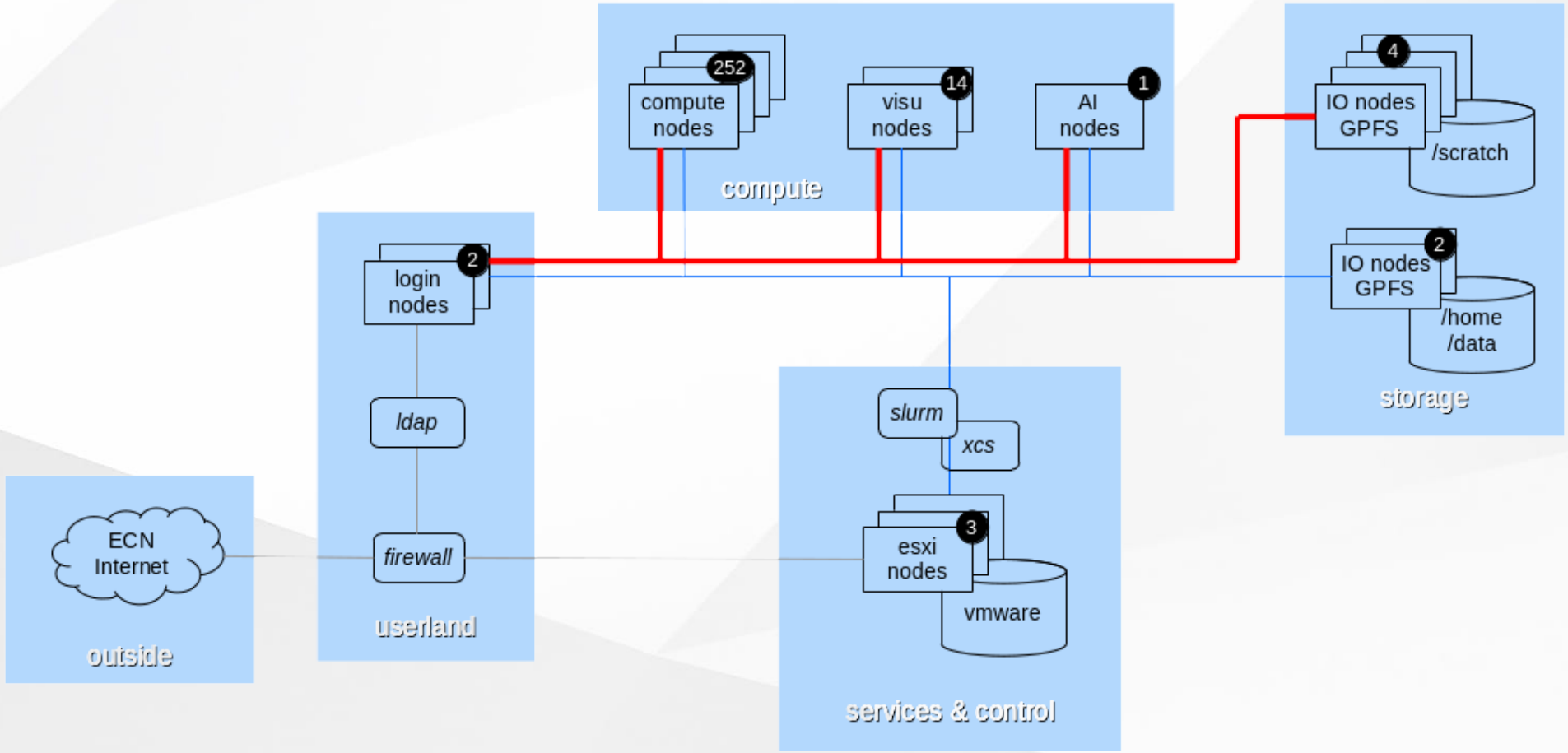
Pierre-Emmanuel Guérin
Davide Rovelli
Hugues Digonnet

https://supercomputing.ec-nantes.fr / @cnscfr

What is SLURM ?

SLURM means **S**imple **L**inux **U**tility for **R**esource **M**anagement.

