On Parallel Computational Technologies of Augmented Domain Decomposition Methods

Y.L Gurieva[1] and V.P Il'in[1,2(✉)]

[1] Institute of Computational Mathematics and Mathematical Geophysics SB RAS, Novosibirsk, Russia
yana@lapasrv.sscc.ru
[2] Novosibirsk State University, Novosibirsk, Russia
ilin@sscc.ru

Abstract. The performance of the parallel domain decomposition methods (DDM) for solving very large systems of linear algebraic equations with non-symmetric sparse matrices depends on the convergence of the iterative algorithms as well as on the efficiency of the computational technologies. Usually in DDM approach the number of iterations grows together with a growth of the degree of freedom. We consider the algorithms for increasing the convergence rate based on the preconditioning with using deflation and aggregation techniques which take low rank approximations of the original systems of linear algebraic equations. The efficiency of the proposed approaches is demonstrated on the representative set of model tasks.

1 Introduction

In general, the modern domain decomposition methods to solve very large systems of linear equations (SLAEs), which arise in the discrete approximation on the non-structured meshes of the multi-dimensional boundary value problems (BVPs) by the finite element or by other grid methods, can be presented by three main mathematical approaches: external Krylov's type iterative process "on subdomains" which presents the additive Schwarz (or special block Jacobi) algorithm, simultaneous solving the auxiliary BVPs in the subdomains which can be carried out by a direct or an iterative algorithm, and preconditioning procedures to accelerate the external iterations, see [1–5] and the literature cited there.

The last factor is very important for strongly scalable parallelized tasks, because for a very large number of subdomains and corresponding block degree of freedom, one can observe the considerable stagnation of the iterative process. In recent decades, various versions of aggregation, deflation, and coarse grid correction accelerators have been investigated and applied successfully by many authors. The main goal of our paper consists namely in the numerical analysis of

The work is supported partially by Russian Science Foundation grant N 14-11-00485.
The experimental part of the paper is supported by the RFBR grant N 14-07-00128.

V. Malyshkin (Ed.): PaCT 2015, LNCS 9251, pp. 35–46, 2015.
DOI: 10.1007/978-3-319-21909-7_4

several versions of aggregation accelerating based on low rank matrix approximations in different coarse subspaces. The program implemtation of the algorithms is realized for the universal compressed sparse matrix format which is necessary to solve the practical problems.

The conventional parallel technologies of DDMs include two levels: application of MPI processes for corresponding subdomains, including interface communications between them at each outer (external) iteration, and implementation of the multi-thread computing for the "internal" parallelezation on the multi-core processors.

The problems are specified by three levels of the degrees of freedom: the number of unknowns of SLAE (10^8-10^{11}), the quantity of subdomains (10^2-10^5, block dimension of the broblem), and coarse grid dimension ($10-10^3$) which determine the scalability of the parallelism of the general computational process.

The bottleneck of DDM approach is in a minimization of a communication time. It can be done by simultaneous data transfer and synchronized computations in the subdomains. In general, DDM performance depends on the convergence properties of the iterative algorithms and on the efficiency of the computational technologies whose variants are discussed later on. In Sect. 2, we describe the algebraic and structured representation of the multi-level preconditioned iterative processes in the Krylov subspaces. Section 3 is devoted to the mapping of the parallel algorithms under consideration onto the computational multi-processor system with the distributed and shared memory architecture. The results of numerical experiments and an analysis of the various approaches is carried out for different orders of the basic interpolation functions and for different placement of the coarse grid nodes. The efficiency of the proposed algorithms is demonstrated on the representative set of the model examples.

2 Statement of the Problem and Algorithms

Let us consider a SLAE

$$Au = \sum_{l' \in \omega_l} a_{l,l'} u_{l'} = f, \;\; A = \{a_{l,l'}\} \in \mathcal{R}^{N,N}, \;\; u = \{u_l\}, \;\; f = \{f_l\} \in \mathcal{R}^N, \tag{1}$$

with the sparse matrix of large order with real entries arising from some discrete approximation of a multi-dimensional BVP by the finite element or the finite volume or other grid methods; ω_l means a set of the indices of off-diagonal entries in the l-th row of matrix A.

