Euro-Par 2005
30 August - 2 September - Lisboa, Portugal
Departamento de Informática
Faculdade de Ciências e Tecnologia
Universidade Nova de Lisboa

Early Experience with Scientific Applications on the Blue Gene/L Supercomputer

George Almasi[1], Gyan Bhanot[1], Dong Chen[1], Maria Eleftheriou[1], Blake Fitch[1], Alan Gara[1], Robert Germain[1], John Gunnels[1], Manish Gupta[1] Philip Heidelberg[1], Mike Pitman[1], Aleksandr Rayshubskiy[1], James Sexton[1], Frank Suits[1], Pavlos Vranas[1], Bob Walkup[1], Chris Ward[1], Yuriy Zhestkov[1], Alessandro Curioni[2], Wanda Andreoni[2], Charles Archer[3], Jose Moreira[3], Richard Loft[4], Henry Tufo[4,5], Theron Voran[5], and Katherine Riley[6]

[1]IBM T. J. Watson Research Center, Yorktown Heights, NY, USA
{gheorghe, gyan, chendong, mariae, bgf, alangara, rgermain, gunnels, mgupta, philiph, pitman, arayshu, sextonjc, suits, vranasp, walkup, tjcw, yuriyz}@us.ibm.com
[2]IBM Zurich Research Laboratory, CH-8803 Rüschlikon, Switzerland
{cur, and}@zurich.ibm.com
[3]IBM Systems and Technology Group, Rochester, MN, USA
{archerc, jmoreira}@us.ibm.com
[4]National Center for Atmospheric Research, Boulder, CO, USA
loft@ucar.edu
[5]University of Colorado at Boulder, Boulder, CO, USA
{tufo, theron.voran}@cs.colorado.edu
[6]Argonne National Laboratory, Argonne, IL, USA
riley@mcs.anl.gov

Abstract. Blue Gene/L uses a large number of low power processors, together with multiple integrated interconnection networks, to build a supercomputer with low cost, space and power consumption. It uses a novel system software architecture designed with application scalability in mind. However, whether real applications will scale to tens of thousands of processors has been an open question. In this paper, we describe early experience with several applications on a 16,384 node Blue Gene/L system. This study establishes that applications from a broad variety of scientific disciplines can effectively scale to thousands of processors. The results reported in this study represent the highest performance ever demonstrated for most of these applications, and in fact, show effective scaling for the first time ever on thousands of processors.

1 Introduction

A popular approach to building supercomputers has been to build clusters of high performance nodes (based on symmetric multiprocessors or vector processors) with high performance interconnection networks. Examples of systems that have been built, or are being built, using this approach include the Earth Simulator [1], ASC Purple [2], and the Columbia [3] systems. At the highest scales, these machines consume a great deal of power and require a lot of floor space. For example, the Earth Simulator, which delivers a peak performance of 41 Teraflop/s, consumes about 7 MW of power, and occupies an area of about 70,000 square feet.

The IBM Blue Gene/L (BG/L) [4] represents a different way of building supercomputers. It uses low power processors, which allows a large number of processors to be packed in a given volume (2048 processors in a rack), with aggregate heat dissipation staying within air cooling limits. Furthermore, it uses system-on-a-chip technology to integrate powerful torus and collective networks, and uses a novel software architecture [5] to support high levels of scalability. While BG/L offers the promise of making massively parallel systems accessible, and promising results have been reported on a 512 node prototype [6], how far the applications can scale has been an open question. Previous results on scaling of MPI applications on any platform have necessarily been limited (by hardware existence) to fewer than ten thousand processors. Many previous studies have shown problems with the scaling of applications to thousands of processors due to factors like computational noise [7].

This paper, together with a companion paper [8] on physics and material science applications developed by researchers at Lawrence Livermore National Laboratory, explores the scaling of applications to thousands of processors. It describes early experience with several scientific applications on a 16,384 node BG/L system. This system, which occupies less than 400 square feet of floor space, and consumes about 400 KW in power, was recently rated as the fastest supercomputer in the world (#1 on the TOP500 list), with a sustained LINPACK performance of 70.72 Teraflop/s [9]. We show that the scientific applications targeted in this study scale well on the BG/L system, thus validating the design of BG/L and establishing the ability to scale MPI applications to several thousand processors. This study also uncovers several opportunities for performance improvements through software optimizations.

