Proceedings of the International Multiconference on Computer Science and Information Technology pp. 579–590

ISSN 1896-7094
© 2006 PIPS

Parallel Performance of a 3D Elliptic Solver

Ivan Lirkov and Yavor Vutov

Institute for Parallel Processing, Bulgarian Academy of Sciences,
Acad. G. Bonchev, Bl. 25A, 1113 Sofia, Bulgaria,
`ivan@parallel.bas.bg` `yavor@parallel.bas.bg`
`http://parallel.bas.bg/~ivan/` `http://parallel.bas.bg/~yavor`

Abstract. It was shown that block-circulant preconditioners applied to a conjugate gradient method used to solve structured sparse linear systems arising from 2D or 3D elliptic problems have good numerical properties and a potential for high parallel efficiency. In this paper the convergence rate and the parallel performance of a circulant block-factorization based preconditioner applied to a 3D problem are analyzed. A portable parallel code is developed based on Message Passing Interface (MPI) standards. The performed numerical tests on parallel computer systems clearly demonstrate the high level of efficiency of the developed algorithm.

Keywords: parallel algorithms, PCG method, preconditioner, circulant. performance

1 Introduction

We are concerned with the numerical solution of linear boundary value problems of elliptic type. After discretization, such problems are reduced to find the solution of linear systems of the form $Ax = b$. We consider here symmetric and positive definite problems. We assume also, that A is a large scale matrix. In practice, large problems of this class are often solved by iterative methods, such as the conjugate gradient (CG) method. At each step of these iterative methods only the product of A with a given vector v is needed. Such methods are therefore ideally suited to exploit the sparsity of the matrix A.

Typically, the rate of convergence of these methods depends on the condition number $\kappa(A)$ of the coefficient matrix A: the smaller $\kappa(A)$ is, the faster convergence. Unfortunately, for elliptic problems of second order, usually $\kappa(A) = \mathcal{O}(n^2)$, where n is the number of mesh points in each coordinate direction, and hence grows rapidly with n. To accelerate the iteration convergence a preconditioner M is combined with the CG algorithm. The theory of the Preconditioned CG (PCG) method says that M is considered as a good preconditioner if it reduces significantly the condition number $\kappa(M^{-1}A)$, and at the same time, if the inverse matrix vector product $M^{-1}v$ can be efficiently computed for a given vector v. A third important aspect should be added to the above two, namely, the requirement for efficient implementation of the PCG algorithm on recent parallel computer systems.

One of the most popular and the most successful class of preconditioners is the class of incomplete *LU* (ILU) factorizations, see e.g. [2, 8]. One potential problem with the ILU preconditioners is that they have limited degree of parallelism. Some attempts to modify the method and to devise other more parallel methods often result in a deterioration of the convergence rate. Another class of preconditioners is proposed in [3]. These preconditioners are based on averaging coefficients of A to form a block-circulant approximation (see also [10, 12, 25]). The usage of the block-circulant approximations is motivated by their fast inversion based on the FFT (see [6, 20]). In addition, the research on circulant preconditioners for Toeplitz systems [5, 4, 11, 12] shows good results for favorable clustering of eigenvalues of the preconditioned system. The block-circulant preconditioners are highly parallelizable, see, e.g., [13, 17, 21], but they are substantially sensitive with respect to a possible high variation of the coefficients of the elliptic operator.

The sensitivity of the block-circulant approximations with respect to a high variation of the problem coefficients was relaxed in the Circulant Block-Factorization (CBF) preconditioners in [18]. This preconditioning technique incorporates the circulant approximations into the framework of the *LU* block-factorization. The computational efficiency and parallelization of the resulting algorithm is as high as of the block circulant one (see [3, 13, 17]).

The goal of the present work is to study the convergence rate of the CBF preconditioners for 3D elliptic problems as well as their parallel performance. The relative condition number $\kappa(M^{-1}A)$ is estimated with $\mathcal{O}(n)$ which is the same as for the 2D elliptic problems. The analysis of the parallel complexity shows that the algorithm is asymptotically optimal. The presented numerical tests demonstrate the features of the CBF preconditioners for 3D elliptic problems.

2 Circulant Block-Factorization Preconditioner

Let us recall that a circulant matrix C has the form $(C_{k,j}) = \left(c_{(j-k) \bmod m}\right)$, where m is the size of C. Let us also denote for any given coefficients $(c_0, c_1, \ldots, c_{m-1})$ by $C = (c_0, c_1, \ldots, c_{m-1})$ the circulant matrix

$$\begin{bmatrix} c_0 & c_1 & c_2 & \ldots & c_{m-1} \\ c_{m-1} & c_0 & c_1 & \ldots & c_{m-2} \\ \vdots & \vdots & \vdots & & \vdots \\ c_1 & c_2 & \ldots & c_{m-1} & c_0 \end{bmatrix}.$$

Any circulant matrix can be factorized as

$$C = F \Lambda F^*, \tag{1}$$

where Λ is a diagonal matrix containing the eigenvalues of C, and F is the Fourier matrix

$$F = \frac{1}{\sqrt{m}} \left\{ e^{2\pi \frac{jk}{m} \mathbf{i}} \right\}_{0 \le j,k \le m-1}.$$

Here $\mathbf{i}$ stands for the imaginary unit.

