A Parallel Fast BEM on Distributed Memory Systems for the Helmholtz Equation as an Extension of SPECFEM3D

Dalibor Lukas, Petr Kovan, Tereza Kovarova, Jan Zapletal

Department of Applied Mathematics, IT4Innovations, VSB-Technical University of Ostrava, Czech Republic

Abstract

In this paper, a new parallel acoustic simulation package has been created, using the boundary element method (BEM). The package is built on top of SPECFEM3D, which is parallel software for doing seismic simulations, e.g. earthquake simulations of the globe. The acoustical simulation relies on a Fourier transform of the seismic elastodynamic data, resulting from SPECFEM3D_GLOBE, which are then postprocessed by a sequence of solutions to Helmholtz equations, in the exterior of the globe. For the acoustic simulations BEM has been employed, which reduces computation to the sphere; however, its naive implementation suffers from quadratic time and memory complexity, with respect to the number of unknowns. To overcome the latter, the method was accelerated by using hierarchical matrices and adaptive cross approximation techniques, which is referred to as Fast BEM. First, a hierarchical clustering of the globe surface triangulation is performed. The arising cluster pairs decompose the fully populated BEM matrices into a hierarchy of blocks, which are classified as far-field or near-field. While the near-field blocks are kept as full matrices, the far-field blocks are approximated by low-rank matrices. This reduces the quadratic complexity of the serial code to almost linear complexity, i.e. $O(n \log(n))$, where $n$ denotes the number of triangles. Furthermore, a parallel implementation was done, so that the blocks are assigned to concurrent MPI processes with an optimal load balance. The novelty of our approach is based on a nontrivial and theoretically supported memory distribution of the hierarchical matrices and right-hand side vectors so that the overall memory consumption leads to $O(n \log(n)/N+n/sqrt(N))$, which is the theoretical limit at the same time.

1. Introduction

The boundary element method (BEM) is an efficient technique for the simulation of physical fields in homogeneous media, relying on partial differential equations. In particular, we are interested in acoustics in unbounded domains, in which case the use of BEM is theoretically proven [4] to be superior to volume discretization techniques. The developed package serves as a post-processing stage to software SPECFEM3D_GLOBE, which solves seismic simulations of the Earth. The package is a new contribution, rather than an improvement, to SPECFEM3D_GLOBE.

Though boundary integral equations were discovered long before the variational approach leading to the methods of finite elements (FEM), for a long time they were not suitable for large scale computations due to the fact that they lead to fully-populated matrices, i.e. to complexity $O(n^2)$, where $n$ stands for the number of boundary triangles. The reveal of BEM dates back to Greengard and Rokhlin [1], who developed the Fast Multipole Method. Their method hierarchically clusters the geometrical domain under consideration into near and far-field couples of clusters and replaces the integral kernel of the far-field by proper low-order expansions, e.g. spherical harmonics. Another important class of Fast BEM relies on hierarchical matrices that were introduced by Hackbusch and Nowak [2]. Instead of the integral kernel, far-field parts of the matrix are hierarchically approximated by matrices of low ranks, which are typically constructed by interpolation. Recently, Bebendorf [3] has proposed the adaptive cross approximation (ACA) method. ACA is a black-box method that interpolates the far-field submatrices without any
knowledge of the related kernel, for which reason it has become a very frequently used variant of Fast BEM. All the mentioned methods reduce the complexity of the matrix assembling as well as solving to related linear systems to $O(n \log(n))$.

Yet a parallel implementation of Fast BEM has been an issue. While a parallel distribution of the computation is rather straightforward, to our best knowledge it has only been done on shared memory systems. We can easily distribute the storage of the compressed dense matrices so that it requires $O(n \log(n)/N)$ memory per core, where $N$ denotes the number of cores. However, yet it has not been considered how to distribute the boundary mesh data, which are needed by the assembling process. By using novel results in decomposition of complete graphs, we propose a technique, which leads to complexity $O(n \log(n)/\sqrt{N})$ for certain $N$, which is the best possible at the same time. The rest of the paper is organized as follows: In Section 2 we describe BEM applied to Helmholtz Equation. In Section 3 sparsification techniques of hierarchical matrices and ACA are recalled. In Section 4, some decompositions of complete graphs are given as well as their relevance to memory distribution of geometrical data. Finally, in Section 5 numerical results document the theoretical parallel scalability.

