Fast Scalable Implicit Solver with Convergence of Equation-Based Modeling and Data-Driven Learning: Earthquake City Simulation on Low-Order Unstructured Finite Element

Tsuyoshi Ichimura

Earthquake Research Institute &
Department of Civil Engineering, The
University of Tokyo,
Center for Advanced Intelligence
Project, RIKEN
Tokyo, Japan

Kohei Fujita

Earthquake Research Institute &
Department of Civil Engineering, The
University of Tokyo
Center for Computational Science,
RIKEN
Tokyo, Japan

Kentaro Koyama

Fujitsu Limited
Nagano, Japan

Yuma Kikuchi

Department of Civil Engineering, The
University of Tokyo
Tokyo, Japan

Ryota Kusakabe

Department of Civil Engineering, The
University of Tokyo
Tokyo, Japan

Kazuo Minami

Center for Computational Science,
RIKEN
Kobe, Japan

Hikaru Inoue

Fujitsu Limited
Tokyo, Japan

Seiya Nishizawa

Center for Computational Science,
RIKEN
Kobe, Japan

Miwako Tsuji

Center for Computational Science,
RIKEN
Kobe, Japan

Tatsuo Nishiki

Fujitsu Limited
Tokyo, Japan

Muneo Hori

Research Institute for
Value-Added-Information Generation,
Japan Agency for Marine-Earth
Science and Technology
Yokohama, Japan

Lalith Maddeggedara

Earthquake Research Institute &
Department of Civil Engineering, The
University of Tokyo
Tokyo, Japan

Naonori Ueda

Center for Advanced Intelligence
Project, RIKEN
Tokyo, Japan

ABSTRACT

We developed a new approach in converging equation-based modeling and data-driven learning on high-performance computing resources to accelerate physics-based earthquake simulations. Here, data-driven learning based on data generated while conducting equation-based modeling was used to accelerate the convergence process of an implicit low-order unstructured finite-element solver.

This process involved a suitable combination of data-driven learning for estimating high-frequency components and coarsened equation-based models for estimating low-frequency components of the problem. The developed solver achieved a 12.8-fold speedup over the state-of-art solver with a 96.4% size-up scalability up to 24,576 nodes (98,304 MPI processes $\times$ 12 OpenMP threads = 1,179,648 CPU cores) of Fugaku with 126,581,788,413 degrees-of-freedom, leading to solving a huge city earthquake shaking analysis in a 10.1-fold shorter time than the previous state-of-the-art solver. Furthermore, to show that the developed method attains high performance on variety of systems with small implementation costs, we ported the developed method to recent GPU systems by use of directive based methods (OpenACC). The equation based modeling and the data-driven learning are of utterly different characteristics, and hence they are rarely combined. The developed approach of combining them is effective, and remarkable results mentioned above are achieved.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

PASC '21, July 5–9, 2021, Geneva, Switzerland

© 2021 Association for Computing Machinery.

ACM ISBN 978-1-4503-8563-3/21/07...\$15.00

<https://doi.org/10.1145/3468267.3470616>

CCS CONCEPTS

• **Applied computing** → **Engineering**; • **Computing methodologies** → **Massively parallel algorithms**.

KEYWORDS

Data driven methods, finite-element modeling, scalability, iterative solver

ACM Reference Format:

Tsuyoshi Ichimura, Kohei Fujita, Kentaro Koyama, Yuma Kikuchi, Ryota Kusakabe, Kazuo Minami, Hikaru Inoue, Seiya Nishizawa, Miwako Tsuji, Tatsuo Nishiki, Munee Hori, Lalith Maddeggedara, and Naonori Ueda. 2021. Fast Scalable Implicit Solver with Convergence of Equation-Based Modeling and Data-Driven Learning: Earthquake City Simulation on Low-Order Unstructured Finite Element. In *Platform for Advanced Scientific Computing Conference (PASC '21)*, July 5–9, 2021, Geneva, Switzerland. ACM, New York, NY, USA, 12 pages. <https://doi.org/10.1145/3468267.3470616>

1 INTRODUCTION

Populated cities around the world have been unexpectedly struck by earthquakes causing considerable fatalities and leaving many injured: Lisbon earthquake in 1755 caused over 50,000 fatalities, San Francisco earthquake in 1906 caused over 3,000 fatalities, Messina earthquake in 1908 caused over 100,000 fatalities, Haiti earthquake in 2010 caused over 300,000 fatalities, Sumatra earthquake in 2004 caused over 220,000 fatalities, Sichuan earthquake in 2008 caused over 80,000 fatalities, and Northeastern Japan earthquake in 2011 caused over 20,000 fatalities. As shown in Fig. 1, a chain of phenomena causes earthquake damage: (a) the accumulation of strain in the crust due to its deformation by plate tectonics, (b) the release of strain causing earthquake wave propagation through the crust, and (c) the earthquake wave input to cities causing damage to society. The movement of plates will continue triggering earthquakes and causing damage. Thus, the reduction of damage arising from earthquakes has become an ambition of humanity. For example, earthquakes with comparable scales with those of the 1906 San Francisco earthquake are expected to occur in the Pacific coast of the United States and Tokyo. Similarly, earthquakes of comparable scales with those of the 2011 Northeastern Japan earthquake are expected to occur in the Cascadia subduction zone of Northern California or Southwestern Japan. Thus, great endeavors have been made to mitigate the damage arising from these disasters. New earthquake observation techniques (e.g., [1]) with new data science-based approaches using large-scale data (e.g., [2, 3]) are under extensive research to mitigate earthquake damage on the one hand and elucidate the earthquake phenomenon on the other. Large-scale international scientific projects are underway to change the scientific perception of earthquakes, such as connecting slow earthquakes without shaking with extremely large earthquakes that have occurred in recent years [4, 5].

Contributions from the supercomputing community for this earthquake problem with high social importance and scientific interest are highly anticipated [6], yielding a variety of outcomes [7]. For example, [8–10] were developed for phenomenon (a), [11–15] were developed for phenomenon (b), and [16–18] were developed for phenomenon (c) in the supercomputing community, which

Figure 1: The earthquake process involves crust deformation due to plate movement, earthquake occurrence and propagation of seismic waves, and the response of cities. As the target area is large and the required resolution is fine, the computational cost is huge. The computation of response of the cities is particularly costly.

made important contributions to the improvement of the analysis capabilities of earthquake problems. The analysis method in this paper targets phenomenon (c) because of its extremely high computational cost compared with that of (a) and (b) at present. In addition, the assessment reliability of phenomenon (c) is crucial for mitigating earthquake disasters.

With the rapid construction and development in the urban environment in recent years, it has become increasingly difficult to predict future damage for phenomenon (c) by extrapolating the data from past damage records. Thus, given the current state of damage estimation, damage levels beyond our experiences or expectations may occur. Using high-resolution urban models of digital twins based on accumulated observation data and equation-based modeling is expected to help us attain a highly reliable estimation. However, this will incur extremely high analysis costs, even with modern advanced computational capabilities. The highly reliable estimation of urban response to large-scale earthquakes requires solving nonlinear time-history problems of fully coupled ground-city models of large areas with localized highly complex structures in super-high resolutions. The target problem has a size of $10^3 \times 10^3 \times 10^{1 \sim 2}$ m with structures that have very complex geometry in a resolution of $10^{-2 \sim -1}$ m. This requires solving nonlinear time-history continuum mechanics problems with three-dimensional finite-element methods using solid elements, which can analytically fulfill the traction-free boundary conditions at the surface. Very small elements are required for modeling complex geometry and heterogeneous nonlinear material properties. For such element sizes, using low-order elements are satisfactory for numerical convergence of the solution. Different from structured mesh, using unstructured mesh leads to random numbering of nodes and thus involves random data access that is not efficient on recent computers. Since the computational cost of element stiffness matrices in low-order elements are small, the relative cost of random data access becomes large, leading to low computational efficiency. Thus, attaining high computational efficiency for low-order unstructured elements on recent computers are regarded challenging. The cost

involved in generating very complex finite-element models of more than 10^{10} degrees-of-freedom and solving 10^3 to 10^4 time steps of unstructured low-order finite-element methods with low computational density and high random data access hinders conducting such analysis. Since small elements are generated locally for models with very complex geometry using solid elements, use of explicit time integration leads to very small time increments and thus huge computational costs. Thus, we use the Newmark- β method ($\beta = 1/4$, $\delta = 1/2$) for time integration and solve the target equation

