Some Geometric and Algebraic Aspects of Domain Decomposition Methods

D.S.Butyugin[1], Y.L.Gurieva[1], V.P.Ilin[1,2], and D.V.Perevozkin[1]

Abstract Some geometric and algebraic aspects of various domain decomposition methods (DDMs) are considered. They are applied to a parallel solution of very large sparse SLAEs resulting from approximation of multi-dimensional mixed boundary value problems on non-structured grids. DDMs are used with parameterized overlapping of subdomains and various types of boundary conditions at the inner boundaries. An algorithm for automatic construction of a balancing domain decomposition for overlapping subdomains is presented. Subdomain SLAEs are solved by a direct or iterative preconditioned method in Krylov subspaces, whereas external iterations are performed by the FGMRES method. An experimental analysis of the algorithms is carried out on a set of model problems.

1 Introduction

The DDMs include a variety of geometric, algebraic, and functional aspects which are aimed at a high performance solution of large-size problems on post-petaflop computers.

Numerous works and Internet sites are devoted to this problem: monographs, papers, conference proceedings, programs, etc. [2], [3]. The issues that are of most interest from the practical point of view are the requirements on high resolution of the numerical approaches to solving multi-dimensional interdisciplinary boundary value problems described by systems of partial differential equations (PDEs) or the corresponding variational statements in the computational domains with complicated piecewise smooth boundaries and contrasting material properties of their subdomains. Approximation of such problems by finite-volume or finite-element methods on nonstructured grids results in very large systems of linear algebraic equations

[1)] Institute of Computational Mathematics and Mathematical Geophysics SB RAS, Novosibirsk, Russia [2)] Novosibirsk State University

(SLAEs) with 10^8–10^{10} unknowns with ill-conditioned or nondefinite sparse matrices with complicated portrait structures.

The solution to the SLAEs is a weak point of modern computing, and the DDMs are the main tool providing scalable parallelizm on multi-processor and multi-core systems. The goal of this paper is to experimentally investigate several approaches to automatic construction of balancing grid subdomains and to parallel solution of the resulting SLAE using the parametrized width of subdomain overlapping, different internal boundary conditions, aggregation techniques, see, for example, [10]. The results of a comparative analysis of the efficiency of various approaches for the model problems are presented. The computations were carried out with the Krylov library [5].

2 Grid domain decomposition without separator nodes

Let the matrix of the SLAE $Au = f$ be split into P subsystems:

$$(Au)_p = A_{p,p}u_p + \sum_{\substack{q=1 \\ q\neq p}}^{P} A_{p,q}u_q = f_p, \;\; p = 1,...,P, \;\; A = \{a_{i,j}\} \in \mathscr{R}^{N,N}, \qquad (1)$$

$$A = \{A_{p,q} \in \mathscr{R}^{N_p,N_q}\}, \;\; u = \{u_p \in \mathscr{R}^{N_p}\}, \;\; f = \{f_p \in \mathscr{R}^{N_p}\}, \;\; p,q = 1,\ldots,P.$$

Assume that SLAE (1) is a system of grid equations approximating a multidimensional boundary value problem for a differential equation, so that the components of the vectors u, f correspond to a grid point, the total number of nodes in the grid computational domain $\Omega^h = \bigcup_{p=1}^{P} \Omega_p^h$ being equal to N. The block decomposition of the matrix and vectors corresponds to the partitioning of Ω^h into P non-overlapping subdomains Ω_p^h, each consisting of N_p nodes, $N_1 + ... + N_p = N$. The decomposition of Ω^h does not use separator nodes, i.e., the boundaries of the subdomains do not pass through the grid nodes.