We can divide the total set of the vector indices $\Omega = \{l\}$ into P non-intersected subsets, or algebraic subdomains,

$$\Omega = \bigcup_{s=1}^{P} \Omega_s, \;\; N = \sum_{s=1}^{N} N_s, \tag{2}$$

each containing approximately equal number of elements N_s. For subdomains Ω_s, let us denote their boundaries Γ_s^0 and closures as the following:

$$\Gamma_s \equiv \Gamma_s^0 = \{l' \in \omega_l, \;\; l \in \Omega_s, \;\; l' \notin \Omega_s\}, \;\; \bar{\Omega}_s^0 = \Omega_s \bigcup \Gamma_s^0. \tag{3}$$

Also, we can define the boundary layers of Ω_s:

$$\Gamma_s^t = \left\{ l' \in \omega_l,\ \ l \in \bar{\Omega}_s^{t-1},\ \ \bar{\Omega}_s^t = \bar{\Omega}_s^{t-1} \bigcup \Gamma_s^t,\ \ t = 1, 2, ..., \Delta_s \right\}. \tag{4}$$

Parameter Δ_s presents the measure of an extension of the subdomain Ω_s. The set of $\bar{\Omega}_s^{\Delta_s}$ forms the algebraic decomposition of the original domain Ω into subdomains with parametrized overlapping. Hystorically, it is known that increasing of the overlapping yields the increasing of the iterative convergence of DDMs and increasing of the cost of each iteration. For the subvectors

$$\bar{u}_s = \{u_l, l \in \bar{\Omega}_s^{\Delta_s}\} \in \mathcal{R}^{\bar{N}_s},\ \ u = \bigcup_{s=1}^{P} \bar{u}_s,$$

the original system can be written in a block form

$$A_{s,s}\bar{u}_s + \sum_{s' \in Q_s} A_{s,s'}\bar{u}_{s'} = f_s,\ \ s = 1, ..., P, \tag{5}$$

where Q_s is the set of subdomains which are adjacent to the extended subdomain $\bar{\Omega}_s^{\Delta_s}$.

To solve (5), the generalized block Jacobi iterative process is used:

$$\bar{B}_s(\bar{u}_s^{n+1} - \bar{u}_s^n) = \bar{f}_s - (\bar{A}\bar{u}^n)_s \equiv \bar{r}_s^n,\ \ \bar{u}_s^n \in \mathcal{R}^{\bar{N}_s}. \tag{6}$$

Here $\bar{r}_s^n$ is the residual subvector and $\bar{B}_s$ is some preconditioning matrix which takes into account the permutations of the "boundary" rows $l \in \Gamma_s^{\Delta_s}$, because of using special interface conditions of Steklov-Poincare type between the neighbour subdomains in the Schwarz iterations, see [2–4] for details.

The vector u^n of the sought for solution of original SLAEs (1) is not defined uniquely in (6), because in the intersections of the neighbour subdomain $\bar{\Omega}_s^{\Delta_s}$ we have several values of the vector components for the various s. In order to avoid such an indefiniteness, different approaches are used. We apply the restricted alternating Schwarz (RAS) slgorithm, which is based on using the restricting operators $R_s \in \mathcal{R}^{N_s, \bar{N}_s}$:

$$u_s^n = R_s\bar{u}_s^n = \{u_l^n = (R_s\bar{u}_s^n)_l,\ \ l \in \Omega_s\} \in \mathcal{R}^{N_s}, \tag{7}$$

where the subdomains $\Omega_s, s = 1, ..., P$, define the domain decomposition without overlapping.

