Millisecond-Scale Molecular Dynamics Simulations on Anton

David E. Shaw*, Ron O. Dror, John K. Salmon, J.P. Grossman, Kenneth M. Mackenzie, Joseph A. Bank, Cliff Young, Martin M. Deneroff, Brannon Batson, Kevin J. Bowers, Edmond Chow, Michael P. Eastwood, Douglas J. Ierardi, John L. Klepeis, Jeffrey S. Kuskin, Richard H. Larson, Kresten Lindorff-Larsen, Paul Maragakis, Mark A. Moraes, Stefano Piana, Yibing Shan, and Brian Towles

D. E. Shaw Research, New York, NY 10036, USA

*Correspondence: David.Shaw@DEShawResearch.com

ABSTRACT

Anton is a recently completed special-purpose supercomputer designed for molecular dynamics (MD) simulations of biomolecular systems. The machine's specialized hardware dramatically increases the speed of MD calculations, making possible for the first time the simulation of biological molecules at an atomic level of detail for periods on the order of a millisecond—about two orders of magnitude beyond the previous state of the art. Anton is now running simulations on a timescale at which many critically important, but poorly understood phenomena are known to occur, allowing the observation of aspects of protein dynamics that were previously inaccessible to both computational and experimental study. Here, we report Anton's performance when executing actual MD simulations whose accuracy has been validated against both existing MD software and experimental observations. We also discuss the manner in which novel algorithms have been coordinated with Anton's co-designed, application-specific hardware to achieve these results.

1. INTRODUCTION

Classical molecular dynamics (MD) simulations of biological molecules give scientists the ability to trace atomic motions and have helped yield deep insights into molecular mechanisms that experimental approaches could not have achieved alone [20, 21, 30]. MD has been limited, however, by the rate at which these simulations can be performed on current computer hardware. With discrete time steps on the order of femtoseconds (10^{-15} seconds), a particular challenge is to observe, *in silico*, functionally important biological events that typically occur on timescales of many microseconds or milliseconds, including the "folding" of proteins into their native three-dimensional structures, the structural changes that underlie protein function, and the interactions between two proteins or between a protein and a candidate drug molecule. Such long-timescale simulations pose a much greater challenge than simulations of larger chemical systems at more moderate timescales, because increasing chemical system size allows for the exploitation of parallelism that is not available when extending a simulation in time.

A variety of projects have focused on accelerating and parallelizing MD, both in general-purpose computers [1, 2, 9, 14], and in

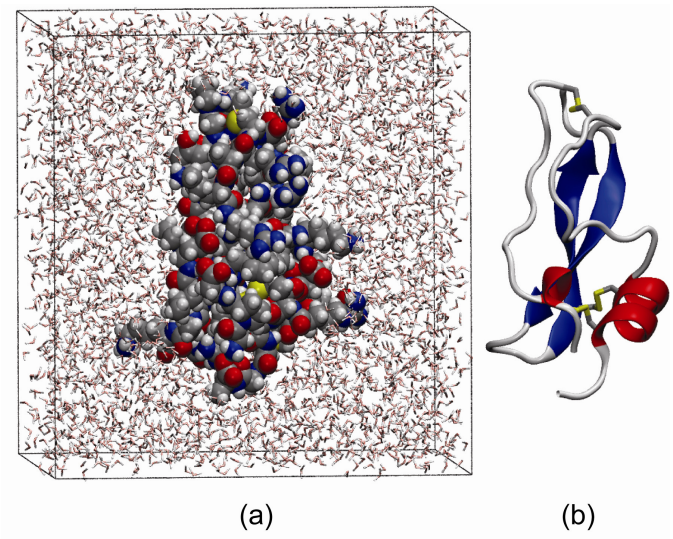

Figure 1: Two renderings of a protein (BPTI) taken from a molecular dynamics simulation on Anton. (a) The entire simulated system, with each atom of the protein represented by a sphere and the surrounding water represented by thin lines. For clarity, water molecules in front of the protein are not pictured. (b) A "cartoon" rendering showing important structural elements of the protein (secondary and tertiary structure).

SC09 November 14-20, 2009, Portland, Oregon, USA.

Length (μs)	Protein	Hardware	Software	Citation
1031	BPTI	Anton	[native]	Here
236	gpW	Anton	[native]	Here
10	WW domain	x86 cluster	NAMD	[10]
2	villin HP-35	x86	GROMACS	[6]
2	rhodopsin	Blue Gene/L	Blue Matter	[25]
2	rhodopsin	Blue Gene/L	Blue Matter	[12]
2	β_2AR	x86 cluster	Desmond	[5]

Table 1: The longest (to our knowledge) published all-atom MD simulations of proteins in explicitly represented water.

specialized hardware [8, 27]. Recently, efforts to extend simulations to longer timescales have concentrated primarily on commodity architectures [1, 2, 14]. Important scientific and medical breakthroughs involving MD simulation may, however, come more quickly into range if it is possible to break off from the curve of commodity-based supercomputing. We have designed Anton, a massively-parallel, specialized supercomputer, to perform MD computations today that might not otherwise be possible for years.

This paper presents initial performance results from Anton and describes the relationships between algorithms and hardware that lead to its high performance. Anton enables scientists to perform simulations at rates above 15 microseconds (μs) per day—roughly two orders of magnitude greater than the rates typically achieved by state-of-the-art parallel simulation packages running on the fastest general-purpose machines at practical levels of parallelism and efficiency (Section 5.1).

The first 512-node Anton machine became operational in October 2008, and has already completed an all-atom simulation of the protein BPTI (Figure 1) spanning more than a millisecond of biological time, along with several other simulations of up to 236 μs (Section 5.3). By way of comparison, the longest all-atom protein simulation previously reported in the literature, which was performed using the MD code NAMD, was 10 μs in length [10]; few others have reached 2 μs (Table 1).

Anton's performance advantage over other systems is the result of an algorithm/hardware co-design process in which the choice of algorithms impacted the design of the hardware, and vice versa (Section 3). The algorithms that work efficiently with Anton's specialized hardware, including some that were created specifically for Anton, differ in a number of ways from those that have proven most effective on general-purpose architectures. We designed algorithms, for example, to increase the fraction of the computational workload that can be performed on fast, hardwired arithmetic pipelines. We also developed parallelization and data choreography methods that are closely tied to the distinctive characteristics of Anton's various computational units and of its communication mechanisms. In addition, we used customized-precision fixed-point arithmetic both to reduce the area requirements of arithmetic units and to efficiently produce simulation results that are deterministic, time reversible, and independent of the number of compute nodes (Section 4).

We have validated Anton's results by comparison to both existing MD software and experimental observations (Section 5.2). A more difficult question is the extent to which the physical models

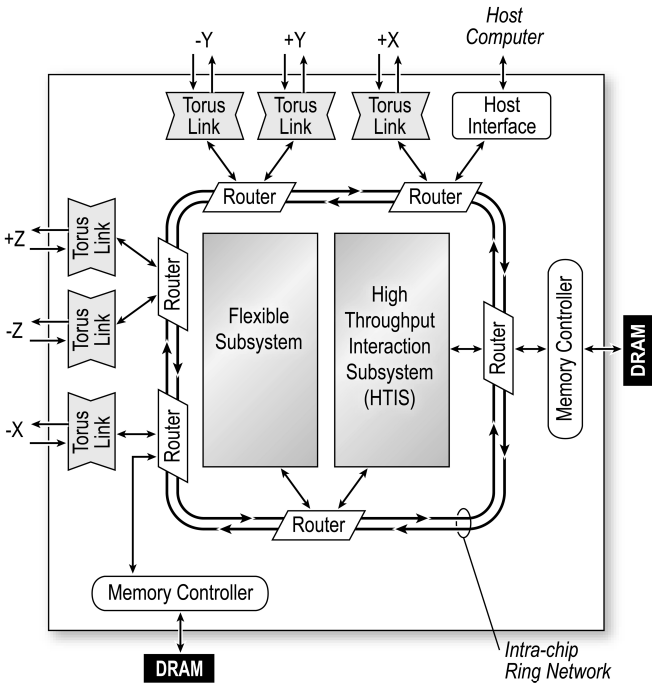

Figure 2: Anton ASIC block diagram.

typically employed in calculating inter-atomic forces permit the behavior of biomolecular systems to be elucidated over the very long timescales that, with the introduction of Anton, are now accessible. This question has not yet been fully answered, but our tests so far suggest that MD simulations may ultimately provide important new insights into biological processes occurring on these timescales, and that such simulations may play a key role in testing and improving these physical models.

