Particle-based simulations in Astrophysics

Jun Makino
Particle Simulator Research Team, AICS/
Earth-Life Science Institute(ELSI),
Tokyo Institute of Technology

Feb 28, 2013 **3rd AICS International Symposium**

Talk Structure

- A few words on Particle Simulator Research Team
  - Motivation
  - Mission
  - Current member
- Examples of large-scale particle-based simulations in astrophysics
- Tools we have
  - Domain Decomposition and Parallelization
  - Time Domain
- Particle-Based Hydrodynamics (mainly SPH)
  - Formulation of "Standard" SPH
  - Discontinuity
  - Solution
- Summary

A few words on Particle Simulator Research Team

- Motivation
- Mission
- Current member

Motivation

- Development of high-performance simulation software for particle-based simulations has become very difficult.
- Three reasons:
  - Advance in physical models, numerical schemes: complex schemes must be implemented
  - Need to make use of parallelism in a number of levels using rather low-level tools: nodes: MPI, cores: OpenMP, SIMD units and superscalar execution: ???
  - Need to find some way to extract reasonable performance on a machine with the multi-level memory system with decreasing bandwidth and increasing latency.
- Most of the difficulty is problem-independent.

A Common Framework for Particle-Based Simulations

Mission

- To develop such a software framework
- Ultimate goals:
  - The only thing a user need to specify is the form of the particle-particle interaction
  - Parallelization and optimization in all levels of parallelism and memory hierarchy are taken care by the framework
  - (Someday...) Performance better than the best human-written codes.

Current member

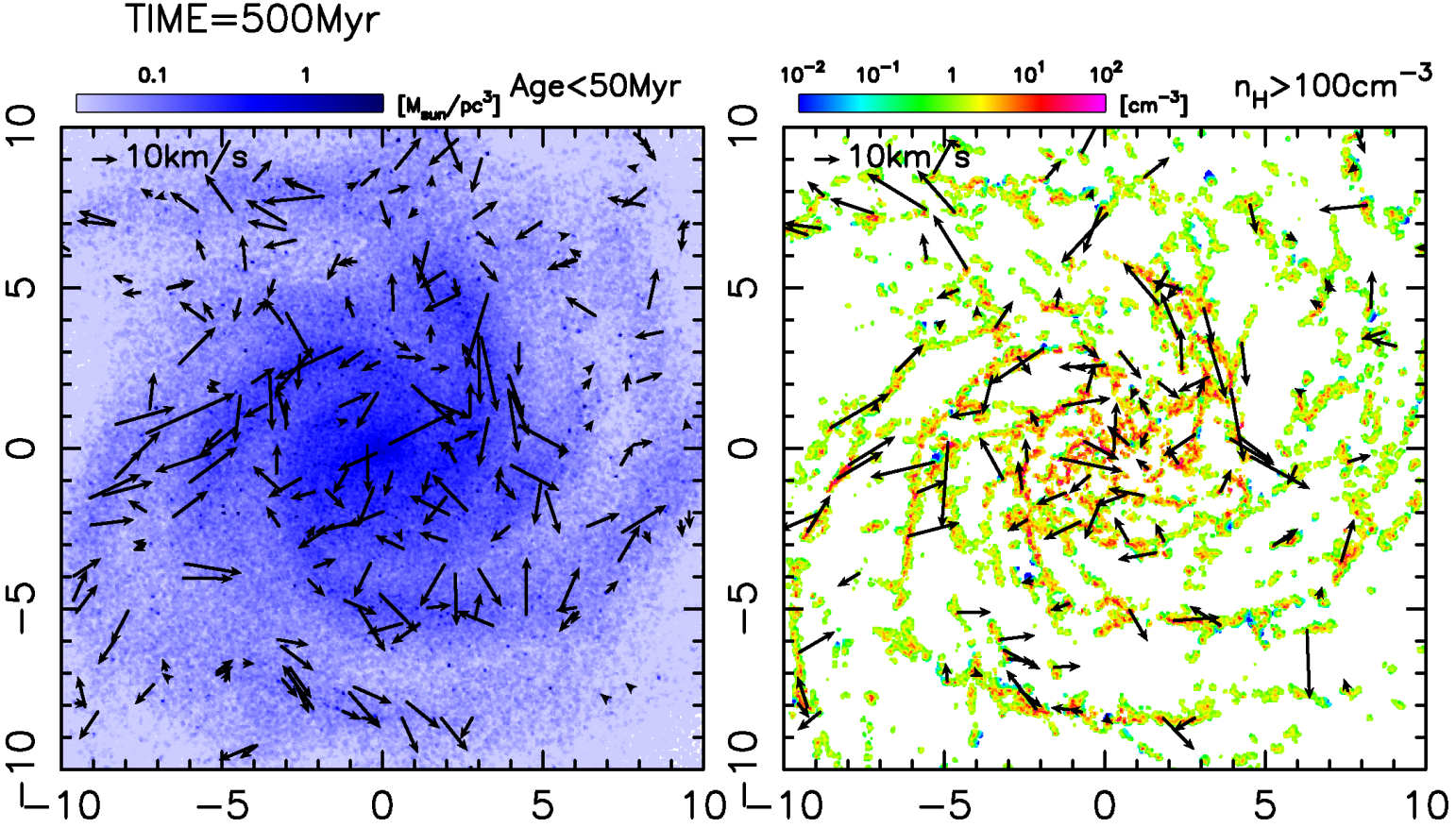
- JM
- Keigo Nitadori
  - Gordon Bell Prize winner for 2009, (2010), 2012
  - Inventor of the "Phantom-GRAPE" library, highly optimized routines for particle-particle interaction calculation
  - many other related works and achievements.
- Masaki Iwasawa: Implemented P^3T scheme (I'll describe later)
- Ataru Tanikawa: Developed an N-body code for point-mass calculation. Phantom-GRAPE for AVX, etc.
- Yuri Iida: secretary

Examples of simulation: Galactic disk

animation (Baba et al 2009) 1 2

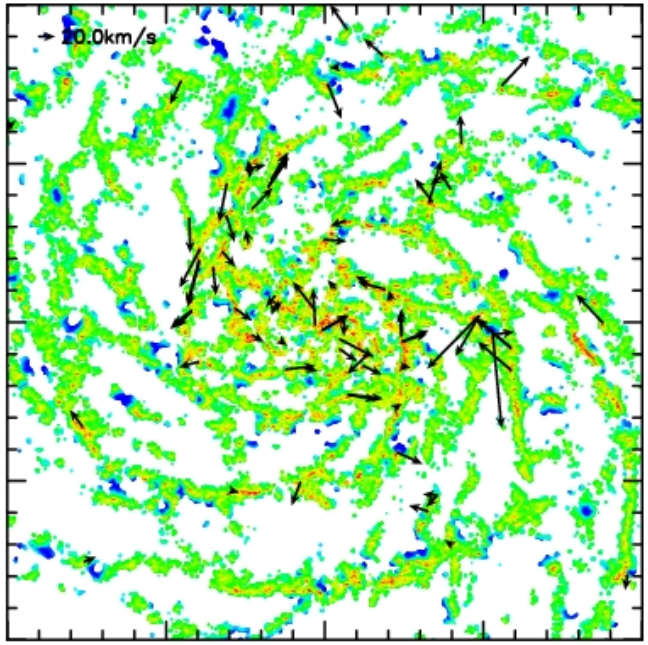
animation 1 2)

