The GRAPE project: Current Status

Junichiro Makino
Department of Astronomy,
School of Science, University of Tokyo,
Tokyo 133-0033, Japan
Email: makino@grape.astron.s..u-tokyo.ac.jp

Abstract

We will briefly overview the history and the present status of the GRAPE project. GRAPE (GRAvity piPE) project is a project to design, develop and use special-purpose computers for astrophysical N-body simulations to do large-scale N-body simulations. Our first machine, GRAPE-1, was completed in 1989 and offered the speed of 240 Mflops. Since then, we have continued to develop newer and faster machines, and the newest machine, the GRAPE-6, has achieved the peak speed of 64 Tflops. We will briefly discuss GRAPE-6 and its parallel architecture.

1 Introduction

In this paper, we give an overview of the present status and the future of the GRAPE project[8, 6]. GRAPE (GRAvity piPE) is a special-purpose computer for simulation of particles interacting through gravitational force. Many astronomical objects, from satellites and rings of planets to large scale structures of the universe, can be expressed as systems of particles interacting through gravity (and in some cases other short-range interactions). Since the gravitational force is a long-range force, the calculation of the gravitational interaction dominates the total calculation time.

A naive approach would cost $O(N^2)$ calculation time, where N is the number of particles. It is certainly possible to use fast algorithms such as Barnes-Hut tree algorithm or FMM, which reduce the calculation cost from $O(N^2)$ to $O(N \log N)$ or even $O(N)$. However, even with such schemes, the calculation of particle-particle interaction is still the most expensive part of the calculation.

One approach to accelerate N-body simulation is, therefore, to accelerate the force calculation by means of a dedicated hardware. Since the calculation of gravitational interaction between particles is relatively simple, requiring some 30 arithmetic operations, we can develop a specialized pipelined processor for force calculation. Such a pipelined processor can integrate fairly large number of arithmetic units, which all operate in parallel to achieve high performance. The basic ideal behind our GRAPE (GRAvity piPE) hardware is to develop such a simple and specialized processor and use it in conjunction with general-purpose programmable computer which handles everything other than the force calculation.

In section 2 we give a overview of basic concept of GRAPE hardware. In section 3 we describe the GRAPE-6 system, which is completed recently. Section 4 is for summary.

2 Overview of GRAPE hardware and software

2.1 Astrophysical N-body problem

The master equation for the gravitational N-body problem is given by:

$$\frac{d^2\mathbf{x}_i}{dt^2} = -\sum_{j\neq i} G m_j \frac{\mathbf{x}_j - \mathbf{x}_i}{|\mathbf{x}_j - \mathbf{x}_i|^3}, \qquad (1)$$

where $\mathbf{x}_i$ and m_i are the position and mass of the particle with index i and G is the gravitational constant. The summation is taken over all particles (typically stars) in the system under study.

The number of particles N varies widely depending on the kind of systems studied. A small star cluster like Hyades might consist of several thousand stars. Globular clusters typically contain one million stars, and galaxies hundreds of billions of stars. In the case of galaxies, it is impossible to model them in a star-by-star basis, and we model the distribution of stars in the six-dimensional phase space with much smaller number of particles.

If N is larger than several tens, the calculation cost of the right hand side of equation (1) dominates the total calculation cost. If N is very large, we could use sophisticated algorithms such as the Barnes-Hut treecode[3], FMM[4], or the particle-mesh method. However, with treecode or FMM, the direct evaluation

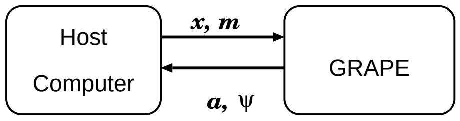

Figure 1: Basic idea of GRAPE.

of gravitational interaction between near-neighbor particles still dominates the total cost. Even in the case of the particle mesh method, if we want high accuracy, near-neighbor interaction must be calculated directly.

A more subtle characteristic of the gravitational force is that it is attractive and diverges for the limit of zero distance. This means that two particles can come arbitrary close, depending on their relative angular momentum, and thus reaching to arbitrary high velocity. In addition, self-gravitating systems tend to be highly inhomogeneous, because the homogeneous state is unstable. These characteristics implies the timescales of particles vary rather widely, and we need a time integration scheme which can vary the timesteps of particles individually [1, 2].

