Parallel ICCG Solvers for a Finite-Element Eddy-Current Analysis on Heterogeneous Parallel Computation Environment

T. Iwashita[1], M. Shimasaki[2], and J. Lu[3]

[1]Academic Center for Computing and Media Studies, Kyoto University, Japan

[2]Graduate School of Engineering, Kyoto University, Japan

[3]School of Microelectronic Engineering, Griffith University, Australia

iwashita@media.kyoto-u.ac.jp, simasaki@kuee.kyoto-u.ac.jp, J.Lu@griffith.edu.au

Abstract – This paper investigates fast electromagnetic field analysis on parallel computers mutually integrated by means of Grid computing technology. To utilize the heterogeneous parallel computation environment, we introduce four parallelized ICCG solvers: the block ICCG, load-balanced block ICCG, algebraic block red-black ordering, and recursive reordering methods. These solvers are evaluated in a finite edge-element eddy-current analysis on integrated parallel computers.

I. INTRODUCTION

Grid computing technology is one of the latest and most important technologies in the area of high-performance computing (see, e.g., [1]). The technology provides various functions related to computing, some of which have the possibility to significantly impact electromagnetic field computations. For example, access to non-limited computer resources enables much larger-scale numerical electromagnetic field simulation to be performed than with conventional analysis.

This paper investigates efficient utilization of heterogeneous parallel computation environments, which involve multiple parallel computers, on a finite-element electromagnetic field analysis. These computational environments are fundamental parts of the Grid computing technique. Here, we use the Globus toolkit that is a standard middleware for Grid computing and the MPICH-G communication library [2], [3]. These two tools provide us with easy access to the integrated multi-computer environment, in which standard MPI-based programs can be run. In this paper, we focus on a solution of a linear system of equations in a finite-element analysis, which is the most time-consuming part in whole FEM-based analysis.

This paper examines four parallelized ICCG (Incomplete Cholesky Conjugate Gradient) [4] solvers in a heterogeneous parallel computation environment. The ICCG method is the most popular solver for a linear system of equations arising in electromagnetic field analyses. While there are several parallelization ways of the ICCG method [5], [6], we focus on four techniques that have an advantage in communication cost. For example, the multi-color ordering method [7] that is one of conventional parallel ICCG methods needs much communication between processors. The first solver is the block ICCG method [8], which is based on localized incomplete factorization, and has the advantage of no communications in parallelized substitutions. The second solver is the load-balanced block ICCG method, in which a load-balancing technique is applied to the block ICCG method. The third solver is the algebraic block red-black ordering method [9], which has a superior convergence rate. The forth solver is the newly presented recursive reordering method. In this technique, in order to utilize both site-by-site and processor-by-processor parallelisms in integrated multi parallel computers, reordering techniques are recursively applied. While there are many variations in choosing reordering methods, we use algebraic block red-black ordering and multi-color ordering. This paper investigates parallel performance of these solvers in a multi parallel computer environment.

II. PARALLEL ICCG SOLVERS

The present paper solves a linear system of equations having a positive or semi-positive definite symmetric coefficient matrix arising in finite element electromagnetic field analyses. To efficiently utilize the Grid computing environment, we use four parallel ICCG solvers for solving this linear system: 1) Block ICCG method, 2) Load-balanced block ICCG method, 3) Algebraic block red-black ordering method, 4) Recursive reordering method. We briefly describe the procedure for each solver, while paying special attention to parallel processing of forward and backward substitutions. Other kernels of the ICCG method are easily parallelized.

A. Block ICCG Method

The block ICCG method is a well-known parallelization

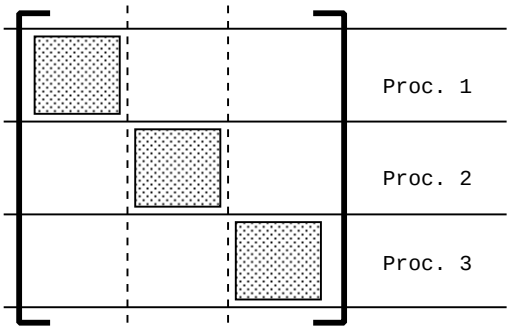

Fig. 1. Preconditioning matrix in block ICCG method (3 Processors).

