N-body Interaction

Rahul Patel :- 201501017
Jeel Prajapati :- 201501213

➢Problem :

- The N-body simulation numerically approximates the evolution of a system of bodies in which each body continuously interacts with every other body.
- Can extend the same idea to simulate other types of interactions.
- e.g.,protein folding is studied using N-body simulation to calculate electrostatic and van der Waals forces. Turbulent fluid flow simulation and global illumination computation in computer graphics are other examples of problems that use N-body simulation.

➢Naive Algorithm

- The all-pairs approach to N-body simulation is a brute-force technique that evaluates all pairwise interactions among the N bodies.
- It is a relatively simple method, but one that is not generally used on its own in the simulation of large systems because of its O(N^2) computational complexity.
- Input : Number of bodies in system, coordinates of body
- Output : Force on each body

➢Barnes Hutt approximation

- The key idea is to approximate long-range forces by replacing a group of distant points with their center of mass. In exchange for a small amount of error, this scheme significantly speeds up calculation, with complexity *n log n* rather than n^2.
- We use a quadtree data structure, which recursively subdivides regions of space into four equal-sized quadrants.
- The Barnes-Hut approximation involves 3 steps(kernels):

    1. Compute bounding box around all bodies (root node)

    2. Divide space into hierarchical quadrants and calculate total mass and center of mass of each quadrant.

    3. Compute forces acting on each body with help of quad

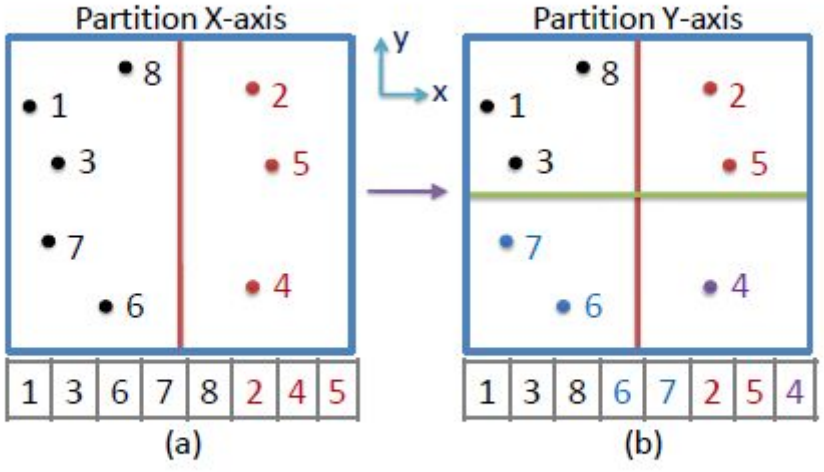


Figure 10. Spatial Clustering of Points

- Here we use top down approach:

  1). Initialize the root node of the quadree as containing all

  points. Set the bounds defining cluster of points

  belonging to the root as 0 and n-1.

  2). Now loop on current level:

  (a) Allocate threads equal to the number of partitions in current level.(Num Threads = 1 initially for the root and then increases as we iterate)

  (b) For every thread, in parallel, do

  (i) STOP the thread if the current partition is a leaf.

  (ii) ELSE, create 4 new partitions and 4 new quadtree nodes. Record the respective partition bounds in the nodes created. To create 4 new partitions, we first divide the current partition along one of the axis (can be any of y or x) and swap the points belonging to one side of the partition with another. We then repeat the same process and divide the 2 new partitions along the remaining axis.

  (c) STOP looping when thread hit the maximum level of tree.

- The net force on a particular body, the nodes of the tree are traversed, starting from the root. If the center of mass of an internal node is sufficiently far from the body, the bodies contained in that part of the tree are treated as a single particle whose position and mass is respectively the center of mass and total mass of the internal node. If the internal node is sufficiently close to the body, the process is repeated for each of its children.
- Whether a node is or isn't sufficiently far away from a body, depends on the quotient r (s / d), where *s* is the width of the region represented by the internal node, and *d* is the distance between the body and the node’s center of mass.
- The node is sufficiently far away when this ratio is smaller than a threshold value *θ*.

- The parameter θ determines the accuracy of the simulation; larger values of θ increase the speed of the simulation but decreases its accuracy.
- If $\theta = 0$, no internal node is treated as a single body and the algorithm degenerates to a direct-sum algorithm.
- It has Average time complexity of nlogn.

➢Serial Implementation

- Input : Number of bodies in system, coordinates of body, approximation constant θ
- Output : Approximate force on each body

- Step 1 : Calculate boundaries of root node by calculating minimum and maximum of X and Y coordinates.
- Step 2 : Generate quadtree a discussed above except instead of assigning thread to each node of current level we would traverse each node using loop.
- Step 3 : Use Barnes Hutt approximation to calculate forces on each body as shown.

