Massively parallel quantum chromodynamics

P. Vranas
M. A. Blumrich
D. Chen
A. Gara
M. E. Giampapa
P. Heidelberger
V. Salapura
J. C. Sexton
R. Soltz
G. Bhanot

Quantum chromodynamics (QCD), the theory of the strong nuclear force, can be numerically simulated on massively parallel supercomputers using the method of lattice gauge theory. We describe the special programming requirements of lattice QCD (LQCD) as well as the optimal supercomputer hardware architectures for which LQCD suggests a need. We demonstrate these methods on the IBM Blue Gene/L™ (BG/L) massively parallel supercomputer and argue that the BG/L architecture is very well suited for LQCD studies. This suitability arises from the fact that LQCD is a regular lattice discretization of space into lattice sites, while the BG/L supercomputer is a discretization of space into compute nodes. Both LQCD and the BG/L architecture are constrained by the requirement of short-distance exchanges. This simple relation is technologically important and theoretically intriguing. We demonstrate a computational speedup of LQCD using up to 131,072 CPUs on the largest BG/L supercomputer available in 2007. As the number of CPUs is increased, the speedup increases linearly with sustained performance of about 20% of the maximum possible hardware speed. This corresponds to a maximum of 70.5 sustained teraflops. At these speeds, LQCD and the BG/L supercomputer are able to produce theoretical results for the next generation of strong-interaction physics.

1. Introduction

Quantum chromodynamics (QCD) is the theory of subnuclear physics. All nuclear particles are made of elementary particles called *quarks* and *gluons*. The gluons mediate the strong nuclear force that binds the quarks together to form stable nuclear particles. The strong nuclear force is one of the four known physical forces, with the other forces being the electromagnetic force, weak nuclear force, and gravity. The strong nuclear force is also responsible for the interactions of nuclear particles and is, therefore, a fundamental area of study in nuclear physics.

Perhaps the best introduction to the theory of QCD was given by Frank Wilczek, a co-recipient of the 2004 Nobel Prize in Physics for his discovery of the properties of QCD. He described QCD as "... our most perfect physical theory" [1] because of the following: QCD embodies deep and beautiful principles (it is a relativistic quantum field theory), it suggests algorithms to answer key questions in physics (one such algorithm is the subject of this paper), it has a wide scope (from nuclear physics to the genesis of the cosmos), it encompasses a wealth of phenomena in physics (e.g., asymptotic freedom and confinement, which are described below), it has few parameters (and is, therefore, simple to describe), it is true (has been verified experimentally), and it lacks flaws (it is fully described by its definition, i.e., it requires no additional assumptions).

Nuclear matter (protons and neutrons) currently constitutes about 90% of the visible universe; however, it is believed that until about 10 μs after the Big Bang, nuclear matter did not exist. The very early universe was so hot that quarks and gluons were in a plasma state called the *quark–gluon plasma*. After 10 μs, the temperature of the universe dropped below two trillion kelvins, and the quark–gluon plasma underwent a phase transition to stable nuclear matter. Currently, at the Brookhaven National Laboratory, an enormously powerful accelerator causes heavy nuclei (in particular gold) to collide at speeds near the speed of light. The Relativistic Heavy Ion Collider (RHIC) produces collisions so powerful that it recreates, if only for a brief moment, the conditions for the formation of the quark–gluon plasma. Strong evidence suggests that RHIC has

0018-8646/08/$5.00 © 2008 IBM

been successful in recreating this state of matter that has not existed in our universe since 10 μs after its birth.

One of the most staggering properties of the theory involves the behavior of its force. Quarks inside nuclear particles behave almost as if they were free (i.e., they experience very little of the nuclear force). This property is called *asymptotic freedom*, which suggests that the interaction between quarks becomes arbitrarily weak at ever shorter distances. However, if one tries to "pull" a quark out of a nuclear particle, the force rapidly becomes extremely strong. A flux tube of gluons forms and forbids the quark from escaping. This property is called *confinement*. Researchers have never observed a single, isolated quark. It is remarkable that both of these dramatically opposite properties are described by a single theory. Furthermore, the theory of QCD is extremely simple in its mathematical description, and it is described by a one-line mathematical formula.

Many physical quantities can be calculated analytically for the case in which the force is weak by using weak-coupling expansions around the zero-force point. However, the calculation of physical quantities becomes extremely difficult when the force is strong. Few analytical calculations are possible, which would have been a serious problem if it were not for the discovery of lattice gauge theory [2, 3]. This theory allows us to calculate physical quantities, such as the masses of nuclear particles or the characteristics of the thermal phase transition in the case of a strong force by using computer simulations. Lattice gauge theory for QCD (LQCD) is described in Section 2.

Even so, with LQCD, the computing requirement is enormous. As a result, LQCD has always required the largest supercomputers available to allow physicists to make scientific progress. In Section 3, we describe the special programming requirements of LQCD as well as the optimal supercomputer hardware architectures from which it benefits. We demonstrate these methods using the IBM Blue Gene/L* (BG/L) massively parallel supercomputer, and we argue that LQCD and the BG/L architecture are well suited to each other because of their curiously common properties. The main result of this paper involves the speedup of LQCD using up to 131,072 CPUs on the largest BG/L supercomputer in 2007, and the result is presented in Section 4. As the number of CPUs is increased, the speedup increases linearly with sustained performance of about 20% of the maximum possible hardware speed. This corresponds to a maximum of 70.5 sustained teraflops (floating-point operations per second) [4, 5]. In Section 5, we present our conclusions.

