3.6 Software Advancement Projects, GPGPU Implementation, and Workshop Support

Development of an integrated dynamical mean-field theory package for correlated electrons

Hiroshi Shinaoka[1], Junya Otsuki[2], Kazuyoshi Yoshimi[3], Mitsuaki Kawamura[3], Takeo Kato[3]
[1]*Department of Physics, Saitama University, Saitama 338-8570*
[2]*Department of Physics, Tohoku University, Sendai 980-8578*
[3]*Institute for Solid State Physics, University of Tokyo, Kashiwa, 277-8581*

In condensed matter physics, dynamical mean-field theory (DMFT) [1] is a widely used tool for the study of strongly correlated electron systems. In a DMFT calculation, a correlated lattice model is mapped to an impurity problem whose bath degrees of freedom are self-consistently determined. DMFT can be combined with density functional theory based *ab-initio* calculations as the DFT+DMFT method, to describe strongly correlated materials such as transition metal oxides [2]. The DFT+DMFT method is useful particularly for investigating one-particle excitation of the systems. The DFT+DMFT allows us to compute one-particle spectral functions, which can be compared directly with angle-resolved photoemission spectroscopy (ARPES). Although there are several open-source computational libraries for DMFT calculations, the use of these libraries requires some expertise. This prevents wider use of the DFT+DMFT method in studies of condensed matter physics.

To make this method available to non-experts(including students) in the community in condensed matter physics, we have developed an open-source software `DCore` ver.1 [3] in Project for advancement of software usability in materials science [3] at the fiscal year of 2017. `DCore` is an abbreviation of "integrated DMFT software for CORrelated Electrons". `DCore` is built on the top of elaborate softwares `TRIQS` [5] and `ALPSCore` libraries [6] and related softwares. `DCore` performs calculations based on DMFT with the help of these libraries. As an impurity solver, one can select continuous-time quantum Monte Carlo method or the Hubbard-I approximation. Because `DCore` provides a well-organized text-file-based interface, users can perform the DFT+DMFT calculation with less effort. In a typical DFT+DMFT calculation, the non-interacting Hamiltonian $\mathcal{H}(k)$ is extracted from the results of DFT calculations by projecting the band structure to maximally localized Wannier functions. In `DCore`, we can import $\mathcal{H}(k)$ from outputs of the DFT codes which support `Wannier90` such as `VASP`, `Wien2k`, `Quantum ESPRESSO`, and `OpenMX`.

`DCore` consists of multiple programs, each of which performs a different step of DMFT calculations. To be more specific, `DCore` consists of three layers: interface layer, DMFT loop, and post-processing. Those are performed by the executables `dcore_pre`, `dcore`, `dcore_post`, respectively. Input parameters are provided by a single text file, which is read by all the three programs.

For the interface layer, there are two types of interfaces: standard interface for tight-binding models and `Wannier90` interface for materials. For the standard interface, one can choose one of predefined tight-binding models. On the other hand, for the `Wannier90` interface, one is able to import a tight-binding model constructed by DFT calculations. The data describing the system generated by `dcore_pre` is

```
[model]
seedname = square
lattice = square
norb = 1
nelec = 1.0
t = -1.0
kanamori = [(2.0, 0.0, 0.0)]

[system]
beta = 40.0
nk = 8
n_iw = 1000
prec_mu = 0.001

[impurity_solver]
name = TRIQS/hubbard-I

[control]
max_step = 7

[tool]
broadening = 0.4
nnode = 4
knode = [(G,0.0,0.0,0.0),(X,0.5,0.0,0.0),(M,0.5,0.5,0.0),(G,0.0,0.0,0.0)]
nk_line = 100
omega_max =6.0
omega_min =-5.0
Nomega = 400
```

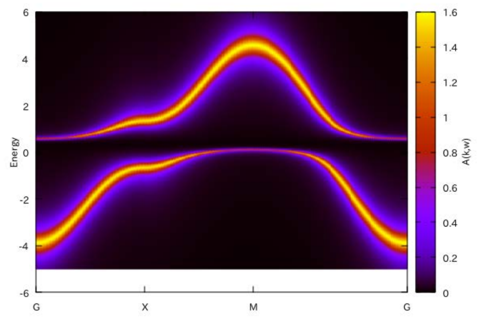

Figure 1: The upper panel shows an example of input file for single-orbital Hubbard model on a square lattice. The lower panel shows the computed momentum-resolved spectrum $A(k, \omega)$.

stored in a file in the HDF5 format, which is read in the later processes. Self-consistent calculations are performed by `dcore` and the results are stored in a separated HDF5 file. One can analyze the result and plot the data by using `dcore_post`.

