LAWRENCE
LIVERMORE
NATIONAL
LABORATORY

LLNL-BOOK-608433

Scalable Molecular Dynamics with NAMD

J. C. Phillips, K. Schulten, A. Bhatele, C. Mei, Y. Sun, L. V. Kale

December 17, 2012

Parallel Science and Engineering Applications: The Charm++ Approach

Disclaimer

Chapter 4

Scalable Molecular Dynamics with NAMD

James C. Phillips, Klaus Schulten

Beckman Institute, University of Illinois at Urbana-Champaign

Abhinav Bhatele

Center for Applied Scientific Computing, Lawrence Livermore National Laboratory

Chao Mei

Intel Corporation

Yanhua Sun, Eric Bohm, Laxmikant V. Kale

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

4.1 Introduction

The NAMD software, used by tens of thousands of scientists, is focused on the simulation of the molecular dynamics of biological systems, with the primary thrust on all-atoms simulation methods using empirical force fields, and with a *femtosecond* time step resolution. Since biological systems of interest are of fixed size, efficient simulation of long time scales requires the

application of fine grained parallelization techniques so that systems of interest can be simulated in reasonable time. This need to improve the time to solution for the simulation of fixed sized systems drives the emphasis on "strong scaling" performance optimization that engendered this collaboration between physical and computer scientists.

Performance improvements motivated by NAMD inspire abstractions, optimized implementations, and robust infrastructure in CHARM++, and complementary improvements to CHARM++ enable the implementation of new features in NAMD. The collaborative and synergistic development underlying the NAMD project (started by principle investigators, Klaus Schulten, Laxmikant V. Kale, and Robert Skeel, in 1992) has contributed to many important achievements in molecular modeling, parallel computing, and numerical algorithms. As recognized in the 2012 IEEE Computer Society Sidney Fernbach Award, jointly awarded to Kale and Schulten, NAMD has been an important contribution to the scientific community.

In this chapter, we will discuss the motivation for biomolecular simulation (§4.2), parallelization techniques for molecular dynamics (§4.3), the parallel design of NAMD (§4.4), its application to ever larger scale simulations (§4.5), overall performance (§4.6), and elaborate upon a few of NAMD's applications (§4.7).

4.2 Need for Biomolecular Simulations

The form and function of all living things originate at the molecular level. Genetic information in nucleic acids encodes the sequence of amino acids for proteins, which once assembled by the ribosome, fold into the specific three-dimensional structures that enable their function in the cell.

Cellular proteins can be isolated, purified, and grown into crystals, from which X-ray diffraction can be used to determine the positions of the protein atoms with great accuracy. Larger aggregates, such as the ribosome, can be studied through cryo-electron microscopy, and the resulting coarse images combined with high-resolution crystal structures to obtain atomic resolution for the complete aggregate. While these experimentally determined structures alone are of great utility in explaining and suggesting mechanisms for the observed chemical and mechanical behavior of biomolecular aggregates, they represent only static and average structures. The detailed atomic motions that lead to function cannot be observed experimentally.

Physics-based simulations step in where experiment leaves off, allowing the study of biomolecular function in full atomic detail. Although atomic interactions are governed by quantum mechanics, the energies found in biological systems are sufficiently low that chemical bonds are only formed or broken in the reaction centers of catalytic proteins. As a result, atomic interactions in

biomolecules can be represented via simple classical potentials for electrostatics, van der Waals, and bonded interactions. While this simplification greatly reduces the computational demands of simulations, many orders of magnitude are required to extend size from atoms to cells and time from femtoseconds to seconds.

Current state-of-the-art simulations may follow millions of atoms for mere microseconds, often employing additional techniques to enhance sampling. Smaller, longer-time simulations may follow the entire protein folding process. Larger simulations allow the study of aggregates such as the ribosome, which builds proteins in all cells and is a common target of antibiotics, the chromatophore, which is the basic photosynthetic unit of plants, and the protein capsids of viruses, which bind to and penetrate the cell membrane to enable the infection process. A recent example is the ground-breaking study, published in *Nature* [256] that determined the structure of the HIV capsid, based on a NAMD simulation with 64 million atoms.

4.3 Parallel Molecular Dynamics

Molecular dynamics (MD) simulations follow molecular systems ranging from a few thousand to millions of atoms for tens of nanoseconds to microseconds. When doing these simulations sequentially, the time period to be simulated is broken down into a large number of time steps of 1 or 2 femtoseconds each. At each time step, forces on each atom (electrostatic, van der Waals and bonded) due to all other atoms are calculated and the new positions and velocities are determined. The atoms are moved to their new positions and the process repeats.

Parallelizing a MD simulation is challenging because of the relatively small number of atoms and large number of time steps involved. Traditionally, three different methods have been used to parallelize MD simulations: atom decomposition, spatial decomposition and force decomposition [205, 206]. Atom decomposition involves distributing the atoms in the MD simulation among the processors evenly. Each processor is responsible for the force calculations for its atoms. Spatial decomposition is similar except that the physical simulation space is divided up spatially to assign atoms to different processors. Force decomposition, on the other hand, involves creating a matrix of force calculations to be performed for pairs of atoms and assigning responsibility for the calculation of a part of the matrix to each processor.

Atom and force decomposition have a high communication-to-computation ratio asymptotically whereas spatial decomposition suffers from load imbalance problems. NAMD pioneered the hybrid decomposition scheme in which the processors holding the atoms and those calculating the forces are decoupled [128]. The parallelization scheme is a hybrid between spatial and force

decomposition. Atoms in the simulation box are divided spatially into smaller boxes and assigned to some processors. The force calculations for a pair of sub-divided boxes are assigned to an arbitrary processor which can be different from the ones holding the two boxes. The scheme is described in detail in the next section. Similar parallelization schemes have been used in other recent MD packages such as Desmond [31] and Blue Matter [70], and in the scheme proposed by Snir [229].

4.4 NAMD's Parallel Design

NAMD is one of the first scientific applications to use Charm++ as the underlying runtime system. Over the last decade, NAMD development has fueled Charm++ research and instigated new features and capabilities in the runtime. In turn, NAMD has benefited from features such as dynamic load balancing and section multicasts that are a part of Charm++.

NAMD has co-developed with Charm++, and served as a confirmation of the utility of some of the features of Charm++, such as message-driven execution and load balancing. An early version of NAMD, which was a precursor to the current version, was written in the mid 1990s. Two different versions were maintained for some time: one was in PVM, and the other in Charm, the C-based precursor to Charm++. The modularity benefits of Charm++ started becoming clear in comparing these variants, especially as they were developed further. The PVM version needed to have a message-driven loop explicitly in its code. Messages belonging to different modules were dealt with in this one loop, and it had to keep track of the progress of different modules, and even different instances of them (such as multiple sub-domains, computation objects, and long-range force calculation objects). In contrast, in the Charm++ version, the objects belonging to different modules were cleanly separated. They naturally allowed adaptive overlap of communication with the computation across modules, yet required no breaching of abstraction boundaries.

At the same time, the separation of objects from processors, which was the hallmark of the original NAMD design, was very naturally supported by Charm++'s object model. The separation of the collection of objects for doing force calculations from the objects housing the atoms also allowed us to (and required us to) do explicit load balancing. With such load balancing, Charm++ was able to exploit an arbitrary number of processors within a reasonable range. In other words, there were no restrictions on the number of processors having to be a cube or even a product of three integers, as was typically required in molecular dynamics applications based on spatial decomposition.

4.4.1 Force Calculations

Molecular dynamics simulations involve calculating forces on each atom (electrostatic, van der Waals and bonded) due to all other atoms. A naïve pairwise calculation of non-bonded forces between all pairs of atoms has a time complexity of $\mathcal{O}(N^2)$. In order to reduce this complexity to $\mathcal{O}(N \log N)$, forces are calculated explicitly only within a cutoff radius, r_c. Beyond this distance the forces are calculated by extrapolating the charge densities of all atoms to a charge grid and using the particle-mesh Ewald (PME) [46] method.

NAMD uses a hybrid decomposition scheme that separates the distribution of data (atoms) from the distribution of work (force calculations). The implementation of the hybrid decomposition scheme and independent calculation of different types of forces is facilitated by the ability to create multiple sets of chares in CHARM++ that can be mapped independently to the processors. The simulation box is divided spatially into smaller boxes called "patches" which collectively form one set of chares (see Figure 4.1 which shows a simplified two-dimensional simulation space). The number of patches can be less than the number of processors in which case, the patches are assigned to a subset of the processors. Force calculations between a pair of patches are assigned to chares from another set called the compute objects, or just "computes". There are three different types of computes – 1. bonded computes that calculate the forces due to bonds, 2. non-bonded computes, that are responsible for calculating short-range non-bonded forces and 3. PME computes, responsible for calculating long-range electrostatic forces.

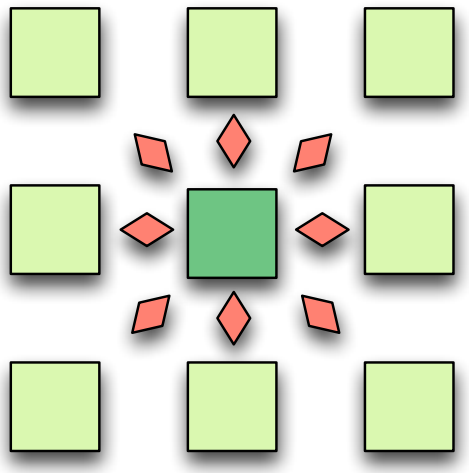

FIGURE 4.1: Hybrid decomposition in NAMD (the square objects are patches and the diamond objects are non-bonded computes)

Each non-bonded compute is responsible for the force calculations between

a pair of patches (or a single patch in case of interactions between atoms within a patch). Hence, each patch sends its atoms to several computes (nine in case of the 2D decomposition shown in Figure 4.1, twenty-seven in case of a 3D decomposition) whereas each compute receives atoms from two patches. The sending of atoms from a patch to all its computes is done via a section multicast that creates a "spanning" tree between the processor holding the patch (root) and the processors holding the associated computes. Forces from all computes to a given patch are also sent back along this tree.

Three different chare arrays are used for the PME computation which uses a two-dimensional pencil decomposition of the charge grid for parallelization: PMEZPencils, PMEYPencils and PMEXPencils. The patches communicate with the PMEZPencils at the beginning and completion of each PME phase. There are several line FFTs within the pencils in each direction and transpose operations between pencils in different directions. Since this phase has a relatively small amount of computation and intensive communication (due to the transposes), it is often done every four time steps instead of every time step.

4.4.2 Load Balancing

The presence of different kinds of chares – patches, bonded computes, non-bonded computes and three types of PME computes makes load balancing a formidable task. However, the load balancing framework in CHARM++ is designed to handle multiple chare arrays in the application. The load balancing framework is measurement-based and relies on the *principle of persistence* of load. This principle assumes that the load distribution in the recent past is a reasonable indicator of that in the near future. The runtime instruments all the chares in the application for their execution times and also records the communication graph between them. This graph is made available to the load balancing framework to make migration decisions. Applications can plug in specific strategies that exploit application-specific knowledge for a better load balance.

In the case of NAMD, all chares – patches, bonded computes, non-bonded computes and PME computes are instrumented for their execution time. The total load on each processor is the sum of the execution times of all objects that reside on it. The loads on each processor in the previous time steps are used to make decisions about migrating the chares for better balance. Only the non-bonded computes, which account for a significant fraction of the execution time, are made migratable. The rest of the computes are assigned statically during program start-up but their loads are considered when balancing the migratable objects.

Load balancing in NAMD is periodic. Before the load balancer is invoked, a few time steps are instrumented and that information is used for balancing load for future time steps. The first time that load balancing is performed, the algorithm reassigns all migratable objects. Subsequent calls perform a refinement-based load balancing that minimizes migrations by preserving the

previous assignments as much as possible. The load balancing strategy is a greedy heuristic-based algorithm that creates a max heap of objects and min heap of processors (based on their execution times) and maps objects iteratively starting with the heaviest ones to the least loaded processors.

The CHARM++ runtime provides a detailed communication graph of the chares involved in an application to the load balancer. The balancing algorithm can use the communication information to minimize communication as well as migration. This information is also used for optimizing the communication on the underlying network topology in the case of torus machines such as the IBM Blue Gene platforms [25]. An interconnect topology aware mapping of the patches and computes in NAMD can optimize communication on the network and minimize network congestion.

4.5 Enabling Large Simulations

The unprecedented growth in the size of parallel machines and the requirements of computational scientists to simulate molecular systems with tens to hundreds of millions of atoms have put the scaling performance of NAMD to test and resulted in significant improvements to the software to enable such use cases.

