Contemporary Mathematics
Volume 180, 1994

IBLU Preconditioners for Massively Parallel Computers

E. DE STURLER

December 1, 1994

ABSTRACT. ILU and MILU are more or less the standard for preconditioning, but their inherent sequentiality precludes efficient parallel implementation. Blocked variants improve the parallelism, but generally they increase the iteration count.

To improve this we adapt approaches from [**2, 4**], and we define parameterized block preconditioners based on subdomains with minimal overlap. Although the choice of the parameters is an open problem, numerical experiments suggest that the iteration count can be kept almost constant going from one to 400 subdomains. We also report on the performance on a 400-processor Parsytec Supercluster at the Koninklijke/Shell-Laboratorium in Amsterdam.

1. Introduction

For efficient iterative solvers on massively parallel computers, the iteration must be efficient and the increase in the number of iterations must be low. Here we focus on preconditioners that have a simple implementation to facilitate their use in large existing programs ported to massively parallel computers. Incomplete Block LU (IBLU) preconditioners with the blocks corresponding to the local equations of the subdomains are good candidates, but they tend to increase the number of iterations significantly. We use adaptations to the approaches in [**2, 4**] to overcome this problem. We take subdomains with minimal overlap, and we modify the local equations on the artificial boundaries. The substitution of the artificial boundary conditions into the matrix gives the so-called gener-

1991 *Mathematics Subject Classification.* Primary 65F10; Secondary 65Y05.

The author wishes to acknowledge Shell Research B.V. and STIPT for the financial support of this research.

This paper is in final form and no version of it will be submitted for publication elsewhere.

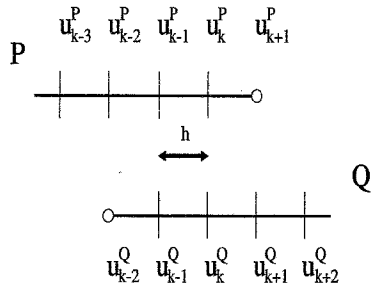

FIGURE 1. Overlapping subdomains P and Q.

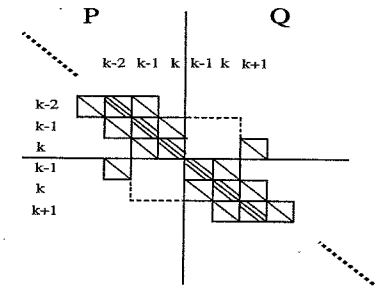

FIGURE 2. The SEM for the overlap of P and Q.

alized Schwarz enhanced matrix (SEM) [4]. From this matrix we use the ILU factorizations of the local equations to form the IBLU preconditioner.

2. Construction of the preconditioner

We describe the construction for a one-dimensional decomposition of a two-dimensional domain and the extension to higher-dimensional decompositions and more general meshes.

2.1. One-dimensional decomposition. We assume a block tridiagonal matrix which has been derived from the well-known 5-point discretization star. We decompose the domain in the y-direction and we duplicate on each subdomain a number of grid lines from the neighbouring subdomain. Figure 2 gives the matrix structure for duplicated grid lines corresponding to an overlap as indicated in Figure 1, where u_k corresponds to the approximation on the k-th grid line in the y-direction, and we have duplicated two grid lines, u_{k-1} and u_k. We will use the artificial boundary conditions

$$\beta_1 u^P_{k-1} + \alpha_1 u^P_k - u^P_{k+1} = \beta_1 u^Q_{k-1} + \alpha_1 u^Q_k - u^Q_{k+1}, \tag{2.1}$$

$$\beta_2 u^P_k + \alpha_2 u^P_{k-1} - u^P_{k-2} = \beta_2 u^Q_k + \alpha_2 u^Q_{k-1} - u^Q_{k-2}, \tag{2.2}$$

to give equations for the exterior grid lines u^P_{k+1} and u^Q_{k-2} that are substituted into the SEM to construct the generalized SEM. If we have only one overlapping grid line we cannot use the parameter β. We can use very general boundary conditions, because the generalized SEM only defines the IBLU preconditioners, and the only requirement for convergence is the nonsingularity of the preconditioner.

