Octree for Particle Simulations On Heterogeneous Computers

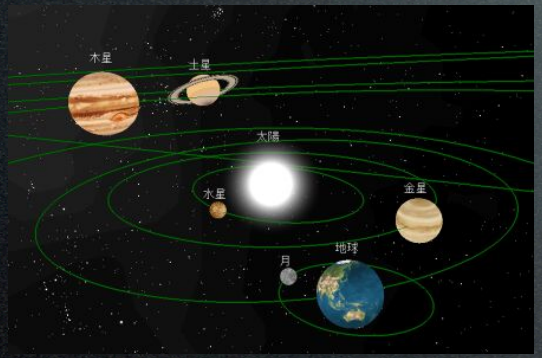
N.Nakasato
University of Aizu & University of Tsukuba

Agenda

- Introduction to particle simulation in astronomy
- Octree implementation for heterogeneous computers
  - Application to SPH method
- Summary

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

$N \sim 10^{11}$

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

Numerical Models in the Universe

solar system

sun&planets

$N \sim 10$

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

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

star cluster

individual stars

$N \sim 10^5$

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

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

We have two classes of problems

1. Small N
2. Lager N

Physical condition

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

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

galaxy

blob of stars&DM

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

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

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

whole universe

blob of DM

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

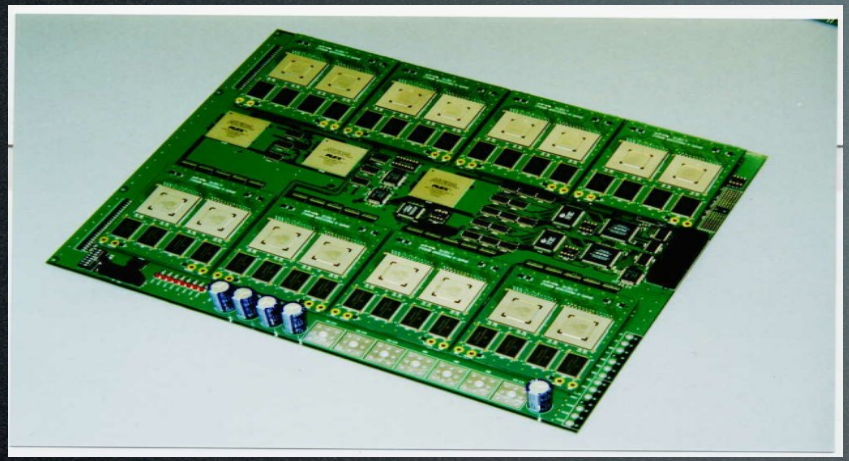
$t_{\rm lifetime} \sim 10^{10}{\rm yr}$

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

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 – 90% See HC-4162 for details
O(N log N)	Octree method	1 - 2%
O(N)	Explicit Hydro code	not high in principle



GRAPE-6A (2002)

- fixed function
- 30 Gflops (90MHz)
- 10W
- 200 Myen (2.4 Tflops)
- super energy efficient

GPU (AMD Cypress, 2010)

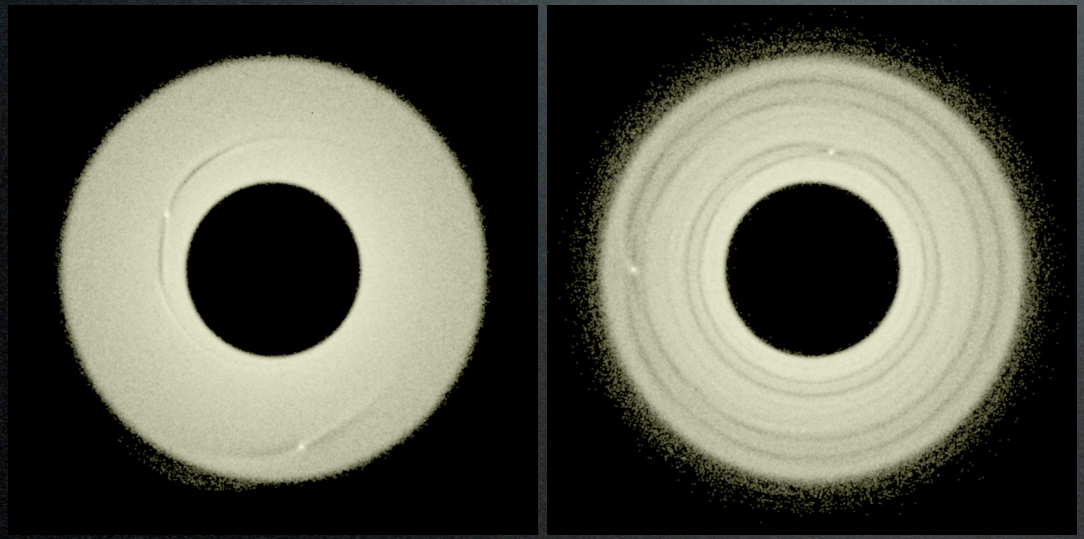
- programmable
- 600 Gflops (850MHz)
- 200W
- 40,000 yen
- highly cost effective

