Proceedings of the International Multiconference on Computer Science and Information Technology pp. 483–492
ISSN 1896-7094
© 2007 PIPS

Comparative Analysis of High Performance Solvers for 3D Elliptic Problems

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. The presented comparative analysis concerns two iterative solvers for 3D linear boundary value problems of elliptic type. After applying the Finite Difference Method (FDM) or the Finite Element Method (FEM) discretization a system of linear algebraic equations has to be solved, where the stiffness matrix is large, sparse and symmetric positive definite. It is well known that the preconditioned conjugate gradient method is the best tool for efficient solution of large-scale symmetric systems with sparse positive definite matrices. Here, the performance of two preconditioners is studied, namely the Modified Incomplete Cholesky factorization MIC(0) and the Circulant Block-Factorization (CBF) preconditioning. Portable parallel codes are developed based on Message Passing Interface (MPI) standards. The comparative analysis is mostly based on the execution times to run the parallel codes. The number of iterations for both preconditioners are also discussed. The performed numerical tests on parallel computer systems demonstrate the level of efficiency of the developed algorithms. The obtained parallel speed-up and efficiency well illustrate the scope of efficient applications.

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 $A\mathbf{u} = \mathbf{b}$. 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 $\mathbf{v}$ 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 preconditioner allows efficient computation of the product $M^{-1}\mathbf{v}$ for a given vector $\mathbf{v}$. 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, see e.g. [5].

2 The 3D Elliptic Problem

Let us consider the following 3D elliptic problem:

$$\begin{gathered}
-\frac{\partial}{\partial x_1}\left(k_1\frac{\partial u}{\partial x_1}\right)-\frac{\partial}{\partial x_2}\left(k_2\frac{\partial u}{\partial x_2}\right)-\frac{\partial}{\partial x_3}\left(k_3\frac{\partial u}{\partial x_3}\right)=f(x_1,x_2,x_3),\\
\forall(x_1,x_2,x_3)\in\Omega, \qquad (1)\\
0<\sigma_{\min}\le k_1(x_1,x_2,x_3),\quad k_2(x_1,x_2,x_3),\quad k_3(x_1,x_2,x_3)\le\sigma_{\max},\\
u(x_1,x_2,x_3)=0,\qquad \forall(x_1,x_2,x_3)\in\Gamma=\partial\Omega,
\end{gathered}$$

on the unit cube $[0,1]^3$. Let the domain be discretized by a uniform grid with mesh size $h=\frac{1}{n}$.

2.1 Nonconforming Finite Element Method

Let $\mathcal{T}_h$ is a decomposition of Ω with $n\times n\times n$ cubes.

The weak formulation of the problem (1) reads as follows: for $f\in L^2(\Omega)$ find $u\in\mathcal{V}\equiv H_0^1(\Omega)=\{v\in H^1(\Omega): v=0 \text{ on } \Gamma\}$, satisfying
$\mathcal{A}(u,v)=(f,v)\quad\forall v\in H_0^1(\Omega)$, where $\mathcal{A}(u,v)=\int_\Omega\sum_{i=1}^3 k_i\frac{\partial u}{\partial x_i}\frac{\partial v}{\partial x_i}dx$.

The above variational problem is then discretized using the finite element method, i.e., the continuous space $\mathcal{V}$ is replaced by a finite dimensional subspace $\mathcal{V}_h$. Then the finite element formulation is:

find $u_h\in\mathcal{V}_h$, satisfying $\mathcal{A}(u_h,v_h)=(f,v_h)\quad\forall v_h\in\mathcal{V}_h$.

The resulting discrete problem to be solved is then a linear system of equations

$$A^{(FEM)}\mathbf{u}_h=\mathbf{f}_h, \qquad (2)$$

where $\mathbf{u}_h$ stands for the vector of unknown degrees of freedom, $A^{(FEM)}$ and $\mathbf{f}_h$ are the corresponding global stiffness matrix and global right hand side.