2.2 Special-purpose hardware

Figure 1 show the basic concept of the GRAPE hardware. The GRAPE part performs only the force calculation, and the general-purpose host computer performs all other calculations. With small changes in algorithm and hardware, we can use both the Barnes-Hut treecode (and in principle FMM) and the individual timestep algorithm, with quite high efficiency.

Clearly, the performance of the GRAPE part should be much higher than that of the general-purpose host computer, since otherwize there is no reason to use the GRAPE part. An obvious question is how we can achieve the performance higher than that of the most advanced microprocessors.

Figure 2 shows the maximum number of floating point operations performed per clock cycle for representative microprocessors. The increase in the operation count per cycle was very fast, until it reaches two (one addition and one multiplication) by 1990. Then the increase stalled, and in 1990s there was essentially no increase. Even in the year 2002, there is no microprocessor which can perform more than four floating point operations per cycle.

In other words, practically *all* the increase in the available number of transistors in 1990s was spent for things other than the arithmetic units. There are two reasons for this trend. The first one is the limit of the off-chip memory bandwidth, which is much more difficult to increase than the on-chip processing speed. The second one is the problem of finding sufficient level of parallelism from programs.

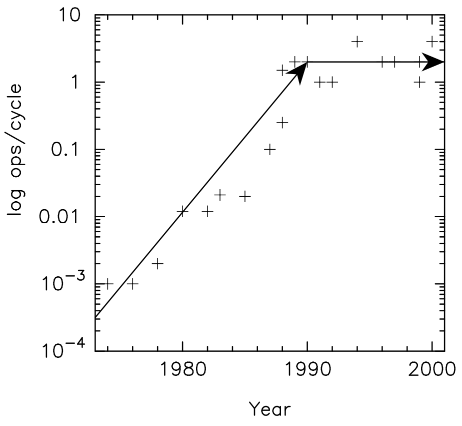

Figure 2: The evolution of the maximum number of floating point operations performed on representative microprocessors, plotted versus time. open circles are the values of actual processors. Arrows indicates the trends.

Thus, even though the theoretical peak performance of microprocessors has been, and will be, increasing, that fact does not imply that they make good use of the device technology. Quite the contrary, they are using ever diminishing fraction of total silicon to do useful floating point operations.

Here is the key for the success of special-purpose computers. If the architecture of the special-purpose computer allows a more efficient use of available transistors than general-purpose microprocessors do, it might be able to achieve better overall performance.

A chip specialized for the calculation of the particle-particle interaction is ideal for the efficient use of the transistors. A pipeline processor would need around 40 arithmetic unit, for the case of the simple gravity. If we implement the Hermite scheme [5], we need to calculate the first time derivative of the force, which adds some 20 arithmetic units. Thus, a single pipeline needs around 60 arithmetic units. We can use these 60 arithmetic units with very small additional control logics, since all connections between arithmetic units are just simple wires (except for some places where pipeline registers are necessary). Thus, if the available transistor count is large enough, it is straightforward to implement a pipeline into a chip. Figure 3 shows such a pipeline.

In the simplest case, this pipeline calculates the force on one particle from all other particles in the system. Thus, at each clock cycle, the pipeline receives the data of one particle.

With the present device technology, we can fit multiple pipelines into a single chip. In this case, we can make use of all pipelines just by letting different pipelines calculate the forces on different processors.

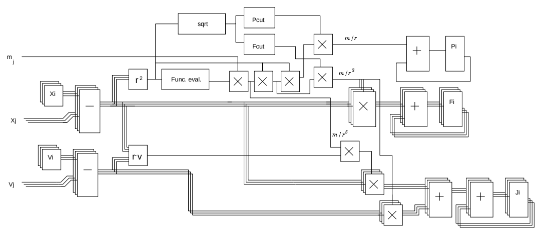

Figure 3: Pipeline to calculate the gravitational force and its time derivative.

