LAWRENCE
LIVERMORE
NATIONAL
LABORATORY

UCRL-CONF-223218

# Simulating Solidification in Metals at High Pressure

F.H. Streitz, J.N. Glosli, M.V. Patel, B.Chan, R.K. Yates, B.R. de Supinski, J.C. Sexton, J.A. Gunnels

July 28, 2006

SciDAC 2006
Denver, CO, United States
June 25, 2006 through June 29, 2006

# Simulating Solidification in Metals at High Pressure:
## *The Drive to Petascale Computing*

**Frederick H. Streitz[1], James N. Glosli[1], Mehul V. Patel[1], Bor Chan[1], Robert K. Yates[1], Bronis R. de Supinski[1],James Sexton[2], John A. Gunnels[2]**

[1] Lawrence Livermore National Laboratory*, Livermore, CA 94550 USA
[2] IBM, Thomas J. Watson Research Center, Yorktown Heights, NY, USA

E-mail: streitz1@llnl.gov

**Abstract.** We investigate solidification in metal systems ranging in size from 64,000 to 524,288,000 atoms on the IBM BlueGene/L computer at LLNL. Using the newly developed *ddcMD* code, we achieve performance rates as high as 103 TFlops, with a performance of 101.7 TFlop sustained over a 7 hour run on 131,072 cpus. We demonstrate superb strong and weak scaling. Our calculations are significant as they represent the first atomic-scale model of metal solidification to proceed, without finite size effects, from spontaneous nucleation and growth of solid out of the liquid, through the coalescence phase, and into the onset of coarsening. Thus, our simulations represent the first step towards an atomistic model of nucleation and growth that can directly link atomistic to mesoscopic length scales.

### 1. Introduction

Understanding the properties of matter under extreme conditions is fundamental to researchers in fields as disparate as astrophysics, planetary science and nuclear physics. Scientists increasingly rely on computer models to develop this understanding since experiments are often impossible (or extremely difficult) at pressures and temperatures of interest.

Simulating complex systems (such as transition metals) under extreme conditions poses several difficulties. As the disorder in the system increases, describing the relevant physics requires increasingly large systems. Further, the length of time necessary to model non-equilibrium behavior is also increasing. Systems with hundreds of billions of atoms have been simulated but these typically use computationally inexpensive pair (or pair-like) potentials that are inadequate for the complex systems that we are discussing. Billion atom simulations are feasible using highly accurate many-body quantum-based interaction potentials, such as MGPT[6, 7, 8, 9]. Ultimately, however, the required time scales to model non-equilibrium behavior require a shorter wall-clock time to solution than has been previously achieved. The new generation of massively parallel computers, such as the IBM BlueGene/L (BG/L) at Lawrence Livermore National Laboratory, provide the computational power necessary to achieve that improvement. This improvement entails reducing the size of the problem on each processor, a severe test of the strong scaling limit for a code.

We developed a parallel classical molecular dynamics (MD) code (*ddcMD*) that achieves the required strong scaling and efficiently implements the MGPT potentials on BG/L. Using *ddcMD* on BG/L, we have modeled the pressure-induced solidification of molten tantalum.

Our simulations have yielded the first determination of the minimum sized needed to model solidification through the rapid nucleation and growth and phase, during which individual grains are formed, to the coarsening phase, during which the metal solidifies into a characteristic microstructure.

The rest of the paper is organized as follows: after a brief overview of the BG/L architecture and the MGPT interaction potentials, we describe the features of *ddcMD* that enable simulation of complex systems under extreme conditions (namely, a particle-based domain decomposition algorithm), including a discussion of the performance and scaling of the code on up to 131,072 processors of BG/L. We then present our simulation results.

## 2. BlueGene/L Architecture

BlueGene/L (BG/L) is a massively-parallel scientific computing system developed by IBM in partnership with the Advanced Simulation and Computing program (ASC) of the US Department of Energy's National Nuclear Security Agency [24, 25]. BG/L's high-density cellular design gives very high performance with low cost, power and cooling requirements. The 65,536-node system at LLNL is at this writing the fastest computer in the world, having achieved a performance of 280.6 Tflop/s on the Linpack benchmark in November, 2005.

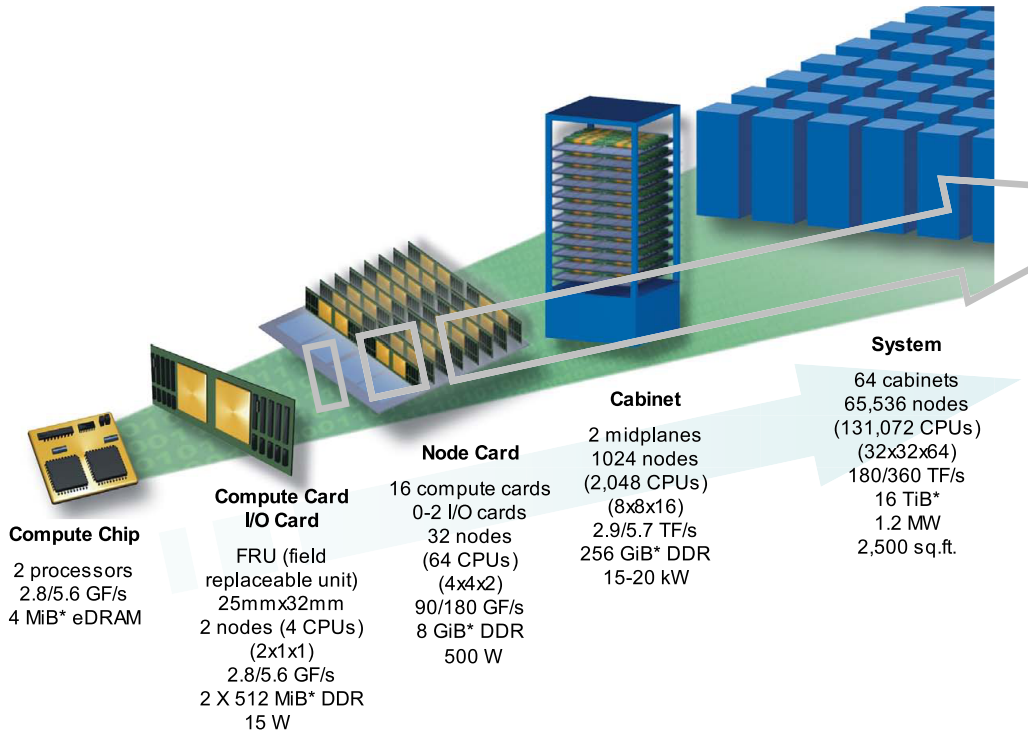


**Figure 1.** High-Level Schematic of the BlueGene/L Platform

A compute node of BG/L is composed of only 10 chips: its 700 MHz compute ASIC plus nine DRAM main memory chips. This highly integrated design drastically lowers power consumption and space requirements while favoring communication and memory performance. The BG/L ASIC has two independent PowerPC 440 cores, each capable of two floating point operations per cycle including fused multiply-adds, yielding a theoretical peak of 4 FLOP/s per cycle, several independent network controllers, three levels of cache, and memory controllers. The two cores on each chip are identical, with their own dual floating point units (FPUs) and symmetric