The RAS Jacobi type method can be written in the following form:

$$\begin{aligned} &u^{n+1} = u^n + B_{ras}^{-1} r^n, \\ &B_{ras}^{-1} = R\hat{A}^{-1}W^T,\ \ \hat{A} = W^T A W = \text{ block-diag } \{A_{s,s} \in \mathcal{R}^{\bar{N}_s, \bar{N}_s}\}, \end{aligned} \tag{8}$$

$W = [w_1 ... w_P] \in \mathcal{R}^{N,P}$ is a rectangular matrix, each its column w_s has the entries equal to one in the nodes from $\bar{\Omega}_s$ and has zero entries otherwise. Let us note that generally even if the original SLAE is symmetric, a preconditioning

matrix B_{ras} from (8) is not a symmetric one. In addition, the inversion of the blocks $A_{s,s}$ of the matrix $\hat{A}$ is actually reduced to the simultaneous solution of independent subsystems in the corresponding subdomains.

We suppose that SLAE (1) is obtained from the approximation of a multi-dimensional BVP for partial differential equations by the finite element, finite volume or other method on some non-structured grid. For example, let the Dirichlet problem for the diffusion-convection equation

$$\begin{array}{l} -\dfrac{\partial^2 u}{\partial x^2} - \dfrac{\partial^2 u}{\partial y^2} + p\dfrac{\partial u}{\partial x} + q\dfrac{\partial u}{\partial y} = f(x,y), \quad (x,y) \in \Omega, \\ u|_\Gamma = g(x,y), \end{array} \tag{9}$$

be solved in a computational domain $\Omega = (a_x, b_x) \times (a_y, b_y)$, where Γ is a boundary of Ω, and the convection coefficients p, q are, for simplicity, the given values. For the sake of brevity, we will use the symbol Ω to denote either the computational domain or the grid domain according to the context.

The given boundary value problem is approximated on a uniform grid

$$\begin{array}{l} x_i = a_x + ih_x, \quad y_j = a_y + jh_y, \\ i = 0, 1, ..., N_x + 1; \quad j = 0, 1, ..., N_y + 1; \\ h_x = (b_x - a_x)/(N_x + 1), \quad h_y = (b_y - a_y)/(N_y + 1), \end{array} \tag{10}$$

by a five-point scheme of the form

$$(Au)_l = u_{l,l}u_l + a_{l,l-1}u_{l-1} + a_{l,l+1}u_{l+1} + a_{l,l-N_x}u_{l-N_x} + a_{l,l+N_x}u_{l+N_x} = f_l, \tag{11}$$

where l is a "global", or natural, number of an inner grid node:

$$l = l(i,j) \equiv i + (j-1)N_x = 1, ..., N = N_x N_y. \tag{12}$$

A particular view of the coefficients $a_{l,l'}$ in (11) can be different, and specific formulae can be found in [6,7]. Eq. (11) are written for the inner grid nodes, moreover, for the nodes near the boundary, whose numbers are from a set of the indices $i = 1, N_x$ or $j = 1, N_y$, the values known from the boundary conditions of the solution are substituted into the corresponding equations and moved to their right-hand sides, so that the corresponding coefficients $a_{l,l'}$ in (11) equal zero (it is the so- called "constraining" procedure).

We can think of the isomorphism between the vector entries in (1)–(5) and the grid nodes: u_l is the value of the grid function u in the l-th node at the grid Ω which is a set of all nodes in the computational domain. The subdomains Ω_s in (2) can be redefined as the grid subdomains, and for the model problem (9) we present a simple decomposition of Ω into a union of an identical non-interesecting rectangle subdomains:

$$\Omega = \bigcup_{s=1}^{P} \Omega_s, \quad P = P_x P_y,$$

each containing an equal number of the grid nodes

$$M = m_x m_y, \quad N_x = P_x m_x, \quad N_y = P_y m_y, \quad N = PM.$$

One can find that the subdomains form a two-dimensional macrogrid, where each macrovertex can be numbered by a pair of indices p, q (similarly to the grid node indices i, j), and a "continuous" number of a subdomain is defined as

$$\begin{aligned} &s = s(p,q) \equiv p + (q-1)P_x = 1, ..., P, \\ &p = 1, ..., P_x; \;\; q = 1, ..., P_y. \end{aligned} \tag{13}$$

We now turn from continuous numbering of nodes to their subdomain-by-subdomain ordering: at first, we number all the nodes in Ω_1, then in Ω_2, etc. The vector components u, f are ordered correspondingly, so that the SLAE (11) takes the block-matrix form (5), where $\bar{u}_s \in \mathcal{R}^{N_s}$ means a subvector of the vector u, whose components correspond to the nodes from the grid subdomain Ω_s, and Q_s means a set of the numbers of the grid subdomains adjacent to subdomain Ω_s. Hereinafter we assume that a local node ordering in every subdomain is a natural one: local pairs of indices $i' = 1, ..., m_x;\; j' = 1, ..., m_y$ are introduced and a continuous number is determined by the formula $l' = i' + (j'-1)m_x$ similar to (12).

