Huge-scale Molecular Study of Multi-bubble Nuclei

Hiroshi WATANABE*[1] and Nobuyasu ITO[2]
[1] *The Institute for Solid State Physics, The University of Tokyo, Kashiwanoha 5-1-5, Kashiwa, Chiba 277-8581, Japan and*
[2] *Department of Applied Physics, School of Engineering, The University of Tokyo, Hongo, Bunkyo-ku, Tokyo 113-8656, Japan*

Abstract

We have developed a molecular dynamics simulation code which adopts MPI/OpenMP hybrid parallelization on the basis of fully domain decomposition scheme. The developed codes is highly optimized and scalable for massively parallel computers. We have succeeded to simulate a huge system containing 38.4 billion Lennard-Jones particles on 76800 processors, and have achieved 193 teraflops which corresponds to a 17.0% of the theoretical peak performance. Using the developed codes, we obtained a precise phase diagram and confirmed that the critical exponents of the gas-liquid transition are consistent with those of the Ising universality class. We also investigate bubble-nucleation phenomena, and observe the Ostwald-like ripening of bubbles. These results demonstrate that the molecular dynamics simulation method is a promising tool for the era of peta-scale computation.

I. INTRODUCTION

Owing to the recent increase in computational power, computer simulation has been more powerful tool to investigate material science. However, it becomes more difficult to utilize the computational power to the full since the increase in computational power is mainly achieved by increase in numbers of CPU-cores. One has to perform some sort of massively parallel computations on such kinds of computers. As a number of processors increases, time spent for communications may increase. If one wants to utilize a computers consisting of a huge number of processors, one has to adopt a parallelization algorithm which exhibits good scaling for more than ten thousands CPU-cores.

We have developed a huge-scale molecular dynamics (MD) codes with the aim of performing direct simulations of gas-liquid multiphase flow. An investigation of gas-liquid multiphase flow is challenging, since it involves phase transition in microscopic regime and flow in macroscopic regime. A direct simulation of the gas-liquid multiphase flow is an approach to reproduce multi-scale and multi-physics phenomena from atomic scale only by assuming particles interactions. While such simulation requires a huge number of particles, the direct simulation of multi-scale physics at the atomic scale are now possible thanks to the development of the computational power. In this manuscript, we report huge-scale MD simulations involving tens of billons of particles on massively parallel computers. We also describe several issues facing programers in dealing with huge-scale simulations.

The present article is organized as follows. In Sec. II, parallelization scheme and efficiency of our code are described. Simulations of multi-bubble nuclei are described in Sec. III. Finally, Sec. IV is devoted to a summary of this study and a perspective of the huge-scale MD simulations.

*hwatanabe@issp.u-tokyo.ac.jp

II. PARALLELIZATION

A. Pseudo-flat-MPI approach

We have two choices for parallelization, flat-MPI and MPI/OpenMP hybrid models. While the flat-MPI model is usually faster than the hybrid for MD simulations with short-range interactions, memory consumption of the flat-MPI is larger than that of the hybrid and it may cause difficulties in substantial simulations. Therefore, OpenMP/MPI hybrid parallelization is required in order to perform a huge job on millions of CPU-cores. One simple way to implement a hybrid code of MD is to adopt the domain decomposition strategy for internode communication with MPI and loop-level decomposition for intranode communication with OpenMP. While this implementation is simple, one has to perform thread parallelization not only for force calculation, but also for other routines, such as pair-list construction, observations, and so forth. A programmer has to consider many different kinds of thread parallelization in order to achieve better performance. This fact imposes a heavy burden on programers. Therefore, we adopts pseudo-flat-MPI approach [1]. By treating thread IDs of OpenMP as ranks of MPI, a full domain decomposition strategy is adopted both for intra- and internode parallelization [2]. The pseudo-flat-MPI approach is similar to the flat-MPI since both model use shared memory as distributed memory in a node. But memory consumption by the pseudo-flat-MPI is less than that by the flat-MPI since the total number of MPI processes decreases. If the system supports MPI subroutine calls from OpenMP threads, then the implementation of communication of the pseudo-flat-MPI is identical to that of a flat-MPI. However, many platforms dose not support or partially support MPI calls from threads [3]. In such platforms, two-level communication is required, *i.e.*, data required from threads belonging to a different processes should be packed and the receiver process should distribute the data to the destination thread (see Fig. 1). One of the disadvantages of adapting the pseudo-flat-MPI model is this two-level communications. However, the CPU-level tuning of the pseudo-flat-MPI is much easier than that of the hybrid. For the naive loop-level decomposition method, load-balancing of threads and SIMD (single instruction multiple data) optimization should be considered simultaneously. The pseudo-flat-MPI model allows a programer to consider only CPU-level tuning, since parallelization of threads is already achieved in higher level.

