Chapter 9

Biomolecular Modeling in the Era of Petascale Computing

Klaus Schulten

Beckman Institute, University of Illinois at Urbana-Champaign

James C. Phillips

Beckman Institute, University of Illinois at Urbana-Champaign

Laxmikant V. Kalé

Department of Computer Science, University of Illinois at Urbana-Champaign

Abhinav Bhatele

Department of Computer Science, University of Illinois at Urbana-Champaign

9.1 Introduction

The structure and function of biomolecular machines are the foundation on which living systems are built. Genetic sequences stored as DNA translate into chains of amino acids that fold spontaneously into proteins that catalyze chains of reactions in the delicate balance of activity in living cells. Interactions with water, ions, and ligands enable and disable functions with the twist of a helix or rotation of a side chain. The fine machinery of life at the molecular scale is observed clearly only when frozen in crystals, leaving the exact mechanisms in doubt. One can, however, employ molecular dynamics simulations to reveal the molecular dance of life in full detail. Unfortunately, the stage provided is small and the songs are brief. Thus, we turn to petascale parallel computers to expand these horizons.

Biomolecular simulations are challenging to parallelize. Typically, the molecular systems to be studied are not very large in relation to the available memory on computers: they contain ten thousand to a few million atoms. Since the size of basic protein and DNA molecules to be studied is fixed, this number does not increase in size significantly. However, the number of time steps to be simulated is very large. To simulate a microsecond in the life of a biomolecule, one needs to simulate a billion time steps. The challenge posed by biomolecules is that of parallelizing a relatively small amount of computation at each time step across a large number of processors, so that billions of time steps can be performed in a reasonable amount of time. In particular, an important aim for science is to effectively utilize the machines of the near future with tens of petaflops of peak performance to simulate systems with just a few million atoms. Some of these machines may have over a million processor cores, especially those designed for low power consumption. One can then imagine the parallelization challenge this scenario poses.

NAMD [15] is a highly scalable and portable molecular dynamics (MD) program used by thousands of biophysicists. We show in this chapter how NAMD's parallelization methodology is fundamentally well-suited for this challenge, and how we are extending it to achieve the goals of scaling to petaflop machines. We substantiate our claims with results on large current machines like IBM's Blue Gene/L and Cray's XT3. We also talk about a few biomolecular simulations and related research being conducted by scientists using NAMD.

9.2 NAMD Design

The design of NAMD rests on a few important pillars: a (then) novel strategy of hybrid decomposition, supported by dynamic load balancing, and adaptive overlap of communication with computation across modules, provided by the Charm++ runtime system [11].

9.2.1 Hybrid decomposition

The current version of NAMD is over ten years old. It has withstood the progress and changes in technology over these ten years very well, mainly because of its from-scratch, future-oriented, and migratable-object-based design. Prior to NAMD, most of the parallel MD programs for biomolecular simulations were extensions of (or based on) their preexisting serial versions [2, 21]. It was reasonable then to extend such programs by using a scheme such as atom decomposition (where atoms were partitioned based on their static atom numbers across processors). More advanced schemes were proposed [16, 8] that used force decomposition, where each processor was responsible for a square section of the $N \times N$ interaction matrix, where N is the total number of atoms.

In our early work on NAMD, we applied isoefficiency analysis [6] to show that such schemes were inherently unscalable: with an increasing number of processors, the proportion of communication cost to computation cost increases even if one were to solve a larger problem. For example, the communication-to-computation ratio for the force decomposition schemes of [16, 8] is of order $\sqrt{P}$, independent of N, where P is the number of processors. We showed that spatial decomposition overcomes this problem, but suffers from load balance issues.

At this point, it is useful to state the basic structure of a MD program: the forces required are those due to electrostatic and van der Waals interactions among all atoms, as well as forces due to bonds. A naïve implementation of the force calculation will lead to an $O(N^2)$ algorithm. Instead, for periodic systems, one uses an $O(N \log N)$ algorithm based on three-dimensional (3-D) fast Fourier transforms (FFTs) called the particle mesh Ewald (PME) method, in conjunction with explicit calculation of pairwise forces for atoms within a cutoff radius r_c. This suggests a spatial decomposition scheme in which atoms are partitioned into boxes of a size slightly larger than r_c. The extra margin is to allow atoms to be migrated among boxes only after multiple steps. It also facilitates storing each hydrogen atom on the same processor that owns its "mother" atom — recall that a hydrogen atom is bonded to only one other atom.

