Parallel Adaptive Cross Approximation for the Multi-trace Formulation of Scattering Problems

Michal Kravčenko^{1,2} , Jan Zapletal^{1,2} , Xavier Claeys³ ,
and Michal Merta^{1,2}

¹ IT4Innovations, VŠB – Technical University of Ostrava,
17. listopadu 2172/15, 708 00 Ostrava-Poruba, Czech Republic

² Department of Applied Mathematics, VŠB – Technical University of Ostrava,
17. listopadu 2172/15, 708 00 Ostrava-Poruba, Czech Republic
{michal.kravcenko,jan.zapletal,michal.merta}@vsb.cz

³ Jacques-Louis Lions Laboratory, Sorbonne Université,
Boite courrier 187, 75252 Paris Cedex 05, France
claeys@ann.jussieu.fr

Abstract. We present a highly parallel version of the boundary element method accelerated by the adaptive cross approximation for the efficient solution of scattering problems with composite scatterers. Individual boundary integral operators are treated independently, i.e. the boundary of every homogeneous subdomain is decomposed into clusters of elements defining a block structure of the local matrix. The blocks are distributed across computational nodes by a graph algorithm providing a load balancing strategy. The intra-node implementation further utilizes threading in shared memory and in-core SIMD vectorization to make use of all features of modern processors. The suggested approach is validated on a series of numerical experiments presented in the paper.

Keywords: Boundary element method · Adaptive cross approximation · Multi-trace formulation · Distributed parallelization

1 Introduction

Boundary integral equations present a valuable tool for the description of natural phenomena including wave scattering problems. Their numerical counterpart, the boundary element method (BEM), has become an alternative to volume based approaches such as finite element or finite volume methods. Except for a natural transition of the problem to the skeleton of the computational domain, BEM has found its place in HPC implementations of PDE solvers. One of the reasons is high computational intensity of system matrix assemblers, which fits well with today's design of HPC clusters, where memory accesses are much more costly than arithmetic operations. To overcome the quadratic complexity of BEM, several methods have been proposed including the fast multipole [4, 15],

© The Author(s)

R. Wyrzykowski et al. (Eds.): PPAM 2019, LNCS 12043, pp. 141–150, 2020.

https://doi.org/10.1007/978-3-030-43229-4_13

or adaptive cross approximation (ACA) [2, 16, 17] methods, with the latter one utilized in this paper.

To describe wave scattering in a composite scatterer and its complement we utilize the local version of the multi-trace formulation (MTF) as introduced in [5–7] and presented here briefly in Sect. 2. The formulation leads to a block matrix with individual boundary integral operators for every homogeneous subdomain and coupling matrices on the off-diagonals. Although not described in detail in this paper, MTF allows for a natural operator preconditioning. In Sect. 3 we propose a strategy to parallelize the assembly of the MTF matrix blocks and their application in an iterative solver based on the approach presented in [11–13] for single domain problems. Except for the distributed parallelism, the method takes full advantage of the BEM4I library [14, 20, 21] and its assemblers parallelized in shared memory and vectorized by OpenMP. We provide the results of numerical experiments in Sect. 4.

2 Multi-trace Formulation

In the following we consider a partitioning of the space into $n + 1$ Lipschitz domains

$$\mathbb{R}^3 = \bigcup_{i=0}^n \overline{\Omega_i}, \quad \Gamma_i := \partial\Omega_i, \quad \Sigma := \bigcup_{i=0}^n \Gamma_i$$

with Ω_0 denoting the background unbounded domain, $\bigcup_{i=1}^n \Omega_i$ representing a composite scatterer, and the skeleton Σ . For a given incident field u_{inc} satisfying $-\Delta u_{\text{inc}} - \kappa_0^2 u_{\text{inc}} = 0$ in $\mathbb{R}^3$, we aim to solve the equation for $u = u_{\text{sc}} + u_{\text{inc}}$,

$$-\Delta u - \kappa_i^2 u = 0 \text{ in } \Omega_i, \quad u - u_{\text{inc}} \text{ satisfies the Sommerfeld condition} \quad (2.1)$$

with the transmission conditions

$$u_i - u_j = 0, \quad \mu_i^{-1} t_i + \mu_j^{-1} t_j = 0 \quad \text{on } \Gamma_{i,j} := \Gamma_i \cap \Gamma_j \quad (2.2)$$

with u_i , t_i denoting the Dirichlet and Neumann traces of $u|_{\Omega_i}$. Note that the normal vector $\mathbf{n}_i$ is always directed outside of Ω_i , see Fig. 1.