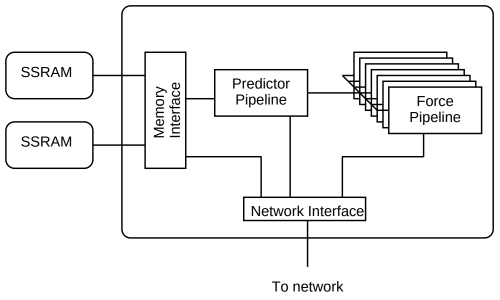

Figure 4: The GRAPE-6 processor chip.

Table 1: GRAPE hardwares

Low-accuracy series

Machine	Date	Speed
GRAPE-1	(89/4 — 89/10)	120 Mflops
GRAPE-1A	(90/4 — 90/10)	240 Mflops
GRAPE-3	(90/9 — 91/9)	14 Gflops
GRAPE-3A	(92/1 — 93/7)	6 Gflops/board
GRAPE-5	(96/8 — 99/4)	5Gflops/chip

High-accuracy series

Machine	Date	Speed
GRAPE-2	(89/8 — 90/5)	40 Mflops
GRAPE-2A	(91/7 — 92/5)	180 Mflops
HARP-1	(92/7 — 93/3)	180 Mflops
GRAPE-4	(92/7 — 95/7)	1 Tflops
MD-GRAPE	(94/7 — 95/4)	1Gflops/chip
GRAPE-6	(97/8 — 02/4)	64 Tflops

Thus, parallel pipelines can be implemented without requiring higher memory bandwidth.

It is also possible to further reduce the required memory bandwidth by letting one pipeline to calculate the force on multiple particles. We call this technique virtual multiple pipeline (VMP) [7]. It is rather similar to simultaneous multithread approach, but the difference is that with our VMP approach the same data from the memory is used by all virtual pipelines. Thus we can reduce the required bandwidth. With multithread architecture, each thread requires its own data stream. Thus, though the requirement for the latency is relaxed, the requirement for the bandwidth remains the same.

Based on this rather simple principle, we started the project to develop GRAPE hardwares in 1989. The first machine, GRAPE-1, was built using ROM chips and fixed-point ALU chips to implement pipeline. GRAPE-3 is out first try to design custom LSI. The GRAPE-3 chip integrated single pipeline, and achieved the speed of 600 Mflops per chip in 1991. GRAPE-4 is a massively parallel version, with 1792 chips each with 640 Mflops. GRAPE-4 was completed in 1995.

Table 1 lists the GRAPE hardwares developed so far. Machines up to GRAPE-4 are described in [6] and references therein.

3 GRAPE-6

GRAPE-6 is the successor of GRAPE-4, which has achieved the peak speed of 1 Tflops in 1995. The processor chip of GRAPE-4 implemented a single force pipeline, which calculates the force between two particles in every three clock cycles using VMP. GRAPE-4 chip was made using a 1μm design rule and the number of transistors was around 400K.

3.1 GRAPE-6 chip and board

For GRAPE-6 chip, we used a 0.25μm design rule. As a result, we could use about 7 million transistors in one chip to implement six pipelines each of which can calculate one interaction per clock cycle. Also, the clock speed was increased from 32 MHz of GRAPE-4 to 90 MHz. These two improvements combined give a factor of 50 improvement in the speed of a chip. In other words, single GRAPE-6 chip offers the speed slightly faster than that of a single GRAPE-4 board with 48 GRAPE-4 chips.

Figure 4 shows the block diagram of the GRAPE-6 chip. A single GRAPE-6 chip integrates all basic functions of a GRAPE system, including the interface to memory chips and interface to the host. The memory interface has the width of 72 bits (with ECC) and operates at 90MHz. It takes 8 clock cycles to read the data of one particle. Therefore, we use one pipeline as eight virtual pipelines. One chip calculates the force on 48 particles in parallel. If we assign 57 operations for calculation of one interaction (this 57 comes from recent Gordon-Bell Prize convention), the peak speed of a chip operating at 90 MHz is 30.8 Gflops.

A GRAPE-6 chip also integrates the control logics and a predictor pipeline. The predictor pipeline is used to predict the positions and velocities of particles at the present time using 4th-order polynomials. This function is used to implement individual timestep algorithms [7].

