UNIL | Université de Lausanne

Using the DCSR Clusters

| le savoir vivant |

mardi 21 janvier 2020

What you will learn

How to interact with a cluster

Different ways of using the clusters

Run simple and not so simple jobs on 1 node

The DCSR cluster environment

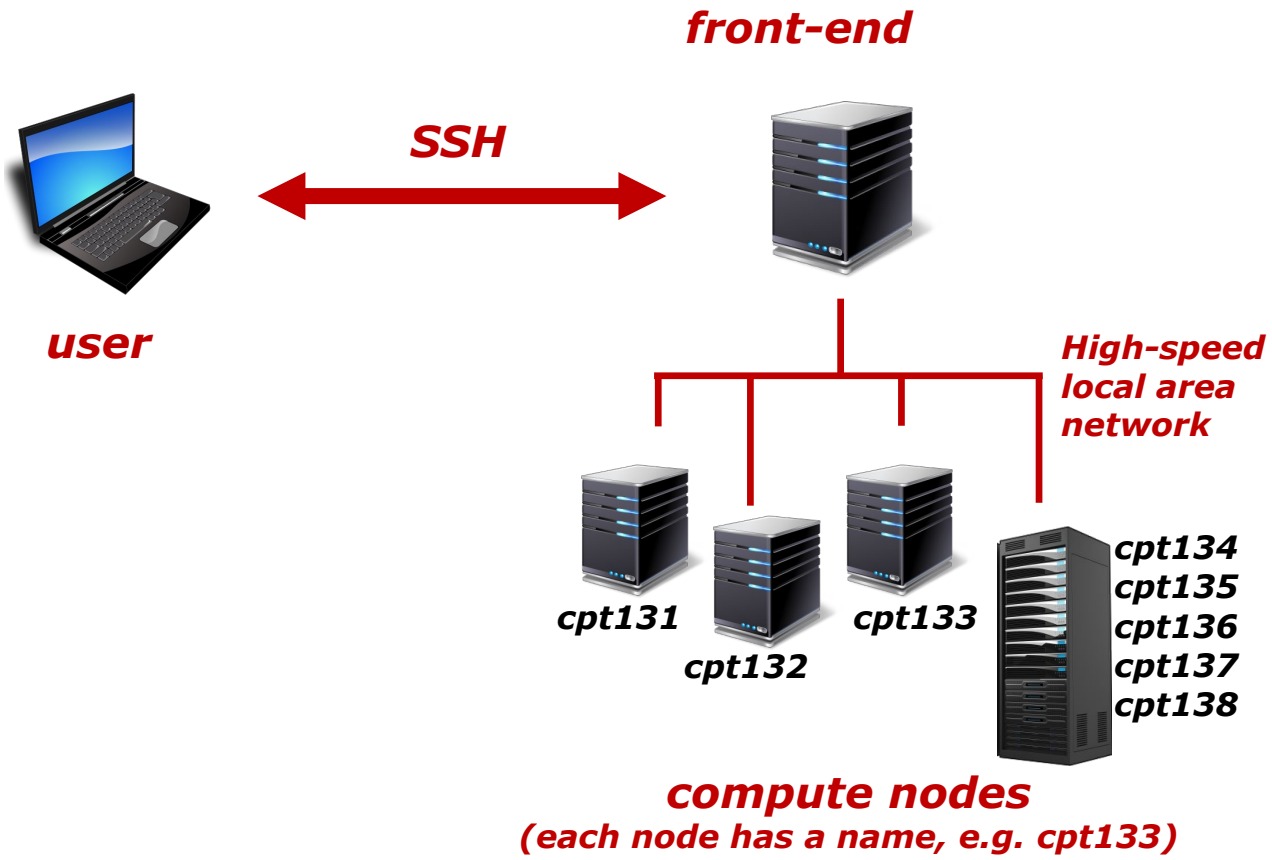
What you will not learn

Running multi-node (MPI) jobs

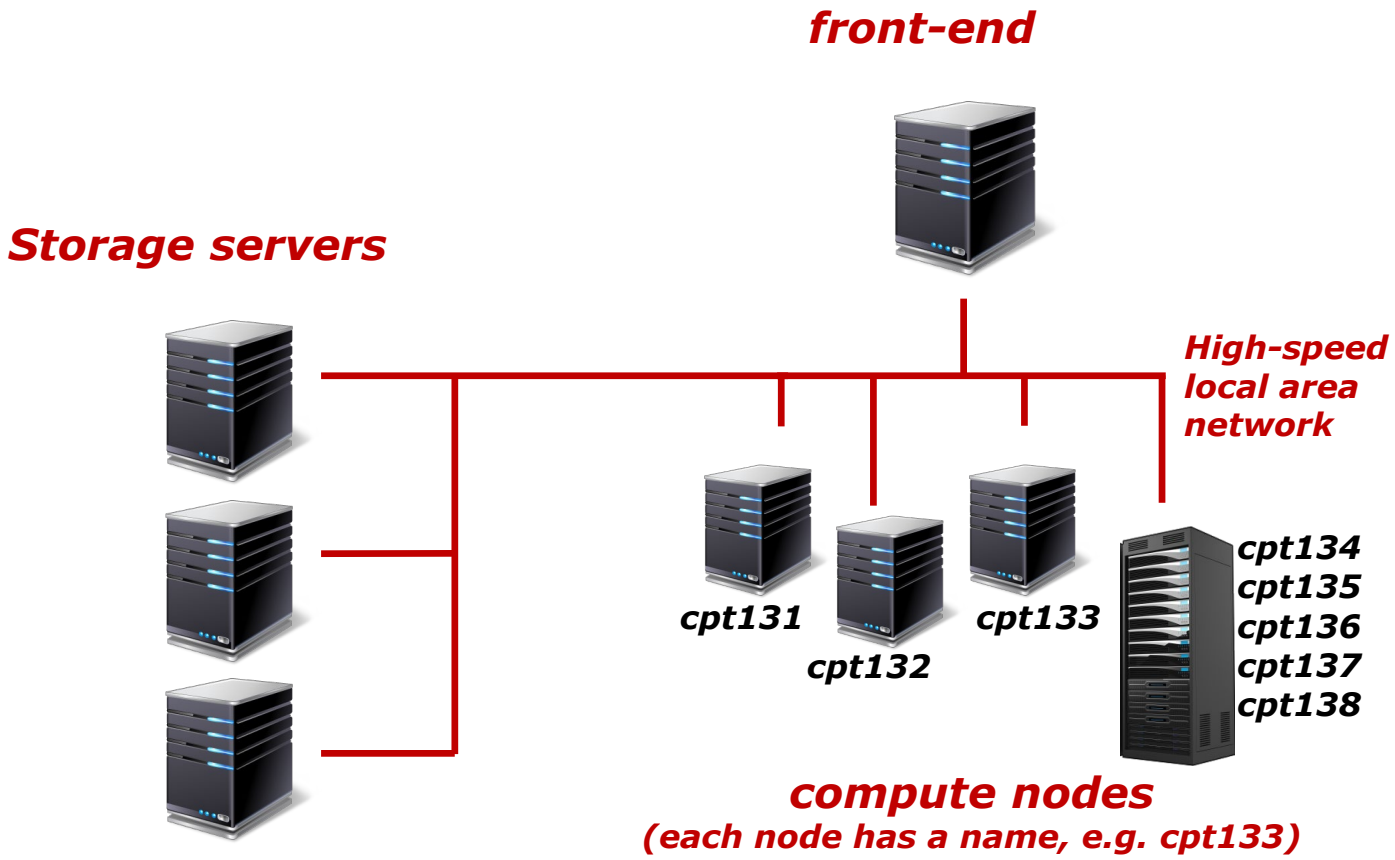
Compiling codes

Programming