2. BEM for Helmholtz Equation

We consider the following exterior Helmholtz problem with the prescribed Neumann datum:

$$\begin{align*}
-\triangle p(x) - \kappa^2 p(x) &= 0 & \text{in } \Omega^e := \mathbb{R}^3 \setminus \overline{\Omega}, \\
\partial p(x)/\partial n(x) &= v_n(x) := \frac{\omega^2 \rho u(x) \cdot n(x)}{\omega^{-2} |x|^{-2}} & \text{on } \Gamma, \\
|\nabla p(x) \cdot x/|x| - i \kappa p(x)| &= O(|x|^{-2}) & \text{as } |x| \to \infty.
\end{align*}$$

where $\Gamma$ denotes the boundary of $\Omega$. The resulting pressure $p$ is a complex valued function, $\kappa := \omega/c$ stands for the wave number, $\omega$ denotes the angular frequency, $c$ denotes the speed of sound, $n$ denotes the unit outer normal to the domain $\Omega$, and $u$ denotes the related component of the displacement field in the frequency domain. Note that, in case of SPECFEM3D GLOBE we shall first translate the resulting time-dependent displacement field to the frequency domain, solve a sequence of Helmholtz problems, and finally transfer the results back to the time-domain.

We shall make use of the following fundamental solution:

$$P(x, y) := \frac{e^{i \kappa r}}{4 \pi r}, \quad x \in \Omega^e$$

We consider the direct boundary integral approach, which leads to the following ansatz:

$$p(x) = -\int_{\Gamma} v_n(y) P(x, y) \, dS(y) + \int_{\Gamma} p(y) \left( \partial P(x, y) / \partial n(y) \right) \, dS(y)$$

Applying the normal derivative to the latter, and using the Galerkin approach, we arrive at

$$\langle D_\kappa p, v \rangle \Gamma = \langle -1/2 I + K'_\kappa v_n, v \rangle \Gamma \text{ on } H^{1/2}(\Gamma)$$

with the following boundary integral operators:

$$\langle D_\kappa p, v \rangle \Gamma := \int_{\Gamma} \int_{\Gamma} P(x, y) \left[ (n(x) \times \nabla \tilde{v}(x)) \cdot (n(y) \times \nabla \tilde{p}(y)) - \kappa^2 n(x) v(x) \cdot n(y) p(y) \right] \, dS(y) \, dS(x)$$

$$\langle v_n, v \rangle \Gamma := \int_{\Gamma} v_n(x) v(x) \, dS(x), \quad \langle K'_\kappa v_n, v \rangle \Gamma := \int_{\Gamma} \int_{\Gamma} \left( \partial P(x, y) / \partial n(y) \right) v_n(y) v(x) \, dS(y) \, dS(x)$$
Furthermore, we triangulate the boundary into n triangles and replace the fractional Sobolev spaces by continuous piecewise linear functions for the pressure and piecewise constants for the velocity field. The inner integrals are evaluated analytically, which gets rid of singularities, while the other integration is efficiently calculated by Gauss quadrature. We refer to [4] for details. This leads to a dense system of linear equations.

3. Hierarchical matrices and ACA

The triangulation of the boundary is hierarchically decomposed into clusters as depicted in Fig. 1. The pairs of clusters are related to submatrices, which are classified as near or far-field, depending on a relation between the cluster sizes and their distance. While the near-field submatrices are kept fully-populated, the far-field submatrices are approximated by a low-rank format.

![Figure 1: Hierarchical clustering of the geometry (left), related hierarchical matrix (right)](image)

The method ACA subsequently chooses pivots in the residuum of the actual approximation of the far-field matrix, and then updates the approximation by a rank-1 matrix. It is a product of the pivoted row and column of the residual matrix. It can be viewed as an interpolation of the original far-field matrix to the pivot rows and columns, see the equation below. The approximation error is the spectral norm of the Schur complement. ACA in a combination with hierarchical matrices reduces the quadratic complexity to almost linear, i.e. $O(n \log n)$.

