Current status of GRAPE project

Jun Makino
Center for Computational Astrophysics
and
Division Theoretical Astronomy
National Astronomical Observatory of Japan

Talk structure

- Hardware
  - GRAPE machines
  - GRAPE-DR
- Science
  - "Dwarf galaxy problem"
- Algorithms
  - Efficiency limit of individual timestep algorithm

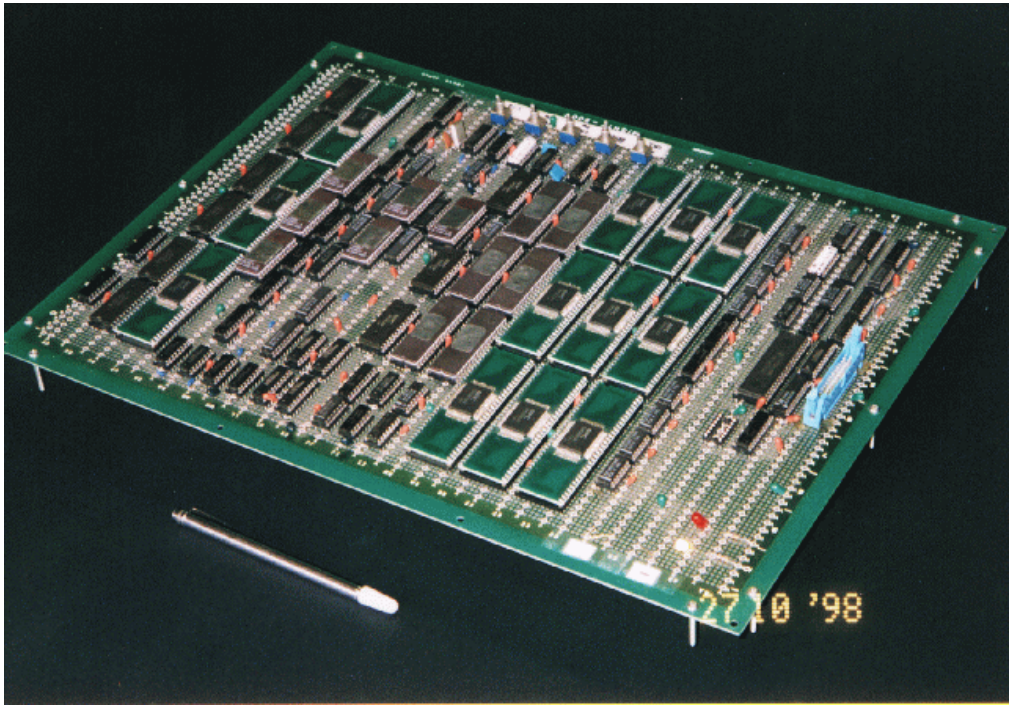
Short history of GRAPE

- Basic concept
- GRAPE-1 through 6

Basic concept

- With N-body simulation, almost all calculation goes to the calculation of particle-particle interaction.
- This is true even for schemes like Barnes-Hut treecode or FMM.
- A simple hardware which just calculates the particle-particle interaction can greatly accelerate overall calculation.

GRAPE-1(1989)

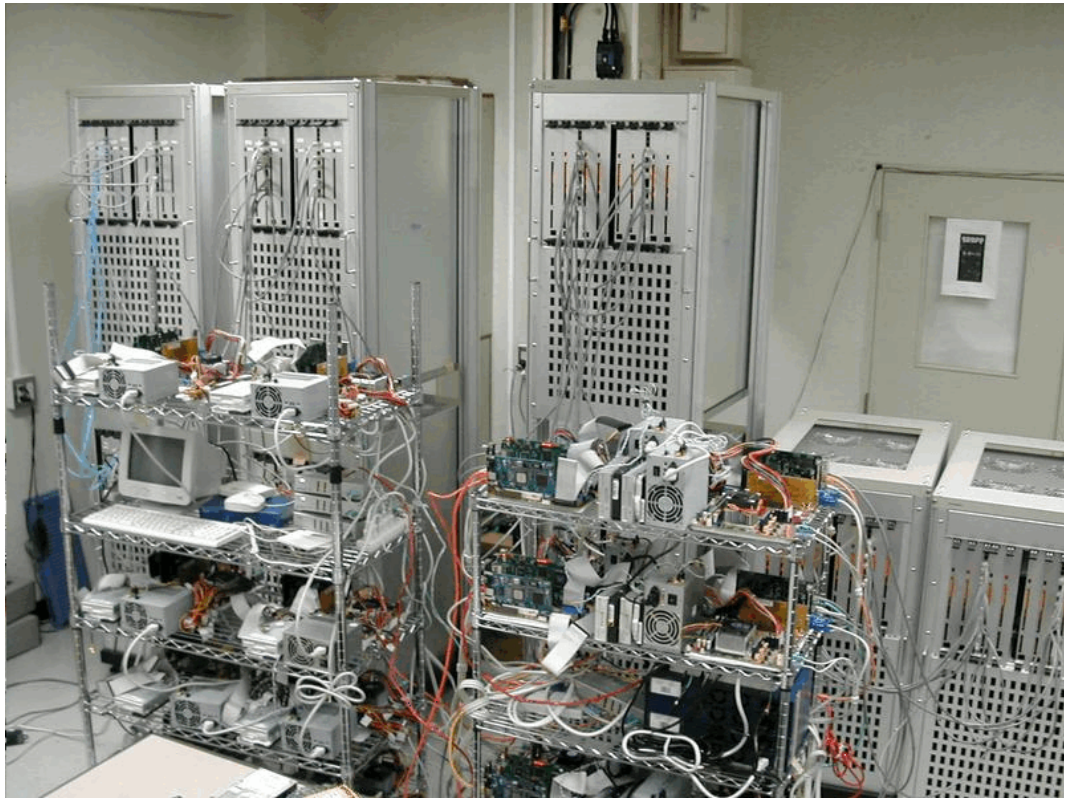


- "Mixed precision".
- $\sim$ 100 IC chips
- $\sim$ USD 2,000
- 240Mflops
- IEEE-488 interface, $\sim$100KB/s

GRAPE-4(1995)

