Tạp chí Tin học và Điều khiển học, T.23, S.3 (2007), 231–239

SPECIAL-PURPOSE COMPUTER ACCELERATED FAST MULTIPOLE METHOD

NGUYEN HAI CHAU[1], ATSUSHI KAWAI[2], TOSHIKAZU EBISUZAKI[3]

[1] *College of Technology, Vietnam National University, Hanoi. Email:* chaunh@vnu.edu.vn

[2] *Saitama Institute of Technology, Saitama, Japan. Email:* kawai@sit.ac.jp

[3] *Advanced Computing Center, Institute of Physical and Chemical Research (RIKEN), Saitama, Japan. Email:* ebisu@riken.jp

Abstract. This paper presents the implementation of fast multipole algorithm (FMM) on a special-purpose computer GRAPE (GRAvity PipE). FMM is one of the fastest algorithms for the calculation of interaction force in a N-particle system. The FMM's computational complexity is $O(N)$. GRAPE is a special-purpose computer dedicated for the calculation of Coulombic or gravitational force. Its calculation speed is 100-1000 times faster than commodity PC computers at the same cost. However, GRAPE is not able to calculate FMM's multipole expansions. We have found new formulae to express multipole expansions so that they are calculable on GRAPE. Consequently, we successfully accelerated FMM using GRAPE. Our numerical experiments show that for close-to-uniform distribution of particle systems, GRAPE accelerates the FMM by a factor of 3 to 60 for low and high accuracy, respectively.

Tóm tắt. Bài báo này trình bày phương pháp cài đặt và tăng tốc thuật toán khai triển đa cực nhanh (FMM) trên máy tính chuyên dụng GRAPE. FMM là thuật toán tính lực tương tác trong hệ N-chất điểm với độ phức tạp tính toán tuyến tính. GRAPE là một họ máy tính chuyên dụng dành để tính lực tương tác tĩnh điện Culông hoặc lực hấp dẫn với tốc độ cao hơn các máy PC thông thường từ 100 đến 1000 lần. Tuy nhiên GRAPE không thể trực tiếp tính được các biểu thức khai triển đa cực của FMM. Chúng tôi đã tìm ra các công thức mới biểu diễn các khai triển đa cực để có thể thực hiện được trên GRAPE và cài đặt thành công thuật toán FMM trên máy tính chuyên dụng GRAPE. Kết quả thực nghiệm của chúng tôi cho thấy FMM được tăng tốc từ 3 lần (đối với độ chính xác thấp) đến 60 lần (đối với độ chính xác cao) trên máy tính GRAPE.

1. INTRODUCTION

Molecular dynamics (MD) simulations require extremely high calculation cost. The most expensive part of MD is for the calculation of Coulombic force among particles (i.e. atoms and ions). In naive direct-summation algorithm, cost of the force calculation scales as $O(N^2)$, where N is the number of particles.

To reduce the cost of force calculation, fast algorithms such as Barnes-Hut treecode [4] and fast multipole method [9] have been developed. Computational complexity of these algorithms are $O(N.\log N)$ and $O(N)$, respectively. These fast algorithms are widely used in the field of MD simulation [16, 17].

Another approach to accelerate the force calculation is to use hardware dedicated to the calculation of inter-particle force. GRAPE (GRAvity PipE) [19, 22] is one of the most widely used hardware of that kind. Figure 1 shows the basic structure of a GRAPE system. It consists of a GRAPE processor board and a general-purpose computer (hereafter referred to as the host computer). The host computer sends positions and charges of particles to GRAPE. GRAPE calculates the force and sends results back to the host.

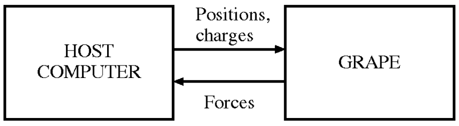

Figure 1. Basic structure of a GRAPE system

A typical GRAPE system performs force calculation 100-1000 times faster than that of commodity computers of the same price. For small-N (say $N \lesssim 10^5$) systems, combination of simple direct-summation algorithm and GRAPE is the fastest calculation scheme. Fast algorithms are not very effective at such a small N. However, for large-N systems, $O(N^2)$ direct-summation becomes expensive, even with GRAPE hardware. Combining one of the fast algorithms and the fast hardware would deliver significant speed up for large-N systems. Makino [18] has successfully implemented a modified treecode [3] on GRAPE, and achieved a factor of 30-50 speed up.

