Modelling, Uncertainty and Data for Engineers (MUDE)

Running code on cluster and queues

Denis Voskov, Ilshat Saifullin

Cluster

Cluster is a set of computational nodes, which are interconnected to a network.

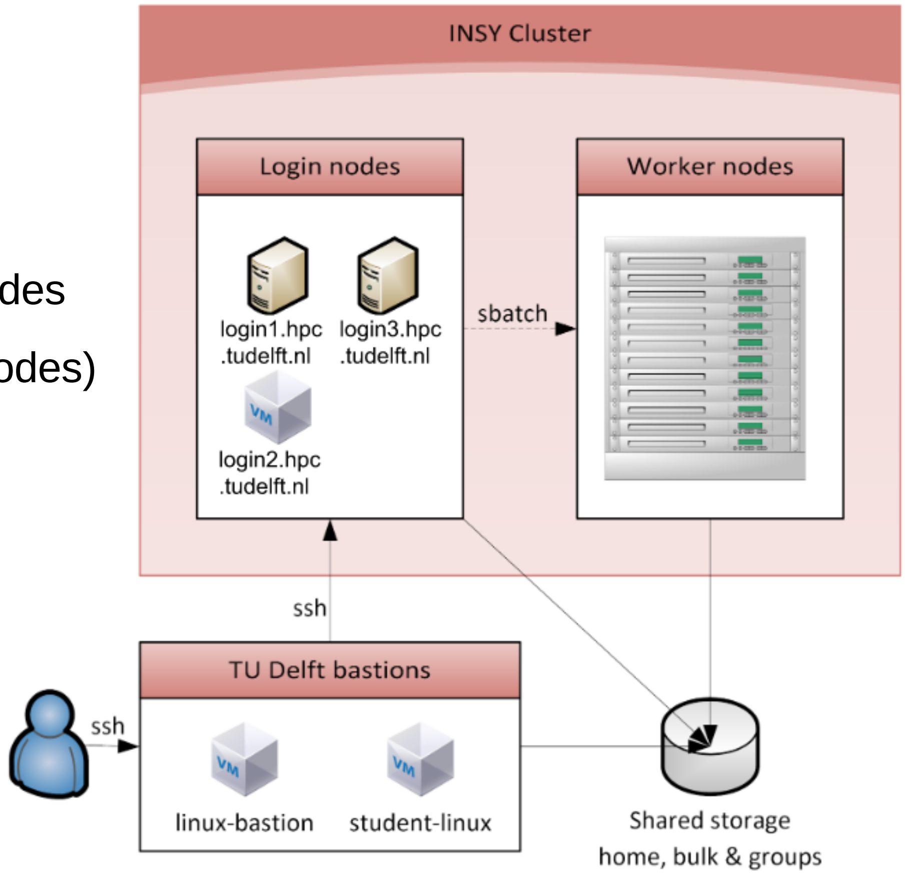
Each node is a computer with CPU, RAM, disk

HPC clusters are actual in case of:

- Computation takes a lot of time
- Series of calculations which can be run simultaneously on the different nodes
- Calculation requires a large amount of memory (one task runs on a few nodes)

Specifics:

- Usually command line interface is used through the login nodes
- Job manager to use the computational nodes
- Shared file system
- Monitoring system (web-interface)



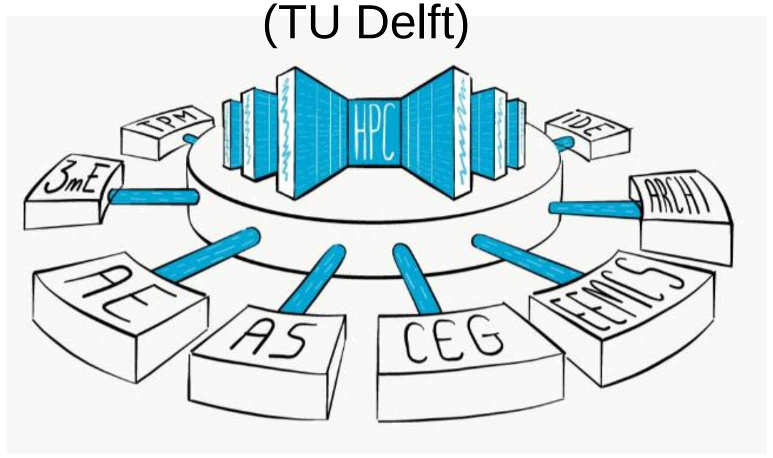
Cluster

jupyterhub

SURFsara national supercomputer

MobaXterm

DelftBlue supercomputer (TU Delft)