We show an example for a single-orbital Hubbard model on a square lattice in Fig. 1 using the standard interface. In the input file, one can choose the lattice model, the type of local interactions, and their strengths. Here, the impurity solver is the Hubbard-I approximation using an implementation in `TRIQS`. The computed results are processed by `dcore_post` and are converted into human-readable formats. One can plot the data by using standard tools such as gnuplot (see Fig. 1.) With the `Wannier90` interface, one can perform DFT+DMFT calculations by using a single similar text input file for `DCore`. We refer the interested reader to the website [3] for more examples for real materials.

Finally, we introduce some of available features in `DCore` ver 1 and a future development plan of `DCore`. The feature of `DCore` is to treat many kinds of interactions such as multi-orbital models with non-density-density interactions and spin-orbit coupling. Thus, we can perform collinear magnetic calculations. A future version will support the computation of (free) energy, two-particle quantities such as local magnetic susceptibilities, and the calculations of non-collinear magnetic structures. The software will be preinstalled on the supercomputer (Sekirei) at ISSP in 2018. We hope that DCore promotes wide use of the DFT-DMFT calculations, which is one of excellent methods for understanding strongly-correlated electron systems.

References

[1] A. Georges, G. Kotliar, W. Krauth, M.J. Rozenberg, Rev. Mod. Phys. **68**, 13 (1996).

[2] G. Kotliar, S. Savrasov, K. Haule, V. Oudovenko, O. Parcollet, C. Marianetti, Rev. Mod. Phys. **78**, 865 (2006).

[3] `https://github.com/issp-center-dev/DCore`

[4] `http://www.issp.u-tokyo.ac.jp/supercom/softwaredev`

[5] O. Parcollet *et al.*, Comput. Phys. Comm. **196**, 398 (2015).

[6] A. Gaenko *et al.*, Comput. Phys. Comm. **213**, 235 (2017).

GPGPU implementation of Fluid Particle Dynamics (FPD) method

Michio Tateno, Kyohei Takae and Hajime Tanaka
Department of Fundamental Engineering, Institute of Industrial Science, University of Tokyo
4-6-1 Komaba, Meguro-ku, Tokyo 153-8505

Structural ordering in colloidal suspensions is significantly influenced by many-body hydrodynamic interactions among colloids, because the liquid flow field is intrinsically coupled with colloid motion. When numerically studying such a process, we inevitably encounter a complicated moving boundary problem, because the non-slip solid-fluid boundary condition has to be satisfied at the surface of the colloidal particles. However, solving such a boundary problem is numerically costly, because we need to generate a complex adaptive mesh depending on the positions of colloids at every time step [1, 2]. This difficulty can be overcome by treating a solid colloidal particle as an undeformable fluid particle with high viscosity. We call this method Fluid Particle Dynamics (FPD) method [3, 4].

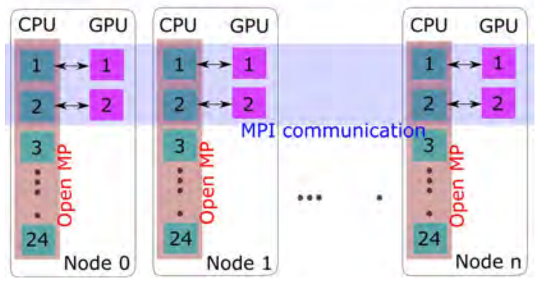

Figure 1: Schematic figure for the way of utilizing computational resources in a large-scale simulation (case(a)). All the computations except for Fast Fourier Transformation (FFT) are performed with a hybrid GPGPU+MPI program. For FFT we transferred the data from devices to hosts and executed it on hosts with a hybrid MPI+OpenMP program.