4.5.1 Hierarchical Load Balancing

Traditionally (before 2010), the load balancing strategies in NAMD were executed serially by collecting the instrumented data (loads and communication graph) on one processor. This becomes infeasible when running a large molecular system or on a large number of processors or with a large number of chares. Collecting the entire communicating graph on one processor and then sending migration decisions out from it leads to a serialization bottleneck in messaging. Storing this information in the memory of one node also becomes infeasible. Finally, the serial load balancing algorithm running on one processor can take a very long time to execute while all other processors are idle, waiting for the decisions. These factors motivated the use of a hierarchical load balancing scheme in NAMD [258].

NAMD uses the hierarchical load balancing support available in CHARM++ [257]. In this scheme, the processors are divided into independent groups that are arranged in a hierarchy forming a tree. The tree can have any number of levels and an arbitrary number of children per node. Every node and its immediate children at the next level form an autonomous group. Within each group, a root node or group leader performs load balancing serially for all processors within its group. At higher levels, group leaders represent the entire sub-tree below them. Load information is first exchanged bottom up and then

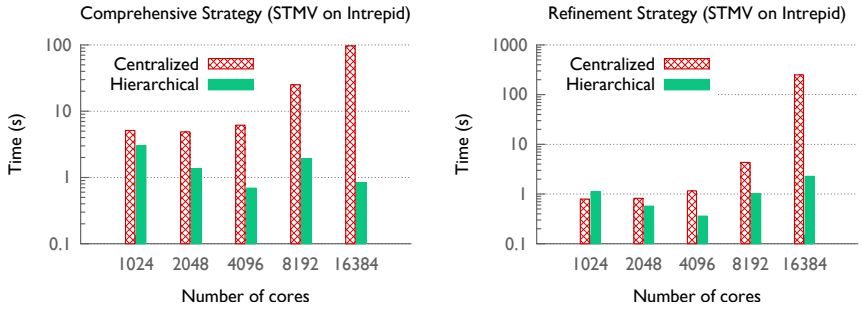

FIGURE 4.2: Improvements in load balancing time from using hierarchical load balancing in NAMD on IBM Blue Gene/P (Intrepid)

load balancing is done in a top down fashion. Within each group, existing load balancing strategies in NAMD such as comprehensive and refinement can be invoked.

Use of hierarchical load balancers leads to significant reductions in the memory consumption in NAMD and more importantly huge savings in the execution time of the load balancing strategies. Figure 4.2 shows the reduction in the time spent in the comprehensive and refinement load balancing strategies simulating the Satellite Tobacco Mosaic Virus (STMV). On 16,384 cores of IBM Blue Gene/P, the time spent in load balancing is reduced by more than 100 times! This improvement is attained while retaining a high quality of load balance achieved, so application performance is almost as good as that with centralized load balancers.

4.5.2 SMP Optimizations

Multicore nodes in parallel machines have motivated the design and implementation of a multi-threaded SMP runtime mode in Charm++ [173]. In this mode, each Charm++ processing element (PE) runs as a thread as opposed to an OS process in the non-SMP runtime mode. All threads (i.e. Charm++ PEs) belonging to the same OS process form a Charm++ "node". The nature of a single memory address space shared by Charm++ PEs on a "node" enables several optimization opportunities.

Reduce Memory Footprint: Read-only data structures or immutable ones (only written once) can be shared among Charm++ PEs on a "node". Exploiting this opportunity can also lead to other benefits such as better cache performance. In NAMD, optimizations are done to share certain information such as the molecule object that contains static physical attributes of atoms and map objects that track the distribution of patch and compute objects. Table 4.1 shows the comparison of average memory usage per core when running

NAMD in non-SMP and SMP modes on the Jaguar machine at Oak Ridge National Laboratory, demonstrating the effectiveness of reducing the memory consumption using SMP mode. In addition, we also observed much better cache performance directly related with the memory footprint reduction [172].

No. of nodes	140	560	2240	4480	8960	17920
No. of cores	1680	6720	26880	53760	107520	215040
non-SMP (MB)	838.09	698.33	798.14	987.37	1331.84	1760.86
SMP (MB)	280.57	141.83	122.41	126.03	131.84	157.76
Reduction factor	2.99	4.92	6.52	7.83	10.10	11.16

TABLE 4.1: Comparison of average memory footprint between SMP and non-SMP during simulation (12 cores per node)

Improve Intra-node Communication: CHARM++ PEs on the same "node" can also exploit the use of shared memory address space to improve the performance of communication. Instead of making a copy of the message when performing intra-node communication in the non-SMP mode, the CHARM++ runtime simply transfers the memory pointer of the message in the SMP mode. This optimization is transparent to the application and is embedded in the runtime. Therefore, NAMD automatically enjoys the benefits from the improved intra-node communication. In addition, considering faster communication within a "node", if a message is sent to multiple PEs on the same "node" from a PE on a remote "node", we can optimize this communication scenario by just sending one inter-node message and then forwarding this message to destination PEs within a node. In this way, the expensive inter-node communication is replaced with the more efficient intra-node one. We refer to this as node-aware communication optimization, and it is exploited as much as possible in communication idioms such as the general multicast/broadcast in the CHARM++ runtime and NAMD-specific multicast operations [172].

Exploit More Fine-grained Parallelism: Several computation functions in the NAMD PME phase have been observed to execute on a few cores on each node with idle neighboring cores. To improve NAMD's performance during this phase, the fine-grained parallelism inherent in those computation functions needs to be exploited and distributed among the idle neighboring cores. OpenMP provides a language directive based approach to realizing this. Using OpenMP threads in conjunction with CHARM++ PEs on the same cores is not straighforward because they are not aware of each other. We have developed a "CkLoop" library for the SMP mode of CHARM++ to use the CHARM++ PEs to mimic the responsibilities of OpenMP threads. [237] shows the performance benefits from using this "CkLoop" library for the PME computation.

Reduce Usage of System Resources: In the SMP node, significantly fewer OS processes are created in the parallel application. This implies that the

usage of system resources which depends on the number of processes also reduces. The benefit of such reduction is exemplified by the decrease in the job launch time. When running NAMD on 224,076 cores of Jaguar where CHARM++ is built to rely on MPI to perform communication, mpirun takes about 6 minutes to launch the job in the non-SMP mode where each core is hosting a MPI rank. In comparison, in the SMP mode, each node is hosting a MPI rank instead, covering 12 cores. As a result, mpirun then only takes about 1 minute to launch the NAMD job.

4.5.3 Optimizing Fine-grained Communication in NAMD

As described earlier, most of the time in the PME phase in NAMD is spent in communication. When scaling NAMD to large number of nodes this communication in PME scales poorly and eventually becomes the major performance bottleneck. Therefore, it is crucial to optimize this communication pattern. Below, a few techniques for optimizing the PME communication are discussed.

Increasing PME message priority. In NAMD, various types of messages play different roles in driving the program execution. For example, the non-bonded function in the compute object is performed when the messages containing atoms from its two patches arrive. Similarly, the arrival of PME messages drives the FFT computation. When different types of messages are queued to be sent or to be executed, the selection of messages to process first can potentially affect the overall performance. When the PME phase becomes the the performance bottleneck, it is highly desirable to process the PME messages as soon as possible. In order to do this, we assign PME messages with higher priority than other messages. Two techniques are applied to implement this idea. On the sender side, messages with high priority are processed first. Only after these messages are injected into the network, the other messages get a chance to be processed. On the receiver side, instead of processing messages in a first-come-first-serve (FCFS) order, incoming messages are queued in the order of priority. Therefore, the computation driven by messages with high priority is performed first. With these two techniques, the delay of processing PME messages is minimized which improves the overall performance and scaling significantly.

Persistent communication for FFT. For most applications in CHARM++, when messages are sent on the sender, the memory to store the message on the destination is usually unknown. Only when the message arrives at the receiver, the corresponding memory is allocated for it. However, in scientific applications, we have observed that there is 'persistent' communication, which means that the communication partners and message sizes for one transaction do not change across time steps. There are two possible benefits of exploiting this persistent communication. First, we can save the time to allocate/free memory on the receiver. The other benefit is to better exploit the underlying

network if it supports remote direct memory access (RDMA). Without using persistent communication, three steps are required to send a message on the RDMA network. First, a small control message including sender's information is sent to the receiver. Based on the information in this small message, the receiver performs a RDMA 'get' transaction to transfer the real data. An ack message is sent back to the sender to notify the completion of the data transfer. Compared with this, using persistent communication, the sender has the information of receiver so that a direct RDMA 'put' can be issued to transfer the data. Hence, the small control message is avoided. In NAMD, we implemented the communication in PME with persistent messages. A 10% performance improvement is observed when running a 100-million-atom simulation on the Titan machine at ORNL.

4.5.4 Parallel Input/Output

As we started to simulate very large molecular systems with tens of millions of atoms using NAMD on hundreds of thousands of processors, we found that the input/output (I/O) in NAMD, i.e., loading molecular data at start-up and outputting atoms' trajectory data to the file system, became a major roadblock. Since existing parallel I/O libraries such as HDF, netCDF etc. do not handle NAMD file formats, we chose to implement parallel I/O natively in NAMD. One main advantage, enabled by the asynchronous message-driven programming model of Charm++, is that we can then optimize for writing trajectory data frame-by-frame, overlapping with the computation on other processors during the simulation.

Traditionally, NAMD loads and processes all molecular data on a single core before broadcasting the data to all other cores. Although this approach is adequate for moderately large molecular systems, it does not scale to molecular systems with several million atoms due to the inherent sequential execution. For example, it requires nearly an hour and about 40 GB of memory to initialize a 100-million-atom STMV simulation on a single core of an Intel Xeon (1.87 GHz) processor. To address this issue, we first developed a compression scheme by extracting "signatures" of atoms from the input data to represent the common characteristics that are shared by a set of atoms. Together with atoms' "signature" input, a binary file containing the information of each atom, constructed from the original input file, is fed into the native parallel input scheme described as follows.

A small number of processors are designated as input and output processors. Considering P "input" processors, one of them first reads the signature file and then broadcasts this to all other input processors. P is usually smaller than the total number of processors and can be automatically tuned to optimize for the memory footprint and performance. After the initial broadcast, each of these P processors loads $1/P$ of the total atoms starting from independent positions in the binary file. Then they shuffle atoms with neighbor input processors according to molecular grouping attributes for later spatial decom-

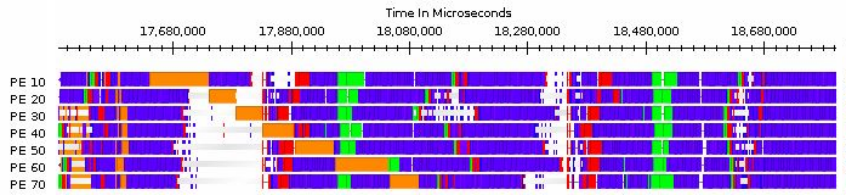

FIGURE 4.3: Parallel Output Timeline of a 2.8-million-atom Ribosome Simulation on Jaguar (Cray XT5)

position. Comparing with the sequential input scheme, for the 100-million-atom STMV simulation, this parallel scheme with 600 input processors on Jaguar (Cray XT5 at ORNL) completes the initialization in 12.37 seconds with an average memory consumption of 0.19 GB on each input processor, a ∼300× reduction in time and a ∼200× reduction in memory footprint!

We faced similar performance and memory footprint challenges in the output of trajectory files, but with an additional one posed by maintaining fast execution time per step under tens of milliseconds in case of frequent output. Similar to the parallel input scheme, with a tunable number of "output" processors, each output processor is responsible for the trajectory output of a subset of atoms. Furthermore, we have implemented a flexible token-based output scheme in which only those output processors that have a token could write to the file system in order to handle I/O contention on different parallel file systems. Reaping benefits from Charm++, the file output on one processor can potentially overlap with useful computation on other processors as clearly illustrated by Figure 4.3 showing the tracing of a single-token-based output scheme. In the figure, the output activity represented by the light gray bars clearly overlaps with useful computation (dark gray) on other cores spanning multiple time steps.

4.6 Scaling Performance

NAMD is run on a variety of supercomputer platforms at national supercomputing centers in the U.S. and elsewhere. It has demonstrated good strong and weak scalability for several benchmarks on different platforms. The platforms vary from small memory and low frequency processors like the IBM Blue Gene machines to fast processors like the Cray XT5 and XK6. The size of molecular systems ranges from benchmarks as small as IAPP with 5570 atoms to an STMV system with 100 million atoms. Table 4.2 lists the various molecular systems (and their simulation details) that were used for obtaining the performance numbers presented here.

System	No. of atoms	r_c (Å)	Simulation box	Time step (fs)
ApoA1	92224	12	108.86 × 108.86 × 77.76	1
F1-ATPase	327506	12	178.30 × 131.54 × 132.36	1
STMV	1066628	12	216.83 × 216.83 × 216.83	1
Ribosome	2820530	12	264.02 × 332.36 × 309.04	1
100M STMV	106662800	12	1084 × 1084 × 867	1

TABLE 4.2: Simulation parameters for molecular systems used for benchmarking NAMD

Performance for various molecular systems: Figure 4.4 shows the execution time per step for five molecular systems running on an IBM Blue Gene/P (BG/P). ApoA1 and F1-ATPase scale well up to 8,192 cores while the bigger systems of STMV and Ribosome scale up to 16,384 cores. The 100 million atom system has demonstrated scalability up to almost the entire machine at Argonne (Intrepid, 163,840 cores). The simulation rate for ApoA1 at 16,384 cores of BG/P is 47 nanoseconds per day (ns/day) or 1.84 ms per time step. The simulation rate for the 100M STMV system at 131,072 cores is 0.78 ns/day or 111.1 ms per time step.

