

# Parallel Soil–Foundation–Structure Interaction Computations

Boris Jeremić

*Department of Civil and Environmental Engineering,  
University of California,  
Davis, CA 95616,  
jeremic@ucdavis.edu*

Guanzhou Jie

*Wachovia Corporation  
375 Park Ave,  
New York, NY,*

**ABSTRACT:** Described here is the Plastic Domain Decomposition (PDD) Method for parallel elastic-plastic finite element computations related to Soil–Foundation–Structure Interaction (SFSI) problems. The PDD provides for efficient parallel elastic-plastic finite element computations steered by an adaptable, run-time repartitioning of the finite element domain. The adaptable repartitioning aims at balancing computational load among processing nodes (CPUs), while minimizing inter-processor communications and data redistribution during elasto-plastic computations. The PDD method is applied to large scale SFSI problem. Presented examples show scalability and performance of the PDD computations.

A set of illustrative example is used to show efficiency of PDD computations and also to emphasize the importance of coupling of the dynamic characteristics of earthquake, soil and structural (ESS) on overall performance of the SFS system. Above mentioned ESS coupling can only be investigated using detailed models, which dictates use of parallel simulations.

## 1 INTRODUCTION

Parallel finite element computations have been developed for a number of years mostly for elastic solids and structures. The static domain decomposition (DD) methodology is currently used almost exclusively for decomposing such elastic finite element domains in subdomains. This subdivision has two main purposes, namely (a) to distribute element computations to CPUs in an even manner and (b) to distribute system of equations evenly to CPUs for maximum efficiency in solution process.

However, in the case of inelastic (elastic–plastic) computations, the static DD is not the most efficient method since some subdomains become computationally slow as the elastic–plastic zone propagates through the domain. This propagation of the elastic–plastic zone (extent of which is not known a-priori) will in turn slow down element level computations (constitutive level iterations) significantly, depending on the complexity of the material model used. Propagation of elastic–plastic zone will eventually result in some subdomains

becoming significantly computationally slow while others, that are still mostly elastic, will be more computationally efficient. This discrepancy in computational efficiency between different subdomains will result in inefficient parallel performance. In other words, subdomains (and their respective CPUs) with mostly elastic elements will be finishing their local iterations much faster (and idle afterward) than subdomains (and their respective CPUs) that have many elastic–plastic elements.

This computational imbalance motivated development of the Plastic Domain Decomposition (PDD) method described in this paper. Developed PDD is applied to a large scale seismic soil–foundation–structure (SFS) interaction problem for bridge systems. It is important to note that the detailed analysis of seismic SFSI described in this paper is made possible with the development of PDD as the modeling requirements (finite element mesh size) were such that sequential simulations were out of questions.

## **1.1 Soil–Structure Interaction Motivation**

The main motivation for the development of PDD is the need for detailed analysis of realistic, large scale SFSI models. This motivation is emphasized by noting that currently, for a vast majority of numerical simulations of seismic response of bridge structures, the input excitations are defined either from a family of damped response spectra or as one or more time histories of ground acceleration. These input excitations are applied simultaneously along the base of a structural system, usually without taking into account its dimensions and dynamic characteristics, the properties of the soil material in foundations, or the nature of the ground motions themselves. Ground motions applied in such a way neglect the soil–structure interaction (SSI) effects, that can significantly change used free field ground motions. A number of papers in recent years have investigated the influence of the SSI on behavior of bridges (7; 9; 25; 31; 32; 34; 43). McCallen and Romstadt (34) performed a remarkable full scale analysis of the soil–foundation–bridge system. The soil material (cohesionless soil, sand) was modeled using equivalent elastic approach (using Ramberg–Osgood material model through standard modulus reduction and damping curves developed by Seed et al. (40)). The two studies by Chen and Penzien (7) and by Dendrou et al. (9) analyzed the bridge system including the soil, however using coarse finite element meshes which might filter out certainly, significant higher frequencies. Jeremić et al. (21) attempted a detailed, complete bridge system analysis. However, due to computational limitations, the large scale pile group had to be modeled separately and its stiffness used with the bridge structural model. In present work, with the development of PDD (described in some details), such computational limitations are removed and high fidelity, detailed models of Earthquake–Soil–Structure systems can be performed. It is very important to note that proper modeling of seismic wave propagation in elastic–plastic soils dictates the size of finite element mesh. This requirement for proper seismic wave propagation will in turn results in a large number of finite elements that need to be used.

## **1.2 Parallel Computing Background**

The idea of domain decomposition method can be found in an original paper from 1870 by H.A. Schwarz (Rixena and Magoulès (38)). Current state of the art in distributed computing in computational mechanics can

be followed to early works on parallel simulation technology. For example, early endeavors using inelastic finite elements focused on structural problems within tightly coupled, shared memory parallel architectures. We mention work by Noor et al. (36), Utku et al. (44) and Storaasil and Bergan (42) in which they used substructuring to achieve distributed parallelism. Fulton and Su (16) developed techniques to account for different types of elements used in the same computer model but used substructures made of same element types which resulted in non-efficient use of compute resources. Hajjar and Abel (17) developed techniques for dynamic analysis of framed structures with the objective of minimizing communications for a high speed, local grid of computer resource. Klaas et al. (27) developed parallel computational techniques for elastic-plastic problems but tied the algorithm to the specific multiprocessor computers used (and specific network connectivity architecture) thus rendering it less general and non-useful for other types of grid arrangements. Farhat (11) developed the so-called Greedy domain partitioning algorithm, which proved to be quite efficient on a number of parallel computer architectures available. However, most of the above approaches develop loss of efficiency when used on a heterogeneous computational grid, which constitutive currently predominant parallel computer architecture. More recently Farhat et al. (12; 13; 14) proposed FETI (Finite Element Tearing and Interconnecting) method for domain decomposition analysis. In FETI method, Lagrange multipliers are introduced to enforce compatibility at the interface nodes.

Although much work has been presented on domain decomposition methods, the most popular methods such as FETI-type are based on subdomain interface constraints handling. It is also interesting to note promising efforts on merging of iterative solving with domain decomposition-type preconditioning (Pavarino (37) and Li and Widlund (30)). From the implementation point of view, for mesh-based scientific computations, domain decomposition corresponds to the problem of mapping a mesh onto a set of processors, which is well defined as a graph partitioning problem (Schlegel et al. (39)). Graph based approach for initial partitioning and subsequent repartitioning forms a basis for PDD method.

### 1.3 Simulation Platform

Developed parallel simulation program is developed using a a number of numerical libraries. Graph partitioning and repartitioning is achieved using parts of the ParMETIS libraries (Karypis et al. (26)). Parts of the OpenSees framework (McKenna (35)) were used to connect the finite element domain. In particular, Finite Element Model Classes from OpenSees (namely, class abstractions Node, Element, Constraint, Load and Domain) where used to describe the finite element model and to store the results of the analysis performed on the model. In addition to that, an existing Analysis Classes were used as basis for development of parallel PDD framework which is then used to drive the global level finite element analysis, i.e., to form and solve the global system of equations in parallel. Actor model (1; 18) was used and with addition of a Shadow, Chanel, MovableObject, ObjectBroker, MachineBroker classes within the OpenSees framework (35) provided an excellent basis for our development. On a lower level, a set of Template3Dep numerical libraries (Jeremić and Yang (24)) were used for constitutive level integrations, nDarray numerical libraries (Jeremić and Sture (23)) were used to handle vector, matrix and tensor manipulations, while FEMtools el-

ement libraries from the UCD CompGeoMech toolset (Jeremić (20)) were used to supply other necessary components. Parallel solution of the system of equations has been provided by PETSc set of numerical libraries (Balay et al. (3; 4; 5)).

