Programs and Algorithms of Numerical Mathematics 18
J. Chleboun, P. Kůs, P. Přikryl, K. Segeth, J. Šístek, T. Vejchodský (Eds.)
Institute of Mathematics CAS, Prague 2017

PARALLEL STRATEGIES FOR SOLVING THE FETI COARSE PROBLEM IN THE PERMON TOOLBOX

Alena Vašatová[1], Jiří Tomčala[1], Radim Sojka[1], Marek Pecha[1,2], Jakub Kružík[1], David Horák[1,2], Václav Hapla[1], Martin Čermák[1]

[1] VSB-Technical University of Ostrava, IT4Innovations National Supercomputing Center
17. listopadu 15, Ostrava, 70833, Czech Republic
vaclav.hapla@vsb.cz, david.horak@vsb.cz
[2] VSB-Technical University of Ostrava, Dep. of Applied Mathematics
17. listopadu 15, Ostrava, 70833, Czech Republic

Abstract: PERMON (Parallel, Efficient, Robust, Modular, Object-oriented, Numerical) is a newly emerging collection of software libraries, uniquely combining Quadratic Programming (QP) algorithms and Domain Decomposition Methods (DDM). Among the main applications are contact problems of mechanics. This paper gives an overview of PERMON and selected ingredients improving scalability, demonstrated by numerical experiments.

Keywords: TFETI, quadratic programming, PERMON, PermonQP, PermonFLLOP, coarse problem

MSC: 65F05, 65M60, 65N55

1. Introduction

PERMON (Parallel, Efficient, Robust, Modular, Object-oriented, Numerical) [10], [12] is a newly emerging collection of software libraries, uniquely combining quadratic programming (QP) and Domain Decomposition Methods (DDM). There are two core modules in PERMON: PermonQP and PermonFLLOP. They are built on top of PETSc [3], [2], mainly its linear algebra part. They extend PETSc with new specific functionality, algorithms for large scale sparse QP problems and DDM of the Finite Element Tearing and Interconnecting (FETI) type. The same coding style is used so that users familiar with PETSc can utilize them with minimal effort. Among the main applications are contact problems of mechanics.

PermonQP provides a base for solution of quadratic programming (QP) problems. It includes data structures, transforms, algorithms, and supporting functions for QP. PermonQP is available for free under the FreeBSD open source license.

PermonFLLOP (FETI Light Layer on Top of PETSc) is an extension of PermonQP that adds support for DDM of the FETI type. PermonFLLOP is currently under preparation for publishing.

DOI: 10.21136/panm.2016.18

2. PermonQP

PermonQP, a general purpose QP solver, allows solving QPs with a symmetric positive semidefinite Hessian and any combination of linear equality and inequality constraints including unconstrained QP. It provides a basic framework for QP solution (data structures, transformations, and supporting functions), a wrapper of PETSc KSP linear solvers for unconstrained and equality-constrained QP, a wrapper of PETSc TAO optimization solvers [13] offering several additional algorithms for unconstrained and box-constrained QP, a variant of the augmented Lagrangian method called Semi-Monotonic Augmented Lagrangian with Bound and Equality (SMALBE), and several specific solvers for bound constrained minimization. General linear inequality constraints can be converted to bound constraints using dualization.

3. PermonFLLOP

PermonFLLOP (FETI Light Layer on Top of PETSc) is an extension of the PermonQP package, implementing the algebraic part of DDMs of the FETI type [7], [6], [5], [4]. Let us show how PermonFLLOP is implemented from the user's perspective. The domain has to be volume-meshed and decomposed using a partitioning software such as METIS [1]. Then virtually arbitrary Finite Element Method (FEM) implementation can be used to generate the subdomain stiffness matrices $\mathbf{K}^s$ and the subdomain load vectors $\mathbf{f}^s$ as sequential data for each subdomain Ω^s, $s = 1, \ldots, N_S$ independently. However, the local-to-global mapping $l2g$, mapping each subdomain's degrees of freedom to the global degrees of freedom, has to be produced in this phase.