Let us consider the following 3D elliptic problem:

$$\begin{gathered} -\frac{\partial}{\partial x_1}\left(k_1(x_1,x_2,x_3)\frac{\partial u}{\partial x_1}\right) - \frac{\partial}{\partial x_2}\left(k_2(x_1,x_2,x_3)\frac{\partial u}{\partial x_2}\right) - \frac{\partial}{\partial x_3}\left(k_3(x_1,x_2,x_3)\frac{\partial u}{\partial x_3}\right) \\ = f(x_1,x_2,x_3), \qquad \forall (x_1,x_2,x_3) \in \Omega, \qquad (2) \\ 0 < \sigma_{\min} \le k_1(x_1,x_2,x_3), \quad k_2(x_1,x_2,x_3), \quad k_3(x_1,x_2,x_3) \le \sigma_{\max}, \\ u(x_1,x_2,x_3) = 0, \qquad \forall (x_1,x_2,x_3) \in \Gamma = \partial\Omega, \end{gathered}$$

on the unit cube $[0,1]^3$. Let the domain be discretized by a uniform grid with n grid points in each coordinate direction. Consider the usual seven-point centered difference approximation. This discretization leads to a system of linear algebraic equations

$$A\mathbf{x} = \mathbf{b}. \tag{3}$$

If the grid points are ordered along the x_1 and x_2 directions first, the matrix A admits a block-tridiagonal structure. The diagonal blocks are block-tridiagonal matrices and the off-diagonal blocks are diagonal matrices. The matrix A can be written in the following form

$$A = tridiag(A_{i,i-1}, A_{i,i}, A_{i,i+1}) \qquad i = 1, 2, \ldots, n, \tag{4}$$

where $A_{i,i}$ are block-tridiagonal matrices which corresponds to one x_3-plane.

We use now the general form of the CBF preconditioning matrix M for the matrix A by

$$M = tridiag(C_{i,i-1}, C_{i,i}, C_{i,i+1}) \qquad i = 1, 2, \ldots n, \tag{5}$$

where $C_{i,j} = Block - Circulant(A_{i,j})$ is block-circulant approximation of the corresponding block $A_{i,j}$. The approach of defining block-circulant approximations can be interpreted as simultaneous averaging of the matrix coefficients and changing of the Dirichlet boundary conditions to periodic ones.

The algorithm (sequential and parallel) of the CBF preconditioner is described in [14–16]. In the next section an estimate of the relative condition number of the preconditioner for a model problem is derived.

3 Model Problem Analysis of the Condition Number

We consider in this section the model 3D elliptic problem

$$\begin{gathered} -k_1 u_{x_1x_1} - k_2 u_{x_2x_2} - k_3 u_{x_3x_3} = f(x_1,x_2,x_3), \qquad \forall (x_1,x_2,x_3) \in \Omega, \qquad (6) \\ u(x_1,x_2,x_3) = 0, \qquad \forall (x_1,x_2,x_3) \in \Gamma = \partial\Omega, \end{gathered}$$

where coefficients k_1, k_2 and k_3 are positive constants. Then, the matrix A can be written in the following form

$$A = k_1 I_n \otimes I_n \otimes T + k_2 I_n \otimes T \otimes I_n + k_3 T \otimes I_n \otimes I_n \tag{7}$$

where $T = tridiag(-1, 2, -1)$. The CBF preconditioner is defined by the following equation:

$$M = k_1 I_n \otimes I_n \otimes C + k_2 I_n \otimes C \otimes I_n + k_3 T \otimes I_n \otimes I_n \tag{8}$$

where $C = (2, -1, 0, \dots 0, -1)$ is circulant matrix. Our goal is to estimate the relative condition number $\kappa(M^{-1}A)$ where matrices A and M are defined in (7) and (8). We estimate in the next lemma the condition number $\kappa(M^{-1}A)$ by the eigenvalues of eigenproblems of a reduced size n^2.

Lemma 1. *The condition number of the preconditioned system satisfies the estimate*

$$\kappa(M^{-1}A) = \frac{\max_m \lambda_{\max}\left(R_m^{-1} Q_m\right)}{\min_m \lambda_{\min}\left(R_m^{-1} Q_m\right)},$$

where

$$\begin{aligned} R_m &= k_1 I_n \otimes C + k_2 C \otimes I_n + k_3 \delta_m I_n \otimes I_n, \\ Q_m &= k_1 I_n \otimes T + k_2 T \otimes I_n + k_3 \delta_m I_n \otimes I_n, \end{aligned}$$

and $\delta_m = 4 \sin^2 \frac{m\pi}{2(n+1)}$, $m = 1, 2, \dots n$.

Proof. For simplicity we will use the notation I for $n \times n$ identity matrix I_n. To estimate the condition number of the CBF preconditioned matrix we shall analyze the eigenvalues of the generalized eigenvalue problem

$$\begin{aligned} &(k_1 I \otimes I \otimes T + k_2 I \otimes T \otimes I + k_3 T \otimes I \otimes I)\, \mathbf{w} = \\ &\lambda\, (k_1 I \otimes I \otimes C + k_2 I \otimes C \otimes I + k_3 T \otimes I \otimes I)\, \mathbf{w}\ . \end{aligned} \tag{9}$$