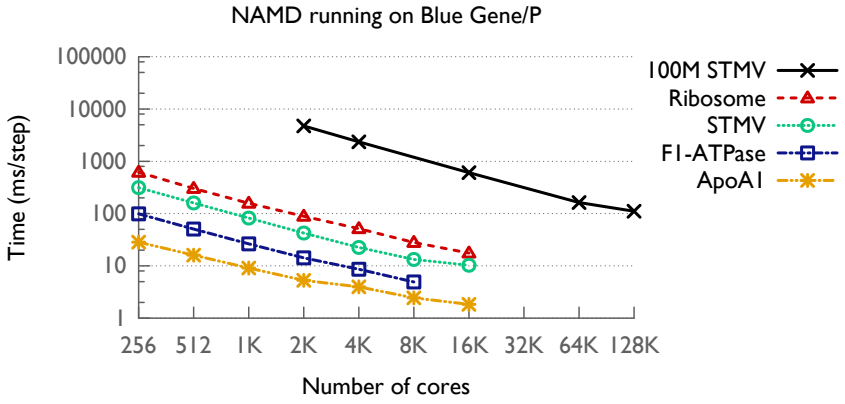

FIGURE 4.4: Performance of NAMD on IBM Blue Gene/P (Intrepid)

Performance on various machines : Figure 4.5 presents the execution times for the 1 and 100 million atom STMV systems on different machines: Blue Gene/P, Ranger and Cray XK6. The 1 million system scales well on the different platforms (left plot). The best performance is achieved on the Cray XK6 with much faster cores and a high-speed interconnect compared to the BG/P. At 16,384 cores, the execution time is 2.5 ms per step (simulation rate of 34.6 ns/day). The benchmarking results for the much bigger 100 million atom STMV running on the Blue Gene/P, Blue Gene/Q and Cray XK6 are shown in the right plot. NAMD demonstrates excellent scalability for this

Machines	Nodes	Cores	Atoms per Core	Time (ms/step)
Blue Gene/Q	4096	65536	1.4	0.683
Cray XC30	512	8192	11.0	0.526

TABLE 4.3: Time step of running ApoA1 with PME every 4 steps on Blue Gene/Q and Cray XC30

molecular system on the Blue Gene/Q and XK6 with good performance at as many as 262,144 cores.

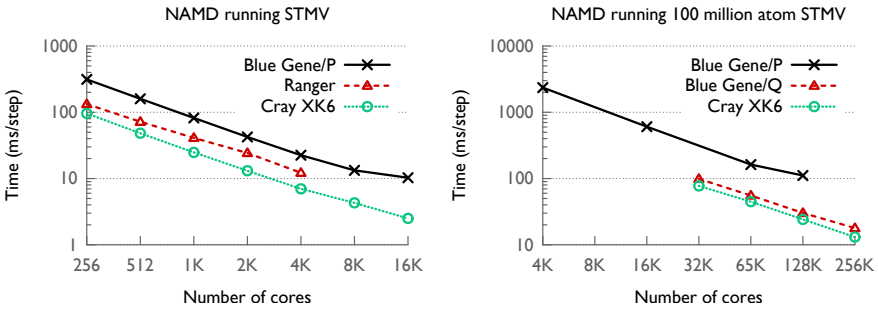

FIGURE 4.5: NAMD performance on several machines

Extreme strong scaling of NAMD. One needs to execute a billion time steps to simulate 1μs in the life of a biomolecular system! Further, a particular system being studied has a fixed size i.e. a fixed number of atoms. So, in order to do the simulations faster, one needs to carry out a single time step as fast as possible. What are the current limits of such strong scaling? Recently, we were able to simulate a time step in about 500μs. This corresponds to a simulation rate of 170 ns/day. Given the amount of communication, coordination and critical-path-bound work one has to do in each time step, such numbers are impressive, and are a testament to NAMD's innate performance orientation, as well as the design of the machines themselves.

Table 4.3 shows the best performance we have achieved for ApoA1 with PME every 4 steps on IBM Blue Gene/Q and Cray XC30. On Blue Gene/Q there are fewer than two atoms per core and we are really pushing the scaling limit using extremely fine-grained decomposition. On Cray XC30, in part due to new Cray Aries interconnect, better performance is obtained on fewer nodes. In both cases, the time per step is below one millisecond, which brings us closer to the goal of simulating longer time scales in the life of biomolecules.

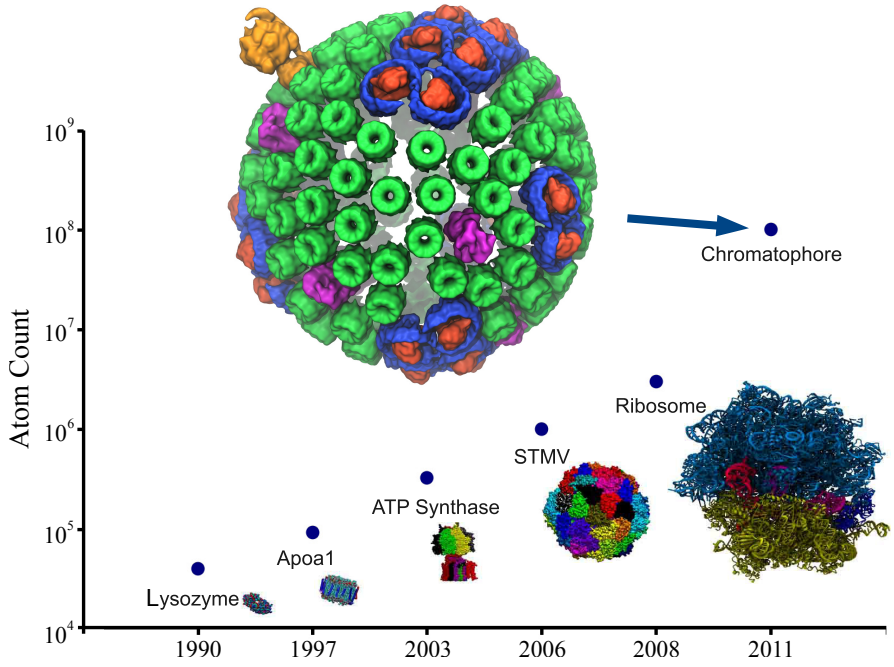

FIGURE 4.6: The size of biomolecular systems that can be studied using all-atom molecular dynamics simulations has steadily increased from that of Lysozyme (40,000 atoms) in the 1990s to the F_1F_0-ATP Synthase and STMV Virus capsid at the turn of the century, and now 100-million atoms as in the spherical chromatophore model shown above. Atom counts include aqueous solvent, not shown. **(see Color Plate 4.)**

4.7 Simulations Enabled by NAMD

NAMD is distributed free of charge as both source code and convenient pre-compiled binaries by the NIH Center for Macromolecular Modeling and Bioinformatics at the University of Illinois. NAMD is a popular program with over 50,000 registered users in the past decade, over 16,000 of whom have downloaded multiple versions of the program. NAMD has been cited in over 5,000 publications and is one of the most used programs at NSF-funded supercomputer centers. NAMD development is driven by the projects of the NIH Center, examples of which are presented below.

In the year 1999, NAMD enabled the study of the photosynthetic purple membrane of *Halobacterium salinarium*, simulating a hexagonal unit cell containing 23,700 atoms distributed over protein, lipid membrane, ion, and water

components [20]. The simulations were used to study how proteins in the membrane capture light energy to pump protons across the membrane. The difference in proton concentration thus established provides energy source that another membrane protein, adenosine triphosphate (ATP) synthase, stores as chemical bonds in ATP for transport and utilization in other processes in the cell [239]. Beginning in 2001, much larger (327,000 atoms) NAMD simulations of ATP synthase were used to study this process. In 2002, a NAMD simulation of a membrane containing aquaporin proteins revealed the mechanism by which aquaporin water channels permit water and other neutral molecules to cross the membrane while preventing the passage of protons and charged molecules [240]. Thus, aquaporins solve the important problem of maintaining a proton gradient to drive ATP synthesis while allowing water to pass.

In 2005, NAMD enabled the first all-atom molecular dynamics study of a complete virus particle [72]. Satellite Tobacco Mosaic Virus (STMV), a small and well studied plant virus, was simulated as a complex of over one million atoms of protein, nucleic acid, and water. By studying the stability of the complete virion and its isolated components, the simulations illustrated that previous speculation that STMV assembly was mostly capsid protein-driven was likely incorrect, and that instead the the virus's genetic payload recruits capsid proteins into a shell around itself.

By 2012, this initial work on virus simulation had matured such that the human HIV virus capsid could be studied, enabled by NAMD simulations on petascale supercomputers such as the "Blue Waters", a Cray XE6 at Illinois. Initial 10-million-atom simulations of a cylindrical HIV assembly have now been extended to a 64-million-atom simulation of the full HIV capsid [256]. Similarly, the earlier studies of photosynthesis have progressed to models of a complete photosynthetic unit, a pseudo-organelle called the chromatophore consisting of several hundred proteins embedded in a spherical lipid membrane [222]. Planned simulations of the chromatophore will exceed 100 million atoms.

4.8 Summary

NAMD is the first science application to use Charm++ as its underlying parallel framework. The Charm++ and NAMD collaboration has come a long way and has benefited both programs immensely. Performance improvements motivated by NAMD have inspired abstractions, optimized implementations, and robust infrastructure in Charm++, and complementary improvements to Charm++ have enabled the implementation of new features in NAMD. NAMD is one of the best scaling parallel molecular dynamics packages and portable to almost any architecture by virtue of using the Charm++ runtime.

NAMD is installed at major supercomputing centers in the U.S. and

around the world and is used by many research groups for their simulations. The study of the influenza virus (A/H1N1) and the HIV capsid are testimony to the impact of NAMD in the field of biophysics and drug design.

Acknowledgments

The many contributors to the development of the NAMD software made this project a success, including : Bilge Acun, Ilya Balabin, Milind Bhandarkar, Abhinav Bhatele, Eric Bohm, Robert Brunner, Floris Buelens, Christophe Chipot, Jordi Cohen, Jeffrey Comer, Andrew Dalke, Surjit B. Dixit, Giacomo Fiorin, Peter Freddolino, Paul Grayson, Justin Gullingsrud, Attila Gursoy, David Hardy, Chris Harrison, Jérôme Hénin, Bill Humphrey, David Hurwitz, Barry Israelewitz, Sergei Izrailev, Nikhil Jain, Neal Krawetz, Sameer Kumar, David Kunzman, Jonathan Lai, Chee Wai Lee, Charles Matthews, Ryan McGreevy, Chao Mei, Esteban Meneses, Mark Nelson, Ferenc Ötvös, Jim Phillips, Osman Sarood, Ari Shinozaki, John Stone, Johan Strumpfer, Yanhua Sun, David Tanner, Kirby Vandivort, Krishnan Varadarajan, Yi Wang, David Wells, Gengbin Zheng, Fangqiang Zhu.

This work was supported in part by a grant from the National Institutes of Health NIH 9P41GM104601 “Center for Macromolecular Modeling and Bioinformatics”. This research is part of the Blue Waters sustained-petascale computing project, which is supported by the National Science Foundation (award number OCI 07-25070) and the state of Illinois. Blue Waters is a joint effort of the University of Illinois at Urbana-Champaign and its National Center for Supercomputing Applications. This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344 (LLNL-BOOK-608433).

This research used resources of the Argonne Leadership Computing Facility at Argonne National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under contract DE-AC02-06CH11357. This research also used resources of the Oak Ridge Leadership Facility at the Oak Ridge National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725. This work used the Extreme Science and Engineering Discovery Environment (XSEDE), which is supported by National Science Foundation grant number OCI-1053575 (project allocations TG-ASC050039N and TG-ASC050040N).

Bibliography

[1] Chombo Software Package for AMR Applications. http://seesar.lbl.gov/anag/chombo.

[2] Leanmd code repository. git clone git://charm.cs.uiuc.edu/benchmarks/leanmd.

[3] A.and A. Bol A. Kasry, M. Kuroda, G.J. Martyna, and G.S. Tulevski. Chemical Doping of Large-Area Stacked Graphene Films for use as Transparent, Conducting Electrodes. *ACS Nano*, 4:3839, (2010).

[4] A. Adcroft, C. Hill, and J. Marshall. Representation of topography by shaved cells in a height coordinate ocean model. *Monthly Weather Review*, 125(9):2293–2315, 1997.