2 Overview of BG/L

This section reviews some architectural features of BG/L that have a significant impact on performance.

2.1 BG/L Hardware

Each BG/L node [1] has two 32-bit embedded PowerPC (PPC) 440 processors, which have 32 KB each of L1 data and instruction caches. The BG/L nodes support prefetching in hardware, based on detection of sequential data access. The prefetch buffer for each processor holds 64 L1 cache lines (16 128byte L2/L3 cache lines) and is referred to as the L2 cache. Each chip also has a 4 MB L3 cache built from embedded DRAM, and an integrated DDR memory controller. A single BG/L node supports 512 MB memory. The PPC 440 processor does not support hardware cache coherence at the L1 level. However, there are instructions to invalidate a cache line or flush the cache, which can be used to manage coherence in software.

BG/L employs a SIMD-like extension of the PPC floating-point unit, which we refer to as the double floating point unit or DFPU [10]. The DFPU adds a secondary FPU to the primary FPU as a duplicate copy with its own register file. BG/L supports a comprehensive set of parallel instructions on double-precision floating-point data.

The BG/L ASIC supports five different networks: torus, collective, global interrupts, Ethernet, and JTAG. The main communication network for point-to-point messages is a three-dimensional torus. Each node contains six bi-directional links for direct connection with nearest neighbors. The raw hardware bandwidth for each torus link is 2 bits/cycle (175 MB/s at 700 MHz) in each direction. The torus network provides both adaptive and deterministic minimal path routing in a deadlock-free manner. The collective network implements broadcasts and reductions with a target hardware latency of 1.5 microseconds for a 64K node system. The global interrupts network supports a fast barrier operation, also with a target latency of 1.5 microseconds for a 64K node system. On BG/L, I/O is supported via special I/O nodes, which are architecturally identical to compute nodes, but are attached to the Gbit/s Ethernet network, which connects the BG/L core to external file servers and host systems. The booting, control and monitoring of the BG/L system is done over the JTAG network.

2.2 BG/L Software

The programming model supported in BG/L is *single program multiple data* (SPMD), with message passing supported via an implementation of the Message Passing Interface (MPI). A BG/L job can be submitted in one of two modes. In *coprocessor mode* (CPM), which is the default mode, a single application (MPI) process runs on each compute node – one of the processors of the compute node is used for computation, and the other is used for offloading part of the communication operations. In *virtual processor mode* (VNM), two application processes are run on each compute node, one on each of the two processors.

BG/L uses a hierarchical organization of software, described in further details in [5]. User applications run exclusively on compute nodes under the supervision of a simple, minimalist *compute node kernel* (CNK). The I/O nodes run a customized version of Linux. Many system calls (such as *read* and *write*) are not directly executed in the compute node, but are function shipped through the collective network to the “parent” I/O node. The control system is implemented as a collection of processes running in an external computer, called the *service node* for the machine. All of the visible state of BG/L is maintained in a commercial database on the service node.

BG/L provides an operating environment with a very low level of “computational noise” (interference from operating system activity), about two orders of magnitude lower than traditional clusters and the ASCI Q system [11]. It also supports low latency communication (latency to nearest neighbor is about 3.3 microseconds, or 2350 processor cycles), with a low *half-bandwidth* point (half of asymptotic bandwidth is achieved at a message size smaller than 1 KB for several MPI bandwidth tests, such as point-to-point sends to all nearest neighbors and *alltoall* collective operation).

3 Applications Performance Results

This section describes the applications we used in this study and the performance of these applications on the BG/L system at IBM Rochester.

3.1 Blue Matter

The *Blue Matter* application framework has been developed at IBM Research to advance our understanding of biologically important phenomena such as protein folding through large scale simulation [12]. In addition to molecular dynamic simulations, the Blue Gene science team also aims to run using the “replica exchange” method in which a large number (32-128) of simulations are run at different temperatures and are coupled via periodic exchanges using a Metropolis Monte Carlo type criterion. This yields an application with a hierarchy of communication—tight coupling within individual trajectories and loose coupling between them. The loose coupling comes about because carrying out the exchange attempts only requires an “all gather” of a single double precision number from each replica once every few hundred to few thousand time steps. Using this technique one can use thousands of nodes with a parallel efficiency determined by that of the component molecular dynamics simulations.