It is easy to compute the eigenvalues of the matrix T, that are expressed by

$$\delta_m(T) = 4 \sin^2 \frac{m\pi}{2(n+1)}, \qquad m = 1, 2, \dots n.$$

Then the matrix T can be factorized in the form $T = V^T D V$, where D is the diagonal matrix of the eigenvalues of T, the matrix V has the corresponding eigenvectors of T and V is orthogonal matrix (i.e., $V^T V = I$). Following the introduced notations we rewrite (9) in the form

$$\begin{aligned} &\left(k_1 (V^T V) \otimes I \otimes T + k_2 (V^T V) \otimes T \otimes I + k_3 (V^T D V) \otimes I \otimes I\right) \mathbf{w} = \\ &\lambda \left(k_1 (V^T V) \otimes I \otimes C + k_2 (V^T V) \otimes C \otimes I + k_3 (V^T D V) \otimes I \otimes I\right) \mathbf{w} \end{aligned} \tag{10}$$

$$\begin{aligned} &(V^T \otimes I \otimes I)(k_1 I \otimes I \otimes T + k_2 I \otimes T \otimes I + k_3 D \otimes I \otimes I)(V \otimes I \otimes I)\mathbf{w} = \\ &\lambda (V^T \otimes I \otimes I)(k_1 I \otimes I \otimes C + k_2 I \otimes C \otimes I + k_3 D \otimes I \otimes I)(V \otimes I \otimes I)\mathbf{w}\ . \end{aligned}$$

Denoting by $\mathbf{u} = (V \otimes I \otimes I)\mathbf{w}$ we obtain

$$(k_1 I \otimes I \otimes T + k_2 I \otimes T \otimes I + k_3 D \otimes I \otimes I)\mathbf{u} = \lambda (k_1 I \otimes I \otimes C + k_2 I \otimes C \otimes I + k_3 D \otimes I \otimes I)\mathbf{u}. \tag{11}$$

It follows from (11) that the eigenvalues of (9) are solutions of the split system of eigenvalue problems

$$\begin{aligned} (k_1 I \otimes T + k_2 T \otimes I + k_3 \delta_m I \otimes I)\mathbf{u}_m &= \lambda (k_1 I \otimes C + k_2 C \otimes I + k_3 \delta_m I \otimes I)\mathbf{u}_m \\ m &= 1, 2, \dots n. \end{aligned} \tag{12}$$

Obviously, the statement of the lemma follows directly from (12).

We denote by

$$\rho = \frac{k_3}{k_1 + k_2} \delta_m. \tag{13}$$

Then the above problem (12) can be rewritten as

$$(k_1 I \otimes T + k_2 T \otimes I + (k_1 + k_2)\rho I \otimes I)\mathbf{u}_m = \lambda(k_1 I \otimes C + k_2 C \otimes I + (k_1 + k_2)\rho I \otimes I)\mathbf{u}_m,$$

$$(k_1 I \otimes (T + \rho I) + k_2(T + \rho I) \otimes I)\mathbf{u}_m = \lambda(k_1 I \otimes (C + \rho I) + k_2(C + \rho I) \otimes I)\mathbf{u}_m.$$

Hence

$$\lambda_{\max}\left(R_m^{-1}Q_m\right) = \lambda_{\max}\left((C + \rho I)^{-1}(T + \rho I)\right)$$

and

$$\lambda_{\min}\left(R_m^{-1}Q_m\right) = \lambda_{\min}\left((C + \rho I)^{-1}(T + \rho I)\right).$$

We will use in the rest part of this section the determinants $\varDelta_i$, defined for a fixed value of ρ.

Definition 1. *We denote by $\varDelta_i = det(tridiag(-1, 2 + \rho, -1))$, where i stands for the dimension of the determinant.*

Now, we derive directly from the definition the recurrence equation

$$\varDelta_i = (2 + \rho)\varDelta_{i-1} - \varDelta_{i-2}, \tag{14}$$

where $\varDelta_0 = 1$ and $\varDelta_1 = 2 + \rho$. We determine $\varDelta_i$ from the recurrence equation (14), and find

$$\varDelta_i = \frac{1 - \psi^{2i+2}}{\psi^i(1 - \psi^2)}, \tag{15}$$

where ψ is one of the roots (to be chosen later) of the square equation

$$\psi^2 - (2 + \rho)\psi + 1 = 0. \tag{16}$$

Here we use Lemma 2 from [19] which define explicitly the eigenvalues of a generalized eigenvalue problem of the form involved in Lemma 1.

Lemma 2. *The matrix $(T + \rho I)^{-1}(C + \rho I)$ has exactly two eigenvalues different from unity, and they are*

$$\lambda_{1,2} = 1 + \frac{1 \pm \varDelta_{n-1}}{\varDelta_n}. \tag{17}$$

Let us remind that the goal of this section is to estimate the condition number of the CBF preconditioned matrix in the terms of k_1, k_2, k_3 and n. This result is the contents of the next theorem.

Theorem 1. *The condition number of the CBF preconditioned matrix for the model problem (6) satisfies the inequality*

$$\kappa\left(M^{-1}A\right) < \sqrt{4 + 2\frac{k_1 + k_2}{k_3}(n + 1)^2} < \sqrt{2\frac{k_1 + k_2}{k_3}}(n + 1) + 2.$$

Proof. From Lemma 1 and Lemma 2 it follows the estimate

$$\kappa(M^{-1}A) = \frac{\max_m \lambda_1}{\min_m \lambda_2}, \tag{18}$$

where $\lambda_{1,2}$ are given by (17), depending on m, as $\rho = \frac{k_3}{k_1 + k_2}\delta_m$. Now we chose ψ to be the larger root of (16), which implies $\psi > 1$. It follows from (15), that $\varDelta_i$ can be expanded in the form

$$\varDelta_i = \frac{1}{\psi^i} + \frac{1}{\psi^{i-2}} + \cdots + \psi^{i-2} + \psi^i.$$