Let us denote the number of processor cores used for the compuation by N_c. We assume here each processor core owns only one subdomain, $N_S = N_c$. PermonFLLOP has nevertheless a new experimental feature of allowing more than one subdomain per core, $N_S > N_c$, i.e. an array of $\mathbf{K}^s$ and $\mathbf{f}^s$ is passed per subdomain.

The "gluing" signed Boolean matrix $\mathbf{B}_g$ is constructed based on $l2g$ as described in [14]. The FEM software can skip the processing of the Dirichlet conditions and rather hand it over to PermonFLLOP, resulting in greater flexibility. PermonFLLOP allows to enforce Dirichlet boundary conditions either by the constraint matrix $\mathbf{B}_d$ (Total Finite Element Tearing and Interconnecting (TFETI) approach), or by a classical technique of embedding them directly into $\mathbf{K}^s$ and $\mathbf{f}^s$ (FETI-1 approach). It is also possible to mix these two approaches.

The inequality constraint matrix $\mathbf{B}_I$ describes linearized non-penetration conditions [5] on the contact zones. It is empty for linear (permanent contact only) problems. The global constraint right-hand side vector $\mathbf{c}$ possesses an analogous structure. Currently, PermonFLLOP requires $\mathbf{B}_I$ and $\mathbf{c}_I$ from the caller.

The subdomain nullspace matrix $\mathbf{R}^s$ is assembled using one of the following options. The first option is to use a numerical approach [8], and the second one is to generate $\mathbf{R}^s$ as rigid body modes from the mesh nodal coordinates [4]. The latter is typical for TFETI and is considered here.

Within PermonFLLOP, the local objects $\mathbf{K}^s$, $\mathbf{R}^s$ and $\mathbf{f}^s$ constitute the global distributed objects

$$\begin{aligned}\mathbf{K} &= \mathrm{diag}(\mathbf{K}^1, \ldots, \mathbf{K}^{N_S}),\\ \mathbf{R} &= \mathrm{diag}(\mathbf{R}^1, \ldots, \mathbf{R}^{N_S}),\\ \mathbf{f} &= [(\mathbf{f}^1)^T, \ldots, (\mathbf{f}^{N_S})^T]^T,\end{aligned}$$

where diag means a block-diagonal matrix consisting of the diagonal blocks in the parentheses.

In the PermonFLLOP's function `FllopSolve`, PermonFLLOP passes the global primal data $\mathbf{K}$, $\mathbf{f}$, $\mathbf{B}_E = \left[\mathbf{B}_g^T\ \mathbf{B}_d^T\right]^T$, $\mathbf{B}_I$ and $\mathbf{R}$ to PermonQP (Section 2), calls a specific series of QP transforms provided by PermonQP, resulting in the bound and equality constrained QP which is then solved with the `QPSSolve` function.

From the mathematical point of view, the called QP transforms (`QPT`) implement the following modifications. The original primal problem

$$\min \frac{1}{2}\mathbf{u}^T\mathbf{K}\mathbf{u} - \mathbf{f}^T\mathbf{u} \ \text{ s.t. } \ \mathbf{B}_I\mathbf{u} \le 0 \ \text{ and } \ \mathbf{B}_E\mathbf{u} = 0, \tag{1}$$

is transformed into the dual one by `QPTDualize`

$$\min \frac{1}{2}\boldsymbol{\lambda}^T\mathbf{F}\boldsymbol{\lambda} - \boldsymbol{\lambda}^T\mathbf{d} \ \text{ s.t. } \ \boldsymbol{\lambda}_I \ge 0 \ \text{ and } \ \mathbf{G}\boldsymbol{\lambda} = \mathbf{e}, \tag{2}$$

We use the standard notation

$$\mathbf{F} = \mathbf{B}\mathbf{K}^\dagger\mathbf{B}^T,\ \mathbf{G} = \mathbf{R}^T\mathbf{B}^T,\ \mathbf{d} = \mathbf{B}\mathbf{K}^\dagger\mathbf{f},\ \mathbf{e} = \mathbf{R}^T\mathbf{f},$$

with matrix $\mathbf{R}$, whose columns span the null space of $\mathbf{K}$ and represent rigid body or zero energy modes of subdomains, and $\mathbf{K}^\dagger$ denoting a generalized inverse of $\mathbf{K}$, i.e. a matrix satisfying $\mathbf{K}\mathbf{K}^\dagger\mathbf{K} = \mathbf{K}$. The constraint matrix $\mathbf{B} = \left[\mathbf{B}_I^T\ \mathbf{B}_E^T\right]^T$ can be constructed so that it has full rank, and then the Hessian $\mathbf{F}$ is positive definite with a relatively favourably distributed spectrum for application of the conjugate gradient method (CG).