The rate of convergence of the iterative process (8) depends on the number of the subdomains, or more precisely, on the diameter of a graph representing a macrogrid formed by the decomposition. This can be clearly explained by the fact that on a single iteration the solution perturbation in one subdomain is transmitted only to the neighbouring, or adjacent, subdomains. To speed up the iterative process, it is natural to use not only the nearest but also the remote subdomain couplings at every step. For this purpose, different approaches are used in decomposition algorithms: deflation, coarse grid correction, aggregation, etc., which to some extent are close to the multigrid principle as well as the low-rank approximations of matrices, see numerous publications cited at a special site [8].

We will consider the following approach based on an interpolation principle. Let Ω_c be a coarse grid with the number of nodes $N_c \ll N$ in the computational domain Ω, moreover, the nodes of the original grid and the coarse grid may not match.

Let us denote by $\varphi_1, ..., \varphi_{N_c}$ a set of the basis interpolating polynomials of order M_c on the grid Ω_c which are supposed to have a finite support and without loss of generality form an expansion of the unit, i.e.

$$\sum_{k=1}^{N_c} \varphi_k(x,y) = 1.$$

Then a sought for solution vector of SLAE (1) can be presented in the form of an expansion in terms of the given basis:

$$u = \{u_{i,j} \approx u^c_{i,j} = \sum_{k=1}^{N_c} c_k \varphi_k(x_i, y_j)\} = \Phi\hat{u} + \psi, \tag{14}$$

where $\hat{u} = \{c_k\} \in \mathcal{R}^{N_c}$ is a vector of the coefficients of the expansion in terms of the basis functions, ψ is an approximation error, and $\Phi = [\varphi_1 ... \varphi_{N_c}] \in \mathcal{R}^{N,N_c}$ is

a rectangular matrix with every k-th column consisting of the values of the basis function $\varphi_k(x_i, y_j)$ at N nodes of the original grid Ω (most of the entris of Φ equal zero in virtue of the finiteness of the basis). The columns, or the functions φ_k, can be treated to be the orthonormal ones but not necessarily. If at some k-th node P_k of the coarse grid Ω_c only one basis function is a nonzero one $(\varphi_k(P_{k'}) = \delta_{k,k'})$, then $\hat{u}_k = c_k$ is the exact value of the sought for solution at the node P_k. With a substitution of (14) into the original SLAE, one can obtain the system

$$A\Phi\hat{u} = f - A\psi, \tag{15}$$

and if to multiply it by Φ^T one can obtain

$$\hat{A}\hat{u} \equiv \Phi^T A\Phi\hat{u} = \Phi^T f - \Phi^T A\psi \equiv \hat{f} \in \mathcal{R}^{N_c}. \tag{16}$$

Assuming further that the error ψ in (14) is sufficiently small and omitting it, one can obtain a system for an approximate coarse grid solution $\check{u}$:

$$\hat{A}\check{u} = \Phi^T f \equiv \check{f}. \tag{17}$$

If the matrix A is a non-singular matrix and Φ is the full-rank matrix(the rank is much less than N), we assume these facts to hold further, then from (16) we have

$$u \approx \tilde{u} = \Phi\check{u} = \Phi\hat{A}^{-1}\hat{f} = B_c^{-1}f,\ B_c^{-1} = \Phi(\Phi^T A\Phi)^{-1}\Phi^T.$$