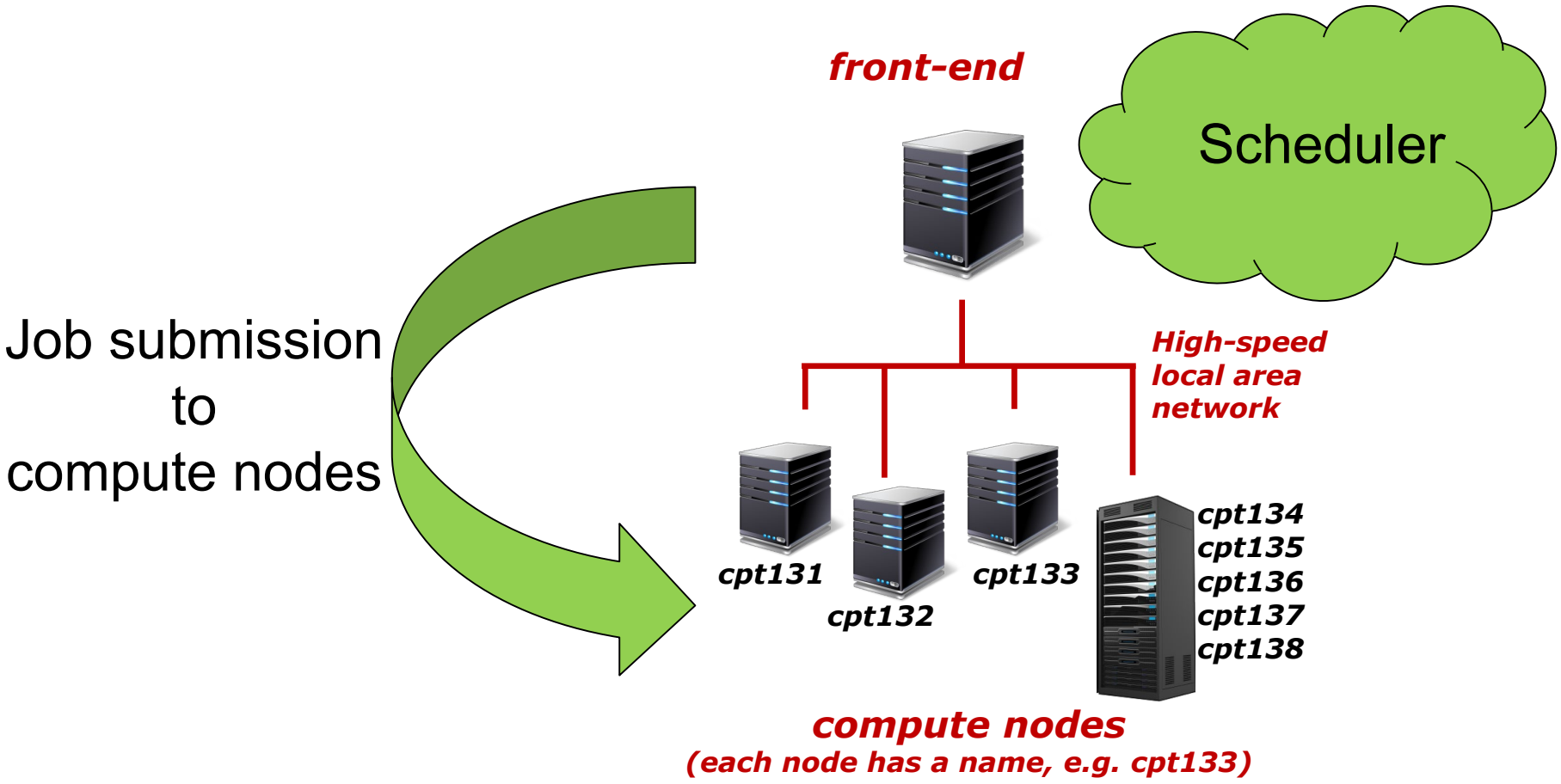
Anatomy of a Cluster



Anatomy of a Cluster



Anatomy of a Cluster



The DCSR Clusters

Wally – General purpose cluster

Axiom – Specialised machines (lots of memory)