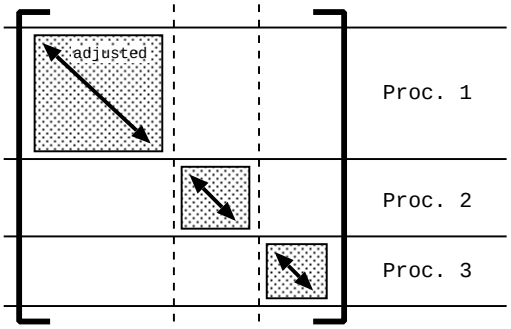

Fig. 2. Preconditioning matrix in load-balanced block ICCG method (3 Processors).

technique for incomplete Cholesky factorization preconditioning. The global matrix is divided into multiple parts that are distributed into processors. Since matrix entries between different processors are ignored in incomplete factorization, its preconditioning matrix can be treated in parallel among processors. Figure 1 shows the form of a preconditioning matrix in the block ICCG method. Forward and backward substitutions are performed in parallel without communications. Therefore, this method has an advantage in its communication cost, and is suitable for parallel computation environments based on lower grade network systems. However, the block ICCG method often suffers from a decline in convergence rate due to ignored matrix entries.

B. Load-Balanced Block ICCG Method

The load-balanced block ICCG method is the enhanced version of the block ICCG method. In a heterogeneous parallel computation environment, load-balancing often plays a key role. While there are several load-balancing techniques, our implementation method is: first, we execute a few iterations of the normal block ICCG method to obtain information about the computation environment. While we can use processor information obtained via Globus functions, we use performance data based on the actual execution. This is mainly because finite element electromagnetic field analysis is generally too complex to predict its computational time using limited information about processors. Next, the computational efforts on each processor are balanced by means of this performance information. The size of the block assigned to each processor is adjusted to the processor performance. Finally, block incomplete factorization is performed on each processor in parallel. Forward and backward substitutions are also parallelized in the same way as the block ICCG method. Fig. 2 depicts a preconditioning matrix in the load-balanced block ICCG method.

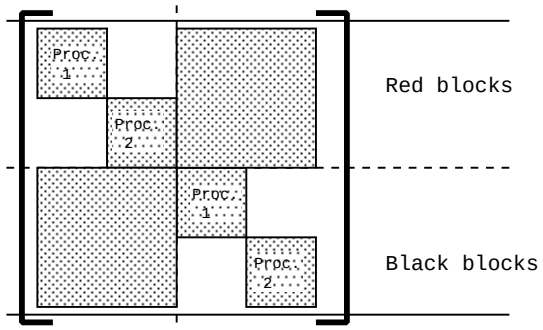

Fig. 3. Coefficient matrix in algebraic block red-black ordering method.

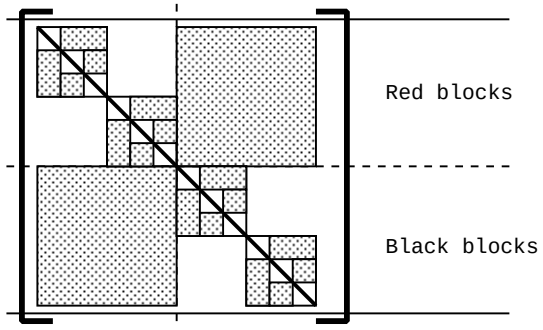

Fig. 4. Coefficient matrix in recursive reordering method.

C. Algebraic Block Red-Black Ordering Method

Reordering (parallel ordering) technique is one of the most popular techniques for parallel processing of the ICCG method. The algebraic block red-black ordering method, which is a relatively new technique, was proposed in [9]. In this method, a set of unknowns are divided into multiple groups (blocks). A well-known red-black ordering scheme is applied to the blocks. This method has two major advantages. The first advantage is

in its high convergence rate, which is generally higher than the block ICCG method, especially when the number of processors is increased. The second advantage is the low communication cost. In this method, only one synchronization point exists in parallelized forward (or backward) substitution. Although its communication cost is larger than the one of the block ICCG method that requires no communication in substitution, it is smaller than other reordering techniques, for example, multi-color ordering. Figure 3 shows a sample of reordered matrix based on algebraic block red-black ordering.