Hence

$$\varDelta_n = \psi\varDelta_{n-1} + \frac{1}{\psi^n},$$

and therefore

$$\lambda_2 = 1 + \frac{1 - \varDelta_{n-1}}{\varDelta_n} = 1 + \frac{1 - \varDelta_{n-1}}{\psi\varDelta_{n-1} + \frac{1}{\psi^n}} = \frac{(\psi - 1)\varDelta_{n-1} + 1 + \frac{1}{\psi^n}}{\psi\varDelta_{n-1} + \frac{1}{\psi^n}}$$

$$> \frac{(\psi - 1)\varDelta_{n-1}}{(\psi + 1)\varDelta_{n-1}} = \frac{\psi - 1}{\psi + 1} = \frac{1}{\sqrt{1 + 4/\rho}}, \tag{19}$$

and

$$\lambda_1 < 2. \tag{20}$$

Combining the estimates (18), (19) and (20) we get the estimate for the condition number

$$\kappa(M^{-1}A) < 2 \max_m \sqrt{1 + \frac{4}{\rho}} = 2 \max_m \sqrt{1 + \frac{4(k_1 + k_2)}{k_3 \delta_m}}.$$

At the end, we use the inequality $\delta_m = 4 \sin^2 \frac{m\pi}{2(n+1)} > \frac{8}{(n+1)^2}$, and obtain the final result of the theorem, namely

$$\kappa(M^{-1}A) < \sqrt{4 + 2\frac{k_1 + k_2}{k_3}(n+1)^2} < \sqrt{2\frac{k_1 + k_2}{k_3}}(n+1) + 2. \tag{21}$$

Corollary 1. *The condition number of the CBF preconditioned matrix for the Poisson problem satisfies the inequality*

$$\kappa\left(M^{-1}A\right) < 2\sqrt{1 + (n+1)^2} < 2(n + \sqrt{2}).$$

It is well known that the ordering of the unknowns has a strong influence on the convergence rate of the PCG algorithms for anisotropic problems. For example, the winning strategy for the ILU, the multilevel, and the multigrid algorithms can be formulated as *following the mesh points along the lines of dominating anisotropy* [7, 9, 22]. The following remark shows that when CBF preconditioners are used, just the opposite ordering (*along the lines of week anisotropy*) improves the convergence.

Remark 1. From equation (21) it follows that the convergence of PCG method with CBF preconditioner is faster if the grid points are ordered along directions with smaller coefficients of the differential equation first.

4 Analysis of the Parallel Complexity

We assume that the computations and communications are not overlapped and therefore, the execution time of the parallel implementation is the sum of the computation time and the communication time.

We shall use in our analysis standard models for the arithmetic and communication times [23]. First, assuming no arithmetic vectorization, the execution of M arithmetic operations on one processor takes time $T_a = M * t_a$, where t_a is the average unit time to perform one arithmetic operation on one processor. Let us consider a parallel computer system consisting of p processors. The communication time of transfer of M words between two neighbor processors is approximated by $T_c = t_s + M * t_c$, where t_s is the start-up time and t_c is the incremental time necessary for each of all M words to be sent.

We denote the following two communication times:

- $b(p)$ — broadcasting a number from one processor to all others;
- $g(M, p)$ — gathering p data packets, each packet with M/p words, in one processor from all others.

For example for cluster the communication times (see [1]) are respectively:

$$b(p) = \lceil \log p \rceil (t_s + t_c)$$
$$g(M, p) = (p-1)\left(t_s + \frac{M}{p} t_c\right)$$

To achieve a low cost of the processor synchronization and communication, we partition the computational domain into p layers so that each layer contains n/p x_1-planes. We map all grid points in one layer onto one processor. In this way, the CG method requires a communication of only values from one grid plane with "neighbor" processors. Parallel algorithm for solving a system with CBF preconditioner is constructed in [14].

We can now estimate the total execution time T_{PCG} for one PCG iteration for the considered circulant block-factorization preconditioner on a parallel system. Each iteration consists of one matrix vector multiplication with the matrix A, one solving a system of equations with preconditioner M, two inner products and three linked triads (a vector updated by a vector multiplied by a scalar). Consequently

$$T_{PCG}(p) = T_{mult} + T_{prec} + 2T_{inn\_prod} + 3T_{triads},$$

$$T_{mult} = 13\frac{n^3}{p}t_a + 4(t_s + n^2t_c), \qquad T_{inn\_prod} = 2\frac{n^3}{p}t_a + b(p),$$

$$T_{triads} = 2\frac{n^3}{p}t_a, \qquad T_{prec} = 4\frac{n^2}{p}T_{FFT}(n) + 12\frac{n^3}{p}t_a + 2g(\frac{n^3}{p}, p),$$

and where $T_{FFT}(n)$ is the time for execution of FFT on a given n-vector on one processor. If n is equal to an exact power of two, i.e., $n = 2^l$ and we use 2-radix algorithm, then $T_{FFT}(n) = 5n \log n t_a$.