[5] L. Adhianto, S. Banerjee, M. Fagan, M. Krentel, G. Marin, J. Mellor-Crummey, and N. R. Tallent. Hpctoolkit: tools for performance analysis of optimized parallel programs http://hpctoolkit.org. *Concurr. Comput. : Pract. Exper.*, 22:685–701, April 2010.

[6] M.P. Allen and D.J. Tildesley. *Computer Simulations of Liquids*. Claredon Press, Oxford, (1989).

[7] R. J. Anderson. Tree data structures for n-body simulation. *SIAM J. Comput.*, 28:1923–1940, 1999.

[8] R. E. Angulo, V. Springel, S. D. M. White, A. Jenkins, C. M. Baugh, and C. S. Frenk. Scaling relations for galaxy clusters in the Millennium-XXL simulation. *ArXiv e-prints*, March 2012.

[9] Gabriel Antoniu, Luc Bouge, and Raymond Namyst. An efficient and transparent thread migration scheme in the PM^2 runtime system. In *Proc. 3rd Workshop on Runtime Systems for Parallel Programming (RTSPP) San Juan, Puerto Rico. Lecture Notes in Computer Science 1586*, pages 496–510. Springer-Verlag, April 1999.

[10] Amnon Barak, Shai Guday, and Richard G. Wheeler. The mosix distributed operating system. In *LNCS 672*. Springer, 1993.

[11] Kevin Barker, Andrey Chernikov, Nikos Chrisochoides, and Keshav Pingali. A Load Balancing Framework for Adaptive and Asynchronous Applications. In *IEEE Transactions on Parallel and Distributed Systems*, volume 15, pages 183–192, 2003.

[12] Kevin J. Barker and Nikos P. Chrisochoides. An Evaluation of a Framework for the Dynamic Load Balancing of Highly Adaptive and Irregular Parallel Applications. In *Proceedings of SC 2003*, Phoenix, AZ, 2003.

[13] J. Barnes and P. Hut. A Hierarchical O(NlogN) Force-Calculation Algorithm. *Nature*, 324:446–449, December 1986.

[14] C. Barrett, R. Beckman, K. Berkbigler, K. Bisset, B. Bush, K. Campbell, S. Eubank, K. Henson, J. Hurford, D. Kubicek, M. Marathe, P. Romero, J. Smith, L. Smith, P. Speckman, P. Stretz, G. Thayer, E. Eeckhout, and M. Williams. TRANSIMS: Transportation Analysis Simulation System. Technical Report LA-UR-00-1725, LANL, 2001.

[15] C. L. Barrett, K. Bisset, S. Eubank, M. V. Marathe, V.S. Anil Kumar, and Henning Mortveit. *Modeling and Simulation of Biological Networks*, chapter Modeling and Simulation of Large Biological, Information and Socio-Technical Systems: An Interaction Based Approach, pages 101–147. AMS, 2007.

[16] C. L. Barrett, S. Eubank, and M. V. Marathe. An interaction based approach to computational epidemics. In *AAAI' 08: Proceedings of the Annual Conference of AAAI*, Chicago USA, 2008. AAAI Press.

[17] C. L. Barrett, H. B. Hunt III, M. V. Marathe, S. S. Ravi, D. J. Rosenkrantz, and R. E. Stearns. Complexity of Reachability Problems for Finite Discrete Dynamical Systems. *J. Comput. Syst. Sci.*, 72(8):1317–1345, 2006.

[18] Christopher L. Barrett, Richard J. Beckman, Maleq Khan, V.S. Anil Kumar, Madhav V. Marathe, Paula E. Stretz, Tridib Dutta, and Bryan Lewis. Generation and analysis of large synthetic social contact networks. In M. D. Rossetti, R. R. Hill, B. Johansson, A. Dunkin, and R. G. Ingalls, editors, *Proceedings of the 2009 Winter Simulation Conference*, Piscataway, New Jersey, December 2009. Institute of Electrical and Electronics Engineers, Inc.

[19] A. Basermann, J. Clinckemaillie, T. Coupez, J. Fingberg, H. Digonnet, R. Ducloux, J.-M. Gratien, U. Hartmann, G. Lonsdale, B. Maerten, D. Roose, and C. Walshaw. Dynamic load balancing of finite element applications with the DRAMA Library. In *Applied Math. Modeling*, volume 25, pages 83–98, 2000.

[20] Jerome Baudry, Emad Tajkhorshid, Ferenc Molnar, James Phillips, and Klaus Schulten. Molecular dynamics study of bacteriorhodopsin and the purple membrane. *Journal of Physical Chemistry B*, 105:905–918, 2001.

[21] A.D. Becke. Density-Functional exchange-energy approximation with correct assymptotic behavior. *Phys. Rev. A*, 38:3098, (1988).

[22] R. J. Beckman, K. A. Baggerly, and M. D. McKay. Creating synthetic baseline populations. *Transportation Research Part A: Policy and Practice*, 30(6):415–429, 1996.

[23] Milind Bhandarkar, L. V. Kale, Eric de Sturler, and Jay Hoeflinger. Object-Based Adaptive Load Balancing for MPI Programs. In *Proceedings of the International Conference on Computational Science, San Francisco, CA, LNCS 2074*, pages 108–117, May 2001.

[24] Abhinav Bhatele, Eric Bohm, and Laxmikant V. Kale. Optimizing communication for charm++ applications by reducing network contention. *Concurrency and Computation: Practice and Experience*, 23(2):211–222, 2011.

[25] Abhinav Bhatelé, Laxmikant V. Kalé, and Sameer Kumar. Dynamic topology aware load balancing algorithms for molecular dynamics applications. In *23rd ACM International Conference on Supercomputing*, 2009.

[26] Scott Biersdorff, Chee Wai Lee, Allen D. Malony, and Laxmikant V. Kale. Integrated Performance Views in Charm ++: Projections Meets TAU. In *Proceedings of The 38th International Conference on Parallel Processing (ICPP)*, pages 140–147, Vienna, Austria, September 2009.

[27] Keith Bisset, Ashwin Aji, Madhav Marathe, and Wu-chun Feng. High-performance biocomputing for simulating the spread of contagion over large contact networks. *BMC Genomics*, 13(Suppl 2):S3, 2012.

[28] E. Bohm, A. Bhatele, L.V. Kale, M.E. Tuckerman, S. Kumar, J.A. Gunnels, and G.J. Martyna. Fine-grained parallelization of the Car-Parrinello ab initio molecular dynamics method on the Blue Gene/L supercomputer. *IBM J. Res. Dev.*, 52 1/2:159–176, (2008).

[29] Eric Bohm, Abhinav Bhatele, Laxmikant V. Kale, Mark E. Tuckerman, Sameer Kumar, John A. Gunnels, and Glenn J. Martyna. Fine Grained Parallelization of the Car-Parrinello ab initio MD Method on Blue Gene/L. *IBM Journal of Research and Development: Applications of Massively Parallel Systems*, 52(1/2):159–174, 2008.

[30] Kevin J. Bowers, Edmond Chow, Huafeng Xu, Ron O. Dror, Michael P. Eastwood, Brent A. Gregersen, John L. Klepeis, Istvan Kolossvary, Mark A. Moraes, Federico D. Sacerdoti, John K. Salmon, Yibing Shan,

and David E. Shaw. Molecular dynamics—scalable algorithms for molecular dynamics simulations on commodity clusters. In *SC '06: Proceedings of the 2006 ACM/IEEE conference on Supercomputing*, page 84, New York, NY, USA, 2006. ACM Press.

[31] Kevin J. Bowers, Edmond Chow, Huafeng Xu, Ron O. Dror, Michael P. Eastwood, Brent A. Gregersen, John L. Klepeis, Istvan Kolossvary, Mark A. Moraes, Federico D. Sacerdoti, John K. Salmon, Yibing Shan, and David 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, NY, USA, 2006. ACM Press.

[32] Brams. http://www.cptec.inpe.br/brams/, Janeiro 2009.

[33] S. Browne, J. Dongarra, N. Garner, K. London, and P. Mucci. A scalable cross-platform infrastructure for application performance tuning using hardware counters. In *Proceedings of Supercomputing'00*, Dallas, Texas, 2000.

[34] Robert K. Brunner and Laxmikant V. Kalé. Adapting to load on workstation clusters. In *The Seventh Symposium on the Frontiers of Massively Parallel Computation*, pages 106–112. IEEE Computer Society Press, February 1999.

[35] Robert K. Brunner and Laxmikant V. Kalé. Handling application-induced load imbalance using parallel objects. In *Parallel and Distributed Computing for Symbolic and Irregular Applications*, pages 167–181. World Scientific Publishing, 2000.

[36] G. T. Camacho and M. Ortiz. Computational modeling of impact damage in brittle materials. *Int. J. Solids Struct.*, 33:2899–2938, 1996.

[37] R. Car and M. Parrinello. Unified approach for molecular dynamics and density functional theory. *Phys. Rev. Lett.*, 55:2471, (1985).

[38] C. Cavazzoni, G.L. Chiarotti, S. Scandolo, E. Tosatti, M. Bernasconi, and M. Parrinello. Superionic and Metallic States of Water and Ammonia at Giant Planet Conditions. *Science*, 283:44, (1999).

[39] Sayantan Chakravorty and L. V. Kale. A fault tolerant protocol for massively parallel machines. In *FTPDS Workshop for IPDPS 2004*. IEEE Press, 2004.

[40] Sayantan Chakravorty and L. V. Kale. A fault tolerance protocol with fast fault recovery. In *Proceedings of the 21st IEEE International Parallel and Distributed Processing Symposium*. IEEE Press, 2007.

[41] K. Channakeshava, K. Bisset, M. Marathe, A. Vullikanti, and S. Yardi. High performance scalable and expressive modeling environment to study mobile malware in large dynamic networks. In *Proceedings of 25th IEEE International Parallel & Distributed Processing Symposium*, 2011.

[42] Karthik Channakeshava, Deepti Chafekar, Keith Bisset, Anil Vullikanti, and Madhav Marathe. EpiNet: A simulation framework to study the spread of malware in wireless networks. In *SIMUTools09*. ICST Press, March 2009. Rome, Italy.

[43] K.L. Chung, Y.L. Huang, and Y.W. Liu. Efficient algorithms for coding Hilbert curve of arbitrary-sized image and application to window query. *Information Sciences*, 177(10):2130–2151, 2007.

[44] A.J. Cohen, Paula Mori-Sanchez, and Weitao Yang. Insights into current limitations of density functional theory. *Science*, 321:792, (2008).

[45] M. C.Payne, M.P. Teter, D.C. Allan, T.A. Arias, and J.D. Joannopoulos. Iterative minimization techniques for ab initio total-energy calculations: molecular dynamics and conjugate gradients. *Rev. Mod. Phys.*, 64:1045, (1992).

[46] T.A. Darden, D.M. York, and L.G. Pedersen. Particle mesh Ewald. An N·log(N) method for Ewald sums in large systems. *JCP*, 98:10089–10092, 1993.

[47] M. Davis, G. Efstathiou, C. S. Frenk, and S. D. M. White. The evolution of large-scale structure in a universe dominated by cold dark matter. *Astrophys. J.*, 292:371–394, May 1985.

[48] M. Davis, G. Efstathiou, C. S. Frenk, and S. D. M. White. The evolution of large-scale structure in a universe dominated by cold dark matter. *Astrophys. J.*, 292:371–394, May 1985.

[49] W. Dehnen. Towards optimal softening in three-dimensional N-body codes - I. Minimizing the force error. *MNRAS*, 324:273–291, June 2001.

[50] S.W. deLeeuw, J.W. Perram, and E.R. Smith. Simulation of Electrostatic Systems in Periodic Boundary Conditions. I. Lattice Sums and Dielectric Constants. *Proc. R. Soc. London A*, 373:27, 1980.

[51] Department of Computer Science,University of Illinois at Urbana-Champaign, Urbana, IL. *The CHARM (5.9) programming language manual*, 2006.

[52] Department of Computer Science,University of Illinois at Urbana-Champaign, Urbana, IL. *The CONVERSE programming language manual*, 2006.

[53] Jayant DeSouza and Laxmikant V. Kalé. MSA: Multiphase specifically shared arrays. In *Proceedings of the 17th International Workshop on Languages and Compilers for Parallel Computing*, West Lafayette, Indiana, USA, September 2004.

[54] K. Devine, B. Hendrickson, E. Boman, M. St. John, and C. Vaughan. Design of Dynamic Load-Balancing Tools for Parallel Applications. In *Proc. Intl. Conf. Supercomputing*, May 2000.

[55] Karen D. Devine, Erik G. Boman, Robert T. Heaphy, Bruce A. Hendrickson, James D. Teresco, Jamal Faik, Joseph E. Flaherty, and Luis G. Gervasio. New challenges in dynamic load balancing. *Appl. Numer. Math.*, 52(2–3):133–152, 2005.

