High Performance Computing for the VPH

A practical introduction to HPC usage

Marco Verdicchio
SURFsara

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 675451.

Introduction to HPC in Computational Modelling

HANDS ON

Getting the training material

The material for this course is available at:
http://bit.ly/HPC_CBM

Transfer the zip file to MarenostrumIV

```
~> scp IntroHPC_CBM-master.zip nct00004@mn1.bsc.es:
```

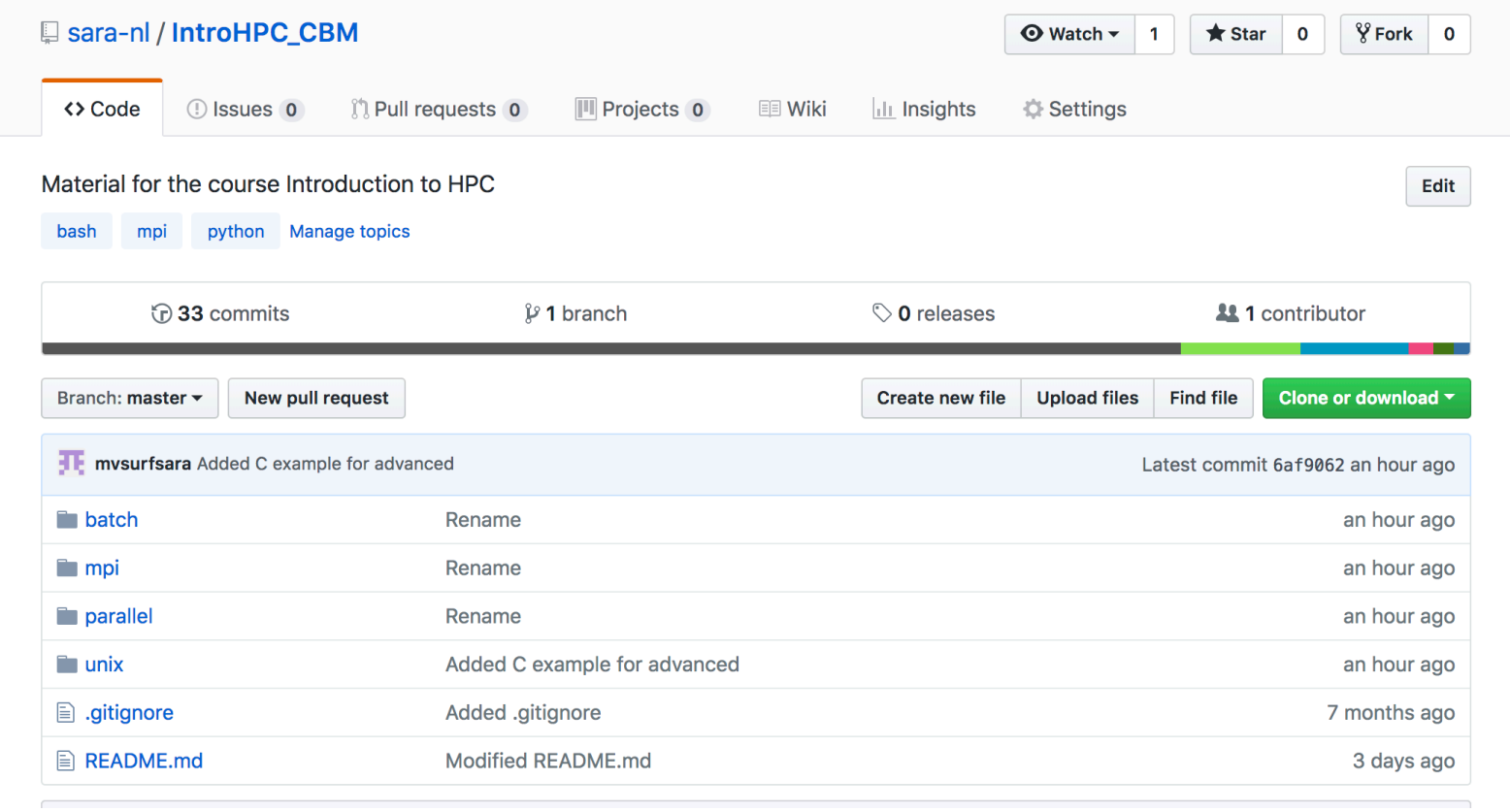
(Windows users can use MobaXterm or Winscp)

Extract the zip in your home:

```
nct00004@login1:~> unzip IntroHPC_CBM-master.zip
```

Getting the training material

Hands on material **http://bit.ly/HPC_CBM**



Bash scripting

Simple bash script

unix/simple.sh

```
#!/bin/bash

echo "Hi, I'm your first script."
echo


lscpu --help > cpu.log
lscpu >> cpu.log


echo "I've left something for you."
echo "Ciao"
```

Use "`nano simple.sh`" to visualize the file. Ctrl-X to exit .