D. Recursive Reordering Method

In a Grid computing environment, the distance between computers is often significantly large. When multiple parallel computers far from each other collaborate with an analysis, the external computer-to-computer network has a large performance degradation compared with an internal network among processors in each parallel computer. The recursive reordering method is proposed for this computation environment. First, we reorder the coefficient matrix in order to distribute data and computations to each site. In this first reordering, the parallel ordering method that has an advantage in its communication cost is preferable. Next, in each parallel computer, another reordering technique is recursively applied to the distributed part of the coefficient matrix. The parallel ordering method that has fast convergence is appropriate to the second reordering technique. In our analysis, we use algebraic block red-black ordering and multi-color ordering for the first and the second reordering techniques, respectively. Figure 4 depicts the reordered matrix in the recursive reordering technique. In the implementation of the method, a hybrid programming tool is required; in the present analysis we use Open MP API (application programming interface) for internal parallel processing in each site, while the MPICH-G library is used for communications between different sites.

III. RESULTS

A. Test Model and Computation Environment

In the present analysis, we use the IEEJ standard benchmark model of 3-D eddy current analyses [10]. The analyzed model is discretized by first-order brick-type edge elements. Table I lists the discretization data. The electromagnetic field equations are solved by using the Galerkin method with *A*-formulation and the backward time difference method. The generated linear system of equations is solved by means of the shifted ICCG method.

The present analyses were implemented on two parallel computers connected via a Giga-bit local area network (see Fig. 5). Both computers are SMP-type parallel computers based on SPARC-compatible processors. First one is a Fujitsu HPC2500, and its peak performance is 5.2 GFlops per processor element. The second is a Fujitsu GP7000F model 900, and its peak performance is 1.2 GFlops per processor element. We used Globus 2.0 for the GRID computing middleware and MPICH-G 1.2.4 for MPI communications between the parallel computers. The parallelized program code is written by using FORTRAN and MPI. In the implementation of the recursive reordering method, we also use Open MP. The convergence criterion of the ICCG method is given by $||\boldsymbol{r}||_2/||\boldsymbol{b}||_2 < 10^{-7}$, where $\boldsymbol{r}$ and $\boldsymbol{b}$ are the residual vector and the right-hand side vector of the linear system, respectively.

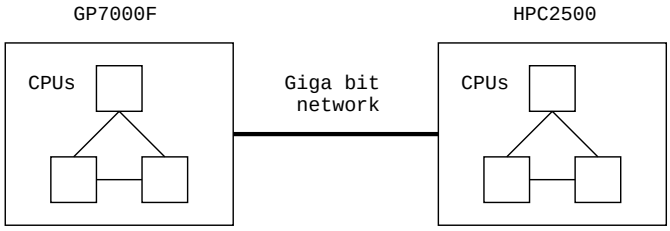

Fig. 5. Computation environment.

Table I. Discretization data.

Number of volume elements	327680
Number of nodes	342225
Number of unknowns	1011920
Time step	1 msec

B. Parallel Performance

Tables II-V list the number of iteration and the elapsed time in the four solvers. In the case of the block ICCG method, the elapsed time is reduced by increasing the number of processors. It is also indicated that the effect of the faster processor (HPC2500) is more significant than that of the processor of the GP7000. This result is caused not only by the performance difference between processors but also by the difference of the internal network performance. In the block ICCG method, the number of iterations is affected by the total number of processors. Table II shows that an increased number of processors leads to a decline in convergence. Accordingly, when more processors are used, the method may suffer from further convergence deterioration.

The results of the load-balanced block ICCG method are examined next. Table VI lists the elapsed time in one iteration of the block ICCG method and the load-balanced block ICCG method. It is shown that the load-balancing technique effectively reduces computational time in all

cases. However, in Table III, the load-balanced block ICCG method does not always obtain a better result than the normal block ICCG method. This is due to a side-effect of the convergence rate of the load-balancing technique. Since the block size assigned to a processor is dynamically changed, the preconditioning effect, i.e., the number of iterations may change in each execution of an analysis. While the side-effect also has the possibility of improving convergence, it does however cause a decline in convergence in the present analysis. Since the side-effect depends on the problem to solve, it should be examined before any given production run.

Next, we examine the results of the algebraic block red-black ordering method. Table IV indicates that the algebraic block red-black ordering technique has a better convergence rate than the block ICCG method. This superior convergence has been reported in [9]. However, it is shown that the elapsed time of the present method is longer than that of the block ICCG method. This is due to overheads of the communications between the two computers. Accordingly, the algebraic block red-black ordering method suffers from a trade-off problem between convergence and time of the present method is longer than that of the block ICCG method. This is due to overheads of the communications between the two computers. Accordingly, the algebraic block red-black ordering method suffers from a trade-off problem between convergence and communication cost in a Grid computing environment.

