Astronomical Particle Simulations with GPU

N.Nakasato (University of Aizu)

JIFT Workshop 2011
" The Next Stage in the Progress of Simulation Science in Plasma Physics "
NIFS, Toki, Japan, 2011/12/02-03

Agenda

- Introduction to particle simulation in astronomy
- Direct summation code on GPU
- Octree implementation on GPU
  - Application to SPH method
- Summary

Accelerators: GPU

- Emergent architecture for HPC
  - "parallel computer" on a chip
  - Good for compute intensive app.

Complexity	Application	Sustained / Peak
$O(N^3)$ or more	Numerical Integration	100%
$O(N^2)$	simple N-Body	90% or more
$O(N^{1.5})$	Matrix Multiplication	60% (so far)
$O(N \log N)$	Octree method	1 - 2%
$O(N)$	Explicit Hydro code	very low in principle

Objects in the Universe

solar system

$$N \sim 10$$

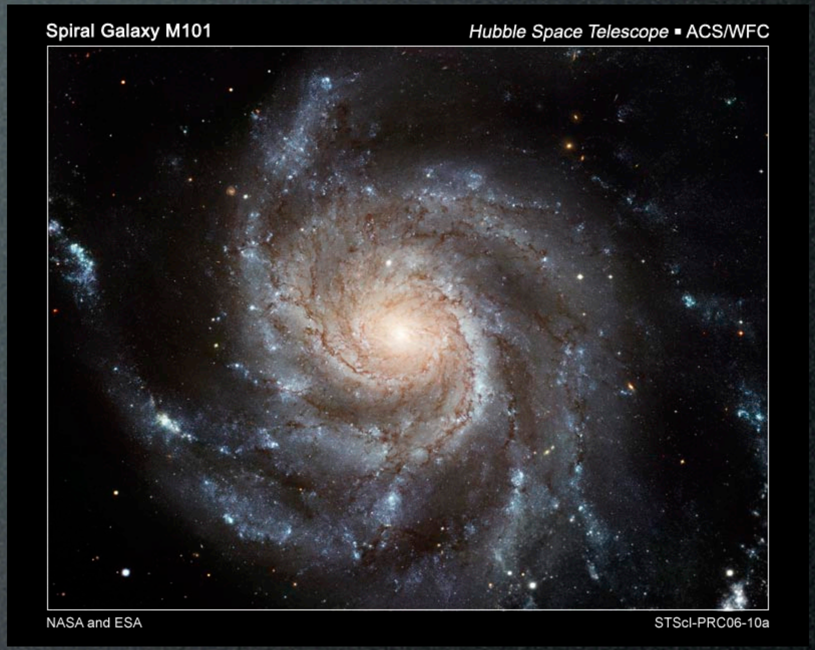
$$t_{\rm lifetime} \sim 10^9 {\rm yr}$$

star cluster

$$N \sim 10^5$$

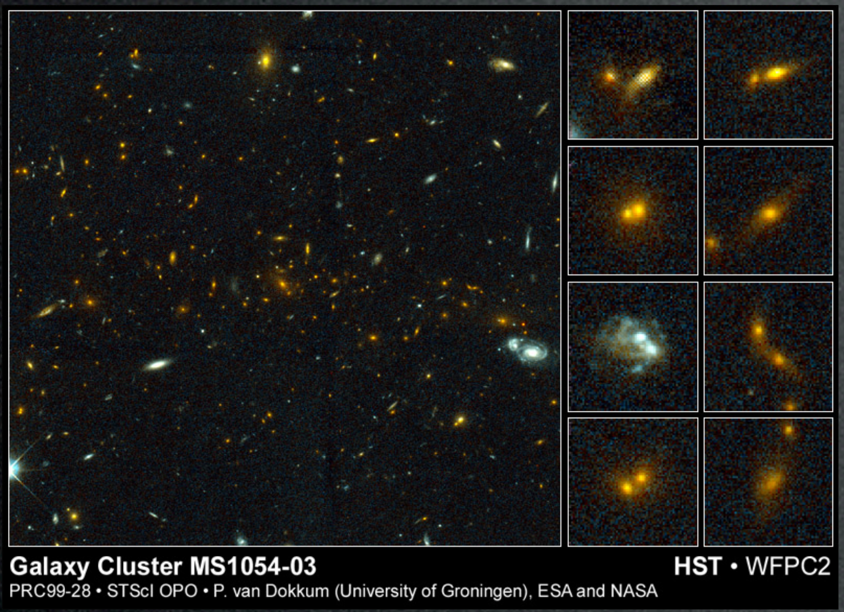
$$t_{\rm lifetime} \sim 10^{10} {\rm yr}$$

galaxy



$$N \sim 10^{11}$$

$$t_{\rm lifetime} \sim 10^{10} {\rm yr}$$



cluster of galaxies

$$N \sim 10^3$$

$$t_{\rm lifetime} \sim 10^{10} {\rm yr}$$

Numerical model

solar system

sun&planets

$N \sim 10$

$t_{\rm lifetime} \sim 10^{10}{\rm yr}$

$t_{\rm dynamical} \sim 1{\rm yr}$

galaxy

blob of stars&DM

$N \sim 10^6 - 10^7$

$t_{\rm lifetime} \sim 10^{10}{\rm yr}$

$t_{\rm dynamical} \sim 10^8{\rm yr}$

star cluster

individual stars

$N \sim 10^5$

$t_{\rm lifetime} \sim 10^{10}{\rm yr}$

$t_{\rm dynamical} \sim 10^5{\rm yr}$

whole universe

blob of DM

$N \sim 10^9 - 10^{11}$

$t_{\rm lifetime} \sim 10^{10}{\rm yr}$

$t_{\rm dynamical} \sim 10^8{\rm yr}$

collisional particle system

$$t_{\text{relaxation}} \sim t_{\text{lifetime}}$$

e.g. simulation of solar system or star cluster
N is small & demand high accuracy

collision-less particle system

$$t_{\text{relaxation}} \gg t_{\text{lifetime}}$$

e.g. simulation of large scale structure in the universe
N is huge & less accuracy

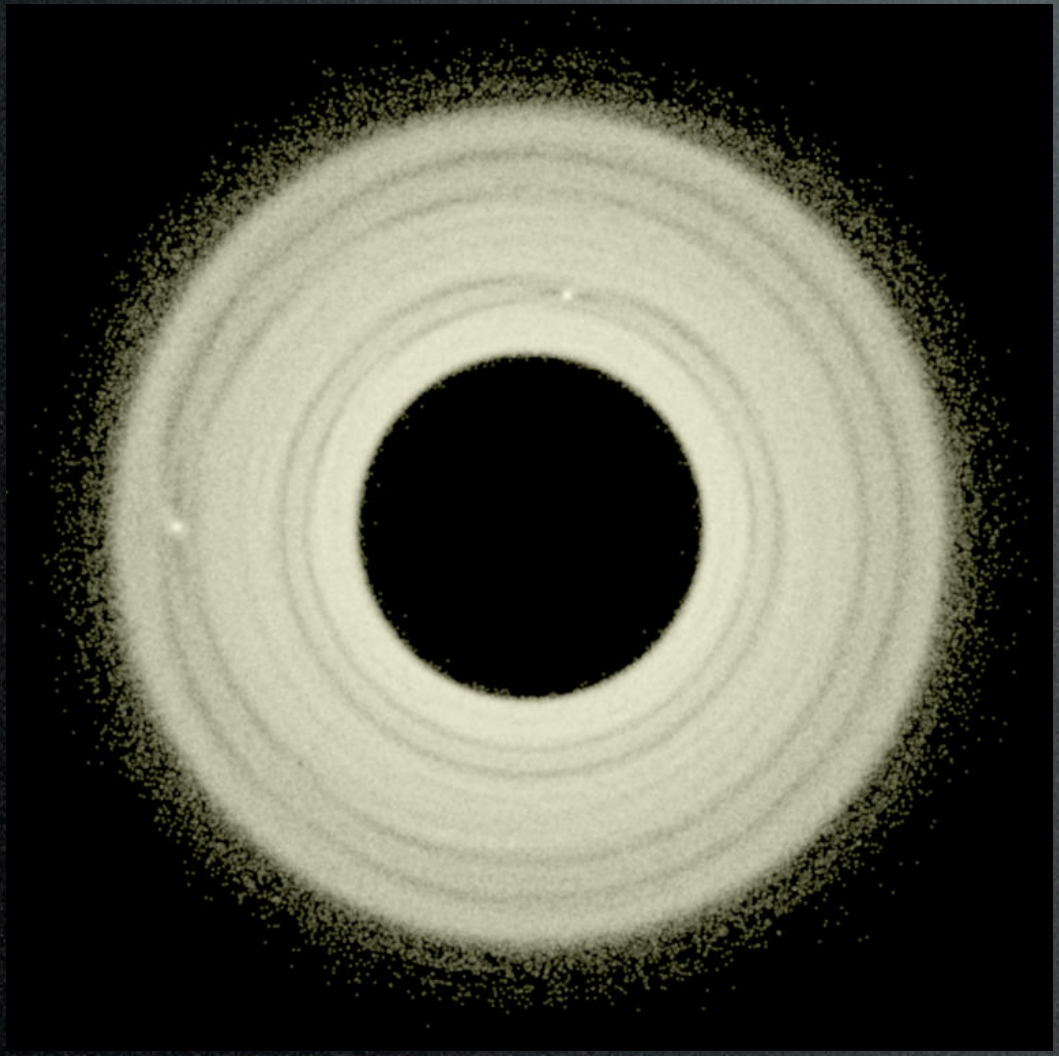
$$t_{\text{relaxation}} = \frac{0.1N}{\ln N} t_{\text{dynamical}}$$

