Optimization and Highly Parallel Implementation of Domain Decomposition Based Algorithms

Lubomír Říha, Tomáš Brzobohatý, Alexandros Markopoulos, Marta Jarošová and Tomáš Kozubek
IT4Innovations National Supercomputing Center, VŠB-Technical University of Ostrava, Ostrava, Czech Republic
Email: {lubomir.riha, tomas.brzobohaty, alexandros.markopoulos, marta.jarosova, tomas.kozubek}@vsb.cz

***Abstract*—We describe an implementation and scalability results of a hybrid FETI (Finite Element Tearing and Interconnecting) solver based on our variant of the FETI type domain decomposition method called Total FETI. In our approach a small number of neighboring subdomains is aggregated into clusters, which results into a smaller coarse problem. Current implementation of the solver is focused on the optimal performance of the main CG solver, including: implementation of communication hiding and avoiding techniques for global communications; optimization of the nearest neighbor communication - multiplication with global gluing matrix; and optimization of the parallel CG algorithm to iterate over local Lagrange multipliers only. The performance is demonstrated on a linear elasticity synthetic 3D cube and real world benchmarks.**

I. Introduction

The goal of this paper is to describe a hybrid FETI method based on our variant of the FETI type domain decomposition method called Total FETI. The original FETI method, also called FETI-1 method, was originally introduced for the numerical solution of the large linear systems arising in linearized engineering problems by Farhat and Roux [1]. In both FETI methods a body is decomposed into several non-overlapping subdomains and the continuity between the subdomains is enforced by Lagrange multipliers. Using a theory of duality, a smaller and relatively well conditioned dual problem can be derived and efficiently solved by suitable variant of the conjugate gradient algorithm.

The original FETI algorithm, where only the favorable distribution of the spectrum of the dual Schur complement matrix [2] was considered, was efficient only for a small number of subdomains. So it was later extended by introducing a natural coarse problem [3], whose solution was implemented by auxiliary projectors so that the resulting algorithm became in a sense optimal [3].

Even if, there are several efficient coarse problem parallelization strategies [4], still there are size limitations of the coarse problem. So several hybrid (multilevel) methods were proposed [6], [5]. The key idea is to aggregate small number of neighboring subdomains into the clusters, which naturally results into the smaller coarse problem. In our Hybrid Total FETI, the aggregation of subdomains into the clusters is enforced again by Lagrange multipliers. Thus Total FETI method is used on both cluster and subdomain levels. This approach simplifies implementation of hybrid FETI methods and enables to extend parallelization of the original problem up to tens of thousands of cores due to less memory requirements. This is positive effect of reducing the coarse space. The negative one is getting worse convergence rate comparing with the original TFETI. To improve it the transformation of basis originally introduced by Klawonn and Widlund [8], Klawonn and Rheinbach [7], and Li and Widlund [9] is applied to the derived hybrid algorithm.

II. Hybrid FETI Solver

Our implementation of the Hybrid FETI solver is implemented in pure C++. Significant part of the development effort was devoted to development of a C++ wrapper for (1) the selected sparse and dense BLAS routines and (2) the sparse direct solvers (MKL version of PARDISO direct solver) of the Intel MKL library. Since the solver development is focused on the future multi and many core architectures, in particular the Intel MIC architecture, the Intel MKL library is the only external tool that our solver uses. In addition, to be able to port the solver to Intel MIC the Intel compiler and Intel MPI are used.

a) Communication Layer Optimization: The solver uses hybrid parallelization suited for multi-socket and multi-core compute nodes as this is the architecture of most of today's supercomputers.

The first level of parallelization is designed for parallel processing of the clusters of subdomains. Each cluster must be assigned to a single node but if necessary multiple clusters can be processed per one node. This distributed memory parallelization is done using MPI. In particular we are using MPI standard 3.0 (implemented in the Intel MPI 5.0 Beta) because the communication hiding techniques implemented in our FETI communication layer require the non-blocking collective operations. In the future we would like to improve the communication layer to use GASPI's PGAS communication model.

The essential part of is a communication layer as it is identical whether the solver runs in FETI or Hybrid FETI mode. It uses novel communication avoiding and hiding techniques for the main iterative solver. In particular we have implemented: (1) the Pipelined Conjuagent Gradient (PipeCG) solver - hides communication of the global dot products in behind the local matrix vector multiplications; (2) the coarse problem solver using distributed inverse matrix - merges two global communication operations (Gather and Scatter) into one (AllGather) and parallelizes the coarse problem processing; and (3) the optimized version of global gluing matrix multiplication (matrix B for FETI and B1 for Hybrid FETI) - written as stencil communication which is fully scalable.

The stencil communication for simple decomposition into four subdomains is shown in the poster where the Lagrange

Multipliers (LMs) that connects different neighboring subdomains are depicted in different colors. In every iteration when the LMs are updated an exchange is performed between the neighboring subdomains to finish the update. This affinity also controls the distribution of the data for the main distributed iterative solver, which iterates over local LMs only. In our implementation each MPI process modifies only those elements of the vectors used by the CG solver that match the LMs associated with the particular domain in case of FETI or the set of domains in a cluster in case of Hybrid FETI.

b) Inter-cluster Processing: The second level of parallelization is designed for parallel processing of subdomains in a cluster. Our implementation enables over subscription of CPU cores therefore each core can process multiple subdomains and therefore the size of the cluster is not limited by the hardware configuration. This shared memory parallelization is implemented using Intel Cilk+. We have chosen the Cilk+ due to its advanced support for C++ language. In particular we are taking advantage of the functionality that allows to create custom parallel reduction operations on top of the C++ objects which in our case are sparse matrices.