2. BACKGROUND

In this section, we summarize the components of MD computation salient to our discussion of Anton. Also, to make this paper self-contained, we briefly review the major components of the Anton architecture, which have been presented in more detail elsewhere [22, 23, 33].

2.1 Molecular Dynamics

A molecular dynamics time step consists of a computationally intensive force calculation for each atom of a chemical system, followed by a less expensive integration step that advances the positions of the atoms according to classical laws of motion. Forces are calculated based on a model known as a *force field*. Commonly used biomolecular force fields express the total force on an atom as a sum of three components: 1) bonded forces, which involve interactions between small groups of atoms connected by one or more covalent bonds, 2) van der Waals forces, which include interactions between all pairs of atoms in the system, but which fall off quickly with distance and are typically only evaluated for nearby pairs of atoms, and 3) electrostatic forces, which include interactions between all pairs of atoms and fall off slowly with distance.

Electrostatic forces are typically computed by one of several fast, approximate methods that account for long-range effects without requiring the explicit interaction of all pairs of atoms. The most widely used methods for such fast electrostatics calculations employ the Ewald decomposition, which divides electrostatic interactions into two contributions. The first decays rapidly with distance, and is thus computed directly for all atom pairs separated by less than a cutoff. This contribution and the van der Waals interactions together constitute the *range-limited interactions*. The second contribution (*long-range interactions*) decays more slowly, but can be expressed as a convolution that can be efficiently computed by taking the fast Fourier transform (FFT) of the charge distribution on a regular mesh, multiplying by an appropriate function in Fourier space, and then performing an inverse FFT [7, 15]. Charge must be mapped from atoms to nearby mesh points before the FFT computation (*charge spreading*), and forces on atoms must be calculated from the convolution result at nearby mesh points after the inverse FFT computation (*force interpolation*).

2.2 Overview of the Anton Architecture

Anton comprises a set of nodes connected in a toroidal topology; the 512-node machines discussed (along with certain smaller and larger configurations) in this paper, for example, have an 8×8×8 toroidal topology, corresponding to an 8×8×8 partitioning of a chemical system with periodic boundary conditions. Each node includes an ASIC with two major computational subsystems (Figure 2). The first is the *high-throughput interaction subsystem* (HTIS) designed for computing massive numbers of range-limited pairwise interactions of various forms using an array of 32 hardwired *pairwise point interaction pipelines* (PPIPs). The PPIPs can compute interactions with a number of different functional forms, as determined by various mode settings and user-specified lookup tables and parameter values. The second is the flexible subsystem, which is composed of programmable cores used for the remaining, less structured part of the MD calculation. The flexible subsystem contains eight geometry cores (GCs) that were designed in-house to perform fast numerical computations, four Tensilica LX processors that control the overall data flow in the Anton system, and four data transfer engines that allow communication to be hidden behind computation. The Anton ASIC also contains a pair of DDR2-800 DRAM controllers, six high-speed (50.6 Gbit/s per direction) channels that provide communication to neighboring ASICs, and a host interface that communicates with an external host computer for input, output, and general control of the Anton system. These units are connected by a bidirectional on-chip communication ring. The ASICs are implemented in 90-nm technology and clocked at 485 MHz, with the exception of the PPIP array in the HTIS, which is clocked at 970 MHz.

3. CO-DESIGN OF ALGORITHMS AND HARDWARE

The algorithms and numerical techniques employed by Anton were co-designed with its hardware architecture. Some of these algorithms were developed specifically for Anton, while others were adapted to exploit its specialized hardware. This section describes these algorithms and their impact on Anton's hardware design.

3.1 Maximizing the Use of Specialized Hardware

To accelerate a computation with specialized hardware, one typically begins by looking for an "inner loop" that accounts for the majority of the computational time. Table 2 shows an execution time profile of the widely used GROMACS MD package [14] on an x86 core. In a typical simulation, the x86 core spends 64% of its time computing range-limited forces—electrostatic and van der Waals interactions between pairs of atoms separated by less than 9 Å. These computations are mapped to Anton's PPIPs, which accelerate the computations by over two orders of magnitude relative to a conventional processor core [23]. While the programmable cores of Anton's flexible subsystem could accelerate the remaining 36% of the workload to a certain extent, algorithmic changes allow us to better leverage Anton's specialized hardware.

The second-largest contribution to the execution time on the x86 is the Fourier computation (14%); this includes both the forward and inverse FFTs. The time required for Fourier computation scales with the number of mesh points used for the FFT, which is

	x86 core		Anton	
	small cutoff (9 Å) fine mesh (64^3)	**large cutoff (13 Å) coarse mesh (32^3)**	**small cutoff (9 Å) fine mesh (64^3)**	**large cutoff (13 Å) coarse mesh (32^3)**
Nonbonded forces				
Range-limited forces	56.6 ms (64%)	164.4 ms (89%)	1.4 μs (4%)	1.9 μs (12%)
FFT & inverse FFT	12.3 ms (14%)	1.4 ms (1%)	24.7 μs (63%)	8.9 μs (58%)
Mesh interpolation	9.6 ms (11%)	8.8 ms (5%)	9.5 μs (24%)	2.0 μs (13%)
Correction forces	4.0 ms (5%)	3.8 ms (2%)	2.5 μs (6%)	2.5 μs (16%)
Bonded forces	2.7 ms (3%)	2.7 ms (1%)	3.5 μs (9%)	4.1 μs (27%)
Integration	3.4 ms (4%)	3.4 ms (2%)	1.6 μs (4%)	1.6 μs (10%)
Total	88.5 ms (100%)	184.5 ms (100%)	39.2 μs (100%)	15.4 μs (100%)

Table 2: Effect of electrostatics parameters on performance, illustrated by execution time profiles for GROMACS on a single x86 core (a 2.66-GHz Xeon X5550 [Nehalem] core) and for Anton, for one time step of the DHFR benchmark system (see Section 5.1). By comparison with a conventional processor, Anton tends to achieve better performance when using a larger cutoff and a coarser mesh. Shading represents tasks that Anton accelerates using special-purpose pipelines. The individual Anton task times, which were measured on one node of a 512-node machine, sum up to more than the total time per time step because certain tasks are performed in parallel. Anton's average execution time per time step is smaller than that shown here because long-range interactions are typically evaluated only every two or three time steps. Anton's overall performance is discussed and compared to that of other parallel platforms in Section 5.1.

determined by tunable parameters in the Ewald decomposition. This decomposition represents the electrostatic potential due to a point charge as a sum of two components: one that falls rapidly to zero as distance from the charge increases, and one that is smooth throughout space and can thus be computed on a mesh. Increasing the smoothness of the second component allows the use of a coarser mesh, but also makes the first component fall to zero less rapidly so that an increased cutoff is required for the range-limited interactions. Because Anton has a much greater performance advantage over conventional processors for the range-limited interactions than for the Fourier computation, Anton's performance is optimized by choosing a larger cutoff and a coarser mesh than are typically used on commodity hardware. On the x86, this parameter change leads to an overall slowdown of nearly twofold, whereas on Anton, it results in a *speedup* of more than twofold (Table 2).