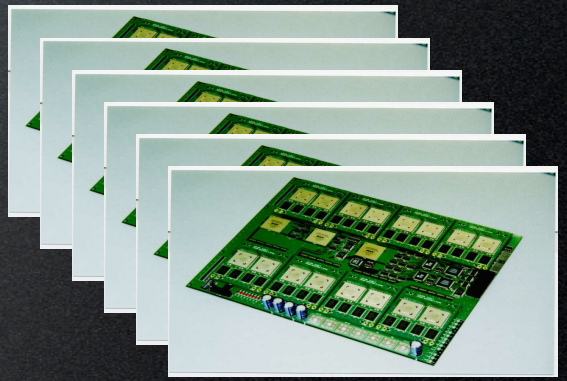
Collisional Particle System

computational complexity $O(N^2)$

direct summation

need high accuracy

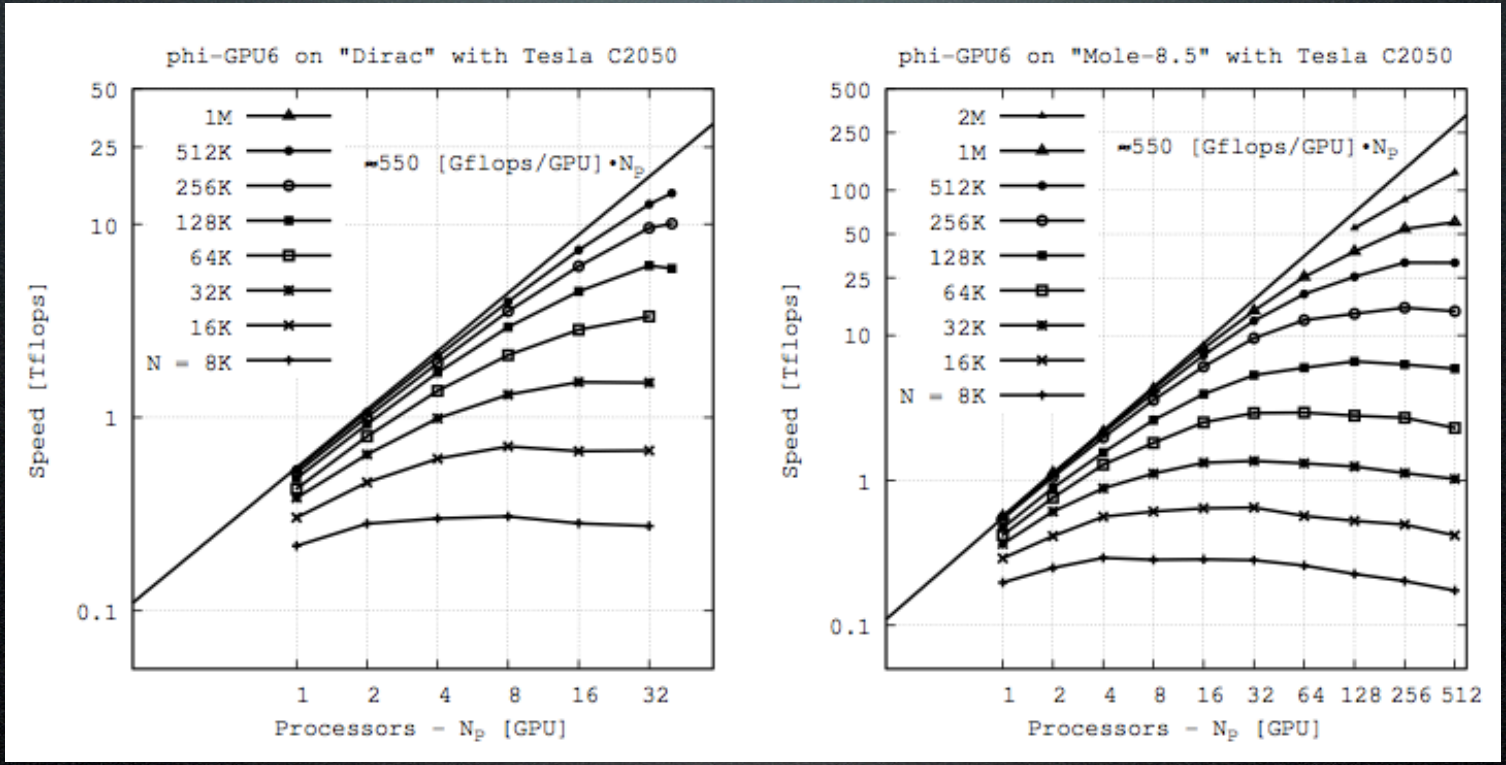




A model of Saturn's ring : Makino etal. 2002

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

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

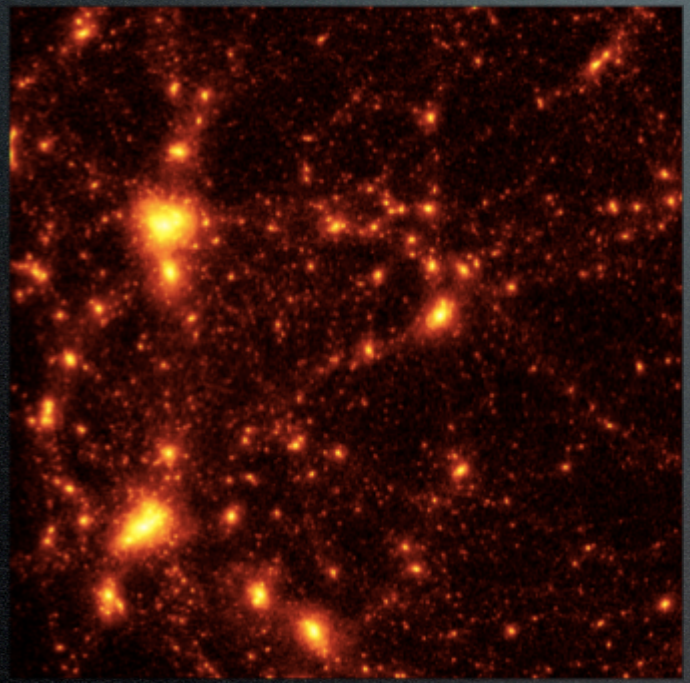
We rely on approximation methods to computer long-range force : Octree method (Barnes&Hut 1986)

