Using the general purpose computing clusters at EPFL

scitas.epfl.ch

February 27, 2018

Welcome

What you will learn

- What is a cluster
- What is a scheduler
- How the environment is organised
- How to run simple jobs on our clusters

What you will not learn

- Writing and compiling codes
- Parallelising code

What is a cluster?

Our clusters

Login	**Hosts**	**Cores**	**RAM**	**Network**	**Storage**
hostname	#	# x GHz	GB	Gbit/s	TB
`castor.epfl.ch`	50	16x2.6	64	10 (Eth)	25
	2		256		
`deneb{1,2}.epfl.ch`	376	16x2.6	64	40 (IB)	350
	144	24x2.5			
	16	16x2.6	+ *4x NVidia K40*		
	8	16x2.6	256		
	2	32x2.6	512		
`fidis.epfl.ch`	335	28x2.6	128	56 (IB)	375
	72		256		

Shared Storage (cluster)

/scratch

- high performance temporary space
- is not backed up
- low redundancy, built for performance
- local to each cluster
- automatically cleanup procedure deletes files without warning
- **for disposable files: intermediary results, temporary files**

/scratch/username

You all have a directory on the scratch filesystem

Shared Storage (global)

/home

- filesystem has per user quotas of 100GB
- backed up to a remote site
- shared, available on all clusters
- **for important files: source code, final results, theses**

Connecting to a cluster

```
ssh -X username@deneb1.epfl.ch
```

- username is your gaspar login
- connect to `deneb1` or `deneb2`
- Linux: connect using `ssh`
- Windows: install and start X server (XMing/Xwin32), connect using PuTTY (with X11 Forwarding enabled)
- OSX: install and start X server (XQuartz), connect using `ssh`

X Forwarding is not strictly needed, but it might be useful.

Batch Systems

Batch

Goal: to take a list of jobs and execute them according to a priority when appropriate resources become available

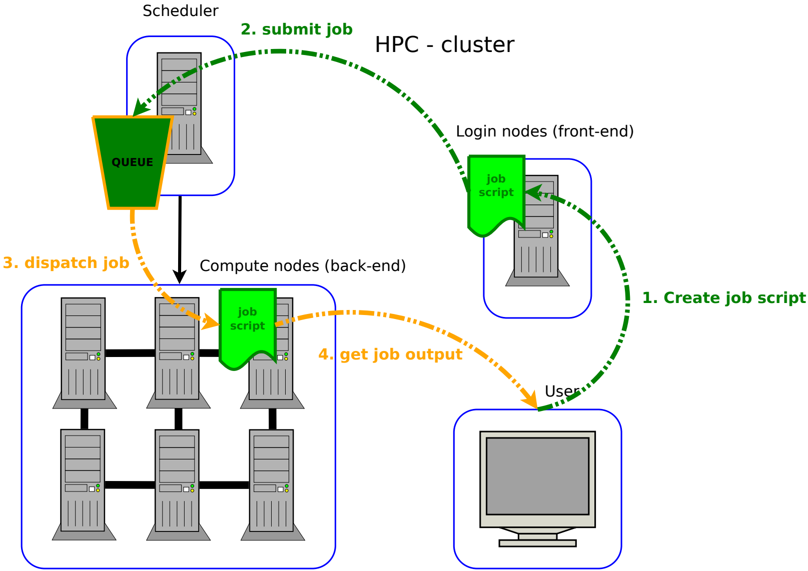
Interactive use is possible but it not the principal way of running jobs!

SLURM

We use SLURM on all our clusters. It's widely used in the HPC world and open source.

`http://slurm.schedmd.com`

Workflow of a job



sbatch

sbatch

The fundamental command is `sbatch` which submits jobs to the batch system.

Worflow

A typical workflow to get your computation done is:

- create a short job-script
- submit it to the batch system
- *it will get executed*
- look at the output

The job **will wait in the queue** until resources are available to run it.

sbatch

civil-459 sbatch directives

```
#!/bin/bash
#SBATCH --workdir /scratch/<put-your-username-here>
#SBATCH --nodes 1
#SBATCH --ntasks 1
#SBATCH --cpus-per-task 1
#SBATCH --mem 8G
#SBATCH --partition gpu
#SBATCH --gres gpu:1
#SBATCH --qos gpu_free
#SBATCH --account civil-459
#SBATCH --reservation civil-459-lab

echo "hello from $(hostname)"
```

SLURM directives

#SBATCH: directive to the batch system

```
--nodes 1
# the number of nodes to use

--ntasks 1
# the number of tasks (in an MPI sense) to run per job

--cpu-per-task 1
# the number of cores per aforementioned task

--mem 4096
# the memory required per node in MB

--time 12:00:00
# the time required in hours:minutes:seconds
```

See the sbatch documentation for the full details!

