GRAPE accelerators

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

IAU 270, Computational Star Formation, Barcelona May 31st - Jun 4th 2010
Talk structure

- Short history of GRAPE
  - GRAPE machines
- GRAPE-DR
  - Architecture
  - Comparison with other architecture
  - Development status
- Next-Generation GRAPE
- GRAPEs and Star-formation simulations
Summary

- GRAPE-DR, with programmable processors, has wider application range than traditional GRAPEs.
- Peak speed of a card with 4 chips is 800 Gflops (DP).
- DGEMM performance 640 Gflops, LU decomposition > 400Gflops
- Currently, 128-card, 512-chip system is up and running.
- We return to custom design with structured ASIC for the next generation (budget limitation...)
- GRAPE-DR might be useful for star formation simulation.
Short history of GRAPE

- Basic concept
- GRAPE-1 through 6
- Software Perspective
Basic concept (As of 1988)

- 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 calculates the particle-particle interaction can accelerate overall calculation.
Chikada's idea (1988)

- Hardwired pipeline for force calculation (similar to Delft DMDP)
- Hybrid Architecture (things other than force calculation done elsewhere)
GRAPE-1 to GRAPE-6

GRAPE-1: 1989, 308Mflops
GRAPE-4: 1995, 1.08Tflops
GRAPE-6: 2002, 64Tflops
Performance history

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

Development cost was around 1/100.
Science on GRAPEs

- Pure $N$-body
  - Planetary formation (Kokubo, Ida, ...)
  - Star clusters (JM, Baumgardt, Portegies Zwart, Hurley, ...)
  - Galactic Dynamics (Athanassoula, Fujii, ...)
  - Galaxies with central BH (JM, Iwasawa, ...)
  - Cosmology (Fukushige, Yoshikawa)

- SPH
  - Galaxy Formation (Steinmetz, Susa, Saitoh)
  - Star formation (Klessen)
Advantage of GRAPEs

- Planetary formation, Star clusters: $N^2$ with individual timestep
  - GRAPE very efficient
  - Difficult to use large parallel machine
- Galactic Dynamics, Cosmology: Treecode
  - GRAPE okay
  - large parallel machines work fine
- Galaxy Formation, Star formation: SPH
  - GRAPE does gravity only
  - Difficult to use large parallel machine efficiently?
“Problem” with GRAPE approach

- Chip development cost has become too high.

<table>
<thead>
<tr>
<th>Year</th>
<th>Machine</th>
<th>Chip initial cost</th>
<th>process</th>
</tr>
</thead>
<tbody>
<tr>
<td>1992</td>
<td>GRAPE-4</td>
<td>200K$</td>
<td>1μm</td>
</tr>
<tr>
<td>1997</td>
<td>GRAPE-6</td>
<td>1M$</td>
<td>250nm</td>
</tr>
<tr>
<td>2004</td>
<td>GRAPE-DR</td>
<td>4M$</td>
<td>90nm</td>
</tr>
<tr>
<td>2010?</td>
<td>GDR2?</td>
<td>&gt; 10M$</td>
<td>45nm?</td>
</tr>
</tbody>
</table>

Initial cost should be 1/4 or less of the total budget. How we can continue?
Next-Generation GRAPE — GRAPE-DR

- New architecture — wider application range than previous GRAPEs
- primarily to get funded
- No force pipeline. SIMD programmable processor
Processor architecture

- Float Mult
- Float add/sub
- Integer ALU
- 32-word registers
- 256-word memory
- Communication port
Chip architecture

- 32 PEs organized to “broadcast block” (BB)
- BB has shared memory. Various reduction operation can be applied to the output from BBs using reduction tree.
- Input data is broadcasted to all BBs.
Computation Model

Parallel evaluation of

\[ R_i = \sum_j f(x_i, y_j) \]

- parallel over both \( i \) and \( j \)
- \( y_j \) may be omitted (trivial parallelism)
- \( S_{i,j} = \sum_k f(x_{i,k}, y_{k,j}) \) also possible
  (matrix multiplication)
The Chip

Sample chip delivered May 2006
90nm TSMC, Worst case 65W@500MHz
PE Layout

Black: Local Memory
Red: Reg. File
Orange: FMUL
Green: FADD
Blue: IALU

0.7mm by 0.7mm
800K transistors
0.13W@500MHz
1Gflops/512Mflops peak (SP/DP)
Processor board

- Around 200W power consumption
- Not quite running at 500MHz yet... (FPGA design not optimized yet)
- 900Gflops DP peak (450MHz clock)
- Available from K&F Computing Research (www.kfcr.jp)

PCIe x16 (Gen 1) interface
Altera Arria GX as DRAM controller/communication interface
GRAPE-DR cluster system
GRAPE-DR cluster system