For an introduction to quantum field theory and QCD, the reader is referred to two books [6, 7]. For an introduction to lattice gauge theory and lattice QCD, the reader is referred to three books [8–10].

2. Lattice QCD

In this section, we give a brief overview of the lattice gauge theory method [2, 3] that allows QCD to be simulated on a computer.

QCD is defined with respect to the continuous four-dimensional (4D) space-time. The quarks and gluons are described by fields over space-time. *Fields* are complex-valued functions of the space-time coordinates and, loosely speaking, indicate the probability of the existence of a particle at each coordinate. This probability is a complicated function of the fields. Specific local and global symmetries constrain these functions.

Since space-time is continuous, one would need an infinite amount of numbers to exactly describe a field even in a finite region. However, a computer is a finite machine with finite memory and computing capability. How then is it possible to simulate QCD?

The first step is to make space-time discrete by replacing it with a 4D lattice. Typically, the lattice is considered hypercubic. Because of the confinement property of QCD, only a finite region of space that contains the nuclear particles must be simulated. In practice, in order to avoid small-volume effects, the region of space used should be several times larger than the particles it contains. Thus, the lattice used is finite, and periodic and antiperiodic boundary conditions are typically implemented. The sites of the lattice are connected by links for which the distance along a link is referred to as lattice spacing a.

This discrete approach could have destroyed the symmetry properties of the theory. However, it turns out that by defining the quark fields on the lattice sites while defining the gluon fields (also called *gauge fields*) on the lattice links, one of the most important symmetries of the theory is preserved. Local gauge invariance is exact.

The rotational and translational symmetries of continuous space-time are destroyed (e.g., the lattice remains invariant if it is shifted by an amount equal to an integer multiple of the lattice spacing, but it does not remain invariant if it is shifted by an arbitrary amount, as is the case for continuous space-time). However, these symmetries are recovered as the lattice spacing a approaches the zero-lattice-spacing limit. By repeating the calculation on lattices with more lattice points and smaller lattice spacing, one can extrapolate to the zero-lattice-spacing limit.

Given this approach, the quark and gluon fields can be defined on a finite set of points. In fact, 24 real numbers per lattice site are needed for each quark field, while 18 real numbers per lattice link are needed for the gauge field. In a typical QCD simulation on the lattice, the computer generates these sets of numbers, called *field configurations*, with a probability that is calculated using the QCD formula. This calculation is complicated and

computationally intensive, but it is possible. From each field configuration, one can then calculate a wealth of physically interesting quantities such as energy and mass. Average values of these quantities are calculated for the full set of field configurations generated by the computer. The method is similar to what is used to simulate statistical-mechanics systems. The equivalent of the Boltzmann weight is present here as well, and it dictates the probability with which field configurations are generated. In particular, molecular dynamics techniques are employed to generate new field configurations from previous ones.

Although it is possible to calculate many physical quantities using numerical simulations, a class of such quantities is still beyond the reach of simulations. For example, equilibration processes, or finite-density physics, involve a severe sign problem (i.e., a complex phase problem) that prohibits use of these techniques. Research efforts known to address these issues have been active for many years. Thankfully, a large class of problems does not suffer from these difficulties. Nuclear physics calculations, calculations of the critical phenomena of the QCD thermal transition, and calculations that relate to the physics of both the current theory of elementary particle physics (the standard model) and theories beyond the standard model are currently being simulated on the largest supercomputers. Such studies involve significant effort because LQCD requires enormous computational resources.

3. Blue Gene/L supercomputer and LQCD

In this section, we discuss LQCD on massively parallel supercomputers and in particular LQCD on the BG/L supercomputer. It may seem strange that a physical theory at the frontier of science would have much in common with a machine designed by engineers working at the frontiers of technology and with strict timetables and architectural guidelines. Theoretical physics has a tradition of "pure thinking" and of analytical calculations in which computers are often barely needed. Conversely, the computing industry is defined by a very rapid development schedule of ever-faster machines that must follow Moore's Law and in which little attention is paid to abstract theories of distant and unrelated disciplines.

However, this interplay between physical theories and computer science is not strange. As described in the previous section, the strong-force regime of QCD would be inaccessible to theoretical calculations if it were not for the largest supercomputers available. In fact, if lattice theoretical physicists had all their wishes fulfilled, today's supercomputers would be viewed as desperately slow. The thirst for computing speed is almost unquenchable. The fact that LQCD absorbs these vast amounts of computing is very interesting. As we discuss in the following section, the weak scaling of QCD on the BG/L system is fortuitous for researchers. (*Weak scaling experiments* refer to studies in which researchers vary the problem size and the number of processors such that the problem size per processor remains constant.) The need for ever-finer lattices that occupy ever-larger volumes indicates that very large lattices are of interest. This implies that QCD can use virtually any size of massively parallel supercomputer that any current and near-future technologies can produce. Petaflop-scale machines are eagerly awaited.

