Title: Parallelization of Tree-Particle Mesh Algorithm for N-Body Galaxy Simulations

Alex Nguyen (alexandn)
Kaiwen Geng (kgeng)

Summary:

We plan on creating an optimized parallel implementation of the Tree-Particle Mesh Algorithm for N-body galaxy simulations and analyzing the performance speedup of our implementation against the performance of the original implementation. By using an algorithm which combines the use of trees (from Barnes-Hut algorithm) and meshes (from Particle Mesh algorithm) in the galaxy simulation calculations, we expect to see greater computational efficiency when compared to the Barnes-Hut algorithm which solely uses trees and Particle Mesh algorithm which solely uses meshes.

Background:

In a Barnes-Hut gravitational N-body simulation, the particles are distributed into an octree data structure, where the force calculations are then performed according to the tree hierarchy. This method is particularly useful for calculating forces between particles whose distances between each other are relatively short, and slightly less effective for calculating forces between particles (or groups of particles) whose distances are relatively far, since it must approximate the farther particle groups as a single particle rather than accurately simulating the group of particles as it exists in space. Since this algorithm breaks down the large set of particles into smaller subsets of particles for force calculation, there is a clear path to parallelization of the algorithm such that the calculation performance can improve significantly with the use of multicore CPUs.

In a Particle Mesh gravitational N-body simulation, the long-distance particles are instead distributed into a mesh data structure. We can then use Poisson's equation on the mesh to obtain gravitational potential and apply them to calculate long-distance forces on particles. This method is more efficient for calculating the forces between long-distance particles, when compared to the Barnes-Hut algorithm, and is not as accurate for particles that are closer together due to the use

of a grid which causes inaccuracies for calculating forces between particles closer together than the dimensions of a single grid unit.

To take advantage of the pros of each algorithm, we can combine these algorithms into a Tree-Particle Mesh algorithm which can utilize the Barnes-Hut algorithm for calculating forces between close-knit particles in 3D space and the Particle Mesh algorithm for calculating forces between long-distance particles in 3D space. There are many different implementations for Tree-Particle Mesh algorithms, especially with how the force calculations are done between the trees from the Barnes-Hut algorithm and the mesh implemented in the Particle Mesh algorithm.

The Challenge:

There are several different ways of implementing a TPM algorithm for cosmological simulations, and our goal is to find a parallel implementation with the highest computational speedup when combining the parallelized versions of both algorithms. Since both algorithms work by splitting the particle calculations by ranges of distribution (by distance in 3D space) we can parallelize each algorithm several ways across each data structure. While task distribution amongst threads will most likely be done through recursive bisection for Barnes-Hut calculations and through spatial decomposition into a volume grid for Particle Mesh calculations, our challenge will be implementing parallelization on the force calculations on interactions between the short-ranged particle groups and long-ranged particle groups.

In each individual algorithm there will be plenty of opportunities for optimization of communication between threads performing calculations for separate groups of particles, since the interactions between groups of neighboring particles must be accounted for in each calculation for both closely grouped particles and long ranged interactions. Additionally, we can explore different ways to parallelize tree creation and traversal for particle interactions calculated using Barnes-Hut as well as mesh parallelization for particle interactions calculated using the Particle Mesh algorithm. Finally, a key component of optimizing our implementation will consist of dynamic work distribution between both algorithms and determining the best way to split particle calculations between both algorithms as well as finding ways to reduce any costs incurred by communication latency.

Resources:

We will be using starter code from previous implementations of the Barnes-Hut algorithm and Particle Mesh algorithm. Additionally, we will be using C++ likely with OpenMPI. We intend to test the scalability of our implementation on the Gates Cluster Machines and potentially the PSC Bridges-2.

Goals and Deliverables:

Our main goal for this project is to increase speedup for cosmological simulations consisting of very large numbers of particles and optimize the computation efficiency of the combination of Barnes-Hut algorithm and Tree-Particle Mesh algorithm, especially when compared to the use of only one algorithm or the other.

We aim to benchmark the performance of each algorithm individually and compare it to benchmarks that we are able to achieve in our combined TPM implementation. We hope to achieve significant speedup between our parallelized TPM implementation and the parallelized Barnes-Hut and Particle Mesh implementations. We also would like to be able to test the scalability of our implementation against large simulations containing very large numbers of particles in 3D space, with a large number of small-range and long-range interactions.

For our final demonstration, we plan on displaying the speedup we are able to achieve from our implementation against the original computation times from the original two implementations for differing cosmological simulations. A stretch goal would be to also be able to include a graphical representation of the cosmological simulations produced by our implementations compared with those produced by the initial implementations. We may also seek to display and compare the differences in bottlenecks between each implementation in our final demo.

Platform Choice:

We plan on using C++ with OpenMPI, running on multicore CPUs.

Schedule:

Week 1: Setup initial testing and benchmarking for Barnes-Hut, Particle Mesh implementations of cosmological simulation

Week 2: Begin implementation of combined Tree-Particle Mesh algorithm using OpenMPI, focusing on load balancing between each algorithm (separating particle calculations by distance)

Week 3: Write milestone report, work on dynamic load balancing for particles within each individual algorithm, start work on message passing between threads for boundary calculations within each algorithm

Week 4: Continue work on communication optimization between threads and test for calculation accuracy for overall combined algorithm

Week 5: Test scalability of implementation, continue work on overall implementation with combined algorithms, reduce any bottlenecks resulting from timing conflicts and communication latencies

Week 6: Final performance testing, prepare final demo, and complete final report

Sources:

Tomoaki Ishiyama, Toshiyuki Fukushige, Junichiro Makino, GreeM: Massively Parallel TreePM Code for Large Cosmological *N*-body Simulations, *Publications of the Astronomical Society of Japan*, Volume 61, Issue 6, 25 December 2009, Pages 1319–1330, https://doi.org/10.1093/pasj/61.6.1319

Bagla, Jasjeet. (2002). TreePM: A code for Cosmological N-Body Simulations. Journal of Astrophysics and Astronomy. 23. 185-196. 10.1007/BF02702282.