Spiral structure and deviation from the circular motion

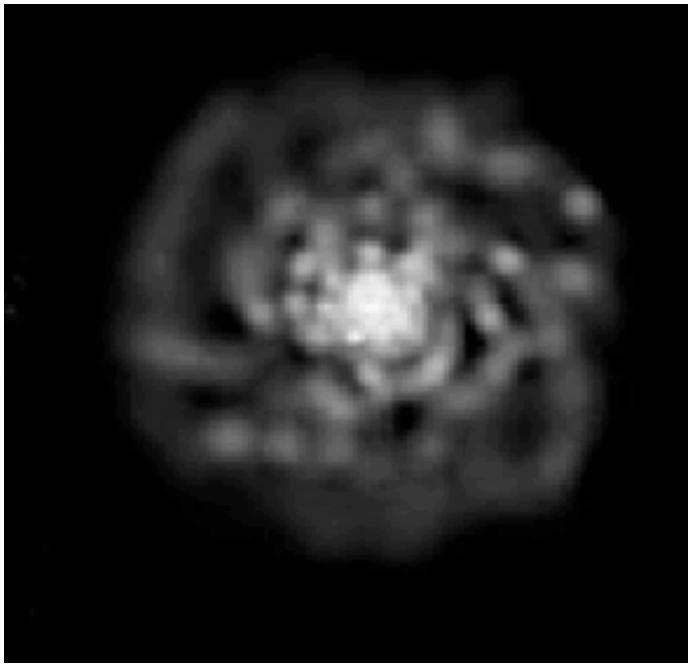


Left: distribution of stars Right: cold gas

High-resolution model and observation



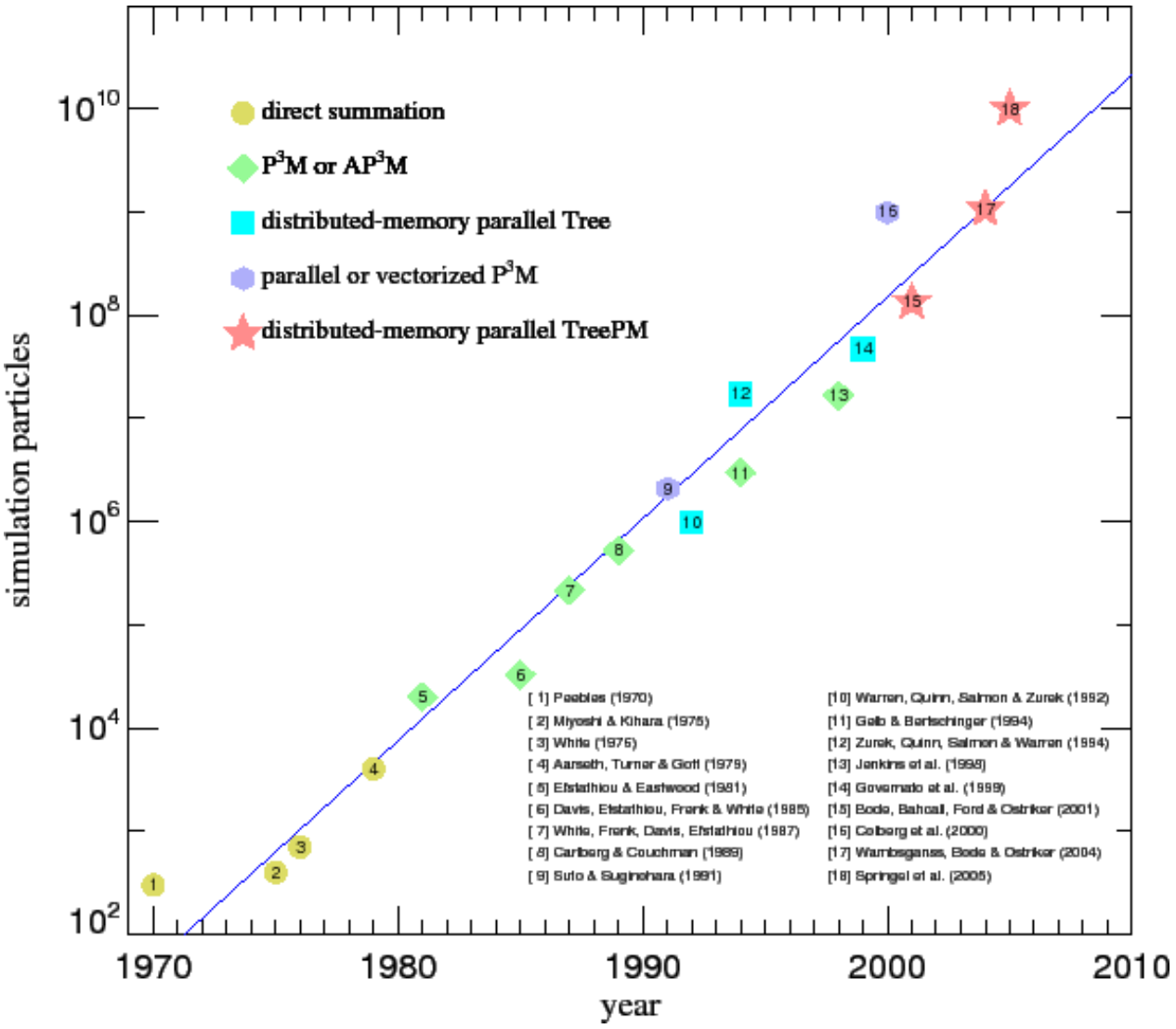
Low-resolution model and observation



Algorithms and Implementations

- Interaction calculation
- Parallelization
- Time domain — individual timestep and Particle-Particle Particle-Tree method

History of the number of particles

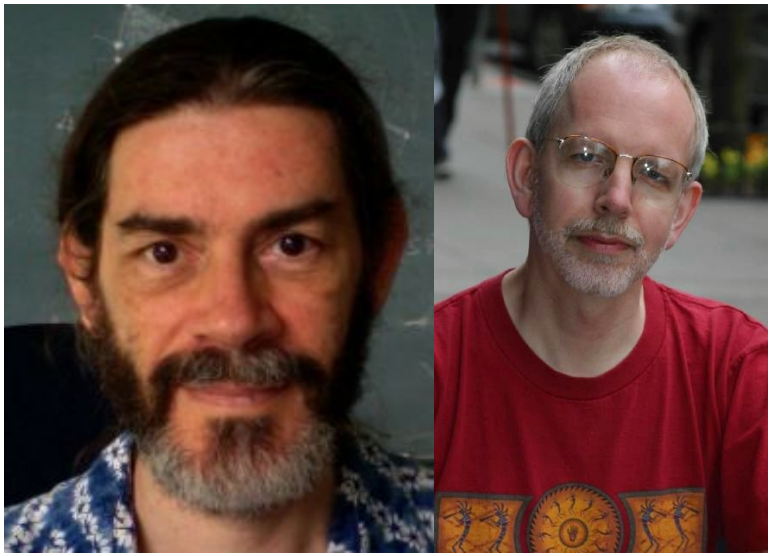


Roughly 100 times in every decade

Calculation cost $\sim O(N)$

How do we calculate gravity?

