
Parallel Scalability of a FETI–DP Mortar Method for Problems with Discontinuous Coefficients

Nina Dokeva and Wlodek Proskurowski

Department of Mathematics, University of Southern California, Los Angeles, CA 90089–1113, USA. dokeva,proskuro@usc.edu

Summary. We consider elliptic problems with discontinuous coefficients discretized by finite elements on non-matching triangulations across the interface using the mortar technique. The resulting discrete problem is solved by a FETI–DP method using a preconditioner with a special scaling described in a forthcoming paper by Dokeva, Dryja and Proskurowski. Experiments performed on up to a thousand processors show that this FETI–DP mortar method exhibits good parallel scalability.

1 Introduction

Parallelization of finite element algorithms enables one to solve problems with a large number of degrees of freedom in a reasonable time, which becomes possible if the method is scalable.

We adopt here the definition of scalability of [3] and [4]: solving n -times larger problem using an n -times larger number of processors in nearly constant cpu time. Domain decomposition algorithms using FETI-DP solvers ([7], [8], [9], [10]) have been demonstrated to provide scalable performance on massively parallel processors, see [4] and the references therein.

The aim of this paper is to experimentally demonstrate that a scalable performance on hundreds of processors can be achieved for a mortar discretization using FETI-DP solvers described in [5] and [6].

In view of the page limitation, Section 2 describing the FETI-DP method and preconditioner is abbreviated to a minimum. For a complete presentation refer to [5]. Section 3 contains the main results.

2 FETI-DP equation and preconditioner

We consider the following differential problem.

Find $u^* \in H_0^1(\Omega)$ such that

$$a(u^*, v) = f(v), \quad v \in H_0^1(\Omega), \quad (1)$$

where

$$a(u, v) = (\rho(x)\nabla u, \nabla v)_{L^2(\Omega)}, \quad f(v) = (f, v)_{L^2(\Omega)}.$$

We assume that Ω is a polygonal region and $\bar{\Omega} = \bigcup_{i=1}^N \bar{\Omega}_i$, Ω_i are disjoint polygonal subregions, $\rho(x) = \rho_i$ is a positive constant on Ω_i and $f \in L^2(\Omega)$. We solve (1) by the finite element method on geometrically conforming non-matching triangulation across $\partial\Omega_i$. To describe a discrete problem the mortar technique is used.

We impose on Ω_i a triangulation with triangular elements and a parameter h_i . The resulting triangulation of Ω is non-matching across $\partial\Omega_i$. Let $X_i(\Omega_i)$ be a finite element space of piecewise linear continuous functions defined on the triangulation introduced. We assume that the functions of $X_i(\Omega_i)$ vanish on $\partial\Omega_i \cap \partial\Omega$.

Let $X^h(\Omega) = X_1(\Omega_1) \times \dots \times X_N(\Omega_N)$ and let $V^h(\Omega)$ be a subspace of $X^h(\Omega)$ of functions which satisfy the mortar condition

$$\int_{\delta_m} (u_i - u_j)\psi ds = 0, \quad \psi \in M(\delta_m). \quad (2)$$

Here, $u_i \in X_i(\Omega_i)$ and $u_j \in X_j(\Omega_j)$ on Γ_{ij} , an edge common to Ω_i and Ω_j and $M(\delta_m)$ is a space of test (mortar) functions.

Let $\Gamma_{ij} = \partial\Omega_i \cap \partial\Omega_j$ be a common edge of two substructures Ω_i and Ω_j . Let Γ_{ij} as an edge of Ω_i be denoted by $\gamma_{m(i)}$ and called *mortar* (master), and let Γ_{ij} as an edge of Ω_j be denoted by $\delta_{m(j)}$ and called *non-mortar* (slave). Denote by $W_j(\delta_{m(j)})$ the restriction of $X_j(\Omega_j)$ to $\delta_{m(j)}$.