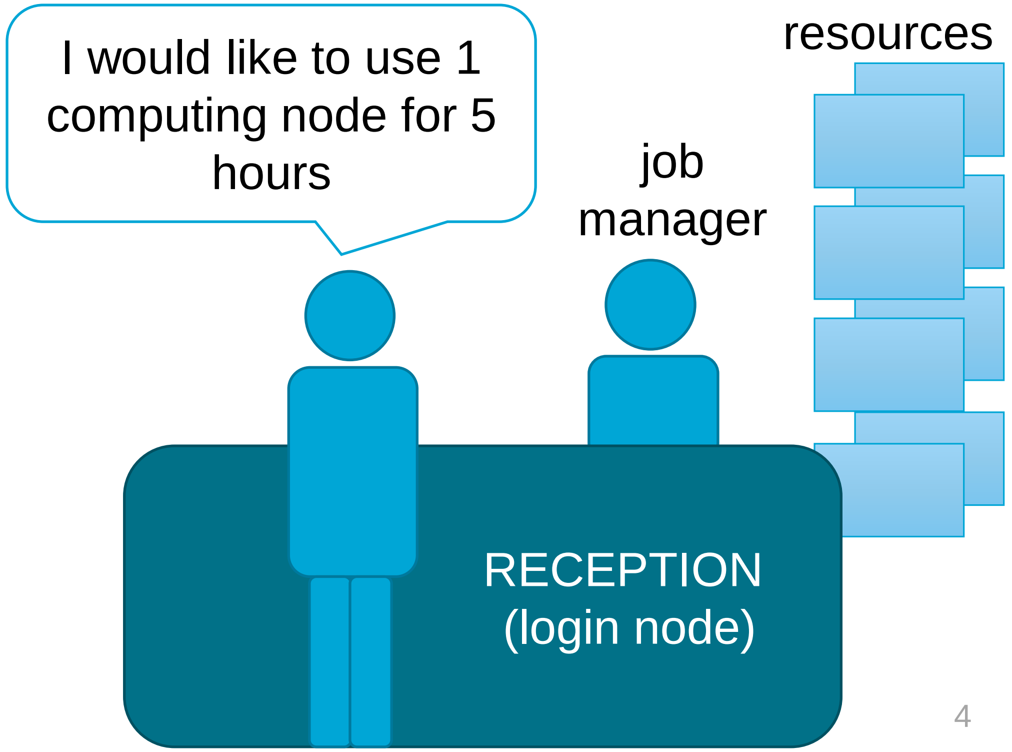


https://www.tudelft.nl/dhpc/ark/delftbluephase1

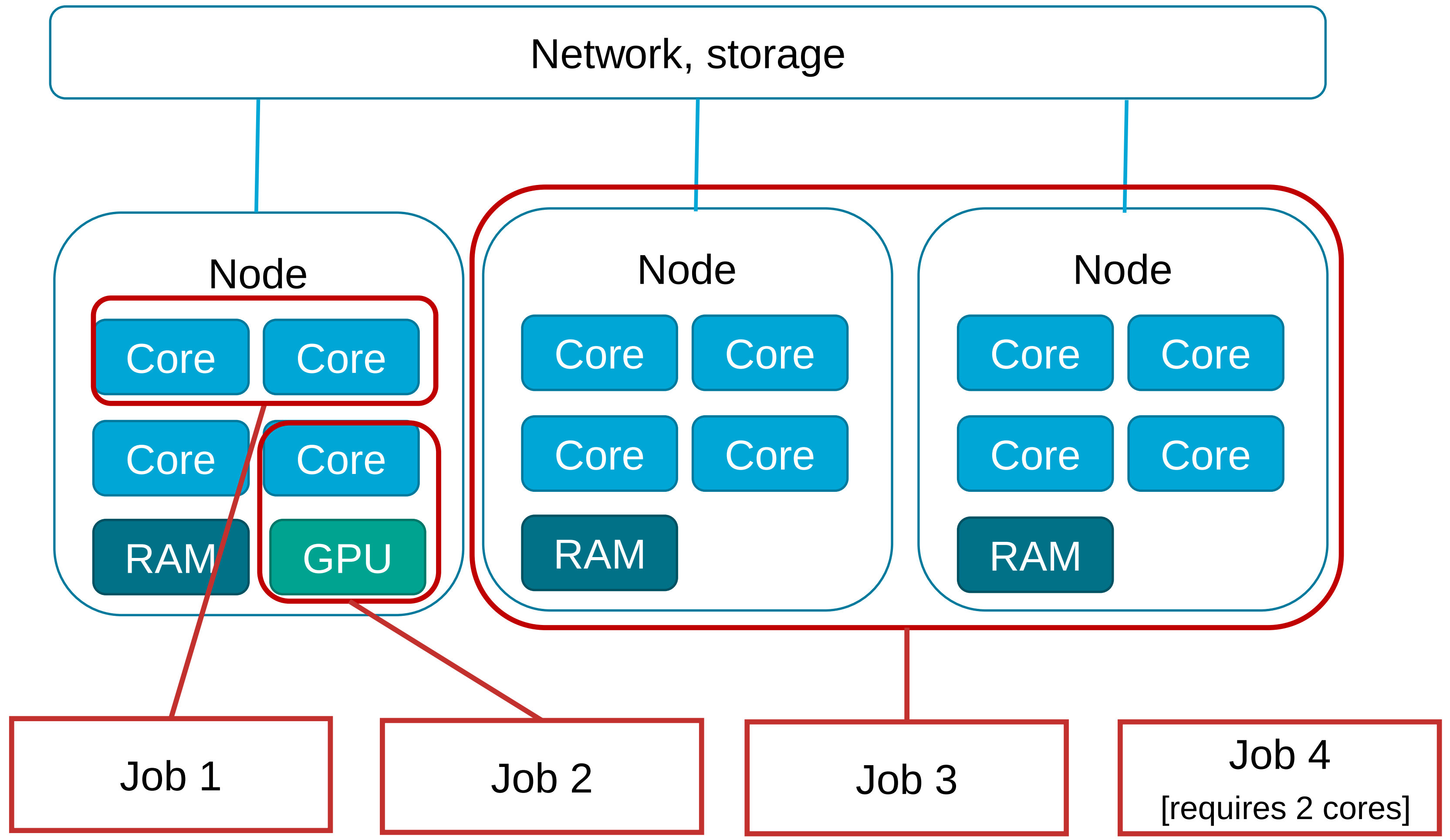
TU Delft

Job manager

- The scheduler handles how compute resources are shared between users and tasks
- Users should not run any task on login nodes
- Users should run tasks on a computing node only through the scheduler
- Job will not run immediately, it depends on the recourses available at that moment
- A task (job) usually has shell script syntax with:
  - commands need to be executed
  - resources request



Job managers



uses 2 CPU cores

uses 1 CPU cores and GPU

uses 2 nodes

waits in the queue (not enough free resources)

SLURM job manager: commands

Command	Description	Example
sbatch *RESOURCES SCRIPT*	submit a batch script	**sbatch myfile.sbatch**
srun *RESOURCES COMMAND*	run a single command	**srun --mem=1G --time=00:10:00 hostname**
