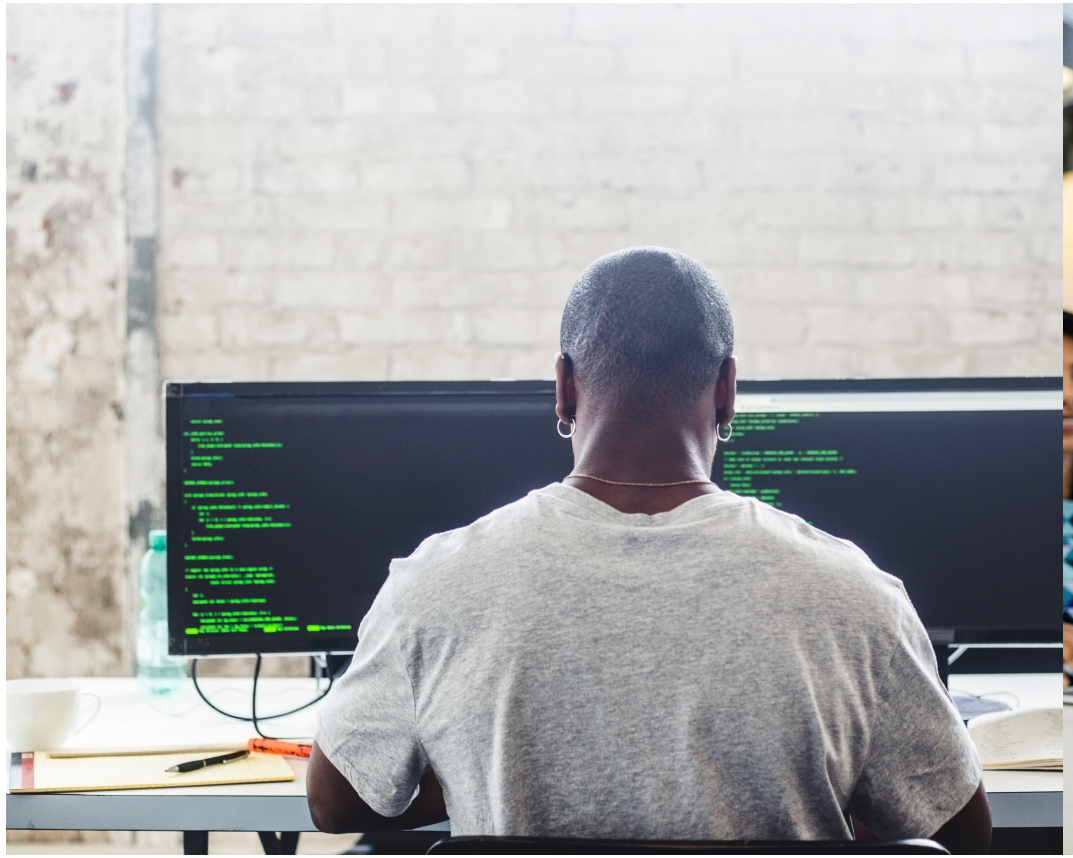
High Performance Computing Center North (HPC2N)

Pedro Ojeda-May

UMEÅ UNIVERSITY

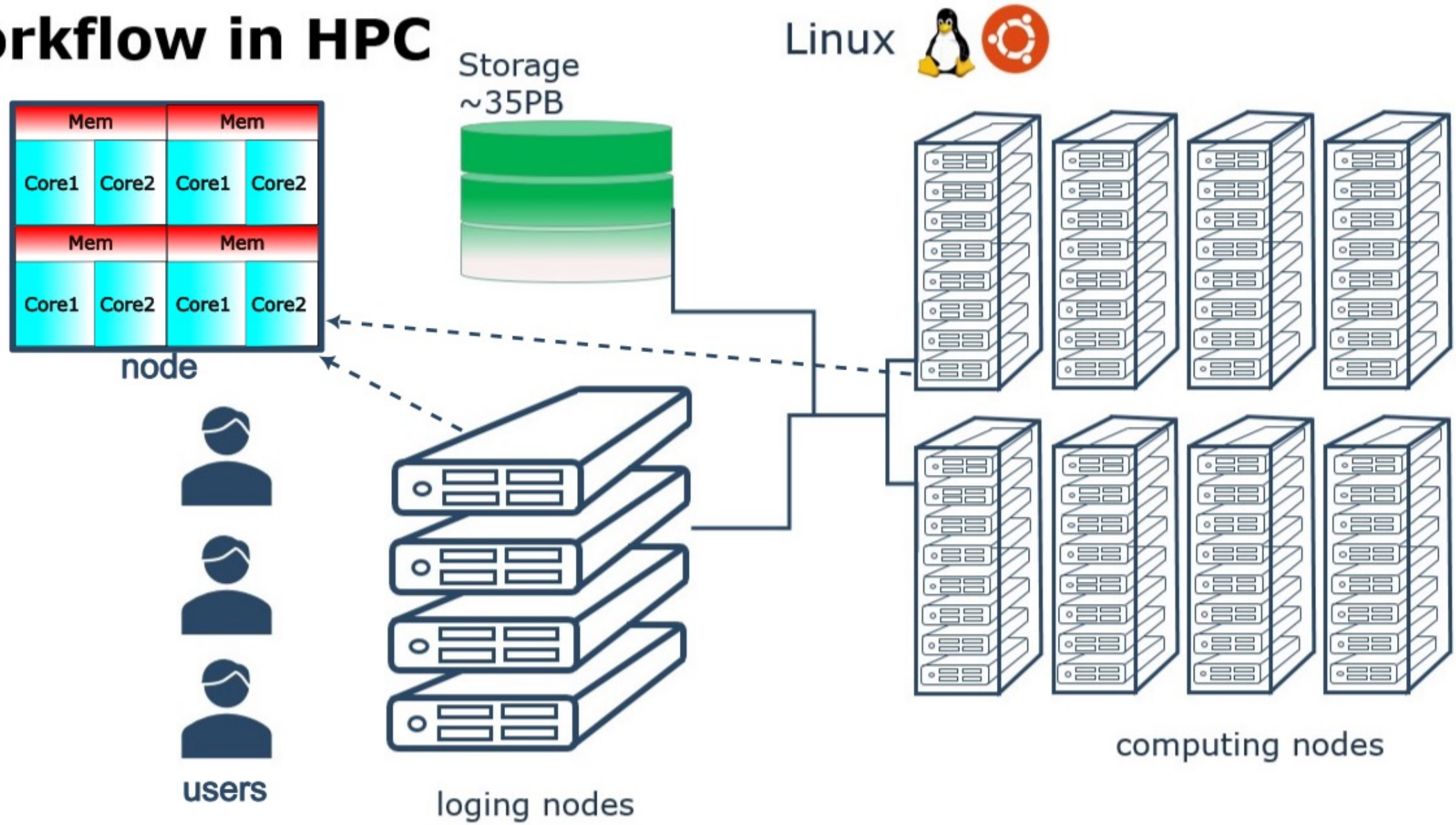
Workflow on a PC



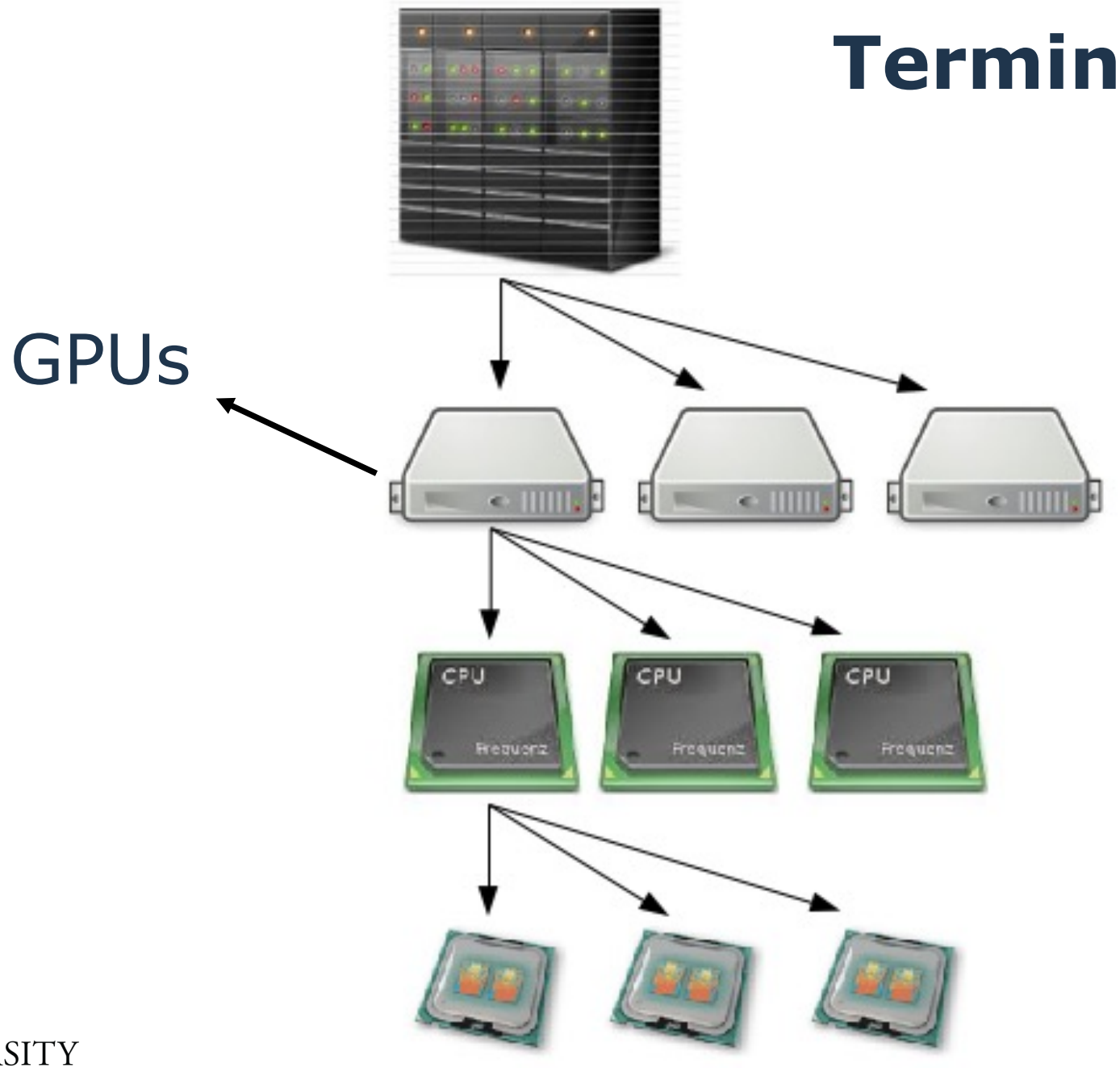
UMEÅ UNIVERSITY

Linux, Windows, Mac, Android

Workflow in HPC



Terminology



UMEÅ UNIVERSITY

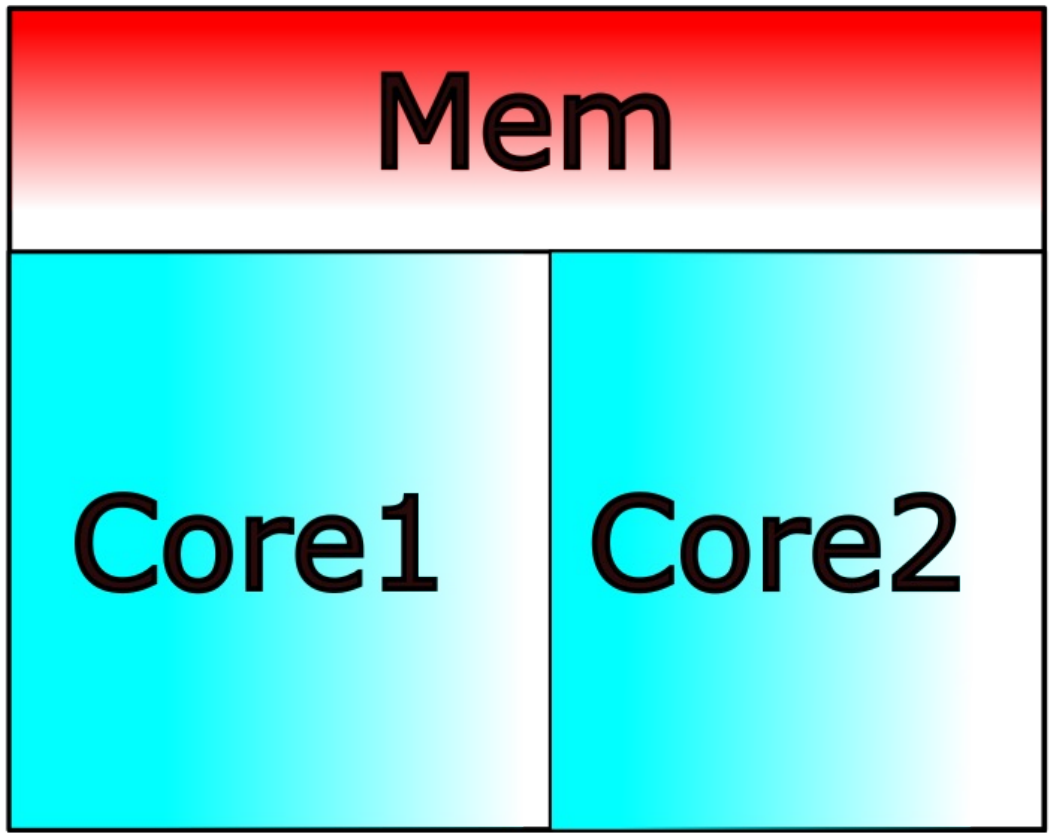
"R in HPC" course HPC2N/PDC (https://github.com/menzzana/parallel_R_course)