Non-conforming finite elements based on *rotated* multilinear shape functions were introduced by Rannacher and Turek [14] as a class of simple elements for the Stokes problem. Some more details about non-conforming finite elements can be found, e.g., in [3, 6, 9]. The cube $[-1,1]^3$ is used as a reference element $\hat{e}$ to define the isoparametric rotated trilinear element $e\in\mathcal{T}_h$. Let $\Psi_e:\hat{e}\to e$ be the corresponding trilinear one-to-one transformation, and let the nodal basis functions be determined by the relation

$\{\phi_i\}_{i=1}^6=\{\hat{\phi}_i\circ\Psi_e^{-1}\}_{i=1}^6,\qquad\{\hat{\phi}_i\}\in\text{span}\{1,x_1,x_2,x_3,x_2^2-x_1^2,x_1^2-x_3^2\}$,

where '$\circ$' denotes the composition of functions. Shape functions $\{\hat{\phi}_i\}_{i=1}^6$ used in presented experiments are found by the point-wise interpolation condition $\hat{\phi}_i(b_e^j)=\delta_{ij}$, where b_e^j, j=1,6 are the centers of the faces of the cube $\hat{e}$.

2.2 Finite Difference Method

Let us consider the usual seven-point centered difference approximation for problem (1). This discretization leads to a system of linear algebraic equations

$$A^{(FDM)}\mathbf{u} = \mathbf{b}$$

where the vector of unknowns $\mathbf{u}$ has size $(n-1)^3$. If the grid points are ordered along the x_1 and x_2 directions first, the matrix $A^{(FDM)}$ admits a block-tridiagonal structure. The matrix $A^{(FDM)}$ can be written in the following form $A^{(FDM)} = tridiag(A_{i,i-1}, A_{i,i}, A_{i,i+1})$, $i = 1, 2, \ldots, n-1$, where the diagonal blocks $A_{i,i}$ are block-tridiagonal matrices which corresponds to one x_3-plane and the off-diagonal blocks are diagonal matrices.

3 MIC(0) Factorization Preconditioning

Let us first recall some well known facts about the modified incomplete factorization MIC(0). Let us decompose the real $N \times N$ matrix A in the form $A = D - L - L^T$, where D is the diagonal and $(-L)$ is the strictly lower triangular part of A. Then we consider the approximate factorization of A which has the following form:

$$M_{MIC(0)}(A) = (X - L)X^{-1}(X - L)^T,$$

where $X = \mathrm{diag}(x_1, \ldots, x_N)$ is a diagonal matrix determined such that A and $M_{MIC(0)}$ have equal row sums. We restrict ourselves to the case when $X > 0$, i.e., when $M_{MIC(0)}$ is positive definite. In this case, the MIC(0) factorization is called *stable*. Concerning the stability of the MIC(0) factorization, we have the following theorem [8].

Theorem 1. *Let $A = (a_{ij})$ be a symmetric real $N \times N$ matrix and let $A = D - L - L^T$ be the splitting of A. Let us assume that (in an element-wise sense)*

$$L \geq 0,\ A\mathbf{e} \geq 0,\ A\mathbf{e} + L^T\mathbf{e} > 0, \qquad \mathbf{e} = (1, \cdots, 1)^T \in \mathbb{R}^N,$$

i.e., that A is a weakly diagonally dominant matrix with non-positive off-diagonal entries and that $A + L^T = D - L$ is strictly diagonally dominant. Then the relation

$$x_i = a_{ii} - \sum_{k=1}^{i-1} \frac{a_{ik}}{x_k} \sum_{j=k+1}^{N} a_{kj} > 0 \tag{3}$$

holds and the diagonal matrix $X = \mathrm{diag}(x_1, \cdots, x_N)$ defines a stable MIC(0) factorization of A.

