Large-Scale MO Calculation with GPU-accelerated FMO Program

Hiroaki Umeda*, Mitsuo Shoji, Taisuke Boku, and Yasuteru Shigeta
Center for Computational Sciences
University of Tsukuba
Tsukuba, Ibaraki, Japan
*umeda@ccs.tsukuba.ac.jp

Toshihiro Hanawa
Information Technology Center
The University of Tokyo
Kashiwa, Chiba, Japan

***Abstract*—GPU-accelerated FMO calculation program has been developed with CUDA and perform large-scale FMO calculations. The performance benchmark shows that GPU-accelerated FMO calculation is 3.8 times faster than the CPU-only FMO calculation on HA-PACS base cluster system. With our GPU-accelerated program, FMO calculation of 23,460 atomic protein was performed in 2 hours with 256 GPUs.**

Keywords—Large-scale Molecular orbital calculation; quantum chemistry; FMO method; GPGPU

I. INTRODUCTION

Ab initio molecular orbital (MO) calculation is a standard tool to obtain molecular properties and to analyze reaction mechanisms in quantum chemistry, and it is wanted to perform MO calculation much faster for large-scale molecule, such as proteins. Fragment molecular orbital (FMO) method [1] is one of such MO calculation technique, in which MO calculation for a molecule is treated as a set of MO calculations for smaller fragments and their inter-fragment interactions under environmental electrostatic potential (ESP). Besides this computational technique, it is also required to execute MO calculation on modern computer architecture, such as accelerator computer, in order to realize large-scale MO calculation. In this presentation, we describe our implementation of GPU-enable FMO program, and evaluate performance of GPU-accelerated FMO calculation on GPU cluster, including the first large-scale GPU-accelerated FMO calculation for 23,460 and 46,641 atomic molecular systems.

II. IMPLEMENTATION

A. Fock Matrix Construction

Fock matrix construction in the FMO calculation is the same as in Hartree-Fock (HF) calculation, and we can find several works for GPU-accelerated two-electron integral calculation [2-6], which is the source of $O(N^4)$ formal computational cost of Fock matrix construction. However, in the case of massive parallel threads with limited memory resources, such as GPU, another point of view becomes considerable, that is, accumulating matrix element to a shared Fock matrix. Because it is difficult to allocate Fock matrix as thread private in GPU local memory, massive slow exclusive addition operation is required to accumulate matrix elements into a shared matrix. Most of previous GPU-study discard advantage of the symmetry property of two-electron integral, which can reduce computational costs, in order to avoid costly exclusive matrix accumulation. Our proposed algorithm [7] is one of the answer to utilize the advantage of integral symmetry.

Proposed algorithm is designed on the basis of our parallel large Fock matrix construction algorithm on distributed PC cluster for FMO-MO method [8]. In our algorithm, each thread has only Gi[] and Gj[] column arrays, and shares resulting Fock matrix G[][] among a thread block. By taking reduction for Gi[] and Gj[] after inner loop, we can accumulate it into G[][] without exclusive control. Noted that GPU-enabled kernel code is activated only for selective integral types in order to overlap CPU and GPU calculations. Applying further CUDA optimizations, such as index-sorting, Schwarz-screening before main-loop, dynamic load-balancing, our GPU-accelerated HF calculation shows three times faster than that with CPU-only on-the-fly integral calculation.

B. Electrostatic potential Calculation

In the FMO method, ESP is calculated as inter-fragment Coulomb interaction (IFC) calculations, and it will be approximated to simplified form depending on the inter-fragment distance. The most time-consuming IFC calculation is four-center IFC (4C-IFC) calculation, and then we have implemented this part with CUDA. Because the 4C-IFC calculation is quite resembles to Fock matrix preparation algorithm, we apply same parallelization techniques: task assignment and code tuning.