Finally, we examine the results of the recursive reordering technique. In the present analysis, we used 100-color ordering for the secondary reordering technique. An advantage of this method is that the number of iterations does not depend on the total number of processors. However, Table V indicates that the recursive reordering method cannot obtain a satisfactory solver performance. This is mainly due to an overhead of multi-thread parallel computation, which is a synchronization cost among processors. In particular, since the GP7000F computer is not equipped with a hardware barrier function, the synchronization cost is large, which results in the deterioration of the total solver performance. In future, when the synchronization function has been improved, this method is expected to obtain a far better solver performance.

In the present analysis, the block ICCG method, in most cases, showed the best performance among the four solvers. This result implies that communication cost is often more significant than convergence in electromagnetic field analyses in a Grid computing environment. On the other hand, the numerical results indicate that the load-balancing technique is effective for reducing computational time of one iteration. The load-balanced block ICCG method obtained the best solver performance among all cases when one GP7000F processor and three HPC2500 processors were used. However, the load-balancing technique has the possibility

Table II. Results of block ICCG method.

# of CPUs on GP7000	# of CPUs on HPC2500	Number of iterations	Elapsed time (sec)
1	1	468	1160
1	2	515	997
1	3	525	825
2	1	515	912
2	2	526	808
2	3	554	787
3	1	525	1080
3	2	553	1203
3	3	545	753

Table III. Results of load-balanced block ICCG method.

# of CPUs on GP7000	# of CPUs on HPC2500	Number of iterations	Elapsed time (sec)
1	1	656	1088
1	2	520	651
1	3	563	610
2	1	672	995
2	2	723	878
2	3	724	797
3	1	783	1177
3	2	853	1357
3	3	740	904

Table IV. Results of algebraic block red-black ordering method.

# of CPUs on GP7000	# of CPUs on HPC2500	Number of iterations	Elapsed time (sec)
1	1	378	1515
1	2	374	1214
1	3	383	1215
2	1	375	1356
2	2	383	1205
2	3	405	1237
3	1	383	1324
3	2	405	1323
3	3	391	1233

of suffering the side-effect of deterioration of convergence. The algebraic block red-black ordering method and the recursive ordering method do not attain better solver performances than the block ICCG method due to their higher communication costs. Since both methods have the advantage of high convergence rates, further development of network technology may improve their solver performances.

Table V. Results of recursive reordering method.

# of CPUs on GP7000	# of CPUs on HPC2500	Number of iterations	Elapsed time (sec)
1	1	504	2792
1	2	504	2757
1	3	504	2752
2	1	504	2330
2	2	504	2349
2	3	504	2283
3	1	504	2181
3	2	504	2265
3	3	504	2276

Table VI. Effect of load-balancing technique. (Time in one ICCG iteration (sec))

# of CPUs on GP7000	# of CPUs on HPC2500	Block ICCG method	Load-balanced block ICCG
1	1	2.48	1.66
1	2	1.93	1.25
1	3	1.62	1.08
2	1	1.77	1.48
2	2	1.53	1.21
2	3	1.42	1.10
3	1	2.06	1.50
3	2	2.17	1.59
3	3	1.38	1.22

IV. CONCLUSION

This paper investigates four parallel ICCG solvers in a multi parallel computer environment in the context of finite element electromagnetic field analyses. The results can be summarized as follows:

- In parallel ICCG solvers for Grid computation environment, a reduction of communication cost is important for getting satisfactory parallel solver performance. Although the block ICCG method is a conventional solver, it is effective in a multi computer environment due to its lower communication cost.
- The load-balancing technique is effective for the reduction of computational time in one ICCG iteration. The technique, however, affects the convergence rate of the solver.
- Although the algebraic block red-black ordering and the recursive ordering techniques have higher convergence rates than the block ICCG method, they can not attain better solver performance due to their communication or synchronization costs.

Further investigation of electromagnetic field analyses in Grid computing environments will be performed in the future; numerical tests on larger numbers of CPUs and computers will be performed for larger models.

ACKNOWLEDGMENT

A part of this work was supported by MEXT (Japan) Grant-in-Aid for Young Scientists (B) (16700060).