[56] J. Diemand, M. Kuhlen, P. Madau, M. Zemp, B. Moore, D. Potter, and J. Stadel. Clumps and streams in the local dark matter distribution. *Nature*, 454:735–738, August 2008.

[57] H.-Q. Ding, N. Karasawa, and W. A. Goddard, III. The reduced cell multipole method for Coulomb interactions in periodic systems with million-atom unit cells. *Chemical Physics Letters*, 196:6–10, August 1992.

[58] P. Domingos and M. Richardson. Mining the Network Value of Customers. In *Proc. ACM KDD*, pages 57–61, 2001.

[59] Isaac Dooley. *Intelligent Runtime Tuning of Parallel Applications With Control Points*. PhD thesis, Dept. of Computer Science, University of Illinois, 2010. http://charm.cs.uiuc.edu/papers/DooleyPhDThesis10.shtml.

[60] D. J. Earl and M.W. Deem. Parallel tempering: Theory, applications, and new perspectives. *Phys. Chem. Chem. Phys.*, 7:3910–3916, (2005).

[61] D. Easley and J. Kleinberg. *Networks, Crowds and Markets: Reasoning About A Highly Connected World*. Cambridge University Press, New York, NY, 2010.

[62] G. Efstathiou, M. Davis, S. D. M. White, and C. S. Frenk. Numerical techniques for large cosmological N-body simulations. *Astrophys. J. Supp.*, 57:241–260, February 1985.

[63] S.N. Eliane, E. Araújo, W. Cirne, G. Wagner, N. Oliveira, E.P. Souza, C.O. Galvão, and E.S. Martins. The SegHidro Experience: Using the Grid to Empower a HydroMeteorological. In *Proceedings of the First International Converence on e-Science and Grid Computing (e-Science/05)*, pages 64–71, 2005.

[64] S. Eubank, H. Guclu, V. S. Anil Kumar, M. Marathe, A. Srinivasan, Z. Toroczkai, and N. Wang. Modelling disease outbreaks in realistic urban social networks. *Nature*, 429:180–184, 2004.

[65] A. E. Evrard. Beyond N-body - 3D cosmological gas dynamics. *MNRAS*, 235:911–934, December 1988.

[66] P. P. Ewald. Die Berechnung optischer und elektrostatischer Gitterpotentiale. *Annalen der Physik*, 369:253–287, 1921.

[67] A. L. Fazenda, J. Panetta, P. Navaux, L. F. Rodrigues, D. M. Katsurayama, and L. F Motta. Escalabilidade de aplicação operacional em ambiente massivamente paralelo. In *Anais do X Simpósio em Sistemas Computacionais (WSCAD-SCC)*, pages 27–34, 2009.

[68] R.P. Feynman. *Statistical Mechanics.* Benjamin, Reading, (1972).

[69] 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.

[70] Blake G. Fitch, Aleksandr Rayshubskiy, Maria Eleftheriou, T. J. Christopher Ward, Mark Giampapa, Michael C. Pitman, and Robert S. Germain. Molecular dynamics—blue matter: approaching the limits of concurrency for classical molecular dynamics. In *SC '06: Proceedings of the 2006 ACM/IEEE conference on Supercomputing*, page 87, New York, NY, USA, 2006. ACM Press.

[71] IT Foster and BR Toonen. Load-balancing algorithms for climate models. In *Proceedings of Scalable High-Performance Computing Conference*, pages 674–681, 1994.

[72] Peter L. Freddolino, Anton S. Arkhipov, Steven B. Larson, Alexander McPherson, and Klaus Schulten. Molecular dynamics simulations of the complete satellite tobacco mosaic virus. *Structure*, 14:437–449, 2006.

[73] SR Freitas, KM Longo, MAF Silva Dias, R. Chatfield, P. Silva Dias, P. Artaxo, MO Andreae, G. Grell, LF Rodrigues, A. Fazenda, et al. The Coupled Aerosol and Tracer Transport model to the Brazilian developments on the Regional Atmospheric Modeling System (CATT-BRAMS). *Atmospheric Chemistry and Physics*, 9(8):2843–2861, 2009.

[74] C. S. Frenk, S. D. M. White, P. Bode, J. R. Bond, G. L. Bryan, R. Cen, H. M. P. Couchman, A. E. Evrard, N. Gnedin, A. Jenkins, A. M. Khokhlov, A. Klypin, J. F. Navarro, M. L. Norman, J. P. Ostriker, J. M. Owen, F. R. Pearce, U.-L. Pen, M. Steinmetz, P. A. Thomas, J. V. Villumsen, J. W. Wadsley, M. S. Warren, G. Xu, and G. Yepes. The Santa Barbara Cluster Comparison Project: A Comparison of Cosmological Hydrodynamics Solutions. *Astrophys. J.*, 525:554–582, November 1999.

[75] D. Frenkel and B. Smit. *Understanding Molecular Simulation*. Academic Press, 1996.

[76] George Karypis and Vipin Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM J. Sci. Comput.*, 20(1):359–392, 1998.

[77] George Karypis and Vipin Kumar. Multilevel k-way Partitioning Scheme for Irregular Graphs. *Journal of Parallel and Distributed Computing*, 48:96–129 , 1998.

[78] T. C. Germann, K. Kadau, I. M. Longini, Jr., and C. A. Macken. Mitigation strategies for pandemic influenza in the United States. *Proc. of National Academy of Sciences*, 103(15):5935–5940, April11 2006.

[79] P. H. Geubelle and J. Baylor. Impact-induced delamination of composites: a 2d simulation. *Composites B*, 29(B):589–602, 1998.

[80] R. Gevaerd, S. R. Freitas, and K. M. Longo. Numerical simulation of biomass burning emission and trasportation during 1998 roraima fires. In *Proceedings of International Conference on Southern Hemisphere Meteorology and Oceanography (ICSHMO) 8*, 2006.

[81] S. Ghan, X. Bian, A. Hunt, and A. Coleman. The thermodynamic influence of subgrid orography in a global climate model. *Climate Dynamics*, 20(1):31–44, 2002.

[82] S. Ghan and T. Shippert. Load balancing and scalability of a subgrid orography scheme in a global climate model. *International Journal of High Performance Computing Applications*, 19(3):237, 2005.

[83] D.S. Ginley and D. Cahen. *Fundamentals of Materials for Energy and Environmental Sustainability*. Cambridge University Press, Cambridge, UK.

[84] Filippo Gioachin and Laxmikant V. Kalé. Dynamic High-Level Scripting in Parallel Applications. In *In Proceedings of the 23rd IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, Rome, Italy, May 2009.

[85] Filippo Gioachin, Amit Sharma, Sayantan Chakravorty, Celso Mendes, Laxmikant V. Kale, and Thomas R. Quinn. Scalable cosmology simulations on parallel machines. In *VECPAR 2006, LNCS 4395, pp. 476-489*, 2007.

[86] Filippo Gioachin, Gengbin Zheng, and Laxmikant V. Kalé. Debugging Large Scale Applications in a Virtualized Environment. In *Proceedings of the 23rd International Workshop on Languages and Compilers for Parallel Computing (LCPC2010)*, number 10-11, Huston, TX (USA), October 2010.

[87] Filippo Gioachin, Gengbin Zheng, and Laxmikant V. Kalé. Robust Record-Replay with Processor Extraction. In *PADTAD '10: Proceedings of the 8th Workshop on Parallel and Distributed Systems: Testing, Analysis, and Debugging*, pages 9–19. ACM, July 2010.

[88] E Goldstein, A Apolloni, B Lewis, J C Miller, M Macauley, S Eubank, M Lipsitch, and J Wallinga. Distribution of vaccine/antivirals and the 'least spread line'; in a stratified population. *J R Soc Interface*, 7(46):755–64, 2010.

[89] R. Gould. Collective action and network structure. *American Sociological Review*, 58:182–196, 1993.

[90] F. Governato, C. Brook, L. Mayer, A. Brooks, G. Rhee, J. Wadsley, P. Jonsson, B. Willman, G. Stinson, T. Quinn, and P. Madau. Bulgeless dwarf galaxies and dark matter cores from supernova-driven outflows. *Nature*, 463:203–206, January 2010.

[91] F. Governato, B. Willman, L. Mayer, A. Brooks, G. Stinson, O. Valenzuela, J. Wadsley, and T. Quinn. Forming disc galaxies in ΛCDM simulations. *MNRAS*, 374:1479–1494, February 2007.

[92] S. L . Graham, P. B. Kessler, and M. K. McKusick. GPROF: a call graph execution profiler. *SIGPLAN 1982 Symposium on Compiler Construction*, pages 120–126, June 1982.

[93] M. Granovetter. Threshold Models of Collective Behavior. *American J. Sociology*, 83(6):1420–1443, 1978.

[94] L. Greengard. *The rapid evaluation of potential fields in particle systems.* PhD thesis, MIT, Cambridge, MA, USA., 1988.

[95] G. Grell and D. Devenyi. A generalized approach to parameterizing convection combining ensemble and data assimilation techniques. *Geophysical Research Letters*, 29(14):38–1, 2002.

[96] G. Grimmett. *Percolation.* Springer, 1989.

[97] A. Gursoy, L.V. Kale, and S.P. Vanka. Unsteady fluid flow calculations using a machine independent parallel programming environment. In R. B. Pelz, A. Ecer, and J. Hauser, editors, *Parallel Computational Fluid Dynamics '92*, pages 175–185. North-Holland, 1993.

[98] A. Haldane and R. May. Systemic risk in banking ecosystems. *Nature*, 469:351–355, 2011.

[99] M. Halloran, N. Ferguson, I. Longini S. Eubank, D. Cummings, B. Lewis, S Xu, C. Fraser, A. Vullikanti, T. Germann, D. Wagener, R. Beckman, K. Kadau, C. Barrett, C. Macken, D. Burke, and P. Cooley. Modeling targeted layered containment of an influenza pandemic in the united states. *PNAS*, 105(12):4639–4644, 2008.

[100] M. Elizabeth Halloran, Neil M. Ferguson, Stephen Eubank, Ira M. Longini, Derek A. T. Cummings, Bryan Lewis, Shufu Xu, Christophe Fraser, Anil Vullikanti, Timothy C. Germann, Diane Wagener, Richard Beckman, Kai Kadau, Chris Barrett, Catherine A. Macken, Donald S. Burke, and Philip Cooley. Modeling targeted layered containment of an influenza pandemic in the united states. *Proceedings of the National Academy of Sciences*, 105(12):4639–4644, March 2008.

[101] R. Halstead. Multilisp: A Language for Concurrent Symbolic Computation. *ACM Transactions on Programming Languages and Systems*, October 1985.

[102] Tsuyoshi Hamada, Tetsu Narumi, Rio Yokota, Kenji Yasuoka, Keigo Nitadori, and Makoto Taiji. 42 tflops hierarchical n-body simulations on gpus with applications in both astrophysics and turbulence. In *Proceedings of the Conference on High Performance Computing Networking, Storage and Analysis*, SC '09, pages 62:1–62:12, New York, NY, USA, 2009. ACM.

[103] Richard Hamming. *Numerical Analysis for Scientists and Engineers*. 1973.

[104] K. Heitmann, P. M. Ricker, M. S. Warren, and S. Habib. Robustness of Cosmological Simulations. I. Large-Scale Structure. *Astrophys. J. Supp.*, 160:28–58, September 2005.

[105] L. Hernquist, F. R. Bouchet, and Y. Suto. Application of the Ewald method to cosmological N-body simulations. *Astrophys. J. Supp.*, 75:231–240, February 1991.

[106] L. Hernquist and N. Katz. TREESPH - A unification of SPH with the hierarchical tree method. *Astrophys. J. Supp.*, 70:419–446, June 1989.

[107] D. Hilbert. Über die stetige abbildung einer linie auf ein flächenstück. *Mathematische Annalen*, 38:459–460, 1891.

[108] R. W. Hockney and J. W. Eastwood. *Computer Simulation Using Particles*. New York: McGraw-Hill, 1981.

[109] P. Hohenberg and W. Kohn. Inhomogeneous electron gas. *Phys. Rev.*, 136:B864, 1964.

[110] Chao Huang and Laxmikant V. Kale. Charisma: Orchestrating migratable parallel objects. In *Proceedings of IEEE International Symposium on High Performance Distributed Computing (HPDC)*, July 2007.

[111] Chao Huang, Orion Lawlor, and L. V. Kalé. Adaptive MPI. In *Proceedings of the 16th International Workshop on Languages and Compilers for Parallel Computing (LCPC 2003), LNCS 2958*, pages 306–322, College Station, Texas, October 2003.

[112] Chao Huang, Gengbin Zheng, Sameer Kumar, and Laxmikant V. Kalé. Performance Evaluation of Adaptive MPI. In *Proceedings of ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming 2006*, March 2006.

[113] J. JáJá. *An introduction to parallel algorithms*. Addison Wesley Longman Publishing Co., Inc. Redwood City, CA, USA, 1992.

