Useful examples

Pedro Ojeda-May, Jerry Eriksson, and Birgitte Brydsö

HPC2N,
UmeåUniversity,

901 87, Sweden.

January 21, 2018

Table of contents

R/Matlab

- R is installed on Abisko and Kebnekaise. The MPI version is only available on Kebnekaise.
- If additional modules are required, they can be installed locally.
- Regarding Matlab, one needs an initial setup (.matlab file)
- Both serial and parallel versions of Matlab are available on Abisko and Kebnekaise.
- Kebnekaise supports the GPU version.

Matlab Scripting

Create a file with your matlab script (funct.m):

```
function D
format long
P1 = rand(10000,10000);
tic;
Q1 = fft(P1);
time1 = toc;
filename = 'log.out';
mid=fopen(filename,'w');
fprintf(mid,'Time = %16.12f\n',time1);
fclose(mid);
```

Create your batch script (job.pbs):

Matlab Scripting

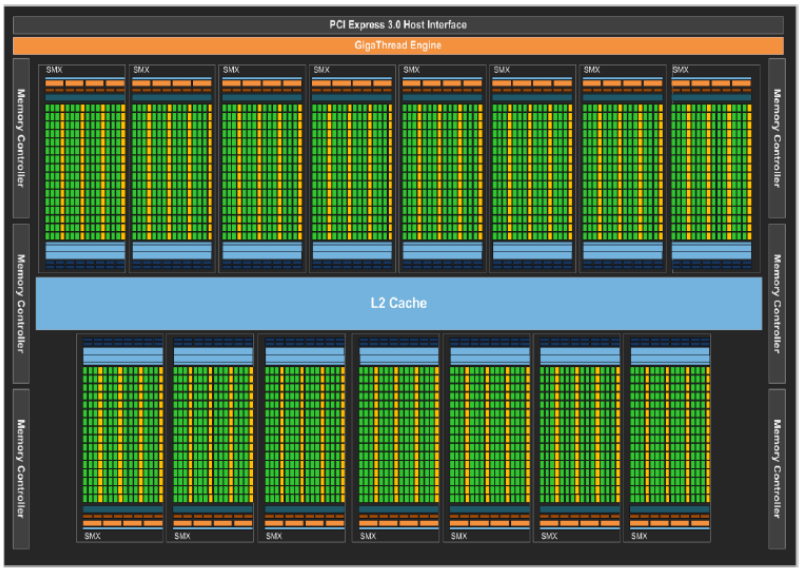
```
#!/bin/bash
#SBATCH -A <your project id>
#Asking for 10 min.
#SBATCH -t 00:10:00
#Number of nodes
#SBATCH -n 1
#Load Matlab
ml MATLAB/2016b
srun -n1 --exclusive matlab -nodesktop -nodisplay \
      -singleCompThread -r "funct;exit" > out.log
```

Submit your script:

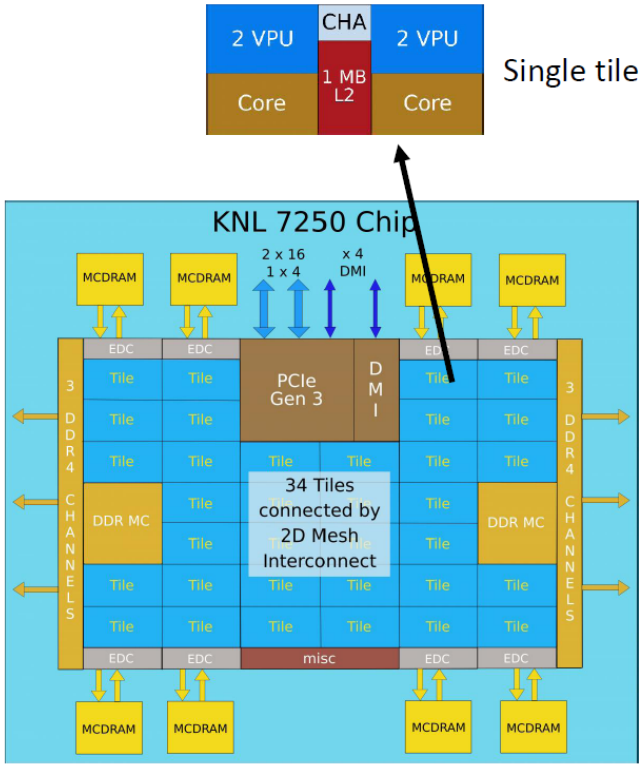
sbatch job.pbs

Accelerating applications

Accelerators



GPU showing the independent units Streaming Multiprocessors (SM).



KNL, composed of several Tiles

Matrix multiplication code (serial)

