LA-UR-05-2925

25 Tflop/s Multibillion-Atom Molecular Dynamics Simulations and Visualization/Analysis on BlueGene/L

Submitted for the 2005 Gordon Bell Performance Prize

Timothy C. Germann * Kai Kadau † Peter S. Lomdahl ‡

Abstract

We demonstrate the excellent performance and scalability of a classical molecular dynamics code, SPaSM, on the IBM BlueGene/L supercomputer at LLNL. Simulations involving up to 160 billion atoms (micron-size cubic samples) on 65,536 processors are reported, consistently achieving 24.4–25.5 Tflop/s for the commonly used Lennard-Jones 6-12 pairwise interaction potential. Two extended production simulations (one lasting 8 hours and the other 13 hours wall-clock time) of the shock compression and release of porous copper using a more realistic many-body potential are also reported, demonstrating the capability for sustained runs including on-the-fly parallel analysis and visualization of such massive data sets. This opens up the exciting new possibility of using atomistic simulations at micron length scales to directly bridge to mesoscale and continuum-level models.

1 Introduction

The IBM BlueGene/L supercomputer at Lawrence Livermore National Laboratory, with 65,536 CPUs connected by multiple high-performance networks, enables a completely new class of physical problems to be investigated. Using either pairwise interactions such as the Lennard-Jones potential, or the many-body embedded atom method (EAM) potential for simple metals, classical molecular dynamics simulations of systems containing as many as 160 billion atoms (or a cube of copper a micron on each side) can be modeled. In order to obtain any new physical insights, however, it is equally important (and typically a greater challenge) that the analysis of such systems be tractable. We demonstrate that this is in fact possible, in large part due to our portable and highly efficient parallel visualization code, which enables the rendering of atomic spheres, Eulerian cells, and other geometric objects at arbitrary resolution in a matter of seconds to minutes, even for tens of thousands of processors and billions of atoms. After briefly reviewing the basic properties of BlueGene/L and our parallel (domain decomposition) molecular dynamics code (SPaSM), we will describe the performance scaling and initial results obtained for shock compression and release of a defective EAM Cu sample containing up to 2.13 billion atoms, illustrating the plastic deformation accompanying void collapse as well as the subsequent void nucleation, growth and linkup upon release.

2 The BlueGene/L architecture

As currently configured (April 2005), BlueGene/L consists of 32,768 nodes, each with two IBM PowerPC 440 processors (at 700 MHz clock speeds) and 512 MB of memory [1, 2, 3]. Each processor contains a superscalar double floating point unit (FPU), allowing up to 4 floating-point operations (a fused multiply-add instruction on each FPU) to be issued by each processor each clock cycle [4]. The theoretical peak performance is thus 5.6 Gflop/s (8 flops $\times$ 700 MHz) per node, or 180 Tflop/s for the current configuration. The final system, scheduled for delivery later this summer, will double the current size, to a peak 360 Tflop/s. Although the default operation mode assigns one processor on each node to computation and the other to communication (in so-called "coprocessor" mode), it is also possible to

*Applied Physics Division (X-7), Los Alamos National Laboratory, Los Alamos, NM 87545, Email: *tcg@lanl.gov*
†Theoretical Division (T-14), Los Alamos National Laboratory, Los Alamos, NM 87545, Email: *kkadau@lanl.gov*
‡Theoretical Division (T-11), Los Alamos National Laboratory, Los Alamos, NM 87545, Email: *pxl@lanl.gov*

run independent MPI tasks on each processor ("virtual node" mode). This is of course most effective for CPU-bound applications which do not demand an excessive amount of memory (under 256 MB per CPU) or communication; fortunately, this class includes the SPaSM code, as we show below.

There are 3 independent internal networks on BlueGene/L (plus two other system control-related networks not of interest to the general user). The principal interconnect is a 3D torus connecting each node to its 6 nearest neighbors, which is particularly amenable to grid-based applications such as the SPaSM cell-based spatial decomposition, described below. A binary-tree network enables efficient global reduction and broadcast operations (only 15 hops are required for the present $2^{15} = 32768$ node system), and a special single-bit binary tree barrier network enables global synchronization of all nodes in less than 10 μs.

3 Classical molecular dynamics simulations

For several decades, molecular dynamics (MD) simulations have been an important tool in statistical mechanics, with applications ranging from equilibrium equation of state calculations, to nonequilibrium behavior such as transport coefficients. The idea behind an MD simulation is very simple: Newton's classical equations of motion, $F = ma$, are solved directly to evolve the positions and velocities of a large collection of atoms comprising the fluid or crystal of interest. While conceptually simple, this task can present a formidable computing problem. If all atoms interact via pairwise, or two-body, forces (such as is the case for gravity, or for Coulombic interactions between charged particles), the direct solution of this general N-body problem will require the calculation of $N(N-1)/2$ forces, i.e., the computational cost will grow as the square of the number of particles, $O(N^2)$. To complicate matters even further, many of the empirical force fields which have been (and are being) developed for materials ranging from elemental silicon, to transition metals and hydrocarbons, involve more complicated many-body potentials with explicit or effective bond-bending and torsional terms, whose cost in theory grows as $O(N^3)$ or $O(N^4)$, respectively. Such direct calculations quickly overwhelm the computing capabilities of even the fastest supercomputers, so clever algorithms have been developed to bring the computational cost under control. For charged systems where Coulombic interactions are unavoidable, or astrophysical simulations requiring gravity, the fast multipole method developed in the late 1980s has become the tool of choice, reducing the cost to $O(N)$ by approximating the long-range interactions in a controllable way. Fortunately, the situation is even better for most non-ionic materials, since the absence of Coulombic interactions makes the neglect of any long-range forces a good approximation. The resulting short-range potentials can be naturally computed with $O(N)$ cost (even for many-body potentials) if the range of the potential is much shorter than the size of the simulation box, since each atom only interacts with a limited number of neighbors. Despite these efforts, until recently (the past decade) most simulations have been limited to a few hundred thousand atoms and a relatively small number of timesteps. To carry out realistic simulations of material properties in three dimensions (3D), however, the inclusion of tens of millions to billions of atoms may be required, depending on the length scale of any initially imposed or naturally emerging microstructure or defect distribution.