Using the nodal basis functions $\varphi_{\delta_{m(i)}}^{(l)} \in W_i(\delta_{m(i)})$, $\varphi_{\gamma_{m(j)}}^{(k)} \in W_j(\gamma_{m(j)})$ and $\psi_{\delta_{m(i)}}^{(p)} \in M(\delta_{m(i)})$, the matrix formulation of (2) is

$$B_{\delta_{m(i)}} u_{i\delta_{m(i)}} - B_{\gamma_{m(j)}} u_{j\gamma_{m(j)}} = 0, \quad (3)$$

where $u_{i\delta_{m(i)}}$ and $u_{j\gamma_{m(j)}}$ are vectors which represent $u_i|_{\delta_{m(i)}} \in W_i(\delta_{m(i)})$ and $u_j|_{\gamma_{m(j)}} \in W_j(\gamma_{m(j)})$, and

$$B_{\delta_{m(i)}} = \left\{ (\psi_{\delta_{m(i)}}^{(p)}, \varphi_{\delta_{m(i)}}^{(k)})_{L^2(\delta_{m(i)})} \right\}, \quad p = 1, \dots, n_{\delta(i)}, \quad k = 0, \dots, n_{\delta(i)} + 1,$$

$$B_{\gamma_{m(j)}} = \left\{ (\psi_{\delta_{m(i)}}^{(p)}, \varphi_{\gamma_{m(j)}}^{(l)})_{L^2(\gamma_{m(j)})} \right\}, \quad p = 1, \dots, n_{\delta(i)}, \quad l = 0, \dots, n_{\gamma(j)} + 1.$$

We rewrite the discrete problem for (1) in V^h as a saddle-point problem using Lagrange multipliers, λ . Its solution is $(u_h^*, \lambda_h^*) \in \tilde{X}^h(\Omega) \times M(\Gamma)$, where $\tilde{X}^h(\Omega)$ denotes a subspace of $X^h(\Omega)$ of functions which are continuous at vertices common to the substructures. We partition $u_h^* = (u^{(i)}, u^{(c)}, u^{(r)})$ into vectors containing the

interior nodal points of Ω_l , the vertices of Ω_l , and the remaining nodal points of $\partial\Omega_l \setminus \partial\Omega$, respectively.

Let $K^{(l)}$ be the stiffness matrix of $a_l(\cdot, \cdot)$. It is represented as

$$K^{(l)} = \begin{pmatrix} K_{ii}^{(l)} & K_{ic}^{(l)} & K_{ir}^{(l)} \\ K_{ci}^{(l)} & K_{cc}^{(l)} & K_{cr}^{(l)} \\ K_{ri}^{(l)} & K_{rc}^{(l)} & K_{rr}^{(l)} \end{pmatrix}, \quad (4)$$

where the rows correspond to the interior unknowns, its vertices and its edges.

Using this notation and the assumption of continuity of u_h^* at the vertices of $\partial\Omega_l$, the saddle point problem can be written as

$$\begin{pmatrix} K_{ii} & K_{ic} & K_{ir} & 0 \\ K_{ci} & \tilde{K}_{cc} & K_{cr} & B_c^T \\ K_{ri} & K_{rc} & K_{rr} & B_r^T \\ 0 & B_c & B_r & 0 \end{pmatrix} \begin{pmatrix} u^{(i)} \\ u^{(c)} \\ u^{(r)} \\ \tilde{\lambda}^* \end{pmatrix} = \begin{pmatrix} f^{(i)} \\ f^{(c)} \\ f^{(r)} \\ 0 \end{pmatrix}. \quad (5)$$

Here, the matrices K_{ii} and K_{rr} are diagonal block-matrices of $K_{ii}^{(l)}$ and $K_{rr}^{(l)}$, while \tilde{K}_{cc} is a diagonal block built by matrices $K_{cc}^{(l)}$ using the fact that $u^{(c)}$ are the same at the common vertices of the substructures. The mortar condition is represented by the global matrix $B = (B_c, B_r)$.

