MONTE CARLO METHODS

Fluid Simulation with the Geometric Cluster Monte Carlo Algorithm

Building on earlier approaches, the geometric cluster algorithm permits the efficient simulation of cases in which conventional simulation methods fail, such as mixtures of different-sized particles. Starting from the celebrated Metropolis Monte Carlo algorithm, the author describes the development of cluster methods for magnetic systems and their connection with the geometric cluster algorithm.

The Monte Carlo method is applied in wide areas of science and engineering to determine the properties of a broad variety of systems. In this article, I focus on its use in the simulation of systems described by classical equilibrium statistical mechanics—specifically, in the simulation of model fluids.

Although the term Monte Carlo *simulation* might suggest otherwise, its goal is not to *mimic* a fluid's behavior in a dynamical sense. Similar to a molecular dynamics (MD) simulation—which does have this goal—a Monte Carlo program generates a sequence of configurations of particles. However, whereas in an MD simulation each configuration is created from the previous one by applying Newton's equations of motion, in a Monte Carlo simulation there does not have to be a physical process connecting two subsequent configurations. Thus, unlike in an MD simulation, the resulting sequence cannot necessarily be viewed as a "movie" of the system's behavior.

The introduction of nonphysical dynamics has the important advantage that it permits the study of systems that evolve over otherwise prohibitively large time scales. Examples of such situations include phase transitions (for instance, melting and freezing) and critical systems (in a fluid, *criticality* corresponds to the conditions at which the distinction between a vapor and a liquid disappears). Near a critical point, a system's dynamical evolution becomes extremely sluggish, and most computer simulation methods are entirely incapable of dealing with the slowdown. Robert Swendsen and Jian-Shen Wang first solved this problem for model magnets—which exhibit the same phenomenon—in 1987 via a *cluster Monte Carlo algorithm*.[1] Because of the fantastic speedup their method provided, researchers wanted to find a similar approach for fluids, but unlike model magnets, most fluid models don't employ a lattice structure. The absence of an underlying lattice breaks a symmetry that is a crucial ingredient for Swendsen and Wang's approach.

Building on important advances[2–4] that have occurred since the introduction of the Swendsen–Wang (SW) algorithm, Jiwen Liu, a graduate student at the University of Illinois, and I have recently been able to formulate its extension to fluids defined in continuum space.[5–7] Here, I describe the resulting highly general algorithm, its connection to existing methods, and its advantages over conventional Monte Carlo algorithms.

1521-9615/06/$20.00 © 2006 IEEE
Copublished by the IEEE CS and the AIP

Erik Luijten
University of Illinois at Urbana-Champaign

Conventional Monte Carlo Algorithms

To understand the conventional Monte Carlo method, we need a fundamental result of statistical mechanics—namely, that a system's thermodynamic properties can be computed by summing over all the possible configurations or *states* of the system. Concretely, we can compute the average value of a property A as

$$\langle A \rangle = \frac{1}{Z} \sum_{\{s\}} A_s \exp(-\beta E_s), \tag{1}$$

where A_s is the value of the observable A if the system resides in state s, and E_s denotes the energy of the corresponding state. Z is a normalization constant (the *partition function*) and $\beta = 1/(k_B T)$, with k_B representing Boltzmann's constant and T the temperature. Equation 1 represents a weighted average, in which each state contributes with a weight factor $\exp(-\beta E_s)/Z$.

The number of configurations a system can have is almost always extremely large, so in numerical calculations the sum in Equation 1 is taken only over a subset of configurations. It might seem straightforward to just generate a set of randomly chosen states (also called *samples*) and compute $A_s \exp(-\beta E_s)$ for each of these states, so that the sum of these individual evaluations provides an estimate of the actual sum. Unfortunately, this approach—called *simple sampling*—doesn't work well because random configurations typically have a large energy, hence the Boltzmann factor $\exp(-\beta E_s)$ is vanishingly small for most samples. A much better solution is to employ *importance sampling*, in which we preferentially sample states that strongly contribute to the integral in Equation 1. We obtain the most accurate estimate if we sample the states with a probability distribution $\exp(-\beta E_s)/Z$.

