A Fast 3D Finite-element Solver for Large-scale Seismic Soil Liquefaction Analysis

Ryota Kusakabe[1], Kohei Fujita[1], Tsuyoshi Ichimura[1], Muneo Hori[2], and Lalith Wijerathne[1]

[1] The University of Tokyo, Bunkyo-ku, Tokyo, Japan
{ryota-k, fujita, ichimura, lalith}@eri.u-tokyo.ac.jp
[2] Japan Agency for Marine-Earth Science and Technology, Yokosuka-shi, Kanagawa, Japan
horimune@jamstec.go.jp

Abstract. The accumulation of spatial data and development of computer architectures and computational techniques raise expectations for large-scale soil liquefaction simulations using highly detailed three-dimensional (3D) soil-structure models; however, the associated large computational cost remains the major obstacle to realizing this in practice. In this study, we increased the speed of large-scale 3D soil liquefaction simulation on computers with many-core wide SIMD architectures. A previous study overcame the large computational cost by expanding a method for large-scale seismic response analysis for application in soil liquefaction analysis; however, that algorithm did not assume the heterogeneity of the soil liquefaction problem, resulting in a load imbalance among CPU cores in parallel computations and limiting performance. Here we proposed a load-balancing method suitable for soil liquefaction analysis. We developed an efficient algorithm that considers the physical characteristics of soil liquefaction phenomena in order to increase the speed of solving the target linear system. The proposed method achieved a 26-fold increase in speed over the previous study. Soil liquefaction simulations were performed using large-scale 3D models with up to 3.5 billion degrees-of-freedom on an Intel Xeon Phi (Knights Landing)-based supercomputer system (Oakforest-PACS).

Keywords: soil liquefaction · fast and scalable solver · large-scale analysis · finite-element method.

1 Introduction

Soil liquefaction induced by earthquakes has caused various kinds of damage, including ground settlement, lateral flow, and tilting and destruction of buildings. When the Great East Japan Earthquake occurred in 2011, soil liquefaction was induced in a large area ranging from Kanto to Tohoku and resulted in sand boiling and the uplift of manholes among other damage [11, 13]. To reduce this type of damage, various methods for soil liquefaction analyses have been developed. The recent accumulation of spatial data and development of

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22741-8_25

computer architectures and computational techniques have enabled larger-scale physical simulations, thereby raising the expectation for realization of large-scale soil liquefaction simulations using highly detailed three-dimensional (3D) soil-structure models. Such simulations will contribute to the mitigation of damage by soil liquefaction, as well as soil liquefaction itself. However, these numerical analyses using large-scale 3D models require huge computational requirements, with most soil liquefaction analyses performed either on two-dimensional (2D) models under plane-strain conditions or on small-scale 3D models. The aim of this study was to increase the speed of large-scale 3D soil liquefaction simulation on computers with many-core wide SIMD architectures, which represent a type of architecture used for the recent development of exascale supercomputers.

For seismic wave-propagation simulations in urban areas not limited to soil liquefaction analyses, parallel 3D finite-element solvers suitable for massively parallel computers have been developed and used to analyze the actual damage. A finalist for the Gordon Bell Prize in SC14 [5] (hereafter, referred to as the SC14 solver) enabled a large-scale earthquake simulation by developing a fast algorithm for solving the huge linear system obtained by discretizing the target physical problem, with the solver algorithm designed based on analyzing the characteristics of the discretized linear system of equations. Such an algorithm can be effective for seismic soil liquefaction simulations. Our previous study [9] expanded the algorithm used for the SC14 solver for use in soil liquefaction simulation to enable large-scale 3D soil liquefaction analysis; however, although the target problem of the SC14 solver was relatively homogeneous, soil liquefaction analysis is a heterogeneous problem, where different physical problems are solved in different soil layers [i.e., non-liquefiable (linear) layers and liquefiable (nonlinear) layers]. Therefore, our previous solver [9], which used the algorithm used for the SC14 solver, displayed low parallel efficiency at each parallelization level of the SIMD lane, thread, and process when executed on a computer with many-core wide SIMD architecture. In the present study, we developed methods to improve the load balance in each parallelization level to overcome this inefficiency and to potentially increase the speed of soil liquefaction analysis and enable larger-scale simulations.

In addition to the development of algorithms appropriate for the discretized linear system, our research group proposed algorithms to reduce the computational cost by considering the characteristics of the target physical problem underlying the linear system (a finalist for the Gordon Bell Prize in SC18 [6]; hereafter referred to as the SC18 solver). In this study, we applied this idea to soil liquefaction analysis. Considering the locality of soil liquefaction phenomena, we partially approximated the heavy computation in the nonlinear layer by using light-weight computation, which is expected to reduce analysis cost and time.

We compared the developed solver with our previous solver [9] and performed large-scale soil liquefaction simulations using 3D ground models with up to 3.5 billion degrees-of-freedom (DOF) on an Intel Xeon Phi (Knights Landing)-based supercomputer system (Oakforest-PACS [8]; OFP).

