Newtonian Dynamics of Particle on a Cluster

Mark Randles
CS490 Independent Project
Dr. Hassan Rajaei
2004-08-05

Abstract.

The purpose of this project was to study both the simulation and resolution of Newtonian particle dynamics on a cluster. Topics to be covered were to be partitioning of the data, interprocess communication, and hybrid algorithms using multi-tier data partitioning.

Introduction.

Newtonian dynamics are the most standard set of dynamics in physics. They are the basis for nearly all motion that we see upon a daily basis. The goal of this project was to use the problem of finding positional data for massive particles, by applying the laws of Newtonian dynamics upon them. Each particle is considered massive, in that it has a fixed mass, along with dimensions and other properties of physical objects.

The theory of data partitioning upon the problem set was one that is quite simple, but can become quite complex. As we shall see, there are may ways to accomplish the same task, not all ways created equal. In addition to the data partitioning, there is the topic of using hybrid code, code such that each compute node in the cluster divides the problem further.

Part I will consist of the problems and ideas associated with the data partitioning of the given problem, and some solutions to it's effect. Part II will discuss the ways hybrid design could further refine and benefit the problems solution.

Part I.

There are my ways to partition the data for this problem. The most obvious is to split the vectors holding the particle up along some sort of arbitrarily partition lines, determined by the number of processors available and other run-time variables. This method is both the most straight forward, and is reasonably fast, as each process does about the same about of work. However, if the full suite of Newtonian

physics is applied, particles will interact with one another, which will complicate and slow down the use of a straight vector division partition.

However the interaction between the particles is not entirely random. Since the motion of a particle will only change with the influence of a outside force, such as a collision with another particle, we can calculate the probable ending position of the particle after a time period T. If any of the paths, from start point to end point, intersect, then it can nearly be assumed that the particles will collide.

Since we now have two sets of particles, those that will collide and those that will not, we can assume that the work for each is independent and can allow for different parts of our cluster to work on each. The particles that will not collide with any others during time T can be handled by the previous discussed vector division partition scheme. The particles that will collide should be handled in a more serial manner, as the collision of the particles is not a very easy to parallelize problem. The situation would exist where only select groups of particles will interact, and the list could be broken down into the smaller sections of interacting particles. This, however, would mean more computation time would need be devoted to the processing of the shortened list of particles.

The cost of this subcomputation and intersection algorithm is in design less then the cost of brute-forcing the entire list. Much of the time saved by the algorithm would be in the reduction of communication between nodes and recovery of the time otherwise spent seeing if non-intersecting particles will collide. However, the algorithm can only speed up by a fraction of the entire computation time of the problem, as we can only reduce the total work done in the math heavy and computation time heavy parts of the code. Only if that time saved is greater then the time spent doing the initial precalculation will we save time.

Part II.

The possible uses of multi-level or hybrid programming in this problem are numerous. Both approaches of data partitioning provide places where a finer, local partitioning is helpful.

The very basic partitioning of the data will yield a noticeable speedup if the computation loop is further broken down by dividing up the work vector, assigning a smaller part to each of the processors available on our test nodes. This provides a further layer of vector division at a local level, which both reduces the number of MPI nodes and the

communication therein, and speeds up the wall time needed for the problem to execute. However this approach is limited by the amount of data provided to each compute node before the second vector division. If the data block is small, then the resulting computation time will approach the time needed to initialize the second vector division, which will affect the speedup.

The second partitioning method discussed would benefit from multi-level programming in much the same way, dividing the data further after course chunks of data were distributed to each compute node. However, using the capabilities of the nodes to partition the work needed in the precalculation section would be the biggest benefit of using multi-level programming. By using the shared memory parallelization available on each node, one can speed up the precalculation, since little or no network communication would need to happen for the computation of the intersections. Also by harnessing the speedup given by the multi-level programming, an algorithm could be developed to do the precompute step once for each globalization step, thus providing a dynamic partitioning of the data depending on the current position of all the particles. However this tactic would be dependent upon the time needed to do the next globalization step vs. the time needed to do the precompute step. If the precompute step is the longer operation, then we would slow down the resulting cycle, and therefore not gain any speedup using that approach.

Conclusion.

As presented above, there are a variety of solutions to computing Newtonian particle simulations on a cluster. However, for each different environment a different solution might prove better then another. As with anything it is a hit or miss proposition, and each problem should be approached independently.