Charge spreading and force interpolation are the next most computationally intensive tasks on the x86. We observed that these mesh interpolation operations could be formulated as "interactions" between atoms and nearby mesh points. In charge spreading, for example, the charge assigned to each mesh point can be written as a sum over the nearby atoms, with the contribution of each atom depending on the charge of that atom and its position relative to the mesh point. This is similar to the range-limited computation performed by the HTIS, with one complication: most high-performance codes use the *Smooth Particle Mesh Ewald* (SPME) algorithm, in which the interaction between an atom and a mesh point is based on B-spline interpolation [7]. Anton's PPIPs, on the other hand, compute interactions between two points as a table-driven function of the distance between them—a radially symmetric functional form that is incompatible with B-splines. We addressed this problem by employing a method we developed called *Gaussian Split Ewald* (GSE) [31], which uses radially symmetric interaction functions—Gaussians—for charge spreading and force interpolation. By using GSE instead of SPME or other mesh-based Ewald methods (e.g., PPPM [15]), we were able to map charge spreading and force interpolation to the HTIS with minimal hardware modification.

In most force fields, the electrostatic and van der Waals forces between pairs of atoms separated by one to three covalent bonds are eliminated or scaled down. The long-range interactions include contributions from these pairs, which must be computed separately as *correction forces* and subtracted out. The PPIP is suited for the computation of these forces, but the HTIS is not: it is designed for interactions between large sets of atoms rather than single-pair interactions. We thus included the *correction pipeline*—a PPIP with the necessary control logic to process a list of atom pairs—in Anton's flexible subsystem.

Of the tasks listed in Table 2, only the Fourier computation, bonded force evaluation and integration are mapped to Anton's flexible cores. These tasks together constitute only 4% of the x86's execution profile with typical Anton parameters; the remaining workload is mapped to Anton's fast, specialized PPIPs. Through a combination of novel algorithms, parameter optimization, and hardware optimization, we significantly reduced the computational workload mapped to Anton's programmable cores.

3.2 Choreographing the Flow of Data

Anton's performance depends on its ability to keep the many arithmetic units on each node busy with useful computation. To achieve this, the various data flows within and between each ASIC must be carefully planned. Here again, the optimal choice of algorithms reflects important differences between Anton and general-purpose hardware. First, a conventional processor typically has a single logical memory that can be used to store or buffer all of the data on which the processor must operate. By contrast, each computational subunit on an Anton ASIC has its own low-latency dedicated memory. Intra-node data transfers between these subunits are carefully choreographed to minimize data movement and to deliver data just when it is needed; this strategy requires algorithmic choices that result in a predictable pattern of data movement. Second, in commodity clusters, inter-node communication latency is measured in microseconds, and per-message overheads disfavor techniques that use many small messages. On Anton, inter-node latency is tens of nanoseconds, and messages with as little as four bytes of data can be sent efficiently, allowing the use of algorithms that require many short messages. Indeed, a typical time step on Anton involves thousands of inter-node messages per ASIC.

The remainder of this section describes inter- and intra-node parallelization methods used by Anton for each of its major computational tasks. Most of Anton's communication patterns take advantage of spatial locality: with the exception of the Fourier transforms and certain global reductions, all of the computations involve sets of particles separated by no more than around 15 Å. For this reason, Anton distributes particle data across nodes using a spatial decomposition, in which the space to be simulated is divided into a regular grid of *boxes*, and each node updates the positions and momenta of atoms in one box, referred to as the *home box* of that node and of those atoms (we also refer to the node containing an atom's home box as the *home node* of that atom). Section 3.2.4 discusses minor modifications to this scheme that may cause atoms located near a box boundary to be assigned to a node responsible for a neighboring box.

3.2.1 Range-Limited Interactions

High-performance MD codes for conventional processors typically organize the computation of range-limited interactions by assembling a *pair list*, which specifies pairs of atoms to be interacted on a processor. On Anton, where the 32 PPIPs in each HTIS can sustain an aggregate input plus output data rate in excess of 10 Tbit/s, a pair list approach that transferred the inputs and outputs of each pipeline between the HTIS and other subsystems would require communication bandwidth beyond the limits of current technology. The HTIS instead uses a systolic-array-like architecture that considers all possible interactions between two sets of atoms [23].

To map the range-limited interactions onto this architecture, Anton implements a variant of the *NT method*, a technique developed in 2005 by Shaw [32] for parallelizing the range-limited *N*-body problem. In this method, each node computes interactions between atoms in a *tower* region and atoms in a *plate* region (Figure 3a). Both of these regions contain the home box, as well as atoms imported from other boxes. Once the interactions have been computed, the resulting forces on atoms in the tower and plate are sent back to the nodes on which those atoms reside.

For typical chemical system sizes, the region from which each node imports atom positions and exports computed forces (the *import region*) is smaller for the NT method than for a traditional spatial decomposition method (Figure 3b); the NT method thus

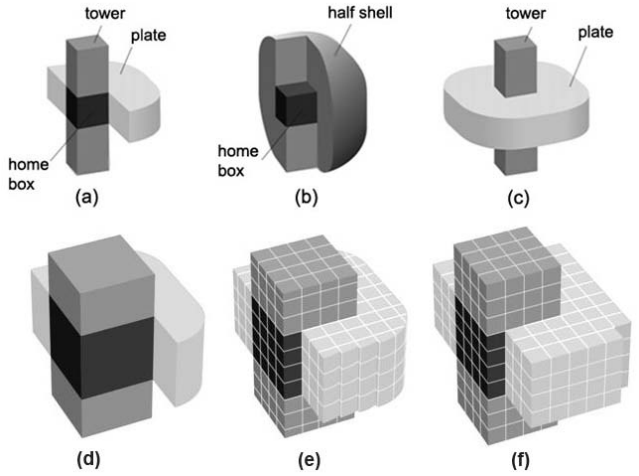

Figure 3: (a–c) Import regions associated with several parallelization methods for range-limited pairwise interactions. (a) In the NT method, each node computes interactions between atoms in a tower region and atoms in a plate region. The asymmetry of the plate region reflects the fact that the interaction between a pair of particles need only be computed once, yielding equal and opposite forces on both particles. (b) In a more traditional parallelization method, each node computes interactions between atoms in its home box and atoms in a larger "half-shell" region, which includes the home box. (c) A variant of the NT method used for charge spreading and force interpolation. A larger, symmetrical plate region is required because these calculations involve interactions between particles and mesh points rather than between pairs of particles. (d–f) Adapting the NT method to Anton. (d) The original NT method, for a chemical system larger than that of (a). (e) The use of subboxes leads to an expansion of the plate region, because the union of the subbox plates is larger than the original plate. (f) On Anton, the import region consists of whole subboxes.

leads to a more communication-efficient implementation, an advantage that grows asymptotically as the level of parallelism increases. In addition, because the two sets of atoms to be interacted are closer in size in the NT method, the ratio of computation to communication bandwidth within each HTIS is higher, so the NT method uses on-chip communication resources more effectively. The NT method is one of a number of *neutral territory* methods—parallelization schemes in which the interaction between two atoms may be computed by a node on which neither resides [2, 3, 11, 19, 29].

In the absence of appropriate countermeasures, one efficiency-limiting factor associated with the NT method arises from the fact that not all atoms in the tower need to interact with all atoms in the plate; many atom pairs, for example, exceed the cutoff radius. In order to achieve high PPIP utilization, each PPIP is thus fed by eight *match units*, which consider pairs of atoms and determine whether they may be required to interact. A given plate atom can be tested against eight tower atoms in a single cycle, with pairs that pass this test moving through a concentrator that feeds the PPIP input queue. As long as the average number of such pairs per cycle per PPIP is at least one, the PPIPs will approach full utilization.