The solution $\mathbf{u}$ can be evaluated by formula

$$\mathbf{u} = \mathbf{K}^\dagger(\mathbf{f} - \mathbf{B}^T\boldsymbol{\lambda}) + \mathbf{R}\boldsymbol{\alpha}. \tag{3}$$

Here,

$$\boldsymbol{\alpha} = -(\mathbf{R}^T\widetilde{\mathbf{B}}^T\widetilde{\mathbf{B}}\mathbf{R})^{-1}\mathbf{R}^T\widetilde{\mathbf{B}}^T\widetilde{\mathbf{B}}\mathbf{K}^\dagger(\mathbf{f} - \mathbf{B}^T\boldsymbol{\lambda})$$

denotes the vector of amplitudes, determining the contribution $\mathbf{R}\boldsymbol{\alpha}$ of the null space $\mathbf{R}$ to the solution $\mathbf{u}$. The matrix $\widetilde{\mathbf{B}}$ is defined as $\widetilde{\mathbf{B}} = \left[\widetilde{\mathbf{B}}_I^T\ \mathbf{B}_E^T\right]^T$ with $\widetilde{\mathbf{B}}_I$ formed by rows of $\mathbf{B}_I$ that correspond to the active constraints.

The problem of minimization on the subset of the affine space is transformed into the problem on subset of vector space by means of arbitrary $\widetilde{\boldsymbol{\lambda}}$ which satisfies $\mathbf{G}\widetilde{\boldsymbol{\lambda}} = \mathbf{e}$

while the solution is looked for in the form $\boldsymbol{\lambda} = \widehat{\boldsymbol{\lambda}} + \widetilde{\boldsymbol{\lambda}}$. The problem obtained by `QPTHomogenizeEq` then reads

$$\min \frac{1}{2}\widehat{\boldsymbol{\lambda}}^T \mathbf{F}\widehat{\boldsymbol{\lambda}} - \widehat{\boldsymbol{\lambda}}^T(\mathbf{d} - \mathbf{F}\widetilde{\boldsymbol{\lambda}}) \text{ s.t. } \widehat{\boldsymbol{\lambda}}_I \geq -\widetilde{\boldsymbol{\lambda}_I} \text{ and } \mathbf{G}\widehat{\boldsymbol{\lambda}} = 0. \tag{4}$$

Further improvement is based on the observation, that the augmented Lagrangian for problem (4) can be decomposed by orthogonal projectors

$$\mathbf{Q} = \mathbf{G}^T(\mathbf{G}\mathbf{G}^T)^{-1}\mathbf{G} \qquad \text{and} \qquad \mathbf{P} = \mathbf{I} - \mathbf{Q}$$

on the kernel of $\mathbf{G}$ and on the image space of $\mathbf{G}^T$: $\mathrm{Im}\mathbf{P} = \mathrm{Ker}\mathbf{G}$, $\mathrm{Im}\mathbf{Q} = \mathrm{Im}\mathbf{G}^T$. Evaluating $(\mathbf{G}\mathbf{G}^T)^{-1}$, i.e. solving the linear system

$$\mathbf{G}\mathbf{G}^T\mathbf{x} = \mathbf{y}, \tag{5}$$

is called the coarse problem (CP). The modified formulation of the problem (4), obtained by `QPTEnforceEqByProjector`, then takes the form

$$\min \frac{1}{2}\widehat{\boldsymbol{\lambda}}^T \mathbf{P}\mathbf{F}\mathbf{P}\widehat{\boldsymbol{\lambda}} - \widehat{\boldsymbol{\lambda}}^T\mathbf{P}\mathbf{d} \text{ s.t. } \widehat{\boldsymbol{\lambda}}_I \geq -\widetilde{\boldsymbol{\lambda}_I} \text{ and } \mathbf{G}\widehat{\boldsymbol{\lambda}} = 0. \tag{6}$$

More details can be found in [12].

