RC23688 (W0508-035) August 5, 2005
Computer Science

IBM Research Report

Blue Matter: Strong Scaling of Molecular Dynamics on Blue Gene/L

Blake G. Fitch[1], Aleksandr Rayshubskiy[1], Maria Eleftheriou[1], T. J. Christopher Ward[2], Mark Giampapa[1], Yuri Zhestkov[1], Michael C. Pitman[1], Frank Suits[1], Alan Grossfield[1], Jed Pitera[3], William Swope[3], Ruhong Zhou[1], Robert S. Germain[1], Scott Feller[4]

[1]IBM Research Division
Thomas J. Watson Research Center
P.O. Box 218
Yorktown Heights, NY 10598

[2]IBM Hursley Park
Hursley SO212JN
United Kingdom

[3]IBM Research Division
Almaden Research Center
650 Harry Road
San Jose, CA 95120-6099

[4]Department of Chemistry
Wabash College
Crawfordsville, Indiana 47933

Research Division
Almaden - Austin - Beijing - Haifa - India - T. J. Watson - Tokyo - Zurich

Blue Matter: Strong Scaling of Molecular Dynamics on Blue Gene/L

BLAKE G. FITCH[1], ALEKSANDR RAYSHUBSKIY[1], MARIA ELEFTHERIOU[1], T.J. CHRISTOPHER WARD[2], MARK GIAMPAPA[1], YURI ZHESTKOV[1], MICHAEL C. PITMAN[1], FRANK SUITS[1], ALAN GROSSFIELD[1], JED PITERA[3], WILLIAM SWOPE[3], RUHONG ZHOU[1], and ROBERT S. GERMAIN[1]
IBM
and
SCOTT FELLER[4]
Wabash College

Strong scaling of fixed-size classical molecular dynamics to large numbers of nodes is necessary to extend the simulation time to the scale required to make contact with experimental data and derive biologically relevant insights. This paper describes a novel n-body spatial decomposition and a collective communications technique implemented on both MPI and low level hardware interfaces. Using Blue Matter on Blue Gene/L, we have measured scalability through 16,384 nodes with measured time per time-step of just over 3 milliseconds for a 43,222 atom protein/lipid system. This is equivalent to a simulation rate of 50 nanoseconds per day and represents an unprecedented time-to-solution for biomolecular simulation as well as scaling to fewer than three atoms per node. On a larger 92,224 atom system, we have achieved floating point performance of over 1.5 TeraFlops/second on 16,384 nodes. Scientific results using Blue Matter on prototype BG/L hardware have been published and additional scientific studies are underway which will grow in scale as hardware resources become available.

Categories and Subject Descriptors: J.3 [**Computer Applications**]: Life and Medical Sciences—*Biology and Genetics*; D.1.3 [**Programming Techniques**]: Concurrent Programming—*Parallel Programming*

General Terms: Algorithms, Performance

Additional Key Words and Phrases: Parallel Programming, N-body Simulations, Biomolecular Simulation, Molecular Dynamics

1. INTRODUCTION

Blue Matter [Fitch et al. 2003; Germain et al. 2005] is a molecular simulation framework and application developed to support the scientific goals of IBM's Blue Gene project [Allen et al. 2001], to serve as a platform for research into application programming patterns for massively parallel architectures, and to explore ways to exploit hardware features of the Blue Gene/L architecture. A major design goal for Blue Matter has been to achieve strong scaling of molecular dynamics for moderately sized systems (10,000 – 100,000 particles) to node counts corresponding to ratios of atoms per node of order one. This supports one of the aims of the scientific component of the project, to carry out simulations on a

Authors' addresses: (1) IBM Thomas J. Watson Research Center, 1101 Kitchawan Rd, Route 134, Yorktown Heights, NY 10598. (2) IBM Hursley Park, Hursley SO212JN, United Kingdom. (3) IBM Almaden Research Center, 650 Harry Road, San Jose, CA, 95120-6099. (4) Department of Chemistry, Wabash College, Crawfordsville, Indiana 47933.

scale that allows meaningful comparisons with experimental data. Results on a 43,222 atom protein/lipid system obtained from early production use of prototype Blue Gene/L hardware were recently published in the Journal of the American Chemical Society, the flagship journal in the field [Pitman et al. 2005].

From the start of the Blue Gene project, the operating assumption has been that future performance gains will come from parallelism rather than increases in processor clock speed. The philosophy of the Blue Gene/L hardware design is to use thousands to tens of thousands of power efficient CPUs to achieve high performance in a relatively small footprint. This presents some new challenges for applications: First, to realize the performance potential of Blue Gene/L, one must efficiently distribute the work across many thousands of nodes, which requires very fine load balancing. Second, the software must take advantage of the communication hardware, as communication is typically the rate-limiting factor at high node counts. Both of these challenges are exacerbated by the nature of biological molecular dynamics simulations: load balancing a large number of particle-particle interactions remains a significant algorithmic challenge and the long-range nature of the electrostatic potential leads to global data dependencies with concomitant communication costs.

The use of Blue Gene hardware to advance our understanding of biologically important processes has been an integral part of the Blue Gene mission from the very beginning [Allen et al. 2001]. As part of that strategy, we started an application effort to support the scientific goals of the project and to also act as a concrete test-bed for research into application development for massively parallel machines. At the start the target machine architecture was quite novel in concept [Allen et al. 2001] and the Blue Matter development effort has tracked the evolution of the machine architecture with the goal of exploiting hardware facilities on the target machine. As part of our efforts to exploit the current Blue Gene/L machine architecture [Gara et al. 2005] we have explored the following:

—Use of the global collective network

—Machine topology effects (3d-torus) [Adiga et al. 2005]

—Low level interfaces vs. MPI

—Use of both processors on chip

At the highest level, the Blue Matter architecture specifies a modular decomposition that spans multiple machines (Blue Gene/L and surrounding host machines) and has been implemented as independent subprograms that cooperate via architected interfaces [Fitch et al. 2003]. The Blue Matter parallel molecular dynamics engine makes extensive use of C++ templates and concepts from generic programming [Austern 1999]. By defining appropriate interfaces, we have been working towards a separation of the complexity of molecular dynamics simulation from the complexity of parallel programming with minimal impact on performance. This has enabled the systematic exploration of parallel decompositions for molecular dynamics targeting massively parallel architectures that we have undertaken and whose latest phase is described in detail below.

