#### **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".
- ~ 100 IC chips
- ~ USD 2,000
- 240Mflops
- IEEE-488 interface, ~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  $1\mu m$

## **GRAPE-6(2001)**



64 Tflops peak
2048 processor
chips
64 processor
boards
16 hosts

## **Processor LSI**



- 0.25  $\mu$ m 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	$90 \mathrm{MHz}$	$3 \mathrm{GHz}$
Peak speed	<b>32.4Gflops</b>	24 G flops
Power	10W	$80 \ W$
$\operatorname{Perf}/W$	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\$	$1 \mu { m m}$
1997	<b>GRAPE-6</b>	1M\$	$250 \mathrm{nm}$
<b>2004</b>	<b>GRAPE-DR</b>	4M\$	<b>90</b> nm
2008?	GDR2?	$\sim 10 \mathrm{M}\$$	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



**Result output port** 

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
  - $\mathbf{FMM}$
  - 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

: 5		ji, ji	<u></u>	<u></u>	1.1.1		<u>l</u>	1.1.1		1.1.1			1							1	)       			1.1.	
-		FEDD	PE01	PE 07	PE03	PE04	PE04	PED3	PEOZ	PEQ1	FEOD				PE CO	PEQI	PEQZ	PEQO	PEGA	FED4	FED3	FE 02	PEOI	PEOD	
	PE D5	PEDB	PE 07	PE OB	PEOP	PE10	FE 10	PED9	PEDS	FED7	FE 06	FEOS	-1	FEOD	FE CO	PE07	PEOB	PEQP	PE10	PE 10	PEDP	PEOB	PE 07	PE06	PE05
	PE 11	PE1Z	PE10	PE 14	PEIS	PE16	PE 16	PE 15	PE 14	PE13	PE12	PE11	Į.	PE11	PE12	PE13	PE14	PE15	PE18	PE 16	PE 10	PE 14	PEID	PE1Z	FEII
	PE 17	PC 16	PE 19		PEZI	PE77	PE 22	<b>PE</b> 21		PE 19	PE 18	PE17		PE17	PE 18	PE19		PE21	PE22	FEZZ	PE21		PE 19	PE 15	PE17
	PE 23	PE 24	PE 25		FE27	PE28	PE 28	PE 27		PE 25	PE 24	PE 23		PE 23	PE 24	PE 25		PE27	FE28	PE 26	PE 27		FE 25	PE24	PE23
	PE 2	e PES	D PESI	PE 20	PE2B	3		PE76	PEZO	PE31 P	E 30 P	E 29		PEZ	9 PE 3	D PEST	FEZD	FE 26			PE26	PE20	PE31 P	E 20 P	E 79
	PEZ	e PEO	D PEOI	PEZO	PEZE			PEZE	PEZO	PE31 P	£30 F	E ZP		PEZ	e PEO	0 PEO	PEZD	PE ZB			PE26	PE20	PE31 P	E 20 P	E 79
-	FE Z3	FE 74	FE ZD		PE27	PE78	PETS	PE 17		PE ZD	FE 74	FE ZO		FE ZO	PE 74	PEZO		PC27	PEZO	FE 78	FE 27		PE 20	PE74	PEZO
	PE 17	PC 18	PC 19		PE21	PE22	PE 22	FE 21		PE 19	PE 18	PE17	2	PE 17	PE 18	PE 19		PE21	PE22	PE 22	PE21		PE 19	PE 15	PE 17
	PE11	PC12	PC 13	PE 14	PE15	PE 18	FE 16	FE 15	FE 14	PE 13	PE 12	PE11		PE11	PE 12	PE13	PE14	PE15	PE16	PE 16	PC 15	PE 1.4	PE13	PE 12	PE 11
	PEDS	PEDB	PE 07	PE OB	PEOP	PE10	PE 10	PED®	FEDS	PED7	PEOB	PE 05		PE 05	PE 06	PE07	PEOB	PE09	PE 10	PE 10	PEDP	PEOB	PE 07	PE06	PE05
1		PEDD	FE01	PE 02	PE03	PE04	PE D4	PE D3	PE D2	PE01	PEOD		R		PE 00	PE01	PE02	FEO3	FEC4	PE D4	PEDƏ	PE 02	PE01	PE 00	
		PEDD	FE01	PE 02	PE03	PE04	PED4	PE D3	PED2	PEO1	PEOD			ųΥ.	PE 00	PE01	PEOZ	FEO3	PEC4	PED4	PED3	PEO2	PE01	PEOD	
	PEDS	PEDE	PE07	PEOB	PEOP	FE 10	PE 10	PED®	FEDS	PED7	PEOB	PEOS	Ĩ	PEOS	PE 06	PE 07	PEOB	FEOP	PE1D	PE 10	PEDP	PEOB	PE 07	PEOS	PE05
	PE11	PE12	PE18	PE14	PEID	PE16	PE 16	FE 15	PE 14	PE13	PE 12	PE11	E	PE11	PE12	PE13	PE14	PE15	PE15	PE 16	PE 16	PE14	PE13	PE12	FE11
	PE 17	PC 18	PC19		PEZI	PE77	PEZZ	FE 21		PC19	PE 18	PE 17		PE 17	PE18	PE 18		PEZI	PE27	FE 72	PE21		PE18	PE18	PE17
	PE 23	PE 24	PE25		PE27	PE28	PE 28	PE 27		PE 25	PE 24	PE 23		PE 23	PE24	PE25		PE27	PE28	PE 28	PE 27		PE 25	PE24	PE23
	PE2	e FE3	D PE31	PE 20	PE 26	(g		PE26	PE20	FE31 P	E 20 P	E 28		PE2	e PES	0 PE31	PE 20	PE 26	- Teri		PE26	PE20	PE31 P	E 20 P	£ 28
	PE 2	9 PE3	OT PEAT	PE 20	PE26				-								Incas	ine as			PE26	PE20	PE21 P	E 30 P	E 29
	195 - 23	DE 24	L.	1	00000	DE CR		PE 10	erse l	-corp	the p	· · · · ·			e l'era	- 1 <b>- E</b> - 5	1 1 20	PE20	. iterat		05.22		ine ce	DE CA	L Interna
	PE 17	PE18	PC19		PE21	PEZZ	PE28	PE 27		PE 25	PE 24	PE 23		PE 23	PE 24	PE 25		PE27	PE28	FEZZ	PE21		PEIR	PE18	PE17
	PE 11	RE 42	RE 13	PEIA	PEAK	PEIE	PE12	HE 21		PE19	PE 18	PE17		PE17	PE18	PE10		PEZI	PE22	and a	PETE	PEIA	PED	PEIZ	EF 11
	PE 05	PE06	PE07	PEOB	PE09	PE10	PE 16	PE 15	PE 14	PE13	PE12	PE11		PE11	PE12	PE13	PE14	PE15	PE16	PE 10	FE 09	FEOB	PE07	PE06	PE05
		FT DD	PEDI	PT D7	PEDA	PT D4	-2.10	-1.09			-LOD	-200 		-100	-208	-207	PLOB	PLUS	-210	ET D4	FEDN	FT D7	PFC1	FEDD	
							PEDA	PE03	PE02	PE01	FE 00				PE 00	PEO1	PE02	PE03	PECA				الفلي حل		