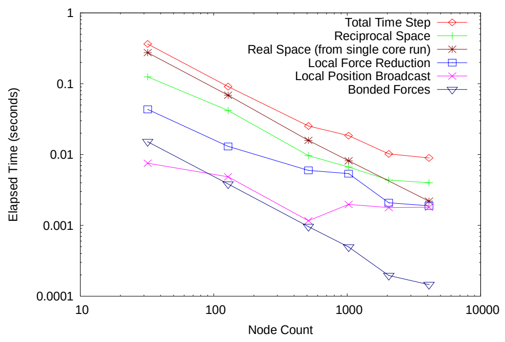


Fig. 1. Performance of Blue Matter on BG/L

Figure 1 shows the total time for a single iteration on a 43,222 atom system as a function of node count, and also the major contributions to the total time. We are now using a parallel decomposition that maps the simulation volume directly onto the 3D torus topology of BG/L and have removed our dependence on the fixed and floating point *Allreduce* collectives used in our previous work [13]. The current decomposition employs a neighborhood broadcast and reduction that is implemented using the *Alltoallv* collective. We carry out the reciprocal space operations requiring the computation of a 3D-FFT on CPU 0 while off-loading the real space non-bond computations (which require no communication) on to CPU 1. The scalability plot of the real space non-bond computation was taken from a separate run that used only one CPU because our tracing utility only functions on the first CPU.

Overall, as Figure 1 shows, we observe nearly ideal speedup up to 512 nodes and a drop in parallel efficiency to about 32% at 4096 nodes. This is scalability to a node count and value of atoms/node in a biomolecular simulation with proper treatment of electrostatics that is unique—for reference, the NAMD package scaled to 2250 nodes (on a 92K atom system) on the PSC Lemieux system before losing performance [14]. From the data plotted in Figure 1, we can extrapolate the parallel efficiency of the replica exchange technique, which uses multiple trajectories, for a 128 replica simulation of the 43K atom system using 512 nodes/replica to be over 90% on 64K nodes (normalizing the parallel efficiency to be 100% at 32 nodes for a single replica).

3.2 Car-Parrinello Molecular Dynamics (CPMD)

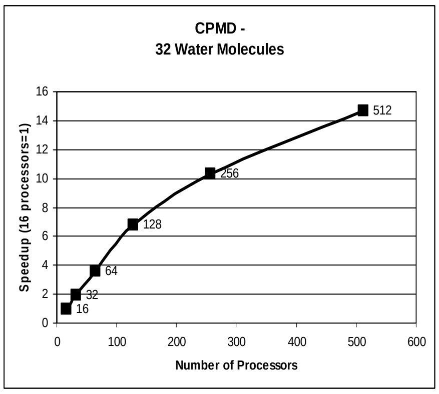
The Car-Parrinello Molecular Dynamics code (CPMD) originated in IBM's Zurich Research Laboratory in the early 1990s [15]. It is an implementation of density functional theory using plane waves and pseudopotentials, and is particularly designed for ab-initio molecular dynamics. The CPMD code is widely used for research in computational chemistry, materials science, and biology. It has been licensed to several thousand research groups in over 50 countries.

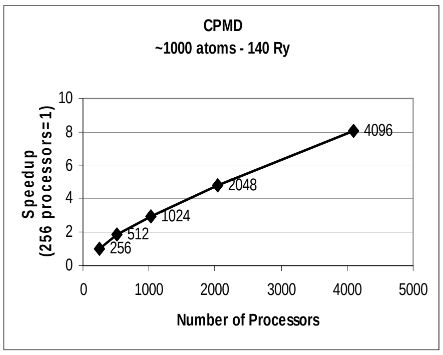
The application is mainly written in Fortran, parallelized for distributed-memory with MPI, with an option to add an additional level of parallelism using OpenMP for multi-processor nodes. CPMD makes extensive use of three-dimensional FFTs, which require efficient all-to-all communication [16]. The scalability was improved using a task-group implementation of the FFT with a special mapping to the BG/L torus [17]. Moreover, overlap matrices, which were replicated in the standard CPMD code, have been distributed on a subset of the nodes – to be able to handle large systems (more than 3000 electronic states). The single processor performance of CPMD was optimized for BG/L using SIMD-enabled routines for the most common calls such as DGEMM, DCOPY, AZZERO, and FFT [6].