For the error of the approximate solution we have

$$u - \tilde{u} = (A^{-1} - B_c^{-1})f. \tag{18}$$

The error of the approximate solution can also be presented via the error of the approximation ψ. Subtracting Eqs. (16) and (17) term by term we have

$$\hat{A}(\hat{u} - \check{u}) = -\Phi^T A\psi$$

what yields the required equation:

$$u - \tilde{u} = \Phi\hat{u} + \psi - \Phi\check{u} = \psi - B_c^{-1}A\psi.$$

The matrix B_c^{-1} introduced above can be regarded as a low rank approximation of the matrix A^{-1} and used as a preconditioner to build an iterative process. In particular, for an arbitrary vector u^{-1} we can choose an initial guess as

$$u^0 = u^{-1} + B_c^{-1}r^{-1},\ \ r^{-1} = f - Au^{-1}. \tag{19}$$

In doing so, the corresponding initial residual $r^0 = f - Au^0$ will be orthogonal to a coarse grid subspace

$$\hat{\Phi} = \text{ span } \{\varphi_1, ..., \varphi_{N_c}\} \tag{20}$$

in the sense of fulfilling the condition

$$\Phi^T r^0 = \Phi^T (r^{-1} - A\Phi \hat{A}^{-1} \Phi^T r^{-1}) = 0. \tag{21}$$

The relations given in [10] are the basis for the conjugate gradient method with deflation, wherein an initial direction vector is chosen by the formula

$$p^0 = (I - B_c^{-1} A) r^0, \tag{22}$$

which ensures that the following orthogonality condition holds:

$$\Phi^T A p^0 = 0. \tag{23}$$

Further iterations are implemented using the following relations:

$$\begin{array}{l} u^{n+1} = u^n + \alpha_n p^n, \;\; r^{n+1} = r^n - \alpha_n A p^n, \\ p^{n+1} = r^{n+1} + \beta_n p^n - B_c^{-1} A r^{n+1}, \\ \alpha_n = (r^n, r^n)/(p^n, Ap^n), \;\; \beta_n = (r^{n+1}, r^{n+1})/(r^n, r^n). \end{array} \tag{24}$$

In this method, which we will refer to as DCG, at every step the following relations hold:

$$\Phi^T r^{n+1} = 0, \;\; \Phi^T A p^{n+1} = 0. \tag{25}$$

If now we turn back to the additive Schwarz method (11), we can try to accelerate it by the coarse grid preconditioner B_c^{-1} (in addition to the preconditioner B_{ras}^{-1}). We will consider this point in a more general formulation assuming that matrix A is a non-symmetric one and that there are several but not only two preconditioning matrices. Moreover, the preconditioners can change from iteration to iteration what corresponds to the so-called dynamic, or flexible, preconditioning.

The SLAE with the non-symmetric matrix A is solved by the well-known BiCGStab algorithm [1].

3 Parallel Technologies of DDM

The objectives of our research consist in the verification, testing, and a comparative analysis of the efficiency of different algorithms and computational technologies of solving big sparse SLAEs aimed at their optimization and including into the KRYLOV library [9] of the parallel algebraic solvers. The main requirements to develop a proper software are high and scalable performance and no formal restrictions on the orders of the SLAEs and on the number of the processors and computational cores used. According to [3], a strong and a weak scalability can be distinguished. The first one describes a decrease in the execution time of one big problem with an increase of the number of computing devices, while the second one stands for approximate preservation of the solution time while increasing the dimension (the number of degrees of freedom) of the problem and the number of processors and/or cores.

The algorithms were coded with taking into account the architecture of the SSCC SB RAS cluster [11] (where KRYLOV library is available) but without GPGPU usage as their effective utilization in the considered domain decomposition methods has its own technological and computational complexity and requires a special study.

Computations are carried out in the following natural way: if a computational domain is divided into P subdomains than the solution is performed on $P+1$ MPI-processes (one is the root process and other ones correspond to their own subdomains). During the program execution, the root MPI-process is used to accumulate partial dot products from the subdomains thus also keeping the synchronization of the computational work in the subdomains and upon completion it accumulates the whole sought for solution vector.