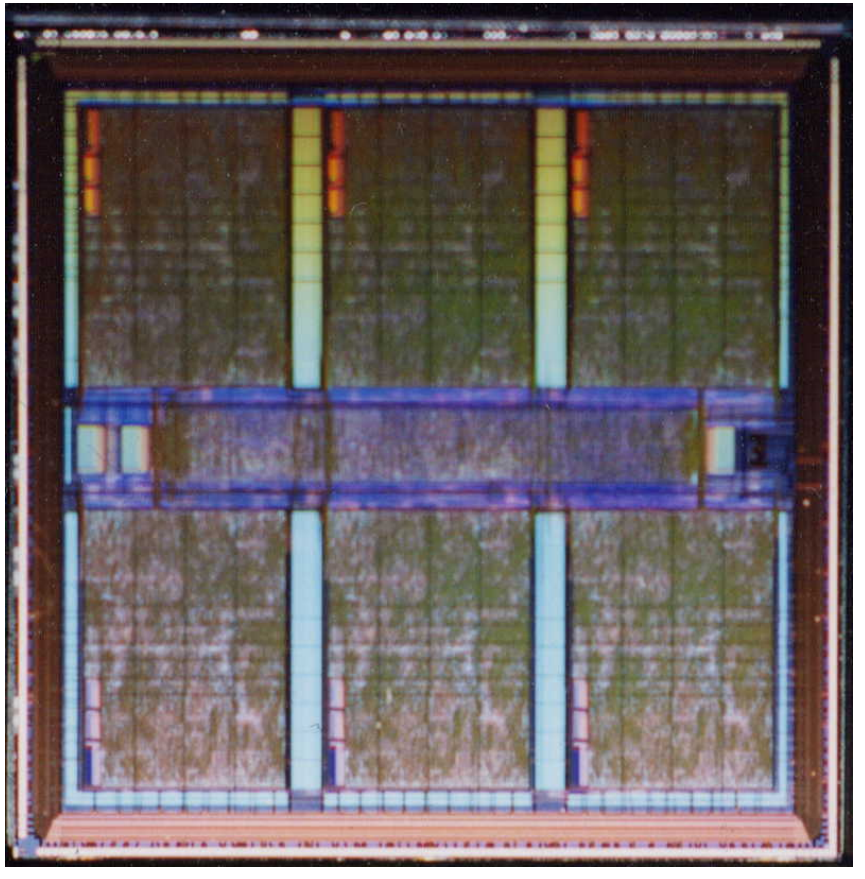
- 1.1 Tflops peak
- 36 boards each with 48 chips
- 640 Mflops per chip (20 operations, 32MHz clock)
- LSI logic 1μm

GRAPE-6(2001)



64 Tflops peak
2048 processor chips
64 processor boards
16 hosts

Processor LSI

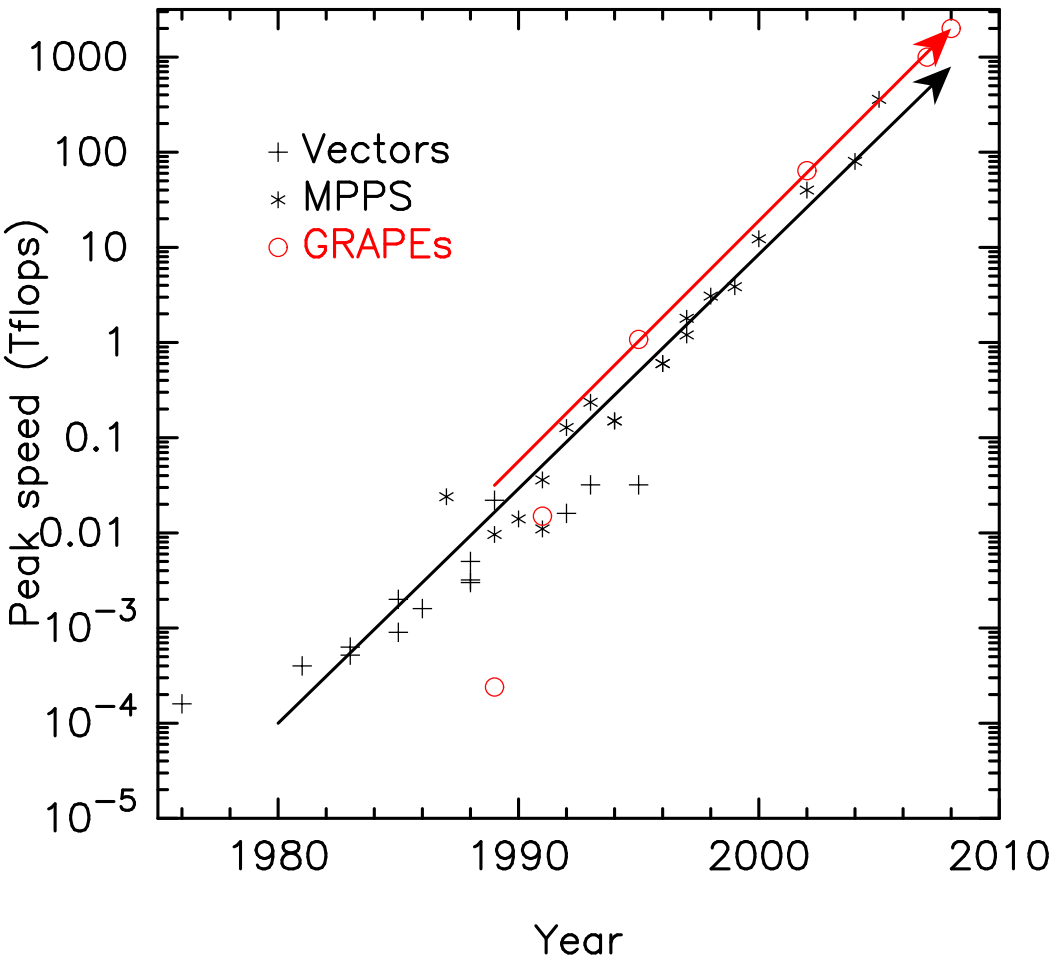


- 0.25 μm design rule (Toshiba TC-240, 1.8M gates)
- 90 MHz Clock
- 6 pipeline processors
- 31 Gflops / chip

Comparison with a recent Intel processor

	GRAPE-6	Intel Woodcrest (Xeon 5160)
Design rule	250nm	65nm
Clock	90MHz	3GHz
Peak speed	32.4Gflops	24Gflops
Power	10W	80 W
Perf/W	3.24Gflops	0.3 Gflops

Performance history



Since 1995 (GRAPE-4), GRAPE has been faster than general-purpose computers.

Development cost was around 1/100.

Should we just continue?

Problem with GRAPE approach

- Chip development cost becomes too high.