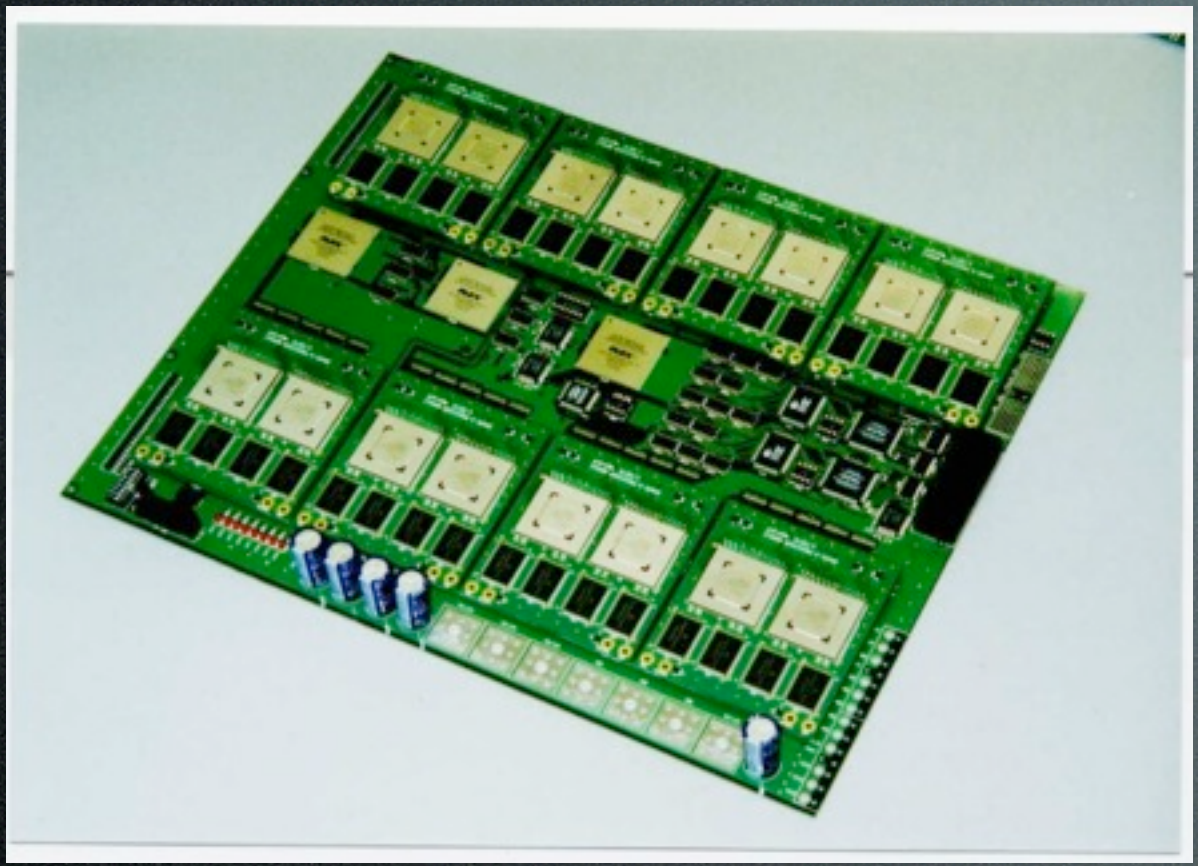
collisional particle system

computational complexity $O(N^2)$

direct summation

need high accuracy





GRAPE-6A (2002)

fixed function

30 Gflops (90MHz)

10W

200 Myen (2.4 Tflops)

super energy efficient

GPU (AMD Cypress, 2010)

programmable

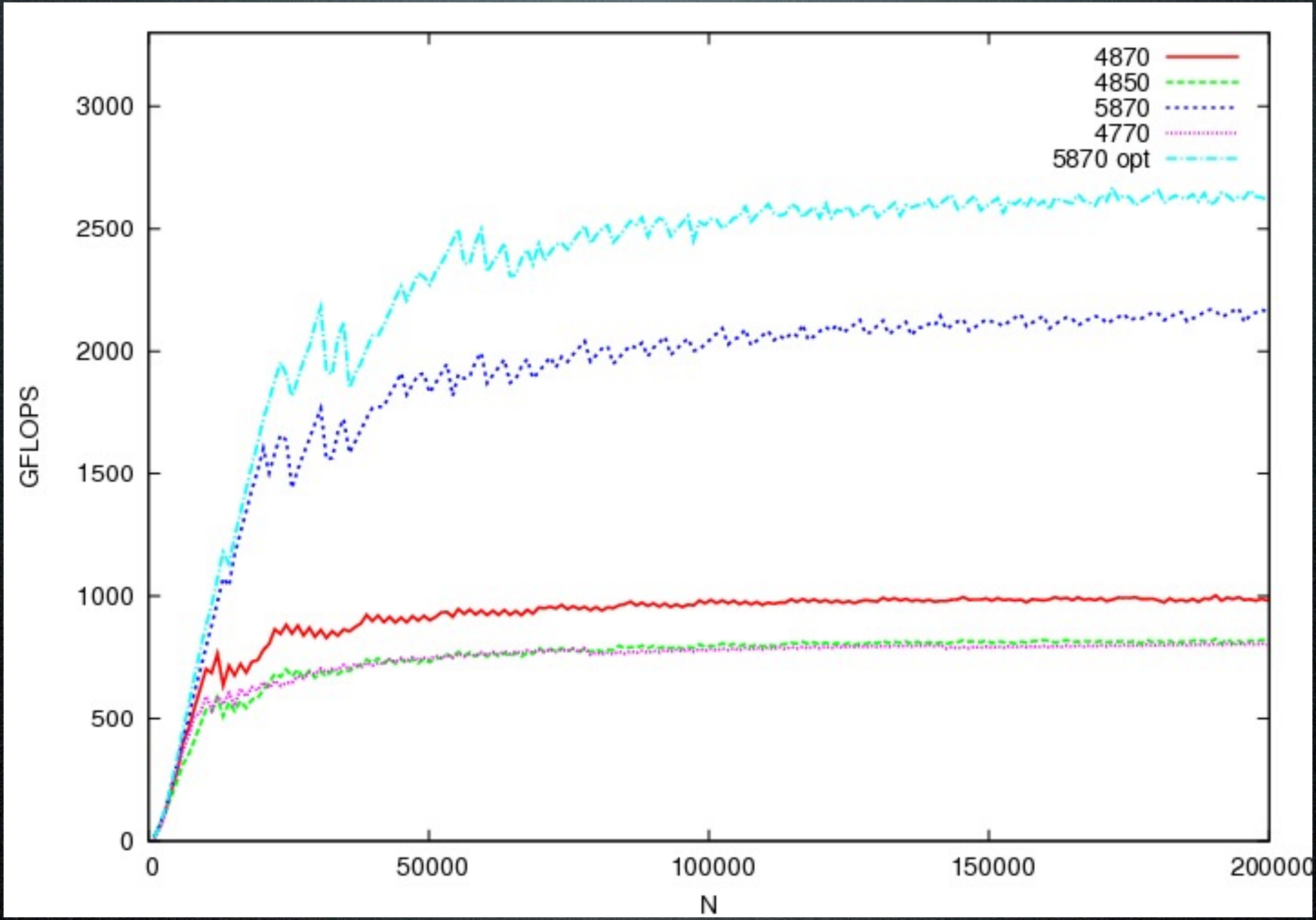
600 Gflops (850MHz)

