GRAPE-4: A TERAFLOPS MACHINE FOR N-BODY SIMULATIONS

MAKOTO TAIJI, JUNICHIRO MAKINO†,
TOSHIYUKI FUKUSHIGE, TOSHIKAZU EBISUZAKI AND DAIICHIRO SUGIMOTO
Department of Earth Science and Astronomy,
† Department of Graphics and Information Science,
College of Arts and Sciences, University of Tokyo
Komaba 3-8-1, Meguro, Tokyo 153, Japan
Internet : taiji@chianti.c.u-tokyo.ac.jp

Abstract. We have developed a massively parallel special-purpose computer system for N-body simulations, GRAPE-4 (GRAvity-PipE 4). The GRAPE-4 system is designed for high-accuracy simulations of dense stellar systems. The GRAPE-4 calculates gravitational forces, their derivatives in time and potential energies. It has a hardware for prediction of positions and velocities, which is used for the individual timestep scheme. Using multi-chip module technology, we integrated 1692 chips of 640 megaflops performance. The peak speed of GRAPE-4 is 1.08 teraflops.

1. Introduction

Gravitational N-body simulation is one of the most important method for the study of star clusters. In N-body simulations, almost all computational time is spent in calculating gravitational forces, since the number of interactions between particles is proportional to the square of the number of particles. Thus, it requires huge computing resources beyond teraflops performance to simulate the post-collapse evolution of real clusters with $10^4 \sim 10^5$ particles. We have developed special purpose computer systems GRAPE (GRAvity PipE) to accelerate for N-body simulations of globular clusters, galaxies, cluster of galaxies, and the universe(Sugimoto *et al.* , 1990; Ebisuzaki *et al.* , 1993). In this paper we describe GRAPE-4, a teraflops massively-parallel special-purpose computer system for gravitational N-

P. Hut and J. Makino (eds.), Dynamical Evolution of Star Clusters, 141–150.

body simulations (Taiji *et al.* , 1994a). GRAPE-4 is suitable for high-accuracy N-body simulations with the hierarchical timestep scheme.

GRAPE calculates only gravitational forces. The host workstation, which is connected to GRAPE, integrates orbits of particles. The GRAPE systems have very long pipelines specialized for the calculations of gravitational interactions. These pipelines perform a few tens of arithmetic operations per cycle. In the large N-body simulations, the force calculation, which costs $O(N^2)$ in time, dominates computational time. On the other hand, both the calculation in a host and the communication between a host and a GRAPE system cost $O(N)$. Therefore, commercial workstations can satisfy the requirements for the communication speed as well as for the calculation speed, although the calculation speed of GRAPE exceeds teraflops.

There are the tree algorithm (Barnes and Hut, 1986), the Particle-Particle Particle-Mesh (PPPM) method (Hockney and Eastwood, 1988), or the fast multipole method (FMM) (Greengard and Rokhlin, 1987), which reduce the cost of force calculations to $O(N \log N)$ or $O(N)$. However, these methods are very difficult to combine with individual or hierarchical timestep schemes, which are essential in simulations of dense stellar systems. Recently McMillan and Aarseth succeeded to implement a tree algorithm with a hierarchical timestep scheme (McMillan and Aarseth, 1993). However, their results indicate that the direct summation is more efficient for simulations at least with $N < 10^5$ on teraflops machines, if we take account of the overhead of parallelization. Thus, the direct summation is still the most efficient for simulations of dense stellar systems. In addition, we can use GRAPE to accelerate these clever algorithms. Makino implemented a tree algorithm (Makino, 1991b) and Brieu et al. implemented the PPPM algorithm on GRAPE-3 (Brieu *et al.* , 1995). Thus, GRAPE accelerates most of particle simulations of large-N self-gravitating systems.

