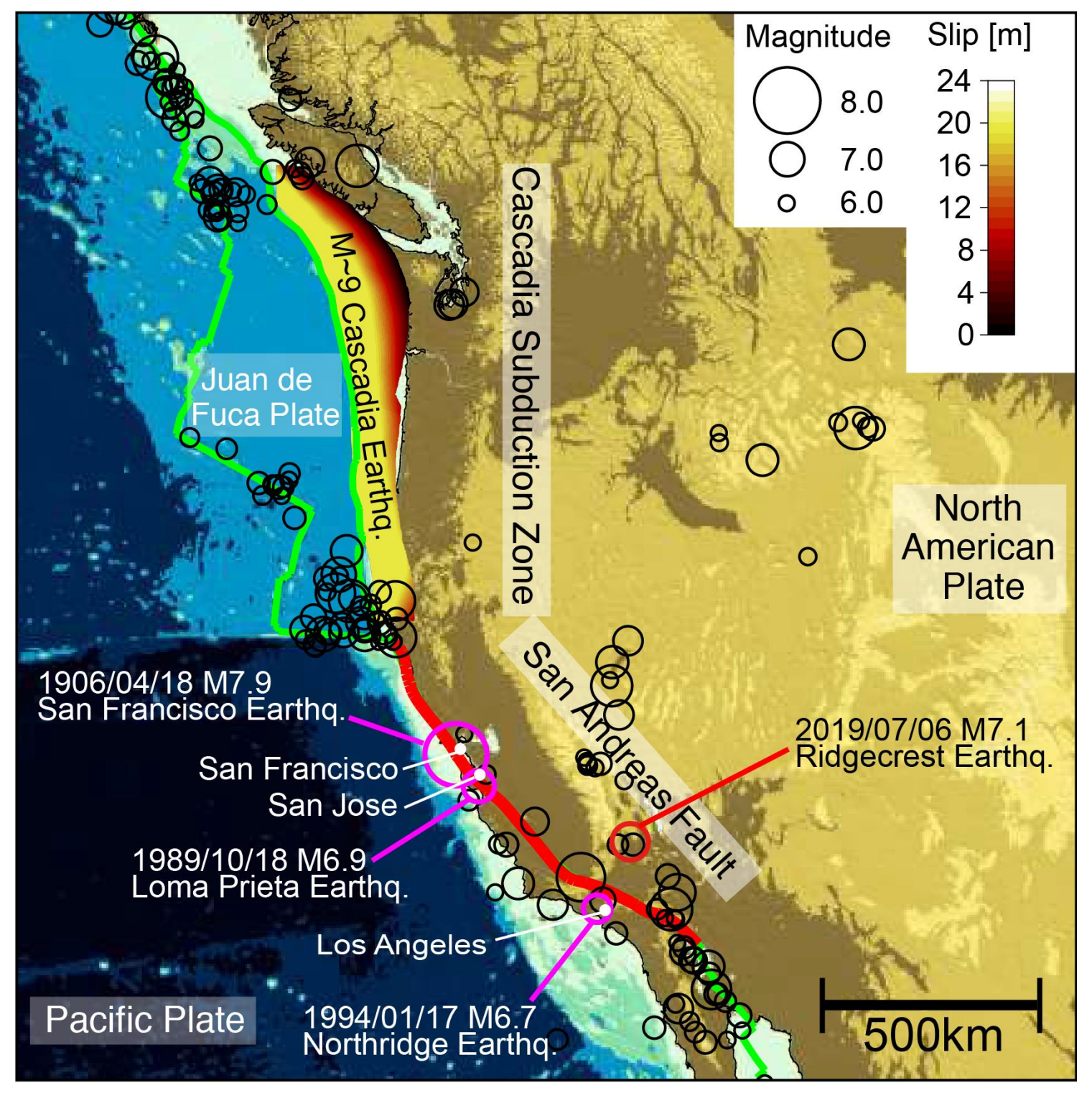
416-PFLOPS fast scalable implicit solver on low-ordered unstructured finite elements accelerated by 1.10-ExaFLOPS kernel with reformulated AI-like algorithm: For equation-based earthquake modeling

Tsuyoshi Ichimura[1,2,3], Kohei Fujita[1,3], Takuma Yamaguchi[1], Akira Naruse[4], Jack C. Wells[5], Christopher J. Zimmer[5],
Tjerk P. Straatsma[5], Takane Hori[6], Simone Puel[7], Thorsten W. Becker[7], Muneo Hori[8], and Naonori Ueda[2]

1. Earthquake Research Institute & Department of Civil Engineering, The University of Tokyo 2. Center for Advanced Intelligence Project, RIKEN 3. Center for Computational Science, RIKEN 4. NVIDIA corporation 5. Oak Ridge National Laboratory
6. Research Institute for Marine Geodynamics, JAMSTEC 7. Jackson School of Geosciences, The University of Texas at Austin 8. Center for Mathematical Science and Advanced Technology, JAMSTEC

Introduction

A better understanding of earthquake physics is a grand challenge because of the potential of large damage to the society and cities.