From the supercomputing engineering perspective, QCD has proven to be of great value for many reasons. To understand this, we briefly describe the QCD code and, in particular, its implementation on the BG/L supercomputer.

It turns out that in most QCD implementations, about 90% of the compute cycles are expended inside a small routine (~1,000 lines of code) called the *QCD kernel* or *D-slash* ($\not{D}$). This kernel calculates the dynamics of the quarks and their interaction with the gluons. Obviously, excellent optimization of D-slash is of great importance. The basic operation that involves D-slash may be expressed as

$$\Psi(x) = \sum_{y} \not{D}(x,y)\,\Psi(y), \tag{1}$$

where $\Psi(x)$ is the quark field at the space-time coordinate x, and $\not{D}(x, y)$ is the D-slash operator. This is a sparse matrix with indices x and y. Most matrix elements are zero except when the lattice sites x and y are adjacent on the lattice grid. Because the D-slash operator is so sparse, it is not stored in memory and its action is calculated operationally. D-slash is given by the following equation:

$$\not{D}(x,\ y) = \frac{1}{2}\sum_{\mu=1}^{4}\big[U_{\mu}(x)(1+\gamma_{\mu})\delta(x+\mu,y) + U_{\mu}^{+}(x-\mu)(1-\gamma_{\mu})\delta(x-\mu,y)\big]. \tag{2}$$

In the above equation, μ represents three spatial directions and one time direction, and the sum over μ is a sum over the four space-time directions. The gluon field residing on a link that originates at location x and is along the μ direction is a 3×3 complex-valued matrix (18 real numbers) represented by $U_{\mu}(x)$. The gluon field carries an internal index, called *color charge*, that can assume three values. The γ_{μ} matrices are 4×4 complex matrices that act on another internal index, called *spin*, carried by the quark field. The function $\delta(a, b)$ is one if $a = b$ and zero otherwise. This function implements the nearest-neighbor feature of the operator. It should be noted that the terms $(1 \pm \gamma_{\mu})/2$ are projection operators and reduce the 24-component quark field (also referred to as *full-spinor* below) into four 12-component intermediate

fields (also referred to as *half-spinors* below). The following is one standard way to efficiently implement Equation (1) so that it allows for possible overlap of computations and communications:

1. Using the four projection operators $(1+\gamma_\mu)/2$ ($\mu=1,2,3,4$), spin project Ψ into four temporary half-spinors Φ^{f}_μ for all local lattice sites and store them in memory. (The superscripts "f" and "b" stand for forward and backward, respectively.)
2. Begin sending and receiving each Φ^{f}_μ that is on the surface of the local lattice to and from the nearest-neighbor nodes along the negative direction μ. Each half-spinor consists of 12 numbers. Using double precision, this corresponds to 96 bytes that must be communicated for each site on the negative-direction surfaces.
3. Using each projection operator $(1-\gamma_\mu)/2$ ($\mu=1,2,3,4$), spin project Ψ and multiply the result with U^+_μ in order to form four half-spinors Φ^{b}_μ for all local lattice sites, and store them in memory.
4. Begin sending and receiving each Φ^{b}_μ that is on the surface of the local lattice to and from the nearest-neighbor nodes along the positive direction μ. As in step 2, each half-spinor consists of 12 numbers, and in double precision, this again corresponds to 96 bytes that must be communicated for each site on the positive direction surfaces.
5. Wait for the Φ^{f}_μ communication to complete. Typically, this involves polling a network register.
6. Now that all needed half-spinors Φ^{f}_μ are in the memory of the node, multiply each of them by U_μ and convert them to full-spinors. Add all four full-spinors for each lattice site and store the resulting full-spinor to memory.
7. Wait for the Φ^{b}_μ communication to complete. Typically, this involves polling a network register.
8. Now that all Φ^{b}_μ are on the node, convert each of them into a full-spinor, and for each site, add them together. For each site, add the result to the full-spinor of step 6 after loading it from memory. This produces the resulting full-spinor for each site.

Notice that in the above steps, the Φ fields are not sequential in memory. The U fields are sequential for the first and second set of four terms but not between the two sets. Also, the loop over lattice sites is over a 4D lattice. As a result, memory accesses from the linear memory are typically sequential only in the internal indices, as indicated above, and therefore involve only a small number of bytes to be transferred. Memory accesses per lattice site consist of 24 numbers for the full-spinors, 12 numbers for the half-spinors, and $4\times 18=72$ numbers for the U field in all four links originating from the same site. Furthermore, the communications involve very-small-sized messages. The half-spinors that are communicated reside on the surfaces of the 4D lattice and typically cannot be grouped into a large message. As a result, each half-spinor is communicated individually. These are short messages of only 96 bytes each. The communications and memory accesses cannot be rearranged because they are associated with the in-between computations. The computations themselves involve only a few operations. For example, the multiplication of the gluon matrix U with a half-spinor involves 72 multiply–add operations that execute in just 36 cycles in a double floating-point unit (FPU). Therefore, the above code (summarized in the eight steps) involves very "bursty" (short, nonsequential) memory accesses, communications, and calculations and, as a result, is very sensitive to memory, communication network, and FPU latencies. Surprisingly, the above code suggests a wealth of architectural requirements in order to achieve maximum performance.