Most of the simulations were carried out on our local parallel computer GeoWulf. Only the largest models (too big to fit in GeoWulf system) were simulated on TeraGrid machine (at SDSC and TACC). It should be noted that program sources described here are available through Author's web site.

## 2 PLASTIC DOMAIN DECOMPOSITION METHOD

Domain Decomposition (DD) approach is one of the most popular methods that is used to implement and perform parallel finite element simulations. The underlying idea is to divide the problem domain into subdomains so that finite element calculations will be performed on each individual subdomain in parallel. The DD can be overlapping or non-overlapping. The overlapping domain decomposition method divides the problem domain into several slightly overlapping subdomains. Non-overlapping domain decomposition is extensively used in continuum finite element modeling due to the relative ease to program and organize computations and is the one that will be examined in this work. In general, a good non-overlapping decomposition algorithm should be able to (a) handle irregular mesh of arbitrarily shaped domain, and (b) minimize the interface problem size by delivering minimum boundary conductivities, which will help reducing the communication overheads. Elastic–plastic computations introduce a number of additional requirements for parallel computing. Those requirements are described below.

### 2.1 The Elastic–Plastic Parallel Finite Element Computational Problem

The distinct feature of elastic-plastic finite element computations is the presence of two iteration levels. In a standard displacement based finite element implementation, constitutive driver at each integration (Gauss) point iterates in stress and internal variable space, computes the updated stress state, constitutive stiffness tensor and delivers them to the finite element functions. Finite element functions then use the updated stresses and stiffness tensors to integrate new (internal) nodal forces and element stiffness matrix. Then, on global level, nonlinear equations are iterated on until equilibrium between internal and external forces is satisfied within some tolerance.

**Elastic Computations.** In the case of elastic computations, constitutive driver has a simple task of computing increment in stresses ( $\Delta\sigma_{ij}$ ) for a given deformation increment ( $\Delta\epsilon_{kl}$ ), through a closed form equation ( $\Delta\sigma_{ij} = E_{ijkl}\Delta\epsilon_{kl}$ ). It is important to note that in this case the amount of work per Gauss point is known in advance and is the same for every integration point. If we assume the same number of integration points per element, it follows that the amount of computational work is the same for each element and it is known in advance.

**Elastic-Plastic Computations.** For elastic-plastic problems, for a given incremental deformation the constitutive driver iterate in stress and internal variable space until consistency condition is satisfied. Number of implicit constitutive iterations is not known in advance. Similarly, if explicit constitutive computations are done, the amount of work at each Gauss point is much higher than it was for elastic step. Initially, all Gauss points are in elastic state, but as the incremental loads are applied, the elastic-plastic zones develop. For integration points still in elastic range, computational load is light. However, for Gauss points that are elastic-plastic, the computational load increases significantly (more so for implicit computations than for explicit ones). This computational load increase depends on the complexity of material model. For example, constitutive level integration algorithms for soils, concrete, rocks, foams and other granular materials can be very computationally demanding. More than 70% of wall clock time during an elastic-plastic finite element analysis might be spent in constitutive level iterations. This is in sharp contrast with elastic computations where the dominant part is solving the system of equations which consumes about 90% of run time. The extent of additional, constitutive level computations is not known before the actual computations are over. In other words, the extent of elastic-plastic domain is not known ahead of time.

The traditional pre-processing type of Domain Decomposition method (also known as topological DD) splits domain based on the initial geometry and mesh connectivity and assigns roughly the same number of elements to every computational node while minimizing the size of subdomain boundaries. This approach might result in serious computational load imbalance for elastic-plastic problems. For example, one subdomain might be assigned all of the elastic-plastic elements and spend large amount of time in constitutive level computations. The other subdomain might have elements in elastic state and thus spend far less computational time in computing stress increments. This results in program having to wait for the slowest subdomain (the one with large number of elastic-plastic finite elements) to complete constitutive level iterations and only proceed with global system iterations after that.

The two main challenges with computational load balancing for elastic-plastic computations are that they need to be:

- Adaptive, dynamically load balancing computations, as the extent of elastic and elastic-plastic domains changes dynamically and unpredictably during the course of the computation.
- Multiphase computations, as elastic-plastic computations follow up the elastic computations and there is a synchronization phase between those two. The existence of the synchronization step between the two phases of the computation requires that each phase be individually load balanced.

## 2.2 PDD Algorithm

The Plastic Domain Decomposition algorithm (PDD) provides for computational load balanced subdomains, minimizes subdomain boundaries and minimizes the cost of data redistribution during dynamic load balancing. The PDD optimization algorithm is based on dynamically monitoring both data redistribution and analysis model regeneration costs during program execution in addition to collecting information about the

cost of constitutive level iterations within each finite element. A static domain decomposition is used to create initial partitioning, which is used for the first load increment. Computational load (re-)balancing will (might) be triggered if, during elastic-plastic computations in parallel, one computations on one of compute nodes (one of subdomains) becomes slower than the others compute nodes (other subdomains). Algorithm for computational load balancing (CLB) is triggered if the performance gain resulting from CLB offsets the extra cost associated with the repartitioning. The decision on whether to trigger repartitioning or not is based on an algorithm described in some details below.

We define the global overhead associated with load balancing operation to consist of two parts, data communication cost  $T_{comm}$  and finite element model regeneration cost  $T_{regen}$ ,

$$T_{overhead} := T_{comm} + T_{regen} \quad (1)$$

Performance counters are setup within the program to follow both. Data communication patterns characterizing the network configuration can be readily measured ( $T_{comm}$ ) as the program runs the initial partitioning. Initial (static) domain decomposition is performed for the first load step. The cost of networking is inherently changing as the network condition might vary as simulation progresses, so whenever data redistribution happens, the metric is automatically updated to reflect the most current network conditions. Model regeneration cost ( $T_{regen}$ ) is a result of a need to regenerate the analysis model whenever elements (and nodes) are moved between computational nodes (CPUs). It is important to note that model (re-) generation also happens initially when the first data distribution is done (from the static DD). Such initial DD phase provides excellent initial estimate of the model regeneration cost on any specific hardware configurations. This ability to measure realistic compute costs allows developed algorithm (PDD) to be used on multiple generation parallel computers.