Combining these results we obtain the following estimates of the execution time for the studied computer system

$$T_{PCG}(p) = 5\left(7 + 4\log n\right)\frac{n^3}{p}t_a + 4\left(t_s + n^2t_c\right) + 2g(\frac{n^3}{p}, p) + 2b(p)$$

and for cluster we obtain

$$T_{PCG}(p) = 2\left(p + \log p + 1\right)t_s + 2\left[(p-1)\frac{n^3}{p^2} + 2n^2 + \log p\right]t_c + 5\left(7 + 4\log n\right)\frac{n^3}{p}t_a.$$

The leading terms of the parallel time complexity functions are:

$$T_{PCG}(p) \approx 2pt_s + 2\frac{n^3}{p}t_c + 5(7 + 4\log n)\frac{n^3}{p}t_a. \tag{22}$$

Our next goal is to analyze the relative speed-up S_p and the relative efficiency E_p, where $S_p = \frac{T(1)}{T(p)} \le p$ and $E_p = \frac{S_p}{p} \le 1$. We apply now (22) and obtain:

$$S_p \approx \frac{5(7 + 4\log n)}{2\frac{p^2}{n^3}\frac{t_s}{t_a} + 2\frac{t_c}{t_a} + 5(7 + 4\log n)}p. \tag{23}$$

Obviously, for our preconditioner, $\lim_{n\to\infty} S_p = p$ and $\lim_{n\to\infty} E_p = 1$, i.e., the algorithm is asymptotically optimal. More precisely, if $\log n \gg \frac{p^2}{n^3}\frac{t_s}{t_a} + \frac{t_c}{t_a}$, then E_p is near to 1. Unfortunately, the start-up time t_s is usually much larger than t_a, and for relatively small n the first term of the denominator in (23) is significant, in this case the efficiency is much smaller than 1.

5 Numerical Tests

The numerical tests presented in this section illustrate the convergence rate as well as the parallel performance of the CBF algorithm for 3D elliptic problems. Both the right hand side and the initial guess are chosen to be random vectors. The computations are done with double precision. The iteration stopping criterion is $||r^{N_{it}}||/||r^0|| < 10^{-6}$, where r^j stands for the residual at the jth iteration step of the preconditioned conjugate gradient method. The code has been implemented in C and the parallelization has been facilitated using the MPI [24] library. We report the results of the experiments executed on two parallel computer systems located in Bologna, Italy.

Example 1. The first test problem is the model problem (6). Table 1 shows the number of iterations as a measure of the convergence rate, where the mesh size n and coefficients k_2 and k_3 are varied. The presented data demonstrate a behavior of the convergence, which confirms the high accuracy of the estimate of the condition number of the preconditioned matrix. In particular, the results confirm Remark 1. One can see that in Table 1 the number of iterations for small k_3 is greater than for large k_3 where the same problem is solved. The "—" sign denotes that the number of iterations is greater than 999.

Example 2. Next we compare the number of iterations for Poisson problem with *mesh anisotropy*, i.e. when the mesh-size is different in each direction. The domain is discretized by a uniform grid with $n_1 \times n_2 \times n_3$ grid points. It is shown in Table 2 that the number of iterations depends only on $\max(n_1, n_2)$ and almost does not depend on n_3.

Example 3. We consider further test problems with variable coefficients in the form

$$\frac{\partial}{\partial x_1}\left[\left(1 + \frac{\epsilon}{2}\sin\left(2\pi\left(x_1 + x_3\right)\right)\right)\frac{\partial u}{\partial x_1}\right] + \frac{\partial}{\partial x_2}\left[\left(1 + \frac{\epsilon}{2}\sin\left(2\pi\left(x_1 + x_2\right)\right)\right)\frac{\partial u}{\partial x_2}\right]$$
$$+ \frac{\partial}{\partial x_3}\left[\left(1 + \epsilon e^{x_1 + x_2 + x_3}\right)\frac{\partial u}{\partial x_3}\right] = f\left(x_1, x_2, x_3\right) \tag{24}$$

Table 1. Number of iterations for the model problem (6) for $k_1 = 1$.

n	k_3	k_2						
		1000	100	10	1	0.1	0.01	0.001
8		35	31	19	12	13	14	13
16		53	42	23	14	16	18	16
32	1	90	57	28	18	20	24	22
64		142	76	36	22	25	31	32
128		210	103	47	29	33	40	43
192		245	114	54	34	39	46	52
256		305	119	61	35	43	51	58
8		37	22	13	8	8	7	7
16		51	29	16	11	11	9	8
32	10	75	38	20	13	15	13	11
64		98	48	25	16	19	18	15
128		136	64	33	21	25	25	21
192		159	75	40	24	28	30	27
256		191	78	44	27	31	34	31
8		23	14	8	5	4	4	4
16		34	19	11	7	6	5	4
32	100	47	25	15	9	8	6	6
64		62	31	19	11	11	9	8
128		85	40	25	15	15	13	11
192		98	47	28	17	18	16	13
256		108	52	31	19	20	19	16
8		13	7	4	3	3	3	3
16		16	9	6	4	3	3	3
32	1000	22	13	8	5	5	4	4
64		32	18	11	7	6	5	4
128		43	25	15	9	8	5	6
192		53	30	18	10	10	8	7
256		58	35	20	12	11	9	8
8		49	48	27	17	19	22	22
16		82	69	35	20	23	29	34
32	0.1	146	91	43	26	28	39	44
64		214	129	55	31	35	49	61
128		326	159	74	41	47	63	84
192		429	197	80	45	53	76	100
256		532	205	86	49	60	78	106
8		54	48	35	21	27	30	37
16		121	94	55	34	34	40	50
32	0.01	282	166	77	43	43	58	74
64		431	249	96	52	54	77	99
128		590	297	120	67	70	104	136
192		796	338	134	72	79	114	170
256		899	395	146	80	81	126	188
8		50	45	36	26	35	42	34
16		136	93	67	40	55	67	61
32	0.001	346	227	125	74	76	85	91
64		714	401	201	104	99	128	142
128		—	665	272	135	119	168	215
192		—	786	381	140	135	190	249
256		—	888	328	158	144	206	298