– Systematically replace distant particles with multipole-moment(MM) of the particles

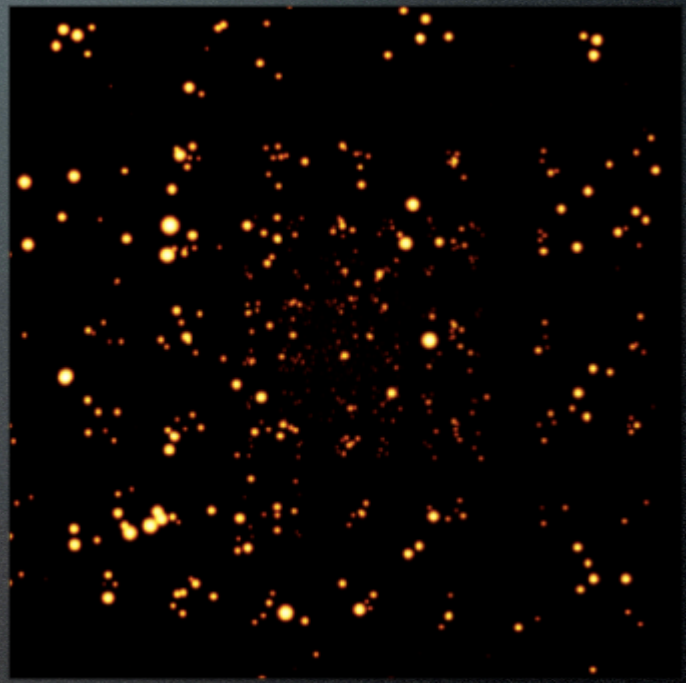
N.Nakasato, Journal of Computational Science, 2011
doi:10.1016/j.jocs.2011.01.006

Reduction of Computing

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



$N \sim 10^6$ distribution



$N \sim 10^3$ particles&nodes

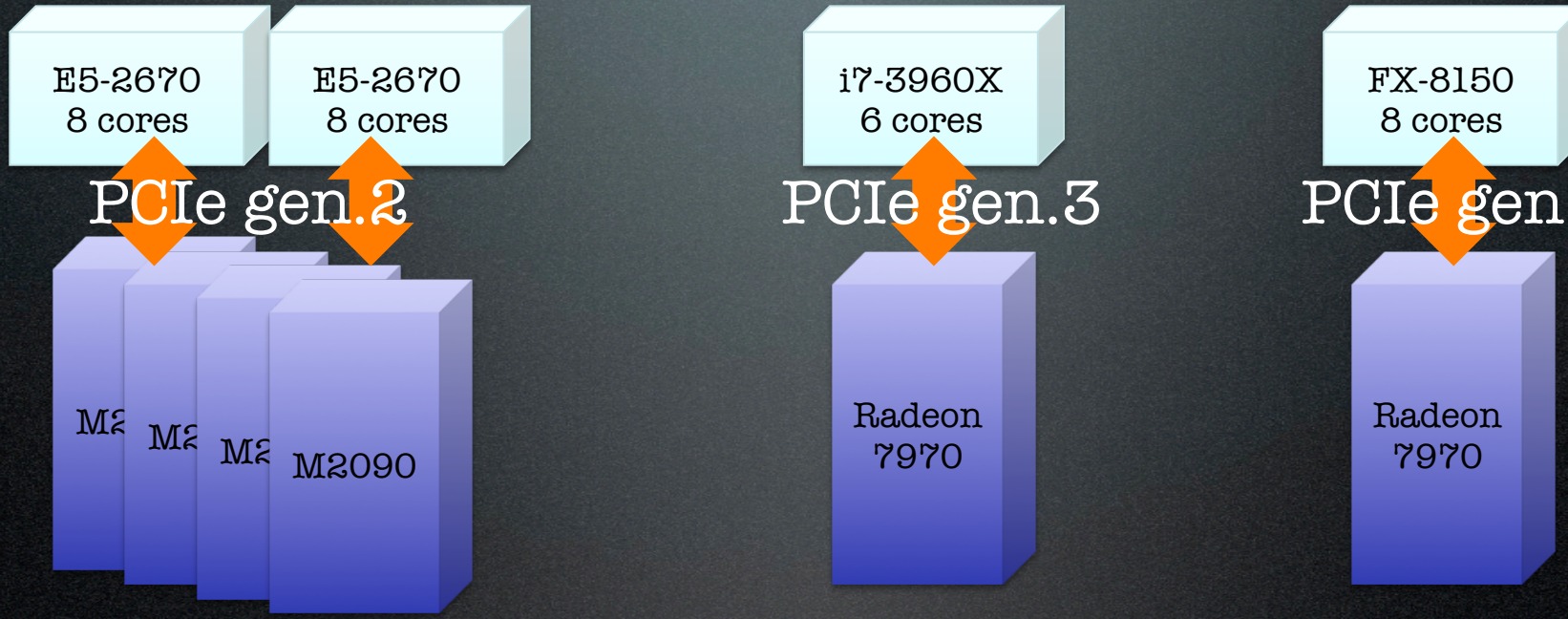
Flow of Octree Code

1. Construction of tree data
    1. Compute “keys”, sort them and “connect” nodes
2. Compute multipole moments
    1. Center of Mass etc...
3. For each particle (most time consuming)
    1. Walk the tree and check the opening-criterion
    2. Either compute the force or further walking the tree

Design Consideration

- We need to consider heterogeneous nature of a compute node for our applications
  - Relative performance of CPU vs. GPU
    - Scalar vs. Vector ratio in vector architecture
  - Memory size
  - Bus technology
  - Balanced Interconnect

Our Heterogeneous Systems



2090HAP (HA-PACS)
at Tsukuba

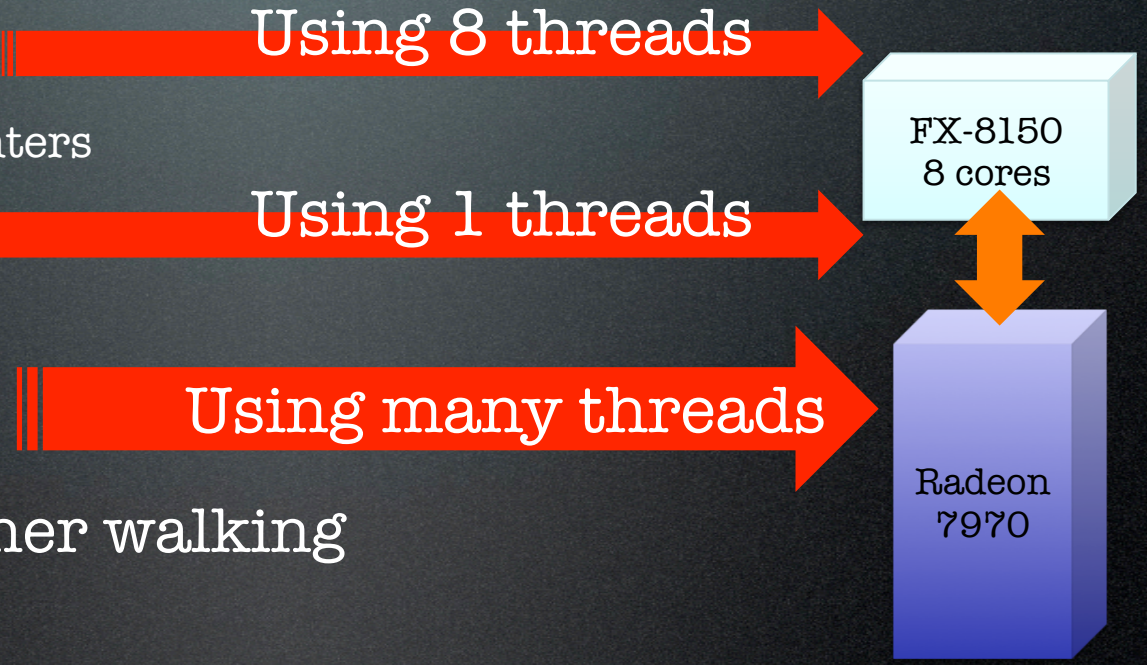
7970SB

7970BD

at Aizu

Work Distribution

1. Tree construction
    1. Sorting & Linking pointers
2. Compute MM
3. For each particle
    1. Walking the tree
    2. Compute or further walking



Other Proposals

- Bedorf, Gaburov, Portegies Zwart (2012)
  - Implement entire octree code in CUDA
    - No need for communications back and forth
  - Drawback is…
    - Tree constcution require atomic operations
    - Paralle tree on GPU is not so effective
    - Tree traversal is based on a stack
  - Offload everything is not always effective