For the load balancing operations to be effective, the  $T_{overhead}$  has to be offset by the performance gain  $T_{gain}$ . Finite element mesh for the given model is represented by a graph, where each finite element is represented by a graph vertex. The computational load imposed by each finite element (FE) is represented by the associated vertex weight  $vwgt[i]$ . If the summation  $SUM$  operation is applied on every single processing node, the exact computational distribution among processors can be obtained as total wall clock time for each CPU

$$T_j := \sum_{i=1}^n vwgt[i], j = 1, 2, \dots, np \quad (2)$$

where  $n$  is the number of elements on each processing domain and  $np$  is the number of CPUs. The wall clock time is controlled by  $T_{max}$ , defined as

$$T_{sum} := sum(T_j), T_{max} := max(T_j), \text{ and } T_{min} := min(T_j), j = 1, 2, \dots, np \quad (3)$$

Minimizing  $T_{max}$  becomes here the main objective. Computational load balancing operations comprises delivering evenly distributed computational loads among processors. Theoretically, the best execution time is,

$$T_{best} := T_{sum}/np, \text{ and } T_j \equiv T_{best}, j = 1, 2, \dots, np \quad (4)$$

if the perfect load balance is to be achieved.

Based on definitions above, the best performance gain  $T_{gain}$  that can be obtained from computational load balancing operation as,

$$T_{gain} := T_{max} - T_{best} \quad (5)$$

Finally, the load balancing operation will be beneficial iff

$$T_{gain} \geq T_{overhead} = T_{comm} + T_{regen} \quad (6)$$

Previous equation is used in deciding if the re-partitioning is triggered in current incremental step. It is important to note that PDD will always outperform static DD as static DD represents the first decomposition of the computational domain. If such decomposition becomes computationally unbalanced and efficiency can be gained by repartitioning, PDD be triggered and the  $T_{max}$  will be minimized.

### 2.3 Solution of the System of Equations

A number of different algorithms and implementations exists for solving unsymmetric<sup>1</sup> systems of equations in parallel. In presented work, use was made of both iterative and direct solvers as available through the PETSc interface (Balay et al. (3; 4; 5)). Direct solvers, including MUMPS, SPOOLES, SuperLU, PLAPACK have been tested and used for performance assessment. In addition to that, iterative solvers, including GMRES, as well as preconditioning techniques (Jacobi, inconsistency LU decomposition and approximate inverse preconditioners) for Krylov methods have been also used and their performance assessed.

Some of the conclusions drawn are that, for our specific finite element models (non-symmetric, using penalty method for some connections, possible softening behavior), direct solvers outperform the iterative solver significantly. As expected direct solver were not as scalable as iterative solvers, however, specifics of our finite element models (dealing with soil-structure interaction) resulted in poor initial performance of iterative solvers, that, even with excellent performance scaling, could not catch up with the efficiency of direct solvers. IT is also important to note that parallel direct solvers, such as MUMPS and SPOOLES provided the best performance and would be recommended for use with finite element models that, as ours did, feature non-symmetry, are poorly conditioned (they are ill-posed due to use of penalty method) and can be negative definite (for softening materials).

### 2.4 PDD Scalability Study

The developed PDD method was tested on a number of static and dynamic examples. Presented here are scalability (speed-up) results for a series of soil-foundation-structure model runs. A hierarchy of models,

---

<sup>1</sup>Non-associated elasto-plasticity results in a non-symmetric stiffness tensors, which result in non-symmetric system of finite element equations. Non-symmetry can also result from the use of consistent stiffness operators as described by Jeremić and Sture (22).

described later in section 3, was used in scaling study for dynamic runs. Finite element models were subjected to recorded earthquakes (two of them, described in section 4), using DRM for seismic load application (Bielak et al. (6) and Yoshimura et al (45)).

Total wall clock time has been recorded and used to analyze the parallel scalability of PDD, presented in Figure 1.

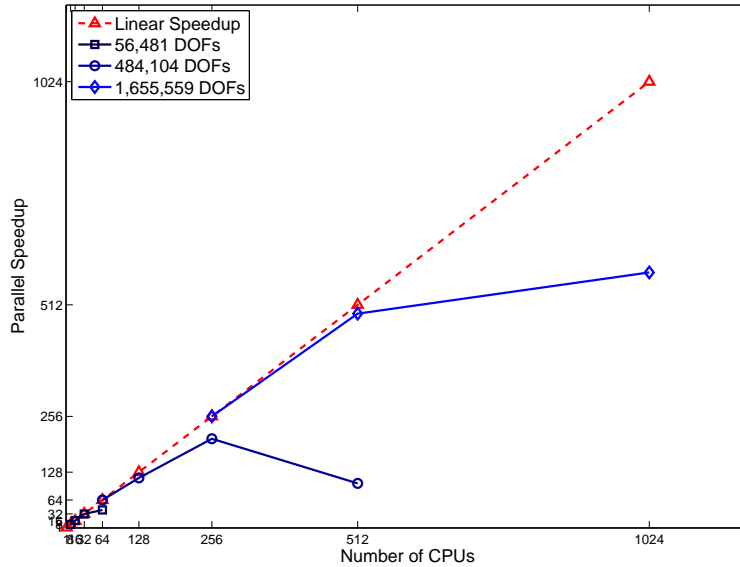


Figure 1: Scalability Study on 3 Bent SFSI Models, DRM Earthquake Loading, Transient Analysis, ITR=1e-3, Imbal Tol 5%.

There is a number of interesting observations about the performance scaling results shown in Figure 1:

- the scalability is quite linear for small number of DOFs (elements),
- there is a link, relation between number of DOFs (elements) and number of CPUs which governs the parallel efficiency. In other words, there exists certain ratio of the number of DOFs to number of CPUs after which the communication overhead starts to be significant. For example for a models with 484,104 DOFs in Figure 1, the computations with 256 CPUs are more efficient that those with 512 CPUs. This means that for the same number of DOFs (elements) doubling the number of CPUs does not help, rather it is detrimental as there is far more communication between CPUs which destroys the efficiency of the parallel computation. Similar trend is observable for the large model with 1,655,559 DOFs, where 1024 CPUs will still help (barely) increase the parallel efficiency.
- Another interesting observation has to do with the relative computational balance of local, element level computations (local equilibrium iterations) and the system of equations solver. PDD scales very



nicely as its main efficiency objective is to have equal computational load for element level computations. However, the efficiency of the system of equations solver becomes more prominent when element level computations are less prominent (if they have been significantly optimized with a large efficiency gain). For example, for the model with 56,481 DOFs it is observed that for sequential case (1 CPU), elemental computation amount for approx. 70 % of wall clock time. For the same model, in parallel with 8 CPUs, element level computation accounts for approx. 40% of wall clock time, while for 32 CPUs the element level computation account for only 10% of total wall clock time. In other words, as the number of CPUs increase, the element level computations are becoming very efficient and the main efficiency gain can then be made with the system of equations solver. However, it is important to note that parallel direct solver are not scalable up to large number of CPUs (Demmel et al. (8)) while parallel iterative solver are much more scalable but difficult to guarantee convergence. This observation can be used in fine tuning of parallel computing efficiency, even if it clearly points to a number of possible problems.