A scalable parallelization of the algorithms is provided by synchronization of the calculations in subdomains and by a minimization of the time losses during interprocessors communication. The solutions to auxiliary algebraic subsystems in the subdomains are obtained simultaneously on the multicore CPUs with the usage of multithread OpenMP calculations. The reduced system 17 is formed and solved in all the processes.

As algorithms from KRYLOV library are designed to solve large sparse SLAEs arising from an approximation of multidimensional boundary value problems on non-structured grids, the well-known compressed sparse row format of the matrix storage is used to keep the non-zero matrix entries. The global matrix A is formed in the root MPI-process (in the simplest implementation) at the preliminary stage, and then the distributed storage of the block rows $\bar{A}_s = \{A_{s,s'}, s' \in Q_s\}$ from (5) is done for the s-th extended subdomain (i.e., on the corresponding MPI-processes). If the original matrix is very big, it can be stored in a row-blocked form already and then the block rows be distributed among computational nodes (subdomains) thus keeping the global matrix on the root MPI-process is not a bottleneck for the problem under consideration.

An important condition for the high performance computing consists in the matching the arithmetic calculations and data communications between the subdomains by using MPI unblocked send-receive means. Moreover, the volume of the data transfer is very small as only the short vectors corresponding to the number of grid ponits on mutual boundary between the subdomains should be exchasnged.

Let us note that for the examined grid boundary value problems, a two-dimensional balanced domain decomposition into subdomains is considered, when for an approximately equal number of nodes $N_S \approx N/P$ in every subdomain the macrogrid daimeter d (for a macrogrid composed of subdomains) is equal, approximately, to $\sqrt{P}$. As the number of the iterations of the additive Schwarz method even with the usage of the preconditioned Krylov methods is proportional to $d^\gamma, \gamma > 0$, this yields a significant advantage over a one-dimensional decomposition for which $d \approx P$.

A solution to the isolated SLAEs in Ω_s is produced by the direct or iterative method requiring $(N/P)^{\gamma_1}, \gamma_1 > 0$ operations at every step of the two-level

process. As it is necessary to exchange the data corresponding to peripheral nodes of the adjacent subdomains only, the volume of such an information is much less and proportional to $(N/P)^{\gamma_1/2}$ (for two-dimensional BVPs) thus allowing one to carry out arithmetic and communication operations simultaneously.

A high performance of the code based on the presented approach is ensured by an active usage of the standard functions and vector-matrix operations from BLAS and Sparse BLAS included into Intel MKL [12].

4 Results of Numerical Experiments

We present the results of methodical experiments on solving five-point SLAEs for 2D Dirichlet problem in the unit square computational domain on the square grids with the number of nodes $N = 128^2$ and 256^2. Calculations were carried out via $P = 2^2, 4^2, 8^2$ MPI-processes each of which corresponded to the subdomains forming the square macrogrid. Iterations over the subdomains were realised with the help of BiCGStab algorithm [1] with the stopping criterion $||r^n||_2 \leq \varepsilon||f||_2$, $\varepsilon = 10^{-8}$. Solving of the auxiliary subdomain subsystems was carried out by the direct solver PARDISO from Intel MKL. The most time-consuming part of LU matrix decomposition was done only once before the iterations.

In the Table 1, each cell contains the numbers of iterations over the subdomains and the times of SLAEs solving (in seconds) on the grids 128^2 and 256^2. The upper figure in each cell corresponds to the zero convection coefficients while the bottom figure – to the convection coefficients $p = q = 4$). Domain decompositions were made for equal overlapping parameters in subdomains: $\Delta_s = \Delta = 0, 1, 2, 3, 4, 5$. Interface boundary conditions of the Dirichlet type between the adjacent subdomains were used in all experiments.

The results demonstrate that with Δ increasing up to 5, the number of the iterations reduces 3 - 4 fold, but when the overlapping value is big, the time of a subdomain solving begins to increase. So, for almost all the grids and the numbers of MPI-processes (subdomains), the optimal Δ value is approximately 3 – 4 in terms of the total execution time. If the convection coefficients p, q are nonzero ones, the number of the iterations increases by approximately 30–50 %. Let us notice that the figures for 4 and 16 subdomains were obtained in the experiments when each MPI-process was ran on its own cluster node in exclusive mode while the data for 64 subdomains were got in a series of experiments on cluster nodes that were given to the tasks in non-exclusive mode yielding some increase of the execution time. So the last line of the Table does not present "pure" speedup of the algorithm.