2.2. Higher dimensional decomposition and general meshes. Higher dimensional decompositions and more general meshes make the construction of the generalized SEM directly from the matrix complicated. Therefore, we adapt the discretization star on the boundaries of the subdomain instead; see [1]. For example, a boundary condition like (2.1) leads to the discretization star in Figure 3. This approach also permits artificial boundary conditions in

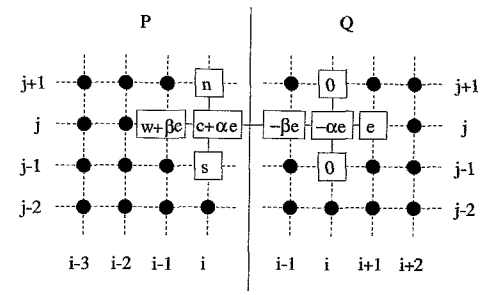

FIGURE 3. The adapted discretization star

directions non-orthogonal to the boundary; see [3].

2.3. Definition of the preconditioner. Let Ω denote the original domain, and let $\tilde{\Omega}$ be the decomposed domain with overlap. The operator $S : \Omega \mapsto \tilde{\Omega}$ assigns to a grid point in $\tilde{\Omega}$ the value of the corresponding grid point in Ω (also for duplicates). The operator $J_\omega : \tilde{\Omega} \mapsto \Omega$ is a projection using a weighted average on duplicate grid points. The operators S and J_ω satisfy the following relations. $J_\omega S : \Omega \mapsto \Omega \ = I$, and $SJ_\omega : \tilde{\Omega} \mapsto \tilde{\Omega}$ computes a weighted average over duplicate grid points and is the identity operator on other grid points.

From the generalized SEM ($\tilde{\Omega} \mapsto \tilde{\Omega}$) we compute the (block) factors $\tilde{L}$, $\tilde{D}$ and $\tilde{U}$. We define the following operators over Ω, $J_\omega \tilde{L}^{-1} S$, $J_\omega \tilde{D} S$, and $J_\omega \tilde{U}^{-1} S$, and we define the preconditioner as $K : \Omega \mapsto \Omega = J_\omega \tilde{U}^{-1} S J_\omega \tilde{D} S J_\omega \tilde{L}^{-1} S$. We can reduce the communication cost by modifying $\tilde{D}$ such that $\tilde{D}S = SD$ for some $D : \Omega \mapsto \Omega$. This only requires averaging the entries of the diagonal matrix $\tilde{D}$ corresponding to duplicate grid points. Then, K is defined by

$$K : \Omega \mapsto \Omega = J_\omega \tilde{U}^{-1} \tilde{D} S J_\omega \tilde{L}^{-1} S. \tag{2.3}$$

3. Experimental results

For the performance of the preconditioner, the iteration count is much more important than the extra computation for the overlaps and the communication with a few nearby processors. Therefore, we focus on the iteration count and discuss other overhead succinctly at the end of the section. We show only the potential of these preconditioners for reducing the number of iterations. For a discussion on the choice of parameters and the convergence see [1].

The first test problem is the Poisson equation on the unit square, $-\Delta u = 0$, with $u = 1$ for $y = 0$, $u = 0$ for $y = 1$, and $u' = 0$ for $x = 0$ and $x = 1$, where u' is the outward normal derivative on the boundary. We discretized this problem on a 100×100 grid using the finite volume method. The second test problem is given by $-(u_{xx} + u_{yy}) + bu_x + cu_y = 0$, on $[0,1] \times [0,4]$, where

$$b(x,y) = \begin{cases} 10 & \text{for } 0 \le y \le 1 \text{ and } 2 < y \le 3, \\ -10 & \text{for } 1 < y \le 2 \text{ and } 3 < y \le 4, \end{cases}$$

