Extreme Scale Multi-Physics Simulations of the Tsunamigenic 2004 Sumatra Megathrust Earthquake

Carsten Uphoff
Sebastian Rettenberger
Michael Bader
uphoff@in.tum.de
rettenbs@in.tum.de
bader@in.tum.de
Technical University of Munich
Boltzmannstr. 3
85748 Garching, Germany

Elizabeth H. Madden
Thomas Ulrich
Stephanie Wollherr
Alice-Agnes Gabriel
madden@geophysik.uni-muenchen.de
ulrich@geophysik.uni-muenchen.de
wollherr@geophysik.uni-muenchen.de
gabriel@geophysik.uni-muenchen.de
Ludwig-Maximilians-Universität München
Theresienstr. 41
80333 Munich, Germany

ABSTRACT

We present a high-resolution simulation of the 2004 Sumatra-Andaman earthquake, including non-linear frictional failure on a megathrust-splay fault system. Our method exploits unstructured meshes capturing the complicated geometries in subduction zones that are crucial to understand large earthquakes and tsunami generation. These up-to-date largest and longest dynamic rupture simulations enable analysis of dynamic source effects on the seafloor displacements.

To tackle the extreme size of this scenario an end-to-end optimization of the simulation code SeisSol was necessary. We implemented a new cache-aware wave propagation scheme and optimized the dynamic rupture kernels using code generation. We established a novel clustered local-time-stepping scheme for dynamic rupture. In total, we achieved a speed-up of 13.6 compared to the previous implementation. For the Sumatra scenario with 221 million elements this reduced the time-to-solution to 13.9 hours on 86,016 Haswell cores. Furthermore, we used asynchronous output to overlap I/O and compute time.

CCS CONCEPTS

• Applied computing → Earth and atmospheric sciences;

KEYWORDS

ADER-DG; earthquake simulation; tsunami coupling; dynamic rupture; petascale performance; hybrid parallelization; local time stepping; asynchronous output

ACM Reference format:

Carsten Uphoff, Sebastian Rettenberger, Michael Bader, Elizabeth H. Madden, Thomas Ulrich, Stephanie Wollherr, and Alice-Agnes Gabriel. 2017. Extreme Scale Multi-Physics Simulations of the Tsunamigenic 2004 Sumatra

Megathrust Earthquake. In *Proceedings of SC17, Denver, CO, USA, November 12–17, 2017*, 16 pages.

DOI: 10.1145/3126908.3126948

1 INTRODUCTION

“Megathrust” earthquakes in subduction zones, caused by oceanic lithosphere sliding into the mantle beneath an overriding tectonic plate, have released most of the seismic energy over the last century. No other type of tectonic activity is known to produce earthquakes that exceed moment magnitudes M_w of 9 and trigger teletsunamis traveling whole oceans.

A particularly devastating example is the M_w 9.1 2004 Sumatra-Andaman Earthquake and Indian Ocean Tsunami. The Sumatra earthquake ruptured the greatest fault length of any recorded earthquake and triggered a series of tsunamis, killing up to 280,000 people in 14 countries, and even displaced Earth’s North Pole by 2.5 cm.

For the coming decades, there is no realistic hope to reliably predict earthquakes, such that the method of choice to mitigate earthquake- and tsunami-related damage to our societies is to forecast seafloor displacement and strong ground motions for likely earthquake scenarios. Dynamic rupture simulations combine non-linear frictional failure and seismic wave propagation in a multi-physics manner to produce physics-based forecasts [e.g. 5, 20, 30, 38, 65] and provide insight into the poorly understood fundamental processes of earthquake faulting [e.g. 2, 4, 29, 32, 33, 68].

A correct representation of subduction zone complexity is crucial, but, in concurrence with the giant spatial extent and temporal duration to be resolved, extremely challenging from a numerical point of view. Simulations must capture slip and seismic waves occurring for hundreds of seconds along hundreds of kilometers but at the same time resolve the meter-scale physical processes taking place at the rupture tip of the earthquake source.

A recent rise of 3D HPC earthquake simulation software [e.g. 18, 43, 48, 49, 72, 75, 76] allows for the simulation of various aspects of earthquake scenarios and enables researchers to answer geophysical questions complicated by the lack of sufficiently dense

Permission to make digital or hard copies of part or all 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 third-party components of this work must be honored. For all other uses, contact the owner/author(s).

SC17, Denver, CO, USA

© 2017 Copyright held by the owner/author(s). 978-1-4503-5114-0/17/11...\$15.00
DOI: 10.1145/3126908.3126948

Figure 1: Left: Tectonic setting of the Sumatra subduction zone including past earthquakes and their magnitudes adapted from [79]. Middle: Unstructured tetrahedral mesh of the modeling domain, including refinement to resolve high-frequency wave-propagation and frictional failure on the fault system. The purple box marks high-resolution (30 arc-second) topography. Blue curves are splay fault traces. Right: Complex 3D geometry in the region marked by the black dotted box in middle figure. The curved megathrust is intersecting bathymetry, as are the 3 splays: one forethrust and two backthrusts. The subsurface consists of horizontally layered continental crust and subducting layers of oceanic crust. Each layer is characterized by a different wave speed and thus requires a different mesh resolution. Red curves mark the megathrust trace in all figures.

data situations. The state-of-the art includes finite difference, finite element and spectral element methods, as well as implicit and explicit time-stepping to solve the elastic wave equations.

We present here the very first dynamic rupture scenario of the 2004 Sumatra earthquake. We resolve the full frictional sliding process in a complex fault network as well as the seismic wave field with frequency content up to 2.2 Hz in the to-date longest (500 s) and largest (1500 km) physics-based earthquake simulation (see Figure 1). We validate the large-scale, high-resolution scenario by geodetic, seismological and tsunami observations. The resulting rupture dynamics shed new light on the activation and importance of splay faults. The modelled high-resolution seafloor displacement will serve as a crucial tool to study time-dependent tsunami generation and propagation on a natural scale.

Resolving the extreme scales of this scenario became feasible only after an end-to-end optimization of the high-performance open-source earthquake simulation software SeisSol for massively parallel HPC infrastructures. In the following, we refer to this newly optimized version as Shaking Corals. The method applies an Arbitrary high-order DERivative Discontinuous Galerkin (ADER-DG) algorithm that does not require simplified geometries nor filtering nor smoothing of the output data as necessary in other modeling approaches [e.g. 9, 74]. However, considering the full complexity of subduction zone geometries leads inevitably to huge differences in element sizes. Implementing local time stepping for all parts, including dynamic rupture, enabled a speedup of 6.8 when compared to global time stepping. The optimization of further "bottlenecks" in time-to-solution and time-to-science, such as the processing of large files via asynchronous I/O, allows for unprecedented temporal and spatial expansion of dynamic rupture simulations.

The largest production run on an unstructured tetrahedral mesh with 221 million elements, requiring up to 3.3 million time steps for a fraction of elements, was computed in 13.9 hours at 0.94 PFLOPS

sustained performance on all 86,016 Haswell cores of SuperMUC. Additionally, we tested local time stepping with dynamic rupture on Shaheen and Cori in our scaling studies. The results indicate that the simulation time can be further reduced to 8.2 hours on 3072 Haswell nodes of Shaheen, running at 1.59 PFLOPS. On 512 Knights Landing nodes of Cori, we measured 467 TFLOPS with a speed-up of 1.28 compared to 512 nodes of Shaheen. From our results, we extrapolate that the full Sumatra scenario would take 7 days and 19 hours on SuperMUC without the presented optimizations and without local time stepping with dynamic rupture.

We demonstrate the essential need for end-to-end-optimization and petascale performance to realize realistic simulations on the extreme scales of great, and extremely devastating, subduction zone earthquakes.

2 DYNAMIC RUPTURE MODELING OF SUBDUCTION ZONE EARTHQUAKES

Many previous studies of earthquake source physics focus on purely vertical strike-slip faulting, representing for example parts of the San Andreas Fault [e.g. 75] or the Landers fault system [e.g. 43]. However, 3D subduction zones are characterized by curved thrust fault geometries that merge with the bathymetry under very shallow angles of narrow subduction wedges. Additionally, complicated networks of fault branches at high angles to the megathrust in the overriding and oceanic plates, termed splay and outerrise faults, potentially generate strong gaining effects of vertical seafloor displacements, making tsunami generation more likely. It was shown that asymmetric fault geometry results in asymmetric near-source ground motion, which increases with decreasing angle even of linear fault segments [23, 63, 69]. Furthermore, if a rupture front

breaks the surface, it strongly excites seismic waves [7] and interacts with reflected waves from the free surface [46] across the uppermost parts of the fault system.

Few dynamic rupture models have tackled subduction zone earthquakes so far, and most of these use simplified fault planes dipping at large angles [24, 46, 62, 85]. Studies investigating the effect of more realistic geometries on shallow slip and seafloor displacement were restricted to 2D [53]. A notable exception is the 3D Tohoku-Oki earthquake scenario presented in [34, 35] that resolves seismic waves up to 0.2 Hz and includes non-planar megathrust geometry. It does not, however, include topography or fault networks. Realistic model setups should include bathymetry, 3D subsurface structure, including subducting layer interfaces and depth-dependent rheology, and intersecting fault geometries with appropriate stress and initial frictional parameterization (Figure 1). All of these factors contribute to the amount and area of vertical uplift of the seafloor, which in turn determines the generation and destructive propagation of the tsunamis that often accompany great subduction zone earthquakes. The role of faults at high angles to the megathrust, including splay and potentially outerrise faults, is of particular interest. This is because their failure changes the geometry of the surface displacements as the rupture front approaches the seafloor, and enhances the potential for generation of a tsunami [45, 67].

In this work, we apply the ADER-DG method [28, 50], which enables high-order accuracy in space and time: seismic waves are accurately propagated over large distances with minimal dispersion errors [26, 52]. ADER-DG is intrinsically dissipative and removes frequencies unresolved by the mesh, caused e.g. by mesh coarsening or dynamic rupture, without affecting longer and physically meaningful wavelengths. DG methods use a (typically high-order) element-local polynomial representation in each element, but restrict information exchange to face-adjacent elements via numerical fluxes based on solutions of the Generalized Riemann Problem. This leads to strongly element-local, compute-bound schemes. Massively parallel implementations can thus achieve high percentages of peak performance, as demonstrated for regional and global seismic wave propagation [14, 15] and dynamic rupture simulation [43]. Integrating earthquake rupture into elastodynamic solvers poses numerical challenges, as the discontinuity of displacements across the fault has to be treated accurately by the scheme. Finite difference, finite element and spectral element methods suffer from spurious high-frequency oscillations, which contaminate the solution over all space-time scales [25] and require numerical regularization [e.g. 4]. In the method presented here, frictional sliding is incorporated as an internal boundary condition, altering the exact solution of an exact Riemann solver (Godunov flux) [56, 84], in contrast to the traction at split-node approach commonly used [3, 19, 21]. The Godunov-flux approach leads to highly accurate solutions without spurious oscillations – see [22, 70] for details.

ADER time stepping, as all explicit methods, needs to enforce a CFL condition, which for ADER is half of the stability limit of Runge–Kutta DG schemes, i.e. $C/(2N + 1) \times (2r_{in}/c_p)$ with r_{in} being the element in-sphere radius, N being the polynomial degree of the basis functions, $C = 0.5$ empirically, c_p = local P-wave speed (fastest wave speed) [22]. A detailed analysis of the linear stability

properties of the ADER-DG schemes via a von-Neumann analysis was published in [28].