In the system (5) we eliminate the unknowns $u^{(i)}$ and $u^{(c)}$ to obtain

$$\begin{pmatrix} \tilde{S} & \tilde{B}^T \\ \tilde{B} & \tilde{S}_{cc} \end{pmatrix} \begin{pmatrix} u^{(r)} \\ \tilde{\lambda}^* \end{pmatrix} = \begin{pmatrix} \tilde{f}_r \\ \tilde{f}_c \end{pmatrix}, \quad (6)$$

where (since $K_{ic} = 0 = K_{ci}$ in the case of triangular elements and a piecewise linear continuous finite element space used in the implementation):

$$\begin{aligned} \tilde{S} &= K_{rr} - K_{ri}K_{ii}^{-1}K_{ir} - K_{rc}\tilde{K}_{cc}^{-1}K_{cr}, & \tilde{f}_r &= f^{(r)} - K_{ri}K_{ii}^{-1}f^{(i)} - K_{rc}\tilde{K}_{cc}^{-1}f^{(c)} \\ \tilde{B} &= B_r - B_c\tilde{K}_{cc}^{-1}K_{cr}, & \tilde{S}_{cc} &= -B_c\tilde{K}_{cc}^{-1}B_c^T, & \text{and} & \tilde{f}_c = -B_c\tilde{K}_{cc}^{-1}f_c. \end{aligned}$$

We next eliminate the unknown $u^{(r)}$ to get for $\tilde{\lambda}^* \in M(\Gamma)$

$$F\tilde{\lambda}^* = d, \quad (7)$$

where

$$F = \tilde{B}\tilde{S}^{-1}\tilde{B}^T - \tilde{S}_{cc}, \quad \text{and} \quad d = \tilde{B}\tilde{S}^{-1}\tilde{f}_r - \tilde{f}_c. \quad (8)$$

This is the FETI-DP equation for the Lagrange multipliers. Since F is positive definite, the problem has a unique solution. This problem can be solved by conjugate gradient iterations with a preconditioner discussed below.

Let $S^{(l)}$ denote the Schur complement of $K^{(l)}$, see (4), with respect to unknowns at the nodal points of $\partial\Omega_l$. This matrix is represented as

$$S^{(l)} = \begin{pmatrix} S_{rr}^{(l)} & S_{rc}^{(l)} \\ S_{cr}^{(l)} & S_{cc}^{(l)} \end{pmatrix}, \quad (9)$$

where the second row corresponds to unknowns at the vertices of $\partial\Omega_l$ while the first one corresponds to the remaining unknowns of $\partial\Omega_l$. Note that B_r is a matrix obtained from B defined on functions with zero values at the vertices of Ω_l and let

$$S_{rr} = \text{diag} \left\{ S_{rr}^{(l)} \right\}_{l=1}^N, \quad S_{cc} = \text{diag} \left\{ S_{cc}^{(l)} \right\}_{l=1}^N, \quad S_{cr} = \left(S_{cr}^{(1)}, \dots, S_{cr}^{(N)} \right). \quad (10)$$

We employ a special scaling appropriate for problems with discontinuous coefficients. The preconditioner M for (7) is defined as, see [5]

$$M^{-1} = \widehat{B}_r \widehat{S}_{rr} \widehat{B}_r^T, \quad (11)$$

where $\widehat{S}_{rr} = \text{diag} \left\{ \widehat{S}_{rr}^{(i)} \right\}_{i=1}^N$, $\widehat{S}_{rr}^{(i)} = S_{rr}^{(i)}$ for $\rho_i = 1$ and we define

$$\widehat{B}|_{\delta_{m(i)}} = \left(\rho_i^{1/2} I_{\delta_{m(i)}}, -\frac{h_{\delta_{m(i)}} \rho_i}{h_{\gamma_{m(j)}} \rho_j} \rho_i^{1/2} B_{\delta_{m(i)}}^{-1} B_{\gamma_{m(j)}} \right), \text{ for } \delta_{m(i)} \subset \partial\Omega_i, i = 1, \dots, N;$$

$h_{\delta_{m(i)}}$ and $h_{\gamma_{m(j)}}$ are the mesh parameters on $\delta_{m(i)}$ and $\gamma_{m(j)}$, respectively.

