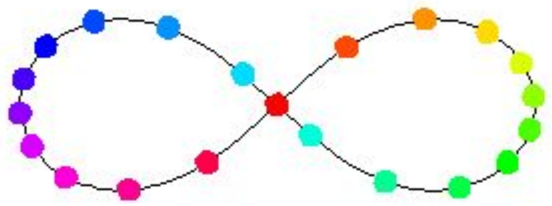
N-Body Simulation

Data Struct & Algs Course Project

Weijia Sun
Zhongze Tang
Zichen Zhu

Background

In physics and astronomy, an N-body simulation is a simulation of a dynamical system of particles, usually under the influence of physical forces, such as gravity. Solving this problem has been motivated by the desire to understand the motions of the Sun, Moon, planets and the visible stars.



Algorithms Deployed 1 - Brute Force

N-Body Simulation is an event-driven simulation. Our project is mainly based on three physics principles:

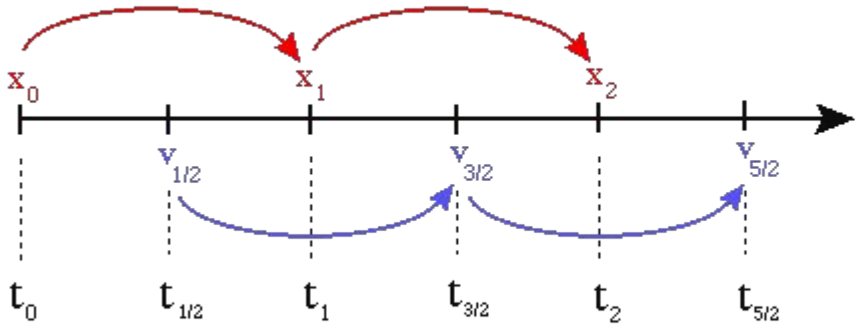
$$\vec{F_i} = -\sum_{j\neq i} \frac{G m_i m_j (\vec{r_i} - \vec{r_j})}{\left(|\vec{r_i} - \vec{r_j}|^2 + \epsilon^2\right)^{3/2}}$$

- **Pairwise force**: Newton's law of universal gravitation
- **Net force**: The principle of superposition says that the net force acting on a particle in the x- or y-direction is the sum of the pairwise forces acting on the particle in that direction
- **Acceleration**: Newton's second law of motion postulates that the accelerations in the x- and y-directions are given by: ax = Fx / m, ay = Fy / m

A naive solution is to use the all-pairs approach to N-Body simulation, which is a brute-force method that go through all pair-wise interactions among N bodies.

Algorithms Deployed 2 - Leapfrog Integration

- Using the leapfrog integration to numerically integrate the above equations
- Discretize time, and update the time variable t in increments of the time quantum Δt.
- The simulation can be more accurate as the Δt get smaller, but this comes with the price of computation.



Algorithms Deployed 3 - Barnes-Hut Algorithm

- Time complexity is O(NlogN)
- Greatly reduce the price of computation with only a little loss of accuracy.
- Group nearby bodies and treat them as a single body to speed up the brute force algorithm. If each group is sufficiently far away from each other, we can approximate its gravitational effects by using its center of mass.
- Use s/d compare to preset θ to determine whether a node is sufficiently far away

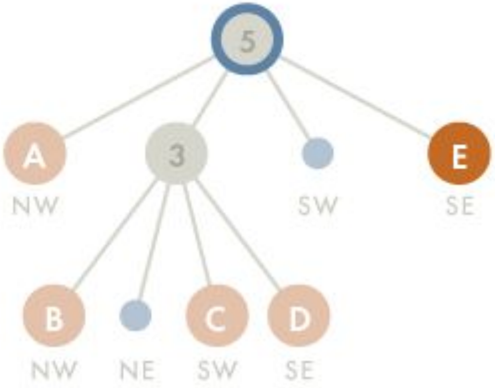
Key Algorithm

- Constructing the Barnes-Hut tree
- Caculating the force acting on each body

Algorithms Deployed 3 -Barnes-Hut Algorithm (con't)