squeue [-u *USERNAME*] [-t STATUS]	check the status of jobs	**squeue -u student** **squeue -u student -t COMPLETED**
scancel *JOBID*	cancel a job and delete it from the queue	**scancel 15168**

srun and sbatch differences:

- srun is blocking
- srun is interactive (outputs directly to the current console)
- if ssh session interrupts, a job running with srun be cancelled

TUDelft

https://doc.dhpc.tudelft.nl/delftblue/Slurm-scheduler/

Slurm: running a serial job (one core)

1. nano test.sbatch

Create a file with the next content

Job specifications

```
#!/bin/sh
#SBATCH --job-name="my script"
#SBATCH --time=01:00:00
#SBATCH --ntasks=1
#SBATCH --cpus-per-task=1
#SBATCH --mem-per-cpu=1G


srun hostname
#srun python my_script.py
```

Job name

Set the job time limit to 1 hour

Number of MPI processes (nodes)

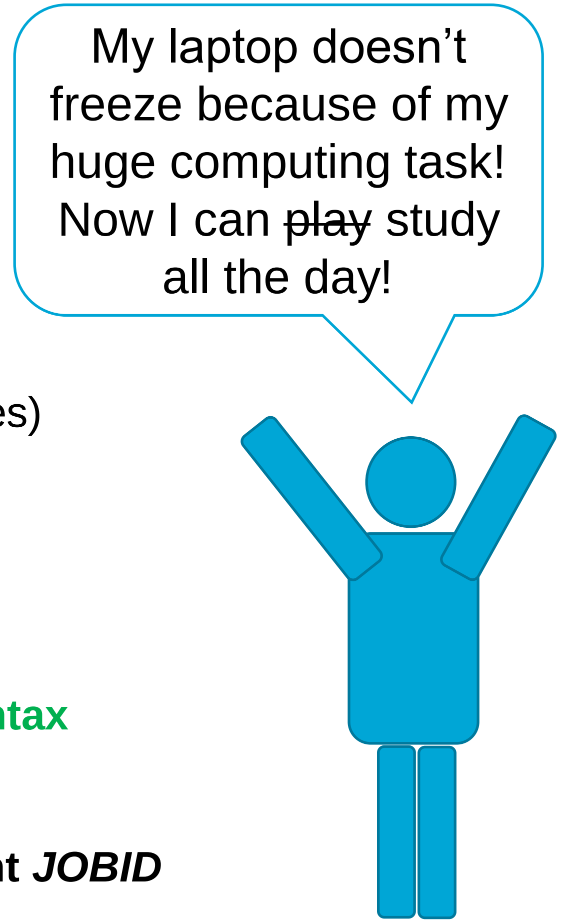
Allocate 1 core of processor

Set the memory limit to 1 GB.