In the Tables below, for the sake of bravity, the results for the Poisson equation are presented, i.e. when there are no convection coefficients in equation (1). The experiments shown that with the moderate values of p, q ($|p| + |q| < 50$) the behavior of the iterative process varied slightly.

The numerous results for the different model and practical problems shown that the behavior of iterations varied slightly in the considered algorithms when

Table 1. The numbers of iterations and the solution times (in seconds) on the grids 128^2 and 256^2 for different overlapping parameter Δ

P	q	$N \setminus \Delta$	0	1	2	3	4	5
	0		18 2.17	11 1.74	9 1.64	7 1.53	7 1.48	6 1.42
4	4	128^2	31 2.85	17 2.10	13 1.87	12 1.81	11 1.74	10 1.74
4	0	256^2	27 8.34	16 5.38	12 4.21	10 3.68	9 3.33	8 2.93
	4		61 16.88	25 7.74	19 6.52	17 5.47	15 5.28	13 4.25
	0		32 1.46	18 1.29	14 1.25	12 1.17	11 1.03	9 0.98
16	4	128^2	41 1.60	25 1.40	19 1.31	16 1.18	14 1.17	14 1.10
16	0	256^2	40 3.23	24 2.23	20 1.97	17 1.77	14 1.27	14 1.24
	4		58 4.32	35 2.83	28 2.46	22 1.98	19 1.62	18 1.52
	0		43 1.56	26 1.66	19 1.39	16 1.50	14 1.56	12 0.86
64	4	128^2	57 2.02	34 1.91	26 1.78	21 1.98	20 1.69	18 1.35
64	0	256^2	60 4.75	36 4.16	27 3.35	22 3.11	20 3.00	18 4.66
	4		87 7.04	47 5.61	38 4.89	31 4.13	28 4.02	25 4.48

the initial error varied. The experiments given above were hold for the initial guess $u^0 = 0$ and the exact SLAE solution $u = 1$.

Table 2 shows the effect of applying of two deflation methods when the conjugate gradient algorithm without any additional preconditioning and without additive Schwarz method is used for three square grids with different numbers of nodes N and for different macrogrids with the number of the macronodes N_c. The macronodes are taken in the vicinity of the subdomain corners, i.e. when $P = 2^2, 4^2, 8^2$ the numbers of the macronodes, or the values of N_c, are $3^2, 5^2$ and 9^2 respectively. The basis functions $\phi_k(x, y)$ were the bilinear finite functions. Three right columns have the number of the iterations (the upper figures in every cell) for the single orthogonalization of the form (23) while the iteration number for the orthogonalization (25) on every iteration is the bottom figure. If to compare these data with the algorithm when the deflation is not used at all (the column with $P = 0$, i.e. no macrogrid is used) one can see the acceleration up to three times when P increases. However, it should be taken into account that an implementation of the multiple orthogonalization makes each iteration more expensive, so an additional investigation is required to optimize the algirithms on practice.

The results from Table 3 present the same data but when using the additive Schwarz method with the domain decomposition into P subdomains. The numbers of the coarse grid nodes are taken the same as that of the macronodes for Table 2. The basis functions $\phi_k(x, y)$ as in the previous series of experiments from Table 2 were the bilinear finite ones. Every cell of Table 3 contains the numbers of the iterations carried out without deflation (the upper figures in each cell) and the numbers of the iterations for the single orthogonalization of the initial guess (the bottom figures in each cell). In every cell, the first column gives the

Table 2. The deflation influence in the conjugate gradient method without additive Schwarz

$N \setminus P$	0	2^2	4^2	8^2
	176	167	166	103
64^2		118	87	56
	338	309	255	181
128^2		220	159	104
	609	544	442	276
256^2		376	294	190

Table 3. Aggregation influence in the additive Schwarz method (decomposition with different overlapping parameter Δ)

$N \setminus P$	2^2			4^2			8^2		
	19	11	8	26	15	12	37	20	15
64^2	23	9	7	21	12	9	28	15	11
	29	15	11	35	22	17	51	31	21
128^2	24	14	10	26	16	12	36	21	15
	38	21	17	53	31	23	71	43	32
256^2	31	18	15	35	21	17	40	26	21

data for the zero value of the overlapping parameter Δ, the second column – for $\Delta = 1$, and the third column – for $\Delta = 2$.

