SANDIA REPORT

SAND99–1158
Unlimited Release
Printed May 1999

A Massively Parallel Sparse Eigensolver for Structural Dynamics Finite Element Analysis

G. M. Reese and David M. Day

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy under Contract DE-AC04-94AL85000.

Sandia National Laboratories

Issued by Sandia National Laboratories, operated for the United States Department of Energy by Sandia Corporation.

Available to DOE and DOE contractors from
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831

Prices available from (703) 605-6000
Web site: http://www.ntis.gov/ordering.htm

Available to the public from
National Technical Information Service
U.S. Department of Commerce
5285 Port Royal Rd
Springfield, VA 22161

NTIS price codes
Printed copy: A04
Microfiche copy: A01

SAND99-1158
Unlimited Release
Printed May 1999

A Massively Parallel Sparse Eigensolver for Structural Dynamics Finite Element Analysis

G. M. Reese
Structural Dynamics Dept.

David M. Day
Applied Mathematics Dept.

Sandia National Laboratories
P. O. Box 5800
Albuquerque, New Mexico 87185-0439

Abstract

Eigenanalysis is a critical component of structural dynamics which is essential for determining the vibrational response of systems. This effort addresses the development of numerical algorithms associated with scalable eigensolver techniques suitable for use on massively parallel, distributed memory computers that are capable of solving large scale structural dynamics problems. An iterative Lanczos method was determined to be the best choice for the application. Scalability of the eigenproblem depends on scalability of the underlying linear solver. A multi-level solver (`FETI`) was selected as most promising for this component. Issues relating to heterogeneous materials, mechanisms and multipoint constraints have been examined, and the linear solver algorithm has been developed to incorporate features that result in a scalable, robust algorithm for practical structural dynamics applications. The resulting tools have been demonstrated on large problems representative of a weapon's system.

Contents

1. Introduction

Structural dynamics analysis is critical to applications in weapons, non-proliferation, space sensors and computational manufacturing. These applications require modeling and simulation which become the building blocks for design optimization, model validation and virtual testing for weapons and other defense programs. While several codes are currently used for structural analysis, these codes lack a crucial component required for large scale applications: a scalable, parallel eigensolver. Without a high performance eigensolver that takes advantage of the computational power and memory of MP systems, a number of encouraging new structural analysis directions will not be realized. This document details the development of this eigensolver and its integration with a parallel structural dynamics code.

A number of technical thrusts are demanding far greater structural dynamics analysis capabilities than are currently available. Two extremely important emerging activities are model based design and advanced manufacturing simulations with optimization and probabilistic analysis procedures. In agile manufacturing, it is necessary to understand and analyze many different designs (so that a number of different manufactured parts can be realized) before producing any hardware. In optimization and probabilistic design, we must solve many structural dynamics problems to find the best possible design, and account for hardware and loading variability.

Within these broad areas, numerous examples can be found of specific applications requiring substantial eigensolver capabilities. Even relatively small component's applications might require large finite element models with greater than a million degrees of freedom and can exceed current technology limits. For example, photo-lithography applications used for manufacturing electronic structures can result in large finite element models particularly when manufacturing details necessary for optimization of the fabricated part are included. Larger component assemblies (such as Sandia's STARS missile) require even bigger models which can only be currently addressed in a limited way through very costly model reduction techniques. Further, when these problems are embedded within an automatic optimization procedure, the computational effort increases dramatically. To successfully analyze these problems, scalable sparse eigenvalue software and algorithm advances are needed that take full advantage of current and future computer architectures.

Sandia is well positioned for development of a parallel eigensolution capability for structural dynamics. We have expertise in sparse eigensolver algorithms, parallel algorithms, and structural mechanics. Further, we can leverage existing Sandia software: a partitioning package, `Chaco` (which is combined into a general purpose partitioning tool

– Nemesis), and a general purpose parallel finite element analysis code, Salinas. The first package has been used to reduce the parallelization and the linear solvers effort as well as enable quick porting to new parallel hardware platforms. The integration of the eigensolver with Salinas yields a sophisticated structural mechanics package. This package has been used to ensure that the solver is robust and performs well for real engineering applications. Additionally, the integrated code allows benchmark comparisons with existing methods. The software is tailored to the ASCI Option Red platform, but has also been ported to a parallel SGI Origin 2000. We have focused on development of portable and scalable software that is applicable to a variety of parallel platforms.

This effort has initiated a very strong relationship with the department of aerospace structures at the University of Colorado. Their solver team has been a leader in the field of multilevel linear solvers for use on massively parallel architectures. They have been extremely successful in applying this technology to structural problems.

This Lab Directed Research and Development (LDRD) effort focuses on development of a massively parallel eigensolver for structural dynamics analysis. The largest single challenge is integration with a robust, scalable and accurate linear solver. Multilevel solvers require especially tight integration of the numerical and finite element analysis fields. Throughout the effort, numerical analysts and structural dynamicists have worked together to insure that the resulting software meets the needs of the finite element applications.

The goal of the study is develop the tools for robust eigensolution of realistic structural dynamics problems. The solvers must be portable and scalable, showing good performance on a variety of platforms. They must be robust enough to be applicable to a general purpose structural dynamics application. The target platforms are distributed memory systems with many (thousands) of compute nodes, and include large clusters of workstations.

2. Selection of Methods

A primary computational task of the `Salinas` package is to compute many of the lowest vibrational modes of structural finite element models. Finite element discretization leads to a sparse symmetric semi-definite generalized eigenvalue problem. The most efficient class of algorithms for these problems are sparse iterative eigensolvers (e.g. Lanczos) applied to a transformed (e.g. shifted and inverted) eigenvalue problem. The computational bottleneck is this transformation, and in the overall solution method, is realized through the solution of a sequence of sparse linear systems. This section describes the linear solver and eigensolver developed for use in `Salinas`.

A linear solver was selected based on the criteria of robustness, accuracy, scalability and efficiency. Direct methods based on sparse Gaussian elimination were considered first. Sequential direct solvers are robust, accurate, and have high performance BLAS3 implementations. Parallel direct solvers are accurate and achieve high performance, but are not scalable. Ironically, the greatest defect of the parallel direct solvers known to the authors is their poor robustness on platforms, such as ASCI Option Red, with fast communication, many processors, and limited per processor memory. The existing parallel direct solvers use non-blocking communication without attention to the total volume of information that has to be buffered (stored) on a given processor. Core dumps due to overflowed message buffers are routine. To factor large matrices, the user must manually set the message buffers to an enormous size determined by trial and error. Direct solvers were rejected due to their poor robustness and lack of scalability.

General purpose iterative solvers, such as the implementation of the preconditioned conjugate gradient method with over-lapping Schwarz preconditioner available in `Aztec`, were also evaluated. These methods converged too slowly for the linear systems obtained from the discretization of structures by high order plate and shell elements. In this case the underlying partial differential equation is the fourth order biharmonic equation for which special purpose iterative solvers are necessary. General purpose iterative solvers were also rejected due to their poor robustness and scalability.

We selected a multi-level domain decomposition method, Finite Element Tearing and Interconnect (`FETI`), that is the most successful parallel solver known to the authors for the linear systems applicable to structural mechanics. `FETI` is a mature solver, with some versions used in commercial finite element packages such as ANSYS. For plates and shells, the singularity in the linear systems has been traced to the subdomain corners. To solve such linear systems, an additional coarse problem is automatically introduced that removes the corner point singularity. `FETI` is scalable in the sense that as the number of unknowns increases and the number of unknowns per processor remains constant,

the time to solution does not increase. Further, FETI is accurate in the sense that the convergence rate does not deteriorate as the iterates converge. Finally the computational bottleneck in FETI, a sparse direct subdomain solve, is amenable to high performance solution methods.

An eigensolver was selected for Salinas based on the same criteria; robustness, accuracy, scalability and efficiency. We evaluated both a Lanczos-based solver and subspace iteration. The Lanczos algorithm solves the minimal number of linear systems required to approximate a set of modes to a given accuracy, and Lanczos-based methods are significantly more efficient than subspace iteration. Subspace iteration is a comparatively simple algorithm that is believed to be somewhat less sensitive to linear solver accuracy than Lanczos-based methods. Structural models are known for which the FETI solver does not converge, but in these cases the accuracy is too low for either subspace iteration or Lanczos-based methods to compute accurate modes. We selected the PARPACK Lanczos-based solver because the memory usage is minimal, the software is reliable, and the number of linear systems solved per mode is nearly minimized. PARPACK is scalable and achieves BLAS2 performance.

3. Development of Current Selections

The capabilities of the eigensolver package as outlined above are limited by several factors. As a result of research done under this LDRD, it was determined that the `Nemesis` domain decomposition package had to be modified to generate (face) connected subdomains and to minimize subdomain geometric aspect ratio. The versatility and robustness of the `FETI` linear solver required further development in several ways detailed below. Many enhancements to the eigensolver are also possible, though at present linear solver issues are more critical to extending the range of applicability of these tools. In this section, the robustness and versatility of `FETI` are discussed. The section concludes with a sketch of possible enhancements to the eigensolver.

Heterogeneous materials

The `FETI` linear solver is not robust if applied to models that consist of highly heterogeneous materials (with stiffnesses that vary by factors of one million). `FETI` uses both a Dirichlet preconditioner and a coarse grid correction. The Dirichlet preconditioner captures the local response of the system and the coarse grid correction captures the global response. On a subdomain with highly heterogeneous materials, a force applied at a point in the stiff material results primarily in a displacement of the soft materials. This nonlocal behavior causes Dirichlet preconditioning to fail. Two solutions to this problem were proposed and researched during the LDRD. The Q projector method and stiffness scaling both scale the problem by the relative stiffness of the elements at the interface. They have different implementations and results. Both methods provide some improvement over standard methods for very inhomogeneous models, but both are only partial solutions. The iteration count (and solution times) almost always increase for these more complicated structures.

The Q projector method, is used to improve the approximation to the stiffness matrix on the coarse grid. This method is detailed in Appendix A. The method effectively scales the solution in the coarse grid problem to improve the matrix conditioning and improve solution time.

The second solution is to normalize the material heterogeneity in the linear system by stiffness scaling. This method uses the diagonal terms of the stiffness matrix to scale the entire problem (coarse grid and subdomain levels). This is discussed in Appendix B.

Modifications to the partitioning software (`Nemesis/Chaco`) to partition the model based on material blocks has been shown to be very effective in reducing the effects of material nonlinearity. However, this solution is somewhat model dependent because par-

titioning by material blocks can be overconstraining. This can result in many partitions that span a major portion of the structure and have poor aspect ratios. Since convergence of the FETI method depends upon the aspect ratio of the subdomain, the improvement gained to overcome inhomogeneous materials may be offset by increased iterations due to poor aspect ratios.

A large demonstration problem was developed which has characteristics of many structural dynamics applications. The model consists of a re-entry vehicle structure with materials and geometry somewhat similar to those found in practice. Stiff, heavy materials are found next to foams. The shell consists of layers of aluminum, glue and phenolics. The overall model size was over one million degrees of freedom - significantly larger than current capabilities at Sandia for eigensolution of problems of this type.

With the improvements in material homogeneity mentioned above, this model was solved using Salinas and the eigensolvers and linear solvers developed in this LDRD. The solution required 512 processors on the ASCI Option Red platform and was completed in under 8 hours of computation.

Multi-point constraints

Multi-point constraints (or MPCs) are important in most structural dynamics applications. MPCs provide an algebraic link between different degrees of freedom in the model. They are used to build rigid elements, to apply boundary conditions and to combine resultants. They are especially important in connecting dissimilar meshes. Parallel solution of systems containing MPCs is particularly difficult because they introduce global communication requirements that are independent of the element connectivity. We have evaluated a number of approaches for implementing MPCs within FETI. These are detailed in Appendix C. Here it is shown that there are natural ways to implement these constraints in a manner similar to the corner point constraints developed to improve performance for shell models. These methods have been demonstrated in development tools, but have not been fully implemented in parallel.

Mechanisms

The linear solver, FETI, is quite dependent on the subdomain rigid body modes because they provide the basis to communicate global information throughout the system. In some automatic decomposition schemes it is quite common to produce subdomains which are not well connected. These subdomains may be only connected at a corner or an edge, which can lead to additional zero energy modes (or mechanisms). Elimination of these mechanisms can significantly improve the performance of the linear solver. We have developed tools to detect and eliminate these spurious zero energy modes. These methods have been included in Nemesis, guaranteeing mechanism free automatic decomposition.

Other Issues

This LDRD identified several needed developments to the FETI solver that were implemented based on the results of this research only after the LDRD ran out. The subdomain rigid body modes, which are used to construct the coarse grid, are now determined directly from the finite element mesh. Because many structures of interest at Sandia National Labs are unconstrained, Salinas and FETI have been developed to model floating structures. This modification was natural because the structural rigid body modes are determined in the FETI method in terms of a reduced model. A high performance (BLAS3) sparse frontal solver developed by Esmond Ng (NERSC) has been added for the subdomain solves, though this is fully implemented only for positive definite systems.

The eigensolver used in Salinas still needs improvements, though as mentioned above, none are critical. In the next release of PARPACK, a feature will be available that reduces the likelihood that modes are missed (e.g. that the ten computed modes are not the ten smallest modes). Including the same feature in Salinas will enhance the robustness of the eigensolver. The cost of orthogonalization may become a bottleneck for very large systems with many eigenvalues. This cost can be reduced by a factor of about six by modifying the code to maintain only semi-orthogonality as is done in MSC/NASTRAN and the PLANSO package developed at NERSC. These topics are discussed in more detail in Appendix B.

This research has also contributed to the development of Anasazi, a high performance parallel eigensolver which is less dependent on the accuracy of the underlying linear solves. The Anasazi eigensolver uses a block method, and achieves high BLAS3 performance compared to the BLAS2 performance of PARPACK. An interface from Salinas to Anasazi will be developed when the solver is sufficiently mature. Anasazi also uses Cayley transformations instead of shift-invert transformations, and this leads to linear systems that are often simpler to precondition. Another class of eigensolvers based on on inexact solves will also be included in Anasazi. In these methods the accuracy of the computed modes is independent of the linear solver accuracy.

4. Conclusions

This effort has shown that iterative eigensolver methods can be successfully applied to large scale structural dynamics problems on MP platforms. The approach taken involves a special purpose, multi-level linear solver, FETI, with characteristics which tune it for structural dynamics applications.

The linear solver has been developed with capabilities required for structural dynamics problems. Special effort has been made to provide the capabilities to treat heterogeneous materials, poor aspect ratios, and structural mechanisms. The framework for support of multi-point constraints has also been established. Robustness has been improved using geometric methods for the solution of rigid body modes of the substructure. The solver is robust, scalable and accurate.

Eigensolver technology has utilized the FETI linear solver framework. We have shown conclusively that the accuracy of the iterative linear solver is sufficient for determination of the lowest eigenmodes using a Lanczos method. Studies indicate that subspace iteration would be another viable eigensolver technique for these architectures.

Both the eigensolver and the linear solver show outstanding scalability and performance on realistic structural dynamics problems. Scalability has been demonstrated for thousands of processors on the ASCI Option Red supercomputer. The appendices include further details on several technical developments resulting from the research supported by this LDRD.

Appendix A
SAND99-0937: Application of the FETI Method to ASCI Problems: Scalability Results on One Thousand Processors and Discussion of Highly Heterogeneous Problems

Printed April 1999

Manoj Bhardwaj
Structural Dynamics Department

David Day
Applied Mathematics Department

Sandia National Laboratories
P. O. Box 5800
Albuquerque, NM 87185-0439

Charbel Farhat, Michel Lesoinne, Kendall Pierson and Daniel Rixen
Department of Aerospace Engineering and Center for Aerospace Structures
University of Colorado at Boulder
Boulder, CO 80309-0429

Abstract

We report on the application of the one-level FETI method to the solution of a class of substructural problems associated with the Department of Energy's Accelerated Strategic Computing Initiative (ASCI). We focus on numerical and parallel scalability issues, and on preliminary performance results obtained on the ASCI Option Red supercomputer configured with as many as one thousand processors, for problems with as many as 5 million degrees of freedom.

Keywords: ASCI, FETI, scalability, domain decomposition, structural heterogeneities

1. Introduction

In 1996, the US Department of Energy announced its Accelerated Strategic Computing Initiative (ASCI) aimed at creating predictive simulation and virtual prototyping capabilities, and accelerating the development of high-performance computing far beyond what might be achieved in the absence of a focused initiative. More specifically, ASCI's vision is to shift promptly from test-based methods to computational-based methods of ensuring the safety, reliability, and performance of the US nuclear weapons stockpile. An initial result of this initiative was the installation in 1997 at the Sandia National Laboratories of an Intel 1.8-Teraflops (trillion floating-point operations per second) peak massively parallel system known as the ASCI Option Red supercomputer. Two additional Teraflop systems known as the ASCI Blue Pacific and ASCI Blue Mountain machines were subsequently sited at the Livermore and Los Alamos National Laboratories, respectively. Harnessing the power of these ASCI computers and exploiting their full potential requires the development of scalable numerical algorithms, which for many applications is a significant challenge.

Part of the ASCI initiative is the development at Sandia of Salinas, a massively parallel implicit structural dynamics code aimed at providing a scalable computational workhorse for highly accurate structural dynamic models. Such large-scale finite element models require significant computational effort, but provide important information including, vibrational loads for components within larger systems, design optimization, frequency response information for guidance and space systems, modal data necessary for active vibration control, and characterization data for structural health monitoring.

As in the case of many other ASCI software research and development projects, the success of Salinas hinges on its ability to deliver scalable performance results. However, unlike many other ASCI computational efforts, Salinas is an implicit code and therefore requires, among others, a scalable equation solver in order to meet its objectives. Because all three ASCI machines are massively parallel computers with thousands of processors, our definition of scalability here is the ability of an algorithm implemented on an ASCI system to solve an n-times larger problem using an n-times larger number of processors in a *nearly* constant CPU time. Achieving such a scalability requires not only a parallel hardware with relatively inexpensive interprocessor communication costs, but most importantly an equation solver that is (a) numerically scalable — that is, with an arithmetic complexity that grows *almost* linearly with the problem size, and (b) amenable to a scalable parallel implementation — that is, which can exploit as large a number of processors as possible while incurring relatively small interprocessor communication costs. Such a stringent definition of scalability rules out sparse

direct solvers because their arithmetic complexity is a nonlinear function of the problem size. Furthermore, for large-scale three-dimensional structural problems with tens of millions of degrees of freedom (d.o.f.), the memory requirements of sparse direct solvers can overwhelm even the largest of the current ASCI machines. This is rather unfortunate because sparse direct methods offer otherwise a robustness that is not matched by any iterative algorithm. On the other hand, several multilevel [1] iterative schemes such as multigrid [2–4] algorithms and domain decomposition methods with coarse auxiliary problems [5] can be characterized by a nearly linear arithmetic complexity, or an iteration count that depends only weakly on the size of the problem to be solved. Such algorithms are prime candidates for a scalable equation solver. For Salinas, the Finite Element Tearing and Interconnecting (FETI) [6–12] solver was chosen because of its underlying mechanical concepts, as well as its potential for delivering a scalable performance.

