ALGORITHMS FOR EXTREME-SCALE SYSTEMS

Allocation: Blue Waters Professor/0.24 Mnh
PI: William Gropp[1]
Collaborators: Luke Olson[1], Jack Dongarra[2]

[1]University of Illinois Urbana–Champaign
[2]University of Tennessee

EXECUTIVE SUMMARY:

Continued increases in the performance of large-scale systems will come from greater parallelism at all levels. At the node level, we see this in both the increasing number of cores per processor and the use of large numbers of simpler computing elements in GPGPUs (general-purpose processing on GPUs). The largest systems must network tens of thousands of nodes together to achieve the performance required for the most challenging computations. Successfully using these systems requires new algorithms and new programming systems. This research looks at the effective use of extreme-scale systems. Over the last year, we have explored alternative formulations of a conjugate gradient method that eliminate some of the strict barrier synchronization as well as use the memory hierarchy more effectively. Other exploratory studies have begun looking at the scalability and fault tolerance of an algebraic multigrid method, scaling for large graph problems, and the benefit of lightweight intra-node balancing to scalability and performance.

INTRODUCTION

At extreme scale, even small inefficiencies can cascade to limit the overall efficiency of an application. New algorithms and programming approaches are needed to address barriers to performance. This work directly targets current barriers to effective use of extreme-scale systems by applications. For example, Krylov methods such as the conjugate gradient method are used in many applications currently being run on Blue Waters (MIMD Lattice Computation—MILC—is one well-known example). Developing and demonstrating a more scalable version of this algorithm would immediately benefit those applications. Longer term, the techniques that are developed will guide the development of highly scalable applications.

METHODS & RESULTS

Early results with alternative Krylov method formulations have revealed several performance effects that can provide a factor of two or more improvement in performance at scale. Current work has been limited by the fact that the non-blocking MPI_Allreduce on Blue Waters is functional but does not provide the expected (or perhaps hoped for) performance, particularly in terms of the ability to overlap Allreduce with other communication and computation. A need for sparse matrix formats for GPUs for several other areas of interest drove work in this area, though it did not use any of the Blue Waters allocation in this year; that work will be used in the upcoming year.

WHY BLUE WATERS?

Scalability research relies on the ability to run experiments at large scale, requiring tens of thousands of nodes and hundreds of thousands of processes and cores. Blue Waters provides one of the few available environments where such large-scale experiments can run. In addition, only Blue Waters provides a highly capable I/O system, which we plan to use in developing improved approaches to extreme-scale I/O