2. What GRAPE-4 calculates

GRAPE-4 calculates forces $\boldsymbol{f}_i$, force derivatives in time $\dot{\boldsymbol{f}}_i$, and potential energies ϕ_i of particle i from the positions $\boldsymbol{x}_i(t_i), \boldsymbol{x}_j(t_j)$, the velocities $\boldsymbol{v}_i(t_i), \boldsymbol{v}_j(t_j)$, and the masses m_j according to the following equation.

$$\begin{aligned} \boldsymbol{f}_i &= \sum_j m_j \frac{\boldsymbol{r}_{ij}}{(r_{ij}^2 + \epsilon^2)^{3/2}}, \\ \dot{\boldsymbol{f}}_i &= \sum_j m_j \left[\frac{\boldsymbol{v}_{ij}}{(r_{ij}^2 + \epsilon^2)^{3/2}} - \frac{3(\boldsymbol{v}_{ij} \cdot \boldsymbol{r}_{ij})\boldsymbol{r}_{ij}}{(r_{ij}^2 + \epsilon^2)^{5/2}} \right], \\ \phi_i &= \sum_j m_j \frac{1}{(r_{ij}^2 + \epsilon^2)^{1/2}}, \end{aligned} \tag{1}$$

https://doi.org/10.1017/S0074180900001479 Published online by Cambridge University Press

where $\boldsymbol{r}_{ij} = \boldsymbol{x}_i(t_i) - \boldsymbol{x}_j(t_i)$, $\boldsymbol{v}_{ij} = \boldsymbol{v}_i(t_i) - \boldsymbol{v}_j(t_i)$, and ϵ is a softening parameter. Force derivatives in time are necessary in fourth-order Hermite scheme, which is more simple and accurate than schemes based on the Newton interpolation (Makino, 1991a; Makino and Aarseth, 1992; Aarseth, 1995). Here, the positions and the velocities at time t_i $(\boldsymbol{x}_j(t_i), \boldsymbol{v}_j(t_i))$ are calculated from those at time t_j using third order predictors

$$\begin{aligned} \boldsymbol{x}_j(t_i) &= \boldsymbol{x}_j + \boldsymbol{v}_j\Delta t + \frac{\boldsymbol{a}_j\Delta t^2}{2} + \frac{\dot{\boldsymbol{a}}_j\Delta t^3}{6}, \\ \boldsymbol{v}_j(t_i) &= \boldsymbol{v}_j + \boldsymbol{a}_j\Delta t + \frac{\dot{\boldsymbol{a}}_j\Delta t^2}{2}, \end{aligned} \tag{2}$$

where $\Delta t = t_i - t_j$. In the hierarchical timestep scheme, we have to evaluate predictors of *all* particles, while we only calculate forces and correctors of particles which share the block step. Therefore, predictor calculations becomes too expensive to be performed by the host computer as N is increased. Since we have to send the predicted coordinates of *all* particles, the communication between the host and GRAPE-4 is also increased. To solve these problems, we added a hardwired pipeline for predictor calculations to GRAPE-4.

It also build the neighbor lists, which contains the indices of particles within a distance h from particle i.

3. System Architecture

Figure 1 shows the block diagram of the GRAPE-4 system. It consists of a host computer, host interface boards (HIB), controller boards (CB), and processor boards (PB). This system has a two-level structure. It has four clusters, and each cluster has one CB and nine PBs. Each PB has 47 HARP (Hermite AcceleratoR Pipe) chips which calculates forces and their derivatives. The HARP chip has the performance of 640 megaflops. Therefore, the system has 1692 HARP chips and the peak performance of 1.08 teraflops. We use Digital Equipment Corporation (DEC) Alpha AXP 3000 series workstation with a TURBOchannel bus as a host computer. However, it is fairly easy to connect the system to another host computer. Since it consumes only 10 kW of power, no heavy cooling system is necessary. It cost about 1.2 million dollars to build the hardware of GRAPE-4. For comparison, a commercial massively-parallel-processors machine with 100–200 gigaflops of theoretical peak speed would cost about 30 million dollars in 1995. Figure 2 shows a photograph of the GRAPE-4 system.

The following two ideas are the keys to achieve such a high performance at a low cost. The first one is the very-long-pipeline architecture, which leads high-performance LSI chips. The second one is massive parallelization. We will explain these ideas in the following sections.

https://doi.org/10.1017/S0074180900001479 Published online by Cambridge University Press