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22741-8_25

2 Target Problem

Eq. (1) is the target linear system of soil liquefaction analysis under undrained conditions. This equation is discretized spatially using the finite-element method and temporally using the Newmark-β method ($\beta = 0.25$).

$$\boldsymbol{A}\delta\boldsymbol{u}^{(n)} = \boldsymbol{b}, \tag{1}$$

where

$$\boldsymbol{A} = \frac{4}{dt^2}\boldsymbol{M} + \frac{2}{dt}\boldsymbol{C}^{(n-1)} + \boldsymbol{K}^{(n-1)},$$
$$\boldsymbol{b} = \boldsymbol{f}^{(n)} - \boldsymbol{q}^{(n-1)} + \boldsymbol{C}^{(n-1)}\boldsymbol{v}^{(n-1)} + \boldsymbol{M}\left(\boldsymbol{a}^{(n-1)} + \frac{4}{dt}\boldsymbol{v}^{(n-1)}\right).$$

Here, $\boldsymbol{M}$, $\boldsymbol{C}$, and $\boldsymbol{K}$ are respectively the consistent mass, Rayleigh damping, and stiffness matrices, $\boldsymbol{f}$, $\boldsymbol{q}$, $\delta\boldsymbol{u}$, $\boldsymbol{v}$, and $\boldsymbol{a}$ are the external force, inner force, displacement increment, velocity, and acceleration vectors, respectively, dt is the time increment, and $*^{(n)}$ is the variable $*$ at the n-th time step. To carry out soil liquefaction analysis, we perform the following at each time step.

1. Read the external force $\boldsymbol{f}$, and calculate the coefficient matrix $\boldsymbol{A}$, and right-hand side vector $\boldsymbol{b}$.
2. Solve the linear system (1) and obtain the displacement increment $\delta\boldsymbol{u}$.
3. Update the displacement $\boldsymbol{u}$, velocity $\boldsymbol{v}$, and acceleration $\boldsymbol{a}$, using $\delta\boldsymbol{u}$.
4. Update the stiffness $\boldsymbol{K}$, and inner force $\boldsymbol{q}$ [Eq. (2) and (3)].

$$\boldsymbol{K}^{(n)} = \sum_e \int_{V_e} \boldsymbol{B}^{\mathrm{T}} \left(\boldsymbol{D}_{\mathrm{s}}^{(n)} + \boldsymbol{D}_{\mathrm{f}}\right) \boldsymbol{B}\ \mathrm{d}V, \tag{2}$$

$$\boldsymbol{q}^{(n)} = \sum_e \int_{V_e} \boldsymbol{B}^{\mathrm{T}} \left(\boldsymbol{\sigma}'^{(n)} - p^{(n)}\boldsymbol{m}\right)\ \mathrm{d}V, \tag{3}$$

where, $\boldsymbol{B}$ is the matrix used to convert the displacement into the strain, $\boldsymbol{D}_{\mathrm{s}}$ and $\boldsymbol{D}_{\mathrm{f}}$ are the elasto-plastic matrices of the soil and pore water, respectively, $\boldsymbol{\sigma}'$ is the effective stress of the soil, p is the excess pore water pressure and $\boldsymbol{m} = \{1\ 1\ 1\ 0\ 0\ 0\}^{\mathrm{T}}$. $\int_e *\mathrm{d}V$ is the volume integration of the variable $*$ in the e-th finite element. $\boldsymbol{D}_{\mathrm{s}}$, $\boldsymbol{D}_{\mathrm{f}}$, $\boldsymbol{\sigma}'$, and p are defined by the constitutive law[3]. For detailed information, consult [3] and a 2D constitutive law [4] which [3] is based on.

3 Previous Method

Solving the linear system (1) and updating the stiffness $\boldsymbol{K}$, and inner force $\boldsymbol{q}$ occupy a considerable amount of the calculation time in liquefaction analysis. Solving the linear system (1) requires a huge computational cost, especially in large-scale 3D analysis, which has been the major obstacle to realizing large-scale

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22741-8_25

3D soil liquefaction analysis. Our previous study [9] overcame the computational cost by applying the algorithm of the SC14 solver for large-scale seismic response analysis to large-scale soil liquefaction analysis. In this section, we explain the algorithm associated with the SC14 solver and the problems in our previous study [9] about implementing the algorithm of the SC14 solver to soil liquefaction analysis.

3.1 Finite-element Solver

The SC14 solver is a fast and scalable method that uses unstructured tetrahedral second-order elements developed for solving the target linear system involved in seismic response analysis without soil liquefaction. It is based on the adaptive conjugate gradient (CG) method [2] and uses hybrid parallelization with message passing interface (MPI) and OpenMP.

The conventional preconditioned CG (PCG) method improves the convergence characteristics by preconditioning $\boldsymbol{z} = \boldsymbol{M}^{-1}\boldsymbol{r}$, where $\boldsymbol{M}$ is a preconditioning matrix that mimics the coefficient matrix $\boldsymbol{A}$. Preconditioning in the adaptive CG method [2] is to *roughly* solve $\boldsymbol{Az} = \boldsymbol{r}$ instead of $\boldsymbol{z} = \boldsymbol{M}^{-1}\boldsymbol{r}$. The preconditioning matrix $\boldsymbol{M}$ in this method is supposed to be $\boldsymbol{M} \approx \boldsymbol{A}$, which describes it as having high preconditioning performance. The PCG method with the 3×3 block Jacobi preconditioner is used to solve $\boldsymbol{Az} = \boldsymbol{r}$ as the preconditioning in order to achieve parallel efficiency.

