Contemporary Mathematics
Volume 180, 1994

Tailoring Domain Decomposition Methods for Efficient Parallel Coarse Grid Solution and for Systems with Many Right Hand Sides

CHARBEL FARHAT
PO-SHU CHEN

ABSTRACT. We present and illustrate a methodology for extending the range of applications of scalable domain decomposition methods to problems with multiple and/or repeated right-hand sides, and for solving efficiently their coarse grid problems on massively parallel processors.

1. Introduction

The condition number of the interface problem associated with a *numerically scalable* domain decomposition (DD) method does not grow (or "grows weakly") asymptotically with the number of subdomains. One of the many reasons why numerical scalability is desirable is that increasing the number of subdomains is the simplest means for increasing the degree of parallelism of a DD based preconditioned conjugate gradient (PCG) algorithm. In other words, numerical scalability is critical for massively parallel processing. This optimal property is usually achieved via the introduction in a DD method of a coarse problem (or coarse grid, by analogy with multigrid methods) that relates to the original problem and that must be solved at each global CG iteration. Direct methods are often chosen for solving the coarse problem despite the fact that they are difficult to implement on a massively parallel processor and do not parallelize well. Therefore in many cases, a numerically scalable DD method loses its appeal because of its lack of *parallel scalability.* One way to restore parallel scalability is to solve iteratively the coarse problem, for example using a CG

1991 Mathematics Subject Classification Primary 65N20, 65N30, 65W05
Supported by RNR NAS at NASA Ames Research Center under Grant NAG 2-827, and by the National Science Foundation under Grant ASC-9217394.
This paper is in final form and will not be published elsewhere.

scheme. However, this approach raises the question of how to solve iteratively and efficiently a system with a constant matrix and repeated right-hand sides. Finding an answer to this question also extends the range of applications of DD methods to design problems, time dependent problems, eigenvalue problems, and several other applications where multiple and repeated right-hand sides always arise and challenge iterative solvers. The iterative solution of systems with multiple right-hand sides has been previously addressed in [1], and recently in [2,3]. Here, we present a CG based methodology for solving such problems that uses the same data structures as those employed in DD methods without a coarse grid. The numerical idea exposed in this paper is related to that analyzed in [1]. However, the specific algorithm proposed herein is different, simpler, and easier to parallelize than that described in [1], and faster but more memory consuming [4] than both schemes presented in [2].

2. Problem formulation and nomenclature

For the sake of clarity, we first discuss the problem and the proposed solution methodology in the absence of any DD method or coarse grid. In Section 4, we highlight the positive implications of domain decomposition on the advocated approach. We are interested in solving iteratively the following problems:

$$(1) \qquad Ax_i = b_i \qquad i = 1, \dots, N_{rhs}$$

where A, $\{b_i\}_{i=1}^{i=N_{rhs}}$, and $\{x_i\}_{i=1}^{i=N_{rhs}}$ denote respectively a symmetric positive definite matrix, a set of N_{rhs} right-hand sides, and the corresponding set of N_{rhs} solution vectors. These problems can be transformed into the following minimization problems:

$$(2) \qquad \min_{x \in \mathcal{R}^{N_A}} \Phi_i(x) = \frac{1}{2}\, x^T A x - b_i^T x \qquad i = 1, \dots, N_{rhs}$$

where N_A is the dimension of matrix A, $\mathcal{R}$ is the set of real numbers, and T is the transpose superscript. If each minimization problem in (2) is solved with a PCG algorithm, the following Krylov subspaces are generated:

$$(3) \qquad \mathcal{S}_i = \{s_i^{(1)}, s_i^{(2)}, \dots, s_i^{(k)}, \dots, s_i^{(r_i)}\} \qquad i = 1, \dots, N_{rhs}$$

where $s_i^{(k)}$ and $r_i < N_A$ denote respectively the search direction vector at iteration k, and the number of iterations for convergence of the PCG algorithm applied to the minimization of $\Phi_i(x)$. Additionally, we introduce the following agglomerated subspaces:

$$(4) \qquad \overline{\mathcal{S}}_i = \bigcup_{j=1}^{j=i} \mathcal{S}_j \qquad i = 1, \dots, N_{rhs}$$

