Parallel Approaches to the N-Bodies Problem

Ben Bole bolebj@dukes.jmu.edu
John Latino latinojr@dukes.jmu.edu
Richard Bimmer bimmerra@dukes.jmu.edu
Kevin Kelly kelly4kc@dukes.jmu.edu

Introduction

The N-bodies problem is a classical physics problem in which the goal is to predict the motion of N celestial bodies, given starting conditions. These starting conditions include the (x, y, z) coordinates, starting velocities, and masses of each of the N celestial bodies. When $N \leq 2$, this problem is trivial, however, when $N \geq 3$, prediction of the motions of bodies becomes much more computationally intensive. There are multiple different methods for efficiently paralellizing the computation of gravitational forces between celestial bodies.

In this paper, we investigate three techniques for approaching the N-bodies problem: the Barnes-Hut Simulation, the Parker-Sochacki Method, and the Particle-Particle-Particle Mesh (P3M) Method. We will be modifying already existing code bases of each respective approach, so that the input and output of each respective program is in the same form. We will be analyzing run-time and scalability of each approach in order to determine which approach would be the best to iterate upon.

The Barnes-Hut Approach

The Barnes-Hut Simulation (BHS) seeks treat the 3-dimensional space as a tree, specifically an octree. In an octree, each internal node of an octree stores exactly eight children. Usually used in the realm of 3-dimensional graphics/simulations, the eight children of each internal node represents the octants of the original 3D space of the parent node. In BHS, each internal node stores a reference to its center of mass, as well as its eight children - internal nodes do not care about individual bodies.

On each time step, the following steps are performed. Space is recursively divided up until each leaf node either contains one or fewer bodies within its bounds. Net force on each body is calculated. This is done by traversing the tree, starting at the starting node. It is then determined if the center of mass for that internal node is significantly close to the body. If the center of mass is not significantly close, then calculations are performed using the center of mass of the internal nodes as a reference. If the center of mass is significantly close, that node's children are traversed; if the node is a leaf node, calculations on the bodies themselves are performed. This process is performed until the specified number of time steps are performed.

Because of the intensive computation required in dividing up space on each time step, BHS is normally used to simulate an extremely large space in which bodies are very far from each other. This makes it so that the algorithm can frequently "skip" calculations on individual bodies, in noting the significance of the distance between the body and internal nodes' "center of mass".

TODO: ANALYSIS AND SCALING OF BHS

The Parker-Sochacki Approach

The Parker Sochacki Method (PSM) seeks to manage the interactions between celestial bodies by manipulating ordinary differential equations (ODEs). Parker and Sochacki proved several years ago that it is possible to manipulate any arbitrary ODE into a system of polynomial ODEs. Having a system of polynomial ODEs then allows us to shift those ODEs into a metric space, allowing for much easier and faster mathematical manipulation.

Since we constructed our metric space using polynomial ODEs - all of which were designed from our initial ODE - we know that our metric space is a Cauchy sequence, making it a complete metric space. Since our constructed metric space is complete, by the Banach-Caccioppoli fixed point theorem, we know that each of our ODEs, once plugged into an iterated function f, will converge onto a fixed point, x - different for each ODE. The fixed point x is the solution to our originial ODE.

This is an extremely convenient method for approaching the N-bodies problem, since the most computationally intensive action is the performance of the iterated function on the metric space. Since we only require the initial ODE, this process of iteration is highly parallelizable.

TODO: ANALYSIS AND SCALING OF PSM

The P3M Approach

The Particle-Particle-Particle Mesh (P3M) Method is a unique approach because it was originally designed to address the n-bodies problem not at a celestial scale, but at a molecular one. The P3M approach is built on top of a now obsolete method called Particle Mesh.

Particle Mesh took in the positions of all bodies given to it and then interpolated those positions onto a grid. Once all positions were populated by the bodies, the forces acting on each point in the grid would be calculated using Poisson's equation. This method could, within a realm of error, accurately simulate N-bodies which did not share the same grid position, however the same could not be said for particles which did share a grid position.

In order to alleviate this issue, P3M requires calculation of the forces of each individual body which shares a spot on the grid. Like BHS, P3M performs faster when bodies are farther apart from each other, due to the individual calculations which must be performed on each body sharing a grid location.

TODO: ANALYSIS AND SCALING OF P3M

Comparison

TODO: COMPARISONS OF EACH APPROACH

Discussion (Mid-Project Only)

The above is how we believe we will format our final report. So far, we have researched many different algorithms for solving the N-Bodies problem, and have settled on 3: the Barnes-Hut Algorithm, the Parker-Sochacki Method, and the Particle-Particle-Particle Mesh (P3M) Method. We have studied these algorithms and included a short description of each above, which will be included in our final report. All the code for these three algorithms is on GitHub at https://github.com/jlat96/CS470_NBody_2019.

We have hit a few roadblocks so far, namely that the code for the P3M method is written in Fortran, so we have little understanding of it and have so far been unable to compile it. We have found a webpage(which is linked in the readme of P3M on GitHub) that seems to include some useful information for compiling and using the code, so we should be able to get it up and running soon. We have also had issues with the format of our input and output files, since each algorithm was implemented by different groups of people. For P3M we have decided to make a python script that will convert our input files for the Parker-Sochacki Method code into the format that the P3M code uses, so we do not have to touch the massive code base that implements the P3M algorithm. Another issue we have had is finding a program to visualize our output. We are talking to Dr. Bernstein to see if he has something we can use, but are still waiting on a response. Finally, porting the Matlab code to C proved to be more difficult than was originally anticipated. An example of a key difficuly in converting this code is that array types in Matlab start at an index of '1', while they start at '0' in C. Converting the algorithm to account for this caused issues that will be resolved before the final deliverable.

Our project poster will include a description and comparison of each of the algorithms, as well as the runtime information of each on the same input data.

We believe we are on track to finish on the final due date. Our next steps are getting the P3M code to compile, finishing the input/output conversion scripts, creating useful input files for our analysis, and finding a way to visualize the output.