High Performance and Parallel Computing for Materials Defects and Multiphase Flows

Dynamic Rupture Earthquake Simulations on Petascale Heterogeneous Supercomputers

M. Bader, A. Breuer, S. Rettenberger, A.-A. Gabriel, C. Pelties, A. Heinecke

Singapore, 14 Jan 2015
SeisSol Core Developers

Alexander Breuer

Alice-Agnes Gabriel

Alexander Heinecke

Christian Pelties

Sebastian Rettenberger
Overview and Agenda

Dynamic Rupture and Earthquake Simulation with SeisSol:
- unstructured tetrahedral meshes
- high-order ADER-DG discretisation
- target applications: earthquake dynamics, tsunami generation

Towards Scalable I/O:
- two-stage scheme to generate and read mesh partitions
- improve scalability of output and checkpointing

Optimisation for Heterogeneous Petascale Platforms:
- code generation to optimize element-local matrix kernels
- offload scheme to address multiphysics

Petascale Runs on Tianhe-2, Stampede and SuperMUC:
- weak scaling of wave propagation component
- strong scaling for 1992 Landers M7.2 earthquake
Part I

Dynamic Rupture and Earthquake Simulation with SeisSol

http://seissol.geophysik.uni-muenchen.de/
Dynamic Rupture and Earthquake Simulation

Landers fault system: simulated ground motion and seismic waves

SeisSol – ADER-DG for seismic simulations:

• adaptive tetrahedral meshes
  → complex geometries, heterogeneous media, multiphysics
• complicated fault systems with multiple branches
  → non-linear multiphysics dynamic rupture simulation
• ADER-DG: high-order discretization in space and time
1992 Landers M7.2 Earthquake

• multiphysics simulation of dynamic rupture and resulting ground motion of a M7.2 earthquake
• fault inferred from measured data, regional topography from satellite data, physically consistent stress and friction parameters
• 1D velocity structure, low velocity near surface
Multiphysics Dynamic Rupture Simulation

- spontaneous rupture, non-linear interaction with wave-field
- featuring rupture jumps, fault branching, etc.
- tackles fundamental questions on earthquake dynamics
- realistic rupture source for seismic hazard assessment
Multiphysics Dynamic Rupture Simulation

- spontaneous rupture, non-linear interaction with wave-field
- featuring rupture jumps, fault branching, etc.
- tackles fundamental questions on earthquake dynamics
- realistic rupture source for seismic hazard assessment
Multiphysics Dynamic Rupture Simulation

- spontaneous rupture, non-linear interaction with wave-field
- featuring rupture jumps, fault branching, etc.
- tackles fundamental questions on earthquake dynamics
- realistic rupture source for seismic hazard assessment
Multiphysics Dynamic Rupture Simulation

- spontaneous rupture, non-linear interaction with wave-field
- featuring rupture jumps, fault branching, etc.
- tackles fundamental questions on earthquake dynamics
- realistic rupture source for seismic hazard assessment
Multiphysics Dynamic Rupture Simulation

• spontaneous rupture, non-linear interaction with wave-field
• featuring rupture jumps, fault branching, etc.
• tackles fundamental questions on earthquake dynamics
• realistic rupture source for seismic hazard assessment
Multiphysics Dynamic Rupture Simulation

- spontaneous rupture, non-linear interaction with wave-field
- featuring rupture jumps, fault branching, etc.
- tackles fundamental questions on earthquake dynamics
- realistic rupture source for seismic hazard assessment
Multiphysics Dynamic Rupture Simulation

- spontaneous rupture, non-linear interaction with wave-field
- featuring rupture jumps, fault branching, etc.
- tackles fundamental questions on earthquake dynamics
- realistic rupture source for seismic hazard assessment
Multiphysics Dynamic Rupture Simulation

- spontaneous rupture, non-linear interaction with wave-field
- featuring rupture jumps, fault branching, etc.
- tackles fundamental questions on earthquake dynamics
- realistic rupture source for seismic hazard assessment
Multiphysics Dynamic Rupture Simulation

- spontaneous rupture, non-linear interaction with wave-field
- featuring rupture jumps, fault branching, etc.
- tackles fundamental questions on earthquake dynamics
- realistic rupture source for seismic hazard assessment
Multiphysics Dynamic Rupture Simulation