The number format used in GRAPE-6 chip is rather complex, since different parts of the pipeline use dif-

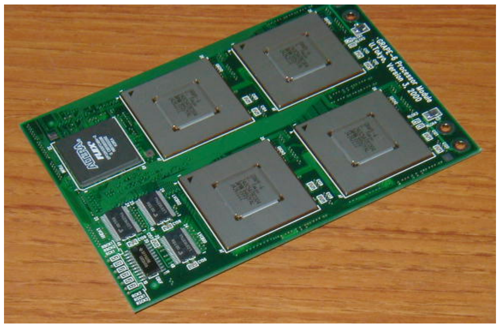

Figure 5: The GRAPE-6 processor module. Four large chips on the top side are the processor chips.

ferent number format. The positions of particles are expressed in 64-bit fixed-point format, and the velocity is expressed in 36-bit floating-point format (1-bit sign, 11-bit exponent and 24-bit mantissa with bias-corrected force-1 rounding). After initial subtractions, calculations for force are done in this 36-bit floating-point format, while that for the time derivative is done with 20-bit mantissa. The final accumulation for force and potential are done in 64-bit fixed-point format with prescaling, so that the accumulator can handle a wide range in the actual value of the force. The prescaling factor can be specified for each particle. For the time derivative, we use the same prescaling but with 32-bit accumulators.

A single processor board of GRAPE-6 houses 32 GRAPE-6 chips, each with its own memory unit. The board is connected to the external interface by one broadcast network for data input and one reduction network for data output. GRAPE-6 board is designed so that different chips calculate the forces on the same particles, but from different set of particles. Thus, it is necessary to have an adder tree to take summation of forces calculated on 32 different chips. This hardware is implemented using FPGA chips from Altera. Since we use the fixed-point format for force and other results, the adder tree is simple and small. Note that we can obtain exactly the same result on machines with different number of GRAPE-6 chips, since fixed-point addition is free of rounding errors.

Figure 5 shows the processor module, which integrates 4 GRAPE-6 chips, 8 memory chips and the first stage of the adder tree. Figure 6 shows the processor board, on which 8 modules are mounted.

3.2 GRAPE-6 system architecture

The entire GRAPE-6 system consists of 64 processor boards. In the present configuration, 64 boards are divided into 4 clusters, each with 16 processor boards and 4 host computers. Within a single cluster, the processor boards are connected to host computers through a rather complex multicast network. The network board (figure 7) implements the multicast network (figure 8). The multicast network is used to reduce the communication between host computers when parallel calculation code is used.

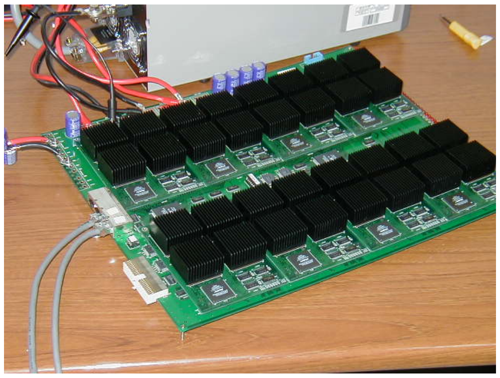

Figure 6: Processor board.

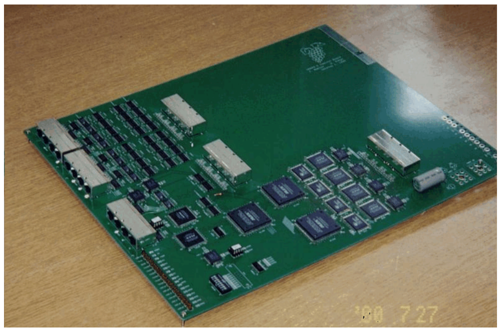

Figure 7: Network board.

In a simple parallel configuration, each processor has its own share of the particles, and calculates the force on them. However, with direct summation, each processor still need the data of particles in other processors, to calculate the forces from them. This means that the communication bandwidth of the single host limits the overall parallel performance.