Jura – Sensitive data cluster

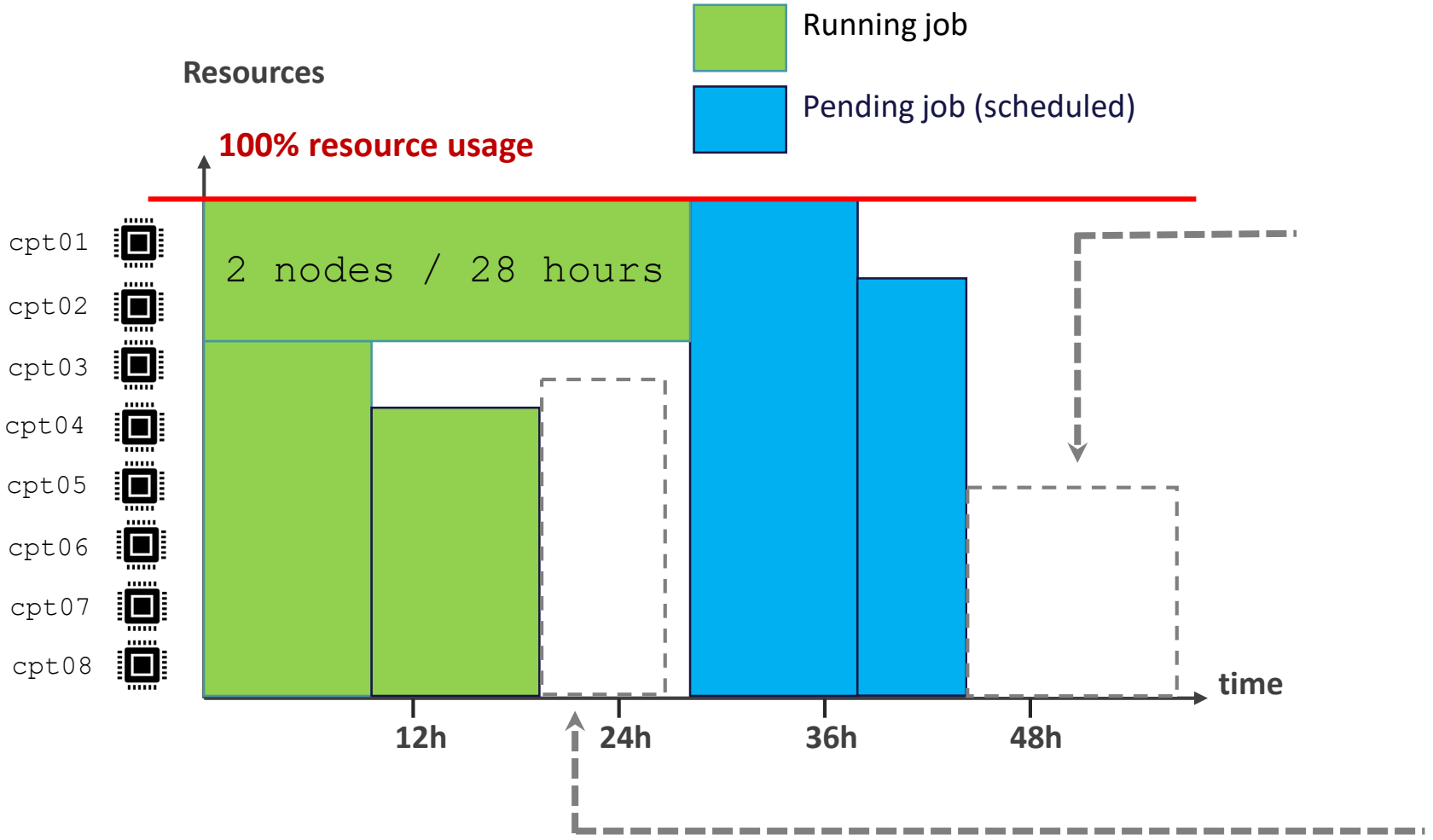
Anatomy of a compute node

Socket – a physical slot where a CPU is plugged in

CPU – an integrated circuit that contains many cores as well as memory management and often graphics