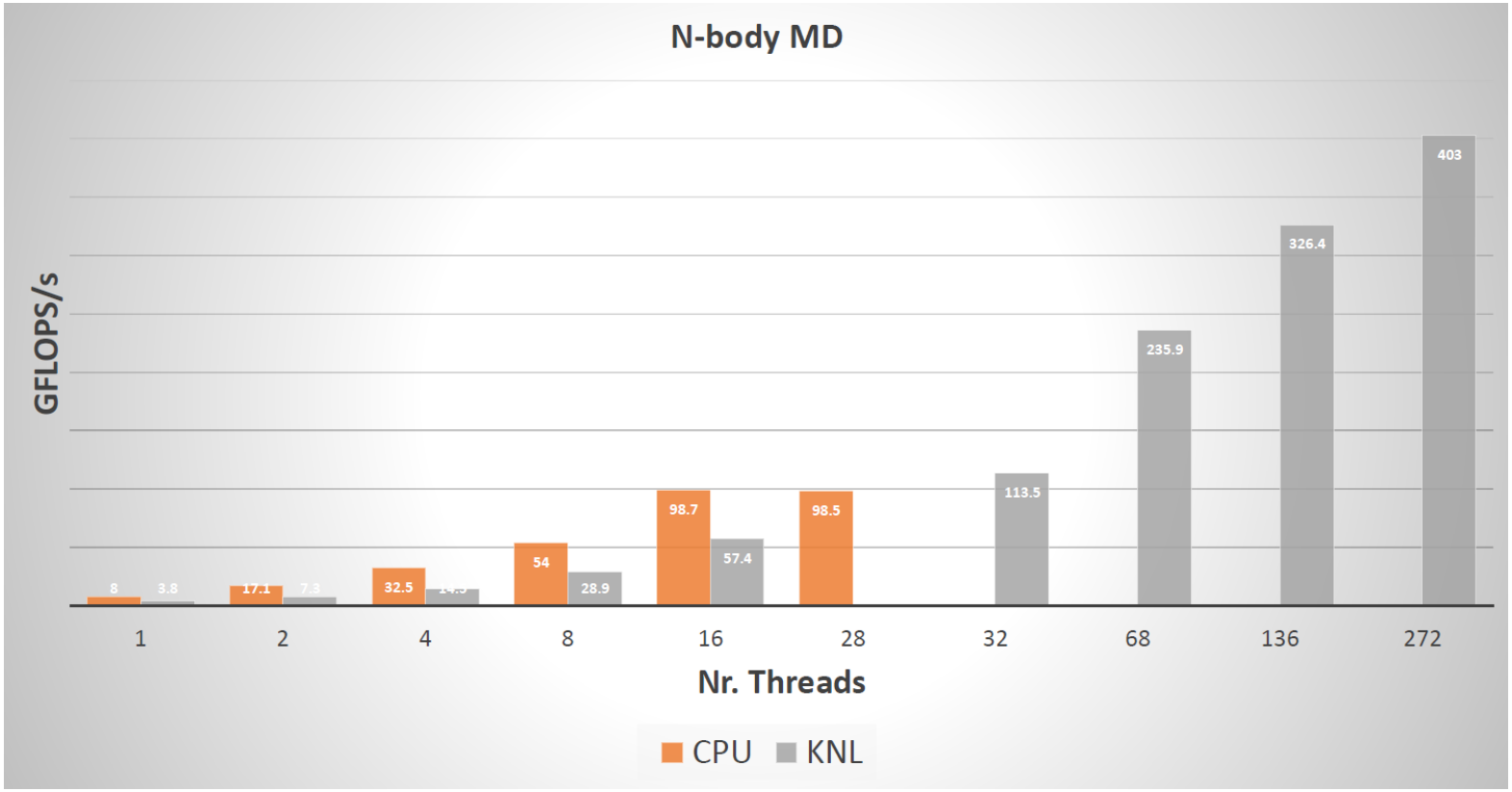
```
do j=1,N
   do i=1,N
      do k=1,N
          p(i,j) = p(i,j) + v1(i,k)* v2(k,j)
      enddo
   enddo
enddo
```

Matrix multiplication code (offload GPU)

```
!$omp target map(tofrom:p) map(to:v1,v2)
!$omp parallel do collapse(2) private(i,j,k,s)
      do j=1,N
         do i=1,N
            s=0.0d0
!$omp simd
            do k=1,N
                s = s + v1(i,k)* v2(k,j)
            enddo
!$omp end simd
            p(i,j) = s
         enddo
      enddo
!$omp end parallel do
```

Matrix multiplication code (OpenMP, KNL)

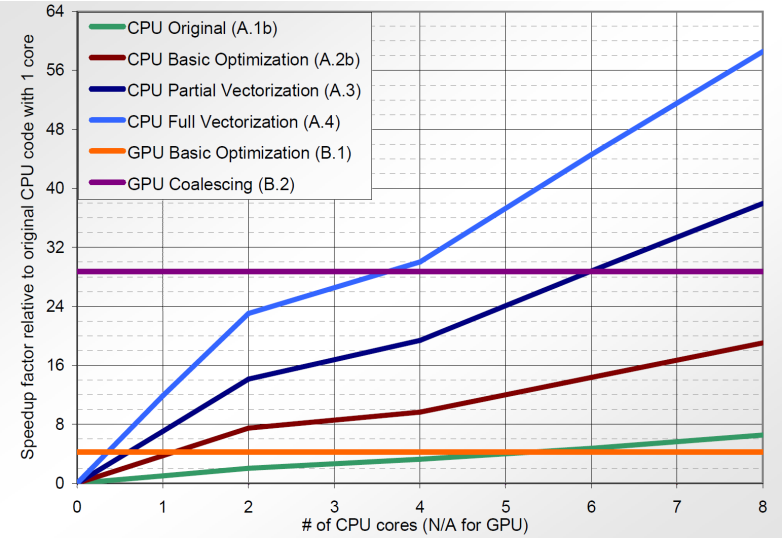
```
!$omp parallel do private(i,j,k) shared(p,v1,v2) collapse(2)
      do j=1,N
         do i=1,N
            s=0.0d0
!$omp simd
            do k=1,N
                s = s + v1(i,k)* v2(k,j)
            enddo
!$omp end simd
            p(i,j) = s
         enddo
      enddo
!$omp end parallel do
```


N-body MD
GFLOPS/s
8
3.8
17.1
7.3
32.5
54
28.9
98.7
57.4
98.5
113.5
235.9
326.4
403
1
2
4
8
16
28
32
68
136
272
Nr. Threads
CPU
KNL

CPU vs. GPU

What if we consider explicit vectorization when using CPUs? Potts model:

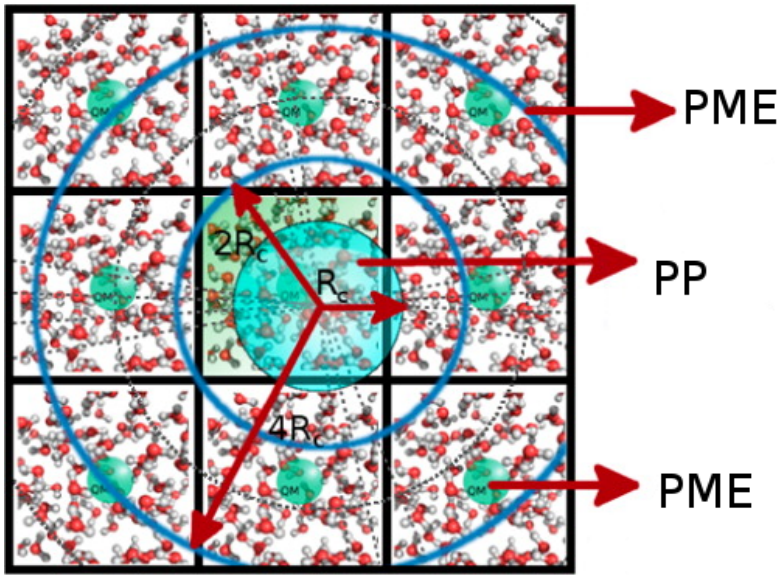
$$U = -\sum_{i} h_i s_i - \sum_{i,j} J_{ij} s_i s_j \tag{1}$$



Credits: J. Comp. Phys., 230, 5383 (2011).