$$\left(\frac{4}{dt^2} \mathbf{M} + \frac{2}{dt} \mathbf{C}^n + \mathbf{K}^n \right) \delta \mathbf{u}^n = \mathbf{F}^n - \mathbf{Q}^{n-1} + \mathbf{C}^n \mathbf{v}^{n-1} + \mathbf{M} \left(\mathbf{a}^{n-1} + \frac{4}{dt} \mathbf{v}^{n-1} \right), \quad (1)$$

with iterative solvers. Here we target material nonlinearity and linearize the nonlinear set of equations in each time step using the nonlinear material property obtained using the solution of the previous time step. Here, $\delta \mathbf{u}$, $\mathbf{u}$, $\mathbf{v}$, $\mathbf{a}$, $\mathbf{F}$, $\mathbf{Q}$ represent incremental displacement, displacement, velocity, acceleration, external force, and internal force vectors, respectively; $\mathbf{M}$, $\mathbf{C}$, $\mathbf{K}$ represent consistent mass, damping, and stiffness matrices, respectively; and dt and n indicate time step increment and time step number, respectively. We used Rayleigh damping for $\mathbf{C}$, where element damping matrix $\mathbf{C}_e^n$ is constructed as a linear combination of element consistent mass matrix $\mathbf{M}_e$ and element stiffness matrix $\mathbf{K}_e^n$. By generating the stiffness matrix, considering material nonlinearity for each time step based on $\delta \mathbf{u}^n$ obtained by solving Eq. (1), the nonlinear time history problem is solved by updating $\mathbf{Q}^n \leftarrow \mathbf{Q}^{n-1} + \mathbf{K}^n \delta \mathbf{u}^n$, $\mathbf{u}^n \leftarrow \mathbf{u}^{n-1} + \delta \mathbf{u}^n$, $\mathbf{v}^n \leftarrow -\mathbf{v}^{n-1} + \frac{2}{dt} \delta \mathbf{u}^n$, $\mathbf{a}^n \leftarrow -\mathbf{a}^{n-1} - \frac{4}{dt} \mathbf{v}^{n-1} + \frac{4}{dt^2} \delta \mathbf{u}^n$. Solving Eq. (1) using iterative solvers leads to bad convergence characteristics for the urban problems, as it involves very complex structures. Unstructured finite-element methods are gaining popularity for modeling problems in phenomena (a)-(c), mainly because of their ability to yield precise geometrical modeling and reflect traction-free boundary conditions with high accuracy. Thus, the outcomes of this paper are expected to advance our scientific knowledge and meet the expectations of society. Although the finite-element method is used as the de facto standard in the manufacturing industry and extensive research has been conducted for fast solvers for Eq. (1), the target problem size remains limited to 10^{6-7} degrees of freedom. Therefore, we cannot merely apply the methods developed in this field; instead, we must develop new methods for solving large-scale earthquake problems. The development of a new method may entail broad ripple effects on the enhancement of analysis capabilities in the manufacturing industry.

In this paper, we develop a new approach by extending the convergence of high-performance computing (HPC) and artificial intelligence (AI). Here, we converge equation-based modeling and data-driven learning by HPC by developing a fast HPC-based scalable implicit solver that uses data generated while conducting equation-based modeling for data-driven learning. Here we propose a method that can accelerate solving huge-scale finite-element problems on recent supercomputer systems by utilizing previously computed data with high parallel computation efficiency on a huge number of computer nodes. This solver will accelerate the convergence process by making constraints in the solution space for local errors. In addition, we develop suitable matrix-vector multiplication kernel algorithms for the target computer architecture to accelerate

computation. We will call this new solver *IRIS* for fast, scalable Iterative Implicit Solver with the convergence of equation-based modeling and data-driven learning. The effectiveness of the solver will be shown on an urban earthquake problem that involves huge computational costs and challenging mathematical properties. We make measurements on CPU-based Fugaku system for evaluating the scalability of the developed method on massively parallel systems, and make measurements on Intel Xeon (Cascade Lake) CPU-based systems and NVIDIA V100 and A100 GPU-based systems to assess the performance of the developed method on widely available computer systems. Here, porting to GPU systems were conducted using directive based methods (i.e., OpenACC) to show that high performance can be attained with small implementation costs. This paper is extended from the preliminary work in SC20 Research Poster [19] with additions in detailed discussions and additional program development and performance measurement for GPU-based systems.

2 COMPARISON WITH RELATED STUDIES

The major challenges in nonlinear dynamic analysis of fully coupled ground-city systems include generating complicated finite-element models of cities with more than 10^{10} degrees of freedom and solving 10^{3-4} time steps of nonlinear time history problems with poor convergence characteristics. Concerning the first challenge, recent studies, such as [20], enable generating high-resolution large-scale finite element models of cities. The remaining challenge involves developing fast and scalable methods for solving computationally costly Eq. (1). Solving Eq. (1), derived from the low-order unstructured finite element method, involves considerable random access. Thus, it is not straightforward to attain performance on recent computer architectures. This is why, to date, it has not yet been possible to develop a general, fast, and scalable solver. Although challenging, the development of such a solver is expected to have substantial utility, as it meets expectations for the mitigation of earthquake disasters, advances the earthquake science, and has great ripple effects in related fields, such as the manufacturing industry. Among the scalable low-order unstructured finite element methods applicable for solving Eq. (1) is the SC14 Gordon Bell Finalist solver *GAMERA* [16] and [17, 18]. [17] and [18] developed high-performance, scalable solvers by exploiting specific characteristics of the target problems. [17] reduced memory access cost by structuring memory access based on the homogeneity of the target problem, and [18] improved the convergence characteristics of the solver by utilizing the locality of the target problem. However, the target problem in this study does not have a homogeneous structure as in the target problem in [17], nor does it have much locality as in the target problem of [18]. Thus, we will compare the proposed solver *IRIS* with the most general scalable solver *GAMERA*, which does not utilize such specific problem characteristics. *PCGE*, which is one of the representative standard methods for solving such complex problems, is also used for comparison to deepen our understanding of the performance of *IRIS*. From the aforementioned reasons, we will examine the performance of the proposed method *IRIS* by comparing it with the state-of-the-art *GAMERA* and the standard solver algorithm *PCGE*. The overview of the performance is as follows:

Figure 2: Computation example of 11,002,859,706 degrees of freedom; 2,593,545,132 element model of fully coupled ground and aboveground/underground structure earthquake shaking analysis using *IRIS* on 98,304 CPU cores (2,048 nodes) of Fugaku: (a) The 2×2 km domain was modeled with 0.5-m 2nd ordered tetrahedral elements with 9323 buildings and (b) an underground complex with 645 tenants. (c) The complex nonlinear seismic response was computed as a result of the three-dimensional configuration.

The scalable implicit solver *IRIS* achieved 15.2-fold speedup over the conventional method *PCGE* and a 12.8-fold speedup over the state-of-the-art *GAMERA*, with 96.4% size-up scalability up to 24,576 nodes of Arm SVE CPU-based Fugaku. *IRIS* increased the capability of solving extremely large-scale problems within a short amount of time. The performance of *IRIS* was 12.7% of FP64 peak on 48 nodes of Fugaku and 12.8% of FP64 peak on 48 nodes of an Intel Xeon-based (Cascade Lake) supercomputer Oakbridge-CX, which is very high for the low-order unstructured finite element method involving random access-dominated sparse-matrix operations and global communication.

With [18], which tried to conduct an urban earthquake simulation, it was difficult to attain high performance in a practical urban setting with complex structures because high-frequency modes dominate the target problem. However, relying on the new equation-based modeling and data-driven learning-based estimation of high-frequency modes, *IRIS* can overcome this problem. Indeed, *IRIS* attained a 10.3-fold speedup over the state-of-the-art solver *GAMERA* in a complex problem shown in Fig. 2. Here, a fully coupled ground-city system, comprising 3 soil layers, 9,323 buildings and an underground complex with 645 tenants, was modeled with tetrahedral finite elements with a minimum element size of 0.5 m. As shown in Fig. 2(c), the seismic response reflecting the building properties as well as the three-dimensional ground structure beneath it was obtained for each building. Furthermore,

it was visible that underground structures affect the propagation of earthquake waves around them, leading to a complex shaking distribution on the ground surface. Attaining such a complex response is only possible by conducting a fully coupled analysis of highly detailed three-dimensional ground-city models. Because the data of the ground and structures, as well as the computational models contain uncertainties, conducting high-quality analyses on digital twins requires the integration of computational models with observation data via optimization methods. However, this entails a huge number of large-scale analyses. By reducing the computation time for each case of the large-scale analysis, many more cases can be conducted at a reasonable cost. Therefore, the development of *IRIS* is expected to advance the research on the integration of equation-based modeling and observation data.