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

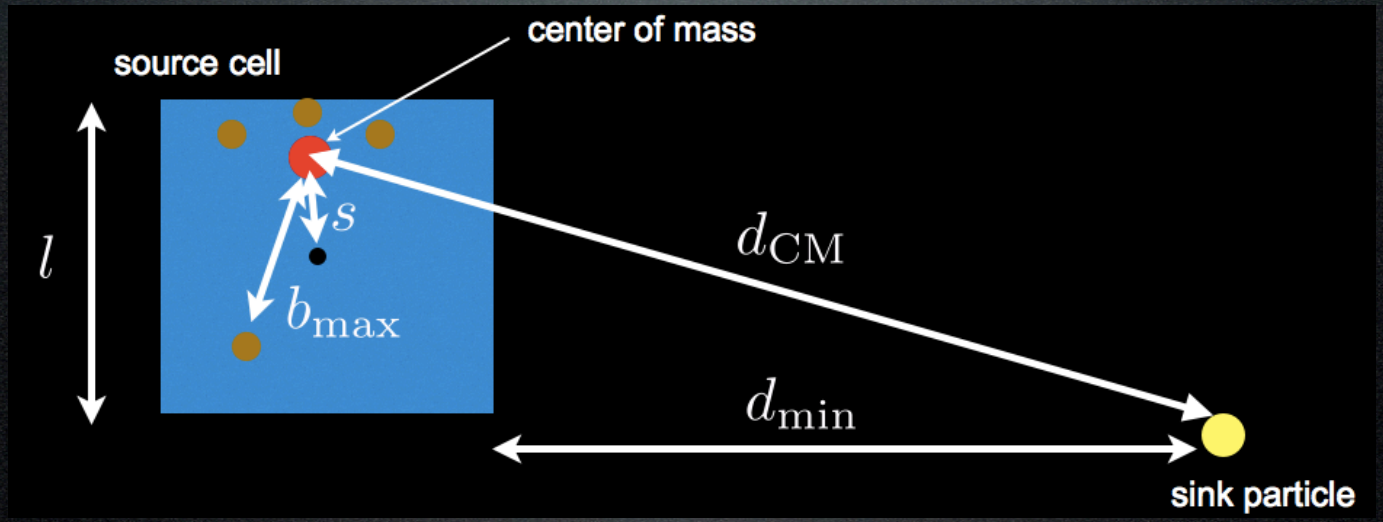
Name	CPU	GPU	PCIe	SP perf.	SDK
2090HAP	dual Xeon E5-2670	M2090 x 4	Gen.2.0 x16	5320	NVIDIA
7970SB	Core i7-3960X	HD7970	Gen.3.0 x16	3789	AMD
7970BD	FX-8150	HD7970	Gen.2.0 x16	3789	AMD
APU	A6-3650	HD6530D	–	284	AMD
HAP	dual Xeon E5-2670	–	–	666	Intel
OPT	dual Opteron 6168	–	–	364	AMD
SANDY	Core i7-3960X	–	–	316	Intel
BD	FX-8150	–	–	230	AMD

Optimization (1) : Construction

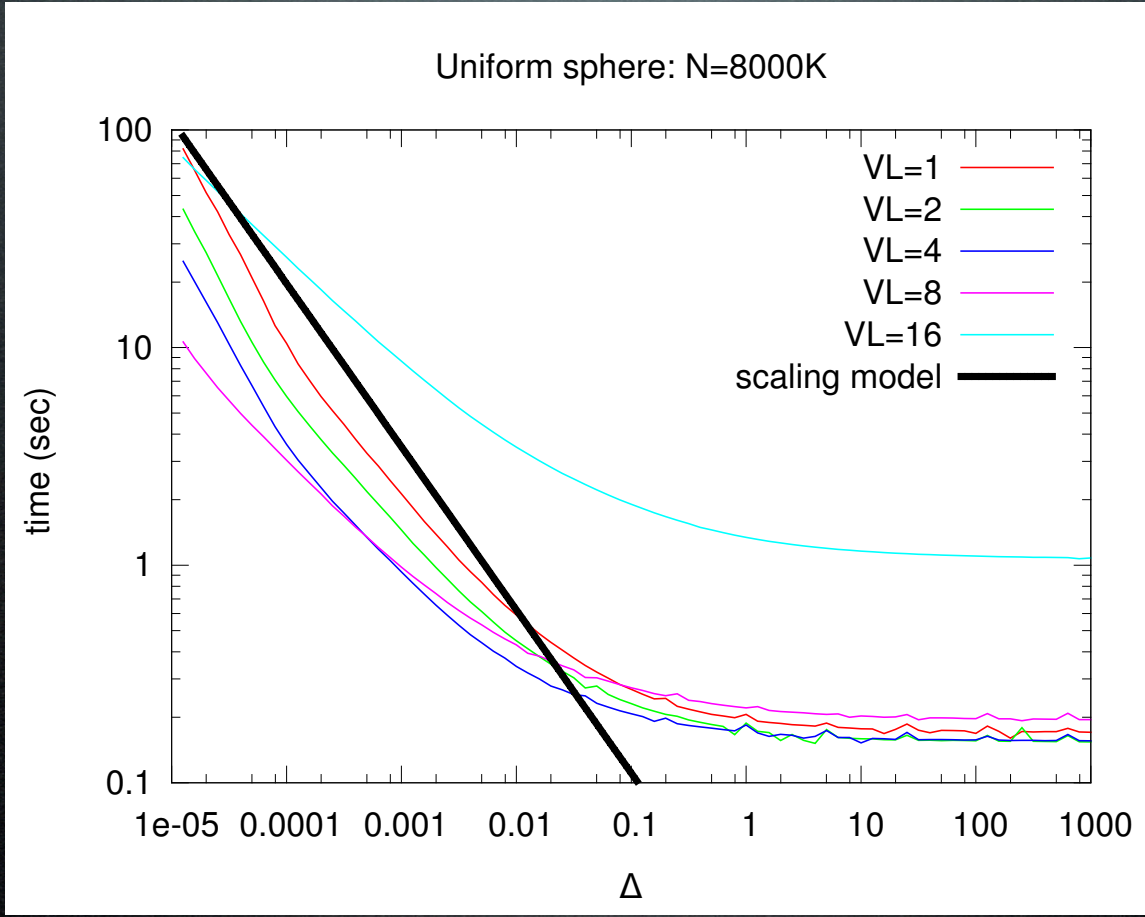
- Create linked-list data structure on CPU
  - Dilated integer
  - Conversion of the Morton key to the PH key
  - Parallel sorting on CPU
  - Computation of the center of mass and multipole moments
  - All in parallel with OpenMP directives

Optimization (2) : Vectorized

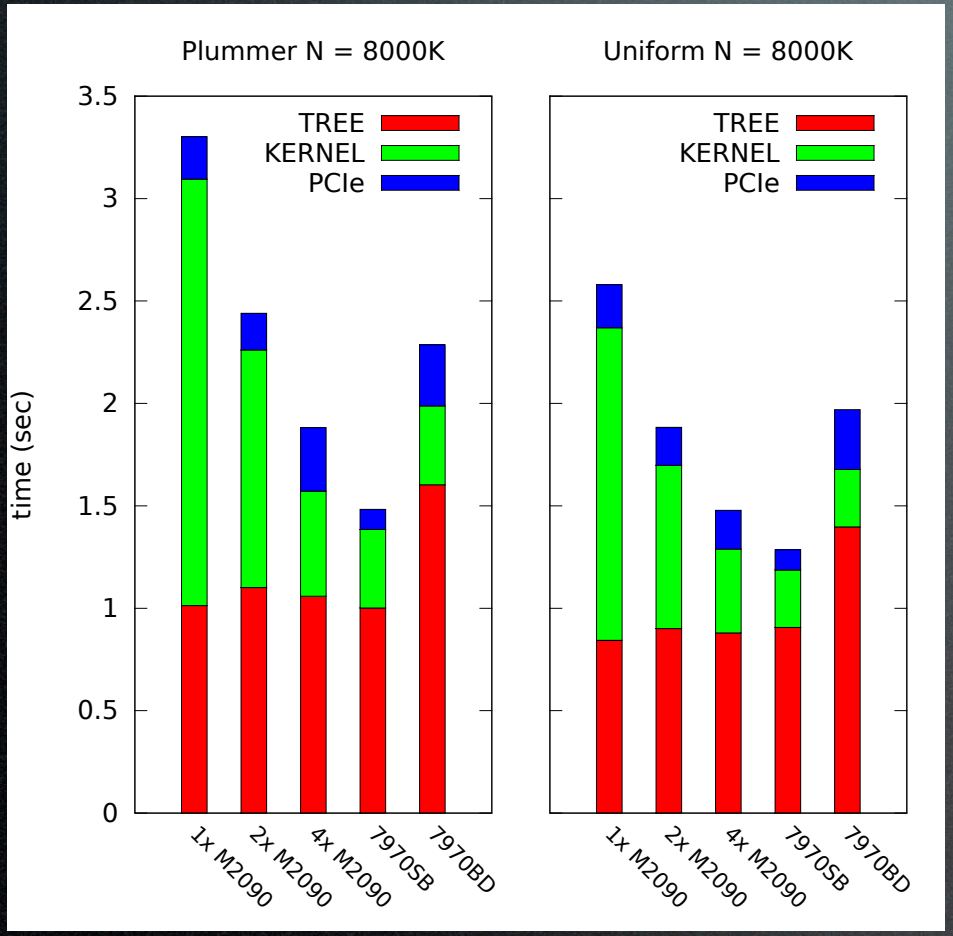
- Tree traversal for multiple particles
  - To make more compute intense
  - But redundant operations



Optimization (3) : Accuracy



Performance of Gravity



- Results on 2090HAP
  - Multi GPU is scalable
  - 2 sec by 4 GPU
- Results on 7970SB
  - 1.5 sec by 1 GPU
  - PCIe v3.0 is effective

Comparison : Gravity

- Our code: 0.16 sec on 7970 for N = 1M
  - We do not use a stack!
- Other tree/FMM code using a stack
  - Bedorf, Gaburov, Portegies Zwart (2012)
    - Tree 0.5 sec C2050 for N = 1M
  - Yokota & Barba (2012)
    - FMM 1.0 sec with GT590 for N = 1M

Not only Gravity but...

- Want to model a merger of two stars

To answer fundamental questions in the universe

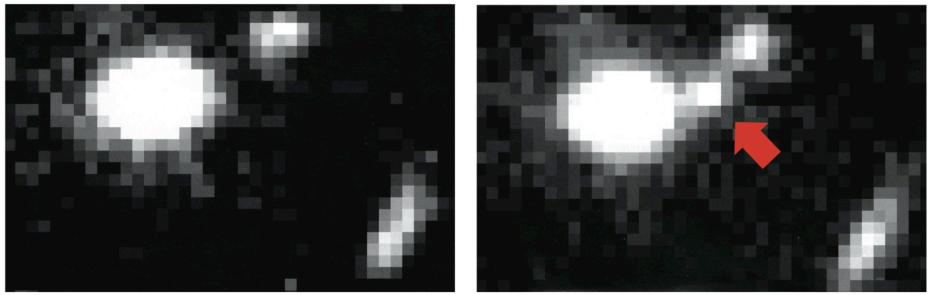
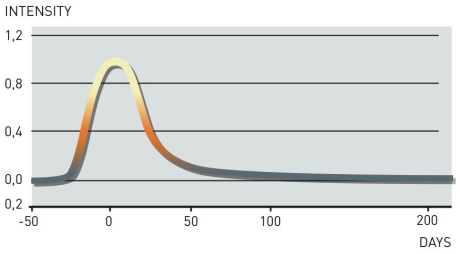
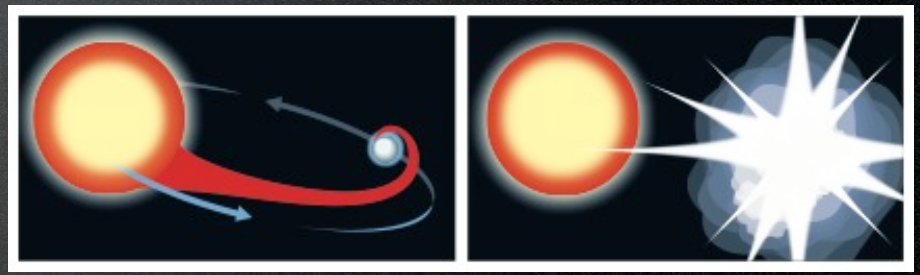