TABLE 1. Problem 1 with local preconditioning.

decomposition	iteration count
1×1	$2 \times 50 + 17$
1×5	$3 \times 50 + 26$
1×10	$4 \times 50 + 2$
1×20	$4 \times 50 + 6$
20×20	$5 \times 50 + 46$

TABLE 2. Problem 2 with local preconditioning.

decomposition	iteration count
1×1	$11 \times 50 + 7$
1×5	$10 \times 50 + 35$
1×10	$11 \times 50 + 33$
1×20	$12 \times 50 + 10$
20×20	$19 \times 50 + 40$

and $c = 10$, with $u = 1$ on $y = 0$, $u = 0$ on $y = 4$, $u' = 0$ for $x = 0$ and $x = 1$. We discretized this problem on a 100×200 grid using the finite volume method. Using these problems we can compare the effects of the preconditioners for diffusion- and convection dominated flow.

We used GMRES(50) for both test problems to compare the convergence results. Tables 1 and 2 show the results for several decompositions without overlap. The one-dimensional decompositions are all in the y-direction. We give the iteration count as the number of complete cycles plus the number of iterations in the last cycle. We see that the effects of the decomposition are much larger for problem 1 than for problem 2.

Tables 3 and 4 show the iteration counts using overlapping grid lines and adapted artificial boundary conditions (2.1,2.2). We give the results without adaptation, $\alpha, \beta = 0$ and (if different) for the optimal α with $\beta = 0$ and for the 1×20 decomposition also with $\beta \neq 0$. Our experiments indicated that the optimal α increases if the number of subdomains increases, and that the convergence is not very sensitive to the choice of α. Although the iteration count as a function of the number of subdomains behaves differently for the two problems, the influence of the parameters α and β seems similar.

We will now discuss the convergence using preconditioners derived from two-dimensional decompositions on a 20×20 processor grid. For simplicity we have taken ω, the overlap size, and α and β (when appropriate) the same for all

TABLE 3. Problem 1 with adapted boundary conditions.

	ovl = 1		ovl = 2	
decomp.	α	it.count	α/β	it.count
1×5	0.0	$2 \times 50 + 28$	0.0/0.0	$2 \times 50 + 25$
	0.1	$2 \times 50 + 26$	0.4/0.0	$2 \times 50 + 24$
1×10	0.0	$2 \times 50 + 39$	0.0/0.0	$2 \times 50 + 38$
	0.4	$2 \times 50 + 37$		
1×20	0.0	$3 \times 50 + 17$	0.0/0.0	$3 \times 50 + 14$
	0.7	$3 \times 50 + 8$	0.6/0.0	$2 \times 50 + 37$
			0.7/0.4	$2 \times 50 + 17$

TABLE 4. Problem 2 with adapted boundary conditions.

decomp.	ovl = 1		ovl = 2	
	α	it.count	α/β	it.count
1×5	0.0	$9 \times 50 + 15$	0.0/0.0	$10 \times 50 + 17$
			0.4/0.0	$9 \times 50 + 45$
1×10	0.0	$11 \times 50 + 44$	0.0/0.0	$11 \times 50 + 11$
	0.4	$10 \times 50 + 1$	0.3/0.0	$9 \times 50 + 41$
1×20	0.0	$10 \times 50 + 22$	0.0/0.0	$10 \times 50 + 28$
	0.5	$10 \times 50 + 8$	0.6/0.0	$10 \times 50 + 1$
			0.6/0.1	$9 \times 50 + 47$

TABLE 5. Problem 1 with adapted boundary conditions on a 20×20 processor grid.

α	β	iteration count
0.0	0.0	$7 \times 50 + 26$
0.5	0.0	$3 \times 50 + 15$

TABLE 6. Problem 2 with adapted boundary conditions on a 20×20 processor grid.

