The Parallelization of the Incomplete LU Factorization on AP1000

Takashi NODERA and Naoto TSUNO

Department of Mathematics
Keio University
3-14-1 Hiyoshi Kohoku Yokohama 223, Japan

Abstract. Using a finite difference method to discretize a two dimensional elliptic boundary value problem, we obtain systems of linear equations $Ax = b$, where the coefficient matrix A is a large, sparse, and nonsingular. These systems are often solved by preconditioned iterative methods. This paper presents a data distribution and a communication scheme for the parallelization of the preconditioner based on the incomplete LU factorization. At last, parallel performance tests of the preconditioner, using BiCGStab(ℓ) and GMRES(m) method, are carried out on a distributed memory parallel machine AP1000. The numerical results show that the preconditioner based on the incomplete LU factorization can be used even for MIMD parallel machines.

1 Introduction

We are concerned with the solution of linear systems of equations

$$Ax = b \tag{1}$$

which arises from the discretization of partial differential equations. For example, a model problem is the two dimensional elliptic partial differential equation:

$$-u_{xx} - u_{yy} + \sigma(x,y)u_x + \gamma(x,y)u_y = f(x,y) \tag{2}$$

which defined on the unit square Ω with Dirichlet boundary conditions $u(x,y) = g(x,y)$ on $\partial\Omega$. We require $\sigma(x,y)$ and $\gamma(x,y)$ to be a bounded and sufficiently smooth function taking on strictly positive values. We shall use a finite difference approximation to discretize the equation (2) on a uniform grid points (**MESH** $\times$ **MESH**) and allow a five point finite difference molecule A. The idea of incomplete LU factorization [1, 4, 7] is to use $A = \tilde{L}\tilde{U} - R$, where

$$A = \begin{Bmatrix} & A^N_{i,j} & \\ A^W_{i,j} & A^C_{i,j} & A^E_{i,j} \\ & A^S_{i,j} & \end{Bmatrix} \quad \tilde{L} = \begin{Bmatrix} 0 & & \\ L^W_{i,j} & L^C_{i,j} & 0 \\ & L^S_{i,j} & \end{Bmatrix}, \quad \tilde{U} = \begin{Bmatrix} & U^N_{i,j} & \\ 0 & U^C_{i,j} & U^E_{i,j} \\ & 0 & \end{Bmatrix} \tag{3}$$

where $\tilde{L}$ and $\tilde{U}$ are lower and upper triangular matrices, respectively. The incomplete LU factorization preconditioner which is efficient and effective for the

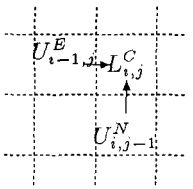

Fig. 1. The data dependency among neighborhoods.

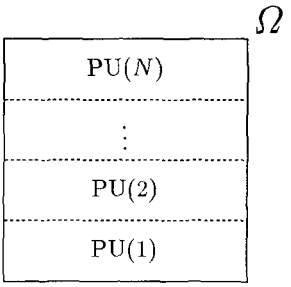

Fig. 2. The data distribution for N processors.

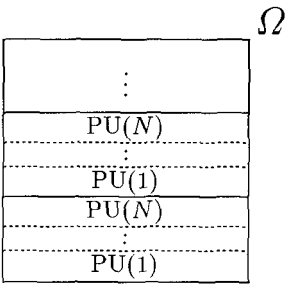

Fig. 3. The cyclic data distribution for N processors.

single CPU machine, may not be appropriate for the distributed memory parallel machine just like AP1000.

In this paper, we describe some recent work on the efficient implementation of preconditioning for a MIMD parallel machine. In section 2, we discuss the parallelization of our algorithm, and then we describe the implementation of incomplete LU factorization on the MIMD parallel machine. In section 3, we present some numerical experiments in which BiCGStab(ℓ) algorithm and GMRES(m) algorithm are used to solve a two dimensional self-adjoint elliptic boundary value problem.

2 Parallelization

In section 1, as we presented the incomplete LU factorization, it involves in two parts. One is the decomposition of A into $\tilde{L}$ and $\tilde{U}$. The another is the inversion of $\tilde{L}$ and $\tilde{U}$ by forward and backward substitution. First of all, we consider the decomposition process. The data dependency of this factorization is given in Fig. 1. Namely, the entry of $L^C_{1,j}$ can be calculated as soon as $L^C_{i-1,j}$ and $L^C_{i,j-1}$ have been calculated on the grid points. The $L^C_{1,j}$ only depends on $L^C_{1,j-1}$ on the west boundary and $L^C_{i,1}$ only depends on $L^C_{i-1,1}$ on the south boundary. Next, we consider the inversion process. The calculation of $w_{i,j} = \tilde{L}^{-1}v$ and $w_{i,j} = \tilde{U}^{-1}v$ is given by forward and backward substitution as follows.