Geometric complexity, especially faults intersecting at shallow angles, may lead to “bad” elements with small in-sphere radii. To avoid that all elements are propagated at the time step of the costliest element, local time stepping (LTS) methods are required. In terms of the numerical scheme, LTS is a straightforward extension for ADER time integration [27]. In SeisSol, a cluster-based implementation of LTS was introduced to ensure scalability and high performance on supercomputers [13]. Speed-ups of up to 4.5 for wave propagation in a geometrically complex volcano model were achieved (mesh with 99.8 million elements). Rietmann et al. [72] developed an LTS scheme for second-order Newmark time stepping in SPECFEM3D [17], which led to a speed up of 3.9 for a simulation of the Tohoku-Oki earthquake on a mesh of 7.5 million elements.

3 END-TO-END OPTIMIZATION

The seismic wave equation in velocity-stress formulation is

$$\begin{aligned} \frac{\partial}{\partial t} \sigma_{ij} - \lambda \delta_{ij} \frac{\partial}{\partial x_k} u_k - \mu \left(\frac{\partial}{\partial x_j} u_i + \frac{\partial}{\partial x_i} u_j \right) &= 0, \\ \rho \frac{\partial}{\partial t} u_i - \frac{\partial}{\partial x_j} \sigma_{ij} &= 0, \end{aligned} \quad (1)$$

where σ is the symmetric stress tensor containing six independent components and u is the particle velocity vector with velocities in x , y , z -direction respectively. (Note that we use tensor notation for subscripts, i.e. an index appearing twice implies summation.) Density ρ and Lamé parameters λ and μ describe the material properties. Regrouping terms we use the compact form

$$\frac{\partial q_p}{\partial t} + A_{pq} \frac{\partial q_q}{\partial x} + B_{pq} \frac{\partial q_q}{\partial y} + C_{pq} \frac{\partial q_q}{\partial z} = 0,$$

where A, B, C are the 9×9 Jacobian matrices of Equation (1) and $q = (\sigma_{xx}, \sigma_{yy}, \sigma_{zz}, \sigma_{xy}, \sigma_{yz}, \sigma_{xz}, u, v, w)$ is the vector of quantities.

3.1 Wave propagation scheme

Equation (1) is numerically solved using an ADER-DG scheme [26, 51]. For the spatial discretization, we approximate q by a polynomial of degree N . The polynomial’s coefficients with respect to basis $\phi_1, \dots, \phi_{B_N}$ are stored in the $B_N \times 9$ matrix Q for every quantity, where $B_N = \frac{1}{6}(N+1)(N+2)(N+3)$. The m -th tetrahedron is advanced in time at time step n using the following update scheme:

$$\begin{aligned} Q_{n+1}^{(m)} - Q_n^{(m)} &= M^{-1} K^\xi \mathcal{I}_n^{(m)} \left(A^{*,(m)} \right)^T \\ &+ M^{-1} K^\eta \mathcal{I}_n^{(m)} \left(B^{*,(m)} \right)^T \\ &+ M^{-1} K^\zeta \mathcal{I}_n^{(m)} \left(C^{*,(m)} \right)^T \end{aligned} \quad (2)$$

$$- \sum_{i=1}^4 \frac{|S_i|}{|J|} M^{-1} F^{-,i} \mathcal{I}_n^{(m)} \left(A^{+,i,(m)} \right)^T \quad (3)$$

$$- \sum_{i=1}^4 \frac{|S_i|}{|J|} M^{-1} F^{+,i,j_i^{(m)}, h_i^{(m)}} \mathcal{I}_n^{(m_i)} \left(A^{-,i,(m)} \right)^T, \quad (4)$$

where M is the mass matrix, K_ξ, K_η, K_ζ the stiffness matrices and F^-, F^+ the flux matrices of size $B_N \times B_N$. These matrices only

depend on the basis functions and can be precomputed. The 9×9 matrices $A^*, B^*, C^*, A^-,$ and A^+ depend on material properties and on the underlying element's geometry. For a detailed description of these matrices we refer to [51]. As the geometry varies for each element, we premultiply them with all constant factors, such as sign, face Jacobian determinant $|S_i|$, or volume Jacobian determinant $|J|$ (and we store them already transposed). Note that due to weak coupling via numerical fluxes in Terms (3) and (4), information from adjacent elements is only required in Term (4).

$\mathcal{I}_n^{(m)}$ in Terms (2) to (4) is the time-integrated Taylor expansion

$$\mathcal{I}_n^{(m)} = \sum_{k=0}^N \frac{\Delta t^{k+1}}{(k+1)!} \frac{\partial^k}{\partial t^k} Q_n^{(m)}, \quad (5)$$

where the time derivatives $\partial^k Q / \partial t^k$ are calculated using a discrete Cauchy-Kovalewski procedure [51]:

$$\begin{aligned} \frac{\partial^{k+1} Q_n^{(m)}}{\partial t^{k+1}} = & -M^{-1} (K^\xi)^T \frac{\partial^k Q_n^{(m)}}{\partial t^k} (A^{*,(m)})^T \\ & -M^{-1} (K^\eta)^T \frac{\partial^k Q_n^{(m)}}{\partial t^k} (B^{*,(m)})^T \\ & -M^{-1} (K^\zeta)^T \frac{\partial^k Q_n^{(m)}}{\partial t^k} (C^{*,(m)})^T \end{aligned} \quad (6)$$

Note that this is a crucial ingredient for local time stepping, as it allows the computation of arbitrary time integrals, only limited by each element's local CFL condition.

3.2 Code generation

Terms (2) to (4) and Equation (6) illustrate that the bulk computational load of the ADER-DG scheme is due to matrix chain products, where the matrix multiplications are of small size. We therefore generate code calling the libxsmm library [44] for every matrix-matrix multiplication. It was shown for several applications that such an approach leads to kernels that achieve a high fraction of available peak performance [42, 44, 47].

Each matrix chain product is automatically analyzed and optimized in the sense that unnecessary computation of zero blocks in sparse matrices are removed, the optimal matrix chain order in terms of number of operations is determined, and zero padding is added for aligned memory access [86]. Finally, all matrices are stored in either dense, sparse, or block partitioned memory layout, where each block may again be sparse or dense. We employ auto-tuning to determine the layouts that minimize time to solution.

3.3 Parallel programming model

We use a carefully tuned hybrid MPI+OpenMP parallelization [13, 43]. A custom scheduler generates work-items, giving priority to work-items that trigger inter-node communication. Inside a work-item, every loop over either elements or dynamic rupture faces is distributed to all threads. Communication is asynchronous using only MPI_Isend, MPI_Irecv, and MPI_Test. In order to ensure MPI progression, one may enable a dedicated communication thread.

3.4 Optimization of wave propagation

In this section we discuss techniques to further improve the matrix multiplication kernels.

Figure 2: Speedup of the neighbor flux kernel due to decomposition of the flux matrices and auto-tuning. We compare the baseline version (BL), without flux matrix decomposition, to our Shaking Corals (SC) version, with flux matrix decomposition. Furthermore, we show the effects of having only 16 flux matrices instead of 48 (suffix 16) and the effects of omitting prefetching (suffix nopf) on HSW (dual-socket) and KNL for discretization orders O4 to O7.

3.4.1 Flux matrix decomposition. A performance problem connected with Term (4) stems from the growth of the flux matrices F^+ with $O(N^6)$, which leads to eviction of the flux matrices from low level caches. We show that expressing the flux matrices via chain products of smaller matrices not only prevents cache eviction, but also reduces the required number of operations.

The flux matrices in Term (4) are defined as in [26]

$$F_{kl}^{+,i,j,h} = \int_{T_2} \phi_k \left(\xi^{(i)}(\chi, \tau) \right) \times \phi_l \left(\xi^{(j)}(\tilde{\chi}^{(h)}(\chi, \tau), \tilde{\tau}^{(h)}(\chi, \tau)) \right) d\chi d\tau, \quad (7)$$

where T_2 is the reference triangle. The linear function $\xi^{(i)} : T_2 \mapsto T_3$, T_3 being the reference tetrahedron, is a parameterization of the i -th face. The linear function $(\tilde{\chi}^{(h)}, \tilde{\tau}^{(h)}) : T_2 \mapsto T_2$ amount for rotation due to the h -th vertex of the neighboring face lying on the first vertex of the local face [26]. As ϕ_k is a polynomial of degree $\leq N$ and $\xi^{(i)}$ is linear, the function $\phi_k \circ \xi^{(i)}$ is element of the space of polynomials on T_2 with degree $\leq N$, which we abbreviate with $\mathbb{P}_{2,N}$. Hence, $\phi_k \circ \xi^{(i)}$ can be represented with a two-dimensional basis $\tilde{\phi}_1, \dots, \tilde{\phi}_{\tilde{B}_N}$, where $\tilde{B}_N = \frac{1}{2}(N+1)(N+2)$. We define R^i as

$$\tilde{M}_{ml} R_{kl}^i = \int_{T_2} \phi_k \left(\xi^{(i)}(\chi, \tau) \right) \tilde{\phi}_m(\chi, \tau) d\chi d\tau,$$

where $\tilde{M}$ is the mass matrix of the basis $\tilde{\phi}$. The sum $R_{kl}^i \tilde{\phi}_l$ can be readily identified as the projection of $\phi_k \circ \xi^{(i)}$ on $\mathbb{P}_{2,N}$. Hence,

$$\phi_k \circ \xi^{(i)} = R_{kl}^i \tilde{\phi}_l, \quad (8)$$

because $\phi_k \circ \xi^{(i)}$ is an element of $\mathbb{P}_{2,N}$. Inserting Equation (8) into Equation (7) yields

$$F_{kl}^{+,i,j,h} = R_{km}^i R_{ln}^j \underbrace{\int_{T_2} \tilde{\phi}_m(\chi, \tau) \tilde{\phi}_n(\tilde{\chi}^{(h)}(\chi, \tau), \tilde{\tau}^{(h)}(\chi, \tau)) d\chi d\tau}_{=: f_{mn}^h}.$$

That is, $F^{+,i,j,h} = R^i f^h (R^j)^T$ in matrix notation, where $R^i \in \mathbb{R}^{B_N \times \tilde{B}_N}$ and $f^h \in \mathbb{R}^{\tilde{B}_N \times \tilde{B}_N}$.

From a computational point of view, decomposing the flux matrices is counter-intuitive, as we have to multiply five matrices instead of three in Term (4). However, once we use an optimal matrix chain multiplication order, for example

$$R^i \left(\left(f^h \left((R^j)^T I \right) \right) (A^-)^T \right),$$

we obtain $O(B_N \tilde{B}_N) = O(N^5)$ flops, instead of $O(B_N^2) = O(N^6)$ flops for the previous approach. In practice, the code generator automatically chooses the matrix chain order that minimizes the number of flops and we observe that we require fewer operations for typical production orders.

As another advantage we do not need to store a matrix $F^{+,i,j,h}$ for every combination of i, j , and h . Instead of these 48 large matrices, we only store the 11 smaller matrices R^i, f^h and $(R^j)^T$, which reduces the required memory from $O(N^6)$ to $O(N^5)$. Note that the 48 $F^{+,i,j,h}$ -combinations can be reduced to 16 by vertex ordering [73]. We discuss the impact of this option on performance in Section 3.4.3.