In order to avoid this problem, we added special network between hosts and GRAPE-6 processor boards. In the normal operation mode, each host controls 4 processor boards. However, in the parallel operation mode, 4 processor boards, which are normally controlled by one host, can receive data from different hosts. By this way, even though each host has only the data of its share, the 4 processor board combined have the data for the entire system, and can calculate the force from entire system.

It is possible to scale up the network to an 8-host, 64-board system, but we chose to have 4 clusters with

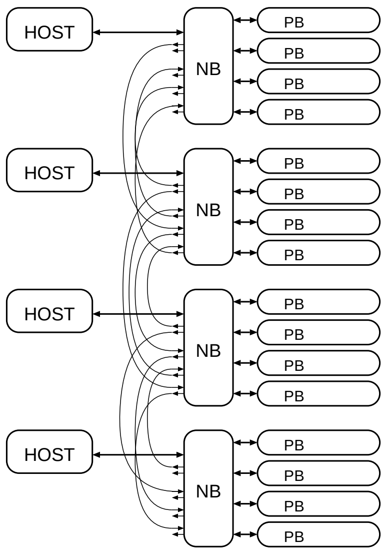

Figure 8: A GRAPE-6 cluster.

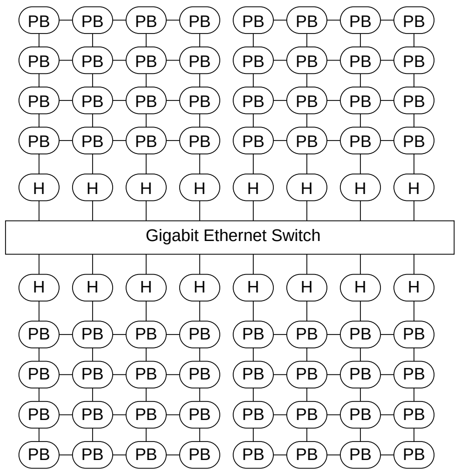

Figure 9: The full GRAPE-6 System.

host computers connected through Gigabit Ethernet. This implies some loss of the performance when we use all 64 boards for single calculation, but allows us more flexible division of the system.

The current GRAPE-6 system consists of 2048 GRAPE-6 chips, for the peak speed of 64 Tflops.

4 Performance

Here, we present the performance of 4×4 single cluster GRAPE-6 for simple individual timestep code and treecode. The host computer is a 1.7 GHz Intel P4 box with i850 chipset, under Linux Kernel version 2.2.17. All timings are done with g++/gcc compilers version egcs-2.91.66.

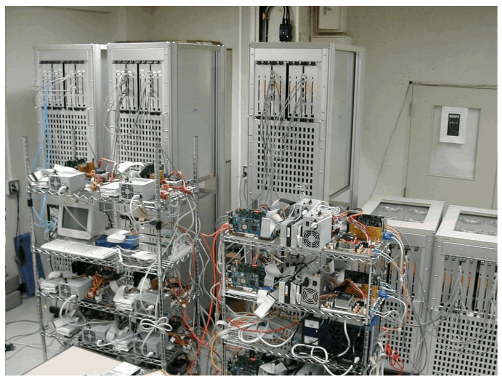

Figure 10: The GRAPE-6 System.

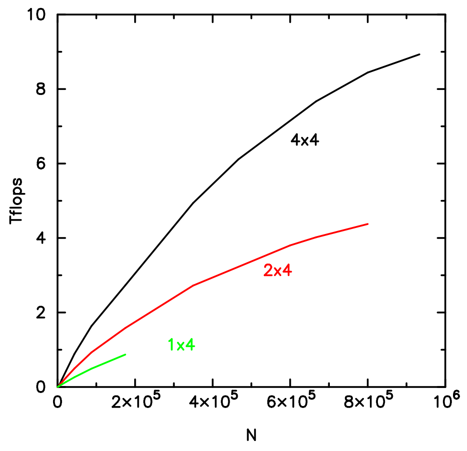

Figure 11: Calculation speed for individual timestep algorithm

Figure 11 shows the speed of the individual timestep algorithm in terms of Tflops, as functions of the number of particles for clusters of different sizes. We used the Plummer model in the standard unit as the initial condition. The softening is 1/64 for all calculations. One can clearly see that the performance scales quite well with the number of hosts and GRAPE boards.