However, even though we can compute the *relative* probability with which two specific states should occur in a set of samples, we can't compute their *absolute* probability because we don't know the normalization constant Z. Fortunately, Nicholas Metropolis and his colleagues[8] found a way to calculate the expectation value in Equation 1 *without* evaluating Z. The basic idea is to create a sequence of states in which each state only depends on the state immediately preceding it (a so-called *Markov chain*). Starting from a configuration s_i that has a Boltzmann factor p_i, we create a new trial configuration s_j, which has a Boltzmann factor p_j. The trial configuration is either *accepted* or *rejected*: if accepted, it's the next member in our sequence of states, and if rejected, then the next member of the sequence is again s_i. This process is repeated iteratively to generate a sequence of configurations; acceptance or rejection is governed by a *transition probability* from each possible state s_i to each state s_j.

In general, finding the transition probabilities that lead to the proper set of configurations can be very complicated, but we can simplify the situation by imposing a restriction called the *condition of microscopic reversibility* or *detailed balance*. This restriction states that, on average, the number of transitions from a state i to a state j is balanced by the number of transitions from state j to state i. Each transition probability is the product of two factors—namely, the *a priori probability* α_{ij} of generating a trial configuration s_j from a configuration s_i, and the *acceptance probability* P_{ij} of accepting the trial configuration as the new state. We can thus write the detailed-balance condition as

$$p_i \alpha_{ij} P_{ij} = p_j \alpha_{ji} P_{ji}. \tag{2}$$

In the simplest choice, the a priori probability is symmetric—that is, $\alpha_{ij} = \alpha_{ji}$—so we have

$$p_i P_{ij} = p_j P_{ji}, \tag{3}$$

which we can rewrite as

$$\frac{P_{ij}}{P_{ji}} = \exp[-\beta(E_j - E_i)], \tag{4}$$

where we have used that p_i and p_j are given by the Boltzmann distribution. Note that the partition function Z has canceled out. This equation does not uniquely define the desired acceptance probability P_{ij}. Metropolis and his colleagues[8] proposed the solution

$$P_{ij} = \begin{cases} \exp[-\beta(E_j - E_i)] & \text{if } E_j > E_i \\ 1 & \text{if } E_j \le E_i \end{cases}, \tag{5}$$

which is sometimes summarized as

$$P_{ij} = \min[\exp(-\beta \Delta_{ij}), 1], \tag{6}$$

with $\Delta_{ij} = E_j - E_i$.

A valid Monte Carlo scheme not only has to obey detailed balance, but it must also be *ergodic*, meaning that a path exists in phase space from every state to every other state via a succession of trial moves. Clearly, if the trial states are chosen in such a manner that certain states can never be reached, the estimator for a thermodynamic observable can differ severely from the correct expectation value.

The trial configuration s_j is generated via a *trial*

Ising Model

Describing the algorithms that underlie the geometric cluster algorithm requires us to first look at the *Ising model*. It's defined on a d-dimensional lattice of linear size L (a square lattice in $d = 2$ and a simple cubic lattice in $d = 3$), with, on each vertex of the lattice, a one-component spin of fixed magnitude that can point up or down. The Hamiltonian

$$\mathcal{H}_{\text{Ising}} = -J\sum_{\langle ij\rangle} s_i s_j$$

describes the system. The spins s take values ±1, and the sum runs over all pairs of nearest neighbors (which are coupled via a ferromagnetic coupling with strength $J > 0$). This is a model for a magnetic material: each spin represents a magnetic domain. At high temperatures, entropy will cause the spins to arbitrarily point up or down. However, below the Curie temperature, the domains line up, and the system becomes magnetized.

move. If we consider an assembly of N particles, for example, a trial move could consist of a small displacement of one particle in a random direction. (Note that a particular trial move's probability is embodied in the factor α_{ij}, which in this example is equal to α_{ji}, as explained later.) The Metropolis criterion (Equation 6) then asserts that the new state is always accepted if the resulting configuration has a lower energy than the original one. If the trial configuration has a higher energy, it's only accepted with a probability equal to the ratio of the Boltzmann factors of the new and the old configuration. In practice, this criterion is implemented by generating a random number $0 \leq r < 1$ and accepting the trial configuration only if $r < P_{ij}$. The use of random numbers to generate trial configurations and to decide on their acceptance or rejection is the origin of the phrase "Monte Carlo simulation" and also explains why these simulations crucially depend on the availability of high-quality random-number generators.