Core – the thing that actually runs your code

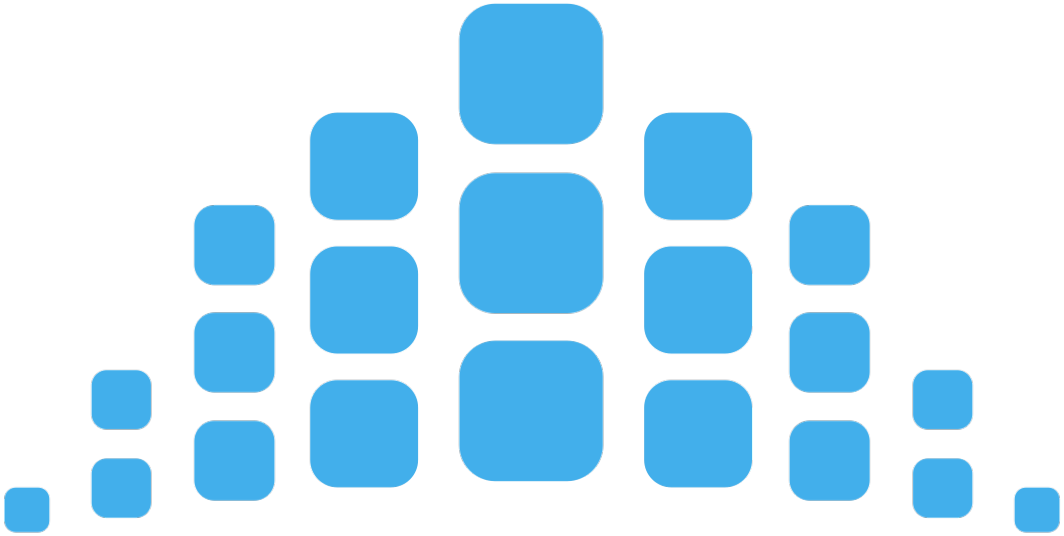
Schedulers


Running job
Pending job (scheduled)
Resources
100% resource usage
cpt01
cpt02
cpt03
cpt04
cpt05
cpt06
cpt07
cpt08
2 nodes / 28 hours
time
12h
24h
36h
48h

SLURM

Batch System and Scheduler

Widely used in research and academia



slurm

workload manager


Slurm

Wally and Axiom

Wally and Axiom have separate login nodes and scratch storage but are part of the same SLURM instance

wally-front1.unil.ch and /scratch/wally

axiom-front1.unil.ch and /scratch/axiom

Batch or Interactive?

Do I really want to sit in front of the computer all day?

What if there was a way I could go to Zelig while my analysis is running?

Job Scripts

What do I want to do and what resources do I need?

Job Scripts

```
ssh <user>@wally-front1.unil.ch
```

```
git clone
https://c4science.ch/source/DCSR-Examples.git
```

Anatomy of a job script

Dear Computer

Please give me 1 node

With 1 task on that node

And 1 compute core for that task

Don't forget to say hello!

Anatomy of a job script

```
#!/bin/bash


Please give me 1 node
With 1 task on that node
And 1 compute core for that task


Don't forget to say hello!
```

Anatomy of a job script

```
#!/bin/bash


#SBATCH --nodes 1
With 1 task on that node
And 1 compute core for that task


Don’t forget to say hello!
```

Anatomy of a job script

```
#!/bin/bash


#SBATCH --nodes 1
#SBATCH --ntasks 1
And 1 compute core for that task


Don’t forget to say hello!
```

Anatomy of a job script

```
#!/bin/bash


#SBATCH --nodes 1
#SBATCH --ntasks 1
#SBATCH --cpus-per-task 1


Don’t forget to say hello!
```

Anatomy of a job script

```
#!/bin/bash


#SBATCH --nodes 1
#SBATCH --ntasks 1
#SBATCH --cpus-per-task 1


echo “hello from `hostname` at `date`”
```

s and S

All slurm commands begin with an s

DCSR provided slurm commands begin with S

sbatch, squeue, sinfo, scancel, srun and many more

Sinteractive, Squeue with more coming soon!

sbatch

Dear SLURM,

Please take my job script and run it as and when the resources I asked for become available.