The process of system (1) solving can be parallelized by the additive Schwarz method:

$$A_{p,p}u_p^n = f_p - \sum_{\substack{q=1 \\ q\neq p}}^{P} A_{p,q}u_q^{n-1} \equiv g_p^{n-1}. \qquad (2)$$

The above matrix-algebraic representation of the structure of SLAE (1) can be extended by introducing a graph describing the same problem. Each ith grid node (or the ith row of the matrix A) can be associated with a vertex v_i of a graph G, and the mesh edge connecting the nodes i and $j \in \Omega^h$, can be associated with the edge of the graph $G = (V,E)$, $V = \{v_i;\ i = 1,...,N\}$, $E = \{(v_i,v_j)\ |a_{i,j} \neq 0, i,j = 1,...,N\}$.

Define an extended subdomain $\bar{\Omega}_p^h \supset \Omega_p^h$ with overlapping, whose breadth is defined in terms of the number of layers, or fronts, of the grid nodes.

Let $\Gamma_p^0 \in \Omega_p^h$ denote a set of internal near-boundary nodes, i.e., nodes $P_i \in \Omega_p^h$, in which one of the neighbors does not lie in Ω_p^h ($P_j \notin \Omega_p^h,\ j \in \omega_i,\ j \neq i$)). In Γ_p^0, define a subset of nodes $\Gamma_{p,q}^0$ in which the neighboring nodes belong to the adjacent subdomain $\Omega_q^h, q \in \bar{\omega}_p$, where $\bar{\omega}_p$ is a set of numbers of the subdomains adjacent to Ω_p^h. Thus, $\Gamma_p^0 = \bigcup\limits_{q \in \bar{\omega}_p} \Gamma_{p,q}^0$, and the subsets $\Gamma_{p,q}^0$ may intersect, i.e. they can contain near-boundary nodes with neighbors from different subdomains.

Let Γ_p^1 denote a set of nodes adjacent to the nodes from Γ_p^0 but not belonging to Ω_p^h and Γ_p^0; and let Γ_p^2 be a set of nodes adjacent to the nodes from Γ_p^1 but not belonging to the union $\Gamma_p^1 \bigcup \Omega_p^h$, etc. These sets will be called the first external layer (front) of nodes, the second layer, etc., respectively. The resulting collection of nodes $\Omega_p^\Delta = \Omega_p^h \bigcup \Gamma_p^1 ... \bigcup \Gamma_p^\Delta$ will be called the extended pth grid subdomain, and Δ, the extension breadth. The case $\Delta = 0$ actually means the decomposition of the domain Ω^h into subdomains without intersections ($\Omega_p^0 = \Omega_p^h$).

The set $\Gamma_p^\Delta \in \Omega_p^\Delta$ presents internal near-boundary nodes of the extended subdomain Ω_p^Δ, and $\Gamma_p^{\Delta+1}$, a set of external near-boundary nodes. Thus the geometric boundary of Ω_p^Δ runs between Γ_p^Δ and $\Gamma_p^{\Delta+1}$. Similarly to Γ_p^0, the set Γ_p^Δ can be partitioned into subsets of near-boundary nodes $\Gamma_p^\Delta = \Gamma_{p,q_1}^\Delta \bigcup \Gamma_{p,q_2}^\Delta ... \bigcup \Gamma_{p,q_{m_p}}^\Delta$ whose neighboring nodes are located, respectively, in the subdomains $\Omega_{q_1}^h, \Omega_{q_2}^h, ..., \Omega_{q_{m_p}}^h$ (here m_p denotes the number of subdomains that intersect Ω_p^Δ, and $q_1, q_2, ..., q_{m_p}$ are the numbers of these subdomains).

Consider iterative process 2 for the equation corresponding to the ith near-boundary node in $\bar{\Omega}_p^h$. Some of the neighbors belong to other subdomains $\bar{\Omega}_q^h$, $q \neq p$ but do not belong to $\bar{\Omega}_p^h$:

$$\Big(a_{i,i} + \theta_i \sum_{j \notin \bar{\Omega}_p} a_{i,j}\Big) u_i^n + \sum_{j \in \bar{\Omega}_p} a_{i,j} u_j^n = f_i + \sum_{j \notin \bar{\Omega}_p} a_{i,j} (\theta_i u_i^{n-1} - u_j^{n-1}). \tag{3}$$

Here $\theta_i \in [0,1]$ are parameters, corresponding for $\theta_i = 0$ or $\theta_i = 1$ to the Dirichlet or Neumann boundary conditions, and for $0 < \theta_i < 1$, to the Robin condition.