With the network architecture discussed in the previous section, the host computers need not exchange any particle data between them. However, still they need to synchronize at the beginning of the each blockstep, and they also need to compare the global minimum time and the local minimum time, to decide if it can perform time integration of particles in its current block. At present, the hosts communicate with MPICH/p4 software with TCP/IP on 100M bit fast ethernet. The startup overhead of the communication is visible for very small values of N. Of course, for systems with small core, the average number of particles that share the same time becomes small, and communication overhead becomes somewhat larger.

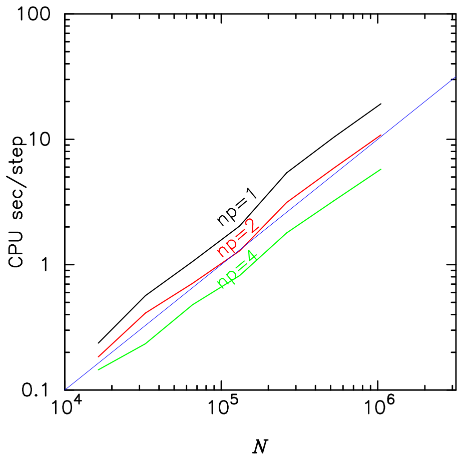

Figure 12: Calculation speed for treecode

Figure 12 shows the performance of treecode. This treecode is a newly written version based on orthogonal recursive multi-section, a generalization of widely used ORB tree which allows a division to arbitrary number of domains in one dimension, instead of allowing only bisection. In this particular timing result, however, the result is the same as that for ORB.

The distribution of the particles is again the Plummer model. One can see that the scaling is again pretty good. 4 processor calculation is 1.9x times faster than 2 processor calculation. We can see that the 100 Mb ethernet is fast enough for the treecode accelerated with GRAPE. Clearly, the performance bottleneck is the speed of the host computer. Over the time GRAPE-6 will be used, we expect the speed of the host computer to be improved by a large factor, which almost directly will be reflected to the speed of the treecode.

5 Summary

We overviewed the GRAPE project, highlighting the newest machine, the GRAPE-6, which achieved the theoretical peak speed of 64 Tflops. We described the network architecture and parallel algorithms used on GRAPE-6, and reported the achieved performance.

Acknowledgments

I would like to thank all of those who involved in the GRAPE project. This work is supported by the Research for the Future Program of Japan Society for the Promotion of Science (JSPS-RFTF97P01102).

References

[1] Aarseth S. J., "Dynamical evolution of clusters of galaxies, I.", *Monthly Notices of Royal Astronomical Society*126, 1963, pp. 223–255.

[2] Aarseth S. J., "Star Cluster Simulations: the State of the Art", *Celestial Mechanics and Dynamical Astronomy*, 73, 1999, pp. 127–137.

[3] Barnes J. and Hut P., "A hierarchical $O(N \log N)$ force calculation algorithm", *Nature*, 324, 1986, pp. 446–449.

[4] Greengard L. and Rokhlin V., "A fast algorithm for particle simulations", *Journal of Computational Physics*, 73, 1987, pp. 325–348.

[5] Makino J. and Aarseth S. J., "On a Hermite integrator with Ahmad-Cohen scheme for gravitational many-body problems", *Publications of the Astronomical Society of Japan* , 44, 1992, pp. 141–151.

[6] Makino J. and Taiji M. *Scientific Simulations with Special-Purpose Computers — The GRAPE Systems.* Chichester:John Wiley and Sons, 1998.

[7] Makino J., Taiji M., Ebisuzaki T., and Sugimoto D., "GRAPE-4: A massively parallel special-purpose computer for collisional N-body simulations", *The Astrophysical Journal* , 480, 1997, pp. 432–446.

[8] Sugimoto D., Chikada Y., Makino J., Ito T., Ebisuzaki T., and Umemura M., "A special-purpose computer for gravitational many-body problems", *Nature*, 345, 1990, pp. 33–35.