Since QCD is defined in a nearest-neighbor lattice of space-time points, it is naturally mapped on a lattice of compute nodes connected with nearest-neighbor physical links. However, some implementations of QCD require local communications that are more distant than nearest-neighbor communications. This implies that a strict nearest-neighbor network would be limiting and leads to the requirement of a more general network.

The above code allows for almost maximal overlap in time between computations, communications, and memory accesses. Given the sensitivity to latencies, a machine that could provide an overlap of all three of these activities would offer a substantial performance advantage over traditional approaches. Thus, the following hardware features are desirable for QCD:

- Load and store accesses in "parallel" with computations and communications.
- Sophisticated memory prefetching that allows block-strided accesses.
- Communications that can overlap with computations and memory accesses. This implies a DMA (direct memory access)-driven network.

Finally, it should be mentioned that Equation (1) is the innermost part of a conjugate gradient (CG) inverter. This inverter requires two global sum reductions per iteration. As a result, fast global-sum-reduction capability is important, suggesting that a good part of the reduction should be supported by hardware.

Although the BG/L supercomputer is a general-purpose computer that is not designed for optimal QCD

performance, many of the above features are present in its hardware. Here is a short description of the BG/L hardware. The reader is referred to [11] for a full description.

The BG/L supercomputer is a massively parallel machine with compute nodes that are interconnected via nearest-neighbor links arranged in a three-dimensional (3D) torus topology. Each node is an IBM ASIC (application-specific integrated circuit) containing two IBM PowerPC* 440 (PPC440) CPU cores. Each core has a custom double multiply–add unit capable of performing up to four floating-point operations per cycle. Therefore, each node can execute up to eight floating-point operations per cycle. Each core has a 32-KB L1 data-cache memory, but the two L1 memories are not coherent. Each core is fed by a small, multistream, sequential prefetcher (L2) that in turn accesses a shared, on-chip 4-MB L3 cache memory. The L3 accesses external DRAM (dynamic RAM) via an on-chip DDR2 (double-data-rate) controller. The ASIC contains a sophisticated, packet-based virtual cut-through router, which allows any node to send packets to any other node without intermediate CPU intervention. Packets that arrive at a node are kept if they are destined for that node or are routed to the appropriate output links in order to reach their final destinations in an optimal way. The network router is accessed from either CPU core by writing and reading packets into hardware addresses that correspond to SRAM (static RAM)-based FIFO (first-in, first-out) queues inside the router. A second, independent collective network is also on the ASIC and provides fast reduction operations such as global sums. Two such ASICs (nodes) are assembled on a small circuit board that also contains the external DRAM (typically 1 GB for both nodes). More functionality is present in the ASIC, but it does not directly relate to the purposes of this paper.

The PPC440 core has a separate load and store pipeline and can have up to three outstanding load instructions. This allows memory access and computations to overlap in time. However, the torus communication network does not allow for overlap of computations and communications because the CPU has to prepare the hardware packets and copy them between memory and the torus FIFOs. Because of this, the earlier code description must be modified for the BG/L platform by consolidating step 2 with step 4 and step 7 with step 5.

Given the above, it is clear that Equation (1) must be carefully coded in a way that is "molded" to the BG/L hardware in order to achieve high sustained performance. This is particularly difficult since the sensitivity to latencies is amplified by the high computing capability of the hardware (eight floating-point operations per CPU cycle). In order to be able to take full advantage of the hardware, we wrote our code as inline assembly code. The main features of our code include the following:

1. All floating-point operations use the double multiply–add instructions by pairing all additions with multiplications in sets of two, whenever possible. The complex numbers used by QCD make this pairing natural.
2. All computations are arranged to avoid pipeline conflicts. These conflicts concern register access rules.
3. The storage order of the quark and gluon fields is chosen to maximize the size of sequential accesses.
4. Load and store operations are arranged to take advantage of the cache hierarchy and the three outstanding load instructions capability of the PPC440 CPU.
5. Since load and store operations can proceed in parallel with floating-point computations, we overlapped memory accesses with computations whenever possible in order to reduce memory access latencies.
6. Since each CG iteration requires two global sums over all the nodes in the machine, we used fast reduction over the global collective network.
7. The BG/L supercomputer does not have a network DMA engine, and as mentioned earlier, the CPUs are responsible for loading and unloading data from the network, reading and storing data to memory, and preparing and attaching the hardware packet headers. Since the transfers that must complete between calculations are very short, we are careful not to introduce any unnecessary latencies. In order to reduce the latencies in step 4, we developed a very fast communications layer directly on the torus hardware. This layer takes advantage of the nearest-neighbor nature of the communication and dispenses with control-related communications. In addition, because the communication pattern is persistent, the packet headers are calculated once only at the beginning of the program. Furthermore, all communications involve direct transfers from and to memory without intermediate copying. Also, although QCD requires a 4D lattice and the BG/L supercomputer has a 3D lattice of nodes, there is a natural way to map QCD onto the BG/L supercomputer. The two CPU cores in each node can serve as a "fourth" dimension. The system software has a virtual node mode of operation in which each core is assigned its own memory footprint, and half the torus FIFOs can be assigned to each core. In this sense, each core is a virtual node. Communication between cores is possible via a commonly mapped

Table 1 Sustained performance for various local lattice sizes. The performance values in the table represent percentages of peak performance.