3 METHODOLOGY

Eq. (1), obtained by discretizing the complex city model with second-order unstructured tetrahedral elements using the three-dimensional nonlinear finite element method, can be denoted as $\mathbf{A}^n \delta \mathbf{u}^n = \mathbf{f}^n$, where both $\mathbf{A}^n \Leftarrow \left(\frac{4}{dt^2} \mathbf{M} + \frac{2}{dt} \mathbf{C}^n + \mathbf{K}^n \right)$ and $\mathbf{f}^n \Leftarrow \mathbf{F}^n - \mathbf{Q}^{n-1} + \mathbf{C}^n \mathbf{v}^{n-1} + \mathbf{M} \left(\mathbf{a}^{n-1} + \frac{4}{dt} \mathbf{v}^{n-1} \right)$ change for each time step n . The characteristics of this problem are as follows: (a) high-frequency modes are dominant, causing extremely poor convergence in the iterative solver; (b) $\mathbf{A}^n$ is a sparse matrix, and, thus, data access to

$\delta \mathbf{u}^n$ is random; and (c) the problem size is more than 10^{10} degrees of freedom, and, thus, $\mathbf{A}^n$ cannot be stored explicitly on memory even in compressed matrix storage formats (e.g., compressed row storage). The widely used standard method suitable for solving such a problem is the *PCGE* method, which is computed in double precision. *PCGE* is a conjugate gradient solver using the 3×3 block Jacobi of $\mathbf{A}^n$ as the preconditioning matrix, with matrix-vector products computed using the element-by-element (EBE) method [21]. In the EBE method, $\mathbf{A}^n$ is not stored explicitly on memory and is processed on the fly. The matrix-vector products are computed by conducting local matrix-vector products $\mathbf{f}_e^n \leftarrow \mathbf{A}_e^n \mathbf{N}_e^T \delta \mathbf{u}^n$ and adding them up as $\mathbf{f}^n \leftarrow \sum_e \mathbf{N}_e \mathbf{f}_e^n$. Here, $\mathbf{A}_e^n$ represents local element matrices computed from the nodal coordinates and material information, whereas $\mathbf{N}_e$ represents the matrices for the mapping of the element-wise nodal values to global nodal values. However, it is difficult to improve the convergence with this preconditioner, leading to poor time to solution. In this paper, we propose a data-driven approach to improve convergence and time to solution. Here, we construct a surrogate model of the physics simulation by applying data-driven learning to the analysis results obtained in the previous time steps and using the surrogate model as a preconditioner. This algorithm not only improves convergence but also enables more efficient computation, as it replaces random access-based computation related to $\mathbf{A}^n$ with sequential access-based data-driven learning computations. As the computation of matrix-vector products of $\mathbf{A}^n$ remains a major kernel, it is also essential to develop matrix-vector algorithms for the latest architectures. Thus, we also show the details of the tuning of the EBE kernel to the latest computer architecture. Here, we focus on CPU-based Fugaku system as it is currently the largest system available. We also show matrix-vector algorithms for GPU-based systems for showing the applicability of the developed method on a variety of computer architectures. Since this kernel is a type of random access dominated on cache computation kernel, the development of algorithms suitable for each architecture is expected to be beneficial for other applications with similar kernels.

3.1 Combining equation-based modeling and data-driven learning for accelerating solver convergence

Various data-driven methods have been proposed for generating surrogate models of nonlinear equation-based problems (e.g., the integration of dynamic mode decomposition[22] and the Koopman operator[23], or methods based on neural differential equations[24]). However, it is difficult to use these methods to obtain the response of a large system with 10^{10} degrees of freedom with nonlinear behavior in both global low-frequency and local high-frequency modes, in high accuracy required in equation-based modeling (i.e., 10^{-8} relative error tolerance) in a shorter time than that of solvers based on equation-based modeling. Furthermore, it is difficult even to store the data required to reconstruct the behavior up to high-frequency modes, as many samples, corresponding to the number of modes of the system, are required to resolve the high-frequency modes of the system. We propose a fast and scalable implicit solver *IRIS*, which is based on a new approach, to be used to overcome the shortcomings of the data-driven approaches by making use of

equation-based modeling. The idea of the proposed model is to capture the global nonlinear behavior by a coarsened equation-based model with small degrees of freedom and to capture the remaining high-frequency modes by localized data-driven expansion in physical and error spaces. By constructing a high-performance hybrid surrogate model based on both equation-based modeling and data-driven approach and using it as a preconditioner within the constraints of equation-based modeling, we can obtain the same accuracy solution for the target equation as that of the pure equation-based modeling approaches.

The proposed method is applicable to general nonlinear dynamic problems $\mathbf{A}^n \delta \mathbf{u}^n = \mathbf{f}^n$ with poor convergence characteristics. Thus, we present the algorithm in its general form with three levels of coarsening to advance our understanding of the algorithm in connection with conventional equation-based modeling. (Although described as a preconditioner, the hybrid surrogate model of equation-based modeling and data-driven approach can be extracted and used for other purposes.) Instead of using a constant preconditioning matrix to solve $\mathbf{A}^n \delta \mathbf{u}^n = \mathbf{f}^n$, we used the adaptive conjugate gradient method [25] to introduce the surrogate model-based preconditioner. Hence, $\mathbf{A}^n \delta \mathbf{u}^n = \mathbf{f}^n$ is solved using the conjugate gradient (CG) method presented in Algorithm 1, where preconditioning equation $\mathbf{A}^n \mathbf{z}^{n,i} = \mathbf{r}^{n,i}$ in the i -th iteration of the CG method is solved with another iterative solver with a relatively large error tolerance. The preconditioner is computed in single precision, as it does not require high accuracy. Since the data-driven approach is only used in the preconditioner, it is important to note that the same solution as that of the standard solver is obtained (i.e., Algorithm 1 is computed in double precision with a relative error of $\epsilon = 10^{-8}$). The EBE method is used to compute matrix-vector products in Algorithm 1, as it is difficult to store $\mathbf{A}^n$ in double precision in memory.

Next, we will explain the surrogate model preconditioner based on the combination of equation-based modeling and data-driven learning. First, we obtain equation $\mathbf{A}_{c1}^n \mathbf{z}_{c1}^{n,i} = \mathbf{r}_{c1}^{n,i}$, which is equivalent to but coarser than $\mathbf{A}^n \mathbf{z}^{n,i} = \mathbf{r}^{n,i}$. Then, we obtain $\mathbf{A}_{c2}^n \mathbf{z}_{c2}^{n,i} = \mathbf{r}_{c2}^{n,i}$, which is equivalent to but coarser than $\mathbf{A}_{c1}^n \mathbf{z}_{c1}^{n,i} = \mathbf{r}_{c1}^{n,i}$. The right-hand vectors are defined as $\mathbf{r}_{c1}^{n,i} \leftarrow \mathbf{P}_{c1} \mathbf{r}^{n,i}$ and $\mathbf{r}_{c2}^{n,i} \leftarrow \mathbf{P}_{c2} \mathbf{r}_{c1}^{n,i}$ using projections $\mathbf{P}_{c*}$. By using projections $\mathbf{R}_{c*}$, $\mathbf{R}_{c1} \mathbf{z}_{c1}^{n,i}$ is expected to approximate $\mathbf{z}^{n,i}$, and $\mathbf{R}_{c2} \mathbf{z}_{c2}^{n,i}$ is expected to approximate $\mathbf{z}_{c1}^{n,i}$. Coarse equations $\mathbf{A}_{c1}^n \mathbf{z}_{c1}^{n,i} = \mathbf{r}_{c1}^{n,i}$ and $\mathbf{A}_{c2}^n \mathbf{z}_{c2}^{n,i} = \mathbf{r}_{c2}^{n,i}$, and projections $\mathbf{P}_{c*}$ and $\mathbf{R}_{c*}$ can be constructed using standard solver methods used for solving equation-based problems (e.g., multi-grid methods). Here, $\mathbf{A}_{c1}^n \mathbf{z}_{c1}^{n,i} = \mathbf{r}_{c1}^{n,i}$ is constructed using the first-order tetrahedral elements through the geometric multi-grid coarsening of $\mathbf{A}^n \mathbf{z}^{n,i} = \mathbf{r}^{n,i}$. Then, $\mathbf{A}_{c2}^n \mathbf{z}_{c2}^{n,i} = \mathbf{r}_{c2}^{n,i}$ is constructed using the algebraic multi-grid coarsening of $\mathbf{A}_{c1}^n \mathbf{z}_{c1}^{n,i} = \mathbf{r}_{c1}^{n,i}$. Although the global behavior of the nonlinear system can be solved using these coarse equations, they are not effective in solving problems that are high-frequency mode dominant because the approximation accuracy of $\mathbf{R}_{c1} \mathbf{z}_{c1}^{n,i}$ for $\mathbf{z}^{n,i}$ and $\mathbf{R}_{c2} \mathbf{z}_{c2}^{n,i}$ for $\mathbf{z}_{c1}^{n,i}$ deteriorates. This is quite obvious because the degree of freedom was reduced by coarsening; therefore, even if the coarse equations are solved accurately, the number of iterations required for solving the fine equation will not decrease because high-frequency modes are obtained only in the fine equation, and, thus, the computation cost