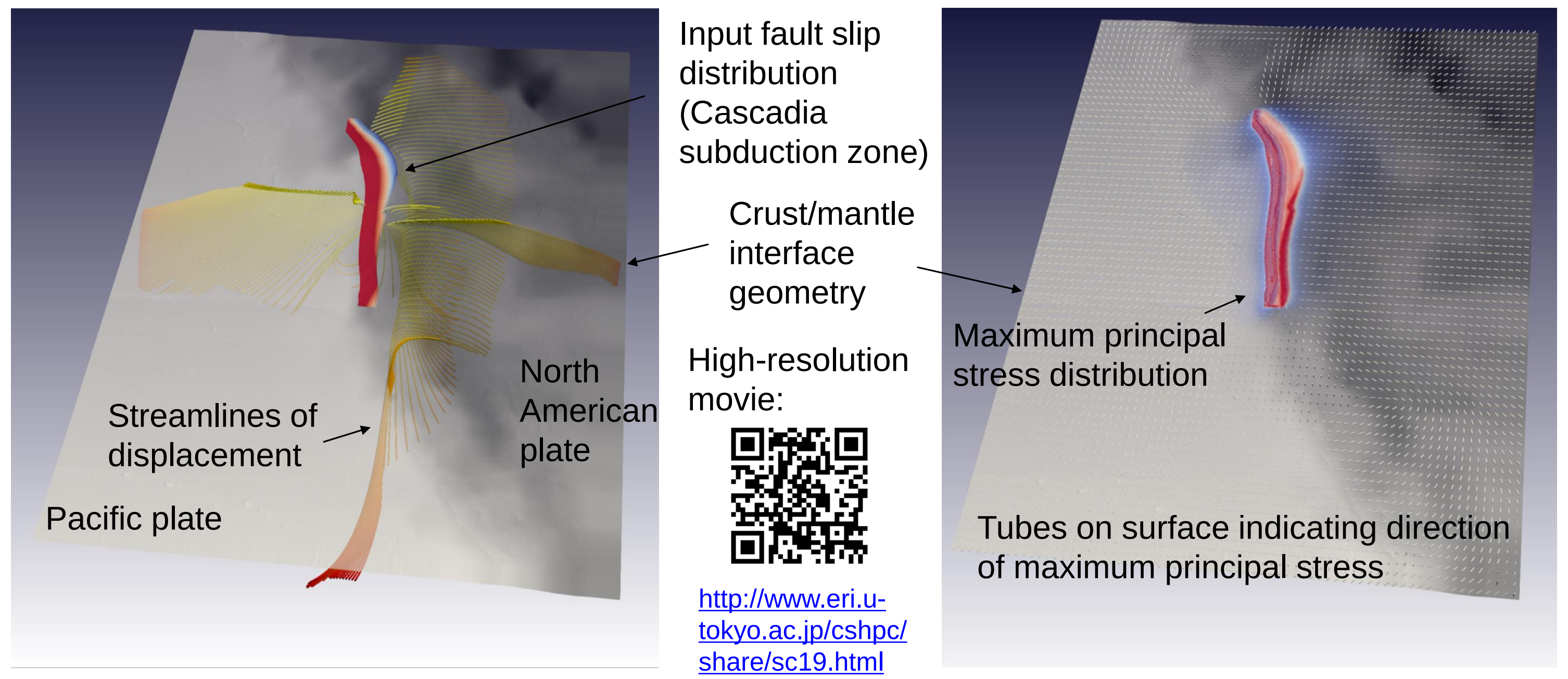
- A magnitude-8 earthquake is anticipated along the San Andreas Fault System, which could also be affected by the plate activity in the Cascadia Subduction Zone, where a magnitude-9 earthquake and a huge tsunami occurred in 1700.
- We expect probabilistic long-term earthquake forecasting to become possible by constructing a physics-based earthquake model with a realistic plate geometry and an assimilation of continuous data while solving the governing equations.
- The computation of governing equations with equation-based modeling considering the crust, plate, and fault geometry in high fidelity is required.
- Unstructured low-order implicit finite-element method is suitable for computing the visco-elastic-plastic time history on a heterogeneous 3D structure.

Huge cost in computing the large spatial- and temporal-scale problem hindered the realization of earthquake forecasting

- Many case analyses for large spatial- and temporal-scale problems in high fidelity are required (10^3 km × 10^3 km region; 10^2 year duration; km-scale resolution; $10^{2\text{-}3}$ iterations for assimilating data and considering uncertainty).
    - The visco-elastic-plastic computation cost is equivalent to solving $10^{10\text{-}12}$ degrees-of-freedom (DOF) elastic analysis for $10^{4\text{-}6}$ times for $10^{2\text{-}3}$ cases.
- At least a 50-fold speedup is required to conduct this analysis even when using the state-of-the-art solver on full Piz Daint.
    - State-of-the-art solver: a directive-based SC16 WACCPD solver [1] designed for P100 GPU based systems, which was developed based on the SC14 Gordon Bell Prize finalist solver [2].