3.1 Empirical interatomic potentials

The most commonly used short-range interaction is the Lennard-Jones 6-12 (LJ) potential, typically truncated at some finite distance r_{cut} so that no particles directly interact beyond this range:

$$V(r) = \begin{cases} 4\epsilon \left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6} \right], & 0 < r \le r_{cut}, \\ 0, & r_{cut} < r. \end{cases} \quad (1)$$

Here σ and ϵ are the usual LJ parameters representing the bond length and well depth. For a more realistic description of metallic bonding, a semiempirical embedded atom method (EAM) [5, 6] potential is often used for simple metals. The atomic potential energy E_i consists of two terms: a pair interaction $\phi(r)$ describing the short-range nuclear repulsion, and an attractive embedding energy $F(\overline{\rho}_i)$ from placing a positively charged nucleus i into a (negatively charged) background electron density $\overline{\rho}_i = \sum_j \rho_j(r_{ij})$ due to the atomic electron densities $\rho_j(r)$ of neighboring atoms j:

$$E_i = \frac{1}{2}\sum_j \phi(r_{ij}) + F\left[\sum_j \rho_j(r_{ij})\right]. \quad (2)$$

Again, both $\phi(r)$ and $\rho(r)$ are short-ranged functions with a cutoff distance r_{cut} typically only including the first few neighbor shells surrounding each atom (a few dozen atoms). Although early EAM potentials used analytic forms for the $\phi(r)$ pair potential, $\rho(r)$ electron density, and $F(\overline{\rho})$ embedding energy, most recent potentials have been developed in tabular form, for both computational efficiency and physical reasons. For instance, a common choice is to require that the lattice energy $E(V)$ under uniform compression and expansion (volume V) exactly follow a "universal equation of state" suggested by Rose *et al* [7]. One may then specify analytic forms for $\phi(r)$ and $\rho(r)$ (no longer restricted to the original physical meanings of these terms), and tabulate the embedding function $F(\overline{\rho})$, since it will not in general be expressible in a simple analytic closed-form expression.

3.2 The SPaSM molecular dynamics code

The SPaSM (Scalable Parallel Short-range Molecular dynamics) code, origianlly developed at Los Alamos National Laboratory for the Thinking Machines CM-5 in the early 1990s, has been described in detail elsewhere [8, 9]. The basic concept, using spatial decomposition to partition space into cells with edge lengths no smaller than r_{cut}, is illustrated in Fig. 1. Rather than the standard neighbor lists used to keep track of which particles interact, the cell construct allows an on-the-fly scan of all particles in the same or immediately adjacent cells, using an "interaction path" to organize such scans and the synchronous message-passing required when adjacent cells reside on different processing nodes. The SPaSM code was ported to several multi-processor platforms supporting both message-passing (on distributed memory computers) and multi-threading (on shared-memory systems), as well as clusters of inexpensive personal computers (Beowulf systems). It shows excellent scaling properties, both in terms of problem size and parallel efficiency [10]. SPaSM has been used extensively for large-scale materials science and physics simulations (involving 10^6 to 10^8 atoms) over the past decade, resulting in over 50 publications (plus another 10 primarily or exclusively devoted to computational aspects), including several in *Science* [11, 12, 13] and *Proceedings of the National Academy of Science* [14]. These studies have provided new insights into such diverse problems as fracture [15], dislocation interactions [11], shock-induced plasticity [12] and phase transformations [13], and the Rayleigh-Taylor fluid instability [14].

The SPaSM code won part of the 1993 IEEE Gordon Bell performance prize for a molecular dynamics simulation with 131 million atoms at 50 Gflop/s on the 1024-node Thinking Machines CM-5 [9]. Subsequently, the portability of the code was increased by replacing CMMD and CM I/O calls with a set of wrapper functions (e.g., to MPI library routines), removing all assembler code, developing a visualization library, and combining these elements into an interactive package using the Python scripting language, all without any significant performance degradation [16, 17]. In 1998, this enhanced version of SPaSM won part of the Gordon Bell price/performance prize for a shockwave simulation on the 70-node Los Alamos Avalon Beowulf cluster, achieving a ratio of $15/Mflop [18].

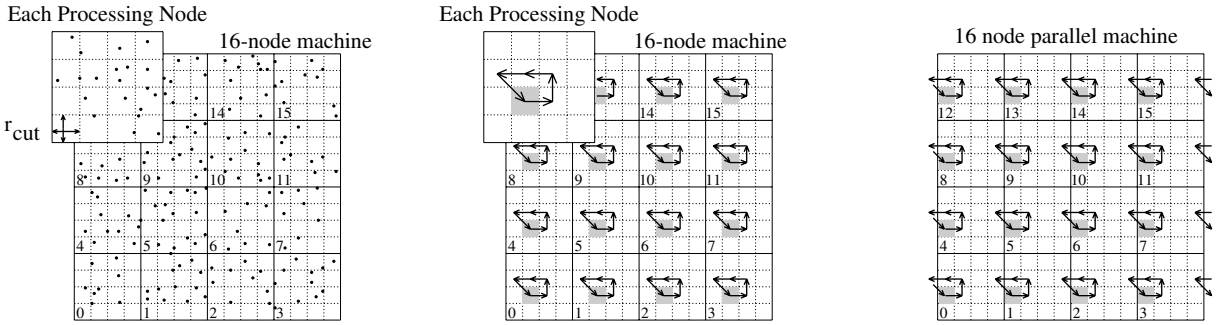

Figure 1: Spatial decomposition of atoms into cells with edge length r_{cut} (left), and a resulting 2D interaction path to scan neighboring cells, illustrating both the cases where the interactions are entirely local (center) or require synchronous send-and-receive message-passing (right). Newton's Third Law is used to restrict the number of neighboring cells visited to 4 (or 13, in 3D).

3.3 Porting to BG/L

Since the stripped-down compute node kernel on BG/L does not support dynamic library loading, porting the current modular (Python-based) SPaSM implementation was deemed impractical, although in principle it would certainly be possible to build a static Python executable with pre-loaded SPaSM simulation, analysis, and visualization libraries. But to expedite porting (into a matter of hours rather than days), we decided to revert to the pre-Python code which had been successfully ported from the original CM-5 version to a variety of platforms, with either MPI, PVM, or shared memory parallelism. The critical code in the LJ force calculation loop is written entirely in C, and hand-optimized for modern superscalar architectures by loop unrolling, macro expansion, and careful use of `register` variables. After testing various compiler options and loop unrolling depths, we used the IBM xlc compiler options `-O2 -qarch=440d -qtune=440` on a 4x unrolled code. The `-qarch=440d` option only gives a 3% speedup over `-qarch=440`, indicating that the double FPU unit is not being efficiently used without explicit hand-tuning [4].