- spontaneous rupture, non-linear interaction with wave-field
- featuring rupture jumps, fault branching, etc.
- tackles fundamental questions on earthquake dynamics
- realistic rupture source for seismic hazard assessment
Multiphysics Dynamic Rupture Simulation

- spontaneous rupture, non-linear interaction with wave-field
- featuring rupture jumps, fault branching, etc.
- tackles fundamental questions on earthquake dynamics
- realistic rupture source for seismic hazard assessment
Multiphysics Dynamic Rupture Simulation

- spontaneous rupture, non-linear interaction with wave-field
- featuring rupture jumps, fault branching, etc.
- tackles fundamental questions on earthquake dynamics
- realistic rupture source for seismic hazard assessment
Multiphysics Dynamic Rupture Simulation

- spontaneous rupture, non-linear interaction with wave-field
- featuring rupture jumps, fault branching, etc.
- tackles fundamental questions on earthquake dynamics
- realistic rupture source for seismic hazard assessment
Multiphysics Dynamic Rupture Simulation

• spontaneous rupture, non-linear interaction with wave-field
• featuring rupture jumps, fault branching, etc.
• tackles fundamental questions on earthquake dynamics
• realistic rupture source for seismic hazard assessment
Multiphysics Dynamic Rupture Simulation

- spontaneous rupture, non-linear interaction with wave-field
- featuring rupture jumps, fault branching, etc.
- tackles fundamental questions on earthquake dynamics
- realistic rupture source for seismic hazard assessment
Landers Earthquake – Results

Observations:

- complex rupture dynamics (fault branching, etc.)
- high-frequency signals from rupture propagate directly into wave field
- synthetic seismograms with frequencies up to 10 Hz
- ground shaking in the engineering frequency band
- 42 s simulated time

SuperMUC Production Run:

- **1.25 PFLOPS** sustained performance
- 7 h 15 min computing time
Project ASCETE
Simulation of Coupled Earthquake-Tsunami Events

prototype simulation by P. Galvez [5] and K. Rahnema

Project Partners:

- earthquake simulation: C. Pelties, A. Gabriel, H. Igel (LMU Munich, SeisSol)
- seismology: L. Dalguer, P. Galvez (ETH Zürich)
- tsunami simulation: J. Behrens, S. Vater (Klima Campus, Univ. Hamburg)
- HPC: M. Bader, A. Breuer, K. Rahnema (TUM)
Simulation of Tsunamigenic Earthquakes

Challenges:

- curved subduction zones that meet surface at shallow angles → high impact on uplift for tsunamigenic earthquakes
- impact of time-dependent earthquake source on tsunami → near-field tsunami with long rupture process (Sumatra 2004) → near-field tsunami “double-slip” earthquake (Tohoku 2011)
- turning oceanic ground motion into shallow-water displacements
Part II

Scalable Meshing and I/O

Mesh Generation and Partitioning

Mesh Generation:

• high-quality meshes required (complicated fault structures, controllable mesh coarsening)
• with $10^8$–$10^9$ grid cells
• using SimModeler by Simmetrix (http://simmetrix.com/)

Wanted: scalable solution to provide parallel mesh partitions

• graph-based partitioning (ParMETIS)
• create customised parallel format (based on netCDF) for mesh partitions (also contains information on neighbour partitions, etc.)
• highly scalable mesh input via netCDF/MPI-IO in SeisSol
Stating Point: Init Mesh Partitions in SeisSol

- strong scaling test on SuperMUC
- LOH.1 benchmark with 7,252,482 grid cells
- combine info from GAMBIT Neutral file and Metis Partition file
- does not scale, memory requirements per task: $O(#\text{cells})$
Two-stage Approach for Scalable Mesh Input

PUMgen: offline partitioning and mesh file generation

- start from mesh file as input
- call ParMETIS to compute partitions and communication structure
- write parallel mesh input file (based on netCDF format)

Resulting Mesh Pipeline:

- Gambit Mesh
- Converter
- netCDF Mesh
- SeisSol

~ 64 Tasks
Up to 9216 Tasks

Runtime: 47.8 min 5.8 sec
Two-stage Approach for Scalable Mesh Input (2)

PUMgen: offline parallel meshing and partitioning

- start from CAD file as input
- parallel mesh generation via Simmetrix interface
- compute partitions and communication structure (ParMETIS)
- write parallel mesh input file (based on netCDF format)

Final Mesh Pipeline using PUMgen:

Parallel Mesh Generation

CAD Model

Serial Mesh Generation

SimModeler/GAMBIT

Gambit Mesh File

Simulation Modeling Suite C++ API

ParMETIS

PUMGen

netCDF Mesh

SeisSol
Parallel Mesh Input

- mesh and partition information combined in a single mesh file (binary file format: HDF5/netCDF)
- includes information on neighbour partitions
- includes information on partition boundaries (ghost layers, etc.)
- relies purely on collective MPI I/O operations to access data
- each processor only reads its own partition
Parallel Mesh Format – Strong Scaling

- SuperMUC
- Strong scaling
- LOH.1
- 7,252,482 cells

- Runtime and memory requirements are reduced to $O(\#\text{cells}/\#\text{partitions})$
- largest test: mesh with 8,847,360,000 tetrahedrons read in 23 seconds on 9216 nodes/ranks
Parallel Mesh Format – Weak Scaling

- Cubic domain, uniformly refined tetrahedral cells
- Weak scaling: 400,000 elements per node
- Largest mesh: > 3.5 billion cells
Parallel Mesh Format – Weak Scaling

Old

- 8000 ranks
- 7.3 million elements
- 4 minutes for initialization

New

- 9216 ranks
- >3.5 billion elements
- 14 seconds for initialization

- Cubic domain, uniformly refined tetrahedral cells
- Weak scaling: 400,000 elements per node
- Largest mesh: > 3.5 billion cells
Wave Field Visualization for SeisSol

- only low-order data visualised
  ⇒ data sets are small (compared to computational cost)
- goal: aggregate all data into a single file
  ⇒ avoids creation of many small files & directory structures
- use **XDMF**: eXtensible Data Model and Format
  (supported by Paraview and Visit, [http://xdmf.org/](http://xdmf.org/))
Writing to a Single File in Parallel

- Files can be partially locked by a process
- “Parts” are defined by the file system blocksize (GPFS)

![Diagram showing processes and blocks on file system]

- SeisSol’s chunks are small (~200 KB per process) compared to the blocksize (8 MB on SuperMUC)
- thus: acquiring locks becomes the bottleneck
Aggregate and Align Data

- Data is redistributed/aggregated to match file system blocks
- Not all processes may be included in the actual I/O phase
XDMF – First test application

- \( \sim 2 \times 78 \text{ KB per process} \)
  (simulates \( \sim 10,000 \text{ elements with 2 variables per process} \))
- Exact element counts can vary \( \pm 5 \text{ elements} \)
  (simulates load imbalance from partitioners)
- SuperMUC: GPFS with 8 MB blocksize
Checkpointing - Latest results

- Weak-scaling scenario with 640,000 elements per node, 6th order accuracy → Checkpoint size: 2.4 GB/node
- Total bandwidth includes additional overhead (writing metadata, flushing file, ...)
- GPFS on SuperMUC: Theoretical peak bandwidth on 9,216 nodes: 200 GB/s
- Overhead for writing a checkpoint every 1,000 timesteps is less than 0.5%
Part III

Optimizing Kernel Operations in SeisSol

Breuer et al. [1,2]
PRACE ISC Award 2014
Seismic Wave Propagation with SeisSol

Elastic Wave Equations: (velocity-stress formulation)

\[ q_t + Aq_x + Bq_y + Cq_z = 0 \]

with \( q = (\sigma_{11}, \sigma_{22}, \sigma_{33}, \sigma_{12}, \sigma_{23}, \sigma_{13}, u, v, w)^T \)

\[ A = \begin{pmatrix}
0 & 0 & 0 & 0 & 0 & 0 & -\lambda - 2\mu & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -\lambda & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -\lambda & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -\lambda & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -\mu & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\mu \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -\mu & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\mu \\
0 & 0 & 0 & 0 & 0 & 0 & -\rho^{-1} & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -\rho^{-1} & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -\rho^{-1} & 0 & 0 & 0
\end{pmatrix} \]

\[ B = \begin{pmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\lambda \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\lambda \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\mu \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\mu \\
0 & -\rho^{-1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -\rho^{-1} & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -\rho^{-1} & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -\rho^{-1} & 0 & 0 & 0 & 0
\end{pmatrix} \]

- high order discontinuous Galerkin discretisation
- **ADER-DG**: high approximation order in space and time:
- additional features: local time stepping, high accuracy of earthquake faulting (full frictional sliding)
  \[ \rightarrow \text{Dumbser, Käser et al. [4,6]} \]
ADER-DG – Benefit of High Order

![Graph showing the relationship between execution time and error linf for different orders of accuracy.](image)

- Error linf decreases as execution time increases for all orders of accuracy.
- The graph compares execution time (s) on a logarithmic scale against error linf for orders 2, 3, 4, 5, 6, and 7.