Remark 1. The numerical tests presented in this work are performed using the perturbed version of MIC(0) algorithm, where the incomplete factorization is

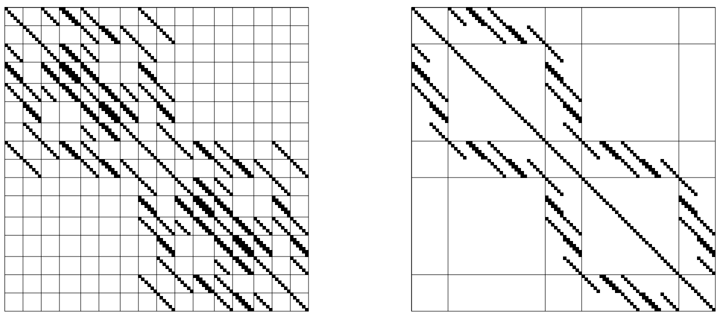
Fig. 1. Sparsity structure of the matrix A on the left and matrix B on the right, for the division of Ω into 2x2x6 hexahedrons. Non-zero elements are drawn with small squares.

applied to the matrix $\tilde{A} = A + \tilde{D}$. The diagonal perturbation $\tilde{D} = \tilde{D}(\xi) = \mathrm{diag}(\tilde{d}_1, \ldots \tilde{d}_N)$ is defined as follows:

$$\tilde{d}_i = \begin{cases} \xi a_{ii} & \text{if} \quad a_{ii} \geq 2w_i, \\ \xi^{1/2} a_{ii} & \text{otherwise,} \end{cases}$$

where $0 < \xi < 1$ is a constant and $w_i = -\sum_{j>i} a_{ij}$.

The idea of the parallel preconditioner is to apply the MIC(0) factorization on an auxiliary matrix B. The matrix B has a special block structure, which allows a scalable parallel implementation.

Following the standard FEM assembling procedure we write A in the form $A = \sum_{e \in \omega_h} L_e^T A_e L_e$, where A_e is the element stiffness matrix, L_e stands for the restriction mapping of the global vector of unknowns to the local one corresponding to the current element e. Let us consider the following approximation B_e of A_e:

$$A_e = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} & a_{16} \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} & a_{26} \\ a_{31} & a_{32} & a_{33} & a_{34} & a_{35} & a_{36} \\ a_{41} & a_{42} & a_{43} & a_{44} & a_{45} & a_{46} \\ a_{51} & a_{52} & a_{53} & a_{54} & a_{55} & a_{56} \\ a_{61} & a_{62} & a_{63} & a_{64} & a_{65} & a_{66} \end{bmatrix}, \quad B_e = \begin{bmatrix} b_{11} & a_{12} & a_{13} & a_{14} & a_{15} & a_{16} \\ a_{21} & b_{22} & a_{23} & a_{24} & a_{25} & a_{26} \\ a_{31} & a_{32} & b_{33} & 0 & 0 & 0 \\ a_{41} & a_{42} & 0 & b_{44} & 0 & 0 \\ a_{51} & a_{52} & 0 & 0 & b_{55} & 0 \\ a_{61} & a_{62} & 0 & 0 & 0 & b_{66} \end{bmatrix}.$$

The local numbering follows the pairs of the opposite nodes of the reference element. The diagonal entries of B_e are modified to hold the row-sum criteria. Assembling the locally defined matrices B_e we get the global matrix $B = \sum_{e \in \omega_h} L_e^T B_e L_e$.

The sparsity structure of the matrices A and B is illustrated by Fig. 1. Lexicographic node numbering is used. The important property of the matrix B is that its diagonal blocks are diagonal matrices. This allows a parallel implementation, see [1, 2].

4 Circulant Block-Factorization Preconditioning

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. Any circulant matrix can be factorized as $C = F\Lambda F^*$, where Λ is a diagonal matrix containing the eigenvalues of C, and F is the Fourier matrix $F = \frac{1}{\sqrt{m}}\left\{e^{2\pi\frac{jk}{m}\mathbf{i}}\right\}$, $0 \leq j,k \leq m-1$. Here $\mathbf{i}$ stands for the imaginary unit.