Figure 2 shows strong-scaling tests on two systems. The first is a small test case of about 100 atoms – a 32 molecule model of liquid water, with a 70 Rydberg cutoff, while the second is a 1000 atom model of the liquid/vapour interface of methanol, with a cutoff of 140 Rydberg. In both cases a gradient corrected form (PBE) of the exchange-correlation functional was used.

For the first system (Figure 2a), a good scaling up to 512 processors is obtained (with a parallel efficiency greater than 40%). This corresponds to the limit of the data granularity of the model. Using more processors would have meant that some processors remain idle for a significant portion of the time. Low latency in the MPI layer and a total lack of system daemons contribute to very good scalability on BG/L. The execution time on 512 BG/L processors was less then 0.35 seconds per step; which is much better than the 1.5 seconds per step that we obtained on IBM p690 SMP servers (1.3 GHz) clustered via double colony switches, where scalability was limited to 128 processors (the Federation switch would enhance this performance by ~30%, but would not enhance scalability). The result on BG/L represents the highest performance obtained for ab-initio molecular dynamics simulation for a system with about 100 atoms [17]. This allows one to simulate a system of this size with fully ab-initio methods with throughput of 175 seconds per day using a fraction of a BG/L rack.

The second test case (Figure 2b) has better data granularity because it is 100 times larger in terms of the amount of data storage required. For this reason, this test case is not latency bound and good scaling up to 4096 processors was observed. The parallel efficiency is more than 90% up to 1024 processors and more than 50% up to 4096 processors. This was obtained by using a taskgroup parallelization scheme and an optimized mapping to the BG/L hardware [17].





(a) (b)

Fig. 2. Strong scaling of CPMD on BG/L: (a) ~100 atom simulation (speedup normalized to 16 procs), (b) ~1000 atom simulation (speedup normalized to 256 procs)

The third test case is a prototypical case in which the linear algebra involved in the orthogonalization of the electronic states starts to dominate the computation (here we have more than 5000 electronic states) and a newly implemented distributed orthogonalization algorithm becomes essential in order to fit the problem into the available memory per node (512MB). Figure 3 shows the sustained performance (considering the number of measured floating point operations and not pseudo operations) for a path-integral ab-initio molecular dynamics simulation, where an additional parallelization layer (over the replicas) is available. We obtain a performance close to 46% of peak at 16K processors and 38% at 32K processors; this value is quite good, especially considering the application, which has memory-intensive and network-intensive algorithms like the 3D-FFT.

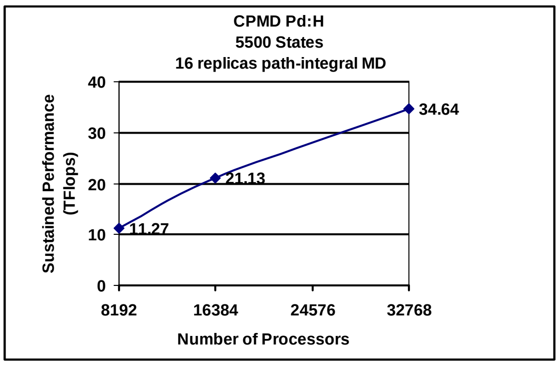


Fig. 3. Strong scaling of CPMD to 32K processors (in virtual node mode)

3.3 FLASH

FLASH is a parallel adaptive-mesh multi-physics simulation code designed to solve nuclear astrophysical problems related to exploding stars [18]. It has been developed as part of an ASC, DOE project at the University of Chicago, and won a Gordon Bell prize in 2000. The FLASH code solves the Euler equations for compressible flow and the Poisson equation for self-gravity.

The test run on BG/L was a highly resolved two-dimensional weak scaling sod (Sod, G. 1978, JCP, 27, 1) problem. The simulation exercises the core piece-wise parabolic hydrodynamics of FLASH and aggressively exercises the adaptive grid. The FLASH sod problem has been a standard benchmark for FLASH developers evaluating the scalability of a system's inter-connect. For most FLASH science runs, total memory and communication between nodes are the most important limiting factors; users need lots of memory and a scalable inter-connect between it all. Therefore, weak scaling is the primary FLASH benchmark target.