Parallelization techniques

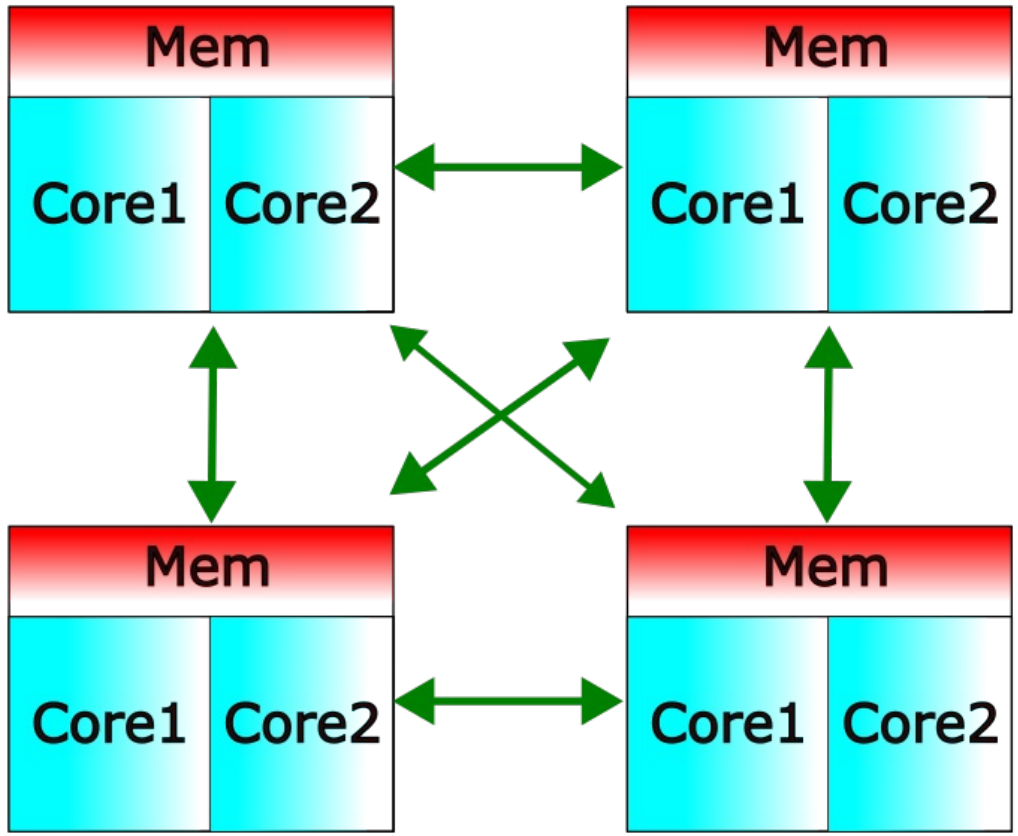
Kebnekaise

Architectures

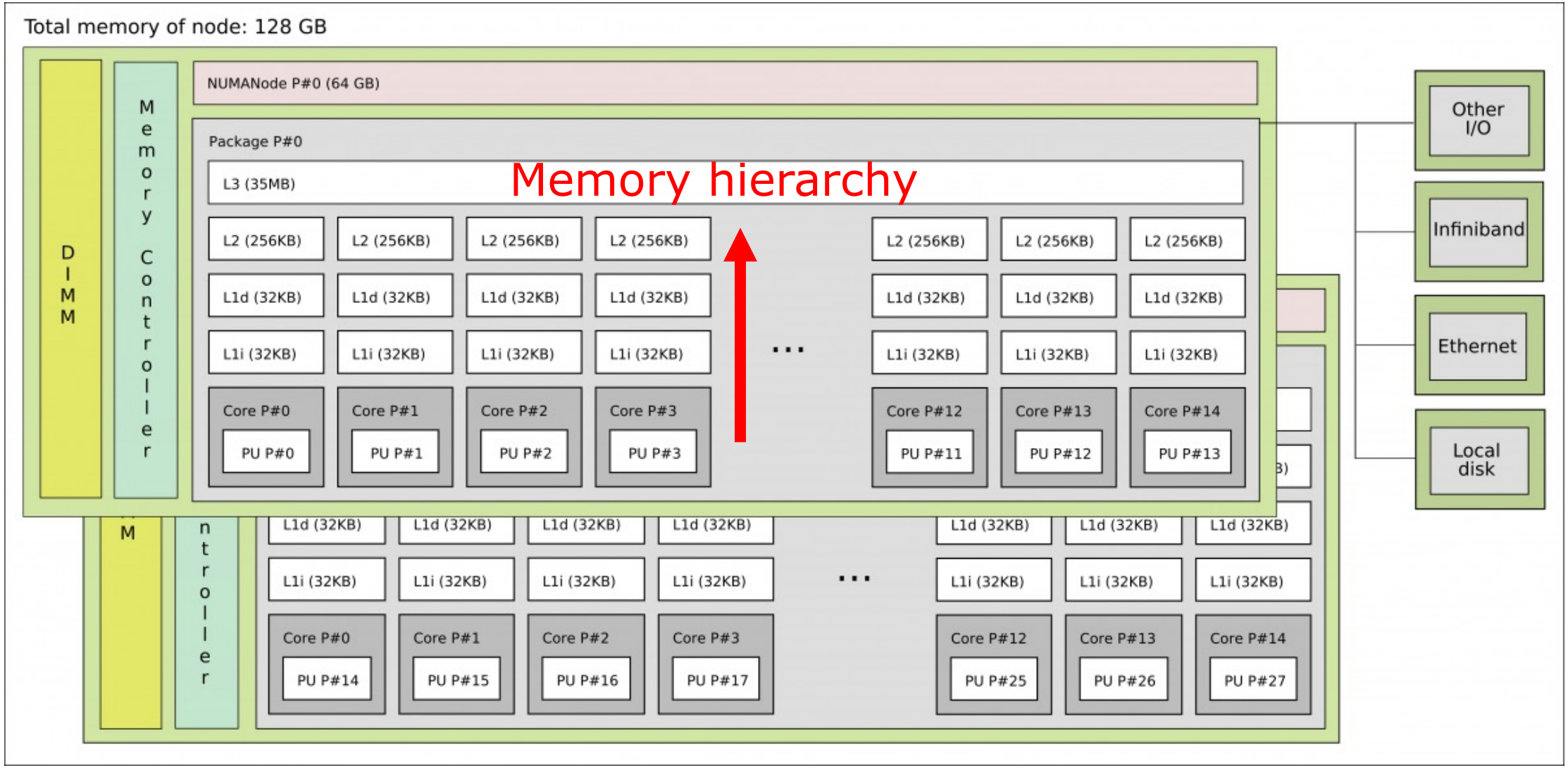
Shared memory



Distributed memory



Node on Kebnekaise



GPU nodes on Kebnekaise

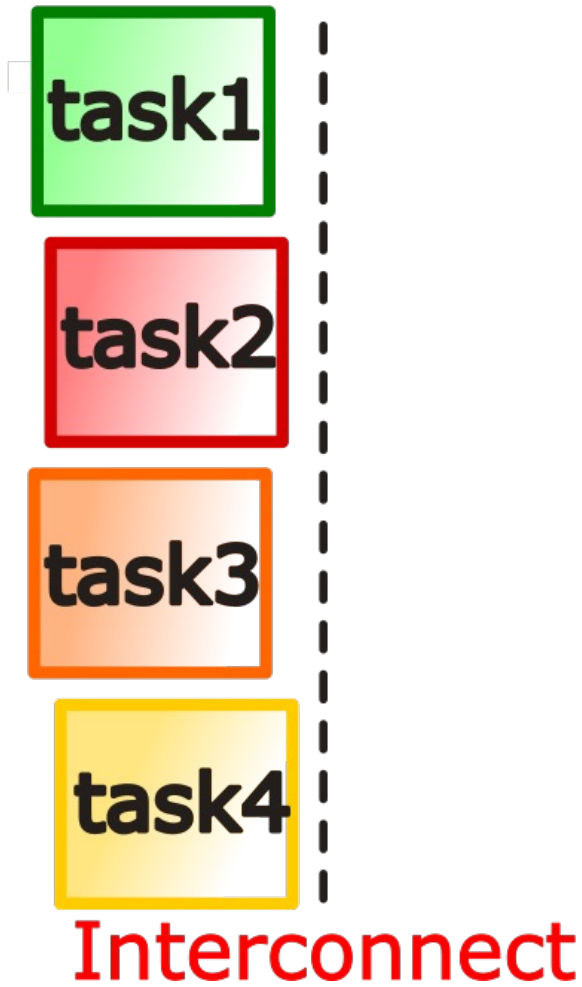
UMEÅ UNIVERSITY

GPU nodes on Kebnekaise

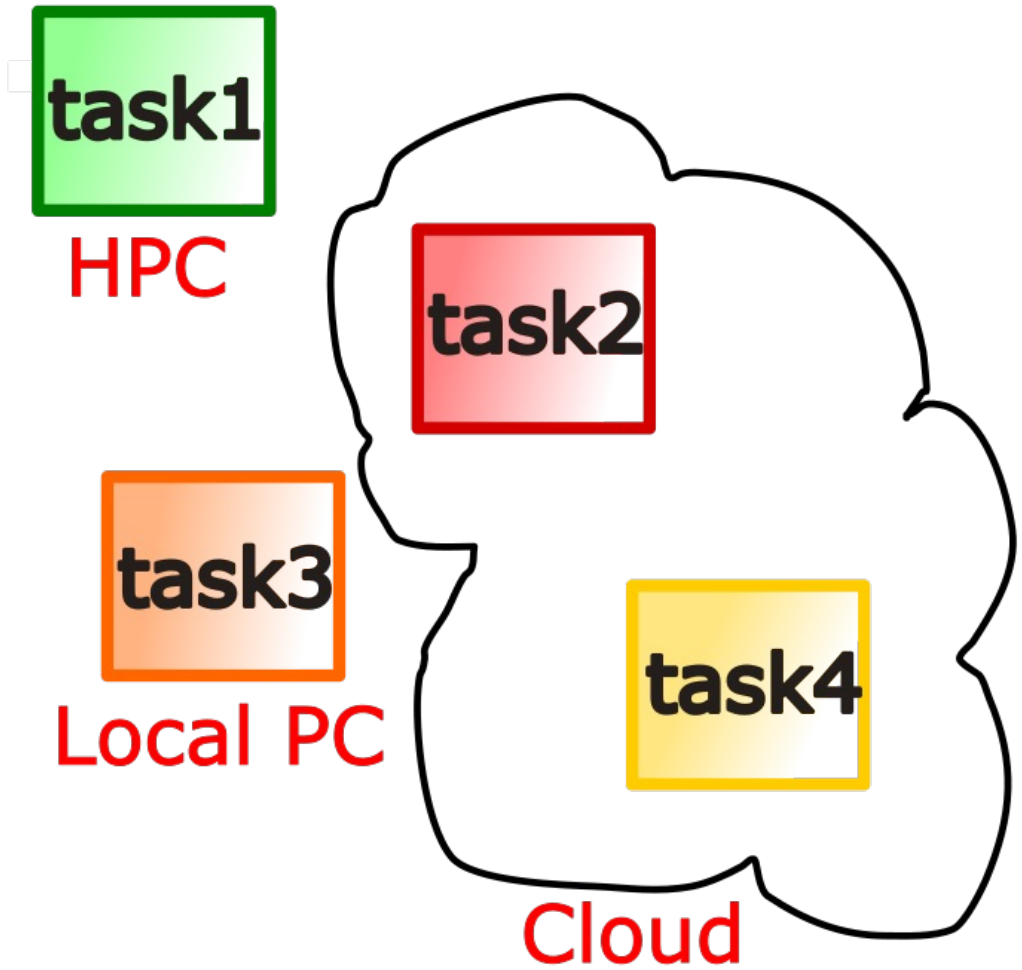
- V100 (NVIDIA)
- A100 (NVIDIA)

Login: kebnekaise-amd,
Thinlinc: kebnekaise-amd-tl

High Performance Computing (HPC)



High Throughput Computing (HTC)

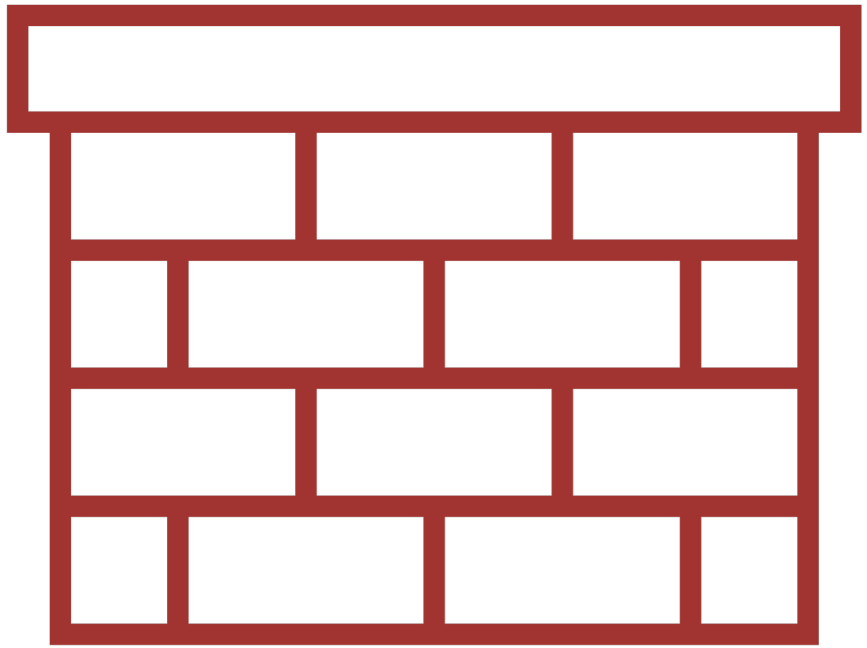


Metrics for performance analysis

Speedup=(1 day)/(0.5 day) = 2

1 day

0.5 day



UMEÅ UNIVERSITY

Metrics for performance analysis

Nr. cores (n)	Time (min)	Speedup (S)	dS/dn
1	19.5	1	-
2	10.8	1.80	0.80
4	5.5	3.54	0.86
8	2.8	6.96	0.85
16	1.5	13	0.75
32	0.9	21.66	0.54
48	0.9	21.66	0

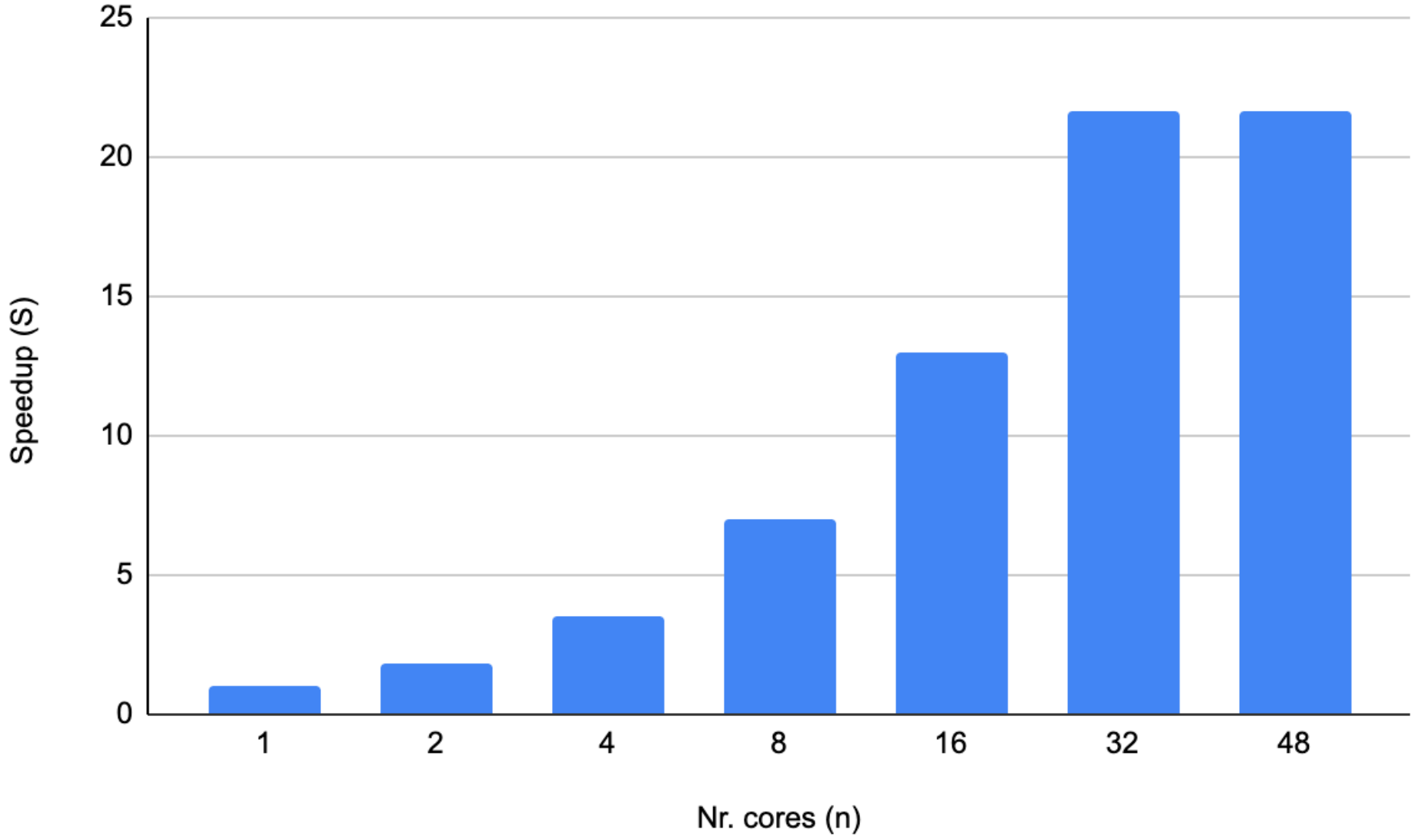
Timing results of a AMBER simulation.