The Blue Matter architecture requires infrastructure to support extensive regression and validation because of the aggressive and experimental nature of the computational platform we are targeting and because of its support for multiple force fields (the models and their parameters used for classical molecular simulation). Many validation techniques are needed to ensure the correctness of the implementation. The two main requirements are

that the force field parameters be properly implemented, and that the integrator correctly measures the forces on each atom and makes the appropriate update of position and velocity for each time step. Many aspects of MD validation have been discussed previously [van Gunsteren and Mark 1998] including one technique specific to Blue Matter [Suits et al. 2005], and the end result is simulations that match the energies expected for each of the force fields, with energy and temperature tightly conserved over a long simulation time. The JACS publication [Pitman et al. 2005] was based on a 118ns NVE simulation of a membrane-bound protein, and the total and kinetic energy drift over that long period of simulation was negligible, indicating that there was consistently correct bookkeeping and integration of all the interaction forces.

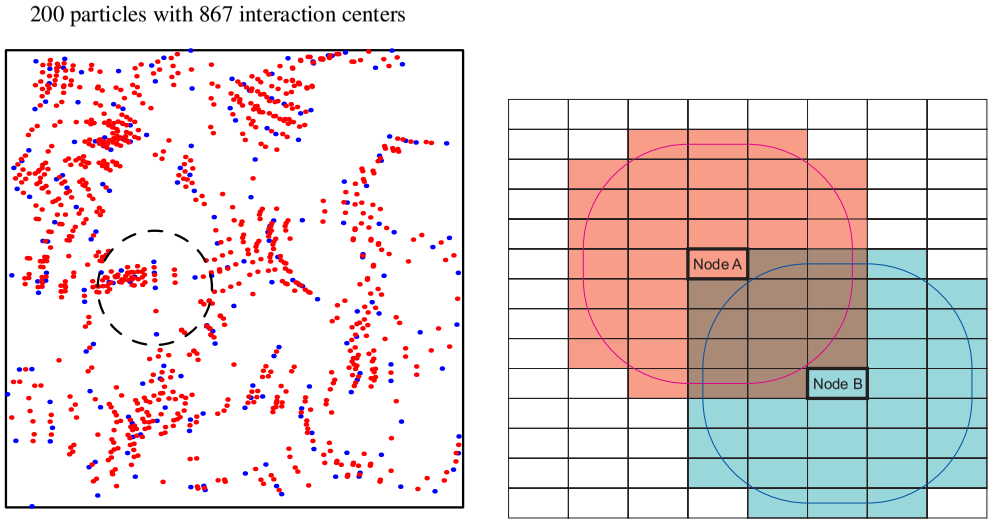
2. PARALLELIZATION STRATEGIES AND CHALLENGES

Classical molecular dynamics uses a model of the interactions between particles as the basis for a numerical integration of the equations of motion of the n-body system. In the case of biomolecular simulation, the existence of molecules with well-separated partial charges means that long range electrostatic interactions must be treated properly or unphysical behavior can be observed [Bader and Chandler 1992]. This issue is most commonly addressed through the use of periodic boundary conditions and the Ewald [De Leeuw et al. 1980] or related mesh [Deserno and Holm 1998] techniques. Use of these techniques involves partitioning the computation of the long range forces into a real-space component that is short-ranged and a reciprocal space component. In the case of the mesh techniques, the reciprocal space component involves a convolution, implemented using three-dimensional FFTs, of the "meshed" charge distribution with a kernel.

One of the design goals for the Blue Matter framework was to allow us to carry out a systematic exploration of parallelization strategies, progressing from the relatively straightforward to the more complex. Our starting point was a version of the "replicated data" [Plimpton and Hendrickson 1996] approach that leveraged the Blue Gene/L hardware collective network (to globalize positions) as well as the torus (to perform a global force reduction) [Germain et al. 2005]. While this approach makes load balancing straightforward, its scalability is limited by the performance of the floating point "all reduce" collective used for the forces.

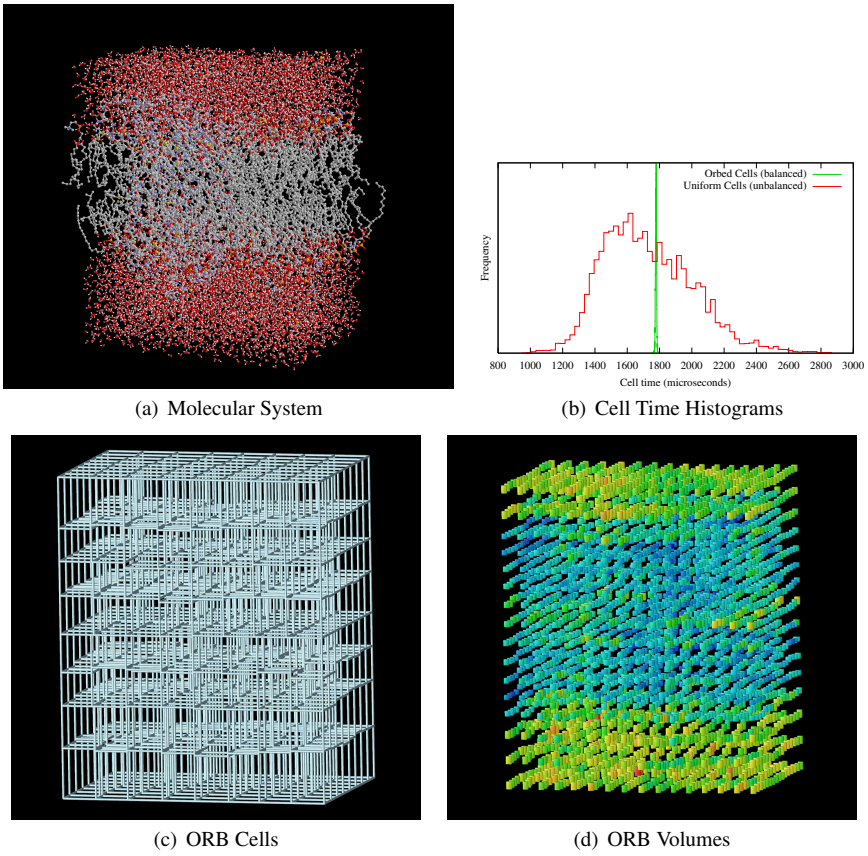
In the current phase of this exploration, we have been using a variant of a spatial decomposition that enables load balancing across a large number of nodes. Our requirements included the ability to load balance based on pair interactions and the maintenance of locality for the real-space portion of the calculation so that a "natural" domain decomposition of the simulation volume onto the 3D torus layout of BG/L would minimize contention on the links.