Currently, simulations based on FLASH are often run on hundreds to thousands of processors on systems like IBM SP Seaborg (NERSC), QSC (LANL) and MCR Linux cluster (LLNL). It has been run up to 16,384 nodes on BG/L and the total times for various platforms for a weak scaling study (fixed problem size per processor) are presented below. The three different BG/L systems listed are evolutions of the system at IBM Watson. The figure shows that on the QSC and MCR systems, FLASH scales poorly beyond 256 processors. On the Seaborg system, FLASH scales well below 1024 processors, but runs into problems beyond that level. On BG/L, FLASH scales almost perfectly up to 16K processors (on a 16K node system in coprocessor mode). Experience on other systems has demonstrated explicit sensitivity to the inter-connect during the re-gridding phase of the multigrid algorithm. We believe that the excellent scaling on BG/L is due to a combination of good message passing performance and low level of computational noise ensured by BG/L software. We suspect that the synchronization maintained by low noise eliminates overhead in global operations normally lost to barriers.

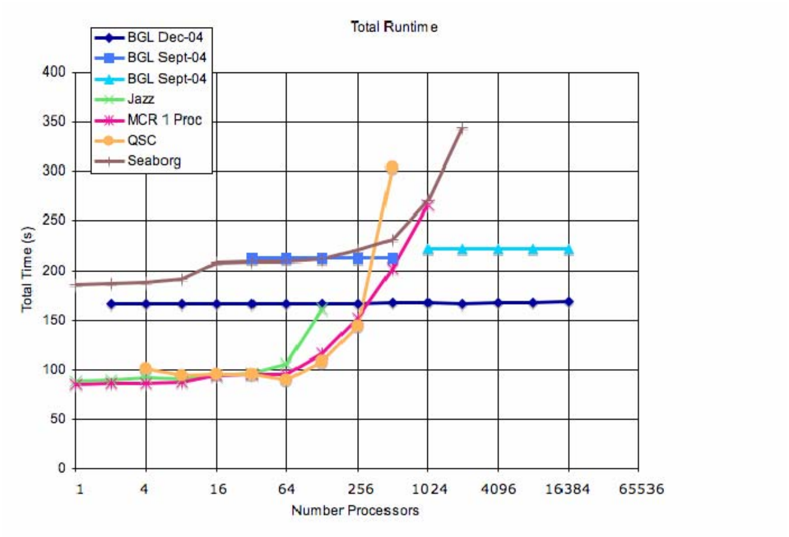


Fig. 4. Weak scaling of FLASH on different architectures

3.4 HOMME

NCAR researchers have built a scalable and efficient spectral-element-based atmospheric dynamical core using the Computational Science Section's High Order Method Modeling Environment (HOMME) [19]. Atmospheric moist processes involving phase changes of water are a fundamental component of atmospheric dynamics and are the most uncertain aspect of climate change research. Simulation of moist processes is challenging because the presence of moisture leads to a new class of fluid motions (moist convection), which is a small-scale phenomenon requiring both very high horizontal and vertical resolution (on the order of a kilometer). The moist Held-Suarez test case extends the standard (dry) Held-Suarez test of the hydrostatic primitive equations by introducing a moisture tracer and simplified physics. It is the next logical test for a dynamical core beyond dry dynamics.

HOMME is written using flexible and efficient F90 modules. The parallel implementation is hybrid MPI/OpenMP. Contiguous groups of elements are distributed to processors and computation is loosely synchronous. The spectral element kernels are similar to level 3 basic linear algebra subroutines such as matrix-matrix multiply. These operations have an O(n) flops to memory access ratio and perform well on modern cache-based microprocessor architectures where CPU-main memory bandwidth increases have significantly lagged processor speed. The physics modules rely heavily on (vector) intrinsic functions. Communication routines are built on top of the MPI message-passing library.