We use now the general form of the CBF preconditioning matrix M for the matrix $A^{(FDM)}$ by

$$M_{CBF} = \mathrm{tridiag}(C_{i,i-1}, C_{i,i}, C_{i,i+1}) \qquad i = 1, 2, \ldots n-1,$$

where $C_{i,j} = Block - Circulant(A_{i,j})$ is block-circulant approximation of the corresponding block $A_{i,j}$ [12, 13]. 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 [10, 11].

5 Numerical Tests

The numerical tests presented in this section illustrate the convergence rate as well as the parallel performance of the developed algorithms for 3D elliptic problems. We consider further test problems with variable coefficients in the form

$$\frac{\partial}{\partial x_1}\left[\left(1+\frac{\epsilon}{2}\sin\left(2\pi\left(x_1+x_3\right)\right)\right)\frac{\partial u}{\partial x_1}\right] + \frac{\partial}{\partial x_2}\left[\left(1+\frac{\epsilon}{2}\sin\left(2\pi\left(x_1+x_2\right)\right)\right)\frac{\partial u}{\partial x_2}\right] + \frac{\partial}{\partial x_3}\left[\left(1+\epsilon e^{x_1+x_2+x_3}\right)\frac{\partial u}{\partial x_3}\right] = f\left(x_1, x_2, x_3\right) \qquad (4)$$

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.

The right hand side f is chosen in such a way that the problem (4) has solution

$$u\left(x_1, x_2, x_3\right) = \sin 2\pi x_1 \sin 2\pi x_2 \sin 2\pi x_3.$$

The computations are done with double precision. The iteration stopping criterion is

$$||\mathbf{r}^{N_{it}}||_{M^{-1}}/||\mathbf{r}^0||_{M^{-1}} < 10^{-3},$$

where $\mathbf{r}^j$ stands for the residual at the jth iteration step of the preconditioned conjugate gradient method. The codes have been implemented in C/C++ and the parallelization has been facilitated using the MPI [15, 16] library. We report

Table 1. Number of iterations and execution times of the algorithms.

N		N_{it}		Error		Execution time T_1			
						CLX		SP5	
MIC(0)	CBF	MIC(0)	CBF	MIC(0)	CBF	MIC(0)	CBF	MIC(0)	CBF
$\epsilon = 0$									
101 376	32 768	13	5	3.29 E-3	3.02 E-3	0.55	0.21	0.23	0.16
338 688	110 592	15	5	1.52 E-3	1.37 E-3	1.29	1.06	0.80	1.10
798 720	262 144	17	5	8.89 E-4	7.78 E-4	5.50	1.62	2.16	1.31
2 681 856	884 736	21	4	4.64 E-4	3.50 E-4	14.79	7.17	9.45	7.79
6 340 608	2 097 152	21	4	3.54 E-4	1.98 E-4	55.29	11.36	23.38	11.06
21 344 256	7 077 888	25	4	2.30 E-4	8.83 E-5		57.94	90.69	71.20
$\epsilon = 0.1$									
101 376	32 768	19	19	5.09 E-3	3.08 E-3	0.79	0.60	0.33	0.43
338 688	110 592	22	21	2.71 E-3	1.40 E-3	1.85	3.69	1.15	3.71
798 720	262 144	29	24	1.62 E-3	7.96 E-4	9.17	6.51	3.61	4.69
2 681 856	884 736	38	32	1.06 E-3	3.58 E-4	26.21	45.02	16.75	47.23
6 340 608	2 097 152	42	37	5.56 E-4	2.02 E-4	108.07	79.48	45.69	72.94
21 344 256	7 077 888	59	48	6.27 E-4	9.04 E-5		547.16	209.27	655.31
$\epsilon = 1$									
101 376	32 768	14	23	1.19 E-2	3.53 E-3	0.59	0.72	0.25	0.34
338 688	110 592	19	30	5.53 E-3	1.60 E-3	1.61	5.14	1.00	4.27
798 720	262 144	28	36	3.13 E-3	9.10 E-4	8.86	9.18	3.49	5.01
2 681 856	884 736	31	47	1.67 E-3	4.09 E-4	21.51	65.25	13.74	59.92
6 340 608	2 097 152	41	58	1.15 E-3	2.31 E-4	105.56	122.69	44.63	93.93
21 344 256	7 077 888	64	80	8.41 E-4	1.03 E-4		882.02	226.71	1001.33