REFERENCES

[1] F. Berman, G. Fox, and A. Hey, *The Grid: past, present, future in Grid Computing*. Ed. West Sussex: John Wiley & Sons Ltd., pp. 9-50, 2003.

[2] I. Foster and C. Kesselman, "The Globus project: a status report," *Future Generation Computer Systems*, vol. 15, pp. 607-621, 1999.

[3] N. T. Karonis, B. Toonen, and I. Foster, "MPICH-G2: A Grid-enabled implementation of the Message Passing Interface," *Journal of Parallel and Distributed Computing*, vol. 63, pp. 551-563, 2003.

[4] J. Meijerink and H. A. van der Vorst, "An iterative solution method for linear systems of which the coefficient matrix is a symmetric M-matrix," *Mathematics of Computation*, vol. 31, pp. 148-162, 1977.

[5] Y. Saad, *Iterative Methods for Sparse Linear Systems*, Second ed., SIAM, Philadelphia, PA, 2003.

[6] H. A. van der Vorst and T. F. Chan, "Parallel Preconditioning for Sparse Linear Equations," *ZAMM. Z. angew. Math. Mech.*, vol. 76, pp. 167-170, 1996.

[7] T. Iwashita and M. Shimasaki, "Algebraic multi-color ordering for parallelized ICCG solver in finite element analyses," *IEEE Trans. Magn.*, vol. 38, pp. 429-432, 2002.

[8] C. Vollaire and L. Nicolas, "Preconditioning techniques for the conjugate gradient solver on a parallel distributed memory computer," *IEEE Trans. Magn.*, vol. 34, pp. 3347-3350, 1998.

[9] T. Iwashita and M. Shimasaki, "Algebraic block red-black ordering method for parallelized ICCG solver with fast convergence and low communication

costs," *IEEE Trans. Magn.*, vol. 39, pp. 1713-1716, 2003.

[10] T. Nakata, N. Takahashi, T. Imai, and K. Muramatsu, "Comparison of various methods of analysis and finite elements in 3-d magnetic field analysis," *IEEE Trans. Magn.*, vol. 27, pp. 4073-4076, 1991.

Takeshi Iwashita received the M.Eng. degree and the Ph.D. degree from the department of electrical engineering in Kyoto University, Japan, in 1995 and 1998, respectively. From 1998 to 1999, he worked as a research associate in Kyoto University for a Japanese national project (JSPS-RFTF project) on software for distributed parallel computing environment. In 2000, he joined the Data Processing Center in the same university. Since 2003 he has worked as an associate professor in the Academic Center for Computing and Media Studies, Kyoto University. From 2003 to 2004, he was a visiting fellow of Griffith University in Australia. His research interests include high performance computing, linear iterative solvers, and electromagnetic field simulations.

Masaaki Shimasaki received the M.Eng. degree from Department of Electronics and Ph.D. degree from Department of Electrical Engineering in Kyoto University, Japan, in 1968 and 1972, respectively. From 1978 to 1989, he worked as an associate professor of Kyoto University and from 1989 to 1997 he worked as professor of Kyushu University. Since 1997, he is a professor of Electrical Engineering of Kyoto University. In the 1974-75 academic years, he was an associate research scientist, Courant Institute of Mathematical Sciences, New York University. His research interests include high performance computing, linear iterative solvers, and electromagnetic field computation.

Junwei Lu graduated from the department of electrical engineering, Xian Jiaotong University, China, in 1976, the M.Eng. degree in electronic and computer engineering from the National Toyama University, Japan, in 1988, and Ph.D., degree in electrical and computer engineering from the National Kanazawa University, Japan, in 1991. From 1976 to 1984, he worked with the Institute of Qinhai Electric Power Testing and Research, China, where he was involved in the various national research projects for electrical power industry. Since 1985 he started his new academic study and research in the area of computational electromagnetics at the laboratory of electrical communications, Toyama University, Japan. Since 1988 he worked on the applied computational electromagnetics and involved in the development of magnetics devices with the laboratory of electrical energy conversion, Kanazawa University, Japan. He joined the School of Microelectronic Engineering, Griffith University in 1992, where he is now an associate professor and director of HPCV Lab. His fields of interest are computational and visual electromagnetics, EMC computer modeling and simulation, high performance cluster computing, smart mobile terminal antennas and RF/MW devices and circuits, and high frequency magnetics.