	Number of nodes					
	2^4	4×2^3	4^4	8×4^3	$8^2 \times 4^2$	16×4^3
D-slash without communications	31.5	28.2	25.9	27.1	27.1	27.8
D-slash with communications	12.6	15.4	15.6	19.5	19.7	20.3
Conjugate gradient inverter	13.1	15.3	15.4	18.7	18.8	19.0

area of memory. We carefully overlap the necessary memory copy time with the time it takes for the network packets to be fully received.

As was mentioned earlier, D-slash is responsible for 90% of the consumed cycles. The remaining 10% are spent by the bulk of the QCD code. This code is tens of thousands of lines long and is written in a high-level language. It encodes both the physics of QCD as well as ingenious algorithms. These codes are written by groups of theoretical physicists and have been developed over many years. It is interesting that the full QCD code stack involves two extremes: a short kernel written in assembly code together with a large amount of code written in a high-level language. In our work, we programmed the D-slash kernel but used the C++ code base of the Columbia Physics System (CPS) that originated at Columbia University [12].

To reiterate, if one wants to design hardware that will perform well for QCD, the design will have to be simple and modular in order to be able to serve various concurrent and competing demands. In particular, trade-off decisions that affect latency can be based on the very-low-latency performance requirements of QCD. For example, this may affect the number of stages of various pipelines, such as CPU, memory, or communications hardware pipelines. Low-latency communications layers would be useful for any QCD type of application in which the communication pattern is fixed and small amounts of data (kilobyte size) are communicated at one time. This is in contrast to the general-purpose heavier type of communication layer, such as Message Passing Interface (MPI). Furthermore, given the importance of low-latency memory access, specialized library functions can be developed for commonly used operations such as the ones found in QCD.

The QCD kernel D-slash can serve as a valuable tool during hardware verification. It can be used to expose bugs (i.e., errors) that may otherwise be unreachable. Bug exposure is facilitated because QCD uses the hardware at high efficiencies as well as at high overlap. For example, the FPU can operate at full performance while the network transfers data at high bandwidth and the memory hierarchy rapidly moves data. This high-demand situation arising from competing and concurrent demands applies pressure on the hardware. Furthermore, the QCD kernel is the full kernel of a real application, so it is of practical importance. Applications often provide excellent verification tools. There have been instances in which bugs were not detected by full verification suites but were apparent during execution of some application. Because most applications tend to be large, they are not suitable as hardware simulators. This is not the case for the small QCD kernel, which can execute in only a few thousand cycles.

During full system validation, QCD can serve as a unique tool for fault isolation for the following reasons. One can program all nodes to perform identical operations on identical datasets. This is possible because the communications are nearest neighbor, their pattern is fixed for all nodes, and the application is strictly SIMD (single instruction, multiple data). All nodes will send and receive the same data to and from their neighbors. At certain intervals, one can check that all nodes have the same value for some intermediate number (e.g., the on-node energy of the gluon field). If a value at a node differs, then the fault is isolated in the neighborhood of that node and corresponding links.

Finally, and very importantly, the QCD kernel can serve as a powerful performance evaluation tool. The performance can be evaluated even before the computer development begins. Because the QCD demands are well defined by Equation (1), these studies can be reliable. Equally significant is that the performance of D-slash can be measured at every stage of the computer development, from verification to a fully built system performance evaluation.

Many of these considerations have been part of the development of several supercomputers, including the BG/L systems. Other examples are the QCD on digital signal processors (QCDSPs) and QCD-on-a-chip (QCDOC) supercomputers [13] that were developed specifically for the study of QCD and have influenced the design of the BG/L supercomputer.

4. Performance

In this section we present the performance results of our code running on the BG/L supercomputer. The strong scaling properties of our kernel were reported in 2005 [14]. (*Strong scaling studies* generally have a fixed problem size, vary the number of processors, and measure the speed.) Our method is not the usual one because we simply kept the number of nodes fixed (to two nodes, with four cores) while we decreased the local problem size. This is akin to strong scaling methodologies, which keep the *global* size fixed while increasing the number of nodes and, thereby, decreasing the *local* problem size. The results are given in **Table 1**.

As can be seen, the smallest local lattice (2^4 sites) without communications achieves 31.5% of peak performance. This high performance is largely due to the fact that the data mostly fits into the L1 cache, resulting in fast memory accesses. However, such a small local lattice has a large surface-to-volume ratio, and therefore, a large number of communications per volume are necessary. Because communications cannot be overlapped with computations on the BG/L supercomputer, the communication cost is additive and the performance drops dramatically to 12.6% when communications are included. For the larger 16×4^3 local lattice, the performance without communications is less (27.8%), but the surface-to-volume ratio is smaller, so the cost of adding communications is less severe, dropping performance to 20.3%.