Increases in speed are limited when merely solving $\boldsymbol{Az} = \boldsymbol{r}$ during preconditioning. The SC14 solver reduces the computational cost by using a first-order tetrahedral-element model generated as the geometric multigrid of the second-order tetrahedral-element model of the target problem. First, $\boldsymbol{A}_{\mathrm{c}}\boldsymbol{z}_{\mathrm{c}} = \boldsymbol{r}_{\mathrm{c}}$ is solved on the first-order element model, which requires much less computational cost than on the second-order element model. Using the solution of the first-order element model as the initial solution, $\boldsymbol{Az} = \boldsymbol{r}$ is then solved on the second-order element model, thereby significantly reducing the amount of computation and communication. The PCG method is used twice in preconditioning on the first- and second-order element models. Hereafter, the former CG is referred to as an inner coarse CG, and the latter is referred to as an inner fine CG. The main CG that uses the two inner CGs for preconditioning is called the outer CG. To further reduce the computational cost, the preconditioning step, which does not require high computational accuracy, is computed in single precision, whereas the other calculation is performed in double precision.

The SC14 solver uses the element-by-element (EBE) method [12] to efficiently compute matrix-vector multiplication, which is the heaviest computation in the CG method. In conventional matrix-vector multiplication calculations, the global coefficient matrix, $\boldsymbol{A} = \sum_e \boldsymbol{A}_e$, is calculated, followed by multiplication of the matrix $\boldsymbol{A}$ by the vector $\boldsymbol{x}$ to obtain the matrix-vector multiplication $\boldsymbol{Ax}$. In the EBE method, the global matrix $\boldsymbol{A}$ is not calculated. The element-wise matrix-vector multiplications $\boldsymbol{A}_e\boldsymbol{x}_e$ are calculated on the fly and summed to obtain the global matrix-vector multiplication $\boldsymbol{Ax} = \sum_e(\boldsymbol{A}_e\boldsymbol{x}_e)$, where $\boldsymbol{x}$ is the component of $\boldsymbol{x}$ for the e-th element. This method reduces the cost to read

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22741-8_25

and write the global coefficient matrix, as well as the memory usage necessary to store it. Therefore matrix-vector multiplication using the EBE method requires less memory access and communication, which shortens computation time.

3.2 Implementation of Our Previous Solver and Its Problems

Our previous study [9] overcame the huge computational cost involved with solving the target linear system (1) in order to allow large-scale soil liquefaction analysis using the method of the SC14 solver, which is explained in the previous section. However, the heterogeneity of soil liquefaction analysis causes load imbalance in parallel computation, which results in low parallel efficiency and long computation times. This is because the target problem for the SC14 solver is seismic response analysis, which is relatively homogeneous; therefore, the algorithm does not assume heterogeneity. The load imbalance occurs during calculation of matrix-vector multiplication using the EBE method and update of the stress and elasto-plastic matrix according to the constitutive law. These two computations have the highest computational costs; therefore, the resulting load imbalance potentially results in a worse time to solution.

The difference in the number of integration points in different elements causes the load imbalance in the calculation of matrix-vector multiplication. When calculating the element-wise coefficient matrix $\boldsymbol{A}_e$ during matrix-vector multiplication using the EBE method, the element-wise stiffness matrix $\boldsymbol{K}_e = \int_{V_e} \boldsymbol{B}^{\mathrm{T}}(\boldsymbol{D}_{\mathrm{s}} + \boldsymbol{D}_{\mathrm{f}})\boldsymbol{B}\mathrm{d}V$, is calculated on the fly. $\boldsymbol{B}$ is a linear function, and the elasto-plastic matrix in the linear layer is constant, suggesting that the integrand in a linear element is second order, and computing the volume integration needs only four integration points. On the other hand, the elasto-plastic matrix in the nonlinear layer spatially varies. Because soil liquefaction occurs locally, the elasto-plastic matrix should be assumed to vary in an element to accurately analyze the phenomenon. Therefore, the integrand in a nonlinear element is more than third order, and more than four integration points are needed. In this study, five integration points were used, enabling calculation of the integration of up to a third-order function.

The difference in the constitutive law at different layers causes load imbalance in the calculation of the effective stress $\boldsymbol{\sigma}'$, and the elasto-plastic matrix $\boldsymbol{D}_{\mathrm{s}}$, of soil. In the linear layer, $\boldsymbol{D}_{\mathrm{s}}$ is constant and unnecessary to update, and $\boldsymbol{\sigma}'$ is calculated with very low computational cost, as $\boldsymbol{\sigma}' = \boldsymbol{D}_{\mathrm{s}}\boldsymbol{\varepsilon}$. Therefore, it is not explicitly calculated and, thus, computed on the fly when $\boldsymbol{q}$ is computed using Eq. (3). In the nonlinear elements, $\boldsymbol{\sigma}'$ and $\boldsymbol{D}_{\mathrm{s}}$ are updated at every time step in the five integration points per element. This requires considerable computation, because calculations for 300 one-dimensional springs are required at each integration point for the implementation in the present study.