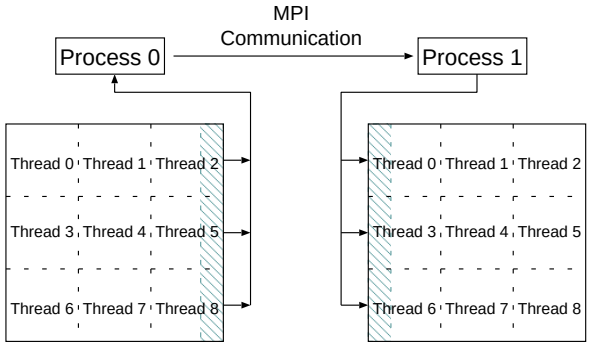

FIG. 1: Schematic illustration of the pseudo-flat-MPI approach. A two-dimensional system is shown for the visibility while the simulations are performed in three-dimensional systems. There are two MPI processes and each process contains nine OpenMP threads. The system is divided into eighteen domains, and each domain is assigned to a thread. Communications between threads of different process are performed via processes.

B. Flat-MPI vs. Hybrid

We perform benchmark simulations of the developed code on FUJITSU PRIMEHPC FX10 at Information Technology Center (ITC) of the University of Tokyo. This machine consists of 4800 SPARC64 IXfx 1.848 GHz processors and each processor contains 16 CPU-cores. We use a truncated Lennard-Jones (LJ) potential with cutoff length 2.5 in LJ units [4]. Keeping a number of particles per node, we in-

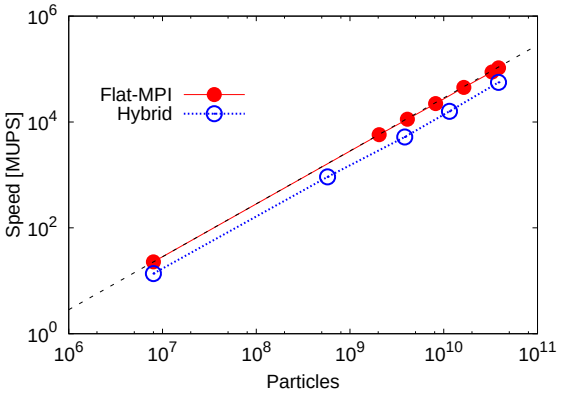

FIG. 2: Parallel efficiency of flat-MPI and hybrid simulations. The results of the flat-MPI and the hybrid are shown as filled and open circles, respectively. Fixing a number of particles per node, we increase a number of nodes from 1 to 4800. The largest run contains 38.4 billion particles and achieved 105229 MUPS.

crease a number of nodes from 1 to 4800. Each node contains 8 million particles, and therefore, the largest run contains 38.4 billion particles. The number density is set to be 0.5. We perform two kinds of simulations, flat-MPI and hybrid. We assign 16 MPI processes on a node for the flat-MPI run and one MPI process with 16 OpenMP threads on a node for the hybrid run. The results is shown in Fig. 2. Computational speeds are measured in the unit millions of updates per second (MUPS). per node, we increase a number of nodes from 1 to 4800. While the parallel efficiency of the flat-MPI is almost perfect, that of the hybrid fluctuates. Computational speed of the flat-MPI is always better than that of the hybrid. The total memory consumption of the hybrid run is less than that for that of flat-MPI, as expected. For the largest run involving 4800 nodes, the hybrid run consumed 10 GB per node out of 32 GB while the flat-MPI run consumed 14 GB per node. Since the computational workload and the memory usage of the user program are identical between two runs, the difference in memory consumption comes from MPI processes.