When examining self-assembly kinetics of colloidal suspensions numerically, we need to choose the length/time scale of our simulations large/long enough for problems we study. As an example, we consider phase separation of a dilute colloidal suspension, which is often accompanied by network formation of colloidal particles. To study this problem numerically, we need to use a large simulation box whose size is far beyond the typical length of aggregates such as the typical pore size of the resulting colloidal gel. Considering the long-range nature of hydrodynamic interactions, there is also a possibility that a gelation process may severely be influenced by finite size effects. Therefore, we need a large-size simulation. Furthermore, when we are interested in kinetics of structure formations in dense suspensions, The time scale of simulation becomes important. This is because the diffusive motions of colloids should largely slow down due to steric hindrance by the surrounding colloids in such a dense system. Consequently, the time scale of the structural ordering such as crystallization is usually much slower than that expected from the Brownian time of a free colloid. Thus, in order to numerically follow such a slow dynamics, a long time simulation is required.

For the above reasons, we need to overcome the problems of the numerical costs associated with simulation size and time. To this end, we perform GPGPU implementation for (a) colloidal gelation and (b) crystallization, utilizing a service provided by ISSP. In this report we explain the outcome of this project.

In ISSP's Supercomputer Center we used queue of F18acc (class B) where 24 CPUs and 2 GPUs (TeslaK40c) are implemented per Node. Each GPU has approximately 12 GB of memory. For case (b), the system size was set as $(L/\sigma)^3 = 17.3^3$ where L and σ are the side length of the simulation box and the diameter of colloids (the corresponding size of the box in the lattice unit is 128^3). In the above setting, we can perform simulations while storing all the data on device since the amount of data (0.45 GB) is less than that of GPU's memory. For case (a) this system size is not large enough to capture a hierarchical structure of a colloidal gel [5] as explained in the previous paragraph. Therefore we performed larger system size simulations, $(L/\sigma)^3 = 69.2^3$. In this case, the required memory is $\sim$ 30GB, which is beyond the capacity of single GPU. We transfer the large amount of data between hosts and devices, developing a simulation code by hybrid GPGPU+MPI parallelization. By dividing all the data with MPI parallelization and distributing them onto multiple GPUs, we can perform most of computations while keeping the data stored on GPUs. In the FPD method we also use Fast Fourier Transformation(FFT) where only part of FFT is performed on host with hybrid MPI+OpenMP parallelization(see Fig.1), because in CUDA and OpenACC no library that can deal with FFT beyond nodes is provided at this moment.

With the above setting, we examine the performance of simulations by the GPGPU adopted codes. For case (b) where we utilize single GPU, the speed measured is 3.7 times faster than that of our previous non-GPGPU simulations, which are performed by MPI parallelization with 16 threads. By this speeding up, we succeeded in simulating the whole process of crystallization in a colloidal suspension from a metastable liquid state to a crystal state. For case (a), we use 32 GPUs (16 Nodes) and realize 10.4 times speeding up compared with our previous non-GPGPU simulations with 16-threads MPI parallelization. With this codes, we successfully perform simulations with 8 times larger system size (in volume) than the simulations without GPGPU implementation and observe power-law growth behavior over one order of magnitude in time. This power-law behavior, which has never been confirmed with the non-GPGPU simulations because of its slow computational speed, implies a characteristic coarsening process of colloidal gelation [5].

In summary, we have made GPGPU implementation of the FPD method to study colloidal gelation and crystallization. We have succeeded in 10.4 and 3.7 time speeding up compared with the performance of our previous simulations without GPU parallelization. We thank ISSP Supercomputer Center for providing computational resources and a service for GPGPU implementation.

References

[1] H. H. Hu, N. A. Patankar, and M. Y. Zhu: J. Comput. Phys., **169**, (2001) 427.

[2] A. S. Khair and J. F. Brady: Proc. R. Soc. A, **463**, (2007) 223.

[3] H. Tanaka and T. Araki: Phys. Rev. Lett., **85**, (2000) 1338.

[4] A. Furukawa, M. Tateno and H. Tanaka: Soft Matter, doi:10.1039/C8SM00189H.

[5] See an activity report "Study of finite-size effects on colloidal gelation originating from momentum conservation".