Table 2. Number of iterations for Poisson problem for $n_3 = 8, 16, 32, 64, 128, 192, 256$.

n_1	n_2						
	8	16	32	64	128	192	256
8	11–12	13–14	16–18	20–23	26–29	31–34	34–37
16	13–14	13–14	17–19	21–24	26–30	31–34	35–38
32	17–18	17–19	17–19	21–23	26–30	32–34	37–39
64	20–23	21–23	21–23	21–23	27–29	32–34	36–38
128	26–30	28–29	27–30	27–30	28–30	32–34	35–37
192	31–33	32–35	32–35	31–34	32–34	31–34	36–38
256	33–37	35–37	37–38	34–38	35–37	35–38	35–37

where $\epsilon \in [0, 1]$ is a parameter. It is well known that the circulant preconditioners are competitive with the incomplete LU factorization for moderately varying coefficients. This reflects the averaging of the coefficients, used in the block-circulant approximations. Such a fact was already observed in [3, 18] for 2D problems. Table 3 shows that for our test problem the anisotropy reduces slightly the number of iterations.

Table 3. Number of iterations for CBF preconditioner for problem (24).

n	$\epsilon = 0$	$\epsilon = 0.01$	$\epsilon = 0.1$	$\epsilon = 1$
8	12	11	11	14
16	14	15	15	17
32	18	19	20	24
64	22	24	28	39
128	29	33	43	63
192	34	41	55	87
256	35	47	66	108

Table 4 shows execution time T_{PCG} for one PCG iteration on an IBM SP Cluster 1600 made of 64 nodes p5-575 (see `http://www.cineca.it/pagine/ibmsp5.htm`). A p5-575 node contains 8 SMP processors IBM Power5 at 1.9 GHz. One can see that the parallel efficiency obtained on up to 8 processors is close to 1. This result was expected because communications between processors in one node are very fast. Moreover, a superlinear speed-up is observed in some cases. The main reason is better usage of cache memories in parallel algorithm.

Table 5 shows execution time on an IBM Linux Cluster 1350 made of 512 2-way IBM X335 nodes (see `http://www.cineca.it/pagine/ibmlinux.htm`). Each computing node contains 2 Xeon Pentium IV processors at 3 GHz. The obtained parallel efficiency confirms the theoretical estimates from the previous section. For relatively large problems the observed parallel efficiency is above 50% even for large number of processors.

Figure 1 shows the execution time for one PCG iteration on two parallel computer systems and figure 2 shows the speed-up obtained for $n = 32, 96$, and 128. The comparison between two IBM clusters shows that IBM Power5 processors and communication between processors in one node of IBM SP cluster are faster but in general the performance of the studied parallel algorithm on the second cluster is much better.

Acknowledgments

This research was supported by grant I-1402/2004 from the Bulgarian NSF. The parallel numerical tests were supported via EC Project HPC-EUROPA RII3-CT-2003-506079.

Table 4. Parallel time, speed-up and parallel efficiency for the CBF preconditioner on IBM SP cluster.

p	T_p	S_p	E_p	T_p	S_p	E_p
	$n = 32$			$n = 64$		
1	0.013			0.130	1.00	
2	0.007	1.94	0.968	0.058	2.23	1.115
4	0.003	3.71	0.926	0.029	4.52	1.129
8	0.002	7.24	0.905	0.026	5.03	0.629
16	0.120	0.11	0.007	0.016	8.22	0.514
32	0.120	0.11	0.003	0.155	0.84	0.026
64				0.167	0.78	0.012
	$n = 48$			$n = 96$		
1	0.135	1.00		1.235		
2	0.066	2.03	1.014	0.620	1.99	0.996
3	0.045	2.97	0.991	0.412	3.00	1.000
4	0.034	3.99	0.997	0.404	3.06	0.765
6	0.022	6.03	1.006	0.223	5.53	0.922
8	0.017	7.75	0.969	0.153	8.09	1.011
12	0.019	7.24	0.603	0.281	4.39	0.366
16	0.053	2.55	0.159	0.210	5.88	0.367
24	0.046	2.90	0.121	0.212	5.83	0.243
32				0.168	7.36	0.230
48	0.121	1.12	0.023	0.109	11.29	0.235
96				0.367	3.36	0.035
	$n = 128$					
1	1.561					
2	0.676	2.31	1.155			
4	0.326	4.79	1.198			
8	0.161	8.52	1.065			
16	0.164	9.55	0.597			
32	0.220	7.10	0.222			
64	0.218	7.15	0.112			
128	0.466	3.35	0.026			

Table 5. Parallel time, speed-up and parallel efficiency for the CBF preconditioner on IBM Linux Cluster.