Introduce matrices $\bar{A}_{p,p} \in \mathscr{R}^{\bar{N}_p, \bar{N}_p}$, $\bar{A}_{p,q} \in \mathscr{R}^{\bar{N}_p, \bar{N}_q}$ for equation (3). Then the iterative process can be transformed to the form

$$\bar{A}_{p,p} \bar{u}_p^n = \bar{f}_p - \sum_{\substack{q=1 \\ q \neq p}}^{P} \bar{A}_{p,q} \bar{u}_q^{n-1} \equiv \bar{g}_p^{n-1}. \tag{4}$$

In the above discussion, we have considered the extension of the subdomain Ω_p^h towards its outer side. The same procedures are performed for the neighboring subdomains, which results in the construction of fronts inside Ω_p^h. These procedures can be implemented at the grid layers (fronts) in the extension of the neighboring subdomains $\Omega_q^h, q \neq p$.

Formula (4) does not describe the iterative process exactly since $\bar{N}_1 + ... + \bar{N}_p \geq N$. The vector u^n can be determined by partitioning the unit:

$$u_i^n = \sum_{q_i \in \bar{\omega}_i} \eta_{q_i} (\bar{u}_{q_i}^n)_i, \quad \sum_{q_i \in \bar{\omega}_i} \eta_{q_i} = 1, \tag{5}$$

where $\bar{\omega}_i$ is a set of the extended subdomains $\bar{\Omega}_{q_i}^h$ including the node P_i. Particular, but important, cases in (5) are $\eta_{q_i} = 1$ for $P_i \in \Omega_{q_i}^h$ and $\eta_{q_i} = 0$ for $P_i \neq \Omega_{q_i}^h$.

An alternative approach is to use iterations "in traces". Let $\bar{\Gamma}_p = \Gamma_p^{\Delta} \bigcup \Gamma_p^{\Delta+1}$ define the trace of the extended subdomain $\bar{\Omega}_p^h$ for $\theta_i \neq 0, P_i \in \Gamma_p^{\Delta}$, and $\bar{\Gamma}_p = \Gamma_p^{\Delta+1}$, for $\theta_i = 0$. We can write $\bar{\Gamma}_q = \bigcup \bar{\Gamma}_{q,p}$, where $\bar{\Gamma}_{p,q} = \bar{\Gamma}_p \bigcap \Omega_q^h$. From 2 we have

$$\bar{u}_p^n = \bar{A}_{p,p}^{-1} (\bar{f}_p - \sum_{\substack{q=1 \\ q \neq p}}^{P} \hat{A}_{p,q} \hat{u}_p^{n-1}). \tag{6}$$

Here the matrices $\bar{A}_{p,p}$ are assumed to be non-singular, $\hat{A}_{p,q} \in \mathscr{R}^{\hat{N}_p, \hat{N}_{p,q}}$ and $\hat{u}_p = \{u_i; P_i \in \bar{\Gamma}_{p,q}\} \in \mathscr{R}^{\hat{N}_{p,q}}, \hat{N}_{p,q}$ being the number of nodes in $\bar{\Gamma}_{p,q}$.

If $\bar{u}_p^n \to \bar{u}_p$ for $n \to \infty$, iterations (6) provide the solution of the preconditioned SLAE

$$\bar{A}u = \bar{f}, \;\; \bar{f} \in \mathscr{R}^N, \;\; \bar{A} \in \mathscr{R}^{N,N}. \tag{7}$$