4. Coarse problem

FETI methods blend iterative and direct solvers. The main loop solving the dual problem is implemented by an iterative solver, e.g. CG. In each iteration, auxiliary problems related to the application of an unassembled system operator are solved: (1) $\mathbf{K}^\dagger$ application and (2) CP solution.

Parallelization is achieved mainly by distributing diagonal blocks of $\mathbf{K}$ over processors, each block reflecting a subdomain. We strive to maximize the number of subdomains to reduce the sizes of the subdomain stiffness matrices, accelerating their factorization and $\mathbf{K}^\dagger$ actions. Furthermore, thanks to the FETI operator condition number estimate [6], decomposition into more subdomains maintaining a fixed discretization parameter h leads to reduction of the condition number of $\mathbf{K}$ and thus the number of iterations.

A drawback is the increasing null space dimension which decelerates the CP solution – it is a kind of a communicating vessels effect. The natural coarse space matrix $\mathbf{G}$ is computed so that each core owns the sparse sequential matrices $\mathbf{R}^s$ and $\mathbf{B}^s$, and computes the local horizontal block $\mathbf{G}^s = (\mathbf{R}^s)^T(\mathbf{B}^s)^T$ without any communication, $\mathbf{G} = [(\mathbf{G}^1)^T, \dots, (\mathbf{G}^{N_S})^T]^T$. The multiplication $\mathbf{G}\mathbf{G}^T = \mathbf{G} * \mathbf{G}^T$, factorization of $\mathbf{G}\mathbf{G}^T$, and the CP solutions (5) should be done in parallel, otherwise they form a computational and memory bottleneck. The sparsity pattern of $\mathbf{G}$ and $\mathbf{G}\mathbf{G}^T$ for the cube decomposed into 27 subdomains is illustrated in Fig. 1 and Fig. 2, respectively.

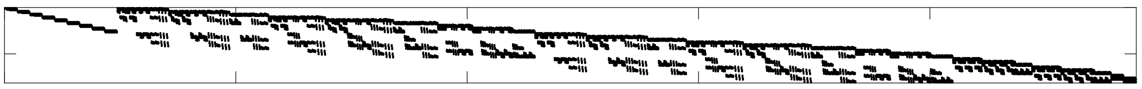

Figure 1: The sparsity pattern of $\mathbf{G}$ for a problem of the elastic cube with 27 subdomains an 2^3 elements per subdomain.

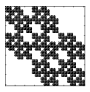

Figure 2: The sparsity pattern of $\mathbf{GG}^T$ for the same problem as in Fig. 1.

We have suggested and compared several strategies for parallel CP solution [11], [9]. The explicit orthonormalization approach starts to fail when the nullspace is large (thousands). Hence, we have abandoned this approach. Let us describe in more detail the strategies tested in this work.

Strategy 1 (S1) Obtain a solution of CP by solving the system (5) iteratively (by CG or pipelined CG (PipeCG)) or by a direct method (by parallel direct solver (SuperLU_DIST)). For a direct solution, $\mathbf{GG}^T$ is factorized in the preprocessing phase: $\mathbf{GG}^T = \mathbf{L}^{\mathbf{GG}^T}(\mathbf{L}^{\mathbf{GG}^T})^T$. During the solution phase, each application of $(\mathbf{GG}^T)^{-1}$ consists of the forward and backward substitution using a parallel direct solver: $\mathbf{x} = (\mathbf{GG}^T)^{-1}\mathbf{y}$ is solved by a two-step procedure as (1) $\mathbf{L}^{\mathbf{GG^T}}\mathbf{w} = \mathbf{y}$, (2) $(\mathbf{L}^{\mathbf{GG^T}})^T\mathbf{x} = \mathbf{w}$.

Strategy 2 (S2) An iterative or a parallel direct solver is employed for the computation of the *explicit inverse* of $\mathbf{GG}^T$. During the preprocessing phase, $(\mathbf{GG}^T)^{-1}$ is computed iteratively or by a direct method. In the solution phase, its application consists in the parallel dense matrix-vector product $(\mathbf{GG}^T)^{-1}\mathbf{y}$ in both cases.