access to resources. Though each FPU is capable of two operations per cycle, the operations are not independent: the second floating point pipe is usable only by 2-way SIMD instructions, or by 2-way "SIMOMD" (i.e., single instruction, multiple operation, multiple data) instructions [25]. The theoretical peak of a single node is 2.8 GFLOPs, hence 367 TFLOP/s for the current 65,536-node system.

The system software supports two modes for applications to use the cores. In communication coprocessor mode there is a single MPI task per node, with one processor running the application, offloading much of the work of message passing to the second processor. In virtual node mode, two MPI tasks run on each node, one task per core. The MPI implementation, based on Argonne National Lab's MPICH-2, has been adapted to make efficient use of BG/L's three independent custom networks (whose controllers are integrated into the BG/L ASIC). Point-to-point and all-to-all communications are handled by a three-dimensional torus. Broadcasts, reductions, and barriers are performed over two tree-topology networks with high bandwidth and low latency (e.g., a full-system barrier in under 2 $\mu$s).

## 3. Classical Molecular Dynamics and MGPT interaction potentials

Classical molecular dynamics (MD) modeling of atomistic processes requires specification of the total potential energy of the system as a function of the coordinates of all particles (atoms). The force on each particle is calculated from the gradient of the energy with respect to it's position:

$$\vec{f}_i = -\nabla_i U\{\vec{r}_i\} \ .$$

The equations of motion for a collection of interacting atoms is expressed as a simple set of first order ODE's:

$$\begin{aligned} d\vec{x}_i/dt &= \vec{v}_i \\ d\vec{v}_i/dt &= \vec{f}_i(x_1, x_2, \ldots, x_N) \ , \end{aligned}$$

where $\vec{x}_i$, $\vec{v}_i$ and $\vec{f}_i$ are respectively the position, velocity and force of the $i^{th}$ atom. A variety of finite difference algorithms can be used to numerically integrate these equations; for the most part none are computationally intensive relative to the evaluation of the forces ($f_i$).

The potential function chosen, $U\{\vec{r}_i\}$, ultimately determines the physical accuracy of the simulation. Common choices for interaction potentials fall into several classes. A pair-wise potential, such as the well-studied Lennard-Jones potential, describes the energy of interaction as simply $U = \sum_i \sum_j \phi(r_{ij})$, where $r_{ij}$ is the separation between atoms $i$ and $j$. Although pair-wise potentials were derived for the study of noble gases, their simplicity leads to their use in a variety of systems. Potentials based on effective medium theory are only slightly more complicated. The embedded atom method (EAM) potentials are the best known example of this class, in which the energy of interaction is written in two parts:

$$U = \sum_i \sum_j \phi(r_{ij}) + \sum_i F_i\left(\rho_i\right) \ , \qquad \rho_i = \sum_j f(r_{ij})$$

The additional term on the right is a many-body term – this term represents the energy $F_i$ to embed a particle at $\vec{r}_i$ in local electron density $\rho_i$, which depends on the position of the $j$ neighbors of atom $i$. The EAM potentials are excellent models of metallic bonding for simple (closed-shell) metals, such as copper or gold, and have been used extensively since their creation in 1983 [13, 14].

Pair-wise (and most EAM-type) potentials produce forces that are radially symmetric on the atoms [15]. This constraint is acceptable in modeling materials for which these potentials were

designed, such as noble gases or closed-shell metals, since their bonding can be approximated closely as spherically symmetric. However, metal elements containing partially-filled $d-$bands or $f-$bands have more complicated bonding environments that result in angularly dependent forces on the atoms. Accurate modeling of these elements requires a more sophisticated interaction potential that accounts for these forces. In order to model solidification in these metals we use MGPT potentials that are derived from first-principles generalized pseudopotential theory (GPT) using density functional theory [6, 8, 9]. Using GPT, we write the total energy as an explicit function of volume:

$$\begin{aligned} U = NV_0(\Omega) &+ \sum_{ij} F(r_{ij}, \Omega) \\ &+ V_3(\Omega) \sum_{ijk} Tr(H_{ij}H_{jk}H_{ki}) \\ &+ V_4(\Omega) \sum_{ijkl} (Tr(H_{ij}H_{jk}H_{kl}H_{li}) \end{aligned}$$

where $\Omega$ is the volume per atom and $V_0, \ldots, V_4$ are functions of $\Omega$. $F$ is a simple function of $\Omega$ and the bond length $r$, and $H_{ij}$ is a hopping matrix along bond $ij$. $H$ is either a $5 \times 5$ matrix for $d-$electron elements or $7 \times 7$ matrix for $f$ electron elements, and the summation is over all pairs, triples, quadruples, etc... of atoms. This series converges to the exact result. For these simulations we retain terms to fourth order, explicitly including interactions up to quadruples. Written in this way, the use of the above equation to calculate energies and forces (which constitutes the kernel of the *ddcMD* code) is comprised of traces of products of very small (no larger than $7 \times 7$) matrices. The extra expense of these potentials causes MGPT calculations to be approximately a factor of twenty more costly than an equivalently sized simulation using EAM potentials.

## 4. Domain Decomposition on BG/L

An innovative domain decomposition scheme is the key to the outstanding performance achieved by *ddcMD* on BG/L. Our particle-based decomposition strategy allows the processors to compute potentials for overlapping spatial regions, which is an essential property for an MD code that supports arbitrarily low numbers of atoms per processor. In addition to this novel scheme for assigning atoms to processors, the domain decomposition scheme of *ddcMD* was designed to minimize redundant calculations in order to minimize time-to-solution. In this section, we describe the domain decomposition scheme used *ddcMD*, including the details of its communication strategy.