- 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



## 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



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

## What's happening

They are integrated in a completely wrong way!



Time

- 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  $r_c \ll r_h N^{-1/3}$
- $\bullet$  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

$$egin{aligned} a_{ij} &= m_j rac{r_{ij}}{r_{ij}^3}, \ j_{ij} &= m_j rac{v_{ij}}{r_{ij}^3} - 3lpha a_{ij}, \ s_{ij} &= m_j rac{a_j - a_i}{r_{ij}^3} - 6lpha j_{ij} - 3eta a_{ij}, \ c_{ij} &= m_j rac{j_j - j_i}{r_{ij}^3} - 9lpha s_{ij} - 9eta j_{ij} - 3\gamma a_{ij}. \end{aligned}$$

#### Acceleration and derivatives (cont'd)

$$egin{aligned} lpha &= rac{r_{ij} \cdot v_{ij}}{r_{ij}^2}, \ eta &= rac{|v_{ij}|^2 + r_{ij} \cdot (a_j - a_i)}{r_{ij}^2} + lpha^2, \ \gamma &= rac{3 v_{ij} \cdot (a_j - a_i) + \mathrm{r}_{ij} \cdot (j_j - j_i)}{r_{ij}^2} + lpha (3eta - 4lpha^2), \end{aligned}$$

### **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:** 

$$egin{aligned} v_{i,c} &= v_{i,0} + rac{\Delta t}{2}(a_{i,1} + a_{i,0}) - rac{\Delta t^2}{10}(j_{i,1} - j_{i,0}) + rac{\Delta t^3}{120}(s_{i,1} + v_{i,c}) \ v_{i,c} &= v_{i,0} \ + rac{\Delta t}{2}(a_{i,1} + a_{i,0}) - rac{3\Delta t^2}{28}(j_{i,1} - j_{i,0}) \ &+ rac{\Delta t^3}{84}(s_{i,1} + s_{i,0}) - rac{\Delta t^4}{1680}(c_{i,1} - c_{i,0}) + O(\Delta t^9), \end{aligned}$$

### **Timestep criterion**

"Generalization" of the standard one:

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

. 11-

~ >

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.