NAMD [10, 9] extends this idea of spatial decomposition, used in its early version in 1994, in two ways: first, it postulates a new category of objects called the *compute* objects. Each compute object is responsible for calculating interactions between a pair of cubical cells (actually brick-shaped cells, called *patches* in NAMD). This allows NAMD to take advantage of Newton's third law easily, and creates a large supply of work units (the compute objects) that an intelligent load balancer can assign to processors in a flexible manner. The Charm++ system, described in Chapter 20, is used for this purpose. As we will show later, it also helps to overlap communication and computation adaptively, even across multiple modules. As our 1998 paper [10] states, "the compute objects may be assigned to any processor, regardless of where the associated patches are assigned." The strategy is a hybrid between spatial and force decomposition. Recently, variations of this hybrid decomposition idea have been used by the programs Blue Matter [3] and Desmond [1] and a proposed scheme by M. Snir [19], and it has been called evocatively the "neutral territory method" [1]. Some of these methods are clever schemes that statically assign the computation of each pair of atoms to a specific processor, whereas NAMD uses a dynamic load-balancing strategy that should be superior due to its adaptive potential (see Section 9.2.2).

NAMD allows spatial decomposition of atoms into boxes smaller than the cutoff distance. In particular, it allows each dimension of a box to be 1/2 or

1/3 of the cutoff radius plus the margin mentioned above. This allows more parallelism to be created when needed. Note that when each dimension of the cell is halved, the number of patches increases eightfold. But since each patch now must interact with patches two-away from it in each dimension (to cover the cutoff distance), a set of $5 \times 5 \times 5$ compute objects must now access its atoms. Accounting for double counting of each compute and for self-compute objects, one gets a total of $8 \times 63/14$ more work units to balance across processors. Note further that these work units are highly variable in their computation load: those corresponding to pairs of patches that share a face are the heaviest (after self-computation objects) and those corresponding to patches that are two hops away along *each* dimension have the least load, because many of their atom-pairs are beyond the cutoff distance for explicit calculation. Early versions of NAMD, in 1998, restricted us to either use full-size patches, or 1/8th-size patches (or 1/27th-size patches, which were found to be inefficient). More recent versions have allowed a more flexible approach: along each dimension, one can use a different decomposition. For example, one can have a two-away X and Y scheme, where the patch size is halved along the X and Y dimensions but kept the same (i.e., $r_c + margin$) along the Z dimension.

9.2.2 Dynamic load balancing

NAMD uses measurement-based load-balancing capabilities provided by the Charm++ runtime [23]. The runtime measures the load of each compute object and each processor during a few instrumented iterations and then assigns objects to processors based on the collected information. After the first load-balancing step, many computes are migrated to under-loaded processors because the initial assignment of computes to processors is arbitrary and as a result suboptimal. The subsequent load-balancing decisions, which use a refinement-based strategy, tend to minimize the number of migrations. This serves to keep communication volume in check and does not break the runtime's assumption of predictability of load.

On machines such as Blue Gene/L, the load balancer also uses knowledge of the three-dimensional (3-D) torus interconnect to minimize the average number of hops traveled by all communicated bytes, thus minimizing contention in the network. While doing the initial mapping of cells to processors, the runtime uses a scheme similar to orthogonal recursive bisection (ORB) [13]. The 3-D torus of processors is divided recursively until each cell can be assigned a processor and then the 3-D simulation box of cells is mapped onto the torus. In subsequent load-balancing steps, the load balancer tries to place the computes on under-loaded processors near the cells, with which this compute will interact.

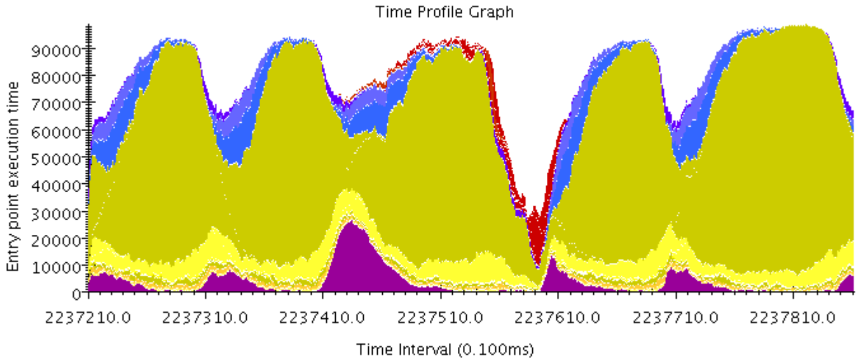


FIGURE 9.1: Time profile of NAMD running ApoA1 benchmark on 1024 processors of Blue Gene/L for five timesteps. Shades of gray show different types of calculations overlapping.