Traditional decomposition algorithms determine the atoms for which a given processor computes potentials through a geometry-based (i.e., spatial) decomposition scheme [10, 11, 12]. The simulated region is divided into smaller regions or *zones*, each of which is assigned to a processor. Communication is determined by proximity: typically only zones that share a boundary (i.e., nearest neighbors within the simulated space) need to exchange locations of the atoms. This limited communication is achieved through the use of the interaction cut-off distance: provided that zones are at least as large as that distance in all three dimensions, then atoms in non-neighboring zones do not interact. Although not an inherent limitation, MD codes that use a geometry-based decomposition frequently assume this communication pattern, which restricts the lower limit of particles per processor that they can support. Also, they cannot achieve good load balance for simulations of systems with large density inhomogeneties, such as cracks and voids.

The systems simulated with *ddcMD* would be especially impacted by these limits. Not only do MGPT potentials have relatively long cut-off distances, but Further, solidification simulations requires fairly long time-scales. Eventually, *ddcMD* will be used to simulate systems with many

cracks and voids. Finally, *ddcMD* is intended to run on truly massively parallel systems, such as BG/L, where load balance issues are important. These factors combine to require a decomposition scheme that supports unprecedented strong scaling: the code must support reduced run-times with very few atoms per processor.

The *ddcMD* particle-based domain decomposition scheme does not assume spatially-implicit communication partners. Instead, each processor maintains a communication list that explicitly tracks the other processors with which it must communicate. The processors on the list are exactly the ones that own atoms within the cut-off distance of an atom owned by that processor. This list allows *ddcMD* to limit communication to (almost) the minimum required during the normal simulation step. This savings comes at the cost of communication required to maintain the communication lists.

**dcInit()**
Initialize function and data pointers and global quantities

**ddcAssignment()**
Specify locations of domain center $R_d$
Assign Particle $r_i$ to the nearest domain center
Send particle to remote domain if not local
Receive particles sent from remote domains
Wait for communication to finish
Determine radius of domain $w_d = \mathrm{Max}\,|r_i R_d|$
Perform an all to all communication of domain information ($R_d$ and $w_d$ ).

**ddcSendRecvList()**
Determine "interacting" domains:
Remote domain $r$ interacts with local domain $d$ if $|R_r - R_d| < w_r + w_d + r_{cut}$
For each "interacting" remote domain $r$:
Determine local particles that interact with remote domain: Particle $i$ interacts with domain $r$ if $|r_i - R_r| < w_r + r_{cut}$
Communicate this list to remote domain $r$
Receive list from remote domain
Construct local-remote particle interaction list.

**ddcUpdateParticle()**
Communicate interacting particles with remote domain[s]

**ddcUpdateForce()**
Communicate and accumulate partial forces from remote domains

**ddcUpdateGlobal()**
Accumulate partial energies, pressures, etc.

**Figure 2.** Description of main *ddc* Routines

Figure 2 lists the routines used in *ddcMD*. The `ddcInit()` routine initializes *ddc* and is called only once, at the beginning of the code. The routine `ddcAssignment()` assigns particles to domains. The routine `ddcSendRecvList` maintains the communication lists. The actual per step communication is performed in `ddcUpdateParticle()`. The routine `ddcUpdateForce()` uses increased communications to reduce redundant calculations. Finally, the routine `ddcUpdateGlobal()` accumulates various partial sums.

The distributed algorithm that `ddcAssignment()` uses to maintain the communication lists begins with each processor calculating the center of its domain and a bounding sphere. This information is exchanged between all processors through a call to `MPI_Allgather()`, from which we determine at each processor the set of processors that own domains that overlap with it. This set is then pruned by a communication that determines which intersections of the bounding spheres actually contain particles, thus limiting the communication lists to processors that actually have particles separated by less than $r_{\mathrm{cut}}$. This set would be exactly those processors with interacting particles, except that we increase the cut-off distance beyond the interaction distance to allow the communication list to be used for multiple time steps.

Spatial decomposition schemes must periodically update ownership of atoms while the explicit communication lists allow a simulation to proceed without changing ownership. However, the communication list can become unacceptably large as the particles move over time. Thus, the `ddcAssignment` routine should be called periodically to limit communication costs. Initially, this routine uses a spatial assignment. Similarly to traditional decomposition schemes, it assigns atoms to the closest domain center. Currently, subsequent calls reassign atoms to the processor with the closest domain center. By adjusting the domain centers one could in principle provide load balancing, which would support hardware systems with heterogeneous processors as well as simulated systems with cracks and voids.

The `ddcUpdateForce()` routine is not strictly part of the *ddcMD* particle-based domain decomposition strategy. As already stated, it is used to eliminate many of the redundant

calculations typically present in MD simulations. Every force calculation is symmetric; when the particles are not owned by the same processor, either both processors compute the same force or one must communicate its result to the other. For simple pair potentials, the communication costs outweigh the computation cost, and redundant computation leads to the fastest time-to-solution. However, the redundant computation increases time-to-solution with complicated many-body potentials, such as the MGPT potential. The `ddcUpdateForce()` routine handles the communication of forces in this case.

The following snippet of pseudo-code demonstrates the use of *ddc* calls to parallelize an MD calculation:

```
main(){
  ddcInit()
  read_input()
  ddcAssignment()
  ddcSendRecvList()
  ddcUpdateParticles()
  loop  {
    Evaluate_Energies_Forces()
    ddcUpdateForces()
    ddcUpdateGlobal()
    Time_Step_Equation_of_Motion()
    if (I/O) Write_Snapshot()
    if (finished) break
    if (need_better_efficiency) ddcAssignment()
    if (needed) ddcSendRecvList()
    ddcUpdateParticles()
  }
}
```

The particle-based domain decomposition strategy provides the strong scaling behavior required by *ddcMD*, as we demonstrate in a later section.

## 5. Performance of *ddcMD* on BlueGene/L

This section describes the performance of our MGPT MD simulation code on BG/L. We start with the application-oriented scaling performance numbers, which are the most important measure of performance since they reflect the time to solution, and show how we have achieved our code design objectives. The code exhibits excellent scaling in both the weak and strong scaling limits. Next, we describe how we compute the floating point operations (Flops) required for each simulation, and then apply this methodology to compute overall performance in our benchmark calculations and production runs. This leads us to a sustained overall performance of over 100 TFlop/s in our largest run.