C. Effect of Hyper Threading

We found that the performance of parallel simulations decreased as the number of processes increased due to the fluctuation of the execution time [5]. While this can be caused from noise from operating system, but details remain unsolved. In order to clarify the cause of the fluctuation, we investigate the effect of hardware multithreading on parallel performance.

We adopt the flat MPI model, that is, we use only MPI for communications. Three types of communication are involved in the simulations, which are (i) exchanging particles that stray from the originally associated process, (ii) exchanging information of particles near boundaries to consider the cross-border interactions, and (iii) synchronization of the validity check of pair lists. The former two are implemented by MPI_Sendrecv function and the last one is implemented by MPI_Allreduce. Note that, only one integer is sent/received by MPI_Allreduce. Therefore, the time spent for the communication is negligible and it plays a role of a barrier.

We perform simulations on SGI Altix ICE 8400EX at the Institute for Solid State Physics (ISSP), the University of Tokyo. It is an integrated blade cluster system consisting of 1920 nodes, 15360 cores and 46TB memory. Each nodes has two Intel Xeon X5570 processors (2.93GHz) which includes 4 cores. Two CPUs in a node is connected by QPI (QuickPath Interconnect) which throughput is 25.6 GB/s. While Intel Xeon X5570 processors support Hyperthreading technology (HT), HT is turned off in the normal operation at ISSP. We turned HT on in monthly large job service and performed benchmarks and compared the results with the benchmark without HT.

We perform the simulation with the identical condition for the largest run in the previous work, that is, the number of MPI processes are 8192 with the same number of physical cores. Therefore, the computational work-

load per core is the same for the both run, with and without HT. In the run with HT, there are two logical cores in one physical core. One MPI process is associated with one logical cores, and therefore, the other logical core is always idle through the simulation (see Fig. 3).

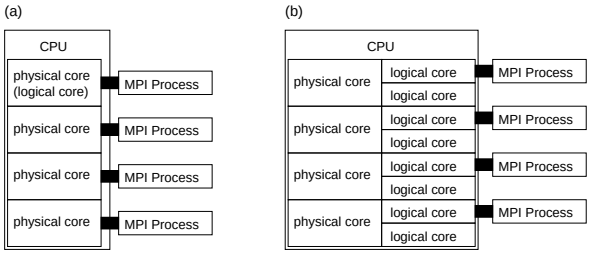

FIG. 3: (a) Process binding without hyperthreading. Each CPU includes 4 physical cores. One MPI process is executed on one physical processor. (b) Process binding with hyperthreading. While each physical core appears as two logical cores, only one MPI process is executed on one physical processor. Therefore, one logical core is always idle on one physical core.

Figure 4 shows the elapsed time vs. a number of processes. Since we perform the weak scaling, the elapsed time would be independent of the number of processes provided if the parallelization efficiency is perfect. Therefore, the increase in time is caused by the parallel overhead. It shows that HT reduces the parallel overhead drastically. We found that the parallel overhead was caused by the fluctuations of the execution time. Due to the global barrier, the slowest process determines the speed of the simulation. It is found that HT reduces the fluctuation of the execution time. Especially, the time of slowest process is drastically decreased by HT. This result implies that the hardware multithreading technology improves not only the efficiency on single physical core, but also for the performance of parallel computing with more than thousand processes. The fact that the HT improves the parallel efficiency suggests that the parallel overhead comes from the system, not from the target application. However, the overhead is too