We implement this load balancing by nominally assigning each non-bonded force interaction to a point in space between the two interacting particles or fragments. A "fragment" is a group of particles that are migrated together from node to node. Fragment location is determined by a tag atom or center of geometry. Fragments always consist of particles within the same molecule although a molecule may be divided up into a number of fragments. For convenience, fragments are currently constructed so that bonds subject to distance constraints (rigid bonds) do not cross fragment boundaries. Fragments typically contain five or fewer atoms. The cost of the interactions between two fragments is assigned to the point in space mid-way between the fragments subject to the minimum image



(a) Interaction Centers (b) Spatial Decomposition

Fig. 1. Figure (a) illustrates the relationship of the particle positions (blue dots) and the interaction centers (red dots) in simulation space. The interaction centers are placed at the mid-point between each pair of particles that fall within the cut-off radius. A dashed circle with radius equal to the cut-off radius chosen is drawn around one of the particles. Static or "structural" load balancing is carried out by using optimal recursive bisection to partition the simulation volume into sub-volumes that contain approximately equal computational burdens. The computational burden of a sub-volume is computed by summing the computational burden of each interaction center contained within that sub-volume. Figure (b) gives a view of the spatial decomposition showing the broadcast zones for two nodes superimposed on the spatial decomposition of the domain onto all nodes (two-dimensional view for simplicity). The nodes that contain areas of simulation space within R_b of the volume element assigned to Node A are shown in orange with a different shading where the nodes also contain areas of simulation space with R_b of the volume element assigned to Node B (which are colored light blue except where overlap occurs). The broadcast radius $R_b > R_c/2$ where R_c is the cutoff radius. The interaction between a particle stored on Node A and a particle stored on Node B can be computed on any of the nodes shaded in brown.



(a) Molecular System (b) Cell Time Histograms

(c) ORB Cells (d) ORB Volumes

Fig. 2. (a) Shows a view of the 43K atom Rhodopsin system. The effects of partitioning on load balance by Orthogonal Recursive Bisection (ORB) on the workload at the interaction centers is shown in Figure (b). Figures (c) and (d) show different views of the modifications to a uniform partitioning of space required in order to balance the interaction workload–(d) shows a symbolic view of the volume of each cell after load balancing using a color scale. As described in the text, the "cost" of computing the interactions between two fragments is assigned to the point midway between the two fragments and space is partitioned using the ORB technique to equalize the workload on each node while maintaining as natural a mapping of the simulation space onto the processor mesh as possible.

convention used with periodic boundary conditions. The simulation volume is then partitioned into sub-volumes with approximately equal computation costs assigned to them via orthogonal (or optimal) recursive bisection (ORB) [Nyland et al. 1997]. The bisections are carried out in such a way that there is a one-to-one correspondence between sub-volumes and nodes and the number of partitions in each dimension is the same as the processor mesh size in that dimension. Of course, as the molecular system evolves, the quality of the load balance degrades, necessitating periodic re-partitionings of the system.

Although orthogonal recursive bisection has been used by others in the context of either partitioning atoms or actual work-load [Nyland et al. 1997; Straatsma and McCammon 2001], our variant of this technique requires that the positions of the particles contained

on a particular node only need to be broadcast to the set of nodes containing points within approximately half the cut-off distance of the originating sub-volume. This ensures that the positions of every particle-pair for which a non-bond interaction must be computed will be available on one or more nodes within the intersection of the two broadcast spheres as shown in Figure 1(b). It also provides a large distribution of load points on which the bisection procedure can be carried out as shown in Figure 1. Additionally, this method makes certain that the positions of particles required for bonded interactions are available where needed and that, with moderate amounts of load imbalance, the positions required for Particle-Particle-Particle Mesh Ewald (P3ME) [Deserno and Holm 1998] calculations will be available as well. Although the number of nodes to which a particular node will broadcast scales as the number of processors, p, the actual count is one eighth of that required by a broadcast to a sphere with radius equal to the full cutoff. This technique also places a limit on the number of inter-node hops required, which scales as $p^{1/3}$.

After the node that is assigned a particular pair interaction receives the necessary position information from the two originating nodes, it can compute the pair interaction, do a local reduction of the forces on a given particle and return the contribution to the force on a given particle to the "home" node of that particle. Note that this requires only a single computation of the interaction. Also, though the pair interactions between two fragments are nominally assigned to the mid-point, they can be performed on any node that receives both particle positions as shown in Figure 1(b). Except for particles that are close to the cut-off distance, it is therefore possible to adjust placement of the interaction computation, providing a mechanism for local load-balancing without a re-partitioning of the entire system.

Other techniques that attempt to combine a spatial decomposition with the advantages of the force decomposition invented by Plimpton and Hendrickson [Plimpton and Hendrickson 1996] have been proposed recently [Snir 2004; Shaw 2004], but neither outlines a detailed strategy for dealing with load imbalance and to our knowledge, no published performance results on biomolecular systems using either technique are available at this time.

Our decomposition requires many-to-many personalized communication operations [Kale et al. 2003] that are not represented by collective operations within the MPI standard. These operations entail each node concurrently originating a broadcast of positions to a local neighborhood and a corresponding concurrent reduction of computed forces back to the originating node. Given a three-dimensional simulation domain and a three-dimensional torus interconnect, communication locality on the machine can be achieved via a "natural" mapping of the simulation domain onto the machine.

Although MPI only allows a task to participate in one collective operation at a time, it is possible to construct equivalent function within the standard in several ways:

(1) Sequentially invoking MPI broadcast/reduce collectives with common members (collectives involving disjoint task groups can proceed concurrently).

(2) Using `ISEND/IRECV` to implement the same communication function in a non-serialized fashion.

(3) Use of `ALLTOALLV` on `MPI_COMM_WORLD` with many node pairs transferring no data to implement the same communication function while avoiding MPI internal message-handling overhead. This can be achieved by using a lower overhead messaging protocol within the implementation of `ALLTOALLV` rather than just using a set

of `ISEND/IRECV` calls.