Due to (2.1) and u_{inc} satisfying the Helmholtz equation with κ_0 in $\mathbb{R}^3 \setminus \overline{\Omega_0}$ we have $u = \tilde{V}_{\kappa_0} t_0 - W_{\kappa_0} u_0 + u_{\text{inc}}$ in Ω_0 and $u = \tilde{V}_{\kappa_i} t_i - W_{\kappa_i} u_i$ in Ω_i , $i > 0$ [16, 18] with the single- and double-layer potentials

$$\begin{aligned} \tilde{V}_{\kappa_i} : H^{-1/2}(\Gamma_i) &\rightarrow H_{\text{loc}}^1(\Omega_i), & \tilde{V}_{\kappa_i} t_i(\mathbf{x}) &:= \int_{\Gamma_i} v_{\kappa_i}(\mathbf{x}, \mathbf{y}) t_i(\mathbf{y}) \, d\mathbf{s}_{\mathbf{y}}, \\ W_{\kappa_i} : H^{1/2}(\Gamma_i) &\rightarrow H_{\text{loc}}^1(\Omega_i), & W_{\kappa_i} u_i(\mathbf{x}) &:= \int_{\Gamma_i} \frac{\partial v_{\kappa_i}}{\partial \mathbf{n}_{i,\mathbf{y}}}(\mathbf{x}, \mathbf{y}) u_i(\mathbf{y}) \, d\mathbf{s}_{\mathbf{y}}, \end{aligned}$$

respectively, and the fundamental solution $v_{\kappa}(\mathbf{x}, \mathbf{y}) = \exp(i\kappa|\mathbf{x} - \mathbf{y}|)/(4\pi|\mathbf{x} - \mathbf{y}|)$. Observing that

$$\begin{bmatrix} -K_{\kappa_0} & V_{\kappa_0} \\ D_{\kappa_0} & K_{\kappa_0}^* \end{bmatrix} \begin{bmatrix} u_{\text{inc}} \\ t_{\text{inc}} \end{bmatrix} = -\frac{1}{2} \begin{bmatrix} u_{\text{inc}} \\ t_{\text{inc}} \end{bmatrix} \quad \text{on } \Gamma_0$$

Fig. 1. Notation and setting for the scattering problem.

results in the relation between the traces of $u|_{\Omega_i}$

$$\begin{bmatrix} -K_{\kappa_i} & V_{\kappa_i} \\ D_{\kappa_i} & K_{\kappa_i}^* \end{bmatrix} \begin{bmatrix} u_i \\ t_i \end{bmatrix} - \frac{1}{2} \begin{bmatrix} u_i \\ t_i \end{bmatrix} = \begin{bmatrix} f_i \\ g_i \end{bmatrix} \text{ on } \Gamma_i, \quad \begin{bmatrix} f_0 \\ g_0 \end{bmatrix} := - \begin{bmatrix} u_{\text{inc}} \\ t_{\text{inc}} \end{bmatrix}, \quad \begin{bmatrix} f_i \\ g_i \end{bmatrix} := 0 \text{ for } i > 0, \quad (2.3)$$

with the boundary integral operators defined as the composition of the trace operators $\gamma_i^D: H_{\text{loc}}^1(\Omega_i) \rightarrow H^{1/2}(\Gamma_i)$, $\gamma_i^N: H_{\text{loc}}^1(\Omega_i) \rightarrow H^{-1/2}(\Gamma_i)$ and the potentials, i.e.

