Astroinformatics
Proceedings IAU Symposium No. 325, 2016
M. Brescia, S.G. Djorgovski, E. Feigelson,
G. Longo & S. Cavuoti, eds.

doi:10.1017/S174392131700045X

Supercomputer simulations of structure formation in the Universe

Tomoaki Ishiyama

Institute of Management and Information Technologies, Chiba University, 1-33, Yayoi-cho, Inage-ku, Chiba, 263-8522, Japan
email: ishiyama@chiba-u.jp

Abstract. We describe the implementation and performance results of our massively parallel MPI† /OpenMP‡ hybrid TreePM code for large-scale cosmological N-body simulations. For domain decomposition, a recursive multi-section algorithm is used and the size of domains are automatically set so that the total calculation time is the same for all processes. We developed a highly-tuned gravity kernel for short-range forces, and a novel communication algorithm for long-range forces. For two trillion particles benchmark simulation, the average performance on the fullsystem of K computer (82,944 nodes, the total number of core is 663,552) is 5.8 Pflops, which corresponds to 55% of the peak speed.

Keywords. dark matter, halo, numerical

1. Introduction

According to the recent observation of the cosmic microwave background (e.g., Planck Collaboration *et al.* 2014), dark matter exists five times as much as baryonic matter. Dark matter is dominant as the source of gravity. Structure formation of the Universe proceeds hierarchically. Smaller-scale dark matter structures formed first from initial density fluctuations imprinted shortly after the Big Bang, and they then merge into larger-scale structures (Figure 1). At about 100 million years after the Big Bang, first stars and galaxies began to form in massive dark matter structures. Therefore, studying dark matter structures is important to understand not only what dark matter is, but also origins of galaxies.

Cosmological N-body simulations have been playing a pivotal role to study the nonlinear structure formation in the Universe. Since dark matter structures exist in wide mass ranges larger than 20 orders of magnitude (earth mass to clusters of galaxies), huge simulations by supercomputers are demanded. Achieving large simulations efficiently on modern supercomputers (more than 10,000 CPUs) is challenging work. Innovative numerical algorithm and optimizing simulation codes are necessary.

In cosmological N-body, simulations, a dark matter particle travels under the gravity from all the other particles in the simulation box. The simplest algorithm to calculate the force of a particle is to calculate the forces from the other $N-1$ particles, where N is the total number of particles in the system. This algorithm is called the direct summation, which is unpractical for large N, since the cost to calculate the forces is proportional to the square of N. Therefore, to accelerate the calculation, sophisticated algorithms with some approximation are commonly adopted in cosmological N-body simulations.

The tree method (Barnes & Hut 1986; Barnes 1990) is the most popular algorithm for N-body simulations. The concept of the tree method is to employ a hierarchical