3.4.2 L2 Prefetching on KNL. The first KNL-optimized version of SeisSol revealed that L2 prefetching of the flux matrices in Term (4) is crucial to achieve highest performance [42]. For order 7, 16 full flux matrices already require 882 KB and 48 flux matrices even 2.6 MB of memory, which exceeds the size of the L2 cache. Thus, flux matrices constantly get evicted from the L2 cache. With the described new flux matrix decomposition we only require 165 KB, which is small enough for the 1 MB shared L2 cache of two KNL cores. Hence, we may omit prefetching of the flux matrices.

L2 prefetching is still necessary for data that is streamed from memory. In Term (4), we prefetch $\mathcal{I}_n^{(m_{i+1})}$ while computing the flux from the i -th face and $\mathcal{I}_n^{((m+1)_1)}$ when $i = 4$.

3.4.3 Single node performance. We measured the performance of our generated flux kernels on two different architectures:

HSW Dual-socket Intel Xeon E5-2697 v3, each with 14 cores, 2.6 GHz non-AVX base frequency, 2.2 GHz AVX base frequency, 32 KB per core L1d/i, 256 KB per core L2, 35 MB per socket L3. Frequency is capped at 2.2 GHz due to the computing center's energy policy.

KNL Single-socket Intel Xeon Phi 7250 with 68 cores, 1.2 GHz AVX base frequency, 1.5 GHz all-tile Turbo frequency, 32 KB per core L1d/i, 1 MB per tile L2, FLAT/QUADRANT mode.

Figure 2 shows the speed-up of the neighbor flux kernel for three different approaches: BL is the first KNL-optimized version of SeisSol, presented in [42], SC denotes our current version, Shaking Corals, and BL16 is the modified BL code using only 16 flux matrices instead of 48. This can be achieved by reordering the local

Figure 3: Single node performance of Shaking Corals (SC) and speed-up compared to the baseline version (BL) for wave propagation on a mesh with 100000 elements.

vertex indices according to a global vertex ordering [73]. On the KNL architecture, we additionally tested the influence of disabled prefetching, which is denoted by the suffix nopf.

On HSW, the difference in speed-up between the 48 flux matrices used by BL and the 16 used by BL16 is relatively small, which is expected as the flux matrices reside in the large L3 cache. In contrast, the SC version of the code benefits from the reduced number of flops giving a speed-up of up to 1.81 compared to BL. On KNL, BL16 is up to 1.64 times faster than BL, due to better L2 cache locality of the flux matrices. Our tests show that the flux matrix decomposition gives an even higher speed-up on KNL, up to 1.39 comparing SC to BL16 and up to 2.27 comparing SC to BL. Our tests additionally show that SC without prefetching of the flux matrices (SCnopf) performs much better than BLnopf because the smaller flux matrices used in SC are more likely to stay in L2 cache.

Figure 3 shows speed-up and absolute single node performance of the whole wave propagation scheme. These results are measured on a proxy application. (The proxy uses the exactly same kernels on random initial data; errors are in the range of a few percent.) The memory layouts (see Section 3.2) were chosen based on auto-tuning, minimizing time-to-solution. This implies different memory layouts on HSW and KNL, hence time-to-solution is the only fair comparison between architectures. On HSW, SC is 1.25 times faster than BL for order 6 and 1.55 times faster for order 7. On KNL, SC is 1.14 and 1.46 times faster, respectively. In terms of machine usage, SC achieves up to 56 % of peak performance on HSW (at 2.2 GHz), and up to 50 % of peak performance on KNL (at 1.2 GHz).

3.5 Dynamic rupture optimization

In 3D dynamic rupture simulations, Equation (1) and the frictional sliding process on a prescribed 2D fault are simultaneously solved. The fault's strength is defined according to the Coulomb failure criterion and post-failure slip behavior is determined from a frictional

constitutive law. Our implementation follows [70] and imposes the friction law as an internal boundary condition on elements aligned with the fault by modifying the numerical fluxes.

For accurately resolving the rupture process, an appropriate resolution of the so called “cohesive zone” [11] at the rupture tip is required, leading to small elements in the vicinity of faults. Nevertheless, elements aligned with the fault are generally only a small fraction of the total number of elements. For example, in the large-scale mesh of Section 5 only 4 % of 221 million elements hold dynamic rupture faces. However, an unoptimized dynamic rupture implementation may lead to a significant performance loss compared to a pure wave propagation problem. This effect leads to a factor of 2 difference in time-to-solution in the scaling tests for global time stepping in Section 4, which cannot be attributed solely to the optimization of the wave propagation kernels (compare Section 3.4.3). By reformulating the numerical scheme in terms of matrix chain products, we are able to use highly efficient matrix multiplication kernels.

3.5.1 Frictional Boundary Conditions. For linear, rotational invariant PDEs the flux term can be written as

$$\mathcal{F}_{kp} = T_{pq} A_{qr} \int_{t_n}^{t_{n+1}} \int_S \phi_k T_{rs}^{-1} \tilde{q}_s \, dS \, dt,$$

where T^{-1} rotates $\tilde{q}$ to a face-aligned system. The vector $\tilde{q}$ is the unknown state on the boundary of a tetrahedron, which is approximated as the solution to a local Riemann problem, the Godunov state, yielding the numerical flux. The key idea for imposing frictional sliding is to modify the wave speeds in the Godunov state such that the shear traction is set to a value imposed by the friction law. This leads automatically to a discontinuity in the particle velocity, which naturally models the movement of the fault in opposite directions. The friction coefficient μ determined by a friction law may depend on a state (for each point in space), e.g. for the linear slip-weakening friction law, μ linearly decreases with increasing slip until it reaches a critical slip distance D_c ; then it stays constant.

The first step in our implementation is to compute the (unmodified) Godunov state for a dynamic rupture face. This requires the derivatives $D^{+,k} = \partial^k / \partial t^k Q^{(r_1)}$ and $D^{-,k} = \partial^k / \partial t^k Q^{(r_2)}$ on both sides of the fault (i.e., from elements r_1 and r_2). We use the derivatives to evaluate the local solution at space-time integration points:

$$\mathcal{N}^{+,l} = V^{+,i} \left(\sum_{k=0}^N \frac{t_l^k}{k!} D^{+,k} \right), \quad \mathcal{N}^{-,l} = V^{-,j,h} \left(\sum_{k=0}^N \frac{t_l^k}{k!} D^{-,k} \right). \quad (9)$$

The time integration points t_l are given by a Gauss-Legendre quadrature rule and the matrices V are generalized Vandermonde matrices, which we define as

$$V_{kl}^{+,i} = \phi_l \left(\xi^{(i)} (\chi_k, \tau_k) \right), \\ V_{kl}^{-,j,h} = \phi_l \left(\xi^{(j)} \left(\tilde{\chi}^{(h)} (\chi_k, \tau_k), \tilde{\tau}^{(h)} (\chi_k, \tau_k) \right) \right).$$

Points χ_k, τ_k are given by the quadrature rule, where we use the rule from [82]. The index i denotes the local face of element r_1 that is part of the fault. Parameters j and h are determined such that the integration points are evaluated at the same location with respect to the basis on elements r_1 and r_2 .

The Godunov state is a linear combination of the degrees of freedom [70]. Hence, we obtain it with

$$\mathcal{G}^l = \mathcal{N}^{+,l} (T^i)^{-T} G^+ + \mathcal{N}^{-,l} (T^i)^{-T} G^-, \quad (10)$$

where $G^\pm$ depends on the material properties. Applying the friction law yields a modified Godunov state for both sides, $\hat{\mathcal{G}}^{+,l}$ and $\hat{\mathcal{G}}^{-,l}$. With quadrature weights w_l , we compute the time integral as

$$\hat{\mathcal{G}}^\pm = \sum_{l=0}^N w_l \hat{\mathcal{G}}^{\pm,l},$$

Finally, we compute the numerical flux as

$$\mathcal{F}^+ = - \frac{|S_i|}{|J(r_1)|} M^{-1} (V^{+,i})^T W \hat{\mathcal{G}}^+ (T^i A^{(r_1)})^T \quad \text{and} \quad (11)$$

$$\mathcal{F}^- = \frac{|S_i|}{|J(r_2)|} M^{-1} (V^{-,j,h})^T W \hat{\mathcal{G}}^- (T^i A^{(r_2)})^T \quad (12)$$

for the plus and minus side, resp., where W is a diagonal matrix containing the quadrature weights for the space integration rule.

3.5.2 Matrix kernels. We precompute the rotations $(TA)^T$ and $T^{-T}G$, because T, A , and G are quadratic and there is no gain in separating them. Furthermore, we premultiply all factors and signs into those matrices and we also precompute the multiplication of $(V)^T$ with the inverse mass matrix and the weight matrix W . In total, this yields matrix chain products of length 3, where all matrices are dense or mostly dense.

Using the code generator for the additional matrix chain products, we measured a performance of 610 GFLOPS for Equations (9) and (10) and 534 GFLOPS for Equations (11) and (12) on HSW. On KNL, we measured 1002 GFLOPS for Equations (9) and (10) and 968 GFLOPS for Equations (11) and (12). This compares well to the kernel performance shown in Figure 3, which shows that a matrix based formulation with the code generation ansatz delivers high performance almost automatically.

3.6 Local time stepping

We use the clustered local time stepping (LTS) scheme described in [13] for pure wave-propagation problems. This scheme uses a binning of elements in time clusters, where each cluster’s time step is a multiple of the previous cluster’s time step. For example, cluster 0 has time step $\Delta t_{\min}$ and cluster k has time step $2^k \Delta t_{\min}$. The cluster-based approach sacrifices part of the theoretical speed-up possible for an element-based LTS in favor of the hardware-oriented data structures and efficient load balancing approaches necessary to achieve high performance and scalability up to petascale [13].

To implement dynamic rupture with clustered LTS, we enforce, as the only constraint, that two elements that share a dynamic rupture face have the same time step. Otherwise we would need to change the numerical scheme of the friction law. The constraint is minor as the size of the two elements, and thus their resulting time step, is similar due to the prescribed mesh resolution on the fault.

Similar to regular elements, we sort dynamic rupture faces into time clusters. Afterwards, we sort dynamic rupture faces based on their adjoining elements’ communication layers. Here, we distinguish two cases: The first case is when a face has an adjoining element that is in the ghost layer. The second case is the converse, i.e. adjoining elements are either in the copy or in the interior layer.

Figure 4: Overhead of writing one checkpoint (830 GB) plus visualization output (17 GB) on 3072 nodes of SuperMUC depending on the I/O mode. The checkpoint is written to the scratch file system (> 120 GB/s) and the visualization output to the project file system (> 150 GB/s). The numbers are the average over 8 checkpoints plus visualization outputs.

These sorting steps allow us to already compute Godunov states of a subset of faces before receiving the ghost layer.

For load-balancing we adopt our strategy from [13], where we weight a time cluster with its relative update rate. The weight for cluster c is $2^{c_{\max}-c}$, with $c_{\max}$ being the largest time cluster. By using multi-constraint partitioning with ParMETIS [78], we obtain a balanced distribution of dynamic rupture faces. For the Sumatra scenario we measured a relative load imbalance of 5 % w.r.t. flops.