Sorry, this is MareNostrum
GRAPE-DR cluster system
**GRAPE-DR cluster system**

- 128-node, 128-card system (105TF theoretical peak @ 400MHz)
- Linpack measured: 360 Gflops/node
- Gravity code: 340Gflops/chip
- Host computer: Intel Core i7+X58 chipset, 12GB memory
- network: x4 DDR Infiniband
- plan to expand to 384-node system.
Software Environment

- Assembly Language
- Kernel libraries
  - matrix multiplication
    * BLAS, LAPACK
    - Particle-Particle interaction
- Compiler Language
- OpenMP-like interface

Idea based on PGDL (Hamada, Nakasato)
— pipeline generator for FPGA
Compiler language example

Nakasato (2008), based on LLVM.

VARI  xi, yi, zi;
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;
rinv = rsqrt(r2);
mr3inv = rinv*rinv*rinv*mj;
fx+= mr3inv*dx;
fy+= mr3inv*dy;
fz+= mr3inv*dz;
Driver functions

Generated from the description in the previous slide

```c
int SING_send_j_particle(struct grape_j_particle_struct *jp, int index_in_EM);
int SING_send_i_particle(struct grape_i_particle_struct *ip, int n);
int SING_get_result(struct grape_result_struct *rp);
void SING_grape_init();
int SING_grape_run(int n);
```
OpenMP-like compiler

Goose compiler (Kawai 2009)

```c
#pragma goose parallel for icnt(i) jcnt(j) res (a[i][0..2])
    for (i = 0; i < ni; i++) {
        for (j = 0; j < nj; j++) {
            double r2 = eps2[i];
            for (k = 0; k < 3; k++) dx[k] = x[j][k] - x[i][k];
            for (k = 0; k < 3; k++) r2 += dx[k]*dx[k];
            rinv = rsqrt(r2);
            mf = m[j]*rinv*rinv*rinv;
            for (k = 0; k < 3; k++) a[i][k] += mf * dx[k];
        }
    }
```

Translated to assembly language and API calls.
Performance and Tuning example

- HPL (LU-decomposition)
- Gravity

Based on the work by H. Koike (Thesis work)
Matrix-multiplication performance

M=N, K=2048, 640 Gflops  N=K=2048, 450 Gflops

FASTEST single-chip and single-card performance on the planet!
LU-decomposition performance

Speed in Gflops as function of Matrix size
430 Gflops (54\% of theoretical peak) for $N=50K$
LU-decomposition tuning

- Almost every previously known techniques
  - except for the concurrent use of CPU and GDR (we use GDR for column factorization as well...)
  - right-looking form
  - TRSM converted to GEMM

- Several other “new” techniques
  - use row-major order for fast $O(N^2)$ operations
  - Transpose matrix during recursive column decomposition
  - Use recursive scheme for TRSM (calculation of $L^{-1}$)
HPL (parallel LU)

- Everything done for single-node LU-decomposition
- Both column- and row-wise communication hidden
- TRSM further modified: calculate $UT^{-1}$ instead of $T^{-1}U$
- More or less working, tuning still necessary

$N=240K$, 64 nodes: 23Tflops/25KW (est.)
920Mflops/W: Better than #1 in Green500 by 25%.
Gravity kernel performance

(Performance of individual timestep code not much different)

Assembly code (which I wrote) is not very optimized yet... Should reach at least 600 Gflops after rewrite.
Comparison with GPGPU

Pros:

• Significantly better silicon usage: 512PEs with 90nm 40% of the peak DP speed of Tesla C2050 with 1/3 clock and 1/8 transistors
  factor 2 better performance per watt

• Designed for scientific applications reduction, small communication overhead, etc

Cons:

• Higher cost per silicon area... (small production quantity)

• Longer product cycle... 5 years vs 1-2 years

Good implementations of $N$-body code on GPGPU are there (Hamada, Nitadori, ...)

GPGPU performance for $N$-body simulation

- x10 compared to a good SSE code for a $N^2$ code with shared timestep.
- $\sim x5$ for production-level algorithms.
- $\sim x3$ or less for the same price (if you buy GTX295, not Tesla).
- $< x2$ if you are not using Keigo Nitadori’s code.
Keigo Nitadori (discussing the use of GPU)
Next-Generation GRAPE

Question:
Any reason to continue hardware development?

- GPUs are fast, and getting faster
- FPGAs are also growing in size and speed
- Custom ASICs practically impossible to make
Next-Generation GRAPE

Question:
Any reason to continue hardware development?

• GPUs are fast, and getting faster
• FPGAs are also growing in size and speed
• Custom ASICs practically impossible to make

Answer?