Multiplying equation (6) by $\bar{A}_{q,p}$ and denoting $\bar{A}_{p,q} \bar{u}_q^n = \hat{A}_{p,q} \hat{u}_q^n = v_{p,q}^n \in \mathscr{R}^{\bar{N}_p}$, we obtain the algebraic system "in traces":

$$v_{q,p} + \bar{A}_{q,p} \bar{A}_{p,p}^{-1} \sum_{\substack{q=1 \\ q \neq p}}^{P} v_{p,q} = \bar{A}_{q,p} \bar{A}_{p,p}^{-1} \bar{f}_p, \;\; p = 1, ..., P; \; q \in \bar{\omega}_p. \tag{8}$$

The degree of freedom of this SLAE is $\hat{N} = \sum\limits_{p=1}^{P} \hat{N}_p = \sum\limits_{p=1}^{P} \sum\limits_{q \in \bar{\omega}_p} \hat{N}_{p,q} \ll N$. Iterative solution of equation (8) can be implemented by a Krylov method. To speed up the iterative DDM process, various approaches, for example, deflation, coarse grid correction, and smoothed aggregation can be used. We consider the SLAE reduction procedure based on an interpolation principle, under the assumption of smooth behavior of the solution to be sought for in each subdomain.

Define a prolongation matrix $W^T = \{w_p\} \in \mathscr{R}^{N,P}$, where the vectors (columns) w_p have nonzero (unit) entries corresponding to the subdomain Ω_p only. Then $\hat{A} = W^T A W \in \mathscr{R}^{P,P}$ presents the global aggregation matrix, and $B = W \hat{A}^{-1} W^T$ is, in a sense, an aggregating preconditioning matrix. For simplicity, we consider non-overlapping subdomains. In this case, the matrix in (7) has a simple form $\bar{A} = B_J A$, where B_J is the block Jacobi preconditioner [10].

DDM-exploiting iterative processes can be consructed in various ways. We use a simple one, namely, the FGMRES [10] with dynamic preconditioner B_n: $B_n = B_A$ for $n = km + 1$, $k = 0, 1, \ldots$, and $B_n = B_J$ otherwise. The stopping criteria of this process are

$$||\hat{r}^n|| \leq ||\hat{f} - \hat{A}u^n|| \leq \varepsilon^e ||\hat{f}||, \;\; \varepsilon^e \ll 1, \text{ or } n \leq n_{max}^e. \tag{9}$$

Subdomain SLAEs are solved by either the direct solver PARDISO [4] or the iterative BiCGStab method [10].

In the latter case of a two-level iterative algorithm, various approaches can be chosen for defining the internal stopping criteria $\varepsilon^i \leq \varepsilon^e$ and n^i_{max}, similarly to (9).

3 Parallel Implementation of Algorithms

The major question in high-performance implementation of DDMs is automatic construction of balancing grid subdomains, based, for instance, on CSR format of the original SLAE. This problem is solved by the graph partitioning approach in two stages. First, we define P subdomains Ω^h_p, $p = 1, \ldots, P$, without intersections. Then extended subdomains $\bar{\Omega}^h_p$ with a given breadth Δ of overlapping are constructed on the basis of the following algorithm.

The non-overlapping grid subdomains Ω^h_p are formulated as subgraphs $G_p(V_p, E_p)$ with possibly small diameters containing approximately equal numbers of vertices $N_p \approx N/P$. In practice, the task consists in transforming the original CSR format to the CSR_p formats for P subdomains, which should be distributed among the corresponding MPI processes.

The graph partitioning is a multi-level aggregation procedure of the sequential macrographs $G^{(l)}(V^{(l)}, E^{(l)}) = \{G^{(l)}_p(V^{(l)}_p, E^{(l)}_p)\}$, $l = 0, 1..., L$, $p = 1, ..., P_l$. Here L and P_l are the number of levels and the number of macrovertices at the lth level, respectively, whose macro-vertices include several vertices of a lower level. If $G^{(0)}(V^{(0)}, E^{(0)})$ denotes the original grid graph, the first aggregation step can be described by the following pseudocode (breadth-first search [8]):