We have implemented the second and third options and have found that as a result of optimizations of the MPI collectives for BG/L [Almasi et al. 2005], the third option gives superior performance. Even so, the realized performance on MPI does not yet reflect the full capabilities of the hardware. We have implemented the collective operations required by Blue Matter via the low-level System Programming Interface (SPI) of the Blue Gene/L Advanced Diagnostics Environment [Giampapa et al. 2005]. This is the environment used by the hardware team to test and validate hardware performance. Results comparing the performance of selected communications kernels, such as the 3D FFT, on both the MPI and SPI communication layers are presented below (see Figure 3 and Table I). These results encouraged us to proceed with modifications of the full Blue Matter application to allow it to run on optimized SPI-based communications routines with shared-memory exploitation of both processors in virtual node mode.

We attribute the differences in performance observed in the communication micro-benchmarks (Table I and Figure 3) and within the full Blue Matter application (presented in Table III below) to the following factors:

(1) The ability to convey detailed knowledge about application requirements to the communications constructs built on the SPI layer. Because communications patterns within the application persist for varying periods of time, the SPI-based implementation can partially or fully pre-compute headers and preallocate buffers. The most extreme example of the benefits of this approach is the 3D-FFT for which the only change from iteration to iteration is in the values placed within packets. This strategy takes advantage of knowledge that cannot be passed via the MPI API.

(2) At the limits of scalability, when very small amounts of data are being sent by one node to another, any additional header data required by MPI may cause a significant incremental increase in the packet size. For a 128^3 FFT on 16,384 nodes, we are sending a single complex number in each packet and that packet is exactly the smallest size permitted by the hardware.

(3) For the personalized many-to-many communication collectives required, it is possible for the SPI-based implementation to take advantage of the hardware multicast capability of BG/L. Although the MPI `ALLTOALLV` collective can send common data to multiple nodes, there is no mechanism for it to realize that the operation is a broadcast to a subset of nodes and to take advantage of that fact in its implementation.

(4) Close coding of the communications kernel can make use of application commitments to data size, alignment, and receive buffer availability required by the hardware. This can frequently result in elimination of memory copies and may reduce cache misses.

In the results below, we have taken advantage of the second CPU on each BG/L node in several different ways with the results compared for the 43,222 atom system in Table III. The two varieties of dual core use are as follows:

—"Dual(1)": Off-load the computation of the real-space non-bond interactions to the second CPU (pure computation) while carrying out the k-space calculations in the first CPU (including the communication-intensive 3D-FFTs). This allows us to realize some amount of overlap between communication and computation.

—"Dual(2)": Measure the amount of time being spent on real-space non-bond calculations and if the real-space calculations take longer than the k-space calculations, then initially

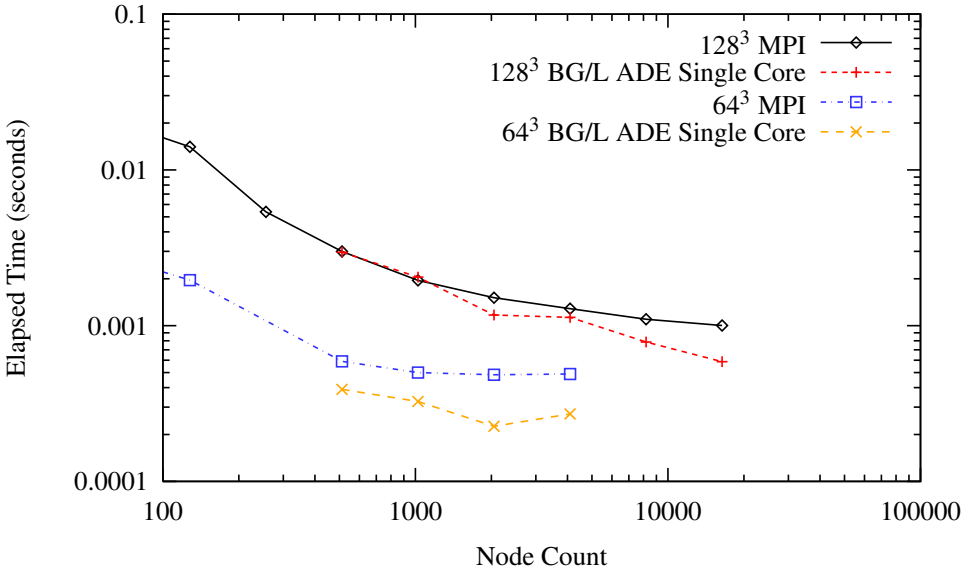


Fig. 3. Measurements of the execution time for the volumetric 3D-FFT [Eleftheriou et al. 2005] running on MPI and on low level communications interfaces derived from the BG/L Advanced Diagnostics Environment [Giampapa et al. 2005] environment.

off-load a portion of the real space non-bond burden to the second CPU. In the case that the k-space computation completes and there is still real-space work remaining, then the real-space calculations are carried out on both cores until completed.

Thus far, "Dual(2)" has been implemented only within the SPI-based version of the code although we plan to implement it within the MPI-based version as well because of the advantages that it has at lower node counts when the real-space interactions consume more time than the k-space interactions.

3. COMMUNICATIONS MICRO-BENCHMARKS COMPARING MPI TO LOW-LEVEL CONSTRUCTS

As discussed above, we have been motivated to explore the use of the SPI used for hardware diagnostics to implement the communications constructs required by Blue Matter in order to fully exploit the hardware capabilities of the Blue Gene/L hardware, especially as we probe the limits of scalability. The communications kernels that we characterized were the 3D-FFT, the spatial neighborhood broadcast, and the corresponding reduction.

In Figure 3 we show measurements of 3D-FFT performance on both MPI and Blue Gene/L Advanced Diagnostics Environment SPI implementations. The preliminary results on the SPI implementation are very encouraging, since the 128^3 FFT continues to speed up out to the limits of scalability where each node carries out a single 1D-FFT during each phase of the row-column computation of the 3D-FFT [Eleftheriou et al. 2005]. Because the bisectional bandwidth of a partition is sensitive to its geometry and because we are scaling to sufficiently high node counts that the data for the 3D-FFT can sit in L1 cache, the scaling of execution time with node count is not smooth.

Node Count	Broadcast		Reduce	
	MPI	SPI	MPI	SPI
512 ($8 \times 8 \times 8$)	1.44 ms	0.67 ms	1.80 ms	0.84 ms
1024 ($8 \times 8 \times 16$)	2.25 ms	1.17 ms	2.69 ms	1.25 ms
2048 ($16 \times 8 \times 16$)	2.30 ms	0.73 ms	2.42 ms	0.81 ms

Table I. Measurements of the performance of the MPI and SPI implementations of the neighborhood broadcast and reduce operations used by Blue Matter. Measurements were made on the 43K atom Rhodopsin system under conditions that guaranteed that both versions would be communicating the same volume of data.

Performance of the MPI and SPI implementations of the neighborhood broadcast and reduce collectives are presented in Table I. Each position or force represents three double precision numbers or 24 bytes of data. The volume of data handled in any one node's portion of these collectives will vary because of number density fluctuations in the molecular system, but the measurements on the MPI and SPI implementations were done under conditions that involve identical sets of communicating partners and equal communication volumes.

4. MEASUREMENT METHODOLOGY

The simulation parameters for all of the Blue Matter results presented here (shown in Table II) are the same as those used for production scientific runs or are the standard ones for those benchmark systems. Because of restrictions on FFT sizes that exist in our current 3D-FFT implementation, our choices of mesh sizes are often larger (smaller mesh spacing) than required for accurate calculations of energies and forces. Except for a specific comparison with NAMD on a benchmark (ApoA1) that used a multiple time stepping technique, all the Blue Matter results presented in this version of the paper were obtained using a velocity Verlet integrator [Swope et al. 1982] that required the P3ME calculation on every time step. This is the technique that when used in our production work on rhodopsin has displayed excellent total energy conservation over long runs (tens to hundreds of nanoseconds) in the constant particle number (N), volume (V), and energy (E) ensemble. With the use of the multiple time step (MTS) RESPA technique with P3ME [Zhou et al. 2001], the P3ME calculation and the FFT's used in that calculation can be performed less frequently.

Timings were obtained via post-analysis of application-level trace data produced by Blue Matter. Using this facility, we can measure the time interval between any pair of trace points on each node and extract statistical information about the set of measured intervals. Blue Matter timings were obtained by averages over 100 time-steps (and all nodes), typically taken several hundred time-steps after ORB-based load-balancing. While diffusion of particles in the system means that the quality of load-balance will decrease over time, our preliminary measurements indicate that redoing the load-balancing by carrying out the ORB process is only necessary after several hundred thousand time-steps and that the performance impact is minimal.

5. PERFORMANCE RESULTS

Scalability results for the MPI version of Blue Matter on a variety of molecular systems are plotted in Figure 4. These systems include solvated proteins of various sizes including the Joint Amber-CHARMM benchmark (DHFR), the Mini FBP system, and the ApoA1 NAMD [Kale et al. 1999] benchmark system as well as the Rhodopsin system that we

System	Total Atoms	Waters (Ions)	Protein/Other Atoms	Cutoff/Switch (Å)	P3ME Mesh
DHFR	23,558	723 (0)	2489	8.0/1.0	64^3
Rhodopsin	43,222	7400 (30)	5608 15,336 (membrane)	9.0/1.0	128^3, 64×128^2
Mini FBP	50,764	16,769 (0)	457	9.0/1.0	128^3
ApoA1	92,224	21,458 (0)	6410 21,440 (lipid)	10.0/2.0	128^3

Table II. Details about the systems benchmarked with Blue Matter. Unless otherwise specified, runs were made with the velocity Verlet integrator [Swope et al. 1982] using the (P3ME) technique to handle long range electrostatic interactions and were constant particle number, volume, and energy (NVE) simulations.

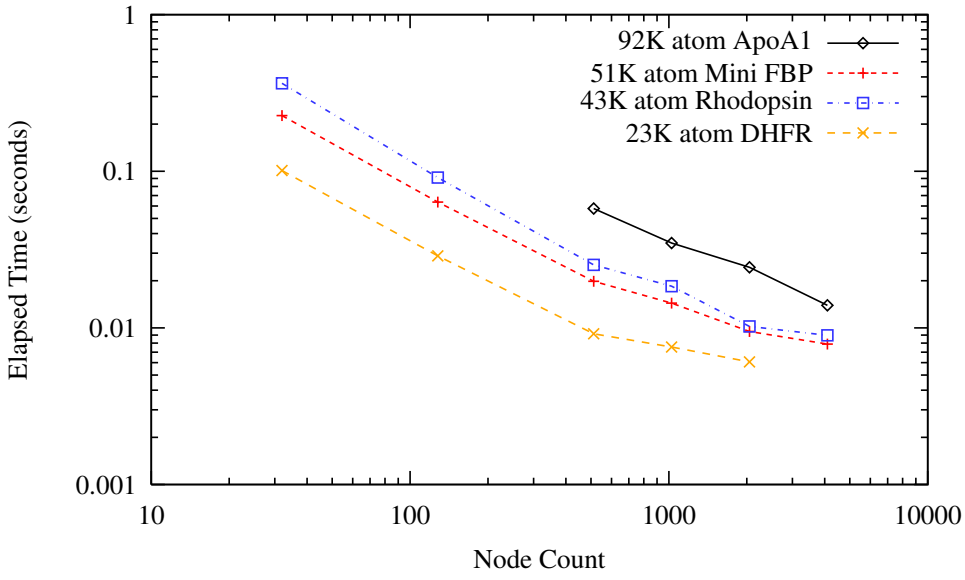


Fig. 4. MPI-based scalability results up to 4096 nodes for Blue Matter on a variety of molecular system sizes and compositions. All of these runs used MPI communications, both CPUs in a mode that used one CPU for real space non-bond calculations and the other for P3ME (distributed 3D-FFTs), and were compiled with options `-O3 -qarch=440`. The node mesh dimensions used were $4 \times 4 \times 2$, $4 \times 4 \times 8$, $8 \times 8 \times 8$, $8 \times 8 \times 16$, $8 \times 16 \times 16$, and $16 \times 16 \times 16$.

have been running in production on Blue Gene/L prototype hardware. All of the plotted results use both cores and when results are available for more than one set of partition dimensions corresponding to the same node count, we only plot the results from the most cubical partition. Results for the Rhodopsin 43K atom system in both single and dual core mode for the MPI version and for two different dual core modes for BG/L ADE SPI version are presented in Table III

The inversion in performance relative to system size for the Rhodopsin and Mini FBP benchmarks seen in Figure 4 can be understood in terms of their compositional differences—Rhodopsin has lipids while Mini FBP is mostly water. We believe that inhomogeneity is not a factor because of the quality of the load balance obtained as seen in Figure 2.