Once we're able to generate a sequence $\{s_1, \ldots, s_M\}$ of configurations, we can calculate the expectation value of a thermodynamic property A by sampling the property A_n for each configuration s_n. The thermodynamic average in Equation 1 is then estimated as a *simple average*,

$$\langle A\rangle \approx \frac{1}{M}\sum_{n=1}^{M} A_n . \qquad (7)$$

The moves used for generating a trial configuration depend on the nature of the system. A simple fluid model is the previously mentioned example of an assembly of N spherical particles, confined to a certain volume. In an elementary move, one randomly selected particle with position $\mathbf{r}$ is displaced to a new position $\mathbf{r}' = \mathbf{r} + \delta\mathbf{r}$. We can choose the displacement $\delta\mathbf{r}$ as a randomly oriented vector on a sphere with radius $0 < |\delta\mathbf{r}| < \ell$, or we can choose the new position $\mathbf{r}'$ within a cube of linear dimension ℓ, centered around the original position $\mathbf{r}$. In either case, the a priori probability α_{ij} is symmetric—that is, the probability to generate the trial configuration from the original configuration is identical to the probability of the reverse process. The parameter ℓ permits control over the simulation's efficiency. Monte Carlo algorithms with trial moves that involve small displacements of individual particles are also called *local-update algorithms*.

The statistical quality of the estimate in Equation 7 depends on the number of *independent* samples in the sequence of configurations, thus a simulation's objective is to maximize the rate at which independent configurations are generated. If a trial configuration is generated via a small change to the previous configuration (that is, the parameter ℓ governing the particle displacement is small), the energy difference Δ_{ij} will typically also be small and the acceptance ratio large. However, many steps will be required before we have an independent configuration. Conversely, if a trial configuration is generated via a big change to the previous configuration, the sequence of configurations would decorrelate quickly were it not that the typical energy difference will be large and the acceptance probability will thus be very small.

In addition to variation of the magnitude of a single-particle displacement, we can also try to increase the rate at which configurations evolve by moving several particles at a time. However, if the moves of these particles are independent, this turns out to be less efficient than a sequence of single-particle moves.[9]

Lattice Cluster Algorithms

Many Monte Carlo methods can be described most easily for *lattice models*. One example of such a system is the Ising model (see the "Ising Model" sidebar). This system can be simulated by the Metropolis algorithm; trial moves consist of the inversion of individual spins and are accepted or rejected on the basis of the change in coupling energy. The SW algorithm[1] for the Ising spin model represents a radical departure from this *single-spin flip* approach. It permits the inversion of entire *clusters* of spins without suffering from the decrease in acceptance probability normally observed in multiple-particle moves. Before discussing its properties, let's review the basic steps of the SW algorithm as applied to a given configuration of spins (see Figure 1 for an illustration):

1. A bond is formed between every pair of *parallel* nearest neighbors with a probability $p_{ij} = 1 - \exp(-2\beta J)$, where J is the coupling constant and β the inverse temperature defined earlier.
2. We identify clusters by determining which spins are connected, directly or indirectly, via bonds. In this way, the system is divided into clusters of parallel spins (the *cluster decomposition*); the bond probability (and hence the typical cluster size) grows with increasing coupling strength βJ (decreasing temperature). For nonzero temperatures, each cluster is generally a subset of all spins of a given sign: two adjacent spins need not belong to the same cluster, even if they have the same sign.
3. For each cluster of spins, we choose a spin value ±1, which is then assigned to all spins that belong to the cluster. Thus, all spins in each cluster are flipped collectively with a probability 1/2.
4. We erase the bonds—which were introduced only to identify the clusters—and the "cluster move" is complete; a new spin configuration has been created. The algorithm restarts at step 1.