Developed solver attains a 75-fold speedup from the state-of-the-art solver

- Accelerated an unstructured low-order implicit finite-element method using local and uniform expansions suitable for computation on Summit.
- Attained a significant speedup compared to the state-of-the-art solver.
    - Developed solver on full Summit corresponded to a 75-fold speedup from the state-of-the-art solver on full Piz Daint.
    - This speedup was very high considering the 215/25 = 8.6-fold difference in the FP64 system peak performance between Summit and Piz Daint.
    - This speedup is expected to be enough to conduct breakthroughs in science.



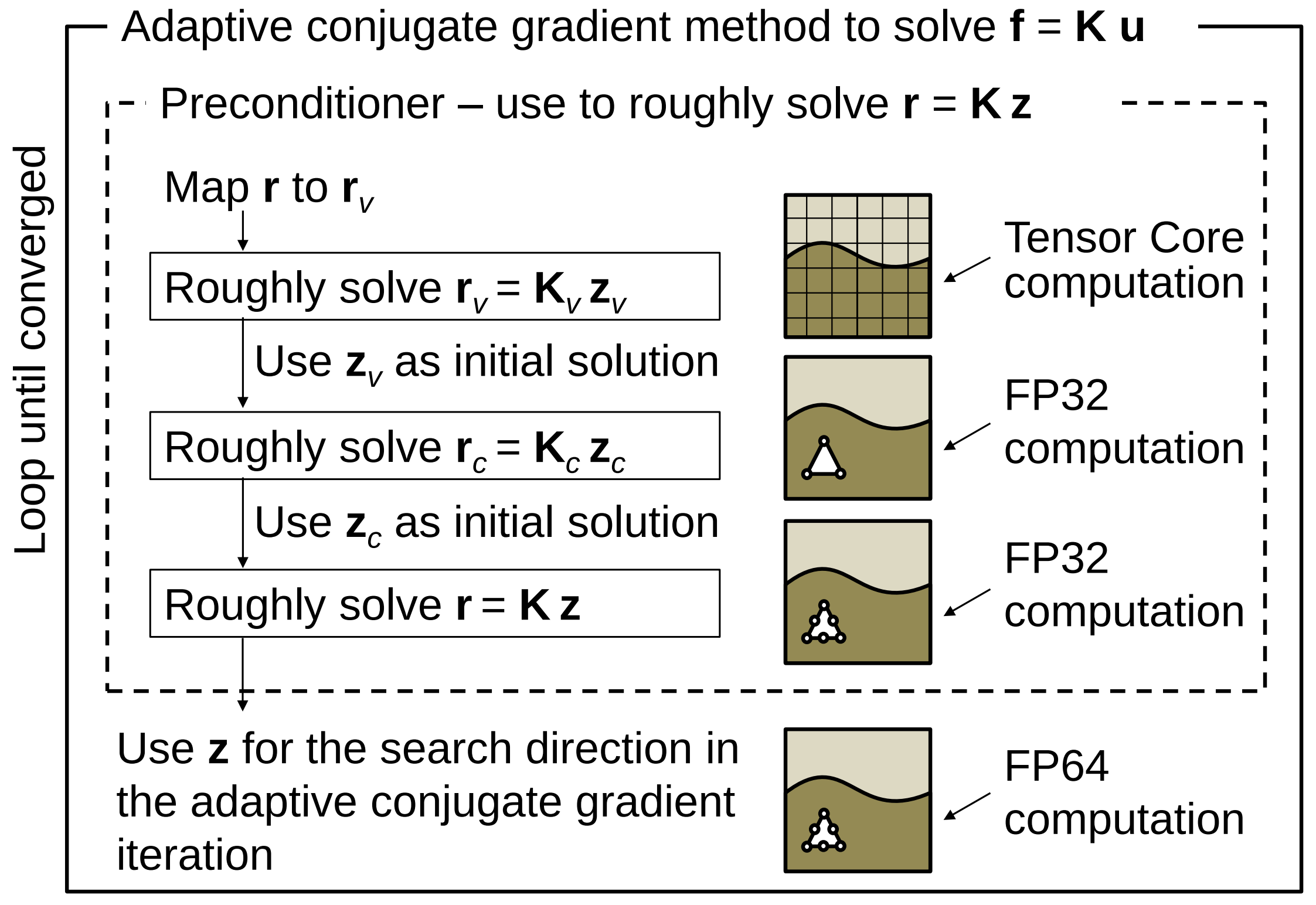
http://www.eri.u-tokyo.ac.jp/cshpc/share/sc19.html