Figure 5 shows the scalability of the major components of a time-step for the 43K atom

Nodes				Time/time-step (seconds)				Atoms/node
				MPI		BG/L ADE SPI		
Total	P_x	P_y	P_z	Single	Dual(1)	Dual(1)	Dual(2)	
32	4	4	2	0.4471	0.3646			1351
128	8	4	4	0.1322	0.0911			338
512	8	8	8	0.0317	0.0253	0.0234	0.0161	84
1024	16	8	8	0.0206	0.0185	0.0162	0.0116	42
2048	16	16	8	0.0137	0.0102	0.0097	0.0072	21
4096	32	16	8	0.0156	0.0135		0.0067	11
4096	16	16	16	0.0104	0.0090	0.0054	0.0039	11
8192	32	16	16				0.0034	5.3
16384	32	32	16				0.0031	2.6

Table III. Tabulated performance data for the 43K atom Rhodopsin system on BG/L. Time per time-step is presented for single core mode and two varieties of dual core mode. In single core mode, only one of the two CPUs on the BG/L chip is actually used by the application. In dual(1) mode, the real space and k-space operations are overlapped by doing k-space operations on CPU 0 and shipping real space operations to CPU 1 via shared memory where coherence is managed by the application. In dual(2) mode, only implemented in the BG/L ADE SPI version at present, a measurement of the real space load is made and if the real space computations take longer than the k-space computations, only a portion of the real-space computations are shipped to CPU 1 initially. When this first portion of the real space computation is complete (and at the same time the k-space operations are complete), the remaining real space computations are carried out on both cores. This provides a considerable win at lower node counts and for larger systems where the real space burden dominates the k-space load. These data were taken using a 2 femtosecond time-step, the value used for production work with this system.

Rhodopsin system. These data were obtained using Blue Matter operating on the BG/L ADE SPI communications layer in Dual(2) mode. The real space computation bar represents the time spent by CPU 1 on its share of the real space computations. In Dual(2) mode, CPU 0 carries out the P3ME operations while CPU 1 computes a portion of the real space calculations required. The fraction of the real space calculations performed on CPU 1 during this phase is chosen so that CPU 0 will complete the P3ME operations at the same time that CPU 1 completes its initial real space computations. If there are any real space calculations that remain to be done after this initial phase, this work is split equally between CPU 0 and CPU 1. Since our measurement tool only supplies information about CPU 0, we infer the amount of time spent on real space calculations by CPU 2 by adding up the amount of time spent on P3ME with the time spent on the "excess" real space calculations by CPU 0. When all of the real space calculations can be completed by CPU 1 in less time than it takes CPU 0 to complete the P3ME operations, then all we can do is place an upper bound on the amount of time spent by CPU 1 on real space. This is the case in Figure 5 for the data taken at 16,384 nodes.

As a very rough way to place Blue Matter running on Blue Gene/L in context, Figure 6 compares published results [Phillips et al. 2002] using the NAMD package [Kale et al. 1999] on the Lemieux system at the Pittsburgh Supercomputing Center and Blue Matter on Blue Gene/L. The results on Lemieux were obtained using a version of the Charm++ library written to the Elan communication library provided by Quadrics. The Blue Matter on Blue Gene/L results were obtained using using the "Dual(2)" version of Blue Matter operating on the BG/L ADE SPI communications layer. The systems benchmarked by Blue Matter and NAMD (ApoA) are identical, and we have made every effort to use either the same (cut-off distances) or higher cost (FFT mesh size) parameters in the Blue Matter runs as were used in the NAMD study to get as close as possible to an "apples-to-apples" com-

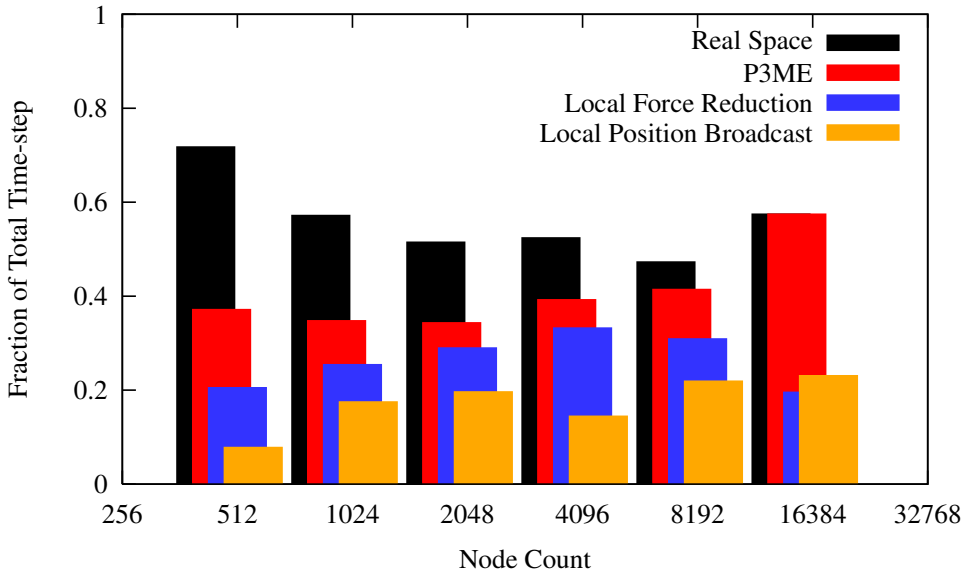
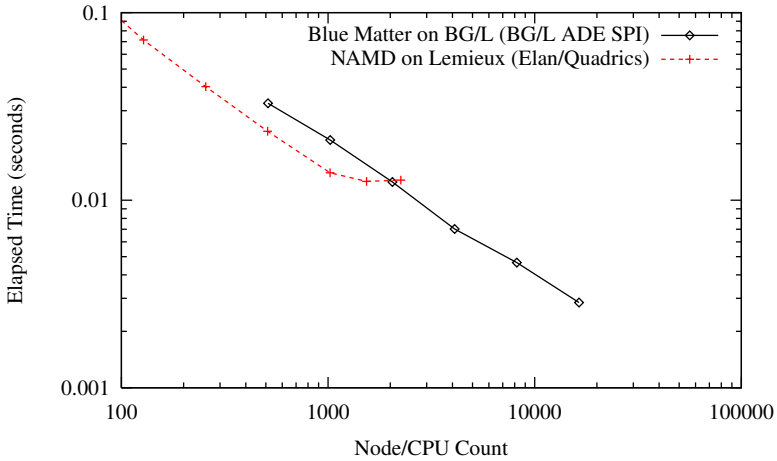


