Distributed Parallel–Adaptive Causal Spacetime Discontinuous Galerkin Method with Application to Earthquake Simulation

Robert B. Haber[1,*], **Amit Madhukar**[1], **Xiao Ma**[2], **Ahmed Elbanna**[2], **Reza Abedi**[3]

[1]Department of Mechanical Science & Engineering, University of Illinois, Urbana, USA
[2]Department of Civil & Environmental Engineering, University of Illinois, Urbana, USA
[3]Department of Mechanical, Aerospace & Biomedical Eng., University of Tennessee, Knoxville, USA

*Email: rbh3@illinois.edu

Abstract

Serial versions of the causal Spacetime Discontinuous Galerkin (cSDG) method compete with other methods running on multi-host platforms for challenging wave problems with wide ranges of length and time scales. This performance reflects the method's asynchronous structure and fine-grained adaptive meshing. However, an effective parallel–adaptive cSDG scheme using traditional domain decomposition has proven elusive. Rebalancing decompositions introduces severe synchronous barriers and is unable to keep pace with dynamic cSDG adaptive meshing.

We use lazy mesh refinement to localize adaptive meshing and, in place of domain decomposition, and a probabilistic scheme for distributing data and tasks continuously maintains load balance. We obtain a fully asynchronous, barrier-free software architecture that delivers excellent performance and scaling efficiency on multi-host platforms. Earthquake simulations exemplify the distributed cSDG implementation's multi-scale simulation capabilities.

Keywords: parallel, adaptive, spacetime, discontinuous Galerkin, earthquake

1 Introduction

The cSDG method depends on specialized spacetime finite element mesh generation. Figure 1 (left) shows a spacetime solution domain with one spatial dimension. Arrows on cell A indicate characteristic trajectories for waves traveling left and right. We impose a *causality constraint* on interior cell faces: they must be spacelike (closer to horizontal than the characteristic trajectories). This ensures that the solution in cell A depends only on solutions in cells B and C. If the overall solution is cell-wise discontinuous and the predecessor-cell solutions are available, the solution in cell A is fully determined and can be solved locally. Level-1 cells on the initial-time boundary can be solved in parallel using only initial data and boundary data. Any level-2 cell can be solved as soon as its level-1 predecessors have been solved, etc.

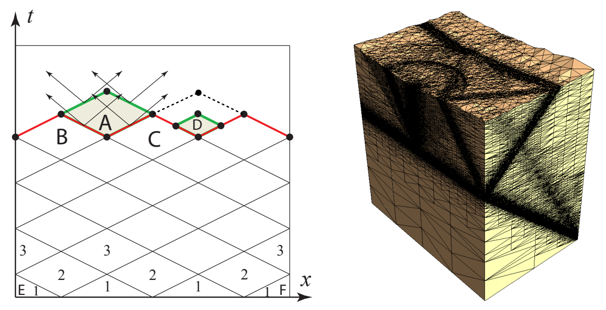

Figure 1: left: cSDG solution scheme in $1d \times$ time; right: adapted causal mesh in $2d \times$ time.

At any stage of the solution process, there is a *front mesh* (shown in red in Fig. 1) below which all cells have been solved. Each step in the cSDG solution process involves advancing a vertex to form a new cell, computing a local solution on the new cell, and updating the front mesh to its new configuration. We repeat this process until the spacetime mesh fills the spacetime analysis domain.

Adaptive refinement of the front mesh induces spacetime mesh refinement. If the solution on a larger cell constructed to the dashed outline near cell D in Fig. 1 has excessive error, we discard the larger cell, bisect its inflow (red) faces to refine locally the front mesh, and restart the cSDG meshing/solution process. This generates cell D which, due to the causality constraint, is refined in both space and time.

In practice, we use the Tent Pitcher algorithm [1] to fill the region between the old and new fronts with a small *patch* of simplex cells. Inter-cell boundaries within the patch are not necessarily causal, so the entire patch has to be solved simultaneously. Fig. 1 (right) shows an adaptively refined cSDG mesh at an intermediate stage of a $2d \times$ time simulation of crack-tip wave scattering in which high-resolution cSDG adaptive meshing captures moving wavefronts and quasi-singular crack-tip fields.