3.7 Asynchronous output

Due to the increasing amount of data processed by massively parallel applications, writing data to disk either for post-processing, visualization, or to allow restarting the application in case of hardware failures becomes a bottleneck. Alternating between computation and I/O phases can waste many CPU cycles if the amount of data written to disk is large. In these cases, overlapping computation and I/O leads to a better utilization of the computational resources. There are several advanced I/O libraries (e.g. ADIOS [59], GLEAN [87], Nessie [60]) that support asynchronous I/O, but restrict the user to certain file format and only support staging nodes.

We implemented a small header-only library that can perform asynchronous output via I/O threads or staging nodes and can be used with libraries without a non-blocking API (e.g. PnetCDF [57], HDF5 [83], SIONlib [31]). The library manages output buffers, data movement and synchronization of the I/O threads or nodes, but does not perform any I/O. Therefore, the application developer is able to choose the appropriate I/O library and format for her task. In Shaking Corals, we use this library to write checkpoints and large-scale output for visualization and tsunami coupling. The asynchronous I/O library works with all checkpoint back-ends (POSIX, MPI-IO, HDF5, SIONlib) and the XDMF writer [71] of Shaking Corals.

Previous work on SeisSol [13] has shown that scalability and time-to-solution is improved on Haswell architecture by using one core solely for communication. We pin our I/O threads to the same core as the communication thread to minimize the slowdown of the computation. Although the I/O thread does not require any CPU cycles from the computing cores, it can have an indirect influence on the performance due to the shared L3 cache. Our measurements showed that the influence is negligible for the comparably small visualization output but noticeable for checkpoints. By enabling direct I/O for checkpoints, we avoid additional buffering and therefore reduce the cache usage of the I/O thread. Figure 4 compares the overhead required for one checkpoint plus visualization output on SuperMUC using synchronous I/O with the I/O thread. The results

Figure 5: Strong scaling of the baseline version (BL) and the Shaking Corals (SC) version on SuperMUC (M) and Shaheen (S). Test cases are LTS (L) and GTS (G) with orders 6 and 7. In the top plot, the solid lines include only flops in the wave propagation kernels whereas the dotted lines include flops in the dynamic rupture kernels, too. The bottom plot shows time to solution, *extrapolated* to a full simulation.

show that computation and I/O perfectly overlap. As one core is sacrificed for communication anyway, I/O is practically for free.

4 SCALABILITY AND TIME-TO-SOLUTION

We tested our implementation on three recent supercomputers:

- M** SuperMUC Phase 2 with 3072 dual-socket Intel Xeon E5-2697 v3 nodes and Infiniband FDR14 interconnect. The nodes are organized in 6 islands and connected via a 4:1 pruned fat tree.
- S** Shaheen II with 6174 dual-socket Intel Xeon E5-2698 v3 nodes and Aries interconnect with Dragonfly topology.
- C** Cori with single-socket Intel Xeon Phi 7250 nodes and Aries interconnect with Dragonfly topology.

Note that all tests are production scenarios with enabled dynamic rupture.

Figure 6: Strong scaling of the baseline version (BL) on Cori (C) and the Shaking Corals (SC) version on Shaheen (S) and Cori (C). Test cases are LTS (L) and GTS (G) with order 6. Solid lines include only flops in the wave propagation kernel, dotted lines include flops in the dynamic rupture kernel, too.

Figure 7: Distribution of elements and dynamic rupture faces to time clusters. The clusters which cover the range $[4\Delta t_{\min}, 64\Delta t_{\min}]$ contain 98.7 % of the elements.

4.1 220,993,734 elements on Haswell

Global time stepping (GTS): In Figure 5 we compare strong scaling of the baseline version with the kernels of [42] (BL G6) with our Shaking Corals version (SC G6), both with GTS and order 6. We observe a speed-up of 2 w.r.t. execution time and an increase of GFLOPS per node from 269 to 425 on 3072 nodes, due to our optimizations described in Section 3. Here, we count only flops in the wave propagation kernel as the baseline version does not have a flop counter for the dynamic rupture kernel. The $2\times$ speed-up is larger than the increase in GFLOPS, because we require fewer flops in the kernels for wave propagation and dynamic rupture. With flops from computing the Godunov state and the numerical flux in the dynamic rupture kernel included, the GTS performance increases from 425 to 437 GFLOPS per node for SC G6.

Figure 5 also shows the *extrapolated* time to solution, which means that we multiply the measured time per time step with the number of time steps for the production run. From our scaling runs

we extrapolate that a full simulation with 500 seconds of simulated time on SuperMUC Phase 2 would take 7 days and 19 hours for BL G6 and 3 days and 22 hours for SC G6.

Local time stepping (LTS): The baseline version did not allow LTS together with dynamic rupture. To allow LTS in BL L6 we enforced that fault elements and all its neighbors share the same time step, which was taken as the minimum over all time steps at the fault. Hence, BL L6 is a mix of GTS on the fault and LTS for wave propagation. In theory, the speed-up due to LTS is limited by the number of GTS updates divided by the number of LTS updates:

$$\left(\sum_{c=0}^{c_{\max}} n_c \right) / \left(\sum_{c=0}^{c_{\max}} n_c \cdot 2^{-c} \right),$$

where $c_{\max}$ is the largest time cluster and n_c is the number of elements in cluster c . The theoretical speed-up of BL L6 is 5.7, whereas the theoretical speed-up of SC L6 is 9.9. Hence, we already obtain a theoretical speed-up of 1.74 by allowing LTS on the fault. Figure 7 shows the distribution of elements to time clusters for SC L6. In total, we have 11 time clusters for wave propagation and 4 time clusters for dynamic rupture.

On 128 nodes we measured a performance of 485 GFLOPS for SC L6. Performance drops to 306 GFLOPS on 3072 nodes. We think this is related to communication, especially due to SuperMUC’s pruned tree interconnect. This argument is supported by our scaling run at order 7, SC L7, which has a higher computation to communication ratio. While having lower performance initially, SC L7 overtakes SC L6 for increasing number of nodes.

For the LTS scaling tests, we executed five time steps of the largest time cluster (which is 5120 time steps for the smallest cluster). Extrapolating time of SC L6 to a full simulation over 500 seconds predicts a simulation time of 13.8 hours, which closely matches the measured simulation time of 13.9 hours for the production run. (Note that the scaling test did not include output whereas the production run included writing 15.8 TB of data.) From our scaling tests we estimate that BL G6 takes 13.6 times longer than SC L6. Comparing BL L6 to SC L6, we achieve a speed-up of 4.6.

We repeated the scaling tests on Shaheen, which has 4 additional cores per node and runs at 2.3 GHz by default. On 3072 nodes we measured 517 GFLOPS per node which is 1.7 times larger than the 306 GFLOPS per node measured on SuperMUC. This speed-up directly translates into time-to-solution, further reducing the estimated time for a full simulation to 8.2 hours, running at 1.59 PFLOPS.

4.2 51,020,237 elements on Knights Landing

Figure 6 shows a strong scaling study for the KNL architecture. As our access to Cori was limited to 512 KNL nodes, we used a smaller mesh. For comparison, we tested the same mesh on Shaheen. The ratio of LTS performance to GTS performance on 16 nodes is 79 % on Cori, whereas it is 92 % on Shaheen for our Shaking Corals version. This shows that the LTS scheme is less efficient on KNL compared to HSW, which we plan to investigate in future work. However on 512 KNL nodes, LTS still yields a speed-up of 7.5 compared to GTS and a speed-up of 1.28 compared to LTS on Shaheen. For machine utilization, we measured 0.9 TFLOPS for LTS and 1.2 TFLOPS for GTS on 512 KNL nodes. The latter number was also given in [42] for a pure wave propagation problem on a single node.

In comparison to the baseline version (BL), performance is increased by about $3\times$ for GTS. From our tests in Section 3.4.3 we would expect almost equal performance in the wave propagation kernel. Hence, the optimization of the dynamic rupture kernel is crucial for high performance on KNL. In terms of time to solution, SC L6 yields a speed-up of 15.2 compared to BL L6.

5 PETASCALE MULTI-PHYSICS SIMULATION OF THE 2004 SUMATRA EARTHQUAKE

On 26 December 2004, failure of 1300–1500 km of the Sumatra subduction zone, caused more than 8 minutes of violent shaking [1, 79]. The resulting M_w 9.1–9.3 megathrust earthquake generated a tsunami that was up to 30 m high along the northern coast of Sumatra. Curiously, observations suggest high fluid pressure and low stress drops [8, 37, 77], limiting the available energy to generate large events. Furthermore, seismic waves have been shown to dynamically trigger nearby faults [40, 61, 68]. In fact, the shallow subduction interface and the height of the tsunami that devastated northern Sumatra suggest that slip did occur on additional faults above the megathrust, dipping at much higher angles.

Incorporating proper initial conditions in terms of rheology, regional state of stress, and fault geometry, as well as modeling the high-resolution frictional failure and seismic waves not restricted to long periods, are essential for understanding the mechanics of these complex systems and mitigating future seismic hazard. Our simulated scenario includes 1500 km of fault length represented at 400 m geometrical resolution and 2.2 Hz frequency content of the seismic wave field. In contrast to previous large-scale earthquake simulations aimed at resolving as high frequencies as possible, our focus here is on understanding fault processes, long-period ground motions (given the lower resonance frequency of tsunamis than those of vulnerable buildings), and the intermediate frequency range in the vicinity of the megathrust potentially transferring energy to activate slip on splay faults. The scenario simulation was performed on an unstructured tetrahedral mesh with 221 million elements and 111 billion degrees of freedom with order 6 accuracy in space and time. The time step varied by a factor of 1024 between the smallest and largest elements with up to 3.3 million time steps, resolving fault-fault and fault-bathymetry intersections and considering the coarsened mesh in the far field. 500 s of simulated time required 13.9 hours computing time at 0.94 PFLOPS sustained performance on all of the 86,016 Haswell cores of SuperMUC. We have written 13 TB of checkpoint data and 2.8 TB for visualization and post-processing.

5.1 Observational modeling constraints

In the following, we summarize observational constraints and the initial conditions of the performed earthquake scenario. We combined geologically constrained fault geometries, topography, local velocity structure, and stress conditions in a highly realistic model of the Sumatra subduction zone. The geometry of the megathrust is based on the community model Slab1.0 [41], combining several independent datasets ranging from historic seismicity to active seismic profiles. We extend the geometry of Slab1.0 north in a self-consistent manner and toward the seafloor to intersect with 30 arc-second resolution topography from GEBCO [88]. The resulting curved megathrust interface dips almost horizontally near the

Figure 8: Rupture propagation across the megathrust and the lower backthrust splay fault and the emitted seismic wave field at 85 s simulation time. The unstructured tetrahedral mesh is refined in the vicinity of the fault and towards lower wave speeds close to the seafloor bathymetry.

seafloor intersection and at 30° at 50 km depth. Our model incorporates three splay faults located to the west of northern Sumatra, synthesized from seismic reflection data [16, 80, 81], bathymetry mapping [80], and relocated seismicity [58]. The material properties for 3 continental layers and 4 oceanic layers, which bend following the dip of the subduction zone interface, are taken from Crust1.0 [55]. Note that the megathrust, but not the splay faults, is embedded in a layer of lower seismic wave speeds than the surrounding rocks.