- A straightforward approach requires $O(N^2)$ operations
- Almost all simulations after 1990 used treecode
- Barnes-Hut tree invented in 1985.



Barnes and Hut

Piet Hut in this summer

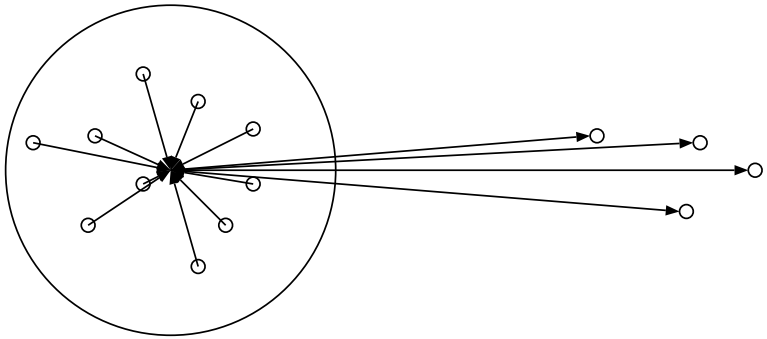
At the banquet of MODEST-12

Basic idea for tree method and FMM

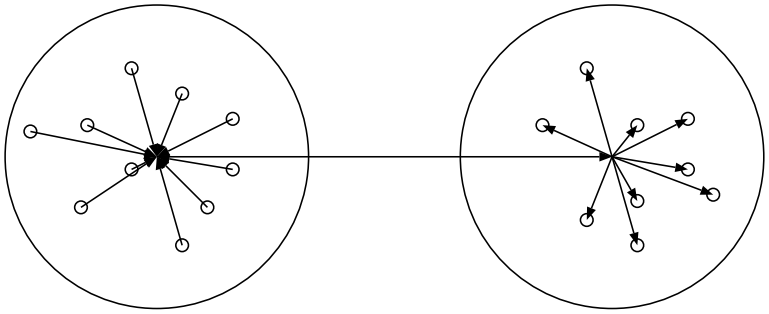
Force from distant particle: Weak

↓

Can't we evaluate many forces at once?



Tree



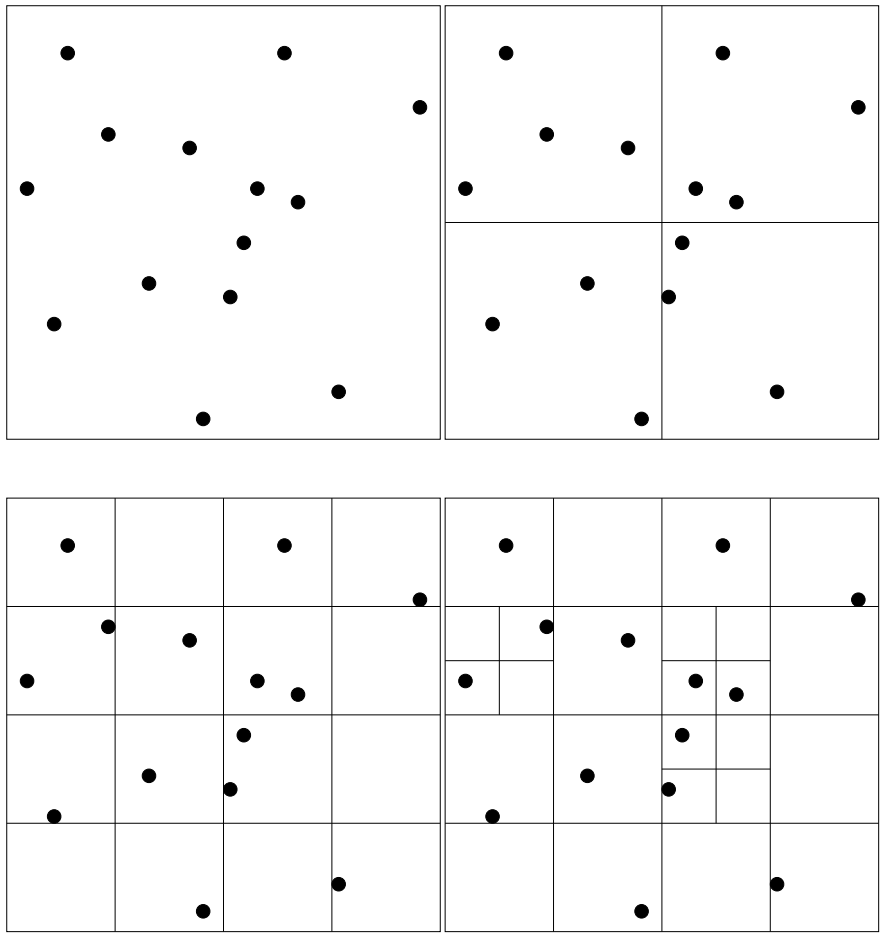
FMM

- **Tree: aggregate stars which exert the forces**
- **FMM: aggregate both side**

How do we aggregate — Barnes-Hut tree

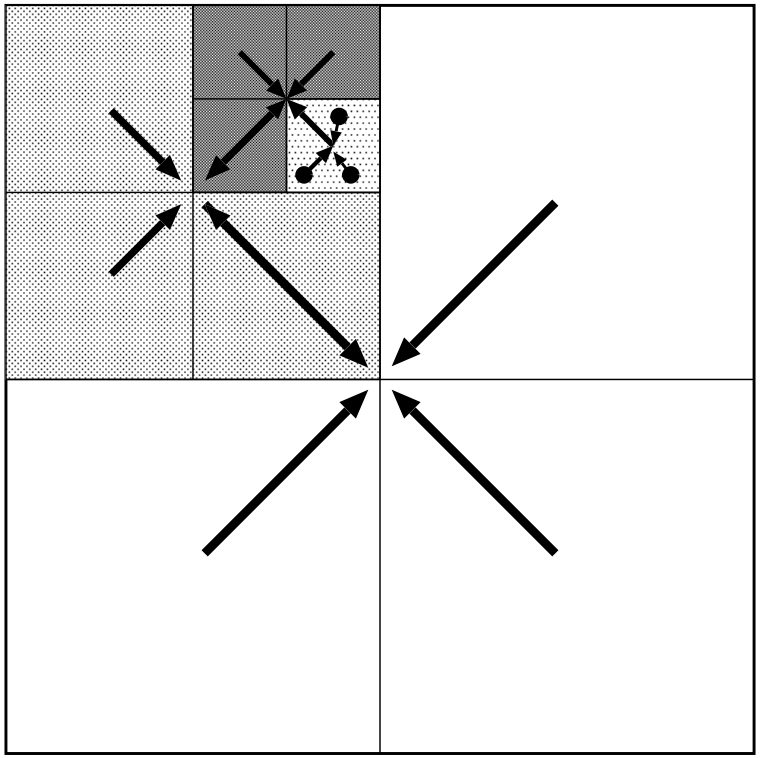
Use tree structure

- First make a cell with all stars in it
- Recursively subdivide the cells to 8 subcells
- Stop if there is small enough stars



Construction of the multipole expansion

Form the expansion for cells.