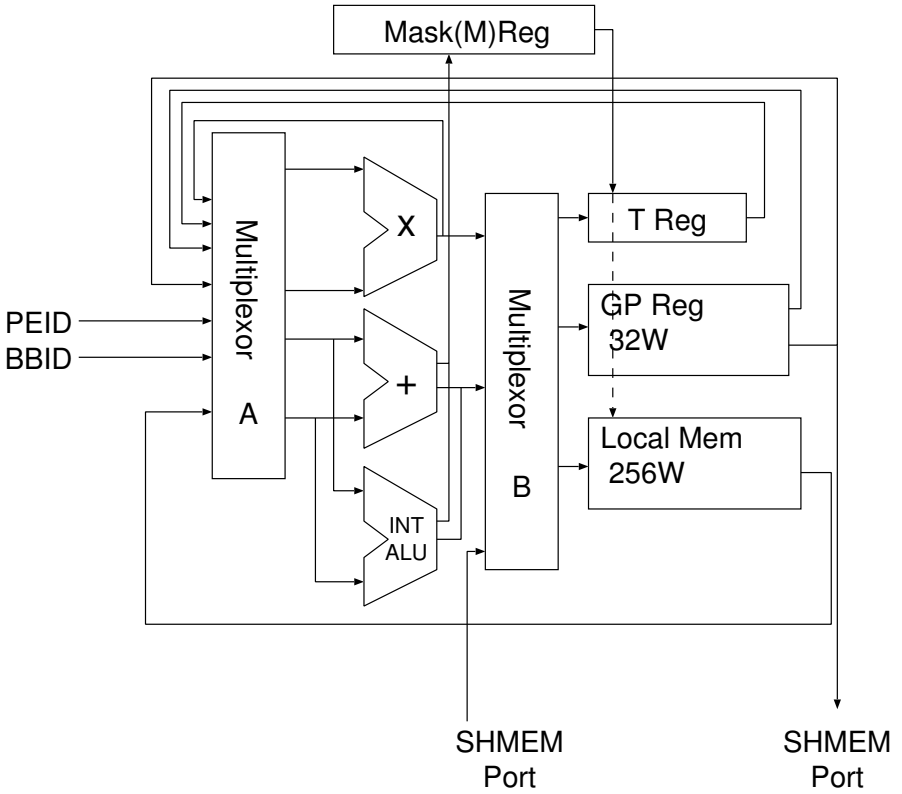
Year	Machine	Chip Initial Cost	process
1992	GRAPE-4	200K$	1μm
1997	GRAPE-6	1M$	250nm
2004	GRAPE-DR	4M$	90nm
2008?	GDR2?	$\sim$ 10M$	65nm?

Initial cost should be 1/4 or less of the total budget.
How we can continue?

Next-Generation GRAPE — GRAPE-DR

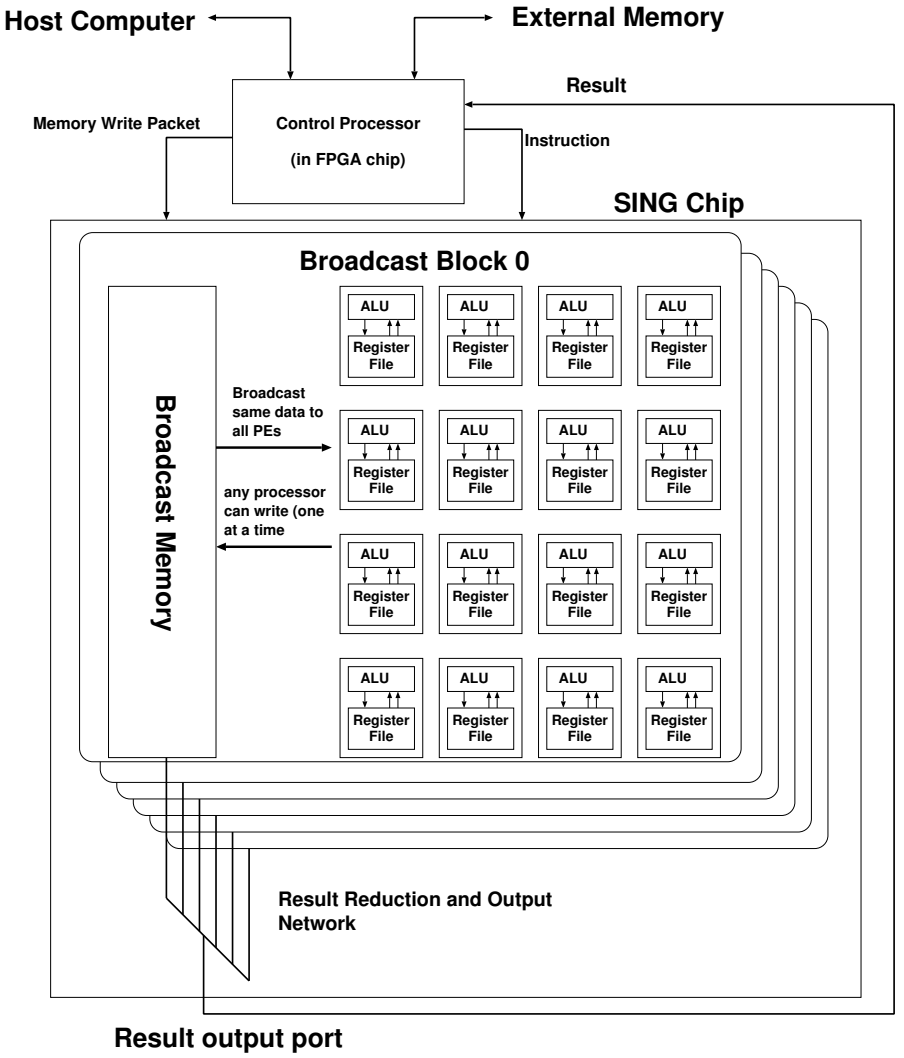
- Planned peak speed: 2 Pflops
- New architecture — wider application range than previous GRAPEs
- primarily to get funded
- No force pipeline. SIMD programmable processor
- Planned completion year: FY 2008 (early 2009)

Processor architecture



- Float Mult
- Float add/sub
- Integer ALU
- 32-word registers
- 256-word memory
- communication port

Chip structure



Collection of small processors.

512 processors on one chip
500MHz clock

Peak speed of one chip: 0.5 Tflops (20 times faster than GRAPE-6).

Why we changed the architecture?

- To get budget (N-body problem is too narrow...)
- To allow a wider range of applications
  - Molecular Dynamics
  - Boundary Element method
  - Dense matrix computation
  - SPH
- To allow a wider range of algorithms
  - FMM
  - Ahmad-Cohen
  - ...

Comparison with FPGA

- much better silicon usage (ALUs in custom circuit, no programmable switching network)
- (possibly) higher clock speed (no programmable switching network on chip)
- easier to program (no VHDL necessary; assembly language and compiler instead)

Comparison with GPGPU

- Significantly better silicon usage
- Higher cost per silicon area... (small production quantity)
- Good implementations of Hermite scheme on GPGPU exist

How do you use it?