S(x cores) = Time(1 core)/Time(x cores)

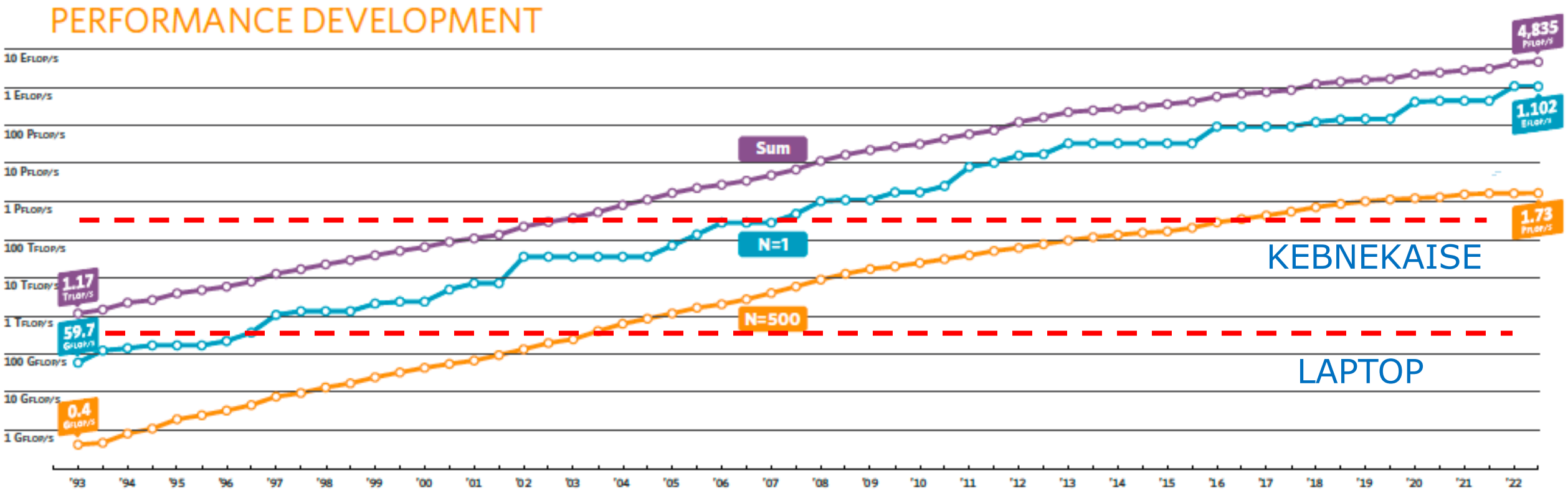
Example:
S(2 cores) = 19.5 min/10.8 = 1.8

Derivative example
$\frac{\partial S}{\partial n}|_{n=32}$=(21.66-13)/(32-16)=0.54

Results from MD simulations with AMBER



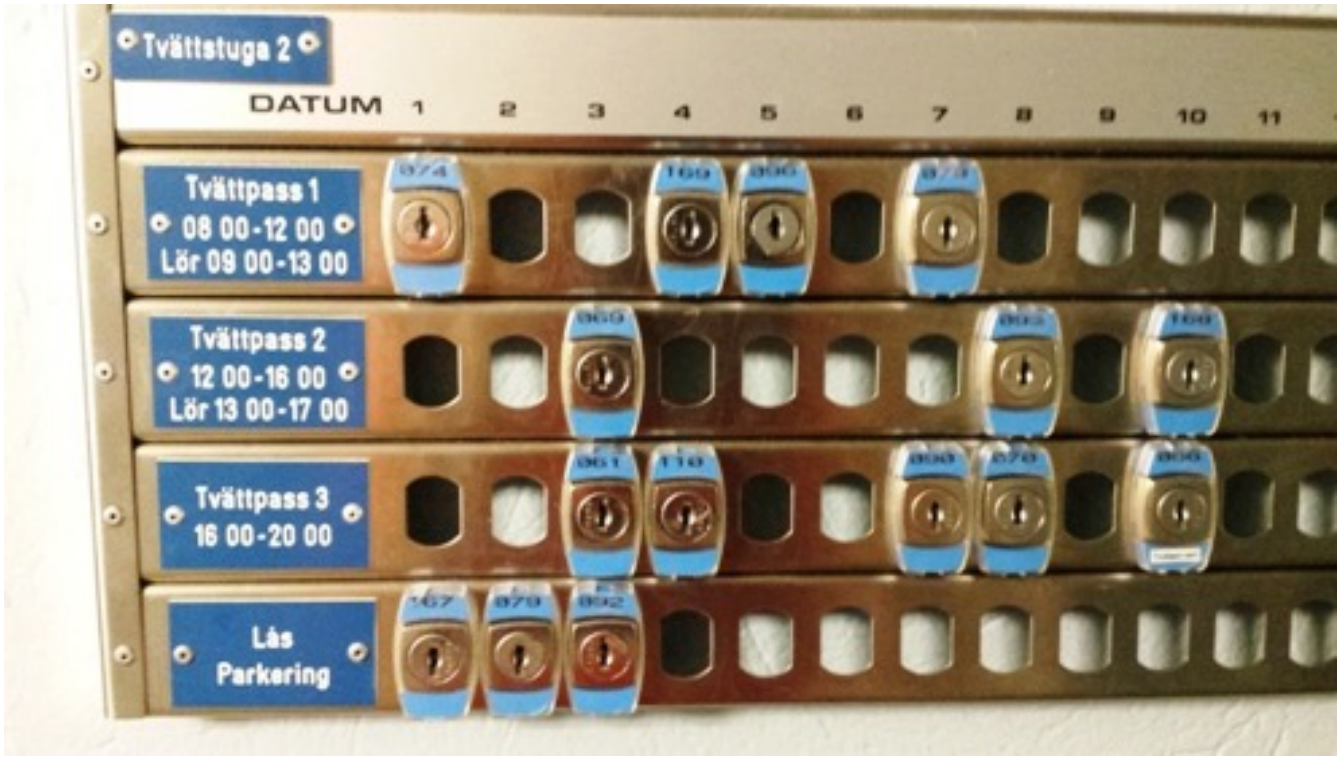
Top 500 list



UMEÅ UNIVERSITY

www.top500.org

Job scheduler (SLURM)



UMEÅ UNIVERSITY

Software management

We use **EasyBuild** and **Lmod** to install and manage software, respectively.

If you are interested in some package whose name contains some *string*, you can see if that package is already installed by typing:

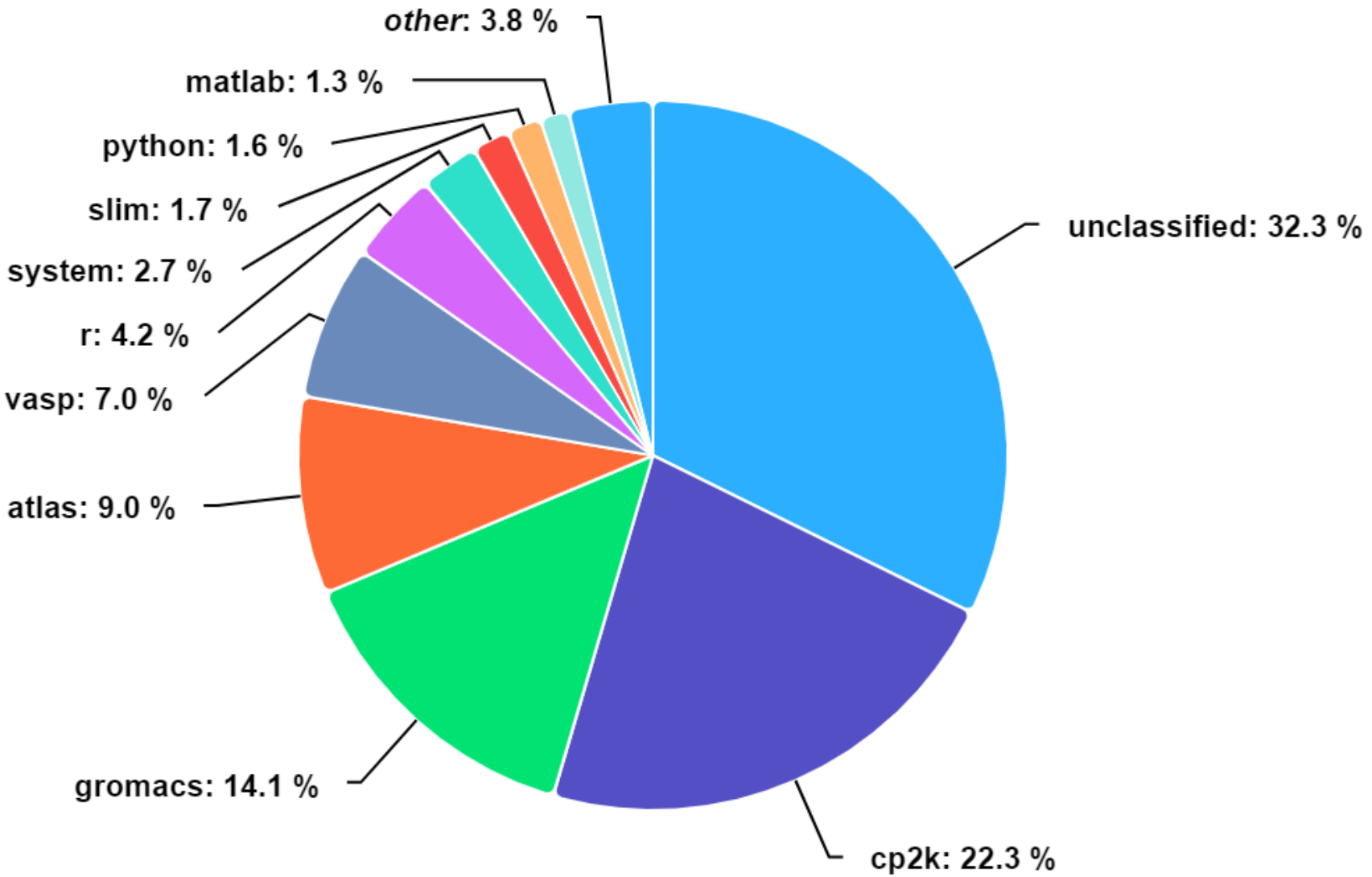
module spider *string*

Here, you will see all the modules that need to be loaded prior to using this package. You can load the modules with

module load *package*

Core hour usage per software for Kebnekaise @ HPC2N

Period: 2023-01-01 - 2023-09-01



FDTD_Solutions (Lumerical)	High performance FDTD-method Maxwell solver for the design, analysis and optimization of nanophotonic devices, processes and materials. External info:
galamost	GALAMOST is a versatile molecular simulation package which is designed to fully utilize computational power of graphics processing units (GPUs). It is developed specially for coarse-graining simulations of polymeric systems by including some latest techniques External info:
GAMESS-US	The General Atomic and Molecular Electronic Structure System (GAMESS) is a general ab initio quantum chemistry package. External info:
Gaussian	Gaussian provides state-of-the-art capabilities for electronic structure modeling. External info:
GPAW	GPAW is a density-functional theory (DFT) Python code based on the projector-augmented wave (PAW) method and the atomic simulation environment (ASE). External info:
Gromacs	A molecular dynamics package primarily designed for biomolecular systems such as proteins and lipids. It is a free software. External info:
HDF5	HDF5 is a data model, library, and file format for storing and managing data. External info:
hwloc	The Portable Hardware Locality (hwloc) software package provides a portable abstraction (across OS, versions, architectures, ...) of the hierarchical topology of modern architectures, including NUMA memory nodes, sockets, shared caches, cores and simultaneous multithreading. It also gathers various system attributes such as cache and memory information as well as the locality of I/O devices such as network

Projects and accounting

One can apply for a local project in SUPR (supr.naiss.se) after signing a User Agreement.

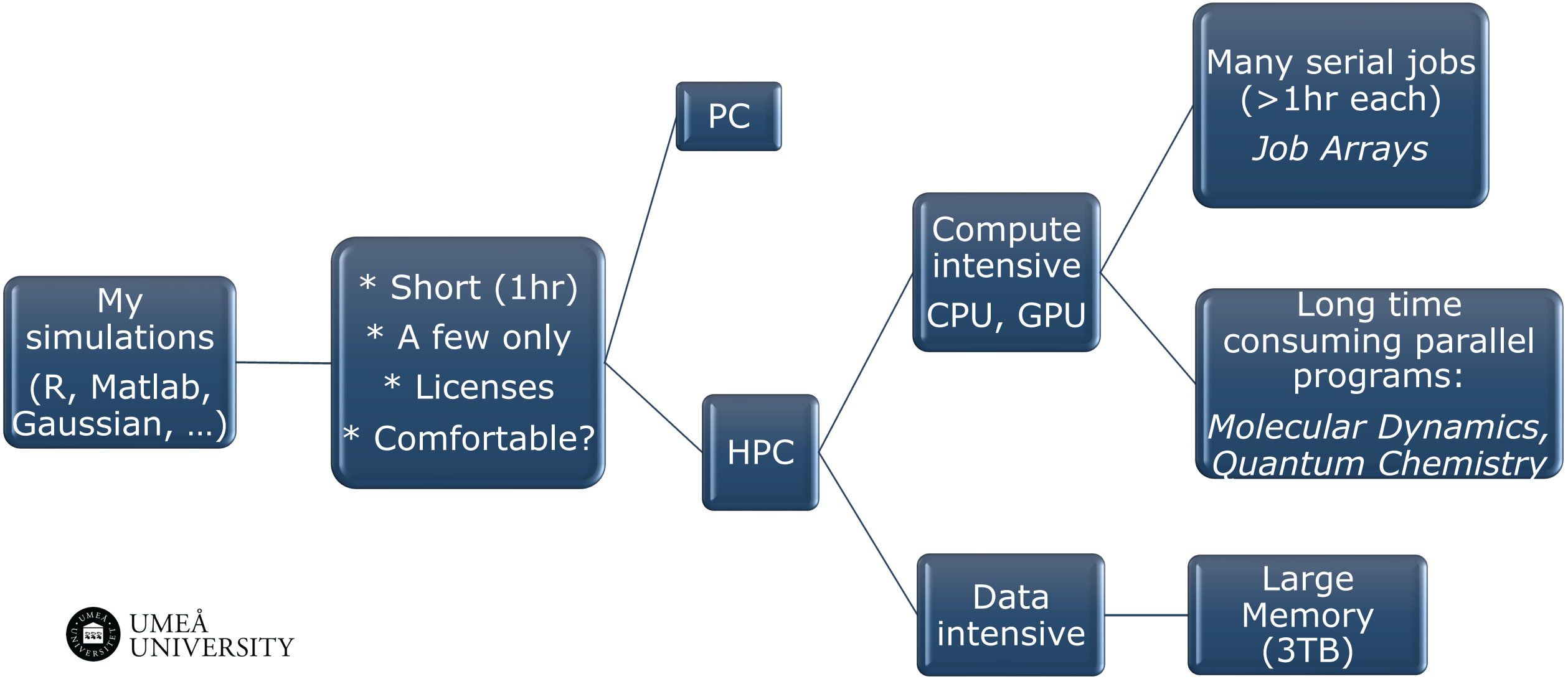
The resources' usage for standard nodes (without GPUs) is measured in

core-hours/month

while for GPU nodes it is measured in

GPU-hours/month

Why/when to use HPC?