To our understanding, there exists no implementation on GRAPE so far. FMM implementation on a similar dedicated-hardware is reported (MD-ENGINE), but its performance is rather modest [1]. The main reason is that the multipole expansions were not able to calculate on MD-ENGINE and only a small fraction of the FMM's calculation amount was accelerated.

In this paper we describe our implementation of the FMM on GRAPE and its performance. Remaining parts of the paper are organized as follows. Section 2 gives a summary of the FMM and related algorithms. In Section 3, we describe the implementation of our FMM code, which is modified so that it runs on GRAPE as referred in Section 4 . Results of numerical tests of the code are shown in Section 5. Section 6 is devoted to discussion. Section 7 summarizes.

2. FMM AND RELATED ALGORITHMS

The fast multipole algorithm (FMM) was developed by Greengard and Rokhlin [9, 10] for fast force calculation. The main difference of FMM from direct summation algorithm is that FMM calculates *approximation forces between groups of particles* but *exact forces between particles*. The accuracy of approximation force is defined by expansion order p. The larger p is, the more accurate force is obtained. Figure 2 illustrates the calculation of approximation force.

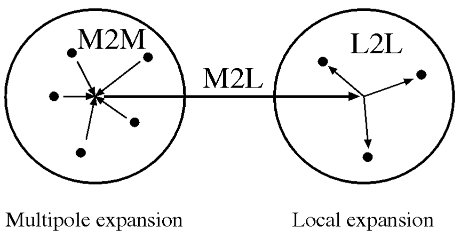

Figure 2. Schematic idea of force approximation in FMM

Forces exerted from left side particles are grouped in one computing element by mutipole to mutipole transition (M2M). Acting force on center of expansion of right side particle is expressed by multipole to local conversion (M2L), then force on each right side particle is calculated by local to local transition (L2L).

Anderson [2] proposed a formulation of the multipole expansion to simplify the implementation of FMM. When potential on the surface of a sphere of radius a is given, the potential Φ at position $\vec{r} = (r, \phi, \theta)$ is expressed as:

$$\Phi(\vec{r}) \approx \sum_{i=1}^{K} \sum_{n=0}^{p} (2n+1) \left(\frac{a}{r}\right)^{n+1} P_n \left(\frac{\vec{s}_i \cdot \vec{r}}{r}\right) \Phi(a\vec{s}_i) w_i \tag{1}$$

for $r \geq a$ (outer expansion) and

$$\Phi(\vec{r}) \approx \sum_{i=1}^{K} \sum_{n=0}^{p} (2n+1) \left(\frac{r}{a}\right)^{n} P_n \left(\frac{\vec{s}_i \cdot \vec{r}}{r}\right) \Phi(a\vec{s}_i) w_i \tag{2}$$

for $r \leq a$ (inner expansion). The function P_n denotes the n-th Legendre polynomial. Here w_i are constant weight values and p (*expansion order*) is the number of untruncated terms. Anderson's method uses outer and inner approximations for M2M and L2L, respectively.

The pseudo-particles multipole method ($\mathrm{P}^2\mathrm{M}^2$) is very similar to Anderson's outer approximation. The difference is that $\mathrm{P}^2\mathrm{M}^2$ uses the mass distribution on the surface of a sphere instead of potential values. Distribution of pseudo-particles is given by spherical t-design [11] and pseudo particle's masses are defined by Makino's approach [20]:

$$Q_j = \sum_{i=1}^{N} q_i \sum_{l=0}^{p} \frac{2l+1}{K} \left(\frac{r_i}{a}\right)^{l} P_l(\cos\gamma_{ij}), \tag{3}$$

where Q_j is charge of pseudoparticle, $\vec{r}_i = (r_i, \phi, \theta)$ is position of physical particle, γ_{ij} is angle between $\vec{r}_i$ and position vector $\vec{R}_j$ of the j-th pseudoparticle [20].