p	T_p	S_p	E_p	T_p	S_p	E_p
	$n = 32$			$n = 64$		
1	0.039			0.355		
2	0.021	1.86	0.932	0.213	1.67	0.835
4	0.011	3.53	0.882	0.119	2.99	0.748
8	0.006	6.47	0.809	0.059	6.00	0.750
16	0.004	10.68	0.667	0.032	11.05	0.690
32	0.003	12.37	0.387	0.016	22.01	0.688
64				0.012	28.54	0.446
	$n = 48$			$n = 96$		
1	0.246			2.311		
2	0.135	1.82	0.911	1.273	1.82	0.908
3	0.092	2.66	0.887	0.864	2.68	0.892
4	0.072	3.42	0.854	0.675	3.42	0.856
6	0.046	5.33	0.888	0.479	4.83	0.804
8	0.038	6.42	0.802	0.371	6.23	0.779
12	0.025	9.70	0.808	0.243	9.50	0.792
16	0.019	12.98	0.811	0.190	12.16	0.760
24	0.012	19.80	0.825	0.120	19.29	0.804
32				0.090	25.58	0.799
48	0.009	26.61	0.554	0.062	37.00	0.771
96				0.041	56.79	0.592
	$n = 128$			$n = 192$		
1	3.689			19.635		
2	2.111	1.75	0.874	10.543	1.86	0.931
3				7.188	2.73	0.910
4	1.185	3.11	0.778	5.553	3.54	0.884
6				3.929	5.00	0.833
8	0.628	5.88	0.734	3.037	6.46	0.808
12				1.999	9.82	0.819
16	0.335	11.02	0.689	1.578	12.44	0.777
24				1.031	19.04	0.793
32	0.165	22.33	0.698	0.832	23.61	0.738
48				0.562	34.93	0.728
64	0.084	44.03	0.688	0.386	50.85	0.795
96				0.288	68.21	0.711
128	0.059	63.00	0.492			
192				0.193	101.74	0.530

