Parallel FETI Solver for Modern Architectures

Lubomír Říha, Tomáš Brzobohatý, Michal Merta, Alexandros Markopoulos, Ondřej Meca, Tomáš Kozubek
IT4Innovations National Supercomputing Center, Ostrava, Czech Republic
Email: lubomir.riha@vsb.cz

***Abstract*—This poster presents a parallel implementation of the Hybrid Total FETI (HTFETI) method. HTFETI is a multilevel domain decomposition method in which a small number of neighboring subdomains is aggregated into clusters. The presented HTFETI solver is able to solve 3D heat transfer problems of size up to 124 billion unknowns on 17,576 nodes of the Titan supercomputer. A superlinear strong scalability of the iterative solver is presented for 20 billion unknown heat transfer problems and 11 billion DOF structural mechanics problem executed on 2,744 to 17,576 nodes. In addition, a super-linear strong scaling is presented for 300 million DOF real world problem.**

The acceleration of the HTFETI method using Local Schur Complement method delivers 7.8 speedup (heat transfer) and 2.7 speedup (structural mechanics) of the iterative solver when executed on the Intel Xeon Phi and 3.4 speedup for structural mechanics delivered by the GPUs on the Titan machine.

I. Introduction

For many years, the Finite Element Tearing and Interconnecting method (FETI) [2], [3] has been successfully used in the engineering community for solving very large problems arising from the discretization of partial differential equations. In such an approach the original structure is decomposed into several non-overlapping subdomains. Mutual continuity of primal variables between neighboring subdomains is enforced afterwards by dual variables, i.e., Lagrange multipliers (LM). They are usually obtained iteratively by one of the Krylov subspace methods, then the primal solution is evaluated locally for each subdomain.

In 2006 Dostál et al. [1] introduced a new variant of an algorithm called Total FETI (or TFETI) in which Dirichlet boundary condition is enforced also by LM.

The HTFETI method is a variant of hybrid FETI methods introduced by Klawonn and Rheinbach [4] for FETI and FETI-DP. In the original approach a number of subdomains is gathered into clusters. This can be seen as a three-level domain decomposition approach. Each cluster consists of number of subdomains and for these, a FETI-DP system is set up. The clusters are then solved by a traditional FETI approach using projections to treat the non trivial kernels. In contrast, in HTFETI, a TFETI approach is used for the subdomains in each cluster and the FETI approach with projections is used for clusters.

The main advantage of HTFETI is its ability to solve problems decomposed into a very large number of subdomains. We have ran tests with over 21 million subdomains organized into 17,576 clusters. This means two things: (i) an extremely large problems can be solved (over 120 billion DOF); (ii) moderate size problems (up to few billion DOF) can be decomposed into very small subdomains which improves memory, computational and numerical efficiency. This approach is used for the real world problems.

II. ESPRESO Library

ESPRESO library is a combination of Finite Element (FEM) and Boundary Element (BEM) tools and TFETI/HTFETI solvers. It supports FEM and BEM (uses BEM4I library) discretization for Advection-diffusion equation, Stokes flow and Structural mechanics. Real engineering problems are imported from Ansys Workbench or OpenFOAM. A C API allows ESPRESO to be used as a solver library for third party application. It has been used for integration with CSC ELMER. For large scale tests library also contains a multiblock benchmark generator. The postprocessing and vizualization is based on the VTK library and Paraview including Paraview Catalyst for inSitu vizualization.

ESPRESO solver is a parallel linear solver which includes highly efficient MPI communication layer [5] designed for massively parallel machines with thousands of compute nodes. The parallelization inside node is done using Cilk++. Three versions of the solver are developed: (i) *ESPRESO CPU* uses sparse matrices and sparse direct solvers to process the system matrices; (ii) *ESPRESO MIC* is an Intel Xeon Phi accelerated version which works with both sparse and dense representation of system matrices; and (iii) *ESPRESO GPU* is a GPU accelerated version which supports dense structures only. Support for sparse structures using cuSolver is under development.

All version can solve both linear (conjugate gradient (CG) solver) and nonlinear systems (GMRES and BiCGStab).

A. Scalability of the HTFETI Method

1) Cube Benchmark: is a tool in the ESPRESO library that generates a uniformly meshed cube problems of size controlled by a number of (i) clusters, (ii) subdomains per cluster and (iii) elements per subdomain. The generator is used for large scalability tests. The HTFETI solver achieves superliner strong scalability for both structural mechanics (11 billion DOF) and heat transfer (20 billion unknowns) benchmarks when executed on 2,744 to 17,576 clusters/compute nodes. For the weak scalability evaluation we have solved up to 124 billion unknown heat transfer problem. Weak scalability shows flattening characteristic all the way to 17,576 nodes. The processing increase at the beginning (from 8 to 512 nodes) is due to numerical scalability i.e. growing number of iterations (from 39 to 47). More details can be found in [7].

2) Real World Problems: are generated by ANSYS Workbench. We present structural mechanics problem of size 300 million DOF is decomposed into 512, 1024, 2048, 4096 and 8192 clusters using our FEM tools. Each cluster is processed by a single MPI process with 22 MPI processes per node of the IT4Innovations Salomon supercomputer. The results show a superliner scalability of the HTFETI method achieved by the combination of parallel scalability of our implementation and the numerical scalability of the HTFETI method.