3.4 Timings and performance

We carried out an extensive series of short (10 timestep) benchmark runs for timing purposes, with different numbers of particles and processors. The atoms are set up in a 3D fcc lattice at a temperature near the melting point, although melting of course cannot occur within such a short simulation time. Nevertheless, the timings are representative of an actual simulation at these particle densities (or, as is typically the case in many realistic simulations, the maximum local particle density at any given instant). The average wall clock times per timestep (including force calculation, timestep integration, and particle redistribution) are shown for the LJ potential with two different choices of r_{cut} in Tables 1 and 2, along with the corresponding Tflop rates obtained by counting the actual number of floating point operations in both the force and integration routines. For the LJ potential and velocity Verlet integrator as implemented, we count $9N_{all} + 20N_{interact} + 65N$ flops, where each divide operation is counted as 5 flops. Here N_{all} is the total number of atom pairs which are considered (all of those in the same or adjoining cells), $N_{interact}$ is the number of pairs which are closer than r_{cut}, and N is the total number of particles.

Except for relatively small problem sizes (less than 100,000 atoms per processor), we consistently measure 17.5–18.5 Tflop/s for the shorter-range LJ potential in Table 1, and 24.5–25.5 Tflop/s for the longer-range LJ potential in Table 2. To give an idea of how large the fraction of time spent in the force calculation is, the $r_{cut} = 5.0\sigma$ benchmark with 128 billion atoms on 32,768 nodes (65,536 processors) required 93.8 s to compute forces (98.5% of the total time), 0.3 s for the (velocity Verlet) integration (0.3%), and 1.1 s to redistribute particles after the integration step (1.1%). The message-passing time in the force calculation and particle redistribution is only 0.7 s, or 0.8% of the total time. As the number of particles is reduced this fraction of course increases, to 10% for the 1 billion particle run. Despite this, the overall performance is still over 22 Tflop/s, and the iteration time of 0.85 s represents a speedup of 14.7 over the 2048-node run, or 92% parallel efficiency. This is also illustrated in Fig. 2, showing both the raw data from Tables 1 and 2 as well as scaled iteration times for $r_{cut} = 5.0\sigma$, which converge to less than 50 μs/particle/processor for all partitions, becoming only gradually slower for fewer numbers of particles per processor. (With 256 MB per processor in virtual node mode, 2.5 million atoms per processor is the maximum that can be stored, since each atom typically requires 80–100 bytes, including overhead for the cell structures.)

We also include a set of timings in coprocessor mode in Table 1, demonstrating the nearly 100% speedup obtained by using both processors on each node for MPI processes (virtual node mode). Timings for the EAM copper potential [19] used in the production runs described below are listed in Table 2. Although the EAM potential has a very short cutoff distance (translating to $r_{cut} \simeq 2.0\sigma$ in LJ units), it involves twice as many interaction path steps (one to compute the electronic density at each atom site, and the second to compute the forces on neighboring atoms due to this density). In lieu of complex (or nonexistant) analytic expressions, it also utilizes fast table lookups for each of the functions $\phi(r)$, $\rho(r)$, and $F(\overline{\rho})$.

4 Parallel visualization code

Typically, checkpoint files (containing positions and velocities of all atoms) are periodically written out during a MD simulation, either for restarting runs or for postprocessing visualization and data analysis. However, such I/O becomes

Billions of Particles	Number of Nodes 2048	4096	8192	16384	32768	coprocessor 32768
1	2.22 (1.09)	1.13 (2.19)	0.60 (4.08)	0.31 (7.94)	0.15 (16.14)	0.30 (8.33)
2	4.30 (1.10)	2.19 (2.21)	1.13 (4.38)	0.59 (8.32)	0.30 (16.57)	0.56 (8.79)
4	8.15 (1.14)	4.22 (2.25)	2.19 (4.41)	1.13 (8.76)	0.58 (16.96)	1.06 (9.27)
8	16.26 (1.13)	8.20 (2.27)	4.27 (4.44)	2.21 (8.75)	1.10 (17.86)	2.09 (9.24)
16		16.12 (2.23)	8.14 (4.57)	4.28 (8.86)	2.16 (17.91)	4.06 (9.34)
32			16.16 (4.56)	8.12 (9.17)	4.18 (18.14)	7.79 (9.55)
64				16.22 (9.09)	8.07 (18.43)	15.53 (9.49)
80				20.52 (8.96)	10.90 (16.87)	19.56 (9.40)
128					16.13 (18.28)	30.88 (9.46)
160					21.83 (16.69)	39.01 (9.34)

Table 1: Average time per timestep in seconds, with Tflop/s in parentheses, for an analytic Lennard-Jones potential with $r_{cut} = 2.5\sigma$, using virtual node mode (2 processors per node) except for the last column, where the default coprocessor mode was used (the second processor on each node is dedicated to communication).

Billions of Particles	Number of Nodes 2048	4096	8192	16384	32768	EAM Cu 32768
1	12.59 (1.53)	6.37 (3.02)	3.39 (5.68)	1.72 (11.19)	0.85 (22.55)	0.71
2	24.73 (1.56)	12.52 (3.08)	6.38 (6.03)	3.29 (11.70)	1.64 (23.45)	1.49
4	48.66 (**1.58**)	24.65 (3.12)	12.66 (6.08)	6.38 (12.08)	3.20 (24.04)	2.67
8	95.81 (1.58)	48.53 (**3.17**)	24.90 (6.19)	12.56 (12.26)	6.30 (24.44)	5.19
16		95.47 (3.17)	48.85 (**6.30**)	24.67 (12.48)	12.43 (24.77)	10.19
32			96.10 (6.29)	48.59 (**12.68**)	24.49 (25.15)	20.03
64				95.68 (12.63)	48.32 (**25.49**)	38.86
80				119.05 (12.55)	60.09 (24.86)	49.35
128					95.21 (25.39)	78.50
160					118.75 (24.69)	97.87

Table 2: Average time per timestep in seconds, with Tflop/s in parentheses (the highest performance for each partition size in **bold**), for an analytic Lennard-Jones potential with $r_{cut} = 5.0\sigma$, using virtual node mode (2 processors per node). For comparison, the last column shows timings for an EAM copper potential using lookup tables.

excessively demanding for multibillion-atom simulations. If one considers the minimal amount of information needed to be double-precision position and velocity vectors for each atom (6 `double`'s) plus perhaps an `int` (atom type for multicomponent systems, or the bond order of each atom) and one or more `float`'s (potential energy, local stress, or other order parameters), the typical checkpoint file requires 50-100 bytes per atom. For a 2-billion atom simulation this is 100-200 GB per checkpoint file. A high-performance parallel file system with 1-10 GB/s bandwidth would make these tolerable (a few minutes to write or read each checkpoint file), but as of April 2005, the Lustre™ global parallel filesystem is not completely operational on BlueGene/L, and striped I/O across 64 NFS disks is unacceptably slow and unreliable (preliminary tests indicate 128 MB/s, or 15-30 minutes per checkpoint file).