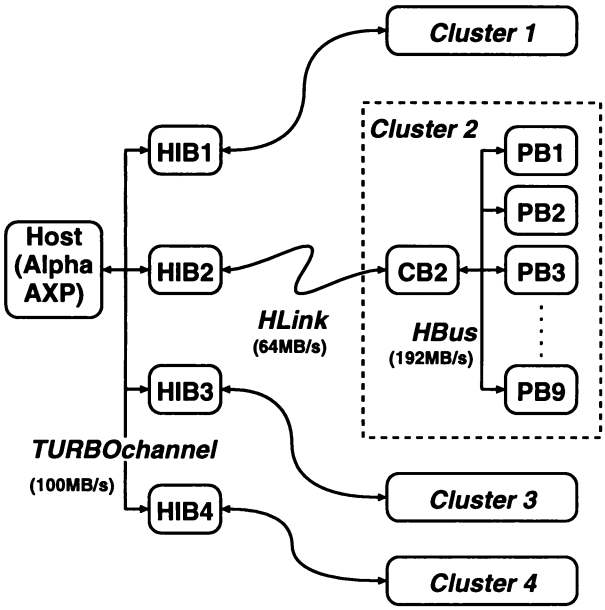

Figure 1. The block diagram of the GRAPE-4 system. HIB: host interface board, CB: controller board, PB: processor board.

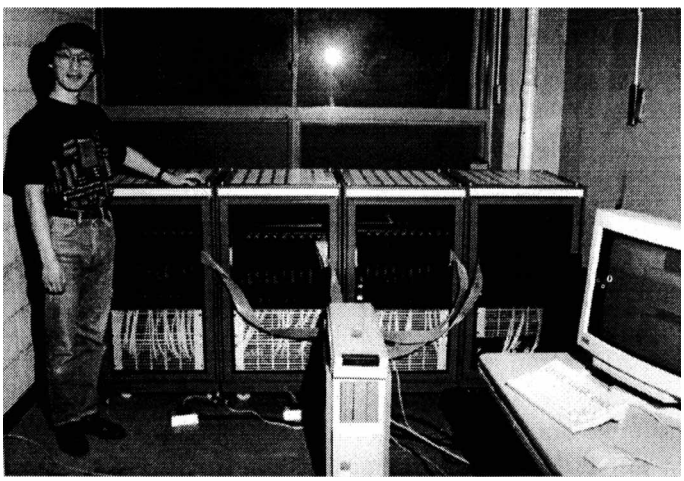

Figure 2. Photograph of the GRAPE-4 system with one of the authors (JM).

4. Very Long Pipeline

Nowadays we can pack $> 10^7$ transistors into one silicon LSI. Since a floating point multiplier and adder need less than $\sim 10^{5.5}$ transistors even in

https://doi.org/10.1017/S0074180900001479 Published online by Cambridge University Press

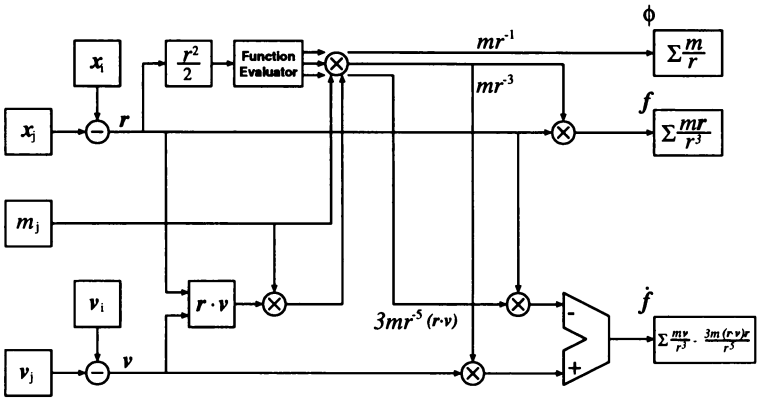

Figure 3. The block diagram of the HARP chip.

double precision, one LSI can have a few tens or hundreds of arithmetic units. It is very difficult to utilize so many arithmetic units efficiently. The very-long-pipeline architecture is one of answers to solve the problem. In this architecture, the whole force calculation are done in one or a few cycles using many arithmetic units. Figure 3 shows the block diagram of the HARP chip, which is developed for the GRAPE-4 system (Taiji *et al.* , 1994b). The HARP chip calculates force $\boldsymbol{f}_i$, force derivative in time $\dot{\boldsymbol{f}}_i$, and potential energy ϕ_i. Each arithmetic unit has the special role and they are connected in the specific sequence to calculate force. These units are fully pipelined so that all of them work simultaneously. The HARP chip calculate one interaction, which usually requires $50 \sim 60$ operations, in three cycles. In other words, it performs about 20 operations per cycle. Thus its performance reaches 640 megaflops at 32 MHz. This is very high performance considering the fact that it is made by 1.0 μm technology, and operates at 32 MHz clock. We have developed another LSI for the evaluation of predictor, the PROMETHEUS chip, which also has the very-long-pipeline architecture.