Fig. 5. This figure shows the relative contributions of major components of a time-step for the Rhodopsin system using the BG/L ADE SPI implementation. The data were taken in dual(2) core mode in k-space (P3ME) calculations are carried out on CPU 0 and part of real space (sufficient to balance k-space) is carried out on CPU 1. If any real space work remains after completing the k-space calculations, both cores work on the remaining real space computations. The "Real Space" bar represents the amount of time spent on real space calculations on CPU 1 and it is inferred by measuring the amount of time spent on real space calculations on CPU 0 after the k-space calculations complete and adding that time to the time spent on k-space. Once the real space calculations take less time than the k-space calculations, there is presently no mechanism to estimate them and therefore the "Real Space" bar at 16,384 nodes is equal in height to the "P3ME" bar because we only know that real space is taking some amount of time that is less than or equal to the time consumed by the P3ME (k-space) calculations.

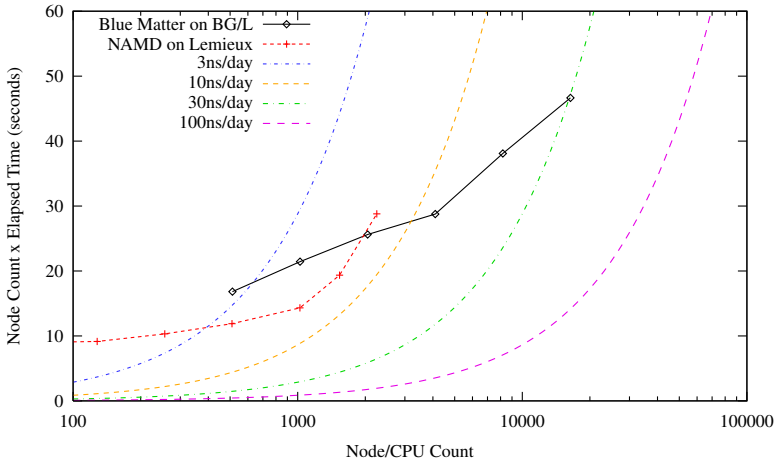
Node Count	Time/Time-step (sec.)	GFLOP/sec.
512	0.0329	140
1024	0.0209	220
2048	0.0125	369
4096	0.0070	657
8192	0.0047	993
16384	0.0029	1621

Table IV. Performance of Blue Matter on the ApoA1 system using multiple time-stepping (P3ME every four time steps) including FLOP rates derived from floating point performance counters in the BG/L chip.

parison. Also, our comparison was made using a multiple time step integration technique that only carried out the P3ME operations once in every four time steps because this was the mode that gave NAMD the best performance on the PSC Lemieux system. Table IV gives the time per time-step and the realized floating point performance for Blue Matter running on the ApoA1 system.



(a) Performance on 92,224 atom ApoA1 system.



(b) Scalability on ApoA1 system with nomograms of throughput. Ideal scalability would be a horizontal line in this plot.

Fig. 6. This plot shows a comparison of the performance of NAMD on the Lemieux Alpha system at the Pittsburgh Supercomputing Center with that of Blue Matter on Blue Gene/L. The molecular system benchmarked was the 92,224 atom ApoA1 system. The NAMD results were obtained using a multiple time-stepping technique to carry out the Particle Mesh Ewald calculation on every fourth time-step (with a mesh size of $108 \times 108 \times 80$) [Phillips et al. 2002] and used a specially tuned version of the Charm++ library written to the Elan communication library provided by Quadrics. In carrying out the measurements with Blue Matter, we attempted to match the parameters reported in the NAMD paper as closely as possible and made our measurements in an NVE (constant particle number, volume, and energy) simulation using a multiple time stepping technique (RESPA [Tuckerman et al. 1992]) with P3ME on every fourth time step (with a mesh size of $128 \times 128 \times 128$). All communication was carried out using the BG/L ADE SPI library [Giampapa et al. 2005].

6. SUMMARY AND CONCLUSIONS

We have described a novel n-body spatial decomposition and a collective communications technique implemented on both MPI and low level hardware interfaces. These constructs have been integrated into the Blue Matter molecular dynamics framework and we have presented strong scaling data on a variety of molecular systems using an MPI-based implementation on Blue Gene/L. Using Blue Matter on BG/L with communications via the BG/L ADE SPI interface, we have achieved close to 3 milliseconds per time-step on 16,384 nodes for a 43,222 atom protein/lipid system. The continued speed-up through values of less than three atoms/node is the first time that this level of strong scaling has been obtained with classical molecular dynamics.

The performance achieved by Blue Matter using the decomposition described in this paper implemented on both MPI and SPI demonstrates the efficacy of our approach. The improvement in performance over the MPI baseline obtained through use of the SPI communications interface shows the advantages that can be realized through use of application-aware communications collectives that fully leverage the available hardware capabilities. This improvement and the results obtained through use of load balancing via the ORB technique as described above also provide support for the hypothesis that planning can give better results than adaptivity for very high levels of scalability.

The time-to-solution measured for the 43,222 atom rhodopsin system on 16,384 nodes corresponds to 50 nanoseconds of simulation time per day or a microsecond of simulation in only twenty days. This capability enables studies of biologically relevant systems on time-scales that were previously impractical. Scientific results using Blue Matter on prototype BG/L hardware have already been published and additional scientific studies are underway.

Work is currently underway to explore further optimizations of the 3D-FFT, such as implementing a real FFT to reduce communication data volume below that of the current complex FFT implementation. We are also continuing to refine our load balancing techniques and are working with the compiler team to improve the floating point efficiency of the Blue Matter code.

ACKNOWLEDGMENTS

We would like thank Jim Sexton for assistance in a multitude of areas and particularly for his work in providing access to the performance counters on the BG/L chip; Nathamuni Ramanujam and Dave Singer for their work in keeping our 20,480 node BG/L system up and running; and the hardware and system software teams in Yorktown and Rochester who created the Blue Gene/L platform that we have been privileged to use.

REFERENCES

ADIGA, N. ET AL. 2005. Blue Gene/L torus interconnection network. *IBM Journal of Research and Development 49,* 2/3, 265–276. http://www.research.ibm.com/journal/rd49-23.html.