Consequently, we have adapted our postprocessing parallel scalable object rendering capability `MD_render` [10] into a subroutine which can be called on-the-fly, directly accessing the SPaSM particle and cell data structures to visualize multi-billion atom data sets. Spheres, cells, cylinders, arrows, and surfaces can be rendered with an arbitrary resolution and 24-bit color-depth. This was realized by developing a graphics library in ANSI C and employing MPI calls for the communication between processes, yielding high performance and guaranteed portability on various supercomputer platforms. The library allows for multiple point light sources, and the color of each surface element of an object is calculated by taking into account the emission, ambient, diffusive, and the specular components. The efficiency was achieved by a parameterization of the individual objects rather than constructing objects like spheres

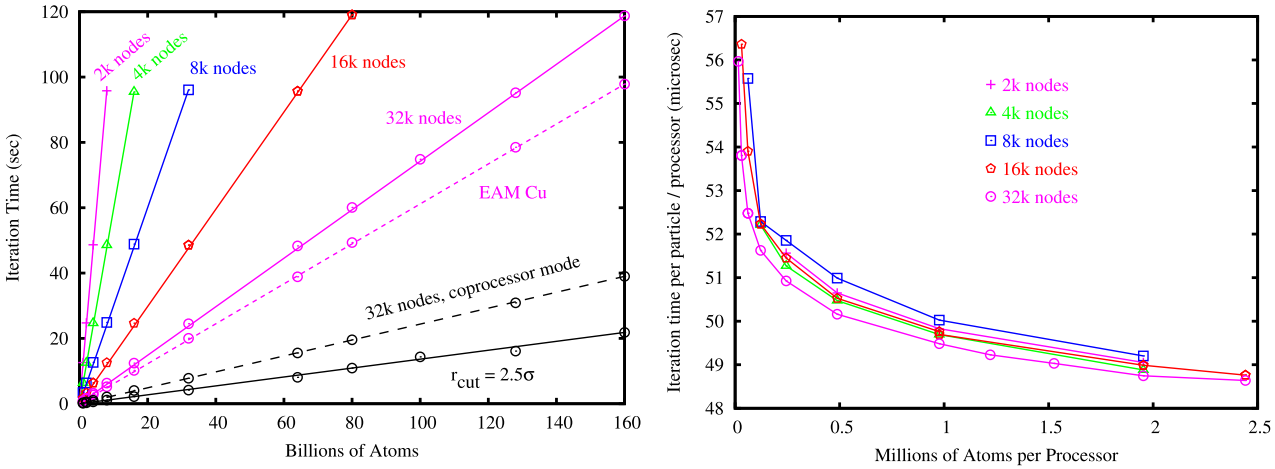

Figure 2: (Left) Average time per timestep in seconds, as a function of problem size on different BlueGene/L partitions in virtual node mode (2 processors per node), for an analytic Lennard-Jones potential with $r_{cut} = 5.0\sigma$. Also shown for comparison are 32k-node results for an EAM copper potential and for $r_{cut} = 2.5\sigma$ LJ, both in virtual node and coprocessor (1 processor per node) mode. (Right) Scaled timings for an analytic Lennard-Jones potential with $r_{cut} = 5.0\sigma$, in virtual node mode on different partition sizes.

with simple polygons (as is the case with graphics libraries like OpenGL [20]). Furthermore, each processor only renders a partial picture of the objects assigned to it (i.e., the atoms or cells in its local memory). A binary tree reduction eventually merges the partial pictures (using depth information assigned to each pixel) into one picture which can be written into a file (with arbitrary resolution). There is almost a 100% parallel efficiency as there is only communication between processors at the end to merge the partial pictures. The rendering scales linearly (assuming the object size is constant) with the number of objects (or even less, since the objects are sorted according to their depths and invisible pixels of objects are not drawn). An earlier version of `MD_render` was used (in the typical standalone postprocessing mode) to generate the 8-million-atom EAM iron shockwave images in [13], and the Rayleigh-Taylor images from a 100-million-atom LJ simulation in [14]. For the typical examples shown below, all images were rendered in 1-5 minutes, depending mainly on the resolution and use of an optional anti-aliasing procedure in which the image is initially rendered with twice the resolution in both directions, then averaged (4:1 pixel ratio) to smooth the final image.

5 Sustained production runs

To demonstrate the capability of BlueGene/L to perform sustained application runs, we present here two simulations performed on a 16,384-node partition (both run in virtual node mode, i.e. utilizing 32,768 processors). Both involve a copper flyer plate (periodic in the x and y directions) initially traveling at a velocity $v_z = 2u_p$ towards a copper target (twice as thick as the flyer plate), resulting in shock waves propagating into both the flyer and target with particle velocity u_p [12, 13]. Upon reaching the free surface at the end of either the flyer or target, a shock wave is replaced by a spreading rarefaction (or release) fan which propagates back into the bulk of the sample, with the target going into tension when the two rarefaction fans (one from the flyer plate and the second from the target) meet. If the tensile stress is sufficiently strong, the material can fail in a process known as spallation. The preliminary simulations here are intended to probe the changes in both shock and spallation behavior for a target which contains a distribution of defects, as might result from radiation damage. As a typical defect which can be created under such conditions, we have chosen to model a porous solid target containing a specified number of spherical voids, each with a radius chosen from a Gaussian distribution with a specified mean size $\overline{r}$. The copper atoms are described using an EAM potential developed by Voter [19].

The first simulation, carried out on 2 February 2005 (on the full 16,384-node system at that time), involved a total of $750 \times 750 \times 950$ fcc unit cells (4 atoms/unit cell), with 150 voids of an average radius $\overline{r} = 3.8$ nm in the target

corresponding to a 0.41% void density. The total number of atoms was 2,131,656,770, and the simulation ran for 7030 timesteps over 13 hours, generating 59 images (each 4800 $\times$ 3600 pixels). The shock was strong enough ($u_p = 0.2$ km/s) to collapse most (if not all) of the voids by emitting partial dislocations (stacking fault loops), as shown in Figs. 3 and 4. Here we use the centrosymmetry parameter [21] to identify the local structure around each atom: atoms in pristine fcc sites are not shown, atoms in hcp stacking faults are grey, and all other atoms (including surfaces and dislocation cores) are red. Fig. 3 shows two views at the end of the simulation (19 ps of simulated time). A third (zoomed-in) view at 18 ps is shown in Fig. 4, both the original 4800×3600 image as well as an expanded view of an 800×600 section, revealing the underlying sphere-rendering which is otherwise impossible to discern.

