BEM4I: A massively parallel boundary element solver

Michal Merta
IT4Innovations National Supercomputing Center
Ostrava, Czech Republic
michal.merta@vsb.cz

Jan Zapletal
IT4Innovations National Supercomputing Center
Ostrava, Czech Republic
jan.zapletal@vsb.cz

Michal Kravcenko
IT4Innovations National Supercomputing Center
Ostrava, Czech Republic
michal.kravcenko@vsb.cz

Lukas Maly
IT4Innovations National Supercomputing Center
Ostrava, Czech Republic
lukas.maly@vsb.cz

ABSTRACT

In this work we present a library of parallel solvers based on the boundary element method (BEM). We provide a brief description of BEM and its parallelization, focus on SIMD vectorization and shared- and distributed-memory parallelization by OpenMP and MPI, respectively. Two approaches for distributed parallelization of BEM are discussed – either based on a novel parallel adaptive cross approximation (ACA) method, or on the boundary element tearing and interconnecting (BETI) domain decomposition method. To demonstrate the efficiency of the library we provide results of numerical experiments on the Xeon and Xeon Phi based clusters.

KEYWORDS

boundary element method, adaptive cross approximation, BETI, parallel computing, SIMD vectorization

ACM Reference format:
Michal Merta, Jan Zapletal, Michal Kravcenko, and Lukas Maly. 2017. BEM4I: A massively parallel boundary element solver. In *Proceedings of Supercomputing 2017 conference, Denver, Colorado USA, November 2017 (SC17),* 2 pages.
https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

The boundary element method [9] is a counterpart to the popular finite element method. Its main advantage is the reduction of the considered problem to the boundary of the computational domain which makes it well suited for problems stated on unbounded domains (such as electromagnetic or acoustic wave scattering) or shape optimization problems. However, the classical BEM produces full matrices and is of quadratic computational complexity with respect to the number of surface DOFs. Moreover, one has to take a special care of the singularities occurring in the numerical integration which leads to a high computational intensity of the integration routines.

An efficient parallel implementation of BEM is necessary to enable solution of large scale real-world problems. The BEM4I library developed at IT4Innovations National Supercomputing Center provides an implementation of BEM parallelized on several levels. On the lowest layer the integration routines are SIMD vectorized using OpenMP pragmas, the system matrix assembly is parallelized in the shared memory by OpenMP threading and the whole procedure can be distributed among computational nodes using MPI. Moreover, assembly of the system matrices can be accelerated by the Intel Xeon Phi coprocessors. The library currently supports solution of problems modelled by the Laplace, Lamé, Helmholtz and time-dependent wave equations.

The contribution has the following structure. After a brief introduction to BEM and the BEM4I library we discuss its vectorization and parallelization in the shared memory. Next, we describe two approaches for parallelization in the distributed memory – one based on the novel parallelization of the adaptive cross approximation (ACA), the second based on the boundary element tearing and interconnecting (BETI) domain decomposition method. Numerical experiments demonstrating the efficiency of the implementation are provided in each section.

2 INTRA-NODE PARALLELIZATION

Implementation of BEM has to deal with integrals of the type

$$[\mathsf{V}_h]_{\ell,k} := \frac{1}{4\pi} \int_{\tau_\ell} \int_{\tau_k} \frac{1}{\|\boldsymbol{x}-\boldsymbol{y}\|} \,\mathrm{d}s_{\boldsymbol{y}} \,\mathrm{d}s_{\boldsymbol{x}}\,.$$

To treat the singularities occurring in the integrals one can either use a fully numerical regularized Gaussian quadrature [8] or semi-analytical approach where the inner integral is treated analytically while the outer one numerically [9]. Both of the approaches lead to a large computational intensity of the quadrature routines – the former case requires evaluation of 4-dimensional integrals with large number of quadrature points, while the latter leads to a frequent evaluation of expensive operations (floating point division, square root, logarithm, or arctangent). Therefore, the assembly of the system matrices is usually one of the most time-consuming parts of a BEM simulation and in BEM4I we pay significant attention to the parallelization and acceleration of the process. Besides the shared-memory threading by OpenMP, the computationally intensive quadrature routines are vectorized by OpenMP SIMD pragmas. The original code was optimized by, e.g., conversion of arrays of structures (AoS) to structure of arrays (SoA), memory alignment, data padding, or manual loop collapsing. In [10, 11] we describe an efficient implementation and provide numerical experiments on Xeon and Xeon Phi (KNC and KNL) architectures for fully numerical and semi-analytical integration. Vectorization by the Vc library is described in [5]. In [6] we describe acceleration of the system matrix assembly by offloading the computation to the KNC coprocessors.

SC17, November 2017, Denver, Colorado USA
2017. ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

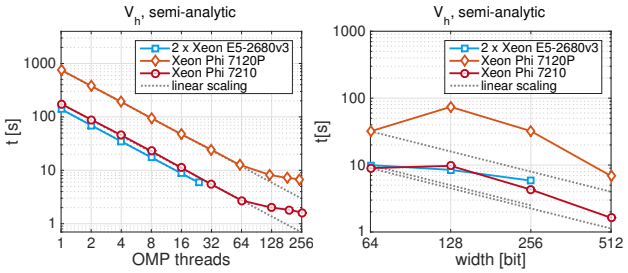

Figure 2.1: Scalability in shared memory (left) and with respect to the SIMD vector length (right).

In Figure 2.1 (left) we provide the results of the scalability tests for the system matrix assembly in shared memory. We obtain perfect scaling up to the physical number of cores. Scaling with respect to the length of the SIMD vector is shown in the right-hand side of Figure 2.1.