Thanks

Dr Ursula Lambda

sbatch

```
[ulambda@login1 ~]$ sbatch hello.run
Submitted batch job 1234567
```

sbatch

```
[ulambda@login1 ~]$ cat slurm-1234567.out

hello from cpt033.wally.unil.ch at Thu Jan 16
17:43:34 CET 2020
```

Exercise: sbatch

$ sbatch hello.run

Note the Job ID!

Wait a minute and look at the output

Working directory

By default the working directory is where you ran sbatch

/users/ulambda

This is probably not a good thing

#SBATCH --chdir /scratch/wally/foo/bar/project

Working directory

```
$ cd /scratch/wally/foo/bar/project
$ sbatch myjob.run
```

or

```
#SBATCH --chdir /scratch/wally/foo/bar/project
```

What's going on?

`squeue`

shows all jobs in the queue

`Squeue`

shows your jobs with useful information

`scontrol -dd show job <JOB ID>`

shows (almost) everything about the job

What's going on?

Your jobs are normally in one of two states

PENDING (PD)

or

RUNNING (R)

I didn’t want to do that…

scancel 12345678

scancel -u ulambda -t PD

scancel -u ulambda -t R

Exercise: squeue and scancel

$ sbatch long_exclusive.run

$ squeue

$ Squeue

$ scancel

Sinteractive

"I need access to a compute node to set up my analysis and check that things work as expected"

Sinteractive

```
$ Sinteractive -R HPC-course -A rfabbret_cours_hpc

Sinteractive is running with the following options:
-A rfabbret_cours_hpc --reservation HPC-course -c 1 --mem 1G
-J interactive -p normal -t 1:00:00

salloc: Granted job allocation 2079911
salloc: Waiting for resource configuration
salloc: Nodes cpt003 are ready for job

[ulabmda@cpt003] $
```

Sinteractive

Behind the scenes Sinteractive still uses the batch system so you might have to wait

Job Arrays

"I have 1000 jobs and the only thing that changes is the input"

1000 job scripts or 1 job script with 1000 elements

Technical Interlude

In order to explain Job Arrays
we need to know a little bit
about shell scripting

Variables

fromage=etivaz

echo ”The best cheese is **$fromage**”

The best cheese is **etivaz**

SLURM Variables

A way for SLURM to pass information to job scripts.

SLURM_TASKS_PER_NODE=1

SLURM_CPUS_PER_TASK=1

SLURM_NPROCS=1

and so on – over 50 in total

SLURM Variables

They are normal shell variables so we access them with `$VARIABLE_NAME`

```
echo $SLURM_JOB_ACCOUNT
```

Job Arrays

```
#!/bin/bash

#SBATCH --nodes 1
..
..
#SBATCH --array=1-1000

echo ${SLURM_ARRAY_TASK_ID}
```

Exercise: Job Arrays

$ sbatch array.run

$ Squeue

What do the output files look like?

Job Arrays

That’s nice but how do I actually use Job Arrays?