† Message Passing Interface
‡ Open Multi-Processing

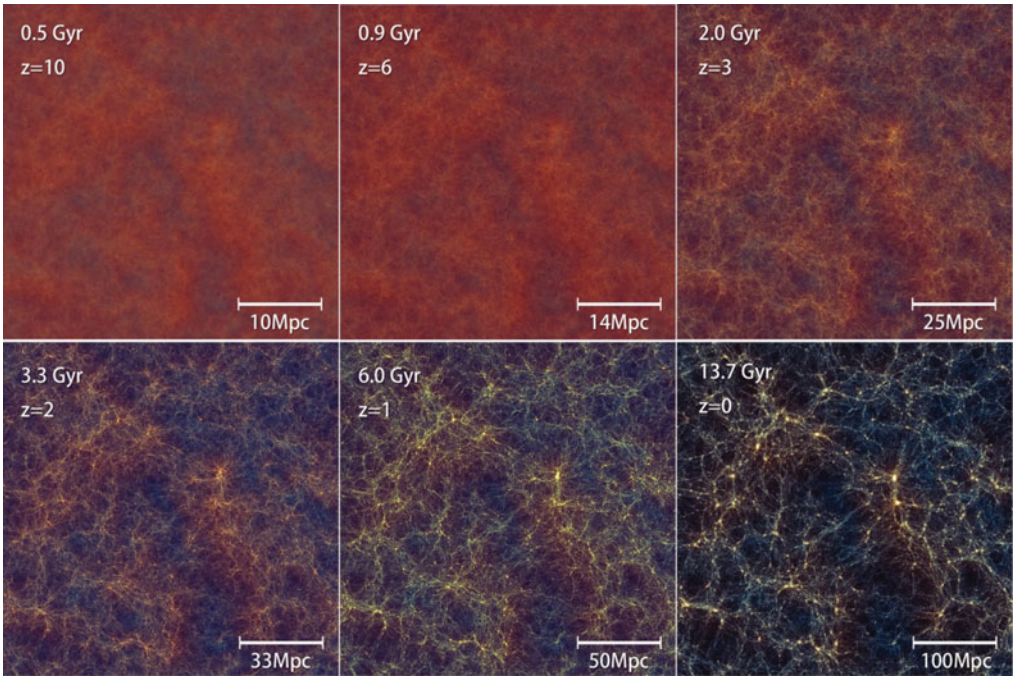

Figure 1. Evolution of large scale structures followed by a cosmological N-body simulation. Brightness is corresponding to dark matter density.

oct-tree structure to represent a system. The force from particles in a tree branch to one particle can be evaluated by calculating the multipole expansion, if the error is sufficiently small (if the branch and the particles are separated enough). If not, the force is evaluated by summing up forces from eight sub-branches. By recursively applying this procedure, the total calculation cost becomes total force on a particle is $\mathcal{O}(N \log N)$, which is drastically reduced from the $\mathcal{O}(N^2)$ cost of the direct summation. Thus, the most codes for cosmological simulations use the tree algorithm.

Yet another way to reduce the calculation cost for cosmological simulations is the PM (Particle Mesh) algorithm. The PM algorithm can calculate the gravitational potential on a regular grid. The mass density at a grid point is calculated by assigning the masses of nearby particles by some kernel function. Then, the Poisson equation is solved using FFT (Fast Fourier Transform). Finally, the force on a particle position is calculated by differentiating and interpolating the potential on the mesh. For details, see Hockney & Eastwood (1981).

In general, the PM algorithm is much faster but less accurate than the tree algorithm since the spatial force resolution is limited by the size of the mesh. In order to overcome this problem, hybrid algorithm such as the TreePM (Tree Particle-Mesh) algorithm has been developed (e.g., Xu 1995; Bode *et al.* 2000; Bagla 2002; Dubinski *et al.* 2004; Springel 2005; Yoshikawa & Fukushige 2005; Ishiyama *et al.* 2009; Ishiyama *et al.* 2012). In this algorithm, the gravitational force is split into two components, short- and long-range forces, which are calculated by the tree and PM algorithm, respectively.

In the TreePM algorithms, the periodic boundary condition is naturally satisfied and high spatial resolution can be achieved. Since the calculation cost of TreePM is $\mathcal{O}(N \log N)$, this algorithm is becoming popular and is used in a number of recent large cosmological N-body simulations (Figure 2).

https://doi.org/10.1017/S174392131700045X Published online by Cambridge University Press

