

COSC3500 Forum Presentation

Max Bo

22nd of October

Outline

Description

Demo

Integration

Barnes-Hut

Correctness

Performance Optimizations

Parallelization

Test Plans

Description

The task was to create a stock-standard, 2-dimensional gravitational n -body simulator.

All bodies were to be assumed to be point masses. The simulation was to be accurate, maintaining a constant total energy, and exhibiting phenomena such as apsidal precession.

Demo

Integration I

$$F = G \frac{m_1 m_2}{r^2}$$

$$a_i = F(x_i)$$

$$v_{i+1} = v_i + a_i \Delta t$$

Integration II

Dehen and Read note that the Euler method ‘performs very poorly in practice’, further noting that ‘errors are proportional to Δt^2 ’.

They contrast it with the second-order *Leapfrog* symplectic integrator, which is ‘heavily used in collisionless N-body applications’.

$$x_i = x_{i-1} + v_{i-1/2} \Delta t$$

$$a_i = F(x_i)$$

$$v_{i+1/2} = v_{i-1/2} + a_i \Delta t$$

which only requires a single acceleration calculation per every two half timesteps

Integration III

and a 'kick-drift-kick' form

$$v_{i+1/2} = v_i + a_i \frac{\Delta t}{2}$$

$$x_{i+1} = x_i + v_{i+1/2} \Delta t$$

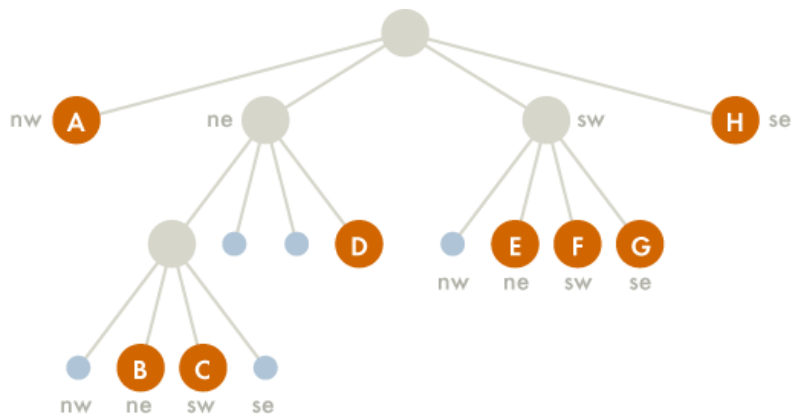
$$v_{i+1} = v_{i+1/2} + a_{i+1} \frac{\Delta t}{2}$$

that is stable with variable timestepping, but incurs an additional acceleration calculation per every two half timesteps.

Barnes-Hut I



Barnes-Hut II



Body

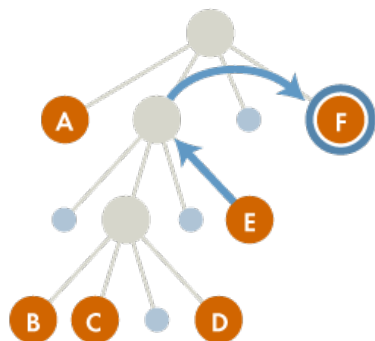
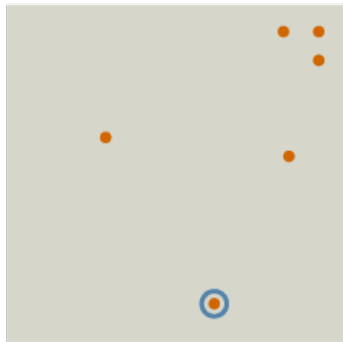