FETI is a domain decomposition based iterative method with Lagrange multipliers. In its simplest form, it is also known as the one-level FETI method, and can be described as a two-step preconditioned conjugate gradient (PCG) algorithm where subdomain problems with Dirichlet (displacement) boundary conditions are solved in the preconditioning step, and related subdomain problems with Neumann (traction) boundary conditions are solved in a second step. The one-level FETI method incorporates a relatively small size auxiliary problem that is based on the subdomain rigid body modes. This coarse problem propagates the error globally during the PCG iterations and accelerates convergence.

For second-order elasticity problems discretized by plane stress/strain and/or solid elements, the condition number of the FETI interface problem preconditioned by the Dirichlet preconditioner [8] grows asymptotically as

$$\kappa \;=\; O\,(1+\log^2\,(\frac{H}{h}\,)) \tag{1}$$

where h denotes the mesh size, and H denotes the subdomain size (Fig. 1). Note that h, H, and h/H are indirect measures of the problem size, the number of subdomains, and the subdomain problem size, respectively. Hence, the condition number estimate (1) establishes the numerical scalability of the FETI method with respect to both the problem size and the number of subdomains. In particular, it proves that in theory, when the mesh discretization is refined, and the number of subdomains is increased as to maintain a constant number of elements per subdomain, the number of FETI iterations required for convergence remains asymptotically constant. This theoretical result has been demonstrated in practice for numerous applications [8–10]. The parallel scalability of the FETI

method has also been demonstrated on various parallel computers with a number of processors ranging between 2 and 128 [13,14]. For fourth-order plate and shell problems, the condition number estimate (1) also holds when the rigid body based coarse problem is enriched by the subdomain corner modes [11,12]. In that case, the FETI method is transformed into a genuine two-level algorithm known as the two-level FETI method.

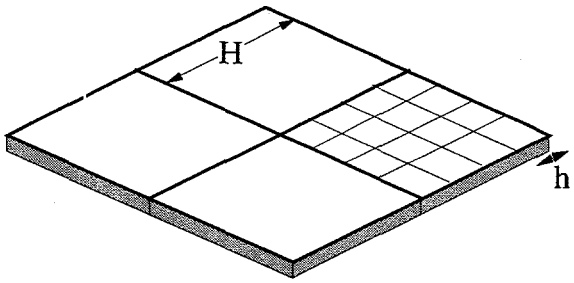

Fig. 1. Mesh size and subdomain size

An important issue in multilevel methods that pertains to parallel scalability is the solution of the lowest level problem, which for domain decomposition methods corresponds to the coarse problem. The size of this coarse problem increases with the number of subdomains. Initially, it was advocated to solve the FETI coarse problems iteratively, using a CG algorithm that is optimized for the solution of problems with repeated right hand-sides [20,21]. That approach was motivated by the fact that the CG method requires only matrix-vector products that can be performed in parallel at the subdomain level, and which necessitate only short range communication between neighboring subdomains. For small mesh partitions and therefore small size coarse problems, it was shown that such a strategy is computationally efficient and allows both one-level and two-level FETI solvers to achieve parallel scalability [10,12]. However, the modified CG algorithm described in [20,21] is not numerically scalable with respect to the the size of the coarse problem, and therefore is not suitable for problems involving a large number of subdomains. Given that the most practical way for implementing domain decomposition methods on distributed memory parallel processors is to generate and assign one or several subdomains to each processor, it follows that the specific iterative solution strategy described in [20,21] is not suitable for ASCI computational platforms. When the given problem is partitioned into a large number of subdomains, it was shown in [14] that for shared memory multiprocessors, solving the FETI coarse problems by a direct method is computationally efficient. Hence, a first objective of this paper is to revisit this issue in the context of the Salinas code, ASCI structural problems, and ASCI computational hardware.

Strictly speaking, the condition number estimate (1) holds for uniform mesh discretizations, uniform mesh partitions with a perfect subdomain aspect ratio,

and homogeneous problems — for example, structural problems with a single material, or different materials but with similar constitutive properties. However, it was shown in [15] that in practice, the numerical scalability of the FETI method holds for irregular discretizations and arbitrary mesh partitions, as long as the subdomains have reasonable aspect ratios. Algorithms for generating subdomains with reasonable aspect ratios can be found in [15–17]. For heterogeneous problems — for example, structural problems involving materials whose constitutive properties differ by several orders of magnitude — an improved coarse problem was proposed in [9] for preserving the numerical scalability of the FETI method. This alternative coarse problem was further investigated in [18] for model structural applications. More recently, a simple and virtually no-cost extension of the FETI preconditioner was proposed in [19] for addressing highly heterogeneous structural problems, but was also mainly verified on academic applications. Since ASCI structural problems are typically heterogeneous, a second objective of this paper is to construct a general strategy that combines both developments exposed in [9] and [19] for addressing the treatment by FETI of structural heterogeneities, and validate it for a realistic ASCI application problem.

Finally, a third objective of this paper is to report on preliminary performance results obtained on a thousand-processor configuration of the ASCI Option Red supercomputer, by the Salinas code equipped with the FETI solver, for various ASCI benchmark and real problems.

For this purpose, the remainder of this paper is organized as follows. In Section 2, we overview the one-level FETI method as our initial effort focuses on three-dimensional solid structures. In Section 3, we discuss a revised strategy for solving the subdomain rigid body mode based FETI coarse problem on a massively parallel processor. We also report some scalability results of the FETI method on the ASCI Option Red machine configured with as many as one thousand processors, for benchmark problems with as many as 5 million d.o.f. In Section 4, we consider the issue of structural heterogeneities and present a strategy for addressing them when using the FETI solver. In Section 5, we apply the FETI method to the finite element analysis on the ASCI Option Red supercomputer of a reentry vehicle, and in Section 6 we conclude this paper.

2. Overview of the one-level FETI method

Stress analyses, implicit linear and a large class of implicit nonlinear dynamic analyses, and vibration (eigenvalue) analyses lead to the solution of one or several systems of equations of the form

$$\mathbf{Ku} = \mathbf{f} \tag{2}$$

where $\mathbf{K}$ is in general a symmetric positive definite matrix, $\mathbf{u}$ is a vector of generalized displacements, and $\mathbf{f}$ a vector of generalized forces. In the FETI method, the structure's computational domain Ω is partitioned into N_s *non-overlapping* subdomains $\Omega^{(s)}$, and Lagrange multipliers $\boldsymbol{\lambda}$ are introduced at the subdomain interfaces to enforce the compatibility of the subdomain generalized displacement field $\mathbf{u}^{(s)}$. Consequently, Eq. (2) above is transformed into the equivalent set of equations

$$\begin{aligned} \mathbf{K}^{(s)}\mathbf{u}^{(s)} &= \mathbf{f}^{(s)} - \mathbf{B}^{(s)^T}\boldsymbol{\lambda} \qquad s = 1, \ldots N_s \\ \sum_{s=1}^{N_s} \mathbf{B}^{(s)}\mathbf{u}^{(s)} &= 0 \end{aligned} \tag{3}$$

where for each subdomain $\Omega^{(s)}$, $\mathbf{K}^{(s)}$ denotes its generalized stiffness matrix, $\mathbf{f}^{(s)}$ its vector of generalized forces, and $\mathbf{B}^{(s)}$ the signed Boolean matrix that extracts from a subdomain vector $\mathbf{v}^{(s)}$ its signed ($\pm$) restriction to the subdomain interface boundary. The first of Eqs. (3) expresses the local equilibrium of the subdomains $\Omega^{(s)}$, and the second of Eqs. (3) the continuity of the subdomain generalized displacement fields across the subdomain interfaces.

The general solution of the first of Eqs. (3) can be written as

$$\mathbf{u}^{(s)} = \mathbf{K}^{(s)^+}\left(\mathbf{f}^{(s)} - \mathbf{B}^{(s)^T}\boldsymbol{\lambda}\right) + \mathbf{R}^{(s)}\boldsymbol{\alpha}^{(s)} \tag{4}$$

where $\mathbf{K}^{(s)^+}$ denotes the inverse of $\mathbf{K}^{(s)}$ if $\Omega^{(s)}$ has sufficient Dirichlet boundary conditions to prevent $\mathbf{K}^{(s)}$ from being singular, or a generalized inverse of $\mathbf{K}^{(s)}$ if $\Omega^{(s)}$ is a floating subdomain. In the latter case, the columns of $\mathbf{R}^{(s)}$ represent the rigid body (or more generally the zero energy) modes of $\Omega^{(s)}$, i.e. $\mathbf{R}^{(s)} = \ker \mathbf{K}^{(s)}$, and $\boldsymbol{\alpha}^{(s)}$ is the set of amplitudes that specifies the contribution of the null space $\mathbf{R}^{(s)}$ to the solution $\mathbf{u}^{(s)}$. These coefficients can be determined by requiring that each subdomain problem be mathematically solvable — that is, each floating subdomain be self-equilibrated — which can be written as

$$\mathbf{R}^{(s)^T}\left(\mathbf{f}^{(s)} - \mathbf{B}^{(s)^T}\boldsymbol{\lambda}\right) = 0 \tag{5}$$

Substituting Eq. (4) into the compatibility equation and exploiting the solvability condition (5) transforms problem (3) into the interface problem

$$\begin{bmatrix} \mathbf{F}_I & -\mathbf{G}_I \\ -\mathbf{G}_I{}^T & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{\lambda} \\ \boldsymbol{\alpha} \end{bmatrix} = \begin{bmatrix} \mathbf{d} \\ -\mathbf{e} \end{bmatrix} \tag{6}$$

where

$$\begin{aligned}
\mathbf{F}_I &= \sum_{s=1}^{N_s} \mathbf{B}^{(s)} \mathbf{K}^{(s)^+} \mathbf{B}^{(s)^T} \\
\mathbf{d} &= \sum_{s=1}^{N_s} \mathbf{B}^{(s)} \mathbf{K}^{(s)^+} \mathbf{f}^{(s)} \\
\mathbf{G}_I &= \left[\mathbf{G}_I^{(1)} \quad \ldots \quad \mathbf{G}_I^{(N_f)} \right] = \left[\mathbf{B}^{(1)} \mathbf{R}^{(1)} \quad \ldots \quad \mathbf{B}^{(N_f)} \mathbf{R}^{(N_f)} \right] \\
\boldsymbol{\alpha} &= \left[\boldsymbol{\alpha}^{(1)^T} \quad \ldots \quad \boldsymbol{\alpha}^{(N_f)^T} \right]^T \\
\mathbf{e} &= \left[\mathbf{f}^{(1)^T} \mathbf{R}^{(1)} \quad \ldots \quad \mathbf{f}^{(N_s)^T} \mathbf{R}^{(N_f)} \right]^T
\end{aligned} \tag{7}$$

and where N_f denotes the number of floating subdomains.

The above indefinite interface problem (6) can be transformed into a semi-definite system of equations as follows. Let $\mathbf{Q}$ be any symmetric matrix for which the product $\mathbf{G}_I^T \mathbf{Q} \mathbf{G}_I$ is invertible. The self-equilibrium condition $\mathbf{G}_I^T \boldsymbol{\lambda} = \mathbf{e}$ can be eliminated from Eqs. (6) by introducing the splitting

$$\boldsymbol{\lambda} = \boldsymbol{\lambda}^0 + \mathbf{P}(\mathbf{Q}) \bar{\boldsymbol{\lambda}} \tag{8}$$

where $\boldsymbol{\lambda}^0$ is a particular solution of $\mathbf{G}_I^T \boldsymbol{\lambda} = \mathbf{e}$ of the form

$$\boldsymbol{\lambda}^0 = \mathbf{Q} \mathbf{G}_I (\mathbf{G}_I^T \mathbf{Q} \mathbf{G}_I)^{-1} \mathbf{e} \tag{9}$$

and $\mathbf{P}(\mathbf{Q})$ is a projector of the form

$$\mathbf{P}(\mathbf{Q}) = \mathbf{I} - \mathbf{Q} \mathbf{G}_I (\mathbf{G}_I^T \mathbf{Q} \mathbf{G}_I)^{-1} \mathbf{G}_I^T \tag{10}$$

Note that $\mathbf{P}$ satisfies

$$\mathbf{P}^2 = \mathbf{P} \qquad \mathbf{G}_I^T \mathbf{P} = 0 \tag{11}$$

Substituting Eq. (8) into the first of Eqs. (6) and premultiplying that equation by $\mathbf{P}^T$ transforms the indefinite interface problem (6) into the projected interface problem

$$\left(\mathbf{P}^T \mathbf{F}_I \mathbf{P} \right) \bar{\boldsymbol{\lambda}} = \mathbf{P}^T \left(\mathbf{d} - \mathbf{F}_I \boldsymbol{\lambda}^0 \right) \tag{12}$$

which is symmetric positive semi-definite for any given matrix $\mathbf{Q}$.

The one-level FETI method consists in transforming the original global problem (2) into the symmetric positive semi-definite interface problem (12), and solving the latter system of equations by a PCG algorithm. Note that the projector $\mathbf{P}$ contains the matrix $(\mathbf{G}_I^T \mathbf{Q} \mathbf{G}_I)^{-1}$, which is symmetric when $\mathbf{Q}$ is symmetric.

In general, each subdomain has at most 6 rigid body modes, and therefore the dimension of $\mathbf{G}_I^T\mathbf{Q}\mathbf{G}_I$ is at most equal to $6N_s$. Hence, this matrix defines an auxiliary coarse problem that couples all the subdomain computations, and which was proved to propagate the error globally and accelerate convergence [8,22].

Two preconditioners have been proposed in the literature for the FETI method: the mathematically optimal Dirichlet preconditioner $\mathbf{P}\overline{\mathbf{F}}_I^{D^{-1}}\mathbf{P}^T$ introduced in [8], and the computationally economical even if not numerically scalable lumped preconditioner $\mathbf{P}\overline{\mathbf{F}}_I^{L^{-1}}\mathbf{P}^T$ proposed in earlier works [6,7]. If each subdomain generalized stiffness matrix is partitioned as

$$\mathbf{K}^{(s)} = \begin{bmatrix} \mathbf{K}_{ii}^{(s)} & \mathbf{K}_{ib}^{(s)} \\ \mathbf{K}_{ib}^{(s)^T} & \mathbf{K}_{bb}^{(s)} \end{bmatrix} \tag{13}$$

where the subscripts i and b designate the subdomain interior and interface boundary d.o.f., respectively, then the component $\overline{\mathbf{F}}_I^{D^{-1}}$ of the Dirichlet preconditioner can be written as

$$\begin{aligned} \overline{\mathbf{F}}_I^{D^{-1}} &= \sum_{s=1}^{N_s} \mathbf{W}^{(s)}\mathbf{B}^{(s)} \begin{bmatrix} 0 & 0 \\ 0 & \mathbf{S}_{bb}^{(s)} \end{bmatrix} \mathbf{B}^{(s)^T}\mathbf{W}^{(s)} \\ \text{where} \\ \mathbf{S}_{bb}^{(s)} &= \mathbf{K}_{bb}^{(s)} - \mathbf{K}_{ib}^{(s)^T}\mathbf{K}_{ii}^{(s)^{-1}}\mathbf{K}_{ib}^{(s)} \end{aligned} \tag{14}$$

and the component $\overline{\mathbf{F}}_I^{L^{-1}}$ of the lumped preconditioner can be written as

$$\overline{\mathbf{F}}_I^{L^{-1}} = \sum_{s=1}^{N_s} \mathbf{W}^{(s)}\mathbf{B}^{(s)} \begin{bmatrix} 0 & 0 \\ 0 & \mathbf{K}_{bb}^{(s)} \end{bmatrix} \mathbf{B}^{(s)^T}\mathbf{W}^{(s)} \tag{15}$$

In the above expressions of $\overline{\mathbf{F}}_I^{D^{-1}}$ and $\overline{\mathbf{F}}_I^{L^{-1}}$, $\mathbf{W}^{(s)}$ is a diagonal "scaling" matrix. In its simplest form, $\mathbf{W}^{(s)}$ stores in each of its entries the inverse of the multiplicity of the corresponding interface node — that is, the inverse of the number of subdomains attached to that node [23,9,11]. For example, if the i-th Lagrange multiplier component $\boldsymbol{\lambda}(i)$ acts on an interface node that is shared between 2 subdomains, then $\mathbf{W}^{(s)}(i) = 1/2$; if it acts on an interface node that is shared by m subdomains, then $\mathbf{W}^{(s)}(i) = 1/m$. Such a matrix $\mathbf{W}^{(s)}$ is referred to as the "topological scaling" matrix.

Recently, both Dirichlet and lumped preconditioners have been extended to address more efficiently heterogeneous structural problems [19]. These extensions are simply obtained by redefining appropriately the scaling matrix $\mathbf{W}^{(s)}$.

Finally, we note that for homogeneous problems, the simplest choice $\mathbf{Q} = \mathbf{I}$ is the most computationally efficient. Most FETI computations reported in the literature have been performed with this trivial choice. However, it was shown in [9] that heterogeneous problems call for a matrix $\mathbf{Q}$ that is physically homogeneous to a generalized stiffness. For this reason, two alternative choices for $\mathbf{Q}$ were first proposed in [9]: $\mathbf{Q} = \mathbf{Q}^L = \overline{\mathbf{F}}_I^{L^{-1}}$, and $\mathbf{Q} = \mathbf{Q}^D = \overline{\mathbf{F}}_I^{D^{-1}}$. These choices were further investigated in [18] and shown to be effective for model heterogeneous problems.