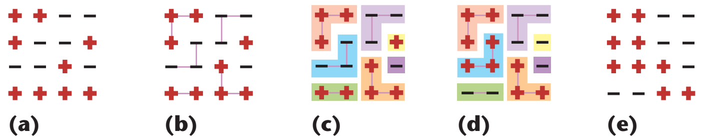

Figure 1. Illustration of the Swendsen-Wang algorithm as described in the text. (a) Original spin configuration; the "+" signs indicate spins that point upward (out of the plane) and "–" signs indicate spins that point downward (into the plane). (b) Bonds (purple lines) are formed between spins of the same sign, with a probability that depends on the coupling strength. (c) All spins that are connected by bonds belong to a single cluster (each color of shading indicates a separate cluster). (d) Each cluster of spins is flipped with a probability of 50 percent. In this example, only the blue, green, and yellow clusters are flipped.(e) All bonds are erased and a new spin configuration is obtained.

This prescription relies on a mathematical relationship discovered in the early 1970s by Dutch mathematical physicists Cees Fortuin and Piet Kasteleyn[10] that connects the Ising model and the random-cluster model. A particularly remarkable aspect of the cluster algorithm is that it is *rejection free*: once the clusters have been formed, each one can be flipped independently, *without* imposing an acceptance criterion involving the energy change induced by such a collective spin-reversal operation. However, the absence of an acceptance criterion does not imply that a cluster flip won't entail an energy difference! Indeed, nothing in the algorithm guarantees this property. Cluster flips result in a sequence of configurations with different energies, but in such a way that they already appear exactly according to the Boltzmann distribution.

Soon after the SW algorithm appeared, Ulli Wolff[2] introduced a single-cluster variant of it. The SW algorithm creates both small and large clusters, and whereas the rapid evolution of the spin configurations is mostly due to the larger clusters, considerable effort is spent constructing the smaller ones. Wolff realized that this problem can be eliminated by avoiding the decomposition of the spin configuration into clusters altogether—in his algorithm, only a single cluster is formed, which is then always flipped. If this cluster turns out to be large, correlations are destroyed as effectively as via the large clusters in the SW algorithm, but without the effort of creating the smaller clusters that make up the rest of the system. If the Wolff cluster is small, not much is gained, but then again, not much computational effort is wasted. This algorithm is even more efficient than the original SW version and—a nice bonus—even easier to implement. Let's see how it compares with the SW algorithm:

1. In the starting configuration, we select a spin i at random.
2. We add all nearest neighbors j of this spin to the cluster with a probability $p_{ij} = 1 - \exp(-2\beta J)$, provided spins i and j are parallel and the bond between i and j hasn't been considered before.
3. Each spin j added to the cluster is also placed on the stack. Once all neighbors of i have been considered for inclusion in the cluster, we retrieve a spin from the stack and consider all its neighbors in turn for inclusion in the cluster, following step 2.
4. Steps 2 and 3 repeat iteratively until the stack is empty.
5. Once the cluster is completed, we invert all spins that belong to the cluster.