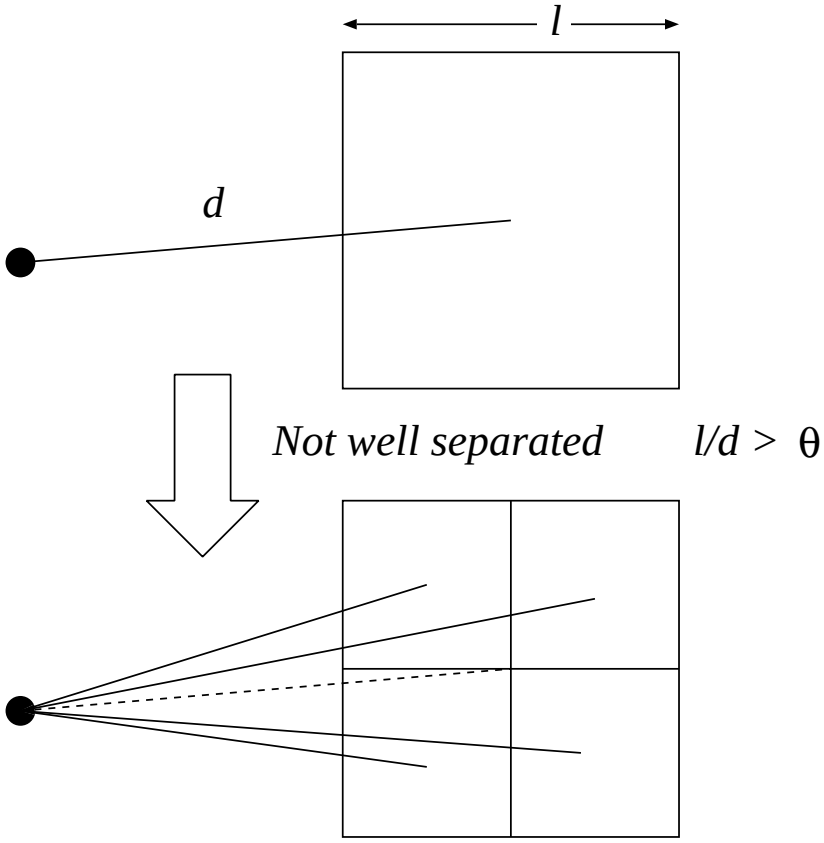


- lowest-level cells: Directly calculate the expansions for stars in it.
- Higher-level cells: Shift and add the expansions for child cells.

Calculate bottom-up.
Calculation cost: $O(Np^4)$ (p: expansion order)

Force calculation in tree method

Recursive expression:



- **Well separated: apply the multipole expansion**
- **not: take summation of the forces from the child cells**

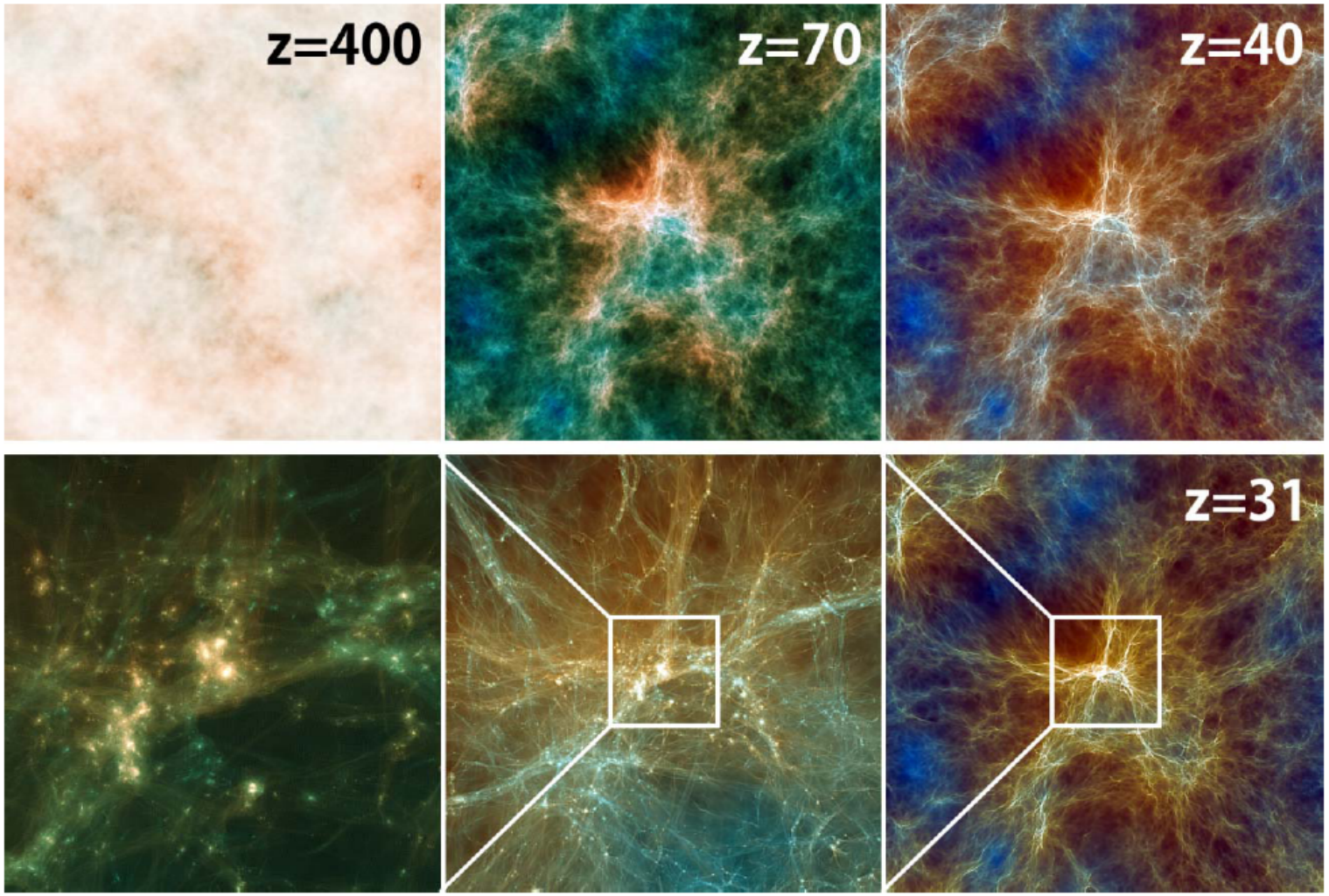
Total force = force from the root cell

The Effect of Tree Method

- Order of the calculation cost reduced from $O(N^2)$ to $O(N \log N)$
- Cray XT4 1024 cores: 2048^3 particles/ several minutes
- K full system: 10240^3 particles/30 sec
- Direct method would take > 100 years/step
- Calculation cost insensitive to the spacial structure