Although the total flop rate varied with the number of tasks and the number of particles per task, *ddcMD* exhibited excellent weak and strong scaling behavior. Figure 3 demonstrates that time-to-solution is consistent over a variety of particles per node as the number of processors is increased. Even more impressive, we saw continuous speedup during strong scaling, a shown in Figure 3. Wall clock time (or time to solution) for a 16,384,000 particle system decreased as more processors were added, up to and including the entire system. The code achieves its objective of scaling to very low particles per processor. In the end, we saw speedup on a classical MD calculation with only 8 particles per processor - such scaling is unprecedented.

Table 1 shows the breakdown between various routines for our full system run discussed in the following section. These results demonstrate that our particle-based decomposition scheme

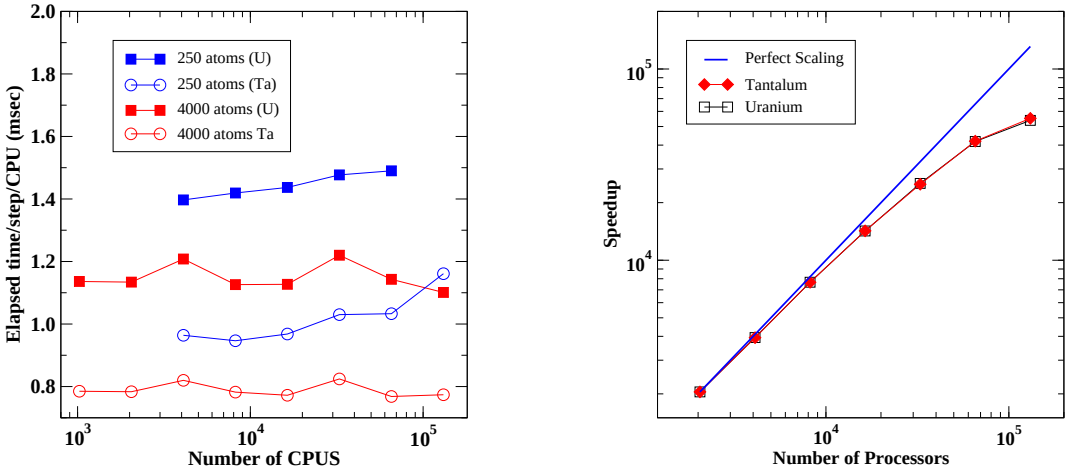


**Figure 3.** (a) Weak scaling performance of *ddcMD* on BlueGene/L, demonstrating how well the code enables a scale-up in the problem size by running on more processors (with the average number of particles/processor held constant). Perfect linear scaling would appear as a horizontal line on this plot. (b) Strong scaling performance of *ddcMD* on BlueGene/L. Speedup achieved by running a fixed size job (16M atoms) on an increasing number of processors.

limits communication costs, as intended. This low communication cost occurs despite the large fraction of every domain that must be communicated to remote domains due to the small spatial size of the domains.

| | | |
|---|---|---|
| I/O | Output | 4.10% |
| MD | Force | 86.57% |
| | Verlet | 4.76% |
| DDC | Barrier | 3.06% |
| | DDC computation | 0.94% |
| | Communication + misc. | 0.57% |

**Table 1.** Breakdown of time spent during ddcMD simulation on 128K CPUs. This profiling data is averaged over the sustained 101.7 TFlop calculation described in the text.

Load balance is a more significant concern for efficient parallelization of *ddcMD*. One measure of the lack of uniformity in the computational load is the percentage of time that domains spend waiting for the last domain to finish. This time is denoted as Barrier time in Table 1 and is about 3% of the total wall clock time. For our largest tantalum simulations, this routine accounts for approximately 7% of the run time, which indicates that load balance could be a problem at even larger node counts. The flexibility of the ddc algorithm allows us to implement a dynamic load balancing algorithm to mitigate this effect, however this has not been exploited yet.

In the extreme strong scaling limit, we find that the time to solution decreases with the number of particles per processor all the way down to 8 particles per processor. Such strong scaling enables us to utilize the *ddcMD* code on BlueGene/L to access not only extremely large systems (billions of atoms) but, when necessary, smaller systems for extremely long times. Our ability to scale strongly down to very few particles per processor is all the more remarkable when one considers how much communication must take place - in the strong scaling limit there are more than an order of magnitude more remote particles per domain than local!

Although we have demonstrated that the performance of *ddcMD* running on BG/L has been exemplary, we have also uncovered some shortcomings in our initial approach to working in this environment. The restartability of our algorithm allowed us to recover successfully from the occasional hardware failure that is to be expected with this number of CPUs. After running on 32,768 processors for approximately 60 hours (including a non-stop run of 22 hours), we encountered a week-long challenge of manipulating (changing permissions, copying/moving, consolidating, etc.) and post-processing this highly distributed data set. Due to file system limitations (notably the inability of NFS to handle large amounts of meta-data), these tasks were much more cumbersome than anticipated. We have since mitigated these problems by writing out fewer (but larger) data files, instead of the one file/CPU paradigm used in the initial runs. Surprisingly, writing and reading from a single large data files also proved non-ideal. In this case, contention arising from the shear number of processes attempting to access the file proved to be the issue.

## 6. Simulation results

The nucleation and growth of a solid out of a liquid is a ubiquitous phenomenon that though well-studied, is hardly well understood. One of the difficulties associated with this process is the broad range of applicable time and length scales: although the initial nucleation of solid-like regions occurs on the atomic scale, the subsequent rapid growth of these nuclei takes a large fraction of a nanosecond and produces grain-like objects which can involve hundreds of thousands of atoms. The eventual coalescence of these objects results in an interconnected network of grains and grain-boundaries which span the entire structures. Computer simulations are a natural way to study this process, but the necessarily finite size of such models is known to color the results. The expectation has always been that a sufficiently large simulation cell would circumvent these issues, producing an accurate model of reality. Researchers have disagreed on exactly how large such a cell must be, with estimates ranging from a few hundred[2] to tens of thousands[4, 5]. To address this question, we performed a series of calculations investigating solidification in metals using MGPT potentials in systems ranging from 64,000 atoms to 525,828,000 atoms. We implemented an NVT ensemble (fixed *N*umber of particles, *V*olume and *T*emperature) for these simulations, with temperature control provided by application of a Langevin thermostat[17, 16]. We used the symplectic integration scheme (with a time step of 1.5 fs), as described by Martyna and co-workers with a slight modification to incorporate the stochastic thermostat[18, 19, 20]. The thermal time constant was set to 1 ps. We use a slightly modified version of the the $Q_6$ local order parameter