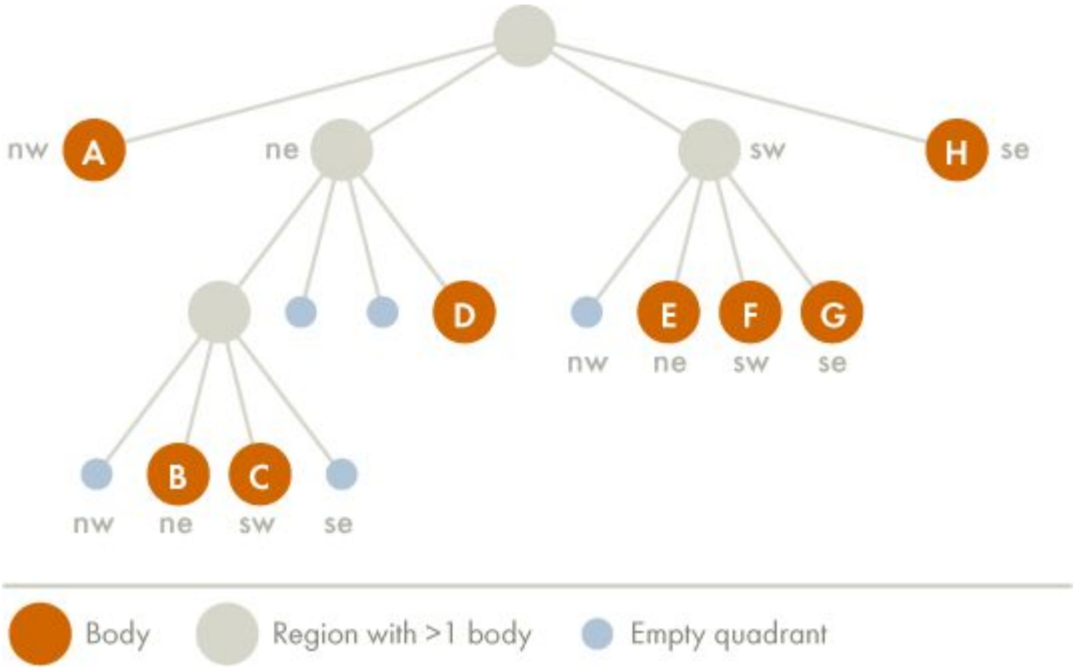
It divides the set of bodies into group by storing them in a so called quad-tree. Quad-tree is similar to binary tree or a multi-branch tree.

Each node in quad-tree has 4 children, some of which may be empty. Each node is a quadrant of the parent area.



Algorithms Deployed 3 - Barnes-Hut Algorithm (con’t)

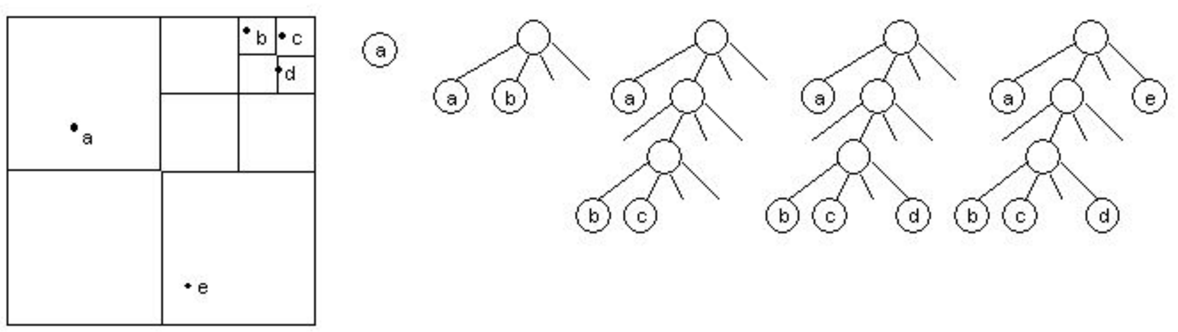
- External node: a single body.
- Internal node: the group of bodies under this node, stores the center-of-mass and the total mass of all its child node.



Barnes-Hut Algorithm - Construct the Quad-tree

When inserting a body b into the quad-tree rooted at node x, you have to follow these steps:

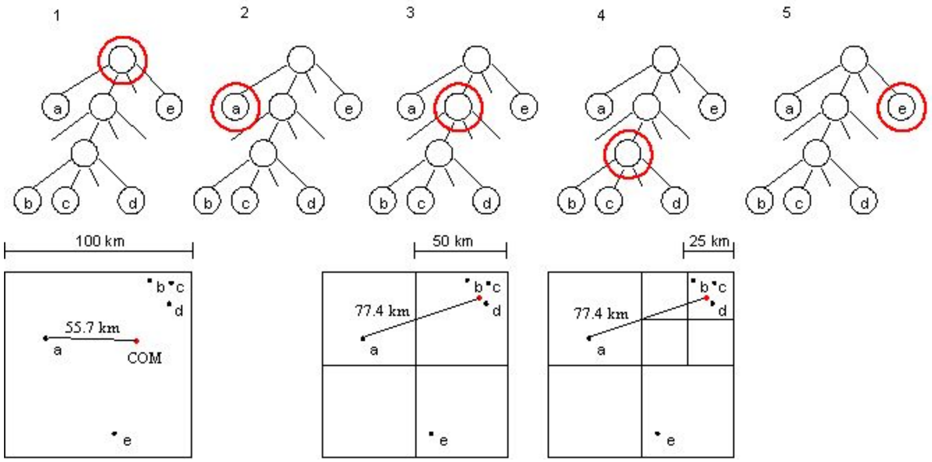
1. If node x is null now, insert the new body b here.
2. If node x is an internal node, update the total mass and the center-of-mass of x and then insert the body b into the corresponding subdivision recursively.
3. If node x is an external node, assume it contains a body called c, then this area has two bodies b and c now. Subdivide the area then insert both b and c into corresponding quadrant recursively. Due to b and c may still end up in the same quadrant, you should keeps subdivide the region until the requirement is meet. Finally, update the center-of-mass and total mass of x



Barnes-Hut Algorithm - Calculating the force

To calculate the net force acting on a body, say that b, follow the following procedure.

1. If the node is an external node, calculate the force exerted by the current node on b, and add it to the b's net force
2. Otherwise, calculate the quotient s/d. If $s/d < \theta$, consider this internal node as a single body and calculate the force exerts on b. Add it to b's net force, too.
3. Otherwise, run the above procedure recursively on each of the current node's child nod



Experimental Configurations

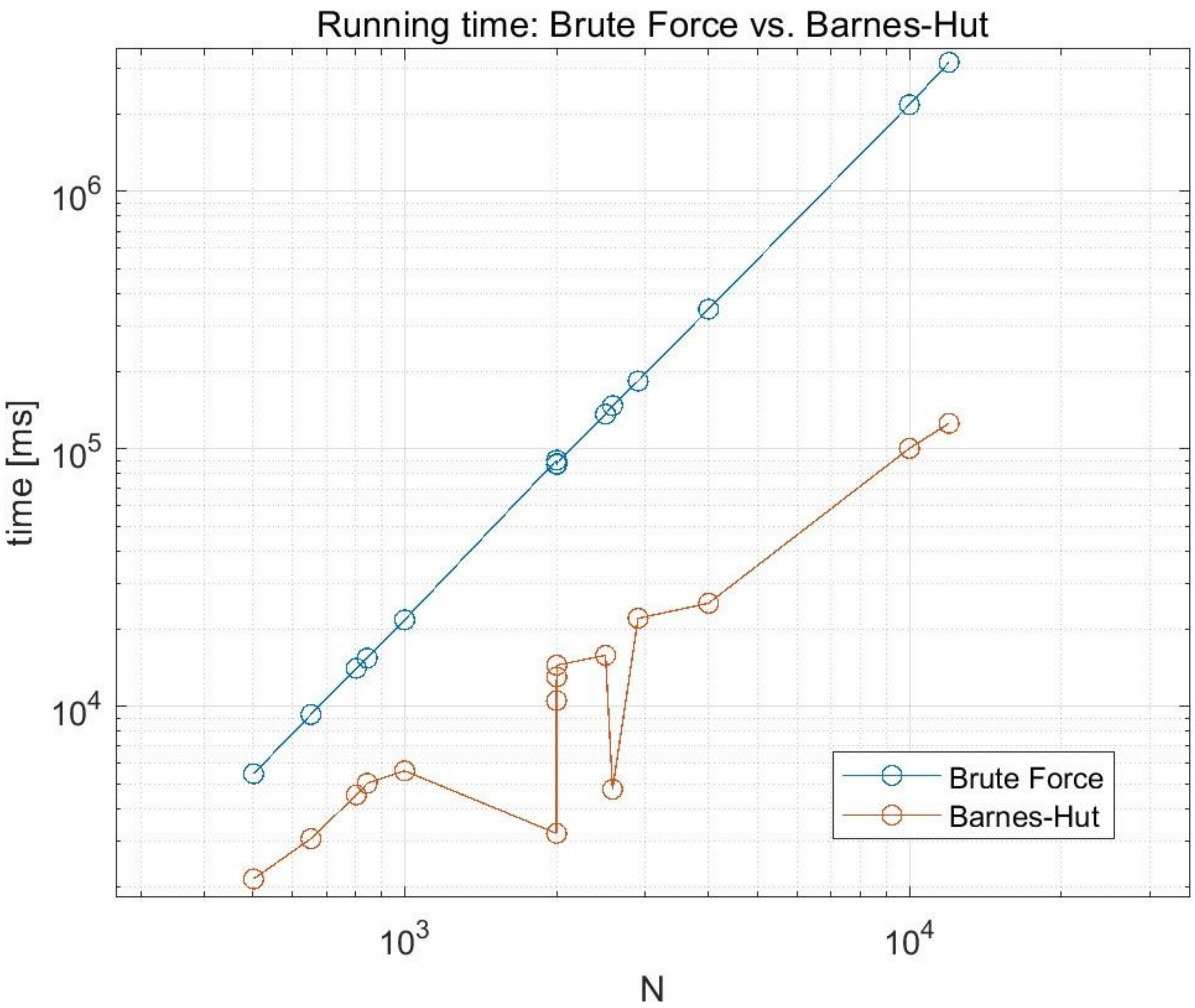
- Suppose that all the bodies are in the same flat. (If in a space, we have to use oct-tree, not quad-tree.)
- Create an animation for N-Body simulation
- Each test file has N bodies and a preset radius of the universe. Each body's initial position, velocity, mass and the color hass been set.
- Run 1024 steps for both the brute force algorithm and the Barnes-Hut algorithm, using the same datasets.

```
5
2.5E11
0.000E00   0.0   0.0   0.000E00   4.97250E41     255   255   0
5.790E10   0.0   0.0   2.395E10   8.25500E34     0     255   0
1.082E11   0.0   0.0   1.750E10   1.21725E36     255   0     255
1.496E11   0.0   0.0   1.490E10   1.49350E36     0     50    255
2.279E11   0.0   0.0   1.205E10   1.60475E35     255   0     0
```