Other fast methods (PME, P^3M) become costly when inhomogeneity develops

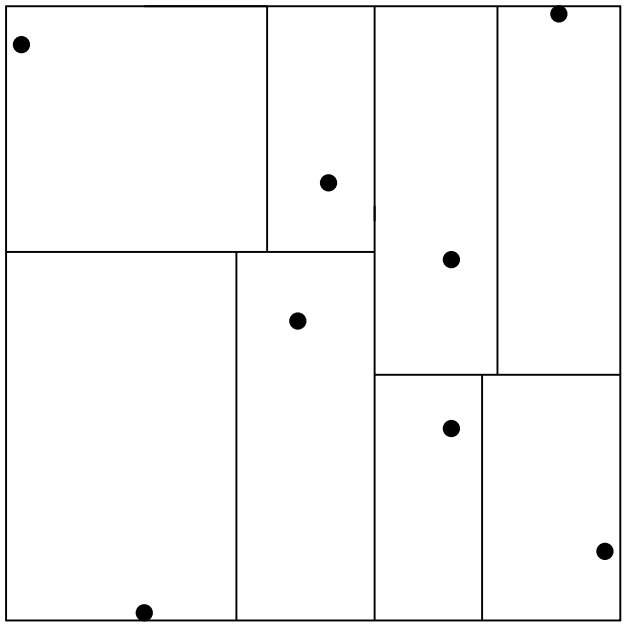
Example of the inhomogeneity



Parallelization

Two known and well-studied methods, both first implemented by Salmon and Warren(Caltech Hypercube group)

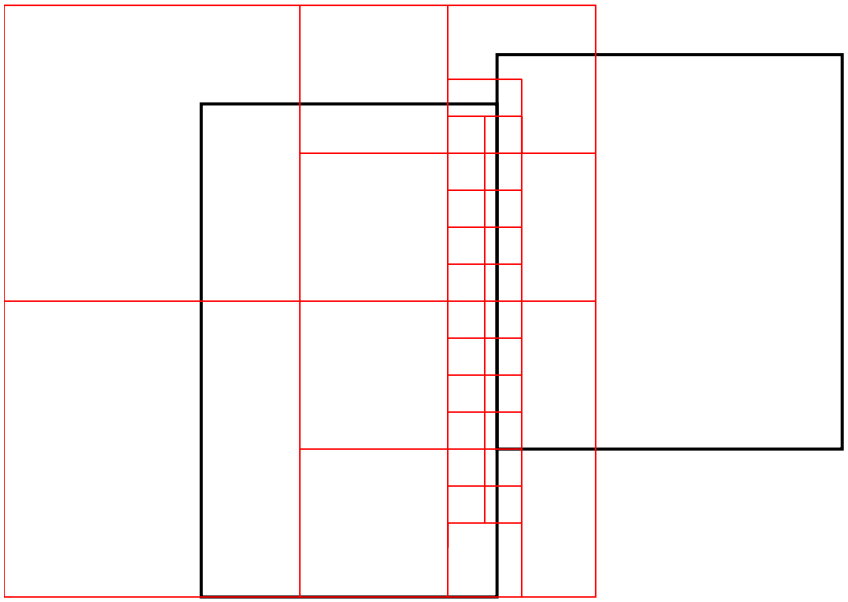
- **Orthogonal Recursive Bysection (ORB)**
- **Hashed Oct Tree (HOT)**



ORB

- Divide the system by a plane perpendicular to x axis (each has same number of particles)
- then do the same thing for y, z, x,... directions, until the number of cells reaches the number of processors

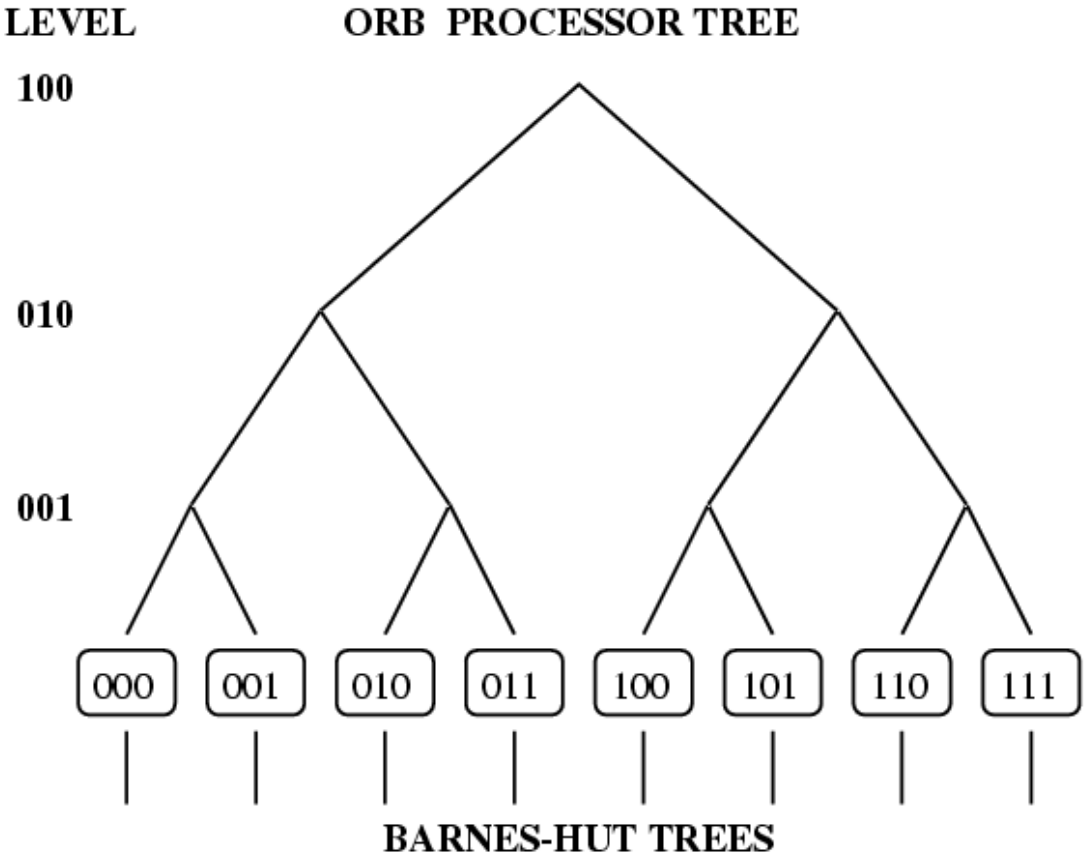
Force from particles in other processors



Get the trees with "unnecessary branches" cut off from other processors (local essential tree, LET) Construct the global tree by combining them with its own tree.

How to combine?