The presented results for the considered grids and macrogrids have approximately the same character as in Table 2 when the increasing of the deflation space yields to the decreasing of the iteration number together with the increasing of the amount of computations at each step. In these experiments, the outer iterations were carried out by the BiCGStab method.

Let us note that the experiments for Table 3 were hold for the initial guess $u^0 = 0$ and the exact SLAE solution $u(x_i, y_j) = x_i^2 - y_j^2$. Naturally, the efficiency of the considered "interpolation" deflation depends on the behaviour of the solution sought for. For example, if it is, e.g., $u(x_i, y_j) = x - y$, then the usage of the bilinear basis functions $\varphi_k(x, y)$ for $N_c \geq 4$ yields to the convergence in one iteration, and this fact was confirmed in the experiments.

5 Conclusion

We have studied experimentally the efficiency and the performance of several advanced approaches for domain decomposition methods. The results presented demonstrate a considerable increasing of the convergence rate of the iterative process when the corresponding overlapping parameters and a coarse grid correction are used to accelerate the additive Schwarz algorithm. It should be

mentioned that the augmented versions of DDM have been implemented without additional communication losses. Obviously, the further research should be held to obtain some practical recommendations to optimize the performance when a combination of various parametrized approaches are used simultaneously. The efficient code optimization for multi-GPGPU and a multi-core implementation is a challenge technology but it is an open question now. For example, it is interesting to analyse two-level iterative FGMRES procedure with some dynamic stopping criterion in subdomains and various basis functions in low rank matrix approximations.

References

1. Saad, Y.: Iterative Methods for Sparse Linear Systems. PWS Publications, New York (2002)
2. Toselli, A., Widlund, O.: Domain Decomposition Methods - Algorithms and Theory. Springer, Heidelberg (2005)
3. Chapman, A., Saad, Y.: Deflated and augmented krylov subspace technique. Numer. Linear Algebra Applic. **4**(1), 43–66 (1997)
4. Il'in, V.P.: Parallel Methods and Technologies of Domain Decomposition (in Russian). Vestnik YuUrGU. Series Computational mathematics and informatics. 46(305), 31–44 (2012)
5. Dubois, O., Gander, M.J., St-Cyr, A., Loisel, S., Szyld, D.: The optimized schwarz method with a coarse grid correction. SIAM J. Sci. Comput. **34**(1), 421–458 (2012)
6. Il'in, V.P.: Finite Difference and Finite Volume Methods for Elliptic Equations. ICMMG Publisher, Novosibirsk (2001). (in Russian)
7. Il'in, V.P.: Finite Element Methods and Technologies. ICMMG Publisher, Novosibirsk (2007). (in Russian)
8. Official page of Domain Decomposition Methods. http://www.ddm.org
9. Butyugin, D.S., Gurieva, Y.L., Il'in, V.P., Perevozkin, D.V., Petukhov, A.V.: Functionality and Algebraic Solvers Technologies in Krylov Library (in Russian). Vestnik YuUrGU. Series Computational mathematics and informatics. 2(3), 92–105 (2013)
10. Gander, M.J., Halpern, L., Santugini, K.: Domain decomposition methods in science and engineering XXI. In: Erhel, J., Gander, M.J., Halpern, L., Pichot, G., Sassi, T., Widlund, O. (eds.) A New Coarse Grid Correction for RAS/AS. LNCSE. Springer-Verlag, Switzerland (2013)
11. Siberian Supercomputer Centre. http://www2.sscc.ru
12. Intel Math Kernel Library (Intel MKL). http://software.intel.com/en-us/intel-mkl