Example of elastic deformation for a fault slip at the Cascadia Subduction Zone computed on Summit (a 1.49×10^{10} DOF model constructed for a 1944 km × 2646 km × 480 km region)

Reformulated AI-like Algorithm for Solving Huge Problems

Reformulate the solver algorithm with local and uniform basis expansions, such that the computation becomes similar to that used in AI training

- An unstructured finite-element analysis can hardly attain performance on Tensor Cores because of the basis functions with complex connectivity and varying strengths.
- We reformulate the solver algorithm such that the local and uniform basis expansions are used.
    - A local and uniform basis is used in the preconditioner of adaptive conjugate gradient method.
    - Major part of the solver becomes an iterative structured-grid solver with uniform element matrices.
    - The usage of matrix-matrix multiplication is enabled in most costly matrix-vector products.
    - Same final FP64 results with those of the standard solvers are attained.
- The solver convergence is improved by designing a special element with a high mapping accuracy.
    - A voxel element consisting of eight smaller sub-voxels is used.
    - The multiplication of the ie-th element matrix and vector becomes:


Adaptive conjugate gradient method to solve $\mathbf{f} = \mathbf{K}\,\mathbf{u}$

Preconditioner – use to roughly solve $\mathbf{r} = \mathbf{K}\,\mathbf{z}$

Map $\mathbf{r}$ to $\mathbf{r}_v$

Roughly solve $\mathbf{r}_v = \mathbf{K}_v \mathbf{z}_v$ — Tensor Core computation

Use $\mathbf{z}_v$ as initial solution

Roughly solve $\mathbf{r}_c = \mathbf{K}_c \mathbf{z}_c$ — FP32 computation

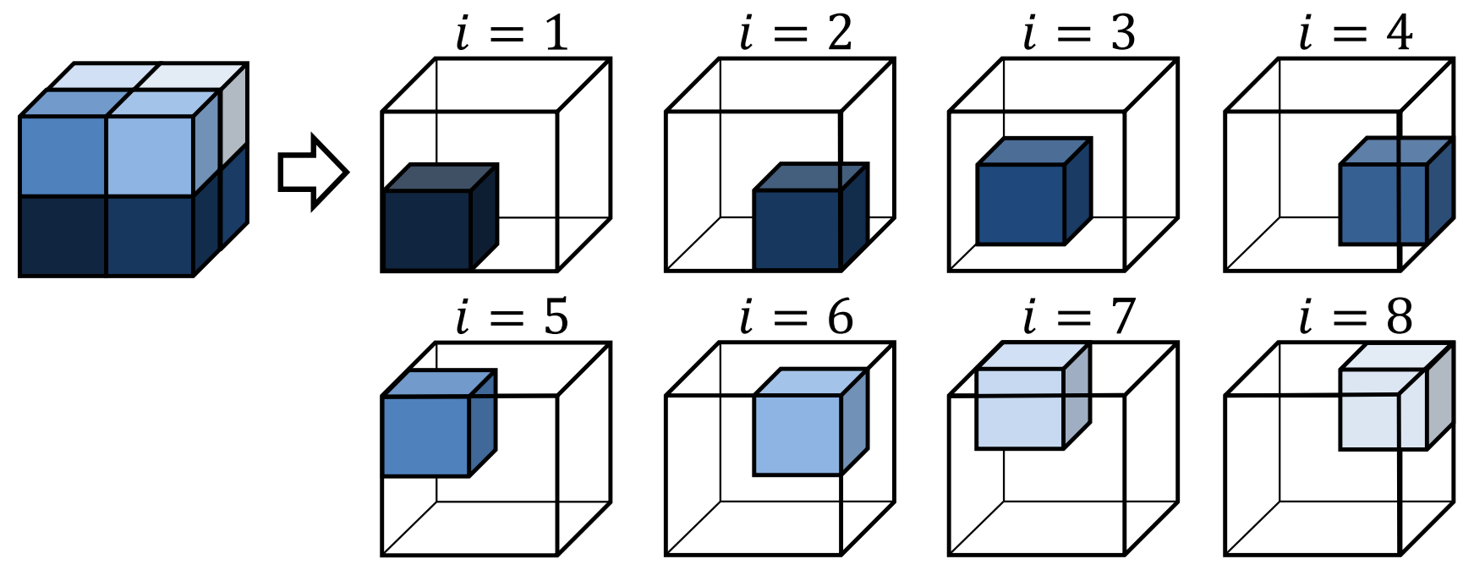
Use $\mathbf{z}_c$ as initial solution

Roughly solve $\mathbf{r} = \mathbf{K}\,\mathbf{z}$ — FP32 computation

Use $\mathbf{z}$ for the search direction in the adaptive conjugate gradient iteration — FP64 computation

Loop until converged

$$\mathbf{q}_{(ie)} = \mathbf{K}_{(ie)}\mathbf{p}_{(ie)} = \left[\sum_{i=1}^{8}\left(\alpha_{i(ie)}\mathbf{A}_i + \beta_{i(ie)}\mathbf{B}_i\right)\right]\mathbf{p}_{(ie)}$$



- $\alpha_{i(ie)}$ and $\beta_{i(ie)}$: scalar values corresponding to the material properties of the ie-th element
- $\mathbf{A}_i$ and $\mathbf{B}_i$: 24 × 24 matrices with constant values
- Use Tensor Cores for matrix-matrix multiplication $\mathbf{A}_i(\mathbf{p}_{(1)}, \mathbf{p}_{(2)}, \dots)$ and $\mathbf{B}_i(\mathbf{p}_{(1)}, \mathbf{p}_{(2)}, \dots)$

Efficient Implementation of Tensor Core

Special care required for using Tensor Cores for small matrices in equation-based modeling

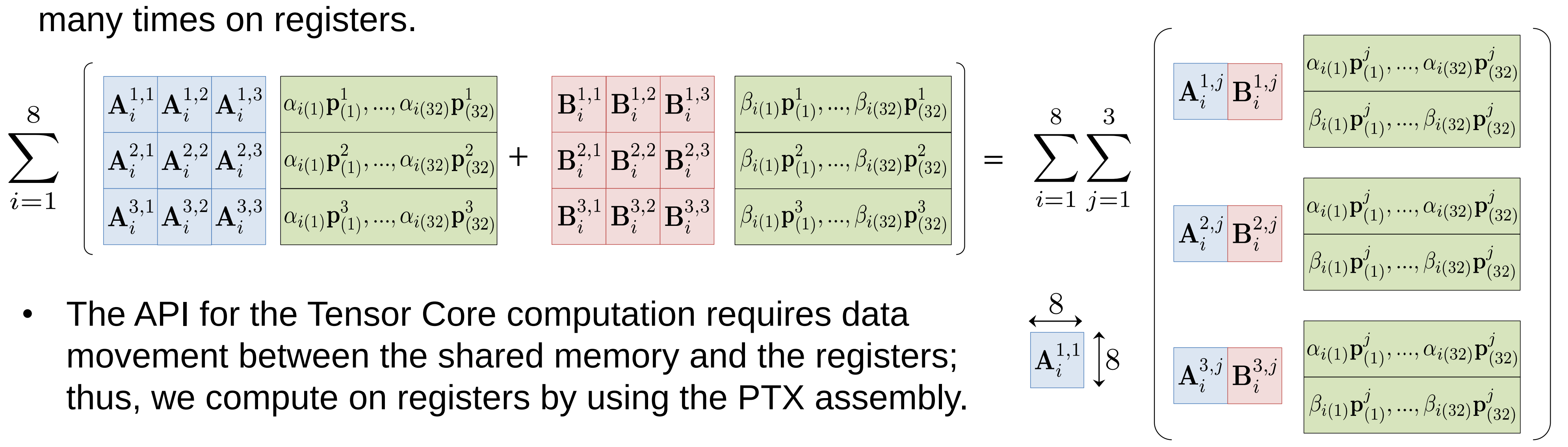
- Tensor Core is designed for large matrix-matrix multiplication with lower precision data types.
- The reduction of data access cost and prevention of loss of accuracy are required.

1. Ensuring convergence of the solver

- Although a low precision is allowed, a very low precision leads to preconditioner failure.
- The values of vectors $\mathbf{p}_{(ie)}$ and $\mathbf{q}_{(ie)}$ are normalized per element to improve accuracy.

2. Efficient data mapping of small matrices

- Frequent data movement leads to inefficiency.
- The computation of 32 elements is subdivided into 72 Tensor Core operations for reuse of matrix many times on registers.

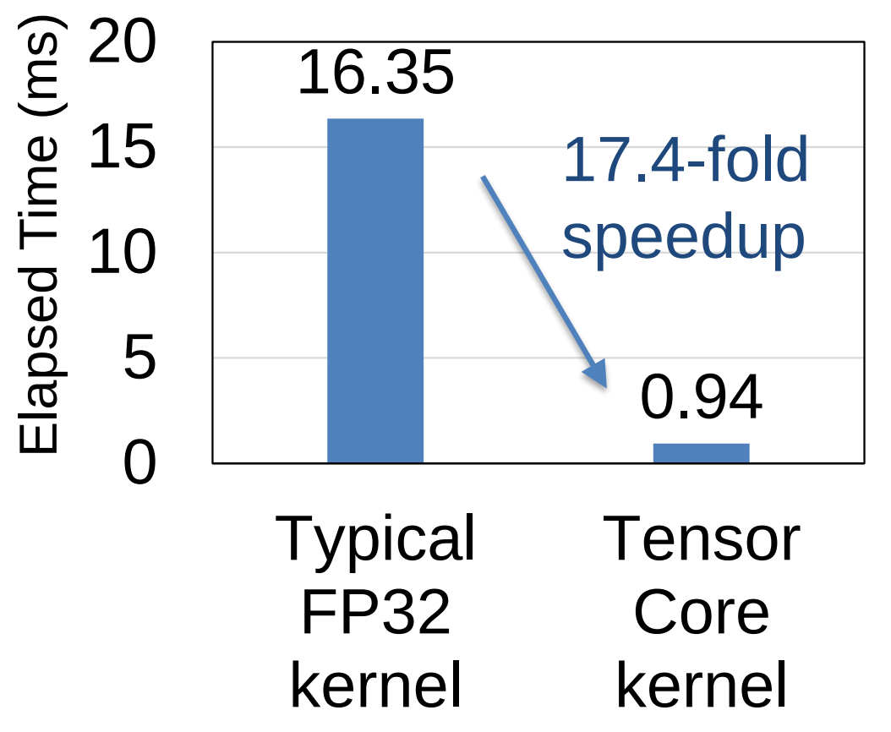

$$\sum_{i=1}^{8}\left(\begin{bmatrix}\mathbf{A}_i^{1,1}&\mathbf{A}_i^{1,2}&\mathbf{A}_i^{1,3}\\\mathbf{A}_i^{2,1}&\mathbf{A}_i^{2,2}&\mathbf{A}_i^{2,3}\\\mathbf{A}_i^{3,1}&\mathbf{A}_i^{3,2}&\mathbf{A}_i^{3,3}\end{bmatrix}\begin{bmatrix}\alpha_{i(1)}\mathbf{p}_{(1)}^{1},\dots,\alpha_{i(32)}\mathbf{p}_{(32)}^{1}\\\alpha_{i(1)}\mathbf{p}_{(1)}^{2},\dots,\alpha_{i(32)}\mathbf{p}_{(32)}^{2}\\\alpha_{i(1)}\mathbf{p}_{(1)}^{3},\dots,\alpha_{i(32)}\mathbf{p}_{(32)}^{3}\end{bmatrix}+\begin{bmatrix}\mathbf{B}_i^{1,1}&\mathbf{B}_i^{1,2}&\mathbf{B}_i^{1,3}\\\mathbf{B}_i^{2,1}&\mathbf{B}_i^{2,2}&\mathbf{B}_i^{2,3}\\\mathbf{B}_i^{3,1}&\mathbf{B}_i^{3,2}&\mathbf{B}_i^{3,3}\end{bmatrix}\begin{bmatrix}\beta_{i(1)}\mathbf{p}_{(1)}^{1},\dots,\beta_{i(32)}\mathbf{p}_{(32)}^{1}\\\beta_{i(1)}\mathbf{p}_{(1)}^{2},\dots,\beta_{i(32)}\mathbf{p}_{(32)}^{2}\\\beta_{i(1)}\mathbf{p}_{(1)}^{3},\dots,\beta_{i(32)}\mathbf{p}_{(32)}^{3}\end{bmatrix}\right)=\sum_{i=1}^{8}\sum_{j=1}^{3}\begin{pmatrix}\begin{bmatrix}\mathbf{A}_i^{1,j}&\mathbf{B}_i^{1,j}\end{bmatrix}\begin{bmatrix}\alpha_{i(1)}\mathbf{p}_{(1)}^{j},\dots,\alpha_{i(32)}\mathbf{p}_{(32)}^{j}\\\beta_{i(1)}\mathbf{p}_{(1)}^{j},\dots,\beta_{i(32)}\mathbf{p}_{(32)}^{j}\end{bmatrix}\\\begin{bmatrix}\mathbf{A}_i^{2,j}&\mathbf{B}_i^{2,j}\end{bmatrix}\begin{bmatrix}\alpha_{i(1)}\mathbf{p}_{(1)}^{j},\dots,\alpha_{i(32)}\mathbf{p}_{(32)}^{j}\\\beta_{i(1)}\mathbf{p}_{(1)}^{j},\dots,\beta_{i(32)}\mathbf{p}_{(32)}^{j}\end{bmatrix}\\\begin{bmatrix}\mathbf{A}_i^{3,j}&\mathbf{B}_i^{3,j}\end{bmatrix}\begin{bmatrix}\alpha_{i(1)}\mathbf{p}_{(1)}^{j},\dots,\alpha_{i(32)}\mathbf{p}_{(32)}^{j}\\\beta_{i(1)}\mathbf{p}_{(1)}^{j},\dots,\beta_{i(32)}\mathbf{p}_{(32)}^{j}\end{bmatrix}\end{pmatrix}$$

- The API for the Tensor Core computation requires data movement between the shared memory and the registers; thus, we compute on registers by using the PTX assembly.

Performance Measurement

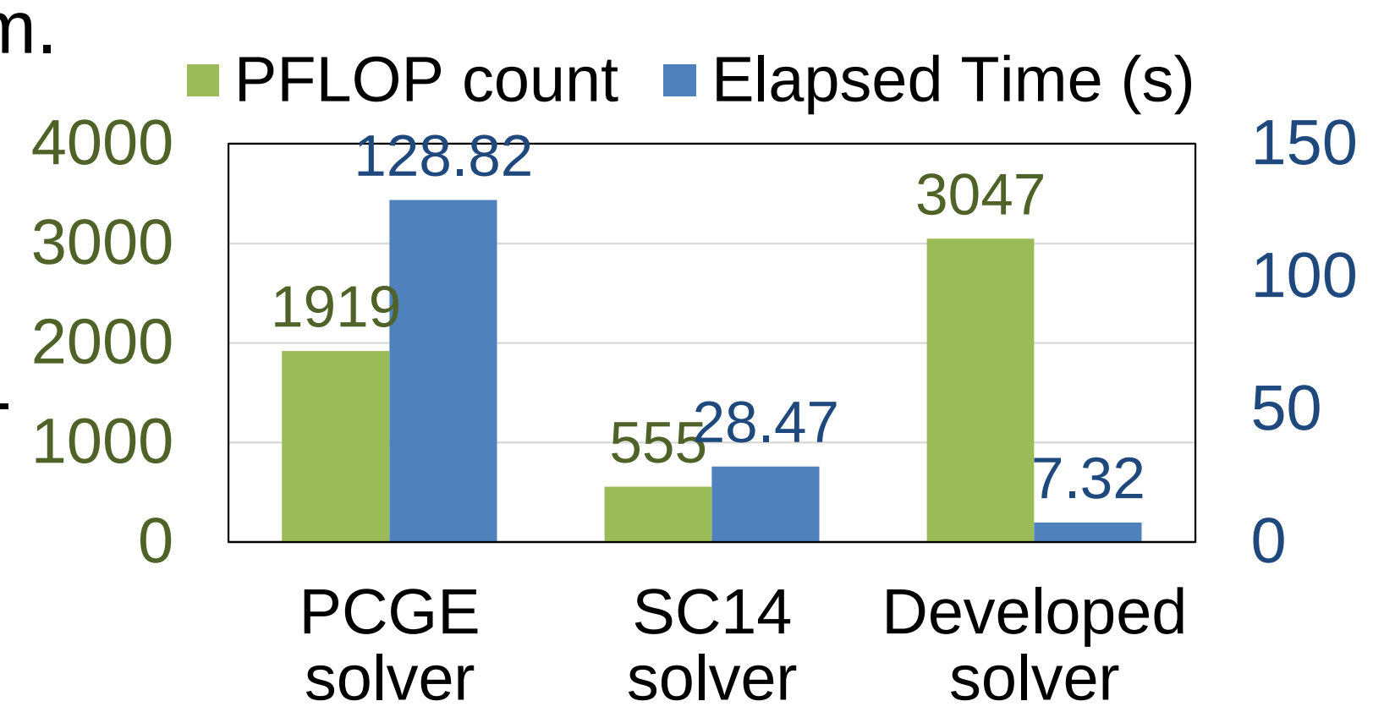
Performance of the matrix-vector kernel

- The performance of the matrix-vector multiplication kernel is measured using one NVIDIA V100 GPU on problem with 2,457,600 voxel elements.
- 49 TFLOPS was achieved using Tensor Cores, corresponding to 17.4-fold speedup from a typical FP32 implementation of the matrix-vector kernel.