Let S_i denote the rectangular matrix associated with $\mathcal{S}_i$. From the orthogonality properties of the conjugate gradient method, it follows that:

$$(5) \qquad S_i^T A S_i = D_i \qquad i = 1, \dots, N_{rhs} \qquad D_i = \begin{bmatrix} d_{i_1} & d_{i_2} & \cdots & d_{i_{r_i}} \end{bmatrix}$$

However, note that in general $\overline{S}_i^T A \overline{S}_i$ is not a diagonal matrix. Finally, we define $\overline{\overline{S}}_i$ as the matrix whose column vectors also span the subspace $\overline{\mathcal{S}}_i$, but are orthogonalized with respect to matrix A. Hence, we have:

(6)

$$range\ (\overline{\overline{S}}_i) \;=\; \overline{\mathcal{S}}_i \qquad i \;=\; 1,\;...,\;N_{rhs}, \qquad \overline{\overline{S}}_i^T A \overline{\overline{S}}_i \;=\; \overline{\overline{D}}_i \qquad i \;=\; 1,\;...,\;N_{rhs}$$

$$\overline{\overline{D}}_i \;=\; \left[\bar{\bar{d}}_{i_1} \;\; \bar{\bar{d}}_{i_2} \;\; \cdots \;\; \bar{\bar{d}}_{i_{\bar{\bar{r}}_i}}\right], \qquad \bar{\bar{r}}_i \;=\; \sum_{j=1}^{j=i} r_j$$

3. Projection and orthogonalization

Suppose that the first problem $Ax_1 = b_1$ has been solved in r_1 PCG iterations, and that the $N_A \times r_1$ matrix S_1 associated with the Krylov subspace $\mathcal{S}_1$ is readily available. Solving the second problem $Ax_2 = b_2$ is equivalent to solving:

$$\min_{x \in \mathcal{R}^{N_A}} \Phi_2(x) \;=\; \frac{1}{2}\; x^T A x \;-\; b_2^T x \tag{7}$$

If $\mathcal{R}^{N_A}$ is decomposed as follows:

$$\text{(8)}\quad \mathcal{R}^{N_A} \;=\; \mathcal{S}_1 \oplus \mathcal{S}_1^*, \quad dim\ (\mathcal{S}_1^*) \;=\; N_A - r_1, \quad \mathcal{S}_1 \text{ and } \mathcal{S}_1^* \text{ are } A-\text{orthogonal}$$

then the solution of problem (7) can be written as:

$$x_2 \;=\; x_2^0 \;+ z_2, \quad x_2^0 \in \mathcal{S}_1, \; z_2 \in \mathcal{S}_1^*, \;\text{ and }\; x_2^{0^T} A z_2 \;=\; z_2^T A x_2^0 \;=\; 0 \tag{9}$$

From Eqs. (7-9), it follows that x_2^0 is the solution of the minimization problem:

$$\min_{x \in \mathcal{S}_1} \Phi_2(x) \;=\; \frac{1}{2}\; x^T A x \;-\; b_2^T x \tag{10}$$

and z_2 is the solution of the minimization problem:

$$\min_{z \in \mathcal{S}_1^*} \Phi_2(z) \;=\; \frac{1}{2}\; z^T A z \;-\; b_2^T z \tag{11}$$

First, we consider the solution of problem (10). Since $x_2^0 \in \mathcal{S}_1$, there exists a $y_2^0 \in \mathcal{R}^{r_1}$ such that:

$$x_2^0 \;=\; S_1 y_2^0 \tag{12}$$

Substituting Eq. (12) into Eq. (10) leads to the following minimization problem:

$$\min_{y \in \mathcal{R}^{r_1}} \widetilde{\Phi}_2(y) \;=\; \frac{1}{2}\; y^T S_1^T A S_1 y \;-\; b_2^T S_1 y \tag{13}$$

whose solution y_2^0 is given by:

$$S_1^T A S_1 \, y_2^0 = \tilde{b}_2, \quad \tilde{b}_2 = S_1^T b_2 \tag{14}$$

From Eq. (5), it follows that the system of equations (14) is diagonal. Hence, the components $[y_2^0]_j$ of y_2^0 can be simply computed as follows:

$$[y_2^0]_j = \frac{[\tilde{b}_2]_j}{d_{1_j}} \qquad j = 1, \ldots, r_1 \tag{15}$$