Algorithm 1 Algorithm based on the adaptive conjugate gradient method to perform nonlinear dynamic response analysis at n -th time step by solving Eq. (1). Line 2 indicates the right hand vector of Eq. (1), and line 3 is the initial solution estimated using the Adams-Bashforth method. The algorithm to solve $A^n \delta u^n = f^n$ is shown in lines 4-22. ϵ is the tolerance for convergence judgment. Double-precision computation is used, except for the computation of line 9.

```

1: read boundary condition of  $n$ -th time step
2:  $f^n \leftarrow F^n - Q^{n-1} + C^n v^{n-1} + M(a^{n-1} + \frac{4}{dt} v^{n-1})$ 
3:  $\delta u^n \leftarrow \frac{dt}{24} (-9v^{n-4} + 37v^{n-3} - 59v^{n-2} + 55v^{n-1})$ 
4:  $r \leftarrow f^n - A^n \delta u^n$ 
5:  $\beta \leftarrow 0, p \leftarrow 0$ 
6:  $i \leftarrow 1$ 
7: while  $\|r\|_2 / \|f\|_2 \geq \epsilon$  do
8:    $r^{n,i} \leftarrow r$ 
9:   solve  $A^n z^{n,i} = r^{n,i}$  by CG method with  $\epsilon_{pre}$ 
10:   $z \leftarrow z^{n,i}$ 
11:  if  $i > 1$  then
12:     $\beta \leftarrow (z, q) / \rho$ 
13:  end if
14:   $p \leftarrow z + \beta p$ 
15:   $q \leftarrow A^n p$ 
16:   $\rho \leftarrow (z, r)$ 
17:   $\alpha \leftarrow \rho / (p, q)$ 
18:   $q \leftarrow -\alpha q$ 
19:   $r \leftarrow r + q$ 
20:   $\delta u^n \leftarrow \delta u^n + \alpha p$ 
21:   $i \leftarrow i + 1$ 
22: end while
23:  $Q^n \leftarrow Q^{n-1} + K^n \delta u^n$ 
24:  $u^n \leftarrow u^{n-1} + \delta u^n$ 
25:  $v^n \leftarrow -v^{n-1} + \frac{2}{dt} \delta u^n$ 
26:  $a^n \leftarrow -a^{n-1} - \frac{4}{dt} v^{n-1} + \frac{4}{dt^2} \delta u^n$ 
27: output results of  $n$ -th time step
28: set  $A^{n+1}$  considering nonlinearity

```

of the fine equation would not be transferred to the less costly computation of the coarser equations. On the other hand, as a time history problem of a physical problem is solved, the projection $D_{c2}^{n,i}$ from $r_{c1}^{n,i} - A_{c1}^n R_{c2} z_{c2}^{n,i}$ to $z_{c1}^{n,i} - R_{c2} z_{c2}^{n,i}$, which is defined on local physical and error spaces excluding the global low-frequency components, is expected to be quite similar among time-steps within a short period of time. The projection $D_{c1}^{n,i}$ from $r^{n,i} - A^n R_{c1} z_{c1}^{n,i}$ to $z^{n,i} - R_{c1} z_{c1}^{n,i}$ is also expected to be quite similar among time-steps within a short period of time. Thus, we can expect to construct a mapping that can reproduce high-frequency modes in the whole domain by constructing a localized mapping capable of considering high-frequency modes at each small domain using data obtained in recent time steps (data of several past time steps $n-1, n-2, \dots$). Here, the domain is partitioned using graph partitioning methods, and the nodal data in each subdomain j is projected using $D_{c*}^{n,i,j}$ which is constructed with the data of the previous m time steps (i.e., the set of $r_{c1}^{n,i} - A_{c1}^n R_{c2} z_{c2}^{n,i}$ and $z_{c1}^{n,i} - R_{c2} z_{c2}^{n,i}$ and the set

Algorithm 2 A solver algorithm combining equation-based modeling and data-driven learning is used to solve $A^n z^{n,i} = r^{n,i}$ in line 9 of Algorithm 1. $diag[A_{c2}^n]$ is the 3×3 block diagonal matrix of A_{c2}^n . The 3×3 block diagonal matrix is used as the preconditioning matrix in each CG method, whose tolerance of convergence judgment is ϵ_{c2} , ϵ_{c1} and ϵ_{pre} , respectively. $D_{c2}^{n,i}$ and $D_{c1}^{n,i}$ are the projections defined in each small domain for the estimation of high-frequency modes obtained from analyzing data of previous steps. By using these projections for the estimation of the high-frequency modes, the number of iterations in the fine equation is transferred to the iterations on less costly coarser CG methods, even for the high-frequency mode-dominated problems. All computation is conducted in single-precision.

```

1:  $r_{c1}^{n,i} \leftarrow P_{c1} r^{n,i}$ 
2:  $r_{c2}^{n,i} \leftarrow P_{c2} r_{c1}^{n,i}$ 
3:  $z_{c2}^{n,i} \leftarrow (diag[A_{c2}^n])^{-1} r_{c2}^{n,i}$ 
4: solve  $A_{c2}^n z_{c2}^{n,i} = r_{c2}^{n,i}$  by CG method with  $\epsilon_{c2}$ 
5:  $z_{c1}^{n,i} \leftarrow R_{c2} z_{c2}^{n,i} + D_{c2}^{n,i} (r_{c1}^{n,i} - A_{c1}^n R_{c2} z_{c2}^{n,i})$ 
6: solve  $A_{c1}^n z_{c1}^{n,i} = r_{c1}^{n,i}$  by CG method with  $\epsilon_{c1}$ 
7: construct/update  $D_{c2}^{n+1,i}$  using  $(z_{c1}^{n,i} - R_{c2} z_{c2}^{n,i})$  and  $A_{c1}^n (z_{c1}^{n,i} - R_{c2} z_{c2}^{n,i})$ 
8:  $z^{n,i} \leftarrow R_{c1} z_{c1}^{n,i} + D_{c1}^{n,i} (r^{n,i} - A^n R_{c1} z_{c1}^{n,i})$ 
9: solve  $A^n z^{n,i} = r^{n,i}$  by CG method with  $\epsilon_{pre}$ 
10: construct/update  $D_{c1}^{n+1,i}$  using  $(z^{n,i} - R_{c1} z_{c1}^{n,i})$  and  $A^n (z^{n,i} - R_{c1} z_{c1}^{n,i})$ 

```

of $r^{n,i} - A^n R_{c1} z_{c1}^{n,i}$ and $z^{n,i} - R_{c1} z_{c1}^{n,i}$). Of the several methods available to construct $D_{c*}^{n,i,j}$ (e.g., pseudo-inverse matrix), we use the Gram-Schmidt orthonormalization, as normal vectors can be added on the fly at a small computational cost. This algorithm is suitable for massively parallel computation, as it does not require communication among subdomains, and the computation cost per node is constant. Furthermore, this orthonormalization process is sequential memory access dominant; thus, the proposed method replaces the random memory access-dominated computation of the unstructured finite element method with a computation more suitable to recent computer architecture. The algorithm is summarized in Algorithm 2.