### 3 FINITE ELEMENT SFSI MODEL DEVELOPMENT

The finite element models used in this study have combined both elastic–plastic solid elements, used for soils, and elastic and elastic–plastic structural elements, used for concrete piles, piers, beams and superstructure. In this section described are material and finite element models used for both soil and structural components. In addition to that, described is the methodology used for seismic force application and staged construction of the model,

#### 3.1 Soil and Structural Model

**Soil Models.** Two types of soil were used in modeling. First type of soil was based on stiff, sandy soil, with limited calibration data (Kurtulus et al. (28)) available from capitol Aggregates site (south–east of Austin, Texas).

Based on the stress-strain curve obtained from a triaxial test, a nonlinear elastic-plastic soil model has been developed using Template Elastic plastic framework (24). Developed model consists of a Drucker-Prager yield surface, Drucker–Prager plastic flow directions (potential surface) and a nonlinear Armstrong-Frederick (rotational) kinematic hardening rule (2). Initial opening of a Drucker–Prager cone was set at  $5^\circ$  only while the actual deviatoric hardening is produced using Armstrong–Frederick nonlinear kinematic hardening with hardening constants  $a = 116.0$  and  $b = 80.0$ .

Second type of soil used in modeling was soft clay (Bay Mud). This type of soil was modeled using a total stress approach with an elastic perfectly plastic von Mises yield surface and plastic potential function. The shear strength for such Bay Mud material was set at  $C_u = 5.0$  kPa. Since this soil is treated as fully saturated and there is not enough time during shaking for any dissipation to occur, the elastic–perfectly plastic model provides enough modeling accuracy.

**Soil Element Size Determination.** The accuracy of a numerical simulation of seismic wave propagation in a dynamic SFSI problem is mainly controlled the spacing of the nodes of the finite element model. In order to represent a traveling wave of a given frequency accurately, about 10 nodes per wavelength are required. In order to determine the appropriate maximum grid spacing the highest relevant frequency  $f_{max}$  that is to be simulated in the model needs to be decided upon. Typically, for seismic analysis one can assume  $f_{max} = 10Hz$ . By choosing the wavelength  $\lambda_{min} = v/f_{max}$ , where  $v$  is the (shear) wave velocity, to be represented by 10 nodes the smallest wavelength that can still be captured with any confidence is  $\lambda = 2\Delta h$ , corresponding to a frequency of  $5f_{max}$ . The maximum grid spacing  $\Delta h$  should therefore not be larger than

$$\Delta h \leq \frac{\lambda_{min}}{10} = \frac{v}{10f_{max}} \quad (7)$$

where  $v$  is the smallest wave velocity that is of interest in the simulation (usually the wave velocity of the softest soil layer). In addition to that, mechanical properties of soil will change with (cyclic) loadings as plastification develops. Moduli reduction curve ( $G/G_{max}$ ) can then used to determine soil element size while taking into account soil stiffness degradation (plastification). Using shear wave velocity relation to shear modulus

$$v_{shear} = \sqrt{\frac{G}{\rho}} \quad (8)$$

one can readily obtain the dynamic degradation of wave velocities. This leads to smaller element size required for detailed simulation of wave propagation in soils which have stiffness degradation (plastification).

Table 1 presents an overview of model size (number of elements and element size) as function of cut-off frequencies represented in the model, material (soil) stiffness (given in terms of  $G/G_{max}$ ) and amount of shear deformation for given stiffness reduction. It is important to note that 3D nonlinear elastic–plastic

Table 1: Variation in model size (number of elements and element size) as function of frequency, stiffness and shear deformation.

model size (# of elements)	element size	$f_{cutoff}$	min. $G/G_{max}$	shear def. $\gamma$
12K	1.0 m	10 Hz	1.0	<0.5 %
15K	0.9 m	3 Hz	0.08	1.0 %
150K	0.3 m	10 Hz	0.08	1.0 %
500K	0.15 m	10 Hz	0.02	5.0 %

finite element simulations were performed, while stiffness reduction curves were used for calibration of the material model and for determining (minimal) finite element size. It should also be noted that the largest FEM model had over 0.5 million elements and over 1.6 million DOFs. However, most of simulations were performed with smaller model (with 150K elements) as it represented mechanics of the problem with appropriate level of accuracy. Working FEM model mesh is shown in Figure 2. The model used, features 484,104

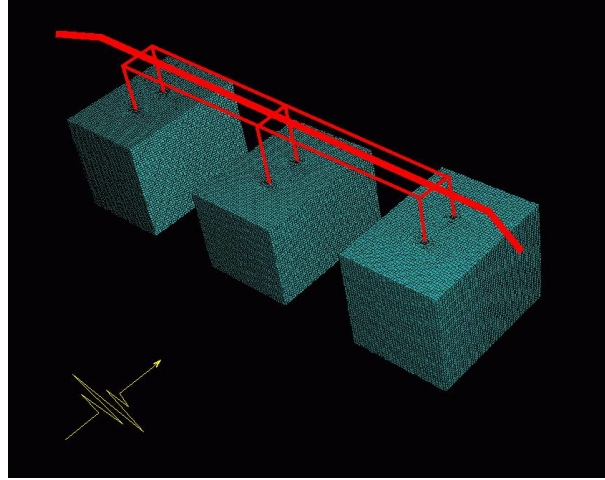


Figure 2: Detailed Three Bent Prototype SFSI Finite Element Model, 484,104 DOFs, 151,264 Elements used for most simulation in this study.

DOFs, 151,264 soil and beam–column elements, and is intended to model appropriately seismic waves of up to 10Hz, for minimal stiffness degradation of  $G/G_{max} = 0.08$ , maximum shear strain of  $\gamma = 1\%$  and with the maximal element size  $\Delta h = 0.3$  m. The largest model (1.6 million DOFs) was able to capture 10 Hz motions, for  $G/G_{max} = 0.02$ , and maximum shear strain of  $\gamma = 5\%$  (see Table 1).

**Structural Models.** The nonlinear structure model (the piers and the superstructure) used in this study were initially developed by Fenves and Dryden (15). This original structural (only) model was subsequently updated to include piles and surrounding soil, and zero length elements (modeling concentrated plastic hinges) were removed from the bottom of piers at the connection to piles. Concrete material was modeled using Concrete01 uniaxial material as available in OpenSees framework (Fenves and Dryden (10; 15)). Material model parameters used for unconfined concrete in the simulation models were  $f'_{co} = 5.9$  ksi,  $\epsilon_{co} = 0.002$ ,  $f'_{cu} = 0.0$  ksi, and  $\epsilon_{cu} = 0.006$ . Material parameters for confined concrete used were  $f'_{co} = 7.5$  ksi,  $\epsilon_{co} = 0.0048$ ,  $f'_{cu} = 4.8$  ksi, and  $\epsilon_{cu} = 0.022$ .

Hysteretic uniaxial material model available within OpenSees framework was selected to model the response of the steel reinforcement. The parameters included in this model are  $F_1 = 67$  ksi,  $\epsilon_1 = 0.0023$ ,  $F_2 = 92$  ksi,  $\epsilon_2 = 0.028$ ,  $F_3 = 97$  ksi, and  $\epsilon_3 = 0.12$ . No allowance for pinching or damage under cyclic loading has been made ( $pinchX = pinchY = 1.0$ ,  $damage1 = damage2 = 0.0$ ,  $beta = 0$ ).