MD applications

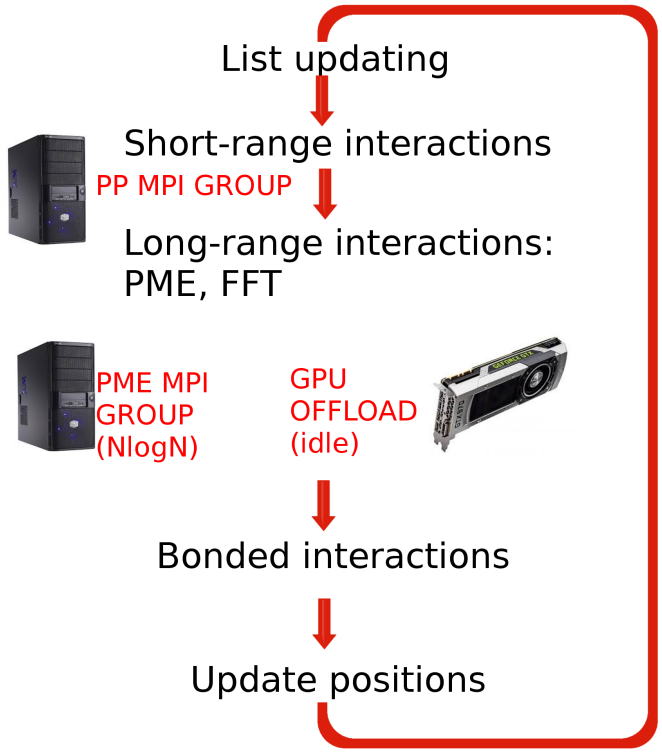
Long-range interactions



adapted from JCTC,10,134 (2014)

- particle-particle interactions are solved in real space (80%)
- PME contribution is solved in reciprocal space (20%)

Workflow MD code



AMBER

```
$module spider amber
----------------------------------------------------------------------
  Amber:
----------------------------------------------------------------------
    Description:
      Amber (originally Assisted Model Building with Energy Refi

     Versions:
        Amber/16-AmberTools-16-patchlevel-20-7-hpc2n (GPU)
        Amber/16-AmberTools-16-patchlevel-20-7
```

AMBER batch script (Sander)

```
#!/bin/bash
#SBATCH -A <Your-Project-Here>
#SBATCH -n 8
#SBATCH --time=01:00:00

module load icc/2017.1.132-GCC-5.4.0-2.26  impi/2017.1.132
module load Amber/16-AmberTools-16-patchlevel-20-7

srun sander.MPI -ng 8 -groupfile equilibrate.groupfile
```

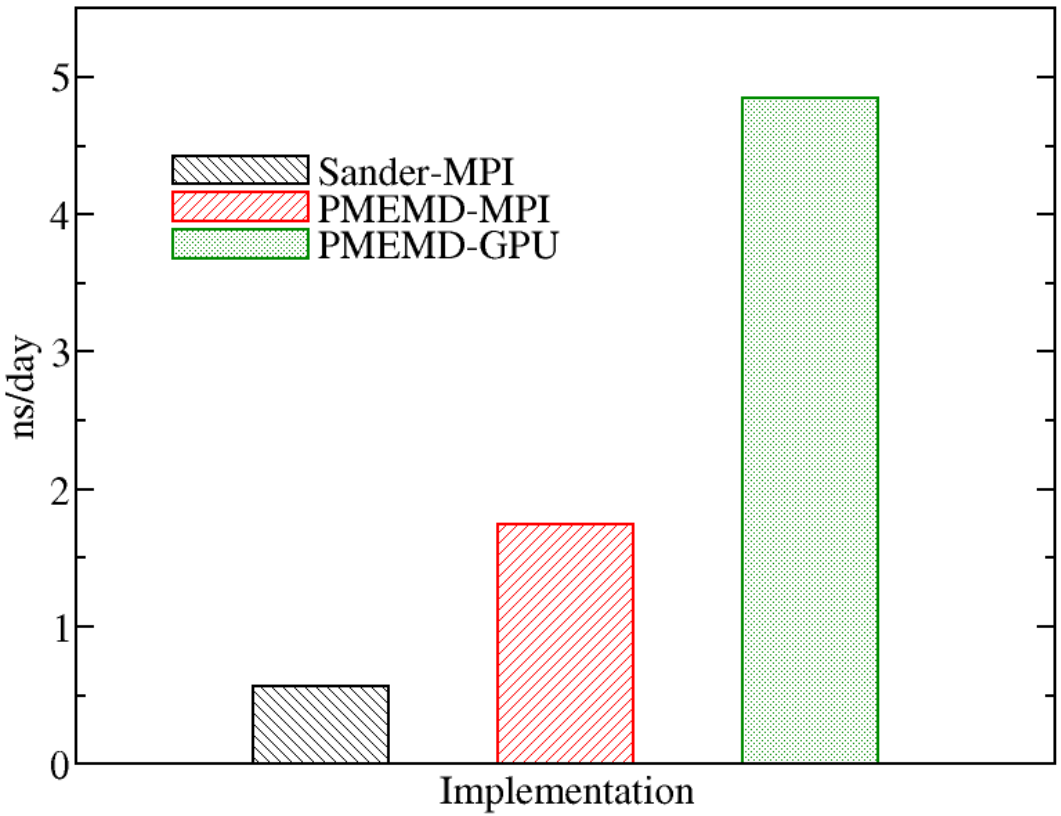
AMBER batch script (Pmemd)

```
#!/bin/bash
#SBATCH -A <Your-Project-Here>
#SBATCH -n 96
#SBATCH --ntasks-per-node=48
#SBATCH --time=01:00:00

module load icc/2017.1.132-GCC-5.4.0-2.26  impi/2017.1.132
ifort/2017.1.132-GCC-5.4.0-2.26
module load Amber/16-AmberTools-16-patchlevel-20-7

srun pmemd.MPI -O -i 02_heat.in -o 02_heat.out -p \
ala_tri.prmtop -c 01_min.rst -r 02_heat.rst -x 02_heat.nc
```