Figure 4. Supernova 1995ar. Two images of the same small piece of the sky taken three weeks apart were compared. Then, on the second image, a small dot of light was discovered! Its status as a type Ia supernova was established after further observations of its light curve. A type Ia supernova can emit as much light as an entire galaxy.The light curve is the same for all type Ia supernovae. Most light is emitted during the first few weeks (see diagram to the right).



Observation of Ia Super Novae
Awarded Nobel Prize
in Physics 2011



Application to SPH method

SPH is solving the Euler equation with particles

We need neighbor interactions

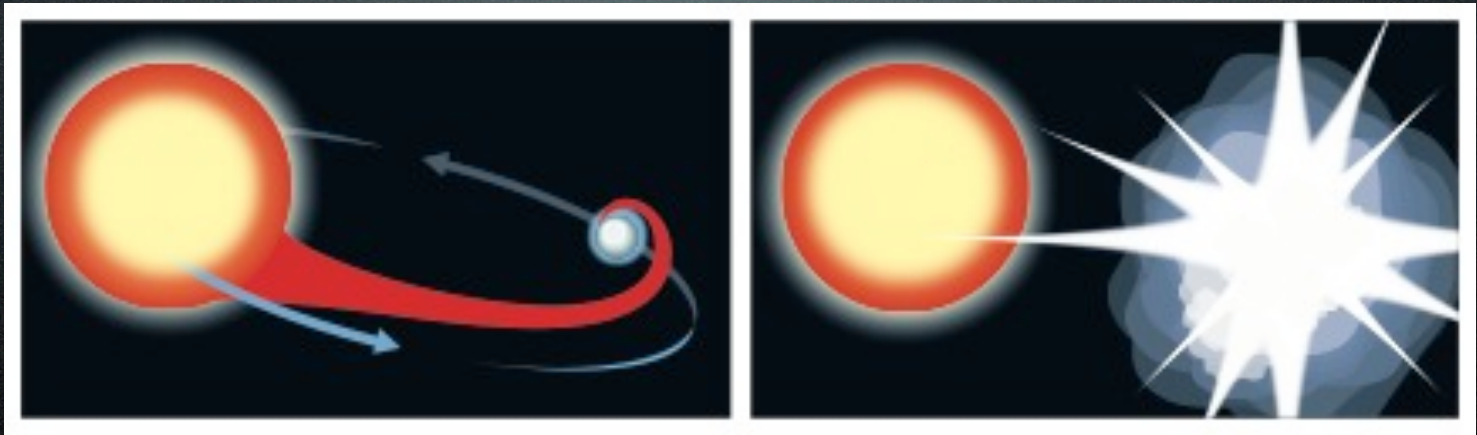
$$\frac{D\boldsymbol{v}_i}{Dt} = -\sum m_j \left(\frac{P_i}{\rho_i^2} + \frac{P_j}{\rho_j^2}\right) \nabla W(\boldsymbol{r}_i - \boldsymbol{r}_j; h) - (\nabla \Phi)_i.$$

With the same octree, we compute the summation on GPU

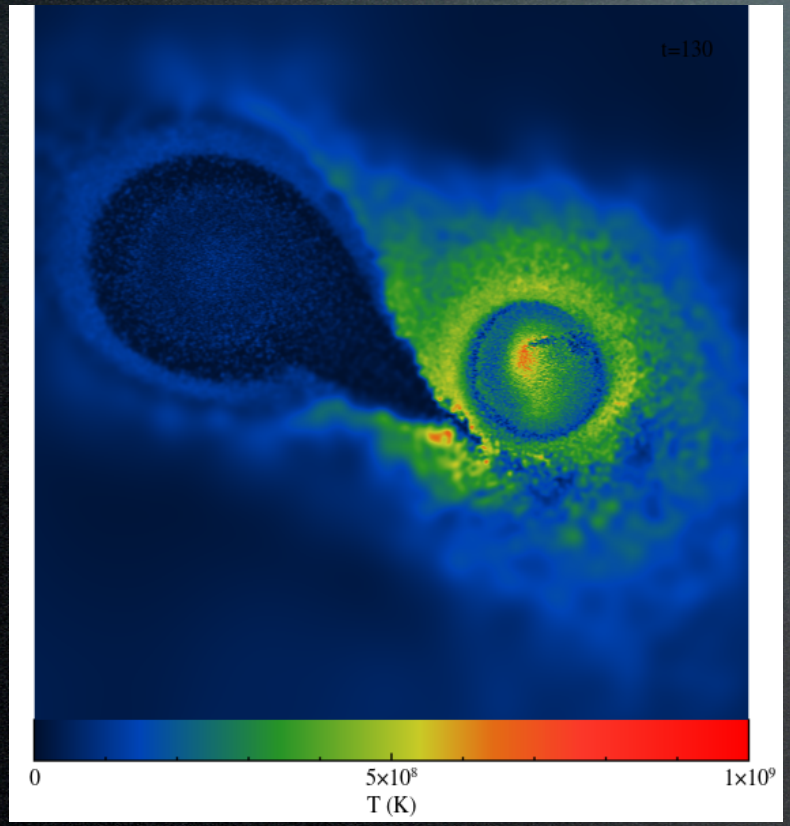
Simulations...