The second simulation, run on half of the present machine on 10-11 April 2005, included $320 \times 320 \times 500$ unit cells, with only 10 voids in the target ($\overline{r} = 2.6$ nm) equivalent to a negligible 0.11% void density. This simulation included 204,655,465 atoms, and ran for 25,917 timesteps (70 ps) over 8 hours (the SLURM queue-imposed time limit), generating 54 images. A much stronger shock was used in this case ($u_p = 1.0$ km/s), so even rendering a culled set of atoms as in Figs. 3 and 4 quickly becomes too complicated to analyze, since as many as a third of all atoms are identified as non-fcc. To get another perspective on the void collapse under shock compression, and the void nucleation, growth, and linkup into a failed spallation zone under release, we also generated images showing only those cells (actually, subcells with edge length $r_{cut}/2$) which *do not* contain any atoms. Examples for this simulation after both shock waves have released from the free surfaces and created a tensile region are shown in Fig. 5. In these frames, we see the nucleation, growth, and linkup of voids which takes place between 36 ps and 48 ps of simulation time. In Fig. 6 we show two views at 56 ps, when a well-developed spall zone is evident near the right end of the sample. (Due to the periodic boundary condition in the z-direction, the sample has wrapped around the right-hand side and caused a second re-shock, shown as the light grey plane in the top frame. This secondary shock is largely responsible for the significant plastic deformation in the left half of the bottom frame.) To give an idea of the rendering times for these images, the top frame showing 20,571,344 cells was generated in 145 seconds (1600×1200 pixel resolution, with antialiasing), and the 65,832,243 spheres shown in the bottom frame in only 81 seconds (2400×1800 pixels).

6 Conclusion

The 180 Tflop/s peak performance of the current BlueGene/L system (half of the planned final installation) takes into account the dual floating-point units on each processor, and the fused multiply-add instruction. Both of these can be efficiently utilized for certain problems, particularly the linear algebra matrix operations embodied in the Linpack benchmark, which achieved 135.3 Tflop/s (75% of the theoretical peak) [3]. However, this becomes increasingly difficult for more complex applications, so a more realistic upper performance target for such production codes without any special hand-tuning (for the current beta-version XL compilers) would omit both the dual-FPUs and fused multiply-add instructions, giving 1/4 of the 180 Tflop/s peak, or 34 Tflop/s. Our 25.5 Tflop/s measurement for the SPaSM code represents 75% of this target, which is quite reasonable when considering the communication, cache and memory contention, and uncounted operations (e.g., particle redistribution) which make up the remaining 25%.

The multibillion-atom simulations enabled by BlueGene/L now open up to study a wide range of problems where an initial or (more often) a naturally emerging defect/microstructure length scale exceeds that of smaller-scale simulations. Already, multimillion-atom simulations in this category include 8 million-atom EAM Fe simulations required to see product grain structure formed by shock-induced solid-solid phase transformations [13], and the 100 million-atom LJ simulations probing turbulent microstructure formed by the Rayleigh-Taylor fluid instability which occurs when a heavy fluid is placed on top of a lighter one in the presence of gravity [14]. Also, a single billion-atom simulation (lasting 4 days on ASCI White) was carried out to study the long-range dislocation structure produced in a LJ solid during fracture [22]. With such simulations now possible for more realistic interaction potentials, and turnaround times of a day or less, we expect to see an exciting variety of previously intractable problems studied by MD on BlueGene/L.

Acknowledgements

We thank our colleagues, Jim Hammerberg, Brad Holian, and Ramon Ravelo for discussions regarding shock physics-related applications; Steve Louis, Michel McCoy, and James Peery for enabling early access to BlueGene/L; and Ryan

Braby, Jeff Fier, Robin Goldstone, Fred Streitz, and Aidan Thompson for their assistance in surmounting various hardware and software hurdles along the way. Los Alamos National Laboratory is operated by the University of California for the U.S. Department of Energy under contract no. W-7405-ENG-36.

References

[1] N.R. Adiga *et al.*. An Overview of the BlueGene/L Supercomputer. In *SC2002 – High Performance Networking and Computing*, Baltimore, MD, November 2002.

[2] `http://www.llnl.gov/asci/platforms/bluegenel/`

[3] `http://www.llnl.gov/pao/news/news_releases/2005/NR-05-03-09.html`

[4] G. Almasi, S. Chatterjee, A. Gara, J. Gunnels, M. Gupta, A. Henning, J. Moreira, and B. Walkup. Unlocking the Performance of the Bluegene/L Supercomputer. In *SC2004 – High Performance Computing, Networking and Storage Conference*, Pittsburgh, PA, November 2004.

[5] M.S. Daw and M.I. Baskes. Embedded-atom method: Derivation and application to impurities, surfaces, and other defects in metals. *Phys. Rev. B* **29***, 6443–6453 (1984).*

[6] M.S. Daw, S.M. Foiles, and M.I. Baskes. The Embedded-Atom Method – A Review of Theory and Applications. *Mater. Sci. Rep.* **9***, 251–310 (1993).*

[7] J.H. Rose, J.R. Smith, F. Guinea, and J. Ferrante. Universal features of the equation of state of metals. *Phys. Rev. B* **29***, 2963–2969 (1984).*

[8] D.M. Beazley and P.S. Lomdahl. Message-passing multi-cell molecular dynamics on the Connection Machine 5. *Parallel Computing* **20***, 173–195 (1994).*

[9] P.S. Lomdahl, P. Tamayo, N. Grønbech-Jensen, and D.M. Beazley. 50 GFlops Molecular Dynamics on the Connection Machine 5. In *Proceedings of Supercomputing 93* (IEEE Computer Society Press, Los Alamitos, CA, 1993), pp. 520-527.