- **Open-source** software created by LLNL and others
- **Resource scheduler** used to optimize HPC resources
- The most used in the world
- https://slurm.schedmd.com/
- https://github.com/SchedMD/slurm

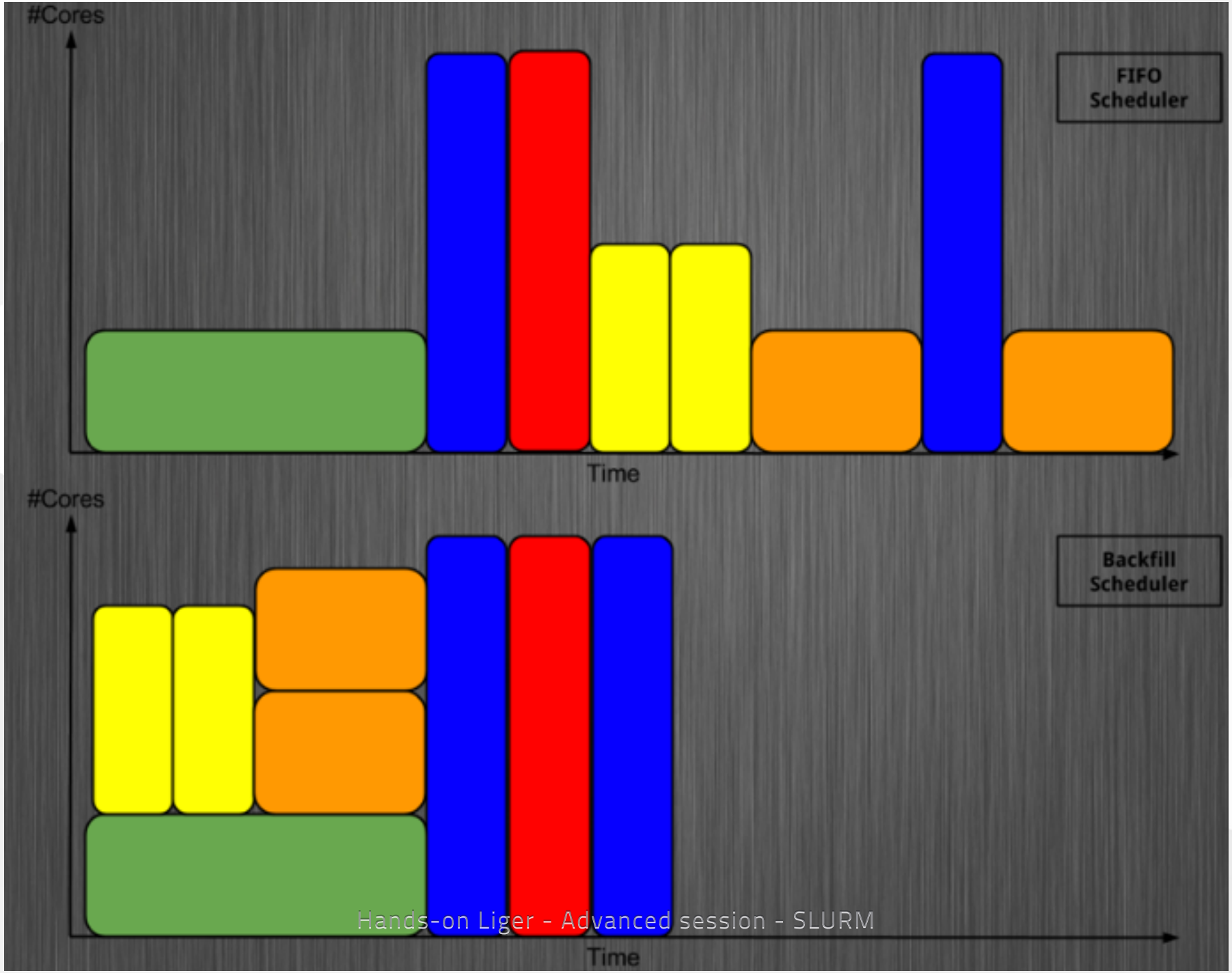

252
compute nodes
14
visu nodes
1
AI nodes
compute
4
IO nodes GPFS
/scratch
2
IO nodes GPFS
/home /data
storage
2
login nodes
ldap
firewall
userland
ECN Internet
outside
slurm
xcs
3
esxi nodes
vmware
services & control
Ethernet 1GB
IB FDR 56 GB
physical node
n
vm

Jobs evaluation order

1. Jobs that can preempt
2. Jobs with an advanced reservation
3. Partition PriorityTier
4. Job priority
5. Job submit time
6. Job ID

Running steps

1. Job put in the correct part of the queue (pending) according to the policy rules.
2. The scheduler checks if any jobs previously breaking policy rules, can now be considered for scheduling.
3. The scheduler calculates new priority for pending jobs.
4. If there are available processor resources the highest priority job(s) will be started.
5. If the highest priority job cannot start, the next job that fits, without changing the predicted startwindow for any higher priority jobs, will be started (backfilling).