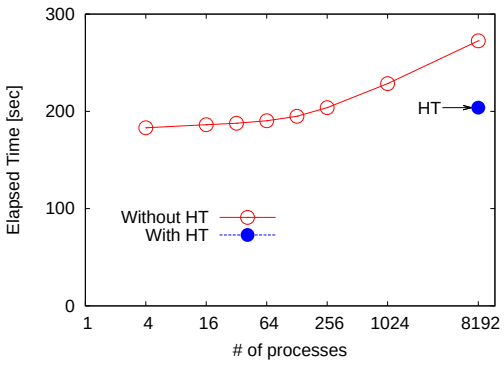

FIG. 4: Parallel Efficiency with and without Hyper-Threading (HT) technology. The open circles denote the results without HT and the solid circles denotes those with HT. The parallel efficiency is improved from 66% to 90% by introducing HT.

large to consider that the overhead is caused by OS jitter, since the timescale of OS jitter is typically at most microseconds [6], while the timescale of the overhead observed here is hundreds milliseconds.

III. MULTI-BUBBLE NUCLEI

A. Phase diagram

In order to perform simulations of multi-bubble nuclei, the precise gas-liquid phase diagram is required. Therefore, we determine the gas-liquid coexisting curve using equilibrium MD simulations [7]. A method to obtain the gas-liquid phase diagram is shown in Fig. 5. First, the liquid-gas coexisting state is realized by isothermal MD simulations with Nosé-Hoover method [8]. The simulation box is a rectangular parallelepiped with sizes $L_x \times L_y \times L_z$. The periodic boundary condition is taken for all directions. We set the ratio of $L_x : L_y : L_z = 1 : 1 : 2$. The studied system sizes are $L_x = 32, 64, 96$, and 128. We place particles densely in the left side of the simulation box so that the liquid phase exists in the left side and the gas phase exists in the right side of the system (Fig. 5 (a)). Suppose the gas-liquid interface is normal to z-axis. Then

the position dependence of the density $\rho(z)$ is of the form,

$$\rho(z) = \frac{(\rho_{\rm l} - \rho_{\rm g})}{2} \left(\tanh\left[(z_c - z)/\lambda\right] + 1\right) + \rho_{\rm g}, \tag{1}$$

where $\rho_{\rm l}$, $\rho_{\rm g}$, and λ are the density of liquid, the density of gas, and the thickness of the interface. The position of the gas-liquid interface is denoted by z_c. Fitting Eq. (1) to the density profile around at the gas-liquid interface, we obtain the coexisting densities of gas and liquid at the chosen temperature. Fig. 5 (b) shows the density profile and the fitting results. Repeating the above procedure at different temperatures, we obtain the gas-liquid phase diagram (Fig. 5 (c)). We perform large scale simulations up to 1583504 particles, and obtain the gas-liquid coexisting densities with high accuracy. Using the obtained densities, we estimate the critical exponent of the order parameter and correlation length to be $\beta = 0.3285(7)$ and $\nu = 0.63(4)$, respectively. These values are consistent with those of the Ising universality class [9–12].

B. Multi-bubble Nuclei

Suppose a particle system in pure liquid phase. If the system is suddenly expanded so that the pressure of the system becomes less than the saturation vapor pressure at the temperature, then the system becomes unstable and bubbles will appear. This phenomenon is called cavitation. When the new state after the expansion is placed between the binodal and the spinodal line in the phase diagram, then the system is in meta-stable. After some waiting time, a bubble will appear [13]. If the expansion is large enough, then the liquid phase becomes unstable and many bubbles will appear. This corresponds to the spinodal decomposition. In order to investigate the multi bubble nuclei, we first equibrate the system in the pure liquid phase, and then expand it strong enough so that the multi bubble nuclei occur.