[114] Pritish Jetley, Filippo Gioachin, Celso Mendes, Laxmikant V. Kale, and Thomas R. Quinn. Massively parallel cosmological simulations with ChaNGa. In *Proceedings of IEEE International Parallel and Distributed Processing Symposium 2008*, 2008.

[115] Pritish Jetley, Lukasz Wesolowski, Filippo Gioachin, Laxmikant V. Kalé, and Thomas R. Quinn. Scaling hierarchical n-body simulations on gpu clusters. In *Proceedings of the 2010 ACM/IEEE International Conference for High Performance Computing, Networking, Storage and Analysis*, SC '10, Washington, DC, USA, 2010. IEEE Computer Society.

[116] Xiangmin Jiao, Gengbin Zheng, Phillip A. Alexander, Michael T. Campbell, Orion S. Lawlor, John Norris, Andreas Haselbacher, and Michael T. Heath. A system integration framework for coupled multiphysics simulations. *Engineering with Computers*, 22(3):293–309, 2006.

[117] J. M. Jiménez, B. L. Lewis, and S. Eubank. Hospitals as complex social systems: agent-based simulation of hospital-acquired infections. In *Proceedings of 2nd International Conference on Complex Sciences: Theory and Applications*, 2012.

[118] John A. Board Jr., Laxmikant V. Kale, Klaus Schulten, Robert D. Skeel, , and Tamar Schlick. Modeling biomolecules: Large sclaes, longer durations. *IEEE Computational Science & Engineering*, 1:19–30, Winter 1994.

[119] Rashmi Jyothi, Orion Sky Lawlor, and L. V. Kale. Debugging support for Charm++. In *PADTAD Workshop for IPDPS 2004*, page 294. IEEE Press, 2004.

[120] L. V. Kale. Application oriented and computer science centered HPCC research. pages 98–105, 1994.

[121] L. V. Kale and Milind Bhandarkar. Structured Dagger: A Coordination Language for Message-Driven Programming. In *Proceedings of Second International Euro-Par Conference*, volume 1123-1124 of *Lecture Notes in Computer Science*, pages 646–653, September 1996.

[122] L. V. Kale and Sanjeev Krishnan. A comparison based parallel sorting algorithm. In *Proceedings of the 22nd International Conference on Parallel Processing*, pages 196–200, St. Charles, IL, August 1993.

[123] L. V. Kale and Sanjeev Krishnan. A comparison based parallel sorting algorithm. In *Proceedings of the 22nd International Conference on Parallel Processing*, pages 196–200, St. Charles, IL, August 1993.

[124] L. V. Kale and Sanjeev Krishnan. Charm++: Parallel Programming with Message-Driven Objects. In Gregory V. Wilson and Paul Lu, editors, *Parallel Programming using C++*, pages 175–213. MIT Press, 1996.

[125] L. V. Kale, B. H. Richards, and T. D. Allen. Efficient parallel graph coloring with prioritization. In *Lecture Notes in Computer Science*, volume 1068, pages 190–208. Springer-Verlag, August 1995.

[126] Laxmikant Kale, Anshu Arya, Abhinav Bhatele, Abhishek Gupta, Nikhil Jain, Pritish Jetley, Jonathan Lifflander, Phil Miller, Yanhua Sun, Ramprasad Venkataraman, Lukasz Wesolowski, and Gengbin Zheng. Charm++ for productivity and performance: A submission to the 2011 HPC class II challenge. Technical Report 11-49, Parallel Programming Laboratory, November 2011.

[127] Laxmikant Kale, Anshu Arya, Nikhil Jain, Akhil Langer, Jonathan Lifflander, Harshitha Menon, Xiang Ni, Yanhua Sun, Ehsan Totoni, Ramprasad Venkataraman, and Lukasz Wesolowski. Migratable objects + active messages + adaptive runtime = productivity + performance a submission to 2012 HPC class II challenge. Technical Report 12-47, Parallel Programming Laboratory, November 2012.

[128] Laxmikant Kalé, Robert Skeel, Milind Bhandarkar, Robert Brunner, Attila Gursoy, Neal Krawetz, James Phillips, Aritomo Shinozaki, Krishnan Varadarajan, and Klaus Schulten. NAMD2: Greater scalability for parallel molecular dynamics. *Journal of Computational Physics*, 151:283–312, 1999.

[129] Laxmikant V. Kalé. The virtualization model of parallel programming : Runtime optimizations and the state of art. In *LACSI 2002*, Albuquerque, October 2002.

[130] Laxmikant V. Kalé. Performance and productivity in parallel programming via processor virtualization. In *Proc. of the First Intl. Workshop on Productivity and Performance in High-End Computing (at HPCA 10)*, Madrid, Spain, February 2004.

[131] Laxmikant V. Kalé, Sameer Kumar, Gengbin Zheng, and Chee Wai Lee. Scaling molecular dynamics to 3000 processors with projections: A performance analysis case study. In *Terascale Performance Analysis Workshop, International Conference on Computational Science(ICCS)*, Melbourne, Australia, June 2003.

[132] Laxmikant V. Kale, Gengbin Zheng, Chee Wai Lee, and Sameer Kumar. Scaling applications to massively parallel machines using projections performance analysis tool. In *Future Generation Computer Systems Special Issue on: Large-Scale System Performance Modeling and Analysis*, volume 22, pages 347–358, February 2006.

[133] L.V. Kalé and S. Krishnan. CHARM++: A Portable Concurrent Object Oriented System Based on C++. In A. Paepcke, editor, *Proceedings of OOPSLA'93*, pages 91–108. ACM Press, September 1993.

[134] L.V. Kalé and Amitabh Sinha. Projections: A preliminary performance tool for charm. In *Parallel Systems Fair, International Parallel Processing Symposium*, pages 108–114, Newport Beach, CA, April 1993.

[135] S.A. Kalogirou. *Solar Energy Engineering: Processes and Systems*. Academic Press, Waltham, MA USA.

[136] George Karypis and Vipin Kumar. METIS: Unstructured graph partitioning and sparse matrix ordering system. University of Minnesota, 1995.

[137] George Karypis and Vipin Kumar. Parallel multilevel k-way partitioning scheme for irregular graphs. In *Supercomputing '96: Proceedings of the 1996 ACM/IEEE conference on Supercomputing (CDROM)*, page 35, 1996.

[138] D. Kempe, J. Kleinberg, and E. Tardos. Maximizing the Spread of Influence Through a Social Network. In *Proc. ACM KDD*, pages 137–146, 2003.

[139] D. Kempe, J. Kleinberg, and E. Tardos. Influential Nodes in a Diffusion Model for Social Networks. In *Proc. ICALP*, pages 1127–1138, 2005.

[140] C.H. Koelbel, D.B. Loveman, R.S. Schreiber, G.L. Steele Jr., and M.E. Zosel. *The High Performance Fortran Handbook*. MIT Press, 1994.

[141] W. Kohn and L.J. Sham. Self-consistent equations including exchange and correlation effects. *Phys. Rev.*, 140:A1133, 1965.

[142] C. Koziar, R. Reilein, and G. Runger. Load imbalance aspects in atmosphere simulations. *International Journal of Computational Science and Engineering*, 1(2):215–225, 2005.

[143] Sanjeev Krishnan and L. V. Kale. A parallel adaptive fast multipole algorithm for n-body problems. In *Proceedings of the International Conference on Parallel Processing*, pages III 46 – III 50, August 1995.

[144] Rick Kufrin. Perfsuite: An Accessible, Open Source Performance Analysis Environment for Linux. In *In Proceedings of the Linux Cluster Conference*, 2005.

[145] Sameer Kumar. *Optimizing Communication for Massively Parallel Processing*. PhD thesis, University of Illinois at Urbana-Champaign, May 2005.

[146] Sameer Kumar, Chao Huang, Gheorghe Almasi, and Laxmikant V. Kalé. Achieving strong scaling with NAMD on Blue Gene/L. In *Proceedings of IEEE International Parallel and Distributed Processing Symposium 2006*, April 2006.

[147] V. Kumar. *Introduction to parallel computing*. Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA, 2002.

[148] Akhil Langer, Jonathan Lifflander, Phil Miller, Kuo-Chuan Pan, , Laxmikant V. Kale, and Paul Ricker. Scalable Algorithms for Distributed-Memory Adaptive Mesh Refinement. In *Proceedings of the 24th International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD 2012). To Appear*, New York, USA, October 2012.

[149] Ilya Lashuk, Aparna Chandramowlishwaran, Harper Langston, Tuan-Anh Nguyen, Rahul Sampath, Aashay Shringarpure, Richard Vuduc, Lexing Ying, Denis Zorin, and George Biros. A massively parallel adaptive fast-multipole method on heterogeneous architectures. In *SC '09: Proceedings of the Conference on High Performance Computing Networking, Storage and Analysis*, pages 1–12, New York, NY, USA, 2009. ACM.

[150] Orion Lawlor, Sayantan Chakravorty, Terry Wilmarth, Nilesh Choudhury, Isaac Dooley, Gengbin Zheng, and Laxmikant Kale. Parfum: A parallel framework for unstructured meshes for scalable dynamic physics applications. *Engineering with Computers*, 22(3-4):215–235, September 2006.

[151] Orion Lawlor, Hari Govind, Isaac Dooley, Michael Breitenfeld, and Laxmikant Kale. Performance degradation in the presence of subnormal floating-point values. In *Proceedings of the International Workshop on Operating System Interference in High Performance Applications*, September 2005.

[152] Orion Sky Lawlor. *Impostors for Parallel Interactive Computer Graphics*. PhD thesis, University of Illinois at Urbana-Champaign, December 2004.

[153] Orion Sky Lawlor and L. V. Kalé. Supporting dynamic parallel object arrays. *Concurrency and Computation: Practice and Experience*, 15:371–393, 2003.

[154] D. Lea and W. Gloger. A memory allocator. `http://web.mit.edu/sage/export/singular-3-0-4-3.old/omalloc/Misc/dlmalloc/malloc.ps`, 2000.

[155] C. Lee, W. Yang, and R.G. Parr. Development of the Calle-Salvetti correlation energy into a functional of the electron density. *Phys. Rev. B*, 37:785, (1988).

[156] Chee Wai Lee. *Techniques in Scalable and Effective Parallel Performance Analysis*. PhD thesis, Department of Computer Science, University of Illinois, Urbana-Champaign, December 2009.

[157] J. K. Lenstra, D. B. Shmoys, and E. Tardos. Approximation algorithms for scheduling unrelated parallel machines. *Math. Program.*, 46(3):259–271, 1990.

[158] J. Leskovec, L. Adamic, and B. Huberman. The Dynamics of Viral Marketing. *ACM Trans. on the Web*, 1(1), 2007.

[159] J.R. Levine. *Linkers and Loaders*. Morgan-Kauffman, 2000.

[160] Bryan Lewis, Stephen Eubank, Allyson M Abrams, and Ken Kleinman. In silico surveillance: evaluating outbreak detection with simulation models. *BMC medical informatics and decision making*, 13(1):12, January 2013.

[161] G. F. Lewis, A. Babul, N. Katz, T. Quinn, L. Hernquist, and D. H. Weinberg. The Effects of Gasdynamics, Cooling, Star Formation, and Numerical Resolution in Simulations of Cluster Formation. *Astrophys. J.*, 536:623–644, June 2000.

[162] X. Li, W. Cai, J. An, S. Kim, J. Nah, D. Yang, R. Piner, A. Velamakanni, I. Jung, E. Tutuc, S.K. Banerjee, L. Colombo, and R.S. Ruoff. Large-Area Synthesis of High-Quality and Uniform Graphene Films on Copper Foils. *Science*, 324:1312, (2009).

[163] X. Liu and G. Schrack. Encoding and decoding the Hilbert order. *Software, practice & experience*, 26(12):1335–1346, 1996.

[164] Kwan-Liu Ma, Greg Schussman, Brett Wilson, Kwok Ko, Ji Qiang, and Robert Ryne. Advanced visualization technology for terascale particle accelerator simulations. In *Supercomputing '02: Proceedings of the 2002 ACM/IEEE conference on Supercomputing*, pages 1–11, Los Alamitos, CA, USA, 2002. IEEE Computer Society Press.

[165] Paulo W. C. Maciel and Peter Shirley. Visual navigation of large environments using textured clusters. In *Proceedings of the 1995 symposium on Interactive 3D graphics*, pages 95–ff. ACM Press, 1995.

[166] Sandhya Mangala, Terry Wilmarth, Sayantan Chakravorty, Nilesh Choudhury, Laxmikant V. Kale, and Philippe H. Geubelle. Parallel adaptive simulations of dynamic fracture events. *Engineering with Computers*, 24:341–358, December 2007.

[167] Achla Marathe, Bryan Lewis, Christopher Barrett, Jiangzhuo Chen, Madhav Marathe, Stephen Eubank, and Yifei Ma. Comparing effectiveness of top-down and bottom-up strategies in containing influenza. *PloS one*, 6(9):e25149, 2011.