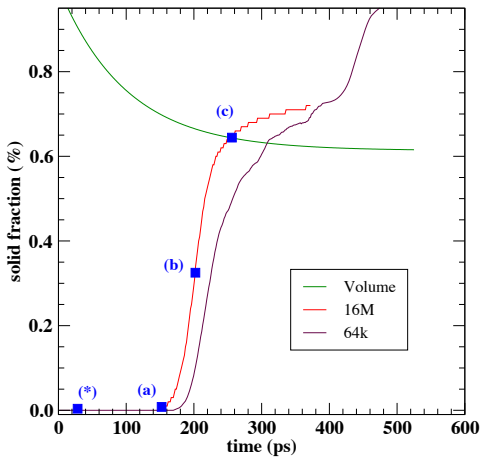


**Figure 4.** Percentage of the simulation cell that has solidified as function of time, along with the compression ratio. The melt curve for Ta is shown in the inset, which also depicts the isothermal compression path (black arrow). The labeled points refer to (*) the crossing of the equilibrium melt curve, (a) the onset of nucleation, (b) the period of explosive growth and (c) the onset of coalescence.

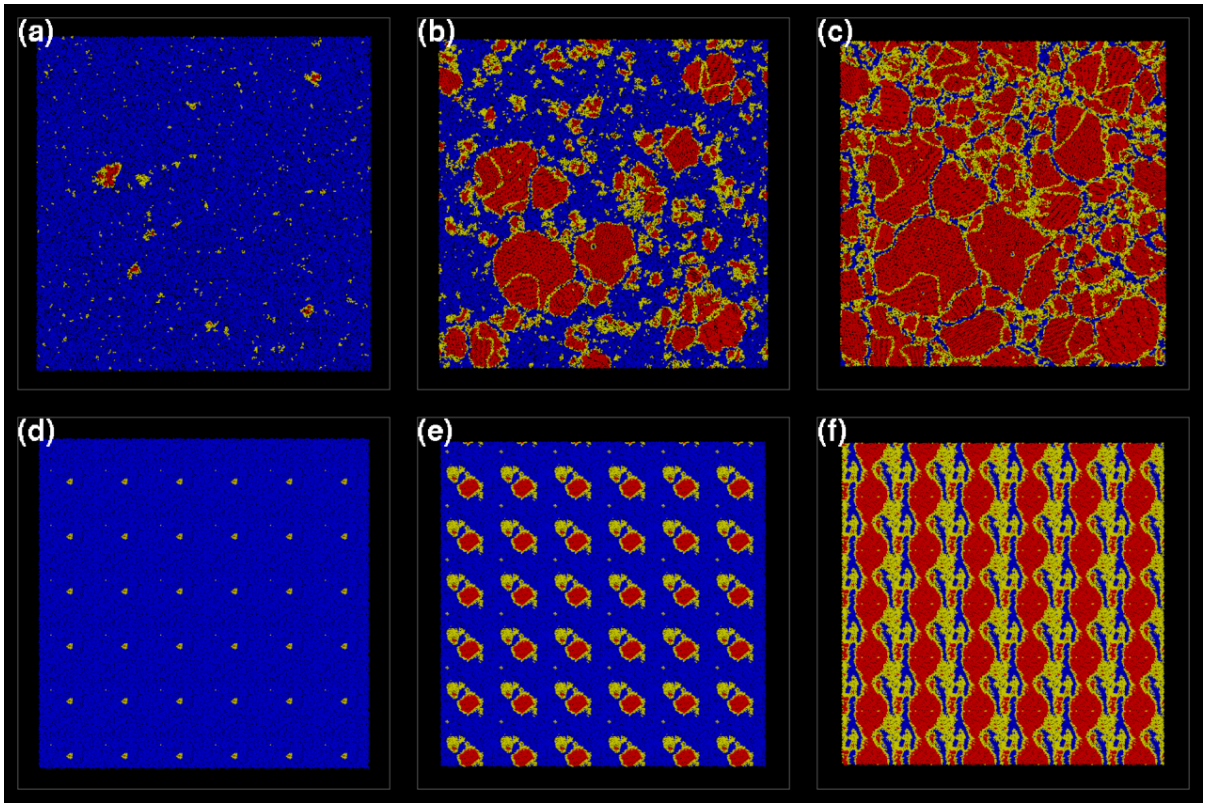


**Figure 5.** Cross sections of the 16M-atom (a-c) and 64k-atom (d-f) simulation taken at equivalent times during the solidification. Labels refer to points shown in Figure 4.

as defined by ten Wolde *et al.* in order to discriminate between "liquid" and "solid" atoms in the cell [21]. This quantity is a measure of the correlation in local bond angles between neighboring atoms, which has a value near unity in a crystalline solid (see Appendix). The initially liquid collection of tantalum atoms was isothermally compressed by exponentially ramping the volume from an initial value of 121.6 au to 74.6 au with a time constant of 100 ps. We show in Figure 4 the evolution of the solid fraction of the system, as well as the compression. The point marked (*) notes the crossing of the equilibrium melt pressure for this temperature (43 GPa). Nucleation is seen to occur at (a), nearly 100 ps after the equilibrium melt pressure was reached.

Rapid growth of the solid grains then occurs (b), with the solidification rate far exceeding the compression rate. This high growth period ends at (c) when the grains have grown into each other (the onset of coalescence). At (c), approximately 65% of the material has solidified - the remaining material comprises an extensive, percolating network of liquid and disorder separating grains of differing orientation[26]. The percolation threshold is reached from above, as liquid is consumed by the growing solid. From this point forward the continued growth of grains is no longer accomplished by the speedy conversion of free liquid atoms but rather by the assimilation of smaller grains by larger grains - a far slower process mediated by the network of disorder which spans the simulation cell [27].

We display in Figure 5 (a-c) a time sequence of cross-sectional images obtained from slices of a 16M-atom simulation cell at the points marked (a-c) in Figure 4. In Figure 5 (d-f) we show the same sequence for a 64k-atom simulation. The atoms have been colored according to a parameter that is a measure of the correlation of local symmetry - liquid (blue) atoms have local

symmetry which is poorly correlated, while solid (red) atoms posses a local symmetry which is highly correlated with their neighbors [21]. Yellow atoms identify a distinct population of "intermediate" atoms that occupy the interfaces and grain boundaries. The images for the 64k-atom simulation were created by tiling the original slices (using periodic boundary conditions) so that the spatial extent for each of the images approximately matches that of the 16M-atom images. (The images are $6 \times 6$ tiles of the original; the exact ratio of simulation box lengths is 6.35:1). Homogenous nucleation in the larger sample is seen to occur earlier and across the entire sample, producing a heirarchy of nucleus-nucleus separations and sizes that lead to the rich grain structure seen in Figure 5(c). By contrast, the early grains nucleated in the 64k-atoms simulation grow very rapidly to fill the simulation cell, so that the grains are interacting with their images. Coalescence in this system produces a very artificial final grain structure, dominated by grains which now spans the entire structure.