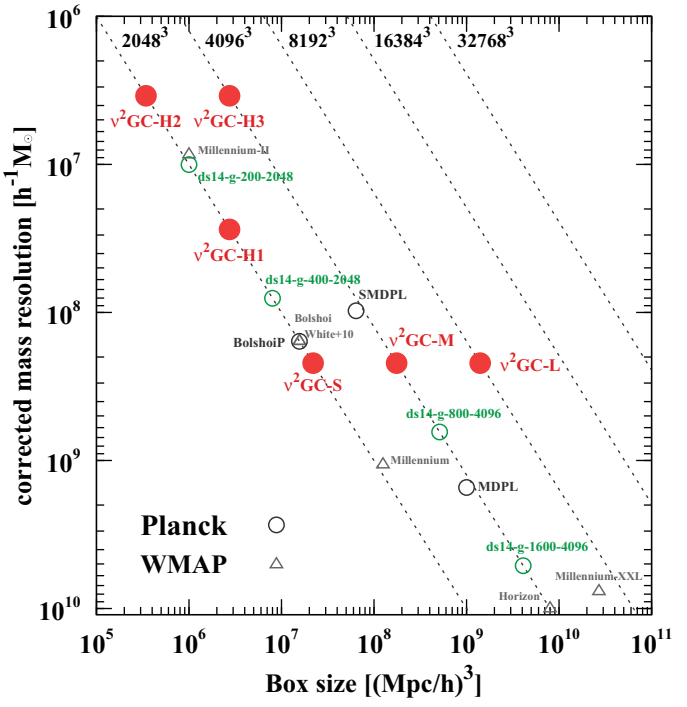

Figure 2. Mass resolution versus simulation volume of recent large cosmological N-body simulations. The mass resolution of each simulation is corrected as the cosmological parameters of all simulations are the same as those we chose in the ν^2GC simulations (Ishiyama *et al.* 2015), indicated as six filled circles ($\Omega_0 = 0.31, \Lambda_0 = 0.69, h_0 = 0.68$). The number of particles along the five dashed lines is constant from 2048^3 to 32768^3. Circles show simulations based on the Planck cosmology (Planck Collaboration *et al.* 2014). The four thin open circles denote four of the five Dark Sky Simulations (DSS; Skillman *et al.* (2014)). The mass resolution of the rest of the DSS simulations is below the range of this figure. The three thick black circles are the BolshoiP, MDPL and SMDPL simulations (Klypin *et al.* 2014). Gray open triangles show simulations based on the WMAP cosmology (e.g., Komatsu *et al.* 2009) by other groups, Millennium simulation (Springel *et al.* 2005), Horizon (Teyssier *et al.* 2009), Millennium-II (Boylan-Kolchin *et al.* 2009), White+10 (White *et al.* 2010), Bolshoi (Klypin *et al.* 2011), Millennium-XXL (Angulo *et al.* 2012).

We introduce our MPI/OpenMP hybrid TreePM implementation GreeM (Ishiyama *et al.* 2009; Ishiyama *et al.* 2012), which is a massively parallel TreePM code based on the implementation of Yoshikawa & Fukushige (2005) for large cosmological N-body simulations, and present performance results up to two trillion particles. The numerical simulations were carried out on K computer at the RIKEN Advanced Institute for Computational Science. It consists of 82,944 SPARC64 VIIIfx oct-core processors with the clock speed of 2.0 GHz (the total number of core is 663,552) and 1.3PB of memory. The peak performance is 10.6 Pflops.

2. Massively Parallel TreePM Code, GreeM

In this section, we briefly introduce three features that enable to achieve high parallel scalability and performance, namely, our domain decomposition algorithm, optimized particle-particle force loop, and relay mesh method. The more details of GreeM are described in Ishiyama *et al.* (2009); Ishiyama *et al.* (2012).

https://doi.org/10.1017/S174392131700045X Published online by Cambridge University Press

2.1. *Domain Decomposition*

In the cosmological N-body simulations, it is difficult to achieve good load balance for the following reason. In cosmological N-body simulations, the particles initially distribute nearly uniformly. The small density fluctuations gradually grow by gravity and form numerous dense structures everywhere. The density of such structures are typically much higher than the average. As a result, the calculation cost of the short-range part becomes highly imbalanced, if the domain decomposition is static, in other words, its geometry of each domain is time invariable and is the same for all domains.

We use a 3-D recursive multi-section domain decomposition (Makino 2004) to overcome this problem. In this method, the shape of a domain is rectangular. To determine the geometries of domains. we use the sampling method (Blackston & Suel 1997), which can drastically reduce the amount of communication needed for performing domain decomposition.

In our method, we adjust the geometries of the domains assigned to individual processes, so that the total calculation time of the force (sum of the short-range and long-range forces) becomes the same for all MPI processes. We achieve good load balance by adjusting the sampling rate of particles in one domain so that it is proportional to the measured calculation time of the short-range and long-range forces. Thus, if the calculation time of a process is larger than the average value, the number of sampled particles of the process becomes relatively larger. After the root process gathers all sampled particles from the others, the new domain decomposition is created so that all domains have the same number of sampled particles. Therefore, the size of the domain for this process automatically becomes somewhat smaller, and the calculation time for the next timestep is expected to become smaller.