9.3 Petascale Challenges and Modifications

When NAMD was designed over ten years ago [14], million-processor machines were beyond the imagination of most people. Yet, by virtue of its parallel design, NAMD has demonstrated good scaling up to thousands of processors. As we moved to terascale machines (typically having tens of thousands of processors), NAMD faced a few challenges to maintain scalability and high efficiency.

The emergence of Blue Gene/L (which has only 256 MB of memory per processor) posed the problem of using a limited amount of memory for the initial startup (loading the molecular structure information), the actual computation, and load balancing. During startup, the molecular structure is read from a file on a single node and then replicated across all nodes. This is unnecessary and limits our simulations to about 100,000 atoms on Blue Gene/L. Making use of the fact that there are some common building blocks (amino acids, lipids, water) from which biomolecular simulations are assembled and their information need not be repeated, this scheme has been changed. Using a compression scheme, we can now run million atom simulations on the Blue Gene/L as we will see in Section 9.3.1.

The other major obstacle to scaling to large machines was the previous implementation of the particle mesh Ewald (PME) method. The PME method uses 3-D fast Fourier transforms (FFTs), which were implemented via a one-dimensional (1-D) decomposition. This limited the number of processors that could be used for this operation to a few hundred depending upon the number of planes in the grid. To overcome this limitation, a commonly used two

TABLE 9.1: Benchmarks and their simulation sizes

Molecular System	Atom Count	Cutoff (Å)	Simulation Cell (Å^3)	Time step (fs)
IAPP	5,570	12	$46.70 \times 40.28 \times 29.18$	2
ApoA1	92,224	12	$108.86 \times 108.86 \times 77.76$	1
STMV	1,066,628	12	$216.83 \times 216.83 \times 216.83$	1

dimensional (2-D) decomposition of the grid into pencils is now used. This has led to higher efficiency on large numbers of processors and has also helped to better overlap the FFT with other operations as can be seen in Figure 9.1.

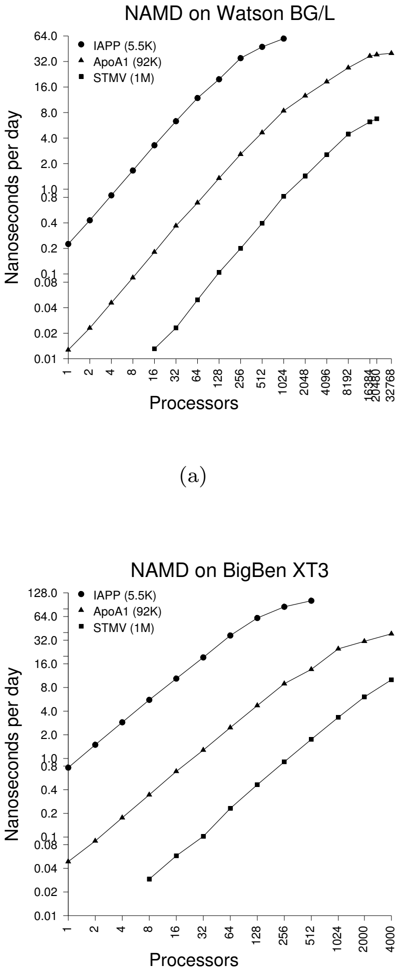
This figure has been generated using the performance analysis tool in the Charm++ framework called Projections. For each 100 μs time interval (along the X-axis), the figure shows the execution time of each function added across all 1024 processors. Various shades of light gray consuming most of the graph represent the compute work. The black peaks at the bottom represent patch integration and the deep gray bordering the hills represents communication. The area in black in the valley in the center represents the PME computation, which overlaps well with other functions.

9.3.1 Current performance

The performance of NAMD on various platforms substantiates the claims made in the previous sections. NAMD has shown excellent scaling to thousands of processors on large parallel machines like the Blue Gene/L and Cray XT3. The benchmarks used for results presented are shown in Table 9.1. These molecular systems are representative of almost all sizes of the simulations interesting to biophysicists. They range from a few thousand atoms to millions of atoms.

The Blue Gene/L (BG/L) machine at IBM T. J. Watson has 20,480 nodes. Each node contains two 700 MHz PowerPC 440 cores and has 512 MB of memory shared between the two. The machine can be operated in coprocessor mode or virtual node mode. In the coprocessor mode, we use only one processor on each node for computation. In the virtual node mode, both processors on each node are used for computation. The nodes on BG/L are connected in a 3-D torus. The Cray XT3 at the Pittsburgh Supercomputing Center (called BigBen) has 2068 compute nodes, each of which has two 2.6 GHz AMD Opteron processors. The two processors on a node share 2 GB of memory. The nodes are connected into a 3-D torus by a custom C-star interconnect.