This pipeline approach is used also in vector processors and RISC processors. Our approach to develop a special-purpose very-long-pipeline LSI has two major advantages over conventional vector processors or RISC pipelines. The first advantage of a special-purpose processor is that we can tune the precision and other features according to the application. The optimization of word lengths reduces the gate count significantly. The second advantage is that it requires rather low input/output (I/O) performance. In the case of general-purpose processors, it is difficult to use many arithmetic units efficiently without increasing the I/O bandwidth between

https://doi.org/10.1017/S0074180900001479 Published online by Cambridge University Press

the memories and the arithmetic units. On the other hand, in the case of a special-purpose processor, arithmetic units are connected in a specific topology following a data flow. Therefore, a LSI chip with many arithmetic units requires a moderate I/O bandwidth. In the case of processors specialized for the N-body simulations, we can use the "virtual multiple pipeline" architecture to further reduce the I/O bandwidth. These advantages make it possible to utilize almost all the transistors available on a chip for calculations. Thus, a special-purpose LSI can house hundreds times more arithmetic units than general-purpose processors.

5. Massive Parallelization

GRAPE-4 consists of 1692 HARP chips to achieve 1.08 teraflops. In this section we explain how we parallelize the task among many pipelines without losing efficiency. As we explained already, GRAPE-4 calculates only forces, their derivative, and potential energies. For example, the force $\boldsymbol{F}_i$ on particle i is given by

$$\boldsymbol{F}_i = \sum_j \boldsymbol{f}_{ij}, \tag{3}$$

where $\boldsymbol{f}_{ij}$ is the force exerted by particle j on particle i. Thus, we have two ways of parallelization: parallelization over index i (i-parallelization) and that over index j (j-parallelization). GRAPE-4 uses both ways of parallelization.

5.1. I-PARALLELIZATION

The parallelization over index i means parallel calculation of forces on many particles. This parallelization scheme is very important in the GRAPE architecture. In this method different pipelines calculate forces on different particles. The coordinates and masses of j-particles are supplied from the predictor (memory) unit to the pipelines. Since these pipelines calculate forces on different particles, we can use the same coordinates of j-particles for calculation of all these forces. Therefore, all the inputs of pipelines can share the common output from the predictor unit. Such a memory architecture cannot be used in general-purpose computers, since it can be applied only for some special applications like classical force calculations.

In the HARP chip we also used the virtual parallelization technique (Makino *et al.* , 1993; Taiji *et al.* , 1994b). The HARP chip accumulates forces, their derivatives in time and potentials of *two* different particles simultaneously. The pipeline accumulate these two forces in turn. Again, we can use the same coordinates and masses of j-particles for calculation of these two particles. The parallelization on an index i can be applied not only for physical pipelines but also for a time-shared virtual pipeline.

https://doi.org/10.1017/S0074180900001479 Published online by Cambridge University Press

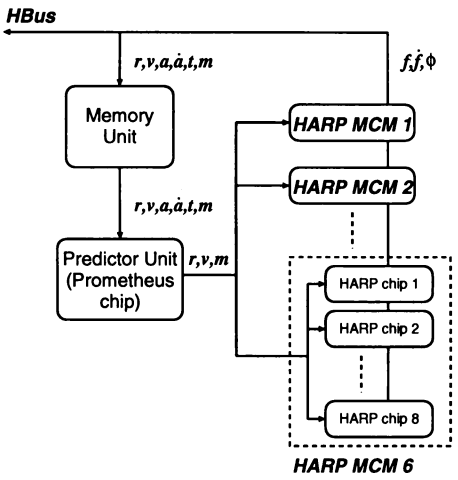

Figure 4. The block diagram of the GRAPE-4 processor board.

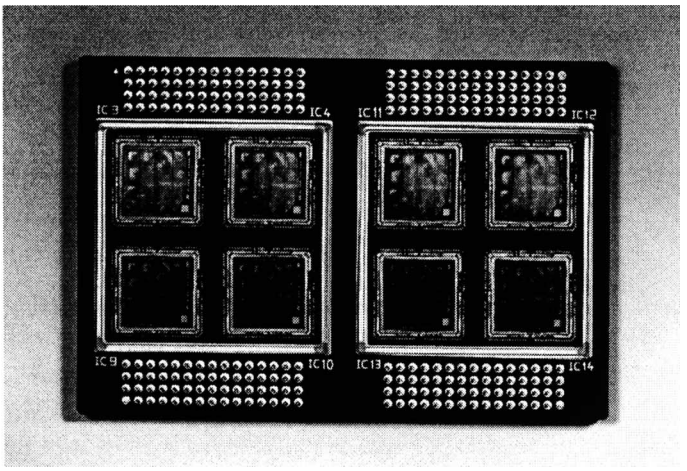

Figure 5. A photograph of a multi-chip module of the HARP chips. It has two cavities and each cavity can house 4 HARP chips.