Commands to run in Bash syntax (your computational task)

2. sbatch test.sbatch

Put the job into queue and print *JOBID*


My laptop doesn't freeze because of my huge computing task! Now I can ~~play~~ study all the day!

3. Check the status of your job with squeue -u username
4. Output will be saved to the file slurm-*JOBID*.out

Additional batch parameters for DelftBlue

1. It is needed to load modules first

```
module load 2022r2
module load python
module load py-numpy
module load py-matplotlib
```

2. Add this line to your sbatch script

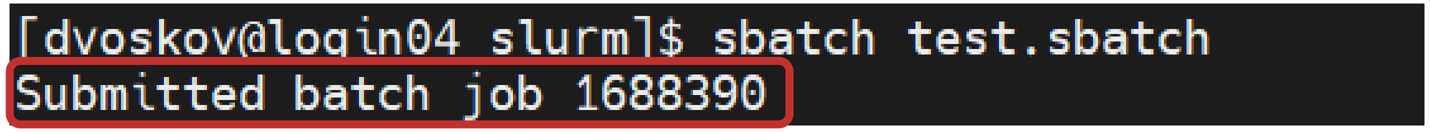
```
#SBATCH --account=Education-CEG-Courses-CEGM1000
```

TU Delft

https://doc.dhpc.tudelft.nl/delftblue/DHPC-modules/

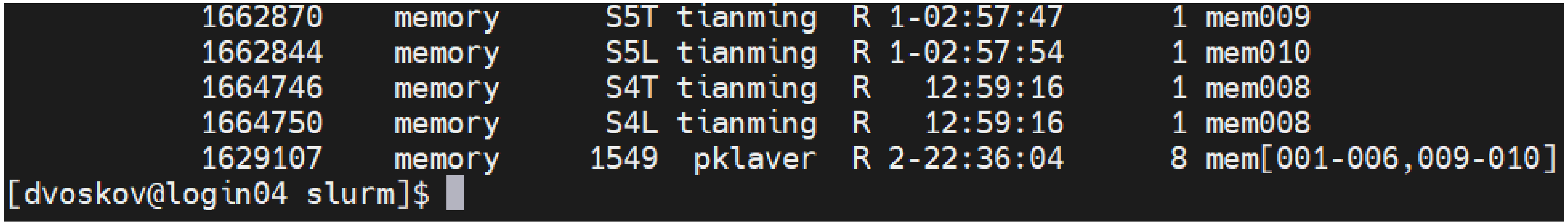
Examples of commands and output

Output of sbatch



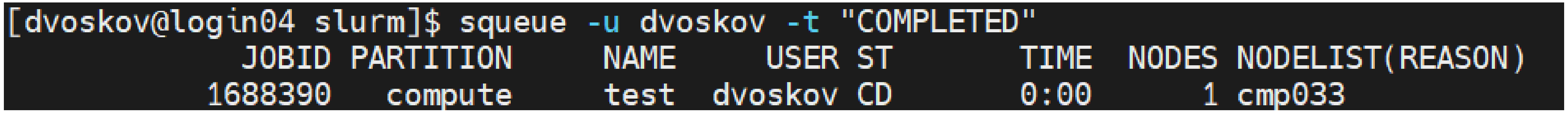
```
[dvoskov@login04 slurm]$ sbatch test.sbatch
Submitted batch job 1688390
```