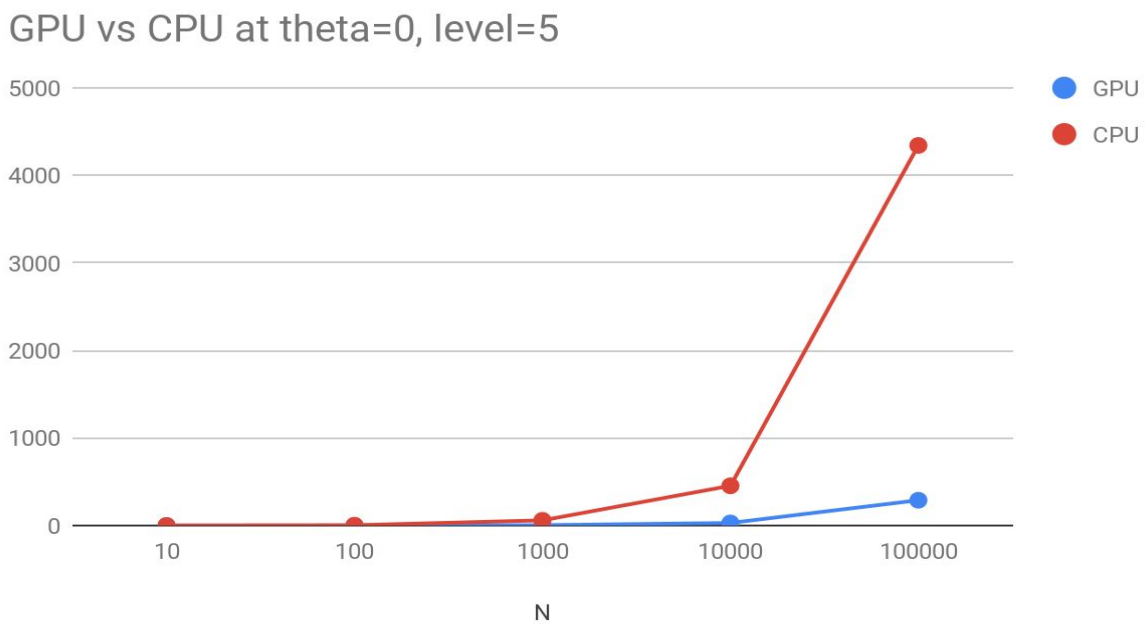
➢Parallel Implementation

- Input : Number of bodies in system, coordinates of body, approximation constant θ
- Output : Approximate force on each body

- Step(kernel) 1 : Parallel reduction strategy is used to find minimum and maximum X and Y coordinates. Implementation of this kernel takes place in two steps. In step 1, each thread block calculates its own local minimum and maximum X and Y coordinates. In step 2, global minimum and maximum X and Y coordinates are calculated.
- Step(kernel) 2 : Generate quad tree level by level. For each level, for each node in that level assign a thread to generate its children.

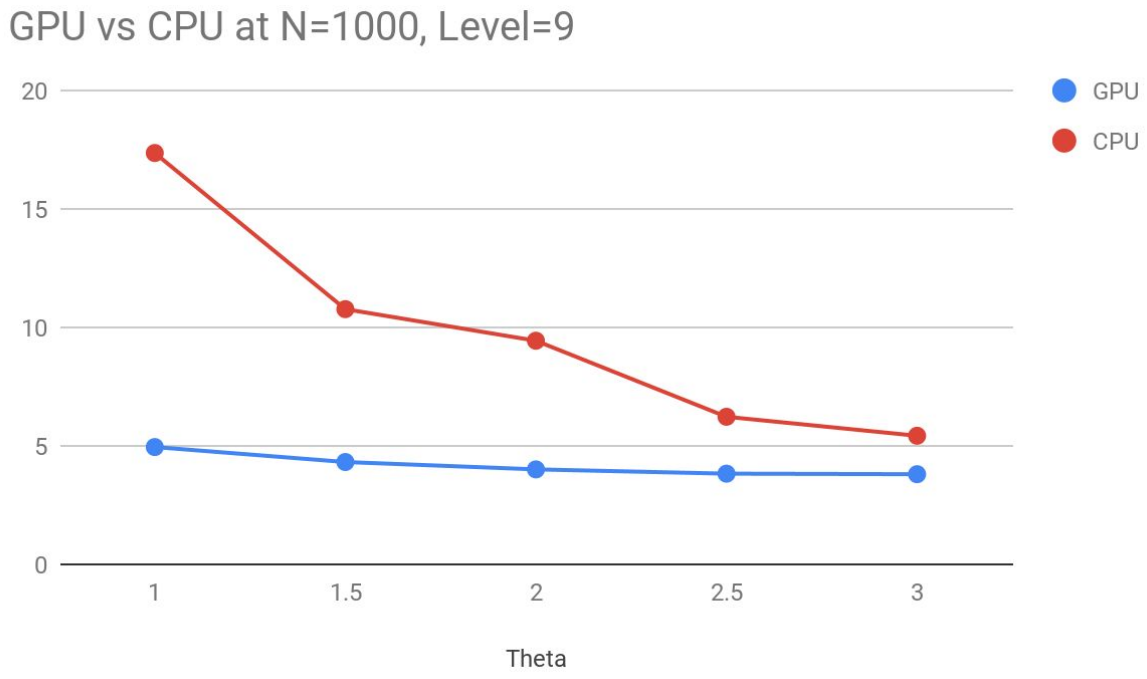
- Step(kernel) 3 : For each body, assign a thread to calculate force on a particular body by Barnes Hutt approximation using above shown method.

➢Performance Analysis



- As number of bodies increase, GPU can make use of parallelization more with more number of threads each working parallely to calculate force on each body(parallelization of kernel 3).

- As number of level increases, number of nodes in each level increases. As a result GPU can use more number of threads in parallel to construct quadtree (parallelization of step 2).



- As theta increase, depth to which we have to traverse the tree decreases. As a result, number of nodes to be traversed decreases. So, it takes less time to calculate force on each body. So, even though GPU will be faster than CPU difference in execution time decreases.

➢Limitations

- As maximum depth of quadtree is fixed before execution of the algorithm, it may happen that effect of very close objects gets miscalculated.

➢Future scope of Improvement

- As amount of GPU resources (number of maximum threads and device memory) increases, we can make quadtree with more depths thus increasing accuracy of forces calculated on bodies

- Reference

  - http://iss.ices.utexas.edu/Publications/Papers/burtscher11.pdf
  - https://cse.buffalo.edu/faculty/miller/Courses/CSE633/Suraj-Balchand-F2010.pdf
  - https://www.cse.iitb.ac.in/~rhushabh/publications/octree