the results of the experiments executed on Linux clusters located in Bologna, Italy. In our experiments, times have been collected using the MPI provided timer and the best results from multiple runs are reported. We present the elapsed time T_p in seconds on p processors, the speed-up $S_p = T_1/T_p$, and the parallel efficiency $E_p = S_p/p$.

Table 1 contains the number of iterations and the execution times collected on two clusters: CLX and SP5. CLX is an IBM Linux Cluster 1350 made of 512 2-way IBM X335 nodes. Each computing node contains 2 Xeon Pentium IV processors running at 3 GHz and 2 GB of RAM. Nodes are interconnected via a Myrinet network with a maximum bandwidth of 256 Mb/s. We have used IBM Visual Age compiler with options "-align -tpp7 -O3 -xN". SP5 is an IBM SP Cluster 1600, made of 64 nodes p5–575 (see `http://www.ibm.com/servers/eserver/pseries/library/sp_books/`) interconnected with a pair of connections to the Federation HPS (High Performance Switch). Globally the machine has 512 IBM Power5 processors and 1.2 TB of memory. A p5–575 node contains 8 SMP processors Power5 at 1.9 GHz and have at least 16 GB of memory. The HPS interconnect is capable of a bandwidth of up to 2 GB/s unidirectional.

First two columns of Table 1 show the size of the discrete problem. Next two columns report the number of iterations, the maximal error of the obtained solution of (4) is shown in next columns. For Laplace equation (for $\epsilon = 0$) for

Table 2. Execution time for parameter $\epsilon = 1$

p	MIC(0)						CBF					
	n											
	32	48	64	96	128	192	33	49	65	97	129	193
IBM Linux Cluster 1350												
1	0.59	1.61	8.86	21.51	105.56		0.72	5.14	9.18	65.25	122.69	882.02
2	0.33	1.44	4.84	18.52	58.88		0.45	3.06	6.03	39.10	90.35	536.99
3								1.98		26.38		358.07
4	0.22	0.94	2.81	10.27	32.04	154.37	0.24	1.66	3.43	21.33	49.82	293.03
6								1.11		14.53		198.28
8	0.18	0.68	2.00	6.14	17.93	91.32	0.12	0.77	1.89	11.35	27.04	157.66
12								0.54		7.89		111.16
16	0.48	0.70	2.43	4.76	13.31	63.31	0.07	0.41	0.86	6.10	14.79	88.26
24								0.28		3.87		58.41
32	0.24	0.54	1.33	3.96	10.08	42.51	0.08		0.11	2.83	8.36	45.11
48		0.66	1.36	3.39	11.88	46.60		0.22		1.97		31.80
64			2.26	3.93	7.51	30.03			0.39		3.59	23.78
IBM SP Cluster 1600												
1	0.25	1.00	3.49	13.74	44.63	226.71	0.34	4.27	5.01	59.92	93.93	1001.33
2	0.16	0.52	1.82	7.16	23.94	121.03	0.18	2.11	2.27	28.71	38.97	438.48
3								1.44		19.11		273.96
4	0.10	0.36	0.98	4.03	13.06	63.31	0.09	1.07	1.12	14.27	18.97	196.92
6								0.71		9.41		131.47
8	0.07	2.70	0.72	2.30	7.77	42.05	0.05	0.56	0.62	7.20	9.83	119.34
12								0.60		5.75		80.12
16	1.27	2.56	0.49	1.31	4.15	21.12	0.38	0.35	0.47	4.40	6.66	61.20
24								0.69		2.99		41.58
32	0.10	0.20	0.40	0.99	2.85	13.65	0.33		0.30	2.36	3.55	31.52
48		2.50	3.97	1.34	2.26	11.31		0.16		1.62		21.63
64			2.05	0.86	1.97	10.27			0.19		2.09	16.64