HPC2N success stories

Clinical Infectious Diseases

MAJOR ARTICLE

Managing Coronavirus Disease 2019 Spread With Voluntary Public Health Measures: Sweden as a Case Study for Pandemic Control

Shina C. L. Kamerlin[1] and Peter M. Kasson[2,3]

[1]Science for Life Laboratory, Department of Chemistry–BMC, Uppsala University, Uppsala, Sweden, [2]Science for Life Laboratory, Department of Cell and Molecular Biology, Uppsala University, Uppsala, Sweden, and [3]Departments of Molecular Physiology and Biomedical Engineering, University of Virginia, Charlottesville, Virginia, USA

UMEÅ UNIVERSITY

HPC2N success stories

Mechanistic Basis for a Connection between the Catalytic Step and Slow Opening Dynamics of Adenylate Kinase

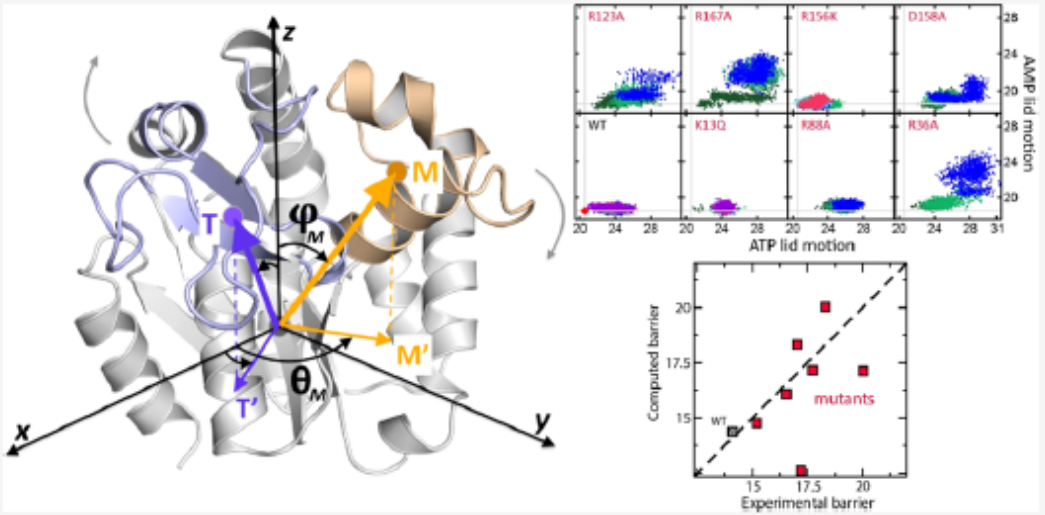
Beata Dulko-Smith, Pedro Ojeda-May, Jörgen Ådén, Magnus Wolf-Watz, and Kwangho Nam*

Cite This: *J. Chem. Inf. Model.* 2023, 63, 1556–1569

Read Online

ACCESS | Metrics & More | Article Recommendations | Supporting Information

ABSTRACT: *Escherichia coli* adenylate kinase (AdK) is a small, monomeric enzyme that synchronizes the catalytic step with the enzyme's conformational dynamics to optimize a phosphoryl transfer reaction and the subsequent release of the product. Guided by experimental measurements of low catalytic activity in seven single-point mutation AdK variants (K13Q, R36A, R88A, R123A, R156K, R167A, and D158A), we utilized classical mechanical simulations to probe mutant dynamics linked to product release, and quantum mechanical and molecular mechanical calculations to compute a free energy barrier for the catalytic event. The goal was to establish a mechanistic connection between the two activities.



HPC2N success stories

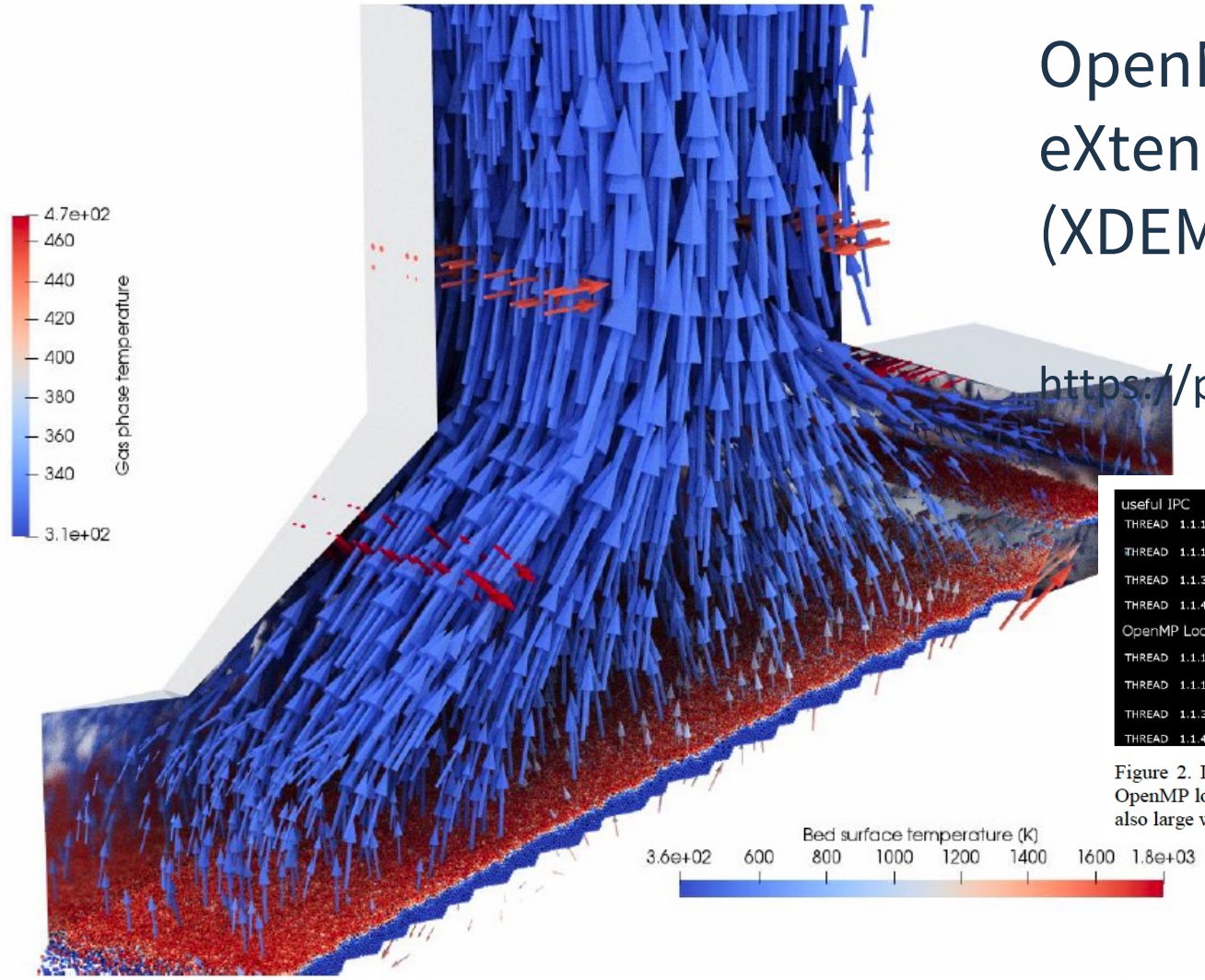


Figure 1. Biomass3D test case showing the fuel bed and gas phase, temperature scales are provided in the boxes

OpenMP optimisation of the eXtended Discrete Element Method (XDEM)

https://prace-ri.eu/wp-content/uploads/WP298.pdf

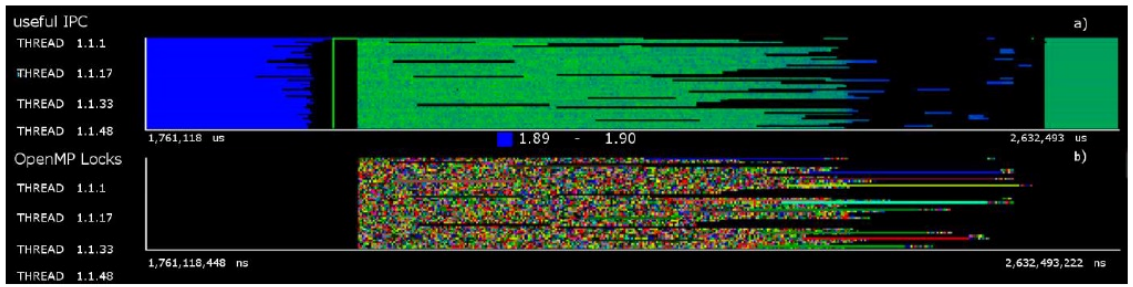
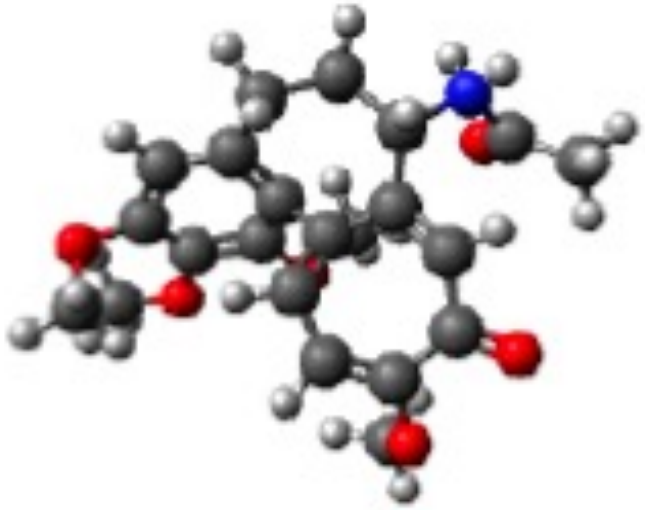


Figure 2. Initial profiling analysis with Extrae/Paraver showing the useful Instructions per Clock cycle (IPC) (a) and the OpenMP locks (b) for a cut region of the collected trace. In (a), blue colour represents high while green low IPC. There were also large waiting time regions in black. In (b) the addresses of memory for the locks are displayed with different colours.

HPC2N success stories

Dr. Christine Gallampois



Collision Cross Sections

$$\Omega_{av} = \frac{1}{4\pi^2}\int_0^{2\pi} d\theta \int_0^{\pi} d\varphi \sin\varphi \int_0^{2\pi} d\gamma \int_0^{\infty} db 2b(1-\cos\chi(\theta,\varphi,\gamma,b))$$

HPC2N success stories

Python or R expensive loops

```
for (i in 1:100) {
      heavy-function(i)
}
```

```
library(doParallel)

cl <- makeCluster(4)
registerDoParallel(cl)
foreach(i=1:100,.combine=cbind) %dopar% {
   heavy-function(i)
}
stopCluster(cl)
```

Reaching out

- Website: https://www.hpc2n.umu.se/
- General questions: **info@hpc2n.umu.se**
- For questions related to Kebnekaise for instance, software, hardware, login, projects: **support@hpc2n.umu.se**
- YouTube channel: https://www.youtube.com/user/HPC2N
- LinkedIn: https://www.linkedin.com/company/hpc2n/

Application’s usage

Tensorflow

```
#!/usr/share/python

import tensorflow as tf
hello = tf.constant('Hello,
TensorFlow!')
sess = tf.Session()
print(sess.run(hello))
```

```
#!/bin/bash
#SBATCH -A staff
#SBATCH -t 00:50:00
#SBATCH -N 1
#SBATCH --exclusive
#SBATCH --gres=gpu:k80:2

ml dependencies_of_Tensorflow
ml Tensorflow/version
python tensor.py
```

```
name: Tesla K80
major: 3 minor: 7 memoryClockRate (GHz)
0.8235
pciBusID 0000:0f:00.0
Total memory: 11.17GiB
Free memory: 11.10GiB
2017-12-05 17:13:03.555397: W
Found device 1 with properties:
name: Tesla K80
>>> print(sess.run(hello))
b'Hello, TensorFlow!'
```

Tensorflow

```
import tensorflow as tf

#Parameters
W = tf.Variable([.3],tf.float32)
b = tf.Variable([-.3],tf.float32)
#Input and output
x = tf.placeholder(tf.float32)
linear_model = W*x+b
y = tf.placeholder(tf.float32)
#Loss
square_delta = tf.square(linear_model-y)
loss = tf.reduce_sum(square_delta)
#Optimize
optimizer =
tf.train.GradientDescentOptimizer(0.01)
train = optimizer.minimize(loss)

init = tf.global_variables_initializer()
sess = tf.Session()
sess.run(init)

for i in range (1000):
    sess.run(train,{x:[1,2,3,4],y:[0,-1,-2,-3]})
print(sess.run([W,b]))
```