- **GRAPE: The necessary software is now ready. Essentially the same as GRAPE-6.**
- **Matrix etc ... RIKEN/NAOJ will do something**
- **New applications:**
  - **Primitive Compiler available**
  - **For high performance, you need to write the kernel code in assembly language**

Primitive compiler

(Nakasato 2006)

```
/VARI  xi, yi, zi, e2;
/VARJ  xj, yj, zj, mj;
/VARF  fx, fy, fz;
dx = xi - xj;
dy = yi - yj;
dz = zi - zj;
r2 = dx*dx + dy*dy + dz*dz + e2;
r3i= powm32(r2);
ff = mj*r3i;
fx += ff*dx;
fy += ff*dy;
fz += ff*dz;
```

- **Assembly code**
- **Interface/driver functions**

are generated from this "high-level description".

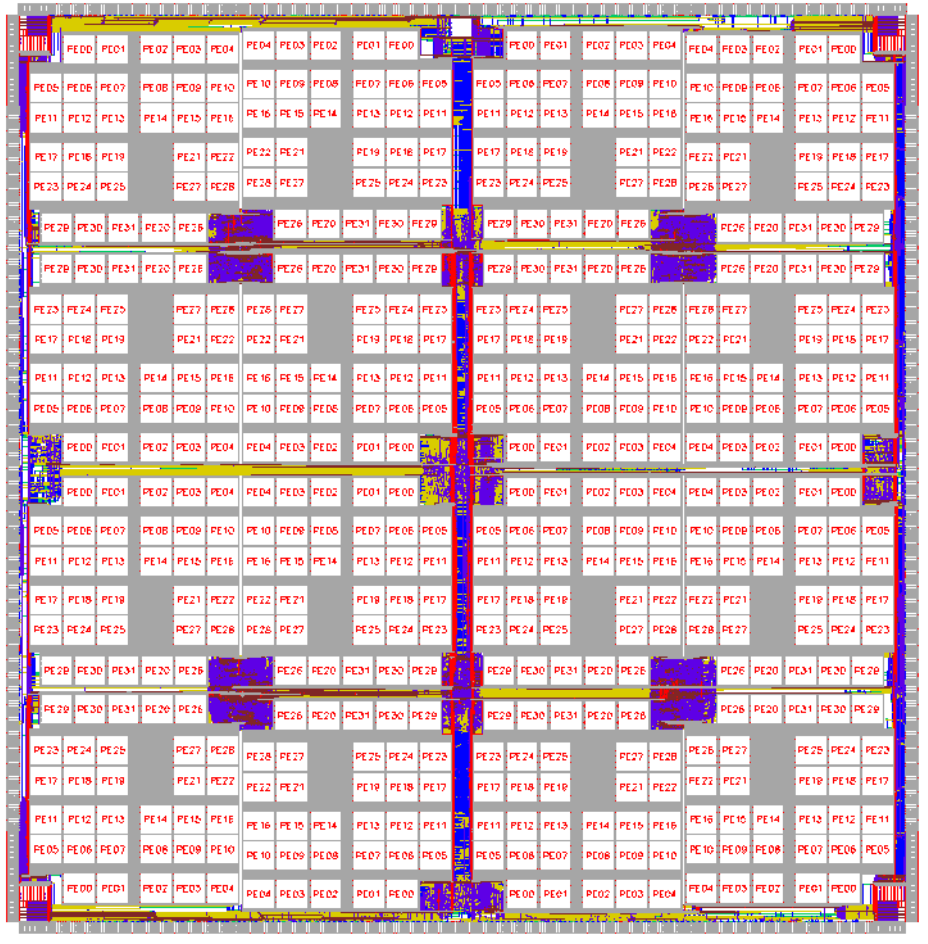
Interface functions

```
struct SING_hlt_struct0{
  double xi;
  double yi;
  double zi;
  double e2;
};
int SING_send_i_particle(struct SING_hlt_struct0 *ip,
                         int n);
...

int SING_send_elt_data0(struct SING_elt_struct0 *ip,
                        int index_in_EM);
...
int SING_get_result(struct SING_result_struct *rp);

int SING_grape_run(int n);
```

Development status

Sample chip delivered May 2006

Chip layout



- 32PEs in 16 groups
- 18mm by 18mm

Prototype board

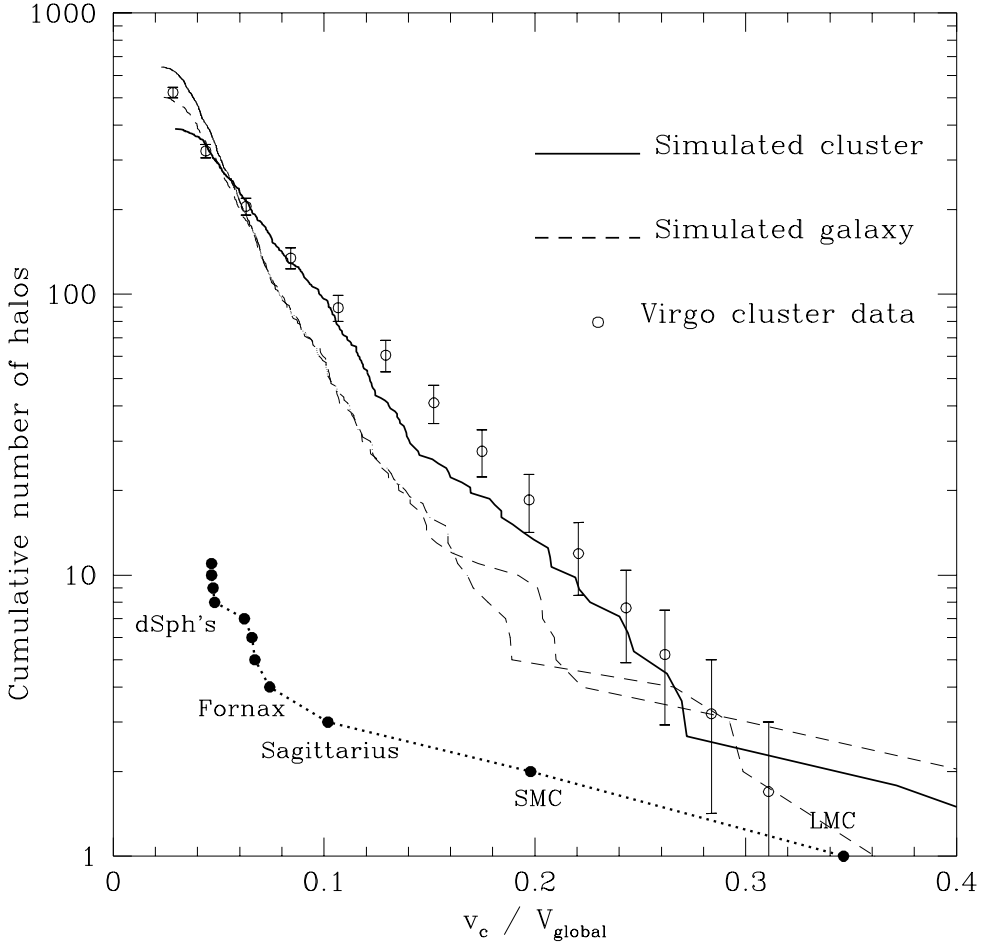
2nd prototype board. (Designed by Toshi Fukushige)
Difference from the 1st one:
PCI-Express x8 interface
On-board DRAM
Designed to run real applications