Next, we turn to the solution of problem (11) via a PCG algorithm. Since the decomposition (9) requires z_2 to be A-orthogonal to x_2^0, at each iteration k, the search directions $s_2^{(k)}$ must be explicitly A-orthogonalized to S_1. This entails the computation of modified search directions $\hat{s}_2^{(k)}$ as follows:

$$\hat{s}_2^{(k)} = s_2^{(k)} + \sum_{q=1}^{q=r_1} \alpha_q s_1^{(q)}, \quad \alpha_q = -\frac{s_1^{(q)^T} A s_2^{(k)}}{s_1^{(q)^T} A s_1^{(q)}} = -\frac{s_2^{(k)^T} A s_1^{(q)}}{s_1^{(q)^T} A s_1^{(q)}} \tag{16}$$

Except for the above modifications, the original PCG algorithm is unchanged. However, convergence is expected to be much faster for the second and subsequent problems than for the first one, because S_1^* and the subsequent supplementary spaces have smaller dimensions than $\mathcal{R}^{N_A}$, and a significant number of the solution components are included in the startup solutions of the form of x_2^0.

The generalization to the case of N_{rhs} right-hand sides of the two-step solution procedure described above is straightforward [3].

4. Application to coarse grids and DD interface problems

Despite its elegance and simplicity, the methodology described in Section 3 can be impractical when applied to the global solution of the problems $Ax_i = b_i$, $i = 1$, N_{rhs}. Indeed, during the PCG solution of the first few problems — that is, before superconvergence can be reached — the cost of the orthogonalizations implied by Eq. (16) can offset the benefits of convergence acceleration via the optimal startup solution and the modified search directions $\hat{s}_i^{(k)}$. Moreover, storing every search direction $\hat{s}_i^{(k)}$ and the corresponding matrix-vector product $A\hat{s}_i^{(k)}$ can significantly increase the memory requirements of the basic PCG algorithm. However, the proposed methodology is computationally feasible in a domain decomposition context because it is applied only to the coarse grid and/or the interface problem.

5. Performance evaluation for coarse problems

First, we consider the static solution of a two-dimensional plane stress elasticity problem using the FETI domain decomposition method [5] on an iPSC-860 parallel processor. The FETI method is numerically scalable. For elasticity problems, the two-norm condition number of its preconditioned interface problem grows asymptotically as $\kappa_2 = O\left(1 + log^2 \frac{H}{h}\right)$, where H and h denote respectively the subdomain and mesh sizes [6]. At each k-th FETI global PCG iteration, the following coarse problem must be solved:

$$(G_I{}^t G_I)\, x_k \;=\; b_k \tag{17}$$

where $G = [\, B_1 R_1 \;\; \ldots \;\; B_s R_s \;\; \ldots \;\; B_{N_s} R_{N_s} \,]$, B_s is a boolean matrix that extracts from a subdomain quantity its interface component, R_s spans the null space of the stiffness matrix associated with a singular subdomain s, N_s denotes the total number of subdomains, and b_k is related to the interface residual at the k-th global PCG iteration. Clearly, the systems in (17) have a constant matrix and repeated right-hand sides. The parallel solution of these problems via the CG scheme described in Section 3 is implemented using the existing subdomain based parallel data structures; it requires communication only between neighboring subdomains. Using 4-node plane stress elements, three finite element models corresponding to 4, 16, and 64 subdomains are constructed: each model has a different size but the ratio H/h is kept constant across all three finite element models. The performance results measured for all three problems are summarized in Table 1 where NEQ, N_p, N_{itr}, and T_{tot} denote respectively the number of equations generated by the finite element model, the number of processors, the number of FETI global PCG iterations, and the total CPU time for solving the plane stress problem.

TABLE 1
Two-dimensional elasticity problem: performance results on an iPSC-860

h	H	NEQ	N_p	N_{itr}	T_{tot}
$\frac{1}{40}$	$\frac{1}{2}$	3,200	4	10	3.45 secs.
$\frac{1}{80}$	$\frac{1}{4}$	12,800	16	16	5.17 secs.
$\frac{1}{160}$	$\frac{1}{8}$	51,200	64	17	5.92 secs.

