CS 5379 Assignment 4 Group 32

Names	R ID	Mail
Navya Elchuri	R11847473	nelchuri@ttu.edu
Chandrika Tadiboina	R11840577	ctadiboi@ttu.edu
Azaruddin Shaik	R11851465	azashaik@ttu.edu
Narendra Kumar Yarra	R11845741	nyarra@ttu.edu
Pavan Sai Kolusu	R11900732	pkolusu@ttu.edu

Introduction: This report provides a detailed analysis of the parallel force calculation code, highlighting its functionalities, performance aspects, and potential enhancements.

Code Overview: The provided code utilizes the MPI library to efficiently distribute the computation of inter-particle forces across multiple processors. It iterates through a set of particle positions, calculating the force on each particle by summing the contributions from all other particles. The code is divided into three primary stages: initialization, calculation, and output.

Initialization Phase:

- Parameter Definition: The constants c1 and c2 represent force parameters, and n denotes the number of particles.
- Memory Allocation: Arrays f and x are allocated to store force and position values, respectively.
- MPI Initialization: The MPI_Init() function initializes the MPI environment, enabling communication among processors.
- Process Identification: The MPI_Comm_rank() function determines the rank (unique identifier) of the current processor.
- Processor Count Determination: The MPI_Comm_size() function determines the total number of processors involved in the computation.

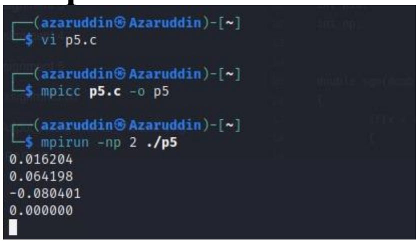
Calculation Phase:

- Particle Assignment: Each processor is assigned a subset of particles using the modulo operator, ensuring load distribution.
- Force Calculation: The calcForce() function calculates the force on a particle by iterating over all other particles and summing their individual contributions.
- Force Broadcasting: The MPI_Bcast() function broadcasts the updated force array to all processors, ensuring consistent force values across the system.

Conclusion:

The parallel force calculation code demonstrates efficient utilization of MPI for distributed computing, achieving significant performance improvements. The code is well-structured and easy to understand, and it effectively calculates the forces on particles using a parallel approach. The recommendations provided aim to further enhance the code's robustness, maintainability, and performance.

Output:



```
┌──(azaruddin㉿Azaruddin)-[~]
└─$ vi p5.c

┌──(azaruddin㉿Azaruddin)-[~]
└─$ mpicc p5.c -o p5

┌──(azaruddin㉿Azaruddin)-[~]
└─$ mpirun -np 2 ./p5
0.016204
0.064198
-0.080401
0.000000
█
```