The SC14 solver achieves load balance by distributing the same number of elements to each CPU core and by assuming that the computation amount in each element is highly similar. However, for soil liquefaction analysis, the computation amount in each element differs; therefore, load imbalance among processes and threads occurred in our previous solver [9], which used the original version

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22741-8_25

As is

```
do ie = 1, ne
    if(ie-th element is linear element)
        [calc. for a linear element]
    else #nonlinear
        [calc. for a nonlinear element]
    endif
end do
```

Reordering

```
[Reordering of the elements]
do ie = 1, ne_linear
    [calc. for a linear element]
enddo
do ie = ne_linear+1, ne
    [calc. for a nonlinear element]
end do
```

Fig. 1: Element reordering to improve thread load balance and exploit automatic vectorization by the compiler.

of the algorithm of the SC14 solver. Another problem in speeding up on our target architecture of many-core wide SIMD is that it is diffcult to exploit a wide SIMD in heterogeneous calculations.

4 Developed Method

In this section, we propose methods for improving the load balance among SIMD lanes, among OpenMP threads, and among MPI processes in order to overcome the problem of load imbalance experienced in our previous study [9]. Additionally, we developed a physics-aware algorithm to solve the target linear system (1) in order to achieve a further increase in speed.

4.1 Improving Load Balance

First, we reordered the elements to improve load balance among OpenMP threads in each process and to exploit efficient use of wide SIMD arithmetic units. In a conventional implementation, variables for both linear and nonlinear elements are stored in arrays in mixed order. We implemented element reordering to enable storing variables for linear elements in the first part of arrays and variables for nonlinear elements in the remainder of the arrays. In this implementation, the computation for the linear and nonlinear elements is separated (Fig. 1) and OpenMP parallelized independently, thereby improving thread load balance. Additionally, separating the computation of the linear and nonlinear elements is suitable for automatic vectorization by the compiler, which will increase the speed of simulation, especially on our target computers of manycore architecture with wide SIMD.

Second, we revised the model-partitioning method in order to improve load balance among MPI processes. The conventional method partitions the model so that the number of elements assigned to each process is almost the same. This causes process load imbalance because the ratio of the number of linear and nonlinear elements could be different. We propose that the linear and nonlinear layers be independently partitioned into the number of MPI processes and that each process be assigned to a domain comprising one partition of the linear layer

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22741-8_25

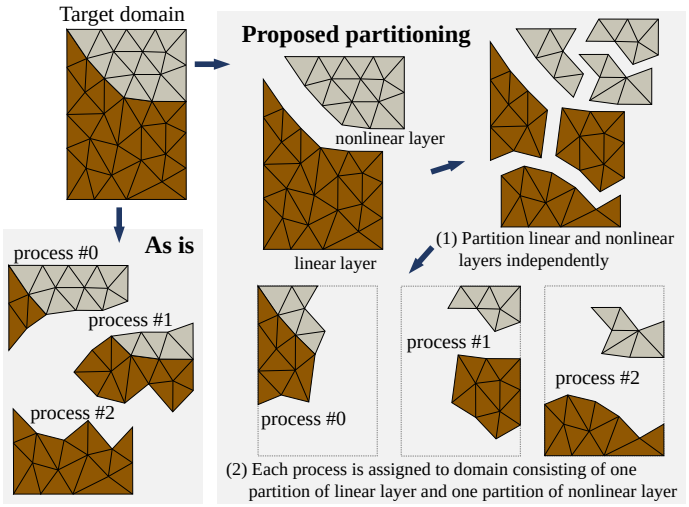


Fig. 2: Overview of the proposed partitioning of the simulation domain.

and one partition of the nonlinear layer (Fig. 2). This method improves load balance among the processes because each process works on the same number of linear and nonlinear elements.

4.2 Physics-aware Preconditioner

We developed a preconditioner that considers the physical characteristics of soil liquefaction based on the preconditioner of the SC14 solver in order to achieve additional increases in speed. The SC18 solver was developed based on the SC14 solver and used the preconditioner based on the physical characteristics of the target problem. The target problem of the SC18 solver includes a domain involving a high degree of stiffness, such as concrete structures. The convergence characteristics of the CG method are extremely poor around these structures. Another inner CG was added between the inner coarse and fine CGs for preconditioning. The only domain having poor convergence characteristics, which is detected by an AI before the analysis, is solved in the additional inner CG, which efficiently improved the convergence characteristics. The physics-aware preconditioner introduced in the present study uses the same concept involving the addition of another inner CG (called the inner middle CG) between the inner coarse and fine CGs. Note that the inner middle CG in this study exploits the locality of soil liquefaction phenomena and differs from the additional inner CG in the SC18 solver.