Clearly, the results summarized in Table 1 demonstrate the combined numerical and parallel scalability of the FETI method and its parallel implementation.

6. Performance evaluation for time dependent problems

Next, we apply the methodology described in this paper to the solution of repeated systems arising from the linear transient analysis using an implicit time-integration scheme of a three-dimensional line-pinched membrane with a circular hole. The structure is discretized in 5,680 4-node elements and 11,640 degrees of freedom. The finite element mesh is partitioned into 32 subdomains. The size of the interface problem is 1,892 — that is, 16.25% of the size of the global problem. The transient analysis is carried out on a 32 processor iPSC-860 system. After all of the usual finite element storage requirements are allocated, there is enough memory left to store a total number of 891 search directions. This number corresponds to 47% of the size of the interface problem. Using a transient version of the FETI method without a coarse grid [7], the system of equations arising at the first time step is solved in 322 iterations. After 3 time steps, 435 search directions are accumulated and only 20 iterations are needed for solving the fourth linear system of equations. After 16 time steps, the total number of accumulated search directions is only 536 — that is, only 28% of the size of the interface problem, and superconvergence is triggered: all subsequent time steps are solved in 1 or 2 iterations and in less than 1.0 second CPU (Fig. 1). When a parallel direct solver is applied to the above problem, at each time step, the pair of forward/backward substitutions consumes 15.0 seconds on the same 32 processor iPSC-860. Therefore, the proposed solution methodology is clearly an excellent alternative to repeated forward/backward substitutions on distributed memory parallel processors.

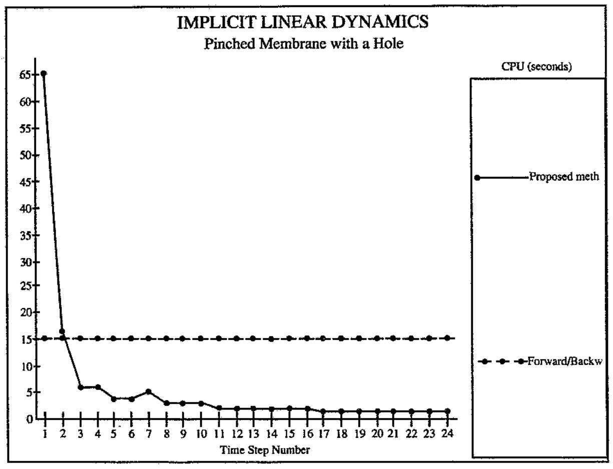

FIG. 1. *CPU history*

REFERENCES

1. Y. Saad, On the Lanczos Method for Solving Symmetric Linear Systems with Several Right-Hand Sides, Math. Comp., **48** No. 178 (1987), 651–662.

2. P. Fischer, Projection Techniques for Iterative Solution of Ax=b with Successive Right-Hand Sides, ICASE Rep. No. 93-90, NASA CR-191571.

3. C. Farhat, L. Crivelli and F. X. Roux, Extending Substructure Based Iterative Solvers to Multiple Load and Repeated Analyses, Comput. Meths. Appl. Mech. Engrg., (1994), in press.

4. P. Fischer, (1993), private communication.

5. C. Farhat and F. X. Roux, A method of finite element tearing and interconnecting and its parallel solution algorithm, Internat. J. Numer. Meths. Engrg., **32** (1991), 1205–1227.

6. C. Farhat, J. Mandel and F. X. Roux, Optimal Convergence Properties of the FETI Domain Decomposition Method, Comput. Meths. Appl. Mech. Engrg., (1994), in press.

7. L. Crivelli and C. Farhat, Implicit transient finite element structural computations on MIMD systems: FETI v.s. direct solvers, AIAA Paper 93-1310, AIAA 34th Structural Dynamics Meeting, La Jolla, California, April 19-21, 1993.

Department of Aerospace Engineering Sciences, Center for Aerospace Structures, University of Colorado, Boulder, Colorado 80309-0429

E-mail address: charbel@boulder.colorado.edu