Blue Gene: A vision for protein science using a petaflop supercomputer

by the IBM Blue Gene team:
F. Allen, G. Almasi, W. Andreoni, D. Beece, B. J. Berne, A. Bright, J. Brunheroto, C. Cascaval, J. Castanos, P. Coteus, P. Crumley, A. Curioni, M. Denneau, W. Donath, M. Eleftheriou, B. Fitch, B. Fleischer, C. J. Georgiou, R. Germain, M. Giampapa, D. Gresh, M. Gupta, R. Haring, H. Ho, P. Hochschild, S. Hummel, T. Jonas, D. Lieber, G. Martyna, K. Maturu, J. Moreira, D. Newns, M. Newton, R. Philhower, T. Picunko, J. Pitera, M. Pitman, R. Rand, A. Royyuru, V. Salapura, A. Sanomiya, R. Shah, Y. Sham, S. Singh, M. Snir, F. Suits, R. Swetz, W. C. Swope, N. Vishnumurthy, T. J. C. Ward, H. Warren, R. Zhou

In December 1999, IBM announced the start of a five-year effort to build a massively parallel computer, to be applied to the study of biomolecular phenomena such as protein folding. The project has two main goals: to advance our understanding of the mechanisms behind protein folding via large-scale simulation, and to explore novel ideas in massively parallel machine architecture and software. This project should enable biomolecular simulations that are orders of magnitude larger than current technology permits. Major areas of investigation include: how to most effectively utilize this novel platform to meet our scientific goals, how to make such massively parallel machines more usable, and how to achieve performance targets, with reasonable cost, through novel machine architectures. This paper provides an overview of the Blue Gene project at IBM Research. It includes some of the plans that have been made, the intended goals, and the anticipated challenges regarding the scientific work, the software application, and the hardware design.

This paper provides an overview of the Blue Gene project at IBM Research. We begin with a brief discussion of why IBM decided to undertake this adventurous research project. We include an overview of proteins and the protein folding problem, including structure prediction and studies of mechanisms. We discuss the limitations of experimental probes of the folding process—a motivation for the use of simulation. This is followed by a brief high-level overview of computational approaches for studying the mechanisms of protein folding, including a survey of some of the challenges, options, and areas of exploration in the field. We then give a brief description of the skeleton science plan now being executed.

After making the case for the utility of large-scale simulation, we focus on the elements of the machine architecture that form the basis for the hardware and software research that the Blue Gene project will pursue. Finally, we describe some of the challenges to be faced in creating a simulation application that can efficiently execute the goals of the scientific pro-

gram on the project hardware, along with some options for meeting those challenges.

Motivation for IBM

There are several reasons why IBM is interested in the use of biomolecular simulations to study protein science. The life sciences are receiving special attention from IBM because the field is demonstrating explosive growth, and the life sciences are creating what will become one of the most significant industries of the new century. Indeed, with advances in bioinformatics and genomics, high-throughput screening of drug candidates, and ready access to information on the Internet, the life sciences have benefited from computational capabilities and will be driving the requirements for data, network, and computational capabilities in the future. The particular area of protein folding was chosen because there is great synergy between IBM's interests and capabilities in high-performance computing and the scientific needs of the field. The understanding of the protein folding phenomenon is a recognized "grand challenge problem" of great interest to the life sciences.

IBM has built research machines to explore novel architectural ideas before, and these projects have frequently been associated with important scientific challenges, such as problems in lattice quantum chromodynamics.[2,3]

The mission of the Blue Gene scientific program is to use large-scale biomolecular simulation to advance our understanding of biologically important processes, in particular our understanding of the mechanisms behind protein folding.

Increased computational power translates into an increased ability to validate the models used in simulations and, with appropriate validation of these models, to probe these biological processes at the microscopic level over long time periods. A critical component of our research program will be the connection of the simulations to the experimental biophysics of protein dynamics.[1] To achieve our high-level scientific goals, it will be essential to collaborate with the worldwide experimental, simulation, and theoretical communities in order to utilize the computational platform in the most intelligent way.

The scientific knowledge derived from research on protein folding can potentially be applied to a variety of related life sciences problems of great scientific and commercial interest, including:

- Protein-drug interactions (docking)
- Enzyme catalysis (through use of hybrid quantum and classical methods)[4]
- Refinement of protein structures created through other methods

Protein science overview

The human genome is currently thought to contain approximately 40000 genes, which code for a much larger number of proteins through alternative splicing and post-translational modification, a molecular toolkit assembled to handle a huge diversity of functions. An understanding of how proteins function is essential for understanding the cell life cycle and metabolism, how cells send signals to their environment, and how cells receive and process signals from their environment. An understanding of protein structure and function can serve as a basis for innovation in new therapies, diagnostic devices, and even industrial applications.

The function of proteins is intimately associated with their structure.[1] The examples shown in Figure 1 illustrate this.[5] When proteins fold into the wrong structure, the results can be fatal, e.g., "mad cow" disease probably results from an autocatalyzed wrong fold in the prion protein[6] and cystic fibrosis is also connected with protein (mis)folding.[7]

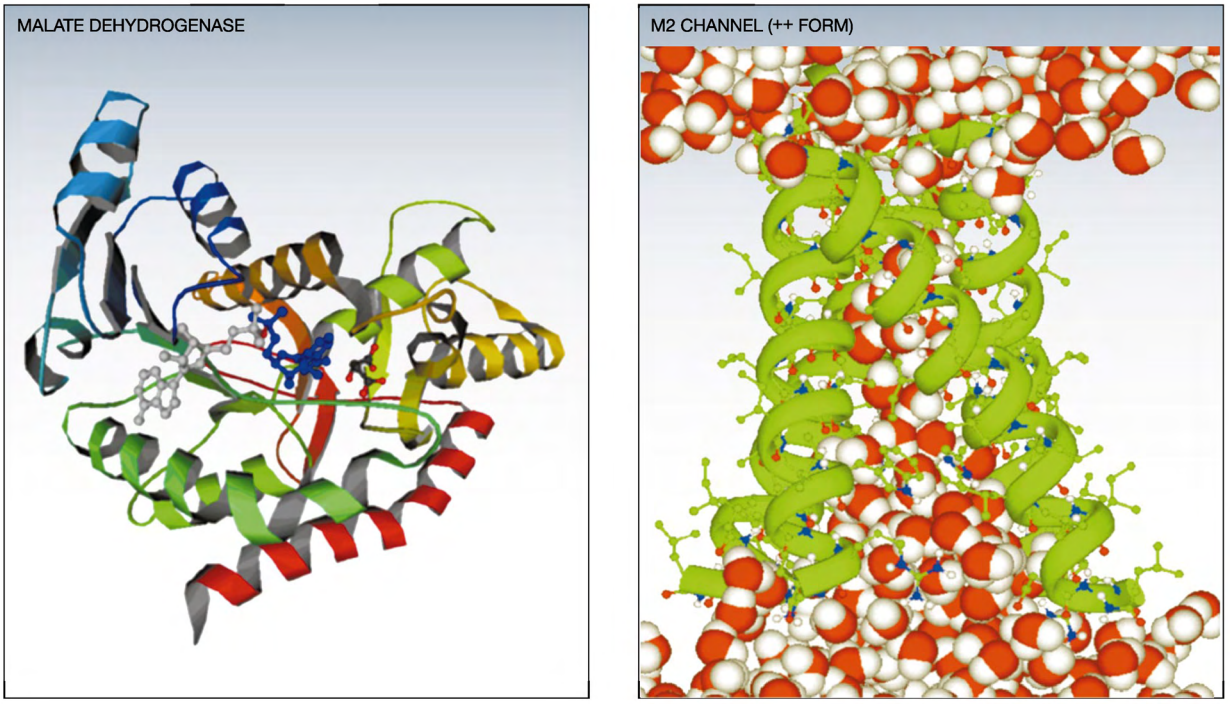
Protein architecture. Protein architecture[8] is based on three principles:

1. The formation of a polymer chain
2. The folding of this chain into a compact function-enabling structure, or *native* structure
3. Post-translational modification of the folded structure

The protein chain (or *peptide chain* if short in length) is a heteropolymer built up from alpha amino acid monomers, as shown in Figure 2. The sequence of amino acid residues in the peptide chain is termed the *primary structure* of the protein. The 20 different choices for each amino acid in the chain give the possibility of enormous diversity, even for small proteins. For example, a peptide of 30 residues yields the astonishing number of about 20^{30}, or approximately 10^{39}, possible unique sequences.

From the enormous number of possible protein sequences that could exist, we observe relatively few in nature. It is thought that the diversity of viable

Figure 1 On the left, the enzyme malate dehydrogenase is shown with the reactants for the catalytic process in place. The site where these reactants are bound, sometimes termed a *cleft*, is specifically tailored in shape and affinity to enormously accelerate this specific reaction, and to be relatively inert to other processes. On the right, the M2 proton channel of the influenza A virus is shown. This ion channel plays an essential role in infection and is the site of attack of a class of channel-blocking flu drugs. In its open state illustrated below, the water column via which protons pass through the channel is seen to be a well-defined structural feature.



Reprinted with permission, from (A) Q. Zhong, T. Husslein, P. B. Moore, D. M. Newns, P. Pattnaik, and M. L. Klein, "The M2 Channel of Influenza A Virus: A Molecular Dynamics Study," *FEBS Letters* **434**, No. 3, 265–271 (1998); (B) Q. Zhong, D. M. Newns, P. Pattnaik, J. D. Lear, and M. L. Klein, "Two Possible Conducting States of the Influenza A Virus M2 Ion Channel," *FEBS Letters* **473**, No. 2, 195–198 (2000).

proteins has been constrained by natural selection to give:

1. Desired function
2. Adequate stability
3. Foldability
4. Evolvability from appropriate evolutionary precursors

The peptide chain has certain local fold characteristics termed *secondary structure*.[7] Steric hindrance and energetic considerations favor certain conformations of the peptide chain. One such conformation is the *alpha helix* (see red helices in Figure 3). Another secondary structure is the *beta sheet* (blue flattened regions in Figure 3), in which two or more strands of the peptide chain are aligned to form a sheet. The relatively organized alpha helix and beta sheet sections of a protein are joined by less organized *loop* or *turn* regions. The way in which the relatively localized secondary structure elements combine to form the overall compact protein is termed the *tertiary* level of structure, as can be seen from the example on the right in Figure 3. Finally, quaternary structure refers to the way that tertiary structures from two or more chains combine to form much larger structures.

The protein folding problem. There are two important facets to the protein folding problem: prediction of three-dimensional structure from amino acid sequence, and understanding the mechanisms and pathways whereby the three-dimensional structure forms within biologically relevant timescales.

The prediction of structure from sequence data is the subject of an enormous amount of research and a series of conferences that assess the state of the art in structure prediction.[9] While this area is extremely important, good progress in the area of structural predictions has been made using only modest amounts of computational power. The effort described in this paper is aimed at improving our understanding of the mechanisms behind protein folding, rather than at structure prediction. Even though biologists have been most interested in structure prediction, there has been an increasing recognition of the role that misfolding of proteins plays in certain disease processes, notably Alzheimer's disease and mad cow disease.[6] The section that follows describes some of the fundamental reasons for interest in the process of protein folding.

Why protein folding mechanisms are important. The fact that a subset of heteropolymers constructed from amino acids and used in biological processes actually take on reproducible three-dimensional structures in a relatively short time of seconds or less is one of the marvels of nature. Heteropolymers typically form a random coil in solution and do not "fold" to any reproducible structure in experimentally accessible times. Consider the paradox noted by Levinthal,[10] which asks the reader to consider that if, say, we allow three possible conformations for every amino acid residue on the peptide chain, a 100-residue protein would have $3^{100} = 10^{47}$ configurations. Any unbiased exploration of this conformational space would take a vast amount of time to complete. Thus, the proteins that fold reproducibly and quickly into particular shapes must have been selected in some way for these properties. It is hypothesized that these proteins conduct this conformational search along particular *pathways* that allow the folding process to proceed quickly. One of the challenges in the study of protein dynamics is to understand the mechanisms behind this behavior. An improved understanding of these mechanisms is not only interesting from a purely scientific perspective, but might eventually allow us to engineer other "self-assembling" structures.

Current view of folding mechanisms. A simplistic but illustrative way of viewing protein folding is to note that the amino acid *R* groups (see Figure 2, caption) fall into three main classes: (1) charged, (2) hydrophilic ("water-loving"), and (3) hydrophobic ("water-hating"). In the simplest picture, the folded state of the peptide chain is stabilized primarily (for a globular protein in water), by the sequestration of much of the hydrophobic groups into the core of the protein—out of contact with water, while the hydrophilic and charged groups remain in contact with water. The stability can be described in terms of the Gibbs free-energy change ΔG

$$\Delta G = \Delta H - T\Delta S,$$

where ΔH is the enthalpy change and ΔS is the entropy change. ΔH is negative due to the more favorable hydrophobic interactions in the folded state, but so is ΔS because the folded state is much more ordered and has lower entropy than the unfolded state. The balance between the enthalpy and entropy terms is a delicate one, and the total free-energy change is only of order 15 kilocalories per mole. Evidently the internal hydrophobic/external hydrophilic packing requirement places strong constraints on the amino acid sequence, as does the requirement that the native state be kinetically accessible.

It is helpful to think of the physics of the folding process as a "free-energy funnel,"[11] shown schematically in Figure 4. Since the folding process is slow relative

Figure 2 The generic formula for an alpha amino acid is $NH_2 - C\alpha HR - COOH$, where the suffix denotes the alpha carbon. The different amino acids, also termed "residues" in protein architecture, differ in the "*R*" group attached to the alpha carbon. There are 20 possible choices of *R* group. In the polymerization process successive monomers are added, resulting in a peptide chain, as shown. The peptide and ultimately the protein chain has the formula $NH_2 - C\alpha HR_1 - CO - NH - C\alpha HR_2 - \cdots - C\alpha HR_L - COOH$.

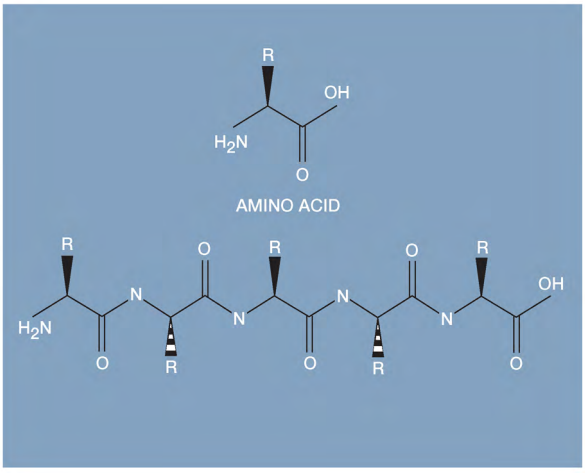
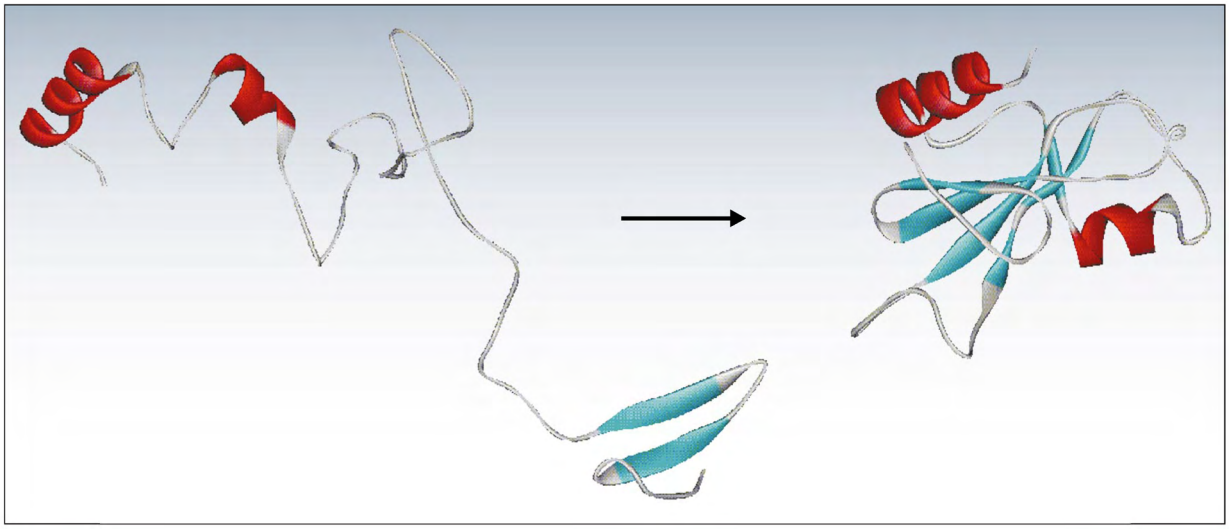


Figure 3 The left portion of the figure shows the unfolded state with some secondary structure formed; the right portion shows the native folded state. PDB ID: 1B2X, A. M. Buckle, K. Henrick, and A. R. Fersht, "Crystal Structural Analysis of Mutations in the Hydrophobic Cores of Barnase," *Journal of Molecular Biology* 234, 847 (1993).



to motions at atomic scale, we can think of partially folded configurations as having a quasi-equilibrium value of the free energy. The free energy surface may be displayed as a function of some reduced dimensionality representation of the system configuration in a given state of the protein.[12] Figure 4 is a vertical plot of free energy in a contracted two-dimensional space (horizontal plane) representing the configuration of the peptide chain. The most unfolded configurations are the most numerous, but have the highest free energy, and occur on the rim of the funnel. Going into the funnel represents a loss of number of configurations (decrease of entropy), but a gradual decrease in free energy, until the native state with very few configurations and the lowest free energy is reached at the bottom of the funnel. The walls of the funnel contain only relatively shallow subsidiary minima, which can trap the folding protein in non-native states, but only for a short time. Now the evolution of the system as it folds can be described in terms of the funnel. The system starts off in a physically probable state on the rim of the funnel, and then makes transitions to a series of physically accessible states within the funnel, until the bottom of the funnel is gradually approached.

Figure 3 illustrates folding. Here the unfolded peptide chain on the left already contains some folded secondary structure, alpha helices (red), and a beta hairpin (blue). It is still a long way from the compact native structure at right. The folding process in different proteins spans an enormous dynamic range from approximately 20 microseconds to approximately 1 second.

Probes of folding. In biophysical studies of protein folding, current emphasis is on the reversible folding of moderate-sized globular proteins under closely controlled conditions *in vitro*. Many globular proteins can be made to unfold *in vitro* and then to undergo a refold back to the native structure without any external assistance.[1] The unfolding process can be driven in several ways, for example by changes in denaturant concentration, by heat pulses, by pressure pulses, etc. In this manner, refolding times down to tens of microseconds can be measured.

Although the study of protein dynamics and folding pathways via experiment is an active area of research and much progress is being made,[1,13] experimental probes do not yet provide as complete a view of protein dynamics at the microscopic length and timescales as one would like. Numerical simulations offer a potential window into this microscopic world, and models that treat the relevant physical phenomena with varying degrees of abstraction have been useful sources of insight, particularly for the theoretical community. As larger computational re-

sources have become available and simulation techniques have improved, more detailed simulations of larger systems and longer timescales have become possible.

The simplest systems for study are those with less than approximately 120 residues, which constitute "two-state folders," i.e., there is no intermediate between the denatured and native states. Such ideal systems should also not contain prosthetic groups, i.e., they should be pure peptides, should not require the assistance of "chaperones" to fold, and should preferably not have disulfide bonds.

Consider the following three types of protein science studies that might employ large-scale numerical simulation techniques:

- Structure prediction
- Folding pathway characterization
- Folding kinetics

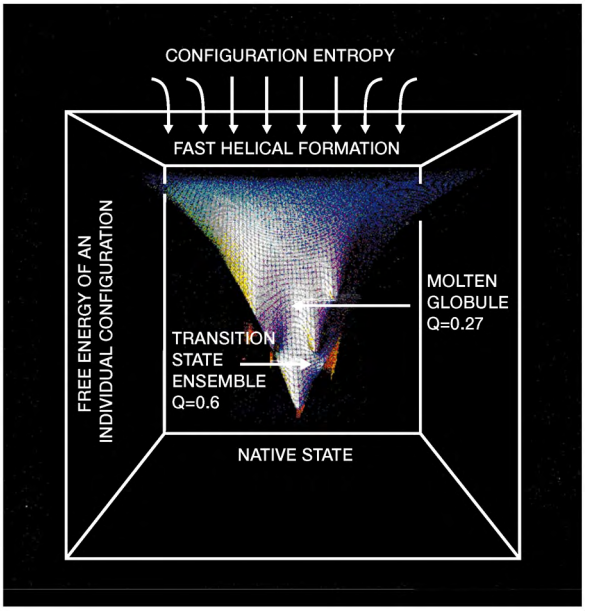
Protein structure prediction can be carried out using a large number of techniques[8] and, as previously discussed, it is unnecessary to spend a "petaflop year" on the prediction of a single protein structure. That said, there is some reason to believe that atomistic simulation techniques may be useful in *refining* structures obtained by other methods.

Folding pathway characterization typically involves the study of thermodynamic properties of a protein in quasi-equilibrium during the folding process. Mapping out the free-energy "landscape" that the protein traverses as it samples conformations during the folding process can give insights into the nature of intermediate states along the folding pathway and into the "ruggedness" of the free-energy surface that is traversed during this process. Because such studies involve computations of average values of selected functions of the system's state, one has the choice of either averaging over time as the system samples a large number of states (molecular dynamics) or averaging over configurations (Monte Carlo). Aggressive sampling techniques that may improve the computational efficiency with which such averages can be computed can be used to good effect in these studies. Simulation techniques to compute these averages over the appropriate thermodynamic ensembles are available.[14]

Simulation studies of *folding kinetics* are aimed at understanding the rates at which the protein makes transitions between various conformations. In this case, the calculation of thermodynamic averages is not enough; the actual dynamics of the system must be simulated with sufficient accuracy to allow estimation of rates. Of course, a large number of transition events must be simulated in order to derive rate estimates with reasonable statistical uncertainties. Another challenge faced in such simulations is that the simulation techniques used to reproduce thermodynamic averages in ensembles other than constant particle number, volume, and energy (NVE) are, strictly speaking, inappropriate for studies of folding kinetics.

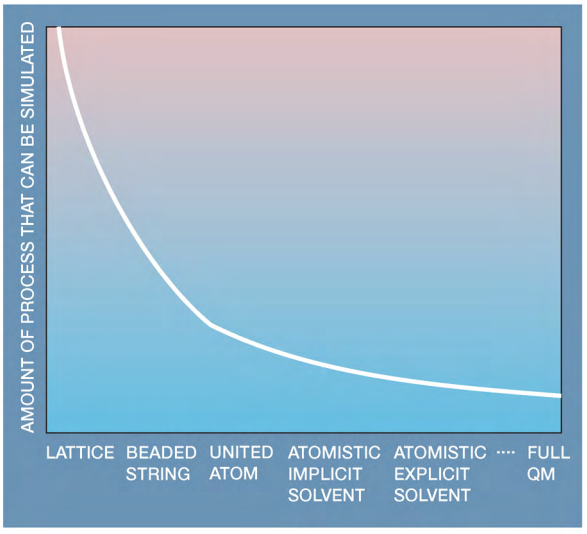
Both pathway and kinetic studies may "zoom in" on selected regions of interest in the folding process rather than trying to simulate the process from end

Figure 4 The free energy is plotted vertically as a function of the accessible configurations of the peptide chain, which are projected into the plane in this representation. The most unfolded configurations are the most numerous, but have the highest free energy, and occur on the rim of the funnel. Going into the funnel represents a loss of a number of configurations (decrease of entropy), but a gradual decrease in free energy, until the native state with very few configurations and the lowest free energy is reached at the bottom of the funnel.



Reprinted from *Folding and Design* **1**, J. N. Onuchic, N. D. Socci, Z. Luthey-Schulten, and P. G. Wolynes, "Protein Folding Funnels: The Nature of the Transition State Ensemble," 441–450. ©1996, with permission from Elsevier Science.

Figure 5 Schematic plot of degree of sophistication of interatomic interactions vs amount of biophysical process that can be simulated



to end. If probing the dynamics of the folding process through repeated simulation of large portions of complete folding trajectories is the method used, then some logical starting points would be peptides or small fast-folding proteins with well-characterized properties.[15] Preferably, folding times should be under 100 microseconds for convenience of simulation.

Computational approaches to studying folding mechanisms

In principle, computer simulations of biomolecular processes such as protein folding can be carried out using techniques spanning a broad range of sophistication in modeling the basic physical processes and spanning a broad range in computational cost.

At one end of the spectrum in sophistication of treatment of the interatomic interactions are the so-called lattice and beaded-string models. Often in these treatments each amino acid residue of a protein is approximated as a point particle. In the lattice models these particles are constrained to occupy sites on a lattice in three dimensions. In the beaded string models this restriction can be relaxed. These models have been extremely useful because they can be exhaustively investigated with a modest amount of computer resources.[16,17]

A more detailed treatment of the interatomic interactions is provided by united-atom models that are simulated in a vacuum or in what is known as implicit solvent. As the name indicates, united-atom models treat groups of atoms within the protein as if they were a single particle. The implicit solvent approximation treats the water surrounding the protein not as discrete molecules, but rather as a continuum, perhaps with a dielectric interaction on the protein. These models, because of their increased sophistication, require significantly more computer resources than the lattice and beaded-string models, and so, many fewer simulations can be run and not as many variables can be easily investigated.

Next in sophistication are the all-atom models of proteins, in which the water solvent is treated with an atomistic resolution. This treatment of the water, known as an explicit solvent treatment, adds considerable computational cost to the calculation, since the interactions between water molecules become the dominant part of the simulation.

Finally, the ultimate in treatment of interatomic interactions would include quantum mechanical (QM) methods that treat the actual electronic wave function of the system explicitly. Such calculations are used, in fact, to study relatively short-time biomolecular processes such as enzyme catalysis.

As Figure 5 indicates, as the degree of sophistication of the treatment of the interatomic interactions increases, it becomes more and more difficult to exhaustively simulate the entire process of protein folding, because the computational requirements rise so quickly. However, all approaches have been fruitful, some workers choosing to take advantage of the ability to simulate the *entire* folding process, albeit at a lower level of sophistication of interatomic interactions. Others have focused their efforts on *higher quality* treatment of the interactions, but with much less exhaustive exploration of the folding process. The addition of hardware with the capabilities planned for Blue Gene should significantly improve the ability to perform simulations of many degrees of sophistication. The current plan for the Blue Gene project is to use all-atom explicit solvent simulations as our first targeted approach.

So far the most ambitious attempt to fold a protein is that of Duan and Kollman[18] on the villin headpiece in water. In three months, on a 256-node Cray T3E processor, the authors were able only to follow the folding trajectory for one microsecond, still too

short a time for folding, and in fact the authors were not able to achieve a fully folded configuration, though extensive secondary structure formation occurred. Subsequently three more trajectories have also been obtained by these authors.

Challenges for computational modeling. The current expectation is that it will be sufficient to use classical techniques, such as molecular dynamics (MD), to model proteins in the Blue Gene project. This is because many aspects of the protein folding process do not involve the making and breaking of covalent bonds. While disulfide bonds play a role in many protein structures, their formation will not be addressed by classical atomistic simulations. In classical atomistic approaches, a model for the interatomic interactions is used. This is known as a potential, or force field, since the forces on all the particles can be computed from it, if one has its mathematical expression and all its parameters. The MD approach is to compute all the forces on all the atoms of the computer model of the protein and solvent, then use that force to compute the new positions of all the atoms a very short time later. By doing this repeatedly, a *trajectory* of the atoms of the system can be traced out, producing atomic coordinates as a function of time.

Newton's equation is integrated for each particle using a small time step of the order of 10^{-15} seconds. This small time-step size is required to accurately describe the fastest vibrations of the protein and solvent system, which tend to be those associated with movement of hydrogen atoms in the protein and water. The magnitude of the computational cost can be seen when one notes that folding times of approximately 10^{-4} seconds are observed in some fast-folding systems, requiring the computation of approximately 10^{11} MD time steps. As can be seen in Table 1, the computational requirements for studying protein folding are enormous.

Various methods can be used to improve the stability and efficiency of the dynamic simulation. These include the use of integrators with good stability properties such as velocity Verlet,[19] and extensions such as RESPA.[20] Additional efficiencies can be realized through freezing the fastest modes of vibration by constraining the bonds to hydrogen atoms to be fixed in length using algorithms such as SHAKE[21] and RATTLE.[22] This approximation is consistent with many of the most popular models for water interactions, such as TIP3P, TIP4P, TIP5P,[23] and SPC and SPC/E,[24] since these models for water interactions

Table 1 The computational effort required to study protein folding is enormous. Using crude workload estimates for a petaflop/second capacity machine leads to an estimate of three years to simulate 100 microseconds.

Physical time for simulation	10^{-4} seconds
Typical time-step size	10^{-15} seconds
Number of MD time steps	10^{11}
Atoms in a typical protein and water simulation	32000
Approximate number of interactions in force calculation	10^{9}
Machine instructions per force calculation	1000
Total number of machine instructions	10^{23}

treat the molecules as completely rigid. The use of RESPA and the fixing of bond lengths involving hydrogen atoms with SHAKE and RATTLE allow the use of larger time-step sizes without much degradation in the accuracy of the simulation.

One would like to study the dynamics of a single protein in an unbounded volume of water. Since only a finite simulation cell can be modeled, the most common approach is to use periodic boundary conditions in the simulation volume. Thus the simulation models a periodic structure of unit cells, such as a simple cubic array, each cell consisting of a box of water containing a protein within it, and all cells being identical. The charges in different cells are partially screened from each other by the high water dielectric constant, which is exhibited by water models commonly used in simulations. Long-range electrostatic interactions for such a periodic structure can be treated via the Ewald summation technique.[25]

One of the key challenges faced in implementing classical atomistic simulations is that of adequately modeling the interatomic interactions in the systems required to study problems of biological interest such as protein folding. The forces that are typically taken into account in MD simulations of protein and water systems (and many others) are illustrated in Figure 6, which displays some of the types of energy expressions involved in a typical model potential. The energy terms may be classified as either *bonded* or *nonbonded* interactions. Bonded interaction terms in most model potentials consist of at least bond stretching, angle bending, and torsion energy terms, but may include more complex terms such as stretch-

Figure 6 Representative functional forms for interparticle interactions used in force fields for atomistic simulations, $U_{total} = U_{Stretch} + U_{Bend} + U_{Torsion} + U_{Coulomb} + U_{LJ}$. Bond stretch interactions involve two particles, angle bending interactions involve three particles, and the torsion terms involve four particles. All of the nonbonded interactions involve pairs of particles. The nonbonded interactions between particles that also have bonded interactions are typically modified or eliminated.

$$U_{\text{Stretch}} = \sum_{\text{bonds (ij)}} K(ij)\,(r_{ij} - r_{ij}^{eq})^2$$

$$U_{\text{Bend}} = \sum_{\text{angles (ijk)}} K(ijk)\,(\theta_{ijk} - \theta_{ijk}^{eq})^2$$

$$U_{\text{Torsion}} = \sum_{\text{torsions (ijkl) n=1, 2, ...}} \sum V_{ijkln}\,[1 + \cos(n\phi_{ijkl} - \gamma_{ijkl})]$$

$$U_{\text{LJ}} = \sum_{\text{nonbonded (ij,i<j)}} \frac{A_{ij}}{r_{ij}^{12}} - \frac{B_{ij}}{r_{ij}^{6}}$$

$$U_{\text{Coulomb}} = \sum_{\text{nonbonded (ij)}} \frac{q_i\,q_j}{\varepsilon r_{ij}}$$

bend interaction. Bond stretch and angle bending movements are generally very small, but torsional movement may involve significant rotational displacements. As Figure 6 indicates, bond and angle expressions are often harmonic in nature, whereas torsional energy contributions are usually expressed as a short sum of cosine functions. Note that the sum in the bond expression is over all pairs of atoms that are considered to be covalently bonded. The sum in the angle expression is over all sets of three atoms that share two covalent bonds. The sum in the torsion expression is over all sets of four atoms that share three covalent bonds.

The nonbonded forces are also illustrated in Figure 6. These consist of Coulomb interactions and Van der Waals interactions. The Coulomb interactions between the atoms are due to charges and have the longest range in terms of the extent of their influence. These interactions play a key role in defining the energies of hydrogen bonds, which are ubiquitous both intraprotein, intrawater, and between protein and water. The binding of solids such as solid hydrocarbons, and many of the interactions within the cores of folded protein molecules, are dominated by the Lennard-Jones (LJ), or Van der Waals interaction. This is comprised of the fluctuating-dipole or dispersive attraction (r^{-6} term), together with exchange repulsion, which in the LJ form of the interaction is artificially modeled by the r^{-12} term. Note that the sums in the Coulomb and LJ expressions include only *nonbonded* pairs of atoms. These are pairs of atoms that are neither covalently bonded nor connected by an angle energy expression. Pairs of atoms that are neither bonded nor connected by an angle expression, yet are coupled by a torsion expression, are sometimes known as 1–4 nonbonded pairs. Depending on the model potential in use, these nonbonded interactions are typically excluded or modified. The modifications may, for example, be different LJ parameters or use some multiplicative attenuation factor.

In these energy expressions, note that there are a number of *force field parameters*, such as spring constants for bond stretch and angle bending, equilibrium bond lengths and angles, torsional energy prefactors and phase angles, LJ parameters and atomic charges. The different model potentials in common use differ not only in the choice of parameters, but also to some extent in the number and complexity of functional forms they employ.

In order for a science program based on large-scale simulation to be a success, the model potentials used must adequately represent the relevant physics and chemistry involved. The project is to some extent hostage to the accuracy of the force fields available. Existing force fields are remarkably successful in some areas. However, the suitability of existing force fields for accurately modeling the dynamics of the folding process is an area of active investigation and refinement of the models used is an ongoing process. The force fields in common use to model the protein and water system include, for example, CHARMM,[26] AMBER,[27] GROMOS,[28] and OPLS-AA.[29] Beyond improved parameterization of the existing models, the inclusion of additional physical phenomena such as polarizability is being investigated.[30,31]

It seems obvious that calculations of protein folding pathways and rates would be very sensitive to variations in the force field. However, strong arguments exist that in fact, paradoxically, the phenomenon of folding may be quite robust. The notion of *designability* of a native fold, meaning that the fold is stable over a significant variation of amino acid content, is one aspect of the robustness. Also, many proteins exist in highly mutated forms across and

within species. These mutated forms often have the same fold. On the other hand, mutated proteins often have much slower folding rates, even if the same structure is ultimately produced. And furthermore, it is well known that a number of diseases exist that can be traced to changes in a single amino acid in the protein, which presumably results in unfoldable or misfolded proteins. This argues for the possibility that folding may be very sensitive to force-field quality. Characterizations, comparisons, and assessments of force fields are expected to represent a significant area of activity for the Blue Gene science program. Thus, the software application infrastructure that supports the science must have the ability to support multiple force fields and new developments in force-field technology.

Simulation options. It is very important to dispel the notion that the Blue Gene resource will be applied to study a single folding event of a single protein. For physically meaningful studies of protein folding, it will be necessary to simulate a large number of trajectories in order to reach conclusions supported by reasonable statistics. Estimates of the number of trajectories required, for each system studied, range from 10 to 100 in order, for example, to derive a crude estimate of the characteristic time for a step in a folding process. This requirement places limitations on the sizes of systems and the lengths of individual dynamical simulations that can be undertaken. There is some evidence that multiple simulations of limited duration provide more information than a single longer trajectory representing an equivalent amount of computation.[32] For force-field characterizations, too, many simulations will be required on a large variety of protein and peptide systems using different force fields.

We therefore anticipate the simulation of a large variety of protein and smaller peptide systems in the Blue Gene program for various amounts of time and, for many, with a high degree of replication in order to obtain meaningful statistics. We also anticipate that effort will be spent on the implementation and investigation of newly developed, and perhaps more efficient, algorithms for investigating protein science.

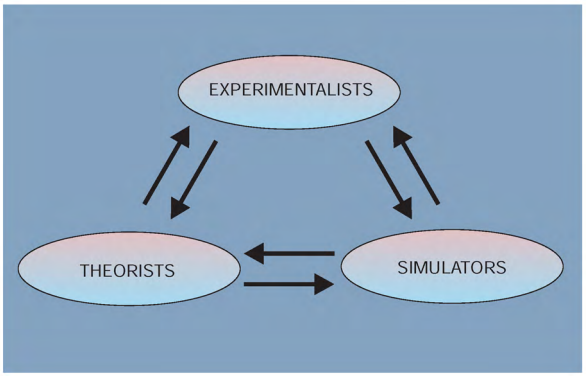
A number of new techniques are becoming available that might prove useful for studying protein processes such as folding. These, and others yet to come, should be examined for applicability in the context of the hardware and science program. For studies of protein folding kinetics, it may be possible to exploit acceleration techniques. Examples include the parallel replica method of Voter,[33,34] the reaction path approach of Elber,[35] and the transition path sampling approach of Chandler and coworkers.[36]

Among the areas we plan to investigate are folding kinetics, pathway characterization, and force-field comparison and assessment. In kinetic studies, many occurrences of the relevant events are required to reach any well-supported conclusions. Simulations that probe for information about kinetics will need to be performed using constant energy dynamics, and highly accurate trajectories will need to be generated to avoid heating the system. Among the important questions to investigate in kinetic studies, in addition to attempting to predict rates, is whether any observations are consistent with recent theories that relate folding to chemical reaction kinetics or nucleation events as observed in first-order phase transitions.[16] These studies will also require the development of new trajectory analysis tools.

For folding pathway characterizations, we might develop and employ technology similar to that of Sheinerman and Brooks,[37] where free energy along folding pathways is characterized using sophisticated thermodynamic sampling techniques. These methods can provide quantitative (subject to the approximations of the model and simulation methods) characterization of the protein folding landscape. Such simulations require the implementation of thermal- and pressure-control technology in the application, as well as the implementation of potential of mean force and umbrella sampling techniques.

Some of the studies will target force-field characterization through a variety of probes. The simplest of these are studies of the structural stability of experimentally generated native structures using different force fields. These are likely to involve both kinetic and thermodynamic sampling simulations[12] of small peptides in water over tens of nanoseconds—times characteristic of fast alpha-helix formation. Another type of study in which known protein structures are partially unfolded by applying a heat pulse, and then refolded, can provide further insight into the ability of the force fields to correctly reproduce free-energy minima. Another possible way to compare and assess force fields is to see if the trends in the free energy of solvation or the partition coefficient for peptides that is observed in experiments can be computed using free-energy methods implemented in the Blue Gene application.

Figure 7 This figure illustrates the interactions between the major intellectual communities participating in the science of protein folding. Extensive opportunities for interchange of ideas exist between each of these communities. Experimentalists can suggest new systems and properties to study in simulations. Simulators can suggest systems to be probed by experimentalists. Theorists can suggest systems for study by experimental and simulation methods. Simulators can feed back microscopic information to theorists that may support or refute existing theories or suggest potential avenues for improvement. Experimentalists can similarly interact with theorists via their data. It is hoped that the Blue Gene project will allow IBM to act as a catalyst to stimulate these interactions.



Overview of planned science program. The Blue Gene science program is planned to be incremental, with early studies being carried out on currently available hardware platforms. An application suite will evolve that makes use of existing software tools and biomolecular modeling and simulation software packages, where appropriate. In fact, a rich set of function exists today in commercial, academic, and public domain tools that allow visualization, problem setup, molecular coordinate manipulation, and trajectory generation and analysis. Some of the initial simulations may be performed using these commercially available software packages. However, new software will ultimately need to be developed for the computational core in order to efficiently exploit the special hardware being designed and to explore novel techniques and algorithms.

Even with the anticipated computational power available to the Blue Gene project for MD simulations, careful consideration must be given to the scientific program in order to utilize the capability of the hardware most effectively to advance our understanding of the protein folding process. Therefore, a key aspect of the Blue Gene science program will be outreach to the worldwide scientific community of protein experimentalists, simulation methodology and application experts, and biophysical theorists in academia, government laboratories, and commercial life sciences institutions. It is hoped that this project can help catalyze interactions within the life sciences community, as shown in Figure 7. This is essential if we are to have a strong and relevant scientific impact in protein science through the application of the hardware.

We intend to engage the scientific community directly by organizing and hosting workshops, conferences, and seminars. At the time this paper was being written, the first IBM-sponsored protein science workshop planned to meet at the University of California in San Diego on March 30–31, 2001. One of the primary functions of the meeting was to generate suggestions for new collaborative work in the field, especially, but not limited to, work that may bear directly on the Blue Gene project.

Hardware

A petaflop/s power (10^{15} floating-point operations per second) computer, with something like 50 times the computing power of all supercomputers in the world today, can only be realized through massive parallelism. Such a machine would need to have high-bandwidth, low-latency communications; otherwise it would be essentially limited to data-parallel operation, a mode resulting in unacceptably long turnaround time for jobs, each of which would be run on a low-power subset of the machine. A conventional implementation of a petaflop/s machine would be too massive, expensive, and unreliable. Because of the communications requirement, solutions involving huge numbers of loosely coupled personal computers are too inefficient for the desired applications.

In the MD GRAPE approach,[38] a set of boards containing a GRAPE chip hard-wired to implement the full $O(N^2)$[39] long-range force calculation is coupled to a host machine in which the short-range forces are computed; in this approach the use of finite-range cutoffs of the long-range forces to improve performance is discarded and compensated for by the efficiency of "pipelining" the calculation of the long-range forces done in the GRAPE chip. There is also a WINE chip for implementing the k-space part of

Ewald. Scaling up such a highly specialized machine to the necessary size would require an appropriate scaling up of the host machine so that the bonded force component (which is calculated more frequently than the long-range forces when using the multiple time-step technique) remains in scale. This would seem to be a highly complex project involving many different types of chips, and therefore an expensive approach.

We next discuss the "cellular" Blue Gene architecture for achieving a petaflop/s machine, a relatively unspecialized approach that seems to be feasible.

High-performance supercomputers are built today by connecting together large numbers of nodes, each consisting of a conventional server system (uniprocessor, or small-scale shared-memory multiprocessor). The ASCI Blue Pacific IBM SP* system, which achieves a performance in excess of 10 teraflops/second (peak), demonstrates the top performance achievable by such systems. However, it will be difficult to scale up the performance of such systems to petaflop/s performance in the near future.

A petaflop/s system built out of conventional server nodes would consume hundreds of megawatts of electrical power; it would require many acres of machine room floor; it would likely have an unacceptably low mean time between failures (an MTBF in excess of 100 hours is considered remarkable in current high-end supercomputers); it would be exceedingly difficult to program and manage; and it would have an unacceptably high price tag.

There are several reasons for this. Conventional microprocessors are built to execute one sequential instruction stream as fast as possible. Increasingly large amounts of hardware are used to extract parallelism from a sequential instruction stream with techniques such as register renaming, speculative execution, branch prediction, and so on. These techniques yield diminishing returns: whereas the number of gates in integrated microprocessors has increased by three orders of magnitude in the last two decades, the number of instructions executed at each cycle has increased by a factor of ten, at best.

A simplified microprocessor design leads to higher efficiency and enables many processor cores to fit on a single chip. This has the added advantage that wires in the processor core can be kept short, further improving performance, and that it is possible to achieve higher chip manufacturing yields by using chips with some faulty processors. This approach is worthwhile if one targets applications with significant amounts of parallelism, so that a large number of individual processors can be usefully applied. Molecular dynamics is one example of such an application.

Current microprocessor architectures suffer from the well-known "Von Neumann bottleneck": memory access time is measured in hundreds of processor cycles, and the compute units often stall, waiting for data to arrive from memory. This leads to increasingly complex logic to support long execution pipelines and to increasingly complex cache hierarchies to satisfy most memory accesses from smaller, but faster, cache memory. While caches may take half of the area of a microprocessor chip, they still do a poor job for many scientific codes. Current CMOS (complementary metal-oxide semiconductor) technology, in particular, IBM Blue Logic technology,[40] provides efficient support for embedded DRAM (dynamic random-access memory) cores on logic chips. The Von Neumann bottleneck is minimized if this technology is used to integrate processor(s) and memory on one chip, thus providing a low-latency, high-bandwidth processor-memory interface. However, while it is quite easy to fit on one chip sufficient floating-point logic to execute tens of billions of floating-point operations per second (flop/s), it is not possible to fit on one processor chip much more than 32 megabytes of memory. The usual "rule of thumb" of one byte of memory per flop/s precludes the use of merged DRAM logic for systems with any interesting performance. The door for the creative use of merged DRAM logic opens, once it is understood that these rules of thumb for balanced systems apply to servers built to accommodate a broad mix of workloads, but may not apply to systems targeting a narrower range of applications. Preliminary analysis indicates that some MD codes can be run efficiently using less than one byte per second per 1000 flop/s.

More generally, thinking on algorithm and system design is still dominated by the invalid perception that compute units are the most expensive resource in a computing system. Applications are designed to minimize the number of operations required to solve a problem. Systems are designed with enough memory and I/O to achieve a high usage of the compute units, even if this leads to low memory or disk utilization. In fact, the cost of systems is dominated by storage cost, and by the cost of the logic required to move data around (caches and buses). If one

thinks anew from basic principles, the conclusion will often be that algorithms that use more computing but less memory, or use more computing but require less communication, coupled with systems designed to ensure more effective use of memory and communication, even at the expense of lower utilization of the processing units, are more cost-effective than conventional algorithms and systems.

The building block for Blue Gene will be a chip that integrates multiple processors, memory, and communication logic.

Conventional high-end systems are clusters of nodes, each controlled by a full operating system. This significantly increases (software) error rates, as well as introduces major inefficiencies. The predominant usage mode for such machines is a partition dedicated for a significant period of time to the execution of one parallel application; the efficient execution of such an application requires the tight coordination of all resources in the partition. Yet, each operating system instance allocates resources (processor time slices and memory) autonomously, with no awareness of the required coupling with other nodes. Indeed, efficient performance is achieved only when the operating system is "out of the way"; many of the mechanisms and policies of the operating system become obstacles to performance, rather than supporting the right virtual parallel machine abstraction. Performance and stability can be significantly improved by using at each node a simple kernel that provides a thin layer of protection atop the hardware, and by providing global mechanisms and policies for the management of partitions.

Even though access to on-chip memory can be much faster than to off-chip memory, DRAM access will still require multiple cycles. We use *simultaneous multithreading* to hide this memory latency. The basic building block for processors is a *thread unit* that executes an autonomous instruction stream. Each thread unit has its own register set and its own instruction sequencer. Multiple thread units within one *processor* share more expensive resources, such as the double-precision floating-point units, instruction cache, and memory bus. If one thread unit stalls as it waits for a load to complete from memory, then the shared processor resources can be used by other thread units. In effect, one has replaced one fast processor by multiple slower thread units that are better matched to the memory latency. This is a worthwhile choice if application-level parallelism is there to be exploited. Assuming a 500-megahertz clock, each processor can execute up to one gigaflop/s (two floating-point instructions at each cycle).

These considerations dictate the design for the Blue Gene system. The building block for Blue Gene will be a single chip that integrates multiple processors, as just described, memory, and communication logic. This will allow us to build a full system, essentially by replicating this one chip.

We have determined that it is possible to fabricate in currently available standard cell technology an inexpensive chip with double-precision scalar performance on the order of 32 gigaflop/s, internal DRAM capacity around 8 megabytes, and external communication bandwidth exceeding 12 gigabytes per second. A novel mechanical packaging scheme maps a $32 \times 32 \times 32$ cube of these chips into a system covering an area of about 40×40 feet. Power consumption for a performance of one petaflop/s is under two megawatts.

The basic programming model offered by a Blue Gene system is very similar to that offered by large-scale clusters today. Each chip is a shared memory multiprocessor that runs simultaneous multiple threads. Threads running on the same chip communicate via shared memory. Communication between chips uses message passing. However, the sheer size of the system will force us to evolve toward higher-level programming models that will be mapped on the actual hardware by a combination of compile-time and run-time techniques.

A major concern on a system of this size is error recovery. We expect that the incidence of software errors will be kept low by keeping the resident software very simple, and by applying correctness-proving techniques to key software subsystems, such as the communication software. However, the large number of components imply that hardware failures will be very likely to occur during any one job run. Furthermore, there will be a non-negligible likelihood that hardware errors will corrupt the state of a computation without being detected by hardware. Software must assume responsibility for error correction and, to some extent, for error detection. A

simple method for detecting errors is to replicate computation and compare results. However, computation mirroring reduces the effective performance by a factor of two. Less onerous error detection may be possible through use of application-dependent redundancy, in the computation state, to periodically validate this state. Error recovery will be based on the ability to isolate faulty components and to restart a computation on the remaining components from a previously taken checkpoint.

The architectural choices outlined in this section have their downside; the low amount of memory restricts the range of applications that can exploit this architecture. Compatibility with existing software is not guaranteed. We believe that an improvement of two orders of magnitude in compute performance per watt, or compute performance per gate count, is well worth this trade-off. At the very high end this might be the only practical way of approaching petaflop/s performance levels.

Application issues

Given the nature of the architecture just described, it is clear that a number of challenges must be faced in crafting an application to execute the science program on the target machine platform. At the highest level, a major challenge is to understand how to extract the maximal amount of concurrency from the application. The scientific goals of the project may require simulation of a fixed-size system for a very large number of time steps.

In the following discussion, some of the known strategies for mapping such simulations onto very large numbers of nodes are described, along with the issues connected with these strategies. A high-level description of selected algorithmic alternatives available for improving the efficiency of the dominant computational burden is provided; we end with a brief survey of some areas of exploration.

Scalability challenge. With an immense number (more than 30000) of nodes, an even larger number of CPUs (approximately 1000000), and a yet larger number of hardware thread contexts (more than 5000000), the challenge of mapping a fixed-size N-body simulation onto such a massively parallel machine is considerable.[41] It is most common to define scalability in terms of the ability to employ greater numbers of nodes to solve larger problems, where "larger" in this case refers to the number of particles, N.

One important research area for the scientific program is the study of very long timescale phenomena within the protein folding process. We would like to use the approximately 100-fold increase in computational power that the hardware potentially offers to realize a similar 100-fold increase in the timescales probed by the scientific program.

This requires the ability to use additional hardware threads to increase the number of time steps that can be simulated in a fixed amount of time on a problem of fixed size. Since the systems most relevant to the science program contain approximately 30000 atoms, the ability to use more hardware threads than there are particles in the simulation is required, if one wishes to run on the full unpartitioned machine.

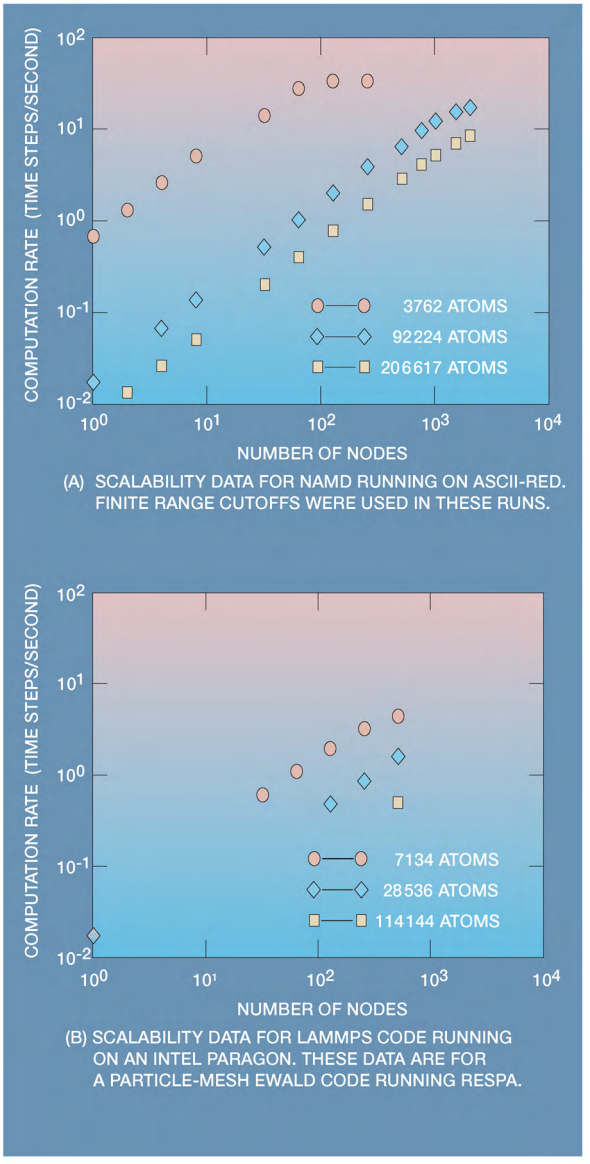
Existing approaches. For concreteness, this discussion focuses on approaches aimed at molecular dynamics. An extensive review of atomistic simulation techniques on parallel machines has recently appeared[42] that provides a broader survey than that provided here. We provide a brief description of current approaches to decomposing the problem on a parallel machine and also some of the currently popular algorithmic alternatives.

In the process of integration of the equation of motion required by molecular dynamics, the integration itself occupies negligible time, since it is of order N in an N-atom system, and involves simple arithmetic. The various contributions to the potential energy of the system are shown in Figure 6, and the evaluation of the corresponding force components on each particle dominates the computation.

There are only $O(N)$ bonded force evaluations and they typically consume anywhere from one to 20 percent of the total force computation in a uniprocessor environment.[25] This variation is due to both variation in system type and to use of multiple time-step integration techniques like RESPA[20] that can change the relative frequencies at which various force terms are evaluated. The force computation is dominated by the nonbonded interactions, comprising the LJ and Coulomb contributions. The LJ forces go out to large numbers of near neighbors, whereas the long-ranged Coulomb forces created by one charge are felt by every other charge in the system. Calculation of the long-range forces is the most expensive part of the MD calculation.

How can the immense parallelism of the machine be exploited to distribute this computational burden

Figure 8 The vertical axis represents the rate at which time steps are computed. With perfect scalability, the data would fall on a line with slope equal to one. The behavior shown here is typical of a parallel molecular dynamics code; scalability is better for systems with large numbers of particles.



without running into communications problems and problems with the machine's limited memory? In a parallel environment, there are three basic approaches:

1. *Atom decomposition* binds the computation of all the forces on a single particle to a particular hardware thread.
2. *Force decomposition* binds the computation of a particular force term (usually a two-body force) to a particular hardware thread.
3. *Spatial decomposition* binds the computation of all the forces on particles within a volume element to a particular hardware thread. Particles may "migrate" from hardware thread to hardware thread during the course of the simulation.

The communication expense of these three approaches has been reviewed by Plimpton[43] with regard to molecular dynamics simulations with short-ranged interactions.

Spatial decomposition is most efficient if the cutoff radius for the real space force sums is relatively small, but there is heightened interest in using Ewald[14,25] methods in simulations of biomolecular systems to take into account long-ranged Coulombic forces via periodic boundary conditions. The use of this technique imposes additional considerations and complicates the issue of choosing the most appropriate approach for problem decomposition.

The Ewald technique expresses the slowly convergent Coulomb sum in terms of two rapidly convergent series, one in real space and one in Fourier (k-) space. The optimal complexity of the Ewald Coulomb sum is $O(N^{3/2})$, i.e., each charge can be thought of as interacting with only $O(N^{1/2})$ other charges. Implementation of the Fourier transform using fast Fourier transform (FFT) techniques[44] enables a speed up to $O(N \log N)$; due to the communication demands of the FFT, this approach presents challenges in a massively parallel environment and may raise the crossover point in particle number at which the FFT-based methods become more efficient than a standard Ewald calculation.

A different approach, with a computational complexity of $O(N)$, uses multipole expansions to evaluate the Coulomb interaction.[45] But the crossover in N at which the multipole approach becomes the most efficient is typically at higher N than for the FFT approach. The optimal choice of algorithm depends on system size and efficiency of implementation on a particular hardware platform. It should be noted that the effective system size, N, is the number of charges, *not* the number of atoms, and can be influenced by the choices made in modeling the system. For the TIP5P[23] model of water, five charge centers per wa-

ter molecule are required as opposed to the three charge centers required by other models, such as TIP3P[23] or SPC/E.[24]

A feeling for the "state of the art" in mapping molecular dynamics codes onto parallel machines is provided in Figure 8.[46,47] It is important to remember that these codes are addressing the challenge of mapping macromolecular system simulations containing bonded as well as nonbonded interactions onto massively parallel machines. One of these studies notes that scalability begins to fall off when the average number of atoms per hardware thread drops below O(10).[46] It is clear that there is a considerable mountain to climb in the course of mapping a single molecular dynamics simulation of modest size onto machine architectures with hardware thread counts much larger than one thousand.

Areas for exploration. There are a number of areas to be explored that might allow one to overcome or evade the scalability challenge just described:

- Improved integrators that might allow use of larger time steps
- Improved implementations and algorithms for long-range force evaluation. It is possible that existing statements about the suitability of specific algorithms for specific problems based on problem size may have to be modified to take into account the suitability of those algorithms for mapping onto a massively parallel machine.
- Improved techniques for calculating thermodynamic ensemble averages. Use of ensemble-averaging techniques such as Monte Carlo may allow more efficient utilization of the machine than conventional molecular dynamics in calculating thermodynamic ensemble averages. Also, free-energy calculation techniques have an embarrassingly parallel component that can utilize a partitioned set of hardware threads.
- Novel methods for studying long-time dynamics in biomolecular systems that may circumvent existing scalability issues[32,33]

In order to explore algorithmic and decomposition alternatives, it is essential to have a good set of simulation environments and tools to support application development and tuning. Modeling the highly threaded machine architecture currently being planned represents a considerable challenge. In order to achieve performance on such platforms, a nonintrusive run-time environment with extremely low overhead will also be needed.[48]

Summary

The Blue Gene project represents a unique opportunity to explore novel research into a number of areas, including machine architecture, programming models, algorithmic techniques, and biomolecular simulation science. As we discussed, every aspect of this highly adventurous project involves significant challenges. Carrying out our planned program will require a collaborative effort across many disciplines and the involvement of the worldwide scientific and technical community. In particular, the scientific program will engage with the life sciences community in order to make best use of this unique computational resource.

*Trademark or registered trademark of International Business Machines Corporation.

Cited references and note

1. A. R. Fersht, *Structure and Mechanism in Protein Science: A Guide to Enzyme Catalysis and Protein Folding*, W. H. Freeman, New York (1998).
2. J. Beetem, M. Denneau, and D. Weingarten, "The GF11 Parallel Computer," *Experimental Parallel Computing Architectures*, J. J. Dongarra, Editor, North Holland Publishing Co., Amsterdam (1987).
3. D. H. Weingarten, "Quarks by Computer," *Scientific American* **274**, No. 2, 104–108 (1996).
4. W. Andreoni, A. Curioni, and T. Mordasini, "DFT-Based Molecular Dynamics as a New Tool for Computational Biology: First Applications and Perspective," *IBM Journal of Research and Development* **45,** Nos. 3 and 4 (2001).
5. Q. Zhong, T. Husslein, P. B. Moore, D. M. Newns, P. C. Pattnaik, and M. L. Klein, *FEBS Letters* **434**, No. 3, 265–271 (1998).
6. J. Collinge, "Variant Creutzfeldt-Jakob Disease," *The Lancet* **354**, No. 9175, 317–323 (1999).
7. E. Strickland, B.-H. Qu, and P. J. Thomas, "Cystic Fibrosis: A Disease of Altered Protein Folding," *Journal of Bioenergetics and Biomembranes* **29**, 483–490 (1997).
8. C. Branden and J. Tooze, *Introduction to Protein Structure*, 2nd Edition, Garland Inc., New York (1999), and references therein.
9. J. Moult, T. Hubbard, K. Fidelis, and J. T. Pedersen, "Critical Assessment of Methods of Protein Structure Prediction (CASP): Round III," *Proteins: Structure, Function, and Genetics* **37**, Supplement 3, 2–6 (1999).
10. C. Levinthal, "Are There Pathways for Protein Folding?" *Journal de Chimie Physique* **65**, 44–45 (1968).
11. W. Jin, J. Onuchic, and P. Wolynes, "Statistics of Kinetic Pathways on Biased Rough Energy Landscapes with Applications to Protein Folding," *Physical Review Letters* **76**, 4861–4864 (1996).
12. F. B. Sheinerman and C. L. Brooks, "Molecular Picture of Folding of a Small Alpha/Beta Protein," *Proceedings of the National Academy of Sciences (USA)* **95**, No. 4, 1562–1567 (1998).
13. V. Munoz, P. A. Thompson, J. Hofrichter, and W. A. Eaton, "Folding Dynamics and Mechanism of Beta-Hairpin Formation," *Nature* **390**, 196–198 (1997).

14. D. Frenkel and B. Smit, *Understanding Molecular Simulation*, Academic Press, San Diego, CA (1996).
15. S. E. Jackson, "How Do Small Single-Domain Proteins Fold?" *Folding & Design* **3**, R81–R91 (August 1998).
16. V. S. Pande, A. Y. Grosberg, T. Tanaka, and D. S. Rokhsar, "Pathways for Protein Folding: Is a 'New View' Needed?" *Current Opinion in Structural Biology* **8**, 68–79 (1998).
17. D. Thirumalai and D. K. Klimov, "Deciphering the Timescales and Mechanisms of Protein Folding Using Minimal Off-Lattice Models," *Current Opinion in Structural Biology* **9**, No. 2, 197–207 (1999).
18. Y. Duan and P. A. Kollman, "Pathways to a Protein Folding Intermediate Observed in a 1-Microsecond Simulation in Aqueous Solution," *Science* **282**, No. 5389, 740–744 (1998).
19. W. C. Swope, H. C. Andersen, P. H. Berens, and K. R. Wilson, "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 (1982).
20. M. Tuckerman, B. J. Berne, and G. J. Martyna, "Reversible Multiple Time Scale Molecular Dynamics," *Journal of Chemical Physics* **97**, No. 3, 1990–2001 (August 1992).
21. J. P. Ryckaert, G. Ciccotti, and H. J. C. Berendsen, "Numerical Integration of the Cartesian Equations of Motion of a System with Constraints: Molecular Dynamics of n-Alkanes," *Journal of Computational Physics* **23**, 327–341 (1977).
22. H. C. Andersen, "RATTLE: A Velocity Version of the SHAKE Algorithm for Molecular Dynamics," *Journal of Computational Physics* **52**, 24–34 (1983).
23. M. W. Mahoney and W. L. Jorgensen, "A Five-Site Model for Liquid Water and the Reproduction of the Density Anomaly by Rigid, Nonpolarizable Potential Functions," *Journal of Chemical Physics* **112**, No. 20, 8910–8922 (2000).
24. H. J. C. Berendsen, J. R. Grigera, and T. P. Straatsma, *Journal of Physical Chemistry* **91**, 6269 (1987).
25. M. P. Allen and D. J. Tildesley, *Computer Simulation of Liquids*, Oxford University Press, New York (1989).
26. A. D. MacKerell, D. Bashford, M. Bellott, R. L. Dunbrack, J. D. Evanseck, M. J. Field, S. Fischer, J. Gao, H. Guo, S. Ha, D. Joseph-McCarthy, L. Kuchnir, K. Kuczera, F. T. K. Lau, C. Mattos, S. Michnick, T. Ngo, D. T. Nguyen, B. Prodhom, W. E. Reiher, B. Roux, M. Schlenkrich, J. C. Smith, R. Stote, J. Straub, M. Watanabe, J. Wiorkiewicz-Kuczera, D. Yin, and M. Karplus, "All-Atom Empirical Potential for Molecular Modeling and Dynamics Studies of Proteins," *Journal of Physical Chemistry B* **102**, 3586–3616 (1998).
27. W. D. Cornell, P. Cieplak, C. I. Bayly, I. R. Gould, K. M. Merz, D. M. Ferguson, D. C. Spellmeyer, T. Fox, J. W. Caldwell, and P. A. Kollman, "A Second Generation Force Field for the Simulation of Proteins and Nucleic Acids," *Journal of the American Chemical Society* **117**, 5179–5197 (1995).
28. W. F. van Gunsteren, X. Daura, and A. E. Mark, "GROMOS Force Field," *Encyclopaedia of Computational Chemistry*, Volume 2 (1998).
29. G. Kaminski, R. A. Friesner, J. Tirado-Rives, and W. L. Jorgensen, "Evaluation and Improvement of the OPLS-AA Force Field for Proteins via Comparison with Accurate Quantum Chemical Calculations on Peptides," submitted to *Journal of Computational Chemistry* (2001).
30. J. L. Banks, G. A. Kaminski, R. Zhou, D. T. Mainz, B. J. Berne, and R. A. Friesner, "Parameterizing a Polarizable Force Field from *ab initio* Data: I. The Fluctuating Point Charge Model," *Journal of Chemical Physics* **110**, 741–754 (1999).
31. J. Ponder, private communication.
32. V. Daggett, "Long Time-Scale Simulations," *Current Opinion in Structural Biology* **10**, 160–164 (2000).
33. A. F. Voter, "Parallel Replica Method for Dynamics of Infrequent Events," *Physical Review B* **57**, No. 22, 13985–13988 (1998).
34. V. S. Pande, private communication (2000).
35. R. Olender and R. Elber, "Calculation of Classical Trajectories with a Very Large Time Step: Formalism and Numerical Examples," *Journal of Chemical Physics* **105**, No. 643, 9299–9315 (1996).
36. P. G. Bolhuis, C. Dellago, P. L. Geissler, and D. Chandler, "Transition Path Sampling: Throwing Ropes over Mountains in the Dark," *Journal of Physics: Condensed Matter* **12**, No. 8A, 147–152 (2000).
37. F. B. Sheinerman and C. L. Brooks, "Calculations on Folding of Segment b1 of Streptococcal Protein g," *Journal of Molecular Biology* **278**, No. 2, 439–456 (1998).
38. T. Fukushige, M. Taiji, J. Makino, T. Ebisuzaki, and D. Sugimoto, "A Highly Parallelized Special-Purpose Computer for Many-Body Simulations with an Arbitrary Central Force: MD:GRAPE," *Astrophysical Journal* **468**, 51–61 (1996).
39. This notation indicates order of complexity. Here $O(N^2)$ means that as N increases, the increase in time required for the computation is proportional to N^2.
40. See http://www.chips.ibm.com/products/asics/products/edram.
41. V. E. Taylor, R. L. Stevens, and K. E. Arnold, "Parallel Molecular Dynamics: Implications for Massively Parallel Machines," *Journal of Parallel and Distributed Computing* **45**, No. 2, 166–175 (September 1997).
42. G. S. Heffelfinger, "Parallel Atomistic Simulations," *Computer Physics Communications* **128**, Nos. 1–2, 219–237 (June 2000).
43. S. Plimpton, "Fast Parallel Algorithms for Short-Range Molecular Dynamics," *Journal of Computational Physics* **117**, No. 1, 1–19 (March 1995).
44. R. W. Hockney and J. W. Eastwood, *Computer Simulation Using Particles*, Institute of Physics Publishing, Bristol, UK (1988).
45. L. Greengard and V. Rokhlin, "A Fast Algorithm for Particle Simulations," *Journal of Computational Physics* **73**, 325–348 (1987).
46. R. K. Brunner, J. C. Phillips, and L. V. Kale, "Scalable Molecular Dynamics for Large Biomolecular Systems," *Supercomputing 2000 Proceedings*, Dallas, TX (November 4–10, 2000), available at http://www.sc2000.org/proceedings/techpapr/papers/pap271.pdf.
47. S. Plimpton, R. Pollock, and M. Stevens, "Particle-Mesh Ewald and rRESPA for Parallel Molecular Dynamics Simulations," *Proceedings of the Eighth SIAM Conference on Parallel Processing for Scientific Computing*, Minneapolis, MN (March 14–17, 1997), pp. 8–21.
48. B. G. Fitch and M. E. Giampapa, "The Vulcan Operating Environment: A Brief Overview and Status Report," *Parallel Supercomputing in Atmospheric Science*, G.-R. Hoffman and T. Kauranne, Editors, World Scientific Publishing Co., Inc., Riveredge, NJ (1993), p. 130.

Accepted for publication March 8, 2001.

IBM Blue Gene team *IBM Research Division, Thomas J. Watson Research Center, P.O. Box 218, Yorktown Heights, New York 10598 (electronic mail: rgermain@us.ibm.com).* The Blue Gene team spans a wide range of technical disciplines and organizations within IBM Research. Blue Gene team members are listed below, grouped by their contributions to the areas of hardware, systems software, and application/science.

Hardware: Ruud Haring is the manager of the Blue Gene Systems Development group and Monty Denneau is the chief architect of the Blue Gene chip and system. Other past and present contributors to the hardware effort are: Daniel K. Beece, Arthur A. Bright, Paul Coteus, Bruce Fleischer, Christos J. Georgiou, Peter H. Hochschild, Kiran K. Maturu, Robert Philhower, Thomas Picunko, Rick A. Rand, Valentina Salapura, Rahul S. Shah, Sarabjeet Singh, Richard Swetz, Nagesh K. Vishnumurthy, and Henry S. Warren, Jr.

Systems software: Manish Gupta is the manager of the High Performance and Cellular Programming Environments group. Jose Moreira is the manager of the Blue Gene System Software group. Other past and present contributors to the systems software effort are: Frances Allen, George Almasi, Jose Brunheroto, Calin Cascaval, Jose Castanos, Paul Crumley, Wilm Donath, Maria Eleftheriou, Mark Giampapa, Howard Ho, Derek Lieber, Matthew Newton, Alda Sanomiya, and Marc Snir.

Application and science: Robert S. Germain is the manager of the Biomolecular Dynamics and Scalable Modeling group and Blake G. Fitch is the application architect. Other past and present contributors to the science and application effort include: Wanda Andreoni, Bruce J. Berne, Alessandro Curioni, Donna Gresh, Susan Hummel, Tiziana Jonas, Glenn Martyna, Dennis Newns, Jed Pitera, Michael Pitman, Ajay Royyuru, Yuk Sham, Frank Suits, William Swope, T. J. Christopher Ward, and Ruhong Zhou.