$$V_{\kappa_i} := \gamma_i^D \tilde{V}_{\kappa_i}, \quad -\frac{1}{2}I + K_{\kappa_i} := \gamma_i^D W_{\kappa_i}, \quad \frac{1}{2}I + K_{\kappa_i}^* := \gamma_i^N V_{\kappa_i}, \quad D_{\kappa_i} := -\gamma_i^N W_{\kappa_i}.$$

The idea of MTF is to replace the identity part of (2.3) by the transmission conditions (2.2). To this end we define the operators

$$X_{i,j} u_i := \begin{cases} \frac{1}{2} u_i|_{\Gamma_{i,j}} & \text{on } \Gamma_{i,j}, \\ 0 & \text{otherwise,} \end{cases} \quad Y_{i,j} t_i := \begin{cases} \frac{1}{2} \frac{\mu_i}{\mu_j} t_i|_{\Gamma_{i,j}} & \text{on } \Gamma_{i,j}, \\ 0 & \text{otherwise.} \end{cases}$$

The variational formulation for the situation in Fig. 1 thus reads

$$\left\langle \begin{bmatrix} -K_{\kappa_0} & V_{\kappa_0} & -X_{1,0} & 0 & -X_{2,0} & 0 \\ D_{\kappa_0} & K_{\kappa_0}^* & 0 & Y_{1,0} & 0 & Y_{2,0} \\ -X_{0,1} & 0 & -K_{\kappa_1} & V_{\kappa_1} & -X_{2,1} & 0 \\ 0 & Y_{0,1} & D_{\kappa_1} & K_{\kappa_1}^* & 0 & Y_{2,1} \\ -X_{0,2} & 0 & -X_{1,2} & 0 & -K_{\kappa_2} & V_{\kappa_2} \\ 0 & Y_{0,2} & 0 & Y_{1,2} & D_{\kappa_2} & K_{\kappa_2}^* \end{bmatrix} \begin{bmatrix} u_0 \\ t_0 \\ u_1 \\ t_1 \\ u_2 \\ t_2 \end{bmatrix}, \begin{bmatrix} s_0 \\ v_0 \\ s_1 \\ v_1 \\ s_2 \\ v_2 \end{bmatrix} \right\rangle_{\Sigma} = - \left\langle \begin{bmatrix} u_{\text{inc}} \\ t_{\text{inc}} \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} s_0 \\ v_0 \\ s_1 \\ v_1 \\ s_2 \\ v_2 \end{bmatrix} \right\rangle_{\Sigma}$$

with

$$(u_i, t_i) \in H^{1/2}(\Gamma_i) \times H^{-1/2}(\Gamma_i), \quad (v_i, s_i) \in \tilde{H}_{\text{pw}}^{1/2}(\Gamma_i) \times \tilde{H}_{\text{pw}}^{-1/2}(\Gamma_i).$$

For a definition of the above Sobolev spaces on manifolds we refer the interested reader to [1, 7, 18, 19] and references therein.

To discretize the system we decompose the skeleton Σ into plane triangles τ_k and define the discrete function space $\text{span}(\varphi_\ell)$ of globally continuous piecewise

linear functions to approximate all the above spaces. The discrete system thus reads

$$\begin{bmatrix} -K_{\kappa_0,h} & V_{\kappa_0,h} & -M_{1,0,h} & O & -M_{2,0,h} & O \\ D_{\kappa_0,h} & K_{\kappa_0,h}^\top & O & \frac{\mu_0}{\mu_1} M_{1,0,h} & O & \frac{\mu_0}{\mu_2} M_{2,0,h} \\ -M_{0,1,h} & O & -K_{\kappa_1,h} & V_{\kappa_1,h} & -M_{2,1,h} & O \\ O & \frac{\mu_1}{\mu_0} M_{0,1,h} & D_{\kappa_1,h} & K_{\kappa_1,h}^\top & O & \frac{\mu_1}{\mu_2} M_{2,1,h} \\ -M_{0,2,h} & O & -M_{1,2,h} & O & -K_{\kappa_2,h} & V_{\kappa_2,h} \\ O & \frac{\mu_2}{\mu_0} M_{0,2,h} & O & \frac{\mu_2}{\mu_1} M_{1,2,h} & D_{\kappa_2,h} & K_{\kappa_2,h}^\top \end{bmatrix} \begin{bmatrix} \mathbf{u}_0 \\ \mathbf{t}_0 \\ \mathbf{u}_1 \\ \mathbf{t}_1 \\ \mathbf{u}_2 \\ \mathbf{t}_2 \end{bmatrix} = - \begin{bmatrix} M_{0,h} \mathbf{u}_{\text{inc}} \\ M_{0,h} \mathbf{t}_{\text{inc}} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix}$$