The finite element model for piers and piles features a nonlinear fiber beam–column element Spacone et al. (41). In addition to that, a zero-length elements is introduced at the top of the piers in order to capture the effect of the rigid body rotation at the joints due to elongation of the anchored reinforcement. Cross section of both piers and piles was discretized using  $4 \times 16$  subdivisions of core and  $2 \times 16$  subdivisions of cover for radial and tangential direction respectively. Additional deformation that can develop at the upper

pier end results from elongation of the steel reinforcement at beam–column joint with the superstructure and is modeled using a simplified hinge model (Mazzoni (33)). Parameters used for steel–concrete bond stress distribution were  $u_e = 12\sqrt{f'_c}$  and  $u_e = 6\sqrt{f'_c}$  (Lehman and Moehle (29)).

The bent cap beams were modeled as linear elastic beam-column elements with geometric properties developed using effective width of the cap beam and reduction of its stiffness due to cracking. The superstructure consists of prismatic prestressed concrete members, and was also modeled as linear elastic beam–column element.

**Coupling of Structural and Soil Models.** In order to create a model of a complete soil–structure system, it was necessary to couple structural and soil (solid) finite elements. Figure 3 shows schematics of coupling between structural (piles) and solid (soil) finite elements. The volume that would be physically occupied by

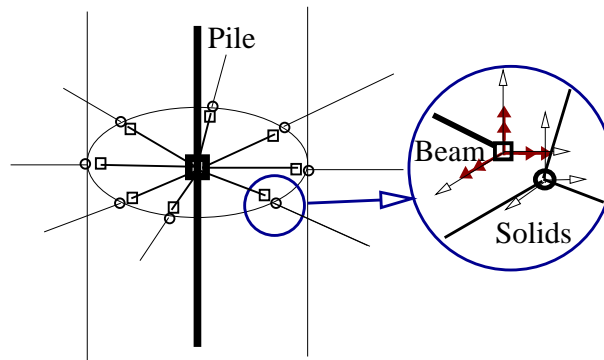


Figure 3: Schematic description of coupling of structural elements (piles) with solid elements (soil).

the pile is left open within the solid mesh that models the foundation soil. This opening (hole) is excavated during a staged construction process (described later). Beam–column elements (representing piles) are then placed in the middle of this opening. Beam–column elements representing pile are connected to the surrounding solid (soil) elements by means of stiff short elastic beam–column elements. These short “connection” beam–column elements extend from each pile beam–column node to surrounding nodes of solids (soil) elements. The connectivity of short, connection beam–column element nodes to nodes of soil (solids) is done only for translational degrees of freedom (three of them for each node), while the rotational degrees of freedom (three of them) from the beam–column element are left unconnected.

### 3.2 Application of Earthquake Motions

Seismic ground motions were applied to the SSI finite element model using Domain Reduction Method (DRM, Bielak et al. (6) and Yoshimura et al (45)). The DRM is an excellent method that can consistently apply ground motions to the finite element model. The method features a two-stage strategy for complex, realistic three dimensional earthquake engineering simulations. The first is an auxiliary problem that simulates

the earthquake source and propagation path effects with a model that encompasses the source and a free field (from which the soil–structure system has been removed). The second problem models local, soil-structure effects. Its input is a set of effective forces, that are derived from the first step. These forces act only within a single layer of elements adjacent to the interface between the exterior region and the geological feature of interest. While the DRM allows for application of arbitrary, 3D wave fields to the finite element model, in this study a vertically propagating wave field was used. Using given surface, free field ground motions, de-convolution was done for this motions to a depth of 100 m. Then, a vertically propagating wave field was (re-) created and used to create the effective forces for DRM. Deconvolution and (back) propagation of vertically propagating wave field was performed using closed form, 1D solution as implemented in Shake program (Idriss and Sun (19)).

### **3.3 Staged Simulations**

Application of loads in stages is essential for elastic–plastic models. This is especially true for models of soil and concrete. Staged loading ensures appropriate initial conditions for each loading stage. Modeling starts from a zero stress and deformation state. Three loading stages, described below, then follow.

**Soil Self–Weight Stage.** During this stage the finite element model for soil (only, no structure) is loaded with soil self–weight. The finite element model for this stage excludes any structural elements, the opening (hole) where the pile will be placed is full of soil. Displacement boundary conditions on the sides of the three soil blocks are such that they allow vertical movements, and allow horizontal in boundary plane movement, while they prohibit out of boundary plane movement of soils. All the displacements are suppressed at the bottom of all three soil blocks. The soil self weight is applied in 10 incremental steps.

**Piles, Columns and Superstructure Self–Weight Stage.** In this, second stage, number of changes to the model happen. First, soil elements where piles will be placed are removed (excavated), then concrete piles (beam–column elements) are placed in the holes (while appropriately connecting structural and solids degrees of freedom, as described above), columns are placed on top of piles and finally the superstructure is placed on top of columns. All of this construction is done at once. With all the components in place, the self weight analysis of the piles–columns–superstructure system is performed.

**Seismic Shaking Stage.** The last stage in our analysis consists of applying seismic shaking, by means of effective forces using DRM. It is important to note that seismic shaking is applied to the already deformed model, with all the stresses, internal variables and deformation that resulted from first two stages of loading.

## 4 SIMULATION RESULTS

Bridge model described above was used to analyze a number of cases of different foundation soils and earthquake excitations. Two sets of ground motions were used for the same bridge structure. Variation of foundation soil, namely (a) all stiff sand and (b) all soft clay. One of the main goals was to investigate if free field motions can be directly used for structural model input (as is almost exclusively done nowadays), that is, to investigate how significant are the SFSI effects. In addition to that, investigated were differences in structural response that result from varying soil conditions. Ground motions for Northridge and Kocaeli earthquakes (free field measurement, see Figure 4) were used in determining appropriate wave field (using DRM). Since the main aim of the exercise was to investigate SFS system a set of short period motions were chosen among Northridge motions records, while long period motions from Kocaeli earthquakes were used for long period example.