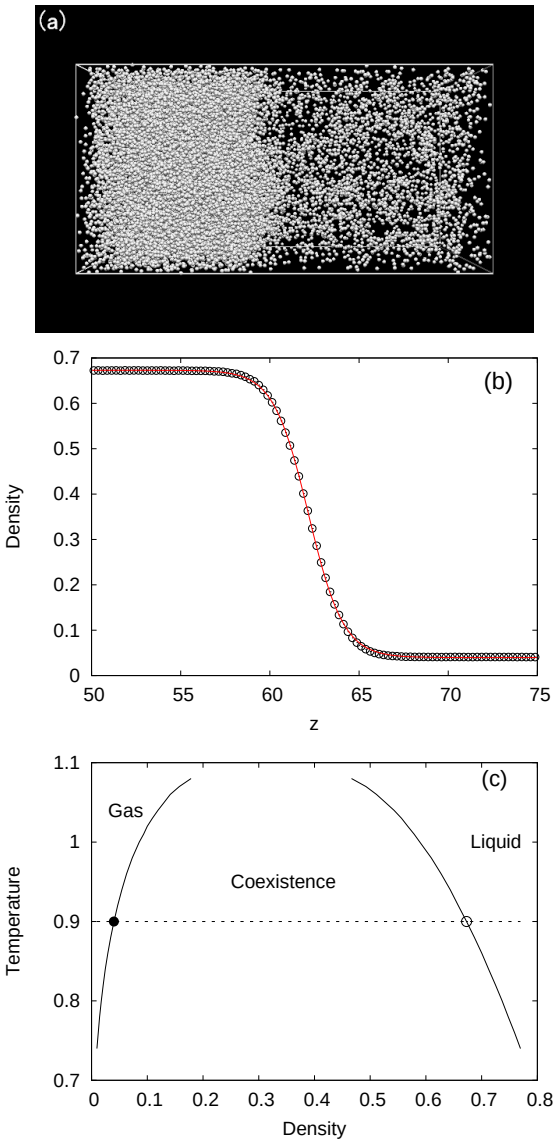

FIG. 5: (a) A snapshot of gas-liquid coexisting phase. Temperature is 0.9. A small system with $L_z = 32$ is shown for the visibility. (b) Density profile at and near the gas-liquid interface. The open circles denote the simulation results and the solid line denotes the fitting result with Eq. (1). (c) Gas-liquid phase diagram of the LJ system with cutoff length 3.0. The coexisting densities of gas and liquid at $T = 0.9$ obtained by the fitting are shown as filled and open circles.

The initial temperature is set to be $T = 0.9$ and the density is set to be $\rho = 0.7$. This corresponds to the pure-liquid phase. After the equilibration, we expand the system adiabatically so that the density becomes $\rho = 0.65$.

After the expansion, we turn off the thermostat and perform NVE simulations. We perform two-sizes of simulations. A smaller run with 22937600 particles is performed at ISSP with the flat-MPI parallelization scheme, and a larger run with 1449776020 particles is performed at ITC with the hybrid scheme. The time evolution of pressure is shown in Fig. 6. We expand the system at $t = 50$ and the pressure becomes negative. Due to the expansion, the temperature of the system decreases from $T = 0.9$ to 0.88. After the expansion, many bubbles appear and start growing. Then the pressure of the system increases due to the entropy production of the bubble growth.

In order to identify bubbles, we divide the system into small subcells with a length of 3.0 and determine the local density for each subcell [13]. We chose the threshold density to be 0.2 and if the local density of a subcell is less than the threshold, then the subcell is considered to be in gas phase, and vice versa. A bubble is identified with the criterion of site percolation, *i.e.*, the adjacent subcells in gas phase are considered to be in the same bubble. The snapshots of bubbles are shown in Fig. 7. While there are many small bubbles just after the expansion, the number of bubbles decreases. Larger bubbles become larger while smaller bubbles become smaller. This phenomenon is called the Ostwald-like ripening.

Size distributions of bubbles at $t = 100$ are shown in Fig. 8. While the data of the smaller and the larger runs are qualitatively same, the statistical errors of the smaller one is much larger than those of the larger one. The bubble-size distribution obtained from the larger run has enough accuracy and it enables us to undertake further analysis of interactions between bubbles during the Ostwald-like ripening. These results shows that at least one billion particles are required to study the time evolution of the bubble-size distribution directly.