[10] K. Kadau, T.C. Germann, and P.S. Lomdahl. Large-Scale Molecular-Dynamics Simulation of 19 Billion Particles. *Int. J. Modern Phys. C* **15***, 193–201 (2004).*

[11] S.J. Zhou, D.L. Preston, P.S. Lomdahl, and D.M. Beazley. Large-Scale Molecular Dynamics Simulations of Dislocation Intersection in Copper. *Science* **279***, 1525–1527 (1998).*

[12] B.L. Holian and P.S. Lomdahl. Plasticity Induced by Shock Waves in Nonequilibrium Molecular-Dynamics Simulations. *Science* **280***, 2085–2088 (1998).*

[13] K. Kadau, T.C. Germann, P.S. Lomdahl, and B.L. Holian. Microscopic View of Structural Phase Transitions Induced by Shock Waves. *Science* **296***, 1681–1684 (2002).*

[14] K. Kadau, T.C. Germann, N.G. Hadjiconstantinou, P.S. Lomdahl, G. Dimonte, B.L. Holian, and B.J. Alder. Nanohydrodynamics simulations: An atomistic view of the Rayleigh-Taylor instability. *Proc. Nat. Acad. Sci. (USA)* **101***, 5851–5855 (2004).*

[15] S.J. Zhou, D.M. Beazley, P.S. Lomdahl, and B.L. Holian. Large-Scale Molecular Dynamics Simulations of Three-Dimensional Ductile Failure. *Phys. Rev. Lett.* **78***, 479–482 (1997).*

[16] D.M. Beazley and P.S. Lomdahl. Lightweight Computational Steering of Very Large Scale Molecular Dynamics Simulations. In *Proceedings of the 1996 ACM / IEEE conference on Supercomputing CD-ROM (IEEE Computer Society, 1996).*

[17] D.M. Beazley and P.S. Lomdahl. Controlling the data glut in large-scale molecular-dynamics simulations. *Computers in Physics* **11***(3), 230–238 (1997).*

[18] M.S. Warren, T.C. Germann, P.S. Lomdahl, D.M. Beazley, J.K. Salmon. Avalon: An Alpha/Linux cluster achieves 10 Gflops for $150K. In *Proceedings of the 1998 ACM / IEEE SC98 Conference CD-ROM (IEEE Computer Society, 1998).*

[19] A.F. Voter. Parallel replica method for dynamics of infrequent events. *Phys. Rev. B* **57***, 13985–13988 (1998).*

[20] A. Sharma, R.K. Kalia, A. Nakano, and P. Vashishta. Scalable and portable visualization of large atomistic datasets. *Comp. Phys. Comm.* **163***, 53–64 (2004).*

[21] C.L. Kelchner, S.J. Plimpton, and J.C. Hamilton. Dislocation nucleation and defect structure during surface indentation. *Phys. Rev. B* **58***, 11085–11088 (1998).*

[22] F.F. Abraham, R. Walkup, H. Gao, M. Duchaineau, T.D. de la Rubia, and M. Seager. Simulating materials failure by using up to one billion atoms and the worlds fastest computer: Work-hardening. *Proc. Natl. Acad. Sci. (USA)* **99***, 5783–5787 (2002).*

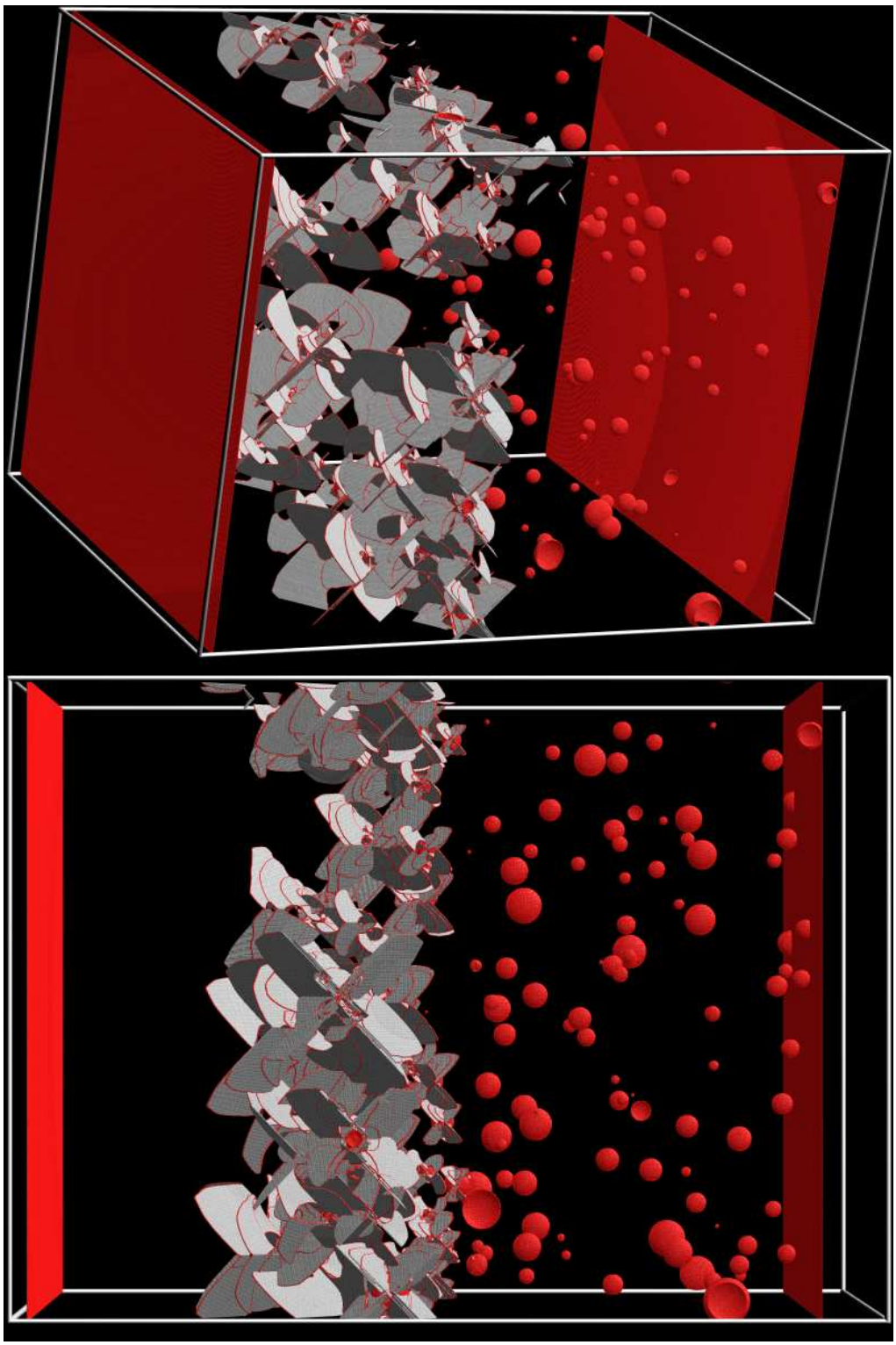

Figure 3: Two perspectives showing only defect atoms (out of a total of 2.1+ billion atoms), 19 ps after a Cu perfect crystal flyer plate (left third of sample) has impacted a porous Cu target, resulting in shock waves to the left and right.