[168] Achla Marathe, Bryan Lewis, Jiangzhuo Chen, and Stephen Eubank. Sensitivity of household transmission to household contact structure and size. *PloS one*, 6(8):e22461, 2011.

[169] Dominik Marx, Mark E. Tuckerman, and M. Parrinello. The nature of the hydrated excess proton in water. *Nature*, 601:397, (1999).

[170] L. Mayer, T. Quinn, J. Wadsley, and J. Stadel. Formation of Giant Planets by Fragmentation of Protoplanetary Disks. *Science*, 298:1756–1759, November 2002.

[171] M. McPherson, L. Smith-Lovin, and J. Cook. Birds of a Feather: Homophily in Social Networks. *Annual Review of Sociology*, 27:415–444, 2001.

[172] Chao Mei, Yanhua Sun, Gengbin Zheng, Eric J. Bohm, Laxmikant V. Kalé, James C.Phillips, and Chris Harrison. Enabling and scaling biomolecular simulations of 100 million atoms on petascale machines with a multicore-optimized message-driven runtime. In *Proceedings of the 2011 ACM/IEEE conference on Supercomputing*, Seattle, WA, November 2011.

[173] Chao Mei, Gengbin Zheng, Filippo Gioachin, and Laxmikant V. Kalé. Optimizing a Parallel Runtime System for Multicore Clusters: A Case Study. In *TeraGrid'10*, number 10-13, Pittsburgh, PA, USA, August 2010.

[174] Esteban Meneses, Greg Bronevetsky, and Laxmikant V. Kale. Dynamic load balance for optimized message logging in fault tolerant hpc applications. In *IEEE International Conference on Cluster Computing (Cluster) 2011*, September 2011.

[175] Esteban Meneses, Celso L. Mendes, and Laxmikant V. Kale. Team-based message logging: Preliminary results. In *3rd Workshop on Resiliency in High Performance Computing (Resilience) in Clusters, Clouds, and Grids (CCGRID 2010).*, May 2010.

[176] J. Michalakes. MM90: a scalable parallel implementation of the Penn State/NCAR Mesoscale Model (MM5). *Parallel Computing*, 23(14):2173–2186, 1997.

[177] John Michalakes, Josh Hacker, Richard Loft, Michael O. McCracken, Allan Snavely, Nicholas J. Wright, Tom Spelce, Brent Gorda, and Robert Walkup. Wrf nature run. In *Proceedings of SuperComputing*, pages 1–6, Los Alamitos, CA, USA, 2007. IEEE Computer Society.

[178] Phil Miller, Aaron Becker, and Laxmikant Kal. Using shared arrays in message-driven parallel programs. *Parallel Computing*, 38(12):66 – 74, 2012.

[179] J. Minkel. The 2003 northeast blackout–five years later. *Scientific American*, 2008. 13 August 2008, `http://www.scientificamerican.com/article.cfm?id=2003-blackout-five-years-later.`

[180] M.Levy. Universal variational functionals of electron densities, first-order density matrices, and natural spin-orbitals and solution of the v-representability problem. *Proc. Natl. Acad. Sci. U.S.A.*, 76:6062, (1979).

[181] P. R. Monge and N. S. Contractor. *Theories of Communication Networks*. Oxford University Press, USA, 2003.

[182] B. Moore, F. Governato, T. Quinn, J. Stadel, and G. Lake. Resolving the Structure of Cold Dark Matter Halos. *Astrophys. J. Lett.*, 499:L5–+, May 1998.

[183] E. Moretti. Social learning and peer effects in consumption: Evidence from movie sales. *Review of Economic Studies*, 78:356–393, 2011.

[184] H. Mortveit and C. Reidys. *An Introduction to Sequential Dynamical Systems*. Springer, New York, NY, 2007.

[185] Martin Mundhenk, Judy Goldsmith, Christopher Lusena, and Eric Allender. Complexity of finite-horizon markov decision process problems. *JACM*, 47(4):681–720, July 2000.

[186] National Institutes of Health, 2009. http://www.nigms.nih.gov/Initiatives/MIDAS/.

[187] J. F. Navarro, C. S. Frenk, and S. D. M. White. A Universal Density Profile from Hierarchical Clustering. *Astrophys. J.*, 490:493, December 1997.

[188] NDSSL. Synthetic data products for societal infrastructures and proto-populations: Data set 2.0. Technical Report NDSSL-TR-07-003, NDSSL, Virginia Polytechnic Institute and State University, Blacksburg, VA, 24061, 2007.

[189] M. Newman. The structure and function of complex networks. *SIAM Review*, 45, 2003.

[190] M.E. Newman. Spread of epidemic disease on networks. *Phys. Rev. E*, 2002.

[191] D.M. Newns, B.G. Elmegreen, X.-H. Liu, and G.J. Martyna. High Response Piezoelectric and Piezoresistive Materials for Fast, Low Voltage Switching: Simulation and Theory of Transduction Physics at the Nanometer-Scale. *Adv. Mat.*, 24:3672, 2012.

[192] D.M. Newns, B.G. Elmegreen, X.-H. Liu, and G.J. Martyna. High Response Piezoelectric and Piezoresistive Materials for Fast, Low Voltage Switching: Simulation and Theory of Transduction Physics at the Nanometer-Scale. *Adv. Mat.*, 24:3672, 2012.

[193] D.M. Newns, B.G. Elmegreen, X.-H. Liu, and G.J. Martyna. The piezoelectronic transistor:A nanoactuator-based post-CMOS digital switch with high speed and low power. *MRS Bulletin*, 37:1071, 2012.

[194] D.M. Newns, J.A. Misewich, A. Gupta C.C. Tsuei, B.A. Scott, and A. Schrott. Mott Transition Field Effect Transistor. *Appl. Phys. Lett.*, 73:780, (1998).

[195] R. Nistor, D.M. Newns, and G.J. Martyna. Understanding the doping mechanism in graphene-based electronics: The role of chemistry. *ACS Nano*, 5:3096, (2011).

[196] A. Odell1, A. Delin1, B. Johansson, N. Bock, M. Challacombe, and A. M. N. Niklasson. Higher-order symplectic integration in Born.Oppenheimer molecular dynamics. *J. Chem. Phys.*, 131:244106, (2009).

[197] Committee on Modeling Community Containment for Pandemic Influenza and Institute of Medicine. *Modeling Community Containment for Pandemic Influenza: A Letter Report*. The National Academies Press, Washington D.C., 2006.

[198] Ehsan Totoni Osman Sarood, Phil Miller and L. V. Kale. 'Cool' Load Balancing for High Performance Computing Data Centers. In *IEEE Transactions on Computer - SI (Energy Efficient Computing)*, September 2012.

[199] J. P. Ostriker and P. J. E. Peebles. A Numerical Study of the Stability of Flattened Galaxies: or, can Cold Galaxies Survive? *Astrophys. J.*, 186:467–480, December 1973.

[200] Douglas Z. Pan and Mark A. Linton. Supporting reverse execution for parallel programs. *SIGPLAN Not.*, 24(1):124–129, 1989.

[201] J. P. Perdew, K. Burke, and M. Ernzerhof. Generalized Gradient Approximation Made Simple. *Phys. Rev. B*, 77:386, (1996).

[202] P. Perzyna. Fundamental problems in viscoplasticity. *Advances in applied mechanics*, 9(C):243–377, 1966.

[203] James C. Phillips, Gengbin Zheng, Sameer Kumar, and Laxmikant V. Kalé. NAMD: Biomolecular simulation on thousands of processors. In *Proceedings of the 2002 ACM/IEEE conference on Supercomputing*, pages 1–18, Baltimore, MD, September 2002.

[204] Planck Collaboration, N. Aghanim, M. Arnaud, M. Ashdown, F. Atrio-Barandela, J. Aumont, C. Baccigalupi, A. Balbi, A. J. Banday, R. B. Barreiro, J. G. Bartlett, E. Battaner, K. Benabed, J.-P. Bernard, M. Bersanelli, H. Böhringer, A. Bonaldi, J. R. Bond, J. Borrill, F. R. Bouchet, H. Bourdin, M. L. Brown, C. Burigana, R. C. Butler, P. Cabella, J.-F. Cardoso, P. Carvalho, A. Catalano, L. Cayón, A. Chamballu, R.-R. Chary, L.-Y. Chiang, G. Chon, P. R. Christensen, D. L. Clements, S. Colafrancesco, S. Colombi, A. Coulais, B. P. Crill, F. Cuttaia, A. Da Silva, H. Dahle, R. J. Davis, P. de Bernardis, G. de Gasperis, G. de Zotti, J. Delabrouille, J. Démoclès, F.-X. Désert, J. M. Diego, K. Dolag, H. Dole, S. Donzelli, O. Doré, M. Douspis, X. Dupac, T. A. Enßlin, H. K. Eriksen, F. Finelli, I. Flores-Cacho, O. Forni, P. Fosalba, M. Frailis, S. Fromenteau, S. Galeotta, K. Ganga, R. T. Génova-Santos, M. Giard, J. González-Nuevo, R. González-Riestra, K. M. Górski, A. Gregorio, A. Gruppuso, F. K. Hansen, D. Harrison, A. Hempel, C. Hernández-Monteagudo, D. Herranz, S. R. Hildebrandt, A. Hornstrup, K. M. Huffenberger, G. Hurier, T. Jagemann, J. Jasche, M. Juvela, E. Keihänen, R. Keskitalo, T. S. Kisner, R. Kneissl, J. Knoche, L. Knox, H. Kurki-Suonio, G. Lagache, A. Lähteenmäki, J.-M. Lamarre, A. Lasenby, C. R. Lawrence, S. Leach, R. Leonardi, A. Liddle, P. B. Lilje, M. López-Caniego, G. Luzzi, J. F. Macías-Pérez, D. Maino, N. Mandolesi, R. Mann, F. Marleau, D. J. Marshall, E. Martínez-González, S. Masi, M. Massardi, S. Matarrese, F. Matthai, P. Mazzotta, P. R. Meinhold, A. Melchiorri, J.-B. Melin, L. Mendes, A. Mennella, M.-A. Miville-Deschênes, A. Moneti, L. Montier, G. Morgante, D. Mortlock, D. Munshi, P. Naselsky, P. Natoli, H. U. Nørgaard-Nielsen, F. Noviello, S. Osborne, F. Pasian, G. Patanchon, O. Perdereau, F. Perrotta, F. Piacentini, E. Pierpaoli, S. Plaszczynski, P. Platania, E. Pointecouteau, G. Polenta, N. Ponthieu, L. Popa, T. Poutanen, G. W. Pratt, J.-L. Puget, J. P. Rachen, R. Rebolo, M. Reinecke, M. Remazeilles, C. Renault, S. Ricciardi, T. Riller, I. Ristorcelli, G. Rocha, C. Rosset, M. Rossetti, J. A. Rubiño-Martín, B. Rusholme, M. Sandri, G. Savini, B. M. Schaefer, D. Scott, G. F. Smoot, J.-L. Starck, F. Stivoli, R. Sunyaev, D. Sutton, J.-F. Sygnet, J. A. Tauber, L. Terenzi, L. Toffolatti, M. Tomasi, M. Tristram, L. Valenziano, B. Van Tent, P. Vielva, F. Villa, N. Vittorio, B. D. Wandelt, J. Weller, S. D. M. White, D. Yvon, A. Zacchei, and A. Zonca. Planck intermediate results. I. Further validation of

new Planck clusters with XMM-Newton. *Astronomy and Astrophysics*, 543:A102, July 2012.

[205] 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.

[206] Steve Plimpton. Fast parallel algorithms for short-range molecular dynamics. *J. Comput. Phys.*, 117(1):1–19, 1995.

[207] C. Power, J. F. Navarro, A. Jenkins, C. S. Frenk, S. D. M. White, V. Springel, J. Stadel, and T. Quinn. The inner structure of ΛCDM haloes - I. A numerical convergence study. *Monthly Notices of the Royal Astronomical Society*, 338:14–34, January 2003.

[208] D. Reed, J. Gardner, T. Quinn, J. Stadel, M. Fardal, G. Lake, and F. Governato. Evolution of the mass function of dark matter haloes. *MNRAS*, 346:565–572, December 2003.

[209] D.K. Remler and P.A. Madden. Molecular Dynamics without effective potentials via the Car-Parrinello approach. *Mol. Phys.*, 70:921, (1990).

[210] E. R. Rodrigues, P. O. A. Navaux, J Panetta, and C. L. Mendes. A new technique for data privatization in user-level threads and its use in parallel applications. In *ACM 25th Symposium On Applied Computing (SAC), Sierre, Switzerland*, 2010.

[211] Eduardo R. Rodrigues, Philippe O. A. Navaux, Jairo Panetta, Alvaro Fazenda, Celso L. Mendes, and Laxmikant V. Kalé. A comparative analysis of load balancing algorithms applied to a weather forecast model. In *Proceedings of 22nd IEEE International Symposium on Computer Architecture and High Performance Computing*, Petrópolis - Brazil, 2010.