with the matrices

$$\begin{aligned} V_{\kappa_i,h}[k,\ell] &:= \langle V_{\kappa_i} \varphi_\ell, \varphi_k \rangle_{\Gamma_i}, & K_{\kappa_i,h}[k,\ell] &:= \langle K_{\kappa_i} \varphi_\ell, \varphi_k \rangle_{\Gamma_i}, \\ D_{\kappa_i,h}[k,\ell] &:= \langle D_{\kappa_i} \varphi_\ell, \varphi_k \rangle_{\Gamma_i}, \\ M_{i,j,h}[k,\ell] &:= \frac{1}{2} \langle \varphi_\ell, \varphi_k \rangle_{\Gamma_{i,j}}, & M_{0,h}[k,\ell] &:= \langle \varphi_\ell, \varphi_k \rangle_{\Gamma_0} \end{aligned} \quad (2.4)$$

and the duality pairings

$$\langle u_i, t_i \rangle_{\Gamma_i} = \int_{\Gamma_i} u_i(\mathbf{x}) t_i(\mathbf{x}) \, d\mathbf{s}_x, \quad \langle u_i, t_j \rangle_{\Gamma_{i,j}} = \int_{\Gamma_{i,j}} u_i(\mathbf{x}) t_j(\mathbf{x}) \, d\mathbf{s}_x.$$

The local indices k, ℓ in (2.4) span over all globally defined basis functions supported on the respective interfaces Γ_i or $\Gamma_{i,j}$.

3 Parallel ACA

The matrices produced by a classical BEM are usually dense. Although the number of degrees of freedom is smaller compared to the volume-based methods (e.g., finite element method), some of the so-called fast BEM have to be applied in order to solve large-scale problems. These methods are usually based on a hierarchical clustering of the underlying mesh and approximations of matrix blocks corresponding to pairs of sufficiently separated clusters. Here we use the adaptive cross approximation (ACA) method since it is a purely algebraic approach and once properly implemented, it can be used for various kind of problems. Other approaches include the fast multipole method [9, 15] or the wavelet compression [8].

After hierarchical clustering, ACA proceeds by approximating sub-matrices corresponding to pairs of well-separated (admissible) clusters by a product of two low-rank matrices. Non-admissible clusters are assembled as dense matrices. Due to a limited scope of this paper, we refer the reader to [3, 16] for more details. The parallelization of the method based on the cyclic graph decomposition was presented in [13] where only certain special numbers of processors were discussed. In [12] we further extended the approach to support general number of processors. In the following section we recollect its basic principle and extend it to support the solution of MTF systems.

Let us first briefly describe the parallelization of a single-domain problem. To distribute BEM system matrices among P processors we decompose the input surface mesh Γ into P disjoint sub-meshes $\Gamma^1, \Gamma^2, \dots, \Gamma^P$ of approximately the same number of elements using, e.g. the METIS library [10]. In this manner we obtain a $P \times P$ block structure of matrices such that the block in row i and column j is induced by the integration over Γ^i and Γ^j . Afterwards, the P^2 blocks are assembled in parallel via P MPI processes. The workload distribution follows the so-called cyclic graph decomposition introduced in [13] for special values of P and later generalized in [12]. This way, each MPI process requires $\mathcal{O}(n/\sqrt{P})$ mesh data for the assembly of the blocks and actively works with $\mathcal{O}(n/\sqrt{P})$ degrees of freedom during matrix-vector multiplication, which has a positive effect on the efficiency of the required MPI synchronization phase. See Fig. 2 for an example of an 8×8 block distribution together with the respective generator graph.