α	iteration count
0.0	$16 \times 50 + 10$
0.2	$13 \times 50 + 41$

directions.

Table 5 shows the results for problem 1 with two overlapping grid lines, and Table 6 shows the results for problem 2 with one overlapping grid line (best choices). The results for $\alpha, \beta = 0$ show that for two-dimensional decompositions without adapted boundary conditions the convergence may be poor. The other results are for the optimal values for α and β. Our experiments indicated that for two-dimensional decompositions the iteration count is much more sensitive to the choice of parameters than for one-dimensional decompositions. The results show that we can keep the increase in iterations small while running the algorithms on as much as 400 processors.

Finally, we discuss the performance for the tests with the minimal number of iterations (to illustrate the potential). We compared the measured runtimes to runtimes of a virtual preconditioned GMRES(50) that has the cycle time of GMRES(50) with a local preconditioner and the number of iterations of the sequential algorithm. This models a perfect speed-up for the preconditioner. We give the relative overhead of the preconditioners for the runtime of a cycle and for

TABLE 7. Measured runtimes and relative performance for problem 1.

overlap (x/y)	measured runtimes			relative performance		
	cycle time (s)	iteration count	solution time (s)	cycle time	iteration count	solution time
0	1.77	$5 \times 50 + 46$	10.2	1.00	2.53	2.53
2	1.91	$3 \times 50 + 15$	6.90	1.08	1.41	1.52

TABLE 8. Measured runtimes and relative performance for problem 2.

overlap (x/y)	measured runtimes: cycle time (s)	measured runtimes: iteration count	measured runtimes: solution time (s)	relative performance: cycle time	relative performance: iteration count	relative performance: solution time
0	2.21	$19 \times 50 + 40$	43.3	1.00	1.78	1.78
1	2.31	$13 \times 50 + 41$	31.8	1.05	1.24	1.30

the number of iterations. The product of these gives approximately the relative overhead for the total solution time. This can be considered as the inverse of the efficiency. The results are given in Tables 7 and 8. For comparison we also give the results for local preconditioning. The adapted preconditioners give a significantly better performance. Moreover, given the fact that we run on 400 processors in parallel, we can stay quite close to the optimal performance: about 65% for problem 1 and almost 80% for problem 2.

4. Conclusions

The IBLU preconditioners described have the potential to keep the iteration count almost constant going from one to 400 subdomains. The overhead within one cycle is only marginal, even on a large number of processors. The a priori choice of good parameters is an open problem, and future research in this direction is necessary. However, for block relaxations see [**3, 4**]. The generation of the preconditioner is straightforward and efficient; it does not introduce any significant parallel overhead.

REFERENCES

1. E. De Sturler. Incomplete Block LU preconditioners on slightly overlapping subdomains for a massively parallel computer. Technical Report CSCS-TR-94-03, Swiss Scientific Computing Center CSCS-ETHZ, 1994.
2. G. Radicati di Brozolo and Y. Robert. Parallel conjugate gradient-like algorithms for solving sparse nonsymmetric linear systems on a vector multiprocessor. *Parallel Comput.*, 11:223–239, 1989.
3. K.H. Tan and M.J.A. Borsboom. Problem-dependent optimization of flexible couplings in domain decomposition methods, with an application to advection-dominated problems. Technical Report 830, Mathematical Institute, University of Utrecht, 1993.
4. W.P. Tang. Generalized Schwarz splittings. *SIAM J. Sci. Statist. Comput.*, 13:573–595, 1992.

FACULTY OF TECHNICAL MATHEMATICS AND INFORMATICS, DELFT UNIVERSITY OF TECHNOLOGY, DELFT, THE NETHERLANDS

Current address: Section of Research and Development, Swiss Scientific Computing Center CSCS-ETHZ, Via Cantonale, CH-6928 Manno, Switzerland

E-mail address: sturler@cscs.ch