Figure 6 displays cross-sectional images obtained from slices of our simulation cells at 250 ps. The three smaller samples have been replicated using periodic boundary conditions (PBC) so that their linear extent is approximately the same as the 16M-atom sample. By the end of the coalescence phase, the 64,000-atom and 250k-atom simulation have each developed a periodic grain structure, which appears rather artificial when compared with the structure seen in the 16M-atom sample. A similar investigation of the structure at coalescence for the 2M atom simulation does not reveal system-spanning grains - the structure appears (to the eye, at least) similar to that shown in Figure 5(c), for the 16M atom simulation. A more careful investigation reveals that even in the absence of such a clear "periodic boundary effect," the distribution of grain sizes in this system has also been cut-off at at an artificially small size. We display in Figure 7 the average size of the largest grains as a function of time for sample sizes spanning over 2 orders of magnitude. Although the grains evolve in similar fashion (i.e, explosive growth followed by a slow coarsening after coalescence), the size of the largest grains at the percolation threshold (as the system transitions from fast to slow growth) exhibits a strong size dependence, with the smaller samples unable to attain the larger grain sizes[29]. The dependence of grain size to system size can be described using finite size scaling theory - close to the percolation threshold the total number of atoms (or mass, $M$) in the largest grains should scale as the linear size of the system $L$ with a fractal dimension $d_f$[28]

$$M \propto N^{d_f/d} \approx N^{0.84}\,.$$

We plot the size at percolation (the points marked in Figure 7) against simulation-cell size in Figure 7. Also shown in Figure 7 are the results expected from finite size scaling (solid line). (The power-law scaling is highlighted in the inset.) We find that for samples smaller than 8,192,000 (8M) atoms, the results can be closely described using finite size scaling, so that the finite size of the simulation cell is determining the size of the largest clusters. Simulations utilizing cells larger than 8M atoms would not produce larger grains at the percolation threshold, as evidenced by the behavior of the 16,364,000 (16M) and 32,768,000 (32M) atom simulations. For these simulations, continuous nucleation of solid (in the vanishing liquid spaces) and growth of the existing solid grains ultimately serve to limit the grain size. The growth of the largest grains for the 16M atom cell is seen in Figure 7 to follow almost identically the growth observed in the 8M-atom simulation, while the size of the largest grains at percolation are seen to have significantly departed from the finite size scaling prediction. We performed a similar simulation on a 1M-atom cell using an NPT ensemble by applying a time-varying hydrostatic stress to contrast with the NVT simulations. (This datum is shown as the magenta square in Figure 7.) We saw little difference in the growth behavior while using this compression technique to follow substantially the same thermodynamic path. The size scale for solidification is thus not influenced by the simulation details. We expect that in general, it is the competition between nucleation rate per volume and growth rate that affects the characteristic size at percolation -