The target problem of the SC14 solver has a relatively small spatial variation in soil-physical properties. The soil elasto-plastic matrix $\boldsymbol{D}_s$, is assumed to be constant not only in a linear element but also in a nonlinear element. On the other hand, for soil liquefaction analysis implemented in this study, the

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22741-8_25

volume integration in nonlinear elements is calculated using $\boldsymbol{D}_{\mathrm{s}}$ at the five integration points, which increases memory access during computation of matrix-vector multiplication via the EBE method. We added the inner middle CG to reduce the data necessary for preconditioning. The computation in the inner middle CG is similar to that in the inner fine CG, except that it uses a matrix $\boldsymbol{A}_{\mathrm{m}}$, instead of the coefficient matrix $\boldsymbol{A}$. Here, $\boldsymbol{A}_{\mathrm{m}} = \frac{4}{dt^2}\boldsymbol{M} + \frac{2}{dt}\boldsymbol{C}_{\mathrm{m}} + \boldsymbol{K}_{\mathrm{m}}$, $\boldsymbol{K}_{\mathrm{m}} = \sum_e \int_{V_e} \boldsymbol{B}^{\mathrm{T}} \boldsymbol{D}_{\mathrm{m}} \boldsymbol{B} \, \mathrm{d}V$, and $\boldsymbol{D}_{\mathrm{m}}$ is constant in an element:

$$\boldsymbol{D}_{\mathrm{m}} = \begin{bmatrix} d_1 & d_2 & d_2 & & & \\ d_2 & d_1 & d_2 & & 0 & \\ d_2 & d_2 & d_1 & & & \\ & & & d_3 & & \\ & 0 & & & d_4 & \\ & & & & & d_5 \end{bmatrix}. \tag{4}$$

d_i $(i = 1, \cdots, 5)$ are calculated as the *average* of the $\boldsymbol{D} = \boldsymbol{D}_{\mathrm{s}} + \boldsymbol{D}_{\mathrm{f}}$ at the five integration points[3]:

$$d_1 = \frac{1}{15}\sum_{j=1}^{5}\left(D^j_{11} + D^j_{22} + D^j_{33}\right), \quad d_2 = \frac{1}{15}\sum_{j=1}^{5}\left(D^j_{12} + D^j_{23} + D^j_{31}\right),$$
$$d_3 = \frac{1}{5}\sum_{j=1}^{5} D^j_{44}, \quad d_4 = \frac{1}{5}\sum_{j=1}^{5} D^j_{55}, \quad d_5 = \frac{1}{5}\sum_{j=1}^{5} D^j_{66}, \tag{5}$$

where $\boldsymbol{D}^j$ is the elasto-plastic matrix at the j-th integration point. This reduces the memory access necessary during the computation. Additionally, the integration in the calculation of $\boldsymbol{K}_{\mathrm{m}}$ uses only four integration points, because $\boldsymbol{D}_{\mathrm{m}}$ is constant, and requires a smaller amount of computation than that in the inner fine CG, which uses five integration points.

Because soil liquefaction occurs locally, there are only a few elements in which the physical property varies greatly and in the most elements, $\boldsymbol{D}^j \approx \boldsymbol{D}_{\mathrm{m}}$ $(j = 1, \cdots, 5)$. Therefore, the computation using the inner fine CG can be partially replaced by the computation using the inner middle CG, which has a lower computational cost.

[3] There are several ways to average $\boldsymbol{D}$, with the optimal method depending upon the problem. Non-zero components of $\boldsymbol{D}_{\mathrm{f}}$ are only present in the upper-left 3×3 components and correspond to the bulk modulus of pore water, which is constant and 10-fold greater than that of soil. Therefore, the upper-left 3×3 components of $\boldsymbol{D}$ are only slightly influenced by soil heterogeneity and are supposed to be approximated using two parameters: d_1 and d_2. On the other hand, a shear wave has a dominant influence on seismic response analysis, and the soil is supposed to show its heterogeneity, such that only shear stiffness in the vibration direction is reduced. Therefore, the lower-right diagonal three components, which correspond to the shear stiffness in different directions are approximated using different parameters: d_3, d_4, and d_5. The zero components of $\boldsymbol{D}_{\mathrm{m}}$ could be non-zero in $\boldsymbol{D}$ but are sufficiently small as compared with d_i $(i = 1, \cdots, 5)$. Approximating these small components by zero efficiently reduces the amount of data in $\boldsymbol{D}_{\mathrm{m}}$.

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22741-8_25

algorithm A: The outline of the proposed method. $\bar{*}$ indicates that the variable $*$ is in single precision. $*_{\mathrm{c}}$ indicates that the variable $*$ is associated with the first-order model. $\bar{\boldsymbol{P}}$ is the mapping matrix from a variable for the first-order model to the equivalent variable for the second-order model. nt is the number of time steps.