Figure 4 shows the block diagram of the GRAPE-4 processor board (PB). It has a memory unit, a predictor unit (PROMETHEUS chip), and 47 HARP chips. We have developed a ceramic multi-chip module (MCM) of 8 HARP chips for efficient packaging. Figure 5 shows the photograph of the MCM. The total number of pins is only 240 since most of input/output pins are shared by all of the HARP chips. Thus, an efficient use of MCM is

https://doi.org/10.1017/S0074180900001479 Published online by Cambridge University Press

again the result of i-parallelization. The area of MCM (75 mm $\times$ 110 mm) is the half of 8 single-chip modules. A Multi-chip modules is very effective in reducing the complexity and size of the board. The processor board can mount 6 MCM (48 HARP chips). However, since we used a MCM with one defect for each board, the number of the HARP chips per PB is 47 in the current GRAPE-4 system.

The parallelization on i is quite easy and effective to achieve high performance as explained, however, there are several problems which limit the number of i-parallel pipelines. In the hierarchical timestep scheme, the number of i-parallel pipelines must be smaller than the average number of particles at a block step, $N_{\rm step}$. For simulations with $N < 10^5$, it becomes smaller than the number of (virtual) pipelines in GRAPE-4 (3384), since $N_{\rm step}$ is roughly 10^3 for $N = 10^5$ (Makino, 1991a). Therefore, we can not apply i-parallelization for all pipelines. In addition, it is physically difficult to connect thousands of pipeline chips to a single memory unit. Therefore, we use i-parallelization only to distribute tasks between pipelines on a processor board. The number of i-parallel pipelines is 94 in GRAPE-4.

5.2. J-PARALLELIZATION

To distribute tasks over processor boards, we use parallelization on index j. In the GRAPE-4 system with 36 processor board, The total force is divided into 36 partial forces. Each processor board calculates a different piece of the force. The total force is calculated as a sum of the partial forces

$$F_i = \sum_{j=1}^{n} f_{ij} + \sum_{j=n+1}^{2n} f_{ij} + \cdots + \sum_{j=35n+1}^{36n} f_{ij}. \tag{4}$$

This means that the communication between the host and GRAPE-4 is increased by a factor of 36. Since the summation of partial forces is done by the host, it also increases the amount of calculations in the host.

The controller board solves these problems. It sums up the partial forces calculated on processor boards and sends the result to the host. Thus, it can reduce the communication and the calculations which arise due to the j-parallelization by a factor of 9.

6. Software and Performance

In this section, we describe the procedure to use GRAPE-4, and discuss the behavior of the sustained performance. Since GRAPE-4 calculates only forces, their derivatives, and potential energies, it is easy to write programs for GRAPE-4. The following is the basic procedure to use GRAPE-4.

1. Select 94 particles in the current block step

2. Calculate predictors at the current time for these particles
3. Send $\boldsymbol{x}$ and $\boldsymbol{v}$ of these particles to GRAPE
4. Calculate forces by GRAPE
5. Receive $\boldsymbol{f}, \dot{\boldsymbol{f}}$, and ϕ from GRAPE
6. Calculate correctors and update $\boldsymbol{x}, \boldsymbol{v}$
7. Send $\boldsymbol{x}, \boldsymbol{v}, \boldsymbol{a}$, and $\dot{\boldsymbol{a}}$

More details are described in the paper by Aarseth (1995).