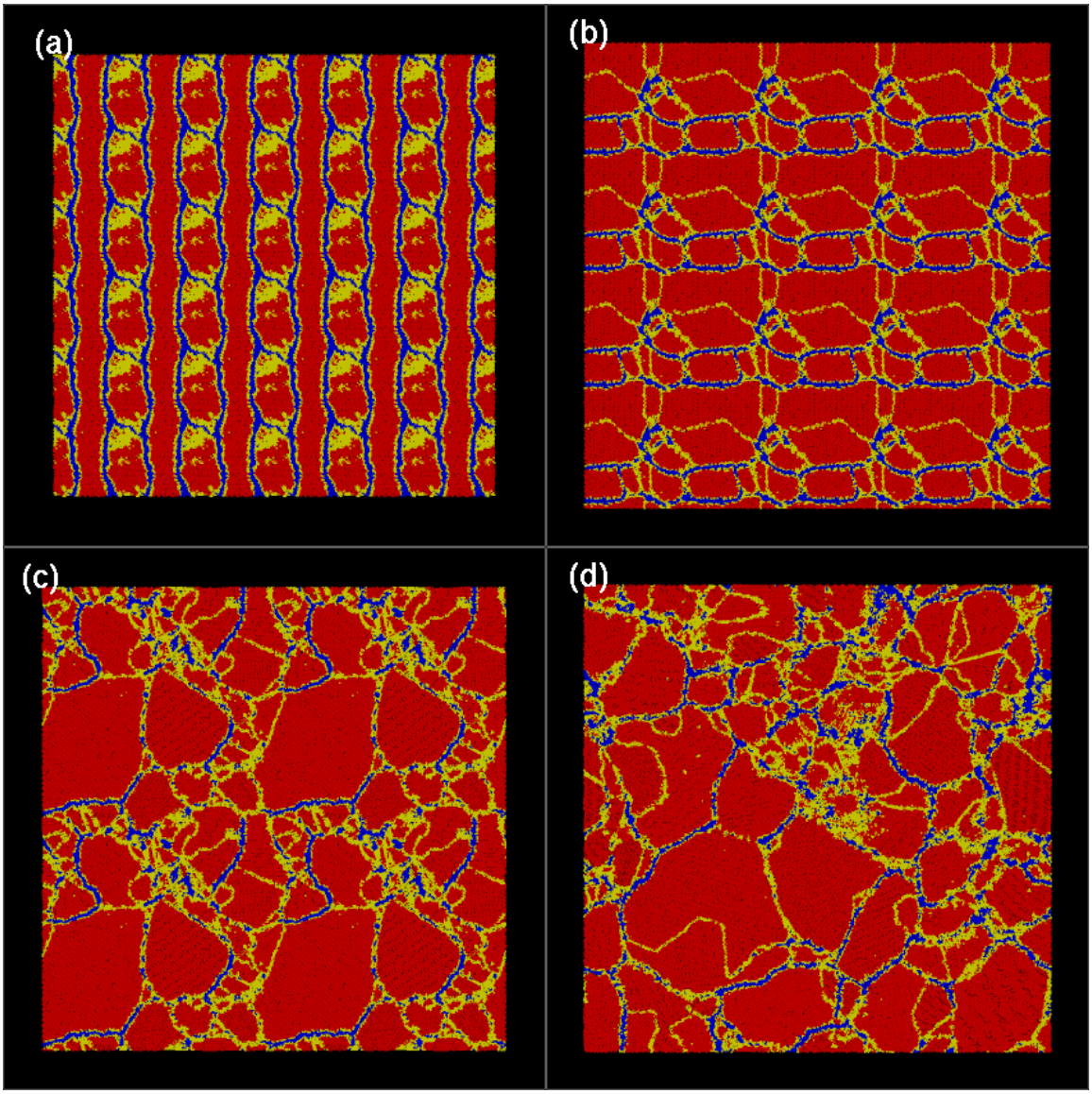


**Figure 6.** Cross sectional images displaying the microstructure obtained in simulations containing (a) 64,000 atoms (b) 256,000 atoms, (c) 2,048,000 atoms and (d) 16,384,000 atoms after the start of the coarsening process. The three smaller sample images have been replicated using periodic boundary conditions to appear approximately the same size as the 16M atom simulation.

any solidification process which leaves these two rates unchanged would result in similar scaling behavior.

We might expect the average size of the largest grain at coalescence to be independent of system size since it is determined entirely by the nucleation rate per volume per time and the growth rate. Neither of those rates vary with system size in a macroscopic system. In a simulated sample that is too small, however, the largest cluster sees an image of itself which is artificially close due to the periodic boundary conditions. The largest cluster in this case will grow until it reaches a certain limiting size that scales with the volume of the box.

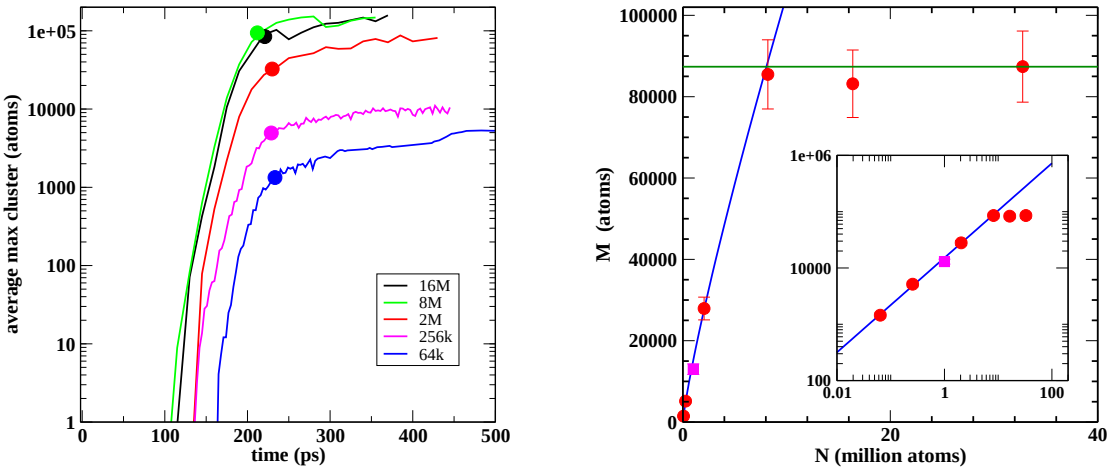


**Figure 7.** (a) The size of the largest cluster in the system, as a function of time, plotted for simulation cells containing 64,000 (green), 250k (blue), 2M (red) and 16M (black) atoms. (b) Maximum cluster size at the start of coarsening as a function of simulation size. The maximum cluster size should scale with the size of the cell if the scale of growth is dictated by the volume of the simulation box, as demonstrated by the 64,000-, 250k- and 2M-atom simulations. The maximum cluster size at the end of coalescence in the 16M simulations, by comparison, is seen no longer to scale with cell volume, suggesting that growth in this simulation occurred with no interference from the cell boundaries.

Figure 7 shows the size of the maximum cluster at the start of the coarsening process as a function of simulation size. We find that the sizes for our three smaller samples scale linearly with system size - an indication that the coarsening process has occurred too early for these systems. The largest grain in the 16M-atom simulation is clearly no longer scaling with cell volume in the same way as the smaller samples. We believe that it is the distribution of nucleation sites during the early stages of the simulation, not the presence of PBC, which limits the grain size in this case.

*ddcMD* will allow us to deepen our understanding of solidification in a wide variety of materials that have to date posed great challenges for atomistic simulation. The results obtained to date suggest that system size is the dominant factor determining not only the growth process but the resulting structure for simulations which are too small.

## 7. Conclusions

The 16M-atom calculation that we performed on tantalum is significant because it provides the first model of metal solidification from the melt made with no approximations due to finite system size. This simulation represents a crucial first step towards our goal of creating a mesoscale model of solidification with input from atomistics, as it demonstrates an ability to directly link accurate atomistic simulations to meso- and higher-scale models – a feat never before achieved for a liquid/solid transition. In addition, our calculations (the largest simulation ever attempted using quantum-based interaction potentials) proves that BlueGene/L's unique architecture can scale as advertised with real-world applications, attaining a performance of 101.7 TFlops. Given the weak scaling of *ddcMD* we could simulate well over a billion atoms, both in order to verify

directly the size independence of our earlier result and to begin an unprecedented atomistic-scale investigation into grain coarsening.

## Appendix: Calculation of $Q_6$ local order parameter

The identification of liquid-like and solid-like atoms is made by using a a slight modification of the $Q_6$ order parameter described by ten Wolde *et al.* which is itself based on the shape spectroscopy parameters introduced by Nelson and Toner.[21, 30] The concept is simple - characterize the angles made by each bond associated a nearest neighbor of atom $i$ using spherical harmonic functions:

$$\vec{q_l}(i) = \langle Y_l^m(\phi_{ij}, \theta_{ij}) \rangle_{N_i}$$

where $\vec{q_l}(i)$ is a $2l+1$-component complex vector whose elements are dependent on the symmetry of the local bonds as well as the choice of coordinate system, and the average is over atoms $j$ that are nearest neighbors of $i$.