Eq. (3) gives solution for outer expansion. By following a similar approach, we obtained solution for inner expansion [6]:

$$Q_j = \sum_{i=1}^{N} q_i \sum_{l=0}^{p} \frac{2l+1}{K} \left(\frac{a}{r_i}\right)^{l+1} P_l(\cos\gamma_{ij}). \tag{4}$$

We use Eq. (4) to speed up our code as showing in Section 4.

3. IMPLEMENTATION OF THE FMM ON GRAPE

The FMM consists of five stages, namely, tree construction, M2M transition, M2L conversion, L2L transition, and force evaluation. Force-evaluation stage consists of near field and far field evaluation parts.

In the case of original FMM, only the near field part of the force-evaluation stage can be performed on GRAPE. In our implementation (hereafter referred to as code A), we modified the original FMM so that GRAPE can handle M2L conversion stage, which is most time consuming. Table 1 summarizes mathematical expressions and operations used at each calculation stage. In the following we describe stages of the code A.

The tree construction stage has no change. It is performed in the same way as in the original FMM.

At the M2M transition stage, we compute positions and charges of pseudoparticles, instead of forming multipole expansion as in the original FMM. This process is totally carried out on the host computer.

The M2L conversion stage is done on GRAPE. Difference from the original FMM is that we do not use the formula to convert multipole expansion to local expansion. We directly calculate potential values with the use of pseudoparticles.

Table 1. Mathematical expressions and operations used in different implementations of the FMM. Underlined parts run on GRAPE

	Original [10]	Code A (section 3)	Code B (Section 4)
M2M	multipole expansion	P^2M^2	P^2M^2
M2L	M2L conversion formula	**evaluation of pseudoparticle potential**	
L2L	local expansion	Anderson's method	P^2M^2
Near field force	**evaluation of physical-particle force**		
Far field force	evaluation of local expansion	Eq. (5)	**evaluation of pseudoparticle force**

The L2L transition is done in the same way as Anderson has done using Eq. (2).

The near field contribution is directly calculated by evaluating the particle-particle force. GRAPE handles this part.

Using Eq. (2), we obtain the far field potential on a particle at position $\vec{r}$. Consequently, far field force is calculated using derivative of Eq. (2):

$$-\nabla\Phi(\vec{r}) = \sum_{i=1}^{K}\sum_{n=0}^{p}\left(n\vec{r}P_n(u) + \frac{u\vec{r} - \vec{s}_i\, r}{\sqrt{1-u^2}}\nabla P_n(u)\right)(2n+1)\frac{r^{n-2}}{a^n}g(a\vec{s}_i)w_i, \tag{5}$$

where $u = \vec{s}_i \cdot \vec{r}/r$. All the calculation at this stage is carried out on the host computer.

4. FURTHER IMPROVED IMPLEMENTATION

With the modification described in Section 3, we have succeessfully to put the bottleneck, namely, the M2L conversion stage, on GRAPE. The overall calculation of the FMM is significantly accelerated. Now the most expensive part is the far field force evaluation. Eq. (5) is complicated and of which, the evaluation would take rather a big fraction of the overall calculation time [5].

If we can convert a set of potential values into a set of pseudoparticles at marginal calculation cost, force from those pseudoparticles can be evaluated on GRAPE, and the bottleneck will disappear. In order for this conversion, we have newly developed a systematic procedure (hereafter A2P conversion).

We have implemented yet another version of FMM (hereafter code B). In the code B, we use A2P conversion to obtain a distribution of pseudoparticles that reproduces the potential field given by Anderson's inner expansion. Once the distribution of pseudoparticles is obtained, L2L stage can be performed using inner-P^2M^2 formula (Eq. (4)), and then the force evaluation stage is totally done on GRAPE (the final column of Table 1). Procedure of A2P conversion is as follows.