200W

40,000 yen

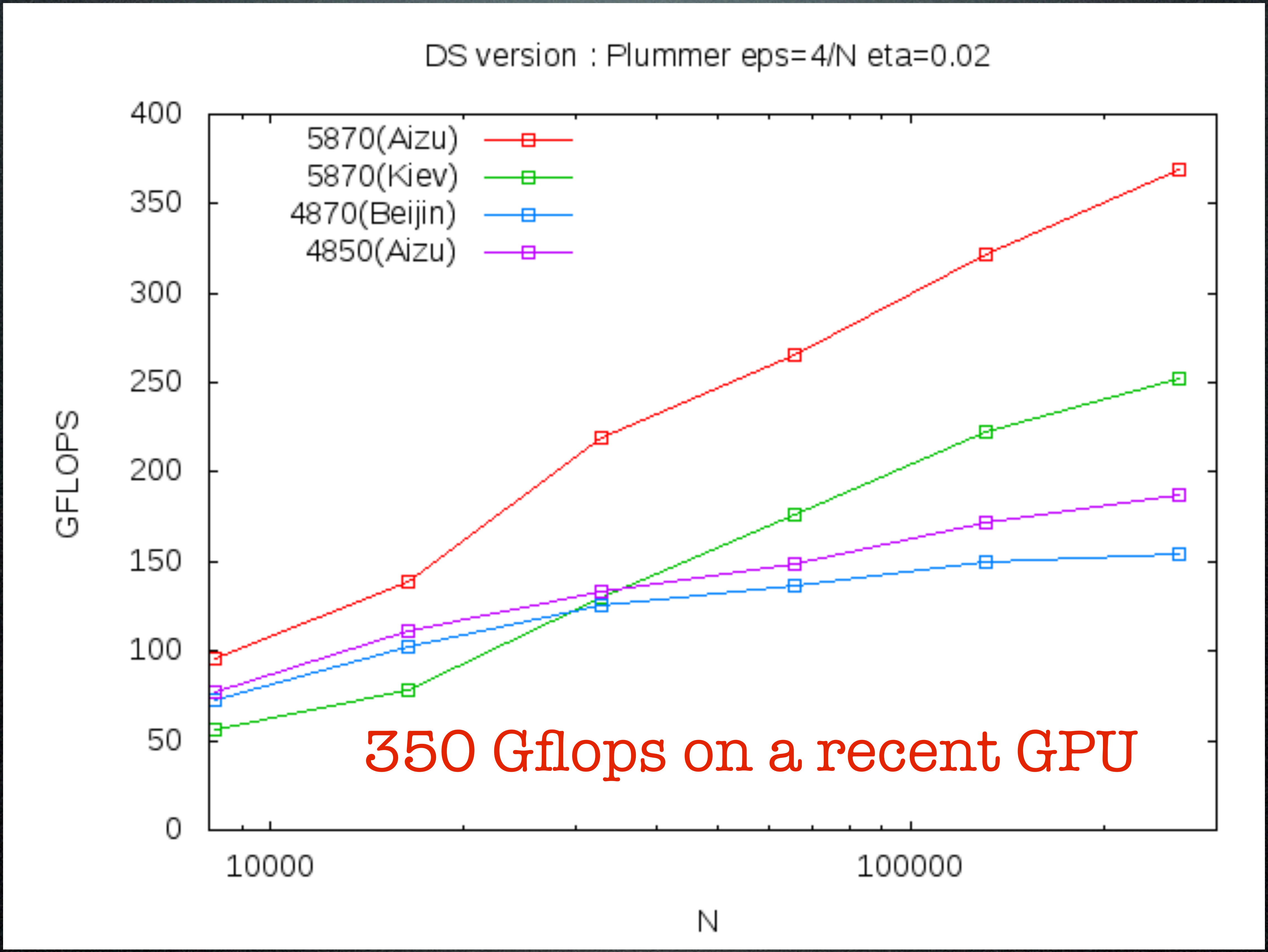
highly cost effective

Performance of O(N^2) algorithm



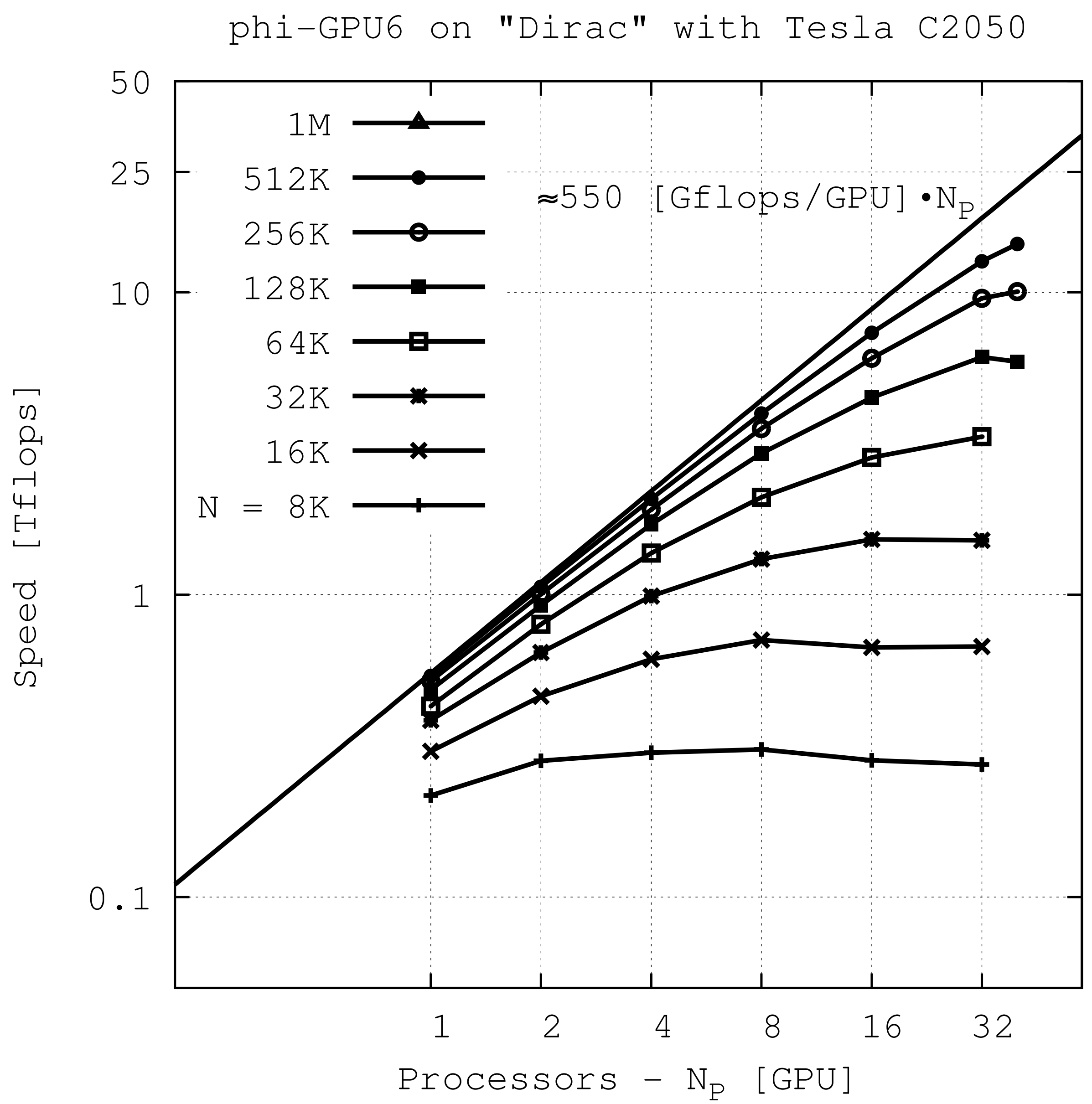
2.6 Tflops in single precision on a recent GPU