Fig. 2. Workload block distribution for 8 MPI processes (left) and the respective generator graph (right). The vertex indices connected by an edge correspond to a block of the matrix assigned to the first process. Other processes are assigned blocks determined by a clockwise rotation of the generator graph.

The extension of the proposed parallelization scheme to local multi-trace operators can be implemented in a straight-forward manner. Let $\Omega_0, \Omega_1, \dots, \Omega_m$ denote the subdomains with their respective boundaries $\Gamma_0, \Gamma_1, \dots, \Gamma_m$. We apply our parallel ACA scheme to each subdomain individually, i.e. each Γ_j is split into P submeshes. The BEM operators $K_{\kappa_j, h}, V_{\kappa_j, h}, D_{\kappa_j, h}$ are treated as $P \times P$ block operators, each assembled in parallel according to the corresponding cyclic graph decomposition. This approach works reasonably well, however, distributing each boundary among the same number of processes may lead to a poor parallel efficiency in the case of Γ_j with small number of elements. Thus, the goal of our future research is to design an advanced parallel scheme for the assembly of local operators such that $K_{\kappa_j, h}, V_{\kappa_j, h}, D_{\kappa_j, h}$ are assembled by various numbers of MPI processes.

4 Numerical Experiments

Our main interest lies in the study of the parallel matrix assembly and efficiency of the matrix-vector multiplication. The results of strong and weak scalability experiments are presented in Tables 1 and 2, respectively.

The numerical experiments were carried out on the Salomon supercomputer at IT4Innovations National Supercomputing Center, Czech Republic. The cluster is composed of 1008 compute nodes, each of them equipped with two 12-core Intel Xeon E5-2680v3 processors and 128 GB of RAM. The nodes are interconnected by the 7D enhanced hypercube InfiniBand network. The theoretical peak performance totals 2 PFlop/s. We used METIS 5.1.0 for the decomposition of the mesh [10] and Intel Parallel Studio 2017.1.132 with Intel Math Kernel Library (MKL) as a BLAS implementation. We use two MPI processes per compute node, 12 OMP threads per one MPI process and set `KMP_AFFINITY="granularity=fine,scatter"`.

Fig. 3. A cut through the test domain geometry (left) and plot of the resulting total field (right). The test domain is a cube split into three parts.

All the tests have been performed on a cubical geometry split into three parts, see Fig. 3 for a central cut of the domain and also the depiction of the resulting total field. The wave numbers are $\kappa_0 = 4$, $\kappa_1 = 2$, $\kappa_2 = 5$, $\kappa_3 = 3$ and $\mu_0 = \mu_1 = \mu_2 = \mu_3 = 1$. We used globally continuous piecewise linear trial and test functions for all operators included in the formulation. The parameters controlling the complexity of ACA were set to $\varepsilon_{\text{ACA}} = 10^{-6}$, $\mu_{\text{ACA}} = 1.2$. The relative precision for the GMRES solver was set to $\varepsilon_{\text{GMRES}} = 10^{-6}$.

The measured quantities are the time in seconds to assemble the matrices, the time to perform one matrix-vector multiplication without MPI synchronization, and also the overhead required to synchronize the results via MPI. We also present the efficiency of the parallel solver. We performed a series of five experiments and the presented values are the averages of the results.

In the case of strong scaling experiments (see Table 1), I_0 contains 258 048 elements and I_1, I_2, I_3 contain 110 592 elements each. Let the real time for parallel matrix assembly and matrix-vector multiplication on P processes be t_P , then

the efficiency of the strong scaling on P processes is calculated as $(2t_2)/(Pt_P)$. Taking into account the relatively small size of the problem, we obtain a reasonable parallel efficiency of the system matrix assembly up to 64 compute nodes (128 processes). The matrix-vector multiplication scales relatively well up to 16 nodes (32 processes). The reason is twofold. The first major factor is increasing time required for vector synchronization after each multiplication. The second factor is the increasing density of the ACA matrices on higher number of processes. In the current implementation, our synchronization scheme is the following:

- we split the outer vector into 4 disjoint intervals (one for each subdomain),
- then we perform all necessary computations for each interval followed by a non-blocking allreduce across all processes,
- to facilitate the progress of the non-blocking reductions, we perform periodic calls to `MPI_Test` on the master OMP thread.