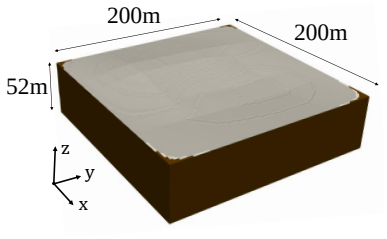
```
1  [Model partitioning]
2  [Element reordering]
3  Calculate the initial values of u, q, K by self-weight analysis.
4  v, a ⇐ 0
5  do it = 1, nt
6     Read the external force f.
7     b ⇐ f − q + Cv + M (a + (4/dt) v)
8     A ⇐ (4/dt^2) M + (2/dt) C + K
9     Set the initial values of δu and other variables for the CG method.
10    r ⇐ b − Aδu
11    (Outer CG starts: solve Aδu = b)
12    while not converged do
13       (Preconditioning starts)
14       r̄ ⇐ r
15       r̄_c ⇐ P̄^T r̄
16       solve Ā_c z̄_c = r̄_c    by PCG method (inner coarse CG)
                                 on first-order-element mesh.
17       z̄_0 ⇐ P̄ z̄_c
18       solve Ā_m z̄ = r̄        by PCG method (inner middle CG)
                                 with initial solution z̄_0,
                                 on second-order-element mesh.
19       z̄_0 ⇐ z̄
20       solve Ā z̄ = r̄          by PCG method (inner fine CG)
                                 with initial solution z̄_0,
                                 on second-order-element mesh.
21       z ⇐ z̄
22       (Preconditioning ends)
23       Update δu and r(= b − Aδu), using z
24    end while
25    (Outer CG ends: displacement increment δu is obtained.)
26    Update u, v and a by Newmark-β Method, using δu.
27    Update σ' and p by constitutive law, using u and δu
28    Q ⇐ Σ_e ∫_{V_e} B^T (σ'^(n) − pm) dV
29    Update D_s by constitutive law, using u and δu
30    K ⇐ Σ_e ∫_{V_e} B^T (D_s + D_f) B dV
31 end do
```

Algorithm A summarizes the proposed method.

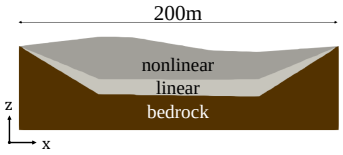
5 Performance Measurement

We performed large-scale soil liquefaction simulations using 3D ground models that mimic actual ground structures in order to demonstrate the effectiveness

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22741-8_25



(a) Model overview



(b) Cross-section view of B

Fig. 3: Model configuration

Table 1: Soil profile properties. ρ: density, V_p, V_s: velocity of primary and secondary wave, h: damping ratio.

	ρ[kg/m^3]	V_p[m/s]	V_s[m/s]	h	constitutive law
Layer1	1500	—	—	5%	nonlinear
Layer2	1800	1380	255	5%	linear
Bedrock	1900	1770	490	0.5%	linear

of the proposed method. The models used in the simulations were generated by replicating the model shown in Fig. 3 in the x and y directions. Table 1 shows the physical properties of the soil. The parameters for soil liquefaction were the same as those used previously [9]. The groundwater level was assumed to be 2-m below the surface of the model. We used the seismic wave observed during the Hyogo-ken Nambu earthquake in 1995 [7], which induced soil liquefaction in large areas, including reclaimed land along the coast, causing significant damage to buildings [1, 10]. The resolution of the models was decided based on a previous study [9], where the maximum element size was 2 m in the nonlinear layer, 20 m in the linear layer, and 40 m in the bedrock, and the time increment was 0.001 s.

All performance measurements were undertaken on Oakforest-PACS [8] (OFP), a computer with many-core wide SIMD architecture. The OFP is a supercomputer introduced by the Joint Center for Advanced High Performance Computing, Japan. It has 8,208 compute nodes, each of which comprises a 68-core Intel Xeon Phi processor 7250 (Knights Landing) CPU, 16GB-MCDRAM memory, and six 16GB-DDR4-2400 memory chips. The network between compute nodes is interconnected by an Intel Omni-Path architecture, and each core has two 512-bit vector units and supports AVX-512 SIMD instructions.

5.1 Time to Solution

First, time to solution was compared with our previous study [9]. Table 2 shows the simulation cases. Asis is the implementation in our previous study [9]. Asis is

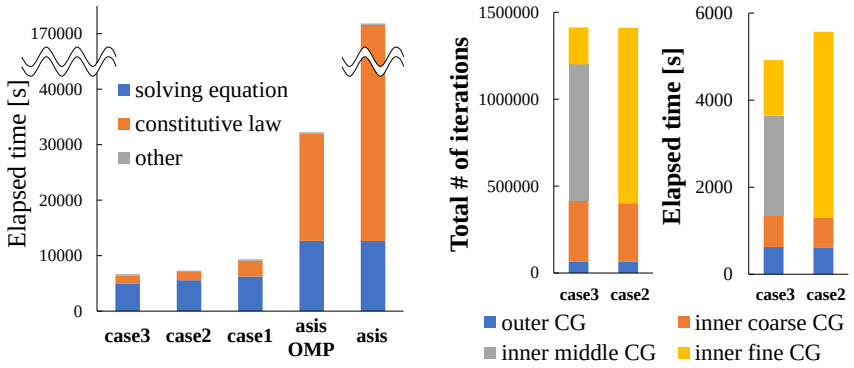
ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22741-8_25