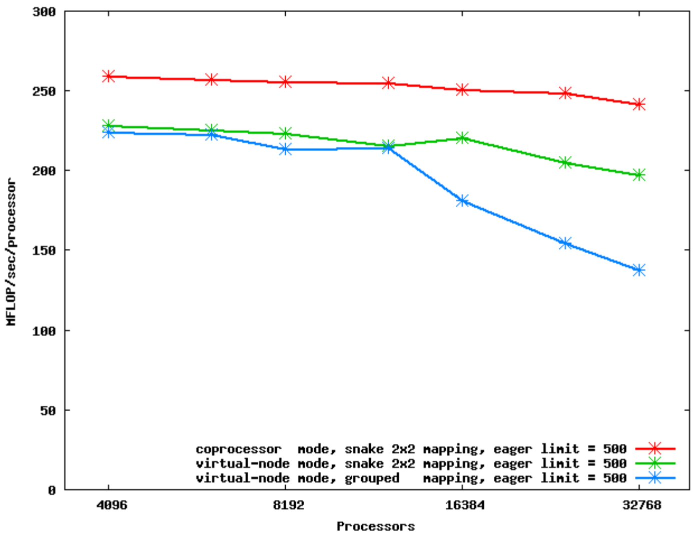


Fig. 5. Strong scaling of HOMME on BG/L: moist Held-Suarez test case, 6 x 128^2 elements, 96 vertical levels, and explicit integration with Δt = 4 seconds

To assess performance on BG/L, a moist Held-Suarez test case configured to match the resolution of the 2002 Gordon Bell AFES run which achieved 26.58 Teraflops on the Earth Simulator was used. This required a grid with 98,304 horizontal elements and 96 vertical levels. Performance results are plotted for a 32,768 node BG/L system at Rochester using both CPM and VNM mode (we were able to run HOMME on the recently installed 32K node system). The total amount of work (total flop count to solve the system) was held constant while increasing the processor count (strong scaling), until, in the 32,768 processor runs there were only three elements per processor. The data is displayed as sustained MFLOPs per processor and ideally should have a flat profile. To show the importance of properly mapping into the torus we show data for both the grouped and snake mapping strategies. In the grouped mapping elements are assigned to the torus in lexicographical order with both processors on a node filled in sequence. The snake mapping places tasks in 2 x 2 node rectangles through the torus, again filling both processors on a node in sequence. At large processor counts the snake mapping is clearly superior as the grouped mapping performance degrades sharply beyond approximately 12,000 processors.

3.5 QCD

The QCD code used in this study does a first-principles numerical simulation of Quantum Chromo Dynamics, the theory of the strong nuclear interactions, using Lattice Gauge Theory. In most full QCD calculations, more than 90% of the cycles are spent in a small kernel that is called the Wilson D-slash operator. It is therefore necessary to optimize this kernel. The kernel was optimized based on specific BG/L hardware features in several ways: FPU operations were grouped to use the DFPU instructions and FPU computations were arranged to avoid pipeline conflicts and to overlap with the load/store pipeline where possible. Memory storage ordering was chosen so that minimal pointer arithmetic was needed. Floating point load/store operations were carefully arranged to take advantage of the cache hierarchy and the CPU's ability to issue up to three outstanding loads before stalling. A thin and effective nearest-neighbor communication layer that interacted directly with the torus network hardware was used to do the data transfers.

Figure 6 shows weak scaling of QCD on up to 4096 processors: in June 2004, BG/L became the first system ever to sustain >1 Teraflop/s performance on QCD. The computation was done using 1024 processors. Furthermore, the code scaled well to 4096 processors. The computational capacity of BG/L can help scientists performing QCD computations avoid approximations that make the results unreliable, and produce more reliable results to this problem.

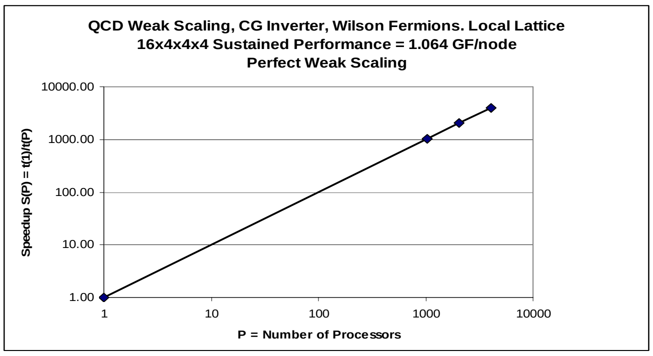


Fig. 6. Weak scaling of QCD on BG/L

4 Conclusions

In this paper, we have presented our experience with several applications on a 16,384 node BG/L system. Our study shows that these applications, which have been targeted to smaller systems so far, can effectively scale to the large BG/L system with a modest level of additional effort. This study represents the first time these applications have been scaled effectively to thousands of processors. These results clearly validate the scalability of the hardware and the software architecture of Blue Gene/L. We hope to improve these results further via software enhancements, and pursue scaling of these applications to levels of a hundred thousand processors in the coming year.