In a nonlinear dynamic problem, it is difficult to estimate the global and local nonlinear time history response with data-driven approach based on past data in accuracy sufficient for equation-based modeling. In this paper, we transformed the target problem with low- and high-frequency components to a problem with only small low-frequency components during the solver process, which enabled estimating results in high-accuracy without the use of more advanced data-driven approaches that may lead to deterioration in scalability or increase in analysis cost.

3.2 EBE kernel for recent architectures

As the hybrid surrogate model, based on equation-based modeling and data-driven approach, can be computed at a minimal cost, the single-precision EBE matrix-vector product in line 9 of Algorithm 2 becomes the most costly component in the whole application. Although the EBE method can reduce the memory footprint and transfer size when compared with methods storing the global matrix

```

1  ! Compute EBE
2  do ie=1,ne (non SIMD computation)
3    cny1=cny(1,ie)
4    cny2=cny(2,ie)
5    cny3=cny(3,ie)
6    cny4=cny(4,ie)
7    ue11=u(1,cny1)
8    ue21=u(2,cny1)

9    ue34=u(3,cny4)
10   xe11=coor(1,cny1)
11   xe21=coor(2,cny1)

12   xe34=coor(3,cny4)
13   ! compute fe using ue11~ue34 and xe11~xe34
14   fe11=...
15   fe21=...

16   fe34=...
17   ! add fe to f
18   f(1,cny1)=fe11+f(1,cny1)
19   f(2,cny1)=fe21+f(2,cny1)

20   f(3,cny4)=fe34+f(3,cny4)
21 enddo

```

Figure 3: Standard EBE kernel algorithm. We show the case for linear tetrahedral elements with four nodes, each with three degrees-of-freedom (i.e., x , y , and z directions).

on memory, data recurrence is involved when adding the element left-hand side vector to the global left-hand side vector. Thus, it is not straightforward to attain performance on recent computer architectures. Furthermore, as the computation of the EBE kernel has many components that are dependent on each other, special tuning is required for attaining performance on architectures with limited register resources. In this paper, we utilize fast EBE algorithm developed in [26] for Fugaku, and also show suitable development for GPU-based systems using OpenACC.

Kernel suitable for Fugaku. Here we describe the kernel algorithms developed for Fugaku in previous study [26]. Fig. 3 shows a straightforward implementation of the EBE kernel. As the data recurrence when adding the results $fe11$ – $fe34$ to the f vector was hindering the use of the SIMD units, we split the innermost loop into two loops: (a) the computation part of $f_i^e \leftarrow A_i^e u_i^e$ (Fig. 4 lines 9–24) and (b) the summation part of f_i^e to f (lines 26–34). This led to the use of the SIMD operations for the computationally rich first loop. Here, we reduce the overhead of loop splitting by blocking the loop with small block size NL , which is typically set to the SIMD vector length. This maintains the temporary buffers $fe11$ – $fe34$ on cache. A cache-friendly thread partitioning method is also developed for efficient multi-core execution of the SIMD-enabled EBE kernel. This thread partitioning technique can be used instead of the coloring methods that color the whole mesh so that elements in each color will not have any shared nodes, yielding better cache reuse of nodal values u , $coor$, and f in comparison with standard coloring methods. Furthermore, loop splitting is applied to the main computation loop (lines 9–24) to facilitate instruction scheduling on the CPU cores.

```

1  !$OMP PARALLEL
2  do icolor=1,ncolor
3  !$OMP DO
4    ! for each thread
5    do iu=1,numberofthreads
6    ! block loop with blocksize NL
7    do ieo=nstart(iu,icolor),nend(iu,icolor),NL
8      ! compute BDBu
9      do ie=1,min(NL, nend(iu,icolor)-ieo+1)
10       cny1=cny(1,ieo+ie-1)
11       cny2=cny(2,ieo+ie-1)
12       cny3=cny(3,ieo+ie-1)
13       cny4=cny(4,ieo+ie-1)
14       ue11=u(1,cny1)
15       ue21=u(2,cny1)

16       ue34=u(3,cny4)
17       xe11=coor(1,cny1)
18       xe21=coor(2,cny1)

19       xe34=coor(3,cny4)
20       ! compute fe using ue11~ue34 and xe11~xe34
21       fe11(ie)=...
22       fe21(ie)=...

23       fe34(ie)=...
24     enddo
25     ! add to vector
26     do ie=1,min(NL, nend(iu,icolor)-ieo+1)
27       cny1=cny(1,ieo+ie-1)
28       cny2=cny(2,ieo+ie-1)
29       cny3=cny(3,ieo+ie-1)
30       cny4=cny(4,ieo+ie-1)
31       f(1,cny1)=fe11(ie)+f(1,cny1)
32       f(2,cny1)=fe21(ie)+f(2,cny1)

33       f(3,cny4)=fe34(ie)+f(3,cny4)
34     enddo
35   enddo ! ieo
36 enddo ! iu
37 !$OMP END DO
38 enddo ! icolor
39 !$OMP END PARALLEL

```

Figure 4: EBE kernel algorithm with SIMD-enhanced computation and multi-core coloring for Fugaku.

Kernel suitable for GPU-based systems. As it was with the EBE kernel algorithm for many-core wide SIMD CPUs, the random access becomes bottleneck for GPUs. However, for GPUs, fast atomics are available and thus use of such hardware enables high performance. Fig. 5 shows the OpenACC implementation of the EBE kernel. Since the OpenACC framework automatically assigns threads and blocks to loop variable i , and fast atomics are available for summation of f , we can expect high performance with small development cost.

4 PERFORMANCE MEASUREMENT SETTINGS

To be able to carefully follow the target city problem, we construct the finite element models of actual city blocks, and measure the performance of the nonlinear response analysis by inputting an earthquake wave from the bottom of the model. Here, we model an actual 128 m $\times$ 128 m city block in Tokyo, comprising buildings and a three-layered ground with topography (Fig. 6). The first and second layers are soft layers that behaves nonlinearly under strong inputs; the third layer is a hard engineering bedrock layer. The model is

```

1 !$acc parallel present(ne,cny,u,coor,f),
2 !$acc& private(ie,cny1, cny2,cny3,cny4,
3 !$acc& ue11,ue12,...,ue34,
4 !$acc& xe11,xe21,...,xe34,fe11,fe21,...,fe34)
5 do ie=1,ne
6   cny1=cny(1,ie)
7   cny2=cny(2,ie)
8   cny3=cny(3,ie)
9   cny4=cny(4,ie)
10  ue11=u(1,cny1)
11  ue21=u(2,cny1)

12  ue34=u(3,cny4)
13  xe11=coor(1,cny1)
14  xe21=coor(2,cny1)

15  xe34=coor(3,cny4)
16  ! compute fe using ue11-ue34 and xe11-xe34
17  fe11=...
18  fe21=...

19  fe34=...
20 !$acc atomic
21   f(1,cny1)=fe11+f(1,cny1)
22 !$acc atomic
23   f(2,cny1)=fe21+f(2,cny1)

24 !$acc atomic
25   f(3,cny4)=fe34+f(3,cny4)
26 enddo
27 !$acc end parallel

```

Figure 5: EBE kernel algorithm for GPUs based on OpenACC.

duplicated in the x and y directions for making large-scale models used for weak-scaling measurements. The models are discretized with a sufficiently small element size (minimum element size was 0.25 m) for convergence in the acceleration response, even if the soil had been softened because of nonlinearity. Table 1 shows the properties of the weak and strong scaling model sets. METIS was used to partition the model, allowing us to expect load balancing and communication characteristics similar to those of a large-scale problem. As typical urban earthquake problems involves solving 10^3 – 4 time steps, we measure the performance of solving several time steps. Here, we measured the performance between the 101 and 200-th time steps for weak and strong scaling. Eq. (1) is solved to the relative error tolerance of $\epsilon = 10^{-8}$ for all solvers. 10^{-1} is used for the relative error tolerances of all the inner solvers (ϵ_{c2} , ϵ_{c1} , and ϵ_{pre}); the previous $m = 32$ steps were used for constructing and updating D_{c*}^{ni} (data of the $m+1$ -th step were discarded from the basis functions of the Gram-Schmidt orthonormalization every time step). The elapsed time for *IRIS* included the time required for equation solving and data-driven learning and inferencing. As shown in Table 1, the number of iterations required for convergence using *PCGE* is almost constant among the models; thus, the model set is suitable for measuring weak scaling. We measured performance on Fugaku, Intel Xeon-based Oakbridge-CX, and NVIDIA V100 & A100 based systems:

Fugaku is a massively parallel Arm CPU-based supercomputer at RIKEN, Center for Computational Science [27]. Each compute node comprises a single A64FX CPU [28] based on the Arm v8.2-A SVE instruction set architecture. Each of the 48 compute-cores of

Figure 6: This finite element model is duplicated in the x and y directions to make large-scale models for performance measurement.

the Arm CPUs has two sets of fused multiply-add (FMA) SIMD units with a width of 512 bits. This leads to FMA (two operations) $\times$ SIMD length 8×2 sets = 32 double-precision operations per clock. Consequently, the peak performance of each CPU is 32 operations $\times$ 48 cores $\times$ 2 GHz = 3072 FP64 GFLOPS. The peak performance for FP32 operations is twice that of the FP64 peak. Each node has 32 GB of HBM2 memory, with a peak memory bandwidth of 1024 GB/s. TofuD, a six-dimensional interconnection network, is used for communication between the nodes [29].

Oakbridge-CX (OBCX) is an Intel Xeon CPU-based system at the University of Tokyo [30]. It comprises 1,368 nodes, each with two 28-core Intel Xeon Platinum (Cascade Lake) 8280 CPUs [31] supporting AVX-512 SIMD instructions. The peak performance of each CPU is 32 operations $\times$ 56 cores $\times$ 2.7 GHz = 4838 FP64 GFLOPS. The peak performance for FP32 operations is twice that of the FP64 peak. Each node has 192 GB of DDR4 RAM with a memory bandwidth of 281 GB/s. A 100-Gbps Intel Omni-Path interconnect is used for communication between the nodes.

The EBE kernel was ported to NVIDIA GPUs and measured on V100 SXM2 GPU and A100 SXM2 GPUs. The peak performance for V100 GPU and A100 GPU are 7.8 and 9.7 FP64 TFLOPS, respectively, while the peak memory bandwidth of V100 GPU and A100 GPU are 900 and 1555 GB/s, respectively.

Twelve or fourteen cores were used per process on Fugaku and OBCX, respectively, both with 4 MPI processes per node, while one process was launched for each GPU. Timers `MPI_WTIME()` were inserted into the code to measure elapsed time. The Fujitsu profiler `fappx` was used to measure the FLOP counts on Fugaku. The FLOPS on OBCX and NVIDIA GPUs were estimated assuming the same FLOP count is involved when running the same code on Fugaku.

5 PERFORMANCE RESULTS

5.1 Algorithm performance

First, we checked the change in the number of iterations using the data-driven mapping algorithm on model A-1 (Fig. 7). The degrees of freedom (DOF) per MPI process for A, the coarsened A_{c1} , and the further coarsened A_{c2} , were 1.28 million, 177,000, and 25,000, respectively. Thus, transferring of the iterations on the fine

Table 1: Configurations of the models used to measure performance (model set A and B is for weak and strong scaling, respectively).

Model	# of compute nodes	# of MPI processes	Total DOF	Mean DOF per MPI domain	Total # of elements	PCGE iterations
A-1 (B-1)	48	192	247,322,799	1,288,139	60,226,447	57,070
A-2	96	384	494,552,673	1,287,897	120,452,894	56,851
A-3	192	768	989,012,421	1,287,776	240,905,788	56,835
A-4	384	1,536	1,977,931,917	1,287,716	481,811,576	56,830
A-5	768	3,072	3,955,770,909	1,287,685	963,623,152	56,819
A-6	1,536	6,144	7,911,448,893	1,287,670	1,927,246,304	56,808
A-7	3,072	12,288	15,822,804,861	1,287,663	3,854,492,608	56,815
A-8	6,144	24,576	31,645,516,797	1,287,659	7,708,985,216	56,812
A-9	12,288	49,152	63,290,940,669	1,287,657	15,417,970,432	56,801
A-10	24,576	98,304	126,581,788,413	1,287,656	30,835,940,864	56,801
B-2	96	384	247,322,799	644,069	60,226,447	57,067
B-3	192	768	247,322,799	322,034	60,226,447	57,069
B-4	384	1536	247,322,799	161,017	60,226,447	57,082

Figure 7: Number of iterations required for model A-1. Total iterations required for solving time steps 101-200 are shown.

model to the coarser models reduces the computational cost. First, we observed the number of iterations of an adaptive CG multi-grid solver without data-driven mapping, corresponding to setting $D_{c1}^{n,i} = 0$ and $D_{c2}^{n,i} = 0$ in Algorithm 2. Although the number of iterations in the fine model ($700 + 46,617 = 47,317$ iterations) was reduced from that of *PCGE* (57,070 iterations), the reduction in the number of iterations was limited to approximately 10,000. Hence, we can see that the problem is high-frequency dominated, hence the effectiveness of using coarse models was limited. When using the data-driven approach for mapping results $z_{c1}^{n,i}$ to $z^{n,i}$, the number of inner *A* iterations was reduced from 46,617 to 25,084, indicating the effectiveness of the developed method for estimating low- to high-frequency modes of the target problem. Using the data-driven mapping from $z_{c2}^{n,i}$ to $z_{c1}^{n,i}$, the inner *A*_{c1} iterations were also reduced. Together with the characteristics of the multi-grid method, where low-frequency modes can be resolved in fewer iterations, we observed that the total number of iterations ($700 + 26,144 + 15,083 + 13,350 = 55,277$) was reduced from that of *PCGE* (57,070), which is expected to be suitable for massively parallel computation where `MPI_Allreduce()` becomes costly.

Next, we observed the performance of the EBE kernel in one compute node of OBCX and Fugaku (Table 2). The standard EBE method denotes the EBE method used in *GAMERA*, which is multi-core enabled, albeit unable to utilize the SIMD units due to data

Table 2: Elapsed time per EBE kernel call on Model A-1.

	OBCX	Fugaku
Standard EBE	16.3 ms	51.4 ms
EBE for multi-core wide-SIMD architecture	2.11 ms	4.02 ms
Speedup from standard EBE	7.73	12.8

Figure 8: Time to solution for model A-1 on OBCX and Fugaku. Elapsed time required for solving time steps 101-200 are shown.

recurrence. The method enabled 7.73- and 12.8-fold speedups on OBCX and Fugaku, respectively. With such improvement in kernel performance and reduction in the number of iterations through the data-driven approach, we can see that a 14.3-fold speedup was attained on OBCX for the whole application when compared with *PCGE* (Fig. 8). Furthermore, the *IRIS* solver was 7.52-fold faster than the state-of-the-art *GAMERA*. Although *GAMERA* was suitable for homogeneous problems (e.g., ground motion problems without structures), the urban earthquake problem was far more complex, which made its performance gain from *PCGE* narrower. However, the hybrid surrogate approach in *IRIS* is effective for such complex problems. 12.8% to peak FP64 performance was attained for the whole *IRIS* application; we can see that the overall algorithm design, combined with architecture-aware tuning, enabled high performance. Similar trends were seen on Fugaku; 12.8- and 15.2-fold speedups were respectively attained compared with *GAMERA* and *PCGE*, with 12.7% of FP64 hardware peak for the full application.

Table 3: Cost breakdown of *IRIS* on Fugaku in model A-1.

	# of iterations	Time per iter.	Total time
Outer	700	9.1 ms	6.3 s
Inner A solver	26144	5.4 ms	140.7 s
Inner A _{c1} solver	15083	1.1 ms	16.7 s
Inner A _{c2} solver	13350	0.45 ms	6.0 s
D _{C*} ⁿⁱ learning/inferencing	700	-	14.9 s
Mapping between grids	700	-	11.7 s
Others (including IO)	-	-	23.3 s
Total	-	-	212.9 s

Table 4: Performance comparison of EBE kernel on Fugaku and NVIDIA GPUs. One process of model A-1 is used on one CMG (12 cores) of Fugaku and one V100 or A100 GPU.