ALLEN, F. ET AL. 2001. Blue Gene: a vision for protein science using a petaflop supercomputer. *IBM Systems Journal 40,* 2, 310–327. http://www.research.ibm.com/journal/sj/402/allen.pdf.

ALMASI, G. ET AL. 2005. Design and implementation of message-passing services for the Blue Gene/L supercomputer. *IBM Journal of Research and Development 49,* 2/3, 393–406. http://www.research.ibm.com/journal/rd/492/almasi.pdf.

AUSTERN, M. 1999. *Generic Programming and the STL: using and extending the C++ standard template library*. Addison-Wesley.

BADER, J. AND CHANDLER, D. 1992. Computer simulation study of the mean forces between ferrous and ferric ions in water. *The Journal of Physical Chemistry 96,* 15.

DE LEEUW, S., PERRAM, J., AND SMITH, E. 1980. Simulation of electrostatic systems in periodic boundary conditions I. lattice sums and dielectric constants. *Proc. Roy. Soc. Lond. A 373*, 27–56. and references therein.

DESERNO, M. AND HOLM, C. 1998. How to mesh up ewald sums. i. a theoretical and numerical comparison of various particle mesh routines. *J. Chem. Phys. 109,* 18, 7678–7693.

ELEFTHERIOU, M., FITCH, B., RAYSHUBSKIY, A., WARD, T., AND GERMAIN, R. 2005. Scalable framework for 3d FFTs on the Blue Gene/L supercomputer: Implementation and early performance measurements. *IBM Journal of Research and Development 49,* 2/3, 457–464. http://www.research.ibm.com/journal/rd/492/eleftheriou.pdf.

FITCH, B., GERMAIN, R., MENDELL, M., PITERA, J., PITMAN, M., RAYSHUBSKIY, A., SHAM, Y., SUITS, F., SWOPE, W., WARD, T., ZHESTKOV, Y., AND ZHOU, R. 2003. Blue Matter, an application framework for molecular simulation on Blue Gene. *Journal of Parallel and Distributed Computing 63*, 759–773.

GARA, A. ET AL. 2005. Overview of the Blue Gene/L system architecture. *IBM Journal of Research and Development 49,* 2/3, 195–212. http://www.research.ibm.com/journal/rd/492/gara.pdf.

GERMAIN, R., ZHESTKOV, Y., ELEFTHERIOU, M., RAYSHUBSKIY, A., SUITS, F., WARD, T., AND FITCH, B. 2005. Early performance data on the Blue Matter molecular simulation framework. *IBM Journal of Research and Development 49,* 2/3, 447–456. http://www.research.ibm.com/journal/rd/492/germain.pdf.

GIAMPAPA, M. ET AL. 2005. Blue Gene/L advanced diagnostics environment. *IBM Journal of Research and Development 49,* 2/3, 319–332. http://www.research.ibm.com/journal/rd/492/giampapa.pdf.

KALE, L., KUMAR, S., AND VARADARAJAN, K. 2003. A framework for collective personalized communication. In *Parallel and Distributed Processing Symposium, 2003. Proceedings. International*. IEEE. http://dx.doi.org/10.1109/IPDPS.2003.1213166.

KALE, L., SKEEL, R., BHANDARKAR, M., BRUNNER, R., GURSOY, A., KRAWETZ, N., PHILLIPS, J., SHINOZAKI, A., VARADARAJAN, K., AND SCHULTEN, K. 1999. NAMD2: Greater scalability for parallel molecular dynamics. *Journal of Computational Physics 151*, 283–312.

NYLAND, L., PRINS, J., YUN, R., HERMANS, J., KUM, H.-C., AND WANG, L. 1997. Achieving scalable parallel molecular dynamics using dynamic spatial decomposition techniques. *Journal of Parallel and Distributed Computing 47,* 2 (December), 125–138.

PHILLIPS, J., ZHENG, G., KUMAR, S., AND KALE, L. 2002. NAMD: biomolecular simulation on thousands of processors. In *Supercomputing 2002 Proceedings*. http://www.sc2002.org/paperpdfs/pap.pap277.pdf.

PITMAN, M. C., GROSSFIELD, A., SUITS, F., AND FELLER, S. E. 2005. Role of cholesterol and polyunsaturated chains in lipid-protein interactions: Molecular dynamics simulation of rhodopsin in a realistic membrane environment. *Journal of the American Chemical Society 127,* 13, 4576–4577. http://dx.doi.org/10.1021/ja042715y.

PLIMPTON, S. AND HENDRICKSON, B. 1996. A new parallel method for molecular dynamics simulation of macromolecular systems. *Journal of Computational Chemistry 17,* 3, 326–337.

SHAW, D. 2004. An asymptotic improvement in the parallel evaluation of pairwise particle interactions. Presented at Philadelpha American Chemical Society meeting.

SNIR, M. 2004. A note on n-body computations with cutoffs. *Theory of Computing Systems 37*, 295–318. DOI: 10.1007/s00224-003-1071-0.

STRAATSMA, T. AND MCCAMMON, J. 2001. Load balancing of molecular dynamics simulation with NWChem. *IBM Systems Journal 40,* 2, 328–341. http://www.research.ibm.com/journal/sj/402/straatsma.html.

SUITS, F., PITMAN, M., PITERA, J., SWOPE, W., AND GERMAIN, R. 2005. Overview of molecular dynamics techniques and early scientific results from the Blue Gene project. *IBM Journal of Research and Development 49,* 2/3, 475–488. http://www.research.ibm.com/journal/rd/492/suits.pdf.

SWOPE, W., ANDERSEN, H., BERENS, P., AND WILSON, K. 1982. A computer simulation method for the calculation of equilibrium constants for the formation of physical clusters of molecules: Application to small water clusters. *Journal of Chemical Physics 76*, 637–649.

TUCKERMAN, M., BERNE, B., AND MARTYNA, G. 1992. Reversible multiple time scale molecular dynamics. *J. Chem. Phys. 97,* 3 (August), 1990–2001.

VAN GUNSTEREN, W. AND MARK, A. 1998. Validation of molecular dynamics simulation. *J. Chem. Phys. 108,* 15, 6109–6116.

ZHOU, R., HARDER, E., XU, H., AND BERNE, B. 2001. Efficient multiple time step method for use with Ewald and particle mesh Ewald for large biomolecular systems. *Journal of Chemical Physics 115,* 5 (August), 2348–2358.

...