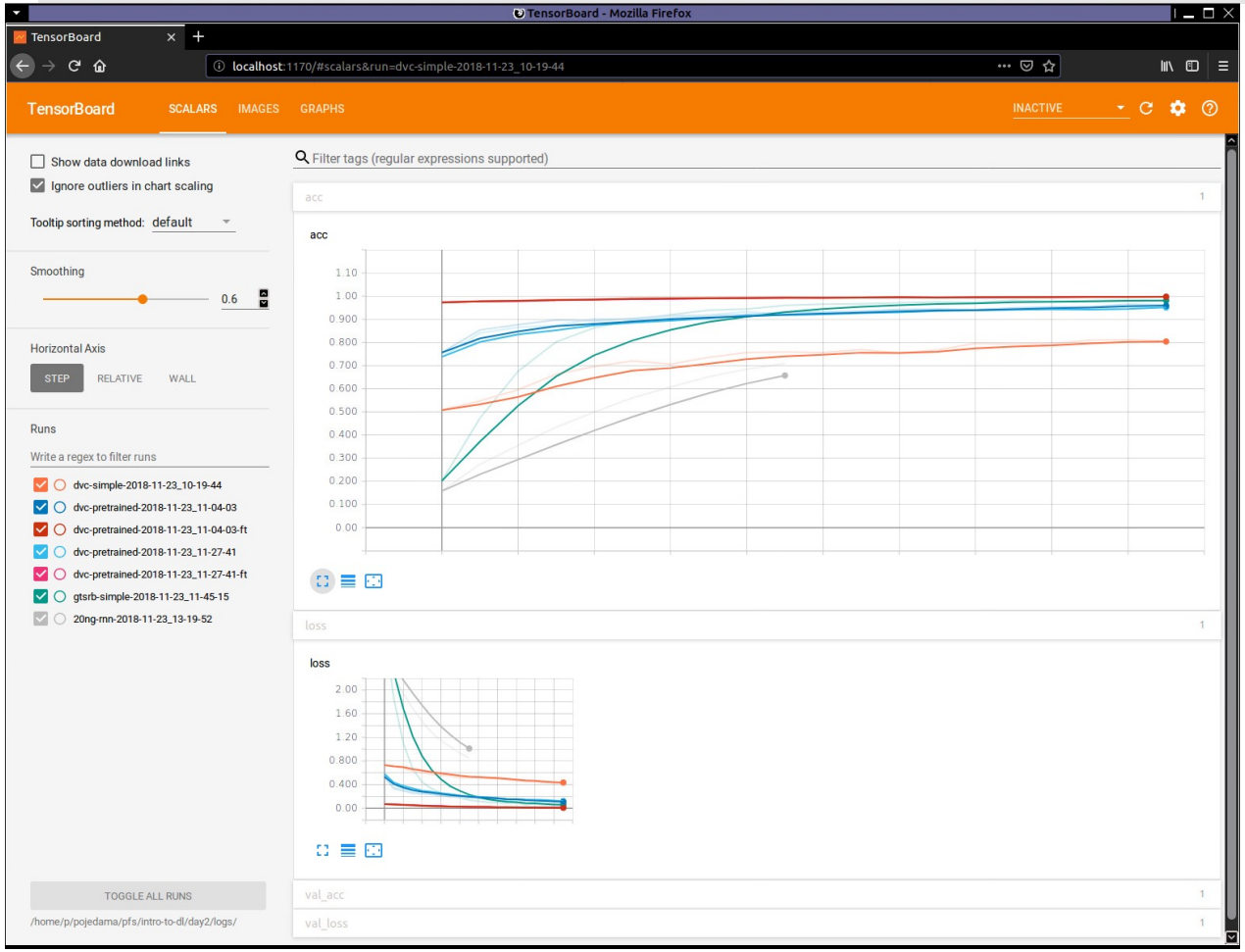
Computing the loss

name: Quadro K6000
major: 3 minor: 5 memoryClockRate (GHz) 0.9015
pciBusID 0000:06:00.0
Total memory: 11.92GiB
Free memory: 11.82GiB
Creating TensorFlow device (/gpu:0) -> (device: 0, name: Quadro K6000, pci bus id: 0000:06:00.0)

[array([-0.9999969], dtype=float32), array([0.99999082], dtype=float32)]

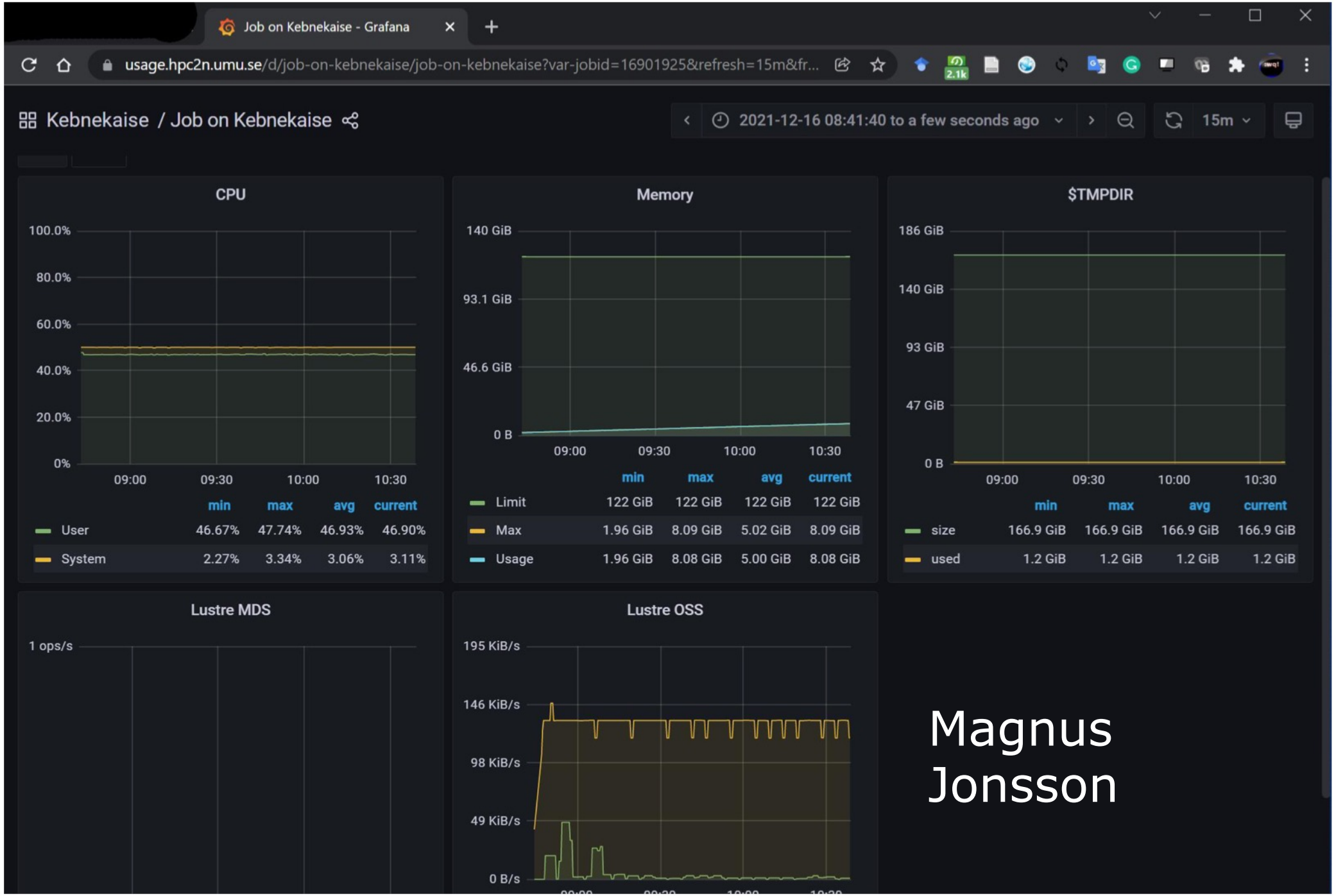
Tensorflow+Tensorboard

```
module load GCC/7.3.0-2.30  CUDA/9.2.88  OpenMPI/3.1.1     #On the command line execute these instructions
module load TensorFlow/1.10.1-Python-2.7.15 Keras/2.2.2-Python-2.7.15
tensorboard --logdir=/home/p/<path-to-directory>/logs --port=1170
```



To monitor the behavior of you jobs open a web browser and type:

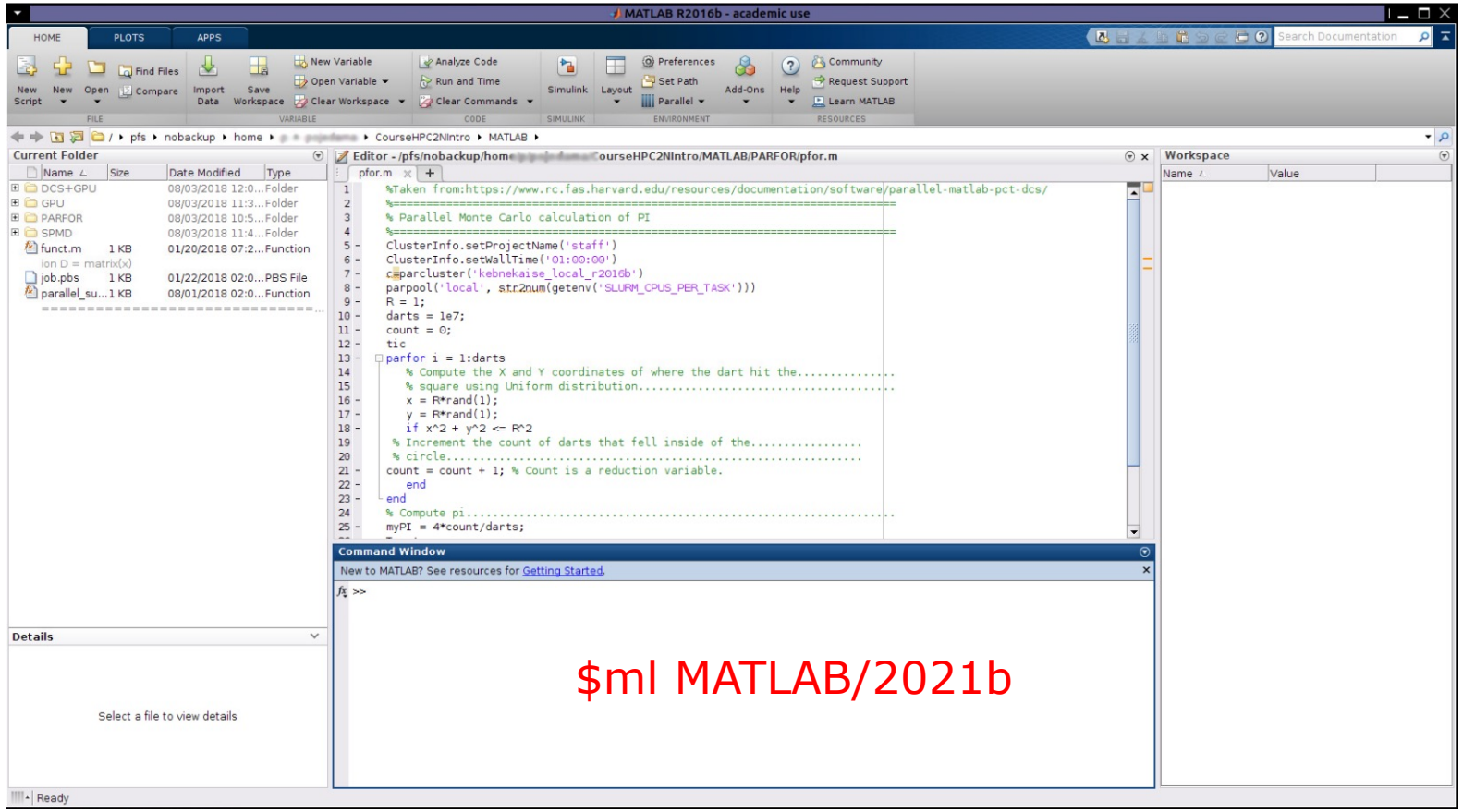
localhost:1170



Magnus Jonsson

MATLAB

One needs to configure MATLAB the first time one uses it:
(https://www.hpc2n.umu.se/resources/software/configure-matlab-2018)



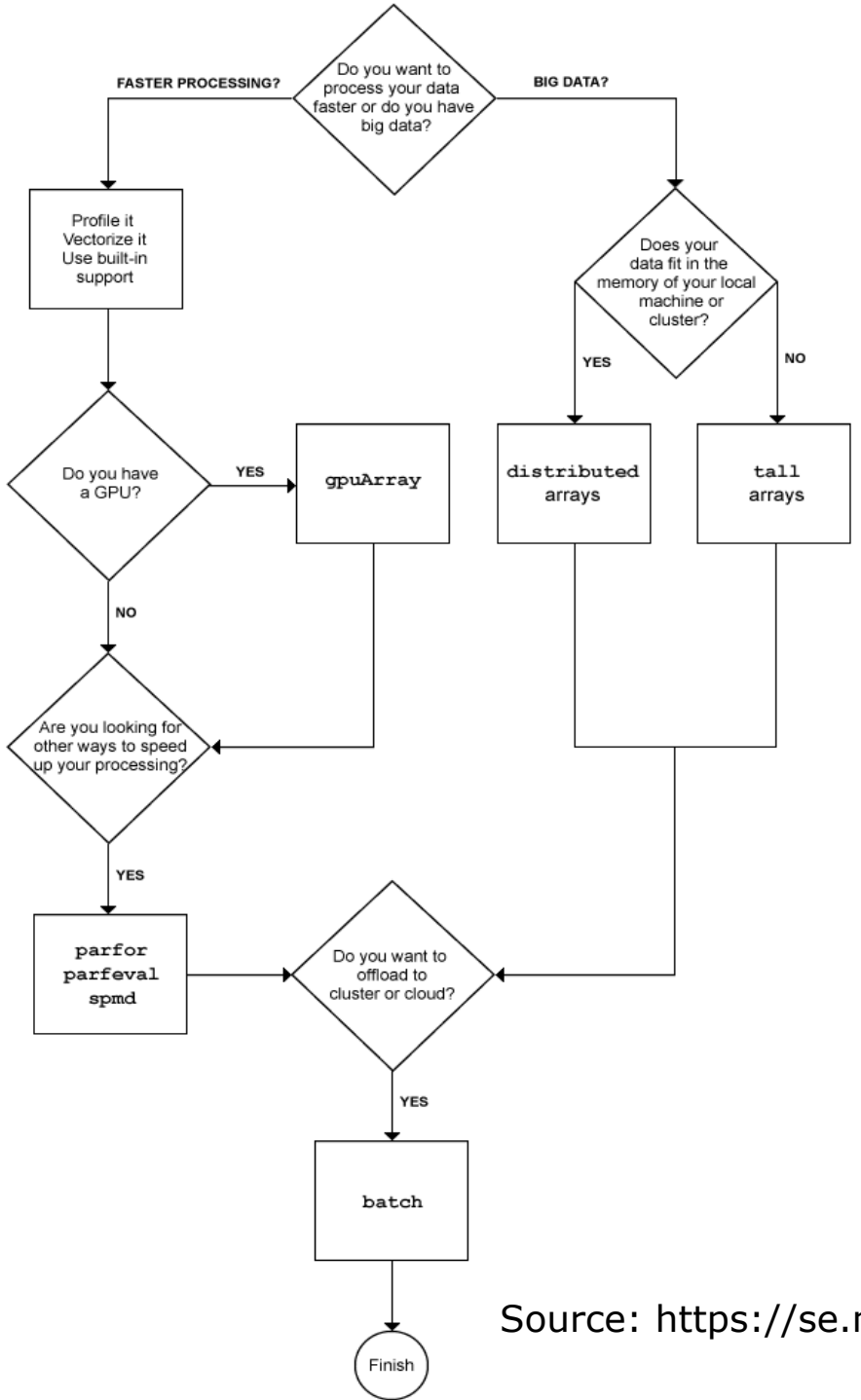
$ml MATLAB/2021b

Please use the GUI just for lightweight tasks, otherwise use the batch queues.

Heavy jobs will be cancelled!