Nevertheless, QCD is typically used as a weak scaling application. The nearest-neighbor nature of the communications as well as the existence of a fast global sum collective network in the BG/L system give linear speedup as the number of compute cores is increased. We were able to increase the number of cores up to the maximum present in the fastest supercomputer (as of the date of this writing), the BG/L 64-rack system at the Lawrence Livermore National Laboratory (LLNL). The result that led to the award in Reference [4] is the culmination of our efforts, as well as of the findings described in this paper. The results appear here for the first time in print, in **Figure 1**, which shows a maximum of 70.5 Tflops sustained on 131,072 CPUs. The local lattice size is $4 \times 4 \times 4 \times 16$, resulting in a maximum global size of $128 \times 128 \times 256 \times 32$ since the grid of compute nodes of the full machine is $32 \times 32 \times 64 \times 2$. The sustained percent of peak speed in this figure is 19.3% for the D-slash kernel and 18.7% for the full CG inverter, which includes the global sum reductions.

5. Conclusions

In this paper, we have given a general description of the physics of QCD and discussed how massively parallel supercomputers are a natural match for this application.

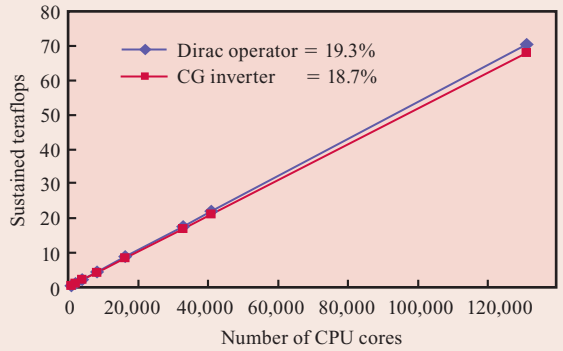

Figure 1

The QCD Dirac operator (D-slash) and conjugate gradient (CG) inverter speedup on the BG/L supercomputer as the number of CPU cores is increased up to the full machine size, 131,072. The highest sustained speed in this graph is 70.5 Tflops [4]. The total lattice has size $128 \times 128 \times 256 \times 32$, while the CPU cores form a grid with size $32 \times 32 \times 64 \times 2$. Therefore, the local lattice on a CPU is of size $4 \times 4 \times 4 \times 16$.

QCD and supercomputing have had a long history. The reader may be interested to know that one of the most popular theoretical physicists and a Nobel laureate Richard Feynman was involved in the development of the Connection Machine 2, a supercomputer that grew out of Danny Hillis's research in the early 1980s at MIT. In fact, Mr. Feynman coded QCD for that machine [15].

Furthermore, we have discussed how QCD can help in the development of massively parallel supercomputers from architecture to final system performance evaluation. Indeed, this has been a component of several supercomputer development efforts, including the IBM Blue Gene* series of machines.

Finally, we have presented the culmination of our efforts in Figure 1, which shows a linear speedup of QCD up to 131,072 CPU cores and 70.5 sustained Tflops. This result was obtained with the 64-rack BG/L system at the LLNL.

Our hope for this paper is that we have shown the close ties between QCD and supercomputing since these ties can serve both fields well in the very interesting and challenging immediate future, when new technologies make it possible to achieve impressive computing speeds and new physics experiments generate new mysteries for LQCD to solve. Readers interested in early research concerning applications of Equation (1) may consult [16].

Acknowledgments

We thank Dr. George Chiu of the IBM Research Division for his help and support. We thank the IBM Research Division and the IBM BG/L team for their support. We

are grateful to the BG/L supercomputing center at the IBM Thomas J. Watson Research Center and to the Lawrence Livermore National Laboratory for allowing us access to these precious resources. We thank the QCDOC collaboration for providing us with the CPS software. Ron Soltz acknowledges the Department of Energy for supporting his research.

*Trademark, service mark, or registered trademark of International Business Machines Corporation in the United States, other countries, or both.