$i = 1$, *while* $\{u \in V \mid C(u) = 0\} \neq \emptyset$
  pick any v *from* $\{u \in V \mid C(u) = 0\}$
  $Q := \{v\}, \quad n = 0$
  while $(n < n_{max}$ *and* $Q \neq \emptyset)$
    $v \leftarrow Q, \qquad C(v) := i$
    $Q \leftarrow (Adj(v) \cap \{u \in V \mid C(u) = 0\}) \setminus Q$
    $n = n + W(v)$
  end while
  $i = i + 1$
end while

Here $C(u)$ and $W(u)$ are the color and weight (integers) of the vertex u, respectively, with the initial values $C(u) = 0, W(u) = 1$, Adj(v) is a set of vertices adjacent to u, and Q is the stack type data structure. Hence, $C(u)$ presents the number of a subdomain (macrovertex) containing the vertex (grid point) u, and $W(u)$ is the resulting number of nodes in the subdomain ($W(u) \leq n_{max}$). This algorithm is repeated for the levels $l = 1, ..., L$.

Parallel implementation of DDM–FGMRES is performed using hybrid programming with MPI processes on distributed memory for subdomains and OpenMP tools for each of the multi-core processors with shared memory.

4 Numerical Experiments

We present the results of some numerical experiments on solving a model Dirichlet boundary value problem for the 2D and the 3D Laplace equation in the unit computational domain $\Omega = [0,1]^d$, $d = 2,3$, which is approximated by a standard (2d+1)-point finite difference scheme on a square mesh (which is cubic in 3D) with the degree of freedom $N = N_x^d$, for different values of N_x. The stopping criteria for FGMRES without restarts were $\varepsilon^e = 10^{-7}$ and $n^e_{max} = \infty$. The exact solution and initial guess for the iterations were taken equal to unit and zero, respectively. All the experiments were carried out on the NKS-30T cluster [1] with standard double-precision arithmetic.

Table 1 shows the efficiency of the proposed algorithm for automatic construction of 3D balancing grid subdomains for $P = 1,8,16,32,64$. The subdomain SLAEs were solved either by the direct method PARDISO from Intel MKL or by the preconditioned BiCGStab method (Eisenstadt modification of incomplete factorization [7]) with the parameters $\varepsilon^i = 0.1$, $n^i_{max} = 5$ (these values are nearly optimal for the given problem data). Note that the PARDISO was run with 12 threads, whereas the BiCGStab was implemented without any parallelization. In Table 1, the upper and lower figures in each line correspond to grids with 128^3 and 256^3 unknowns, respectively, and the left and right figures in each column present the numbers of external iterations and execution time in seconds. In this case, the DDM parameters $\Delta = \theta = 0$ were used.

Table 1 Comparative analysis of DDM without overlapping for direct and iterative subdomain solvers, $\theta = 0, N = 128^3, 256^3$

method \ P	1		8		16		32		64	
direct	1	885	53	30.1	75	20.4	108	12.6	130	18.1
		- -	72	332	102	212	142	138	169	189
iterative	18	64.9	68	20.5	92	12.5	103	13.0	197	11.9
	18	606	99	296	132	203	197	139	262	115

Table 2 presents the number of iterations for the aggregation approach for the same model SLAEs with the exact solution $u = 1000 + x + y$ and initial guess $u^0 = 0$. The aggregation preconditioner was used once every m steps (with $m = 10$ as an optimal value). Note that the behavior is also observed for different numbers of

subdomains, whereas the results are given here for $P = 16$ and 32 (upper and lower cell values, respectively). The case $m = 0$ means solving without aggregation.

Table 2 Numbers of external iterations for solving SLAEs with aggregation preconditioning

$N \setminus m$	0	1	5	10	15	20
	82	50	46	41	42	46
128^3	132	62	53	52	57	58
	143	70	54	51	53	53
256^3	193	60	72	61	62	68

In the other experiments, 2D problems were solved on square meshes with $N = 128^2, 256^2$, and $P = 4, 16, 64$ equal square subdomains. The systems in the subdomains were solved by the PARDISO, and the external iterations were carried out by the iterative BiCGStab method "in traces".