Output of squeue



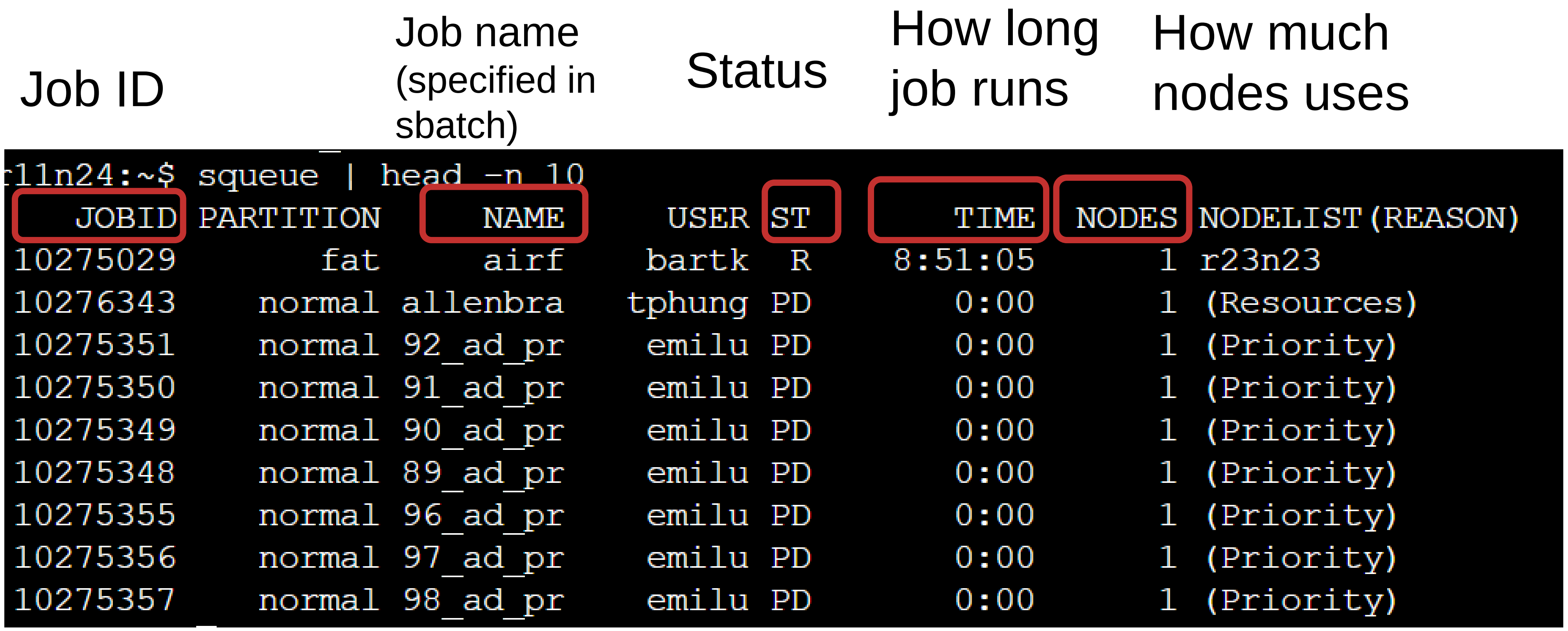
```
           1662870    memory     S5T tianming  R 1-02:57:47      1 mem009
           1662844    memory     S5L tianming  R 1-02:57:54      1 mem010
           1664746    memory     S4T tianming  R   12:59:16      1 mem008
           1664750    memory     S4L tianming  R   12:59:16      1 mem008
           1629107    memory    1549  pklaver  R 2-22:36:04      8 mem[001-006,009-010]
[dvoskov@login04 slurm]$ 
```

squeue doesn’t show completed tasks by default. How to show them:



```
[dvoskov@login04 slurm]$ squeue -u dvoskov -t "COMPLETED"
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
           1688390   compute     test  dvoskov CD       0:00      1 cmp033
```

Squeue command output



```
r11n24:~$ squeue | head -n 10
   JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
10275029       fat     airf    bartk  R    8:51:05      1 r23n23
10276343    normal allenbra   tphung PD       0:00      1 (Resources)
10275351    normal 92_ad_pr    emilu PD       0:00      1 (Priority)
10275350    normal 91_ad_pr    emilu PD       0:00      1 (Priority)
10275349    normal 90_ad_pr    emilu PD       0:00      1 (Priority)
10275348    normal 89_ad_pr    emilu PD       0:00      1 (Priority)
10275355    normal 96_ad_pr    emilu PD       0:00      1 (Priority)
10275356    normal 97_ad_pr    emilu PD       0:00      1 (Priority)
10275357    normal 98_ad_pr    emilu PD       0:00      1 (Priority)
```