III. PERFORMANCE BENCHMARK

A. GPU-Acceleration Speedups

To evaluate performance of our GPU-enabled FMO program, we perform benchmark FMO calculation of middle-size protein (1,961 atoms) on eight nodes of HA-PACS base cluster, whose node has 2 Intel E5-2680 CPU and 4 NVIDIA M2090 GPUs. The results shows our GPU-accelerated FMO calculation is 3.8 times faster than that with CPU on-the-fly integral calculation. It should be noted that IFC calculation

"Research and Development on Unified Environment of Accelerated Computing and Interconnection for Post-Petascale Era" project of Basic Research Programs: CREST "Development of System Software Technologies for post-Peta Scale High Performance Computing," Japan Science and Technology Agency.

becomes dominant as molecular size, that is, number of fragments. When performing larger FMO calculation, GPU-acceleration will become large.

B. Large-scale FMO application

Two large-scale FMO calculation are examined. First application is influenza HA3 protein (HA3, 23,460 atoms), which is considered as acting important roles with influenza virus [9]. Second application is vitamin B receptor (VBR), which is a member of nuclear receptor, and a binding ligand within water solvent (46,641 atoms in total). For these binding model, the pair interaction energy (PIE) analysis of FMO calculation could be helpful to analyze binding mechanism. Both applications are performed as FMO-HF/6-31G(d) calculations on HA-PACS base cluster. GPU-accelerated FMO calculation of HA3 and VBR are successfully executed with 64 and 128 nodes of GPU cluster in 2 hours, respectively. It should be noted that this is the first large-scale GPU-accelerated FMO calculation.

ACKNOWLEDGMENT

The FMO benchmarks have been carried out under the "Interdisciplinary Computational Science Program" in CCS, University of Tsukuba. H.U. thanks T. Sawada, PhD. (AIST, Fujifilm Corporation) for structure modeling of Influenza HA3 protein.

REFERENCES

[1] K. Kitaura, T. Sawai, T. Asada, T. Nakano, and M. Uebayasi, "Pair interaction molecular orbital method: an approximate computational method for molecular interactions," Chem. Phys. Lett., vol. 312, pp. 319-324, 1999.

[2] I. S. Ufimtsev and T. J. Martinez, "Quantum Chemistry on Graphical Processing Units. 1. Strategies for Two-Electron Integral Evaluation," J. Chem. Theory Comput., vol. 4, pp. 222-231, 2008.

[3] K. Yasuda, "Two-electron integral evaluation on the graphics processor unit," J. Computat. Chem., vol. 29, pp. 334-342, 2008.

[4] A. Asadchev, V. Allada, J. Felder, B. M. Bode, M. S. Gordon, and T. L. Windus, "Uncontracted Rys Quadrature Implementation of up to G Functions on Graphical Processing Units," J. Chem. Theory Comput., vol. 6, pp. 696-704, 2010.

[5] K. A. Wilkinson, P. Sherwood, M. F. Guest, and K. J. Naidoo, "Acceleration of the GAMESS-UK electronic structure package on graphical processing units," J. Computat. Chem., vol. 32, pp. 2313-2318, 2011.

[6] S. A. Losilla, M. A. Watson, A. Aspuru-Guzik, and D. Sundholm, "Construction of the Fock Matrix on a Grid-Based Molecular Orbital Basis Using GPGPUs," J. Chem. Theory Comput., vol. 11, pp. 2053-2062, 2015/05/12 2015.

[7] H. Umeda, T. Hanawa, M. Shoji, T. Boku, Y. Inadomi, "Fock Matrix Preparation in Fragment Molecular Orbital Method with GPGPU," IPSJ Transactions on Advanced Computing Systems, vol. 6, no. 4, pp. 26-37, 2013.

[8] H. Umeda, Y. Inadomi, T. Watanabe, T. Yagi, T. Ishimoto, T. Ikegami, H. Tadano, T. Sakurai, and U. Nagashima, "Parallel Fock matrix construction with distributed shared memory model for the FMO-MO method," J. Comput. Chem., vol. 31, pp. 2381-2388, 2010.

[9] T. Sawada, D. G. Fedorov, and K. Kitaura, "Binding of Influenza A Virus Hemagglutinin to the Sialoside Receptor Is Not Controlled by the Homotropic Allosteric Effect," J. Phys. Chem. B, vol. 114, pp. 15700-15705, 2010/12/02 2010.