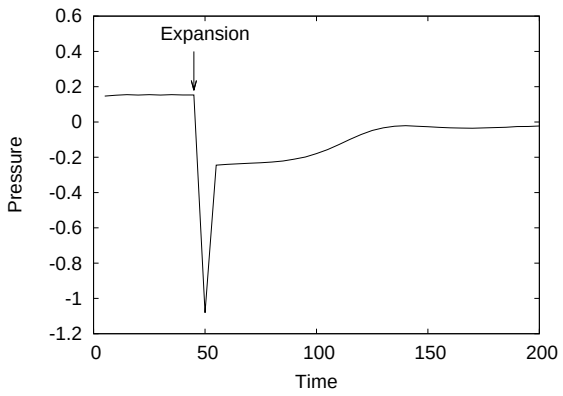

FIG. 6: Time evolution of the pressure. After the expansion at $t = 50$, the pressure becomes negative. Then the pressure increases due to growth of bubbles.

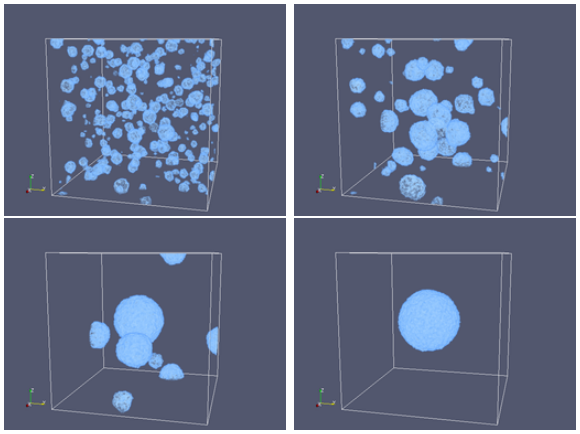

FIG. 7: Snapshots of bubbles. Just after the expansion, there are many bubbles in the system. Then larger bubbles become larger while the smaller bubble becomes smaller due to the interactions between bubbles (Ostwald-ripening). Finally, there is only one bubble in the system.

IV. SUMMARY AND PERSPECTIVE

We have developed a classical MD simulation program with the pseudo-flat-MPI parallelization which works efficiently on the petaflops machine consisting 76800 CPU-cores. We have observed the Ostwald-like ripening of bubbles after multi-bubble nuclei. The Ostwald-like ripening is a multi-scale and multi-physics phenomena since it results from

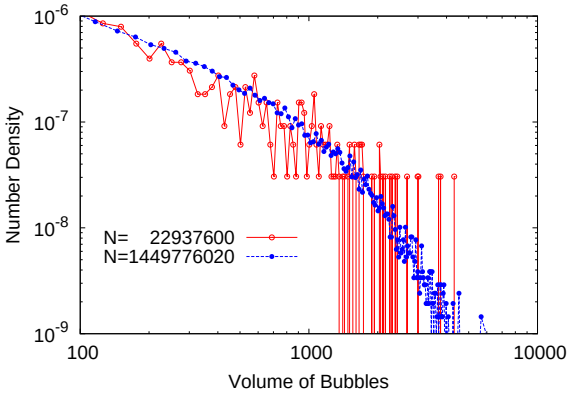

FIG. 8: Bubble-size distribution. The snapshots at $t = 100$ are shown. The decimal logarithms are taken for both axes.

interactions between bubbles, and a bubble results from interactions between atoms. Therefore, we have succeeded in simulating multi-scale and multi-physics phenomena directly from the atomic-scale. As number of particles increases, more complex phenomena can be reproduced. The relation between a number of particles and observable phenomena is summarized in Table I. A thousand particles are enough to observe phenomena in linear response regime, such as steady thermal conduction, Couette flow, etc. If you have several tens of thousands of particles, you can simulate dynamics of a bubble induced by heat or laser pulse [14]. If you want to reproduce a bubble nucleation in metastable liquid, several millions of particles are required [13]. As presented in this paper, tens of millions of particles are necessary to reproduce the Ostwald-like ripening and billions of particles are required if one want to observe the time evolution of bubble-size distribution with acceptable accuracy. Bubble flow will be beyond trillion-particle simulations. As the computational power increases, so does the possibility of the MD simulation. The classical MD simulation with short-range interaction is a promising application which can utilize a peta- or larger scale computers to the full.