	Fugaku	V100	A100
Peak hardware FP64 TFLOPS	0.768	7.8	9.7
Elapsed time per kernel call (ms)	4.02	0.303	0.233
Ratio to FP64 peak	20.6%	27.0%	28.2%

Table 3 shows that each iteration of Inner A_{c1} was conducted 5-fold faster than that of Inner A solver; it also shows that each iteration of Inner A_{c2} was conducted 12-fold faster than that of Inner A solver. The time required for learning/inferencing of the data-analytics mapping took only 1/14 of the total application time, which led to the speedup of the whole application.

Table 4 shows the performance measurement results of the EBE kernel on NVIDIA GPUs. Compared to the high peak performance of 20.6% obtained on Fugaku, we can see further improved peak performance ratio of 27.0% and 28.2% on V100 and A100 GPUs, respectively. Although fine tuning is possible by conducting local summation of f in shared memory to reduce global memory accesses using CUDA, we can see that kernel algorithm development suitable for the computer architecture allows high performance with small amount of implementation cost by using OpenACC. In total, we can see that the developed method can attain high performance on a wide range of computer architecture.

5.2 Scalability on Fugaku

We used up to 27,648 compute nodes of Fugaku to measure scalability on large-scale problems. We generated models fixing the problem size per process to 1.28 million DOF and increasing the number of compute nodes. Fig. 9 shows the elapsed time for the model set. The smallest model (A-1) with 247 million DOF was solved in 212.7 s with 18.7 TFLOPS (12.7% of peak) when using 48 nodes of Fugaku, whereas the largest model, A-10 with 126 billion DOF, was solved in 220.8 s with 8.87 PFLOPS (11.8% of peak) when using 24,576 nodes of Fugaku. The size-up scalability up to 24,576 nodes was 96.4%; we can see that excellent size-up scalability was attained. The time required for MPI_Allreduce() in the total application increased from 0.85 (48 nodes) to 8.90 s (24,576 nodes), and the point-to-point communication increased from 6.10 (48 nodes) to 11.51 s (24,576 nodes). Thus, we can see that most of the increase in time was due to the increase of time in communication. The time increase in MPI_Allreduce() time followed the theoretical time (i.e., proportional to $\log_2 n$ where n is the number of processes). Hence,

Figure 9: Size-up scalability for model A on Fugaku. Elapsed time for solving time steps 101-200 is shown. Numbers in brackets indicate ratio to FP64 peak FLOPS. **Figure 10: Speedup scalability for model B on Fugaku. The measured elapsed time for solving time steps 101 to 200.**

we can see that the solver algorithm is expected to be scalable to further larger systems. The elapsed time of *GAMERA* and *PCGE* on 24,576 nodes was 2,702 and 3,227 s, respectively, corresponding to 12.2- and 14.6-fold speedup of *IRIS* from *GAMERA* and *PCGE*, respectively. Although measurements were only made on CPU-based Fugaku system, we can expect high scalability on GPU-accelerated systems as the algorithm is designed to have good load balancing as well as small amount of communication.

Fig. 10 shows the strong scaling of the developed solver. For the smallest core count of 192 processes (model B-1), the 247 million DOF problem was solved in 212.7 s. Although some degradation in performance with an increase in CPU cores was visible, we can see that high scalability is attained. The speedup efficiency from 192 → 384 processes was 94.8%, the scalability from 192 → 768 processes was 82.4%, and the scalability from 192 → 1,536 processes was 66.6%. We can see that high scalability was attained by the solver algorithm that has excellent load balancing and reduced communication.

6 CLOSING REMARKS

By extending the convergence of HPC and AI using a more generalized combination of equation-based modeling and data-driven learning in the HPC resources, we designed an algorithm that benefits from the strengths of both equation-based modeling and data-driven learning to accelerate solvers. By considering the local characteristics of time space and error space in the solving process, by obtaining low-frequency components using equation-based modeling in a coarse model on the one hand and obtaining high-frequency components using data-driven learning on data of past computation results on the other, we designed a method that can improve the estimation accuracy for all low- to high-frequency components of the solution, and, thus, improve the convergence characteristics of the scalable implicit solver. This can be regarded as change in the standard solver algorithm to an algorithm that is suitable for recent computer architecture. The equation-based modeling (with random memory access-dominant low-order unstructured finite-element computation) was shifted to data-driven learning (with sequential memory access-dominant vector computation). Together with the development of equation-based modeling techniques suitable for the computer architecture (e.g., the EBE methods shown in this paper) led to a high-performance solver. Although the difference in accuracy and characteristics of the equation-based modeling and data-driven learning methods made it seem difficult to integrate these methods, we managed to facilitate the connection between these methods by designing a suitable interface, which solved the problems that were difficult to solve either by equation-based modeling or by data-driven learning. We can expect constructing more accurate data-rich data-driven models by utilizing last-level storage in the developed approach. This approach can be expected as a way to utilize the diverse storage hierarchies with different bandwidths and capacities which are difficult to utilize in terms of accelerating equation-based modeling. The EBE kernel accelerated in this study is a type of random data access dominant on cache computation kernel. Such kernels exist in many applications, and thus the algorithm development suitable for each target computer architecture show in this study is expected to be useful for accelerating a variety of other applications.

It is important to improve the reliability of urban models to enhance the reliability of earthquake damage estimations based on equation-based modeling. Thus, constructing digital twins of actual cities in cyberspace and using observed urban response data for small earthquakes for the continuous optimization of urban models in extreme-computing systems, such as Big Data & Extreme Computing (BDEC) systems[32], are crucial for the next generation of disaster mitigation. Fast analysis methods, as the one developed in this paper, is useful for conducting optimization within a reasonable time frame. Although the optimization of complex models is challenging, methods suitable for BDEC systems using AI-based methods are being developed (e.g., [33]). Therefore, it is becoming feasible to tackle such problems in the BDEC systems. Furthermore, the proposed approach is not confined to structures but can be generalized to other problems, including modeling the flow of people and goods in an urban environment in digital twins. We can expect substantial contribution from digital twins in the research for suitable designs of cities in the post-COVID-19 era. By utilizing

extreme computing for analyzing big data in physics, cyberspace, and analysis space by equation-based modeling and data-driven learning, we can expect valuable contributions to the HPC field and expansions of the frontiers in urban simulations.

ACKNOWLEDGMENTS

Results were obtained using Fugaku at Center for Computational Science, RIKEN, and Oakbridge-CX at Information Technology Center, The University of Tokyo. This work was supported by MEXT as “Program for Promoting Researches on the Supercomputer Fugaku” (Large-scale numerical simulation of earthquake generation, wave propagation and soil amplification: hp200126, hp210171) and as “Post K computer project” (Priority Issue 3 - Development of integrated simulation systems for hazards and disasters induced by earthquakes and tsunamis: hp190177) and JSPS KAKENHI Grant Numbers JP18H05239, JP18H03795, JP17K14719. We thank GDEP Advance, Inc. for providing A100 GPU environment for performance measurement.