Start Matlab with:
matlab -singleCompThread

Further information: https://www.hpc2n.umu.se/resources/software/matlab



MATLAB Parallel computing tools

Source: https://se.mathworks.com/help/parallel-computing/choosing-a-parallel-computing-solution.html

Serial job

File name: funct.m

```
function D

format long
A1 = rand(10000,10000);
tic;
B1 = fft(A1);
time1 = toc;

filename = 'log.out';
mid=fopen(filename,'w');
fprintf(mid,'Time = %16.12f\n',time1);
fclose(mid);
```

```
#!/bin/bash
#SBATCH -A hpc2n202X-XYZ
#Asking for 10 min.
#SBATCH -t 00:10:00
#SBATCH -n 1
#SBATCH -p batch

#Load modules necessary for running
MATLAB
ml MATLAB/2021b

matlab -nodisplay -singleCompThread -r
"funct;exit" > out.log
```

Parfor

```
parpool('local', str2num(getenv('SLURM_CPUS_PER_TASK')))
parfor i = 1:darts
x = rand(1);
y = rand(1);
   if x^2 + y^2 <= R^2
    count = count + 1; % reduction variable.
   end
end
```

File name: pfor.m

Batch script:

```
#SBATCH -c 4
#SBATCH -p batch
#Load modules necessary for MATLAB
ml MATLAB/2019b.Update2

matlab –nodisplay -r "pfor"
```

Timing

Nr. Workers	Time(sec)
1	26.43
2	17.17
3	11.82
4.	7.71
28	1.82

UMEÅ UNIVERSITY

Source: https://www.rc.fas.harvard.edu/resources/documentation/software/parallel-matlab-pct-dcs/

GPU jobs

On the MATLAB GUI:

```
c=parcluster('kebnekaise')

% submit the job to the queue
j = c.batch(@parallelex, 1, {})

% Wait for the job to finish.
j.wait
% Fetch the result after the job has finished
j.fetchOutputs{:}
```

File: parallelex.m

```
function speedUp =
parallel(x)
format long
A1 = rand(10000,10000);
tic;
B1 = fft(A1);
time1 = toc;


A2 = gpuArray(A1);
tic;
B2 = fft(A2);
time2 = toc;

speedUp = time1/time2;
end
```

R/Rstudio

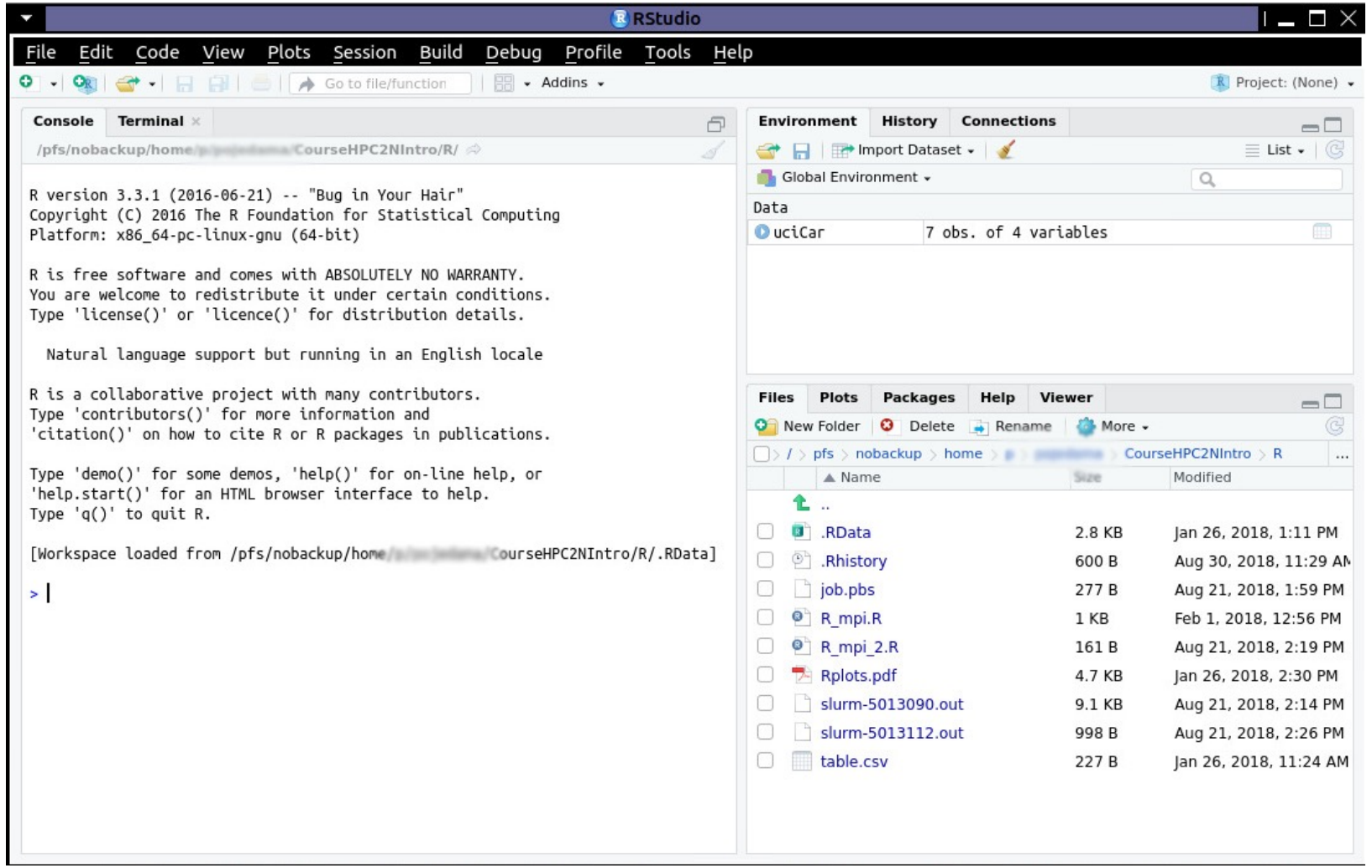
$ml dependecies_of_R

$ml R/version
$rstudio

Rstudio will use the loaded R version

Please use the GUI just for lightweight tasks, otherwise use the batch queues.

Heavy jobs will be cancelled!



UMEÅ UNIVERSITY

R/Rstudio

Installing packages

In case you need to install some R package that is not included in the installed R version, you need to follow this guideline:

https://www.hpc2n.umu.se/resources/software/user_installed/r

R serial job

```
#!/bin/bash
### SNAC project number, enter your own
#SBATCH -A hpc2n202X-XYZ
# Asking for one core
#SBATCH -n 1
#SBATCH --time=00:10:00

ml dependencies_of_R
ml R/version

R --no-save --quiet < input.R > Rexample.out
```

UMEÅ UNIVERSITY

Further information:
https://www.hpc2n.umu.se/resources/software/r

R parallel job (Rmpi)

File: Rmpi.R

```
library("Rmpi")
mpi.spawn.Rslaves(nslaves=5)

x <- c(10,20,30,40,50)
mpi.apply(x,runif)

# Close down the MPI processes and
quit R
mpi.close.Rslaves()
mpi.finalize()
```

```
#!/bin/bash
#SBATCH -A hpc2n202X-XYZ
#Asking for 10 min.
#SBATCH -t 00:10:00
#SBATCH -n 6

export OMPI_MCA_mpi_warn_on_fork=0

ml dependencies_of_R
ml R/version

R --no-save --quiet -f Rmpi.R
```


Further information:
https://www.hpc2n.umu.se/resources/software/r

R parallel job (doParallel)

```
#!/bin/bash
#SBATCH -A hpc2n202X-XYZ
#Asking for 10 min.
#SBATCH -t 00:10:00
#SBATCH -n 8

ml dependencies_of_R
ml R/version

R -q --slave -f doParallel_ML.R
```

doParallel_ML.R

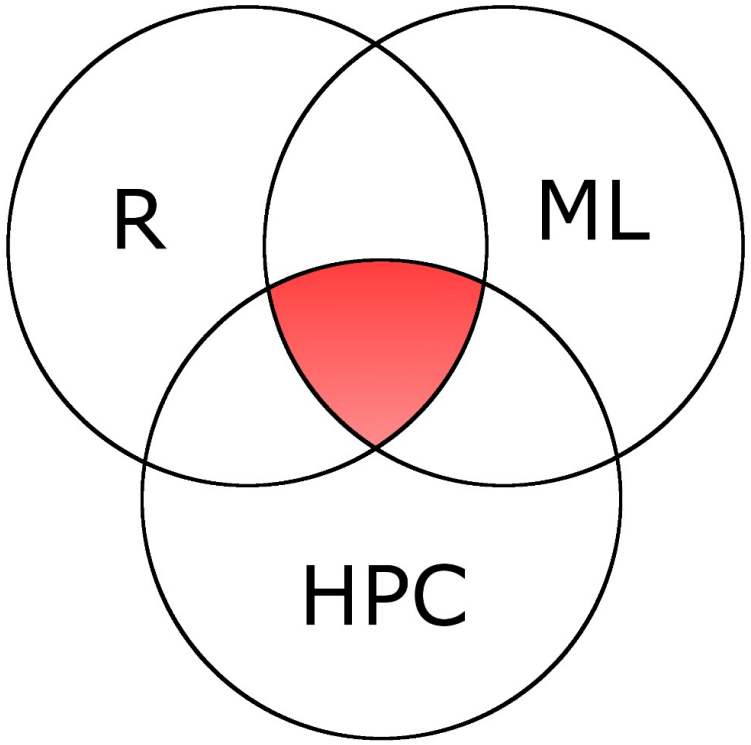
```
library(doParallel)
registerDoParallel(cores=8)
getDoParWorkers()

library(caret)
library(MASS)
library(klaR)
library(nnet)
library(e1071)
library(rpart)
```

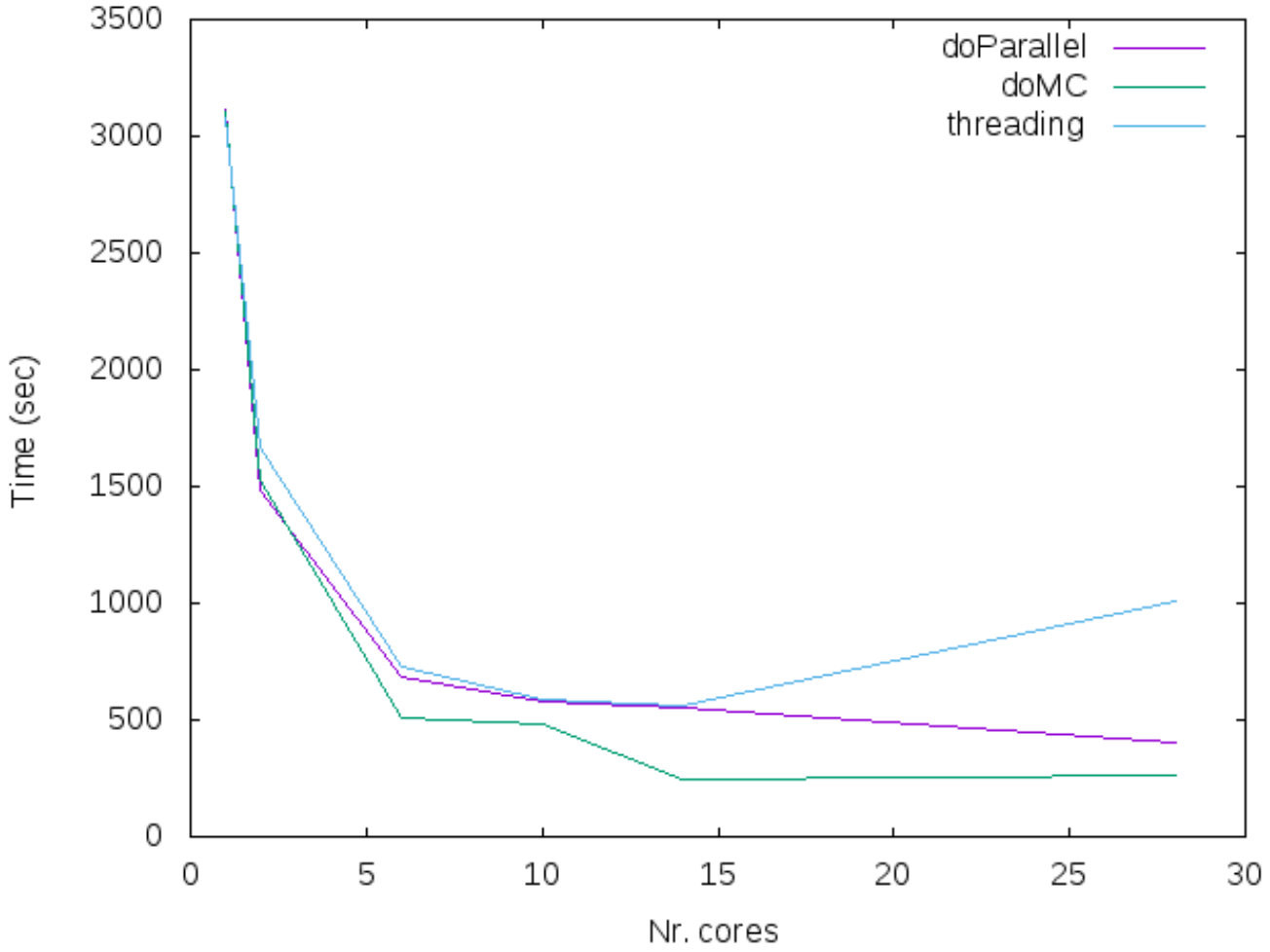
UMEÅ UNIVERSITY

Further information:
https://www.hpc2n.umu.se/resources/software/r

R Machine Learning job

Example: Boosted classification model using "**xgboost**" (7 tunning parameters). Further details: http://appliedpredictivemodeling.com/blog/2018/1/17/parallel-processing).





UMEÅ UNIVERSITY

N-body Dynamics

List updating

Short-range interactions

PP MPI GROUP

Long-range interactions:
PME, FFT

PME MPI
GROUP
(NlogN)

GPU
OFFLOAD
(idle)

Bonded interactions

Update positions

Benchmark

- Solvated protein
- 158945 atoms
- 1.2 nm cutoff radius
- 1 fs time step
- PME for electrostatics

http://www.charmm-gui.org/

AMBER

- Collection of independent routines
- It uses Sander/PMEMD for solving the Newton's equations
- It offers a robust set of analysis tools
- Resources: https://ambermd.org/

```
# module load dependencies_of_amber
ml Amber/version
srun pmemd.MPI -O -i  input.mdin
srun pmemd.cuda.MPI -O –I input.mdin
```