References

[1] S. Habata, M. Yokokawa, S. Kitawaki. The Earth Simulator. In NEC Research and Development, 44(1), January 2003.

[2] ASC Purple: Fifth Generation ASC Platform. http://www.llnl.gov/asci/platforms/purple/.

[3] NASA Unveils its Newest, Most Powerful Supercomputer. http://www.nasa.gov/centers/ames/research/lifeonearth/lifeonearth-projectColumbia.html

[4] N. R. Adiga et al. An overview of the BlueGene/L supercomputer. In *SC2002 – High Performance Networking and Computing,* Baltimore, MD, November 2002.

[5] G. Almasi, R. Bellofatto, J. Brunheroto, C. Cascaval, J. Castaños, L. Ceze, P. Crumley, C. Erway, J. Gagliano, D. Lieber, X. Martorell, J. Moreira, A. Sanomiya, and K. Strauss. An Overview of the BlueGene/L System Software Organization (Distinguished Paper). *Proceedings of the 2003 International Conference on Parallel and Distributed Computing (Euro-Par 2003).* August 26-29, 2003. Klagenfurt, Austria. pp. 543-555.

[6] G. Almasi, S. Chatterjee, A. Gara, J. Gunnels, M. Gupta, A. Henning, J. Moreira, B. Walkup, A. Curioni, C. Archer, L. Bachega, B. Chan, B. Curtis, S. Brunett, G. Chukkapalli, R. Harkness, W. Pfeiffer. Unlocking the Performance of the BlueGene/L Supercomputer. *SC 2004: High Performance Computing, Networking and Storage Conference*, Pittsburgh, PA, November 2004.

[7] F. Petrini, D. Kerbyson and S. Pakin. The Case of the Missing Supercomputer Performance: Achieving Optimal Performance on the 8,192 Processors of ASCI Q. In *IEEE/ACM SC2003*, Phoenix, AZ, November 2003.

[8] G. Almasi et al. Scaling physics and material science applications on a massively parallel Blue Gene/L system. *International Conference on Supercomputing*, Cambridge, MA, June 2005.

[9] TOP500 Supercomputer Sites, http://www.top500.org.

[10] L. Bachega, S. Chatterjee, K. Dockser, J. Gunnels, M. Gupta, F. Gustavson, C. Lapkowski, G. Liu, M. Mendell, C. Wait, T.J.C. Ward. A High-Performance SIMD Floating Point Unit Design for BlueGene/L: Architecture, Compilation, and Algorithm Design. *Parallel Architecture and Compilation Techniques (PACT 2004)*, Antibes Juan-les-Pins, France, Sept-Oct 2004.

[11] K. Davis, A. Hoisie, G. Johnson, D. Kerbyson, M. Lang, S. Pakin and F. Petrini. A Performance and Scalability Analysis of the BlueGene/L Architecture. In *IEEE/ACM SC2004*, Pittsburgh, PA, November 2004.

[12] B.G. Fitch, R.S. Germain, M. Mendell, J. Pitera, M. Pitman, A. Rayshubskiy, Y. Sham, F. Suits, W. Swope, T.J.C. Ward, Y. Zhestkov, R. Zhou. Blue Matter, an application framework for molecular simulation on Blue Gene, *Journal of Parallel and Distributed Computing*, 2003, pp. 759-773.

[13] R.S. Germain, et al. Early performance data on the Blue Matter molecular simulation framework. *IBM Journal of Research and Development*, 49(2/3):447–456, 2005.

[14] J.C. Phillips, G. Zheng, S. Kumar, and L.V. Kale. NAMD: biomolecular simulation on thousands of processors. *Supercomputing 2002 Proceedings*, 2002.

[15] CPMD home page. http://www.cpmd.org.

[16] J. Hutter and A. Curioni. Dual-level parallelism for ab initio molecular dynamics: Reaching teraflop performance with the CPMD code, Parallel Computing, (31), 2005, pp. 1-17.

[17] J. Hutter and A. Curioni. Car-Parrinello Molecular Dynamics on Massively Parallel Computers, ChemPhysChem, 2005, in press.

[18] FLASH code. www.flash.uchicago.edu/

[19] HOMME code. http://www.homme.ucar.edu/