#Cores
FIFO
Scheduler
Time
#Cores
Backfill
Scheduler
Time

Job states

You have 2 different (main) states with SLURM :

- RUNNING (R)
- PENDING (PD), check the reason with `squeue`

```
user@liger# squeue
JOBID PARTITION USER  ST TIME     N CPUS QOS  PRIORITY NODELIST(REASON)
1300903 compute user  PD  -        1    1 normal 403753 (Resources)
1300905 compute azerty R 21:18:22 1    1 normal 403753 node188
1300901 compute other  R 21:19:35 1    1 normal 403753 node001
```

Reservation types

You have 2 different types of reservations :

- exclusive : get entire node
- shared : get only needed cores (default mode)

```
[user@login02 ~]$ salloc
salloc: Granted job allocation 1333793

[user@login02 ~]$ Mysqueue
JOBID PARTITION USER ST TIME N CPUS QOS    PRIORITY NODELIST NAME
1333793 compute login R 0:14 1    1 normal   314671 node001  bash
```

```
[user@login02 ~]$ scontrol show job 1333793
JobId=1333793 JobName=bash
   UserId=login(1020) GroupId=ici(506)
   Priority=314671 Nice=0 Account=ici QOS=normal
   JobState=RUNNING Reason=None Dependency=(null)
   Requeue=0 Restarts=0 BatchFlag=0 Reboot=0 ExitCode=0:0
   RunTime=00:00:27 TimeLimit=01:00:00 TimeMin=N/A
   SubmitTime=2020-01-30T09:06:26 EligibleTime=2020-01-30T09:06:26
   StartTime=2020-01-30T09:06:26 EndTime=2020-01-30T10:06:26
   PreemptTime=None SuspendTime=None SecsPreSuspend=0
   Partition=compute AllocNode:Sid=login02:25901
   ReqNodeList=(null) ExcNodeList=(null)
   NodeList=node001
   BatchHost=node001
   NumNodes=1 NumCPUs=1 CPUs/Task=1 ReqB:S:C:T=0:0:*:*
   Socks/Node=* NtasksPerN:B:S:C=0:0:*:* CoreSpec=*
   MinCPUsNode=1 MinMemoryCPU=5G MinTmpDiskNode=0
   Features=(null) Gres=(null) Reservation=(null)
   Shared=OK Contiguous=0 Licenses=(null) Network=(null)
   Command=(null)
   WorkDir=/home/login
```

With exclusive mode

```
[user@login02 ~]$ salloc --exclusive
salloc: Granted job allocation 1333794

[user@login02 ~]$ Mysqueue
JOBID PARTITION USER ST TIME N CPUS QOS    PRIORITY NODELIST NAME
1333794 compute login R 0:03 1   24 normal   314671 node045  bash
```

```
[user@login02 ~]$ scontrol show job 1333794
JobId=1333794 JobName=bash
   UserId=login(1020) GroupId=ici(506)
   Priority=314671 Nice=0 Account=ici QOS=normal
   JobState=RUNNING Reason=None Dependency=(null)
   Requeue=0 Restarts=0 BatchFlag=0 Reboot=0 ExitCode=0:0
   RunTime=00:00:06 TimeLimit=01:00:00 TimeMin=N/A
   SubmitTime=2020-01-30T09:08:07 EligibleTime=2020-01-30T09:08:07
   StartTime=2020-01-30T09:08:07 EndTime=2020-01-30T10:08:07
   PreemptTime=None SuspendTime=None SecsPreSuspend=0
   Partition=compute AllocNode:Sid=login02:25901
   ReqNodeList=(null) ExcNodeList=(null)
   NodeList=node045
   BatchHost=node045
   NumNodes=1 NumCPUs=24 CPUs/Task=1 ReqB:S:C:T=0:0:*:*
   Socks/Node=* NtasksPerN:B:S:C=0:0:*:* CoreSpec=*
   MinCPUsNode=1 MinMemoryCPU=5G MinTmpDiskNode=0
   Features=(null) Gres=(null) Reservation=(null)
   Shared=0 Contiguous=0 Licenses=(null) Network=(null)
   Command=(null)
   WorkDir=/home/login
```

Cancel a job

```
[user@login02 ~]$ scancel 1333794
salloc: Job allocation 1333794 has been revoked.
```

Show running jobs

```
[user@login02 ~]$ squeue
JOBID PARTITION USER ST   TIME   N CPUS QOS  PRIORITY NODELIST   NAME
1335738 compute ogvh R 13:55:44 8 192 normal 397871 node[067-074] LG
1335737 compute ofiv R 14:07:34 5  96 normal 397866 node[062-066] run8
1334818 compute eaxj R 19:57:38 4  96 normal 394546 node[050-053] test
1335082    visu ofiv R 16:56:33 1   1 normal 392328 viz01          Job17
1335055    visu ofiv R 17:42:07 1   1 normal 392271 viz01          Job17
1334793 compute udcm R 20:44:45 1   1 normal 252116 node005        test
1335271 compute udcm R 20:48:13 1   1 normal 239714 node005        test
1335398 compute udcm R 13:58:10 3  60 normal 236858 node[037-039] test
1335452 compute exia R 09:04:23 1   1 normal 234187 node040        test
1335927 compute udcm R  4:00:22 2  36 normal 227338 node[024-025] test
1335484 compute bnbe R 00:08:03 1   8 lheea   89638 node007        test
1335485 compute bnbe R 23:53:13 1   8 lheea   89630 node007        test
```

Priorities

Job prioritization factors

- **Age** : the length of time a job has been waiting in the queue, eligible to be scheduled
- **Association** : a factor associated with each association
- **Fair-share** : the difference between the portion of the computing resource that has been promised and the amount of resources that has been consumed
- **Job size** : the number of nodes or CPUs a job is allocated
- **Nice** : a factor that can be controlled by users to prioritize their own jobs. Same usage as Linux.

Job prioritization factors

- **Partition** : a factor associated with each node partition
- **QOS** : a factor associated with each Quality Of Service
- **Site** : a factor dictated by an administrator or a site-developed job_submit or site_factor plugin
- **TRES** : each TRES Type has its own factor for a job which represents the number of requested/allocated TRES Type in a given partition

Priority calculation

```
Job_priority =
        site_factor +
        (PriorityWeightAge) * (age_factor) +
        (PriorityWeightAssoc) * (assoc_factor) +
        (PriorityWeightFairshare) * (fair-share_factor) +
        (PriorityWeightJobSize) * (job_size_factor) +
        (PriorityWeightPartition) * (partition_factor) +
        (PriorityWeightQOS) * (QOS_factor) +
        SUM(TRES_weight_cpu * TRES_factor_cpu,
            TRES_weight_<type> * TRES_factor_<type>,
            ...)
        - nice_factor
```

Priority details

`sprio` to see the detailed priorities of pending jobs.

```
$ sprio
  JOBID PRIORITY  AGE FAIRSHARE JOBSIZE  PARTITION     QOS
  65539    62664    0     51664    1000      10000       0
  65540    62663    0     51663    1000      10000       0
  65541    62662    0     51662    1000      10000       0
```

Nodes availability

```
[user@login02 ~]$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
compute*     up 15-00:00:0      1  down* node006
compute*     up 15-00:00:0      3  drain node[026,035,049]
compute*     up 15-00:00:0      3   resv node[032,047,054]
compute*     up 15-00:00:0      6    mix node[025,040-041,045,048,066]
compute*     up 15-00:00:0     40  alloc node[003,005,007-018,...]
compute*     up 15-00:00:0    199   idle node[001-002,004,019-024,...]
[...]

[user@login02 ~]$ sinfo -Rl
Thu Feb  6 09:19:23 2020
REASON               USER         TIMESTAMP           STATE  NODELIST
Node unexpectedly re root(0)      2020-01-14T14:10:16 down*  node006
NHC: check_hw_mcelog root(0)      2020-02-04T16:21:46 drain  node026
NHC: check_hw_mcelog root(0)      2020-02-02T06:11:46 drain  node035
NHC: check_hw_mcelog root(0)      2020-01-31T17:01:46 drain  node049
```

```
[user@login02 ~]$ scontrol show part
PartitionName=compute
   AllowGroups=ALL AllowAccounts=ALL AllowQos=ALL
   AllocNodes=ALL Default=YES
   DefaultTime=01:00:00 DisableRootJobs=NO GraceTime=0 Hidden=NO
   MaxNodes=UNLIMITED MaxTime=15-00:00:00 MinNodes=1 LLN=NO MaxCPUsPerNode=UNLIMITED
   Nodes=node[001-252]
   Priority=1 RootOnly=NO ReqResv=NO Shared=YES:4 PreemptMode=OFF
   State=UP TotalCPUs=6048 TotalNodes=252 SelectTypeParameters=N/A
   DefMemPerCPU=5120 MaxMemPerNode=117760

PartitionName=visu
   AllowGroups=ALL AllowAccounts=ALL AllowQos=ALL
   AllocNodes=ALL Default=NO
   DefaultTime=01:00:00 DisableRootJobs=NO GraceTime=0 Hidden=NO
   MaxNodes=2 MaxTime=3-00:00:00 MinNodes=1 LLN=NO MaxCPUsPerNode=UNLIMITED
   Nodes=viz[01-14]
   Priority=1 RootOnly=NO ReqResv=NO Shared=NO PreemptMode=OFF
   State=UP TotalCPUs=336 TotalNodes=14 SelectTypeParameters=N/A
   DefMemPerCPU=10240 MaxMemPerNode=235520
```

Which accounts can I use ?

```
[user@login02 ~]$ id
uid=1020(login) gid=506(ici) groups=506(ici),501(ecn),546(TPCALCUL)
```

Use `-a` option for running computation with another project.