Completed tasks are not shown by default

Most important job statuses

Code	Status
PD	PENDING
R	RUNNING
CG	COMPLETING
CD	COMPLETED
F	FAILED
CA	CANCELLED

TU Delft

How to connect to DelftBlue

ssh command

1. From Linux
2. From MobaXTerm application

https://doc.dhpc.tudelft.nl/delftblue/Remote-access-to-DelftBlue/

How to transfer your data to DelftBlue

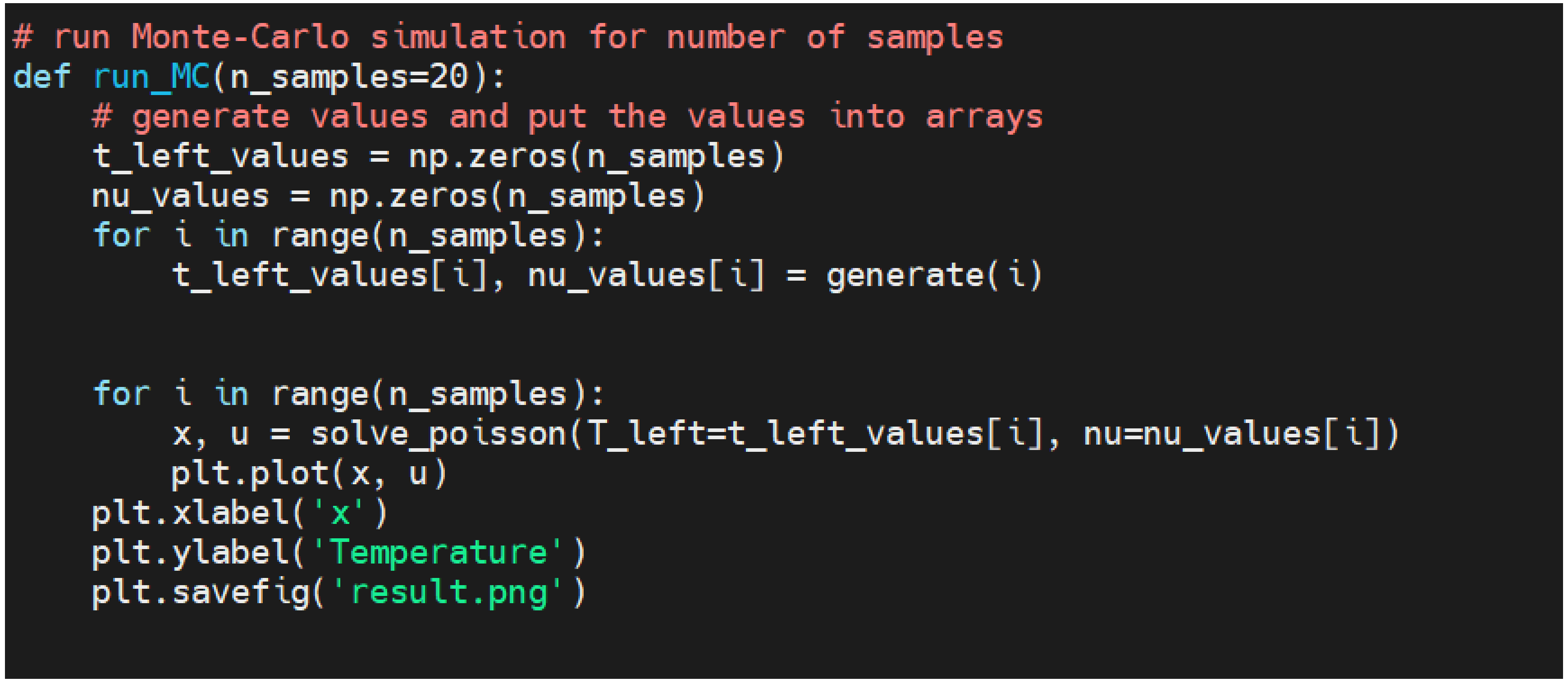
1. Drag and drop from/to file tree in MobaXterm (left area)
2. OnDemand web-interface
3. SFTP clients (GUI)
4. scp command
5. rsync command

Details:

https://doc.dhpc.tudelft.nl/delftblue/Data-transfer-to-DelftBlue/#scp

Use MC solver for Poisson equation

- Lets run Poisson solver in Monte-Carlo simulation using left temperature and nu as an uncertain input based on seed:

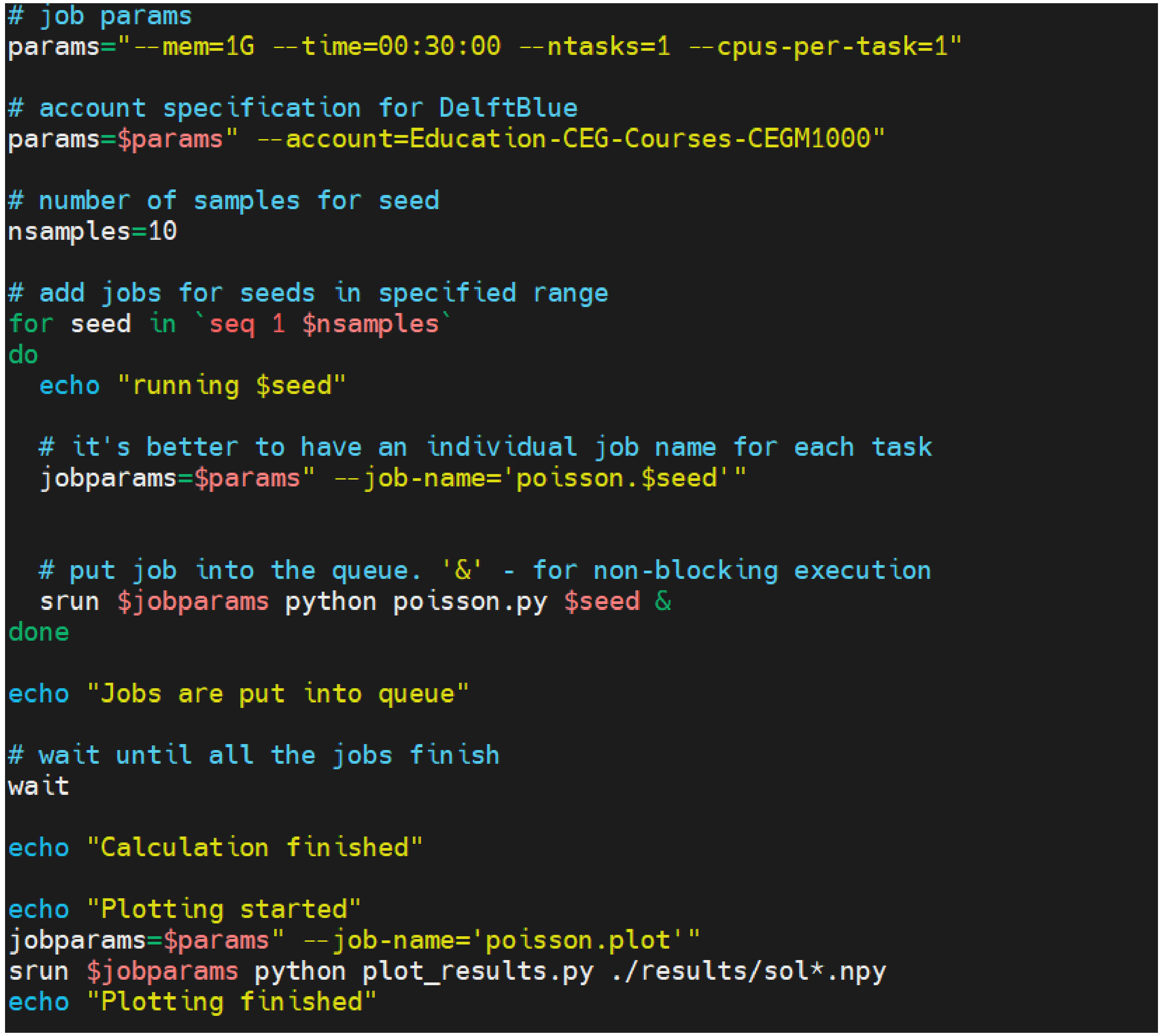


```python
# run Monte-Carlo simulation for number of samples
def run_MC(n_samples=20):
    # generate values and put the values into arrays
    t_left_values = np.zeros(n_samples)
    nu_values = np.zeros(n_samples)
    for i in range(n_samples):
        t_left_values[i], nu_values[i] = generate(i)


    for i in range(n_samples):
        x, u = solve_poisson(T_left=t_left_values[i], nu=nu_values[i])
        plt.plot(x, u)
    plt.xlabel('x')
    plt.ylabel('Temperature')
    plt.savefig('result.png')
```

Slurm: Solving Poisson equation on cluster

How to solve with different parameters simultaneously (on different cores/nodes)?

1. Modify the Python script poisson.py to use cmd parameters
2. Pass the seed parameter from bash script
3. Make a loop in bash - to put the jobs into queue
4. Wait until computations finish
5. Plot the results

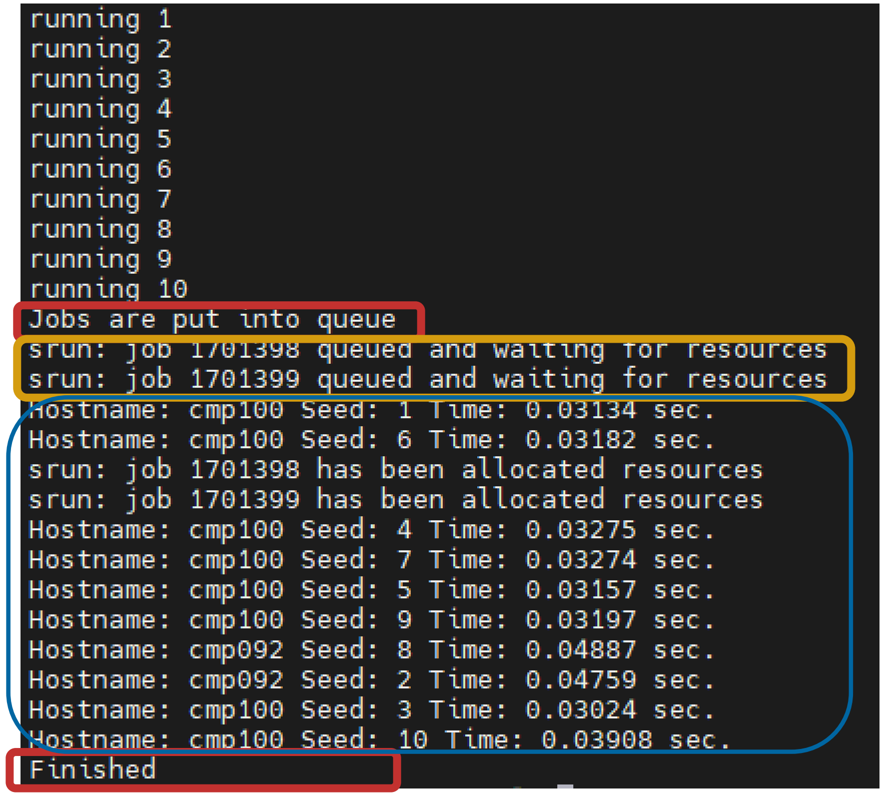


```
# job params
params="--mem=1G --time=00:30:00 --ntasks=1 --cpus-per-task=1"

# account specification for DelftBlue
params=$params" --account=Education-CEG-Courses-CEGM1000"

# number of samples for seed
nsamples=10

# add jobs for seeds in specified range
for seed in `seq 1 $nsamples`
do
  echo "running $seed"

  # it's better to have an individual job name for each task
  jobparams=$params" --job-name='poisson.$seed'"


  # put job into the queue. '&' - for non-blocking execution
  srun $jobparams python poisson.py $seed &
done

echo "Jobs are put into queue"

# wait until all the jobs finish
wait

echo "Calculation finished"

echo "Plotting started"
jobparams=$params" --job-name='poisson.plot'"
srun $jobparams python plot_results.py ./results/sol*.npy
echo "Plotting finished"
```

