Implementation of Multiple-precision arithmetic Krylov Subspace Method on GPU using CUMP

Soichiro Ikuno[1], Yuta Hirokawa[1], and Taku Itoh[1]

[1]School of Computer Science, Tokyo University of Technology, Japan

1 Introduction

The Large and the sparse matrices appear as a part of the nonlinear engineering and physical simulation, and the matrices must be solved in a short amount of time. The General Purpose Computing on Graphics Processing Unit (GPGPU) is a one of the solution of reducing the calculation time. In recent years, GPU architectures are capable of scientific computing more than the specific graphics operation, and various researches of GPGPU have been proposed.

It is known that the number of iteration of the Krylov subspace method depends on the accuracy of the calculation. As might be expected from above reason, the number of iteration can be reduced by using high precision arithmetic. On the other hand, it is easy to expect that the CPU time and transmission time of data from memories of multiple-precision operation are much slower than that of normal precision operation because a multiple-precision arithmetic use long bit length for each operation. From this reason, the parallelization of multiple-precision arithmetic must be needed for reducing the CPU time.

The purpose of the present study is to develop the numerical code for solving the linear system obtained from the electromagnetic analysis using edge element. And to avoid the accumulation of error and to get a high performance, multiple-precision arithmetic is used on GPU.

2 Multiple-precision arithmetic Krylov Subspace Method and Evaluation

As is well known that it takes much coding cost to make a high performance program using GPU. Moreover, we will implement the multiple-precision arithmetic on GPU. Thus, we adopt the Conjugate Gradient (CG) method for solver for linear system obtained by an electromagnetic static field problem that discretized by using FEM with an edge element.

In the present study, The GNU Multiple Precision Arithmetic Library (GMP) and The CUDA Multiple Precision Arithmetic Library (CUMP) are used for implementing multiple-precision arithmetic on CPU and GPU [1, 2]. GMP is a free library for arbitrary precision arithmetic, and there is no limitation of the precision. In addition, GMP can be implemented on UNIX-type systems, such as Linux, BSD and Mac OS X. CUMP is a free library for arbitrary precision arithmetic on Compute Unified Device Architecture (CUDA), and it is developed based on GMP.

The procedures such as inner product of CG method are calculated on not only GPU but also CPU. For example, a procedure of an inner product is divided into number of block in GPU for parallelization, and values are calculated by convolutional technique in each block using many threads. The values that calculated in each block are gathered in main memory, and final summation is calculated by CPU. Note that we implement 100 digit precision in decimal by using GMP and CUMP. 332 bit, 32 bit and 32 bit is allocated for mantissa, exponent part and sign part, respectively.

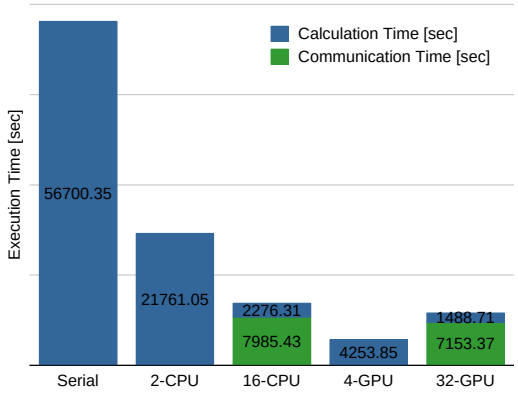

Fig. 1. The calculation time and communication time of the various parallelized CG.

The dimension size of coefficient matrix is $N = 1709028$, and 42 nonzero elements include in unit column. Thus, we adopt the Jagged Diagonal Storage (JDS) for data compression. In addition, GPU cluster super computer HA-PACS [3] installed at University of Tsukuba is used for the evaluation. In HA-PACS, four M2090 GPUs and two CPUs are implemented on unit node. The eight nodes are used for the evaluation, and MPI and OpenMP are adopted for parallelization.

In Fig. 1, we show the CPU time of the various parallelized CG. This figure indicates that the CPU time of four GPUs is much smaller than that of the serial calculation. Actually, CPU time of four GPUs is about 13.33 times faster than that of serial calculation. From this results, multiple-precision arithmetic CG method on GPU is effective methodology for electromagnetic analysis. The further detail for numerical results will describe at the conference. Also the results of Variable Preconditioned Krylov subspace method will be shown [4].

3 Conclusion

References

[1] `http://gmplib.org/`

[2] `http://www.hpcs.cs.tsukuba.ac.jp/~nakayama/cump/`

[3] `http://www.ccs.tsukuba.ac.jp/CCS/research/project/ha-pacs`

[4] K. Abe, S. L. Zhang, “A variable preconditioning using the SOR method for GCR-like methods”, Int. J. Numer. Anal. Model. 2, no. 2, (2005) 147-161.