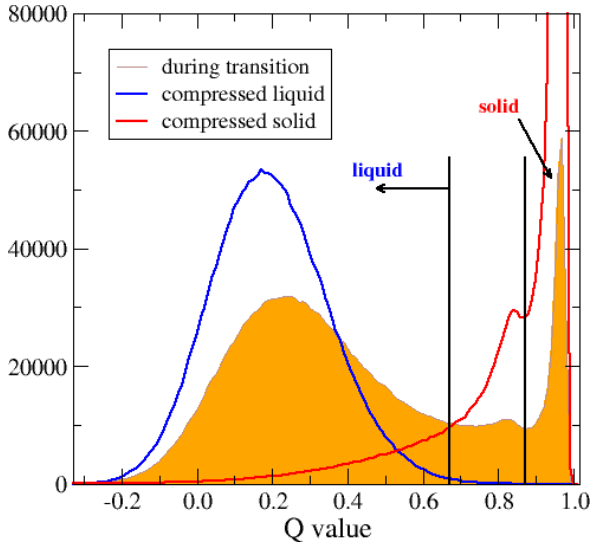


**Figure 8.** Distributions of $Q_6$ measured in a compressed liquid, a compressed liquid during solidification and in a compressed solid at the point of coalescence.

We define a dot product:

$$\vec{q_l}(i) \cdot \vec{q_l}(j) = \sum_{m=-l}^{l} q_{lm}(i) q_{lm}^*(j)$$

which produces a spherically symmetric quantity that is now independent of the choice of coordinates [31].

The $q_6$ parameter has been widely used to characterize incipient order in liquids [31, 32, 33], since it has approximately the same value for most cubic structures. By itself, it is difficult to use as an order parameter due to the broad distribution of values present in an ordering liquid. However, by characterizing the correlation of values between neighboring atoms, we can easily identify those atoms whose bond angles are highly correlated with their neighbors (i.e., are solid-like) and those whose bond angles are poorly correlated with their neighbors (i.e., are liquid-like). We define for each atom $i$ a scalar quantity $Q_6$:

$$Q_6 = \frac{\langle \vec{q_6}(i) \cdot \vec{q_6}(j) \rangle_{N_i}}{\vec{q_6}(i) \cdot \vec{q_6}(i)}$$

which is an average of such correlations among nearest neighbors of $i$, normalized by $|q_6|$.

We show in Figure 7 the distributions of $Q_6$ measured in a compressed liquid, a compressed liquid nucleating crystals, and a sample at the point of coalescence. Making use of the clear separation of values present in the distribution, we arbitrarily assign the label "liquid" or the color blue to atoms with $Q_6 < 0.67$ and "solid" or the color red to atoms with $Q_6 > 0.87$. Yellow or "interface" atoms are those with middle values of $Q_6$ - corresponding to a distinct population, as demonstrated by the small peak in the distribution around the value $Q_6 = 0.8$.

## Acknowledgments

This work was performed under the auspices of the U.S. Department of Energy by University of California, Lawrence Livermore National Laboratory under Contract W-7405-Eng-48.

## References


[1] Alder B J and Wainwright T E 1959 *J. Chem. Phys* **31** 459
[2] Mandell M J, McTague J P and Rahman A 1976 *J. Chem. Phys.* **64** 3699
[3] Honeycutt J D and Andersen H C 1984 *Chem. Phys. Lett.* **108** 535
[4] Swope W C and Andersen H C 1990 *Phys. Rev. B* **41** 7042
[5] O'Malley B and Snook I 2003 *Phys. Rev. Lett.* **90** 085702
[6] Moriarty J A 1994 *Phys. Rev* B **49** 12431
[7] Moriarty J A 1990 *Phys. Rev.* B **42** 1609
[8] Moriarty J A, Belak J F, Rudd R E, Söderlind P, Streitz F H and Yang L H 2002 *J. Phys.:Condens. Matter* **14** 2825
[9] Söderlind P and Moriarty J A 1998 *Phys. Rev.* B **57** 10340
[10] Brunner R K , Phillips J C and Kale L V 2000 *Proceedings of Supercomputing SC2000, Dallas, Texas, Nov. 8-15, 2000* 67
[11] Phillipsgbin J C , Kumary Z S and Kaley L V 2002 *Proceedings of Supercomputing SC2002, Baltimore, Maryland, Nov. 16-22, 2002*
[12] Warren M S, Germann T C, Lomdahl P S, Beazley D M, Salmon J K 1998 *Proceedings of Supercomputing SC98, Orlando, Florida, Nov. 7-13, 1998*
[13] Daw M S and Baskes M I 1983 *Phys. Rev. Lett.* **50** 1285
[14] Foiles S M, Baskes M I and Daw M S, 1986 *Phys. Rev.* B **33** 7983
[15] see, however, Baskes M I 1992 *Phys. Rev.* B **46** 2727 for a modification of the EAM potential which introduces angular forces
[16] Allen M P and Tildesley D J 1987 *Computer Simulation of Liquids* (Clarendon Press, Oxford).
[17] Ermak D L and Buckholz H 1980 *J. Comput. Phys.* **35** 169
[18] Tuckerman M , Martyna G J and Berne B J 1990 *J. Chem. Phys.* **97** 1990
[19] Martyna G J, Tobias D J and Klein M L 1994 *J. Chem. Phys.* **101** 4177
[20] Martyna G J, Tuckerman M, Tobias D J and Klein M L 1996 *Molec. Phys.* **87** 1117
[21] ten Wolde P R, Ruiz-Montero M J and Frenkel D, 1995 *Phys. Rev. Lett.* **75** 2714
[22] Klein W and Layvraz F, 1987 *Phys. Rev. Lett.* **57** 2845
[23] Luo S-N, Strachan A and Swift D C 2004 *J. Chem. Phys.* **120** 11640
[24] Gara A *et al.* 2005 *IBM J. Res. and Devel.* **49** 195
[25] Bachega L *et al.* 2004 *Proceedings of Parallel Arch. and Comp. Tech. (PACT2004)*
[26] The critical percolation probability for random perocolation in three dimensions is about 31%.
[27] Stauffer D 1979 *Phys. Rep.* **54** 1
[28] $d_f = d - \beta/\nu$, where the critical exponents are known to be $\beta = 0.41$ and $\nu = 0.88$ for $d = 3$ dimensions
[29] Both the rapid initial growth rate and the slow (coarsening) growth rate appear to be independent of size.
[30] Nelson D R and Toner J 1981 *Phys. Rev.* B. **24** 363
[31] Steinhardt P J, Nelson D R and Ronchetti M 1983 *Phys. Rev.* B **28** 784
[32] Mitus A C and Patashinskii A Z 1988 *Physica* A **150** 371
[33] Patashinskii A Z, Mitus A C and Ratner M A 1997 *Phys. Rep.* **288** 409