Dubinski: Upper structure is the ORB tree

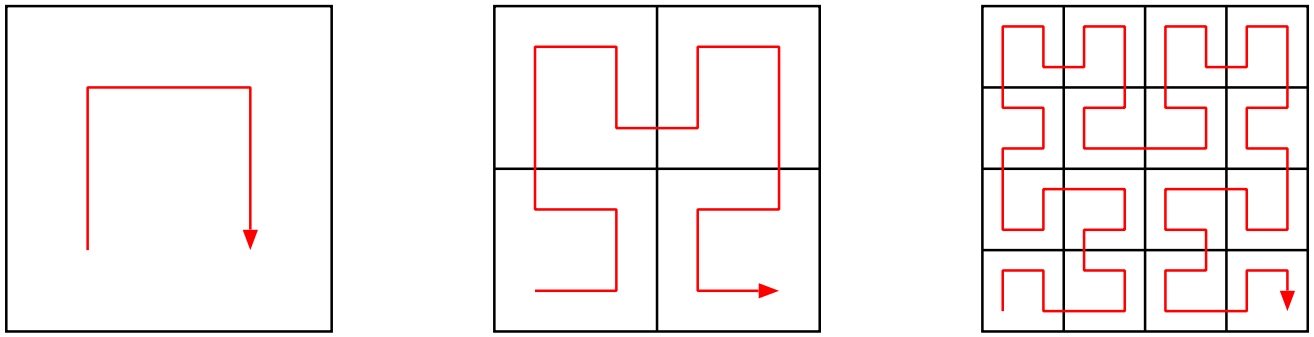


Problems with ORB tree

- Complex implementation
  - Different tree structures for the ORB tree and local tree
  - LET should be transferred maintaining the tree structure
- Poor scalability
  - Communication proportional to the number of processors
- Calculation result depends on the number of processors (within the tree accuracy, but...)

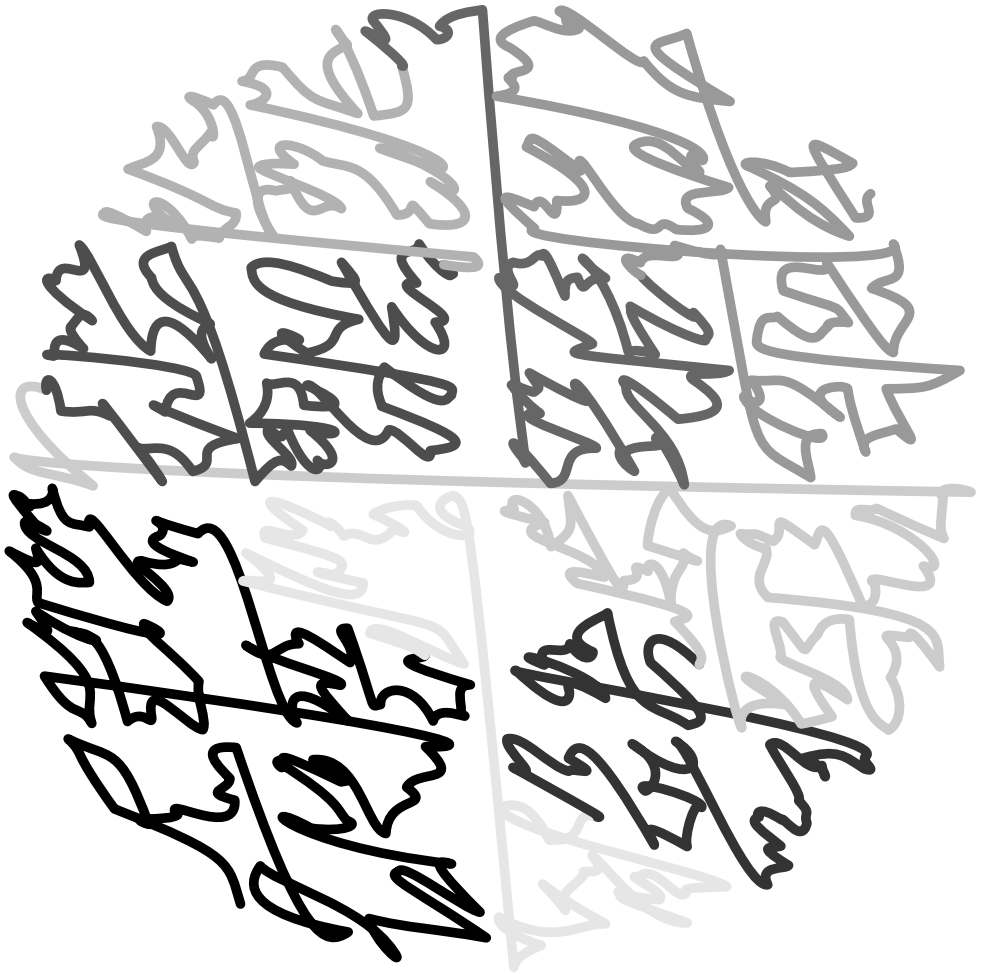
HOT

Peano-Hilbert curve



- Order particles on the Peano-Hilbert curve
- Assign contiguous particles to each processors

HOT



(This one uses Morton Ordering)

Tree construction and interaction calculation with HOT

Tree construction

- Assign Peano key to each particle
- Perform global parallel sort

Interaction calculation

- On-demand communication: Request necessary data to other processors

Fairly sophisticated message combining, async operation of calculation and communication, delayed evaluation etc...

Our approach

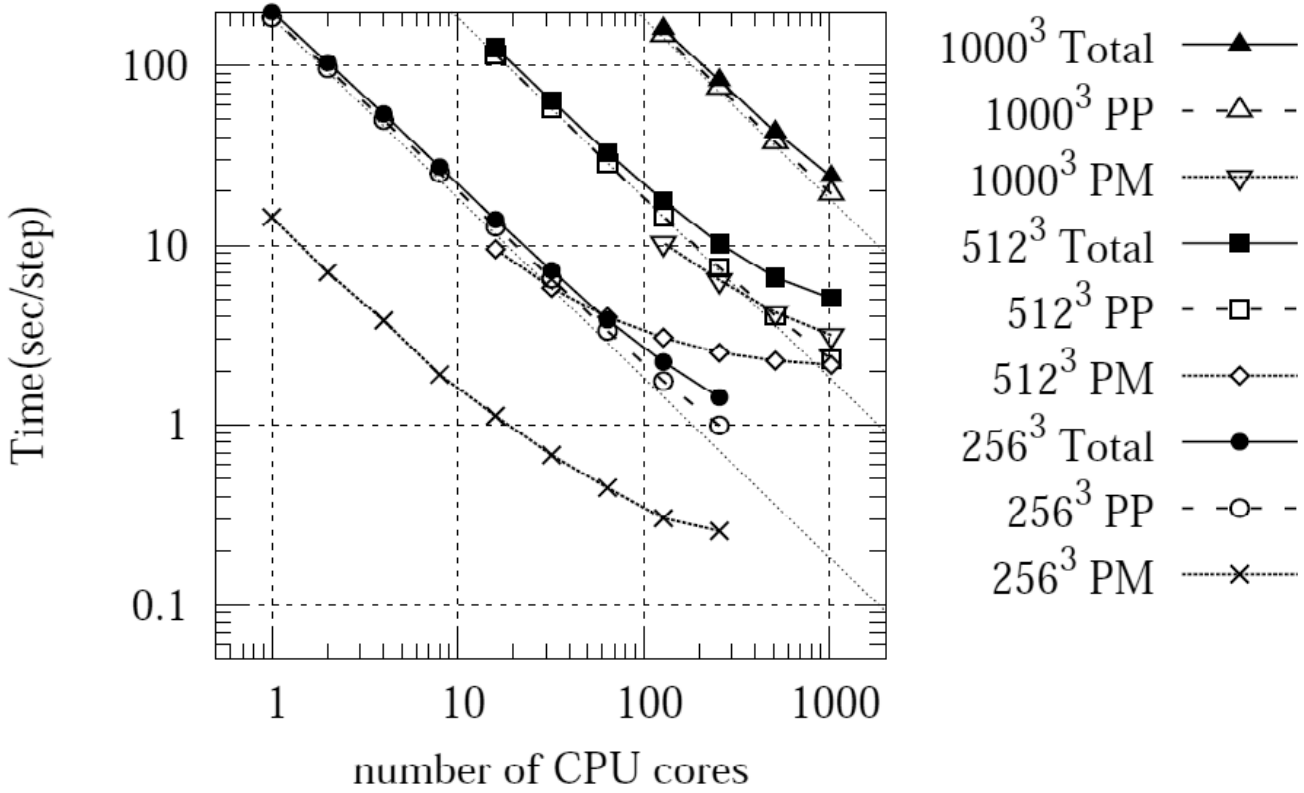
(Makino 2004, Ishiyama et al 2009, 2012)
Modify ORB in two ways