Table 2: Simulation cases: *OMP(constitutive law)* describes OpenMP parallelization in the calculation of the constitutive law; *Thread load balance/SIMD* describes load balancing among OpenMP threads in an MPI process and SIMD vectorization by reordering elements; *Process load balance* describes load balancing among MPI processes by independently partitioning the linear and nonlinear domains; and *Physics-aware preconditioner* describes preconditioning using the inner coarse, middle, and fine CGs. "+" indicates that the feature is implemented, and "−" indicates that it is not implemented.

	asis	asisOMP	case1	case2	case3
OMP(constitutive law)	-	+	+	+	+
Thread load balance/SIMD	-	-	+	+	+
Process load balance	-	-	-	+	+
Physics-aware preconditioner	-	-	-	-	+

an incomplete implementation that does not implement OpenMP parallelization in the calculation of the constitutive law. In the present study, asisOMP, which implements OpenMP parallelization for calculation of the constitutive law, was used to measure the reference performance. Case1 implements element reordering in order to improve the thread load balance and exploit the wide SIMD. Case2 implements revised model partitioning to improve the process load balance in addition to case1 implementation. Case3 implements the physics-aware preconditioner to increase the speed of solving the linear system (1) in addition to case2 implementation. We used a 54,725,427-DOF model generated by replicating the model shown in Fig. 3 in the x and y directions. Soil liquefaction analysis for 7,500 time steps was performed with 512 MPI processes $\times$ eight OpenMP threads per one MPI process = 4,096 CPU cores (64 OFP nodes). Fig. 4a shows the time to solution. Case1 compared with asisOMP achieved a 2.03-fold increase in speed in solving the linear system and a 6.78-fold increase in the speed of calculating the constitutive law. This result showed that thread load balancing and SIMD vectorization successfully reduced the computation time. Specifically, calculation of the constitutive law exploited the wide SIMD based on the SIMD width of the OFP at eight. Case2 compared with case1 achieved a 1.12-fold increase in speed in solving the linear system and a 1.89-fold increase in the speed of calculating the constitutive law, with calculation of the constitutive law showing greater performance improvement, because the difference in computational cost between the linear and nonlinear layer was larger. Case3 compared with case2 achieved a 13% increase in the speed of solving the linear system. Fig. 4b shows the total number of iterations in the outer and inner CGs and the analysis times for case2 and case3. The number of iterations in the inner fine CG in case2 and the summation of the number of iterations in the inner middle and fine CGs in case3 were similar. Case3 reduced computation time by using the inner middle CG accompanied by a lower computational cost. Case3 achieved a 26-fold increase in speed over asis and a 4.85-fold increase in speed

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22741-8_25



(a) Time to solution of the proposed method. Elapsed time to analyze 7,500 time steps.

(b) Number of iterations and the elapsed time in the outer CG and inner CGs. Results represent analyses of 7,500 time steps.

Fig. 4: Measurement of the performance of the proposed method.

Table 3: Weak-scale configuration.

# of OFP nodes	# of MPI procs.	DOF	DOF per process	# of elements
64	512	109,425,213	213,721	26,810,034
128	1,024	218,798,889	213,671	53,620,068
256	2,048	437,546,541	213,646	107,240,212
512	4,096	874,990,053	213,621	214,480,424
1,024	8,192	1,749,877,677	213,608	428,961,000
2,046	16,368	3,499,549,341	213,804	857,922,000

over asisOMP, thereby demonstrating the effectiveness of the proposed method.

5.2 Scalability

We then measured the weak scalabilities of case3 and asisOMP. The model shown in Fig. 3 was replicated in the x and y directions in order to generate models, and Table 3 shows the model configuration. 100 time steps of soil liquefaction analyses were performed using 512 to 16,368 MPI processes while maintaining a constant DOF per MPI process. The number of OpenMP threads per MPI process was eight. In the analysis using 16,368 MPI processes, up to 29.2 GiB per OFP node was used out of 16GB-MCDRAM memory and 96GB-DDR memory. Results are shown in Fig. 5.

For all problem sizes, case3 showed an about 5-fold increase in speed relative to asisOMP. The largest-scale problem used 2,046 of the 8,208 OFP compute nodes, indicating that the proposed solver showed a high level of performance

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22741-8_25

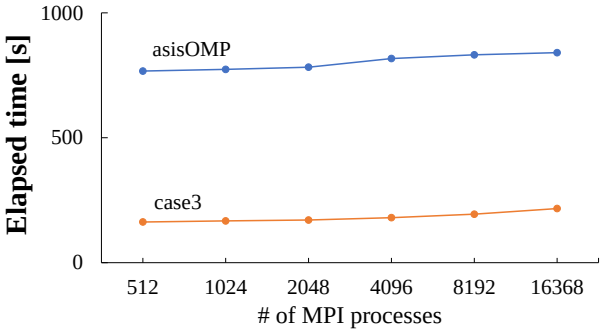


Fig. 5: weak scalability

using 25% of the OFP. The weak scalability of case3 from 512 MPI processes to 16,368 MPI processes was 75 %. This can be considered a high level of performance based on the complexities of the target problem. The finite-element method using unstructured elements is complicated, making it difficult to achieve a high degree of scalability. Soil liquefaction analysis involves additional complexities in regard to load balancing, and the high memory usage for calculating the constitutive law limits the problem scale per MPI process, resulting in a relatively larger percentage of communications.