Table 1. Strong scalability of operator assembly, matrix-vector multiplication and total runtime of the solver on 1, 2, $\dots$, 64 compute nodes (MPI synchronization included).

P	Matrix assembly		Matrix-vector multiply			Total time	
	[s]	eff. [%]	apply [s]	sync. [s]	eff. [%]	[s]	eff. [%]
2	627.1	100.0	2.55	0.010	100.0	1614.6	100.0
4	316.2	99.2	1.27	0.028	98.8	884.1	91.3
8	177.2	88.5	0.66	0.028	92.8	542.4	74.4
16	112.5	69.7	0.34	0.039	83.5	377.6	53.4
32	54.7	71.7	0.19	0.052	66.7	273.7	36.9
64	26.8	73.0	0.10	0.076	45.1	226.9	22.2
128	14.3	68.5	0.06	0.091	26.0	214.1	11.8

The weak scaling experiments were performed on 1, 4 and 16 compute nodes each running two MPI processes (see Table 2). On a single node, Γ_0 contains $k_0 := 81\,648$ elements, and $\Gamma_1, \Gamma_2, \Gamma_3$ contain $k_1 := 34\,992$ elements each. On $P/2$ nodes, the number of mesh elements is proportionally increased, i.e. $Pk_0/2$ for Γ_0 and $Pk_1/2$ for Γ_1, Γ_2 and Γ_3 . Let the expected complexity of parallel matrix assembly and matrix-vector multiplication on P processes be $e_P := (k_0 \log(\frac{P}{2}k_0) + 3k_1 \log(\frac{P}{2}k_1))/2$ and let the real time be t_P , respectively. The efficiency of the weak scaling on P processes is then calculated as $(t_2 e_P)/(t_P e_2)$.

Table 2. Efficiency of the weak scaling of operator assembly and matrix-vector multiplication on 1, 4, 16 compute nodes (MPI synchronization included).

P	Matrix assembly		Matrix-vector multiply		
	t [s]	eff. [%]	apply t [s]	sync. t [s]	eff. [%]
2	206.7	100.0	0.79	0.0069	100.0
8	247.9	94.0	1.11	0.0476	80.2
32	322.2	80.6	1.99	0.2080	49.6

5 Conclusion

We briefly presented a local multi-trace formulation for scattering problems with heterogeneous scatterers and applied the so-called cyclic graph decomposition to define a block-based workload distribution for distributed parallel matrix assembly and matrix-vector multiplication. We performed experiments on up to 64 compute nodes and presented strong and weak scaling properties of our approach. In our following work, we plan to refine and generalize the proposed methods to directly decompose the skeleton of the scatterer instead of the individual subdomains. We believe this could lead to a better scalability and ability to deal with problems with differently sized subdomains.

Acknowledgement. The authors acknowledge the support provided by The Ministry of Education, Youth and Sports from the National Programme of Sustainability (NPS II) project ‘IT4Innovations excellence in science – LQ1602’ and from the Large Infrastructures for Research, Experimental Development and Innovations project ‘IT4Innovations National Supercomputing Center – LM2015070’. MK was further supported by the ESF in ‘Science without borders’ project, reg. nr. CZ.02.2.69/0.0./0.0./16_027/0008463 within the Operational Programme Research, Development and Education.

References

1. Amann, D.: Boundary element methods for Helmholtz transmission problems. Master’s thesis, TU Graz (2014)
2. Bebendorf, M.: Hierarchical Matrices: A Means to Efficiently Solve Elliptic Boundary Value Problems. Lecture Notes in Computational Science and Engineering. Springer, Heidelberg (2008). <https://doi.org/10.1007/978-3-540-77147-0>
3. Bebendorf, M., Rjasanow, S.: Adaptive low-rank approximation of collocation matrices. Computing **70**(1), 1–24 (2003). <https://doi.org/10.1007/s00607-002-1469-6>
4. Chaillat, S., Collino, F.: A wideband fast multipole method for the Helmholtz kernel: theoretical developments. Comput. Math. Appl. **70**(4), 660–678 (2015)
5. Claeys, X., Dolean, V., Gander, M.: An introduction to multi-trace formulations and associated domain decomposition solvers. Appl. Numer. Math. **135**, 69–86 (2019)