Submitting a job

sbatch myjob.sh

```
$ sbatch myjob.sh
Submitted batch job 1617329

$ cat /scratch/eroche/slurm-1617329.out
hello from r08-nodegpu01
```

Cancelling jobs

scancel

To cancel a specific job:

```
    scancel <JOB_ID>
```

To cancel all your jobs:

```
    scancel -u <username>
```

To cancel all your jobs that are not yet running:

```
    scancel -u <username> -t PENDING
```

What's going on?

squeue

With no arguments squeue will list all jobs currently in the queue! The output and information shown can be refined somewhat by giving options.

- `squeue -t R -u username`
- `squeue -t PD -u username`
- `squeue -t PD -u username --start`

Squeue

Squeue is an custom squeue that shows only your jobs with more useful information.

Modules

Why is a module system needed

- The OS version is restricted to an older one due to compatibility requirements of storage systems and specialized interconnects.
- The above is often in direct conflict with the needs of the HPC community, for which newer versions bring performance improvements and support for newer hardware (new CPU features).
- Many scientific codes are not even packaged under most Linux distributions.

Lmod

- **Lmod** is a utility that allows multiple, often incompatible, tools and libraries to co-exist on a system.
- It's a backwards-compatible evolution of the older GNU Modules.

Modules

How software is organised under Lmod

- Packages are organized hierarchically: `Compiler / MPI / blas`
- **Lmod** is designed to maintain the environment consistent
- **Lmod** does everything possible to automatically reload any software when one of the hierarchy layers is changed

Basic commands

- `module avail`
- `module load / unload <module-name>`
- `module spider <name>`
- `module save / restore <mnemonic-name>`
- `module purge`

python with modules

python3

```
$ python3 --version
-bash: python3: command not found

$ module load gcc python

$ module list
Currently Loaded Modules:
  1) gcc/5.4.0   2) python/3.6.1

$ python3 --version
Python 3.6.1
```

For tools such as python make sure that you are using the version provided by modules!

Interactive access

Why interactive?

For debugging or running applications such as ipython interactively we don't want to submit a batch job.

Sinteract or salloc

There are two main ways of getting access depending on what you want to achieve:

- `Sinteract` - custom tool to access a node
- `salloc` - standard tool for an interactive allocation

Behind the scenes both use the same mechanism as `sbatch` to get access to resources.

Sinteract

Sinteract --help

```
usage: Sinteract [-c cores] [-n tasks] [-t time] [-m memory]
[-p partition] [-a account] [-q qos] [-g resource] [-r reservation]
options:
  -c cores       cores per task (default: 1)
  -n tasks       number of tasks (default: 1)
  -t time        as hh:mm:ss (default: 00:30:00)
  -m memory      as #[K|M|G] (default: 4G)
  -p partition   (default: parallel)
  -a account     (default: scitas-ge)
  -q qos         as [normal|gpu|gpu_free|mic|...] (default: )
  -g resource    as [gpu|mic][:count] (default is empty)
  -r reservation reservation name (default is empty)

examples:
    /usr/bin/Sinteract -c 16 -p serial
    /usr/bin/Sinteract -p gpu -q gpu_free -g gpu:1
```

Sinteract

Sinteract

```
[user@deneb1 ]$ /usr/bin/Sinteract -p gpu -q gpu_free -g gpu:1 -a civil-459 -r civil-459-lab
Cores:            1
Tasks:            1
Time:             00:30:00
Memory:           4G
Partition:        gpu
Account:          civil-459
Jobname:          interact
Resource:         gpu:1
QOS:              gpu_free
Reservation:      civil-459-lab

salloc: Pending job allocation 1617334
salloc: Granted job allocation 1617334
salloc: Waiting for resource configuration
salloc: Nodes r08-nodegpu01 are ready for job
[user@r08-nodegpu01 ]$
```

civil-459 Notes

reservation civil-459-lab

- 6 GPU nodes with a total of 24 GPUs
- Wednesday mornings 08h00 until 13h00
- To allow you to work interactively

reservation civil-459-project

- 2 GPU nodes with a total of 8 GPUs
- All day every day until the end of the semester
- To allow you to submit jobs

Note on QoS

The gpu_free QoS limits you to 12 hours per job and one running job at a time.

Getting Help

`man` pages are your friends!

- `man sbatch`
- `man gcc`

Ask your TA

- All problems should be directed to the TA
- If needed they will contact us
- Do not send mails to scitas!

Useful links

links

Change your shell at:

`https://cadiwww.epfl.ch/cgi-bin/accountprefs/`

SCITAS web site:

`http://scitas.epfl.ch`

(in particular) SCITAS documentation space:

`http://scitas.epfl.ch/kb`

SLURM man pages:

`http://slurm.schedmd.com/man_index.html`