M. Bader et al.: Dynamic Rupture Earthquake Simulations on Petascale Heterogeneous Supercomputers
SeisSol in a Nutshell – ADER-DG

Update scheme

\[
\begin{align*}
Q_{k}^{n+1} &= Q_k - \left| S_k \right| J_k^{-1} \left( \sum_{i=1}^{4} F^{-,i} I(t^n, t^{n+1}, Q_k^n) N_{k,i} A_k^+ N_{k,i}^{-1} \\
&\quad + \sum_{i=1}^{4} F^{+,i,j,h} I(t^n, t^{n+1}, Q_k^n(t_i)) N_{k,i} A_{k(i)}^- N_{k,i}^{-1} \right) \\
&\quad + M^{-1} K^{\xi} I(t^n, t^{n+1}, Q_k^n) A_k^* \\
&\quad + M^{-1} K^{n} I(t^n, t^{n+1}, Q_k^n) B_k^* \\
&\quad + M^{-1} K^{\zeta} I(t^n, t^{n+1}, Q_k^n) C_k^*
\end{align*}
\]

Cauchy Kovalewski

\[
I(t^n, t^{n+1}, Q_k^n) = \sum_{j=0}^{J} \frac{(t^{n+1} - t^n)^{j+1}}{(j + 1)!} \frac{\partial^j}{\partial t^j} Q_k(t^n)
\]

\[
(Q_k)_t = -M^{-1} \left( (K^{\xi})^T Q_k A_k^* + (K^{n})^T Q_k B_k^* + (K^{\zeta})^T Q_k C_k^* \right)
\]
Optimisation of Matrix Operations

Apply sparse matrices to multiple DOF-vectors $Q_k$

\[
(K^\xi)^T \quad \frac{\partial j}{\partial t} Q_k \quad A_k^*
\]

**Code Generator for Sparse Kernels:** (Breuer et al. [1])

- avoid overhead of CSR (or similar) data structures; store CSR elements vector, only
- full “unrolling” of all element operations using a code generator
- use intrinsics and apply blocking to improve vectorisation
Code Generator – Sparse Matrix Kernels
Code Generator – Dense Kernels
Optimisation of Matrix Operations

Apply sparse matrices to multiple DOF-vectors $Q_k$

Dense vs. Sparse Kernels: (Breuer et al. [2])

- switch to dense kernels depending on achieved time to solution
- for sparse and dense kernels: exploit zero-blocks generated during recursive CK computation
Performance Optimization

Switch between Sparse/Dense Kernels:

- auto-tuning approach on benchmark scenarios
- measure sparse vs. dense performance for each matrix
- select sparse vs. dense kernel based on best time to solution

<table>
<thead>
<tr>
<th>order</th>
<th>sparse</th>
<th>sparse</th>
<th>sparse</th>
<th>sparse</th>
<th>sparse</th>
<th>sparse</th>
<th>boundary</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>13%</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>dense</td>
<td>26%</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td>dense</td>
<td></td>
<td></td>
<td></td>
<td>17%</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>sparse</td>
<td></td>
<td>23%</td>
</tr>
<tr>
<td>6</td>
<td>dense</td>
<td>dense</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>9%</td>
</tr>
</tbody>
</table>

Hybrid MPI+OpenMP Parallelisation:

- careful OpenMP parallelisation of all parts (not only main kernels → communication buffers, etc.)
- targeted at manycore platforms, such as Intel Xeon Phi
- OpenMP improvements for Xeon Phi also lead to noticeable improvements for “standard” CPUs
Part IV

Dynamic Rupture Simulation at Petascale – Xeon Phi Supercomputers

Heinecke et al.: *Petascale High Order Dynamic Rupture Earthquake Simulations on Heterogeneous Supercomputers* [3]
ACM Gordon Bell Finalist 2014
Supercomputing Platforms