Batch system

Run “simple.sh” interactively

Give execution permissions to the script:

```
nct00004@login1:~> chmod u+x simple.sh
```

Run the script:

```
nct00004@login1:~> ./simple.sh
```

```
nct00004@login1:~> bash simple.sh
```

Bash scripting

Advanced bash script

unix/advanced.sh

```
#!/bin/bash
# <- this is a comment and everything that follow is ignored

args=$# # Number of args passed.

if [ $args = 0 ]
then
 echo "This script needs a number as argument input."
 echo "Ex: ./advanced.sh 10"
 exit
fi
```

Bash scripting

unix/advanced.sh (cont'd)

```
num=$1 # First argument passed.

echo "The input number is:" $1
echo

cd src

gcc -o ../fact.exe fact.c 1> compile.log 2> compile.err
# Redirect stdout and stderr to files

cd ..

./fact.exe $num | tee fact.log # Pipe output into command

echo
echo "Done"
```

Batch system

Run “advanced.sh” interactively

Give execution permissions to the script:

```
nct00004@login1:~> chmod u+x advanced.sh
```

Run the script:

```
nct00004@login1:~> ./advanced.sh 10
```

```
nct00004@login1:~> ls –l *.log
```

Batch jobs

batch/slurm1.sub

```
#!/bin/bash
#SBATCH --job-name="test_serial"
#SBATCH --ntasks=1
#SBATCH --time=00:02:00
#SBATCH --workdir=.
#SBATCH --output=serial_%j.out
#SBATCH --error=serial_%j.err

echo "Who am I?"
whoami
echo "Where ?"
srun hostname

sleep 600
echo "Ciao"
```

batch/slurm2.sub

```
#!/bin/bash
#SBATCH --job-name="test_multinode"
#SBATCH --nodes=2
#SBATCH --tasks-per-node=3
#SBATCH --time=00:02:00
#SBATCH --workdir=.
#SBATCH --output=multinode_%j.out
#SBATCH --error=multinode_%j.err

echo "Who am I?"
whoami
echo "Where ?"
srun hostname

sleep 600
echo "Ciao"
```

Batch system

Submitting jobs with SLURM

Submits a “job script” to the queue system:

```
nct00004@login1:~> sbatch slurm1.sub
```

Check the status of the submitted jobs:

```
nct00004@login1:~> squeue

JOBID   PARTITION   NAME  USER    ST  TIME  NODES  NODELIST(REASON)
2262040      main test_s nct00001 PD  0:00      2  (None)
```

MPI jobs

Running an MPI application

mpi/run_wave (I)

```
...

SCRATCH=/gpfs/scratch/nct00/$USER/$SLURM_JOBID

mkdir –p $SCRATCH || exit 1

cp wave $SCRATCH
...

cd $SCRATCH
```

MPI jobs

Running an MPI application

mpi/run_wave (II)

```
...
# if wave was compiled with OpenMPI support, set the number of
threads
# export OMP_NUM_THREADS=4

srun ./wave $m $n $dx $dy $nt $dt $c $zeta \
           $f0x0 $f0y0 $f0r0 $nt_i $outfile

perl h52anim.pl wave.h5 u

cp wave.h5 wave.h5.gif $SLURM_SUBMIT_DIR
```

Batch system

Prepare and runs mpi example

Read the README.md for more information:

```
nct00004@login1:~> nano README.md
```

Requires compilers, MPI, HDF5, perl:

```
nct00004@login1:~> module load hdf5
nct00004@login1:~> module load mpi
nct00004@login1:~> module load perl
nct00004@login1:~>
```

Batch system

Prepare and run mpi example

The code needs to be compiled and linked to right libraries:

```
nct00004@login1:~> nano Makefile
nct00004@login1:~> make
```

Submit to the batch:

```
nct00004@login1:~> sbatch run_wave
```

Check that it runs:

```
nct00004@login1:~> squeue
```

Batch system

Prepare and run mpi example

It generates a gif that we can copy back at the end:

```
~> scp nct00004@mn1.bsc.es:IntroHPC_CBM-master/mpi/wave.h5.gif .
```

Or can be visualized remotely:

```
~> ssh –X nct00004@mn1.bsc.es
```

```
nct00004@login1:~> display IntroHPC_CBM-master/mpi/wave.h5.gif
```