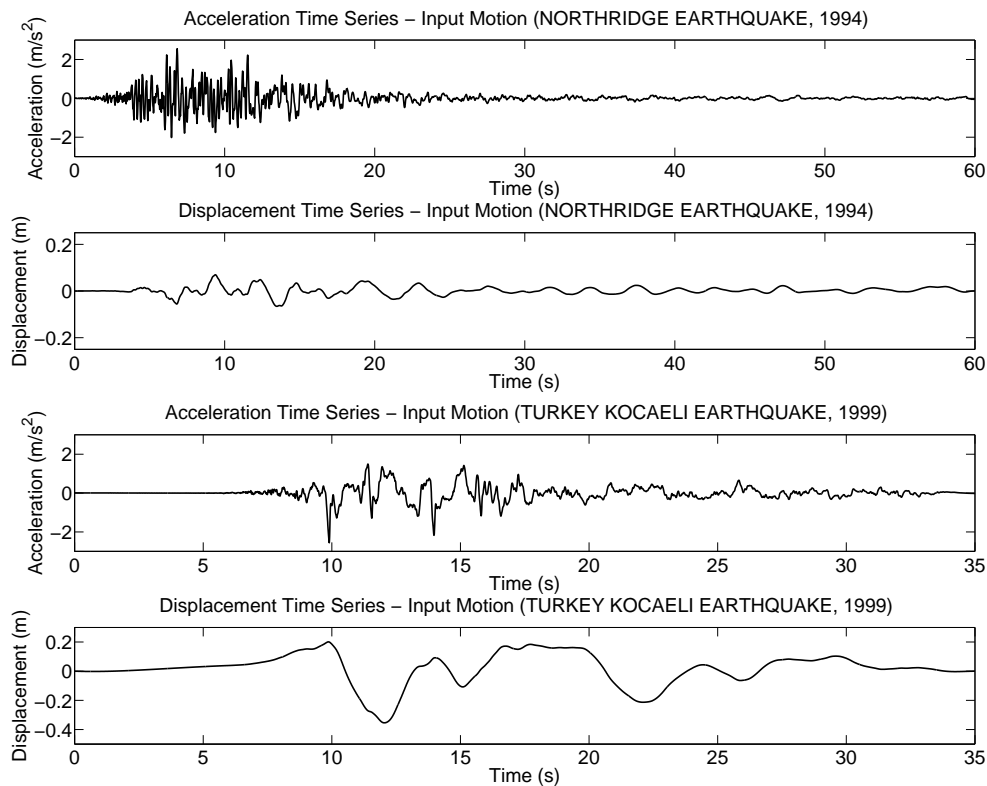


Figure 4: Input motions: short period (Northridge) and long period (Kocaeli).

A number of very interesting results were obtained and are discussed below.

**Free Field vs. SSI Motions.** A very important aspect of SFSI is the difference between free field motions and the motions that are changed (affected) but the presence of the structure. Figure 5 shows comparison of

free field short period motions (obtained by vertical propagation of earthquake motions through the model without the presence of bridge structure and piles) and the ones recorded at the base of column of the left bent in stiff and soft soils. It is immediately obvious that the free field motions in this case do not correspond

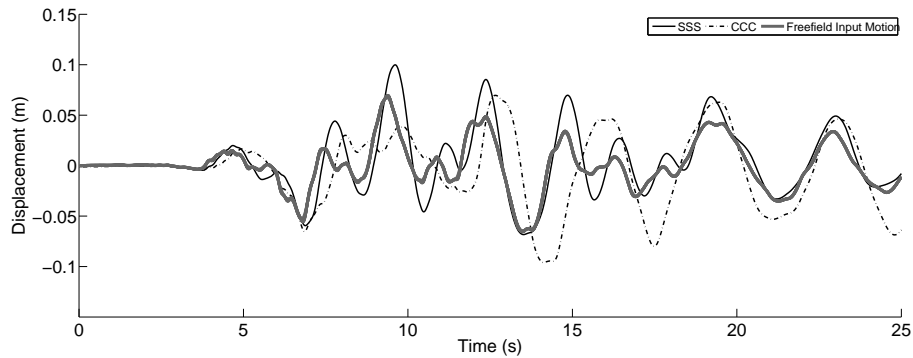


Figure 5: Comparison of free field versus measured (prototype model) motions at the base of left bent for the short period motions (Northridge) for all clay (CCC) and all sand (SSS) soils.

to motions observed in bridge SFS system with stiff or soft soils. In fact, both the amplitude and period are significantly altered for both soft and stiff soil and the bridge structure. This quite different behavior can be explained by taking into account the fact that the short period motions excite natural periods of stiff soil and can produce (significant) amplifications. In addition to that, for soft soils, significant elongation of period is observed.

On the other hand, as shown in Figure 6 the same SFS system (same structure with stiff or soft soil beneath) responds quite a bit different to long period motions. The difference between free field motions

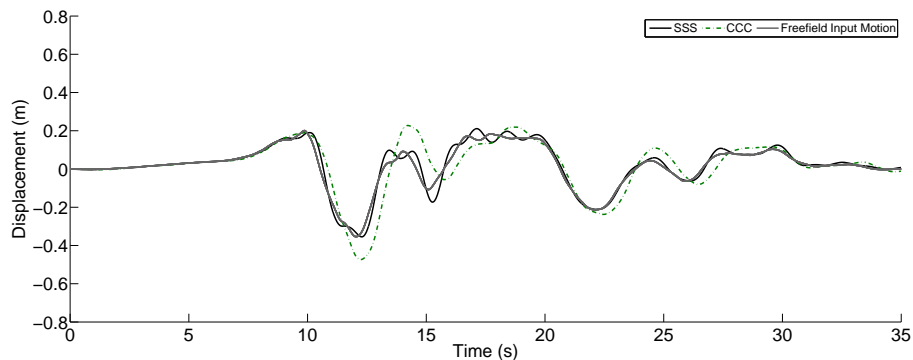


Figure 6: Comparison of free field versus measured (prototype model) motions at the base of left bent for the long period motions (Kocaeli) for all clay (CCC) and all sand (SSS) soils.

and the motions measured (simulated) in stiff soils is smaller in this case. This is understandable as the stiff

soil virtually gets carried away as (almost) rigid block on such long period motions. For the soft soil one of the predominant natural periods of the SFS system is excited briefly (at 12-17 seconds) but other than that excursion, both stiff and soft soil show fair matching with free field motions. In this case the SFS effects are not that pronounced, except during the above mentioned period between 12 and 17 seconds.

**Bending Moments Response.** Influence of variable soil conditions and of dynamic characteristic of earthquake motions on structural response is followed by observing bending moment response. For this particular purpose, a time history of bending moment at the top of one of the piers of bent # 1 (left most in Figure 2) is chosen to illustrate differences in behavior.

Figure 7 shows time history of the bending moment at top of left most pier of bent # 1 for all sand (SSS) and all clay (CCC) cases for short period motion (Northridge).

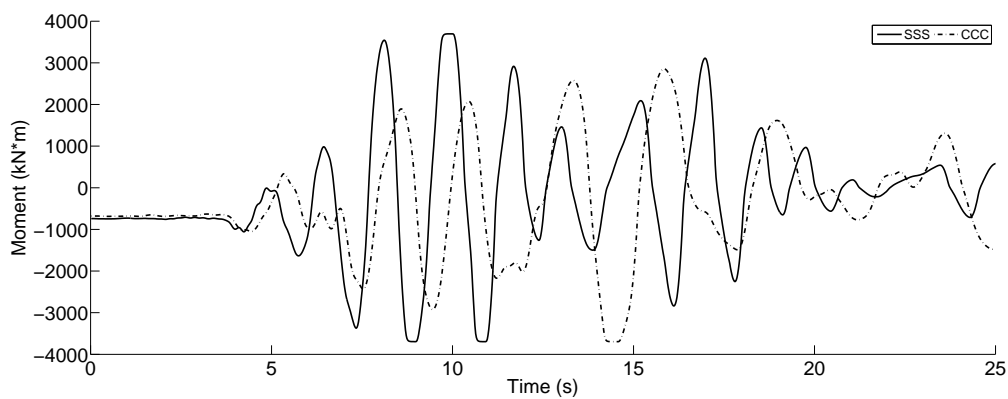


Figure 7: Simulated bending moment time series (top of left pier) for short period motion (Northridge), for all clay (CCC) and all sand (SSS) soils.