AMBER batch script (single GPU engine)

```
#!/bin/bash
#SBATCH -J Amber
#SBATCH -A <Your-Project-Here>
#SBATCH -n 1
#SBATCH --gres=gpu:k80:1
#SBATCH --time=1:00:00

module load icc/2017.1.132-GCC-5.4.0-2.26 ifort/2017.1.132-
module load CUDA/8.0.44
module load Amber/16-AmberTools-16-patchlevel-20-7-hpc2n
pmemd.cuda -O -i mdinfile -o mdoutfile -c inpcrdfile \
-p prmtopfile -r restrtfile
```

AMBER performance 158944 atoms system



NAMD

```
$module spider namd
----------------------------------------------------------------------
  NAMD:
----------------------------------------------------------------------
    Description:
      NAMD is a parallel molecular dynamics code designed for hi

     Versions:
        NAMD/2.12-mpi
        NAMD/2.12-nompi (GPU support)
```

NAMD 6 cores

```
#!/bin/bash
#SBATCH -A SNICXXXX-Y-ZZ
#SBATCH -t 00:10:00
#SBATCH -N 1
#SBATCH -c 6

module add icc/2017.1.132-GCC-6.3.0-2.27  impi/2017.1.132
module add NAMD/2.12-nompi

namd2 +p6 +setcpuaffinity config_file > output_file
```

NAMD all cores

```
#!/bin/bash
#SBATCH -A SNICXXXX-Y-ZZ
#SBATCH -t 00:10:00
#SBATCH -N 1
#SBATCH -c 28

module add icc/2017.1.132-GCC-6.3.0-2.27 impi/2017.1.132
module add NAMD/2.12-nompi

namd2 +p28 config_file > output_file
```

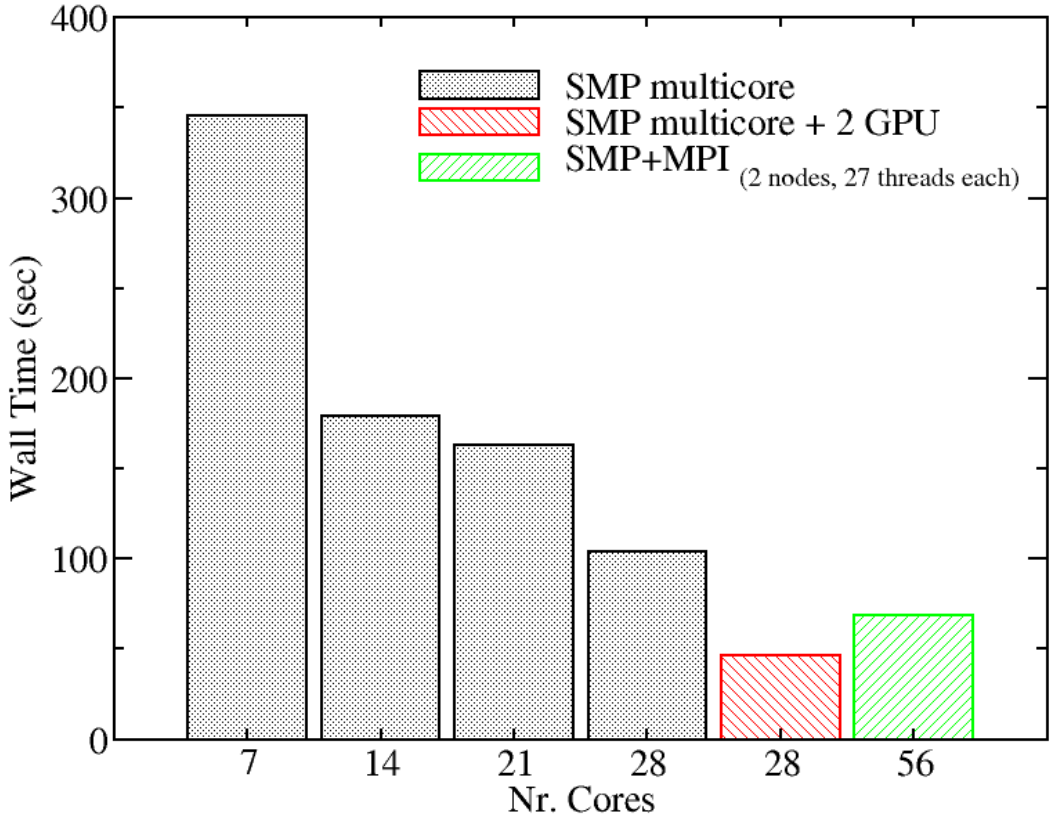
NAMD on GPUs

```
#!/bin/bash
#SBATCH -A SNICXXXX-Y-ZZ
#SBATCH -t 00:10:00
#SBATCH -N 1
#SBATCH -c 28
#SBATCH --exclusive
#Ask for 2 GPU cards
#SBATCH --gres=gpu:k80:2

module add GCC/5.4.0-2.26  CUDA/8.0.61_375.26  OpenMPI/2.0.
module add NAMD/2.12-nompi
```