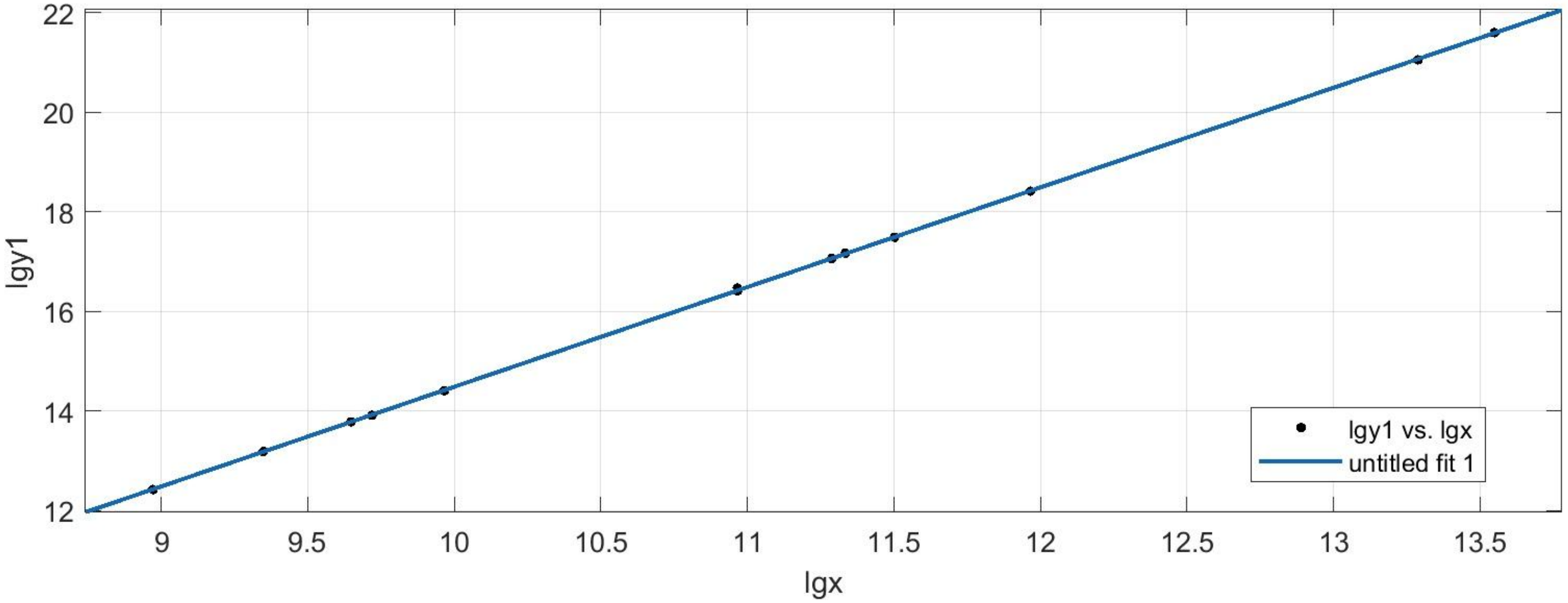
Results and Analysis

- Brute Force Algorithm
  - Time Complexity: O(N^2).
    - For each body you have to go through all the other bodies and calculate the pair-wise force
  - Space Complexity: O(N)
    - Algorithm need store every body.
- Barnes-Hut Algorithm
  - Time Complexity:O(NlogN).
    - The N bodies will construct an quad-tree with average height O($log_2 N^{1/3}$)
    - Time required to construct the tree is O(NlogN)
    - Time required to update the center-of-mass and total mass of each node in quad-tree is O(NlogN).
  - Space Complexity: Hard to calculate
    - Best case: A full 4-tree
    - Worst case: If two nodes are infinitely close enough, the tree can be infinit tall.

