Project Proposal

Group Member

Jack Sun, TzuKao Wang, Zhengtong Zhang

Project description

Molecular dynamics is a kind of simulation method to analyze atoms and molecules's physical movements.When given the original position and the velocity of the particles, we could use many-body Newton functions to predict the particles movement in the future and hence simulate the whole process of the system.

Since we could consider the system would become an ergodic system in the long term, the computation of each particles' movement could be processed in the parallel approach. Thus we use OpenMP as the parallel computation method to assist the simulation of molecular dynamics. We will test the parallel program and develop the performance model based on the parameter of processors, the number of particles and the process time from computation and communcation.

Technical approach

- In the coding file, it utilizes the language of C++ and the parallelization platform of Expanse.
- The underlying algorithm is the velocity Verlet algorithm.
- When calculating the potential energy and forces, the techniques of OpenMP are deployed. Details of OpenMP techniques are listed below:
    - Working sharing: For approach
    - Data environment: shared(f , nd, np, pos, vel), private(i, j, k, rij, d, d2), reduction(+: pe, ke)

Timetable

Sunday, May 1: Understanding the program
Sunday, May 15: Test the process time with different parameters
Sunday, May 29: Develop the performance model based on the result
Sunday, June 5: Organize the result and write the paper.

Final deliverable

The Performance Model with different data and the testing result

Team cooperation

Jack Sun, TzuKao Wang do the testing and Zhengtong Zhang collects the results and analyzes them. Afterwards, every member will develop the performance model simultaneously.

Reference

[1] John burkardt. (n.d.). *Md_openmp*.
https://people.sc.fsu.edu/~jburkardt/cpp_src/md_openmp

Bonus Part

Pseudo-codes of function: Compute Forces and Energies in *md_openmp.cpp*

OpenMp parallelization set up:
Shared variables: force of particles (Array), number of dimension (int), number of particles (int), position (Array), velocity (Array)
Private variables: i (dimension iterator index), j (particles iterator index), k (particles iterator index), rij, d, d2 (energy function parameters)

OpenMp feature: For loop reduction. Set two PRIVATE variables **pe** and **ke** to accumulate the PUBLIC variables **pot** and **kin**.

```
For each particle k in the simulation box:
    For each dimension i:
        Initialize the Force of kth particle in dimension i to 0.0
    For each particle j in the simulation box:
        If particle j is not particle k:
            Compute the displacement and norm between j and k.
            Assign half of the potential energy of k to particle j.
            For each dimension i:
                Update the Force of the kth particle \
                according to the potential function.
    For each dimension i:
        Compute the kinetic energy of the kth particle in dimension i.

Update the kinetic energy based on mass.

Return the potential energy and the kinetic energy and \
assign the calculation result to the corresponding pointer.
```