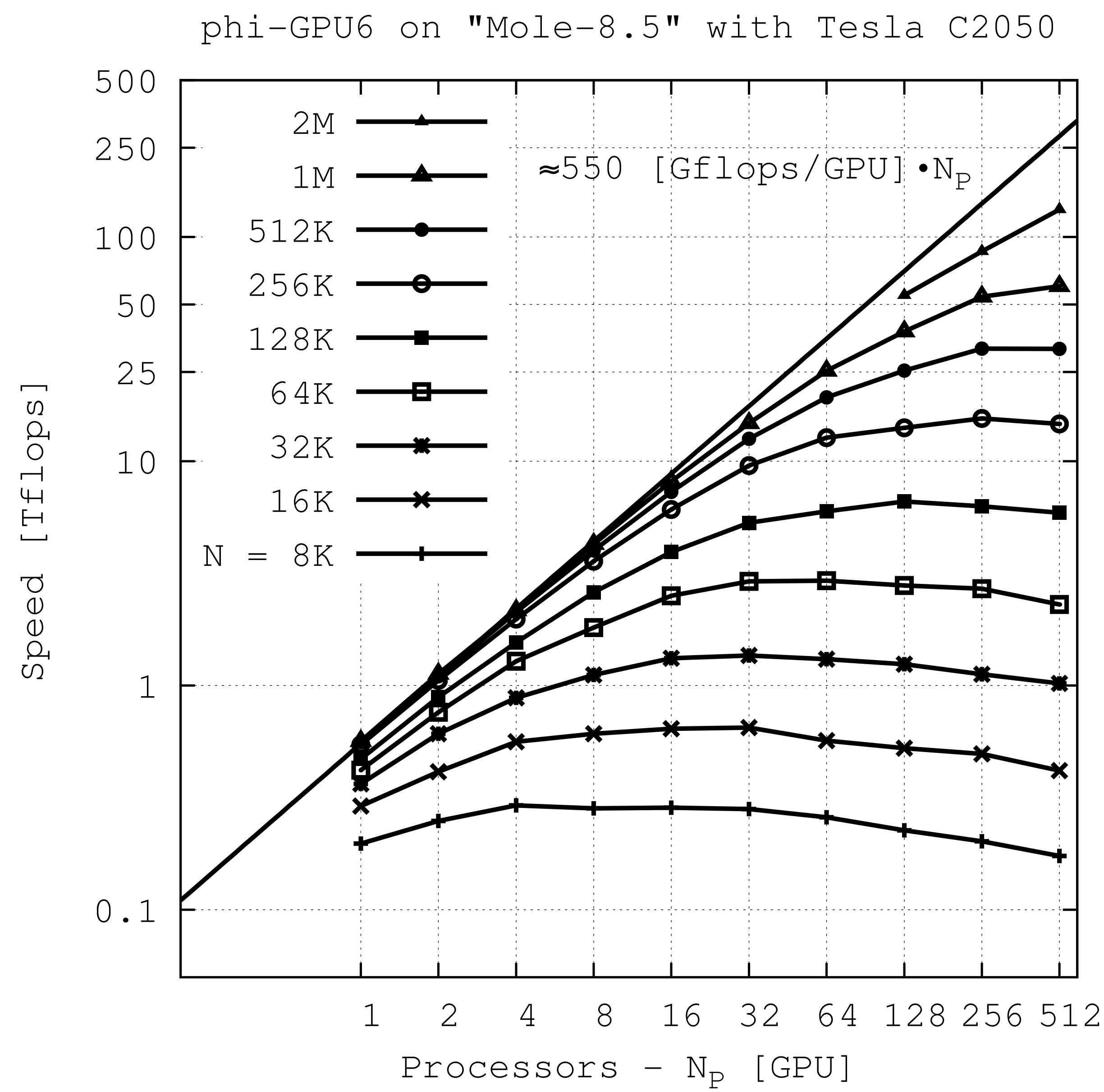
GRAPE-6 emulation library on GPU



phi-GPU6 on Tesla

100 Tflops on a recent GPU cluster





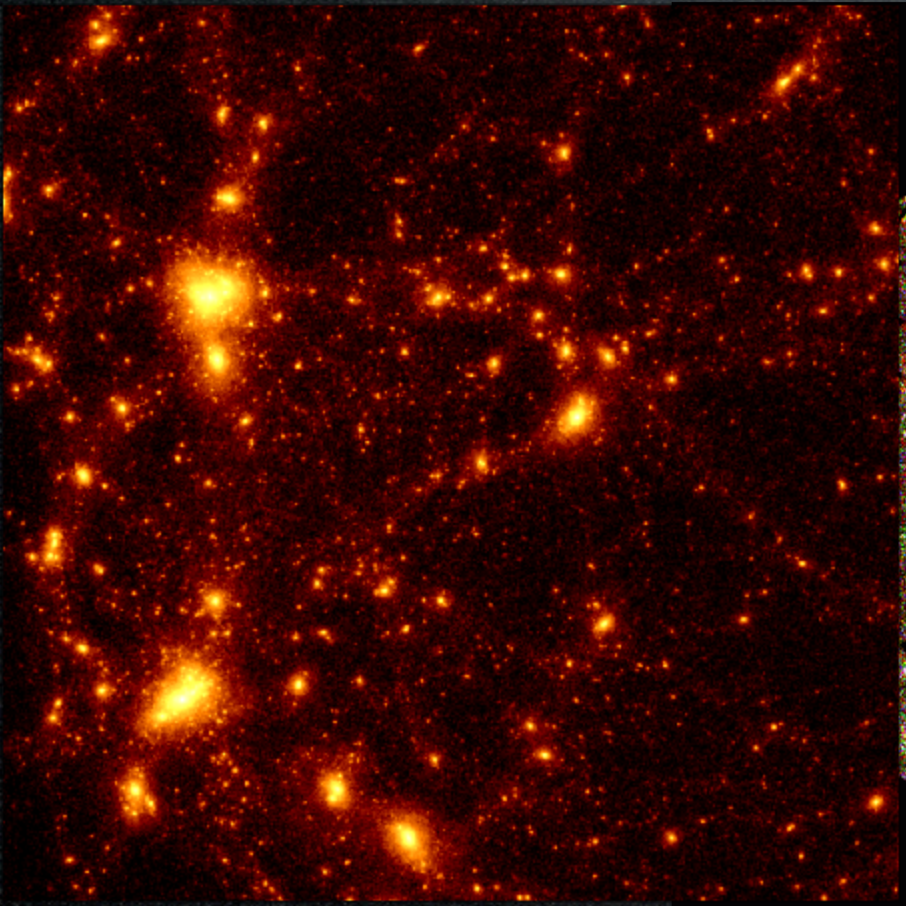
Spurzem, Berczik, Berentzen etal. 2011

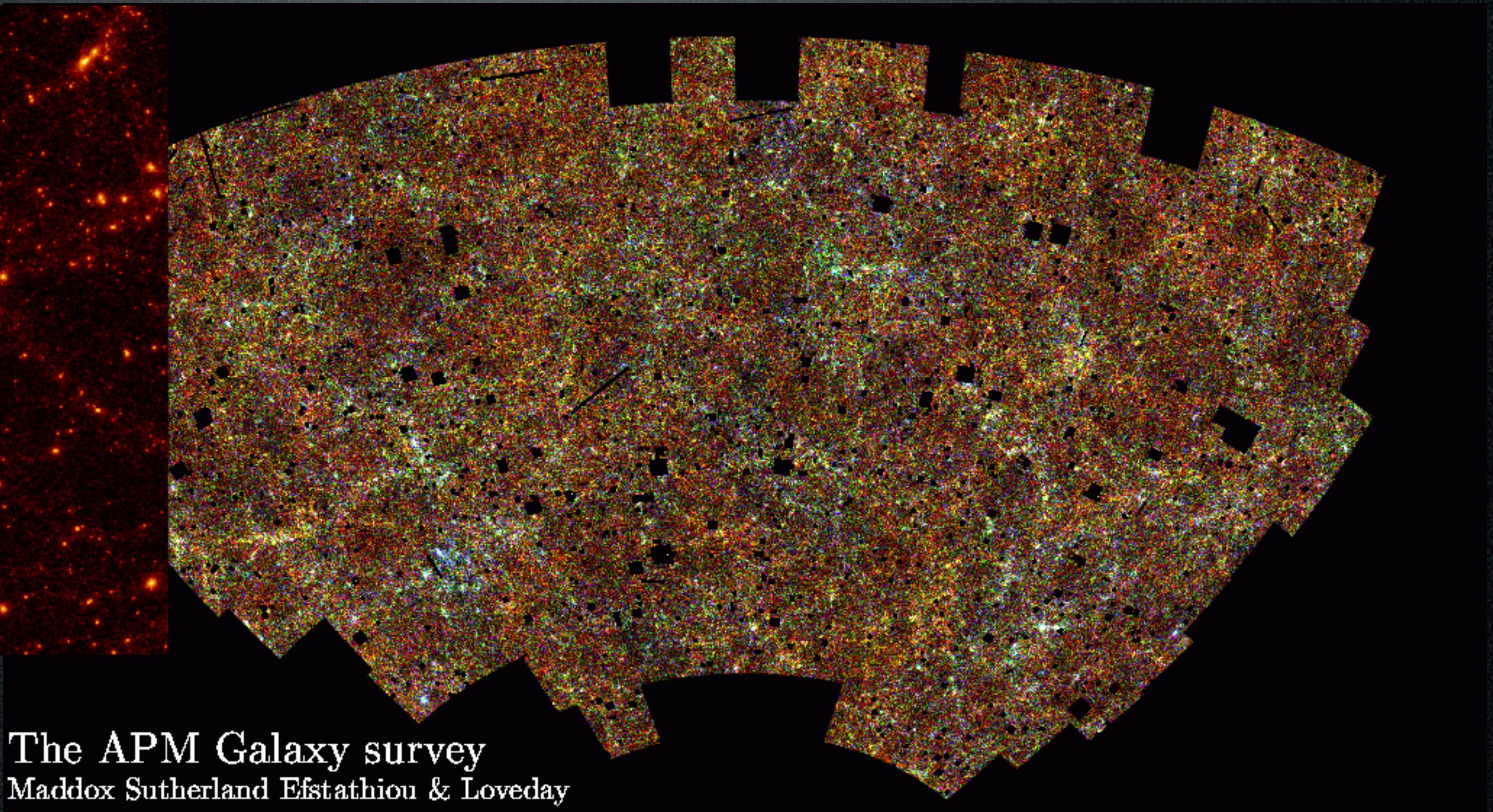
collision-less particle system

computational scheme : O(N log N) or O(N)

