

Multilevel Methods Combine All Properties that are Essential for Fast and Efficient Performance at the Extreme Scale

Computational science is facing several major challenges with future architectures: non-increasing clock speeds are being offset with added concurrency (more cores) and limited power resources are leading to reduced memory per core, complex heterogeneous architectures, and higher levels of hardware failures (faults). To meet these challenges and yield fast and efficient performance, solvers need to exhibit extreme levels of parallelism, minimize data movement, and demonstrate resilience to faults.

For many DOE problems, multilevel methods are the only truly scalable solver approaches, because they are *mathematically optimal* and they display *excellent parallelization properties*. As a result, these solvers are now widely used in scientific simulation codes. For example, algebraic multigrid has shown excellent weak scaling on up to more than 1 million cores and about 4.5 million hardware threads for diffusion problems. The importance of these methods does not lessen because of the new architecture challenges we are facing. Rather the need to develop new multilevel methods and techniques is greater than ever.

To elaborate on this without overcomplicating the discussion, consider multigrid methods for linear systems, keeping in mind that many of the basic comments and conclusions carry over to the general setting. Multigrid methods are called *optimal* methods because they can solve a linear system with N unknowns with only $O(N)$ work. This property gives them the potential to solve ever larger problems on proportionally larger parallel machines in constant time. Multigrid methods achieve this optimality by employing two complementary processes: *smoothing* and *coarse-grid correction*. In the classical setting of scalar elliptic problems, the smoother (or relaxation method) is a simple iterative algorithm like Gauss-Seidel that is effective at reducing high-frequency error. The remaining low-frequency error is then accurately represented and efficiently eliminated on coarser grids via the coarse-grid correction step. Applying this simple multigrid idea to get a scalable multilevel method often involves considerable algorithmic research, however. One has to decide which iterative method to use as a smoother, how to coarsen the problem, and how to transfer information between the levels. When designed properly, a multilevel solver will be algorithmically scalable; it will uniformly damp all error frequencies with a computational cost that depends only linearly on the problem size. One consequence of this design is that multilevel solvers have *natural resilience properties*, as has been demonstrated for algebraic multigrid [1].

It is well known that the overall time spent by a parallel multilevel method on an architecture with p processors depends on $O(\log p)$ and consequently for very large p , scalability will be impacted. However, because of their optimality, multilevel approaches will still be the fastest methods for various problems. The approach most commonly used to parallelize multilevel methods is a straightforward data decomposition of the fine and coarse grid systems such that all coarse grid subdomains owned by a processor are nested (or nearly so). This minimizes communication during inter-grid transfers, but it has the side effect that on the coarsest levels, some processors do not own grid points and hence become idle during phases of the algorithm that involve those grids. It is important to note that, in the sense of solving systems as quickly as possible, this is not a major algorithmic issue at any scale of parallelism,

since multilevel methods not only yield optimal-order work, but also exhibit optimal-order data motion in parallel. Since data movement is responsible for a majority of the power consumed in a system, this property also makes them *naturally power efficient*.

The development of multilevel methods at the extreme scale will require both the investigation of old ideas that were abandoned since they turned out to be inefficient on older or even current architectures, such as the utilization of idle processes on coarser levels to accelerate convergence, multilevel domain decomposition methods, or additive multigrid methods, as well as new ideas. It will require both evolutionary as well as revolutionary approaches.

One approach that has the potential to revolutionize time stepping methods is the application of multigrid methods for computing multiple time steps simultaneously. Since clock speeds are no longer increasing, a significant challenge for the computational science community on future computer architectures is to overcome the sequential nature of current time integration methods, which will be a significant bottleneck. Solving for multiple time steps in parallel would remove this bottleneck, and a feasible way to achieve this is through multilevel methods. However, developing such methods requires significant research. The development of such an algorithm in a two-grid setting (parareal) has already been successful, and there are promising multi-level results for a model test problem [2], which indicate the potential for speedups of several orders of magnitudes on future exascale architectures.

Since exascale computers are currently not available, it will be important to develop performance models [3] that will guide our research. They can help to evaluate new algorithms as well as modifications to current algorithms, identify bottlenecks, and predict performance on machines that have not been built yet.

In conclusion, well-designed multilevel algorithms are ideally suited for exascale computing, since they combine all properties that are essential for efficient performance at the extreme scale, i.e. they are mathematically optimal, efficiently parallelizable, resilient, power efficient, and will be the fastest solvers for various problems.

[1] M. C. Guix, B. R. de Supinski, G. Bronevetsky, and M. Schulz, *Fault Resilience of the Algebraic Multigrid Solver*, in International Conference on Supercomputing (ICS), 2012.

[2] S. Friedhoff, R. Falgout, T. Kolev, S. MacLachlan, and J. Schroder. *A Multigrid-in-Time Algorithm for Solving Evolution Equations in Parallel*, 2013. Student paper winner at the Copper Mountain Conference on Multigrid Methods, Copper Mountain, CO, March 17-22, 2013.

[3] H. Gahvari, A. H. Baker, M. Schulz, U. M. Yang, K. E. Jordan, and W. Gropp, *Modeling the Performance of an Algebraic Multigrid Cycle on HPC Platforms*, in Proceedings of the 25th International Conference on Supercomputing, Tucson, AZ, 2011, pp. 172–181.

Contact: Ulrike Meier Yang

Lawrence Livermore National Laboratory, Box 808, L-561, Livermore, CA 94551

Email: umyang@llnl.gov

ph: (925)422-2850