At the first step, we distribute pseudoparticles on the surface of a sphere with radius b using the spherical t-design. Here, b should be larger than the radius of the sphere a on which Anderson's potential values $\Phi(a\vec{s}_i)$ are defined. According to Eq. (4), it is guaranteed that we can adjust the charge of the pseudoparticles so that $\Phi(a\vec{s}_i)$ are reproduced. Therefore, the relation

$$\sum_{j=1}^{K} \frac{Q_j}{|\vec{R}_j - a\ \vec{s}_i|} = \Phi(a\ \vec{s}_i) \tag{6}$$

should be satisfied for all $i = 1, ..., K$. Using a matrix $\mathcal{R} = \{1/|\vec{R}_j - a\ \vec{s}_i|\}$ and vectors $\vec{Q} = {}^T[Q_1, Q_2, ..., Q_K]$ and $\vec{P} = {}^T[\Phi(a\ \vec{s}_1), \Phi(a\ \vec{s}_2), ..., \Phi(a\ \vec{s}_K)]$, we can rewrite Eq. (6) as

$$\mathcal{R}\vec{Q} = \vec{P}. \tag{7}$$

In the next step, we solve the linear Eq. (7) to obtain charges Q_j. By numerical experiment, we found that appropriate value of radius b is about 6.0, for particles inside a cell with side length 1.0. Anderson specified in his paper [2] that a should be about 0.4.

5. NUMERICAL TESTS

Here we show the performance of the FMM code B measured on MDGRAPE-2 [23]. MDGRAPE-2 is one of the latest hardware of the GRAPE series. It is developed for MD simulation and has additional functions so that it can handle forces which do not decay as $1/r^2$, such as Van der Waals force. However, in our test we use MDGRAPE-2 only to calculate Coulombic force and potential. The additional functions are not used.

We used two GRAPE systems for numerical tests. The first one contains a MDGRAPE-2 board (64 pipelines, 192 GFlops) and a host computer COMPAQ DS20E (Alpha 21264 667MHz). The second one consists of one MDGRAPE-2 board (16 pipelines, 48GFlops) and a self-assembled host computer (Pentium 4 2.2GHz, Intel D850 motherboard). We refer the former system as “system I”, and the latter as “system II”.

In the tests, we distributed particles uniformly within a unit cube centered at origin, and evaluated force on all particles. We measured the calculation time at high ($p = 5$) and low ($p = 1$) accuracy, with and without GRAPE. The finest refinement level $l_{\max}$ is set to $l_{\max} = 4$ and 5, for running with and without GRAPE, respectively. These values are chosen so that the overall calculation time is minimized.

Results are shown in Figures 3, 4, 5, 6 and Tables 2, 3. Figures 3 and 5 are results of system I. Figure 4, 6 and tables 2, 3 are of system II.

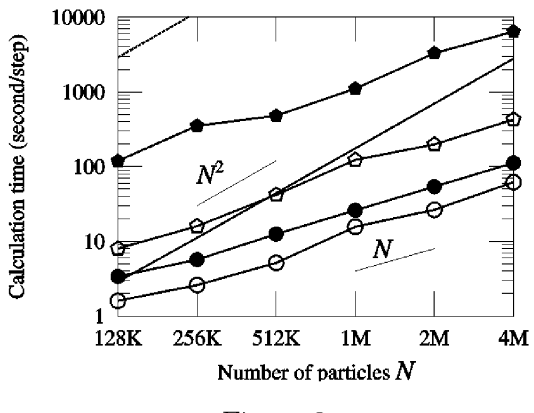

Figure 3

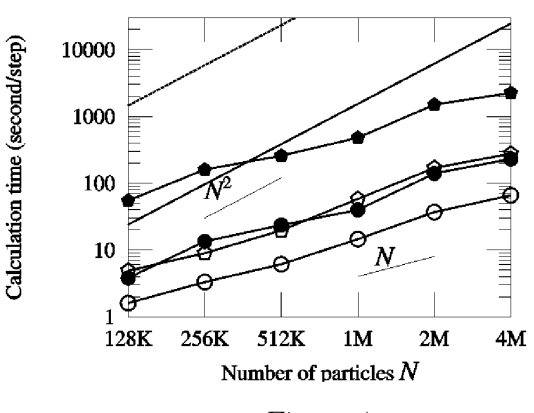

Figure 4

Figure 3: Force calculation time of FMM and direct-summation algorithm on system I. Circles are performance of FMM on MDGRAPE-2. Pentagons are that on the host computer.

Open and filled symbols are for low ($p = 1$) and high accuracy ($p = 5$), respectively. Solid and dashed curves without symbols are performance of direct method on MDGRAPE-2 and the host computer, respectively.

Figure 4: Same as Figure 3, but on system II.

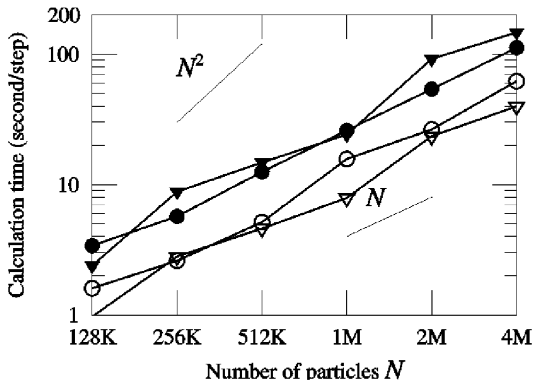

Figure 5

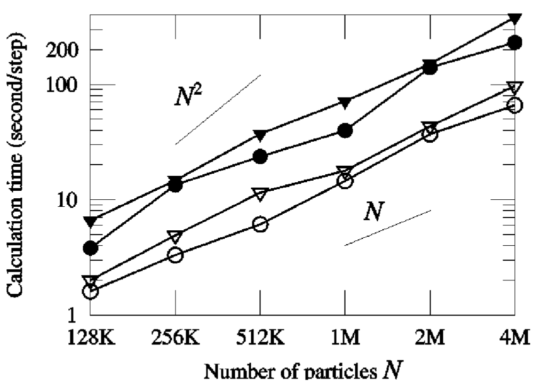

Figure 6

Figure 5. Comparison of force calculation time for FMM and treecode on MDGRAPE-2 on system I. Circles are performance of FMM on MDGRAPE-2. Triangles are that of the treecode on MDGRAPE-2. Open and filled symbols are for low and high accuracy, respectively. Parameter pairs (p, θ) to obtain low and high accuracy of the treecode are $(1, 1.0)$ and $(2, 0.33)$, respectively.

Figure 6. Same as Figure 5, but on system II.

In Figures 3 and 4, calculation time of code B is plotted against the number of particles N. Our code scales as $O(N)$ while direct method scales as $O(N^2)$. On system I, runs with GRAPE are faster than those without GRAPE by a factor of 5 and 60 for low and high accuracy, respectively. On system II, the speed-up factors are 3 and 14.5. Since calculation amount of the M2L stage becomes more significant at higher p (Table 2), the speed up factor is larger for higher accuracy.

Table 2. Pair wise interaction count for 1M particle run

	With GRAPE	$(l_{\max} = 4)$	Without GRAPE	$(l_{\max} = 5)$
Accuracy	Low	High	Low	High
M2L	6.8×10^5	2.8×10^8	7.7×10^6	3.2×10^9
Force evaluation				
far field	1.6×10^8	9.1×10^9	1.8×10^8	5.6×10^9
near field	6.1×10^9	6.1×10^9	8.2×10^8	8.2×10^8

Table 5 shows the breakdown of the calculation time for 1M particles runs. We can see GRAPE significantly accelerates the M2L and force evaluation parts. The overall performance of our implementation is limited by the speed of communication bus between the host and GRAPE, rather than the speed of GRAPE itself. For further acceleration, we need to switch from legacy PCI bus (32-bit/33MHz) to the faster buses, such as PCI-X, or PCI Express.

Figure 5 shows the calculation time of our FMM code and the treecode [15], both running on GRAPE. The order of the multipole expansion p and the opening angle θ for the treecode is set to $(p, \theta) = (1, 1.0)$ and $(2, 0.33)$ for low and high accuracy, respectively. These values are chosen so that the treecode gives roughly the same root-mean-square (RMS) force error as that of the FMM. The RMS force errors at low and high accuracy are $\sim 5 \times 10^{-2}$ and $\sim 2 \times 10^{-5}$, respectively.