tree method, P3M, FMM

$N = 10^6{}_{\text{(galaxy)}} - 10^{12}{}_{\text{(large scale structure)}}$



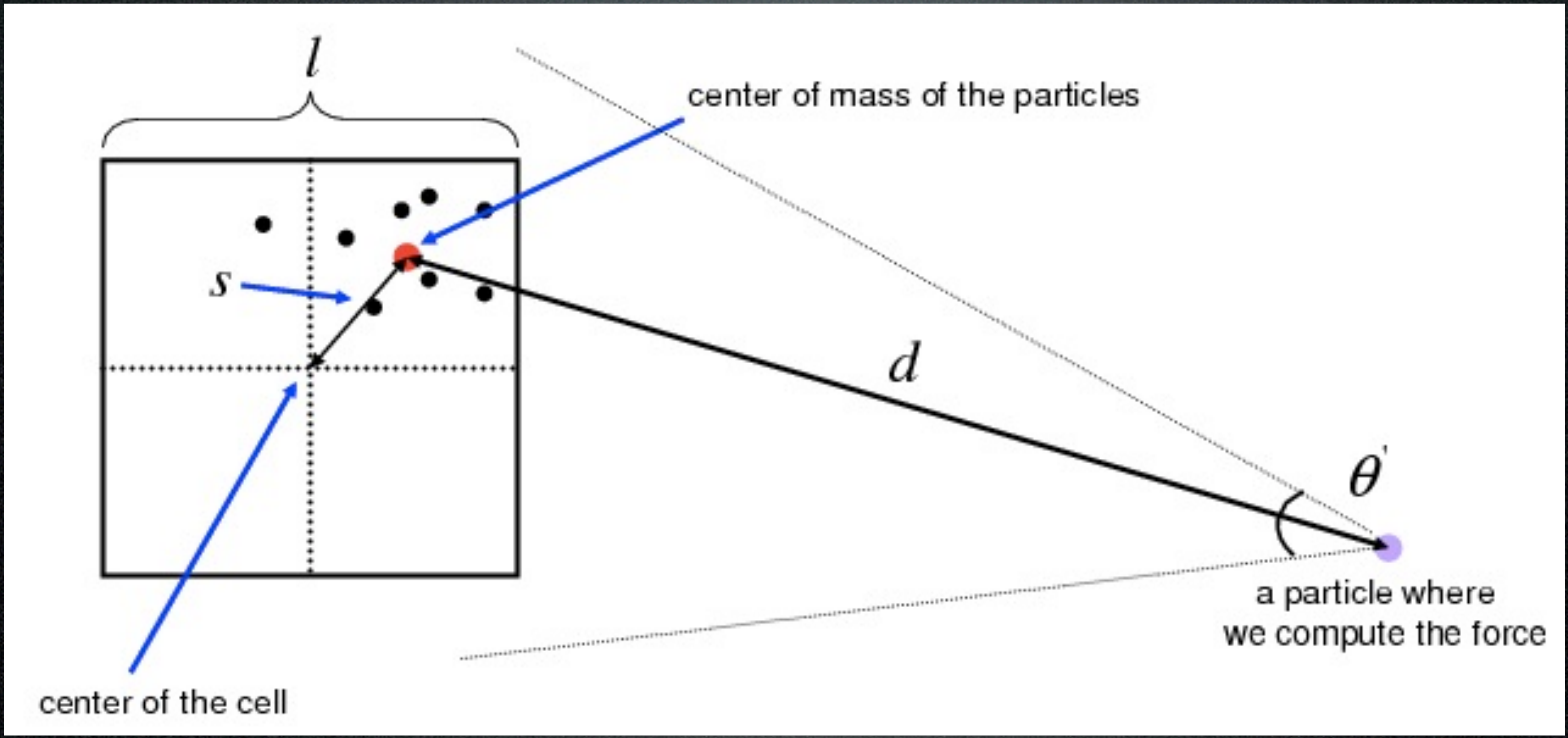


Collision-less case

- $O(N^2)$ algorithm works up to N <100 k
  - It is effective to make FP units busy but slow
    - High accuracy is not always demanded
- There are faster methods
  - O(N log N) methods are
    - Particle-Mesh (FFT based)
    - Octree method
      - Method of choice in many astronomical simulations
  - O(N) method
    - Fast-Multipole Method

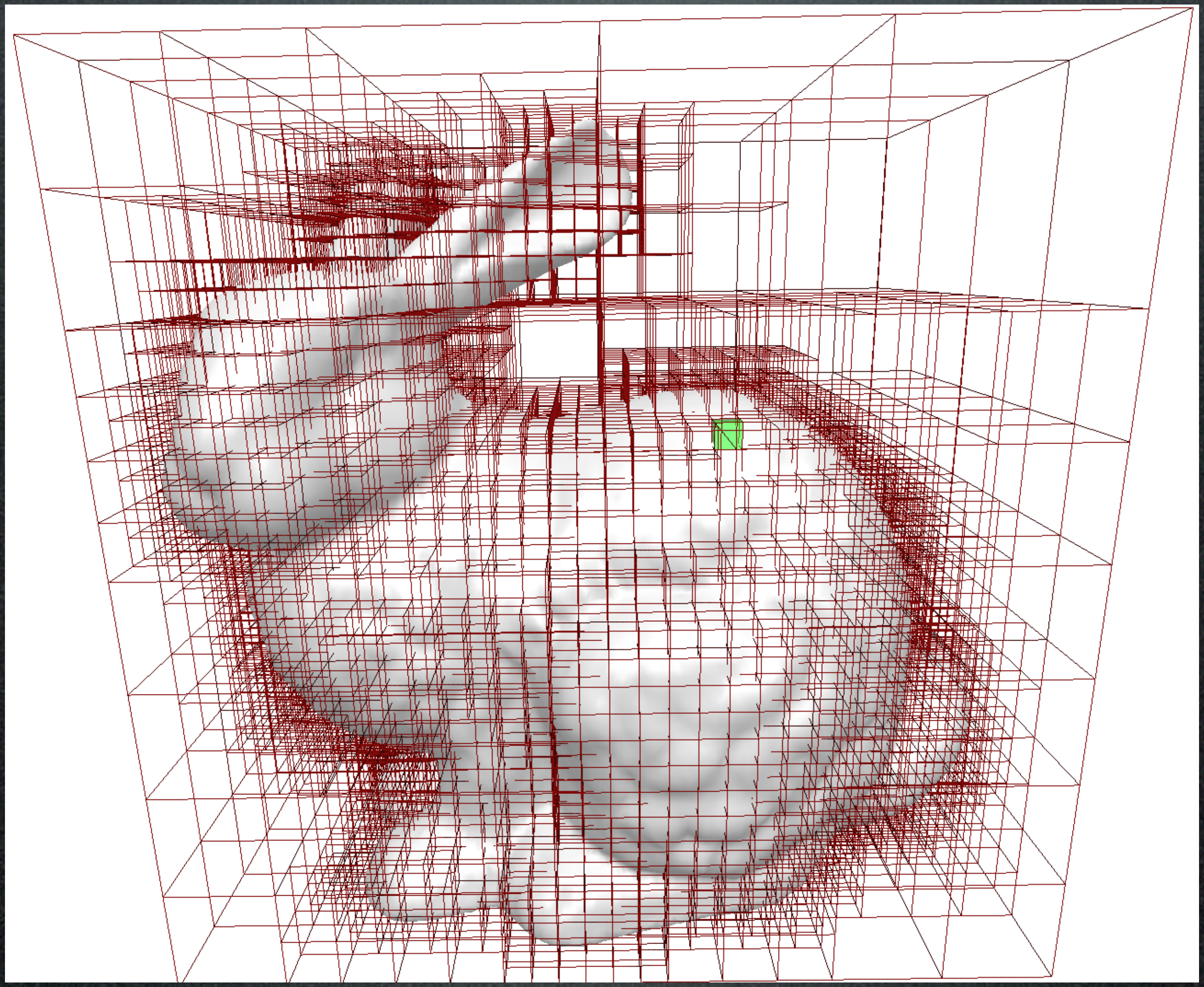
Octree Method

- Approximation method to computer long-range force
  - Systematically replace distant particles with multipole-moment(MM) of the particles



Octree structure

- Recursively sub-divide the space into 2x2x2 cubes where a particle resides
  - Relation between cubes are represented as TREE