As the chemical system size increases, the NT method's match efficiency (defined as the ratio of necessary interactions to pairs of atoms considered) falls to a point where even eight match units cannot keep a PPIP occupied (Table 3). We address this issue by

Match efficiency of NT method				
		Subboxes per box		
		1×1×1	2×2×2	4×4×4
Box side length	8 Å	25%	40%	51%
	16 Å	12%	25%	40%
	32 Å	4%	12%	25%

Table 3: Match efficiency of the NT method for several box sizes, each divided into 1 (1×1×1), 8 (2×2×2), or 64 (4×4×4) subboxes. These figures assume a 13-Å cutoff radius.

dividing each home box into a regular array of subboxes, and applying the NT method separately to each one. The use of subboxes significantly increases match efficiency and thus PPIP utilization (Table 3), at the cost of slightly enlarging a node's total import region (Figure 3e). The effective import region is enlarged further (Figure 3f) to take advantage of Anton's multicast mechanism, which sends all atoms in a given subbox to the same set of nodes.

A variant of the NT method is also used to parallelize the charge spreading and force interpolation operations, with the HTIS computing interactions between atoms in the tower and mesh points in the plate. Because of the asymmetric nature of these interactions, the plate region must be enlarged relative to that used for range-limited interactions (Figure 3c). In addition, mesh point positions are regular and fixed, so each node can simply compute them locally rather than importing them. To perform charge spreading, for example, each node imports position data for atoms in the tower, computes interactions with mesh points in the plate, and then exports a charge for each of these mesh points. Because the tower region that must be imported for the range-limited force computation always includes the charge-spreading tower region, no additional atom position communication is required.

3.2.2 FFT

In order to evaluate long-range electrostatic forces, Anton must perform two sequentially dependent FFTs: a forward FFT followed by an inverse FFT. With our choice of Ewald parameters, the mesh on which these FFTs is computed is small—just 32×32×32 for a cubical chemical system 40–80 Å on a side, with only 64 mesh points stored on each of Anton's 512 nodes. As such, the actual FFT computation is relatively inexpensive, and most of the FFT time is due to communication. Although Anton's toroidal interconnect is optimized for local communication, the three-dimensional FFT can still be parallelized effectively using a straightforward decomposition into sets of one-dimensional FFTs oriented along each of the three axes. This parallelization strategy involves sending a large number of messages (hundreds per node); alternative strategies that reduce the number of messages but use greater communication volume generally perform better on commodity clusters [2, 14]. Computation of the FFT on Anton is described in more detail in a separate paper [36].

3.2.3 Bond Terms and Correction Forces

In contrast to range-limited forces, which are computed between pairs of atoms dynamically selected according to their current positions, each bonded force term (*bond term*) is specified prior to

the simulation as a small set of atoms along with parameters governing their interaction. At every time step, the data representing each bond term must be brought together with the data representing the positions of its constituent atoms to compute both the bond term and associated correction forces. On a conventional processor, this can be accomplished by ensuring that copies of the required atoms are available in memory on the node containing the bond term; the atoms can then be retrieved to evaluate the bond term. For Anton this technique is impractical: the overhead of dynamically retrieving atoms from memory would significantly increase the overall computation time. Instead, bond terms are statically assigned to GCs, so that each atom has a fixed set of "bond destinations." On every time step an atom's position is sent directly to the flexible subsystems containing its bond destinations, whence it is replicated to the individual GCs as well as the correction pipeline. In addition to eliminating unnecessary communication latency, this approach allows us to perform static load-balancing among the GCs so that the worst-case load is minimized. To ensure that the bond destinations for each atom remain on nodes close to the atom's home node as the chemical system evolves, we recompute the assignment of bond terms to GCs roughly every 100,000 time steps.

3.2.4 Integration and Constraints

Most MD simulations can be accelerated by incorporating *constraints* during integration that fix the lengths of bonds to hydrogen atoms as well as angles between certain bonds. This eliminates the fastest vibrational motions, allowing the simulation to take longer time steps. Applying constraints requires operating on small groups of neighboring atoms (*constraint groups*) that may reside on different home boxes according to the spatial subdivision used for the NT method. This poses a design challenge: on the one hand, we require the atoms to be on the same node during integration; on the other, we require them to be on different nodes in order to perform the NT method's position import.

We address this issue by slightly altering the assignment of atoms to nodes. In particular, we ensure that all atoms in a constraint group reside on the same node; this node takes full responsibility for updating those atoms' positions and velocities. In order to ensure that all required range-limited interactions are computed despite the fact that some atoms may not reside on the "correct" home node according to the spatial subdivision, we slightly expand the NT method's import region, as if the cutoff radius were somewhat larger. The match units and PPIPs still use the original cutoff radius to determine which pairs of atoms should interact, so the set of particle interactions performed remains exactly the same. An alternative approach would involve replicating the computation associated with integration of each constraint group across every node that contains one of that group's atoms, thus ensuring that all updated atom positions are available on their home nodes following integration. We implemented both alternatives for purposes of comparison, and found that the former approach afforded significantly better performance due to both a reduced computational workload and much simpler (and faster) bookkeeping code, which more than offset the communication costs associated with the larger import region.

We use a similar strategy to reduce the cost associated with migrating atoms between nodes as atoms move through space, crossing box boundaries. Atom migration can significantly impact performance, as it requires a large number of sequential bookkeeping operations that are on the critical path of the time step. Anton mitigates this expense by performing migration operations only every N time steps, where N is typically between 4 and 8. Thus, an atom can reside on an "incorrect" node for one or both of two reasons: because it is part of a constraint group that straddles two or more nodes, or because of a delay of several time steps from when it crosses a box boundary to when the next migration operation is performed. Because we can place a conservative upper bound on the distance a particle will move in a time step, a slight expansion of the NT method import region is again sufficient to ensure execution of the correct set of range-limited interactions.

4. NUMERICAL REPRESENTATIONS AND COMPUTATION

Anton achieves both a performance advantage and desirable numerical properties by using customized numerical formats. Throughout the Anton ASIC, we use a fixed-point number system[1], in which a B-bit, signed fixed-point number can represent 2^B evenly spaced distinct real numbers in [–1, 1). This fixed-point representation takes advantage of the fact that all of the arithmetic in an MD simulation involves quantities that are bounded by physical considerations.

The use of fixed-point arithmetic requires increased attention to the units in which different physical quantities are represented on Anton and the scaling factors used to convert between them, but it has two substantial advantages over floating point, which is the arithmetic of choice for modern commodity processors and nearly all MD codes. First, fixed-point hardware adders have significantly lower area and latency than their floating-point equivalents: a 32-bit fixed-point adder is about one-tenth the size and has one-quarter the latency of a 32-bit (single-precision) floating-point adder. Second, unlike floating-point addition, fixed-point addition is associative. In other words, the order of summation of fixed-point numbers will not affect numerical results[2], which simplifies the algorithms necessary to achieve the following desirable numerical properties on Anton:

Determinism. Many popular MD codes are not deterministic; repeated simulations on the same hardware with the same inputs may produce different results, because, for example, the order in which the forces on an atom are summed depends on the order in which messages arrive at the node computing the sum. Simulations on Anton are deterministic, as we have verified by repeating simulations of over four billion time steps and checking that the results are bitwise identical.

Parallel invariance. To our knowledge, no commonly used MD code produces bitwise identical results regardless of the number of processors used in a simulation. In contrast, a given simulation will evolve in exactly the same way on any single- or multi-node Anton configuration. As part of the bringup process, we regularly

[1] Single-precision floating-point arithmetic is available on the Tensilica cores, but is used for very limited purposes.

[2] Furthermore, on Anton, addition and subtraction are allowed to "wrap" in the natural way for twos-complement arithmetic. Thus, a collection of values can be added correctly as long as the final sum is representable, regardless of whether intermediate partial sums wrap. In 4-bit arithmetic, for example, 3/8, 7/8 and −5/8 can be summed and will correctly produce 5/8 regardless of the order of operation, even though 3/8+7/8 wraps to −3/4.