$$
\mathbf{P}_{C_x} \mathbf{A} \mathbf{P}_{C_y}^T =: \begin{pmatrix} \tilde{A}_{11} & \tilde{A}_{12} \\
\tilde{A}_{21} & \tilde{A}_{22} \end{pmatrix} \approx \begin{pmatrix} \tilde{A}_{11} & \tilde{A}_{12} \\
\tilde{A}_{21} & \tilde{A}_{21} \tilde{A}_{11}^{-1} \tilde{A}_{12} \end{pmatrix} = \begin{pmatrix} \tilde{A}_{11} \\
\tilde{A}_{21} \end{pmatrix} \begin{pmatrix} \tilde{A}_{11}^{-1} & \begin{pmatrix} \tilde{A}_{11} & \tilde{A}_{12} \end{pmatrix} \\
\tilde{A}_{21}^{-1} & \tilde{A}_{22} \end{pmatrix} \\
=: (\mathbf{u}_1, \ldots, \mathbf{u}_r) (\mathbf{v}_1, \ldots, \mathbf{v}_r)^T.
$$

4. Distribution of memory by decomposition of complete graphs

Our approach to a parallel implementation of ACA-BEM relies on a subdivision of the boundary mesh into $N$ pieces. Subsequently, matrix $\mathbf{A}$ is decomposed into $N$ by $N$ blocks. We aim to distribute the blocks to processes so that the numbers of triangles per process are minimized. Additionally, we assign the $i$-th diagonal block to the $i$-th process. This minimization problem can be formulated in terms of graph theory as a cyclic decomposition of a complete undirected graph. Consider $N$ vertices and $(N-1)N$ edges of such a graph. Then, for certain $N$ we are able to construct a subgraph on $O(\text{sqrt}(N))$ vertices and $N-1$ edges so that the original graph is reconstructed by $N$ rotations.
of the subgraph. In particular, so far we have found these decompositions for $N=2, 3, 7, 21, 31, 57, \text{ and } 91$ so that the related subgraphs have 2, 3, 5, 6, 8, and 10 vertices, respectively, which is an $O(\sqrt{N})$ function. We depict an example for $N=7$ on Fig. 2.

![Figure 2: Cyclic decomposition of a complete undirected graph on 7 vertices into triangles](image)

5. Parallel implementation and numerical results

We employ $N$ processes. Each of them reads numbers of the boundary mesh parts from the prepared cyclic decomposition. Then the related parts of mesh are loaded from disk and related $N$ blocks of the system matrix are assembled, each compressed by means of hierarchical matrices and ACA. When the assembling part is done, the processes are waiting for a signal from the master to contribute to the matrix-times-vector multiplication within GMRES iteration method for solution to a linear system of equations.

Table 1 and Figure 3 demonstrate the parallel scalability of the matrix assembling (left) and the related memory requirement (right) per core of the proposed method for the single-layer matrix of the Laplace operator on a cube. In the legend of Figure 3, the level of discretization $n$ and the corresponding matrix compression rate $c$ are indicated. The compression rates do not depend on numbers of employed cores. The computed results support theoretical computational complexity of matrix assembling $O(n \log(n)/N)$ as well as expected memory complexity $O(n \log(n)/N + n/\sqrt{N})$. Note that a comparable FEM discretization of the largest problem would lead almost to a billion of volume unknowns.
<table>
<thead>
<tr>
<th>n</th>
<th>compression</th>
<th>CPU [s], memory [MB] per core</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>N:=1</td>
</tr>
<tr>
<td>3072</td>
<td>19.6%</td>
<td>20s, 156MB</td>
</tr>
<tr>
<td>12288</td>
<td>10.8%</td>
<td>148s, 245MB</td>
</tr>
<tr>
<td>49152</td>
<td>5.0%</td>
<td>965s, 878MB</td>
</tr>
<tr>
<td>196608</td>
<td>1.7%</td>
<td>5676s, 3676MB</td>
</tr>
<tr>
<td>786432</td>
<td>0.7%</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 1: Efficiency of the parallel Fast BEM method for distributed memory system

![Figure 3: Parallel scalability of the matrix assembling (left) and the corresponding memory requirement (right) per core.](image)

6. Conclusion

In this paper we presented a novelty parallel method for fast BEM on distributed memory systems. We documented its performance up to almost a million of boundary elements and 31 computational cores. At the moment, we still debug our implementation. We have not posted it on the internet yet. It will take a quite effort to make it robust, stable, and user friendly.

Acknowledgements

This work was financially supported by the PRACE project funded in part by the EUs 7th Framework Programme (FP7/2007-2013) under grant agreement no. RI-26155.
References