similar mesh size used in FEM and FDM discretization the obtained accuracy has the same order but the FEM discretization leads to a system of linear equations with approximately three times more unknowns. This leads to the larger execution time of MIC(0) algorithm. For problems with strongly varying coefficients MIC(0) algorithm has a preference and regardless of larger size of the discrete problem the MIC(0) algorithm is faster.

Table 2 shows the parallel execution time for the problem with strongly varying coefficients (for parameter $\epsilon = 1$). The comparison shows that the MIC(0) algorithm is faster except for relatively large mesh size on large number of processors. This fact confirms our general expectations that the incomplete Cholesky factorization is robust preconditioner (in this case — for problems with strongly varying coefficients) and that the CBF algorithm is highly parallelizable.

Table 3 compare the parallel execution time per one PCG iteration. The comparison shows that the MIC(0) algorithm is faster except execution on more than 16 processors on CLX cluster. The speed-up obtained on both clusters

Table 3. Execution time per iteration of the algorithms.

n	T_1^{it}	T_2^{it}	T_3^{it}	T_4^{it}	T_6^{it}	T_8^{it}	T_{12}^{it}	T_{16}^{it}	T_{24}^{it}	T_{32}^{it}	T_{48}^{it}	T_{64}^{it}	T_{96}^{it}
MIC(0) algorithm, IBM Linux Cluster 1350													
32	0.039	0.022		0.015		0.012		0.032		0.016			
48	0.080	0.072		0.047		0.034		0.035		0.027	0.033		
64	0.305	0.167		0.097		0.069		0.084		0.046	0.047	0.078	
96	0.672	0.579		0.321		0.192		0.149		0.124	0.106	0.123	
128	2.513	1.402		0.763		0.427		0.317		0.240	0.283	0.179	
192				2.375		1.405		0.974		0.654	0.717	0.462	
MIC(0) algorithm, IBM SP Cluster 1600													
32	0.016	0.011		0.007		0.005		0.085		0.007			
48	0.049	0.026		0.018		0.135		0.128		0.010	0.125		
64	0.120	0.063		0.034		0.025		0.017		0.014	0.137	0.071	
96	0.429	0.224		0.126		0.072		0.041		0.031	0.042	0.027	
128	1.062	0.570		0.311		0.185		0.099		0.068	0.054	0.047	
192	3.487	1.862		0.974		0.647		0.325		0.210	0.174	0.158	
CBF algorithm, IBM Linux Cluster 1350													
33	0.023	0.018		0.009		0.004		0.003		0.003			
49	0.147	0.097	0.061	0.052	0.035	0.023	0.016	0.012	0.008		0.006		
65	0.211	0.160		0.091		0.048		0.021		0.012		0.009	
97	1.193	0.806	0.534	0.438	0.295	0.234	0.153	0.120	0.075	0.056	0.038		0.025
129	1.718	1.500		0.830		0.453		0.238		0.118		0.059	
193	9.750	6.514	4.233	3.529	2.388	1.913	1.244	0.985	0.654	0.517	0.364	0.243	0.174
CBF algorithm, IBM SP Cluster 1600													
33	0.013	0.007		0.003		0.002		0.001		0.001			
49	0.135	0.066	0.045	0.033	0.022	0.017	0.015	0.011	0.007	0.004	0.004		
65	0.130	0.058		0.029		0.015		0.011		0.007		0.004	
97	1.232	0.588	0.391	0.292	0.192	0.147	0.117	0.089	0.060	0.046	0.031		0.020
129	1.561	0.643		0.312		0.161		0.109		0.057		0.033	
193	12.149	5.377	3.356	2.381	1.608	1.462	0.980	0.748	0.507	0.383	0.262	0.201	0.141