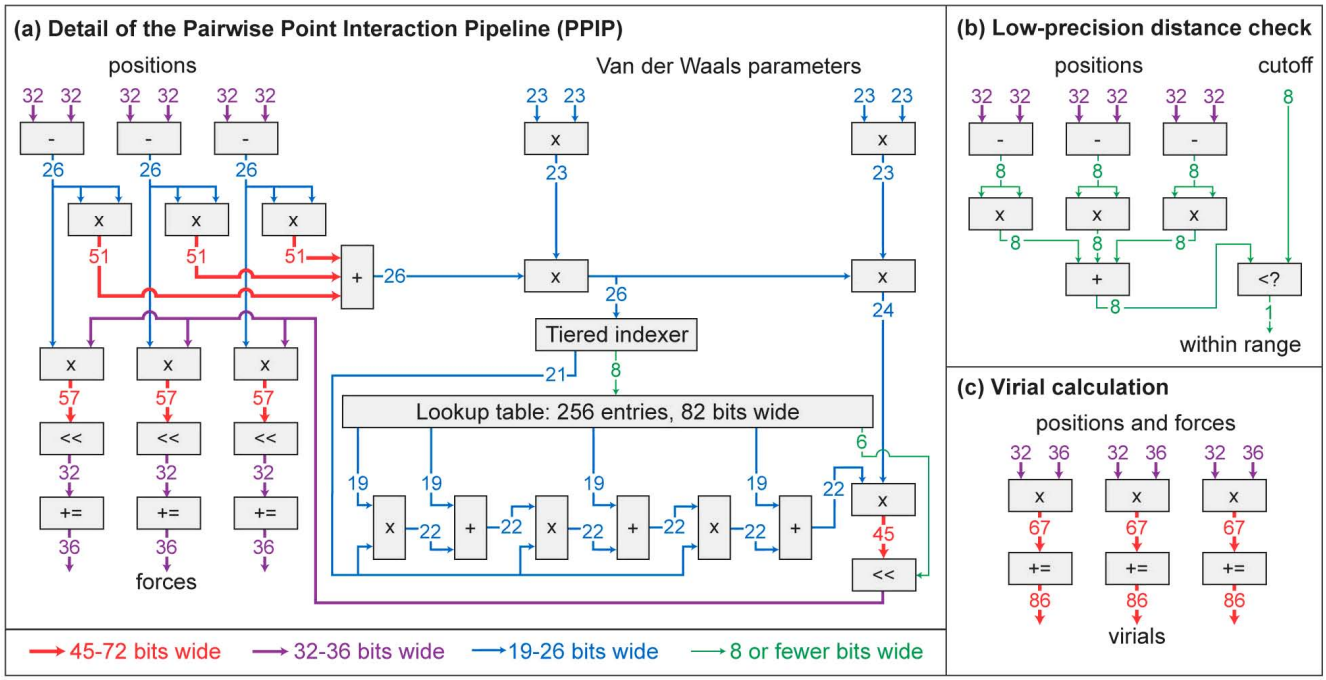

Figure 4: Bit widths of principal data paths and arithmetic operators in the HTIS. (a) One of 32 PPIPs on each ASIC, including an arbitrary function evaluator used to compute van der Waals interactions (a similar function evaluator for electrostatic interactions is not shown). (b) One of 256 low-precision distance check units (components of the match units). (c) Three of 12 multiply/accumulators used in the computation of virials (the large bit widths allow Anton to guarantee determinism and parallel invariance for pressure-controlled simulations). All rounding is performed using a round-to-nearest/even rule.

compared simulation results on one Anton machine to those computed previously on a smaller machine, beginning with simulations performed on a single Anton node. We verified, for example, that 2.7 billion time steps produced identical results on 128-node and 512-node Anton configurations.

Exact reversibility. Anton simulations are exactly reversible when run without constraints, temperature control or pressure control: we have run a simulation for 400 million time steps, negated the instantaneous velocities of all the atoms, and then run another 400 million time steps, recovering the initial conditions bit-for-bit. Anton achieves exact time reversibility by representing the continuum space of atomic positions and momenta with a uniformly dense discrete approximation [2, 34], and by eliminating variability associated with the order of force summation, both of which are natural consequences of fixed-point arithmetic.

The HTIS is the most computationally dense part of the Anton ASIC. Whereas the cores of the flexible subsystem perform most computation in 32-bit arithmetic, the bit width of each arithmetic unit within the HTIS was optimized to minimize die area while maintaining sufficient accuracy. Figure 4 illustrates the variety of datapath widths used in Anton, including 8-bit low-precision distance checks and 86-bit accumulators for the tensor products of force and position (*virials*) used in pressure-controlled simulations.

Each PPIP computes two arbitrary functions of a distance, r, to evaluate the electrostatic and van der Waals forces between two atoms. To provide flexibility while conserving die area, these functions are implemented as tabulated piecewise-cubic polynomials. The tables are indexed by r^2 rather than r, avoiding an unnecessary square root in the distance computation [2]. A tiered indexing scheme divides the domain of r^2 into non-uniform segments, allowing for narrower segments where the function is rapidly varying. In a system with a cutoff of R, for example, the electrostatic table might be configured with 64 entries for $(r/R)^2$ in [0, 1/128), 96 entries for $(r/R)^2$ in [1/128, 1/32), 56 entries for $(r/R)^2$ in [1/32, 1/4) and 24 entries for $(r/R)^2$ in [1/4, 1). In each entry, the table stores the four coefficients of a cubic polynomial and a single exponent common to all four coefficients, as in block-floating-point schemes. Polynomial coefficients, associated exponents, and

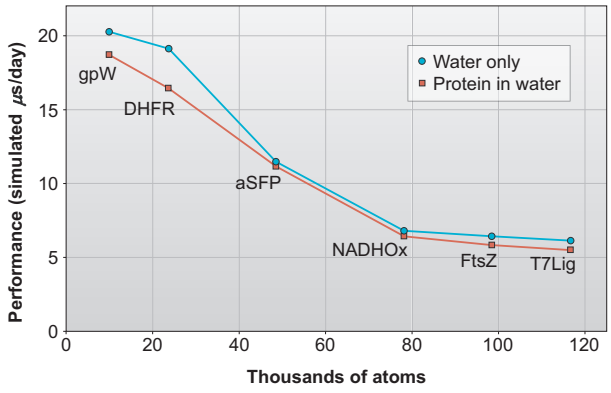

Figure 5: Performance of a 512-node Anton machine for chemical systems of different sizes. Simulation parameters are specified in Table 4 (water systems use the same parameters as similarly sized protein systems). Performance was measured over ten million time steps for each system.

Chemical system (PDB ID)	Number of atoms	Side length (Å)	Cutoff radius (Å)	Mesh size	Performance (μs/day)	Energy drift (kcal/mol/DoF/μs)	Total force error	Numerical force error
gpW (1HYW)	9865	46.8	10.5	32^3	18.7	0.035	80.7×10^{-6}	9.8×10^{-6}
DHFR (5DFR)	23558	62.2	13.0	32^3	16.4	0.053	73.9×10^{-6}	9.0×10^{-6}
aSFP (1SFP)	48423	78.8	15.5	32^3	11.2	0.036	67.3×10^{-6}	11.5×10^{-6}
NADHOx (1NOX)	78017	92.6	10.5	64^3	6.4	0.015	58.4×10^{-6}	8.3×10^{-6}
FtsZ (1FSZ)	98236	99.8	11.0	64^3	5.8	0.015	62.0×10^{-6}	8.9×10^{-6}
T7Lig (1A0I)	116650	105.6	11.0	64^3	5.5	0.021	60.6×10^{-6}	8.9×10^{-6}

Table 4: Accuracy measurements for the protein-in-water systems of Figure 5, and the adjustable parameters used in each simulation. Force errors are expressed as fractions of the rms force. The first column lists the name of each protein and the Protein Data Bank ID of its crystal structure. "DoF" stands for "degree of freedom." All simulations used 2.5-femtosecond time steps with long-range interactions evaluated at every other time step. Bond lengths to hydrogen atoms were constrained. The gpW and DHFR simulations used the AMBER99SB force field [17], while the others used OPLS-AA [18]; water was represented by the rigid TIP3P model.

the parameters of the tiered indexing scheme are computed off-line as part of system preparation; the Remez exchange algorithm is used to compute the minimax polynomial on each segment, after which the coefficients are adjusted to make the function continuous across segment boundaries. Together, these features allow the function evaluators to use relatively narrow (19–22 bit) data paths but still accurately capture functions with large dynamic range.