$$w_{i,j} = (v_{i,j} - L^S_{i,j}w_{i,j-1} - L^W_{i,j}w_{i-1,j})/L^C_{i,j} \tag{4}$$

$$w_{i,j} = v_{i,j} - U^N_{i,j}w_{i,j+1} - U^E_{i,j}w_{i+1,j} \tag{5}$$

We now consider brief introduction to the method of parallelization by Bastian and Horton [4]. They proposed the algorithm of dividing the domain Ω, as shown in Figure 2, when we do the parallel processing by N processors such as PU(1), ..., PU(N). Namely, PU(1) starts with its portion of the first low of grid points. After end, a signal is sent to PU(2), and then it starts to process its

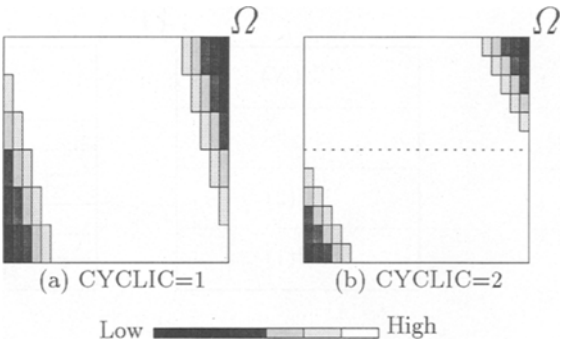

Fig. 4. The efficiency of parallel computation in the grid points on Ω.

section of the first row. At the time, PU(1) has started with the calculation of its component of the second row. All processors are running when PU(N) begins with the calculation of its component of the first row.

2.1 Generalization of the algorithm by Bastian and Horton

As we can show in Fig. 3, we divide the domain Ω by N processors cyclically. Here, we denote CYCLIC which denotes the number of cyclic division of the domain. The parallelization of Fig. 2 is equal to CYCLIC=1. If it divides as shown in the Fig. 3, the computation of the k-th processor PU(k) can be begun more quickly than the case of Bastian and Horton. Therefore, in this case, the efficiency of parallel processing will be higher than the ordinary one.

In Fig. 4, we show the efficiency of parallel computation in the square grid points for CYCLIC= 1, 2, respectively. In these figures, the efficiency of parallel processing is shown a deeper color of the grids is low.

The efficiency E of the parallel processing by this division is calculated as following equation 6. The detailed derivation of this equation is given in the forthcoming our paper [10,11].

$$E = \frac{\text{CYCLIC} \times \text{MESH}}{\text{CYCLIC} \times \text{MESH} + N - 1} \tag{6}$$

On the other hand, for distributed memory parallel machine, the number of CYCLIC is increased, the communication between processors is also increased. Moreover, concealment of the communication time by calculation time also becomes difficult. Therefore, we will consider the trade-off between the efficiency of parallel processing and the overhead of communication.

3 Numerical Experiences

In this section we demonstrate the effectiveness of our proposed implementation of ILU factorization on a distribute memory machine Fujitsu AP1000 for some

Table 1. AP1000 specification

Architecture	Distributed Memory, MIMD
Number of processors	64
Inter processor networks	Broadcast network(50MB/s) Two-dimensional torus network (25MB/s/port) Synclronization network

Table 2. The computational time (sec.) for the multiplication of matrix $A(\tilde{L}\tilde{U})^{-1}$ and vector v on example 1.

Mesh Size	CYCLIC	$(\tilde{L}\tilde{U})^{-1}v$	Av	Total
128×128	1	2.52×10^{-2}	2.88×10^{-3}	2.81×10^{-2}
	2	5.86×10^{-2}	6.08×10^{-3}	6.47×10^{-2}
256×256	1	5.44×10^{-2}	9.18×10^{-3}	6.36×10^{-2}
	2	1.41×10^{-1}	1.41×10^{-2}	1.55×10^{-1}
	4	2.12×10^{-1}	2.68×10^{-2}	2.38×10^{-1}
512×512	1	1.56×10^{-1}	4.16×10^{-2}	1.98×10^{-1}
	2	3.31×10^{-1}	5.05×10^{-2}	3.81×10^{-1}
	4	5.05×10^{-1}	6.70×10^{-2}	5.72×10^{-1}
	8	8.26×10^{-1}	1.21×10^{-1}	9.47×10^{-1}

large sparse matrices problems. The Specification of AP1000 is given in Table 1. Each cell of AP1000 employs RISC-type SPARC or SuperSPARC processor chip. As Table 2 shows, the computational time for $A(\tilde{L}\tilde{U})^{-1}v$ for various CYCLIC, which obtained by the following example 1. From these results, it is considered suitable by the AP1000 to decrease data volume of the communication between processors, using CYCLIC= 1. So, we use 1 as the number of CYCLIC in the following numerical experiments.