Job Arrays

We use the Array ID to select different inputs

Job Arrays

"I need to analyse 100 different input datasets"

Job Arrays

"I need to run my code with 100 different input values"

Job Arrays

Prepare a file with one input (variables or filename) per line and apply the following recipe:

```
IN=$(sed -n ${SLURM_ARRAY_TASK_ID}p in.list)

echo "Running analysis on $IN”

mycode.x $IN
```

Job Arrays

```
$ cat in.list


Rock_GraniteGneiss.in
Rock_FeldsparDolerite.in
Rock_QuartzGranite.in
Rock_SchistSandstone.in
..
..
```

Job Arrays

```
$ cat in.list

-A 10 -B 21 -C 16
-A 10 -B 22 -C 54
-A 11 -B 13 -C 71
-A 11 -B 64 -C 98
..
..
```

Exercise: Job Arrays

$ sbatch arrayselect.run

$ Squeue

Check the script, input file and output

Job Arrays

On the DCSR clusters the maximum number of array elements is 5000

Job Dependencies

"I want to run this job once all of my array jobs have finished but it's nice and warm in Zelig and I left my laptop in the office"

Job Dependencies

```
$ sbatch array.run
Submitted batch job 1234
```

We take the Job ID and use it:

```
$sbatch --depend=afterany:1234 post_array.run
```

Partitions and Limits

Partition ~ Physical group of compute nodes

…but…

We can have multiple partitions on the same nodes but with different resource limits

Partitions and Limits

normal – wally for 1 day

long – wally for 10 days

ax-normal – axiom for 1 day

ax-long – axiom for 10 days

Partitions and Limits

If nothing is specified then it's the same as

```
#SBATCH --partition normal
#SBATCH --time 12:00:00
```

This may or may not be what you want and the defaults can change without warning.

Software on the Clusters

A cluster without software isn’t much use

Modules

A utility to allow multiple conflicting software packages to live happily together

The standard software management tool on clusters but implementation details vary wildly

Modules

$ module avail

$ module load <MODULE NAME>

$ module list

$ module purge

Exercise: Modules

$ sbatch module.run

Have a look at the script

Why do we start with “module purge”?

Modules

Modules can also make other modules appear

To query what software is available use vit_soft tool

```
$ vit_soft -s <PART OF NAME>
$ vit_soft -s BLAS
```

(This will be replaced by in the near future)

Modules on Wally & Axiom

For historical reasons:

```
$ module load HPC/Software
```

```
$ module load Bioinformatics/Software/vital-it
```

This will also be changing in the near future

Modules on Wally & Axiom

```
$ module load HPC/Software

$ module load R/3.6.1

$ R --version
R version 3.6.1 (2019-07-05) -- "Action of the Toes”
```

MATLAB

MATLAB is only installed on the front-end nodes

To run MATLAB jobs on the cluster you must first compile your .m files and then use the MATLAB runtime

See our documentation for the recipe!

Front nodes

`MyMatlabScript.m`

Compile with Matlab mcc

`MyMatlabScript`
`run_MyMatlabScript.sh`

Compute nodes

Run with Matlab Runtime

```
sh run_MyMatlabScript.sh $MCR_PATH
```

$MCR_PATH

- `$ module load\ Development/Languages/Matlab_Compiler_Runtime/v96`

- The version has to match the version of Matlab used to compile the code!

https://ch.mathworks.com/products/compiler/matlab-runtime.html

https://ch.mathworks.com/help/compiler/mcc.html

Node level Parallelism

Some programs can and will make use of more than one CPU core at a time

Keywords: Multi-threaded / OpenMP

Check the program documentation for all the details

Node level Parallelism

```
#SBATCH --nodes 1
#SBATCH --ntasks 1
#SBATCH --mem-per-cpu 2048
#SBATCH --cpus-per-task 16

# run the SpeedAnalyse code with 16 threads

./SpeedAnalyse --nthreads=16
```

Node level Parallelism

```
#SBATCH --nodes 1
#SBATCH --ntasks 1
#SBATCH --mem-per-cpu 2048
#SBATCH --cpus-per-task 16

export OMP_NUM_THREADS=$SLURM_CPUS_PER_TASK

./SpeedAnalyseOpenMP
```

HELP!

To: helpdesk@unil.ch

Subject: DCSR problem with MATLAB on Wally

Dear DCSR,

...

...

Merci beaucoup, Dr Ursula Lambda

HELP!

But before clicking send please:

Ask your colleagues

Check our documentation and FAQ

Going Further

Courses from

DCSR

SIB

PRACE