Table 3 presents the iterative process versus the overlapping value Δ. The cells present the same data as in Table 1 for $\theta = 0$, and $N = 128^2, 256^2$ (upper and lower lines in each row, respectively). We see that the number of iterations decreases monotonically with increasing Δ, but for the run time there is some minimum for a sufficiently small value $\Delta \leq 4$.

Table 3 Numerical results for different overlapping values Δ, $\theta = 0$, $N = 128^2, 256^2$

$P \setminus \Delta$	0		1		2		3		4		5	
	18	1.75	11	1.45	9	1.37	7	1.26	7	1.26	6	1.20
4												
	27	6.85	16	4.37	12	3.51	10	3.02	9	2.82	8	2.49
	32	1.42	18	1.18	14	1.19	12	1.09	11	0.89	9	0.79
16												
	41	3.85	24	2.83	20	2.20	17	1.80	14	1.38	14	1.66
	43	1.56	26	1.66	19	1.39	16	1.50	14	1.56	12	0.86
64												
	60	4.75	36	4.16	27	3.35	22	3.11	20	3.00	18	4.66

Table 4 contains the number of iterations versus θ values. The left and right cell values correspond to $N = 128^2$ and $N = 256^2$, respectively. No overlapping takes place, i.e. $\Delta = 0$.

Table 4 Number of iterations for non-overlapping DDMs ($\Delta = 0$) with different θ, $N = 128^2, 256^2$

$P \setminus \theta$	0		0.5		0.6		0.7		0.9975	
4	18	27	16	26	16	24	14	23	10	12
16	32	41	28	40	27	39	27	40	31	75
64	43	60	42	56	40	55	41	55	93	86

These results demonstrate that the constant parameter θ is appropriate only for a sufficiently small P. The experiments have also shown that for the overlapping decomposition ($\Delta > 0$) it is better to take $\theta = 0$.

5 Conclusions

Our preliminary numerical results show that the DDMs considered have reasonable efficiency. However, there are too many approaches needing systematic experimental investigation to construct high-performance code. This concerns, in particular, the application of various optimized Schwarz methods [6], [9] with different values of parameter θ and coarse grid correction for overlapping or non-overlapping DDM. Of course, the problem of creating an adapted environment for robust SLAE solvers on modern supercomputers requires coordinated efforts of algebraists and programmers.

References

[1] NKS-30T cluster, 2009. URL http://www2.sscc.ru/HKC-30T/HKC-30T.htm.
[2] 22nd international conference on domain decomposition methods (DD22), 2013. URL http://dd22.ics.usi.ch/.
[3] Domain decomposition methods, 2013. URL http://ddm.org.
[4] *Intel (R) Math Kernel Library*, 2013. URL http://software.intel.com/en-us/articles/intel-mkl/.
[5] Dmitry S. Butyugin, Valery P. Il'in, and Danil V. Perevozkin. Parallel methods for solving the linear systems with distributed memory in the library Krylov (in Russian). *Herald of SUSU. Series "Computational Mathematics and Informatics"*, 47 (306):5–19, 2012.
[6] Martin J. Gander. Optimized Schwarz methods. *SIAM J. Num. Anal.*, 44: 699–731, 2006.
[7] V.P. Ilin. *Finite element methods and technologies (in Russian)*. Novosibirsk, ICM&MG SBRAS Publ., 2007.
[8] Sergio Pissanetsky. *Sparse matrix technology*. Academic Press, New York, 1984.
[9] Lizhen Qin and Xuejun Xu. On a parallel Robin-type nonoverlapping domain decomposition method. *SIAM J. Numer. Anal.*, 44(6):2539–2558, 2006. ISSN 0036-1429.
[10] Yousef Saad. *Iterative methods for sparse linear systems*. Society for Industrial and Applied Mathematics, Philadelphia, PA, second edition, 2003. ISBN 0-89871-534-2. doi: 10.1137/1.9780898718003. URL http://dx.doi.org/10.1137/1.9780898718003.