5. RESULTS

We have thus far constructed four 512-node Anton machines, each of which typically performs an independent simulation. In this section, we present initial measurements of Anton's execution speed, the magnitude of which is illustrated by simulations that have reached a timescale far beyond that of the longest previous MD simulations. We also present the results of various tests performed to assess the accuracy of Anton simulations.

5.1 Performance

Figure 5 shows the performance of a 512-node Anton machine on several different chemical systems, varying in size and composition. On the widely used Joint AMBER-CHARMM benchmark system, which contains 23,558 atoms and represents the protein dihydrofolate reductase (DHFR) surrounded by water, Anton simulates 16.4 μs per day of wall-clock time. The fastest previously reported simulation of this system was obtained using a software package, called Desmond, which was developed within our group for use on commodity clusters [2]. This Desmond simulation executed at a rate of 471 nanoseconds (ns) per day on a 512-node 2.66-GHz Intel Xeon E5430 cluster connected by a DDR InfiniBand network, using only two of the eight cores on each node in order to maximize network bandwidth per core [4]. (Using more nodes, or more cores per node, leads to a decrease in performance as a result of an increase in communication requirements.) Due to considerations related to the efficient utilization of resources, however, neither Desmond nor other high-performance MD codes for commodity clusters are typically run at such a high level of parallelism, or in a configuration with most cores on each node idle. In practice, the performance realized in such cluster-based simulations is generally limited to speeds on the order of 100 ns/day. The previously published simulations listed in Table 1, for example, ran at 100 ns/day or less—over two orders of magnitude short of the performance we have demonstrated on Anton.

For chemical systems with more than 25,000 atoms, the simulation rate of a 512-node Anton scales inversely with the number of atoms simulated (Figure 5 and Table 4). Below 25,000 atoms, the simulation rate gradually plateaus as the ratio of communication to computation increases. Systems containing only water run 3–24% faster than systems of the same size containing a protein solvated in water, because the water molecules do not require bond terms (they are held rigid by constraints); this indicates that bond term computation is sometimes on the critical path and represents an area for performance optimization.

Although the configuration used to generate most of the results presented in this paper contains 512 nodes, Anton machines may incorporate a number of nodes equal to any power of two from 1 to 32,768 nodes.[3] During the bringup process we tested configurations with 1, 2, 4, 8, 64, 128 and 256 nodes. We continue to use configurations with fewer than 512 nodes to simulate certain smaller chemical systems, for which such configurations may prove more cost-effective. A 512-node Anton machine can be partitioned, for example, into four 128-node machines, each of which achieves 7.5 μs/day on the DHFR system—well over 25% of the performance achieved when parallelizing the same simulation across all 512 nodes. Configurations with more than 512 nodes will deliver increased performance for larger chemical systems, but will likely not benefit chemical systems with only a few thousand atoms, for which the increase in communication latency will outweigh the increase in parallelism.

5.2 Accuracy

Anton's determinism, parallel invariance, and reversibility properties (Section 4) provide powerful correctness tests, which we exploited throughout the bringup process. In addition, we performed a variety of tests to ensure that Anton produces results comparable in accuracy to those of widely used MD codes running on commodity hardware.

We examined errors in the per-atom forces computed on Anton by comparing them with forces computed in Desmond using double-precision floating-point arithmetic and extremely conservative values for adjustable parameters (cutoffs, grid size, etc.). Table 4 lists the root-mean-squared (rms) total force errors, expressed as a

[3] The number of nodes in an Anton machine need not actually be a power of two, but the current software only supports power-of-two configurations.

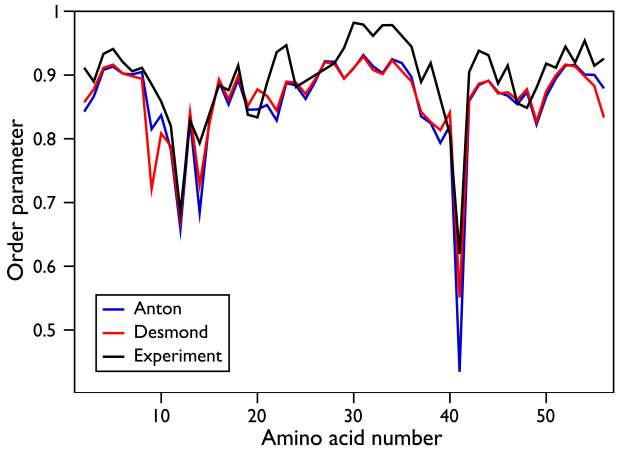

Figure 6: Amount of motion of each amino acid in the protein GB3, as estimated from 1-μs simulations on Anton (blue) and Desmond (red), and as measured experimentally by NMR (black). Smaller values on the vertical axis indicate more motion. Simulations used the AMBER99SB force field [17].

fraction of the rms force, for the protein systems of Figure 5. In each case, the ratio is less than 10^{-4}; ratios of 10^{-3} are generally considered acceptable for MD simulation [37]. The table also lists "numerical force error," defined as the difference between Anton forces and forces computed in double-precision floating-point arithmetic using the same adjustable parameter values as Anton. This error is nearly an order of magnitude smaller than the total force error, indicating that the majority of the total force error is not inherent to Anton's numerics, and could be reduced by adjusting parameters at the cost of performance.

Calculating forces with negligible error is necessary but not sufficient for the execution of accurate MD simulations. Even small errors can accumulate over time if they are correlated or biased. Energy drift, the rate of change of total system energy (which is exactly conserved by the underlying equations of motion), is more sensitive to certain errors that could adversely affect the physical predictions of a simulation.[4] Table 4 lists energy drift figures measured on Anton, which are lower than those typically obtained with widely used single-precision and double-precision codes on commodity hardware [2, 14], although certain double-precision codes obtain significantly lower energy drift [9].

Higher-level tests of Anton have included the estimation of experimentally observable quantities through protein simulations. One example, shown in Figure 6, involves quantities called *backbone amide order parameters*, which are measured by nuclear magnetic resonance (NMR) experiments and which characterize the amount of movement of each amino acid in a protein (an order parameter near 1 indicates that the amino acid has little mobility, while a lower order parameter indicates that it has more). Using a previously described method [24], we estimated these order parameters for 55 amino acids of the protein GB3 from a one-microsecond Anton simulation and from a one-microsecond simulation on a cluster running Desmond, with the same force field. The two estimates are highly similar; the differences reflect the fact that one finite-length trajectory will encounter certain rare events that the other will not, because the systems we simulate are chaotic and any two MD simulations that are not bitwise identical will follow divergent trajectories. Figure 6 also shows the corresponding order parameters measured by recent NMR experiments [13]. Again, the results are similar; differences reflect a combination of experimental measurement errors and errors in the estimates from simulation (resulting from finite simulation length and from limitations in the force field employed).

[4] Most simulations are run with a thermostat that removes the most immediate effects of energy drift, but energy drift in unthermostatted simulations (such as those reported in Table 4) provides a useful diagnostic for implementation accuracy.