The CP dimension is not large enough to justify the use of the whole global communicator. Instead, we propose a proper *partial* parallelization of this CP solution. We divide all processes of the global `PETSC_COMM_WORLD` communicator into the subcommunicators using PETSc built-in “pseudopreconditioner” `PCREDUNDANT`; the number of these subcommunicators is N_r (number of cores doing redundant work); this means the number of cores in each subcommunicator is $\approx N_c/N_r$.

In Strategy 2, the explicit inverse is assembled in the following way. Each of N_r subcommunicators is assigned a contiguous portion of N_n/N_r columns of the identity matrix taken as the right-hand side, where N_n is the dimension of the nullspace of $\mathbf{K}$, i.e. the number of columns of the matrix $\mathbf{R}$. The result of the forward/backward substitutions is the corresponding portion of N_n/N_r columns of the resulting explicit inverse $(\mathbf{GG}^T)^{-1}$, stored as a $N_n \times (N_n/N_r)$ dense matrix distributed vertically across the subcommunicator. Taking advantage of the symmetry of $(\mathbf{GG}^T)^{-1}$, each subcommunicator's block is transposed in parallel and the blocks are then merged one

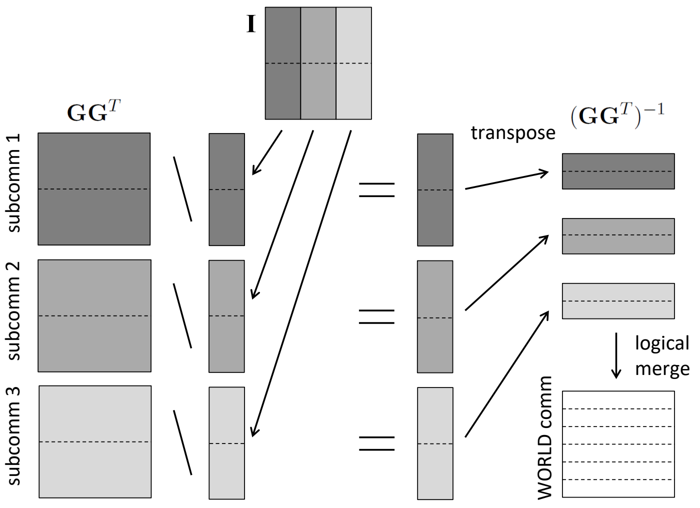

Figure 3: Scheme of $(\mathbf{GG}^T)^{-1}$ implementation using Strategy 2. Different colours represent different communicators.

below the other in the proper order forming the complete $(\mathbf{GG}^T)^{-1}$ matrix, divided into horizontal blocks distributed across the global communicator. Note that this merge means only logical reassignment from the subcommunicator to the global communicator with no actual data movements. A scheme of this strategy is depicted in Fig. 3.

5. Numerical experiments

The numerical experiments were performed at ARCHER, the latest UK National Supercomputing Service. It is based on a Cray XC30 supercomputer with 4920 nodes, 118,080 cores and 1.56 Petaflops of theoretical peak performance. All compute nodes are connected together in the Dragonfly topology by the Aries interconnect. Each compute node contains two 2.7 GHz, 12-core Ivy Bridge processors.

Firstly, we have performed a comparison of the CP strategies. For the CP solution the SuperLU_DIST solver performs better than the MUMPS solver. The GGtinv phase of the S1 is much cheaper in comparison with S2. On the other hand, S1 has much more expensive CP actions compared with S2. For a high number of expected CP actions, the second strategy starts to payoff because the high cost of preprocessing phase is offset by the cheapness of the CP action. The choice of an appropriate strategy therefore depends on the number of expected CP actions. This can be interesting for ill conditioned elasto-static problems but even more interesting for contact problems where the number of iterations is always higher. Finally, the greatest effect will be seen for all problems that are solved using outer iteration on top of FETI such as shape optimization, transient problems, or elasto-plasticity. Graphs in Fig. 4 show performance of both strategies on 8,000 subdomains (the CP size 48,000).