- We model the merging of White Dwarf stars
  - Very dense stars as a final stage of our Sun...
- Physics we need to model
  - Hydrodynamics of high density plasma
  - Gravity
  - Nuclear reactions (possibly)

N = 4M Particles Model

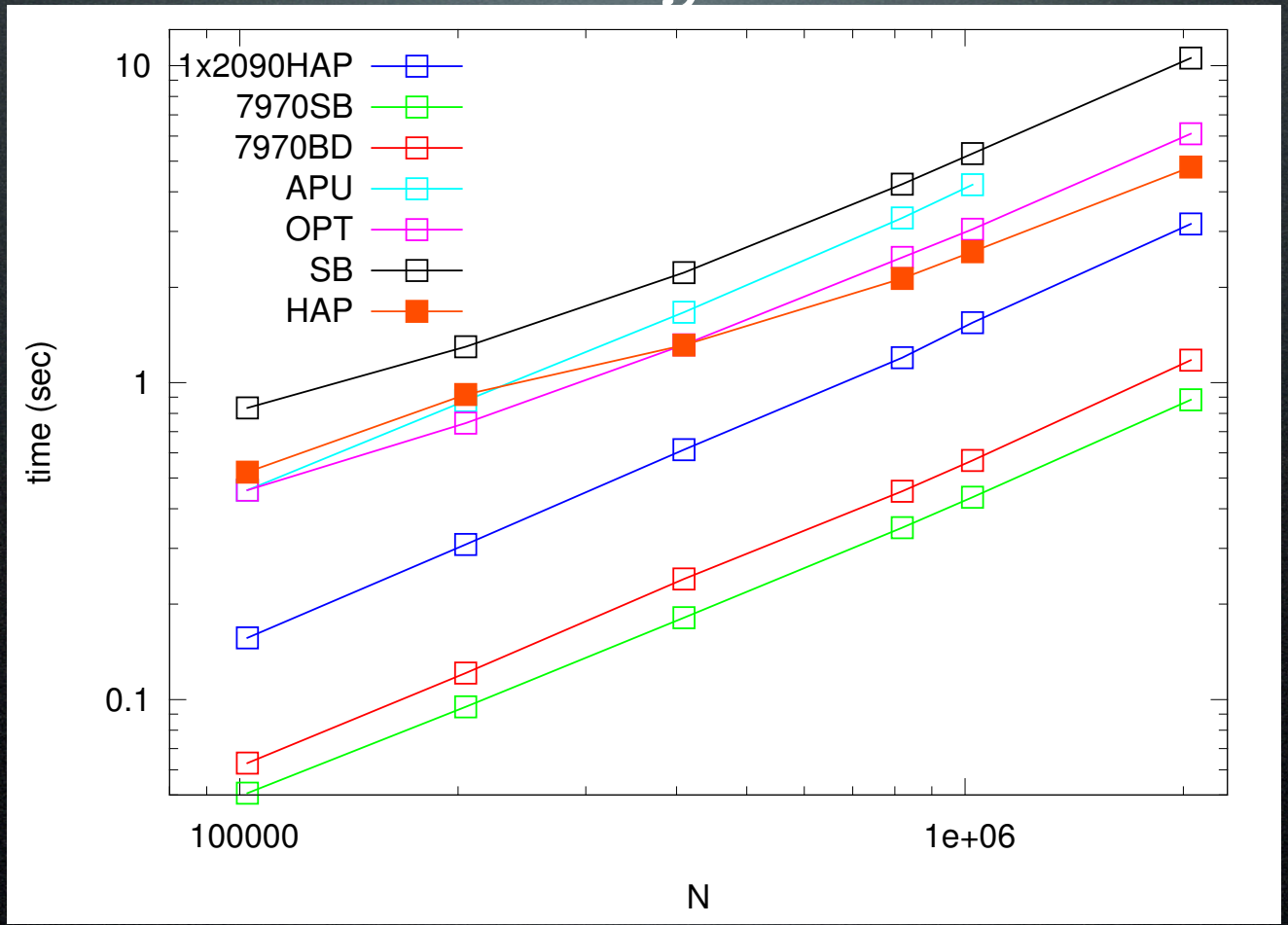


A Simulation of the merger



- Results on 2090HAP
  - 3.53 sec by 4GPU (g++)
  - Scalable
- Results on 7970SB
  - 3.20 sec by 1GPU (g++)
  - 2.52 sec by 1 GPU (icpc)
    - GPU 1.41 sec

SPH & Gravity Benchmark



Summary

- Our Octree code successfully and effectively solve astrophysical particle models
  - 4M model of two white dwarf stars on 4GPU system
- OpenCL works great on many systems
  - Multiple OpenCL devices scale well
- Optimal work distribution is a key

preprint http://arxiv.org/abs/1206.1199