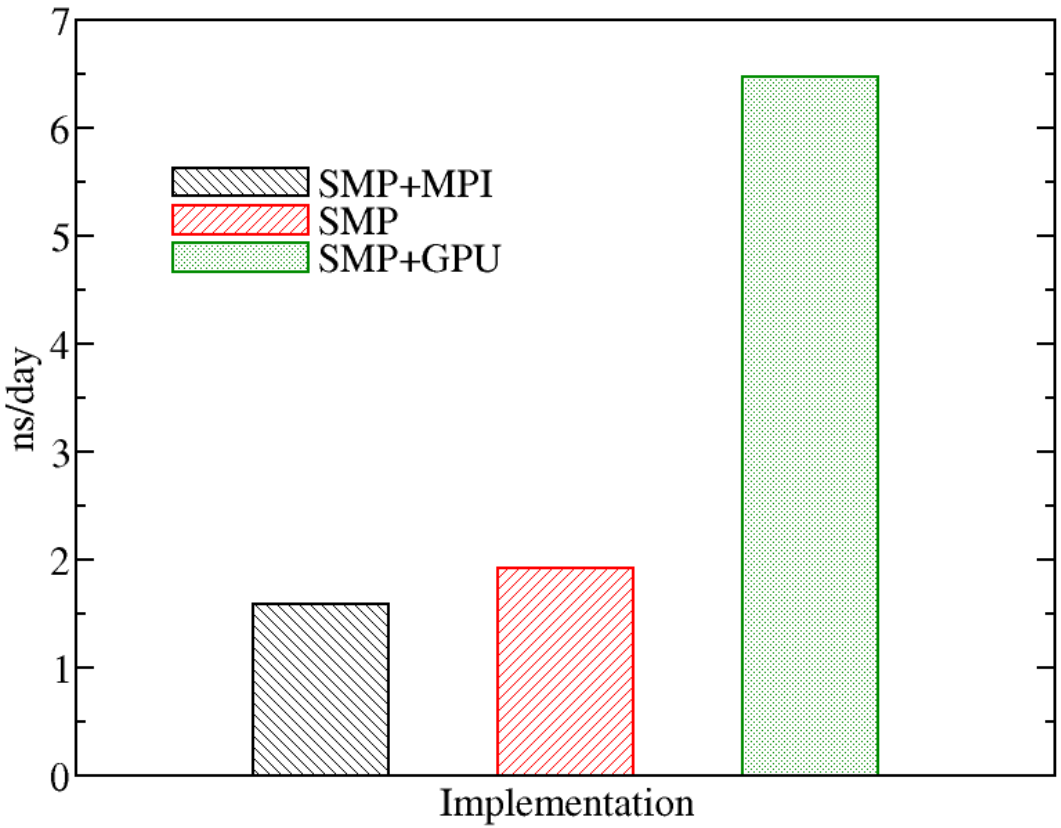
NAMD more than 1 node

```
#!/bin/bash
#SBATCH -A SNICXXXX-Y-ZZ
#SBATCH -t 00:10:00
#SBATCH -N 2
#SBATCH -n 2
#SBATCH -c 28
#SBATCH --exclusive

module add GCC/6.3.0-2.27  OpenMPI/2.0.2
module add NAMD/2.12-mpi

srun -N 2 -n 2 namd2 +ppn 27 config_file > output_file
```

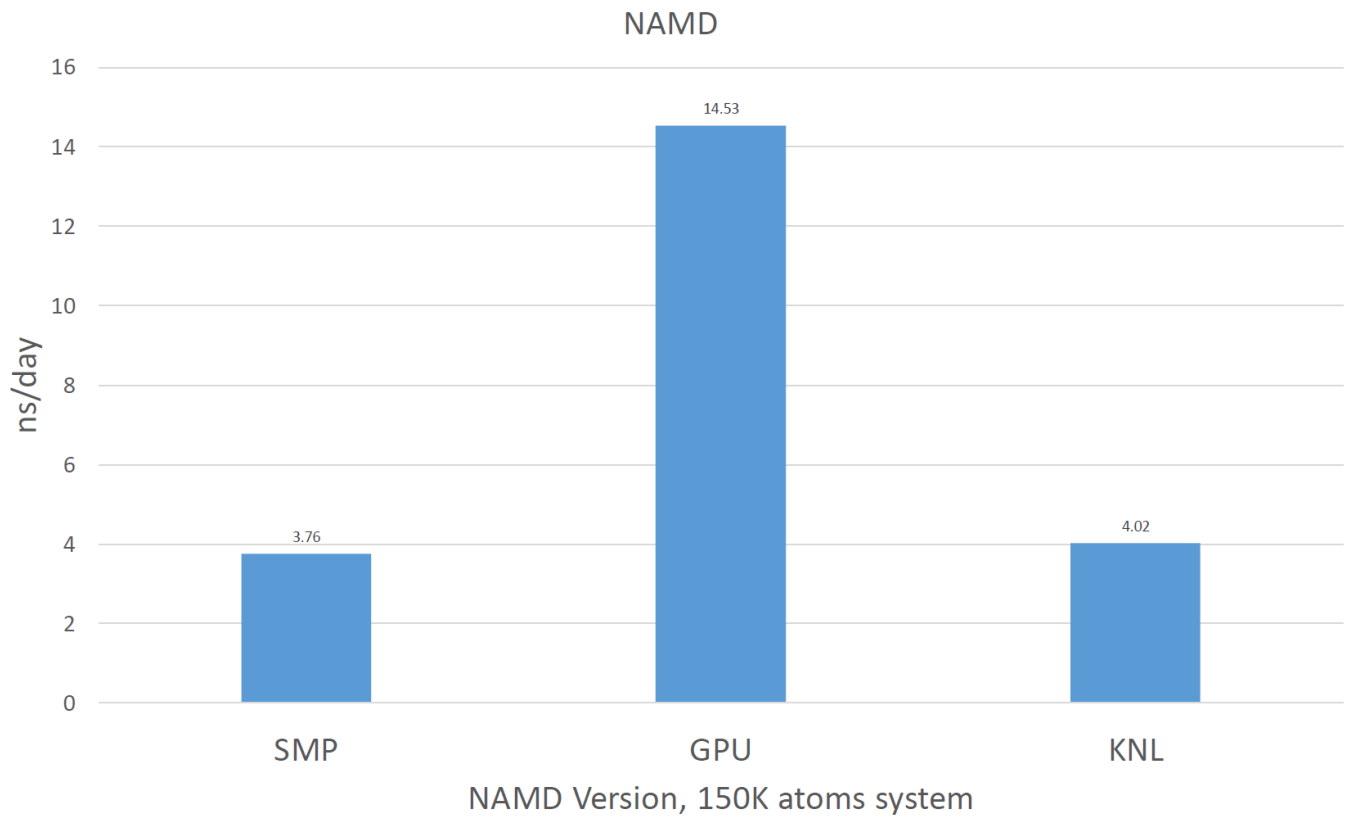
NAMD performance 23074 atoms system



NAMD performance 158944 atoms system



NAMD performance (+setcpuaffinity)



GROMACS

```
$module spider gromacs
----------------------------------------------------------
  GROMACS:
----------------------------------------------------------
    Description:
      GROMACS is a versatile package to perform molecular
      systems with hundreds to millions of particles. -

     Versions:
        GROMACS/5.1.4-hybrid
        GROMACS/5.1.4-mt
        GROMACS/2016-hybrid
        GROMACS/2016-mt
```