References

1. F. Wilczek, "What QCD Tells Us About Nature—and Why We Should Listen," *Nuclear Phys. A* **663**, 3–20 (2000).
2. K. G. Wilson, "Confinement of Quarks," *Phys. Rev. D* **10**, No. 8, 2445–2459 (1974).
3. K. G. Wilson, "Quarks and Strings on a Lattice," *New Phenomena in Subnuclear Physics, Part A*, A. Zichichi, Ed., Plenum Press, New York, 1974, pp. 69–142.
4. P. Vranas, G. Bhanot, M. Blumrich, D. Chen, A. Gara, M. Giampapa, P. Heidelberger, V. Salapura, J. C. Sexton, and R. Soltz, "2006 Gordon Bell Prize for Special Achievement," *Proceedings of Supercomputing 2006*, Tampa, FL; see *http://sc06.supercomputing.org/news/press_release.php?id=14*.
5. P. Vranas, G. Bhanot, M. Blumrich, D. Chen, A. Gara, P. Heidelberger, V. Salapura, and J. C. Sexton, "The Blue Gene/L Supercomputer and Quantum Chromodynamics," *Proceedings of Supercomputing 2006*, Tampa, FL; see *http://sc06.supercomputing.org/schedule/pdf/gb110.pdf*.
6. T. Cheng and L. Li, *Gauge Theory and Elementary Particle Physics*, Oxford University Press, New York, 1984.
7. M. E. Peskin and D. V. Schroeder, *An Introduction to Quantum Field Theory*, Perseus Books, New York, 1995.
8. M. Creutz, *Quark, Gluons and Lattices*, Cambridge University Press, New York, 1983.
9. I. Monvay and G. Munster, *Quantum Fields on a Lattice*, Cambridge University Press, New York, 1994.
10. J. Kogut, *Milestones in Lattice Gauge Theory*, Kluwer, New York, 2004.
11. "Blue Gene," *IBM J. Res. & Dev.* **49**, No. 2/3 (2005), entire issue.
12. The Columbia Physics System (CPS); see *http://www.epcc.ed.ac.uk/~ukqcd/cps*.
13. P. A. Boyle, D. Chen, N. H. Christ, M. A. Clark, S. D. Cohen, C. Christian, Z. Dong, et al., "Overview of the QCDSP and QCDOC Computers," *IBM J. Res. & Dev.* **49**, No. 2/3, 351–365 (2005).
14. G. Bhanot, D. Chen, A. Gara, J. Sexton, and P. Vranas, "QCD on the Blue Gene/L Supercomputer," *Nucl. Phys. B Proc. Suppl.* **140**, 823–825 (2005); see *http://xxx.lanl.gov/ps/hep-lat/0409042*.
15. W. D. Hillis, "Richard Feynman and the Connection Machine"; see *http://www.kurzweilai.net/articles/art0504.html?printable=1*.
16. H. Hamber and G. Parisi, "Numerical Estimates of Hadronic Masses in Pure SU (3) Gauge Theory," *Phys. Rev. Lett.* **47**, No. 25, 1792–1795 (1981).

Received March 14, 2007; accepted for publication April 9, 2007; Internet publication December 11, 2007

Pavlos Vranas *Lawrence Livermore National Laboratory, Livermore, California 94550 (vranas1@llnl.gov)*. Dr. Vranas received his B.S. degree in physics from the University of Athens, Greece, in 1985 and his Ph.D. degree in theoretical elementary particle physics from the University of California at Davis in 1990. He continued his research in theoretical physics as a postdoctoral researcher at the Supercomputing Computations Research Institute at Columbia University and at the University of Illinois at Urbana-Champaign. From 2000 to 2007, Dr. Vranas worked at the IBM Thomas J. Watson Research Center as a Research Staff Member with the core hardware architecture, design, and development team of the Blue Gene series of supercomputers while continuing his research in theoretical physics. Dr. Vranas performed the first numerical simulations using domain wall fermions and has played a key role in their application to quantum chromodynamics (QCD) and related theories. He received the Gordon Bell Prize in 1998 for his work on the Columbia University QCDSP supercomputer. In 2006, he received the 2006 Gordon Bell Prize for Special Achievement for simulations of QCD on the Blue Gene/L system, and in 2007 he received the IBM Outstanding Invention Achievement Award. Dr. Vranas has authored more than 70 papers in theoretical physics and supercomputing as well as 18 patents.

Matthias A. Blumrich *IBM Research Division, Thomas J. Watson Research Center, P.O. Box 218, Yorktown Heights, New York 10598 (blumrich@us.ibm.com)*. Dr. Blumrich is a Research Staff Member in the Blue Gene Systems Development group at the IBM Thomas J. Watson Research Center. He received a B.E.E. degree from the State University of New York at Stony Brook in 1986, and M.A. and Ph.D. degrees in computer science from Princeton University in 1991 and 1996, respectively. He joined IBM Research in 1998, where he has worked on scalable networking and memory systems for servers and the Blue Gene supercomputers. Dr. Blumrich is an author and coauthor of many patents and technical papers.

Dong Chen *IBM Research Division, Thomas J. Watson Research Center, P.O. Box 218, Yorktown Heights, New York 10598 (chendong@us.ibm.com)*. Dr. Chen is a Research Staff Member in the Deep Computing Systems Department of IBM. He received his B.S. degree in physics from Peking University in 1990, and M.A., M.Phil., and Ph.D. degrees in theoretical physics from Columbia University. He continued as a postdoctoral researcher at Massachusetts Institute of Technology before joining the IBM Server Group in 1999. He has been working in many areas related to the Blue Gene systems since 2000. Dr. Chen is an author or coauthor of more than 30 technical journal papers. He has received an IBM Outstanding Technical Achievement Award and five IBM Invention Achievement Awards. He also received two Gordon Bell Prizes for his contributions to QCDSP and Blue Gene/L supercomputers.

Alan Gara *IBM Research Division, Thomas J. Watson Research Center, P.O. Box 218, Yorktown Heights, New York 10598 (alangara@us.ibm.com)*. Dr. Gara is a Research Staff Member at the IBM Thomas J. Watson Research Center. He received his Ph.D. degree in physics from the University of Wisconsin at Madison in 1986. In 1998 Dr. Gara received the Gordon Bell Prize in the most cost-effective category for the QCDSP supercomputer, and in 2006 he received the Gordon Bell Prize for Special Achievement. He is the Chief Architect of the Blue Gene line of supercomputers. In 2006, he was named a member of the IBM Academy and an IBM Fellow.