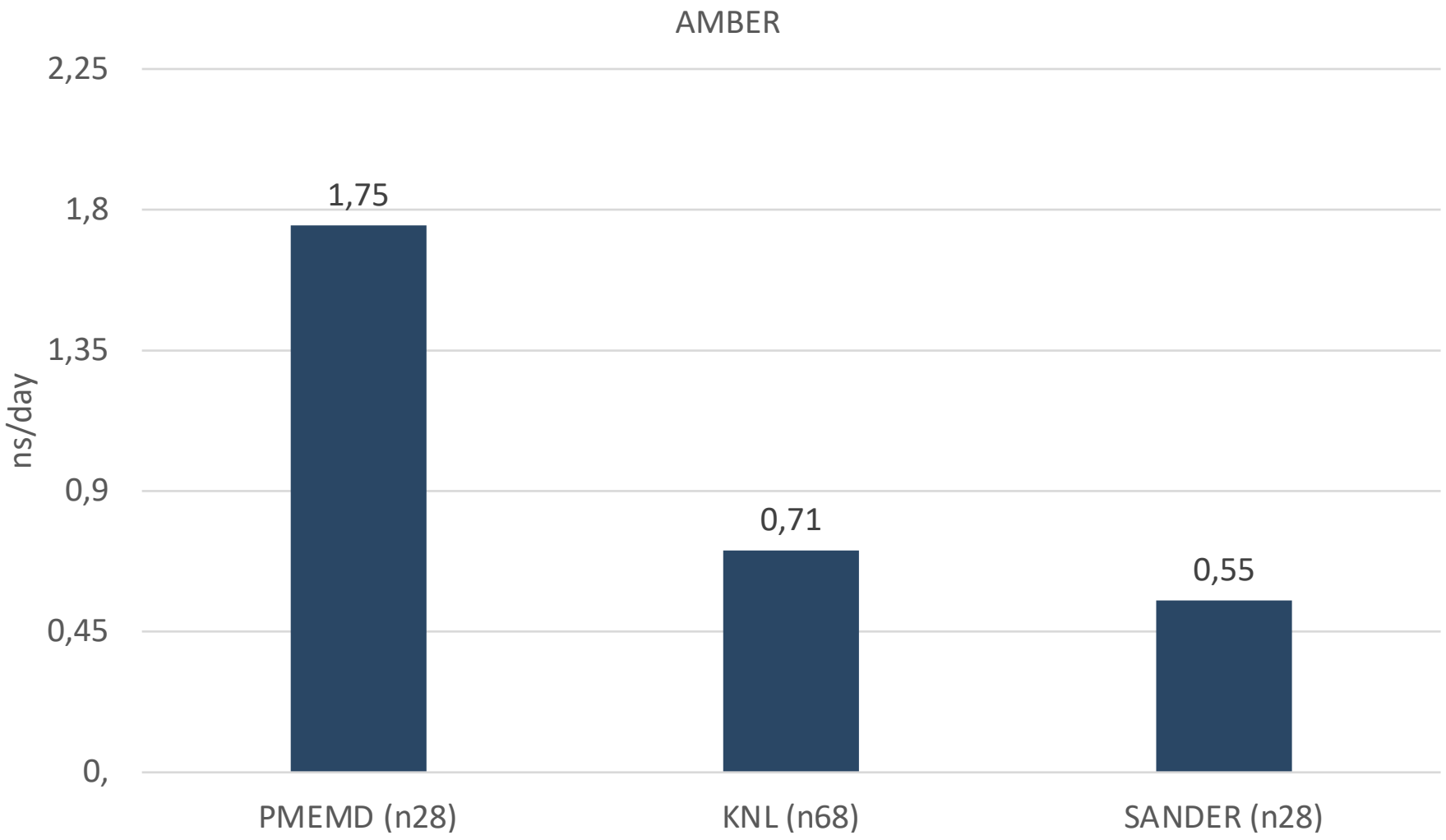
AMBER Tools

- For setting up a simulation (initial structure, solvation, ions, …):

On the command line:

$tleap, antechamber, cpptraj, …

AMBER

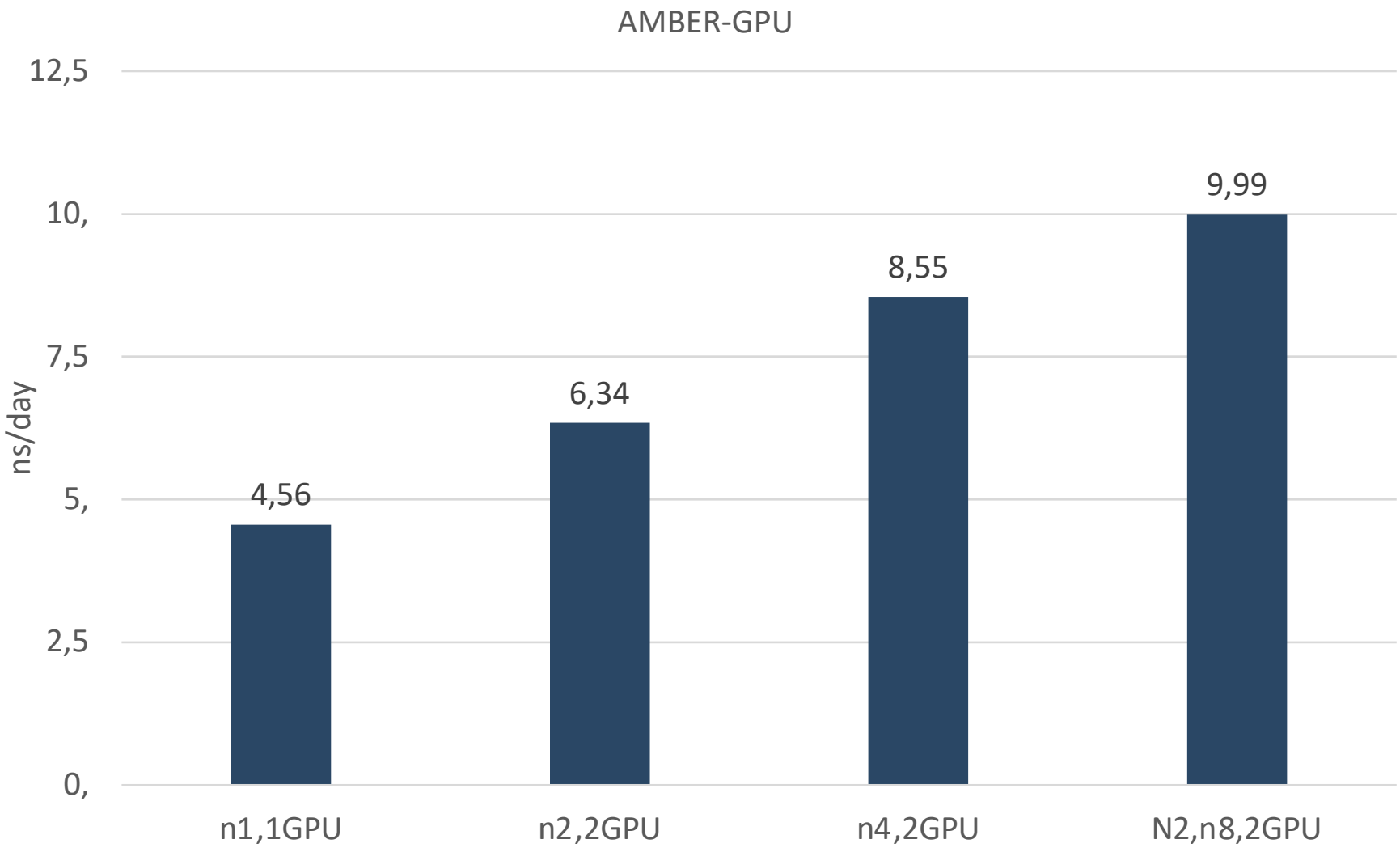


UMEÅ UNIVERSITY

AMBER

On Kebnekaise, best performance is achieved with 4 MPIs/Node and using 2 GPU cards

Notice that, the remaining CPUs are not used.



NAMD

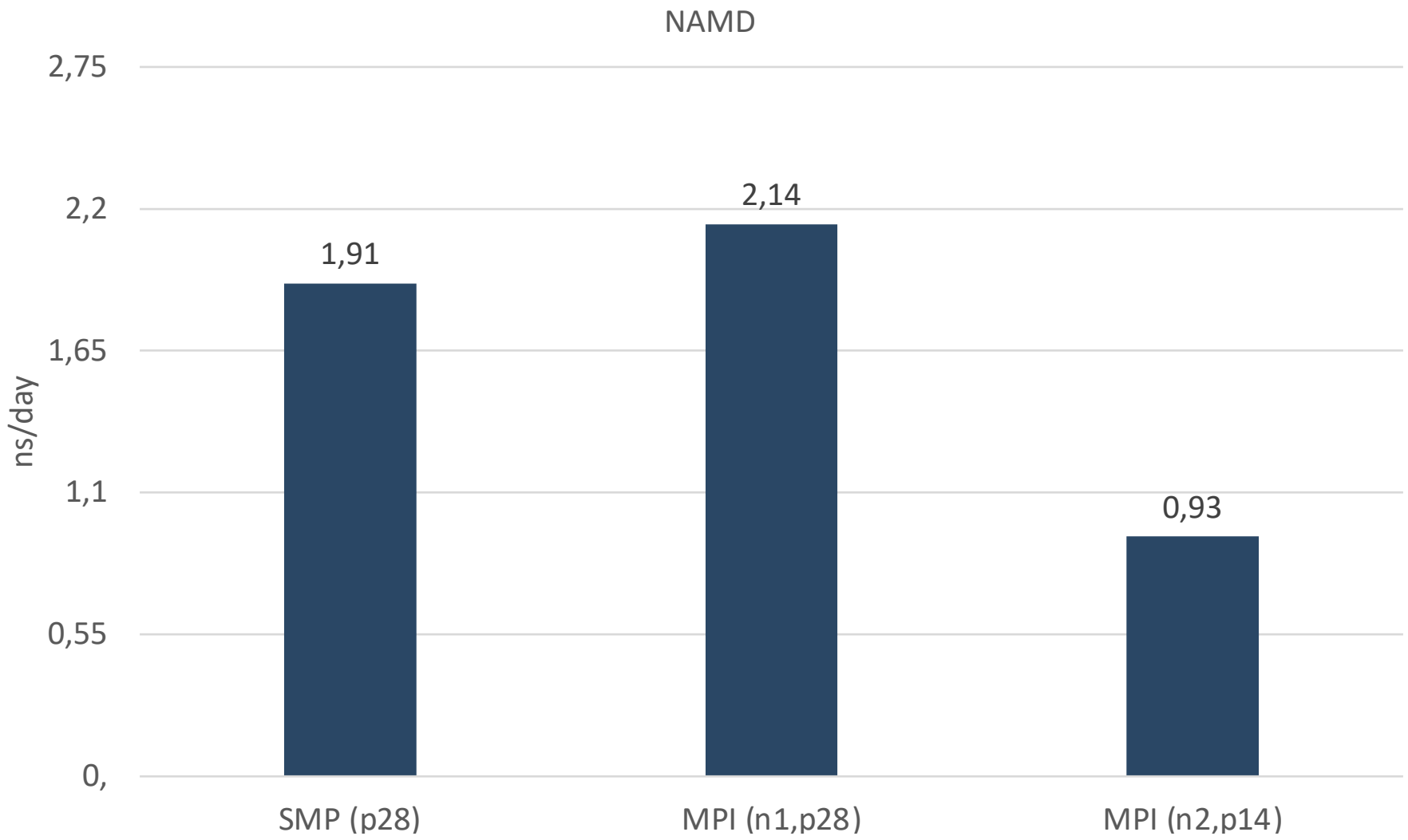
- Based on charm++ communication protocol
- Versions: single node, multi-node, and GPU
- Highly scalable
- Resources: https://www.ks.uiuc.edu/Research/namd/

NAMD GPU

```
#!/bin/bash
#SBATCH -A staff
#SBATCH -t 00:50:00
#SBATCH -n 28
#SBATCH –exclusive
#Ask for 2 GPU cards
#SBATCH --gres=gpu:k80:2

#Load modules necessary for running NAMD
ml GCC/9.3.0  CUDA/11.0.2  OpenMPI/4.0.3
ml NAMD/2.14-nompi
#Execute NAMD
namd2 +p 28 +setcpuaffinity step4_equilibration.inp > output.dat
```

NAMD

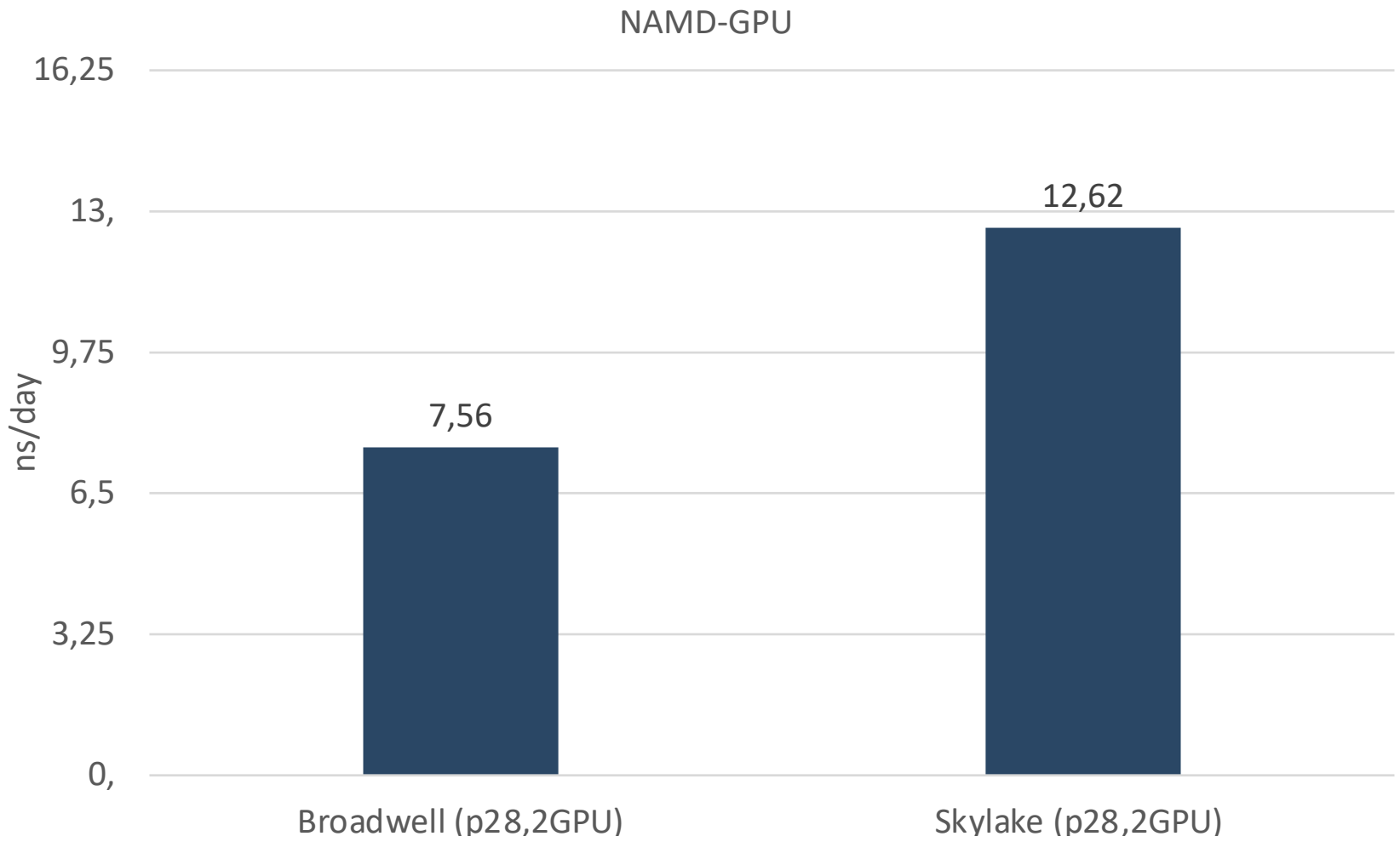
Use the
+setcpuaffinity flag
for a 10% speedup



UMEÅ UNIVERSITY

NAMD

With Volta GPU cards one can speed up the simulation by 1.66x



UMEÅ UNIVERSITY

GROMACS

- Resources:
  - https://www.gromacs.org/index.html (website)
  - https://gromacs.bioexcel.eu/top?period=monthly (forum)

GROMACS

```
#SBATCH -n 4
#SBATCH -c 7
# Asking for 2 GPUs
#SBATCH --gres=gpu:k80:2
#SBATCH -p batch

ml dependencies_of_gromacs
ml GROMACS/version

if [ -n "$SLURM_CPUS_PER_TASK" ]; then
    mdargs="-ntomp $SLURM_CPUS_PER_TASK"
else
    mdargs="-ntomp 1"
fi

srun -n $SLURM_NTASKS gmx_mpi mdrun $mdargs -npme 0 -dlb yes  -v -deffnm step4.1_equilibration
```