2.2. *Optimized Particle-Particle Force Loop*

The calculation of the pairwise force is the highest cost part. We can optimize this part by utilizing Phantom-GRAPE (Nitadori *et al.* 2006; Tanikawa *et al.* 2012a,b) software accelerator, which is originally developed for the x86 architecture with the SSE (Streaming SIMD† Extensions) instruction set. We have extended Phantom-GRAPE with support for the short-range force of TreePM to the HPC-ACE (High Performance Computing Arithmetic Computational Extension) architecture of K computer using SIMD built-in functions provided by the Fujitsu C++ compiler. To get the maximum performance, the force loop was unrolled eight times by hand so that 16 pairwise interactions, forces from 4-particles to 4-particles are evaluated in one iteration.

The LINPACK peak per core of SPARC64 XIIIfx is 16 Gflops [4 FMA (Fused Multiply Add) units running at 2.0 GHz]. However, the theoretical upper limit of our force loop is 12 Gflops because it consists of 17 FMA and 17 non-FMA operations (51×2 floating-point operations in total) for two (one SIMD) interactions. Our force loop reaches 11.65 Gflops on a simple $\mathcal{O}(N^2)$ kernel benchmark, which is 97% of the theoretical limit.

2.3. *Relay Mesh Method*

For the parallel FFT of the PM part, we can use the MPI version of the FFTW 3.3 library (http://www.fftw.org/). The parallel FFTW supports the 1-D slab decomposition only, in which the number of processes that perform FFT is restricted by the number of grid point of the PM part in one dimension. Since the calculation cost of FFT is relatively small in most cases, this 1-D parallel FFT may not decrease the performance significantly. However, in the situation that the number of MPI processes is very large,

† Single Instruction/Multiple Data

https://doi.org/10.1017/S174392131700045X Published online by Cambridge University Press

communication becomes problematic since the number of processes that send the local mesh to an FFT process is proportional to $p^{2/3}$, where p is the number of MPI processes. Thus, the communication time of the conversion of the mesh from 3-D rectangular to 1-D slab can become bottlenecks on modern massively parallel supercomputers such as the full system of K computer.

To overcome this problem, we developed a novel communication algorithm, *Relay Mesh Method*. The basic idea of this method is to split the global all-to-all communication on the conversion of the mesh structures into two local communication. Processes are divided into small groups whose sizes are equal or larger than that of the FFT processes. One of the groups contains the FFT processes, we call this group the root group. For example, consider a simulation with 2-D decomposed 6×6 processes and the number of PM grids in one axis $N_{\rm PM} = 8^3$. In this case, the number of FFT processes is eight since the FFT is parallelized for only one axis. We make four groups that consist of $3 \times 3 = 9$ processes. The eight processes of the root group perform FFT.

The 1-D slab decomposed density mesh is constructed in the following two steps. First, each group compute the contribution of its particles to the mesh, and then the total mesh is constructed by adding up the contributions from all groups. Each group constructs the 1-D distributed density mesh that covers the full simulation box. The decomposition of the density slabs are the same for all groups, however, the density slabs of each group include the only contributions of particles in the group (partial slab density). Then each group communicate their slabs so that the root process contain the complete slabs. In this method, the global communication in the second step (previous page) is replaced by two local communications, one within groups and the other over groups. The first communication is done to construct the 1-D distributed density mesh in the same way as the second step of the original method, but the communication is closed within each group. In this example, the nine processes of each group send the mass density to eight processes within the same group. After the first communication, each group has the 1-D distributed partial density slabs. Then, all groups *relay* the partial slabs to the root group, and the root group reduces them to construct the complete slabs. In this example, four processes in different groups communicate. The more details of this method are described in Ishiyama *et al.* (2012).

Using this method, we can avoid network congestion. Here we show the performance result for 4096^3 FFT on 12288 nodes. Without this algorithm, $\sim$10 and $\sim$3 seconds took for for the conversion of mesh structures and backward potential conversion, respectively. With this method using three groups, these are reduced to $\sim$3 and $\sim$0.3 seconds. Our novel communication algorithm could decrease the communication time by a factor of more than four. On the other hand, the calculation time of FFT itself was $\sim$4 seconds. Thus, FFT became a bottleneck after the optimization of these communication parts. However, we confirmed that the performance is improved for FFT by using a 3-D parallel FFT library and this novel technique is also applicable for the simplification of the conversion.