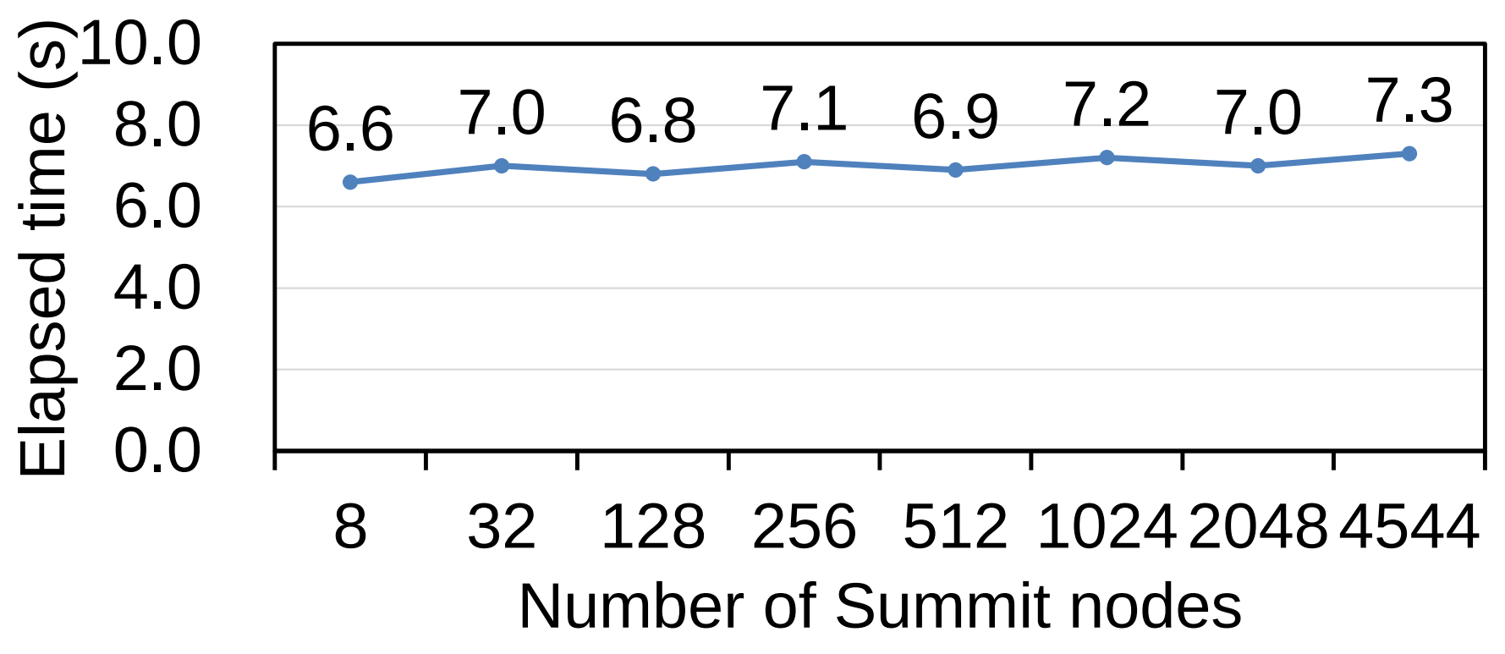
Time-to-solution of the whole solver

- Compared developed solver with a standard solver (PCGE; conjugate gradient solver with 3 × 3 block Jacobi conditioning) and the SC14 Gordon Bell Finalist solver [2] well-tuned for V100 GPUs.
- Measured performance using 4544 nodes of Summit (27,264 NVIDIA V100 GPUs) on a 1.67×10^{12} DOF problem.
- The developed method has increased the arithmetic count, but Tensor Cores accelerated the matrix-vector kernel, leading to 3.89-fold speedup from the SC14 solver.



Weak scaling on Summit

- The developed solver attains a high scalability of 90.5% from eight to 4544 nodes.
- Led to 416 PFLOPS and 1.10 ExaFLOPS for the whole solver and the matrix-vector kernel, respectively, on the nearly full system (4544 nodes).



Summary and Future Prospects

- An equation-based earthquake modeling algorithm is transformed to an algorithm suitable for high-performance hardware originally designed for AI.
- High performance and scalability on full Summit are achieved.
- Our approach using local and uniform expansions is applicable to other problems according to the target computer architecture characteristics.
- We plan to use the developed method to enable long-term earthquake forecasting, which is expected to advance earthquake disaster mitigation.

Acknowledgments

Our results were obtained using the Summit at Oak Ridge Leadership Computing Facility, a US Department of Energy, Office of Science User Facility at Oak Ridge National Laboratory (ORNL). We thank Yukihiko Hirano (NVIDIA) for coordination of the collaborative research project. We thank Christopher B. Fuson, Don E. Maxwell, Oscar Hernandez, Scott Atchley, Veronica Melesse-Vergara (ORNL), Jeff Larkin, Stephen Abbott (NVIDIA), Lixiang Luo (IBM), Richard Graham (Mellanox Technologies) for generous support concerning use of Summit. We thank Lalith Maddegedara (The University of Tokyo) for support and discussion at the Earthquake Research Institute. We thank Noda Tomoyuki and Hikaru Inoue (Fujitsu Limited) for support in program development. We acknowledge support from Post K computer project (Priority Issue 3 - Development of integrated simulation systems for hazards and disasters induced by earthquakes and tsunamis: proposal numbers hp180217, hp190177), and Japan Society for the Promotion of Science (18H05239, 26249066, 25220908, and 17K14719). Simone Puel and Thorsten W. Becker acknowledge support from NSF-EAR 1722680.

References

[1] Fujita, Kohei, et al. “Acceleration of Element-by-Element Kernel in Unstructured Implicit Low-order Finite-element Earthquake Simulation using OpenACC on Pascal GPUs,” WACCPD, 2016.
[2] Ichimura, Tsuyoshi, et al. ’’Physics-based urban earthquake simulation enhanced by 10.7 BlnDOF × 30 K time-step unstructured FE non-linear seismic wave simulation,’’ SC’14, 2014.