III. Results

c) Synthetic 3D cube benchmark: The performance of the solver has been measured in both Hybrid FETI and FETI mode to evaluate the scalability of both methods. The focus at this development stage is to minimize the time per iteration.

On the poster is shown the weak scalability of the solver on a synthetic 3D cube benchmark where domain size of $3 \cdot (5+1)^3 = 648$ DOFs are fixed. The small size of the subdomains helps observe the behavioral of the FETI bottlenecks and the effect of the (1) Pipelined CG algorithm (in the graph identified as a "CG with 1 reduction"), (2) use of the distributed inverse matrix of coarse problem (GGTINV) and the (3) Hybrid FETI (HFETI) method on these bottlenecks. Following observations can be made from the results: (1) Pipelined CG helps both FETI and Hybrid FETI method; (2) using GGTINV helps even more for both methods; (3) Hybrid FETI iteration is faster than FETI iteration for more then 512 subdomains without GGTINV and for more than 1000 subdomains with GGTINV.

d) Real world benchmark - Engine 2.5 millions DOFs: The second benchmark is a 2.5 milion DOF model of a car engine. Using this benchmark we have evaluated the behavioral of the communication layer during a strong scaling test. We have run the benchmark decomposed into 32 to 1024 subdomains in FETI mode. During all tests 8 MPI processes/subdomains have been assigned per node using only half of the CPU cores of each processor. This work distribution was selected to increase the memory bandwidth per subdomains.

On the poster is shown the single iteration time of the solver running FETI method on 32 to 1024 MPI processes. It can be seen that for up to 1024 MPI processes our solver exhibits the super-linear scaling showing the efficiency of the communication layer. Also to be able to continue scaling from 512 to 1024 cores.

IV. Conclusion

Current implementation of the solver is focused primarily on the optimization of the main iteration loop, including: implementation of communication hiding and avoiding techniques for global communications; optimization of the nearest neighbor communication - multiplication with global gluing matrix; optimization of the parallel CG algorithm to iterate only over local Lagrange multipliers. In other words: make a highly scalable FETI solver to achieve (if possible) super linear scaling. We have also implemented parallel routines for inter cluster processing used by the Hybrid FETI method. Cluster processing is distributed among the CPU cores (shared memory model) only and exploits two levels of parallelism: the data parallelism - delivered by the MKL library and the task parallelism - achieved using Intel Cilk over multiple subdomains in a cluster.

Acknowledgment

This paper has been elaborated in the framework of the project New creative teams in priorities of scientific research , reg. no. CZ.1.07/2.3.00/30.0055, supported by Operational Programme Education for Competitiveness and co-financed by the European Social Fund and the state budget of the Czech Republic and by Grant Agency of the Czech Republic GAČR grant 13-30657P.
The work was also supported the European Regional Development Fund in the IT4Innovations Centre of Excellence project (CZ.1.05/1.1.00/02.0070) and the Project of major infrastructures for research, development and innovation of Ministry of Education, Youth and Sports with reg. num. LM2011033, and by the project EXA2CT funded from the EU's Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 610741.
We thank SURFsara (www.surfsara.nl) for the support in using the Cartesius supercomputer.

References

[1] C. Farhat, F-X. Roux, "An unconventional domain decomposition method for an efficient parallel solution of large-scale finite element systems", SIAM J. Sci. Stat. Comput. 13: 379–396, 1992.

[2] F-X. Roux, "Spectral analysis of interface operator", Proceedings of the 5th Int. Symp. on Domain Decomposition Methods for Partial Differential Equations, ed. D. E. Keyes et al., SIAM, Philadelphia 1992; 73–90.

[3] C. Farhat, J. Mandel, F-X. Roux, "Optimal convergence properties of the FETI domain decomposition method", Comput. Methods Appl. Mech. Eng. 115, 1994; 365–385.

[4] T. Kozubek, V. Vondrak, M. Mensik, D. Horak, Z. Dostal, V. Hapla, P. Kabelikova, M. Cermak, "Total FETI domain decomposition method and its massively parallel implementation", Advances of Engeneering Software, accepted

[5] A. Klawonn, O. Rheinbach, "Highly scalable parallel domain decomposition methods with an application to biomechanics", ZAMM, 90, No. 1, pp.5-32, 2010

[6] Jungho Lee: "A hybrid domain decomposition method and its applications to contact problems in mechanical engineering", PhD thesis, New York University, 2009.

[7] A. Klawonn and O. Rheinbach, "A parallel implementation of Dual-Primal FETI methods for three dimensional linear elasticity using a transformation of basis", SIAM J. Sci. Comput., 28(5):1886–1906, 2006

[8] A. Klawonn and O. B. Widlund, "Dual-primal FETI methods for linear elasticity", Communications on pure and applied mathematics, 59(11):1523–1572, 2006

[9] J. Li and O. B. Widlund, "FETI-DP, BDDC, and block Cholesky methods", International journal for numerical methods in engineering, 66:250–271, 2006