GROMACS

```
$module spider GROMACS/2016-hybrid
----------------------------------------------------------
  GROMACS: GROMACS/2016-hybrid
----------------------------------------------------------
 Description:
   GROMACS is a versatile package to ...
   Homepage: http://www.gromacs.org

   You will need to load all module(s) on any one of
   the lines below before the "GROMACS/2016-hybrid" ...

   GCC/5.4.0-2.26  CUDA/8.0.44  OpenMPI/2.0.1
```

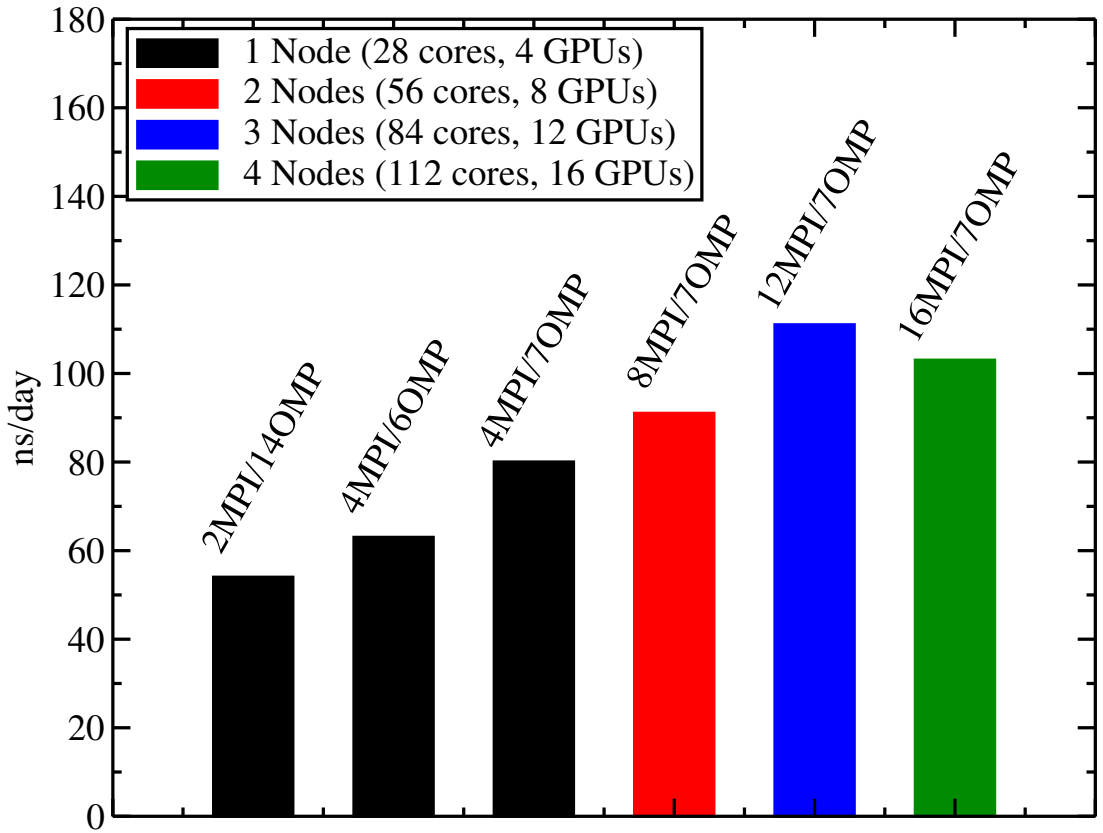
GROMACS batch script

```
#!/bin/bash
#SBATCH -A ~SNICYYYY-XX-NN
#SBATCH -t 01:00:00
#SBATCH -n 12
#SBATCH -c 7
#SBATCH --gres=gpu:k80:2
#SBATCH -p batch
module add CUDA/8.0.44   GCC/5.4.0-2.26
module add OpenMPI/2.0.1 GROMACS/2016-hybrid
mdargs="-ntomp $SLURM_CPUS_PER_TASK"
mpirun -np $SLURM_NTASKS gmx_mpi mdrun $mdargs \
-dlb yes  -v -deffnm npt
```

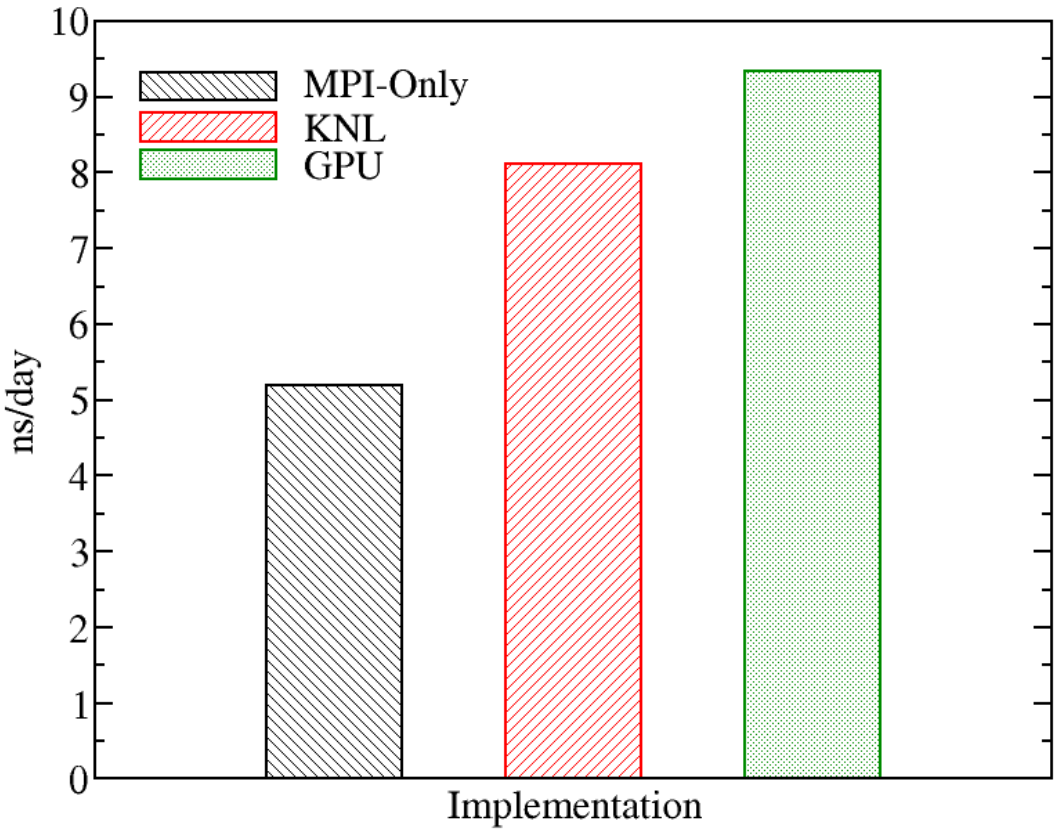
GROMACS output

```
Running on 3 nodes with total 84 cores,
84 logical cores, 12 compatible GPUs
  Cores per node:            28
  Logical cores per node:    28
  Compatible GPUs per node:  4
  All nodes have identical type(s) of GPUs
```

