

CSCI596 (Scientific Computing and Visualization) Assignment 5

Hybrid MPI+OpenMP Parallel Molecular Dynamics

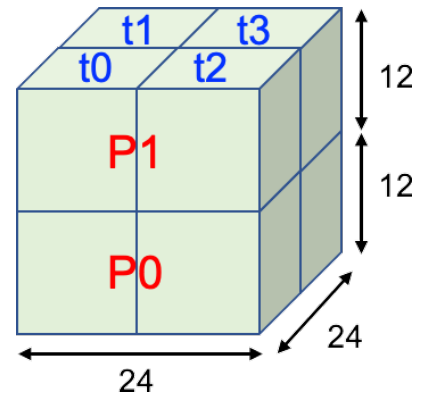
Due: October 13 (Wed), 2021

1. Write a hybrid MPI+OpenMP parallel molecular dynamics (MD) program (name it `hmd.c`), starting from the MPI parallel MD program, `pmd.c`, following the lecture note on “hybrid MPI+OpenMP parallel MD”. *Submit the source code of `hmd.c`, with your modifications from `pmd.c` clearly marked.*
2. (Verification) Run your `hmd.c` on two 4-core nodes (in total of 8 cores) with 2 MPI processes, each with 4 OpenMP threads, using the following input parameters: `InitUcell = {24,24,12}`, `Density = 0.8`, `InitTemp = 1.0`, `DeltaT = 0.005`, `StepLimit = 100`, `StepAvg = 10`. Use the following number of MPI processes and that of OpenMP threads, `vproc = {1,1,2}`, `nproc = 2`, `vthrd = {2,2,1}`, `nthrd = 4`, in the header file. Note the global number of atoms is: 4 atoms/unit cell \times (24 \times 24 \times 12 unit cells) \times 2 MPI processes = 55,296. *Submit the standard output from the run.* Make sure that the total energy is the same as that calculated by `pmd.c` using the same input parameters (shown below) at least for ~5-6 digits.

```

al = 4.103942e+01 4.103942e+01 2.051971e+01
lc = 16 16 8
rc = 2.564964e+00 2.564964e+00 2.564964e+00
nglob = 55296
0.050000 0.877345 -5.137153 -3.821136
0.100000 0.462056 -4.513097 -3.820013
0.150000 0.510836 -4.587287 -3.821033
0.200000 0.527457 -4.611958 -3.820772
0.250000 0.518668 -4.598798 -3.820796
0.300000 0.529023 -4.614343 -3.820808
0.350000 0.532890 -4.620133 -3.820798
0.400000 0.536070 -4.624899 -3.820794
0.450000 0.539725 -4.630387 -3.820799
0.500000 0.538481 -4.628514 -3.820792
CPU & COMT = 3.836388e+00 2.632065e-02

```



3. (Scalability) Run your `hmd.c` on an 8-core node with one MPI process and the number of threads varying from 1, 2, 4, to 8, with input parameters: `InitUcell = {24,24,24}`, `Density = 0.8`, `InitTemp = 1.0`, `DeltaT = 0.005`, `StepLimit = 100`, `StepAvg = 101`. *Plot the strong-scaling parallel efficiency as a function of the number of threads and submit the plot.*

(Potential Final Project)

Optimize the performance of the hybrid MPI+OpenMP MD code. For example, we could enclose the entire MD loop in a parallel clause in the main function to avoid the excessive fork-join overhead. We could also use a lock variable for synchronization.