Similarly, Figure 8 shows time history of the bending moment for same pier, for same soil conditions, but for long period motion (Kocaeli). Time histories of bending moments are quite different for both types of soil conditions (SSS and CCC) and for two earthquake motions. For example, it can be seen from Figure 7 that short period motion earthquake, in stiff soil (SSS) produces (much) larger plastic deformation, which can be observed by noting flat plateaus on moment – time diagrams, representing plastic hinge development. Those plastic hinge development regions are developing symmetrically, meaning that both sides of the pier have yielded and full plastic hinge has formed. On the other hand, the short period earthquake in soft soil (CCC) produces very little damage, one side of a plastic hinge is (might be) forming between 14 and 15 seconds.

Contrasting those observation is time history of bending moments in Figure 8, where, for a long period motion, stiff soil (SSS) induces small amount of plastic yielding (hinges) on top of piers. However, soft soil (CCC) induces a (very) large plastic deformations. Development of plastic hinges for a structure in soft soil also last very long (more than two seconds, see lower plateau for CCC case in Figure 8) resulting in



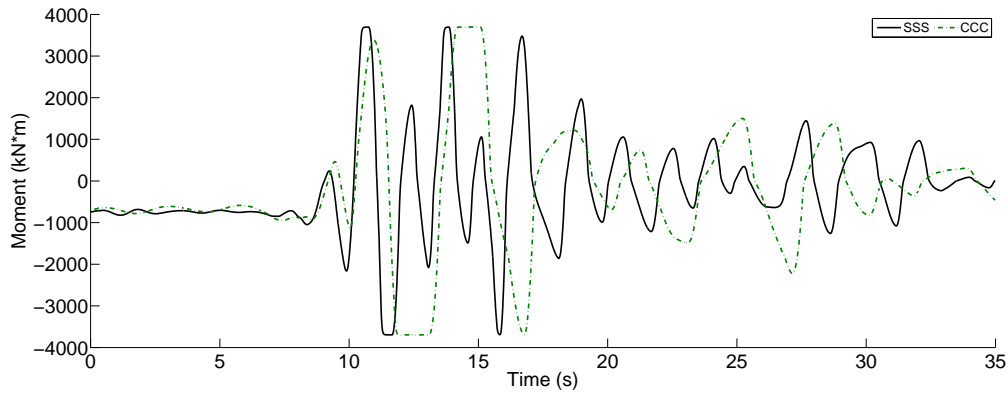


Figure 8: Simulated bending moment time series (top of left pier) for long period motion (Kocaeli), for all clay (CCC) and all sand (SSS) soils.

significant damage development in thus formed plastic hinges. Observed behavior also somewhat contradicts common assumption that soft soils are much more detrimental to structural behavior. It is actually the interaction of the dynamic characteristic of earthquake, soil and structure (ESS) that seem to control the ultimate structural response and the potential damage that might develop.

## 5 SUMMARY

In this paper, an algorithm, named the Plastic Domain Decomposition (PDD), for parallel elastic–plastic finite element computations was presented. Presented was also a parallel scalability study, that shows how PDD scales quite well with increase in a number of compute nodes. More importantly, presented details of PDD reveal that scalability is assured for inhomogeneous, multiple generation parallel computer architecture, which represents majority of currently available parallel computers.

Presented also was an application of PDD to soil–foundation–structure interaction problem for a bridge system and Earthquake–Soil–Structure (ESS) interaction effects were emphasized. The importance of the (miss–) matching of the ESS characteristics to the dynamic behavior of the bridge soil–structure system was shown on an example using same structure, two different earthquakes (one with short and one with long predominant periods) and two different subgrade soils (stiff sand and soft clay)

The main goal of this paper is to show that high fidelity, detailed modeling and simulations of geotechnical and structural systems are available and quite effective. Results from such high fidelity modeling and simulation shed light on new types of behavior that cannot be discovered using simplified models. These high fidelity models tend to reduce modeling uncertainty, which (might) allow practicing engineers to use simulations tools for effective design of soil–structure systems.

## ACKNOWLEDGMENT

The work presented in this paper was supported in part by a grant from the Civil and Mechanical System program, Directorate of Engineering of the National Science Foundation, under Award NSF–CMS–0337811 (cognizant program director Dr. Steve McCabe)

## References

- [1] G. Agha. Actors: A Model of Concurrent Computation in Distributed Systems. MIT Press, 1984.
- [2] P.J. Armstrong and C.O. Frederick. A mathematical representation of the multiaxial bauschinger effect. Technical Report RD/B/N/ 731., C.E.G.B., 1966.
- [3] Satish Balay, Kris Buschelman, Victor Eijkhout, William D. Gropp, Dinesh Kaushik, Matthew G. Knepley, Lois Curfman McInnes, Barry F. Smith, and Hong Zhang. PETSc users manual. Technical Report ANL-95/11 - Revision 2.1.5, Argonne National Laboratory, 2004.
- [4] Satish Balay, Kris Buschelman, William D. Gropp, Dinesh Kaushik, Matthew G. Knepley, Lois Curfman McInnes, Barry F. Smith, and Hong Zhang. PETSc Web page, 2001. <http://www.mcs.anl.gov/petsc>.
- [5] Satish Balay, William D. Gropp, Lois Curfman McInnes, and Barry F. Smith. Efficient management of parallelism in object oriented numerical software libraries. In E. Arge, A. M. Bruaset, and H. P. Langtangen, editors, Modern Software Tools in Scientific Computing, pages 163–202. Birkhäuser Press, 1997.
- [6] Jacobo Bielak, Kostas Loukakis, Youshaiki Hisada, and Chaiki Yoshimura. Domain reduction method for three-dimensional earthquake modeling in localized regions. part I: Theory. Bulletin of the Seismological Society of America, 93(2):817–824, 2003.
- [7] Ma Chi Chen and J. Penzien. Nonlinear soil–structure interaction of skew highway bridges. Technical Report UCB/EERC-77/24, Earthquake Engineering Research Center, University of California, Berkeley, August 1977.
- [8] James W. Demmel, Stanley C. Eisenstat, John R. Gilbert, Xiaoye S. Li, and Joseph W. H. Liu. A supernodal approach to sparse partial pivoting. SIAM J. Matrix Analysis and Applications, 20(3):720–755, 1999.
- [9] B. Dendrou, S. D. Werner, and T. Toridis. Three dimensional response of a concrete bridge system to traveling seismic waves. Computers and Structures, 20:593–603, 1985.
- [10] M. Dryden. Validation of simulations of a two-span reinforced concrete bridge. Submitted in partial satisfaction of the requirements for degree of Ph.D., 2006.
- [11] Charbel Farhat. Multiprocessors in Computational Mechanics. PhD thesis, University of California, Berkeley, 1987.
- [12] Charbel Farhat. Saddle-point principle domain decomposition method for the solution of solid mechanics problems. In Fifth International Symposium on Domain Decomposition Methods for Partial Differential Equations, Norfolk, VA, USA, May 6-8 1991.
- [13] Charbel Farhat and M. Geradin. Using a reduced number of lagrange multipliers for assembling parallel incomplete field finite element approximations. Computer Methods in Applied Mechanics and Engineering, 97(3):333–354, June 1992.
- [14] Charbel Farhat and Francois-Xavier Roux. Method of finite element tearing and interconnecting and its parallel solution algorithm. International Journal for Numerical Methods in Engineering, 32(6):1205–1227, October 1991.
- [15] Gregory Fenves and Mathew Dryden. Nees sfsi demonstration project. NEES project meeting, TX, Austin, August 2005.
- [16] R. E. Fulton and P. S. Su. Parallel substructure approach for massively parallel computers. Computers in Engineering, 2:75–82, 1992.