3. Scalability and Performance

Figure 3 shows the strong scaling of our code, namely CPU time per step as a function of the number of computational nodes. To measure the scalability of our code, we performed simulations with 2048^3, 4096^3, 8192^3, and 10240^3 dark matter particles. We set the number of PM grids in one axis as $N_{\rm PM} = N^{1/3}/2$.

Regardless of the number of particles, the parallel speedup is excellent up to the full

https://doi.org/10.1017/S174392131700045X Published online by Cambridge University Press

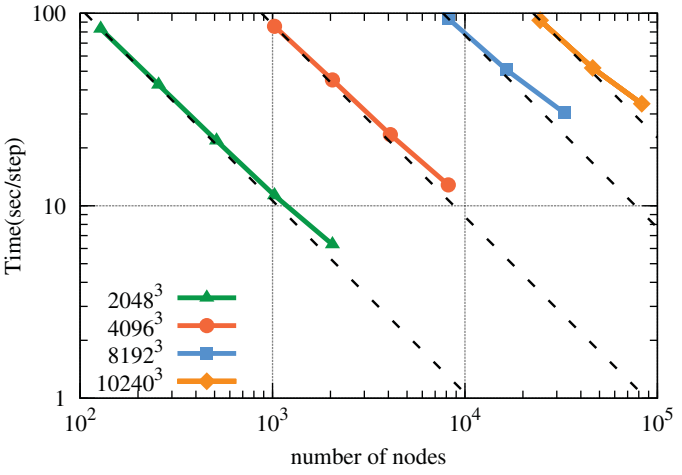

Figure 3. Calculation time per step of our code as a function of the number of nodes on K computer. The results of 2048^3, 4096^3, 8192^3, and 10240^3 dark matter simulations are shown. The dashed lines show the perfect strong scaling.

system of K computer (82,944 nodes). In particular, for the region with less than 10,000 nodes, parallel speedup is almost perfect.

For two trillion particles benchmark simulation, the average performance on the full-system of K computer is about 5.8 Pflops , which corresponds to 55% efficiency. If we focus on the only force calculation cycle, it achieves 71% efficiency, which corresponds to 95% efficiency since the theoretically maximum efficiency is 75%. It is important to keep in mind that the performance is underestimated since we use only the particle-particle interaction part to estimate the performance. Actually, we obtained a few percent higher efficiency by he Fujitsu sampling profiler since it counts all floating-point operations.

4. Conclusion

We present the implementation and performance results of our massively TreePM code on the full system of K computer. The average performance achieved is 5.8 Pflops. The efficiency of the entire calculation reaches 55%. The efficiency of the gravity kernel is 71%. These high efficiency is achieved by a highly optimized gravity kernel for short-range force calculation on the HPC-ACE architecture of K computer and by developing a novel domain decomposition and communication algorithm for the calculation of long-range forces. Our implementation enables us to perform huge cosmological N-body simulations within practical time (e.g., Ishiyama 2014; Ishiyama *et al.* 2015, 2016).

Acknowledgment

The development of our simulation code was carried out on Cray XT4 and XC30 at Center for Computational Astrophysics, CfCA, of National Astronomical Observatory of Japan, and the K computer at the RIKEN Advanced Institute for Computational Science (Proposal numbers hp120286, hp130026, hp140212 and hp150226) This work has been funded by MEXT HPCI STRATEGIC PROGRAM. We thank the support by MEXT/JSPS KAKENHI Grant Number 24740115, 15H01030, and 15K12031.