We have, following [5]

Theorem 1. *Let the mortar side be chosen where the coefficient ρ_i is larger. Then for $\lambda \in M(\Gamma)$ the following holds*

$$c_0 \left(1 + \log \frac{H}{h} \right)^{-2} \langle M\lambda, \lambda \rangle \leq \langle F\lambda, \lambda \rangle \leq c_1 \left(1 + \log \frac{H}{h} \right)^2 \langle M\lambda, \lambda \rangle, \quad (12)$$

where c_0 and c_1 are positive constants independent of h_i, H_i , and the jumps of ρ_i ; $h = \min_i h_i, H = \max_i H_i$.

This estimate allows us to achieve numerical scalability, an essential ingredient in a successful parallel implementation.

3 Parallel implementation and results

Our parallel implementation problem is divided into three types of tasks: solvers on the subdomains (with different meshes of discretization) which run individually and in parallel, a problem on the interfaces between the subdomains which can be solved in parallel with only a modest amount of global communication, and a "coarse" problem on the vertices between the subdomains which is a global task. A proper implementation of the coarse problem is crucial when the number of processors/subdomains is large.

We discuss some details of the implementation and present experimental results demonstrating that this method is well scalable. The numerical experiments were performed on up to 1024 processors provided by the University of Southern California Center for High Performance Computing and Communications (<http://www.usc.edu/hpcc>). All jobs were run on identically configured nodes equipped with dual Intel Pentium 4 Xeon 3.06 GHz processors, 2 GB of RAM and low latency Myrinet networking. Our code was written in C and MPI, using the PETSc toolkit (see [2]) which interfaces many different solvers.

The test example for our experiments is the weak formulation of

$$-\text{div}(\rho(x)\nabla u) = f(x) \text{ in } \Omega, \quad (13)$$

with Dirichlet boundary conditions on $\partial\Omega$, where $\Omega = (0, 1) \times (0, 1)$ is a union of $N = n^2$ disjoint square subregions $\Omega_i, i = 1, \dots, N$ and $\rho(x) = \rho_i$ is a positive

constant in each Ω_i . The coefficients $\rho(x)$ are chosen larger on the mortar sides of the interfaces, see Theorem 1.

The distribution of the coefficients ρ_i and grids h_i in Ω_i , $i = 1, \dots, 4$ with a maximum mesh ratio 8 : 1 used in our tests (for larger number of subregions, this pattern of coefficients is repeated) is here with $h = \frac{1}{32n}$:

$$\begin{pmatrix} 1e6 & 1e4 \\ 1e2 & 1 \end{pmatrix}, \quad \begin{pmatrix} h/8 & h/4 \\ h/2 & h \end{pmatrix}. \tag{14}$$

Each of the N processors works on a given subdomain and communicates mostly with the processors working on the neighboring subdomains.

For the subdomain solvers, we employ a symmetric block sparse Cholesky solver provided by the SPOOLES library (see [1]). The matrices are factored during the first solve and afterwards only a forward and backward substitutions are needed.

In each preconditioned conjugate gradient (PCG) iteration to solve the FETI-DP equation (7) for the Lagrange multipliers, there are two main operations:

1. multiplication by the preconditioner $M^{-1} = \tilde{B}_r \tilde{S}_{rr} \tilde{B}_r^T$ which involves solving N Dirichlet problems that are uncoupled, and some operations on the interfaces between the neighboring subdomains.
2. multiplication by $F = \tilde{B} \tilde{S}^{-1} \tilde{B}^T - \tilde{S}_{cc}$ which involves solving N coupled Neumann problems connected through the vertices.

The latter task involves solving a system with the global stiffness matrix K , see (5), of the form:

$$\begin{pmatrix} K_{ii} & 0 & K_{ir} \\ 0 & \tilde{K}_{cc} & K_{cr} \\ K_{ri} & K_{rc} & K_{rr} \end{pmatrix} \begin{pmatrix} v_i \\ v_c \\ v_r \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ p \end{pmatrix}. \tag{15}$$

Its Schur complement matrix C with respect to the vertices is

$$C = \tilde{K}_{cc} - (0, K_{cr}) \begin{pmatrix} K_{ii} & K_{ir} \\ K_{ri} & K_{rr} \end{pmatrix}^{-1} \begin{pmatrix} 0 \\ K_{rc} \end{pmatrix}. \tag{16}$$

C is a sparse, block tridiagonal $(n - 1)^2 \times (n - 1)^2$ matrix which has 9 nonzero diagonals. Solving a "coarse" problem with C is a global task while the subdomain solvers are local and run in parallel.