6 Conclusion

In this study, we developed an efficient algorithm for soil liquefaction analysis based on that reported previously [9]. Additionally, we developed a preconditioner that reduced the amount of computation and memory access by considering the locality of soil liquefaction phenomena. The proposed method achieved a 26-fold increase in speed as compared with that demonstrated previously [9]. We performed soil liquefaction simulations using large-scale 3D models with up to 3.5 billion DOFs, which is 1,000-fold larger than the 3.4-million-DOF model computed in the previous study [9]. These results demonstrated that the proposed method is fast and scalable.

The load-balancing method proposed in this study can be applied not only to soil liquefaction analysis but also to other heterogeneous problems, such as other multi-physics coupled analyses. The physics-aware preconditioner developed in this study achieved increases in speed by considering the physical characteristics of soil liquefaction phenomena, implying that smarter algorithms for solving mathematical problems can be developed by considering the characteristics of the underlying physical phenomenon, which the SC18 solver [6] demonstrated for a different physical problem.

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22741-8_25

References

1. Akai, K.: Geotechnical reconnaissance of the effects of the January 17, 1995, Hyogoken-Nanbu earthquake, Japan. Earthquake Engineering Research Center, University of California at Berkeley (1995)
2. Golub, G.H., Ye, Q.: Inexact preconditioned conjugate gradient method with inner-outer iteration. SIAM Journal on Scientific Computing **21**(4), 1305–1320 (1999), https://doi.org/10.1137/S1064827597323415
3. Iai, S.: Three dimensional formulation and objectivity of a strain splace multiple mechanism model for sand. Soils and Foundations **33**(1), 192–199 (1993), https://doi.org/10.3208/sandf1972.33.192
4. Iai, S., Matsunaga, Y., Kameoka, T.: Strain space plasticity model for cyclic mobility. Soils and Foundations **32**(2), 1–15 (1992), https://doi.org/10.3208/sandf1972.32.2_1
5. Ichimura, T., Fujita, K., Tanaka, S., Hori, M., Lalith, M., Shizawa, Y., Kobayashi, H.: Physics-based urban earthquake simulation enhanced by 10.7 blndof × 30 k time-step unstructured fe non-linear seismic wave simulation. In: SC '14: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis. pp. 15–26 (2014), https://doi.org/10.1109/SC.2014.7
6. Ichimura, T., Fujita, K., Yamaguchi, T., Naruse, A., Wells, J.C., Schulthess, T.C., Straatsma, T.P., Zimmer, C.J., Martinasso, M., Nakajima, K., Hori, M., Maddegedara, L.: A fast scalable implicit solver for nonlinear time-evolution earthquake city problem on low-ordered unstructured finite elements with artificial intelligence and transprecision computing. In: Proceedings of the International Conference for High Performance Computing, Networking, Storage, and Analysis. pp. 49:1–49:11. SC '18 (2018), http://dl.acm.org/citation.cfm?id=3291656.3291722
7. JMA observatory, Japan Meteorological Agency: Strong ground motion of the southern hyogo prefecture earthquake in 1995 observed at kobe. http://www.data.jma.go.jp/svd/eqev/data/kyoshin/jishin/hyogo_nanbu/dat/H1171931.csv ([accessed 3 January 2019])
8. Joint Center for Advanced High Performance Computing: Basic specification of oakforest-pacs. http://jcahpc.jp/files/OFP-basic.pdf
9. Kusakabe, R., Ichimura, T., Fujita, K., Hori, M., Wijerathne, L.: A finite element analysis method for simulating seismic soil liquefaction based on a large-scale 3d soil structure model. submitted to *Soil Dynamics and Earthquake Engineering* (2019)
10. Okimura, T., Takada, S., Koid, T.H.: Outline of the great hanshin earthquake, japan 1995. Natural hazards **14**(1), 39–71 (1996), https://doi.org/10.1007/BF00229911
11. Towhata, I., Maruyama, S., Kasuda, K., Koseki, J., Wakamatsu, K., Kiku, H., Kiyota, T., Yasuda, S., Taguchi, Y., Aoyama, S., Hayashida, T.: Liquefaction in the kanto region during the 2011 off the pacific coast of tohoku earthquake. Soils and Foundations **54**(4), 859–873 (2014), https://doi.org/10.1016/j.sandf.2014.06.016
12. Winget, J.M., Hughes, T.J.R.: Solution algorithms for nonlinear transient heat conduction analysis employing element-by-element iterative strategies. Computer Methods in Applied Mechanics and Engineering **52**(1-3), 711–815 (1985), https://doi.org/10.1016/0045-7825(85)90015-5
13. Yamaguchi, A., Mori, T., Kazama, M., Yoshida, N.: Liquefaction in tohoku district during the 2011 off the pacific coast of tohoku earthquake. Soils and Foundations **52**(5), 811–829 (2012), https://doi.org/10.1016/j.sandf.2012.11.005

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22741-8_25