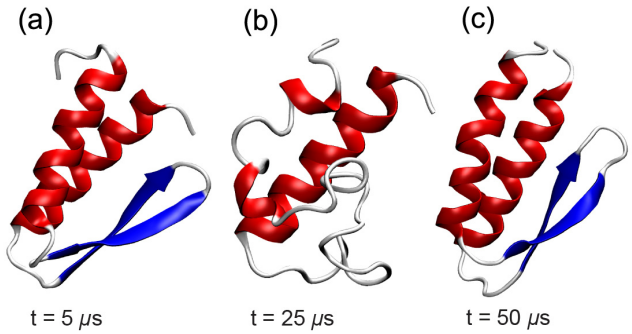

Figure 7: Unfolding and folding events in a 236-μs simulation of the protein gpW. (a) is a folded structure early in the simulation, (b) is a snapshot after the protein has partially unfolded, and (c) is a snapshot after it has folded again.

5.3 Long-Timescale Simulations

Although Anton is still the focus of a substantial software development effort, it has already enabled the longest MD simulations to date by a very large factor. Few previously reported simulations have reached 2 μs, the longest being a 10 μs simulation reported in 2008 that took over three months on the NCSA Abe machine [10] (Table 1). By contrast, we have already performed a 1031-μs simulation on Anton.

This millisecond-scale simulation models a protein called bovine pancreatic trypsin inhibitor (BPTI) (Figure 1). This protein has been the subject of many previous MD simulations; in fact, the first MD study of a protein, published in 1977 [26], simulated BPTI for 9.2 picoseconds. Our simulation, which is over 100 million times longer, reveals behavior that is not evident at timescales of picoseconds, nanoseconds, or even a few microseconds. Certain aspects of this behavior were unanticipated, and have provided an explanation for experimental data on the dynamical properties of BPTI. The biophysical observations and results from this simulation will be detailed elsewhere.

Anton's performance is dependent on the size of the system being simulated, and on a number of parameters associated with the execution of molecular dynamics simulations. Our simulation of BPTI involves a cubic system measuring 51.3 Å on a side and containing 17,758 particles: 892 protein atoms and 6 chloride ions, each represented by a single particle using the AMBER99SB force field [17] with a correction to the parameters describing isoleucine [28]; and 4215 water molecules, each represented by 4 particles using the TIP4P-Ew model [16] (each of the four particles in this water model is treated computationally as an atom, even though a water

molecule contains only three physical atoms). We used a 10.4-Å cutoff radius for range-limited interactions, a 7.1-Å cutoff for charge spreading and force interpolation, a 32×32×32 FFT mesh, Berendsen temperature control, and 2.5-femtosecond time steps, with long-range electrostatics evaluated every other time step. It proceeded at 9.8 μs/day, but more recent software and clock speed improvements have allowed a comparable simulation of this system to proceed at 18.2 μs/day.

We have also performed a variety of other simulations on Anton with durations of 10–236 μs, including several of protein folding. In the hope of observing both protein folding and protein unfolding events, for example, we simulated a viral protein called gpW for 236 μs at a temperature that, experimentally, equally favors the folded and unfolded states. We observed a sequence of folding and unfolding events, such as those illustrated by the snapshots in Figure 7.

6. CONCLUSION

Commodity computing benefits from economies of scale but imposes limitations on the extent to which an MD simulation can be accelerated through parallelization. Due to a combination of a unique hardware architecture and carefully co-designed algorithms, Anton gives scientists, for the first time, the ability to perform MD simulations on the order of a millisecond—two orders of magnitude longer than any atomically detailed simulation reported on general-purpose hardware, and three orders of magnitude longer than any simulation reported previously on special-purpose hardware (a one-microsecond simulation using MDGRAPE-3 was reported in 2007 [35]). Anton's actual measured performance exceeds our previously published, simulator-based predictions [33], both because we were able to operate the system at a higher clock rate than anticipated and because we discovered further opportunities for improvement in algorithms and software. (We predicted a simulation rate of 14.5 μs/day for the DHFR benchmark system, for example, and achieved a simulation rate of 16.4 μs/day.)

Although it has only been operational for a few months, Anton has already begun to serve as a "computational microscope," allowing the observation of biomolecular processes that are inaccessible to laboratory experiments and that, until now, have been well beyond the reach of computer simulation. Our hope is that, over the coming years, Anton will contribute not only to the advancement of basic scientific knowledge but to the development of safe, effective, precisely targeted medicines capable of relieving suffering and saving human lives.

7. ACKNOWLEDGMENTS

We thank Joseph Gagliardo, Jon Peticolas, Daniel Ramot, and Jochen Spengler for major contributions to the bringup process; Amos Even, Lev Iserovich, and Ben Vitale for development of systems software and test infrastructure; Tiankai Tu for helpful comments on the paper; and Rebecca Kastleman and Jennifer McGrady for editorial assistance.

8. REFERENCES

[1] A. Bhatele, S. Kumar, C. Mei, J. C. Phillips, G. Zheng, and L. V. Kalé. 2008. Overcoming scaling challenges in biomolecular simulations across multiple platforms. In Proc. IEEE Int. Parallel and Distributed Processing Symp. (Miami, FL, 2008).

[2] K. J. Bowers, E. Chow, H. Xu, R. O. Dror, M. P. Eastwood, B. A. Gregersen, J. L. Klepeis, I. Kolossváry, M. A. Moraes, F. D. Sacerdoti, J. K. Salmon, Y. Shan, and D. E. Shaw. 2006. Scalable algorithms for molecular dynamics simulations on commodity clusters. In Proc. ACM/IEEE Conf. on Supercomputing (Tampa, FL, 2006).

[3] K. J. Bowers, R. O. Dror, and D. E. Shaw. 2007. Zonal methods for the parallel execution of range-limited N-body problems. J. Comput. Phys. 221 (1), 303–329.

[4] E. Chow, C. A. Rendleman, K. J. Bowers, R. O. Dror, D. H. Hughes, J. Gullingsrud, F. D. Sacerdoti, and D. E. Shaw. 2008. Desmond performance on a cluster of multicore processors. D. E. Shaw Research Technical Report DESRES/TR—2008-01. http://deshawresearch.com.

[5] R. O. Dror, D. H. Arlow, D. W. Borhani, M. Ø. Jensen, S. Piana, and D. E. Shaw. 2009. Identification of two distinct inactive conformations of the β_2-adrenergic receptor reconciles structural and biochemical observations. Proc. Natl. Acad. Sci. USA 106, 4689–4694.

[6] D. L. Ensign, P. M. Kasson, and V. S. Pande. 2007. Heterogeneity even at the speed limit of folding: large-scale molecular dynamics study of a fast-folding variant of the villin headpiece. J. Mol. Biol. 374, 806–16.

[7] U. Essmann, L. Perera, M. L. Berkowitz, T. Darden, H. Lee, and L. G. Pedersen. 1995. A smooth particle mesh Ewald method. J. Chem. Phys. 103 (19), 8577–8593.

[8] R. D. Fine, G. Dimmler, and C. Levinthal. 1991. A special purpose, hardwired computer for molecular simulation. Proteins: Struct., Funct., Genet. 11 (4), 242–253.

[9] B. G. Fitch, A. Rayshubskiy, M. Eleftheriou, T. J. C. Ward, M. E. Giampapa, M. C. Pitman, J. W. Pitera, W. C. Swope, and R. S. Germain. 2006. Blue Matter: approaching the limits of concurrency for classical molecular dynamics. In Proc. ACM/IEEE Conf. on Supercomputing (Tampa, FL, 2006).

[10] P. L. Freddolino, F. Liu, M. H. Gruebele, and K. Schulten. 2008. Ten-microsecond MD simulation of a fast-folding WW domain. Biophys. J. 94 (10), L75–L77.

[11] R. S. Germain, B. Fitch, A. Rayshubskiy, M. Eleftheriou, M. C. Pitman, F. Suits, M. Giampapa, and T. J. C. Ward. 2005. Blue Matter on Blue Gene/L: massively parallel computation for biomolecular simulation. In Proc. 3rd IEEE/ACM/IFIP Int. Conf. on Hardware/Software Codesign and System Synthesis (New York, NY, 2005).