Program Flow of Octree

1. Construct a tree structure

   Tree consist of nodes and particles

2. Walk through the tree and compute the MM at each node.

3. For each particle

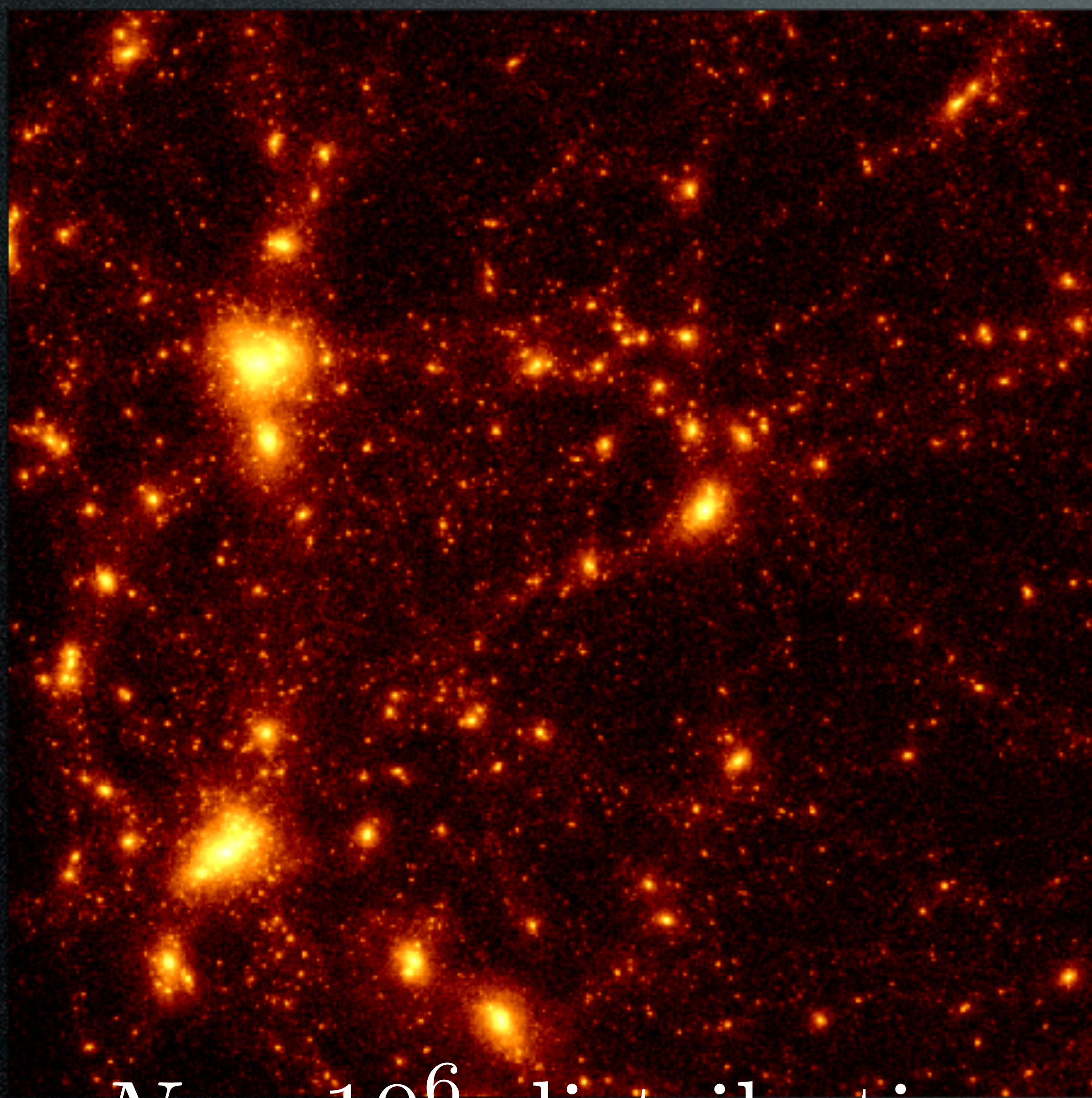
   1. Walk through the tree and check the opening criterion

      1. If it is particle, compute the force

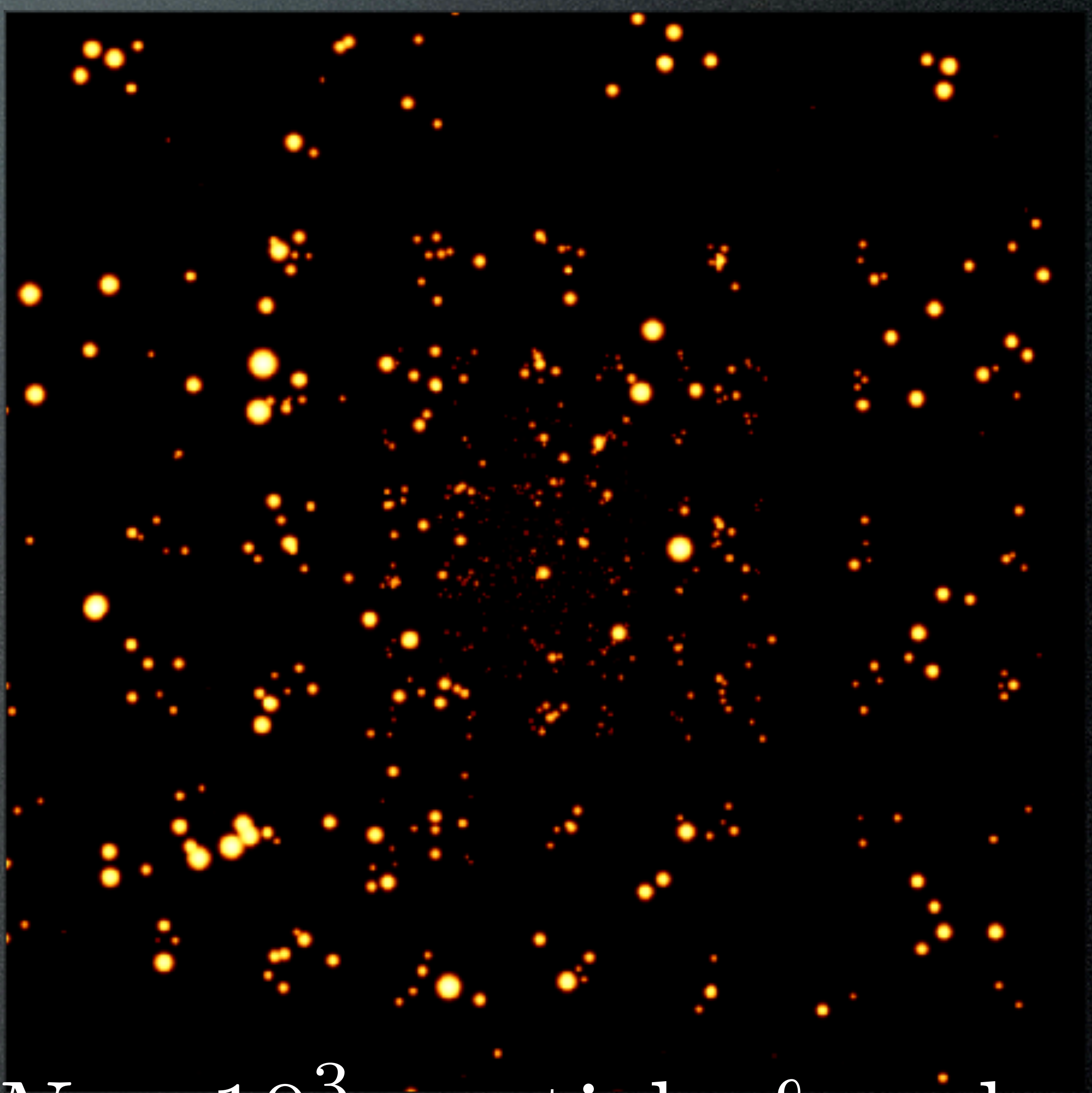
      2. If it is node, compute the force or further walk the children nodes

Reduction of computing

- Distant particles (a node) are replaced with its MM



$N \sim 10^6$ distribution



$N \sim 10^3$ particles&nodes

Recursive Tree Walk

```
procedure treewalk(i, cell)
  if cell has only one particle
    force += f(i, cell)
  else
    if cell is far enough from i
      force += f_multipole(i, cell)
    else
      for i = 0, 7
        if cell->subcell[i] exists
          treewalk(i, cell->subcell[i])
```

Fig. 3. A pseudo code for the force-calculation by traversing the oct-tree

Note on Octree

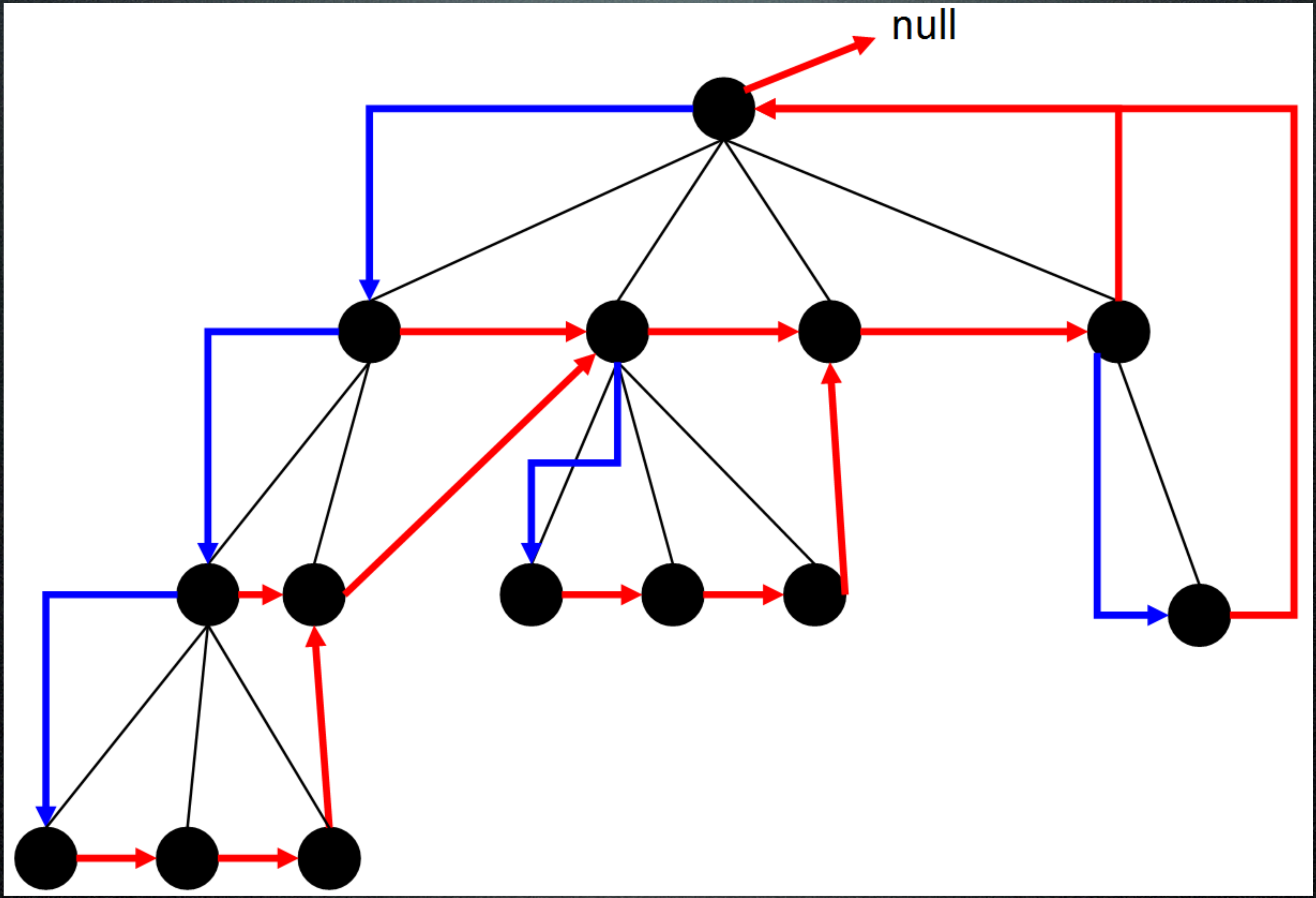
- At stage 3, we can compute force acting on each particle in parallel
  - Force calculation by octree is a parallel problem
    - Vectorized tree, parallel tree code
  - But stage 1 & 2 is not highly parallel
    - these part could be bottleneck