Results and Analysis

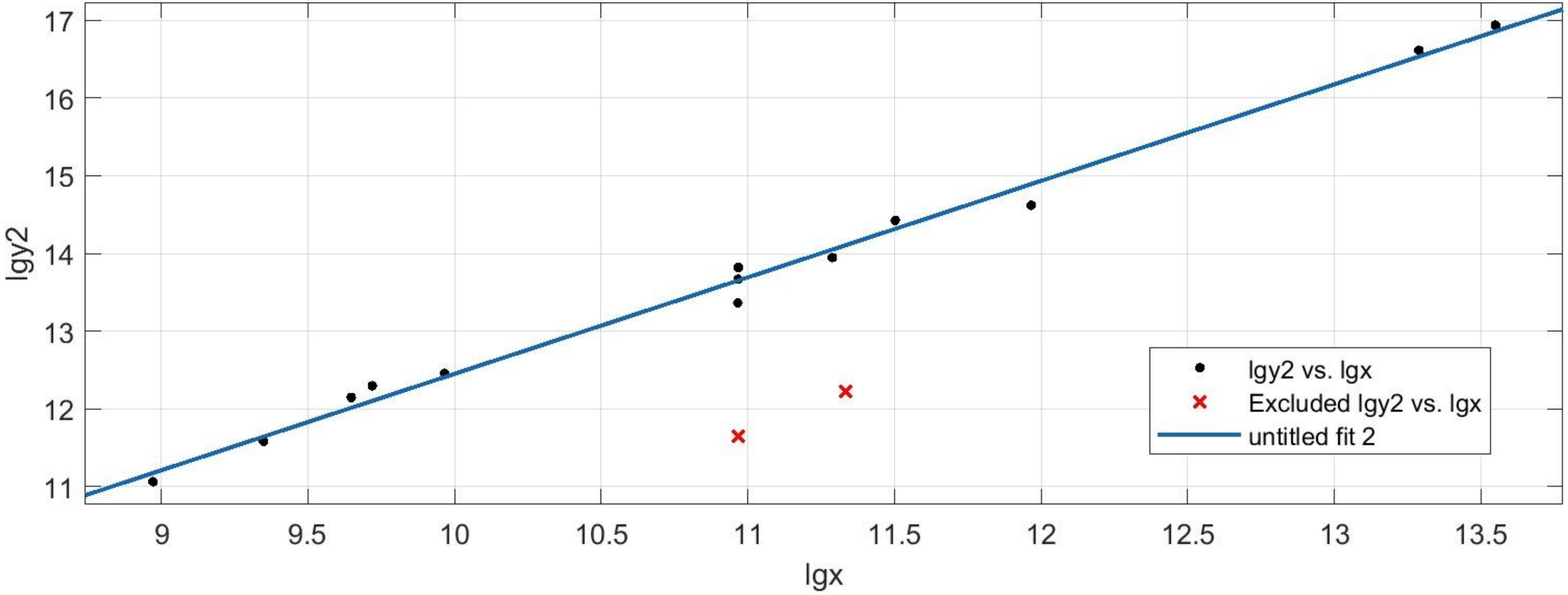


Results and Analysis



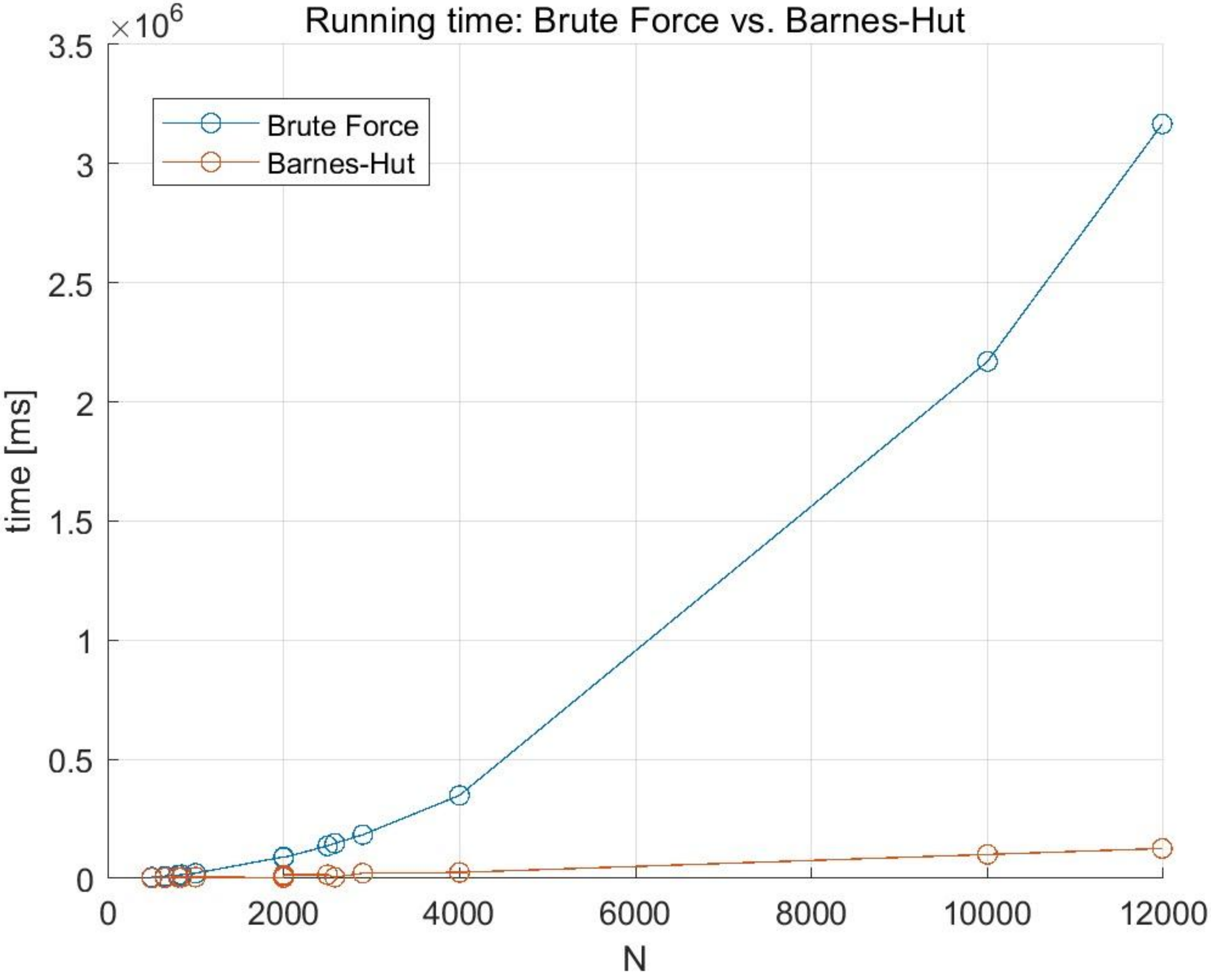
lg(Size of dataset) - lg(Running time)

Results and Analysis



lg(Size of dataset) - lg(Running time)

Results and Analysis



Discussion

Quad-tree

Quad-tree is a tree data structure in which each internal node has exactly four children. Quad-tree is the 2D analog of oct-rees and are most used to partition a 2D space by recursively subdividing it into four quadrants.

Typical use of quad-tree:

- Image processing
- Mesh generation
- spatial indexing
- Efficient collision detection in two dimensions

DEMO TIME!