6. Claeys, X., Hiptmair, R.: Multi-trace boundary integral formulation for acoustic scattering by composite structures. *Commun. Pure Appl. Math.* **66**(8), 1163–1201 (2013)
7. Claeys, X., Hiptmair, R., Jerez-Hancknes, C., Pintarelli, S.: Novel multi-trace boundary integral equations for transmission boundary value problems. In: *Unified Transform for Boundary Value Problems: Applications and Advances*, pp. 227–258. SIAM (2015)
8. Dahlke, S., Harbrecht, H., Utzinger, M., Weimar, M.: Adaptive wavelet BEM for boundary integral equations: theory and numerical experiments. *Numer. Functional Anal. Optim.* **39**(2), 208–232 (2018)
9. Greengard, L., Rokhlin, V.: A fast algorithm for particle simulations. *J. Comput. Phys.* **135**(2), 280–292 (1997)
10. Karypis, G., Kumar, V.: A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM J. Sci. Comput.* **20**(1), 359–392 (1999)
11. Kravčenko, M., Maly, L., Merta, M., Zapletal, J.: Parallel assembly of ACA BEM matrices on Xeon Phi clusters. In: Wyrzykowski, R., Dongarra, J., Deelman, E., Karczewski, K. (eds.) *PPAM 2017. LNCS*, vol. 10777, pp. 101–110. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-78024-5_10
12. Kravčenko, M., Merta, M., Zapletal, J.: Distributed fast boundary element methods for Helmholtz problems. *Appl. Math. Comput.* **362**, 124503 (2019)
13. Lukáš, D., Kovář, P., Kovářová, T., Merta, M.: A parallel fast boundary element method using cyclic graph decompositions. *Numer. Algorithms* **70**(4), 807–824 (2015)
14. Merta, M., Zapletal, J.: BEM4I. IT4Innovations, VŠB - Technical University of Ostrava, 17. listopadu 2172/15, 708 00 Ostrava-Poruba, Czech Republic (2013). <http://bem4i.it4i.cz/>
15. Of, G.: Fast multipole methods and applications. In: Schanz, M., Steinbach, O. (eds.) *Boundary Element Analysis. Lecture Notes in Applied and Computational Mechanics*, vol. 29, pp. 135–160. Springer, Berlin Heidelberg (2007). https://doi.org/10.1007/978-3-540-47533-0_6
16. Rjasanow, S., Steinbach, O.: *The Fast Solution of Boundary Integral Equations. Mathematical and Analytical Techniques with Applications to Engineering*. Springer, Heidelberg (2007). <https://doi.org/10.1007/0-387-34042-4>
17. Rjasanow, S., Weggler, L.: Matrix valued adaptive cross approximation. *Math. Methods Appl. Sci.* **40**(7), 2522–2531 (2017)
18. Sauter, S.A., Schwab, C.: *Boundary Element Methods. Springer Series in Computational Mathematics*. Springer, Heidelberg (2010). <https://doi.org/10.1007/978-3-540-68093-2>
19. Steinbach, O.: *Numerical Approximation Methods for Elliptic Boundary Value Problems: Finite and Boundary Elements. Texts in Applied Mathematics*. Springer, New York (2008). <https://doi.org/10.1007/978-0-387-68805-3>
20. Zapletal, J., Merta, M., Malý, L.: Boundary element quadrature schemes for multi- and many-core architectures. *Comput. Math. Appl.* **74**(1), 157–173 (2017). 5th European Seminar on Computing ESCO 2016
21. Zapletal, J., Of, G., Merta, M.: Parallel and vectorized implementation of analytic evaluation of boundary integral operators. *Engineering Analysis with Boundary Elements* **96**, 194–208 (2018)

Open Access This chapter is licensed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.