Octree on GPU

- We implement the stage 3 on GPU
  - Possible because of highly parallel nature
  - Originally it was proposed for vecterization of the tree method (Makino 1990)
  - It is applicable to any interaction
    - Gravity/Coulomb force
    - short-range MD force
    - Hydrodynamics (SPH) : explained later
    - Any algorithm required neighbor particles

Threaded Tree Structure

Convert a recursion to an iteration



Iterative Tree Walk

```
procedure treewalk_iterative(i)
  cell = the root cell
  while cell is not null
    if cell has only one particle
      force += f(i, cell)
      cell = cell->next
    else
      if cell is far enough from i
        force += f_multipole(i, cell)
        cell = cell->next
      else
        cell = cell->more
```

Fig. 6. A pseudo code for an iterative treewalk procedure.

See our paper for details

N.Nakasato, Journal of Computational Science, 2011

doi:10.1016/j.jocs.2011.01.006

Flow of Octree on GPU

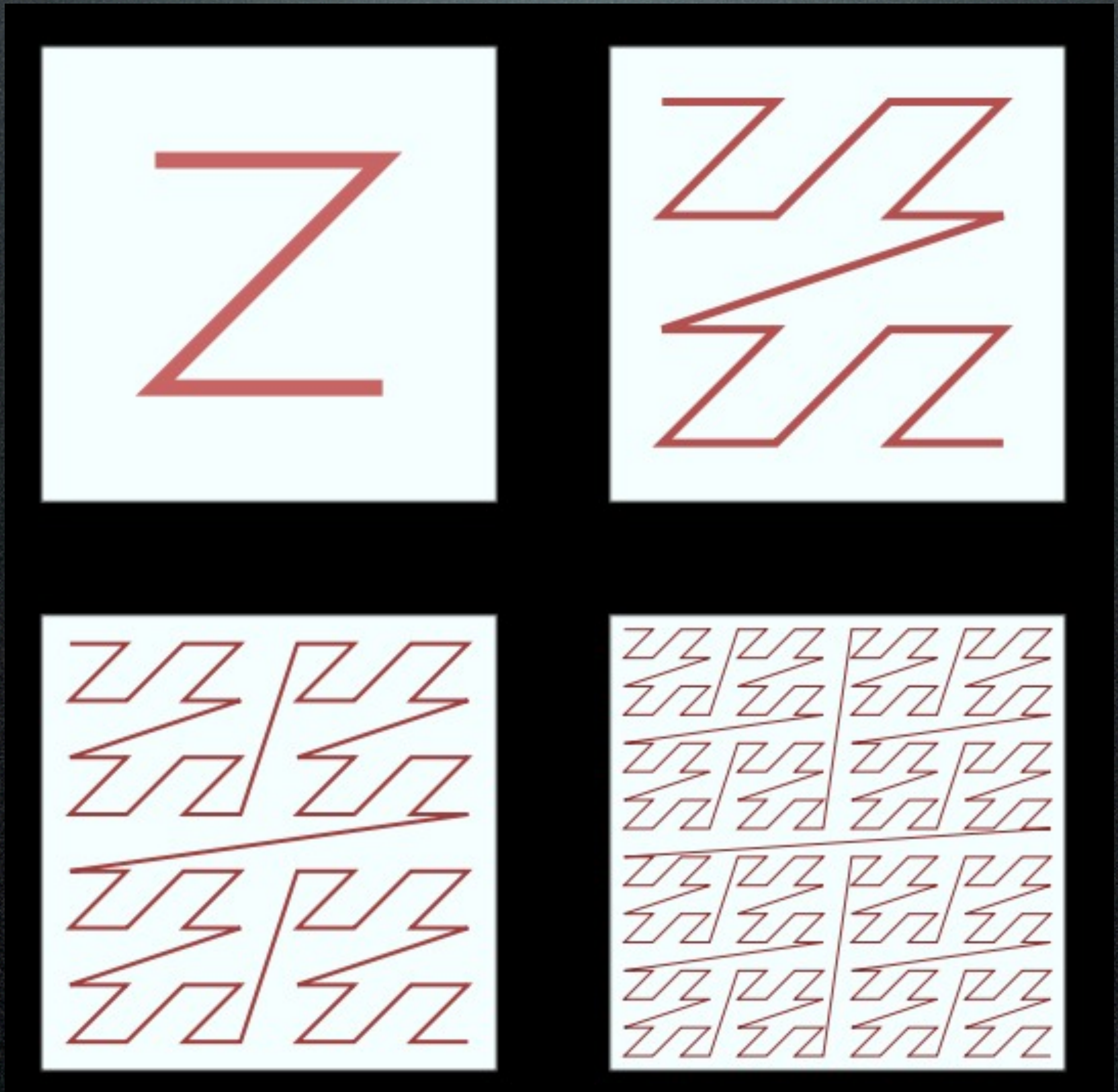
1. Tree construction
2. Compute MM
3. Send the tree-data to GPU
4. For each particle (on GPU)
    1. Walk the tree and check the opening-criterion
    2. Either compute the force or further walking the tree
5. Receive the results from GPU

GPU Programming

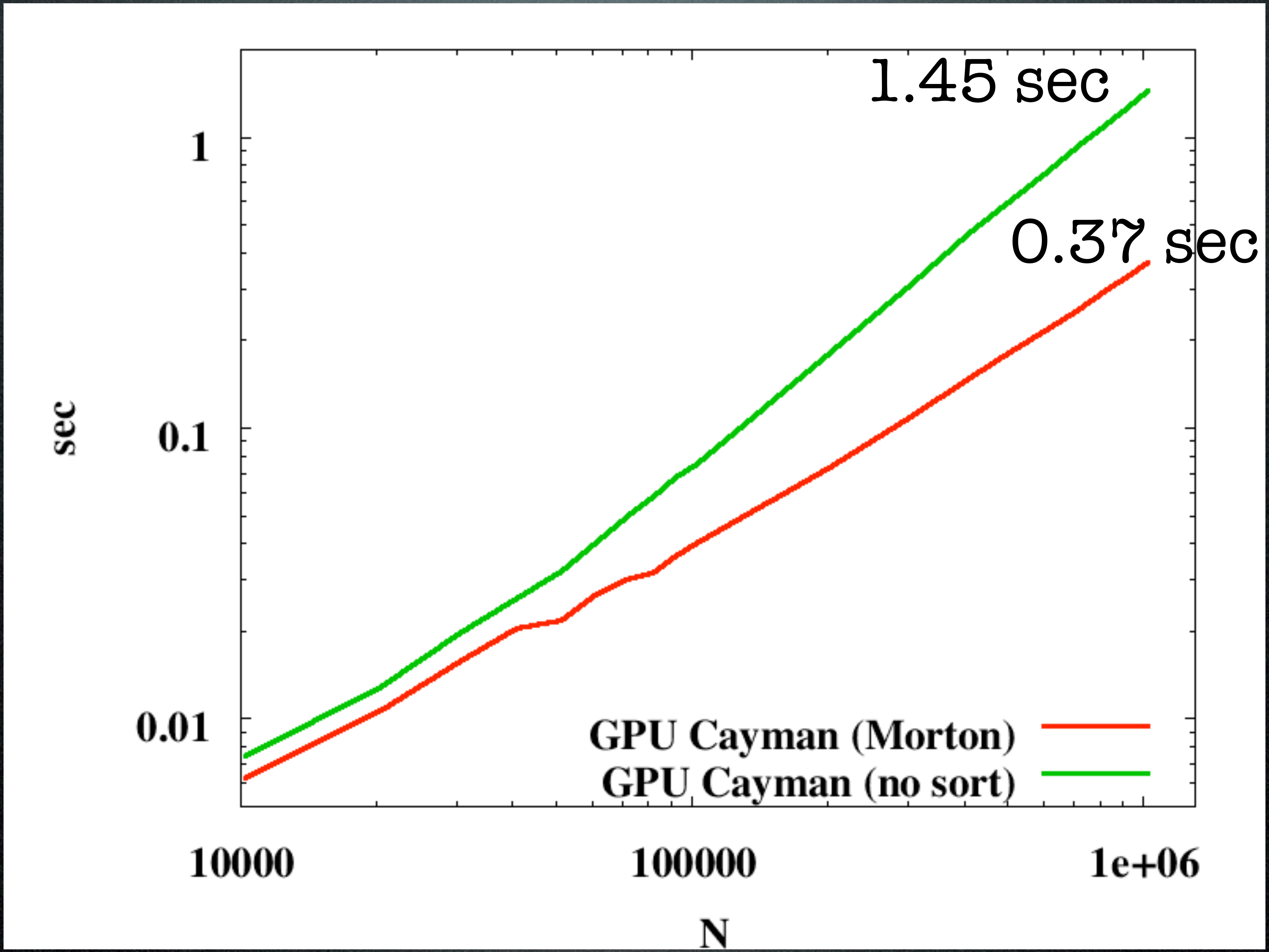
- We use OpenCL for implementing the octree code on GPU and CPU
  - Supported by many devices (CPU,GPU,Cell,DSP)
  - Effectively use multi-core on recent CPUs
  - Recent SDKs are much more mature than before