Proper implementation of the coarse system solving is important for the scalability especially when the number of processors/subdomains, N is large. Without assembling C , the coarse system could be solved iteratively (for example, with PCG using symmetric Gauss-Seidel preconditioner). Since the cpu cost then depends on N , it is preferable to assemble C .

We implemented two approaches discussed in [4]. In the case of relatively small C studied here one can invert C in parallel by duplicating it across a group of processors so that each computes a column of C^{-1} by a direct solver, for which we employed SPOOLES.

When C is larger the above approach may not be efficient or even possible; in that case one can use distributed storage for C and then a parallel direct solver. In a second implementation, we employed the block sparse Cholesky solver from the MUMPS package (see [11] and [12]) interfaced through PETSc. For simplicity, the

matrix C was stored on $n - 1$ or $(n - 1)^2$ processors, with the first choice yielding better performance.

In the tests run on up to (the maximum available to us) $N = 1024$ processors the two implementations performed almost identically. In Table 1 and Fig. 1 and 2 we present results from our first implementation when the coarse problem is solved by computing columns of C^{-1} .

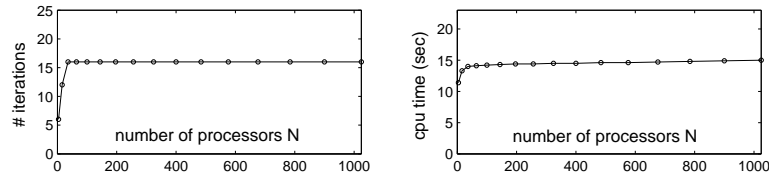


Fig. 1. Iterations and execution time vs number of processors.

Fig. 1 shows that the number of PCG iterations remains constant after $N = 36$ when the number of subdomains/processors is increased. The graph of the execution time (on the right) has a similar pattern. Although the number of degrees of freedom is increasing, the cpu time remains almost constant, see Table 1.

| N | # it | d.o.f. | cpu time |
|------|------|------------|----------|
| 4 | 6 | 87 037 | 11.4 |
| 16 | 13 | 350 057 | 13.3 |
| 36 | 16 | 789 061 | 14.0 |
| 64 | 16 | 1 404 049 | 14.1 |
| 100 | 16 | 2 195 021 | 14.2 |
| 144 | 16 | 3 161 977 | 14.3 |
| 196 | 16 | 4 304 917 | 14.4 |
| 256 | 16 | 5 623 841 | 14.4 |
| 324 | 16 | 7 118 749 | 14.5 |
| 400 | 16 | 8 789 641 | 14.5 |
| 484 | 16 | 10 636 517 | 14.6 |
| 576 | 16 | 12 659 377 | 14.6 |
| 676 | 16 | 14 858 221 | 14.7 |
| 784 | 16 | 17 233 049 | 14.8 |
| 900 | 16 | 19 783 861 | 14.9 |
| 1024 | 16 | 22 510 657 | 15.0 |

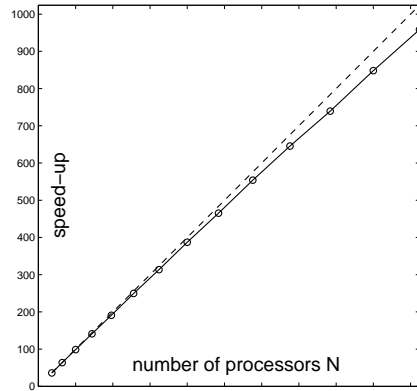


Fig. 2. Speed-up.

Table 1. Number of iterations, number of degrees of freedom and execution time in seconds.

Fig. 2 shows the speed-up of the algorithm, where the dashed line represents the ideal (linear) and the solid line the actual speed-up, respectively.