Region with >1 body

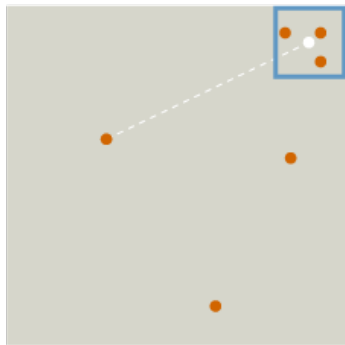


Empty quadrant

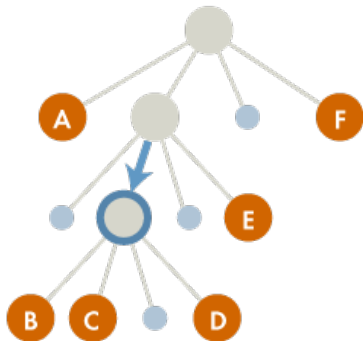
Barnes-Hut III



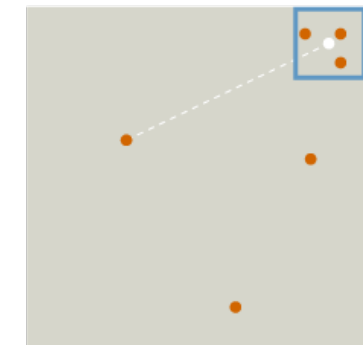
Barnes-Hut IV



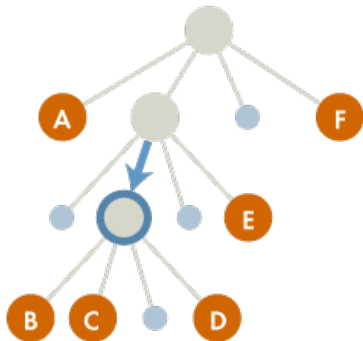
d 66.9
 s ——— 25
 θ



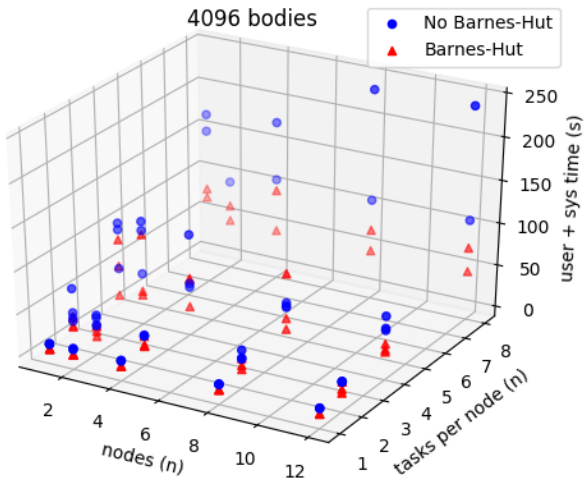
Barnes-Hut V



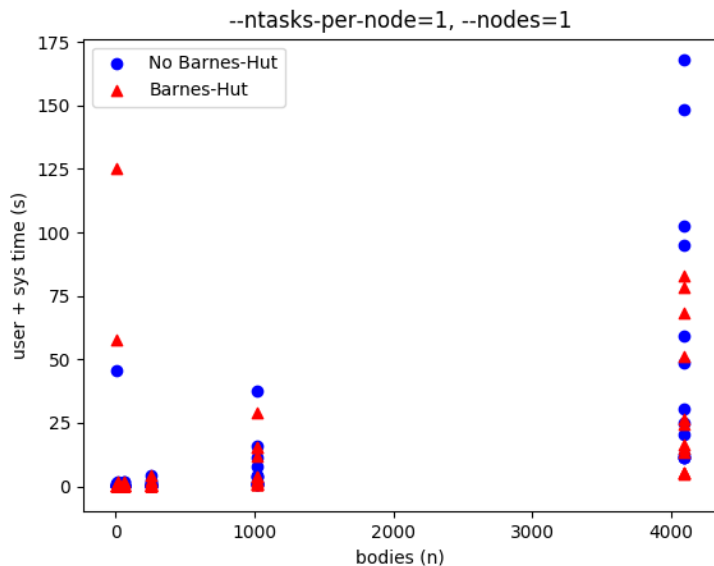
d 66.9
 s — 25 |
 θ



Barnes-Hut VI



Barnes-Hut VII

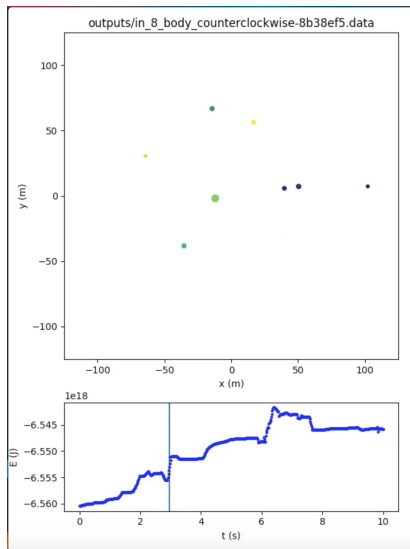


Correctness I

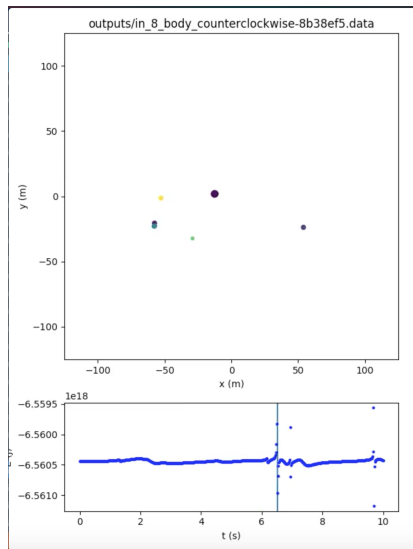
$$U = -G \frac{mM}{R}$$

$$E_k = \frac{1}{2}mv^2$$

Correctness II



Correctness III

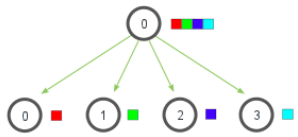


Performance Optimizations I

Parallelization I

```
while (True) {  
    #pragma omp parallel for shared(bodies)  
    for (size_t i = 0; i < bodies.size(); i++) {  
        auto& body = sbodies[i];  
        quad_tree.calculate_force(body);  
    }  
}
```

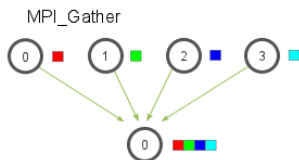
MPI_Scatter



Parallelization II

```
#pragma omp parallel for shared(bodies)
  for (size_t i = 0; i < sbodies.size(); i++) {
    auto& body = sbodies[i];

    if (step % 2 == LEAP) {
      body.leap(timestep);
    }
    else {
      body.frog(timestep);
    }
  }
}
```



Test Plans I