Figures 9.2(a) and 9.2(b) show NAMD scaling to 32,768 processors of the Blue Gene/L machine and to 4,000 processors of Cray XT3. Different techniques are at work together as we run on machines with large numbers of processors. At some point on the plots, depending on the atoms per processors



(a)

(b)

FIGURE 9.2: Performance of NAMD on Blue Gene/L and Cray XT3.

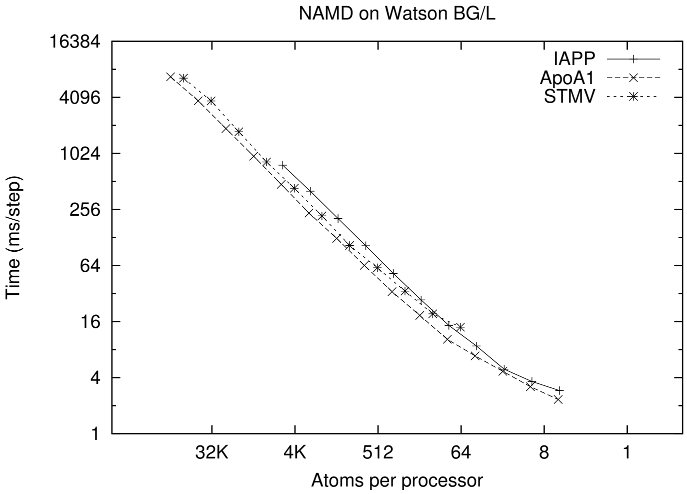


FIGURE 9.3: Time per step plotted against ratio of atoms to processors for NAMD running on BG/L.

and the machine, we switch from 1-Away to 2-AwayX and then 2-AwayXY decomposition of the patches. We also shift from 1-D decomposition of grids to 2-D for the PME computation when required. These decisions are automated so as to relieve the user of the burden of identifying optimal configuration parameters.

9.3.2 Performance on future petascale machines

To model running large molecular systems (many millions of atoms) on petascale machines with millions of processors, we plotted number of atoms per processor versus time step for different molecular systems. As can be seen in Figure 9.3 we get similar performance for a given ratio of number of atoms to processors for all the three benchmarks (which are quite varied in their sizes).

This plot suggests that NAMD will perform well for larger sized molecular systems on new petascale machines. For example, consider a 100-million atom molecular system which we wish to run on a hypothetical petascale machine consisting of 5-million processors. This gives us an atom-to-processor ratio of 20, which is within the regime presented in the above plot. The fraction of CPU cycles spent of FFT/PME increases as $N \log N$ as the number of atoms (N) increases, but this is a relatively small effect. We also validated these conclusions using our BigSim performance prediction framework [24] for some petascale designs.

9.3.3 Acceleration co-processors

There is currently excitement about the potential of heterogeneous clusters in which the bulk of computation is off-loaded to a specialized (or at least less-flexible) but higher performance coprocessor. Examples include the Cell Broadband Engine processor, graphics processors (GPUs), field-programmable gate arrays (FPGAs), and the special-purpose MD-GRAPE. Although these application accelerators are capable of sustaining hundreds of gigaflops for well-suited application kernels, data transfer between the CPU and the coprocessor limits performance to perhaps a factor of ten over a traditional CPU core. Since accelerated nodes are likely to outrun interconnect bandwidth, their impact will be seen most on throughput-oriented clusters, while leadership-class petascale machines will employ multicore processors for maximum code portability and a more balanced design. The parallel design of NAMD would be little-changed by the addition of acceleration coprocessors.

9.4 Biomolecular Applications

Scientifically interesting simulations of biomolecular systems currently range from ten thousand to a few million atoms, while future simulations may extend to 100,000,000 atoms. Progress will be on two fronts: supporting simulations of larger systems and increasing simulation rates on more powerful machines as they appear. For smaller systems the length of simulation that can be obtained is limited by latency and serial bottlenecks (strong scaling), while for larger simulations the size of the available machine limits performance (weak scaling). Table 9.2 summarizes expected simulation capabilities for NAMD running on the latest leadership-class hardware. Values for years 2004 and 2006 reflect capabilities already achieved. All simulations use a 12 Å cutoff, PME full electrostatics, and 1 femtosecond time steps. Achieving simulation rates higher than 100 nanoseconds/day would require significant improvement in network latencies, which is not easily foreseen in the next 5 years. Examples of biomolecular simulations in each size range are given in the table, illustrated in Figure 9.4, and described below.