Preliminary data for production board

- Design finished, prototype board in Oct 2007
- 4 Chips on a board (2Tflops peak)
- PCI-Express x16 interface
- 300W...
- Early 2009....
- 5-10K USD

Science: Dwarf galaxy problem

(Ishiyama et al arXiv:0708.1987)



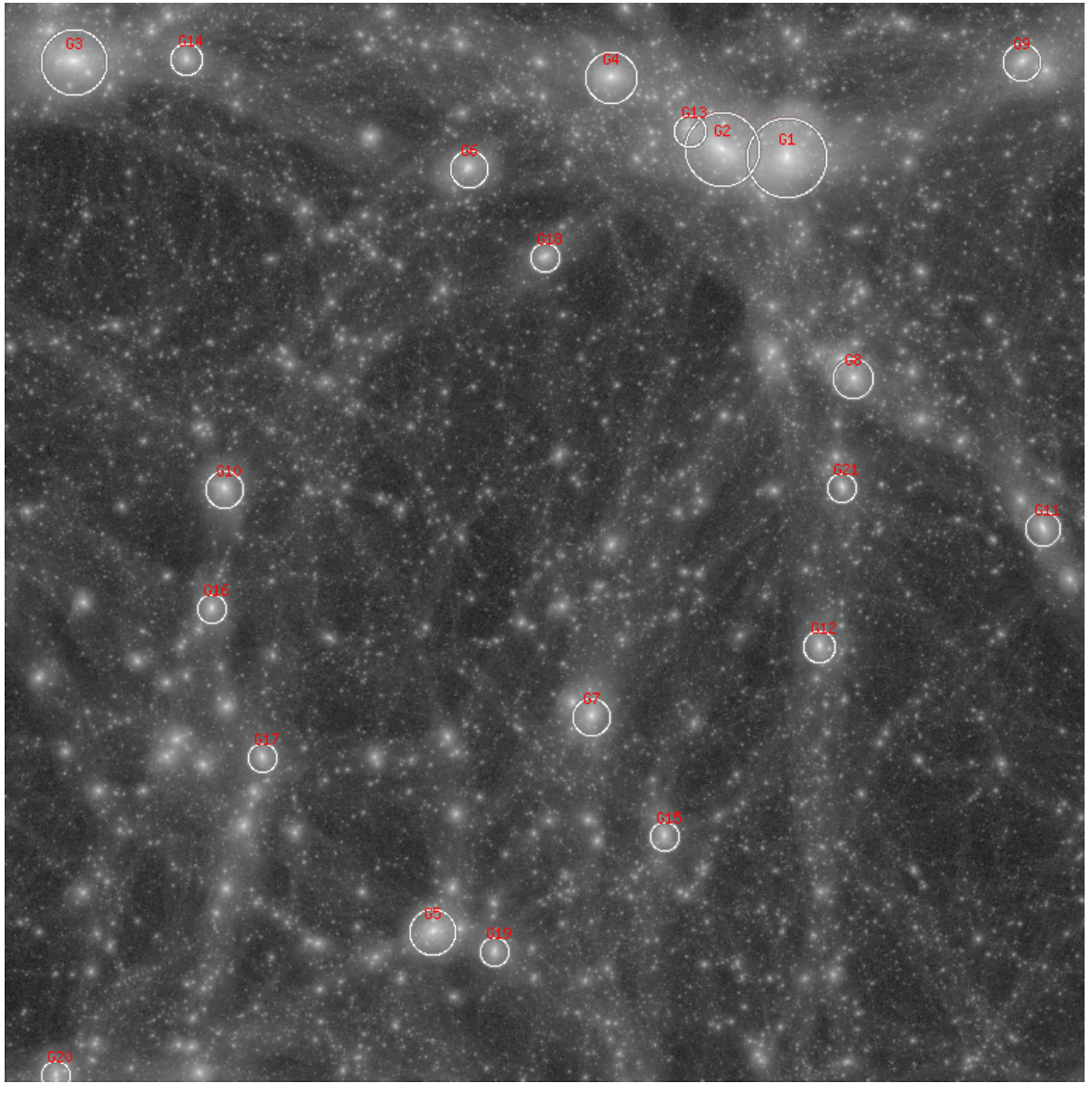
Moore et al 1999

- Too many CDM subhalos in galaxy-sized halos
- Or too few dwarf galaxies
- SCDM
- re-simulation method

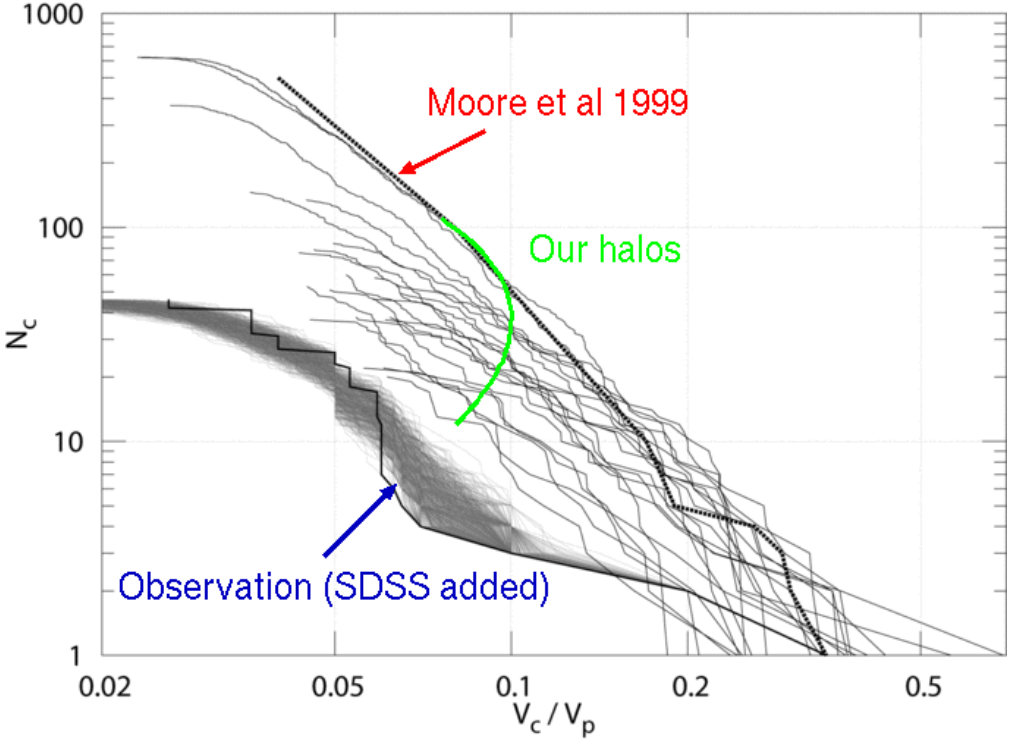
Our simulation

- **Unbiased sample of ALL halos in one simulation box**
- **TreePM code on GRAPE-6A cluster**
- **512^3 particles**
- **21.4 Mpc cube, LCDM**

Snapshot


G3
G14
G4
G9
G13
G2
G1
G6
G10
G8
G21
G11
G16
G12
G7
G17
G15
G5
G19
G20

Result

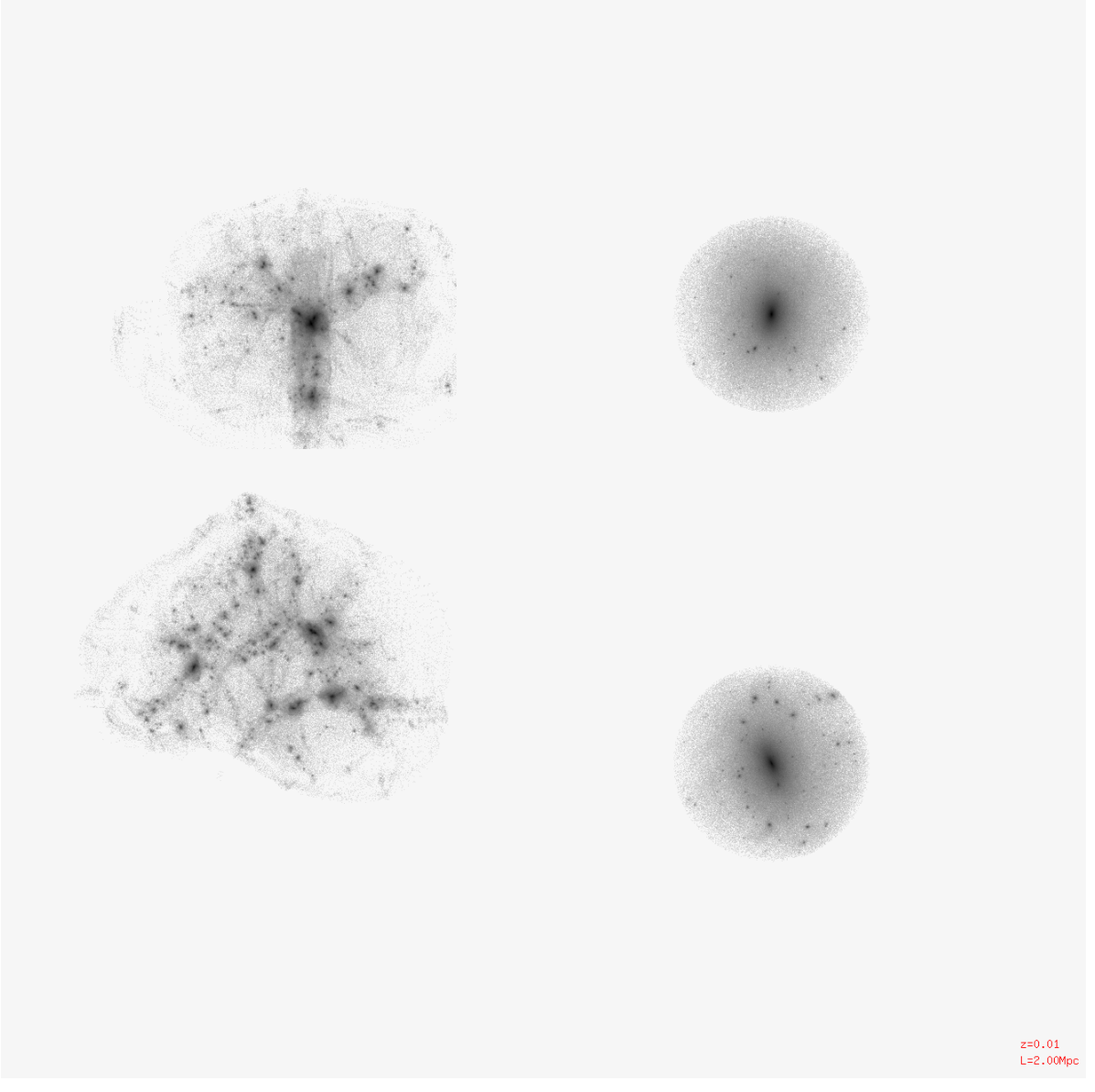


- Large variation in number of subhalos
- The richest ones agree with Moore's result

The poorest ones are within a factor of two with observations

= Dark CDM subhalos are not necessary

Poor and Rich halos



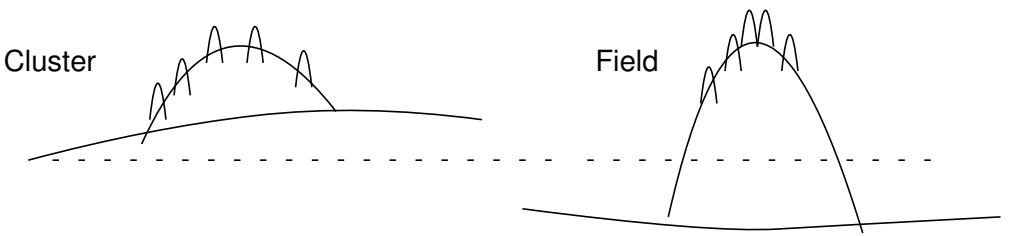
A poor halo at z=3 (left) and 0 (right)

A rich halo at z=3 (left) and 0 (right)

Reason for large variation

"Field" halos have fewer subhalos than "cluster" halos

- **form earlier: subhalos tidally stripped strongly**
- **subhalos born closer to the center of the parent halo**

