

Partitioned Global Address Space Languages on the IBM BlueGene/P

Rajesh Nishtala[†], Paul H. Hargrove[‡], Dan O. Bonachea[†], Katherine A. Yelick^{†‡}

[†]University of California at Berkeley, [‡]Lawrence Berkeley National Labs, Berkeley, CA, USA
{rajeshn,bonachea,yelick}@cs.berkeley.edu phhargrove@lbl.gov

As the number of cores in a socket and within a system continue to grow at an exponential rate, the ability to scale communication systems, programming models, and applications to this large scale once again takes center stage. Further aggravating the problem, the pressures of power and machine cost at these large scales motivate the use of lower core clock rates leading to weak integer performance relative to lower-scale predecessor systems. A modern example of a machine in this class is the IBM BlueGene/P (BG/P). The relatively weak compute cores on the BG/P magnify the software overheads associated with the communication subsystem. Semantic matching between the higher level programming model and the underlying network hardware can help alleviate some of these overheads.

A new class of languages, called Partitioned Global Address Space (PGAS) languages, has recently emerged to aid in the performance and scalability of HPC applications. Rooted in traditional shared memory programming models, these languages expose a global address space that is logically partitioned across the threads. To provide the illusion of a globally shared memory these languages employ a one-sided communication model whereby a thread may directly read and write the memory located at a remote node, without the explicit cooperation of the application thread(s) on the remote node. One-sided communication semantics have been shown to enable increased performance by decoupling processor synchronization from data transfer, operations which are implicitly linked in two-sided communication models such as MPI message-passing.

In our most recent work [1] we have ported the GASNet communication layer [2] and the Berkeley Unified Parallel C (UPC) [3] compiler to the BG/P. Like MPICH2 1.0.7, the GASNet implementation on BG/P targets the lower level open source Deep Computing Messaging Framework (DCMF) [4] to manage the interprocess communication. To the best of our knowledge our Berkeley UPC compiler is the first PGAS compiler available on the BG/P and this work comprises some of the largest-scale PGAS application numbers published to date.

In order to benchmark communication performance we use the NAS Parallel Benchmark FT [5]; the core of which is a large three-dimensional FFT, a problem that is limited by the bisection bandwidth

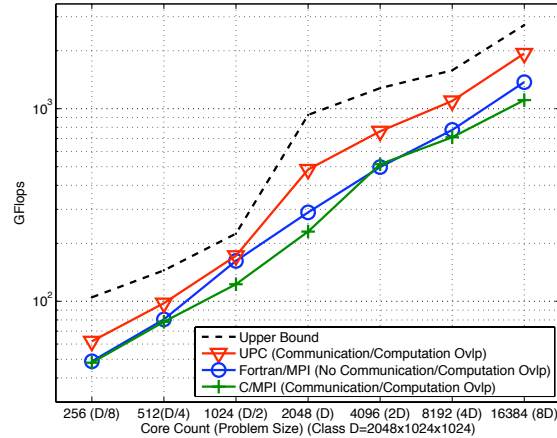


Figure 1: NAS FT Weak Scaling on IBM BlueGene/P

performance of the network. We focus on how the one-sided communication semantics found in UPC are a closer fit to the underlying hardware and thus improve overall application performance. In addition, we study the efficacy of communication/computation overlap. By simultaneously using both the computation and communication subsystems we can realize better application performance. As shown in Figure 1, we scale the benchmark up to 16,384 cores of the BG/P and demonstrate that UPC consistently outperforms MPI by as much as 66% for some processor configurations and an average of 32%. In addition, the results demonstrate the scalability of the PGAS model and the Berkeley implementation of UPC, the viability of using it on machines with multicore nodes, and the effectiveness of the BG/P communication layer for supporting PGAS languages.

References

- [1] Rajesh Nishtala et al. Scaling Communication-Intensive Applications on BlueGene/P Using One-Sided Communication and Overlap (to appear). In *International Parallel and Distributed Processing Symposium (IPDPS)*, 2009.
- [2] GASNet website. <http://gasnet.cs.berkeley.edu/>.
- [3] The Berkeley UPC Compiler. <http://upc.lbl.gov>.
- [4] Sameer Kumar et al. The Deep Computing Messaging Framework: Generalized scalable message passing on the BlueGene/P supercomputer. In *International Conference on Supercomputing (ICS)*, 2008.
- [5] David H. Bailey et al. The NAS Parallel Benchmarks. *The International Journal of Supercomputer Applications*, 5(3):63–73, Fall 1991.