9.4.1 Aquaporins

Water constitutes about 70% of the mass of most living organisms. Regulation of water flow across cell membranes is critical for maintaining fluid balance within the cell. The transportation of water in and out of a cell is mediated by a family of membrane proteins named aquaporins (AQPs), which are widely distributed in all domains of life. Through modulating water permeability of cellular membranes, AQPs play a crucial role in water

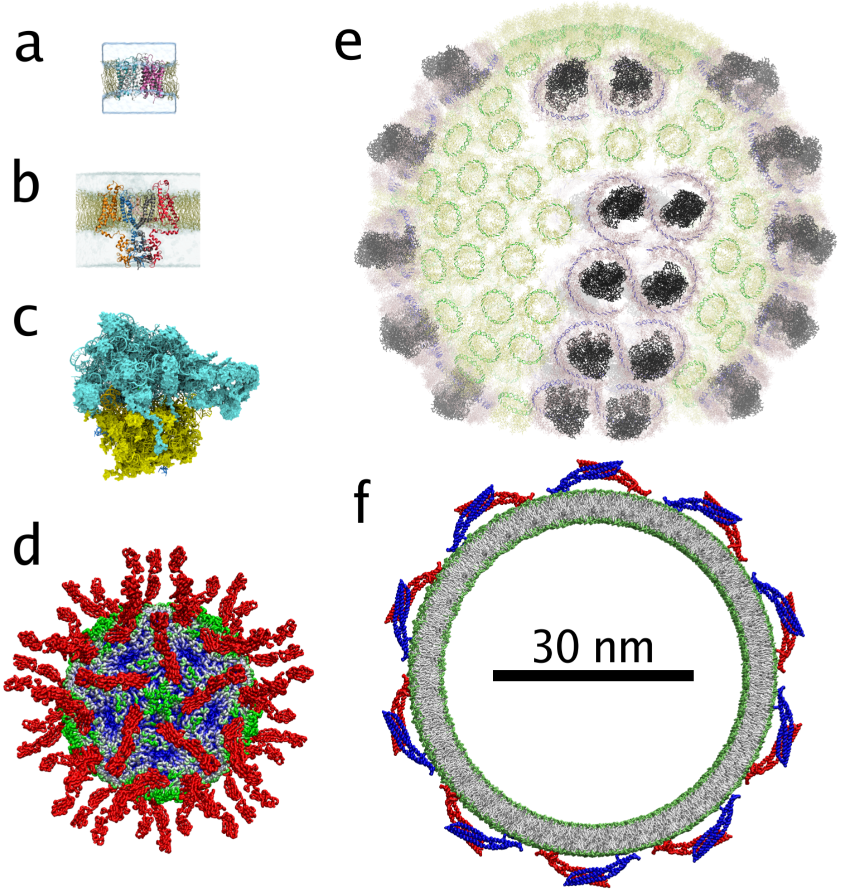


FIGURE 9.4: (See color insert following page 18.) Example biomolecular simulations: (a) aquaporin in membrane with solvent, (b) potassium channel in membrane with solvent, (c) ribosome, (d) poliovirus with cell surface receptors, (e) photosynthetic chromatophore, (f) BAR domain vesicle cross section.

homeostasis of living cells. In the human body, there are at least 11 different AQPs, whose physiological importance is reflected in the many pathophysiological situations associated with their absence/malfunction. For example, cataracts and diabetes insipidus have been linked to the impaired functions of AQP0 and AQP2, respectively. A particularly intriguing property of AQPs is their ability to block protons while allowing water to pass. In the past few

TABLE 9.2: Forecast of production simulation capabilities

Size	**100 K atoms**	**1 M atoms**	**10 M atoms**	**100 M atoms**
e.g.	aquaporin, potassium chan.	STM virus, ribosome	poliovirus	chromatophore, BAR dom. vesicle
2004	4 ns/day			
	500-core Alpha			
2006	10 ns/day	4 ns/day		
	500-core XT3	2000-core XT3		
2008	100 ns/day	10 ns/day	1 ns/day	
	5,000-core machine			
2010	100 ns/day	100 ns/day	10 ns/day	1 ns/day
		50,000-core machine		
2012	100 ns/day	100 ns/day	100 ns/day	10 ns/day
			500,000-core machine	

years, MD simulations [20] have contributed significantly to the understanding of this unique property of AQPs, and also of the molecular basis of their function and selectivity. A single solvated and membrane-embedded AQP tetramer simulation [22] comprises ∼100,000 atoms.