3. Scalability results on the ASCI Option Red supercomputer

3.1. Implementation of Salinas and FETI on massively parallel distributed memory systems

Like FETI, Salinas is based on substructuring, and relies on the same concept of mesh partitioning. For this reason, interfacing both codes was a relatively straightforward task. Using an automatic mesh decomposer [29,30], a given finite element structural model is first decomposed into $N_s \geq N_p$ subdomains, where N_p denotes the target number of processors. The potential advantages of generating more subdomains than there are processors are discussed in [14,31], among other references. Next, the generated subdomains are re-arranged into N_p clusters containing each one or several subdomains, and each cluster is assigned to one processor. Most if not all Salinas and FETI computations can be performed concurrently at the subdomain level, and necessitate interprocessor communication only between neighboring clusters. As far as FETI is concerned, only the solution at each PCG iteration k of a coarse problem of the form

$$(\mathbf{G}_I^T \mathbf{Q} \mathbf{G}_I)\boldsymbol{\alpha}^k = \mathbf{G}_I^T \mathbf{w}^k \tag{16}$$

deserves special attention. Such a coarse problem is associated with a matrix-vector multiplication of the form $\mathbf{P}^T\mathbf{w}^k$ or $\mathbf{P}\mathbf{y}^k$, where $\mathbf{w}^k$ and $\mathbf{y}^k$ denote respectively two vectors generated by the PCG algorithm applied to the solution of the interface problem $(\mathbf{P}^T\mathbf{F}_I\mathbf{P})\bar{\boldsymbol{\lambda}} = \mathbf{P}^T(\mathbf{d} - \mathbf{F}_I\boldsymbol{\lambda}^0)$ (12). Hence, it arises twice at each FETI iteration. Before addressing this issue, we note that

- the system matrix $\mathbf{G}_I^T\mathbf{Q}\mathbf{G}_I$ is independent of the iteration number k. Only the right hand side vector $\mathbf{G}_I^T\mathbf{w}^k$ varies throughout the FETI iterations.
- for any $\mathbf{Q}$, the system matrix $\mathbf{G}_I^T\mathbf{Q}\mathbf{G}_I$ has the same size which depends on the number of floating subdomains N_f, and the dimensions of the null spaces $\ker\mathbf{K}^{(s)}$, $s = 1, \ldots, N_f$. In general, the size of $\mathbf{G}_I^T\mathbf{Q}\mathbf{G}_I$ is of the order of $6N_s$.
- for $\mathbf{Q} = \mathbf{I}$, $\mathbf{G}_I^T\mathbf{G}_I$ is a sparse symmetric positive matrix. Its sparsity pattern is dictated by the connectivity of the mesh partition, and is identical to the sparsity pattern of any finite element matrix obtained by treating each subdomain as a "superelement". More specifically, $\mathbf{G}_I^{(s)^T}\mathbf{G}_I^{(q)} \neq 0$ if and only if $\Omega^{(s)}$ and $\Omega^{(q)}$ are neighboring subdomains.
- for $\mathbf{Q} = \mathbf{Q}^L = \overline{\mathbf{F}}_I^{L^{-1}}$ and $\mathbf{Q} = \mathbf{Q}^D = \overline{\mathbf{F}}_I^{D^{-1}}$, $\mathbf{G}_I^T\mathbf{Q}\mathbf{G}_I$ is also a sparse symmetric positive matrix, and its sparsity pattern is also dictated by the connectivity of the mesh partition. However, in these two cases $\mathbf{G}_I^T\mathbf{Q}\mathbf{G}_I$ is

slightly more populated than $\mathbf{G}_I^T\mathbf{G}_I$. More specifically, $\mathbf{G}_I^{(s)^T}\mathbf{Q}\mathbf{G}_I^{(q)} \neq 0$ if and only if $\Omega^{(q)}$ is a neighbor of $\Omega^{(s)}$, or the neighbor of a neighbor of $\Omega^{(s)}$.

- for all three choices of $\mathbf{Q}$ specified above, $\mathbf{G}_I^T\mathbf{Q}\mathbf{G}_I$ can be assembled in parallel using only subdomain level computations, and a small amount of interprocessor communication between processors mapped onto neighboring clusters of subdomains.

As mentioned in the introduction, we consider here solving the coarse problem (16) by a direct method. Such a strategy improves the robustness of the FETI method, but complicates its implementation on massively parallel distributed memory systems. As stated earlier, parallel sparse direct algorithms do not scale well in the sense defined in this paper, particularly for these small size coarse problems. Furthermore, we note that because the system matrix $\mathbf{G}_I^T\mathbf{Q}\mathbf{G}_I$ needs be factored only once, but the coarse problem (16) must be solved twice at each FETI iteration, it is essential to focus on a strategy that addresses not only the factorization of $\mathbf{G}_I^T\mathbf{Q}\mathbf{G}_I$, but most importantly the subsequent forward and backward substitutions. Indeed, the scalable parallelization of the direct solution of sparse lower and upper triangular systems is even more challenging than that of the factorization of a sparse matrix.

For all of the above reasons, we consider here the following approach for solving the coarse problem (16) on a massively parallel distributed memory system such as the ASCI Option Red supercomputer. For the sake of notational simplicity, but without any loss of generality, we assume in the following algorithmic description that each floating subdomain has exactly 6 rigid body modes.

a) form $\mathbf{G}_I^T\mathbf{Q}\mathbf{G}_I$ in parallel and replicate this relatively small size sparse matrix in each processor.

b) request that each processor factor $\mathbf{G}_I^T\mathbf{Q}\mathbf{G}_I$.

c) compute in parallel a distributed inverse of $\mathbf{G}_I^T\mathbf{Q}\mathbf{G}_I$ as follows. For each floating subdomain $\Omega^{(j)}$ assigned to processor p_j, request that p_j performs 6 forward and backward substitutions to solve

$$(\mathbf{G}_I^T\mathbf{Q}\mathbf{G}_I)\mathbf{X}_j = \mathbf{I}_j \tag{17}$$

where $\mathbf{I}_j$ contains the 6 columns of the identity matrix $\mathbf{I}$ that are assigned to subdomain $\Omega^{(j)}$ in conjunction with its 6 rigid body modes $\mathbf{R}^{(j)}$.

d) at each FETI iteration k, solve each coarse problem of the form given in (16) by performing a parallel distributed matrix-vector multiplication. Indeed,

from Eqs. (7,16,17) it follows that

$$\boldsymbol{\alpha}^k \;=\; \mathbf{X}(\mathbf{G}_I^T\mathbf{w}^k) \;=\; \sum_{j=1}^{j=N_f} \mathbf{X}_j[\mathbf{G}_I^{(j)^T}\mathbf{w}^k] \tag{18}$$

which shows that the evaluation of $\boldsymbol{\alpha}^k$ can be performed using subdomain-by-subdomain parallel computations and requires only one global range communication.

The strategy outlined above for solving the coarse problem (16) is essentially composed of three sequences of embarrassingly parallel computations.

The first one has two caveats. From a computational viewpoint, requesting that all processors perform the factorization of the same matrix $\mathbf{G}_I^T\mathbf{Q}\mathbf{G}_I$ is equivalent to serializing this computation. This serialization does not significantly affect the overall performance of FETI, as long as the cost of the factorization of $\mathbf{G}_I^T\mathbf{Q}\mathbf{G}_I$ is negligible compared to the cost of the other FETI operations — that is, as long as the number of subdomains is below a certain critical value N_s^{cr}. However, because of Amdahl's law, there also exists a certain number of processors N_p^{cr} beyond which this serialization will prevent FETI from scaling well on a massively parallel system. Furthermore, given that the size of $\mathbf{G}_I^T\mathbf{Q}\mathbf{G}_I$ increases with the number of subdomains N_s, and that N_s increases with the number of processors N_p, there also exists a critical number of subdomains and/or processors beyond which storing $\mathbf{G}_I^T\mathbf{Q}\mathbf{G}_I$ in a single processor of a local memory system will not be feasible. However, note that after the $\mathbf{X}_j$ column matrices have been computed, $\mathbf{G}_I^T\mathbf{Q}\mathbf{G}_I$ can be deleted, which frees memory for other usage, for example, by Salinas.

The second sequence (c) of embarrassingly parallel computations is an effective one from both computational complexity and parallel scalability viewpoints. The third sequence (d) is also perfectly scalable from a parallel processing viewpoint. It is also computationally efficient if the size of each cluster of subdomains is such that the total number of column matrices $\mathbf{X}_j$ assigned to a processor p_j is comparable to the average number of nonzero entries in a row of the factors of the sparse matrix $\mathbf{G}_I^T\mathbf{Q}\mathbf{G}_I$. In particular, if one and only one subdomain is assigned to each processor ($N_s = N_p$), the embarrassingly parallel steps (c) and (d) are both numerically and parallel-wise scalable.

In summary, one can reasonably expect that the FETI method equipped with the coarse problem solver described above will scale well on massively parallel distributed memory systems, up to a certain problem and/or machine size (number of processors) beyond which the storage scheme and factorization method of the

coarse matrix $\mathbf{G}_I^T \mathbf{Q} \mathbf{G}_I$ will need to be revisited. Hence, a first objective of this work is to assess this limit in the context of the solution of three-dimensional second-order elasticity problems on the ASCI Option Red supercomputer.

3.2. The ASCI Option Red supercomputer

The ASCI initiative supports the ASCI Option Red supercomputer, a massively parallel processor with a distributed memory multiple instruction and multiple data architecture, as well as the ASCI Option Blue Mountain and ASCI Option Blue Pacific supercomputers. The ASCI Option Red and Blue Mountain systems run MP LINPACK, one of the computer industry's standard speed tests for large systems, at 1.3 and 1.6 Teraflops respectively [24].

The ASCI Option Red supercomputer, also known as the Intel Teraflops machine, is the first large-scale supercomputer built entirely of commodity, commercial, off-the-shelf components. It has 4,536 compute and 72 service nodes each with 2 Pentium Pro processors, 594 Gbytes of real memory, and two independent 1-Terabyte disk systems. It occupies 1600 sq. ft. of floor-space (Fig. 2). The system's 9,216 Pentium Pro processors are connected by a $38 \times 32 \times 2$ mesh.

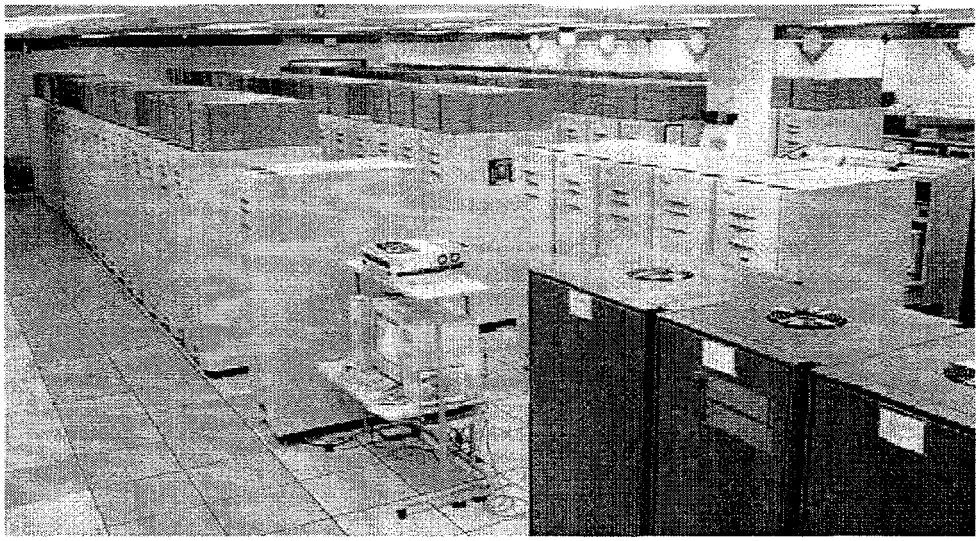

Fig. 2. The ASCI Option Red supercomputer

The Pentium Pro processor runs at 200 MHz and has a peak floating-point rate of 200 Mflops (million floating-point operations per second). It has separate on-chip data and instruction L1 caches of 8 Kbytes each. It has also an L2 cache of 256 Kbytes packaged with the CPU in a single dual-cavity PGA package. All

cache lines are 32 bytes wide. The system was delivered with 128 Mbytes of memory per node, but supports up to 256 Mbytes of memory per node. The two processors on each node support two on-board PCI interfaces; each of these interfaces provides 133 Mbytes/sec I/O bandwidth. The memory subsystem is structured as four rows of four independently controlled and sequentially interleaved banks of DRAM to produce up to 533 Mbytes/sec of data throughput. Each memory bank is 72 bits wide. The router supports bi-directional bandwidths of up to 800 Mbytes/sec over each of six ports. As many as four message streams can pass on any given port and at any given time.

Two UNIX-based operating systems collectively called the Teraflops OS run on the ASCI Option Red supercomputer and present a single system image to the user. Compute nodes run an efficient small operating system called Cougar [25–27]. Service nodes run POSIX 1003.1 and XPG3, and AT&T System V.3 and 4.3 BSD Reno VFS [28]. The file system is concentrated on a small set of specialized nodes that process I/O requests. Symbios RM20 Redundant Arrays of Independent Disks (RAIDs) are used for secondary storage. A Symbios RM20 RAID has two bays of ten drives each and two controllers. The disk drives are Seagate 4-GbytesBarracudas with a 3.5" form-factor [28].

3.3. Preliminary scalability results

Assessing the scalability (in the sense defined in this paper) of both the FETI method and its massively parallel implementation described in Section 3.2 requires generating, for a given problem, a sequence of finite element models where the total number of d.o.f. is increased proportionally to an increasing sequence of number of processors, in order to maintain the ratio problem size over machine size constant. Generating such a sequence of finite element models and the corresponding sequence of mesh partitions is in general a tedious task. For this reason, and because the numerical scalability of the FETI method has already been established and repeatedly demonstrated for realistic structural problems [9–14,31], we consider here two simple three-dimensional benchmark problems that are easy to generate and manipulate for scalability studies. Both benchmark problems correspond to homogeneous structures uniformly discretized by 8-noded brick elements, and partitioned into cubic subdomains. For this reason, we set $\mathbf{Q} = \mathbf{I}$ for both problems. In both cases, we generate the sequence of finite element meshes by fixing the size of each cubic subdomain, and increasing the number of subdomains N_s to match the target number of processors N_p. Hence, we consider here only the case $N_s = N_p$, because we consider large values of N_p ranging between 8 and 1000 processors. We fix the subdomain size to 1728 elements ($12 \times 12 \times 12$), which corresponds to the maximum subdomain size affordable by FETI for one of the two benchmark problems on a single processor

with 128 Mbytes of memory, after the Salinas memory requirements have been met. The two benchmark problems considered here differ as follows

1) in benchmark problem BP1, the structure has a cubic shape and is partitioned into $n \times n \times n$ subdomains (Fig. 3.). It is clamped at one end, and subjected to a distributed vertical load at the other.

2) in benchmark problem BP2, the structure has the shape of a rectangular parallelepiped and is partitioned into $2 \times 2 \times n$ subdomains (Fig. 4.). It is clamped at one end, and subjected to a distributed vertical load at the other.

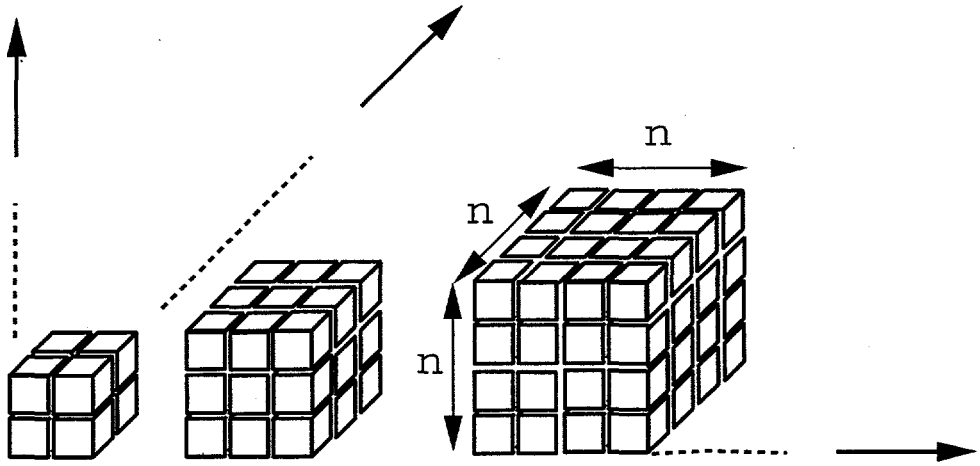

Fig. 3. The benchmark problem BP1

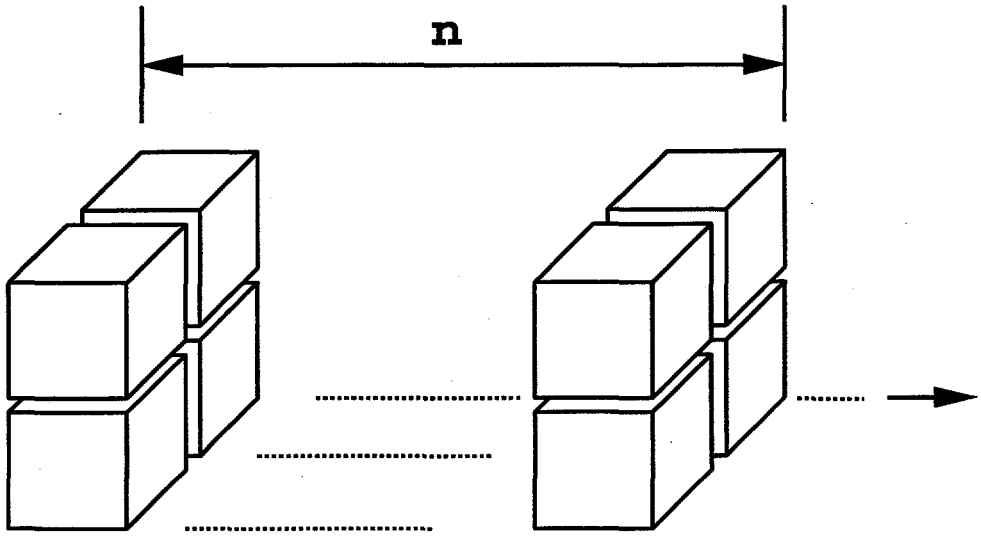

Fig. 4. The benchmark problem BP2

In both cases, the size of the $\mathbf{G}_I^T \mathbf{G}_I$ matrix increases linearly with the number of subdomains, and the amount of fill-in per row suffered during the factorization of this matrix grows with the number of the subdomains lying in a plane perpendicular to the main axis of the structure. Hence, for benchmark problem BP1, this amount of fill-in per row increases as $O\ (n^2)$, while for benchmark problem BP2 it remains constant. Since the parallel implementation of the one-level FETI method described in Section 3.2 calls for replicating the storage of $\mathbf{G}_I^T \mathbf{G}_I$ in every processor, and effectively serializes the factorization of this matrix, it follows that problems BP1 and BP2 provide a worst-case and best case scenarios, respectively, for the memory requirements and parallel scalability of the FETI method. In particular, given that the size of each subdomain is fixed to 1728 elements, that each processor of the target ASCI Option Red supercomputer has 128 Mbytes of memory only, and that Salinas has its own memory requirements that must be accommodated, the $O\ (n^2)$ growth of fill-in per row for the factorization of the coarse problem associated with the benchmark problem BP1 limits the number of subdomains that can be considered in this investigation to $N_s = 1000$, and therefore limits the number of processors to $N_p = 1000$. Of course, this number of subdomains and/or processors can be increased by decreasing the subdomain problem size. However, because access to more than 1000 processors on the ASCI Red Option machine is also a practical challenge by itself, we limit here our investigation of the performance and scalability of the FETI method on the ASCI Option Red supercomputer to a maximum number of 1000 processors.