3 DISTRIBUTED ACA METHOD

To reduce the memory requirements and computational complexity of BEM to the almost linear case, one can leverage the ACA method [1]. However, its parallel implementation is not straightforward as one has to deal with load balancing issues. In [4], a new approach for parallelization of ACA was presented. The original boundary mesh is decomposed into N submeshes, pairs of which define blocks in the system matrix. The related $N \times N$ submatrices are assigned to N concurrent processes. Each of the blocks is assembled as a separate low-rank ACA matrix. The distribution minimizing the number of submeshes assigned to a single MPI process is defined using cyclic decompositions of undirected complete graphs.

Performance of the method for the Laplace equation is demonstrated in Table 3.1 on meshes with up to 55.6 million surface elements (which translates to hundreds of millions volume elements). The method enables the assembly of the system matrices in a reasonable time while compressing their size down to 0.08 – 0.19 %. Initial results on a KNL-based cluster are presented in [2].

4 BETI DOMAIN DECOMPOSITION METHOD

Another approach for distributed-memory parallelization of BEM is the boundary element tearing and interconnecting method (BETI) which is a counterpart to the well-known FETI method [3]. BETI is a non-overlapping domain decomposition method – the original mesh is decomposed into smaller submeshes and the global continuity is enforced by Lagrange multipliers. In addition to parallelization, the method enables solution of problems with materials which are only piecewise homogeneous.

BETI in BEM4I is enabled via an interface to the domain decomposition library Espreso [7]. While BEM4I assembles local Dirichlet-to-Neumann maps (Steklov-Poincaré operators), Espreso takes care of the gluing of the subdomains and the global solution process. Using BETI we are able to scale up to 864 Salomon nodes and solve problems with 7.7 billion surface elements.

# surface elements	24.7 million	55.6 million
# nodes	64	256
# MPI ranks	128	512
assembly V_h/K_h [s]	163.2/280.5	242.8/414.5
compression V_h/K_h [%]	0.081/0.163	0.092/0.185
total RAM [GB]	3932/3987	22688/22873

Table 3.1: Distributed assembly of BEM matrices V_h and K_h on the Salomon supercomputer at IT4Innovations.

5 CONCLUSION

We have presented a library of boundary element solvers. It is parallelized on several levels, starting from SIMD vectorization, to shared-memory parallelization, and distribution among multiple computational nodes. It is well-suited for problems on unbounded domains, such as modelling of wave propagation. Moreover, due to its high computational intensity and coalesced memory access pattern, the method can be efficiently employed on modern many-core systems with wide SIMD registers.

ACKNOWLEDGEMENTS

The work was supported by The Ministry of Education, Youth and Sports from the National Programme of Sustainability (NPU II) project 'IT4Innovations excellence in science – LQ1602', from the Large Infrastructures for Research, Experimental Development and Innovations project 'IT4Innovations National Supercomputing Center – LM2015070' and by VŠB – Technical University of Ostrava under the grant SP2017/165.

REFERENCES

[1] M. Bebendorf. 2000. Approximation of boundary element matrices. *Numer. Math.* 86 (2000), 565–589.
[2] M. Kravcenko, L. Maly, M. Merta, and J Zapletal. 2017. Parallel boundary element method for many-core architecture. *Lecture Notes in Computer Science* (2017), Accepted. PPAM17 Conference.
[3] U. Langer and O. Steinbach. 2003. Boundary Element Tearing and Interconnecting Methods. *Computing* 71, 3 (01 Nov 2003), 205–228. https://doi.org/10.1007/s00607-003-0018-2
[4] D. Lukas, P. Kovar, T. Kovarova, and M. Merta. 2015. A parallel fast boundary element method using cyclic graph decompositions. *Numerical Algorithms* 70, 4 (2015), 807–824. https://doi.org/10.1007/s11075-015-9974-9
[5] M. Merta and J. Zapletal. 2015. Acceleration of boundary element method by explicit vectorization. *Advances in Engineering Software* 86 (2015), 70–79. https://doi.org/10.1016/j.advengsoft.2015.04.008
[6] M. Merta, J. Zapletal, and J. Jaros. 2016. Many core acceleration of the boundary element method. *Lecture Notes in Computer Science* 9611 (2016), 116–125. https://doi.org/10.1007/978-3-319-40361-8_8
[7] L. Riha, T. Brzobohaty, A. Markopoulos, O. Meca, and T. Kozubek. 2016. Massively Parallel Hybrid Total FETI (HTFETI) Solver. In *Proceedings of the PASC Conference (PASC16)*. ACM, New York, NY, USA, Article 7, 11 pages. https://doi.org/10.1145/2929908.2929909
[8] S. Sauter and C. Schwab. 2010. *Boundary Element Methods.* Springer Berlin Heidelberg, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-68093-2_4
[9] O. Steinbach. 2008. *Numerical Approximation Methods for Elliptic Boundary Value Problems: Finite and Boundary Elements.* Springer, New York.
[10] J. Zapletal, M. Merta, and L. Maly. 2017. Boundary element quadrature schemes for multi- and many-core architectures. *Computers and Mathematics with Applications* 74, 1 (2017), 157–173. https://doi.org/10.1016/j.camwa.2017.01.018
[11] J. Zapletal, G. Of, and M. Merta. 2017. Parallel and vectorized implementation of analytic evaluation of boundary integral operators. *Computer Methods in Applied Mechanics and Engineering* (2017). Submitted.