is reported in Table 4. As it was expected, the parallel features of the CBF algorithm lead to higher speed-up. Moreover, a super-linear speed-up is observed on SP5 cluster. The main reasons for this fact can be related to splitting the entire problem into subproblems which helps memory management, in particular allows for better usage of cache memories of individual parallel processors.

6 Concluding Remarks and Future Works

In this paper we concerned with the numerical solution of 3D elliptic problems. After discretization, such problems reduce to the solution of linear systems. We use two preconditioners: the Modified Incomplete Cholesky factorization MIC(0) and the Circulant Block-Factorization (CBF) preconditioning. Presented numerical results show that the rate of convergence of the CBF preconditioner is the same as for ILU factorization. We reported on the parallel performance of the studied preconditioners applied to the PCG algorithm. The developed MPI

Table 4. Speed-up of the algorithms.

n	S_2	S_3	S_4	S_6	S_8	S_{12}	S_{16}	S_{24}	S_{32}	S_{48}	S_{64}	S_{96}
MIC(0) algorithm, IBM Linux Cluster 1350												
32	1.751		2.505		3.122		1.219		2.385			
48	1.111		1.702		2.350		2.292		2.902	2.432		
64	1.826		3.118		4.394		3.633		6.615	6.389	3.883	
96	1.160		2.091		3.489		4.500		5.389	6.325	5.446	
128	1.792		3.290		5.882		7.919		10.430	8.861	14.003	
192			1.000		1.690		2.438		3.627	3.312	5.131	
MIC(0) algorithm, IBM SP Cluster 1600												
32	1.450		2.143		2.885		0.190		2.239			
48	1.852		2.765		0.369		0.389		4.593	0.397		
64	1.886		3.475		4.718		6.945		8.485	0.876	1.677	
96	1.913		3.396		5.893		10.364		13.588	10.029	15.355	
128	1.862		3.410		5.719		10.725		15.526	19.526	22.204	
192	1.872		3.579		5.386		10.720		16.566	20.040	21.983	
CBF algorithm, IBM Linux Cluster 1350												
33	1.33		2.73		5.88		9.16		8.37			
49	1.52	2.40	2.82	4.26	6.28	9.11	12.00	17.52		22.97		
65	1.32		2.32		4.36		9.96		17.95		24.76	
97	1.48	2.23	2.73	4.04	5.10	7.79	9.93	16.01	21.32	31.25		47.48
129	1.15		2.07		3.80		7.22		14.56		29.35	
193	1.50	2.30	2.76	4.08	5.10	7.84	9.90	14.90	18.86	26.80	40.17	55.96
CBF algorithm, IBM SP Cluster 1600												
33	1.94		3.80		7.24		8.73		13.76			
49	2.03	3.02	4.09	6.12	7.97	9.27	12.75	18.11	35.74	32.65		
65	2.23		4.52		8.59		11.52		18.88		31.60	
97	2.09	3.15	4.22	6.43	8.41	10.52	13.84	20.53	26.88	39.38		63.06
129	2.43		5.01		9.71		14.36		27.41		47.82	
193	2.26	3.62	5.10	7.56	8.31	12.39	16.24	23.95	31.68	46.33	60.49	85.93

codes provide new effective tool for solving of large-scale problems in realistic time on a coarse-grain parallel computer systems.