We adopt the definition of the speed-up of [3]. Here, it is adjusted to $N_0 = 36$ as a reference point, after which the number of iterations remains constant, see Table 1:

$$Sp = \frac{36 \times T_{36}}{T_{N_p}} \times \frac{N_{dof_{N_s}}}{N_{dof_{36}}},$$

where T_{36} and T_{N_p} denote the CPU time corresponding to 36 and N_p processors, respectively, and $N_{dof_{36}}$ and $N_{dof_{N_s}}$ denote the number of d.o.f. of the global problems corresponding to 36 and N_s subdomains, respectively.

This definition accounts both for the numerical and parallel scalability.

4 Conclusions

In this paper we study the parallel performance of the FETI-DP mortar preconditioner developed in [5] for elliptic 2D problems with discontinuous coefficients. The computational evidence presented illustrates good scalability of the method (an almost linear speed-up).

We would like to thank Max Dryja for his collaboration throughout. The first author would like to thank Panayot Vassilevski for the guidance during her summer internship at the Lawrence Livermore National Laboratory. The USC Center for High Performance Computing and Communications (HPCC) is acknowledged for generously providing us with the use of its Linux cluster.

References

1. C. ASHCRAFT AND R. G. GRIMES, *Spooles: An object-oriented sparse matrix library*, in Proceedings of the Ninth SIAM Conference on Parallel Processing for Scientific Computing, 1999.
2. S. BALAY, K. BUSCHELMAN, W. D. GROPP, D. KAUSHIK, L. C. MCINNES, AND B. F. SMITH, *PETSc home page*. <http://www.mcs.anl.gov/petsc>, 2001.
3. M. BHARDWAJ, D. DAY, C. FARHAT, M. LESOINNE, K. PIERSON, AND D. RIXEN, *Application of the FETI method to ASCI problems - scalability results on one thousand processors and discussion of highly heterogeneous problems*, Int. J. Numer. Meth. Engng., 47 (2000), pp. 513–535.
4. M. BHARDWAJ, K. PIERSON, G. REESE, T. WALSH, D. DAY, K. ALVIN, J. PEERY, C. FARHAT, AND M. LESOINNE, *Salinas: A scalable software for high-performance structural and solid mechanics simulations*, in Proceedings of 2002 ACM/IEEE Conference on Supercomputing, 2002, pp. 1–19. Gordon Bell Award.
5. N. DOKEVA, M. DRYJA, AND W. PROSKUROWSKI, *A feti-dp preconditioner with a special scaling for mortar discretization of elliptic problems with discontinuous coefficients*, SIAM J. Numer. Anal., 44 (2006), pp. 283–299.
6. M. DRYJA AND O. B. WIDLUND, *A FETI-DP method for a mortar discretization of elliptic problems*, vol. 23 of Lecture Notes in Computational Science and Engineering, Springer, 2002, pp. 41–52.
7. C. FARHAT, M. LESOINNE, P. LETALLEC, K. PIERSON, AND D. RIXEN, *FETI-DP: A dual-primal unified FETI method - part i: A faster alternative to the two-level FETI method*, Internat. J. Numer. Methods Engng., 50 (2001), pp. 1523–1544.

8. C. FARHAT, M. LESOINNE, AND K. PIERSON, *A scalable dual-primal domain decomposition method*, Numer. Lin. Alg. Appl., 7 (2000), pp. 687–714.
9. A. KLOWONN, O. B. WIDLUND, AND M. DRYJA, *Dual-Primal FETI methods for three-dimensional elliptic problems with heterogeneous coefficients*, SIAM J. Numer. Anal., 40 (2002), pp. 159–179.
10. J. MANDEL AND R. TEZAUER, *On the convergence of a dual-primal substructuring method*, Numer. Math., 88 (2001), pp. 543–558.
11. I. S. D. PATRICK R. AMESTOY AND J.-Y. L'EXCELLENT, *Multifrontal parallel distributed symmetric and unsymmetric solvers*, Comput. Methods Appl. Mech. Engrg., 184 (2000), pp. 501–520.
12. I. S. D. PATRICK R. AMESTOY, J.-Y. L'EXCELLENT, AND J. KOSTER, *A fully asynchronous multifrontal solver using distributed dynamic scheduling*, SIAM J. Matrix Anal. Appl., 23 (2001), pp. 15–41.