[Example 1.] This example involves discretizations of boundary value problem for the partial differential equation (Joubert [5]).

$$-u_{xx} - u_{yy} + \sigma u_x = f(x, y)$$
$$u(x, y)|_{\partial\Omega} = 1 + xy$$

where the right-hand side is taken such that the true solution is $u(x, y) = 1 + xy$ on Ω. We discretize above equation 7 on a 256×256 mesh of equally spaced discretization points. By varying the constant σ, the amount of nonsymmetricity of the matrix may be changed. We utilize the initial approximation vector of $x_0 = 0$ and use simple stopping criterion $\|r_k\|/\|r_0\| \leq 10^{-12}$. We also used double precision real arithmetic to perform the runs presented here.

In Table 3, we show the computational times required to reduce the residual norm by a factor of 10^{-12} for an average over 3 trials of preconditioned or un-

Table 3. The computational time (sec.) for example 1.

Algorithm	σh									
	0	2^{-3}	2^{-2}	2^{-1}	2^{0}	2^{1}	2^{2}	2^{3}	2^{4}	2^{5}
GMRES(5)	*	*	63.25	31.74	33.51	32.15	32.98	32.82	36.68	43.91
GMRES(5)+ILU	*	48.21	25.74	27.29	22.91	18.10	12.24	9.23	7.07	6.13
GMRES(10)	*	117.04	50.52	47.75	50.41	50.89	50.75	47.82	44.42	43.54
GMRES(10)+ILU	296.18	36.25	39.58	40.95	37.02	25.45	16.08	11.94	8.31	6.63
GMRES(20)	*	109.68	89.45	88.79	94.16	92.41	92.00	90.39	84.65	79.01
GMRES(20)+ILU	197.47	60.54	76.01	70.29	54.82	36.51	29.32	15.55	9.83	5.89
BiCGStab(1)	30.11	23.83	20.34	20.66	23.18	23.04	48.42	101.04	*	*
BiCGStab(1)+ILU	30.34	21.28	17.82	17.25	15.09	10.91	8.46	6.73	4.86	3.13
BiCGStab(2)	35.16	26.78	24.14	24.67	26.38	29.65	30.07	25.64	24.69	24.90
BiCGStab(2)+ILU	31.86	22.87	19.67	21.27	16.47	10.37	7.47	6.51	5.23	3.63
BiCGStab(4)	47.28	33.40	30.93	32.86	39.23	39.51	36.47	36.75	32.33	30.96
BiCGStab(4)+ILU	34.38	24.56	22.47	23.17	18.99	12.00	8.49	7.79	5.69	4.30

* It does not converge after 3000 iterations.

preconditioned BiCGStab(ℓ) and GMRES(m) algorithm. For the preconditioner of incomplete LU factorization, in most cases both methods worked quite well.

References

1. Meijerink, J. A. and van der Vorst, H. A.: An iterative solution method for linear systems of which the coefficient matrix is symmetric M-matrix, *Math. Comp.*, Vol. 31, pp. 148-162 (1977).
2. Gustafsson, I.: A class of first order factorizations, *BIT* Vol. 18, pp. 142-156 (1978).
3. Saad, Y. and Schultz, M. H.: GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems, *SIAM J. Sci. Stat. Comput.*, Vol. 7, No. 3, pp. 856–869 (1986).
4. Bastian, P. and Horton, G.: Parallelization of robust multigrid methods: ILU factorization and frequency decomposition method, *SIAM J. Sci. Stat. Comput.*, Vol. 12, No. 6, pp. 1457–1470 (1991).
5. Joubert, W.: Lanczos methods for the solution of nonsymmetric systems of linear equations, *SIAM J. Matrix Anal. Appl.*, Vol. 13, No. 3, pp. 926–943 (1992).
6. Sleijpen, G. L. G. and Fokkema, D. R.: BiCGSTAB(ℓ) for linear equations involving unsymmetric matrices with complex spectrum, *ETNA*, Vol. 1, pp. 11–32 (1993).
7. Bruaset, A. M.: A survey of preconditioned iterative methods, Pitman Research Notes in Math. Series 328, Longman Scientific & Technical (1995).
8. Nodera, T. and Noguchi, Y.: Effectiveness of BiCGStab(ℓ) method on AP1000, Trans. of IPSJ (in Japanese), Vol. 38, No. 11, pp. 2089-2101 (1997).
9. Nodera, T. and Noguchi, Y.: A note on BiCGStab(ℓ) Method on AP1000, *IMACS Lecture Notes on Computer Science*, to appear (1998).
10. Tsuno, N.: The automatic restarted GMRES method and the parallelization of the incomplete LU factorization, Master Thesis on Graduate School of Science and Technology, Keio University (1998).
11. Tsuno, N. and Nodera, T.: The parallelization and performance of the incomplete LU factorization on AP1000, Trans. of IPSJ (in Japanese), submitted.