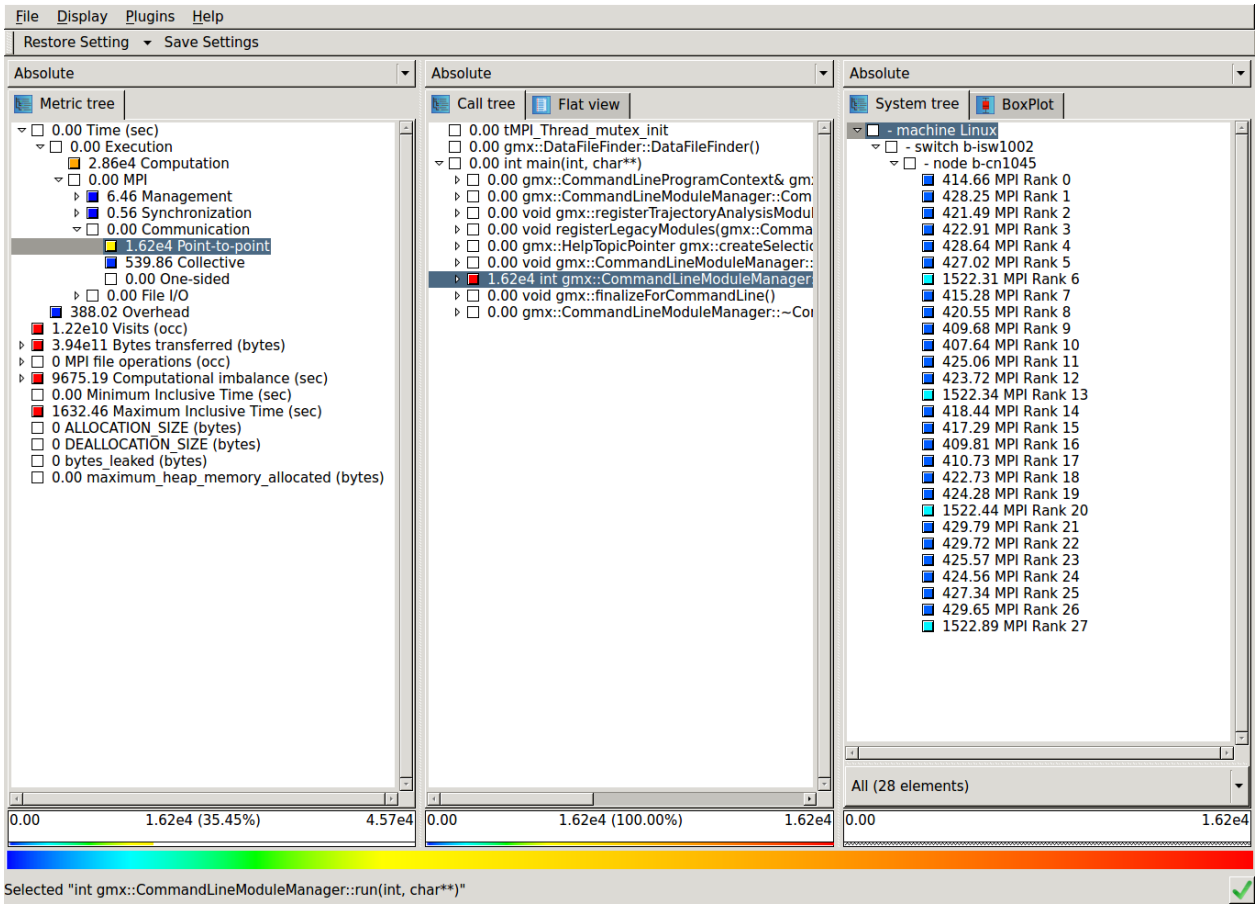
GROMACS performance 100K atoms



GROMACS performance 158944 atoms system



Profiling Tools SCALASCA/GROMACS



DOMDEC CHARMM

```
module add GCCcore/5.4.0
module add GCC/5.4.0-2.26
module add CUDA/8.0.44

./install.com gnu M fftw  domdec_gpu
```