Mark E. Giampapa *IBM Research Division, Thomas J. Watson Research Center, P.O. Box 218, Yorktown Heights, New York 10598 (giampapa@us.ibm.com)*. Mr. Giampapa is a Senior Engineer in the Exploratory Server Systems Department. He received a B.A. degree in computer science from Columbia University. He joined the IBM Research Division in 1984 to work in the areas of parallel and distributed processing, and he has focused his research on distributed memory and shared memory parallel architectures and operating systems. Mr. Giampapa has received three IBM Outstanding Technical Achievement Awards for his work in distributed processing, simulation, and parallel operating systems. He holds 15 patents, with several more pending, and has published ten papers.

Philip Heidelberger *IBM Research Division, Thomas J. Watson Research Center, P.O. Box 218, Yorktown Heights, New York 10598 (philiph@us.ibm.com)*. Dr. Heidelberger received a B.A. degree in mathematics from Oberlin College in 1974 and a Ph.D. degree in operations research from Stanford University in 1978. He has been a Research Staff Member at the IBM Thomas J. Watson Research Center since 1978. His research interests include modeling and analysis of computer performance, probabilistic aspects of discrete event simulations, parallel simulation, and parallel computer architectures. He has authored more than 100 papers in these areas. He has been working on the Blue Gene Project since 2000. Dr. Heidelberger has served as Editor-in-Chief of the *ACM Transactions on Modeling and Computer Simulation*. He was the general chairman of the ACM Special Interest Group on Measurement and Evaluation (SIGMETRICS) Performance 2001 Conference, the program co-chairman of the ACM SIGMETRICS Performance 1992 Conference, the program chairman of the 1989 Winter Simulation Conference, and he was the vice president of ACM SIGMETRICS. He is a Fellow of the ACM and the IEEE.

Valentina Salapura *IBM Research Division, Thomas J. Watson Research Center, P.O. Box 218, Yorktown Heights, New York 10598 (salapura@us.ibm.com)*. Dr. Salapura has been a technical leader for the Blue Gene program since its inception. She has contributed to the architecture and implementation of several generations of Blue Gene systems focusing on multithreaded, multicore architecture design and evaluation, and multiprocessor memory subsystems, interconnect, and synchronization. Most recently, she has been Unit Lead for several units of the Blue Gene/P* system, as well as a leader of the chip and system bring-up effort. She is currently working on power/performance characterization of the Blue Gene/P system and on the architecture of future IBM systems. Before joining IBM, Dr. Salapura was Assistant Professor with Technische Universität Wien. She received her Ph.D. degree from Technische Universität Wien, Vienna, Austria, and M.S. degrees in electrical engineering and computer science from University of Zagreb, Croatia. She is the recipient of the 2006 Gordon Bell Prize for Special Achievement for the Blue Gene/L supercomputer and quantum chromodynamics. She is the author of more than 60 papers on processor architecture and high-performance computing, and she holds many patents in this area. She was general co-chair of the 2006 ACM Computing Frontiers conference and program co-chair for the System Architecture and Applications track of the IEEE International Conference on Computer Design in 2006 and 2007. Dr. Salapura is a senior member of the IEEE.

James C. Sexton *IBM Research Division, Thomas J. Watson Research Center, P.O. Box 218, Yorktown Heights, New York 10598 (sextonjc@us.ibm.com)*. Dr. Sexton is a Research Staff Member at the IBM Thomas J. Watson Research Center. He received his Ph.D. degree in theoretical physics from Columbia University and has held research positions at the Fermi National Accelerator Laboratory (Fermilab), the Institute of Advanced Studies at Princeton University, and Trinity College, Dublin.

Ron Soltz *Physics and Advanced Technologies Directorate, Lawrence Livermore National Laboratory, 7000 East Avenue, Livermore, California 94550 (soltz@llnl.gov)*. Dr. Soltz is a Staff Physicist at Lawrence Livermore National Laboratory (LLNL). He received his Ph.D. degree in physics from the Massachusetts Institute of Technology in Cambridge, Massachusetts, in 1994. He currently leads the LLNL research program for the Relativistic Heavy Ion Collider (RHIC) at Brookhaven National Laboratory. His interest in understanding nuclear matter at very highest temperatures and densities led him to begin calculations in lattice quantum chromodynamics, and this work led to the Gordon Bell Prize for Special Achievement in 2006.

Gyan Bhanot *Department of Biomedical Engineering, Rutgers University, Piscataway, New Jersey 08855 (gyanbhanot@gmail.com)*. Professor Bhanot received his Ph.D. degree in theoretical physics from Cornell University in 1979. He was a Research Staff Member at the IBM Research Division in the Physics Department from 1994 to 2001 and in the Computational Biology Group from 2001 to 2006. He worked on BG/L applications at IBM from 2003 to 2005. He is currently Professor of Biomedical Engineering at Rutgers University with joint appointments at the Cancer Institute of New Jersey and the BioMaPS Institute. His current research interest is in developing models for the initiation, progression, and metastasis of cancer and understanding complex disease phenotypes.