B. Support for Modern Architectures

The key component for the efficient usage of many-core accelerators is the Local Schur Complement (LSC) method which converts the sparse system matrices, originally processed by a sparse direct solver, into dense SC matrices. The the most time consuming operation of the iterative FETI solver is the forward and backward substitutions. This operation, which can take up to 95% of the iterative solver runtime, is replaced by a more efficient dense matrix vector multiplication (GEMV). This dense BLAS level 2 operation has contiguous memory access and therefore yields significantly better performance for this memory bounded application.

The LSC method for the TFETI, described in [6], can be extended for HTFETI in a way that workload is split between CPU (all sparse BLAS routines) and the accelerator (the LSC processing). Therefore the overall processing rate of the iterative solver is not affected by the additional workload required by the HTFETI method.

The main bottleneck of the the LSC method is the preprocessing required to calculate the LSC. For this operation we use optimized algorithm for SC calculation implemented in the PARDISO sparse direct solver. This approach significantly reduces the processing time, but it is still 5 to 10 times slower the factorization (required if TFETI uses sparse system matrices) depending on the subdomains sizes. However, the preprocessing is performed only once during the initialization stage, while the processing of the LSC is performed tens to thousand times, depending on the number of iterations.

The weak scalability evaluation of the HTFETI with LSC exhibits similar behavioral as the non-accelerated version. For both GPU and Xeon Phi architectures the LSC is processed by the accelerator and the rest of the processing is done by a CPU. Firstly, we have compared 2x Xeon Phi 7120p accelerators to two 12core Haswell based Xeon E5-2680v3 CPUs. For this configuration Xeon Phi is 2.7 times faster for structural mechanics and 7.8 times faster for heat transfer when executed on 343 compute nodes. In case of the GPU acceleration we have used the Titan machine with 1x Tesla K20X GPU and 1x 16core AMD Opteron 6274. For this configuration the GPU is 3.4 times faster if SC is stored as general matrix. If SC is stored as symmetric matrix the speedup is only 1.4. The Intel Xeon Phi handles symmetric SC more efficiently.

Intel has recently introduce the second generation of Xeon Phi chips based on Knights Landing architecture. This architecture contains fast MCDRAM memory with bandwidth over 400GB/s for STREAM benchmark. Due to early access grated to us under the umbrella of the Intel PCC we have ported the ESPRESO library to KNL system and evaluated the benefits of the HBM memory for the HTFETI solver. The promising results show that single KNL chip (Xeon Phi 7210) with MCDRAM delivers the performance of the two Xeon Phi 7120p accelerators if LSC are stored in MCDRAM and remaining data structures are stored in DRAM. This is a major improvement.

III. Conclusion

This paper describes the HTFETI solver and its acceleration by selected modern many-core architectures. The solver achieves superliner strong scalability and flattening weak scalability for up 17,576 compute nodes of ORNL Titan machine. The acceleration of the HTFETI using GPGPU accelerators and Intel Xeon Phi coprocessors with the local Schur complement method shows significant speedups of the iterative CG solver. The new Xeon Phi (KNL) generation is two times faster then the first generation (KNC) due to fast MCDRAM memory.

Acknowledgment

This 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 and from the Large Infrastructures for Research, Experimental Development and Innovations project IT4Innovations National Supercomputing Center LM2015070.

This research used resources of the Oak Ridge Leadership Computing Facility at the Oak Ridge National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725.

This research has been supported by Intel Parallel Computing Center at IT4Innovations.

References

[1] Z. Dostál, D. Horák, and R. Kučera. Total FETI-an easier implementable variant of the FETI method for numerical solution of elliptic PDE. *Communications in Numerical Methods in Engineering*, 22(12):1155–1162, jun 2006.

[2] C. Farhat, J. Mandel, and F.-X. Roux. Optimal convergence properties of the FETI domain decomposition method. *Computer Methods in Applied Mechanics and Engineering*, 115:365–385, 1994.

[3] P. Gosselet and C. Rey. Non-overlapping domain decomposition methods in structural mechanics. *Archives of Computational Methods in Engineering*, 13(4):515–572, 2006.

[4] A. Klawonn and O. Rheinbach. Highly scalable parallel domain decomposition methods with an application to biomechanics. *ZAMM*, 90(1):5–32, jan 2010.

[5] L. Riha, T. Brzobohaty, A. Markopoulos, M. Jarosova, T. Kozubek, D. Horak, and V. Hapla. Implementation of the efficient communication layer for the highly parallel total feti and hybrid total feti solvers. *Parallel Computing*, 2016.

[6] L. Riha, T. Brzobohaty, A. Markopoulos, T. Kozubek, O. Meca, O. Schenk, and W. Vanroose. Efficient implementation of total feti solver for graphic processing units using schur complement. *Lecture Notes in Computer Science*, 9611:85–100, 2016.

[7] L. Riha, T. Brzobohaty, A. Markopoulos, O. Meca, and T. Kozubek. Massively parallel hybrid total feti (htfeti) solver. In *Proceedings of the Platform for Advanced Scientific Computing Conference*, PASC '16, New York, NY, USA, 2016. ACM.