SuperMUC @ LRZ, Munich
- 9216 compute nodes (18 “thin node” islands)
  147,456 Intel SNB-EP cores (2.7 GHz)
- Infiniband FDR10 interconnect (fat tree)
- #12 in Top 500: 2.897 PFlop/s

Stampede @ TACC, Austin
- 6400 compute nodes, 522,080 cores
  2 SNB-EP (8c) + 1 Xeon Phi SE10P per node
- Mellanox FDR 56 interconnect (fat tree)
- #7 in Top 500: 5.168 PFlop/s

Tianhe-2 @ NSCC, Guangzhou
- 8000 compute nodes used, 1.6 Mio cores
  2 IVB-EP (12c) + 3 Xeon Phi 31S1P per node
- TH2-Express custom interconnect
- #1 in Top 500: 33.862 PFlop/s
Optimization for Intel Xeon Phi Platforms

Offload Scheme:
- hides communication with Xeon Phi and between nodes
- use “heavy” CPU cores for dynamic rupture

Hybrid parallelism:
- on 1–3 Xeon Phis and host CPU(s)
- reflects multiphysics simulation
- manycore parallelism on Xeon Phi
Weak Scaling of Wave Propagation

- goal: test scalability towards large problem sizes
- cubic domain, uniformly refined tetrahedral cells
- weak scaling: 400,000 elements per card/node
Weak Scaling of Wave Propagation

- more than 90% parallel efficiency on Tianhe-2 and Stampede
- 87% on full SuperMUC (no overlapping)
Weak Scaling of Wave Propagation

- 8.6 PFlop/s on Tianhe-2 (8000 nodes)
- 2.3 PFlop/s on Stampede (6144 nodes)
- 1.6 PFlop/s on SuperMUC (9216 nodes)
Weak Scaling – Peak Efficiency

![Graph showing weak scaling efficiency for different supercomputers.](image)
Strong Scaling of Landers Scenario

- 191 million tetrahedrons; 220,982 element faces on fault
- 6th order, 96 billion degrees of freedom
Strong Scaling of Landers Scenario

- more than 85% parallel efficiency on Stampede and Tianhe-2 (when using only one Xeon Phi per node)
- multiple-Xeon-Phi performance suffers from MPI communication
Strong Scaling of Landers Scenario

- 3.3 PFlop/s on Tianhe-2 (7000 nodes)
- 2.0 PFlop/s on Stampede (6144 nodes)
- 1.3 PFlop/s on SuperMUC (9216 nodes)
Landers Strong Scaling – Peak Efficiency

SuperMUC, gr. buff.  
Stampede  
SuperMUC, classic

Tianhe-2

% peak: hardware

% peak: non-zero

# nodes

M. Bader et al.: Dynamic Rupture Earthquake Simulations on Petascale Heterogeneous Supercomputers  
SeisSol: Latest/Current Work on Optimization

Further Performance Engineering:

- improved memory layout: separate ghost/copy layers, NUMA awareness
- alignment of data for matrix kernels (esp. important for MIC)
- overlap computation and communication (using more effective data layout)
- improved matrix kernels: assembler/machine code, AVX2 for Haswell

Towards Local Time Stepping:

- communication of time-integrated variables (instead of simple ghost cell exchange)
- more asynchronous updates
SeisSol: Earthquake Simulation @ Petascale

Multiphysics Dynamic Rupture Simulations with SeisSol:

- high-order ADER-DG on complicated geometries
- non-linear interaction of rupture process and seismic waves
- physics-based seismic hazard analysis

Petascale Performance on Heterogeneous Platforms:

- scalable mesh input (and output) of more than 9 billion cells
- exploits high computational intensity of ADER-DG
- requires careful tuning of the entire simulation pipeline
- code generation to accelerate element kernels
- offload-scheme for multiphysics with Xeon Phi
- extreme (hybrid) parallelism with approx. 1.6 million cores
Acknowledgements

Special thanks go to . . .

- Intel:
  - Mikhail Smelyanskiy, Karthikeyan Vaidyanathan
  - Pradeep Dubey

- all colleagues from the three supercomputing centers, esp.:
  - Arndt Bode (Leibniz Supercomputing Centre)
  - William Barth (Texas Advanced Computing Centre)
  - Xiang-Ke Liao (NCSS and NUDT)

- for financial support:
  - Volkswagen Foundation (project ASCETE)
  - German Research Foundation
  - KONWIHR

- the entire SeisSol team and all contributors
References