We also note that most realistic problems are neither cube-shaped, nor parallelepiped-shaped. The decomposition of their meshes seldom generates perfectly load-balanced subdomains, or subdomains with a perfect aspect ratio. However, based on the arguments presented above, we can reasonably argue that the scalability results of FETI for the benchmark problems BP1 and BP2 provide lower and upper bounds of the scalability results to be expected for the solution by FETI on the ASCI Option Red supercomputer of more realistic problems.

Even though Salinas is primarily a structural dynamics implicit code, we report here on the scalability of this software equipped with the FETI solver for linear static analysis. This is because the complexity of solving a system of equations arising from one step of an implicit structural dynamic (large time-step) analysis, or from a static analysis, is essentially the same. For the optimization of FETI to the solution of repeated systems arising form the linear dynamic analysis or the eigenvalue analysis of a structure, we refer the reader to [10,20,21]. For both benchmark problems, we equip FETI with the topological scaling matrix $\mathbf{W}^{(s)}$. We report in Table 1 and Table 2 the performance results obtained for problem BP1 on a 1000-processor configuration of the ASCI Option Red supercomputer,

using the Dirichlet and lumped preconditioners, respectively. Similarly, we report in Table 3 and Table 4 the performance results obtained for problem BP2. In all cases, we use the following stopping criterion

$$||\mathbf{Ku} - \mathbf{f}||_2 \ < 10^{-6} \times ||\mathbf{f}||_2 \tag{19}$$

Table 1
Solution by FETI equipped with the Dirichlet preconditioner of the benchmark problem BP1 on the ASCI Option Red supercomputer

n	$N_p = N_s$	N_{dof}	N_{itr}	Factor $\mathbf{G}_I^T\mathbf{G}_I$	FETI	Salinas + FETI
2	8	46,875	14	0.001 sec	209 sec	336 sec
3	27	151,959	20	0.006 sec	216 sec	346 sec
4	64	352,947	25	0.05 sec	222 sec	355 sec
5	125	680,943	27	0.3 sec	225 sec	358 sec
6	216	1,167,051	30	1.1 sec	229 sec	365 sec
7	343	1,842,375	31	2.9 sec	235 sec	367 sec
8	512	2,738,019	33	5.8 sec	239 sec	380 sec
9	729	3,885,087	33	14.9 sec	252 sec	405 sec
10	1,000	5,314,683	34	32.4 sec	275 sec	413 sec

Table 2
Solution by FETI equipped with the lumped preconditioner of the benchmark problem BP1 on the ASCI Option Red supercomputer

n	$N_p = N_s$	N_{dof}	N_{itr}	Factor $\mathbf{G}_I^T\mathbf{G}_I$	FETI	Salinas + FETI
2	8	46,875	27	0.001 sec	140 sec	267 sec
3	27	151,959	36	0.006 sec	148 sec	278 sec
4	64	352,947	45	0.05 sec	155 sec	288 sec
5	125	680,943	48	0.3 sec	157 sec	290 sec
6	216	1,167,051	51	1.1 sec	161 sec	297 sec
7	343	1,842,375	55	2.9 sec	167 sec	299 sec
8	512	2,738,019	55	5.8 sec	171 sec	312 sec
9	729	3,885,087	58	14.9 sec	187 sec	340 sec
10	1,000	5,314,683	60	32.4 sec	216 sec	399 sec

Table 3
Solution by FETI equipped with the Dirichlet preconditioner of the benchmark problem BP2 on the ASCI Option Red supercomputer

n	$N_p = N_s$	N_{dof}	N_{itr}	Factor $\mathbf{G}_I^T\mathbf{G}_I$	FETI	Salinas + FETI
2	8	46,875	14	0.001 sec	209 sec	336 sec
7	28	159,375	18	0.007 sec	214 sec	343 sec
16	64	316,875	18	0.018 sec	215 sec	354 sec
31	124	699,375	18	0.038 sec	216 sec	347 sec
54	216	1,216,875	19	0.073 sec	216 sec	350 sec
86	344	1,936,875	19	0.131 sec	217 sec	351 sec
128	512	2,881,875	19	0.203 sec	218 sec	354 sec
182	728	4,096,875	19	0.298 sec	218 sec	355 sec
250	1,000	5,626,875	19	0.414 sec	222 sec	360 sec

Table 4
Solution by FETI equipped with the lumped preconditioner of the benchmark problem BP2 on the ASCI Option Red supercomputer

n	$N_p = N_s$	N_{dof}	N_{itr}	Factor $\mathbf{G}_I^T\mathbf{G}_I$	FETI	Salinas + FETI
2	8	46,875	27	0.001 sec	140 sec	267 sec
7	28	159,375	30	0.007 sec	142 sec	271 sec
16	64	316,875	31	0.018 sec	143 sec	282 sec
31	124	699,375	31	0.038 sec	143 sec	274 sec
54	216	1,216,875	31	0.073 sec	144 sec	278 sec
86	344	1,936,875	31	0.131 sec	144 sec	278 sec
128	512	2,881,875	32	0.203 sec	146 sec	282 sec
182	728	4,096,875	33	0.298 sec	147 sec	284 sec
250	1,000	5,626,875	34	0.414 sec	151 sec	289 sec

The performance results reported in Tables 1–4 show that

- the FETI method equipped with the Dirichlet preconditioner achieves numerical scalability (constant asymptotic iteration count) for both benchmark problems BP1 and BP2.

- when equipped with the lumped preconditioner, the FETI method achieves numerical scalability for problem BP2. It also exhibits a reasonable numerical scalability for problem BP1. For both benchmark problems, the FETI method performs on average 1.6 times more iterations when equipped with the lumped preconditioner than when equipped with the Dirichlet preconditioner. However when equipped with the lumped preconditioner, the FETI method is on average 1.3 times faster (problem BP1) and 1.5 times faster (problem BP2) than when equipped with the Dirichlet preconditioner. This demonstrates the computational efficiency of the lumped preconditioner for the solution of second-order elasticity problems by the FETI method.
- for problem BP1, the cost of the factorization of the matrix $\mathbf{G}_I^T \mathbf{G}_I$ — which is the only sequential operation performed by the current implementation of FETI on massively parallel local memory machines — is shown to increase dramatically with the number of the subdomains and processors, as $O\left(N_s^{7/3} = N_p^{7/3}\right)$, which is consistent with the $O\left(n^2\right)$ $\left(n = N_s^{1/3}\right)$ growth of the fill-in per row predicted for the factorization of this matrix for problem BP1. Nevertheless, the results reported in Table 2 show that for the benchmark problem BP1, the FETI method equipped with the lumped preconditioner solves 5,314,683 equations in 216 seconds CPU on a 1000-processor configuration of the ASCI Option Red supercomputer.
- for problem BP2, the CPU time consumed by the sequential factorization of the coarse problem of the FETI method is reported to grow only linearly with the number of subdomains. This is consistent with our analytical prediction that is based on the fact that the size of $\mathbf{G}_I^T \mathbf{G}_I$ grows linearly with the number of subdomains, and the fact that for problem BP2, the fill-in per row suffered during the factorization of this matrix is independent of the number of subdomains. For the benchmark problem BP2, the FETI method equipped with the lumped preconditioner solves 5,626,875 equations on 1000 processors in 151 seconds.

In order to quantify the scalability of the current implementation of the FETI method on the ASCI Option Red supercomputer, we introduce the following definition of the speed-up

$$Sp = \frac{8 \times T_8}{T_{N_p}} \times \frac{N_{dof_{N_s}}}{N_{dof_8}} \tag{20}$$

where T_8 and T_{N_p} denote respectively the CPU timings corresponding to 8 and N_p processors, and N_{dof_8} and $N_{dof_{N_s}}$ denote respectively the sizes (in d.o.f.) of the global problems corresponding to 8 and N_s subdomains. Here, the case $N_p = 8$ is taken as a reference point. Note that the above definition of the

speed-up is a *strict* one: it accounts for both concepts of numerical and parallel scalability. It assesses the combined performances of the given algorithm, its parallel implementation, and the parallel hardware on which this algorithm is executed. For the benchmark problems BP1 and BP2, the speed-ups achieved by the FETI method are reported Fig. 5 for the case of the Dirichlet preconditioner, and in Fig. 6 for the case of the lumped preconditioner.

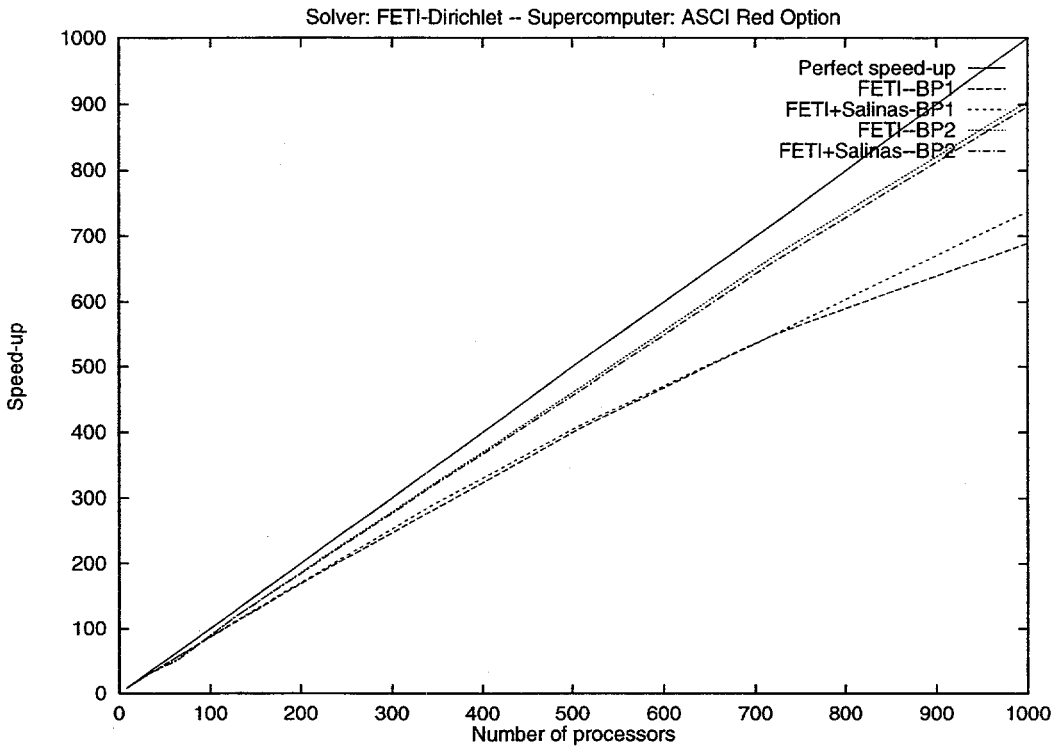

Fig. 5. Scalability results of the FETI method equipped with the Dirichlet preconditioner on the ASCI Option Red supercomputer

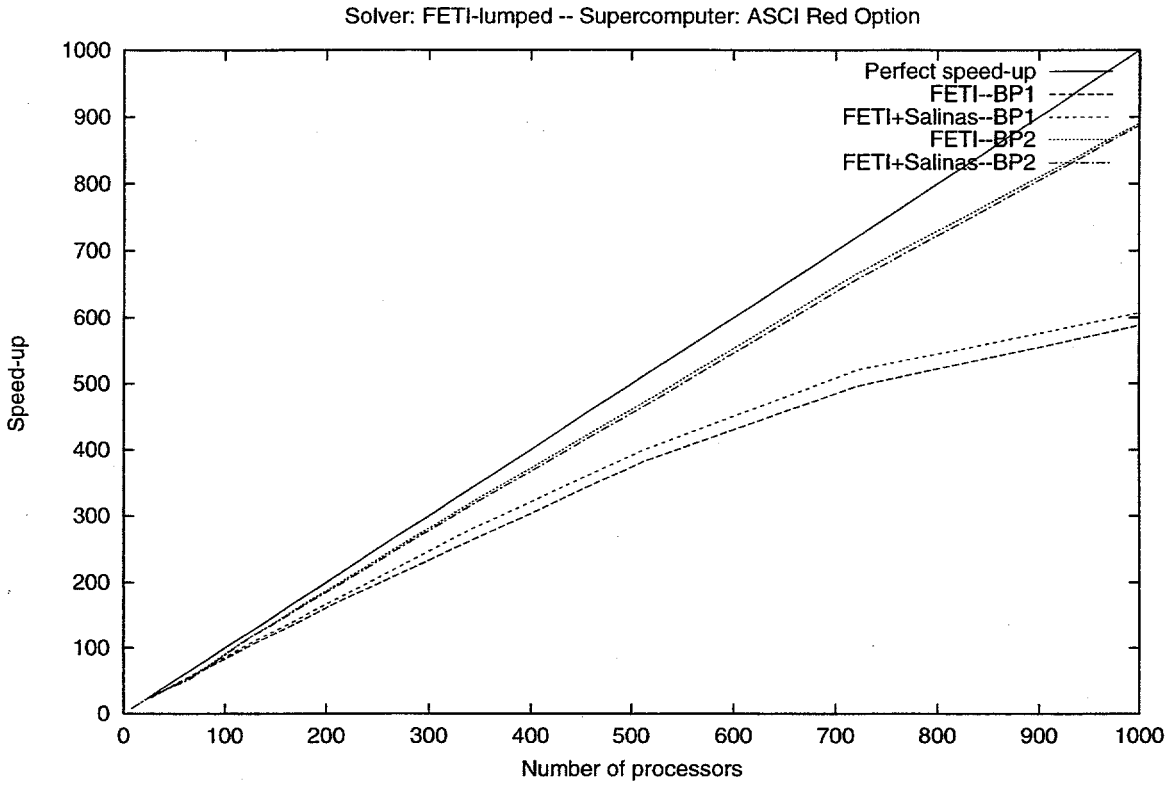

Fig. 6. Scalability results of the FETI method equipped with the lumped preconditioner on the ASCI Option Red supercomputer

From Fig. 5, we conclude that when equipped with the Dirichlet preconditioner, the one-level FETI method implemented on a 1000-processor configuration of the ASCI Option Red machine, as described in Section 3.1 can be expected to achieve for realistic second-order elasticity problems a speed-up in the range of 700 to 900, and therefore an efficiency (speed-up per processor) ranging between 70% and 90%. The lower bound of this trend for efficiency suggests that for $N_p > 1000$, maintaining this level of speed-up will require the parallelization of the factorization of the matrix $\mathbf{G}_I^T \mathbf{G}_I$ of the coarse problem in order to address the effect of Amdahl's law.

4. Highly heterogeneous structural problems

Benchmark problems BP1 and BP2 are homogeneous problems, and their mesh partitions are characterized by perfect subdomain aspect ratios. This ex-

plains the relatively low iteration counts reported in Tables 1–4 for their solution by the FETI method.

The structural problems targeted by the ASCI initiative are typically heterogeneous, with some material properties differing by as much as five orders of magnitude. For such problems, FETI delivers the same parallel scalability as for the benchmark problems BP1 and BP2. However, maintaining the numerical scalability of the FETI method for highly heterogeneous structural problems requires equipping it with different choices for $\mathbf{Q}$ and $\mathbf{W}^{(s)}$ than the identity and topological scaling matrices, respectively. For example, it was argued in [9] that for heterogeneous structural problems, $\mathbf{Q}$ must be chosen as a matrix that captures the different stiffnesses of the various subdomains. It was also shown in [9] that the two specific choices $\mathbf{Q}^D = \overline{\mathbf{F}}_I^{D^{-1}}$ and $\mathbf{Q}^L = \overline{\mathbf{F}}_I^{L^{-1}}$ not only meet this requirement, but also offer a computational advantage because $\mathbf{P}(\overline{\mathbf{F}}_I^{D^{-1}})\overline{\mathbf{F}}_I^{D^{-1}}\mathbf{P}^T(\overline{\mathbf{F}}_I^{D^{-1}}) = \overline{\mathbf{F}}_I^{D^{-1}}\mathbf{P}^T(\overline{\mathbf{F}}_I^{D^{-1}})$, and $\mathbf{P}(\overline{\mathbf{F}}_I^{L^{-1}})\overline{\mathbf{F}}_I^{L^{-1}}\mathbf{P}^T(\overline{\mathbf{F}}_I^{L^{-1}}) = \overline{\mathbf{F}}_I^{L^{-1}}\mathbf{P}^T(\overline{\mathbf{F}}_I^{L^{-1}})$. Subsequently, it was verified numerically in [18] that for highly heterogeneous model problems, these two specific choices of $\mathbf{Q}$ maintain indeed the numerical scalability of the FETI method with respect to both the mesh size h and subdomain size H.