Following the Coulomb criterion, frictional failure of the megathrust and splay faults is controlled by the interplay between the strength of the faults and the tectonic stress field loading the faults. In our scenario, fault strength is at the low end of the inferred range for subduction zones (static coefficient of friction $\mu_s = 0.30$ following [37]) and accompanied by depth-dependent cohesion to mimic plastic deformation that decreases slip near the surface. During rupture, a fault’s frictional resistance decreases over $D_c = 0.8$ m to its dynamic value of $\mu_D = 0.25$. The pore fluid pressure is set at twice the hydrostatic fluid pressure, approaching but always less than the lithostatic fluid pressure [77].

The remote stress field is oriented optimally for full dip-slip at the earthquake’s hypocenter. Its amplitudes are constrained by a fixed ratio $R = 0.7$ of stress change to strength drop on the fault [6]. The maximum principal stress plunges 8° and is oriented at 309° in the southern part of the model and 330° in the northern part. It rotates gradually between these orientations in the model region from $5.5^\circ\text{N } 92.5^\circ\text{E}$ and $4^\circ\text{N } 93^\circ\text{E}$. The loading that all faults experience prior to the earthquake increases with depth and varies laterally due to the non-planar geometry of the slip interfaces.

5.2 Geophysical interpretation

The performed scenario reproduces the main observed characteristics of the real event, including its magnitude, fault slip distribution, and seafloor displacements.

The fault rupture process cannot be observed directly, but is typically determined by inverting data such as waves recorded

by seismic stations, displacements from GPS stations, satellite imagery or geologic data. In the particular case of Sumatra, there was no seismic station in the vicinity of the rupture. On the other hand, unusual additional information helped to constrain the coseismic displacement, including the displacement of local coral [66]. A-priori assumptions and the enormous scale of the Sumatra earthquake lead to a range of results [79]. For example, determined magnitudes are M_w 9.1–9.3, estimated rupture duration is 480–600 s, and estimated rupture length is 1300–1500 km.

We validate our simulation against the most recent inversion of geodetic and tsunami data [12] that accounts for data uncertainties in a Bayesian framework. Their preferred model has M_w 9.25 and returns slip of 40 m at the trench in two locations, near the hypocenter and near 8° latitude. Slip reaches 20 m in the northern part of the fault. These three regions of large coseismic slip are in agreement with previous results [54, 79].

Our simulated earthquake has M_w 9.18, with maximum horizontal displacement reaching 46.5 m near the trench in the two locations that match those determined by [12]. Horizontal and vertical surface displacements match the GPS observations used by [12] extremely well in both magnitude and orientation (Figure 9). Only near the epicenter, where we prescribe artificial nucleation, does the model exceed inferred observations, hinting at a more complex nucleation process than we implement. The simulated rupture lasts about 440 s, which is slightly faster than the consensus of most source inversions and may be due to simplified modeling assumptions (see Section 5.3).

The complex rupture dynamics evolving across the fault system shed new light on the mechanism of splay faults and their impact on observables. We capture rupture transfers to both backthrusts, whereas the forethrust stays locked, indicating a high sensitivity to orientation. The lower backthrust (see Figure 1) is dynamically triggered after passage of the main rupture front along the megathrust, then breaks in the opposite direction and produces a maximum slip of 7 m. The upper backthrust is weakly activated after 150 s, with a maximum slip of 2.2 m. Splay fault slip transfers into vertical seafloor displacement, resulting in 4 m of vertical displacement above the lower backthrust (see right inset of Figure 9)

Rupture across the megathrust, in distinction to the splay faults, happens inside a narrow layer with seismic wave speeds lower than the surrounding rocks. The high model resolution allows us to observe reflected and head waves generated at the layer interfaces, interacting with the rupture front and inducing multiple slip pulses (see Figure 8), as predicted by 2D simulations [46].

5.3 Impact and Outlook

The high-resolution seafloor displacement produced by our model will serve as a valuable initial condition for tsunami models. These tsunami models will promote our understanding of how megathrust dynamics affect tsunamigenesis. First, we already produced a similar scenario without splay faults; coupling this seafloor displacement with additional tsunami models will highlight the role of splay faults in producing a large tsunami wave. Furthermore, 2D dynamic rupture simulations [62, 64] on planar dipping faults show that plastic deformation in the overriding wedge results in a distinct, more vertical seafloor uplift, which might enhance the

Figure 9: Synthetic horizontal (left) and vertical (right) seafloor displacement. Arrows depict comparison to observations from geodetic and tsunami data summarized in [12]. The inset shows the considerable contribution of the lower splay fault to vertical uplift.

tsunami caused by the earthquake. Incorporating off-fault plasticity is computationally challenging and would increase computational time by up to 50 % [75]. Finally, our model earthquake is breaking the megathrust quickly, probably due to the smoothness of the fault on geometric scales below 400 m. However, evaluating the effects of small-scale, fractal roughness requires increasing the fault mesh resolution considerably. Although we have tested the complete work flow (incl. mesh generation and output) with meshes up to 1 billion elements, we are constrained by the computational resources available for production runs.

6 CONCLUSIONS

We presented the first dynamic rupture simulation of the devastating 2004 Sumatra earthquake that considers detailed geometry of the megathrust-splay fault system, 3D layered media, and high resolution topography. The earthquake scenario matches coseismic slip and horizontal and vertical surface displacements inferred from observations. The obtained results allow analysis of splay fault activation and will serve as time-dependent seafloor initial conditions to study tsunami generation and propagation.

The extreme duration of the dynamic rupture simulation (500 s) required an end-to-end optimization of the entire SeisSol software, targeted at strong scalability, which made extremely large simulations (1500 km faults) with up to 221 million elements and 111 billion degrees of freedom feasible w.r.t. time-to-solution. We implemented local time stepping for dynamic rupture, which, despite the low contribution to the overall computational load, had turned into a major bottleneck for realistic earthquake geometries. Using

code generation based on matrix chain products, we optimized the flux and dynamic rupture kernels leading to an additional speed-up of 2. Detailed scaling tests on SuperMUC revealed a speed-up 13.6 compared to the previous implementation and a speed-up of 6.8 compared to the optimized global time stepping implementation.

ACKNOWLEDGMENTS

The work presented in this paper was supported by the Volkswagen Foundation (project ASCETE – Advanced Simulation of Coupled Earthquake–Tsunami Events, grant no. 88479), by the German Research Foundation (DFG) (project no. KA 2281/4-1, AOBJ 584936 / TG-92), by the Bavarian Competence Network for Technical and Scientific High Performance Computing (KONWIHR) (project GeoPF – Geophysics for PetaFlop Computing), and by Intel as part of the Intel Parallel Computing Center ExScMIC-KNL. Computing resources were provided by the Leibniz Supercomputing Centre (LRZ, project no. pr45fi and h019z, on SuperMUC), by P. Martin Mai, King Abdullah University of Science and Technology (KAUST, on Shaheen-II) and by the National Energy Research Scientific Computing Center (NERSC, on Cori). We especially thank Nicolay Hammer (LRZ), as well as Richard Gerber and Jack Deslippe (NERSC) for their highly valuable support.