REFERENCES

- [1] N. J. Lindsey, T. C. Dawe, J. B. Ajo-Franklin, Illuminating seafloor faults and ocean dynamics with dark fiber distributed acoustic sensing, *Science* 29 Nov 2019, Vol. 366, Issue 6469, pp. 1103–1107, 2019.
- [2] Z. E. Ross, D. T. Trugman, E. Hauksson, P. M. Shearer, Searching for hidden earthquakes in Southern California, *Science* 24 May 2019, Vol. 364, Issue 6442, pp. 767–771, 2019.
- [3] K. J. Bergen, P. A. Johnson, M. V. de Hoop, G. C. Beroza, Machine learning for data-driven discovery in solid Earth geoscience, *Science* 22 Mar 2019, Vol. 363, Issue 6433, eaau0323, 2019.
- [4] K. Obara, A. Kato, Connecting slow earthquakes to huge earthquakes, *Science* 15 Jul 2016, Vol. 353, Issue 6296, pp. 253–257, 2016.
- [5] T. Nishikawa, T. Matsuzawa, K. Ohta, N. Uchida, T. Nishimura, S. Ide, The slow earthquake spectrum in the Japan Trench illuminated by the S-net seafloor observatories, *Science*, 23 Aug 2019, Vol. 365, Issue 6455, pp. 808–813, 2019.
- [6] T. H. Jordan, SC15 Invited Talk Spotlight: Societal Impact of Earthquake Simulations at Extreme Scale.
- [7] J. de la Puente, J. E. Rodriguez, M. Monterrubio-Velasco, O. Rojas, A. Folch, Urgent Supercomputing of Earthquakes: Use Case for Civil Protection, In *Proceedings of the Platform for Advanced Scientific Computing Conference (PASC '20)*, ACM, 2020.
- [8] K. Fujita, T. Ichimura, K. Koyama, M. Horikoshi, H. Inoue, L. Meadows, S. Tanaka, M. Hori, L. Maddeggedara, T. Hori, A Fast Implicit Solver with Low Memory Footprint and High Scalability for Comprehensive Earthquake Simulation System, Research Poster for SC16: International Conference for High Performance Computing, Networking, Storage and Analysis, 2016.
- [9] K. Fujita, T. Ichimura, K. Koyama, H. Inoue, M. Hori, L. Maddeggedara, Fast and Scalable Low-order Implicit Unstructured Finite-element Solver for Earth's Crust Deformation Problem, In *Proceedings of the Platform for Advanced Scientific Computing Conference (PASC '17)*, 2017.
- [10] T. Yamaguchi, K. Fujita, T. Ichimura, A. Naruse, J. C. Wells, C. J. Zimmer, T. P. Straatsma, M. Hori, L. Maddeggedara, N. Ueda, Low-Order Finite Element Solver with Small Matrix-Matrix Multiplication Accelerated by AI-Specific Hardware for Crustal Deformation Computation, In *Proceedings of the Platform for Advanced Scientific Computing Conference (PASC '20)*, 2020.
- [11] L. Carrington, D. Komatitsch, M. Laurenzano, M. Tikir, D. Michea, N.L. Goff, A. Snively, J. Tromp, High-frequency simulations of global seismic wave propagation using SPECfem3Dglobe, *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis (SC'08)*, IEEE Press, pp. 1–11, 2008.
- [12] A. Heinecke, A. Breuer, S. Rettenberger, M. Bader, A.-A. Gabriel, C. Pelties, A. Bode, W. Barth, X.-K. Liao, K. Vaidyanathan, M. Smelyanskiy, P. Dubey, Petascale High Order Dynamic Rupture Earthquake Simulations on Heterogeneous Supercomputers, *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis (SC'14)*, IEEE Press, 2014.
- [13] H. Fu, C. He, B. Chen, Z. Yin, Z. Zhang, W. Zhang, T. Zhang, W. Xue, W. Liu, W. Yin, G. Yang, X. Chen, 15-Pflops Nonlinear Earthquake Simulation on Sunway TaihuLight: Enabling Depiction of Realistic 10 Hz Scenarios, *SC '17 Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, Article No. 2, 2017.

- [14] T. Ichimura, K. Fujita, T. Yamaguchi, M. Hori, M. Lalith, N. Ueda, AI with Super-Computed Data for Monte Carlo Earthquake Hazard Classification, Research Poster for SC17: International Conference for High Performance Computing, Networking, Storage and Analysis, 2017.
- [15] C. Uphoff, S. Rettenberger, M. Bader, E. H. Madden, T. Ulrich, S. Wollherr, A. Gabriel, Extreme scale multi-physics simulations of the tsunamigenic 2004 sumatra megathrust earthquake, In Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis (SC '17), 2017.
- [16] T. Ichimura, K. Fujita, S. Tanaka, M. Hori, M. Lalith, Y. Shizawa, H. Kobayashi, Physics-based urban earthquake simulation enhanced by 10.7 BlnDOF x 30 K time-step unstructured FE non-linear seismic wave simulation. Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis (SC'14). IEEE Press, pp. 15–26, 2014.
- [17] T. Ichimura, K. Fujita, P. E. B. Quinay, L. Maddeggedara, M. Hori, S. Tanaka, Y. Shizawa, H. Kobayashi, K. Minami, Implicit Nonlinear Wave Simulation with 1.08T DOF and 0.270T Unstructured Finite Elements to Enhance Comprehensive Earthquake Simulation. Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis (SC'15), ACM, 2015.
- [18] T. Ichimura, K. Fujita, T. Yamaguchi, A. Naruse, J. C. Wells, T. C. Schulthess, T. P. Straatsma, C. J. Zimmer, M. Martinasso, K. Nakajima, M. Hori, L. Maddeggedara, A fast scalable implicit solver for nonlinear time-evolution earthquake city problem on low-ordered unstructured finite elements with artificial intelligence and transprecision computing. Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis (SC'18). ACM, Article No. 49, 2018.
- [19] T. Ichimura, K. Fujita, K. Koyama, R. Kusakabe, K. Minami, H. Inoue, S. Nishizawa, M. Tsuji, T. Nishiki, M. Hori, L. Maddeggedara, N. Ueda, Fast Scalable Implicit Solver with Convergence of Physics-Based Simulation and Data-Driven Learning: Toward High-Fidelity Simulation with Digital Twin City, Research Poster for the International Conference on High Performance Computing, Networking, Storage and Analysis (SC'20), 2020.
- [20] K. Fujita, K. Katsushima, T. Ichimura, M. Horikoshi, K. Nakajima, M. Hori, L. Maddeggedara, Wave Propagation Simulation of Complex Multi-Material Problems with Fast Low-Order Unstructured Finite-Element Meshing and Analysis, Proceedings of the International Conference on High Performance Computing in Asia-Pacific Region (HPC Asia 2018), 2018.
- [21] J. M. Winget, T. J. Hughes, Solution algorithms for nonlinear transient heat conduction analysis employing element-by-element iterative strategies, *Computer Methods in Applied Mechanics and Engineering*, vol. 52(1–3), 1985.
- [22] J. H. Tu, C. W. Rowley, D. M. Luchtenburg, S. L. Brunton, J. N. Kutz, On dynamic mode decomposition: Theory and applications. *Journal of Computational Dynamics*, 1(2): 391–421, 2014.
- [23] I. Mezić, Analysis of fluid flows via spectral properties of the Koopman operator, *Annu. Rev. Fluid Mech.*, 2013.
- [24] R. T. Q. Chen, Y. Rubanova, J. Bettencourt, D. Duvenaud, Neural Ordinary Differential Equations, 32nd Conference on Neural Information Processing Systems (NeurIPS 2018), Montreal, Canada, 2018.
- [25] G. H. Golub, Q. Ye, Inexact conjugate gradient method with inner-outer iteration, *SIAM Journal on Scientific Computing*, vol. 21(4), pp. 1305–1320, 1997.
- [26] K. Fujita, K. Koyama, K. Minami, H. Inoue, S. Nishizawa, M. Tsuji, T. Nishiki, T. Ichimura, M. Hori, L. Maddeggedara, High-fidelity nonlinear low-order unstructured implicit finite-element seismic simulation of important structures by accelerated element-by-element method, *Journal of Computational Science*, <https://doi.org/10.1016/j.jocs.2020.101277>, 2020.
- [27] Supercomputer Fugaku, RIKEN Center for Computational Science [Online]. <https://www.r-ccs.riken.jp/en/postk/project>
- [28] T. Yoshida, Fujitsu High Performance CPU for the Post-K Computer, IEEE Hot Chips: A Symposium on High Performance Chips, San Jose, August 21, 2018.
- [29] Y. Ajima, et al., The Tofu Interconnect D, IEEE Cluster 2018, 2018.
- [30] Oakbridge-CX. [Online]. <https://www.cc.u-tokyo.ac.jp/en/supercomputer/obcx/system.php>
- [31] A. Kumar, S. Kottapalli, I. M. Steiner, B. Valentine, I. Hirsh, G. Vedaraman, L. P. Looi, M. Arafa, A. Rudoff, S. Mandava, B. Fahim, and S. A. Vora, Future Intel Xeon Scalable Processor (Codename: Cascade Lake-SP), Hot Chips 30, [Online]. https://www.hotchips.org/hc30/2conf/2.15_Intel_Cascade_Lake_HC30.Intel.Akhilesh.CLXCPU.Final.pdf
- [32] Big Data and Extreme-scale Computing. [Online]. <https://www.exascale.org/bdec/>
- [33] T. Yamaguchi, T. Ichimura, K. Fujita, M. Hori, L. Wijerathne, N. Ueda, Data-Driven Approach to Inversion Analysis of Three-dimensional Inner Soil Structure via Wave Propagation Analysis, Computational Science - ICCS 2020, Lecture Notes in Computer Science, 2020.