Next, we discuss the sustained performance of GRAPE-4. The computing time per particle per timestep is expressed as

$$T_{\rm step} = T_{\rm host} + T_{\rm comm} + N t_{\rm force} \tag{5}$$

where $T_{\rm host}$ is the time for calculations on the host computer, $T_{\rm comm}$ is the time for communication between the host and GRAPE-4, and $N t_{\rm force}$ is the time for force calculation on processor boards. Since $N t_{\rm force}$ is proportional to the number of particles N but $T_{\rm host}$ and $T_{\rm comm}$ are constant, the sustained performance strongly depends on N. Thus, the efficiency $\eta = T_{\rm force}/T_{\rm step}$ behaves as

$$\eta = \frac{N}{N + N_{\rm half}}, \tag{6}$$

where $N_{\rm half}$ is the number of particle which is necessary to obtain the efficiency of 50%. For the full GRAPE-4 system, we obtained $N_{\rm half} \sim 4 \times 10^5$. $N_{\rm half}$ is scaled roughly to the peak performance of the GRAPE system. More details will be discussed in another paper (Taiji *et al.* , 1996).

7. Summary and future prospects

We have developed GRAPE-4, a special-purpose computer system for astrophysical N-body simulations. It calculates gravitational forces and its time-derivative and has a hardware for predictor calculations. It is suitable for simulations of dense stellar systems using the hierarchical timestep scheme. The GRAPE-4 system is a massively parallelized system of 1692 force calculation pipeline chips. Its peak performance reached at 1.08 teraflops and it cost 1.2 million dollars.

The GRAPE architecture will become more important in future. General-purpose computers cannot utilize efficiently the increasing amount of transistors. On the other hand, GRAPE can use all of them to increase arithmetic units. The speed of GRAPE will increase by 10^3 in ten years, though general-purpose computers become about 100 times faster at the same period. GRAPE-4 used the technology available at 1990. If we start to develop the next generation GRAPE (GRAPE-TNG) at 1998, we will be able to build a pipeline chips with 5 times faster clock and 50 times more transistors. Thus we get a speedup by a factor of 250. Therefore, we will be able

https://doi.org/10.1017/S0074180900001479 Published online by Cambridge University Press

to build a petaflops GRAPE in the end of the twentieth century for the price of 10 million dollars. GRAPE-TNG will open a new era of petaflops computing, and advance our understanding of dynamics of star clusters.

ACKNOWLEDGMENTS

This work was partially supported by the Grant-in-aid for Specially Promoted Research (04102002) of the Ministry of Education, Science, and Culture.

References

* The viewgraphs of the presentation are available at "`http://butterfly.c.u-tokyo.ac.jp:8080/pub/people/taiji/grape4`".

Aarseth, S. J. 1995. *In this volume.*

Barnes, J., and Hut, P. (1986) *Nature*, **324**, 446.

Brieu, P. P., Summers, F. J., and Ostriker, J. P. 1995. *Astrophys. J.*, in press.

Ebisuzaki, T., Makino, J., Fukushige, T., Taiji, M., Sugimoto, D., Ito, T., and Okumura, S. K. (1993) *Publ. Astron. Soc. Japan*, **45**, 269, and references therein.

Greengard, L., and Rokhlin, V. (1987) *J. Comp. Phys.*, **73**, 325–348.

Hockney, R. W., and Eastwood, J. W. (1988) *Computer Simulation Using Particles.* Bristol: Adam Hilger.

Makino, J. (1991a) *Astrophys. J.*, **369**, 200.

Makino, J. (1991b) *Publ. Astron. Soc. Japan*, **43**, 621–638.

Makino, J., and Aarseth, S. J. (1992) *Publ. Astron. Soc. Japan*, **44**, 141.

Makino, J., Kokubo, E., and Taiji, M. (1993) *Publ. Astron. Soc. Japan*, **45**, 349.

McMillan, S. L. W., and Aarseth, S. J. (1993) *Astrophys. J.*, **414**, 200.

Sugimoto, D., Chikada, Y., Makino, J., Ito, T., Ebisuzaki, T., and Umemura, M. (1990) *Nature*, **345**, 33.

Taiji, M., Makino, J., Ebisuzaki, T., and Sugimoto, D. (1994a) *Pages 280–287 of: Proceedings of the 8th International Parallel Processing Symposium.* Los Alamitos: IEEE Computer Society Press.

Taiji, M., Makino, J., Kokubo, E., Ebisuzaki, T., and Sugimoto, D. (1994b) *Pages 302–311 of: Proceedings of the 27th Hawaii International Conference on System Sciences.* Los Alamitos: IEEE Computer Society Press.

Taiji, M., Makino, J., Ebisuzaki, T., and Sugimoto, D. 1996. *Publ. Astron. Soc. Japan*, to be submitted.