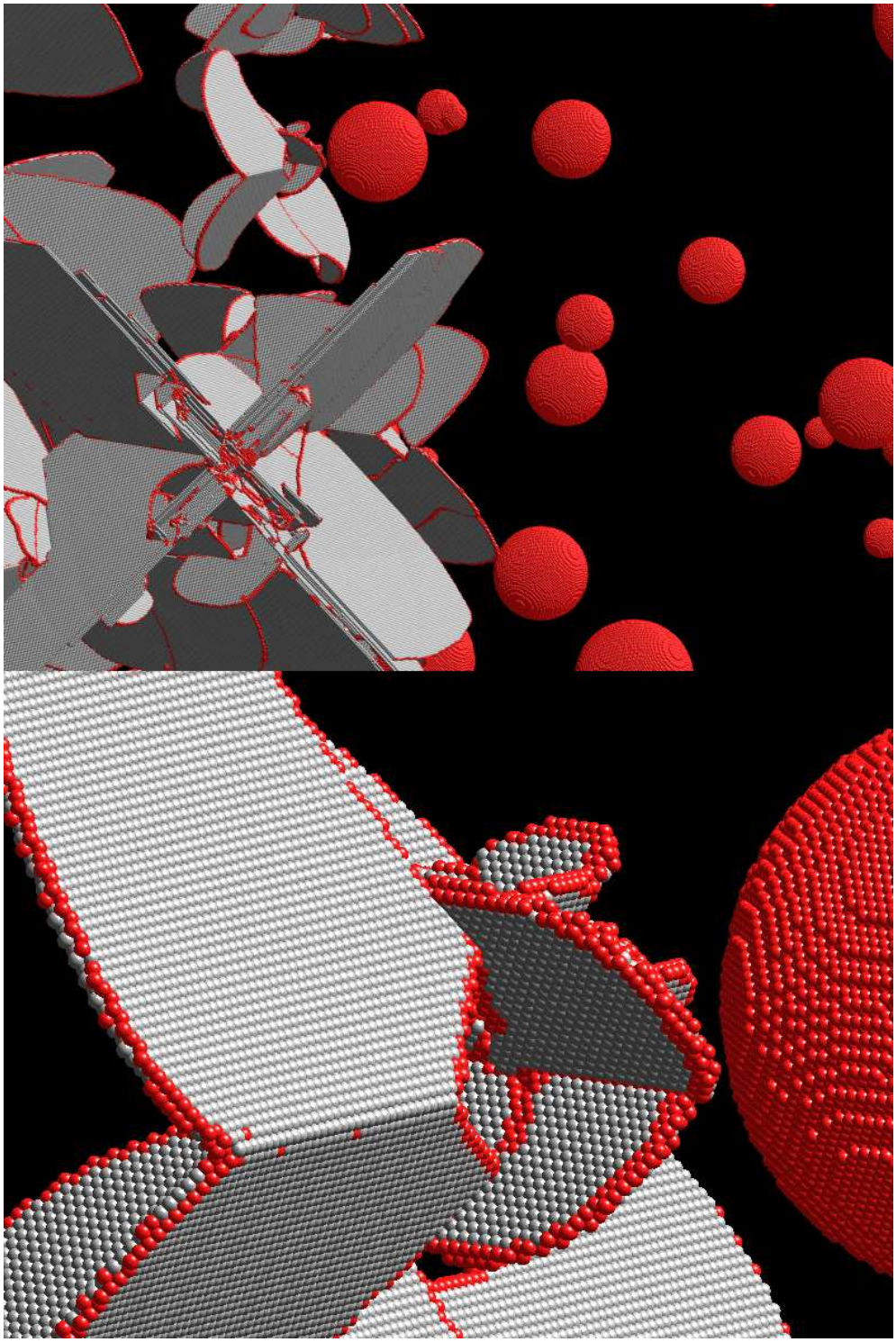

Figure 4: A third view of the same 2.1 billion atom simulation, at 18 ps. The top frame shows the as-rendered 4800 $\times$ 3600-pixel image, while the lower frame is zooming in on an 800 $\times$ 600-pixel region near the top, just left of center.