TUDelft

Running Poisson solver with slurm

Output of ./**run.sh**

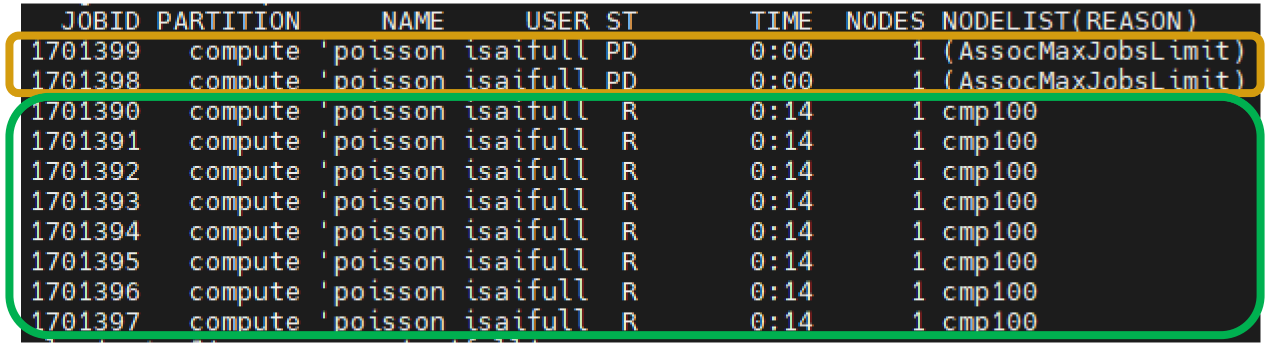


```
running 1
running 2
running 3
running 4
running 5
running 6
running 7
running 8
running 9
running 10
Jobs are put into queue
srun: job 1701398 queued and waiting for resources
srun: job 1701399 queued and waiting for resources
Hostname: cmp100 Seed: 1 Time: 0.03134 sec.
Hostname: cmp100 Seed: 6 Time: 0.03182 sec.
srun: job 1701398 has been allocated resources
srun: job 1701399 has been allocated resources
Hostname: cmp100 Seed: 4 Time: 0.03275 sec.
Hostname: cmp100 Seed: 7 Time: 0.03274 sec.
Hostname: cmp100 Seed: 5 Time: 0.03157 sec.
Hostname: cmp100 Seed: 9 Time: 0.03197 sec.
Hostname: cmp092 Seed: 8 Time: 0.04887 sec.
Hostname: cmp092 Seed: 2 Time: 0.04759 sec.
Hostname: cmp100 Seed: 3 Time: 0.03024 sec.
Hostname: cmp100 Seed: 10 Time: 0.03908 sec.
Finished
```

Pending jobs

Output from jobs

Output of squeue



```
   JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
 1701399   compute 'poisson isaifull PD       0:00      1 (AssocMaxJobsLimit)
 1701398   compute 'poisson isaifull PD       0:00      1 (AssocMaxJobsLimit)
 1701390   compute 'poisson isaifull  R       0:14      1 cmp100
 1701391   compute 'poisson isaifull  R       0:14      1 cmp100
 1701392   compute 'poisson isaifull  R       0:14      1 cmp100
 1701393   compute 'poisson isaifull  R       0:14      1 cmp100
 1701394   compute 'poisson isaifull  R       0:14      1 cmp100
 1701395   compute 'poisson isaifull  R       0:14      1 cmp100
 1701396   compute 'poisson isaifull  R       0:14      1 cmp100
 1701397   compute 'poisson isaifull  R       0:14      1 cmp100
```

Running jobs