[212] Eduardo R. Rodrigues, Philippe O. A. Navaux, Jairo Panetta, Celso L. Mendes, and Laxmikant V. Kalé. Optimizing an MPI weather forecasting model via processor virtualization. In *Proceedings of IEEE International Conference on High Performance Computing (HiPC 2010)*, Goa - India, 2010.

[213] D. Romero, B. Meeder, and J. Kleinberg. Differences in the Mechanics of Information Diffusion Across Topics: Idioms, Political Hashtags, and Complex Contagion on Twitter. In *Proceedings of the 20th International World Wide Web Conference (WWW 2011)*, 2011.

[214] Michiel Ronsse and Koen De Bosschere. RecPlay: a fully integrated practical record/replay system. *ACM Trans. Comput. Syst.*, 17(2):133–152, 1999.

[215] H.G. Rotithor. Taxonomy of dynamic task scheduling schemes in distributed computing systems. In *Proceedings of IEE: Computers and Digital Techniques*, volume 141, pages 1–10, 1994.

[216] J. J. Ruan, T. R. Quinn, and A. Babul. The Observable Thermal and Kinetic Sunyaev-Zel'dovich Effect in Merging Galaxy Clusters. *ArXiv e-prints*, April 2013.

[217] Ruth Rutter. Run-length encoding on graphics hardware. Master's thesis, University of Alaska at Fairbanks, 2011.

[218] J. K. Salmon and M. S. Warren. Skeletons from the treecode closet. *Journal of Computational Physics*, 111:136–155, March 1994.

[219] Yanhua Sun Sameer Kumar and L. V. Kale. Acceleration of an Asynchronous Message Driven Programming Paradigm on IBM Blue Gene/Q. In *Proceedings of 26th IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, Boston, USA, May 2013.

[220] Osman Sarood and Laxmikant V. Kalé. A 'cool' load balancer for parallel applications. In *Proceedings of the 2011 ACM/IEEE conference on Supercomputing*, Seattle, WA, November 2011.

[221] Martin Schulz, Jim Galarowicz, Don Maghrak, William Hachfeld, David Montoya, and Scott Cranford. Open|speedshop: An open source infrastructure for parallel performance analysis. *Scientific Programming*, 16(2-3):105–121, 2008.

[222] Melih Sener, Johan Strumpfer, John A. Timney, Arvi Freiberg, C. Neil Hunter, and Klaus Schulten. Photosynthetic vesicle architecture and constraints on efficient energy harvesting. *Biophysical Journal*, 99:67–75, 2010.

[223] D. Shakhvorostov, R.A. Nistor, L. Krusin-Elbaum, G.J. Martyna, D.M. Newns, B.G. Elmegreen, X. Liu, Z.E. Hughesa, S. Paul, C. Cabral, S. Raoux, D.B. Shrekenhamerd, D.N. Basovd, Y. Songe, and M.H. Mueser. Evidence for electronic gap-driven metal-semiconductor transition in phase-change materials. *PNAS.*, 106:10907–10911, (2009).

[224] S. Shende and A. D. Malony. The TAU Parallel Performance System. *International Journal of High Performance Computing Applications*, 20(2):287–331, Summer 2006.

[225] S.A. Shevlin, A. Curioni, and W. Andreoni. Ab Initio Design of High-k Dielectrics: $La_xY_{1-x}AlO_3$. *Phys. Rev. Lett.*, 94:146401, (2005).

[226] S. Shingu, H. Takahara, H. Fuchigami, M. Yamada, Y. Tsuda, W. Ohfuchi, Y. Sasaki, K. Kobayashi, T. Hagiwara, S. Habata, et al. A 26.58 tflops global atmospheric simulation with the spectral transform method

on the earth simulator. In *Proceedings of the 2002 ACM/IEEE conference on Supercomputing*, pages 1–19. IEEE Computer Society Press, 2002.

[227] D. Siegel. Social networks and collective action. *Americal Journal of Political Science*, 53:122–138, 2009.

[228] A. Sinha and L.V. Kalé. Information Sharing Mechanisms in Parallel Programs. In H.J. Siegel, editor, *Proceedings of the 8th International Parallel Processing Symposium*, pages 461–468, Cancun, Mexico, April 1994.

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

[230] Edgar Solomonik and Laxmikant V. Kale. Highly Scalable Parallel Sorting. In *Proceedings of the 24th IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, April 2010.

[231] R. Souto, RB Avila, POA Navaux, MX Py, N. Maillard, T. Diverio, HC Velho, S. Stephany, AJ Preto, J. Panetta, et al. Processing mesoscale climatology in a grid environment. In *Proceedings of the Seventh IEEE International Symposium on Cluster Computing and the Grid–CCGrid*, 2007.

[232] V. Springel. The cosmological simulation code GADGET-2. *MNRAS*, 364:1105–1134, December 2005.

[233] V. Springel, J. Wang, M. Vogelsberger, A. Ludlow, A. Jenkins, A. Helmi, J. F. Navarro, C. S. Frenk, and S. D. M. White. The Aquarius Project: the subhaloes of galactic haloes. *MNRAS*, 391:1685–1711, December 2008.

[234] V. Springel, S. D. M. White, A. Jenkins, C. S. Frenk, N. Yoshida, L. Gao, J. Navarro, R. Thacker, D. Croton, J. Helly, J. A. Peacock, S. Cole, P. Thomas, H. Couchman, A. Evrard, J. Colberg, and F. Pearce. Simulations of the formation, evolution and clustering of galaxies and quasars. *Nature*, 435:629–636, June 2005.

[235] J. Stadel, D. Potter, B. Moore, J. Diemand, P. Madau, M. Zemp, M. Kuhlen, and V. Quilis. Quantifying the heart of darkness with GHALO - a multibillion particle simulation of a galactic halo. *MNRAS*, 398:L21–L25, September 2009.

[236] J. G. Stadel. *Cosmological N-body Simulations and their Analysis*. PhD thesis, Department of Astronomy, University of Washington, March 2001.

[237] Yanhua Sun, Gengbin Zheng, Chao Mei Eric J. Bohm, Terry Jones, Laxmikant V. Kalé, and James C.Phillips. Optimizing fine-grained communication in a biomolecular simulation application on cray xk6. In *Proceedings of the 2012 ACM/IEEE conference on Supercomputing*, Salt Lake City, Utah, November 2012.

[238] Yanhua Sun, Gengbin Zheng, L. V. Kale, Terry R. Jones, and Ryan Olson. A uGNI-based Asynchronous Message-driven Runtime System for Cray Supercomputers with Gemini Interconnect. In *Proceedings of 26th IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, Shanghai, China, May 2012.

[239] Emad Tajkhorshid, Aleksij Aksimentiev, Ilya Balabin, Mu Gao, Barry Isralewitz, James C. Phillips, Fangqiang Zhu, and Klaus Schulten. Large scale simulation of protein mechanics and function. In Frederic M. Richards, David S. Eisenberg, and John Kuriyan, editors, *Advances in Protein Chemistry*, volume 66, pages 195–247. Elsevier Academic Press, New York, 2003.

[240] Emad Tajkhorshid, Peter Nollert, Morten Ø. Jensen, Larry J. W. Miercke, Joseph O'Connell, Robert M. Stroud, and Klaus Schulten. Control of the selectivity of the aquaporin water channel family by global orientational tuning. *Science*, 296:525–530, 2002.

[241] Claudia Taylor, Achla Marathe, and Richard Beckman. Same influenza vaccination strategies but different outcomes across us cities? *International Journal of Infectious Diseases*, 14(9):e792 – e795, 2010.

[242] T.N. Theis and P.M. Solomon. In Quest of the "Next Switch": Prospects for Greatly Reduced Power Dissipation in a Successor to the Silicon Field-Effect Transistor. *Proceedings of the IEEE*, 98:2005, (2010).

[243] GJ Tripoli and WR Cotton. The Colorado State University three-dimensional cloud/mesoscale model. Technical Report 3, Atmos, 1982.

[244] M. Tuckerman, G. Martyna, M.L. Klein, and B.J. Berne. Efficient Molecular Dynamics and Hybrid Monte Carlo Algorithms for Path Integrals. *J. Chem. Phys.*, 99:2796, (1993).

[245] Ramkumar V. Vadali, Yan Shi, Sameer Kumar, L. V. Kale, Mark E. Tuckerman, and Glenn J. Martyna. Scalable fine-grained parallelization of plane-wave-based ab initio molecular dynamics for large supercomputers. *Journal of Comptational Chemistry*, 25(16):2006–2022, Oct. 2004.

[246] J. W. Wadsley, J. Stadel, and T. Quinn. Gasoline: a flexible, parallel implementation of TreeSPH. *New Astronomy*, 9:137–158, February 2004.

[247] R.L. Walko, L.E. Band, J. Baron, T.G.F. Kittel, R. Lammers, T.J. Lee, D. Ojima, R.A. Pielke Sr, C. Taylor, C. Tague, et al. Coupled atmosphere–biophysics–hydrology models for environmental modeling. *Journal of Applied Meteorology*, 39(6), 2000.

[248] Yuhe Wang and John Killough. A new approach to load balance for parallel compositional simulation based on reservoir model over-decomposition. In *2013 SPE Reservoir Simulation Symposium*, 2013.

[249] M. S. Warren and J. K. Salmon. A parallel hashed oct-tree n-body algorithm. In *Proceedings of the 1993 ACM/IEEE conference on Supercomputing*, Supercomputing '93, pages 12–21, New York, NY, USA, 1993. ACM.

[250] M.S. Warren, J.K. Salmon, D.J. Becker, M.P. Goda, T. Sterling, and W. Winckelmans. Pentium pro inside: I. a treecode at 430 gigaflops on asci red, ii. price/performance of $50/mflop on loki and hyglac. In *Supercomputing, ACM/IEEE 1997 Conference*, page 61, nov. 1997.

[251] S. D. M. White, C. S. Frenk, and M. Davis. Clustering in a neutrino-dominated universe. *Astrophys. J. Lett.*, 274:L1–L5, November 1983.

[252] X.-P. Xu and A. Needleman. Numerical simulation of fast crack growth in brittle solids. *Journal of the Mechanics and Physics of Solids*, 42:1397–1434, 1994.

[253] M. Xue, K.K. Droegemeier, and D. Weber. Numerical Prediction of High-Impact Local Weather: A Driver for Petascale Computing. *Petascale Computing: Algorithms and Applications*, pages 103–124, 2007.

[254] Jae-Seung Yeom, Abhinav Bhatele, Keith Bisset, Eric Bohm, Abhishek Gupta, Laxmikant V. Kale, Madhav Marathe, Dimitrios S. Nikolopoulos, Martin Schulz, and Lukasz Wesolowski. Overcoming the scalability challenges of contagion simulations on Blue Waters. Technical Report 13-057, NDSSL, Virginia Bioinformatics Institute at Virginia Tech, 2013.

[255] Y. B. Zeldovich and R. A. Sunyaev. The Interaction of Matter and Radiation in a Hot-Model Universe. *Astrophysics & Space Science*, 4:301–316, July 1969.

[256] Gongpu Zhao, Juan R. Perilla, Ernest L. Yufenyuy, Xin Meng, Bo Chen, Jiying Ning, Jinwoo Ahn, Angela M. Gronenborn, Klaus Schulten, Christopher Aiken, and Peijun Zhang. Mature HIV-1 capsid structure by cryo-electron microscopy and all-atom molecular dynamics. *Nature*, 497:643–646, 2013. doi:10.1038/nature12162.

[257] Gengbin 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.

[258] Gengbin Zheng, Abhinav Bhatele, Esteban Meneses, and Laxmikant V. Kale. Periodic Hierarchical Load Balancing for Large Supercomputers. *International Journal of High Performance Computing Applications (IJHPCA)*, March 2011.

[259] Gengbin Zheng, Orion Sky Lawlor, and Laxmikant V. Kalé. Multiple flows of control in migratable parallel programs. In *2006 International Conference on Parallel Processing Workshops (ICPPW'06)*, pages 435–444, Columbus, Ohio, August 2006. IEEE Computer Society.

[260] Gengbin Zheng, Xiang Ni, and L. V. Kale. A Scalable Double In-memory Checkpoint and Restart Scheme towards Exascale. In *Proceedings of the 2nd Workshop on Fault-Tolerance for HPC at Extreme Scale (FTXS)*, Boston, USA, June 2012.

[261] Gengbin Zheng, Lixia Shi, and Laxmikant V. Kalé. FTC-Charm++: An In-Memory Checkpoint-Based Fault Tolerant Runtime for Charm++ and MPI. In *2004 IEEE International Conference on Cluster Computing*, pages 93–103, San Diego, CA, September 2004.

[262] Gengbin Zheng, Terry Wilmarth, Praveen Jagadishprasad, and Laxmikant V. Kalé. Simulation-based performance prediction for large parallel machines. In *International Journal of Parallel Programming*, volume 33, pages 183–207, 2005.