[12] A. Grossfield, M. C. Pitman, S. E. Feller, O. Soubias, and K. Gawrisch. 2008. Internal hydration increases during activation of the G-protein-coupled receptor rhodopsin. J. Mol. Biol. 381, 478–486.

[13] J. B. Hall and D. Fushman. 2006. Variability of the 15N chemical shielding tensors in the B3 domain of protein G from 15N relaxation measurements at several fields. Implications for backbone order parameters. J. Am. Chem. Soc. 128 (24), 7855–7870.

[14] B. Hess, C. Kutzner, D. van der Spoel, and E. Lindahl. 2008. GROMACS 4: algorithms for highly efficient, load-balanced, and scalable molecular simulation. J. Chem. Theory Comput. 4 (2), 435–447.

[15] R. W. Hockney and J. W. Eastwood. 1988. Computer Simulation Using Particles. Adam Hilger, Bristol, England, 1988.

[16] H. W. Horn, W. C. Swope, J. W. Pitera, J. D. Madura, T. J. Dick, G. L. Hura, and T. Head-Gordon. 2004. Development of an improved four-site water model for biomolecular simulations: TIP4P-Ew. J. Chem. Phys. 120 (20), 9665–9678.

[17] V. Hornak, R. Abel, A. Okur, B. Strockbine, A. Roitberg, and C. Simmerling. 2006. Comparison of multiple Amber force fields and development of improved protein backbone parameters. Proteins: Struct., Funct., Bioinf. 65, 712–725.

[18] W. L. Jorgensen, D. S. Maxwell, and J. Tirado-Rives. 1996. Development and testing of the OPLS all-atom force field on conformational energetics and properties of organic liquids. J. Am. Chem. Soc. 118 (45), 11225–11236.

[19] L. V. Kalé, R. Skeel, M. Bhandarkar, R. Brunner, A. Gursoy, N. Krawetz, J. Phillips, A. Shinozaki, K. Varadarajan, and K. Schulten. 1999. NAMD2: greater scalability for parallel molecular dynamics. J. Comput. Phys. 151 (1), 283–312.

[20] F. Khalili-Araghi, J. Gumbart, P.-C. Wen, M. Sotomayor, E. Tajkhorshid, and K. Shulten. 2009. Molecular dynamics simulations of membrane channels and transporters. Curr. Opin. Struct. Biol. 19 (2), 128–137.

[21] J. L. Klepeis, K. Lindorff-Larsen, R. O. Dror, and D. E. Shaw. 2009. Long-timescale molecular dynamics simulations of protein structure and function. Curr. Opin. Struct. Biol. 19 (2), 120–127.

[22] J. S. Kuskin, C. Young, J.P. Grossman, B. Batson, M. Deneroff, R. O. Dror, and D. E. Shaw. 2008. Incorporating flexibility in Anton, a specialized machine for molecular dynamics simulation. In Proc. 14th Int. Symp. on High-Performance Computer Architecture (Salt Lake City, UT, 2008).

[23] R. H. Larson, J. K. Salmon, R. O. Dror, M. M. Deneroff, C. Young, J. P. Grossman, Y. Shan, J. L. Klepeis, and D. E. Shaw. 2008. High-throughput pairwise point interactions in Anton, a specialized machine for molecular dynamics simulation. In Proc. 14th Int. Symp. on High-Performance Computer Architecture (Salt Lake City, UT, 2008).

[24] P. Maragakis, K. Lindorff-Larsen, M. Eastwood, R. O. Dror, J. L. Klepeis, I. T. Arkin, M. Ø. Jensen, H. Xu, N. Trbovic, R. A. Friesner, A. G. Palmer III, and D. E. Shaw. 2008. Microsecond molecular dynamics simulation shows effect of slow loop dynamics on backbone amide order parameters of proteins. J. Phys. Chem. B 112, 6155–6158.

[25] K. Martinez-Mayorga, M. C. Pitman, A. Grossfield, S. E. Feller, and M. F. Brown. 2006. Retinal counterion switch mechanism in vision evaluated by molecular simulations. J. Am. Chem. Soc. 128, 16502–16503.

[26] J. A. McCammon, B. R. Gelin, and M. Karplus. 1977. Dynamics of folded proteins. Nature 267, 585–590.

[27] T. Narumi, Y. Ohno, N. Okimoto, T. Koishi, A. Suenaga, N. Futasugi, R. Yanai, R. Himeno, S. Fujikawa, M. Taiji, and M. Ikei. 2006. A 55 TFLOPS simulation of amyloid-forming peptides from yeast prion Sup35 with the special-purpose computer system MDGRAPE-3. In Proc. ACM/IEEE Conf. on Supercomputing (Tampa, FL, 2006).

[28] S. Piana-Agostinetti, K. Lindorff-Larsen, P. Maragakis, M. P. Eastwood, R. O. Dror, and D. E. Shaw. 2009. Improving molecular mechanics force fields by comparison of microsecond simulations with NMR experiments. Biophys. J. 96 (3), 406a.

[29] S. J. Plimpton, S. Attaway, B. Hendrickson, J. Swegle, C. Vaughan, and D. Gardner. 1996. Transient dynamics simulations: parallel algorithms for contact detection and smoothed particle hydrodynamics. In Proc. ACM/IEEE Conf. on Supercomputing (Pittsburgh, PA, 1996).

[30] B. Roux. 2007. A proton-controlled check valve for sodium ion transport. Nature Chem. Bio. 3, 609–610.

[31] Y. Shan, J. L. Klepeis, M. P. Eastwood, R. O. Dror, and D. E. Shaw. 2005. Gaussian split Ewald: a fast Ewald mesh method for molecular simulation. J. Chem. Phys. 122, 054101.

[32] D. E. Shaw. A fast, scalable method for the parallel evaluation of distance-limited pairwise particle interactions. 2005. J. Comput. Chem. 26 (13), 1318–1328.

[33] D. E. Shaw, M. M. Deneroff, R. O. Dror, J. S. Kuskin, R. H. Larson, J. K. Salmon, C. Young, B. Batson, K. J. Bowers, J. C. Chao, M. P. Eastwood, J. Gagliardo, J. P. Grossman, C. R. Ho, D. J. Ierardi, I. Kolossváry, J. L. Klepeis, T. Layman, C. McLeavey, M. A. Moraes, R. Mueller, E. C. Priest, Y. Shan, J. Spengler, M. Theobald, B. Towles, and S. C. Wang. 2007. Anton: a special-purpose machine for molecular dynamics simulation. In Proc. 34th Int. Symp. on Computer Architecture (San Diego, CA, 2007). ACM Press, New York, NY, 1–12.

[34] R. D. Skeel. 1999. Symplectic integration with floating-point arithmetic and other approximations. In Proc. NSF/CBMS Regional Conf. on Numerical Analysis of Hamiltonian Differential Equations (Golden, CO, 1999) 29, 3–18.

[35] A. Suenaga, T. Narumi, N. Futatsugi, R. Yanai, Y. Ohno, N. Okimoto, and M. Taiji. 2007. Folding dynamics of 10-residue b-hairpin peptide chignolin. Chem. Asian J. 2, 591–598.

[36] C. Young, J. A. Bank, R. O. Dror, J. P. Grossman, J. K. Salmon, and D. E. Shaw. 2009. A 32×32×32, spatially distributed 3D FFT in four microseconds on Anton. In Proc. ACM/IEEE Conf. on Supercomputing (Portland, OR, 2009).

[37] R. Zhou, E. Harder, H. Xu, and B. J. Berne. 2001. Efficient multiple time step method for use with Ewald and particle mesh Ewald for large biomolecular systems. J. Chem. Phys. 115 (5), 2348–2358.