Run a very simple job

```
[user@login02 ~]$ srun hostname
node045
```

How to run an automatic serial job ?

```
[user@login02 ~]$ cat /softs/liger/slurm/examples/serial_job.slurm

#!/bin/bash
#SBATCH -J TestJob
#SBATCH -N 1
#SBATCH -o TestJob%j.out
#SBATCH -e TestJob%j.err
#SBATCH --time=0-00:30:00

sleep 5
hostname
```

```
[user@login02 ~]$ sbatch /softs/liger/slurm/examples/serial_job.slurm
```

How to run an automatic parallel job ?

```
[user@login02 ~]$ cat /softs/liger/slurm/examples/parallel_job.slurm
#!/bin/bash
#SBATCH -J TestJob
#SBATCH -N 4
#SBATCH -o TestJob-%j.out
#SBATCH -e TestJob-%j.err
#SBATCH --time=0-00:60:00

srun --ntasks-per-node=1 hostname
```

```
[user@login02 ~]$ sbatch /softs/liger/slurm/examples/serial_job.slurm
```

Specific variables for slurm scripts

```
$SLURM_ARRAY_JOB_ID    (job id for the array)
$SLURM_ARRAY_TASK_ID   (job array index value)
$SLURM_CPUS_PER_TASK   (CPUs per MPI task)
$SLURM_JOB_ACCOUNT     (account name)
$SLURM_JOB_ID          (job id)
$SLURM_JOB_NAME        (job name)
$SLURM_JOB_NODELIST    (names of nodes allocated to job)
$SLURM_JOB_NUM_NODES   (number of nodes allocated to job)
$SLURM_JOB_PARTITION   (partition running the job)
$SLURM_JOB_UID         (job owner user id)
$SLURM_JOB_USER        (job owner name)
$SLURM_NNODES          (number of nodes)
$SLURM_NTASKS          (number of MPI tasks)
$SLURM_PROCID          (task ID with MPI rank)
$SLURM_SUBMIT_DIR      (directory job was submitted from)
$SLURM_WORKING_DIR     (change your working directory)
```

Computation usage (Liger only)

```
[user@login02 ~]$ Mybalance
Fri Feb  7 09:53:51 CET 2020
Account = Liger Project Id (LIPID) - Usage and limit are in CPU hrs
(1) start from 2020-01-01T00:00:00 till 2020-02-06T23:59:59
User    Usage(1)|  Account   Usage | Account Limit Available (CPU hrs)
------- ------- + --------- ------ + ------------- ----------
 login       26 |     ici   117809 |
```

Information about completed jobs

```
[user@login02 ~]$ sacct
  JobID   JobName   Partition   Account  AllocCPUS      State ExitCode
------- --------- --------- -------- ---------- ---------- --------
1336201   hostname   compute       ici          1  COMPLETED      0:0
1336202   hostname   compute       ici          1  COMPLETED      0:0
```

Specific cases (advanced usage)

- Run the same job many times
  - JobArray : `--array=0-31`
- Run a job after a specific one
  - Dependencies : `-d after:job_id`
- Get de specific node : `--nodelist=<nodeXXX,...>`
- Exclude specific node : `--exclude=<nodeXXX,...>`

Parallelism methods

- MPI : distributed memory
  - use `srun --ntasks=value` and MPI lib in your code
- OpenMP : shared memory
  - use OpenMP pragma in your code and :

```
# to put in your slurm script
#SBATCH --cpus-per-task=value
if [ -n "$SLURM_CPUS_PER_TASK" ]; then
  omp_threads=$SLURM_CPUS_PER_TASK
else
  omp_threads=1
fi
export OMP_NUM_THREADS=$omp_threads
```

Reminder

sacct	show finished jobs
salloc	run interactive mode job
sbatch	run script for automatic job
scancel	cancel a job
scontrol	job informations
sinfo	view partitions and nodes
slurmtop	view cluster load

smap	view jobs and reservations (CLI mode)
sprio	view priority details for pending jobs
squeue	view all jobs (running or pending)
srun	run a single command job
Mysqueue	view only your jobs (Liger only)
Mybalance	track you consumption (Liger only)