The experimental data collected from two clusters is only preliminary. In the near future we plan, first, to complete the performance studies by running our code on a number of additional machines. Second, we will extend our work to non-uniformly shaped domains, non-uniform discretization as well as situations when the proposed approach is embedded in a solver for non-linear problems. The implementation of the possibility for overlapping of computations and communications (see, e.g. [2, 7]) will be analyzed at the next step in development of the parallel code.

Acknowledgments. This research was partially supported by grant I-1402/2004 from the Bulgarian NSF and by the project BIS-21++ funded by FP6 INCO grant 016639/2005. The parallel numerical tests were supported via the EC Project HPC-EUROPA RII3-CT-2003-506079.

References

1. P. Arbenz, S. Margenov, Parallel MIC(0) preconditioning of 3D nonconforming FEM systems, *Iterative Methods, Preconditioning and Numerical PDEs*, Proceedings of IMET Conference, Prague, 2004, 12-15.
2. P. Arbenz, S. Margenov, Y. Vutov, Parallel MIC(0) Preconditioning of 3D Elliptic Problems Discretized by Rannacher–Turek Finite Elements, *Comput. Math. Appl.* (to appear)
3. D. N. Arnold, F. Brezzi, Mixed and nonconforming finite element methods: implementation, postprocessing and error estimates, *RAIRO Model. Math. Anal. Numer.*, **19**, 1985, 7–32.
4. O. Axelsson, *Iterative solution methods*, Cambridge University Press, Cambridge, 1994.
5. A. O. H. Axelsson, M.G. Neytcheva, *Supercomputers and numerical linear algebra*, KUN, Nijmegen, 1997.
6. G. Bencheva, S. Margenov, Parallel incomplete factorization preconditioning of rotated linear FEM systems, *J. Comput. Appl. Mech.*, **4** (2), 2003, 105–117.
7. G. Bencheva, S. Margenov, J. Starý. Parallel PCG Solver for Nonconforming FE Problems: Overlapping of Communications and Computations. *Large-Scale Scientific Computing*, I. Lirkov, S. Margenov, and J. Waśniewski eds., *Lecture notes in computer sciences*, **3743**, Springer Verlag, 646–654, 2005.
8. R. Blaheta, Displacement decomposition—incomplete factorization preconditioning techniques for linear elasticity problems, *Numer. Linear Algebra Appl.*, **1**, 1994, 107–126.
9. D. Braess, *Finite elements. Theory, fast solvers, and applications in solid mechanics*, Cambridge University Press, 1997.
10. I. Lirkov, 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, P. V. Vassilevski, eds., World Scientific, Singapore, 1999, 482–490.
11. I. Lirkov, S. Margenov, M. Paprzycki, Parallel performance of a 3D elliptic solver, *Numerical Analysis and Its Applications II*, L. Vulkov, J. Waśniewski, P. Yalamov eds., *Lecture Notes in Computer Sciences*, **1988**, Springer Verlag, 2001, 535–543.
12. I. Lirkov, Y. Vutov, Parallel Performance of a 3D Elliptic Solver, *Proceedings of the International Multiconference on Computer Science and Information Technology*, Volume 1, M. Ganzha, M. Paprzycki, J. Wachowicz, K. Wecel eds., (CD-ROM), ISSN 1896–7094, 2006, 579–590.
13. I. Lirkov, Y. Vutov, The Convergence Rate and Parallel Performance of a 3D Elliptic Solver, *System Science*, to appear.
14. R. Rannacher, S. Turek, Simple nonconforming quadrilateral Stokes Element, *Numer. Methods Partial Differential Equations*, **8** (2), 1992, 97–112.
15. M. Snir, St. Otto, St. Huss-Lederman, D. Walker, J. Dongara, *MPI: The Complete Reference*, Scientific and engineering computation series, The MIT Press, Cambridge, Massachusetts, 1997, Second printing.
16. D. Walker and J. Dongara, MPI: a standard Message Passing Interface, *Supercomputer*, **63**, 1996, 56–68.