A variational approach for tuning FETI to the solution of heterogeneous structural problems was also proposed in [32], then simplified in [19] to provide computational efficiency. This alternative approach does not focus on the projector $\mathbf{P}$, and therefore does not affect the choice of the matrix $\mathbf{Q}$. It focuses on the scaling matrix $\mathbf{W}^{(s)}$, and proposes a "superlumped stiffness" scaling procedure rather than the topological one described in Section 2. The variational theory exposed in [19] suggests that an efficient scheme for accelerating the convergence of the FETI method applied to the solution of highly heterogeneous problems is to construct the diagonal matrix $\mathbf{W}^{(s)}$ as follows. If $\boldsymbol{\lambda}(i)$ is the i-th component of the Lagrange multiplier vector viewed by subdomain $\Omega^{(s)}$ and connecting the interface displacement d.o.f. $\mathbf{u}^{(s)}(j_s)$ in $\Omega^{(s)}$ to the interface displacement d.o.f. $\mathbf{u}^{(q)}(j_q)$ in the neighboring subdomain $\Omega^{(q)}$, then $\mathbf{W}^{(s)}(i)$ is set to

$$\mathbf{W}^{(s)}(i) \;=\; \frac{k^{(q)}_{j_q j_q}}{k^{(s)}_{j_s j_s} + \displaystyle\sum_{\Omega^{(l)} \in \mathcal{N}(\Omega^{(s)})} k^{(l)}_{j_l j_l}} \tag{21}$$

where $\mathcal{N}(\Omega^{(s)})$ denotes the set of neighbors of $\Omega^{(s)}$, $k^{(l)}_{j_l j_l}$ is the diagonal coefficient of the subdomain stiffness matrix $\mathbf{K}^{(l)}$ associated with the displacement d.o.f. $\mathbf{u}^{(l)}(j_l)$, and j_l is such that the displacement d.o.f. $\mathbf{u}^{(l)}(j_l)$ and the displacement d.o.f. $\mathbf{u}^{(s)}(j_s)$ correspond to the same displacement d.o.f. of the global finite

element model. Note that if all the subdomains in $\mathcal{N}(\Omega^{(s)})$ have the same material and discretization properties as subdomain $\Omega^{(s)}$, then $\mathbf{W}^{(s)}(i) = 1/m$ (see Section 2), which shows that the topological scaling overviewed in Section 2 is a particular case of the superlumped stiffness scaling summarized in Eq. (21). In [19], using a set of model problems and a few realistic ones, it was shown that the FETI method equipped with the stiffness scaling matrix $\mathbf{W}^{(s)}$ specified in Eq. (21) is numerically scalable with respect to both the problem and mesh partition sizes.

The superlumped stiffness scaling (21) does not increase neither the computational complexity nor the storage requirements of the FETI method by any significant amount. Therefore, it can be invoked by default. On the other hand, equipping the FETI method with $\mathbf{Q}^D = \overline{\mathbf{F}}_I^{D^{-1}}$ or $\mathbf{Q}^L = \overline{\mathbf{F}}_I^{L^{-1}}$ increases the computational complexity and storage requirements of the projection steps in FETI by a small percentage. Hence, a first objective of this section is to investigate when and whether equipping the FETI method not only with the stiffness scaling procedure (21) but also with a matrix $\mathbf{Q} \neq \mathbf{I}$ is worthwhile.

Furthermore, since both the $\mathbf{Q}$- and $\mathbf{W}^{(s)}$-approach address in an explicit manner only the structural heterogeneities viewed by the subdomain interfaces, a second objective of this section is to investigate whether the subdomain interfaces should include all the mesh boundaries separating the different materials of a heterogeneous finite element model, which would affect the mesh partitioning strategy.

Finally, a third objective of this section is to devise a general strategy for optimizing the solution of highly heterogeneous structural problems by the FETI method.

4.1. Findings and recommendations

In our numerous experimentations with the solution of heterogeneous problems by the FETI method, we have observed the following

O1) in addition to $\mathbf{Q}^D$ and $\mathbf{Q}^L$, the following matrix should be considered

$$\mathbf{Q}^{SL} = \sum_{s=1}^{N_s} \mathbf{W}^{(s)} \mathbf{B}^{(s)} \begin{bmatrix} 0 & 0 \\ 0 & \mathrm{diag}(\mathbf{K}_{bb}^{(s)}) \end{bmatrix} \mathbf{B}^{(s)^T} \mathbf{W}^{(s)} \tag{22}$$

This matrix is a diagonal (lumped) approximation of the lumped preconditioner. Hence, we refer to it as the superlumped matrix $\mathbf{Q}^{SL}$, which explains the SL superscript. Equipping the projector $\mathbf{P}$ with the superlumped matrix $\mathbf{Q}^{SL}$ can be interpreted as preconditioning the coarse problem by the superlumped stiffness scaling procedure (21). This matrix is inexpensive to

compute and store, and is such that $\mathbf{G}_I^T \mathbf{Q}^{SL} \mathbf{G}_I$ has the same sparsity pattern — and therefore the same memory requirements — as $\mathbf{G}_I^T \mathbf{G}_I$.

O2) if *all* the subdomain interfaces separate regions with similar high jumps in the material properties, then the FETI method equipped with the superlumped stiffness scaling procedure (21) performs well, and converges even faster than when the problem is homogeneous. We have observed this behavior of FETI even for mesh partitions with poor subdomain aspect ratios.

O3) if *some* but not all of the subdomain interfaces separate regions with similar high jumps in the material properties, then FETI exhibits a good convergence when equipped with the superlumped stiffness scaling procedure (21) and $\mathbf{Q} \neq \mathbf{I}$, and when the mesh partition has good subdomain aspect ratios. By $\mathbf{Q} \neq \mathbf{I}$, we mean here $\mathbf{Q} = \mathbf{Q}^D$, or $\mathbf{Q} = \mathbf{Q}^L$, or $\mathbf{Q} = \mathbf{Q}^{SL}$.

Based on these observations, some of which are illustrated in the next section, we make the following recommendations for the solution of highly heterogeneous structural problems by the FETI method

R1) by default, use the superlumped stiffness scaling matrix (21), and equip FETI's projector $\mathbf{P}$ with $\mathbf{Q}^{SL}$ (10). This improves the convergence of FETI at almost zero additional storage and computational cost.

R2) if possible, design a mesh partition where all subdomain interfaces are along boundaries between materials with similar jumps in their properties. Unfortunately, this may be possible only for cyclic structures, or academic problems where the number of materials matches the desired number of subdomains.

R3) in the general case, partition the mesh along the material boundaries, then refine the obtained partition to generate the target number of subdomains N_s. If, because of topological reasons, this process can be expected to create subdomains with poor aspect ratios, modify this strategy as follows. First, re-organize the material groups into a smaller number of clusters each containing materials with relatively similar properties. Then, decompose the mesh along the boundaries of the clusters, and refine the obtained mesh partition to generate the desired number N_s of balanced subdomains. If needed for ensuring good subdomain aspect ratios, include in a cluster a neighboring material even if it has significantly different properties.

4.2. Justifications

In order to highlight the importance of the recommendations formulated above, we consider here the stress analysis of the heterogeneous cantilever structure shown in Fig. 7. This structure has a length $L_x = 4$, a depth $L_y = 1$, a

thickness $th = 0.01$, and a Poisson ratio $\nu = 0.3$. It is constructed by gluing together 8 slices of 3 different materials M_i in the following sequence: M_1, M_2, M_3, M_2, M_1, M_2, M_3, M_2 (Fig. 7). The Young moduli E_i of the materials M_i are such that

$$E_1 = 1000 \times E_2 \quad \text{and} \quad E_2 = 100 \times E_3 = 2.05 \times 10^{11} \tag{23}$$

Hence, $E^{max}/E^{min} = 10^5$.

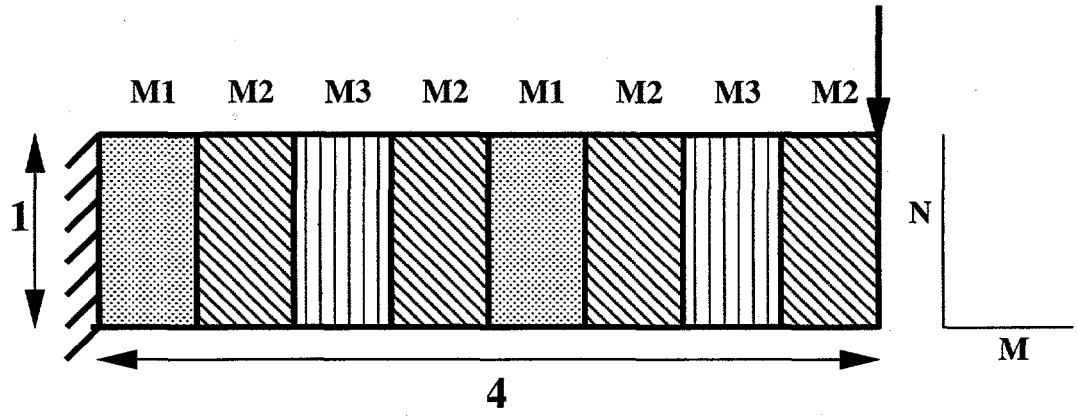

Fig. 7. A heterogeneous cantilever problem

We uniformly discretize this structure by 80×20 plane stress elements, and generate several $M \times N$ mesh partitions with different characteristics. For each mesh partition, we solve the corresponding system of equations by the FETI method equipped with the lumped and Dirichlet preconditioners, and with $\mathbf{Q} = \mathbf{I}$, $\mathbf{Q} = \mathbf{Q}^D$, $\mathbf{Q} = \mathbf{Q}^L$, and $\mathbf{Q} = \mathbf{Q}^{SL}$. In all cases, we use the superlumped stiffness scaling procedure (21). We adopt the stopping criterion (19) and report the obtained iteration counts in Table 5.

Table 5
Iteration count for FETI applied to the solution of the heterogeneous problem graphically depicted in Fig. 7

		Lumped preconditioner			Dirichlet preconditioner		
$N_s(M \times N)$	AR	$\mathbf{I}$	$\mathbf{Q}^{SL}$	$\mathbf{Q}^{L}$	$\mathbf{I}$	$\mathbf{Q}^{SL}$	$\mathbf{Q}^{D}$
4 (4×1)	1	18	17	17	5	5	4
8 (8×1)	1/2	23	23	23	7	7	6
16 (16×1)	1/4	43	42	41	19	17	22
16 (8×2)	1	34	21	19	22	15	15
40 (40×1)	1/10	113	112	112	82	81	81
40 (8×5)	1/2	68	37	35	53	25	27
64 (4×16)	1	66	20	19	52	14	17

From Fig. 7 and the results summarized in Table 5, the reader can check that

- for both 4×1 and 8×1 mesh partitions, all subdomain interfaces separate regions with high jumps of Young's modulus. In both cases, the FETI method performs well, particularly when equipped with the Dirichlet preconditioner, which is in agreement with the general observation O2.
- both 16×1 and 8×2 mesh partitions have the same number of subdomains. In both cases, only half the subdomain interfaces separate regions with high jumps of Young's modulus. However, the subdomains of the 8×2 mesh partition have a better aspect ratio (AR = 1) than the subdomains of the 16×1 decomposition (AR = 1/4). This explains why FETI performs better for the 8×2 mesh partition than for the 16×1 decomposition.
- similarly, both 40×1 and 8×5 mesh partitions have the same number of subdomains, same number of homogeneous interfaces, and same number of heterogeneous interfaces. However, the subdomains of the 8×5 decomposition have a better aspect ratio (AR = 1/2) than those of the 40×1 mesh partition (AR = 1/10). Consequently, FETI performs better for the 8×5 mesh partition than for the 40×1 decomposition.
- the 16×4 mesh partition is another example of a mesh partition with both homogeneous and heterogeneous subdomain interfaces. For this decomposition and $\mathbf{Q} \neq \mathbf{I}$, the FETI method performs as well as for the 8×2 mesh

partition, which is also characterized by a uniformly perfect subdomain aspect ratio.

- in all cases, the FETI method equipped with $\mathbf{Q}^{SL}$ performs almost as well as when equipped with $\mathbf{Q}^{D}$ or $\mathbf{Q}^{L}$, and in some cases it performs even better (because of variations in the initial residual). In a few cases the FETI method equipped with $\mathbf{Q} \neq \mathbf{I}$ performs much better than when equipped with $\mathbf{Q} = \mathbf{I}$. These two observations are consistent with the general observations O1 and O3.
- the Dirichlet preconditioner is needed when the mesh partition has poor subdomain aspect ratios.
- $\mathbf{Q} \neq \mathbf{I}$ is justified and needed when the mesh partition has heterogeneous crosspoints.
- when the recommendations formulated in the previous section are followed, FETI exhibits a reasonable numerical scalability with respect to the number of subdomains.

Next, we consider the case of a realistic ASCI-type heterogeneous structural problem, and illustrate in particular the importance of recommendation R3.

5. Application to the analysis of a mockup reentry vehicle

Here, we report on the performance of the FETI method applied to the finite element analysis of a mockup reentry vehicle (RV) on the ASCI Option Red supercomputer.

An RV can be expected to experience different loadings in normal and hostile environments. Its structural response during vibration is usually predicted by a modal analysis, while its shock response is usually simulated by a direct transient analysis. The predictive computation of responses at component levels requires a detailed finite element model of the full body as well as individual components.

We focus on a large-scale finite element model of a mockup RV with 330,300 elements, and 334,759 nodes. With slightly more than one million d.o.f., this model requires significant computational power, and provides a reasonable benchmark for massively parallel computational platforms. All elements of the mesh are either 8-noded brick or 6-noded wedge elements. Decomposing this mesh into subdomain with good aspect ratios is a difficult task because the RV shown in Fig. 8 has a thin wall tubular-like overall structure. Hence, the finite element model considered herein poses serious computational challenges to substructure-based methods.

There are eight different materials that are scattered within the RV model (Fig. 8), and their Young's moduli vary from 10^2 psi to 3×10^7 psi. Hence, this RV

structure is a highly heterogeneous one with $E^{max}/E^{min} = 3 \times 10^5$, and therefore can be expected to challenge any iterative solver. For the same reasons described in Section 3.3, we consider here only the stress analysis of this RV model using Salinas equipped with the FETI solver.

The results of the analysis performed in Section 3.3 (see Tables 1–4) suggest that the solution by FETI of this million d.o.f. problem requires (the memories of) at least 216 processors of the ASCI Option Red supercomputer. For this reason, we consider partitioning the given RV mesh into 250 subdomains and assigning each subdomain to one processor. We also consider partitioning this mesh into 500 subdomains for computations on 500 processors, in order to provide at least one example of the parallel scalability for a fixed problem size of our current massively parallel implementation of the FETI method. More specifically, in order to illustrate recommendation R3, we consider three different mesh decomposition strategies

- partitioning the mesh as is, with particular attention to the subdomain aspect ratio using the optimizers described in [15,16].
- partitioning the mesh along its material boundaries, then refining the obtained mesh partition to generate the requested number of subdomains. In that case, the subdomain aspect ratio optimizer [15,16] is applied locally, within each material group.
- re-organizing all the material groups of the RV finite element model into two clusters and partitioning each cluster independently from the other.

Furthermore, our mesh decomposer [29] automatically post-processes each mesh partition to remove any internal mechanism generated by the partitioning algorithm, in order to allow a robust evaluation of the rigid body modes and generalized inverse of the stiffness matrix of each floating subdomain [33]. For this reason, and because of other issues associated with the clustering process, the number of generated subdomains N_s may differ from the requested number of subdomains N_s^{req}, usually by less than 5 %.

Following recommendation R1, we equip FETI with the superlumped stiffness scaling procedure (21). However, as stated earlier, the topology of the RV model shown in Fig. 8 is such that mesh partitions with good subdomain aspect ratios cannot be reasonably expected. For this reason, we employ the Dirichlet preconditioner. Furthermore, we equip FETI's projector $\mathbf{P}$ with $\mathbf{Q} = \mathbf{Q}^D$, because the three other choices discussed in this paper turned out to be ineffective. We monitor the convergence of FETI with the stopping criterion (19), and report in Table 6 the obtained performance results.

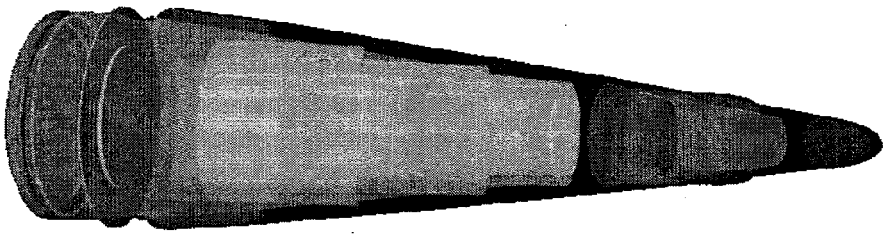

Fig. 8. Mockup reentry vehicle: each color indicates a different material

Table 6
Solution by FETI on the ASCI Option Red supercomputer of the RV problem with 1,004,277 d.o.f. (Fig. 8)

N_s^{req}	$N_p = N_s$	Type	N_{itr}	Factor $\mathbf{G}_I^T \mathbf{Q}^D \mathbf{G}_I$	FETI	Salinas + FETI
250	252	Regular	290	1.8 sec	378 sec	474 sec
250	251	Material	463	1.7 sec	563 sec	657 sec
250	257	Cluster	221	1.5 sec	261 sec	350 sec
500	513	Regular	325	8.8 sec	167 sec	219 sec
500	505	Material	584	10.6 sec	443 sec	502 sec
500	517	Cluster	276	11.6 sec	162 sec	218 sec

For both requested numbers of subdomains, the following trend can be observed

- the FETI method performs better on the regular mesh partition than on the material based mesh partition. This can be explained as follows. Many of the material interface boundaries run parallel to the longitudinal axis of the RV, within its thin wall structure. Consequently, each material group defines a substructure with a poor aspect ratio. Partitioning this substructure into tens of subdomains generates subdomains with poorer aspect ratios than partitioning the original mesh.

- on the other hand, the FETI method performs much better for the cluster based decomposition, which highlights the relevance of recommendation R3. Again, we remind the reader that this notion of clustering is motivated here by topological reasons and the objective of generating subdomains with as good an aspect ratio as possible.

For the cluster based mesh decompositions, the performance results reported in Table 6 demonstrate a reasonable numerical scalability of the FETI method for this highly heterogeneous problem. They also show that the CPU time of the FETI method is reduced by a factor equal to 1.6 when the number of processors is increased from 257 to 517. This corresponds to an efficiency of 80%, which demonstrates a good parallel scalability of our implementation of FETI on the ASCI Option Red supercomputer.