9.4.2 Potassium channels

Ions crossing potassium channels are responsible for the generation and spread of electrical signals in the nervous system. A number of high-resolution crystal structures of potassium channels have been resolved over the last decade, recognized in part by the 2003 Nobel prize, awarded to Dr. MacKinnon for his pioneering work on structure-function relationships of these channels. However, how potassium channels dynamically transit between open, inactive, or conductive states upon application of voltage remains highly debated. Molecular dynamics simulations seem to be an ideal tool to tackle this question, but relevant gating events occur on the millisecond timescale, while current MD simulations only reach nanosecond timescales. Moreover, all-atom descriptions are required to faithfully model the behavior of the channel, e.g., its ion selectivity. A system of 350,000 atoms containing one potassium channel, a patch of membrane, water, and ions is being simulated already at 100 ns per month on 512 Cray XT3 CPUs [12]. To study channel gating, however, a tenfold improvement in simulation speed (1 μs per month) is required.

9.4.3 Viruses

Satellite Tobacco Mosaic Virus (STMV), one of the smallest and simplest viruses known, consists of a ball-shaped RNA genome enclosed in an icosahedral capsid composed of 60 identical protein units; the complete particle is roughly 17 nm in diameter. Although small for a virus, the complete simulation of STMV immersed in a water box with ions contains about one million atoms. Based on the results of the first simulation of the complete STMV particle [5], researchers were able to propose a possible assembly pathway for the virus. Further efforts on this project focus on the disassembly of STMV, which is known to be mediated by a change in pH (this holds for many other viruses), although the exact mechanism is unclear.

The poliovirus is larger and more complex than STMV (the polio capsid is about 30 nm in diameter and composed of 240 protein units), and the disassembly is believed to be triggered by contact between the capsid and host cell membrane and receptor proteins. Structures of the poliovirus itself and of the virus in complex with the receptor are available, although structure of the receptor is known only at a low resolution of ∼10 Å. Researchers have already developed a homology model of the poliovirus receptors and started MD simulations of a single capsid-receptor bundle (∼250,000 atoms). The systems intended for further simulations, focusing on capsid disassembly, include a portion of the capsid in contact with a membrane (about 3 million atoms) and the complete poliovirus capsid (∼10 million atoms). Further, a portion of the capsid in contact with a membrane will be simulated (up to more than 3 million atoms), to elucidate the role of the cellular membrane in the opening of the capsid and release of the genome. Finally, a simulation of the complete poliovirus capsid might be necessary for investigation of how the disassembly proceeds over the surface of the whole virus, which would require building a system consisting of about 10 million atoms.

9.4.4 Ribosome

The translation of genetic information into protein sequences is essential for life. At the core of the translation process lies the ribosome, a 2.5–4.5 MDa ribonucleoprotein complex where protein synthesis takes place. The ribosome is not only interesting because of its fundamental role in the cell, it is also a major target for drug discovery and design. Many antibiotics in clinical use block protein synthesis in the bacterial ribosome. With the emergence of high-resolution structures of the ribosome complexed with antibiotics, it has become clear that chemically diverse antibiotics target only a few ribosomal sites [17]. Structure-based drug design targeting these specific sites is an attractive option for discovering new antibiotics [4]. The Sanbonmatsu team at Los Alamos National Laboratory performed ground-breaking large-scale (2,640,000 atoms) all-atom MD simulations of the entire ribosome beginning in 2003. These simulations, performed with NAMD, discovered a corridor of 20 universally conserved ribosomal RNA bases interacting with the tRNA during accommodation. The study was published in 2005 and also demonstrated the feasibility of simulating conformational changes in multimillion-atom molecular machines using NAMD [18].

9.4.5 Chromatophore

One of the most fundamental processes for life on Earth is the transformation of light energy into the synthesis of ATP. This transformation is achieved through different specialized organelles, one such organelle being the chromatophore of the purple bacterium *Rhodobacter sphaeroides*. Chromatophores are sheet-like or bulb-like indentations of the bacterial plasma membrane. The chromatophore contains six types of proteins: about twenty photosynthetic reaction centers, about twenty light-harvesting complexes 1 (LH1), about 150 light harvesting complexes 2 (LH2), about ten bc1 complexes, about five cytochrome c2s, and usually one ATP synthase. These proteins are all individually structurally known, though not all from the same species. The chromatophore with its 200 proteins carries out a cardinal function in the bacterium, the absorption of sunlight by about 4,000 chlorophylls (contained in LH1 and LH2 along with 1,300 carotenoids) and the transformation of its energy into the synthesis of adenosine-triphosphate (ATP) from adenosine-diphosphate (ADP). The entire chromatophore model, an archetypal example of systems studied in structural systems biology, consists of more than 200 proteins in a (90 nm)^3 system containing about 70 million atoms.

