TN

ALGORITHMS FOR EXTREME-SCALE SYSTEMS

Allocation: Blue Waters Professor/80 Knh
PI: William Gropp[1]
Collaborator: Luke Olson[1]
Students: Amanda Bienz[1], Paul Eller[1], Philipp Samfass[1]

[1]University of Illinois at Urbana-Champaign

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 both the increasing number of cores per processor and the use of large numbers of specialized computing elements in GPGPUs (general-purpose graphics processing units). 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. Over the last year, we have explored two ways to reduce communication costs in large-scale systems. One is the redesign of algorithms to take into account the difference between on-node and off-node communication performance. The other is to look at the effect of performance irregularity and the use of nonblocking collectives to improve performance of algorithms that use MPI (message passing interface) collectives. We also developed an improved communication model that better matches the performance of modern parallel processors.

RESEARCH CHALLENGE

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 sustained performance.

This work directly targets current barriers to effective use of extreme scale systems by applications. For example, Krylov methods such as Conjugate Gradient are used in many applications currently being run on Blue Waters (MILC is one well-known example) and other leadership-class systems. Developing and demonstrating a more scalable version of this algorithm would immediately benefit those applications. Also of importance to many computations, including Krylov methods for solving large systems of linear equations as well as methods for large-scale graph computations, are sparse-matrix vector multipliers. These involve significant communication between nodes and can lead to scalability limits; by improving methods to exploit internode and intranode communication, many applications can improve their scalability. In the longer term, the techniques that are developed in this project will provide guidance for the development of highly scalable applications.

METHODS & CODES

To address the challenges of parallelism and scale, we developed several codes that allow us to benchmark the performance of these operations, gather detailed timing results, and perform experiments with different approaches. For example, we have been developing a "noise injector" to allow us to better experiment with different amounts of performance variation in multicore nodes. We have also developed a set of benchmark codes that better measure the achievable communication performance of the communication patterns commonly used in applications.

RESULTS & IMPACT

Early results with alternative Krylov formulations have revealed several performance effects that can provide a factor of two or more improvement in performance at scale [1]. We have been using Blue Waters over the last year for an investigation into the impact of large-scale system performance variation on parallel numeric algorithms. This includes developing code for measuring and processing network performance counters, injecting network noise into nodes running another algorithm, and kernels with a variety of common communication patterns. Our initial experiments have used smaller core counts to assist in developing the code and improving the experiments in preparation for a more detailed study in the coming months involving large runs. The goal of this study is to better understand what network noise looks like on supercomputers and how we can develop parallel numeric algorithms that perform well despite noise.

We have also explored the performance models used to guide both the development of algorithms and the analysis of application performance. We discovered that the classic "postal model" is no longer effective for systems with multi-core nodes, and we developed a simple extension to the postal model that explains the observed performance of current systems [2]. This work has also informed the development of an improved approach to sparse matrix–vector multiplication that takes into account the different performance of inter- and intra-node communication [3].

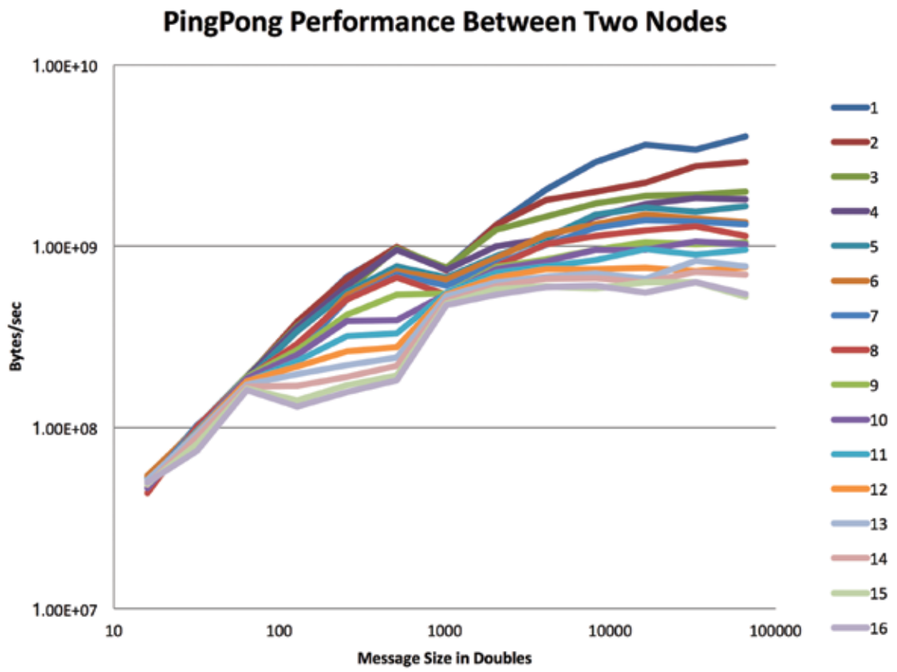

Figure 1: Communication performance between two nodes with 1–16 processes communicating at the same time. Note that as the number of communicating processes increases, the achieved performance per process drops, with nearly an order of magnitude loss between one and 16 communicating processes at large messages.

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 environments available for large-scale experiments. 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.

PUBLICATIONS AND DATA SETS

Eller, P. R., and W. Gropp, Scalable nonblocking preconditioned conjugate gradient methods. *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis* (IEEE Press, Salt Lake City, Utah, November 13–18, 2016), pp. 1–12.

Gropp, W., et al., Modeling MPI Communication Performance on SMP Nodes: Is it Time to Retire the Ping Pong Test? *Proceedings of the 23rd European MPI Users' Group Meeting* (ACM Digital Library, Edinburgh, United Kingdom, September 25–28, 2016), pp. 41–50.