Finally, from Table 1 and Table 6, the reader can observe that the performance of the FETI method for this heterogeneous RV problem is consistent with that of the BP1 problem with 216 subdomains.

6. Closure

We have presented an initial implementation of the FETI method on the ASCI Option Red supercomputer, reported on its incorporation within the Salinas structural dynamics code, and its application to the solution of highly heterogeneous problems. This initial implementation of FETI on a massively parallel distributed memory system is characterized by (a) the redundant storage of the sparse matrix $\mathbf{G}_I^T \mathbf{Q} \mathbf{G}_I$ of the FETI coarse problem in every processor, (b) the serialization of the factorization of this matrix, but (c) the perfect and efficient parallelization of the subsequent forward and backward solves associated with this matrix via an inverse matrix approach. For up to 1000 subdomains and 1000 processors with 128 Mbytes each, this implementation delivers a good scalability. For a larger number of subdomain and processors, it must be revisited to distribute the storage of the sparse matrix of the coarse problem among a group if not all of the processors, and perform in parallel the factorization of this matrix. For heterogeneous structural problems, we recommend equipping FETI by default with the superlumped stiffness scaling procedure fully described in [19], and the superlumped version (22) of the heterogeneous projector presented in [9]. However, some problems may require equipping FETI's projector with the Dirichlet $\mathbf{Q}^D$ matrix. We also recommend generating mesh partitions whose interface boundaries include the mesh material boundaries. If doing so prevents the generation of subdomains with good aspect ratios, we recommend clustering the materials into groups with different but relatively similar properties before partitioning the mesh. When these recommendations are followed, the FETI method

achieves a good combined numerical/parallel scalability for highly heterogeneous structural problems.

Acknowledgements

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the U.S. Department of Energy under Contract No. DE-AC04-94AL85000. Charbel Farhat acknowledges partial support by the Sandia National Laboratories under Contract No. BD-2435. Michel Lesoinne and Daniel Rixen acknowledge partial support by the Department of Energy under Award No. B347880/W-740-ENG-48. Daniel Rixen acknowledges also support by the Fonds National de la Recherche Scientifique, Belgium. Kendall Pierson acknowledges the support by the National Science Foundation under Grant No. GE49355046. All authors thank their colleagues Ken Alvin, David Martinez, John RedHorse, and Garth Reese at the Sandia National Laboratories for their various contributions to this effort.

References

[1] S. F. McCormick, Multilevel adaptive methods for partial differential equations, Frontiers in Applied Mathematics, SIAM, 1989.

[2] S. F. McCormick, ed., Multigrid methods, Frontiers in Applied Mathematics, SIAM, 1987.

[3] W. Briggs, A multigrid tutorial, SIAM, 1987.

[4] P. Vanek, J. Mandel and M. Brezina, Algebraic multigrid on unstructured meshes, Computing 56, 179-196 (1996).

[5] P. LeTallec, Domain-decomposition methods in computational mechanics, Computational Mechanics Advances 1, 121-220 (1994).

[6] C. Farhat, A Lagrange multiplier based divide and conquer finite element algorithm, J. Comput. Sys. Engrg. 2, 149-156 (1991).

[7] 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, 1205-1227 (1991).

[8] C. Farhat, J. Mandel and F. X. Roux, Optimal convergence properties of the FETI domain decomposition method, Comput. Meths. Appl. Mech. Engrg. 115, 367-388 (1994).

[9] C. Farhat and F. X. Roux, Implicit parallel processing in structural mechanics, Computational Mechanics Advances 2, 1-124 (1994).

[10] C. Farhat, P. S. Chen and J. Mandel, A scalable Lagrange multiplier based domain decomposition method for implicit time-dependent problems, Internat. J. Numer. Meths. Engrg. 38, 3831-3858 (1995).

[11] C. Farhat and J. Mandel, The two-level FETI method for static and dynamic plate problems - part I: an optimal iterative solver for biharmonic systems, Comput. Meths. Appl. Mech. Engrg. 155, 129-152 (1998).

[12] C. Farhat, P. S. Chen, J. Mandel and F. X. Roux, The two-level FETI method - Part II: extension to shell problems, parallel implementation and performance results, Comput. Meths. Appl. Mech. Engrg. 155, 153-180 (1998).

[13] C. Farhat, P. S. Chen and P. Stern, Towards the ultimate iterative substructuring method: combined numerical and parallel scalability, and multiple load cases, J. Comput. Sys. Engrg. 5, 337-350 (1995).

[14] M. Lesoinne and K. Pierson, An efficient FETI implementation on distributed shared memory machines with independent numbers of subdomains and processors, Contemporary Mathematics 218, 318-324 (1998).

[15] C. Farhat, N. Maman and G. Brown, Mesh partitioning for implicit computations via iterative domain decomposition: impact and optimization of the subdomain aspect ratio, Internat. J. Numer. Meths. Engrg. 38, 989-1000 (1995).

[16] D. Vanderstraeten, C. Farhat, P. S. Chen, R. Keunings and O. Zone, A retrofit and contraction based methodology for the fast generation and optimization of mesh partitions: beyond the minimum interface size criterion, Comput. Meths. Appl. Mech. Engrg. 133, 25-45 (1996).

[17] R. Diekmann, B. Meyer and B. Monien, Parallel decomposition of unstructured FEM-Meshes, Concurrency: Practice and Experience 10, 53-72 (1998).

[18] D. Rixen, Substructuring and dual methods in structural analysis, Ph.D. Thesis, Université de Liège, Belgium, Collection des Publications de la Faculté des Sciences appliquées, n° 175, 1997.

[19] D. Rixen and C. Farhat, A simple and efficient extension of a class of substructure based preconditioners to heterogeneous structural mechanics problems, Internat. J. Numer. Meths. Engrg. 44, 489-516 (1999).

[20] C. Farhat and P. S. Chen, Tailoring domain decomposition methods for efficient parallel coarse grid Solution and for systems with many right hand sides, Contemporary Mathematics 180, 401-406 (1994).

[21] C. Farhat, L. Crivelli and F. X. Roux, Extending substructure based iterative solvers to multiple load and repeated analyses, Comput. Meths. Appl. Mech. Engrg. 117, 195-200 (1994).

[22] J. Mandel and R. Tezaur, On the convergence of a substructuring method with Lagrange multipliers, Numerische Mathematik 73, 473-487 (1996).

[23] R. Glowinski, G. H. Golub, G. A. Meurant, and Jacques Périaux, eds., First International Symposium on Domain Decomposition Methods for Partial Differential Equations, Philadelphia, PA, USA, SIAM, 1988.

[24] ASCI Blue Mountain web site, http://www.lanl.gov/projects/asci/bluemtn.

[25] S. L. Lillevik, The Touchstone 30 Gigaflop DELTA prototype, in Proceedings of the Sixth Distributed Memory Computer Conference, p. 671, IEEE Computer Society Press, 1991.

[26] B. Traversat, B. Nitzberg and S. Fineberg, Experience with SUNMOS on the Paragon XP/S-15, in Proceedings of the Intel Supercomputer Users' Meeting, San Diego, 1994.

[27] L. Shuler, R. Riesen, C. Jong, D. van Dresser, A. Maccabe, L. A. Fisk, and T. M. Stallcup, The Puma operating system for massively parallel computers, in Proceedings of the Intel Supercomputer North America Users' Conference, 1995.

[28] S. Garg, R. Godley, R. Griffths, A. Pfiffer, T. Prickett, D. Robboy, S. Smith, T. M. Stallcup, and S. Zeisset, Achieving large scale parallelism through operating system resource management on the Intel TFLOPS supercomputer, Intel Technology Journal, Q1'98 issue, 1998.

[29] C. Farhat, S. Lanteri and H. D. Simon, TOP/DOMDEC, A software tool for mesh partitioning and parallel processing, J. Comput. Sys. Engrg. 6, 13-26 (1995).

[30] B. Hendrickson and R. Leland, The Chaco user's guide: version 2.0, Tech. Report SAND94–2692, 1994.

[31] D. Rixen, C. Farhat, R. Tezaur and J. Mandel, Theoretical comparison of the FETI and algebraically partitioned FETI methods, and performance comparisons with a direct sparse solver, Internat, J. Numer. Methds. Engrg., (in press).

[32] C. Farhat and D. Rixen, A new coarsening operator for the optimal preconditioning of the dual and primal domain decomposition methods: application to problems with severe coefficient jumps, in Proceedings of the Seventh Copper Mountain Conference on Multigrid Methods, N. Duane Melson, T. A. Manteuffel, S. F. McCormick and C. C. Douglas, eds., pp. 301-316 (1995)

[33] C. Farhat, K. Pierson and M. Lesoinne, The second generation of FETI methods and their application to the parallel solution of large-scale linear and geometrically nonlinear structural analysis problems," Comput. Meths. Appl. Mech. Engrg., (in press).

Appendix B
SAND 98-0410C: A Basic Parallel Sparse Eigensolver for Structural Dynamics

David Day [1]

Abstract

In this work the basic Finite Element Tearing and Interconnecting (FETI) linear system solver and the PARPACK eigensolver are combined to compute the smallest modes of symmetric generalized eigenvalue problems that arise from structures modeled 'primarily' by solid finite elements. Problems with up to one million unknowns are solved. A comprehensive and relatively self-contained description of the FETI method is presented.

B.1 Introduction and Summary

We seek to compute the left-most modes of a symmetric generalized eigenvalue problem $Ax = Mx\lambda$ with more than one million unknowns arising from structural dynamics using a distributed memory platform. In this work both A and M are positive definite. This generalized eigenvalue problem is solved using PARPACK [B.19, B.21] in shift-invert mode. The traditional approach is to use a sparse direct linear solver for inversion [B.15]. The limitation of this approach is that all current parallel direct sparse matrix factorization codes require essentially infinite per processor memory to be scalable. Instead we use an iterative linear solver. The Finite Element Tearing and Interconnecting (FETI hereafter) iterative linear system solver [B.13] is used to invert [B.6]. This work describes the basic FETI multilevel used method for solving linear systems whose coefficient matrix is a positive definite stiffness matrix for solid structures. The underlying partial differential equation for solid structures is second order elliptic.

Section two describes the reformulation of the linear system as a distributed interface linear system. Section three develops the FETI iterative solution algorithm. In section four the combined FETI/PARPACK method is applied to several model eigenvalue problems of order up to one million.

Notation. In this work upper case Roman letters denote matrices, lower case

[1]Sandia National Laboratories, Albuquerque, New Mexico 87185. Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy under Contract DE-AC04-94AL85000.

Roman letters denote vectors and Greek letters denote scalars. $\mathcal{R}(A)$ and $\mathcal{N}(A)$ denote the range and null spaces of the specified matrix.

B.2 Formulation of the Interface Linear System

FETI is a domain decomposition method in which the compatibility equation for the subdomain interfaces is solved iteratively. A subdomain is a set of elements (not nodes). This approach is natural for finite element modeling because assembling the subdomain stiffness matrix requires no interprocessor communication.

Substructuring is a type of domain decomposition in which an unassembled structural stiffness matrix is represented in the form $A = L^T K L$ where K is a block diagonal matrix of subdomain stiffness matrices and L is a Boolean assembly matrix [B.22]. Consider for example if $-\ddot{u}(x) = g$ with Dirichlet boundary conditions on the unit interval is discretized by finite differences on a regular grid with mesh spacing 1/4. Paritioning into two subdomains, each with a single element, yields the decomposition

$$\begin{bmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 2 & -1 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & -1 & 2 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Note that in this case $L^T L = \mathrm{diag}(1, 2, 1)$. In general the linear system $Au = f$ is traditionally solved by multi-frontal Gaussian elimination [B.17, pp. 216–225] FETI solvers exploit the parallelism inherent in the representation $A = L^T K L$.

A subdomain boundary node is represented on each subdomain sharing the node. The global solution is a particular set of subdomain solutions that have consistent values along subdomain interfaces. This interface compatibility condition is a discrete analog of the Neumann boundary condition $\nabla u \cdot n = 0$. For this reason most subdomain stiffness matrices are singular. To illustrate this point, return to the example problem $-\ddot{u}(x) = g$ on the unit interval discretized with regular mesh spacing $1/(n+1)$. For $n > 3$, paritioning into one element subdomains yields

$$K = \mathrm{diag}\left(\begin{bmatrix} 2 & -1 \\ -1 & 1 \end{bmatrix}, \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}, \ldots, \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}, \begin{bmatrix} 1 & -1 \\ -1 & 2 \end{bmatrix} \right)$$

A more precise definition of L will be required. L maps the unknowns to subdomain unknowns. Each row of L corresponds to a subdomain unknown and each column of L corresponds to a structure unknown. In each row of L there is precisely one nonzero entry. $L_{i,j} = 1$ if subdomain node i corresponds to node j. Moreover $\Delta := L^T L$ is a diagonal matrix whose (j, j) entry is the number of subdomains sharing the jth global unknown.

The FETI approach is based on the decomposition of $Au = f$ into the three linear systems

$$\begin{array}{ll} L^T p = f & p \text{ not unique} \\ K u_s = p & u_s \text{ not unique} \\ & u_s \text{ exists} \iff p \in \mathcal{R}(K) \quad (p \perp \mathcal{N}(K)) \\ L u = u_s & u \text{ exists} \iff u_s \in \mathcal{R}(L) \quad (u_s \perp \mathcal{N}(L^T)) \end{array}$$

Here the fact that the range of a matrix is orthogonal to the null space of the transpose is used. The constraints that u_s and u exist determine p and u_s.

Because K is singular, it is necessary to compute the action of the pseudo-inverse, K^+, on a vector. We digress to discuss this task in detail. A stable algorithm for computing K^+ involves the eigendecomposition of K which is too costly. We are stuck using an unstable algorithm. A much more efficient but potentially unstable method is to use the factorization $K = R^T D R$, where R is unit upper right triangular and D is diagonal, to determine the null space $\mathcal{N}(K)$ [B.25]. In general (complete) pivoting tends to reduce instability of a factorization [B.16]. Currently complete pivoting is not used. Instead using the properties of solid elements, it is possible to permute the subdomain unknowns so that on connected subdomains the pivots are all positive until only a small South East submatrix remains unfactored. Instead of factoring, an eigendecomposition of this small symmetric South East submatrix is computed. This technique is potentially unstable due to a kind of element growth.

Particular solutions to the first two equations are $\tilde{p} := L\Delta^{-1} f$ (simpler to verify than derive) and

$$\tilde{u}_s := K^+ p. \tag{B.1}$$

The general solutions can be written in the following form. First p is the sum of a particular solution and a vector in the null space of L^T. Let the rows of B span the null space of L^T, so that $BL = 0$. Because p is determined by $L^T p = f$ to within the null space of L^T, there exits a vector x such that

$$p = \tilde{p} - B^T x, \tag{B.2}$$

Similarly u_s is determined by $K u_s = p$ to within the null space of K. Let the columns of N_r be an orthonormal basis for the null space of K. Then u_s must be of the form

$$u_s = \tilde{u}_s - N_r x_r. \tag{B.3}$$

Because u_s satisfies the interface compatibility condition, there holds $u_s = Lu$. Multiply by $\Delta^{-1} L^T$ to find

$$u = \Delta^{-1} L^T u_s \tag{B.4}$$

Engineers use the symbol λ in place of x to remind the reader that this is a vector of Lagrange multipliers. We reserve the symbol λ for eigenvalues.

If a subdomain stiffness matrix is poorly conditioned, it may be nontrivial to distinguish the null space from the eigenspaces that correspond to the smallest positive eigenvalues. In this case the columns of N_r are ortho-normal and span a space that contains $\mathcal{N}(K)$ and the span of the eigenvectors corresponding to the smallest eigenvalues. The indices of x_r corresponding to eigenvectors of positive eigenvalues are not required to vanish.

We conclude this section by deriving the linear system that defines the interface unknowns. There exist unique vectors x and x_r such that u_s and u exist. The vector u_s is contained in $\mathcal{R}(L)$ if $Bu_s = \mathbf{0}$. First substitute equations (B.3), (B.1) and (B.2) to find

$$\begin{aligned} \mathbf{0} = Bu_s &= B\tilde{u}_s - BN_r x_r \\ &= BK^+ p - BN_r x_r \\ &= BK^+(\tilde{p} - B^T x) - N^T N_r x_r \\ &= BK^+\tilde{p} - BK^+ B^T x - BN_r x_r \end{aligned}$$

Second because $\mathcal{N}(K)$ is the orthogonal complement of $\mathcal{R}(K)$,

$$\mathbf{0} = N_r^T p = N_r^T \tilde{p} - N_r^T B^T x$$

implies that $p \in \mathcal{R}(K)$. Rewriting these two simultaneous equations for x and x_r in matrix notation yields the reduced interface linear system

$$\begin{bmatrix} BK^+B^T & BN_r \\ N_r^T B^T & \mathbf{0} \end{bmatrix} \begin{bmatrix} x \\ x_r \end{bmatrix} = \begin{bmatrix} BK^+\tilde{p} \\ N_r^T \tilde{p} \end{bmatrix} \tag{B.5}$$

Section three describes the FETI iterative solution method for this equation. Given the solutions x and x_r, first compute p, then $\tilde{u}_s$, next u_s and finally the solution u.

B.3 The FETI Method

In this section the basic FETI method for solving the reduced interface linear system is developed. The indefinite interface linear system is first reduced to a symmetric positive semi-definite linear system. Dirichlet preconditioning is used. The resulting iterative method is an instance of a preconditioned conjugate projected gradient method [B.13].

Next we discuss the particular choice of B used in this work. B is defined by two properties. First the rows of B are of the form $[0 \ldots 010 \ldots 0 - 10 \ldots 0]$ where the location of the two nonzero entries corresponds to a pair of unknowns that correspond to

the same global unknown. Second the subdomain constraint matrix is constructed in the same way for each subdomain. These choices are made to ease implementation. The disadvantage of constructing B in this way is that redundant constraints are included in B. If one structure unknown corresponds to three subdomain unknowns, then two constraints would suffice to enforce the interface compatibility condition, but two constraints are added to B for each of the three subdomains, resulting in a total of three constraints. If one structure unknown corresponds to four subdomain unknowns, then then three constraints are added to B for each of the four subdomains, resulting in a total of six constraints. In general if a subdomain shares a node with k other processors, then k constraints would suffice, but a total of $k(k-1)/2$ constraints are used. This choice of B is inefficient due the the excessive number of redundant constraints if a node is shared by very many subdomains (think of the hub of a spoked wheel). We will revisit this subject in more detail in §3.2.