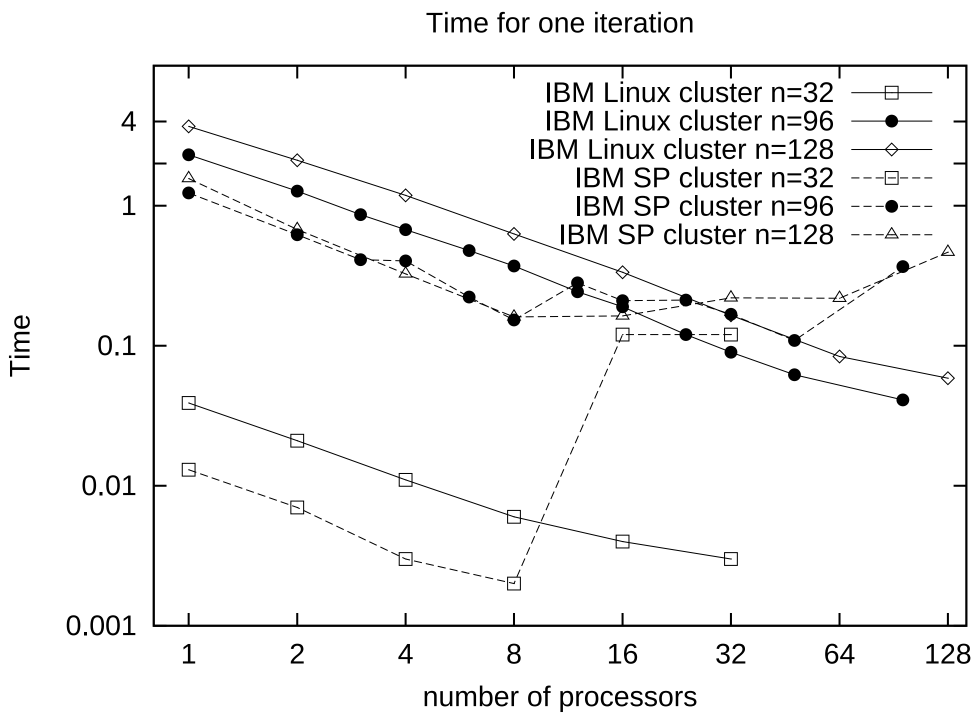

Fig. 1. Time for one iteration on the parallel computer systems

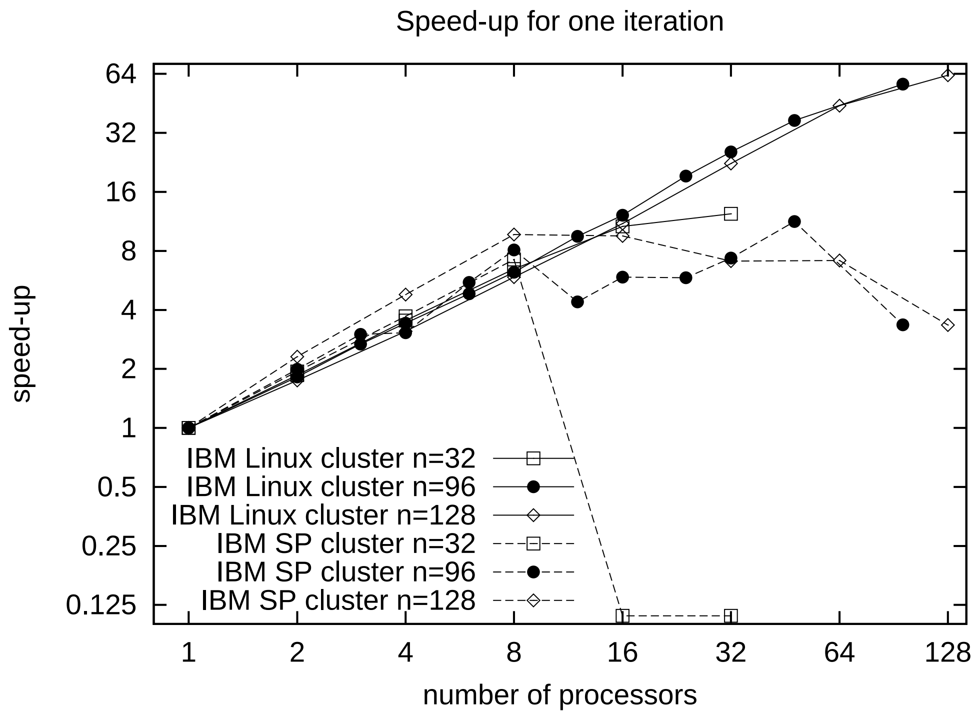

Fig. 2. Speed-up for one iteration on the parallel computer systems

References

1. A. O. H. Axelsson and M. G. Neytcheva. *Supercomputers and numerical linear algebra.* KUN, Nijmegen, 1997.
2. O. Axelsson. *Iterative solution methods.* Cambridge University Press, Cambridge, 1994.
3. R. H. Chan and T.F. Chan. *Circulant preconditioners for elliptic problems,* J. Num. Lin. Alg. Appl., **1**, 77–101, 1992.
4. R. H. Chan and G. Strang. *Toeplitz equations by conjugate gradients with circulant preconditioner,* SIAM J. Sci. Stat. Comp., **10**, 104–119, 1989.
5. T. F. Chan. *An optimal circulant preconditioner for toeplitz systems,* SIAM J. Sci. Stat. Comp., **9**, 766–771, 1988.
6. P. J. Davis. *Circulant matrices.* Johns Wiley, New York, 1979.
7. I. S. Duff and G. A. Meurant. *The effect of ordering on preconditioned conjugate gradients,* BIT, **29**, 635–657, 1989.
8. G. H. Golub and C. F. Van Loan. *Matrix Computations.* Johns Hopkins Univ.Press, Baltimore, 2nd edition, 1989.
9. W. Hackbusch. *The frequency decomposition multi-grid method. 1. Application to anisotropic equations,* Numer. Math., **56**, 219–245, 1989.
10. S. Holmgren and K. Otto. *Iterative solution methods for block-tridiagonal systems of equations,* SIAM J. Matr. Anal. Appl., **13**, 863–886, 1992.
11. T. Huckle. *Circulant and skewcirculant matrices for solving toeplitz matrix problems,* SIAM J. Matr. Anal. Appl., **13**, 767–777, 1992.
12. T. Huckle. *Some aspects of circulant preconditioners,* SIAM J. Sci. Comput., **14**, 531–541, 1993.
13. I. Lirkov and S. Margenov. *Parallel complexity of conjugate gradient method with circulant preconditioners,* Proceedings of the Parcella'96, R. Volmar, W. Erhard, and V. Jossifov, editors, *Mathematical research*, **96**, 279–286, Berlin, 1996. Akademie Verlag.
14. I. Lirkov and S. Margenov. Parallel complexity of conjugate gradient method with circulant block-factorization preconditioners for 3D elliptic problems. *Recent Advances in Numerical Methods and Applications*, O.P. Iliev, M.S. Kaschiev, Bl. Sendov, and P.V. Vassilevski, editors, 482–490, Singapore, 1999. World Scientific.
15. I. Lirkov, S. Margenov, and M. Paprzycki. *Parallel conjugate gradient method with circulant block-factorization preconditioners for 3D elliptic problems,* Proceedings of the Ninth SIAM Conference on Parallel Processing for Scientific Computing, B. Hendrickson et. al., editor, Philadelphia, 1999. SIAM. files Marcin˜1.pdf and Marcin˜1.ps.
16. I. Lirkov, S. Margenov, and M. Paprzycki. *Parallel performance of a 3D elliptic solver,* Numerical Analysis and Its Applications II, L. Vulkov, J. Wasniewski, and P. Yalamov, editors, *Lecture Notes in Computer Sciences*, **1988**, 535–543, Springer Verlag, 2001.
17. I. Lirkov, S. Margenov, M. Paprzycki, and R. Owens. *A shared memory parallel implementation of block-circulant preconditioners,* Large-Scale Scientific Computations of Engineering and Environmental problems, M. Griebel, O. Iliev, S. Margenov, and P. Vassilevski, editors, *Notes on Numerical Fluid Mechanics*, **62**, 319–327, Braunschweig, Germany, 1998. Vieweg Verlag.
18. I. Lirkov, S. Margenov, and P.S. Vassilevski. *Circulant block-factorization preconditioners for elliptic problems,* Computing, **53** 1, 59–74, 1994.
19. I. Lirkov, S. Margenov, and L. Zikatanov. *Circulant block–factorization preconditioning of anisotropic elliptic problems,* Computing, **58** 3, 245–258, 1997.
20. C. Van Loan. *Computational frameworks for the fast Fourier transform.* SIAM, Philadelphia, 1992.
21. S. Margenov and I. Lirkov. *Preconditioned conjugate gradient iterative algorithms for transputer based systems* Parallel and distributed processing, K. Boyanov, editor, 406–415, Sofia, 1993. Bulgarian Academy of Sciences.
22. S. D. Margenov and P. S. Vassilevski. *Algebraic multilevel preconditioning of anisotropic elliptic problems,* SIAM J. Matrix Anal. Appl., **15** 5, 1026–1037, 1994.
23. Y. Saad and M. H. Schultz. *Data communication in parallel architectures,* Parallel Comput., **11**, 131–150, 1989.
24. M. Snir, St. Otto, St. Huss-Lederman, D. Walker, and J. Dongarra. *MPI: The Complete Reference.* Scientific and engineering computation series. The MIT Press, Cambridge, Massachusetts, 1997. Second printing.
25. G. Strang. *A proposal for toeplitz matrix calculations,* Stud. Appl. Math., **74**, 171–176, 1986.