- [17] J. F. Hajjar and J. F. Abel. Parallel processing for transient nonlinear structural dynamics of three-dimensional framed structures using domain decomposition. Computers & Structures, 30(6):1237–1254, 1988.
- [18] Carl Hewitt, Peter Bishop, and Richard Steiger. A Universal Modular ACTOR Formalism for Artificial Intelligence. In Proceedings of the 3rd International Joint Conference on Artificial Intelligence, Stanford, CA, August 1973.
- [19] I. M. Idriss and J. I. Sun. Excerpts from USER’S Manual for SHAKE91: A Computer Program for Conducting Equivalent Linear Seismic Response Analyses of Horizontally Layered Soil Deposits. Center for Geotechnical Modeling Department of Civil & Environmental Engineering University of California Davis, California, 1992.
- [20] Boris Jeremić. Lecture notes on computational geomechanics: Inelastic finite elements for pressure sensitive materials. Technical Report UCD-CompGeoMech-01-2004, University of California, Davis, 2004. available online: <http://sokocalo.engr.ucdavis.edu/~jeremic/CG/LN.pdf>.
- [21] Boris Jeremić, Sashi Kunnath, and Feng Xiong. Influence of soil–structure interaction on seismic response of bridges. International Journal for Engineering Structures, 26(3):391–402, February 2004.
- [22] Boris Jeremić and Stein Sture. Implicit integrations in elasto–plastic geotechnics. International Journal of Mechanics of Cohesive–Frictional Materials, 2:165–183, 1997.
- [23] Boris Jeremić and Stein Sture. Tensor data objects in finite element programming. International Journal for Numerical Methods in Engineering, 41:113–126, 1998.
- [24] Boris Jeremić and Zhaohui Yang. Template elastic–plastic computations in geomechanics. International Journal for Numerical and Analytical Methods in Geomechanics, 26(14):1407–1427, 2002.
- [25] A. J. Kappos, G. D. Manolis, and I. F. Moschonas. Sismic assessment and design of R/C bridges with irregular configuration, including SSI effects. International Journal of Engineering Structures, 24:1337–1348, 2002.
- [26] George Karypis, Kirk Schloegel, and Vipin Kumar. ParMETIS: Parallel Graph Partitioning and Sparse Matrix Ordering Library. University of Minnesota, 1998.
- [27] O. Klaas, M. Kreienmeyer, and E. Stein. Elastoplastic finite element analysis on a MIMD parallel–computer. Engineering Computations, 11:347–355, 1994.
- [28] Asli Kurtulus, Jung Jae Lee, and Kenneth H. Stokoe. Summary report — site characterization of capital aggregates test site. Technical report, Department of Civil Engineering, University of Texas at Austin, 2005.
- [29] D.E. Lehman and J.P. Moehle. Seismic performance of well-confined concrete bridge columns. Technical report, Pacific Earthquake Engineering Research Center, 1998.
- [30] Jing Li and Olof B. Widlund. On the use of inexact subdomain solvers for BDDC algorithms . Computer Methods in Applied Mechanics and Engineering, 196(8):1415–1428, January 2007.
- [31] N. Makris, D. Badoni, E. Delis, and G. Gazetas. Prediction of observed bridge response with soil–pile–structure interaction. ASCE Journal of Structural Engineering, 120(10):2992–3011, October 1994.
- [32] N. Makris, D. Badoni, E. Delis, and G. Gazetas. Prediction of observed bridge response with soil-pile-structure interaction. ASCE J. of Structural Engineering, 120(10):2992–3011, 1995.
- [33] S. Mazzoni, G.L. Fenves, and J.B. Smith. Effects of local deformations on lateral response of bridge frames. Technical report, University of California, Berkeley, April 2004.
- [34] D. B. McCallen and K. M. Romstadt. Analysis of a skewed short span, box girder overpass. Earthquake Spectra, 10(4):729–755, 1994.
- [35] Francis Thomas McKenna. Object Oriented Finite Element Programming: Framework for Analysis, Algorithms and Parallel Computing. PhD thesis, University of California, Berkeley, 1997.
- [36] A. K. Noor, A. Kamel, and R. E. Fulton. Substructuring techniques – status and projections. Computers & Structures, 8:628–632, 1978.
- [37] Luca F. Pavarino. BDDC and FETI-DP preconditioners for spectral element discretizations. Computer Methods in Applied Mechanics and Engineering, 196(8):1380–1388, January 2007.

- [38] Daniel Rixena and Frédéric Magoulès. Domain decomposition methods: Recent advances and new challenges in engineering. Computer Methods in Applied Mechanics and Engineering, 196(8):1345–1346, January 2007.
- [39] Kirk Schloegel, George Kaypis, and Vipin Kumar. Graph partitioning for high performance scientific simulations. Technical report, Army HPC Research Center, Department of Computer Science and Engineering, University of Minnesota, 1999.
- [40] H. B. Seed, R. T. Wong, I. M. Idriss, and K. Tokimatsu. Moduli and damping factros for dynamic analysis of cohesionless soils. Earthquake Engineering Research Center Report UCB/EERC 84/14, University of California Berkeley, 1984.
- [41] E. Spacone, F.C. Filippou, and F.F. Taucer. Fibre beam-column model for non-linear analysis of r/c frames 1. formulation. Earthquake Engineering and Structural Dynamics, 25:711–725, 1996.
- [42] O. O. Storaasli and P. Bergan. Nonlinear substructuring method for concurrent processing computers. AIAA Journal, 25(6):871–876, 1987.
- [43] J. Sweet. A technique for nonlinear soil–structure interaction. Technical Report CAI-093-100, Caltrans, Sacramento, California, 1993.
- [44] S. Utku, R. Melosh, M. Islam, and M. Salama. On nonlinear finite element analysis in single– multi– and parallel processors. Computers & Structures, 15(1):39–47, 1982.
- [45] Chaiki Yoshimura, Jacobo Bielak, and Yohaiki Hisada. Domain reduction method for three–dimensional earthquake modeling in localized regions. part II: Verification and examples. Bulletin of the Seismological Society of America, 93(2):825–840, 2003.