B.3.1 The Projected Linear System

In this section we discuss the solution method for the linear system (B.5). We first introduce the notation

$$F = BK^+B^T \quad G = BN_r \quad d = BK^+\tilde{p} \quad e = N_r^T\tilde{p}$$

and rewrite (B.5) in the more compact form

$$\begin{bmatrix} F & G \\ G^T & \mathbf{0} \end{bmatrix} \begin{bmatrix} x \\ x_r \end{bmatrix} = \begin{bmatrix} d \\ e \end{bmatrix} \tag{B.6}$$

Iterative solvers such as MINRES applied to the the symmetric indefinite linear system (B.6) converge somewhat less rapidly than conjugate gradient iteration for symmetric positive definite matrices [B.29]. The above system is solved by deflating the G term and applying PCG to the constrained linear system. The resulting algorithm has been called the preconditioned conjugate projected gradient method (PCPG) [B.13].

The eponymous projector in PCPG annihilates G,

$$\Pi = I - G(G^TG)^{-1}G^T.$$

Multiply row 1 of equation (B.6) by Π to eliminate x_r from

$$\Pi Fx + \Pi Gx_r = \Pi Fx = \Pi d.$$

To compute x approximate the correction c, $x = c + x_0$, to $x_0 = G(G^TG)^{-1}e$. Now $G^Tx_0 = e = G^Tx$ implies that $G^Tc = \mathbf{0}$. Thus $c = \Pi c$ solves

$$(\Pi F\Pi)c = \Pi(d - Fx_0)$$

Note that once x has been computed, x_r is given by $x_r = (G^TG)^{-1}G^T(d - Fx)$.

The basic FETI method works well for elasticity problems, for stuctures modeled by elements (hex and wedge elements) that correspond to a second order partial differential equation. But, for structures modeled primarily by elements that correspond to a fourth order partial differential equation such as plate and shell elements, modifications to the coarse space are worthwhile [B.11] [B.3]. We give a new possible explanation for this. It is possible to view FETI as a multilevel method with coarse space $\mathcal{N}(K)$ and coarse space preconditioner $(G^TG)^{-1}$. Recall that $G = BR$ where the rows of B, $[..1\ldots -1..]$, act like a gradient operator. And G^TG contains a product of gradients, like a *Laplacian*. Thus $(G^TG)^{-1}$ acts like an inverse Laplacian preconditioner, suitable for second order elliptic partial differential equations.

B.3.2 Dirichlet Preconditioning

A Dirichlet preconditioner $\tilde{F}$ is an approximate inverse of $F = BK^+B^T$. To define $\tilde{F}$, more notation will be needed. Let p denote the number of subdomains and let $K^{(i)}$ and B_i denote the subdomain stiffness and constraint matrices:

$$K = \text{diag}(K^{(1)}, \ldots, K^{(p)}) \qquad B = [B_1, \ldots, B_p]$$

Then $F = \Sigma_i B_i K^{(i)+} B_i^T$. Next a kind of inverse of $B_i K^{(i)+} B_i^T$ is defined. Each row of B represents a constraint and is of the form $[\ldots 1 \ldots -1 \ldots]$. But the columns of B that correspond to interior nodes vanish. We define $\hat{B}_i$ to be the matrix obtained by deleting the zero columns from B_i. If a global unknown corresponds to an unknown on only one subdomain, then the corresponding subdomain unknown is called an interior unknown. But if a global unknown corresponds to unknowns on more than one subdomain, then the corresponding subdomain unknowns are called boundary unknowns. Thus $B_i K^{(i)+} B_i^T = \hat{B}_i X_i \hat{B}_i^T$ where X_i consists of the elements of $K^{(i)+}$ that correspond to boundary nodes.

Next we show that X_i is related to a Schur complement in $K^{(i)}$. Since we will be discussing one subdomain only in this paragraph, the superscript i is dropped. K can be permuted so that interior unknowns precede all boundary unknowns. In this case,

$$K = \begin{bmatrix} K_{ii} & K_{ib} \\ K_{bi} & K_{bb} \end{bmatrix} = \begin{bmatrix} I & 0 \\ K_{bi}K_{ii}^{-1} & I \end{bmatrix} \begin{bmatrix} K_{ii} & K_{ib} \\ 0 & S \end{bmatrix}$$

where $S = K_{bb} - K_{bi}K_{ii}^{-1}K_{ib}$. If S were nonsingular then

$$K^{-1} = \begin{bmatrix} K_{ii}^{-1} & -K_{ii}^{-1}K_{ib}S^{-1} \\ 0 & S^{-1} \end{bmatrix} \begin{bmatrix} I & 0 \\ -K_{bi}K_{ii}^{-1} & I \end{bmatrix} = \begin{bmatrix} * & * \\ * & S^{-1} \end{bmatrix}$$

and $X = S^{-1}$.

Dirichlet preconditioning approximates the inverse of the sums,

$$F^+ = (\Sigma_i \hat{B}_i S_i^+ \hat{B}_i^T)^+$$

by the sum of the inverses $\tilde{F} = \Sigma_i \hat{B}_i S_i \hat{B}_i^T$.

The effectiveness of the Dirichlet preconditioner is enhanced by including a diagonal scaling matrix W^{-1} and using $W^{-1}\tilde{F}W^{-1}$ as an approximate inverse of F [B.14]. We present a derivation that the scaling makes the preconditioner an exact inverse in a special case. The derivation is condensed and relatively difficult to follow. The scaling can be derived in the special case in which each $K^{(i)} = I$. First observe that $\tilde{F} = F = BB^T$. The remainder of the derivation depends on a special property of B that we now digress to derive. Constraining two unknowns to be equal is achieved using $C_2 = [1 \quad -1]$. Constraining three unknowns to be equal is achieved using

$$C_3 = \begin{bmatrix} 1 & -1 & 0 \\ 1 & 0 & -1 \\ 0 & 1 & -1 \end{bmatrix}$$

Constraining $n+1$ unknowns to be equal is achieved using

$$C_{n+1} = \begin{bmatrix} e & -I \\ 0 & C_n \end{bmatrix}$$

where e is a vector of ones of appropriate length. A remarkable property of this choice of redundant constraints is that $C_k C_k^T$ has only one nonzero eigenvalue, k, with multiplicity $k-1$. The null space of $C_k C_k^T$ has dimension $(k-1)(k-2)/2$. Thus the matrix BB^T can be permuted into a direct sum of matrices of the form $C_k C_k^T$ and the zero matrix. Recall that $C_k k^{-1} C_k^T$ is a projection. We define $W = \text{diag}(W_1, ..., W_p)$ where the (k,k) entry of W_i is the number of unknowns in any subdomain that correspond to the same (global) unknown as the k-th boundary unknown on the i-th subdomain. Then (in the special case $K^{(i)+} = I$) $W^{-1/2} F W^{-1/2}$ is a projection and $F^+ = W^{-1}\tilde{F}W^{-1}$. This completes the derivation of the scaling.

B.3.3 Scalings for Highly Heterogeneous Materials

A structural model consists of highly heterogeneous matrerials if the material stiffnesses (Young's moduli) vary over many orders of magnitude. For these applications a generalized scaling of the Dirichlet preconditioner is used. If a global unknown corresponds to unknowns i_k on subdomains p_k, then the entry of W that corresponds to i_j is

$$(\Sigma_k K^{(p_k)}_{i_k,i_k}) / K^{(p_j)}_{i_j,i_j}$$

instead of $\Sigma_k 1$ [B.12, B.20]. This scaling is called super lumped scaling. In section 4.2, numerical evidence is given that, even with this scaling, FETI is not robust for structural models that consist of highly heterogeneous matrerials. In this section, we present an additional diagonal scaling, called K scaling, for highly heterogeneous materials designed to extend the range of applicability of the FETI solver. K-scaling is used in concert with the super lumped scaling.

In general DAD scaling of a symmetric positive definite matrix A to a unit diagonal matrix is known to be nearly optimal, in the sense that the condition number of DAD is nearly minimized. In FETI the dual problem with coefficient matrix F is solved iteratively. K-scaling nearly minimizes the condition number of F. The ideal matrices to scale are F or K^+, but neither is assembled. The most closely related matrix that is assembled is

$$\begin{bmatrix} K & B^T \\ B & 0 \end{bmatrix}$$

We will scale this matrix on the left and right to reduce the condition number of F,

$$\begin{bmatrix} D_1 & 0 \\ 0 & D_2 \end{bmatrix} \begin{bmatrix} K & B^T \\ B & 0 \end{bmatrix} \begin{bmatrix} D_1 & 0 \\ 0 & D_2 \end{bmatrix}$$

by choosing the scaling for which the diagonal elements of K are as near to unity as possible. At unknowns that correspond to interior subdomain nodes (not shared), the diagonal entries of D_1 are chosen so that $D_1 K D_1$ is unit diagonal. And diagonal entries of D_1 that correspond to the same global unknown are chosen to be equal. In this case, due to the special structure of the signed Boolean constraint matrix B, there exists a diagonal matrix Δ such that

$$BD_1 = \Delta B$$

The constraint matrix B is invariant under scalings such that $D_2 = \Delta^{-1}$;

$$\begin{bmatrix} D_1 & 0 \\ 0 & D_2 \end{bmatrix} \begin{bmatrix} K & B^T \\ B & 0 \end{bmatrix} \begin{bmatrix} D_1 & 0 \\ 0 & D_2 \end{bmatrix} = \begin{bmatrix} D_1 K D_1 & B^T \\ B & 0 \end{bmatrix}$$

This vastly simplifies the implementation of the K scaling.

The diagonal entries of D_1 that correspond to shared unknowns are chosen so that $D_1 K D_1$ is nearly unit diagonal on a logarithmic scale. As in the discussion of super lumped scaling, suppose that a global unknown corresponds to local unknowns $\{i_1, i_2, \ldots\}$ on the corresponding subdomains $\{p_1, p_2, \ldots\}$. The corresponding entries of D_1 each have the value δ that minimizes

$$\max_j \mid \log \ \delta^2 K^{(p_j)}_{i_j,i_j} \mid$$

The mini-max solution is

$$\delta = \left(\frac{\min_j K^{(p_j)}_{i_j,i_j}}{\max_j K^{(p_j)}_{i_j,i_j}} \right)^{1/4}$$

Numerical experiments using this scaling will be presented in a future report.

B.3.4 Algorithm Implementation

In summary, the basic FETI method is implemented as follows.

Initialize: $x_0 = G(G^T G)^{-1} e$ and $r_0 = d - Fx$

For $k = 1, 2, \ldots$

1. $w_{k-1} = \Pi r_{k-1}$
2. $z_{k-1} = W^{-1}\tilde{F}W^{-1}w_{k-1}$
3. $y_{k-1} = \Pi z_{k-1}$
4. $\zeta_{k-1} = y_{k-1}^T z_{k-1} / y_{k-2}^T z_{k-2} \quad (\zeta_1 = 0)$
5. $p_k = y_{k-1} + p_{k-1}\zeta_{k-1} \quad (p_1 = y_0)$
6. $p_k := (I - P\Omega^{-1}Q^T)p_k \quad (k > 0)$
7. $P(:,k) = p_k$
8. $Q(:,k) = Fp_k$
9. $\Omega(k,k) = P(:,k)^T Q(:,k)$
10. $\nu_k = y_{k-1}^T z_{k-1} / \omega_k$
11. $x_k = x_{k-1} + p_k \nu_k$
12. $r_k = r_{k-1} - Fp_k\nu_k$

Remark. In steps 6, 7 and 8 the search directions are stored and the current search direction is explicitly re-orthogonalized against all the previous search directions. This extra work is observed to be worthwhile [B.23].

Remark. Iteration stops once $\|w_k\| < TOL\|w_0\|$. The reduction in $\|w\|$ is observed to be proportional to the reduction in $\|Au - f\|$.

B.4 Eigenvalue Problems

In this section the FETI method is applied to a few model generalized eigenvalue problems. We first discuss the extension of the algorithm to eigenvalue problems and we conclude with a summary of proposed future work.

B.4.1 The Influence of Limited Accuracy Linear Solves

In this theoretical section the influence of linear solver accuracy on the eigenpair residuals $Ax - Mx\lambda$ is analyzed. Eigenpairs of (A, M) are approximated using the shift-invert Lanczos algorithm. The result of this section is to show that if a direct linear solver is used and the relative eigenvalue tolerance, τ, (for the shifted problem) is not too small, then the absolute residual error is proportional to τ. And the same result holds if an iterative linear solver is used, provided that the tolerance for the linear solver is somewhat smaller than τ. The use of a finite tolerance does not influence the orthogonality of the

Lanczos vectors. This simple analysis does not in any way account for implicit restarts; the initial deflation scheme in ARPACK is sensitive to τ [B.18], but deflation schemes are in progress that are not sensitive to τ [B.28].

The expression $fl(A^{-1})$ is introduced to denote the computed result of (implicitly) computing the action of A^{-1}. For example if a direct solver is used, then the computed Cholesky decomposition of A satisfies $A = U^T U + E$ for $\|E\| = \mathcal{O}(\epsilon\|A\|)$ and $fl(A^{-1}) = (U^T U)^{-1}$.

The governing equation for the shift-invert Lanczos algorithm (zero shift) is

$$fl(A^{-1})MQ_j = Q_j T_j + q_{j+1}\beta_{j+1}e_j^T + E_j, \quad \|E_j\| = \mathcal{O}(\epsilon\|Q_j\|\|T_j\|).$$

For clarity we set $E_j = \mathbf{0}$. Let τ denote the relative residual tolerance

$$fl(A^{-1})MQ_j s - Q_j s\lambda^{-1} = -r, \quad \|r\| \leq \tau\lambda^{-1} \tag{B.7}$$

Here $T_j s = s\lambda^{-1}$ and $\|Qs\|_M = 1$. But the significant quantity is the norm of

$$AQ_j s - MQ_j s\lambda.$$

Multiply (B.7) by $-A\lambda$ to find

$$-Afl(A^{-1})MQ_j s\lambda + AQ_j s = Ar\lambda$$

Adding $(Afl(A^{-1}) - I)MQ_j s\lambda$ to both sides yields

$$AQ_j s - MQ_j s\lambda = Ar\lambda + (Afl(A^{-1}) - I)MQ_j s\lambda \tag{B.8}$$

In the direct solver case, for $\xi = \|Afl(A^{-1}) - I\| = \|E(U^T U)^{-1}\|$ and $\|E\| = \mathcal{O}(\epsilon\|A\|)$, one can derive that $\xi = \mathcal{O}(\epsilon\|A\|\|A^{-1}\|)$. Take norms, substitute $\|r\| \leq \tau\lambda^{-1}$ and there appears

$$\|AQ_j s - MQ_j s\lambda\| \leq \tau\,\|A\| \;+\; \xi\,\|MQ_j\|\,\|s\lambda\|$$

That is the residual error depends on the eigensolver tolerance τ until τ decreases to the threshold $\xi\|MQ_j\|\|s\lambda\|$.

The error in computing $\tilde{X} = fl(A^{-1})(MQ_j)$ is different if an iterative linear solver is used. Assuming that the linear solver converges, the columns of $\tilde{X} = [\tilde{x}_1 \cdots \tilde{x}_j]$ satisfy the inequality

$$\|A\tilde{x}_i - Mq_i\| \leq \rho\|Mq_i\|.$$

Use of this strict stopping criterion in the iterative linear solver is based on the assumption that all backward error is in the right hand side; this criterion is unachievable for sufficiently small ρ [B.1]. The following analysis captures the essential relations. First it

is not too much of a simplification to say that an iterative linear solver computes $\tilde{X}$ such that

$$\|A\tilde{X} - MQ_j\| \leq \rho\|MQ_j\| \tag{B.9}$$

for a user supplied scalar ρ. Substitute

$$(Afl(A^{-1}) - I)MQ_j s\lambda = (A\tilde{X} - MQ_j)s\lambda$$

into (B.8), take norms and substitute (B.9) to find

$$\|AQ_j s - MQ_j s\lambda\| \leq \tau \|A\| + \rho \|MQ_j\| \|s\lambda\| \tag{B.10}$$

The residual error depends on both the eigensolver tolerance τ and the linear solver tolerance ρ. Ideally the error due to limited accuracy linear solves is smaller than the error from the eigensolver, $\rho\|MQ_j\|\|s\lambda\| < \tau\|A\|$, and is still achievable by the iterative linear solver.

B.4.2 Numerical Results

In this section examples are presented that demonstrate the scalability of the linear solver and the eigensolver. Performance results are obtained for large model problems. And a parameter study of the sensitivity of the computed eigenvalues to the linear solver and eigensolver accuracy is performed. Before discussing these results, the choice of partitioner and coarse solver are discussed.

A given structure is first decomposed or partitioned using the DOMDEC decomposition package [B.8, B.24, B.9]. Using DOMDEC makes it possible to determine subdomains with low geometric aspect ratio, which is critical for FETI [B.13, B.10].

To apply the FETI method, it is necessary to solve many linear systems with coefficient matrix G^TG. The order of G^TG is proportional to the number of subdomains. We use a full sweep of the conjugate gradient method with stable DGKS [B.7] reorthogonalization to compute the decomposition $(G^TG)^{-1} = PD^{-1}P^T$ where D is diagonal and P is dense. This algorithm is not as efficient as a sparse direct solver, but it is scalable, simple to implement and easy to adapt to more complex applications (e.g. singular G^TG). More efficient algorithms could be used here, such as a parallel sparse direct solver. But for eigenvalue problems the cost of inverting G^TG is amortized over many solves, and the difference in cost of these approaches has been insignificant so far.

The PARPACK package is used in shift-invert mode to compute the smallest eigenvalues of (A, M). A zero shift is used because we have not yet implemented a FETI solver for the indefinite problem $A - M\sigma$. Computing the action of M involves communication along the subdomain interfaces. The tolerance for the eigensolver is used is $\tau = 10^{-9}$. Eigenpairs are approximated using Krylov-based subspaces of dimension three times the number of requested eigenvalues, unless on one eigenvalue is requested, in which case the Krylov subspace dimension is six. The tolerance used for the FETI linear solver is

Table B.1. Number of FETI iterations required to reduce the relative residual error a fixed amount for the linear system arising from modeling a steel cube clamped at one face and loaded on the opposite face. The grid size and number of processors are increased to keep the number of unknowns per processor constant.