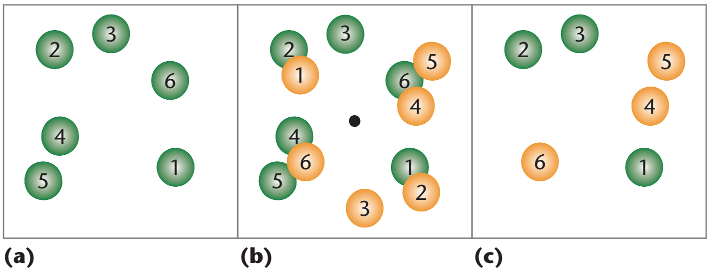

Figure 2. Geometric cluster algorithm for hard disks.[3] From (a) the original configuration, (b) we can create a new configuration (orange circles) via a point reflection of all particles with respect to a randomly chosen pivot point (small black circle). The superposition of the original and new configuration leads to groups of overlapping particles. This example has three *pairs* (or clusters) of groups ({1, 2}, {3}, and {4, 5, 6}). The particles in any one of these clusters can be point-reflected with respect to the pivot without affecting the other two. In (c) the final configuration results, if we start from the original configuration, only the particles in the third cluster {4, 5, 6} are point-reflected. (Reprinted with permission from "Generalized Geometric Cluster Algorithm for Fluid Simulation."[6] Copyright 2005, American Institute of Physics.)

Again, this is a rejection-free algorithm in the sense that the cluster is always flipped. Just as in the Metropolis algorithm, the cluster-construction process uses random numbers, but the probabilities p_{ij} involve energies of individual spin pairs, in contrast with an acceptance criterion that depends on the *total* energy change induced by inverting an entire group of spins. We can simplify this implementation even further: in step 2, we can immediately invert each spin j added to the cluster, thus guaranteeing that a spin is never added twice and eliminating step 5.

Cluster Algorithms for Continuum Systems

Are there also cluster Monte Carlo algorithms for continuum systems? We will first look at the special case of a hard-sphere liquid and then show how we can generalize it to arbitrary fluids. To demonstrate the efficiency of this new method, we apply it to a suspension of colloids and nanoparticles.

Geometric Cluster Algorithm for Hard-Sphere Mixtures

The advantages of lattice cluster algorithms—in particular, the suppression of critical slowdowns—made it a widely pursued goal to generalize the SW and Wolff algorithms to fluid systems in which particles aren't confined to lattice sites, but can take arbitrary positions in continuum space. Unfortunately, the absence of a lattice structure breaks a fundamental symmetry. We can interpret an Ising model as a *lattice gas*, in which a spin +1 corresponds to a particle, and a spin –1 corresponds to an empty site. Accordingly, a spin-inversion operation corresponds to a particle being inserted into or removed from the system. This *particle-hole symmetry* is absent in continuum systems: we can delete a particle in a fluid configuration straightforwardly, but there is no unambiguous way to transform empty space into a particle.

In 1995, Christophe Dress and Werner Krauth[3] proposed a method to efficiently generate particle configurations for a hard-sphere liquid. In this system, particles are represented by impenetrable spheres (or disks, in the two-dimensional variant) that don't interact unless they overlap. Because of this *hard-core repulsion*, a Monte Carlo algorithm involving local moves is relatively inefficient, since any move that generates a particle overlap is rejected. Figure 2 shows the *geometric cluster algorithm* (GCA),[3] which is designed to avoid such overlaps while generating a new configuration by proceeding as follows:

1. In a given configuration C of particles, we choose a "pivot" at random.
2. We generate a new configuration $\tilde{C}$ by performing a point reflection with respect to the pivot for all particles in C.
3. We superimpose the configuration C and its transformed counterpart $\tilde{C}$, which leads to groups of overlapping particles. The groups generally come in pairs, except possibly for a single group that's symmetric with respect to the pivot; each pair is denoted a "cluster."
4. For each cluster, C and $\tilde{C}$ can exchange particles without affecting particles in other clusters. This exchange is performed for each cluster independently with a probability 1/2, thus if the superposition of C and $\tilde{C}$ decomposes into N clusters, we have 2^N possible new configurations. The configurations that are actually realized are denoted C' and $\tilde{C}'$, that is, the original configuration C is transformed into C'; its point-reflected counterpart $\tilde{C}$ is transformed into $\tilde{C}'$.
5. We discard the configuration $\tilde{C}'$—which is essentially just a mirror copy of C' used for constructing the clusters—and C' is the new configuration that serves as the starting point for the algorithm's next iteration. (Note that a new pivot location is chosen in every iteration.)