9.4.6 BAR domain vesicle

Proteins containing BAR domains play an important role in essential cellular processes (such as vesicle endocytosis at synaptic nerve terminals) by inducing or sensing membrane curvature. The U.S. National Science Foundation (NSF) solicitation *Leadership-Class System Acquisition—Creating a Petascale Computing Environment for Science and Engineering* provides the following model problem, involving protein BAR domains, for the proposed machine:

> A molecular dynamics (MD) simulation of curvature-inducing protein BAR domains binding to a charged phospholipid vesicle over 10 ns simulation time under periodic boundary conditions. The vesicle, 100 nm in diameter, should consist of a mixture of dioleoylphosphatidylcholine (DOPC) and dioleoylphosphatidylserine (DOPS) at a ratio of 2:1. The entire system should consist of 100,000 lipids and 1,000 BAR domains solvated in 30 million water molecules, with NaCl also included at a concentration of 0.15 M, for a total system size of 100 million atoms. All system components should be modeled using the CHARMM27 all-atom

> empirical force field. The target wall-clock time for completion of the model problem using the NAMD MD package with the velocity Verlet time-stepping algorithm, Langevin dynamics temperature coupling, Nose-Hoover Langevin piston pressure control, the particle-mesh Ewald algorithm with a tolerance of 1.0e-6 for calculation of electrostatics, a short-range (van der Waals) cut-off of 12 Angstroms, and a time step of 0.002 ps, with 64-bit floating point (or similar) arithmetic, is 25 hours. The positions, velocities, and forces of all the atoms should be saved to disk every 500 time steps.

The requirements for this simulation are similar to the requirements of the chromatophore simulation described above. In both cases, systems containing hundreds of proteins and millions of atoms need to be simulated to gain insights into biologically relevant processes. In order to accomplish such projects, NAMD must be ported to petascale parallel computers.

9.5 Summary

Each time step in a biomolecular simulation is small, yet we need many million of them to simulate a small interval of time in the life of a biomolecule. Therefore, one has to aggressively parallelize a small computation with high parallel efficiency. The NAMD design is based on the concept of Charm++ migratable objects and is fundamentally adequate to scale to petascale machines—this is indicated by the 1–2 milliseconds time per step achieved by NAMD for some benchmarks, with ratio of atoms to processor in a similar range that we expect to see on petascale machines. We have demonstrated scalability to machines with tens of thousands of processors on biomolecular simulations of scientific importance. Implementation strategies have been reworked to eliminate obstacles to petascale through memory footprint reduction and fine grained decomposition of the PME computation. All this has made the study of large molecules such as the ribosome and entire viruses possible today and will enable even larger and longer simulations on future machines.

9.6 Acknowledgments

Various students and staff of the Parallel Programming Laboratory and the Theoretical and Computational Biophysics Group at the University of Illinois have been of great assistance in the development of NAMD. Sameer Kumar, at IBM, has helped significantly in improving NAMD performance on Blue Gene/L. The images of Figure 9.4 were created with the program VMD [7] by Anton Arkhipov, Jen Hsin, Fatemeh Khalili, Leonardo Trabuco, Yi Wang, and Ying Yin. Access to the Blue Gene/L machine at the IBM T. J. Watson Research Center was provided by Glenn J. Martyna and Fred Mintzer. Access to the Cray XT3 was provided by the Pittsburgh Supercomputing Center (PSC) via Large Resources Allocation Committee grant MCA93S028. We are grateful to Shawn T. Brown for helping us with the runs at PSC. This work was supported by the National Institutes of Health under grant P41-RR05969.

References

[1] K. J. Bowers, E. Chow, H. Xu, R. O. Dror, M. P. Eastwood, B. A. Gregersen, J. L. Klepeis, I. Kolossvary, M. A. Moraes, F. D. Sacerdoti, J. K. Salmon, Y. Shan, and D. E. Shaw. Scalable algorithms for molecular dynamics simulations on commodity clusters. In *SC '06: Proceedings of the 2006 ACM/IEEE Conference on Supercomputing*, New York, 2006. ACM Press.

[2] B. R. Brooks, R. E. Bruccoleri, B. D. Olafson, D. J. States, S. Swaminathan, and M. Karplus. CHARMM: A program for macromolecular energy, minimization, and dynamics calculations. *Journal of Computational Chemistry*, 4(2):187–217, 1983.

[3] B. Fitch, R. Germain, M. Mendell, J. Pitera, M. Pitman, A. Rayshubskiy, Y. Sham, F. Suits, W. Swope, T. Ward, Y. Zhestkov, and R. Zhou. Blue Matter, an application framework for molecular simulation on Blue Gene. *Journal of Parallel and Distributed Computing*, 63:759–773, 2003.

[4] F. Franceschi and E. M. Duffy. Structure-based drug design meets the ribosome. *Biochem. Pharmacol.*, 71:1016–1025, 2006.

[5] P. L. Freddolino, A. S. Arkhipov, S. B. Larson, A. McPherson, and K. Schulten. Molecular dynamics simulations of the complete satellite tobacco mosaic virus. *Structure*, 14:437–449, 2006.

[6] A. Grama, A. Gupta, and V. Kumar. Isoefficiency: Measuring the scalability of parallel algorithms and architectures. *IEEE Parallel & Distributed Technology*, 1(August), 1993.

[7] W. F. Humphrey, A. Dalke, and K. Schulten. VMD – Visual molecular dynamics. *Journal of Molecular Graphics*, 14(1):33–38, 1996.

[8] Y.-S. Hwang, R. Das, J. Saltz, M. Hodoscek, and B. Brooks. Parallelizing molecular dynamics programs for distributed memory machines. *IEEE Computational Science & Engineering*, 2(Summer):18–29, 1995.

[9] L. Kalé, R. Skeel, M. Bhandarkar, R. Brunner, A. Gursoy, N. Krawetz, J. Phillips, A. Shinozaki, K. Varadarajan, and K. Schulten. NAMD2: Greater scalability for parallel molecular dynamics. *Journal of Computational Physics*, 151:283–312, 1999.

[10] L. V. Kalé, M. Bhandarkar, and R. Brunner. Load balancing in parallel molecular dynamics. In *Fifth International Symposium on Solving Irregularly Structured Problems in Parallel*, volume 1457 of *Lecture Notes in Computer Science*, pages 251–261, 1998.

[11] L. V. Kale and S. Krishnan. Charm++: Parallel programming with message-driven objects. In G. V. Wilson and P. Lu, editors, *Parallel Programming Using C++*, pages 175–213. MIT Press, 1996.

[12] F. Khalili-Araghi, E. Tajkhorshid, and K. Schulten. Dynamics of K^+ ion conduction through Kv1.2. *Biophys. J.*, 91:L72–L74, 2006.

[13] S. Kumar, C. Huang, G. Zheng, E. Bohm, A. Bhatele, J. C. Phillips, H. Yu, and L. V. Kalé. Scalable molecular dynamics with NAMD on Blue Gene/L. *IBM Journal of Research and Development: Applications of Massively Parallel Systems*, 2007. (to appear).

[14] M. Nelson, W. Humphrey, A. Gursoy, A. Dalke, L. Kale, R. D. Skeel, and K. Schulten. NAMD—a parallel, object-oriented molecular dynamics program. *Intl. J. Supercomput. Applics. High Performance Computing*, 10(Winter):251–268, 1996.

[15] J. C. Phillips, R. Braun, W. Wang, J. Gumbart, E. Tajkhorshid, E. Villa, C. Chipot, R. D. Skeel, L. Kalé, and K. Schulten. Scalable molecular dynamics with NAMD. *Journal of Computational Chemistry*, 26(16):1781–1802, 2005.

[16] S. J. Plimpton and B. A. Hendrickson. A new parallel method for molecular-dynamics simulation of macromolecular systems. *J Comp Chem*, 17:326–337, 1996.

[17] J. Poehlsgaard and S. Douthwaite. The bacterial ribosome as a target for antibiotics. *Nat. Rev. Microbiol.*, 3:870–881, 2005.

[18] K. Y. Sanbonmatsu, S. Joseph, and C. S. Tung. Simulating movement of tRNA into the ribosome during decoding. *Proc. Natl. Acad. Sci. USA*, 102:15854–15859, 2005.

[19] M. Snir. A note on N-body computations with cutoffs. *Theory of Computing Systems*, 37:295–318, 2004.

[20] S. Törnroth-Horsefield, Y. Wang, K. Hedfalk, U. Johanson, M. Karlsson, E. Tajkhorshid, R. Neutze, and P. Kjellbom. Structural mechanism of plant aquaporin gating. *Nature*, 439:688–694, 2006.

[21] P. K. Weiner and P. A. Kollman. AMBER: Assisted model building with energy refinement a general program for modeling molecules and their interactions. *Journal of Computational Chemistry*, 2:287, 1981.

[22] J. Yu, A. J. Yool, K. Schulten, and E. Tajkhorshid. Mechanism of gating and ion conductivity of a possible tetrameric pore in aquaporin-1. *Structure*, 14:1411–1423, 2006.

[23] G. Zheng. *Achieving High Performance on Extremely Large Parallel Machines: Performance Prediction and Load Balancing.* PhD thesis, Department of Computer Science, University of Illinois at Urbana-Champaign, 2005.

[24] G. Zheng, G. Kakulapati, and L. V. Kalé. BigSim: A parallel simulator for performance prediction of extremely large parallel machines. In *18th International Parallel and Distributed Processing Symposium (IPDPS)*, page 78, Santa Fe, NM, April 2004.