Suggested members of the Scientific Committee:

2 Parallel–Adaptive Implementation

Figure 2 diagrams a previous parallel–adaptive cSDG software architecture for a single multi-core, shared-memory host with two hardware threads per core. The front mesh and the *Pitchable Vertex Queues* (PVQs determine the patch-building order) are the only global, shared data. On each core i, one hardware thread handles meshing while the other solves patches. Threads run asynchronously and communicate exclusively via queues. All meshing and solve operations are embarrassingly parallel, with the exception of operations that access the global data to build and store front-mesh fragments called *footprints*.

In our new architecture, a templated *distributed vector* container class distributes front-mesh and PVQ data across meshing threads on multiple hosts. The distributed vector class handles internally non-blocking MPI access commands, recycling of storage freed by delete operations, and a probabilistic load-balancing distribution scheme. Solve threads only access private local data and are unchanged in the distributed architecture.

3 Simulation of Super-Shear Earthquake

Earthquake simulations present extreme computational challenges due to the wide range of scales — regional fault systems extend over thousands of kilometers while rupture process zones measure in hundreds of microns. State-of-the-art earthquake simulations are unable to bridge this range of scales, so it is common practice to use much larger process zone sizes in numerical models. For example, the Southern California Earthquake Center (SCEC) TPV205-2D benchmark problem uses a critical slip-weaking distance, D_c, that is 10^4 times larger than is physically reasonable.

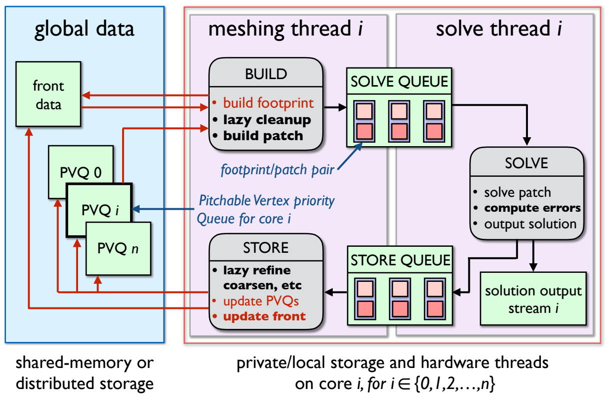

Figure 2: Parallel software architecture

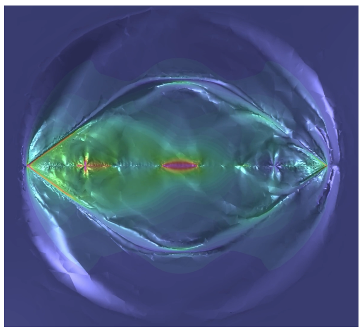

Figure 3: Super-shear rupture for reduced d_c in SCEC TPV205-2D benchmark problem.

We combined adaptive cSDG models for elastodynamics, dynamic contact, and dynamic fracture [2] with a slip-weakening friction model to obtain a powerful new earthquake simulation tool. When we modeled SCEC TPV205-2D as specified, our cSDG solutions were in good agreement with previous results where rupture-tip speeds remained below the shear-wave speed. Figure 3 shows the cSDG solution when we reduced D_c by a factor of 10. The left and right Mach cones confirm the presence of super-shear rupture speeds that approach the dilatational wave speed.

4 Conclusions

The SCEC TPV205-2D results underline the importance of resolving all relevant length and time scales in science and engineering simulation. The results shown here, obtained with a serial cSDG code, demonstrate the power of adaptive cSDG methods. Parallel–adaptive cSDG codes running on HPC platforms promise break-through performance capable of solving problems with realistic material-parameter values on regional-scale fault systems. New results will demonstrate the the distributed cSDG scheme's performance and scalability.

References

[1] R. Abedi, S.-H. Cheng, J. Erickson, Y. Fan, M. Garland, D. Guoy, R. Haber, J. Sullivan, S. Thite and Y. Zhou. Spacetime meshing with adaptive refinement and coarsening, in *20th Ann. Symp. Comp. Geometry, New York, USA, June 8–11, 2004*, pp. 300–309.

[2] R. Abedi, R.B. Haber and P.L. Clark, Effect of random defects on dynamic fracture in quasi-brittle materials, *Int J Fract* **208** (2017), pp. 241–268.

Suggested members of the Scientific Committee: