11th World Congress on Computational Mechanics (WCCM XI)
5th European Conference on Computational Mechanics (ECCM V)
6th European Conference on Computational Fluid Dynamics (ECFD VI)
July 20–25, 2014, Barcelona, Spain

TASK-BASED DECOMPOSITION OF A HIGHER-ORDER METHOD ON COMPLEX GEOMETRIES

Tobias Weinzierl[1] **and Roland Wittmann**[2]

[1] School of Engineering and Computing Sciences, Durham University,
Stockton Road, Durham DH1 3LE, GREAT BRITAIN, tobias.weinzierl@durham.ac.uk
[2] Department of Informatics, Technische Universität München,
Boltzmannstr. 3, 85748 Garching, GERMANY, wittmanr@in.tum.de

Key words: *Spacetrees, Finite Cell Method, Task-based Programming*

Spacetrees, a generalisation of the octree concept [5], and their variations have established themselves as mature technique to describe dynamically adaptive block-structured meshes. While they inherently facilitate h-adaptivity with in-situ mesh generation and yield a multiscale domain decomposition, they suffer from an unfavourable boundary approximation once the domain becomes nontrivial. Their meshes are not boundary-fitted and thus in $\mathcal{O}(h)$. Furthermore, a tree representation of a block-structured mesh does not fit to today's computing hardware tailored to vector operations—trees have to be processed element-by-element if no regularity constraint or preprocessing is applied, and the computational workload per spacetree cell is limited while tracking the geometric mesh layout and handling adaptivity requires some pre- and postprocessing effort. This effort might be acceptable if the tree processing phase consumes only a minor time of the overall algorithm. It is unacceptable for codes that realise matrix-free in-situ linear algebra or have to reassemble over and over again. The latter are targeted by the present work. We study as use case a simple parabolic partial differential equation applying a matrix-free iterative scheme where the elliptic solve process degenerates to one or two sweeps after few initial time steps [4].

Facing the hardware challenge, higher order methods improve the ratio of operations per loaded byte. The less sparser the underlying matrices are the better suit the operations to vector facilities. A breakthrough however is obtained once we merge multiple spacetree nodes into one regular grid Cartesian grid and apply the higher order stencil—with loops accurately fused and vector units exploited—there. This can be done on-the-fly, i.e. we can identify regular subgrids on the fly and handle them as one patch rather than individual cells [1, 6].

Facing the boundary approximation challenge, it is a natural choice to apply immersed boundary techniques such as the finite cell method ([2], e.g.) within such a higher-order setting. However, this misfits to the cell-merge/-fusion technique discussed before: code for boundary cells typically is not a plain nested loop anymore since it introduces additional code fragments and case distinctions to handle the boundary and boundary conditions, code for boundary cells typically comes along with successive domain oversampling (either as a cascade of finer and finer regular grids or as recursive bisection) yielding nonuniform computational cost per cell, and code resolving the boundary accurately often still yields a certain h-adaptivity despite sophisticated boundary treatment techniques.

We propose to label boundary cells in the spacetree as well as regular subgrids, and to restrict these labels bottom-up within the tree. Such a formalism is an analysed (space-)tree grammar. Subsequent mesh traversals realised as spacetree push-back automatons then can decompose the mesh traversal into a set of tasks. Atomic tasks either represent boundary cells or standard cells that do not belong to a regular subgrid structure. Or they represent collections of multiple inner cells that are tied into a regular Cartesian grid patch [6]. The latter tasks yield the MFLOPSs, while the boundary tasks improve the algorithm's accuracy. Task stealing on shared memory architectures ensures that the boundary handling does not thwart all cores' vector processing units. All code is available as open source [3].

REFERENCES

[1] W. Eckhardt and T. Weinzierl. A Blocking Strategy on Multicore Architectures for Dynamically Adaptive PDE Solvers. In R. Wyrzykowski, J. Dongarra, K. Karczewski, and J. Wasniewski, editors, *Parallel Processing and Applied Mathematics, PPAM 2009*, volume 6068 of *Lecture Notes in Computer Science*, pages 567–575. Springer-Verlag, 2010.

[2] M. Ruess, D. Schillinger, A.I. Özcan, and E. Rank. Weak coupling for isogeometric analysis of non-matching and trimmed multi-patch geometries. *Computer Methods in Applied Mechanics and Engineering*, 269:46–71, 2014.

[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.

[6] T. Weinzierl, R. Wittmann, K. Unterweger, M. Bader, A. Breuer, and S. Rettenberger. Hardware-aware block size tailoring on adaptive spacetree grids for shallow water waves. In *HiStencils 2014—1st International Workshop on High-Performance Stencil Computations*, 2014. (submitted).