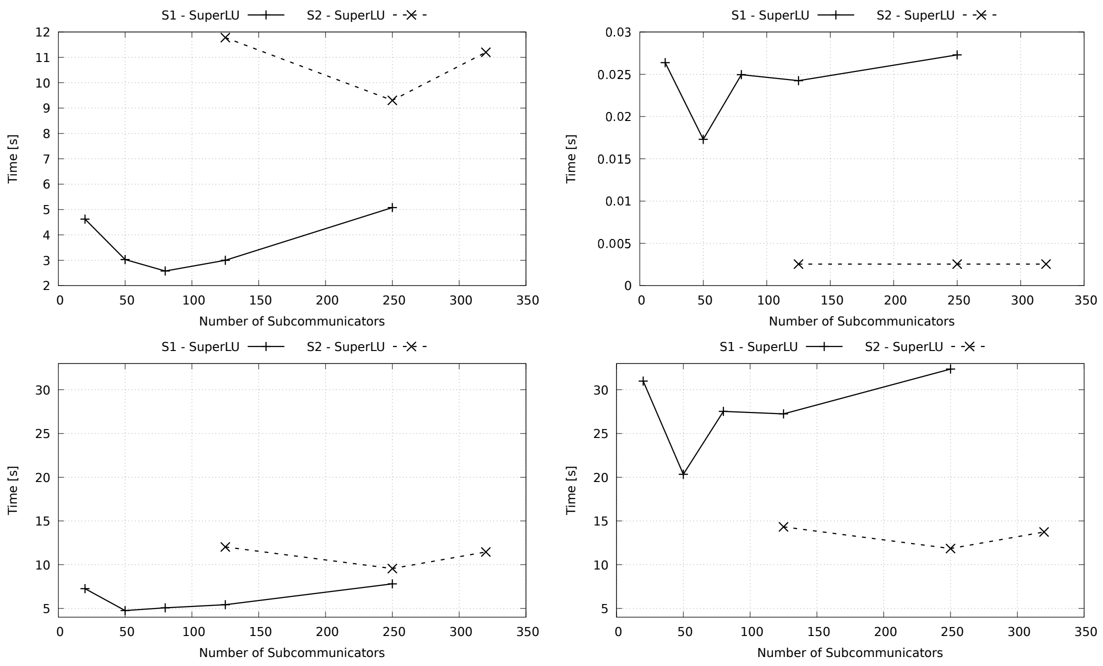

Figure 4: CP performance for the cube benchmark with 8,000 subdomains. $\mathbf{GG}^T$ size is 48,000. Top left: CP setup time. Top right: CP solution time. Bottom left: Time of CP setup + 100 CP actions. Bottom right: Time of CP setup + 1000 CP actions actions.

Secondly, we have demonstrated PERMON capabilities using weak scalability tests with the S1 strategy. As a model 3D linear elasticity problem, we consider an elastic cube with the bottom face fixed generated by our PermonCube benchmark generation package. For the linear case (see Fig. 5), the top face is loaded with a vertical surface force $f_z = 465$ [N/mm^2] directed upwards. For the nonlinear case (see Fig. 7), the top face is loaded with a vertical surface force $f_z = -465$ [N/mm^2] directed downwards, and the right one is partially in contact with a rigid obstacle. In both cases, Young modulus is $E = 2 \cdot 10^5$ [MPa], and Poisson ratio is $\mu = 0.33$. The graphs in Fig. 6 and Fig. 8 demonstrate both numerical and weak parallel scalability up to 701 millions of unknowns and 10,648 subdomains with one subdomain per one computational core. The contact problem was solved using SMALBE and Modified Proportioning and Reduced Gradient Projection (MPRGP) with our new adaptive expansion steplength which significantly improved this scalability and reduced not only the number of expansion steps but also the number of CG steps.

6. Conclusion

The PERMON team was successful to push the scalability limits for both linear and nonlinear benchmarks using ARCHER up to 702 millions of unknowns and 10,648 subdomains (cores). The implemented matrix formats and efficient parallel

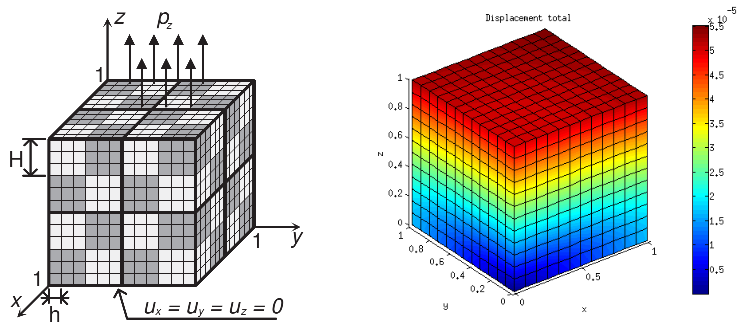

Figure 5: Linearly elastic cube problem.

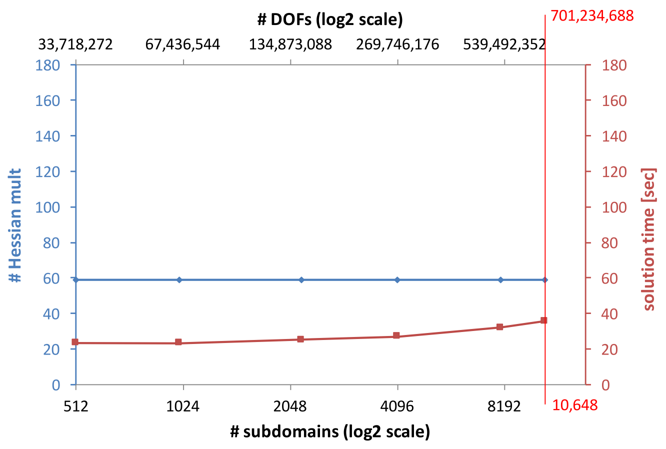

Figure 6: Scalability results for the linearly elastic cube problem.

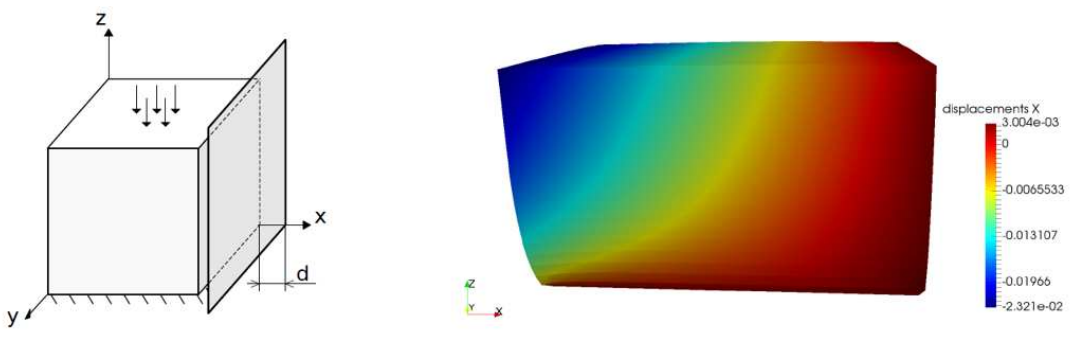

Figure 7: Contact problem – a linearly elastic cube with an obstacle.

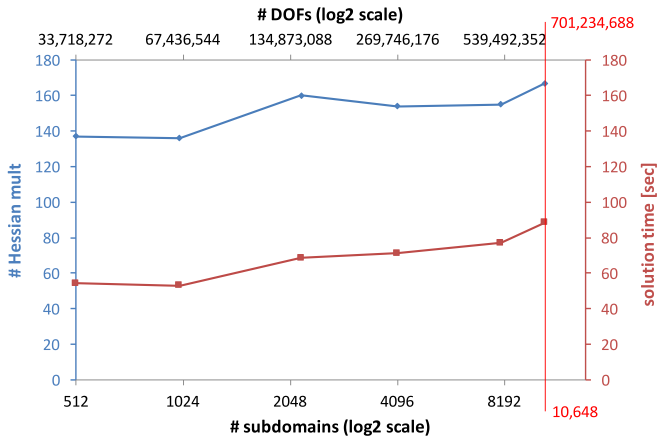

Figure 8: Scalability results for the contact problem.

direct solvers were employed. Furthermore, two strategies for CP solution were studied: (1) factorization + forward/backward substitutions, (2) factorization + explicit inverse assembly + dense matrix-vector products. It was demonstrated that the optimal strategy depends on the number of subdomains and the expected number of CP actions. The latter depends on the class of the solved problem.