Grid	Procs	Iters
6^3	8	20
12^3	64	37
18^3	216	41
24^3	512	43

Table B.2. Solution time for PARPACK/FETI eigensolver on different numbers of processors. The smallest eigenpair of a steel cube clamped at one side is computed. The problem dimension is $100,000$.

Procs	Time (Min)
64	24
125	11
256	6

$\rho = 10n\epsilon$ where ϵ is the machine precision and n is the number of global unknowns. The FETI method is used to compute u such that $\|Au - f\| < \rho\|f\|$.

The first set of model problems are steel cubes clamped at one face and discretized using brick elements. The subdomains are irregular. First the FETI solver was applied to problems on different grids and solved using different numbers of processors, but holding the number of unknowns per processor fixed. The results are tabulated in Table 1. We observed that the FETI linear solver is scalable in the sense that the number of iterations required to solve the linear system to a fixed accuracy grows only modestly as the number of processors increases.

Next the eigensolver was applied to a $32 \times 32 \times 32$ cube with roughly $100,000$ unknowns, and computed the smallest mode using different numbers of processors. These results tabulated in Table 2 were obtained on an INTEL PARAGON platform with 16 megabytes of memory per processor. The solution time scaled with the number of processors. Computing the smallest mode for a $64 \times 64 \times 64$ cube with $800,000$ unknowns took 36 minutes using 512 processors. The time to compute additional modes increases roughly linearly in the number of requested modes.

We have also applied the FETI solver to a family of model structures with more

complex geometry. The structures were analyzed on an INTEL TERAFLOP platform with 128 megabytes of memory per processor. Several materials are used in these structures whose stiffnesses range over six orders of magnitude, and the scaling described in [B.12, B.20] for highly heterogeneous materials is used.

For a model with over one million unknowns, attempts to reduce the residual error by a factor of approximately 10^{-9} failed; after 600 iterations the residual had been reduced by a factor of 10^{-5}. If all of the materials are replaced by a single material, then 200 iterations suffice to solve a typical linear system. If the stiffnesses are changed from the original model so that they range over three orders of magnitude, then approximately 260 iterations suffice to solve a typical linear system and the ten left-most modes are computed in 3.5 hours on 256 processors.

We have investigated the influence on the accuracy of the computed eigenpairs of the accuracies of the FETI linear system solver and PARPACK. The following data is for a model with approximately 3600 unknowns and in which the material stiffnesses range over six orders of magnitude. The model was partitioned into four subdomains and the ten left-most modes were computed using a subspace of dimension thirty. Of the ten lowest modes, the first and the second are equal, the fourth and the fifth are equal, the sixth and the seventh are equal and the tenth and the eleventh are equal. The tenth and eleventh modes are approximately $10^{5.7}$. In each experiment, the modes computed were in fact the left-most modes. Based on quantities computed during computation, the stiffness matrix has condition number greater than 10^6 and norm greater than $10^{8.2}$. One use of this experiment is to demonstate the bound (B.10). In our case $\|s\lambda\| = \lambda$. Because M is a diagonal matrix with entries ranging from 3×10^{-5} to 3×10^{-3} and $Q_j^T M Q_j = I$, we use the rough estimate $\|MQ_j\| \approx 10^{-2}$.

Table 3. requires some explanation. ρ denotes the maximum residual error norm achieved by the FETI linear solver divided by the right hand side norm. τ is the relative tolerance for PARPACK. The eigenpair residuals are given by $r = Ax - Mx\lambda$ where $x^T M x = 1$. The maximum residual error norm predicted by equation (B.10) is compared to the observed value. All logarithms are computed base ten. The columns labeled Solves and Iterations display the number of linear systems solved and the total number of iterations of the FETI linear system solver respectively.

The number of linear solver iterations is large. At first glance the residual errors norms also appear large, but relative to $\|A\| > 10^8$, the norms are reasonable. The values of ρ are observed *a posteriori*. The computation of ρ is nontrivial. The values of τ are upper bounds specified *a priori*.

Table 4. tabulates the number of linear solves and the number of eigen pairs computed for a similar model.

Future work on this approach starts with modifications to FETI necessary to solve highly heterogeneous problems. We are experimenting with modifying the decomposition,

Table B.3. The influence of linear solver accuracy, ρ, and eigensolver accuracy, τ, on the eigenpair residual error norms for computing the ten smallest modes of a 3000 unknown model problem. The number of linear systems solved (Solves), the number of linear solver iterations (Iterations), and the solution time in seconds are also tabulated.

$\log\rho$	$\log\tau$	$\log\|r\|_{obs}$	$\log\|r\|_{pred}$	Solves	Iterations	Seconds
-1.0	-8	3.1	2.7	108	3101	462
-2.1	-8	1.8	1.6	109	3373	496
-3.2	-8	-.9	.7	119	3934	584
-4.4	-10	-.2	-.7	122	4320	647
-5.2	-11	-1.1	-1.5	133	4942	715
-6.2	-12	-1.9	-2.5	135	5268	750
-7.1	-12	-3.0	-3.2	135	5534	770
-8.4	-12	-4.2	-3.7	135	5806	802
-8.8	-5	1.4	3.2	70	3153	447
-8.8	-6	1.4	2.2	70	3153	448
-8.8	-7	-.3	1.2	80	3738	523
-8.8	-8	-2.2	.2	119	5358	752
-8.8	-9	-3.0	-.8	122	5500	759
-8.8	-10	-3.0	-1.8	122	5500	755

Table B.4. The the number of linear systems solved (Solves) and the number of eigen pairs computed (NumEigs) for a 3000 unknown model problem.

NumEigs	4	14	24	34	44	54	64	74	84	94	194
Solves	60	95	116	140	135	163	193	223	211	236	273

the scaled Dirichlet preconditioner and the coarse space. Of course the number of linear system solves per eigenpair can be substantially reduced by shifting. Extensions to solve linear systems in which the stiffness matrix is singular are needed. In this case it is possible to determine $\mathcal{N}(A)$ from $\mathcal{N}(G^TG)$ [B.6] and it is necessary to check for nondegeneracy, $\mathcal{N}(A) \cap \mathcal{N}(M) \neq 0$, given $\mathcal{N}(A)$. Experiments with an eigensolver based on limited accuracy solves, such as a restarted Jacobi-Davidson method for symmetric generalized eigenvalues problems, are needed [B.27, B.26]. Such methods are meant to require fewer linear solver iterations. Techniques to (usually) detect missed modes by restarting will be included in a future release of PARPACK.

An alternative algorithm (that we hope to experiment with) for computing the null space of the subdomain stiffness matrices is to shift, factor, and apply a partial eigensolver on each subdomain. Another approach that we have recently implemented is to determine the null space directly from the subdomain geometry, with out using the subdomain stiffness matrix.

We are indebted to Gene Golub for the observation that instead of implicitly representing the projection as $\Pi = I - G(G^TG)^{-1}G^T$, one can perform a sparse QR factorization $G = QR$ and use $\Pi = I - QQ^T$. This approach is advantageous in the important case in which A is symmetric positive semi-definite. The null space of A must be determined before solving linear systems or eigensystems. The null space of A can be determined from the right null space of G [B.6]. The standard practice is to apply a rank-revealing Cholesky decomposition to G^TG, but this is unstable [B.16]. In contrast a rank-revealing QR factorization of G *is* stable [B.2, B.4, B.5]. This is a subject for further study.

Acknowledgements I would like to thank Raymond Tuminaro for sharing his understanding of the FETI method with me. I would also like to thank John Red-Horse and the referees for their suggestions on improving this document. I would also like to thank Garth Reese, Dan Segalman for developing Salinas and Charbel Farhat, Michel Lesionne, Daniel Rixen and Po Shu Chen for their assistance with the FETI method.

References

[B.1] M. Arioli, I. Duff, and D. Ruiz. Stopping criteria for linear solvers. *SIAM J. Matrix Anal. Applic.*, 13:138–144, 1992.

[B.2] C. Bischof. Incremental condition estimation. *SIAM J. Matrix Anal. Applic.*, 11:312–322, 1990.

[B.3] P. S. Chen C. Farhat and F. Roux. The two-level feti method - part ii: Extension to shell problems, parallel implementation and performance results. Technical Report CU-CAS-96-10, University of Colorado, College of Engineering, Campus Box 429, Boulder, CO 80309, 1996.

[B.4] T. Chan. Reak revealing qr factorizations. *Linear Alg. Appl.*, 88:67–82, 1987.

[B.5] S. Chandrasekaran and Ipsen I. On rank-revealing factorizations. *SIAM J. Matrix Anal. Applic.*, 15:592–622, 1994.

[B.6] J.-M. Cros and F. Léné. Parallel iterative methods to solve large-scale eigenvalue problems in structural dynamics. In Espedal M. Bjø rstad P. and Keyes D., editors, *Proc. Ninth Int. Conf. on Domain Decomposition Meths.* John Wiley and Sons Ltd., 1996.

[B.7] J. W. Daniel, W. B. Gragg, L. Kaufman, and G. W. Stewart. Reorthogonalization and stable algorithms for updating the Gram-Schmidt QR factorization. *Math. Comp.*, 30:772–795, 1976.

[B.8] C. Farhat. simple and efficient automatic fem domain decomposer. *Comput. and Struct.*, 28:579–602, 1988.

[B.9] C. Farhat and M. Lesoinne. Automatic partitioning of unstructured meshes for the parallel solution of problems in computational mechanics. *Internat. J. Numer. Meths. Engrg.*, 36:745–764, 1993.

[B.10] C. Farhat, N. Maman, and M. Lesoinne. Mesh partitioning for implicit computations via iterative domain decomposition: impact and optimization of the subdomain aspect ratio. *Internat. J. Numer. Meths. Engrg.*, 38:989–1000, 1996.

[B.11] C. Farhat and J. Mandel. The two-level feti method for static and dynamic plate problems - part i: An optimal iterative solver for biharmonic systems. Technical Report CU-CAS-95-23, University of Colorado, College of Engineering, Campus Box 429, Boulder, CO 80309, 1995.

[B.12] C. Farhat and D. Rixsen. Preconditioning the feti and balancing domain decomposition methods for problems with intra- and inter-subdomain coefficient jumps. In Espedal M. Bjø rstad P. and Keyes D., editors, *Proc. Ninth Int. Conf. on Domain Decomposition Meths.* John Wiley and Sons Ltd., 1996.

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

[B.14] C. Farhat and F. Roux. Implicit parallel processing in structural mechanics. Technical Report CU-CAS-93-26, University of Colorado, College of Engineering, Campus Box 429, Boulder, CO 80309, 1993.

[B.15] Lewis J. Grimes R. and Simon H. A shifted block lanczos algorithm for solving sparse symmetric generalized eigenproblems. *SIAM J. Matrix Anal. Applic.*, 15:228–272, 1994.

[B.16] N. Higham. *Accuracy and stability of numerical algorithms.* SIAM, Phildelphia, PA, 1996.

[B.17] A. Erisman I. Duff and J. Reid. *Direct methods for sparse matrices.* Clarendon Press, Oxford, 1986.

[B.18] R. Lehoucq and D. Sorensen. Deflation techniques for an implicitly restarted arnoldi iteration. *SIAM J. Matrix Anal. Applic.*, 17:789–821, 1996.

[B.19] R. B. Lehoucq, D. C. Sorensen, and C. Yang. *ARPACK Users' Guide: Solution of Large Scale Eigenvalue Problems with Implicitly Restartëd Arnoldi Methods.* SIAM, Phildelphia, PA, 1998.

[B.20] J. Mandel and M. Brezina. Balancing domain decomposition for problems with large jumps in coefficients. *Math. Comp.*, 65:1387–1401, 1996.

[B.21] K. Maschhoff and D. Sorensen. *PARPACK: An Efficient Portable Large Scale Eigenvalue Package for Distributed Memory Parallel Architectures.* Lecture Notes in Computer Science 1184, Springer-Verlag Berlin, 1995.

[B.22] K. C. Park and C. Felippa. A variational framework for solution method developments in structural mechanics. Technical Report CU-CAS-96-21, University of Colorado, College of Engineering, Campus Box 429, Boulder, CO 80309, 1996.

[B.23] F.-X. Roux. Acceleration of the outer conjugate gradient by reorthogonalization for a domain decomposition method for structural analysis problems. In Chan T., Glowinski R., Périaux J., and Widlund O., editors, *Proc. Third Int. Conf. on Domain Decomposition Meths.*, pages 314–319. SIAM, Philadelphia., 1990.

[B.24] H. Simon. Partitioning of unstructured problems for parallel processing. *Comput. Sys. Engrg.*, 2:135–148, 1991.

[B.25] I. Slapničar. Componentwise analysis of direct factorization of real symmetric and hermitian matrices. *Linear Alg. Appl.*, 272:227–275, 1998.

[B.26] G. L. G. Sleijpen and H.A. van der Vorst. A Jacobi-Davidson iteration method for linear eigenvalue problems. *SIAM J. Matrix Anal. Applic.*, 17(2):401–425, 1996.

[B.27] G.L.G. Sleijpen, J.G.L. Booten, D.R. Fokkema, and H.A. van der Vorst. Jacobi-Davidson type methods for generalized eigenproblems and polynomial eigenproblems. *BIT*, 36(3):595–633, 1996.

[B.28] D.C. Sorensen. Deflation for implicitly restarted arnoldi methods. Technical Report TR98-12, Rice University, Houston, TX 77251-1829, 1998.

[B.29] A. Wathen, B. Fischer, and D. Silvester. The convergence of iterative solution methods for symmetric and indefinite linear systems. In D.F. Griffiths & G.A. Watson, editor, *Numerical Analysis 1997, Pitman Research Notes in Mathematics Series*, pages 230–243. Addison Wesley Longman, Harlow, England, 1997.

Appendix C
Multipoint Constraints

Reference

Charbel Farhat, Catherine Lacour and Daniel Rixen, "Incorporation of Linear Multipoint Constraints in Substructure Based Iterative Solvers. Part 1: A Numerically Scalable Algorithm", Int. J. Numer. Meth. Engng. **43**, 997-1016 (1998).

Summary Development

The above reference outlines the details of the theory governing different methods of implementing linear multipoint constraints (MPCs) in the FETI framework. The constraint equations can be written as

$$Cu = g$$

Where C is a $m \times n$ real matrix, g is a n-long vector, and u is the vector of generalized displacements.

The problem is formulated as a constraint problem using Lagrange multipliers. The degrees of freedom are partitioned into three distinct sets.

1. u, the physical degrees of freedom in the model,
2. λ, the standard constraint equations of FETI, i.e. those equations necessary to insure that displacements are continuous across subdomain boundaries, and
3. μ, the Lagrange multipliers introduced by the MPCs.

The two sets of constraints may be solved simultaneously or one set of constraints may be solved in an inner loop, while the second set is solved in an outer loop. Simultaneous solution for both constraints is difficult to precondition. Solution for μ in the outer iteration is inefficient. Solution for λ in the outer iteration is very similar to the two-level FETI method previously developed. The equations for solution are,

$$\begin{bmatrix} \begin{bmatrix} K & C^T \\ C & 0 \end{bmatrix} & \begin{bmatrix} B & 0 \end{bmatrix}^T \\ \begin{bmatrix} B & 0 \end{bmatrix} & 0 \end{bmatrix} \begin{bmatrix} \begin{bmatrix} u \\ \mu \end{bmatrix} \\ \lambda \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} f \\ g \end{bmatrix} \\ 0 \end{bmatrix} \tag{C.1}$$

Here K is the stiffness matrix, B is the Boolean matrix for application of subdomain interface continuity and f are the applied forces.

The form of this equation is exactly the same as the two-level FETI method (see the reference above or[2]) . In this method corner nodes for shell and plate finite element subdomains are constrained to insure continuity between subdomains. In the case of shell/plate problems, the constraint is optional since the method does converge without it. However, including the constraint greatly improves the convergence rate. For applied MPCs, the constraint is required for correct solution of the equations, but the formalism is identical.

[2]C. Farhat, P. S. Chen, F. Risler and F. X. Roux, "A unified framework for accelerating the convergence of iterative substructuring methods with Lagrange multipliers", Int. J. Numer. Meth. Engng., **42**, 257-288 (1998).

Distribution List:

MS0316 E. Gorham, 9209
MS0318 G. Davidson, 9201
MS0318 P. Heermann, 9215
MS0321 A. L. Hale, 9224
MS0321 W. J. Camp, 9200
MS0439 B. J. Driessen, 9234
MS0439 M. J. Bhardwaj, 9234
MS0439 D. R. Martinez
MS0439 D. J. Segalman, 9234
MS0439 G. M. Reese, 9234 (5)
MS0439 J. R. Red-Horse, 9234
MS0439 K. F. Alvin, 9234
MS0441 R. W. Leland, 9226
MS0443 H. S. Morgan, 9117
MS0557 T. J. Baca, 9119
MS0819 J. Peery, 9231
MS0820 P. Yarrington, 9232
MS0825 W. H. Rutledge, 9115
MS0826 W. Hermina, 9111
MS0827 J. D. Zepper, 9136
MS0827 R. Griffith, 9114
MS0828 J. Garcia, 9106
MS0828 J. Moya, 9105
MS0828 R. K. Thomas, 9104
MS0828 T. C. Bickel, 9101
MS0834 A. C. Ratzel, 9112
MS0835 S. N. Kempka, 9113
MS0836 C. W. Peterson, 9116
MS0841 C. Hartwig, 9102
MS0841 P. J. Hommert, 9100
MS1110 D. M. Day, 9222 (5)
MS1110 D. Womble, 9222
MS1110 N. Pundit, 9223
MS1110 R. S. Tuminaro, 9222 (5)
MS1111 G. Heffelfinger, 9225
MS1111 S. Dosanjh, 9221
MS1135 D. B. Davis, 9134
MS9018 Central Technical File, 8940-2 (1)
MS0899 Technical Library, 4916 (2)
MS0619 Review & Approval Desk, 12690
for DOE/OSTI (1)

Charbel Farhat
University of Colorado at Boulder
Center for Aerospace Structures
Boulder, CO 80309-0429

Michel Lesoinne
University of Colorado at Boulder
Center for Aerospace Structures
Boulder, CO 80309-0429

Kendall Pierson
University of Colorado at Boulder
Center for Aerospace Structures
Boulder, CO 80309-0429

Daniel Rixen
University of Colorado at Boulder
Center for Aerospace Structures
Boulder, CO 80309-0429