Implementation of GPGPU computing in full diagonalization for $\mathcal{H}\Phi$

Takahiro Misawa and Kazuyoshi Yoshimi
Institute for Solid State Physics, University of Tokyo
Kashiwa-no-ha, Kashiwa, Chiba 277-8581

From the 2015 fiscal year, we have been developed the open-source software for exact diagonalization $\mathcal{H}\Phi$ [1, 2] under the support of "Project for advancement of software usability in materials science" by Institute for Solid State Physics (ISSP), University of Tokyo. At the initial stage of the development ($\mathcal{H}\Phi$ ver. 1.0 released at the 2015 fiscal year), we implemented the full diagonalization using the LAPACK routine [3], the exact diagonalization using the Lanczos method [4], and the finite-temperature calculations using the thermal pure quantum states [5] for the general Hamiltonians in the quantum lattice models. Then, in the 2016 fiscal year, we implemented the functions for calculating the dynamical correlation functions using the Lanczos method [4] and the shifted Krylov method [6]. We also implemented the LOBCG method [7] that enables us to obtain many low-energy excited states in one calculations. In the 2017 fiscal year, the real-time evolution is implemented and the developers' manual has been written.

Several useful methods without full diagonalization such as the Lanczos method and the thermal pure quantum states have been implemented in $\mathcal{H}\Phi$ so far. Although the full diagonalization is widely used in the condensed matter physics and is also useful for examining the accuracy of other methods, it is still done by the LAPACK routine for single CPU processor in $\mathcal{H}\Phi$. To get rid of this weak point in $\mathcal{H}\Phi$, under the support of supercomputing center at ISSP, we have implemented the GPGPU

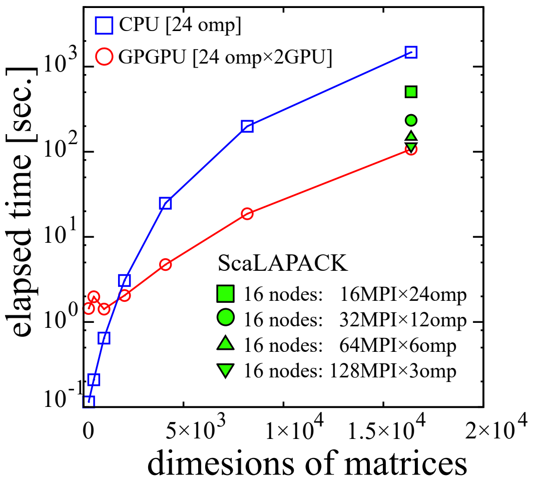

Figure 1: Comparison of elapsed time of full diagonalization for the one-dimensional Heisenberg chain by the LAPACK routine **zheev** with 24 openmp threads and the MAGMA routine **magma_zheevd** with 2 GPGPUs. The calculations are done using single node in F18acc at the supercomputer system B (sekirei). We perform the calculation up to $L = 14$ (matrix dimension is $2^{14} = 16384$) from $L = 8$ (matrix dimension is $2^8 = 256$). Full diagonalization by the GPGPU becomes faster for $L \gtrsim 2500$. We also show the results by the ScaLAPACK performed by 16 nodes in i18cpu for $L = 14$. In the ScaLAPACK, by increasing the number of MPI process, the elapsed time becomes shorter. The full diagonalization by the GPGPU by 1 node is, however, still faster than the ScaLAPACK by 16 nodes.

(General Purpose Graphics Processing Unit) computing for the full diagonalization in this project. We also implemented the full diagonalization by the ScaLAPACK [8] for multi processors in another project and compare the results with the GPGPU. In this activity report, we explain how the GPGPU computing accelerates the full diagonalization in $\mathcal{H}\Phi$.

We replace the **zheev** function in LAPACK with the **magma_zheevd** in the MAGMA library for GPGPU computing [9]. We note that the MAGMA library only supports the diagonalization in single process and multi GPGPUs. Thus, we perform the benchmark in single process. In Fig.1, we show elapsed time of the full diagonalization for the one-dimensional Heisenberg chain as a function of the dimension of the matrix. In this calculation, we do not specify z component of the spin (S_z), the dimension of the matrix is given by 2^L, where L is the system size. We find that the GPGPU computing becomes faster than the LAPACK library around $L \sim 2500$. For larger system sizes ($L \geq 5000$), the full diagonalization by GPGPU is about 10 times faster than the LAPACK routine. In addition, surprisingly, we find that the GPGPU computing with 1 node is still faster than the ScaLAPACK with 16 nodes. This result indicates that the GPGPU computing is efficient in performing the full diagonalization for intermediate size of matrices (size of matrices is about 10^4), which can be treated in single node.