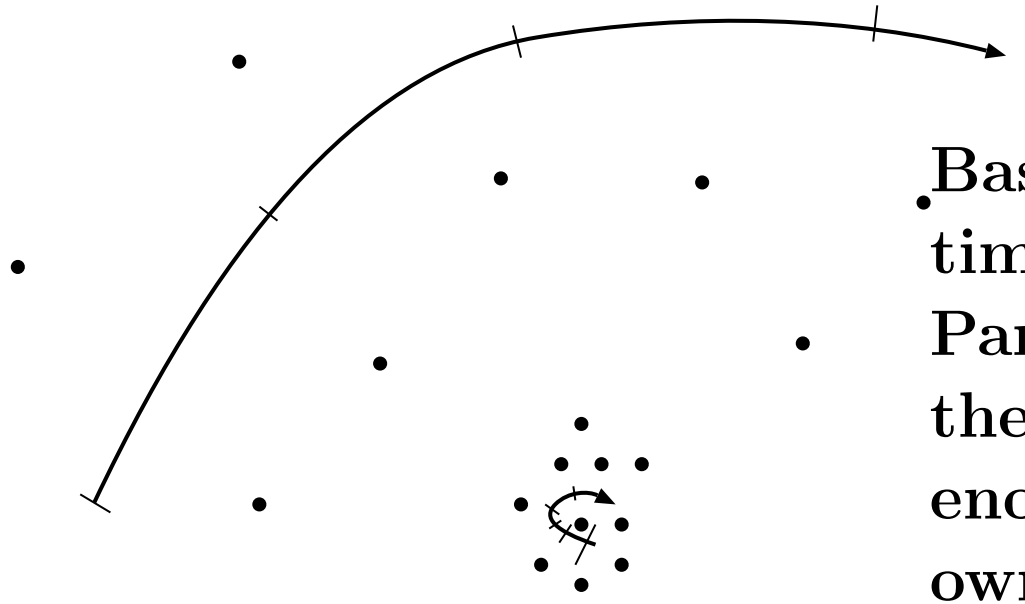


- **less external tidal field: subhalos have smaller orbital angular momentum**

Implication to globular cluster formation scenario

- Many massive CDM halos ($V_c > 0.1V_p$) were formed, but they suffered very strong tidal stripping.
- If they have developed compact stellar nuclei before stripping starts, stripped remnants would look like massive globular clusters.

Limit of individual timestep algorithm

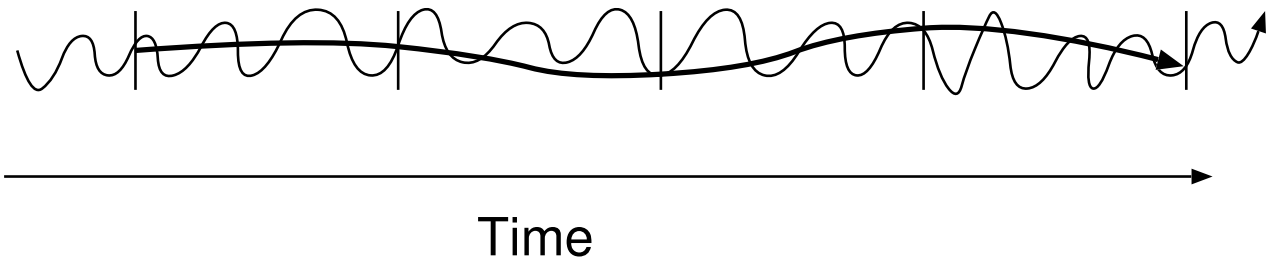


Basic idea of individual timestep:
Particles should have the timestep just enough to resolve their own orbits.

What happens to the forces from short-timescale particles to long-timescale particles?

What's happening

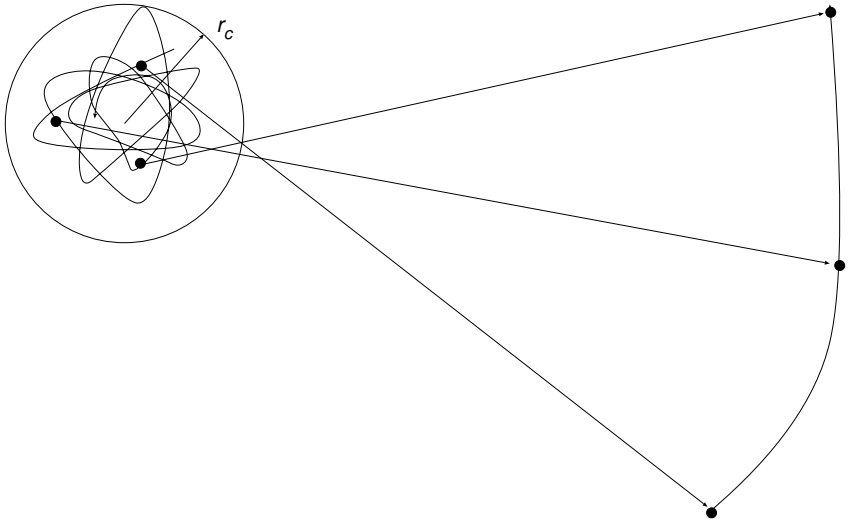
They are integrated in a completely wrong way!



- **Forces do have rapidly changing components**
- **If the timestep is large, forces are sampled "randomly" (if the orbit is not periodic)**

When does this happen?

- When the orbital timescale of particles in the core becomes less than the timestep of typical particles in the cluster.
- Roughly speaking: If $r_c \ll r_h N^{-1/3}$
- Just before bounce: $r_c \sim r_h/N \ll r_h N^{-1/3}$



Does this really matter?

In the case of a singular isothermal cusp

- **The velocity change due to this error can be comparable to two-body relaxation (smaller by $N^{1/6}$).**
- **Reduction of timestep helps, but only as $\Delta t^{1.5}$**
- **The only way to suppress this error completely is to reduce the timesteps of all particles to less than the core crossing time**

Impact on the calculation cost

- Hopefully not so severe for normal star clusters
  - the fraction of time for which the core size is small is small
  - Mass spectrum makes the core size larger
- Any system with central massive BH might be problematic.

Possible solutions

- **Individual timestep for interactions, not particles (Nitadori's talk)**
- **Time-averaged force from particles in the central region**

Time-symmetric individual timestep (JM et al 2006) might help....

Summary

- **GRAPE-DR, with programmable processors, will have wider application range than traditional GRAPEs.**
- **Second prototype (close to production version) is just arrived.**
- **Commercial version should be ready by... sometime around the end of this year.**
- **Peak speed of a card with 4 chips will be 2 Tflops**

6th and 8th-order Hermite schemes

- fourth-order Hermite scheme is not widely used.
- For many problems, higher order schemes can be advantageous.
- GRAPE-DR (unlike previous GRAPEs) can be used with whatever schemes.

Two different ways to achieve higher orders

- Use previous timesteps
- Calculate 2nd (for 6th) and 3rd (for 8th) time derivatives directly.

