Parallel Space-time Spacetrees for Simple Parabolic Benchmarks

Tobias Weinzierl

Technische Universität München,
Boltzmannstrasse 3,
85748 Garching
weinzier@in.tum.de

Space-time schemes for parabolic partial differential equations (PDEs) promise advantages compared to traditional time stepping: First guesses of the solution's evolution in time are delivered quickly (an important ingredient for computational steering or interactive computing), periodic boundary conditions are straightforward to implement (important for steady cycle systems), backward problems interacting with the PDE can be solved directly (important for optimisation and calibration based upon the adjoint), and so forth [4]. The most prominent selling points of space-time however are the increased level of concurrency and superconvergence exploiting solution smoothness in time (see for example [1] for historic remarks). All these properties render space-time a promising candidate for the dawning exascale age.

Three major showstoppers for space-time meshings do exist: memory, efficient solvers, and software complexity. We propose to pick up the concept of spacetrees, a generalisation of the well-known quadtree/octree concept for two and three dimensions, to use three-partitioning, and to apply it to a four-dimensional space-time setting: here, a bounded space-time domain is embedded into a $4d$-hypercube. This cube then is cut into 3^4 equally-sized subcubes[1]. The process continues recursively and yields a cascade of adaptive Cartesian grids. Such grids can be traversed and serialised along a space-filling curve and thus stored efficiently with basically one bit per vertex [2, 5]. Such a grid cascade delivers a multiscale representation of the computational domain well-suited for geometric multigrid solvers. Such a grid finally has simple tensor-product structure. Existing solvers for Cartesian meshes with standard single step time integration can straightforwardly be rewritten as space-time code.

The present talk studies a plain geometric multigrid solver acting on the space-time domain, and it discusses some of the space-time advantages: We observe that the spacetree's adaptivity in both space and time can, if multiple snapshots at different time steps have to be held anyway, coarse more aggressively than time stepping and provide a nice framework to realise local time stepping in combination with dynamic adaptivity in space. We observe that the spacetree's multiscale data structure can, in combination with a full approximation storage scheme, yield first guesses of the solution quickly and, in accordance with textbook knowledge, superconverge. We observe that the additional temporal

[1] The factor three results from the use of the PDE framework Peano [3] as software base.

degree of freedom increases the concurrency. Different to traditional approaches, the spacetree allows us to distribute not only time slices or follow a spatial domain decomposition, but it facilitates to deploy whole space-time subdomains to different ranks. However, the load balancing for such a data structure is delicate, the adaptivity in time interplays with the communication and data flow, the h-adaptivity of the spacetree suffers from a lack of accuracy at the domain boundaries due to the $\mathcal{O}(h)$ accuracy, and so forth. Some of these issues and potential solutions are addressed and sketched.

References

1. M. Gander and S. Vandewalle. Analysis of the parareal time-parallel time-integration method. *SIAM Journal on Scientific Computing*, 29(2):556–578, 2007.
2. T. Weinzierl. *A Framework for Parallel PDE Solvers on Multiscale Adaptive Cartesian Grids*. Verlag Dr. Hut, 2009.
3. T. Weinzierl et al. Peano—a Framework for PDE Solvers on Spacetree Grids, 2012. www.peano-framework.org.
4. T. Weinzierl and T. Köppl. A geometric space-time multigrid algorithm for the heat equation. *Numerical Mathematics: Theory, Methods and Applications*, 5(1):110–130, February 2012.
5. T. Weinzierl and M. Mehl. Peano – a traversal and storage scheme for octree-like adaptive cartesian multiscale grids. *SIAM Journal on Scientific Computing*, 33(5):2732–2760, October 2011.