REFERENCES

- [1] C. J. Ammon, C. Ji, H.-K. Thio, D. Robinson, S. Ni, V. Hjorleifsdottir, H. Kanamori, T. Lay, S. Das, D. Helmberger, G. Ichinose, J. Polet, and D. J. Wald. 2005. Rupture Process of the 2004 Sumatra-Andaman Earthquake. *Science* 308, 5725 (2005), 1133–1139.
- [2] J.-P. Ampuero and Y. Ben-Zion. 2008. Cracks, pulses and macroscopic asymmetry of dynamic rupture on a bimaterial interface with velocity-weakening friction. *Geophys. J. Int.* 173, 2 (2008), 674–692.
- [3] D. J. Andrews. 1999. Test of two methods for faulting on finite-difference calculations. *B. Seismol. Soc. Am.* 89, 4 (1999), 931–937.
- [4] D. J. Andrews. 2005. Rupture dynamics with energy loss outside the slip zone. *J. Geophys. Res.* 110, 1 (2005), 1–14.
- [5] D. J. Andrews, T. C. Hanks, and J. W. Whitney. 2007. Physical limits on ground motion at Yucca Mountain. *B. Seismol. Soc. Am.* 97, 6 (2007), 1771–1792.
- [6] H. Aochi, R. Madariaga. 2003. The 1999 Izmit, Turkey, Earthquake: Nonplanar Fault Structure, Dynamic Rupture Process, and Strong Ground Motion. *B. Seismol. Soc. Am.* 93, 3 (2003), 1249–1266.
- [7] H. Aochi, R. Madariaga, and E. Fukuyama. 2003. Constraint of fault parameters inferred from nonplanar fault modeling. *Geochem. Geophys.* 4, 2 (2003), 1020.
- [8] P. Audet, M. G. Bostock, N. I. Christensen, and S. M. Peacock. 2009. Seismic evidence for overpressured subducted oceanic crust and megathrust fault sealing. *Nature* 457, 7225 (2009), 76–78.
- [9] M. Barall. 2009. A grid-doubling finite-element technique for calculating dynamic three-dimensional spontaneous rupture on an earthquake fault. *Geophys. J. Int.* 178 (2009), 845–859.
- [10] M. Barall and R. A. Harris. 2014. Metrics for Comparing Dynamic Earthquake Rupture Simulations. *Seismol. Res. Lett.* 86, 1 (Nov. 2014), 223–235.
- [11] G. I. Barenblatt. 1962. The Mathematical Theory of Equilibrium Cracks in Brittle Fracture. *Advances in Applied Mechanics* 7 (1962), 55–129.
- [12] Q. Bletery, A. Sladen, J. Jiang, and M. Simons. 2016. A Bayesian source model for the 2004 great Sumatra-Andaman earthquake. *J. Geophys. Res. Solid Earth* 121, 7 (2016), 5116–5135.
- [13] A. Breuer, A. Heinecke, and M. Bader. 2016. Petascale Local Time Stepping for the ADER-DG Finite Element Method. In *2016 IEEE Int. Parallel and Distrib. Process. Symp. (IPDPS)*. 854–863.
- [14] A. Breuer, A. Heinecke, S. Rettenberger, M. Bader, A.-A. Gabriel, and C. Pelties. 2014. Sustained Petascale Performance of Seismic Simulations with SeisSol on SuperMUC. In *Supercomputing: 29th Int. Conf.* Springer, 1–18.
- [15] C. Burstedde, O. Ghattas, N. Gurnis, T. Isaac, G. Stadler, T. Warburton, and L. Wilcox. 2010. Extreme-Scale AMR. In *Proc. Int. Conf. for High Performance Computing, Networking, Storage and Anal. (SC '10)*. IEEE Computer Society, 1–12.
- [16] A. P. S. Chauhan, S. C. Singh, N. D. Hananto, H. Carton, F. Klingelhoefer, J.-X. Dessa, H. Permana, N. J. White, and D. Graindorge. 2009. Seismic imaging of forearc backthrusts at northern Sumatra subduction zone. *Geophys. J. Int.* 179, 3 (Dec. 2009), 1772–1780.
- [17] Computational Infrastructure for Geodynamics. 2017. SPECfem3D Cartesian. www.geodynamics.org/cig/software/specfem3d/. (2017).
- [18] Y. Cui, E. Poyraz, K. B. Olsen, J. Zhou, K. Withers, S. Callaghan, J. Larkin, C. Guest, D. Choi, A. Chourasia, and others. 2013. Physics-based seismic hazard analysis on petascale heterogeneous supercomputers. In *Proc. Int. Conf. for High Performance Computing, Networking, Storage and Anal.* ACM, 70.
- [19] L. A. Dalguer and S. M. Day. 2007. Staggered-grid split-node method for spontaneous rupture simulation. *J. Geophys. Res.* 112, 2 (2007), 1–15.
- [20] L. A. Dalguer, K. Irikura, J. D. Riera, and H. C. Chiu. 2001. The importance of the dynamic source effects on strong ground motion during the 1999 Chi-Chi, Taiwan, earthquake: Brief interpretation of the damage distribution on buildings. *B. Seismol. Soc. Am.* 91, 5 (2001), 1112–1127.
- [21] S. M. Day, L. A. Dalguer, N. Lapusta, and Y. Liu. 2005. Comparison of finite difference and boundary integral solutions to three-dimensional spontaneous rupture. *J. Geophys. Res.* 110 (2005), 1–23.
- [22] J. de la Puente, J.-P. Ampuero, and M. Käser. 2009. Dynamic rupture modeling on unstructured meshes using a discontinuous Galerkin method. *J. Geophys. Res. Solid Earth* 114, B10 (2009), B10302.
- [23] N. DeDontney, J. R. Rice, and R. Dmowska. 2012. Finite Element Modeling of Branched Ruptures Including Off-Fault Plasticity. *B. Seismol. Soc. Am.* 102, 2 (2012), 541–562.
- [24] B. Duan. 2012. Dynamic rupture of the 2011 Mw 9.0 Tohoku-Oki earthquake: Roles of a possible subducting seamount. *J. Geophys. Res. Solid Earth* 117, B5 (2012), B05311.
- [25] B. Duan and S. M. Day. 2008. Inelastic strain distribution and seismic radiation from rupture of a fault kink. *J. Geophys. Res.* 113, B12 (Dec. 2008), B12311.
- [26] M. Dumbser and M. Käser. 2006. An arbitrary high-order discontinuous Galerkin method for elastic waves on unstructured meshes – II. The three-dimensional isotropic case. *Geophys. J. Int.* 167 (2006), 319–336.
- [27] M. Dumbser, M. Käser, and E. F. Toro. 2007. An arbitrary high-order discontinuous Galerkin method for elastic waves on unstructured meshes V: Local time stepping an p -adaptivity. *Geophys. J. Int.* 171, 2 (2007), 695–717.
- [28] M. Dumbser and C.-D. Munz. 2005. Arbitrary high order discontinuous Galerkin schemes. *Numerical methods for hyperbolic and kinetic problems* (2005), 295–333.
- [29] E. M. Dunham. 2007. Conditions governing the occurrence of supershear ruptures under slip-weakening friction. *Geophys. J. Int.* 172, B7 (2007), 1–24.
- [30] G. P. Ely, S. M. Day, and J.-B. Minster. 2010. Dynamic rupture models for the southern San Andreas fault. *B. Seismol. Soc. Am.* 100, 1 (2010), 131–150.
- [31] W. Frings, F. Wolf, and V. Petkov. 2009. Scalable Massively Parallel I/O to Task-local Files. In *Proc. Int. Conf. for High Performance Computing, Networking, Storage and Anal. (SC '09)*. ACM, New York, NY, USA, 1–11.
- [32] A.-A. Gabriel, J.-P. Ampuero, L. A. Dalguer, and P. M. Mai. 2012. The Transition of Dynamic Rupture Modes in Elastic Media Under Velocity-Weakening Friction. *J. Geophys. Res.* 117, B9 (2012), 0148–0227.
- [33] A.-A. Gabriel, J.-P. Ampuero, L. A. Dalguer, and P. M. Mai. 2013. Source Properties of Dynamic Rupture Pulses with Off-Fault Plasticity. *J. Geophys. Res.* (2013).
- [34] P. Galvez, J.-P. Ampuero, L. A. Dalguer, S. N. Somala, and T. Nissen-Meyer. 2014. Dynamic earthquake rupture modelled with an unstructured 3-D spectral element method applied to the 2011 M9 Tohoku earthquake. *Geophys. J. Int.* 198, 2 (2014), 1222–1240.
- [35] P. Galvez, L. A. Dalguer, J.-P. Ampuero, and D. Giardini. 2016. Rupture Reactivation during the 2011 Mw 9.0 Tohoku Earthquake: Dynamic Rupture and Ground-Motion Simulations. *B. Seismol. Soc. Am.* 106, 3 (2016), 819–831.
- [36] C. Geuzaine and J. F. Remacle. 2009. Gmsh: A 3-D finite element mesh generator with built-in pre- and post-processing facilities. *Internat. J. Numer. Methods Engrg.* 79, 11 (2009), 1309–1331.
- [37] J. L. Hardebeck. 2015. Stress orientations in subduction zones and the strength of subduction megathrust faults. *Science* 349, 6253 (Sept. 2015), 1213–1216.
- [38] R. A. Harris, M. Barall, D. J. Andrews, B. Duan, S. Ma, E. M. Dunham, A.-A. Gabriel, Y. Kaneko, Y. Kase, B. T. Aagaard, D. D. Oglesby, J.-P. Ampuero, T. C. Hanks, and N. Abrahamson. 2011. Verifying a Computational Method for Predicting Extreme Ground Motion. *Seismol. Res. Lett.* 82, 5 (2011), 638–644.
- [39] R. A. Harris, M. Barall, R. J. Archuleta, E. M. Dunham, B. T. Aagaard, J.-P. Ampuero, H. Bhat, V. Cruz-Atienza, L. A. Dalguer, P. Dawson, S. M. Day, B. Duan, G. P. Ely, Y. Kaneko, Y. Kase, N. Lapusta, Y. Liu, S. Ma, D. D. Oglesby, K. B. Olsen, A. Pitarka, S. Song, and E. Templeton. 2009. The SCEC/USGS Dynamic Earthquake Rupture Code Verification Exercise. *Seismol. Res. Lett.* 80, 1 (2009), 119–126.
- [40] R. A. Harris and S. M. Day. 1999. Dynamic 3D simulations of earthquakes on en echelon faults. *Geophys. Res. Lett.* 26, 14 (1999), 2089–2092.
- [41] G. P. Hayes, D. J. Wald, and R. L. Johnson. 2012. Slab1.0: A three-dimensional model of global subduction zone geometries. *J. Geophys. Res.* 117, B1 (Jan. 2012), B01302.
- [42] A. Heinecke, A. Breuer, M. Bader, and P. Dubey. 2016. High Order Seismic Simulations on the Intel Xeon Phi Processor (Knights Landing). In *High Performance Computing: 31st International Conference, ISC High Performance 2016, Frankfurt, Germany, June 19–23, 2016, Proceedings*. Springer, 343–362.

- [43] A. Heinecke, A. Breuer, S. Rettenberger, M. Bader, A.-A. Gabriel, C. Pelties, A. Bode, W. Barth, X.-K. Liao, K. Vaidyanathan, M. Smelyanskiy, and P. Dubey. 2014. Petascale High Order Dynamic Rupture Earthquake Simulations on Heterogeneous Supercomputers. In *Proc. Int. Conf. for High Performance Computing, Networking, Storage and Anal.* IEEE, New Orleans, 3–14. Gordon Bell Finalist.
- [44] A. Heinecke, G. Henry, M. Hutchinson, and H. Pabst. 2016. LIBXSMM: Accelerating Small Matrix Multiplications by Runtime Code Generation. In *Proc. Int. Conf. for High Performance Computing, Networking, Storage and Anal. (SC '16)*. IEEE Press, Piscataway, NJ, USA, 84:1–84:11.
- [45] K. Hiroo and K. Masayuki. 1993. The 1992 Nicaragua earthquake: a slow tsunami earthquake associated with subducted sediments. *Nature* 361, 6414 (Feb. 1993), 714–716.
- [46] Y. Huang, J.-P. Ampuero, and H. Kanamori. 2014. Slip-Weakening Models of the 2011 Tohoku-Oki Earthquake and Constraints on Stress Drop and Fracture Energy. *Pure Appl. Geophys.* 171, 10 (2014), 2555–2568.
- [47] M. Hutchinson, A. Heinecke, H. Pabst, G. Henry, M. Parsani, and D. Keyes. 2016. Efficiency of High Order Spectral Element Methods on Petascale Architectures. In *High Performance Computing: 31st Int. Conf. Springer*, 449–466.
- [48] T. Ichimura, K. Fujita, P. E. B. Quinay, L. Maddegada, M. Hori, S. Tanaka, Y. Shizawa, H. Kobayashi, and K. Minami. 2015. Implicit Nonlinear Wave Simulation with 1.08T DOF and 0.270T Unstructured Finite Elements to Enhance Comprehensive Earthquake Simulation. In *Proc. Int. Conf. for High Performance Computing, Networking, Storage and Anal. (SC '15)*. ACM, New York, 4:1–4:12.
- [49] T. Ichimura, K. Fujita, S. Tanaka, M. Hori, M. Lalith, Y. Shizawa, and H. Kobayashi. 2014. Physics-based urban earthquake simulation enhanced by 10.7 BlnDOF×30 K time-step unstructured FE non-linear seismic wave simulation. In *Proc. Int. Conf. for High Performance Computing, Networking, Storage and Anal.* IEEE, 15–26.
- [50] M. Käser and M. Dumbser. 2006. An Arbitrary High Order Discontinuous Galerkin Method for Elastic Waves on Unstructured Meshes I: The Two-Dimensional Isotropic Case with External Source Terms. *Geophys. J. Int.* 166, 2 (2006), 855–877.
- [51] M. Käser, M. Dumbser, J. de la Puente, and H. Igel. 2007. An arbitrary high-order discontinuous Galerkin method for elastic waves on unstructured meshes – III. Viscoelastic attenuation. *Geophys. J. Int.* 168 (2007), 224–242.
- [52] M. Käser and F. Gallovič. 2008. Effects of complicated 3-D rupture geometries on earthquake ground motion and their implications: a numerical study. *Geophys. J. Int.* 172, 1 (2008), 276–292.
- [53] J. E. Kozdon and E. M. Dunham. 2013. Rupture to the trench: Dynamic rupture simulations of the 11 March 2011 Tohoku earthquake. *B. Seismol. Soc. Am.* 103, 2B (2013), 1275–1289.
- [54] F. Krüger and M. Ohrnberger. 2005. Spatio-temporal source characteristics of the 26 December 2004 Sumatra earthquake as imaged by teleseismic broadband arrays. *Geophys. Res. Lett.* 32, 24 (2005), 1–4.
- [55] G. Laske, G. Masters, Z. Ma, and M. Pasyanos. 2013. Update on CRUST1.0 - A 1-degree Global Model of Earth's Crust, Abstract EGU2013-2658. European Geophysical Union.
- [56] R. J. LeVeque. 2002. *Finite volume methods for hyperbolic problems*. Vol. 31. Cambridge University Press.
- [57] J. Li, W. keng Liao, A. Choudhary, R. Ross, R. Thakur, W. Gropp, R. Latham, A. Siegel, B. Gallagher, and M. Zingale. 2003. Parallel netCDF: A High-Performance Scientific I/O Interface. In *Supercomputing, 2003 ACM/IEEE Conference*. 39–39.
- [58] J.-Y. Lin, X. Le Pichon, C. Rangin, J. C. Sibuet, and T. Maury. 2009. Spatial after-shock distribution of the 26 December 2004 great Sumatra-Andaman earthquake in the northern Sumatra area. *Geochem. Geophys.* 10, 5 (2009).
- [59] Q. Liu, J. Logan, Y. Tian, H. Abbasi, N. Podhorszki, J. Y. Choi, S. Klasky, R. Tchoua, J. Lofstead, R. Oldfield, M. Parashar, N. Samatova, K. Schwan, A. Shoshani, M. Wolf, K. Wu, and W. Yu. 2014. Hello ADIOS: the challenges and lessons of developing leadership class I/O frameworks. *Concurrency and Computation: Practice and Experience* 26, 7 (2014), 1453–1473.
- [60] J. Lofstead, R. Oldfield, T. Kordenbrock, and C. Reiss. 2011. Extending Scalability of Collective IO Through Nessel and Staging. In *Proceedings of the Sixth Workshop on Parallel Data Storage (PDSW '11)*. ACM, New York, NY, USA, 7–12.
- [61] J. C. Lozos. 2016. A case for historic joint rupture of the San Andreas and San Jacinto faults. *Science advances* 2, 3 (2016), e1500621.
- [62] S. Ma. 2012. A self-consistent mechanism for slow dynamic deformation and large tsunami generation for earthquakes in the shallow subduction zone. *Geophys. Res. Lett.* 39 (2012), 1–7.
- [63] S. Ma and G. C. Beroza. 2008. Rupture dynamics on a bimaterial interface for dipping faults. *B. Seismol. Soc. Am.* 98, 4 (2008), 1642–1658.
- [64] S. Ma and E. T. Hirakawa. 2013. Dynamic wedge failure reveals anomalous energy radiation of shallow subduction earthquakes. *Earth Planet. Sci. Lett.* 375 (2013), 113–122.
- [65] R. Madariaga. 1983. High frequency radiation from dynamic earthquake fault models. *Annales Geophysicae* 1, 1 (1983), 17–23.
- [66] A. J. Meltzner, K. Sieh, M. Abrams, D. C. Agnew, K. W. Hudnut, J.-P. Avouac, and D. H. Natawidjaja. 2006. Uplift and subsidence associated with the great Aceh-Andaman earthquake of 2004. *J. Geophys. Res. Solid Earth* 111, B2 (2006).
- [67] G. F. Moore, N. L. Bangs, A. Taira, S. Kuramoto, E. Pangborn, and H. J. Tobin. 2007. Three-Dimensional Splay Fault Geometry and Implications for Tsunami Generation. *Science* 318, 5853 (2007), 1128–1131.
- [68] D. D. Oglesby. 2008. Rupture termination and jump on parallel offset faults. *B. Seismol. Soc. Am.* 98, 1 (2008), 440–447.
- [69] D. D. Oglesby, R. J. Archuleta, and S. B. Nielsen. 2000. The three-dimensional dynamics of dipping faults. *B. Seismol. Soc. Am.* 90, 3 (2000), 616–628.
- [70] C. Pelties, J. de la Puente, J.-P. Ampuero, G. B. Brietzke, and M. Käser. 2012. Three-dimensional dynamic rupture simulation with a high-order discontinuous Galerkin method on unstructured tetrahedral meshes. *J. Geophys. Res. Solid Earth* 117, B2 (2012). B02309.
- [71] S. Rettenberger and M. Bader. 2015. Optimizing Large Scale I/O for Petascale Seismic Simulations on Unstructured Meshes. In *2015 IEEE International Conference on Cluster Computing (CLUSTER)*. IEEE Xplore, Chicago, IL, 314–317.
- [72] M. Rietmann, M. Grote, D. Peter, and O. Schenk. 2017. Newmark local time stepping on high-performance computing architectures. *J. Comput. Phys.* 334 (2017), 308–326.
- [73] M. E. Rognes, R. C. Kirby, and A. Logg. 2010. Efficient Assembly of $H(\text{div})$ and $H(\text{curl})$ Conforming Finite Elements. *SIAM Journal on Scientific Computing* 31, 6 (2010), 4130–4151.
- [74] O. Rojas, S. Day, J. Castillo, and L. A. Dalguer. 2008. Modelling of rupture propagation using high-order mimetic finite differences. *Geophys. J. Int.* 172, 2 (2008), 631–650.
- [75] D. Roten, Y. Cui, K. B. Olsen, S. M. Day, K. Withers, W. H. Savran, P. Wang, and D. Mu. 2016. High-frequency nonlinear earthquake simulations on petascale heterogeneous supercomputers. In *Proc. Int. Conf. for High Performance Computing, Networking, Storage and Anal.* IEEE, 82.
- [76] D. Roten, K. B. Olsen, H. Magistrale, J. C. Pechmann, and V. M. Cruz-Atienza. 2011. 3D simulations of M7 earthquakes on the Wasatch fault, Utah, part I: Long-period (0–1 Hz) ground motion. *B. Seismol. Soc. Am.* 101, 5 (2011), 2045–2063.
- [77] D. M. Saffer and H. J. Tobin. 2011. Hydrogeology and Mechanics of Subduction Zone Forearcs: Fluid Flow and Pore Pressure. *Annu. Rev. Earth Planet. Sci.* 39 (2011), 157–86.
- [78] K. Schloegel, G. Karypis, and V. Kumar. 2002. Parallel static and dynamic multi-constraint graph partitioning. *Concurrency and Computation: Practice and Experience* 14, 3 (2002), 219–240.
- [79] P. Shearer and R. Bürgmann. 2010. Lessons Learned from the 2004 Sumatra-Andaman Megathrust Rupture. *Annu. Rev. Earth Planet. Sci.* 38 (2010), 103–131.
- [80] J.-C. Sibuet, C. Rangin, X. Le Pichon, S. Singh, A. Cattaneo, D. Graindorge, F. Klingelhoefer, J.-Y. Lin, J. Malod, T. Maury, and others. 2007. 26th December 2004 great Sumatra-Andaman earthquake: Co-seismic and post-seismic motions in northern Sumatra. *Earth Planet. Sci. Lett.* 263, 1–2 (Nov. 2007), 88–103.
- [81] S. C. Singh, H. Carton, P. Tapponnier, N. D. Hananto, A. P. S. Chauhan, D. Hartoyo, M. Bayly, S. Moeljopranoto, N. Bunting, P. Christie, H. Lubis, and J. Martin. 2008. Seismic evidence for broken oceanic crust in the 2004 Sumatra earthquake epicentral region. *Nature Geoscience* 1, 11 (Oct. 2008), 777–781.
- [82] A. H. Stroud. 1971. *Approximate calculation of multiple integrals*. Prentice-Hall, Englewood Cliffs, NJ.
- [83] The HDF Group. 2017. HDF5. <https://www.hdfgroup.org/HDF5/>. (2017).
- [84] E. F. Toro. 1999. *Riemann Solvers and Numerical Methods for Fluid Dynamics, A Practical Introduction*. Springer, Berlin, Heidelberg, New York.
- [85] K. Tsuda, S. Iwase, H. Uratani, S. Ogawa, T. Watanabe, J. Miyakoshi, and J.-P. Ampuero. 2017. Dynamic Rupture Simulations Based on the Characterized Source Model of the 2011 Tohoku Earthquake. *Pure Appl. Geophys.* (2017), 1–12.
- [86] C. Uphoff and M. Bader. 2016. Generating high performance matrix kernels for earthquake simulations with viscoelastic attenuation. In *2016 Int. Conf. High Performance Computing Simulation (HPCS)*. 908–916.
- [87] V. Vishwanath, M. Hereld, and M. E. Papka. 2011. Toward simulation-time data analysis and I/O acceleration on leadership-class systems. In *2011 IEEE Symposium on Large Data Analysis and Visualization*. 9–14.
- [88] P. Weatherall, K. M. Marks, M. Jakobsson, T. Schmitt, S. Tani, J. E. Arndt, M. Rovere, D. Chayes, V. Ferrini, and R. Wigley. 2015. A new digital bathymetric model of the world's oceans. *Earth and Space Science* 2, 8 (2015), 331–345.

A ARTIFACT DESCRIPTION: EXTREME SCALE MULTI-PHYSICS SIMULATIONS OF THE 2004 SUMATRA MEGATHRUST EARTHQUAKE

A.1 Abstract

This artifact description contains information about the complete workflow required to set up simulations with the Shaking Corals version of SeisSol. We describe how the software can be obtained and the build process as well as necessary preprocessing steps to generate the input dataset for the node level performance measurements. Input datasets for the scaling and production runs are not publicly available due to their size. In addition, the artifact description outlines the complete workflow from the raw input data to the final visualization of the output.

A.2 Description

A.2.1 Check-list (artifact meta information).

- **Algorithm:** Arbitrary high-order DERivative Discontinuous Galerkin (ADER-DG) with clustered local time stepping.
- **Program:** SeisSol (www.seissol.org); version: *Shaking Corals*.
- **Compilation:** Intel C/C++ and Fortran Compiler.
- **Binary:** -
- **Data set:** CAD model of Sumatra subduction zone and Bay of Bengal assembled with GoCAD; Mesh generated with the Simulation Modeling Suite from Simmetrix (<http://simmetrix.com/>); see Section 5.1 for input data sets concerning topography, etc.
- **Run-time environment:** Lenovo NeXtScale nx360M5 WCT with IBM MPI (SuperMUC), Cray XC40 with Cray MPI (Shaheen II and Cori)
- **Hardware:** Optimized code is available for Intel Haswell, Intel Knights Landing and other Intel architectures. An unoptimized fallback version is also available.
- **Output:** Timings from the log file; optional receiver (seismic stations) and visualization output.
- **Experiment workflow:** See below.
- **Publicly available?** Code and example datasets are publicly available. The original input datasets for this paper are available upon request.

A.2.2 How software can be obtained. The software can be obtained from Github <https://github.com/SeisSol/SeisSol>:

```
$ git clone --recursive https://github.com/SeisSol/SeisSol
```

To use Shaking Corals as in this paper run:

```
$ git checkout 201703
$ git submodule update
```

A.2.3 Hardware dependencies. Shaking Corals uses libxsmm (<https://github.com/hfp/libxsmm>) to generate highly optimized assembly kernels. Hence, the performance results reported in this paper can only be achieved on modern Intel architectures. A fallback C implementation of the kernels is available in the code repository.

A.2.4 Software dependencies. Shaking Corals requires netCDF-4 (with MPI-IO support) for reading large unstructured meshes. Additional requirements are:

- SCons and Python 2.7 (cf. Section A.3)
- Intel C/C++ and Fortran Compiler
- libxsmm (cf. Section A.2.3)

A.2.5 Datasets. A setup including a mesh with over 3 million elements for the 2004 Sumatra-Andaman earthquake can be obtained from Zenodo <https://dx.doi.org/10.5281/zenodo.439946>. Due to the large size of the production-run meshes, these are only available upon request.

A.3 Installation

Shaking Corals uses SCons for compilation and supports various options to customize the binary. The following command compiles the release version using optimized Intel Haswell kernels for convergence order 6. The resulting binary will support a hybrid MPI+OpenMP parallelization and netCDF for mesh initialization. Note that netCDF is optional but recommended for large runs.

```
$ scons order=6 compileMode=release \
    generatedKernels=yes arch=dhsw \
    parallelization=hybrid commThread=yes \
    netcdf=yes
```

To get a full list of all available options, run:

```
$ scons --help
```

For a more detailed description, see <https://github.com/SeisSol/SeisSol/wiki>.

A.4 Experiment workflow

SeisSol requires at least three input files: The file DGPATH, a parameter file, and a mesh file. DGPATH should contain a single text line with the full path to the Maple folder inside the repository.

The parameter file is in Fortran namelist format and contains all parameters required to setup the simulation. Parameters can reference other files, e.g. to specify a list of receiver stations or more complex material parameters. The parameter file for the Sumatra earthquake can be used as a starting point for simulations.

Meshes are usually constructed from CAD models using either the Simulation Modeling Suite from Simmetrix or Gmsh. To convert meshes to the custom netCDF format, the preprocessing tool PUMGen is available on Github <https://github.com/TUM-I5/PUMGen/tree/201703>. Since meshes contain a fixed partitioning, PUMGen can also be used to repartition existing meshes.

For a parameter file `parameters.par`, SeisSol can be invoked with the following commands:

```
$ export OMP_NUM_THREADS=<threads>
$ mpiexec -n <processes> ./SeisSol parameters.par
```

If the communication thread is enabled, ensure that no OpenMP thread is pinned to the last core (e.g. by setting KMP_AFFINITY). For this paper, we used the following settings (using SMT on Haswell platforms):

```
# SuperMUC Phase 2
$ export OMP_NUM_THREADS=54
$ export KMP_AFFINITY=compact,granularity=thread
# Shaheen II
$ export OMP_NUM_THREADS=62
$ export KMP_AFFINITY=compact,granularity=thread
# Cori
$ export OMP_NUM_THREADS=65
$ export KMP_AFFINITY=proclist=[2-66],explicit,granularity=thread
```

On Cori we additionally left the first tile for the operating system.

A.5 Evaluation and expected result

SeisSol directly provides performance results (timings and FLOP counters from libxsmm) at the end of the simulation. For example, the production run from this paper (cf. Section 5) printed the following information:

```
Wall time (via gettimeofday): 50007.9 seconds.
Total calculated HW-GFLOP: 4.68102e+10
Total calculated NZ-GFLOP: 2.14946e+10
WP calculated HW-GFLOP: 4.42614e+10
WP calculated NZ-GFLOP: 1.9178e+10
DR calculated HW-GFLOP: 2.54884e+09
DR calculated NZ-GFLOP: 2.31664e+09
Time wave field writer backend: 103.908
Time wave field writer frontend: 0.222447
Time checkpoint frontend: 0.922489
Time checkpoint backend: 188.573
Time fault writer backend: 59.5376
Time fault writer frontend: 0.293896
Time free surface writer backend: 99.9163
Time free surface writer frontend: 0.136493
```

Single-node performance results can also be obtained with a proxy application. The proxy application can be found on Github https://github.com/SeisSol/SeisSol/tree/201703/auto_tuning/proxy.

Physical results are written into separate files. The number of files, the time interval and the location can be configured in the parameter file. Receivers can be used to generate high frequency output at certain points. In addition, SeisSol can produce large scale outputs for the fault, the surface and the whole wave field. Receivers are stored in simple text files, the large-scale output uses the XMDF format. The latter can be directly visualized with ParaView and VisIt.

A.6 Experiment customization

To use the I/O thread with direct I/O and enable optimizations for 8 MB block size of SuperMUC's GPFS, we set the following environment variables:

```
$ export XDMFWRITE_ALIGNMENT=8388608
$ export XDMFWRITE_BLOCK_SIZE=8388608
$ export SEISSOL_CHECKPOINT_ALIGNMENT=8388608
$ export SEISSOL_CHECKPOINT_DIRECT=1
$ export ASYNC_MODE=THREAD
$ export ASYNC_BUFFER_ALIGNMENT=8388608
```

A.7 Notes

The Github Wiki of the project contains further information for setting up simulations with SeisSol as well as additional example setups.

B COMPUTATIONAL RESULTS ANALYSIS: EXTREME SCALE MULTI-PHYSICS SIMULATIONS OF THE 2004 SUMATRA MEGATHRUST EARTHQUAKE

B.1 Abstract

In the following we summarize our approach to ensure the trustworthiness of our results, by: i) the verification of the dynamic rupture and seismic wave propagation implementation, with local time stepping (LTS) in Shaking Corals, ii) the validation of the earthquake scenario based on the 2004 Sumatra earthquake, iii) the verification of the performance evaluation based on flop counting.

To verify the pure seismic wave propagation part of SeisSol, version Shaking Corals, we perform convergence tests under varying order of accuracy and mesh refinement. We consider the L^∞ -error with respect to an analytical solution, as presented in [26]. In contrast, spontaneous dynamic rupture simulations suffer from a lack of analytical reference solutions. Therefore, we verify the new dynamic rupture solver based on a well-established community benchmark [e.g. 39], of the Southern California Earthquake Center (SCEC) which is publicly available at <http://scedata.usc.edu/cvws/>. The test problem covers very heterogeneous initial conditions and is here extended to varying time steps (element sizes) across the fault. Lastly, the large-scale simulation results of the Sumatra earthquake scenario are validated against geodetical and tsunami observation.

B.2 Results Analysis Discussion

Verification of dynamic rupture coupled to seismic wave propagation in Shaking Corals. In order to ensure correctness of the presented optimizations of the pure wave propagation part of Shaking Corals, we perform 3D convergence tests considering h - and p -refinement w.r.t. to an analytic solution. We solve a plane wave problem on a sequence of meshes and increasing order of accuracy, as presented in [26]. Figure 10 shows the results for order 2–7 and mesh spacings of 2.71–43.30 m. Each line slope in Figure 10 represents the achieved order of accuracy under mesh refinement. Our results demonstrate that we correctly recover the theoretical order of accuracy. The measured errors are in the range of the errors obtained in [26].

For dynamic rupture, no analytic solutions are available. A well established verification procedure for solvers tackling dynamic rupture coupled to seismic wave propagation is the comparison with state-of-the-art numerical methods in terms of key dynamic rupture parameters as well as synthetic ground motions. The SCEC Dynamic Earthquake Rupture Exercise (<http://scedata.usc.edu/cvws/>, [10, 38, 39]), provides community designed test problems of varying complexity and online evaluation of misfits. We present here verification of Shaking Corals for TPV16 (http://scedata.usc.edu/cvws/download/tpv16/TPV16_17_Description_v03.pdf), a 3D benchmark featuring linear slip-weakening friction and heterogeneous initial stress conditions across the fault resembling our production run. Since TPV16 is not featuring complex fault nor subsurface geometries we added sufficient complexity to verify the local time stepping implementation of Shaking Corals, in the following denoted as SC_LTS. We created a mesh consisting of three zones of

Figure 10: Convergence test for orders of accuracy 2–7 with decreasing mesh spacing. Numbers above the markers show the computed convergence order with respect to the next coarser mesh in the sequence of meshes.

Figure 11: Mesh with three different fault discretizations for the TPV16 benchmark with Shaking Corals.

different element sizes: 50 m around the nucleation patch, 200 m surrounding the 50 m box and 150 m elsewhere on the fault. Figure 11 visualizes the fault surface discretization. Including the smooth transitions between the different areas, the mesh leads to four distinct time clusters of the fault elements. Only 0.1 % of the almost 108 000 fault elements are assigned to the cluster with the minimal time step $\Delta t_{\min} = 6.5 \cdot 10^{-5}$ whereas 27.9 %, 31.7 % and 40.2 % are contained in the $2\Delta t_{\min}$, $4\Delta t_{\min}$, $8\Delta t_{\min}$ time cluster respectively. The mesh is created using the open source meshing software Gmsh [36] and the full setup is available at Zenodo <https://dx.doi.org/10.5281/zenodo.439950>.

We compare to TPV16 solved with Shaking Corals using global time stepping (GTS) and a mesh with uniform discretization on the fault, here denoted by SC_GTS. The results of both simulations are compared against well-established dynamic rupture software, namely FaultMod [9], a second-order accurate finite element code and SPECfEM3D [17] a high-order spectral element method.

Figure 12 compares slip rate and shear stress on the fault at 9 km along strike and 9 km in depth, which is located in the 150 m discretization zone (see Figure 11). Both versions of Shaking Corals

Figure 12: Comparison of FaultMod, SPECFEM3D, Shaking Corals GTS (SC_GTS) and LTS (SC_LTS) for on-fault dynamic rupture parameters 9 km along-strike and at 9 km depth of TPV16.

show excellent agreement with FaultMod and SPECFEM3D. Furthermore, the LTS and GTS versions of Shaking Corals give near-identical solutions, although the compared element is not updated at the minimal time step.

Scenario-based validation: We demonstrated in the previous paragraph, that Shaking Corals solves the multiphysics problem in accordance with independent implementations based on very different numerical schemes. However, we cannot claim by this means to be able to capture the full complexity of natural earthquake source dynamics. Therefore, conducting a realistic scenario based on a real event, such as presented in Section 5, is crucial to validate Shaking Corals against measured data.

The simulated magnitude M_w 9.18, which is calculated from the size of the ruptured area and the amount of displacement, is very similar to the officially observed size of M_w 9.1 defined by the US Geological Survey. The modeled slip of 40 m at the trench both near the hypocenter and at 8° latitude matches the slip distribution most recently inferred by [12], as well as previous inversions [79], and ground motions records of teleseismic broadband arrays [54].

As we presented in Figure 9, the modeled seafloor displacements, crucial for tsunami genesis, are also in excellent agreement with geodetic and tsunami data [12]. The modeled displacements do differ considerably from observations for one measurement very close to the epicenter, hinting at a more complex earthquake nucleation process than implemented in the scenario. In addition, the duration of the simulated earthquake is faster (440 s) than the 500–600 s inferred by most source inversions. This may be due to the considerable non-uniqueness of source inversion methods or simplified modeling assumptions such as a smooth fault geometry on scales smaller than 400 m (see Section 5.3).

Performance Results: Shaking Corals’ code generator provides a flop count of every matrix chain product. Before the simulation loop starts, the cost of every loop over elements is calculated exactly using the flop counter of the code generator. The cost of simple BLAS 1 loops is also calculated and added to this count. During the simulation, whenever a loop over elements is encountered, the pre-calculated flop count is added to the internal flop counter. After

the simulation, the flop count is summed over all partitions and printed to the log.

In order to verify this approach, we compared the flop count to libxsmm’s flop counter in debug mode, which adds the respective flops whenever a matrix matrix multiplication is calculated. In debug mode, we also update libxsmm’s flops counter whenever one of our custom BLAS 1 routines is called. For implementation reasons, we have chosen to neglect some of the BLAS 1 operations such that libxsmm’s flop counter may yield a higher flop count than our internal flop counter. Furthermore, we neglect some flops in the friction law for dynamic rupture. Here, counting flops would be difficult due to branches.

In total, the flop count we use to calculate performance results is a lower bound of the actual flop count. Also, we think that the amount of neglected flops is minor.

B.3 Summary

The previous section shows that both parts, the wave propagation and the dynamic rupture part of Shaking Corals, are well tested for the use in challenging dynamic rupture problems. We ensure that the results of the Sumatra earthquake scenario agree reasonably well to available observation. The flop count based method to evaluate performance is verified against libxsmm.