• GPU speed improvement might have slowed down
• FPGAs are becoming far too expensive
• Power consumption might become most critical
• Somewhat cheaper way to make custom chips
GPU speed improvement slowing down?

Clear “slowing down” after 2006 (after G80)

Reason: shift to more general-purpose architecture

Discrete GPU market is eaten up by unified chipsets and unified CPU+GPU

But: HPC market is not large enough to support complex chip development
FPGA

“Field Programmable Gate Array”

• “Programmable” hardware
• “Future of computing” for the last two decades....
• Telecommunication market needs: large and fast chips (very expensive)
Power Consumption

1kW \cdot 1 \text{ year} \sim 1000 \text{ USD}

You (or your institute) might be paying more money for electricity than for hardware.

Special-purpose hardware is quite energy efficient.

<table>
<thead>
<tr>
<th>Chip</th>
<th>Design rule</th>
<th>Gflops/W</th>
</tr>
</thead>
<tbody>
<tr>
<td>GRAPE-7(FPGA)</td>
<td>65nm</td>
<td>&gt; 20</td>
</tr>
<tr>
<td>GRAPE-DR</td>
<td>90nm</td>
<td>4</td>
</tr>
<tr>
<td>GRAPE-6</td>
<td>250nm</td>
<td>1.5</td>
</tr>
<tr>
<td>Tesla C2050</td>
<td>40nm</td>
<td>&lt; 2</td>
</tr>
<tr>
<td>Opteron 6128</td>
<td>45nm</td>
<td>&lt; 1.2</td>
</tr>
</tbody>
</table>
Structured ASIC

- Something between FPGA and ASIC
- eASIC: 90nm (Fujitsu) and 45nm (Chartered) products.

- Compared to FPGA:
  - 3x size
  - 1/10 chip unit price
  - non-zero initial cost

- Compared to ASIC:
  - 1/10 size and 1/2 clock speed
  - 1/3 chip unit price
  - 1/100 initial cost (> 10M USD vs ~ 100K)
GRAPEs with eASIC

- Completed an experimental design of a programmable processor for quadruple-precision arithmetic. 6PEs in nominal 2.5Mgates.
- Started designing low-accuracy GRAPE hardware with 7.4Mgates chip.

Summary of planned specs:

- around 8-bit relative precision
- 100-200 pipelines, 300-400 MHz, 2-5Tflops/chip
- small power consumption: single PCIe card can house 4 chips (10 Tflops, 50W in total)
Will this be competitive?

Rule of thumb for a special-purpose computer project:

Price-performance goal should be more than 100 times better than that of a PC available when you start the project.

— x 10 for 5 year development time
— x 10 for 5 year lifetime

Compared to CPU: Okay
Compared to GPU: ??? (Okay for electricity)
Will this be competitive?

Rule of thumb for a special-purpose computer project:

Price-performance goal should be more than 100 times better than that of a PC available when you start the project.

— x 10 for 5 year development time
— x 10 for 5 year lifetime

Compared to CPU: Okay
Compared to GPU: ???? (Okay for electricity)

Will GPUs exist 10 years from now?
GRAPEs and Star-formation simulations

SPH simulation with GRAPE

- Early efforts — Steinmetz, Klessen, Susa
  - Let GRAPE do gravity
  - SPH and all other physics on host
  - Speedup rather limited: Gravity is dominant, but not something like 99.99%...

- Possibility with GRAPE-DR
  - Do SPH interaction (and other physics) on GRAPE-DR (and GPU and other accelerators)
Practical problems with SPH on accelerators

- Neighbor list
  - neighbor lists of different particles are all different
  - Hopeless with an SIMD architecture with hundreds of cores...

- Individual timestep
  - Only a small fraction of particles are integrated with small timesteps
  - reduce the total calculation cost, but reduces parallelism...
Neighbor list

• *If* the accelerator is fast enough, we can use a shared neighbor list to reduce the communication cost.

• Same technique as that we use with treecode (Barnes 89, JM 90).

• roughly 10x more computation to reduce communication by a factor of 10.
Individual timestep

- Wadsley *et al.* (2004): Particles with relatively small timesteps dominate the cost. (But: If you resolve high-density gas, there appear small number of particles with very short timestep)

- With sink particles, there is an artificial lower limit for the timestep.

Traditional individual timestep might be an overkill. Something much simpler might be enough.
Summary

- GRAPE-DR, with programmable processors, has wider application range than traditional GRAPEs.
- Peak speed of a card with 4 chips is 800 Gflops (DP).
- DGEMM performance 640 Gflops, LU decomposition > 400Gflops
- Currently, 128-card, 512-chip system is up and running.
- We return to custom design with structured ASIC for the next generation (budget limitation...)
- GRAPE-DR might be useful for star formation simulation.