- **Limit the depth to three**
- **Allow divisions to more than two cells**

1000 nodes: $10 \times 10 \times 10$. For 2, 4, 8 nodes, The same as traditional ORB.
In principle can be used even on prime numbers of nodes.
Happens to be ideal for K-computer with "54" nodes in y dimension...

Parallel performance

(Ishiyama et al. 2009, TreePM)



Scaling is OK if we have 10^4-10^5particles/core

Performance on the K computer

Ishiyama et al 2012 (SC12, Gordon Bell winner)

- **10240^3 particles**
- **82944 nodes (full system)**
- **27 sec/step**
- **5.67 Pflops**

"Similar" code on BG/Q (another Gordon Bell finalist): 14PF, but 70 sec/step for the same problem size.

Individual timestep

For many problems, we use individually variable timesteps for particles.

- Different particles can have different time and timestep
- Each particle can change its timestep at any time

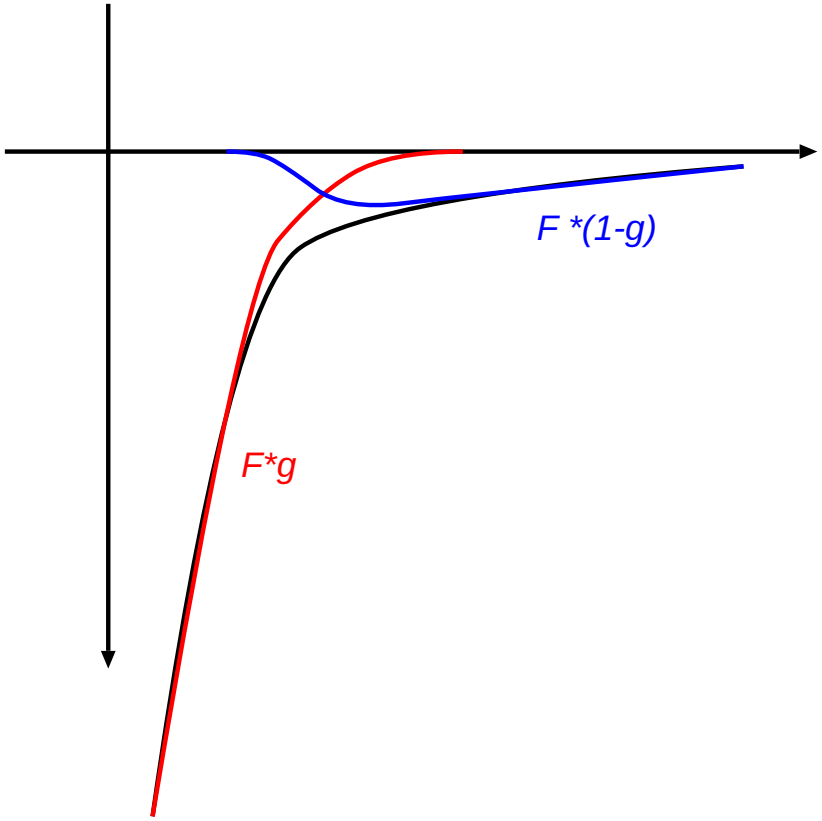
Great in reducing the calculation cost, but

- Not an ideal way to use large-scale parallel computers
- Hard to combine with tree or FMM (they need to calculate forces on many particles at one time)

Particle-Particle Particle-Tree

Divide the pairwize interaction into near- and far- terms

$$\boldsymbol{F_{ij} = -Gm_i m_j \frac{r_{ij}}{|r_{ij}|^3} = F_{ij}(1 - g(|r_{ij}|) + F_{ij} g(|r_{ij}|)}$$



- $\boldsymbol{F * g}$**+kinetic energy integrated wit individual variable timestep**
- $\boldsymbol{F * (1 - g)}$ **tree + constant timestep**

Same idea as P^{3}M, PME Oshino et al. 2011, PASJ, 63, 881-.

Seems to work extremely well.

Particle-Based Hydrodynamics: SPH and its problems

Advantages of particle-based method for fluid

- Naturally adaptive (particles moves to where the mass is there)
- Naturally gives Lagrange picture. Useful for low-temperature, high-speed objects
- Parallelization fairly easy