For the standard models in the condensed matter physics such as the Heisenberg or the Hubbard model, users can easily perform the full diagonalization using the standard mode in $\mathcal{H}\Phi$ [1, 2] by preparing only one input file as follows:

```
L       = 12
model   = "SpinGC"
lattice = "chain"
method  = "FullDiagh"
J       = 1.0
NGPU    = 2
```

By using the expert mode, users also treat the general Hamiltonian with the arbitrary one-body potentials and the arbitrary two-body interactions. Furthermore, in $\mathcal{H}\Phi$, it is possible to input the arbitrary Hermite Hamiltonians in the Matrix Market format [10] and perform the full diagonalization. Thus, the implemented efficient full diagonalization method by the GPGPU computing in $\mathcal{H}\Phi$ is useful for a wide range of the scientists in the fields of condensed matter physics and the mathematical science, who want to diagonalize and analyze the large-scale matrices.

Acknowledgment: This work was supported by support service of program portability to General Purpose Graphics Processing Unit.

References

[1] M. Kawamura, K. Yoshimi, T. Misawa, Y. Yamaji, S. Todo, N. Kawashima, Comp. Phys. Commun. **217**, 180 (2017).

[2] http://issp-center-dev.github.io/HPhi /index.html.

[3] E. Anderson, Z. Bai, C. Bischof, L. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, D. Sorensen, LAPACK Users' Guide, 3rd Edition, Society for Industrial and Applied Mathematics, 1999.

[4] E. Dagotto, Rev. Mod. Phys. **66**, 763 (1994).

[5] S. Sugiura, A. Shimizu, Phys. Rev. Lett. **108**, 240401 (2012).

[6] A. Frommer, Computing **70**, 87 (2003).

[7] A. V. Knyazev, SIAM journal on scientific computing **23**, 517 (2001).

[8] L. S. Blackford, J. Choi, A. Cleary, E. D'Azevedo, J. Demmel, I. Dhillon, J. Dongarra, S. Hammarling, G. Henry,

A. Petitet, K. Stanley, D. Walker, R. C. Whaley, ScaLAPACK Users' Guide, Society for Industrial and Applied Mathematics, Philadelphia, PA, 1997.

[9] http://icl.cs.utk.edu/magma/software.

[10] https://math.nist.gov/MatrixMarket/.

Supercomputer course of Computational Materials Design (CMD®) workshop

Masaaki GESHI[1], Yoshitada MORIKAWA[2], Tomoya ONO[3]

[1]*Institute for NanoScience Design,*
Osaka University, Machikaneyama, Toyonaka, Osaka 560-8531

[2]*Department of Precision Science and Technology,*
Osaka University, Yamada-oka, Suita, Osaka 565-0871

[3]*Center for Computational Science,*
University of Tsukuba, Tenno-dai, Tsukuba, Ibaraki 305-8577

The 31th Computational Materials Design (CMD®) workshop (CMD31) has been held from September 11 to September 15 and the 32th CMD® workshop (CMD32) has been done from February 26 to March 2 at Graduate School of Engineering Science, Osaka University. In this workshop we have the supercomputer course to train up human resources to advance researches by using system B supercomputer of ISSP, the University of Tokyo.

In CMD31 six participants took the supercomputer course and got a tutorial on STATE-Senri developed by Y. Morikawa. After explaining how to use the supercomputer of ISSP and explaining how to use STATE-Senri, calculation models on each research subject of the participants were built and their calculations were carried out. Concrete themes were adsorption states, vibration modes and quantum effect of CH_4 molecule on the Pt surface, formic acid adsorption structure on Cu surface and its vibration modes and decomposition reaction process, and structure and reactivity of Pt atoms supported on graphene and so on. The participants performed the calculations and examined the results.

In CMD32 one participant took the supercomputer course and got a tutorial on RSPACE developed by T. Ono. After describing the calculation method of electronic states and electron conduction property using RSPACE, exercises published in the manual were carried out. Then, electronic state calculations were carried out on a plurality of molecular systems, and the electronic density distribution was visualized. Finally, the atomic structure optimization of the system in which molecules are sandwiched between metal electrodes was carried out, and the calculation of electron conduction properties of molecules was analyzed.