References

Angulo, R. E., Springel, V., White, S. D. M., Jenkins, A., Baugh, C. M., & Frenk, C. S. 2012, *Mon. Not. R. Astron. Soc.*, 426, 2046

Bagla, J. S. 2002, *Journal of Astrophysics and Astronomy*, 23, 185

Barnes, J. & Hut, P. 1986, *Nature*, 324, 446

Barnes, J. E. 1990, *Journal of Computational Physics*, 87, 161

Blackston, D. & Suel, T. 1997, in Proceedings of the 1997 ACM/IEEE conference on Supercomputing (CDROM), Supercomputing '97 (New York, NY, USA: ACM), 1–20

Bode, P., Ostriker, J. P., & Xu, G. 2000, *Astrophys. J.Supp.*, 128, 561

Boylan-Kolchin, M., Springel, V., White, S. D. M., Jenkins, A., & Lemson, G. 2009, *Mon. Not. R. Astron. Soc.*, 398, 1150

Dubinski, J., Kim, J., Park, C., & Humble, R. 2004, *New Astronomy*, 9, 111

Hockney, R. W. & Eastwood, J. W. 1981, Computer Simulation Using Particles (New York: McGraw-Hill)

Ishiyama, T. 2014, *Astrophys. J.*, 788, 27

Ishiyama, T., Enoki, M., Kobayashi, M. A. R., Makiya, R., Nagashima, M., & Oogi, T. 2015, *Publ. of the Astron. Society of Japan*, 67, 61

Ishiyama, T., Fukushige, T., & Makino, J. 2009, *Publ. of the Astron. Society of Japan*, 61, 1319

Ishiyama, T., Nitadori, K., & Makino, J. 2012, in Proc. Int. Conf. High Performance Computing, Networking, Storage and Analysis, SC'12 (Los Alamitos, CA: IEEE Computer Society Press), 5:, (arXiv:1211.4406)

Ishiyama, T., Sudo, K., Yokoi, S., Hasegawa, K., Tominaga, N., & Susa, H. 2016, *Astrophys. J.*, 826, 9

Klypin, A., Yepes, G., Gottlober, S., Prada, F., & Hess, S. 2014, arXiv: 1411.4001

Klypin, A. A., Trujillo-Gomez, S., & Primack, J. 2011, *Astrophys. J.*, 740, 102

Komatsu, E., Dunkley, J., Nolta, M. R., Bennett, C. L., Gold, B., Hinshaw, G., Jarosik, N., Larson, D., Limon, M., Page, L., Spergel, D. N., Halpern, M., Hill, R. S., Kogut, A., Meyer, S. S., Tucker, G. S., Weiland, J. L., Wollack, E., & Wright, E. L. 2009, *Astrophys. J.Supp.*, 180, 330

Makino, J. 2004, *Publ. of the Astron. Society of Japan*, 56, 521

Nitadori, K., Makino, J., & Hut, P. 2006, *New Astronomy*, 12, 169

Planck Collaboration, Ade, P. A. R., Aghanim, N., Armitage-Caplan, C., Arnaud, M., Ashdown, M., Atrio-Barandela, F., Aumont, J., Baccigalupi, C., Banday, A. J., *et al.* 2014, *Astron. Astrophys.*, 571, A16

Skillman, S. W., Warren, M. S., Turk, M. J., Wechsler, R. H., Holz, D. E., & Sutter, P. M. 2014, arXiv: 1407.2600

Springel, V. 2005, *Mon. Not. R. Astron. Soc.*, 364, 1105

Springel, V., White, S. D. M., Jenkins, A., Frenk, C. S., Yoshida, N., Gao, L., Navarro, J., Thacker, R., Croton, D., Helly, J., Peacock, J. A., Cole, S., Thomas, P., Couchman, H., Evrard, A., Colberg, J., & Pearce, F. 2005, *Nature*, 435, 629

Tanikawa, A., Yoshikawa, K., Nitadori, K., & Okamoto, T. 2012a, ArXiv e-prints

Tanikawa, A., Yoshikawa, K., Okamoto, T., & Nitadori, K. 2012b, *New Astronomy*, 17, 82

Teyssier, R., Pires, S., Prunet, S., Aubert, D., Pichon, C., Amara, A., Benabed, K., Colombi, S., Refregier, A., & Starck, J. 2009, *Astron. Astrophys.*, 497, 335

White, M., Cohn, J. D., & Smit, R. 2010, *Mon. Not. R. Astron. Soc.*, 408, 1818

Xu, G. 1995, *Astrophys. J.Supp.*, 98, 355

Yoshikawa, K. & Fukushige, T. 2005, *Publ. of the Astron. Society of Japan*, 57, 849

https://doi.org/10.1017/S174392131700045X Published online by Cambridge University Press