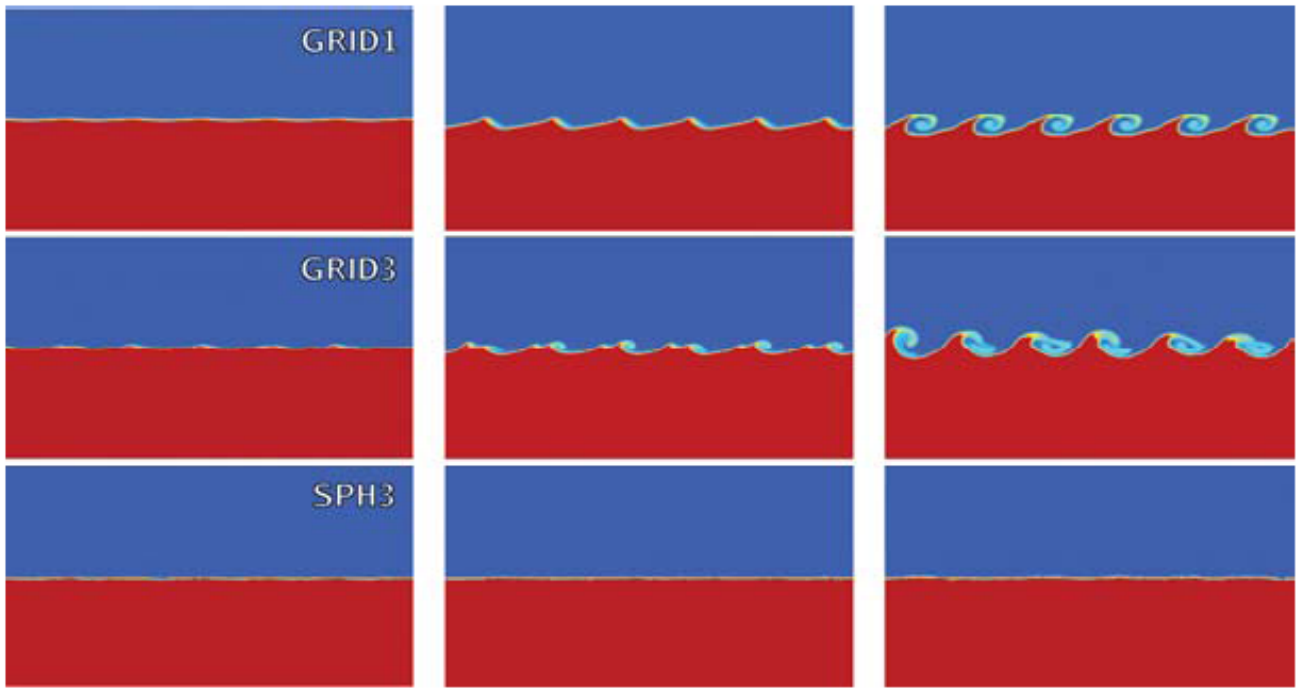
However, there are quite a few problems...

SPH and Contact Discontinuity, KH instability

Agertz et al (MN 2007, 380, 963)

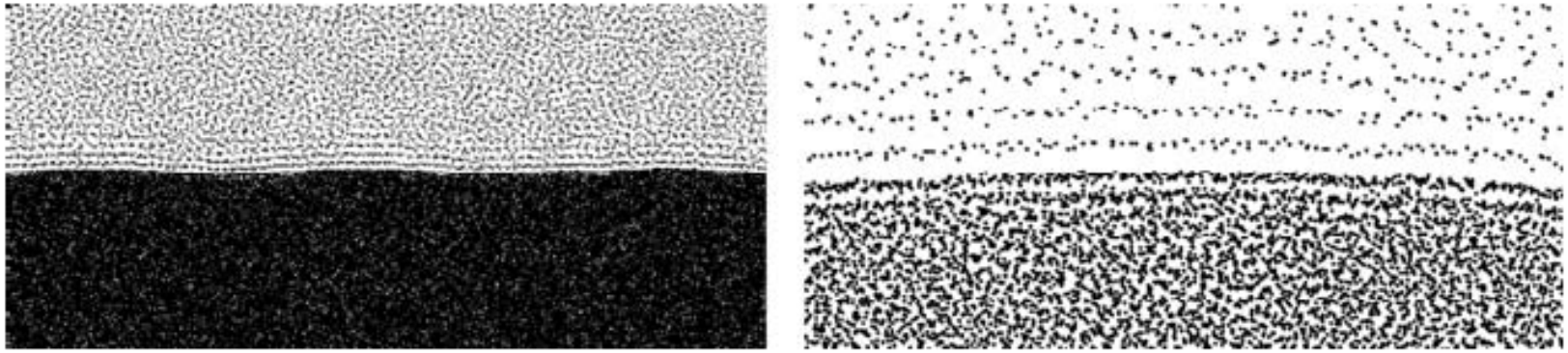
- **Kelvin-Helmholtz Instability is not correctly handled with SPH**
- **Something very bad is occuring at the boundary of two fluids.**
- **Is SPH usable?**

How different? (1)



SPH suppress KHI

How different? (2)



Strange-looking gap of particles at the two-fluid boundary.

Why does this happen?

Fundamental problem with SPH approximation

Density estimate

$$\rho(x) = \sum_j m_j W(x - x_j), \tag{1}$$

Gradient of a quantity f

$$\langle \nabla f \rangle(x) \sim \sum_j m_j \frac{f(x_j)}{\rho(x_j)} \nabla W(x - x_j). \tag{2}$$

ρ must be smooth and differentiable

Not satisfied at the contact discontinuity

Solution?

"Fundamental" reason

Smooth estimate of ρ contains $O(1)$ error at CD.

We could solve the problem by smoothing real ρ.

- **Let u diffuse (artificial conductivity)**
- **Use density which is continuous at CD.**

Sort of working, but not a "true" solution.

Our proposal: Basic idea

- **In SPH, we use m/ρ as "volume element" for numerical integration over smoothing kernel**

$$\langle f \rangle(x) = \sum_j \frac{m_j f(x_j)}{\rho(x_j)} W(x - x_j). \tag{3}$$

- **We can use other forms of the volume element and derive a consistent set of SPH equations.**

Pressure-based SPH

Use $(\gamma-1)U/p$ as the volume element, where U is the internal energy of particle.

SPH equation of motion:

$$m_i \dot{v}_i = -\sum_j (\gamma-1) U_i U_j \left(\frac{1}{q_i} + \frac{1}{q_j} \right) \nabla W(x_i - x_j). \quad (4)$$

- **RHS does not depend on mass**
- **This form is symmetric (between i and j particles)**

Examples

Standard SPH1
New SPH1
Standard SPH2
New SPH2

Is this the ultimate solution?

- Maybe not...
- Not well-behaved at very strong shocks
- Breaks down at liquid surface (near-zero pressure)

A "natural" volume element

One way to define the "volume" of a particle, V_i, in the context of SPH is to use the following implicit equation:

$$\sum_j V_j W(x_i - x_j) = 1. \tag{5}$$

Equation of motion:

$$m_i \dot{v}_i = -\sum_j V_i V_j (p_i + p_j) \nabla W(x_i - x_j). \tag{6}$$

Time evolution of V_i:

$$\dot{V}_i = -V_i \sum_j V_j (v_i - v_j) \nabla W(x_i - x_j). \tag{7}$$

Eq. (5) is implicit, but Eq. (7) is explicit. We do not need to solve the implicit equation during the time integration.

We are currently testing schemes of this type. Results seem promising.

Summary

- Efficient highly-parallel algorithms for particle-based simulations are now available.
- We can now handle long-range interaction of trillion particles on the K-computer, with very high overall efficiency.
- We can combine schemes for problems with wide range of spacial scale and timescale with fast parallel algorithms.
- We hope to develop a "general-purpose" framework for particle-based simulation, based on these algorithms.
- Particle-based hydrodynamics is widely used, but there are still many rooms for improvements. We are working on some of them.

Our Implementation

- Do not send LET. Send only leaf nodes ("particles") of LET
- Insert these "particles" to the local tree (JM's code. Ishiyama et al. uses a bit different approach)

Insertion method: The method used in Barnes' original tree code.