Acknowledgements

The authors acknowledge the support of the Czech Science Foundation (GACR) project no. 15-18274S. The work was also 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", and by the internal student grant competition project SP2016/178 "PERMON toolbox development II".

References

[1] METIS. URL http://tinyurl.com/libmetis.

[2] Balay, S., Gropp, W. D., McInnes, L. C., and Smith, B. F.: Efficient management of parallelism in object oriented numerical software libraries. In: E. Arge, A. M. Bruaset, and H. P. Langtangen (Eds.), *Modern Software Tools in Scientific Computing*, pp. 163–202. Birkhäuser Press, 1997. doi:10.1007/978-1-4612-1986-6_8.

[3] Balay, S. et al.: PETSc – Portable, Extensible Toolkit for Scientific Computation. URL http://www.mcs.anl.gov/petsc.

[4] Dostál, Z., Horák, D., and Kučera, R.: Total FETI – an easier implementable variant of the FETI method for numerical solution of elliptic PDE. Commu-

nications in Numerical Methods in Engineering **22** (2006), 1155–1162. doi:10.1002/cnm.881.

[5] Dostál, Z. et al.: FETI based algorithms for contact problems: scalability, large displacements and 3D Coulomb friction. Comput. Methods Appl. Mech. Engrg. **194** (2005), 395–409. doi:10.1016/j.cma.2004.05.015.

[6] Farhat, C., Mandel, J., and Roux, F. X.: Optimal convergence properties of the FETI domain decomposition method. Comput. Methods Appl. Mech. Engrg. **115** (1994), 365–385. doi:10.1016/0045-7825(94)90068-X.

[7] Farhat, C. and Roux, F. X.: A method of finite element tearing and interconnecting and its parallel solution algorithm. International Journal for Numerical Methods in Engineering **32** (1991), 1205–1227. doi:10.1002/nme.1620320604.

[8] Gosselet, P. and Rey, C.: Non-overlapping domain decomposition methods in structural mechanics. Arch. Comput. Methods Engrg. **13** (2006), 515–572. doi:10.1007/BF02905857.

[9] Hapla, V., Horák, D., and Merta, M.: Use of direct solvers in TFETI massively parallel implementation. In: PARA 2012, Revised Selected Papers, *Lecture Notes in Computer Science*, vol. 7782, pp. 192–205. Springer-Verlag Berlin Heidelberg, 2013. doi:10.1007/978-3-642-36803-5_14.

[10] Hapla, V. et al.: PERMON (Parallel, Efficient, Robust, Modular, Object-oriented, Numerical). URL http://permon.it4i.cz.

[11] Hapla, V. and Horák, D.: TFETI coarse space projectors parallelization strategies. In: PPAM 2011, Revised Selected Papers, Part I, *Lecture Notes in Computer Science*, vol. 7203, pp. 152–162. Springer-Verlag Berlin Heidelberg, 2012. doi:10.1007/978-3-642-31464-3_16.

[12] Hapla, V. et al.: Solving contact mechanics problems with PERMON. In:HPCSE 2015, Revised Selected Papers, *Lecture Notes in Computer Science*, vol. 9611, pp. 101–115. Springer International Publishing Switzerland, 2016. doi:10.1007/978-3-319-40361-8_7.

[13] Munson, T., Sarich, J., Wild, S., Benson, S., and McInnes, L. C.: TAO users manual. Tech. Rep. ANL/MCS-TM-322, Argonne National Laboratory, 2015. URL http://tinyurl.com/tao-man.

[14] Vašatová, A., Čermák, M., and Hapla, V.: Parallel implementation of the FETI DDM constraint matrix on top of PETSc for the PermonFLLOP package. In: PPAM 2015, Revised Selected Papers, Part I, *Lecture Notes in Computer Science*, vol. 9573, pp. 150–159. Springer International Publishing Switzerland, 2016. doi:10.1007/978-3-319-32149-3_15.