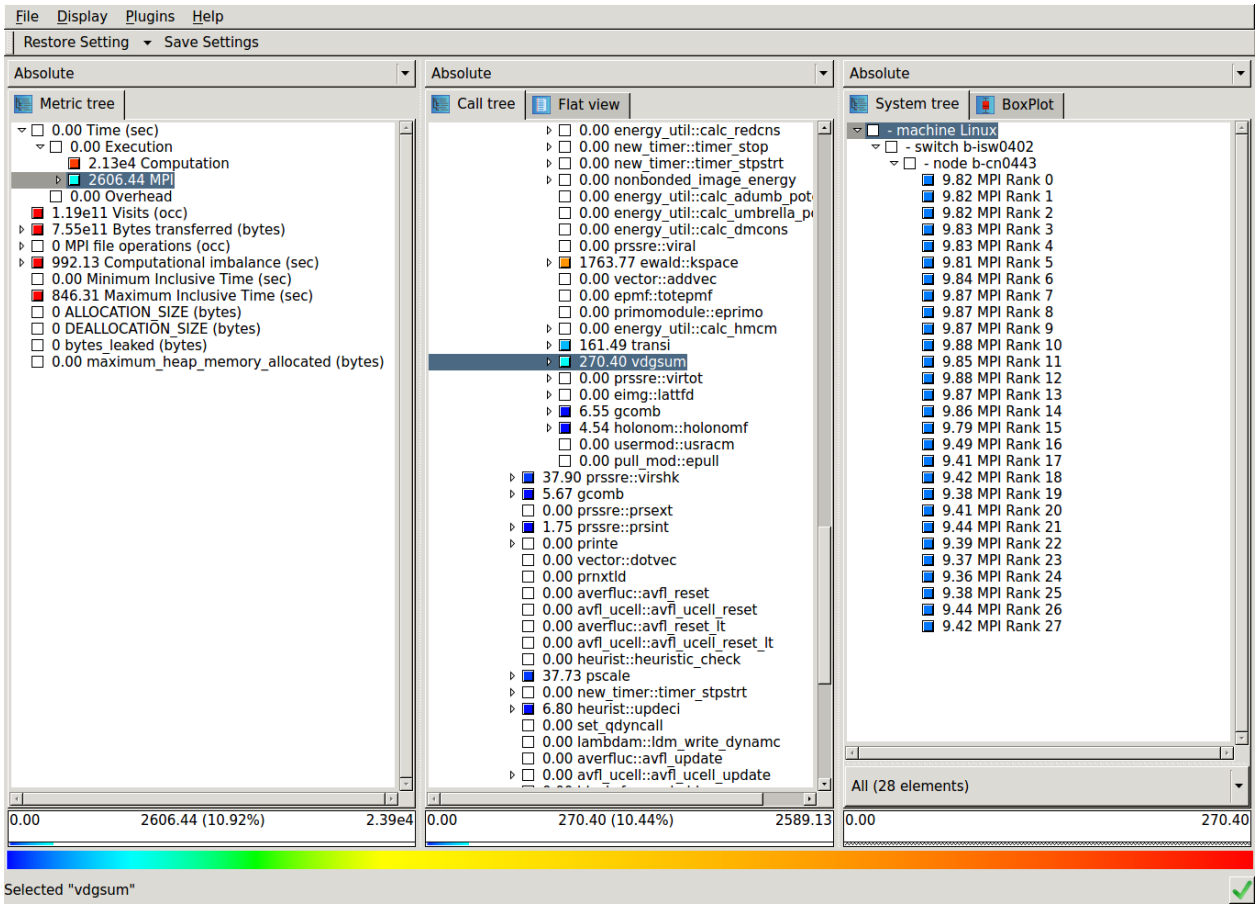
DOMDEC CHARMM

```
<domdec_dr_common> No direct/recip split, using all nodes c
Number of CUDA devices found 4
Using CUDA driver version 8000
Using CUDA runtime version 8000
Node 0 uses CUDA device 3 Tesla K80 with CUDA_ARCH 350
Intel CPU | Using CUDA version of non-bonded force loops ar
Initializing DOMDEC with NDIR =   1  1  1
Number of threads per MPI node =  28
Dynamic Load Balancing disabled
Splitting recip cores into (y by z):   1 by    1
```

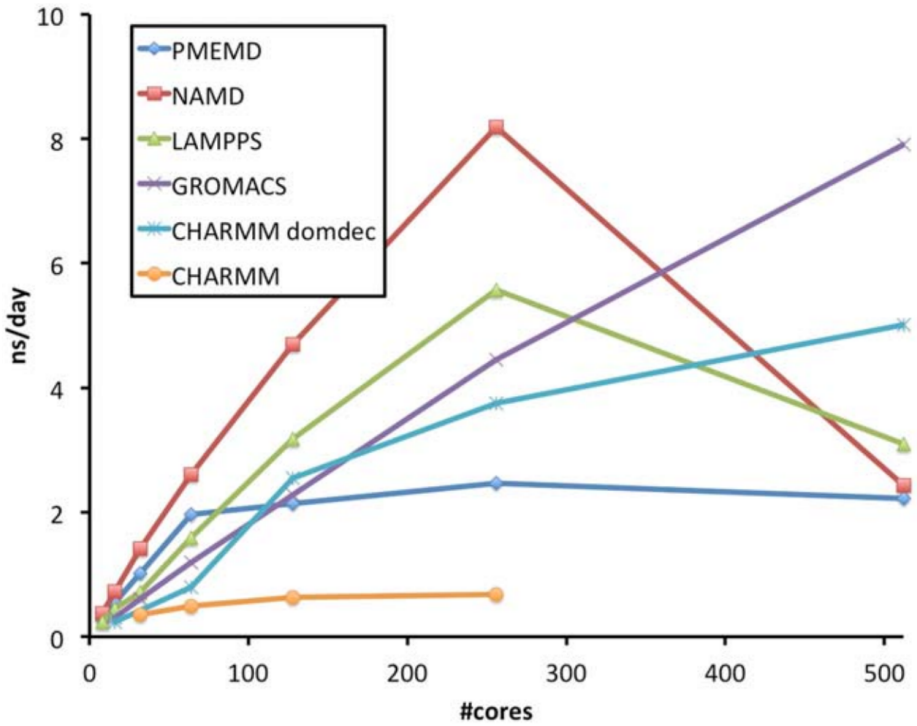
batch script DOMDEC CHARMM

```
#!/bin/bash
#SBATCH -A staff
#SBATCH -J G2016-gpu
#SBATCH -t 12:00:00
#SBATCH -n 1          ##number of mpis
#SBATCH -c 28          ##number of openmp threads
#SBATCH --gres=gpu:k80:2
#SBATCH -p batch
module add GCCcore/5.4.0 GCC/5.4.0-2.26 CUDA/8.0.44
mpirun -np $SLURM_NTASKS -x OMP_NUM_THREADS=28 \
 --bind-to none /home/p/pojedama/pfs/c42a1/ \
exec/gnu_M/charmm < m.inp > out.dat
```

Profiling Tools SCALASCA/CHARMM



Scaling behavior modern MD software



source: hEGFR with 465,404 atoms. JCC, 35, 406 (2014).

Ab-initio applications

batch script GAUSSIAN

```
#!/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 in the submit direct
g16.set-cpu+gpu-list input.com
time g16 input
```

GAUSSIAN on GPUs

Initial input file:

```
%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:

```
%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

45 atoms structure, RESP

+5 15
```

Installing your own applications

Installing your own applications

- We use EasyBuild to install software on Abisko/Kebnekaise.
- You can install your own software on your local directory using EasyBuild and make it available for later use.