GROMACS recognizes the number of available GPU cards

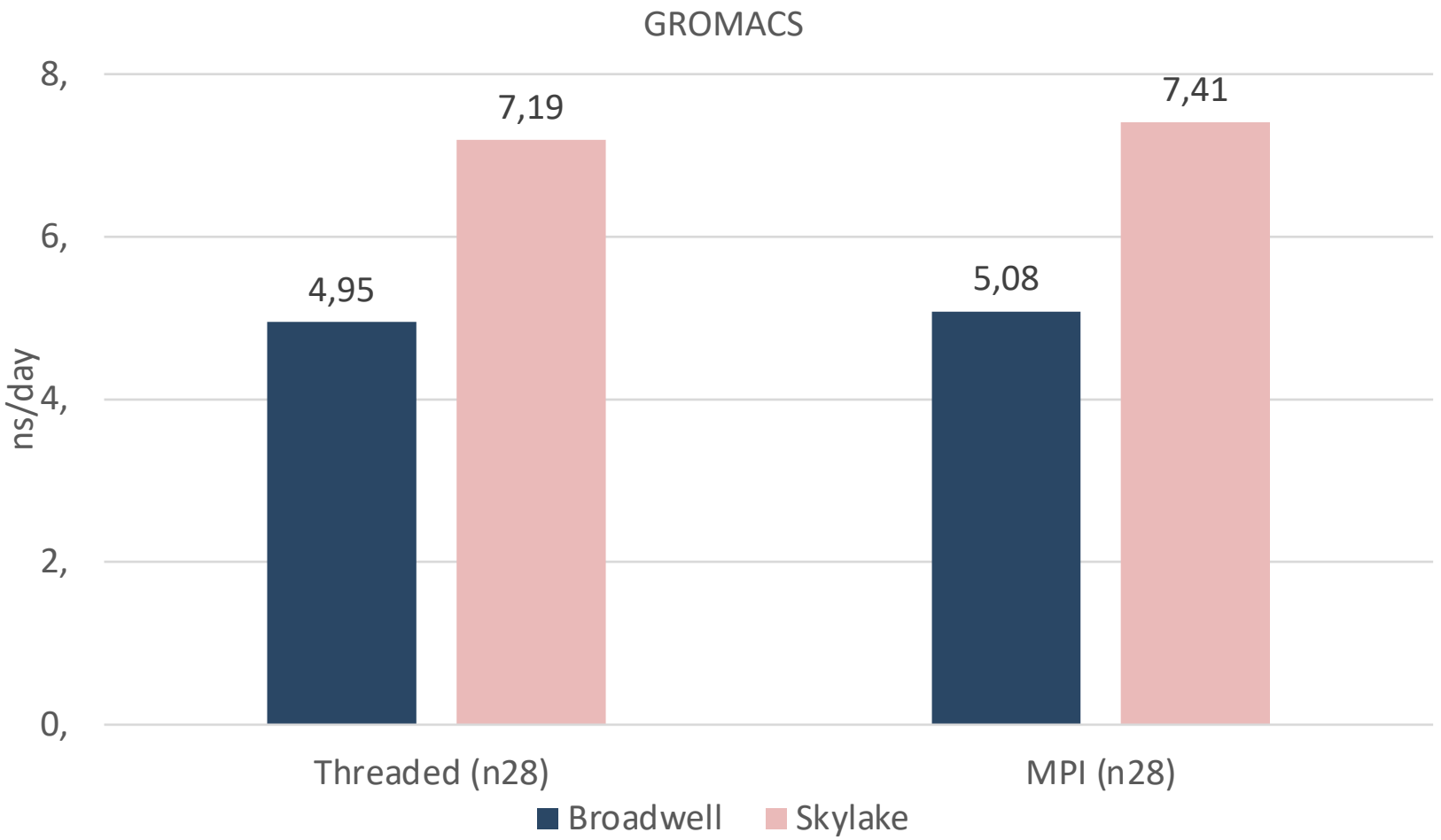
GROMACS

gmx tune_pme

```
Individual timings for input file 0 (npt_bench00.tpr):
PME ranks      Gcycles       ns/day        PME/f    Remark
   0         4355.019      57.776         -      OK.
   0         4547.105      55.335         -      OK.
   0         4289.420      58.659         -      OK.
  -1(  0)    4455.791      56.469          -     OK.
  -1(  0)    4440.157      56.668          -     OK.
  -1(  0)    4275.551      58.850          -     OK.

Tuning took     7.7 minutes.
------------------------------------------------------------
Summary of successful runs:
Line tpr PME ranks  Gcycles Av.     Std.dev.       ns/day       PME/f    DD grid
   0   0    0         4397.181      133.917       57.257         -      4   1   1
   1   0   -1(  0)    4390.500       99.855       57.329          -     4   1   1

------------------------------------------------------------
Best performance was achieved with the automatic number of PME ranks (see line 1)
Please use this command line to launch the simulation:

mpirun -np 4 gmx_mpi mdrun -npme -1 -s npt.tpr -ntomp 7 -dlb yes
```

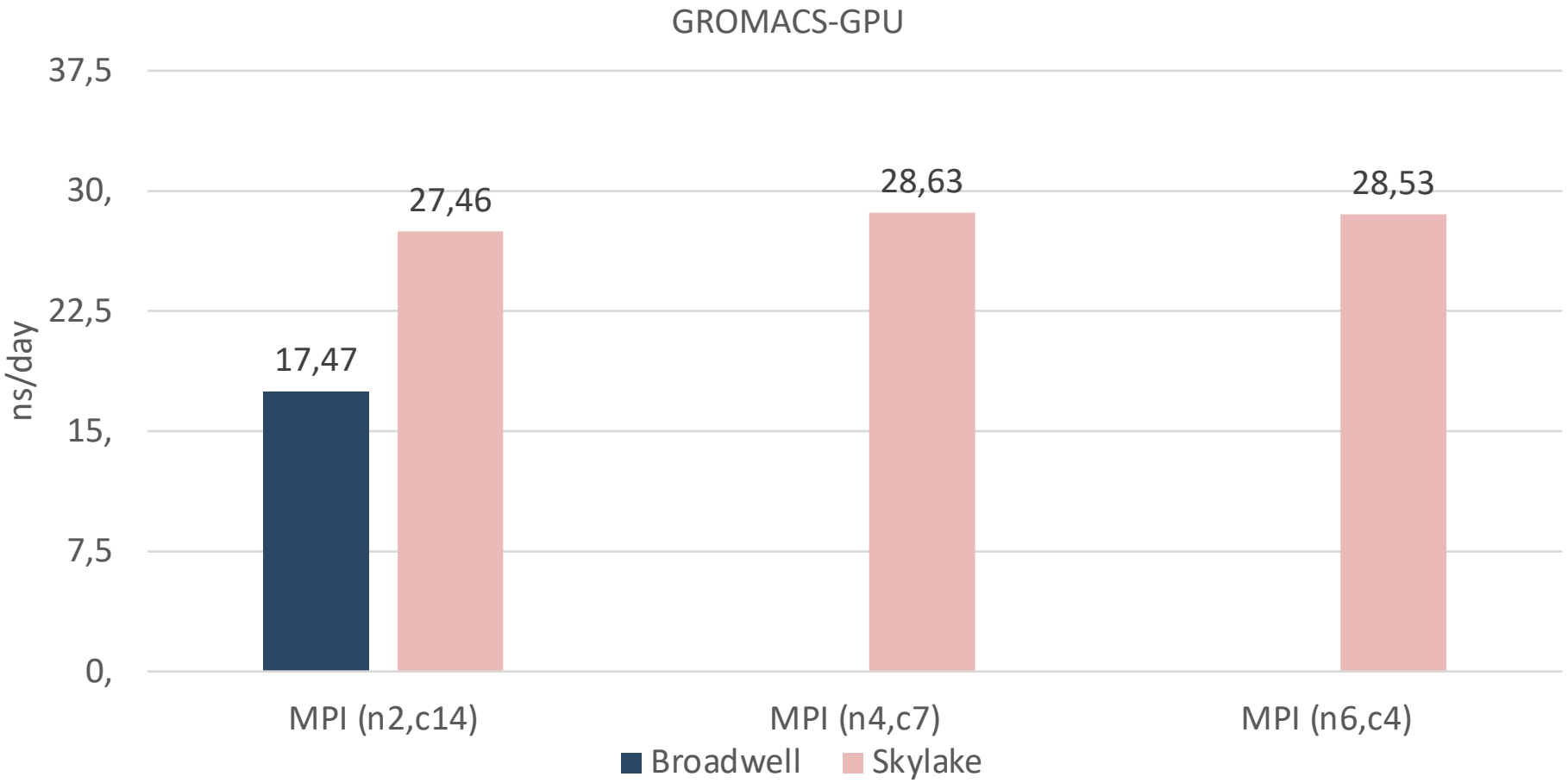
UMEÅ UNIVERSITY

GROMACS



UMEÅ UNIVERSITY

GROMACS



UMEÅ UNIVERSITY

GROMACS

- PLUMED, threaded MPI version is not supported.

Instead of:

gmx mdrun –ntmpi X

Use:

mpirun gmx_mpi …

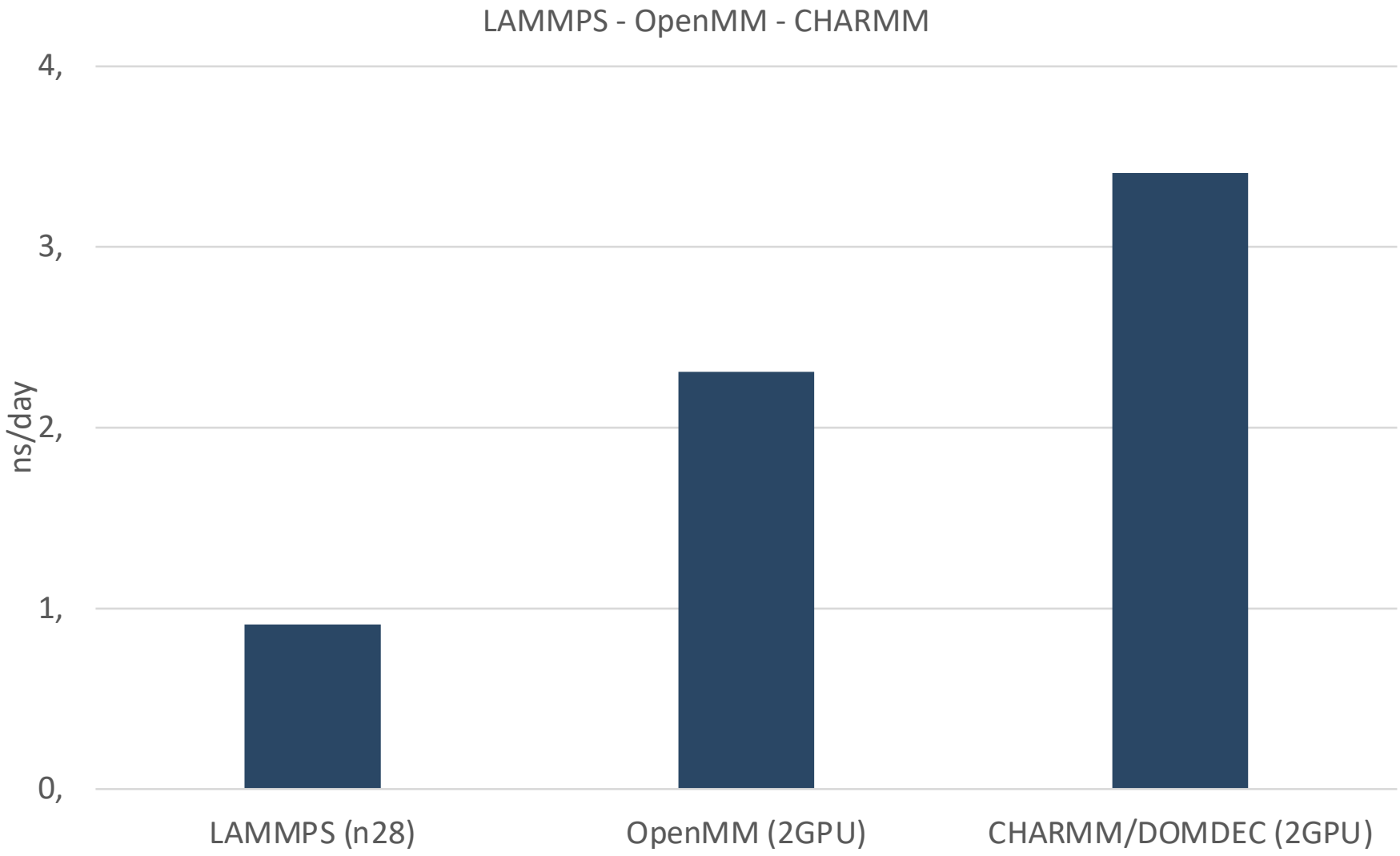
- Avoid writing energies

- Offloading features starting from 2018 version

LAMMPS

```
#!/bin/bash
#SBATCH -A staff
#Asking for 10 min.
#SBATCH -t 02:10:00
#Ask for 28 processes
#SBATCH -n 28

#Load modules necessary for running LAMMPS
module load LAMMPS/version

#Execute LAMMPS
srun lmp -in step4.1_equilibration.inp
```

LAMMPS-OpenMM-CHARMM



UMEÅ UNIVERSITY

Suggestions

- Check if the software version you are trying is MPI (-n), OpenMP (-c), or hybrid (-n -c). Also, if you are using a GPU version.
- Is your simulation scaling properly?
- Is the cutoff radius for long-range interactions appropriate? PME (gmx tune_pme for GROMACS)?

Gaussian

Initial input file:

```
$more input_bk.com
%chk=geom_optim.chk
%mem=16GB
#UB3LYP/6-31+G(d) OPT=(ModRedun) SCF=(MaxCycle=256) pop=none NoSymm

45 atoms structure, RESP

+5 15
O       3.744336        -1.126487       6.111505
P       2.893853        -1.251776       4.246949
O       4.150424        -3.154051       4.078061
```

Corrected input file:

```
$more input.com
%cpu=0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27
%gpucpu=0-3=0,7,14,21
%chk=geom_optim.chk
%mem=16GB
#UB3LYP/6-31+G(d) OPT=(ModRedun) SCF=(MaxCycle=256) pop=none NoSymm
```

```
#!/bin/bash
#SBATCH -A staff
#SBATCH -N 1
#SBATCH -c 28
#SBATCH --exclusive
#SBATCH --gres=gpu:k80:2
#SBATCH --time=00:10:00

module add gaussian/16.A.03-AVX2
# Assume that the job file are located
g16.set-cpu+gpu-list input.com
time g16 input
```

Integral=(FineGrid,Acc2E=10)
Constants=2006

Comments

- Check if the software version you are trying is MPI (-n), OpenMP (-c), or hybrid (-n -c). Also, if you are using a GPU version.
- Is the number of cores/nodes you request optimal? Trying a short simulation could help.
- Is your simulation scaling well?
- Use job-usage script on the command line, this will allow you to monitor the resources usage in your local browser

Comments

- Upon asking a question to the support team: Make a folder with the case which displays the issue. Including all relevant files would be useful.
- Keep the topics of the support tickets independent (one ticket per case)
- Avoid using SLURM batch scripts from other centers, use those provided in our web site.

- Don’t load modules in your *.bashrc* file, this can cause troubles with software.
- Don’t forget to cite HPC2N upon publishing.