Today, one has to treat many different kinds

Phenomena	Required Particles
Linear response regime	10^3
Laser-induced bubble	10^4
Single bubble nucleation	10^6
Multi-bubble nuclei	10^7
Bubble-size distribution	10^9
Bubble flow	10^{12}

TABLE I: Relation between a number of particles and observable phenomena.

of parallel paradigms simultaneously such as SIMD in a CPU-core, thread-parallelization in a node, process-parallelization between nodes, and so forth. Parallelization is not only a thing to be considered. There are many other things that need to be taken care of such as multi-level CPU-cache, different CPU-architecture, and so forth. Even now, the development costs is burdensome. The development costs may increase, but will surely not decrease. If this trend continues, then a person who writes programs and a person who writes scientific papers will be specialized and separated. While it is true that some kinds of specializations are unavoidable, we would like to place a high value on writing programs and scientific papers by a same person. We will happy if the present paper and our codes [15] might help a people who writes a code for huge-scale simulations from the scratch and writes a scientific paper simultaneously.

Acknowledgements

The part of the present report is based on the recent works done with C.-K. Hu and M. Suzuki. The authors would like to thank S. Todo and T. Komatsu for fruitful discussions. HW is grateful to K. Nitadori for giving valuable comments. This work was partially supported by Grants-in-Aid for Scientific Research (Contract No. 23740287) and by KAUST GRP (KUK-I1-005-04). The computational resource of Fujitsu FX10 was awarded by "Large-scale HPC Challenge" Project, In-

formation Technology Center of the University of Tokyo. The computations were also carried out using the facilities of the Supercomputer Center, Institute for Solid State Physics, University of Tokyo, and the Research Institute for Information Technology, Kyushu University.

[1] H. Watanabe, M. Suzuki, and N. Ito, arXiv:1210.3450.
[2] M. J. Berger, M. J. Aftosmis, D. D. Marshall, S. M. Murman, J. Parallel Distrib. Comput. **65**, (2005) 414.
[3] The specifications of MPI 2.2 is available from: ⟨http://www.mpi-forum.org/⟩.
[4] S. D. Stoddard and J. Ford, Phys. Rev. A **8**, (1973) 1504.
[5] H. Watanabe, M. Suzuki, and N. Ito, Prog. Theor. Phys. **126**, (2011) 203 .
[6] P. Beckman, K. Iskra, K. Yoshii, and S. Coghlan, ACM SIGOPS Operating Systems Review **40**, (2006) 29.
[7] H. Watanabe, N. Ito, and C.-K. Hu J. Chem. Phys. **136**, (2012) 204102.
[8] W. G. Hoover, Phys. Rev. A **31**, (1985) 1695.
[9] H. E. Stanley, *Introduction to Phase Transitions and Critical Phenomena* (Oxford Univ. Press, New York 1971).
[10] A. L. Talapov and H. W. J. Blöte, J. Phys. A: Math. Gen. **29**, (1996) 5727.
[11] N. Ito, K. Hukushima, K. Ogawa, and Y. Ozeki, J. Phys. Soc. Jpn. **69**, (2000) 1931.
[12] M. Hasenbusch, Phys. Rev. B **82**, (2010), 174433.
[13] H. Watanabe, M. Suzuki, and N. Ito, Phys. Rev. E **82**, (2010) 051604.
[14] H. Okumura and N. Ito, Phys. Rev. E **67**, 045301(R) (2003).
[15] ⟨http://mdacp.sourceforge.net/⟩.