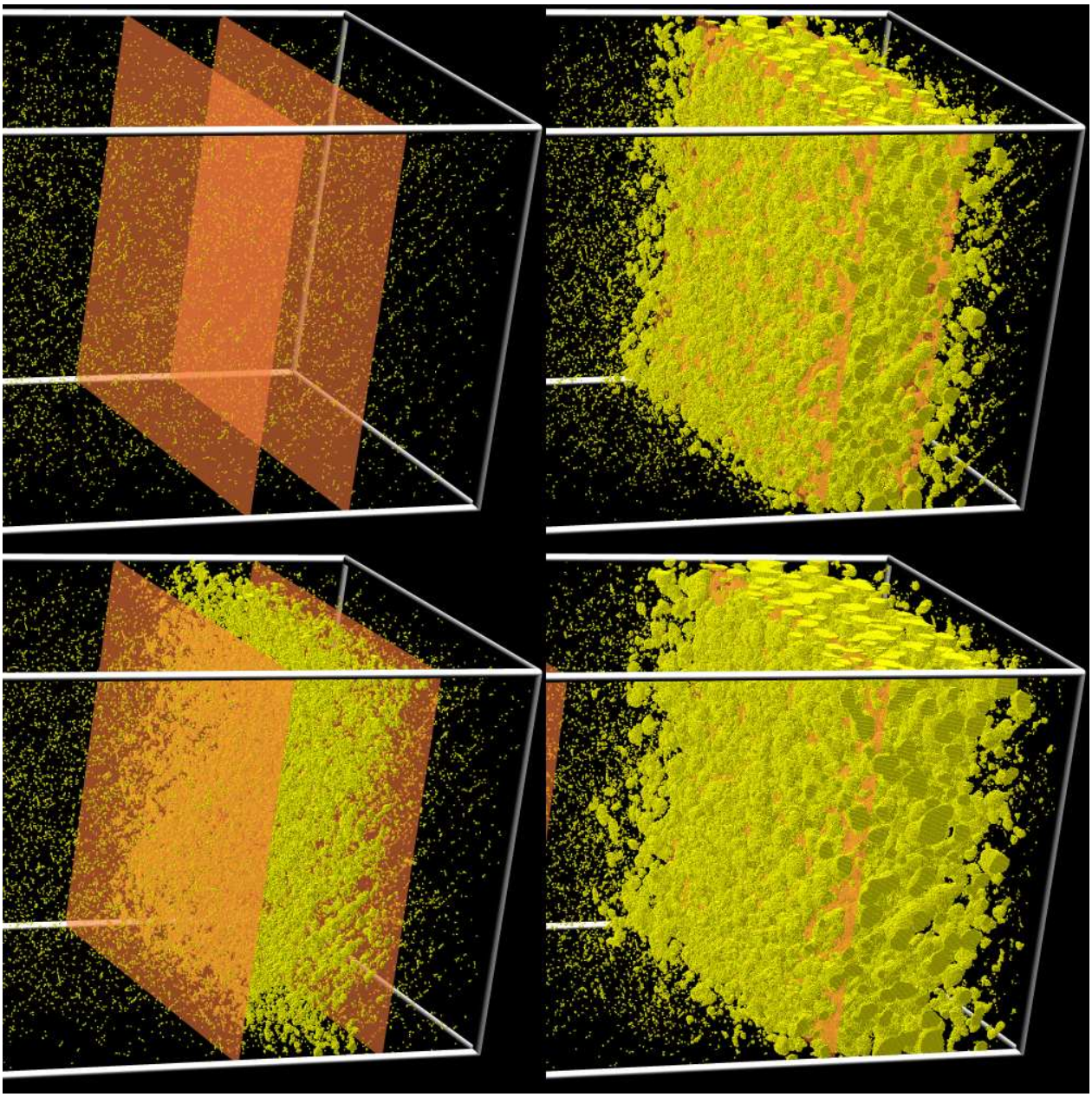

Figure 5: Images showing only those Eulerian cells within the bulk material which do not contain any atoms, from a 200+ million-atom simulation of Cu shock compression and release. Approximately 11 ps after the initial impact, the shock wave in the flyer plate releases from the free surface, and release from the far surface of the target occurs at 22 ps. The two spreading rarefaction fans collide and lead to a tensile zone at 33 ps, just before the top left frame (36 ps). The two planes shown here are bounding the tensile zone, but the sample only contains scattered residual defects from shock compression-induced plastic deformation and subsequent release. Void nucleation is evident in the bottom frame (40 ps), with growth and linkup in the next two frames on the right (44 ps and 48 ps).

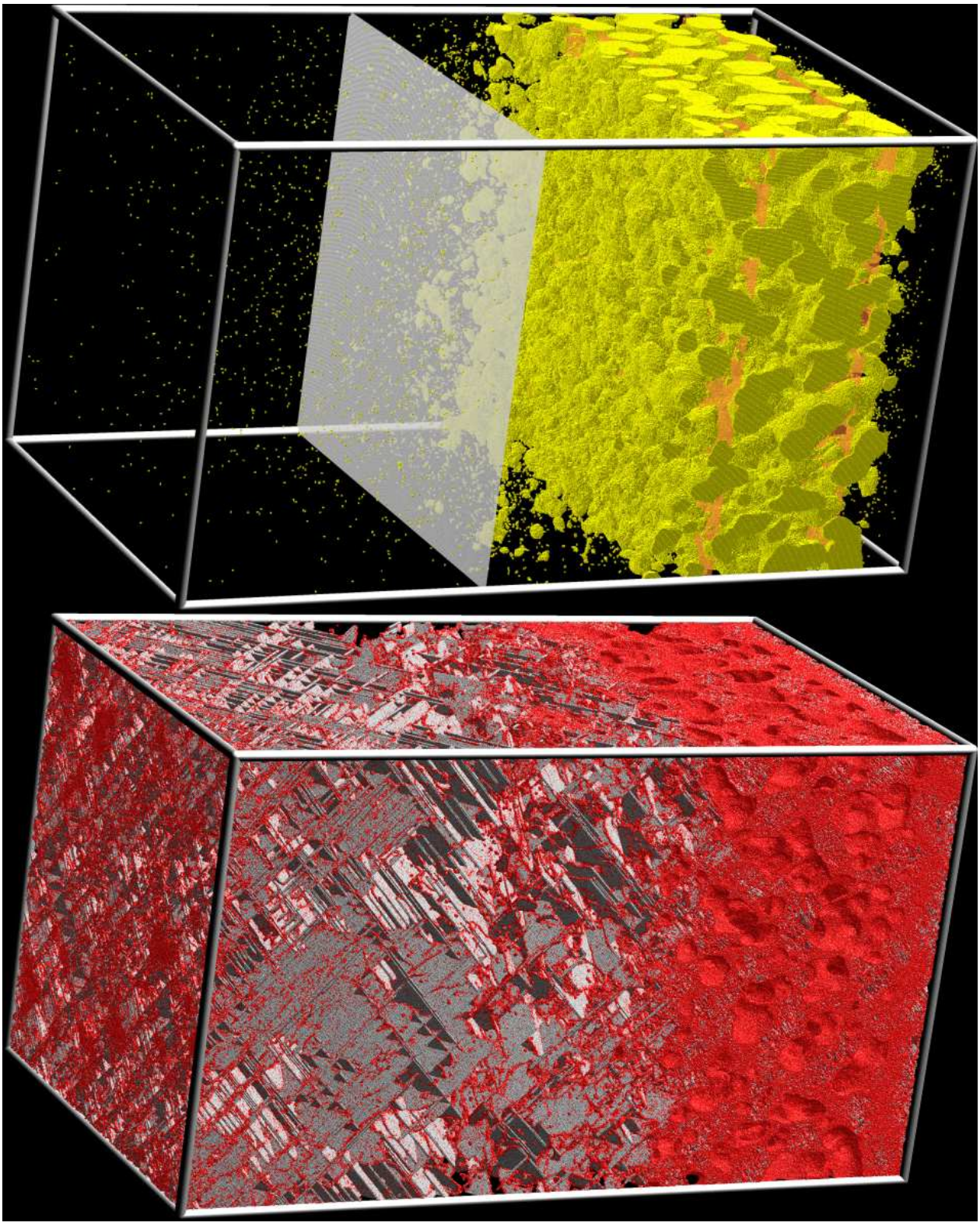

Figure 6: Two views of the same 200+ million-atom simulation at 56 ps, after void linkup has led to a clear spall zone. The top panel shows the empty Eulerian cells as in Fig. 5, while the bottom shows non-fcc atoms as in Figs. 3 and 4.