The latter approach

- is easier to program.
- has much smaller error coefficient
- can be made time-symmetric

Acceleration and derivatives

$$
\begin{aligned}
\boldsymbol{a}_{ij} &= m_j \frac{\boldsymbol{r}_{ij}}{\boldsymbol{r}_{ij}^3}, \\
\boldsymbol{j}_{ij} &= m_j \frac{\boldsymbol{v}_{ij}}{\boldsymbol{r}_{ij}^3} - 3\alpha \boldsymbol{a}_{ij}, \\
\boldsymbol{s}_{ij} &= m_j \frac{\boldsymbol{a}_j - \boldsymbol{a}_i}{\boldsymbol{r}_{ij}^3} - 6\alpha \boldsymbol{j}_{ij} - 3\beta \boldsymbol{a}_{ij}, \\
\boldsymbol{c}_{ij} &= m_j \frac{\boldsymbol{j}_j - \boldsymbol{j}_i}{\boldsymbol{r}_{ij}^3} - 9\alpha \boldsymbol{s}_{ij} - 9\beta \boldsymbol{j}_{ij} - 3\gamma \boldsymbol{a}_{ij}.
\end{aligned}
$$

Acceleration and derivatives (cont'd)

$$\alpha = \frac{\boldsymbol{r}_{ij} \cdot \boldsymbol{v}_{ij}}{\boldsymbol{r}_{ij}^2},$$

$$\beta = \frac{|\boldsymbol{v}_{ij}|^2 + \boldsymbol{r}_{ij} \cdot (\boldsymbol{a}_j - \boldsymbol{a}_i)}{\boldsymbol{r}_{ij}^2} + \alpha^2,$$

$$\gamma = \frac{3\boldsymbol{v}_{ij} \cdot (\boldsymbol{a}_j - \boldsymbol{a}_i) + \mathrm{r}_{ij} \cdot (\boldsymbol{j}_j - \boldsymbol{j}_i)}{\boldsymbol{r}_{ij}^2} + \alpha(3\beta - 4\alpha^2),$$

Predictor and corrector

Predictors: Usual polynomial form.
Caution: need to predict acceleration (and jerk for 8th order) and need to use one previous value(s) to construct higher-order terms.

Correctors:

$$\boldsymbol{v}_{i,c} = \boldsymbol{v}_{i,0} + \frac{\boldsymbol{\Delta t}}{\boldsymbol{2}}(\boldsymbol{a}_{i,1} + \boldsymbol{a}_{i,0}) - \frac{\boldsymbol{\Delta t^2}}{\boldsymbol{10}}(\boldsymbol{j}_{i,1} - \boldsymbol{j}_{i,0}) + \frac{\boldsymbol{\Delta t^3}}{\boldsymbol{120}}(\boldsymbol{s}_{i,1} +$$

$$\boldsymbol{v}_{i,c} = \boldsymbol{v}_{i,0} + \frac{\boldsymbol{\Delta t}}{\boldsymbol{2}}(\boldsymbol{a}_{i,1} + \boldsymbol{a}_{i,0}) - \frac{\boldsymbol{3\Delta t^2}}{\boldsymbol{28}}(\boldsymbol{j}_{i,1} - \boldsymbol{j}_{i,0})$$
$$+ \frac{\boldsymbol{\Delta t^3}}{\boldsymbol{84}}(\boldsymbol{s}_{i,1} + \boldsymbol{s}_{i,0}) - \frac{\boldsymbol{\Delta t^4}}{\boldsymbol{1680}}(\boldsymbol{c}_{i,1} - \boldsymbol{c}_{i,0}) + \boldsymbol{O(\Delta t^9)},$$

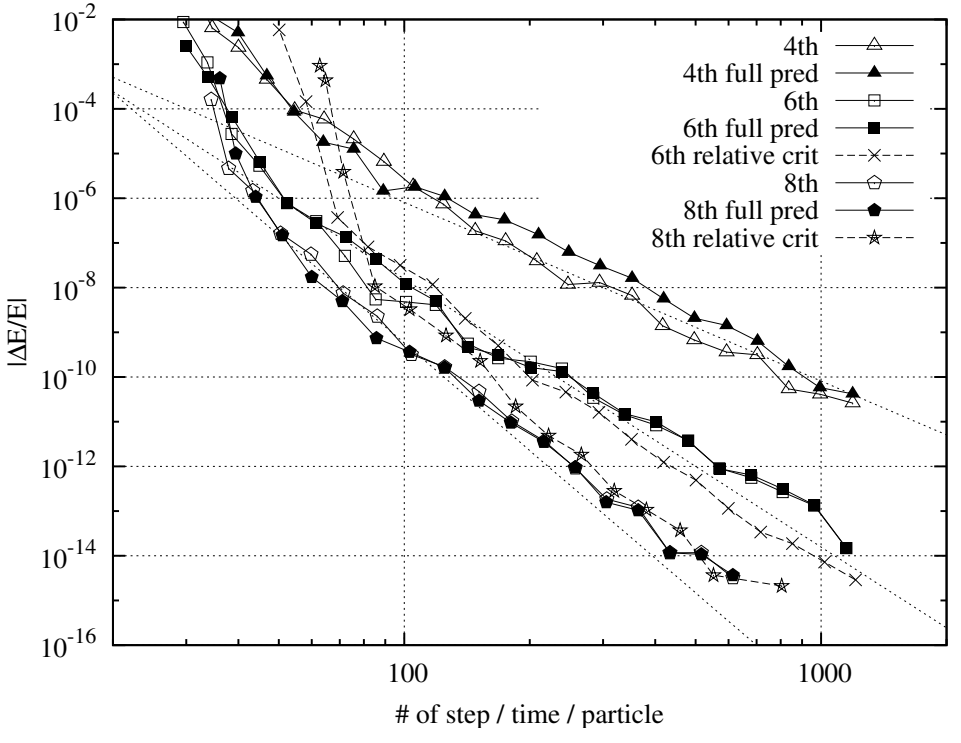
Timestep criterion

"Generalization" of the standard one:

$$\Delta t \;=\; \eta \left(\frac{|\boldsymbol{a}^{(0)}||\boldsymbol{a}^{(2)}| + |\boldsymbol{a}^{(1)}|^2}{|\boldsymbol{a}^{(p-3)}||\boldsymbol{a}^{(p-1)}| + |\boldsymbol{a}^{(p-2)}|^2} \right)^{1/(2p-6)} .$$

seems to work fine.

Numerical result



- $N = 1024$, Plummer model, $\epsilon = 4/N$
- Higher order schemes actually work.
- They allow much larger timesteps than that for the 4th order scheme for practical range of accuracy.