Table 3. Time breakdown for 1M particles run on system II

	With GRAPE ($l_{\max} = 4$)		Without GRAPE ($l_{\max} = 5$)	
Accuracy	Low	High	Low	High
Tree construction	1.05	1.03	1.02	1.06
Building neighbor and interaction lists	0.06	0.08	1.89	2.31
M2M	0.22	5.92	0.26	5.97
M2L				
Host	0.01	0.21	0.36	133.88
Data transfer	0.16	4.78	0	0
GRAPE	0.0004	0.18	0	0
	0.17	5.17	0.36	133.88
L2L	0.01	0.34	0.05	4.11
Force evaluation				
Host	0.78	0.97	54.35	330.99
Data transfer	8.57	17.37	0	0
GRAPE	3.92	9.48	0	0
	13.27	27.82	54.35	330.99
Total	**14.78**	**40.36**	**57.93**	**478.32**

We can see that the performance of our FMM code and the treecode is mostly the same. The FMM is better than the treecode at high accuracy, and worse at low accuracy.

6. DISCUSSION

6.1. Comparison with other implementation

We compared the performance of our FMM implementation (code B) with Wrankin's Distributed Parallel Multipole Tree Algorithm (DPMTA) [24].

As for Wrankin's code, we measured the performance on the system II, using the serial version of DPMTA 3.1.3 available at `http://www.ee.duke.edu/~wrankin/Dpmta/`.

For the measurement, particles are distributed in a unit cube. The expansion order and other parameters of each code are chosen so that relatively high accuracy ($\sim 10^{-5}$) is achieved, and the performance is optimized.

Table 4 summarizes the comparison. Using GRAPE, our code outperforms Wrankin's codes by tenfold. Without GRAPE, our code is slower than Wrankin's code by a factor of 1.1-1.4, mainly because our code requires larger number of operation counts so that it takes full advantage of GRAPE.

Table 4. Performance comparison with Wrankin's code

	Wrankin's code	Our code	
N		with GRAPE	without GRAPE
98,304	33.2	2.9	34.1
393,216	190.2	16.4	196.5
1,572,864	629.6	64.0	878.8

7. SUMMARY

Using special-purpose hardware GRAPE, we have successfully accelerated the FMM. In order to take full advantage of the hardware, we have modified the original FMM using the Anderson's method, the pseudoparticle multipole method, and two conversion techniques we have newly invented. The experimental results show that GRAPE accelerates the FMM by a factor of 3 to 60, and the factor increases as the required accuracy becomes higher. Comparison with the treecode shows that our FMM is faster at high accuracy, while the treecode is faster at low accuracy.

Acknowledgement

Thanks are due to Dr. T. Iitaka at the Institute of Physical and Chemical Research (RIKEN) for his valuable advices.

We are grateful to Prof. J. A. Smith from Bridge to Asia for refining the manuscript.

This work is supported by the Advanced Computing Center, RIKEN and the College of Technology, Vietnam National University, Hanoi under QC.05.01 project. Part of this work is carried out while N. H. Chau is a contract researcher of RIKEN and A. Kawai is a special postdoctoral researcher of RIKEN.

REFERENCES

[1] T. Amisaki, S. Toyoda, H. Miyagawa, K. Kitamura, Development of hardware accelerator for molecular dynamics simulations: a computation board that calculates nonbonded interactions in cooperation with fast multipole method. *Journal of Computational Chemistry* **24** (2003) 582–592.

[2] C. R. Anderson, An implementation of the fast multipole method without multipoles, *SIAM J. Sci. Stat. Comput.* **13** (4) (1992) 923–947.

[3] J. E. Barnes, A modified tree code: Don't laugh; It runs, *Journal of Computational Physics* **87** (1990) 161–170.

[4] J. E. Barnes, P. Hut, A hierarchical $O(N \log N)$ force-calculation algorithm, *Nature* **324** (1986) 446–449.