Ordering of Particles

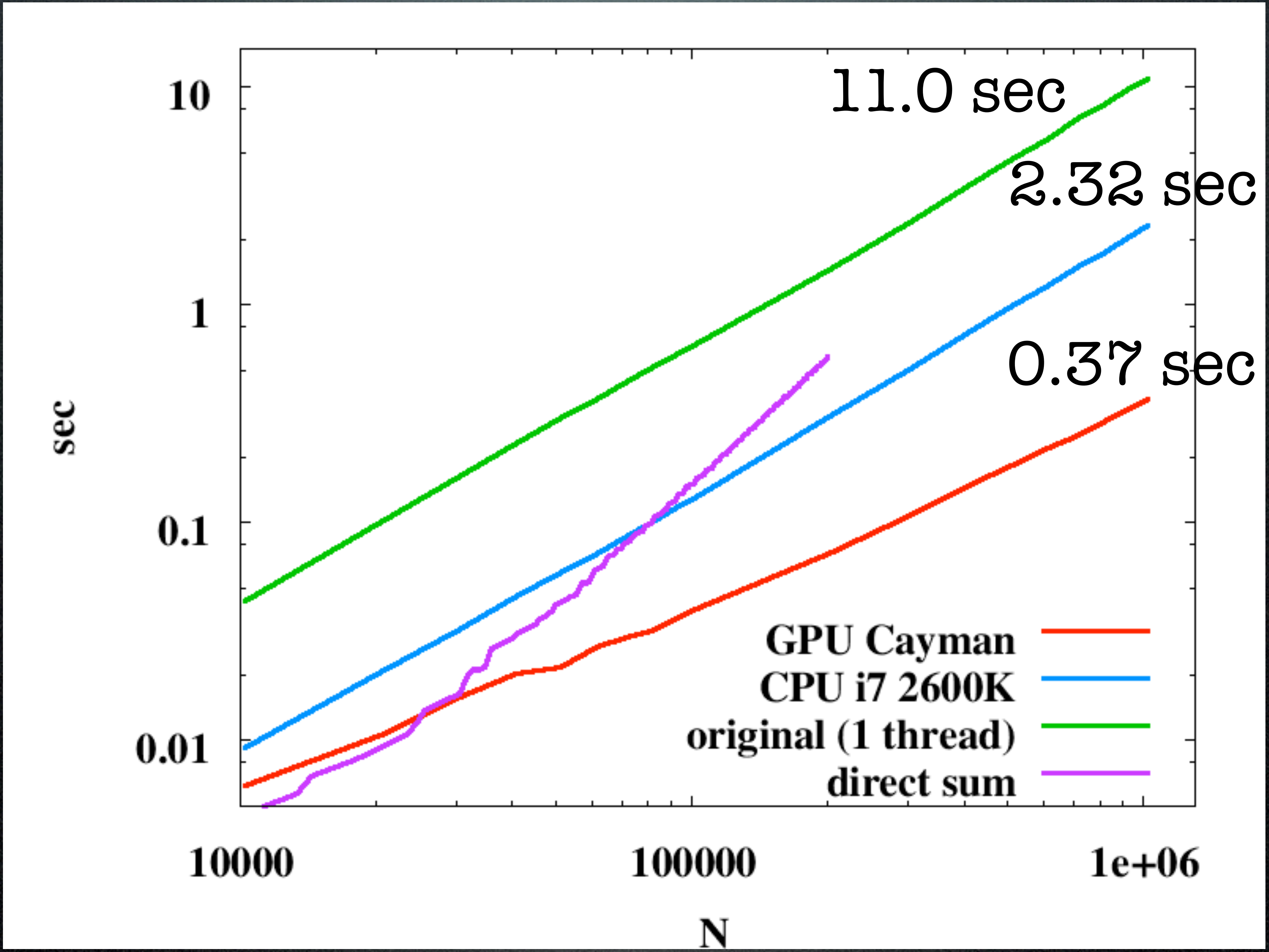
- Morton-order to preserve locality of data



Cache friendly ordering

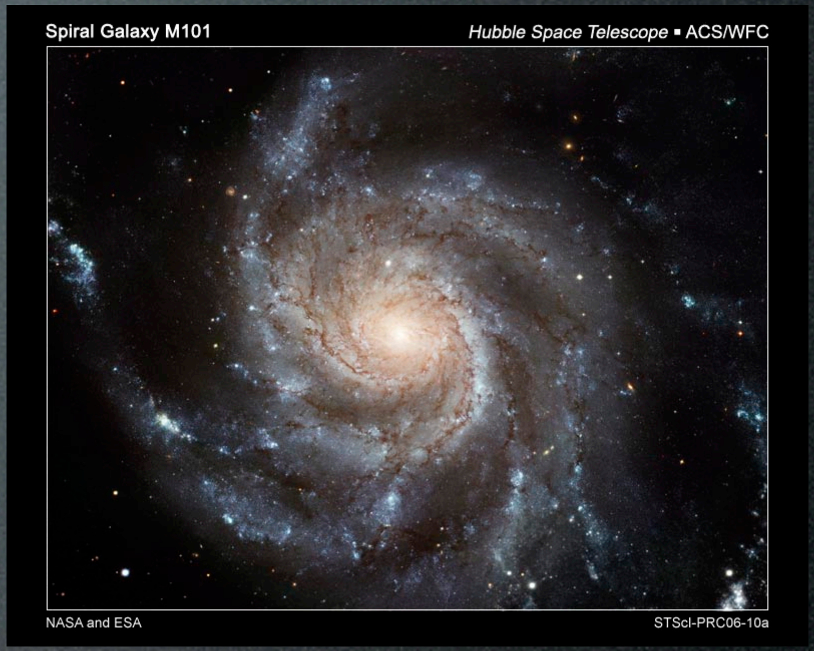


O(N²) vs. O(N log N)



Not only gravity but...

- Ingredient of a galaxy
  - DM (gravity)
  - Plasma (hydro)
  - Stars condensed from plasma (gravity)



- To model a realistic galaxy evolution
  - We couple gravity and hydro : SPH method
  - + radiative cooling + star formation + SN explosions + chemical enrichment + ...

Neighbor interaction

Application to Smoothed Particle Hydrodynamics
Solving the Euler equation with particles

$$\rho_i = \sum m_j W(\boldsymbol{r}_i - \boldsymbol{r}_j; h)$$

$$\frac{D\boldsymbol{v}_i}{Dt} = -\sum m_j (\frac{P_i}{\rho_i^2} + \frac{P_j}{\rho_j^2}) \nabla W(\boldsymbol{r}_i - \boldsymbol{r}_j; h) - (\nabla \Phi)_i.$$

$$\frac{Du_i}{Dt} = \frac{1}{2} \sum m_j (\frac{P_i}{\rho_i^2} + \frac{P_j}{\rho_j^2}) (\boldsymbol{v}_i - \boldsymbol{v}_j) \nabla W(\boldsymbol{r}_i - \boldsymbol{r}_j; h)$$

Summary

- Particle simulations in Astronomy involves wide range of timescale
- GPU is now used as replacement to special purpose systems (GRAPE)
- GPU is effective to speed-up O(N log N) the octree method
  - application to the SPH method