[5] N. H. Chau, A. Kawai, T. Ebisuzaki, Implementation of fast multipole algorithm on special-purpose computer MDGRAPE-2, *Proceedings of the 6th World Multiconference on Systemics, Cybernetics and Informatics SCI2002*, (Orlando, Colorado, USA, July 14-18, 2002) 477–481.

[6] N. H. Chau, A new formulation for fast calculation of far field force in molecular dynamics simulations, *Journal of Science*, Vietnam National University (2007).

[7] T. Fukushige, A. Kawai, J. Makino, Structure of dark matter halos from hierarchical clustering. III. Shallowing of the inner cusp, *Astrophysical Journal* **606** (2004) 625–634.

[8] T. Fukushige, J. Makino, A. Kawai, Constructing PC-GRAPE Cluster Using GRAPE-6A, submitted to *Publications of the Astronomical Society of Japan*

[9] L.Greengard, V. Rokhlin, A fast algorithm for particle simulations, *Journal of Computational Physics* **73** (1987) 325–348.

[10] L. Greengard, V. Rokhlin, A new version of the fast multipole method for the Laplace equation in three dimensions, *Acta Numerica* **6** (1997) 229–269.

[11] R. H. Hardin, N. J. A. Sloane, McLaren's improve snub cube and other new spherical design in three dimensions, *Discrete and Computational Geometry* **15** (1996) 429–441.

[12] Y. Hu, S. L. Johnsson, A data-parallel implementation of $O(N)$ hierarchical N-body methods, Intl. J. of Supercomput. Appl. and High Perf. Comput., 10(1), 1996, 3-40.

[13] A. Kawai, J. Makino, Pseudoparticle multipole method: A simple method to implement a high-accuracy treecode, *The Astrophysical Journal* **550** (2001) L143–L146.

[14] A. Kawai, J. Makino, High-accuracy treecode based on pseudoparticle multipole method, *Proceedings of the 208th Symposium of the International Astronomical Union* (Tokyo, Japan, July 10-13, 2001) 305–314.

[15] A. Kawai, J. Makino, T. Ebisuzaki, Performance analysis of high-accuracy tree code based on the pseudoparticle multipole method, *The Astrophysical Journal Supplement* **151** (2004) 13–33.

[16] P. Lakshminarasimhulu, J. D. Madura, A cell multipole based domain decomposition algorithm for molecular dynamics simulation of systems of arbitrary shape, *Computer Physics Communications* **144** (2002) 141–153.

[17] J. A. Lupo, Z. Q. Wang, A. M. McKenney, R. Pachter, W. Mattson, A large scale molecular dynamics simulation code using the fast multipole algorithm (FMD): performance and application, *Journal of Molecular Graphics and Modelling* **21** (2002) 89–99.

[18] J. Makino, Treecode with a special-purpose processor, *Publ. Astron. Soc. Japan* **43** (1991) 621–638.

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

[20] J. Makino, Yet another fast multipole method without multipoles - Pseudoparticle multipole method, *Journal of Computational Physics* **151** (1999) 910–920.

[21] T. Narumi, A. Kawai, T. Koishi, An 8.61 Tflop/s molecular dynamics simulation for NaCl with a special-purpose computer: MDM, *Proceedings of SC2001*, (Denver, Colorado, USA, November 10-16, 2001) in CD-ROM.

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

[23] R. Susukita, T. Ebisuzaki, B. G. Elmegreen, H. Furusawa, K. Kato, A. Kawai, Y. Kobayashi, T. Koishi, G. D. McNiven, T. Narumi, K. Yasuoka, Hardware accelerator for molecular dynamics: MDGRAPE-2, *Computer Physics Communications* **155** (2003) 115–131.

[24] W. T. Wrankin, J. A. Board, A Portable Distributed Implementation of the Parallel Multipole Tree Algorithm, *Proceedings of the Fourth IEEE International Symposium on High Performance Distributed Computing–HPDC 95*, (The Ritz Carlton Pentagon City, Virginia, USA, August 1-4, 1995) 17–22.

Received on June 28 - 2006
Revised on August 6 - 2007