Periodic boundary conditions must be employed so that an arbitrary placement of the pivot is possible. However, other self-inverse operations are permissible, such as a reflection in a plane.[11] In this case, we must choose various orientations of the plane to ensure that the algorithm is ergodic.

If we compare the GCA to lattice cluster algorithms, we notice that the SW and Wolff algorithms operate in the grand-canonical ensemble (phrased in the lattice-gas interpretation, the total number of particles fluctuates during the simulation), whereas the GCA operates in the canonical ensemble (meaning the number of particles is constant). Nevertheless, there is a remarkable resemblance between the GCA and the SW algorithm: we can decompose the original configuration into clusters by exploiting a symmetry operation that leaves the Hamiltonian invariant if applied to the entire configuration. In the SW algorithm, this is the spin-inversion operation; in the GCA, it is a geometric symmetry operation. Subsequently, we can create a new configuration by moving each cluster independently with a certain probability.

This approach is very general—for example, it isn't restricted to monodisperse systems, and Krauth and his colleagues have applied it successfully to binary[12] and polydisperse[13] mixtures. Indeed, conventional simulations of size-asymmetric mixtures typically suffer from jamming problems, in which a very large fraction of all trial moves is rejected because of particle overlaps. In the GCA, particles move in a nonlocal fashion, yet overlaps are avoided.

The GCA's most important limitation is the fact that the average cluster size increases very rapidly if the total density approaches the percolation threshold of the combined system containing the superposition of the configurations C and $\tilde{C}$. For high densities, each cluster spans the entire system, so the system no longer evolves, and the algorithm is no longer ergodic.

To emphasize the analogy with lattice cluster algorithms, we can formulate a single-cluster (Wolff) variant of the GCA:[5,11]

1. In a given configuration C, choose a "pivot" at random.
2. Select a particle i as the first particle that belongs to the cluster. This particle moves via a point reflection with respect to the pivot. In its new position, we refer to the particle as i'.
3. Repeat the point reflection in step 2 iteratively for each particle j that overlaps with i'. Thus, if the (moved) particle j' overlaps with another particle k, particle k moves as well. Note that all translations involve the same pivot.
4. Once all overlaps have been resolved, the cluster move is complete.

Just as in the Wolff algorithm, only a single cluster of particles is constructed and moved, eliminating the overhead involved in creating a point-reflected copy of the entire system and then determining groups of overlapping particles.

Generalized GCA for Interacting Particles

A clear difference between the GCA and other MC algorithms is the absence of an acceptance criterion (as in the Metropolis algorithm) or energy-dependent cluster addition probability (as in the SW and Wolff algorithms). This is because the GCA is formulated only for particles that interact via hard-core repulsions. To apply it to systems with other types of pair potentials, Dress and Krauth[3] suggested accepting or rejecting entire cluster moves, based on the energy difference involved in such moves. For pair potentials that can be split into a hard-core contribution and an attractive or repulsive tail, this is indeed possible. The cluster-construction procedure takes care of the hard core by guaranteeing that no overlaps are generated, and the acceptance criterion accounts for the tail of the interactions.

For potentials without a hard core, such as a Lennard-Jones interaction, this solution is already less elegant. We're forced to choose an arbitrary "effective hard core," which is used in the cluster construction. The diameter of this excluded volume must be smaller than any particle separation that would typically occur because the algorithm won't generate configurations in which a pair of particles is separated by less than this distance. More important, clusters are now constructed on the basis of only *part* of the pairwise interactions, and the evaluation of the remainder of the energy change resulting from a cluster move is deferred until the acceptance step. The consequences for the computational efficiency can be very severe, because the costly rejection of a cluster move (which involves both its construction and the evaluation of the resulting energy change) becomes quite likely.

However, an entirely different approach is possible by carrying the analogy to lattice cluster algorithms further. First, we phrase the probability p_{ij} of adding a spin j (adjacent to a spin i) to a cluster in the SW algorithm in terms of the corresponding difference in the pair energy of i and j. The relative energy $\Delta_{ij}^{\mathrm{SW}}$ between a spin i that be-

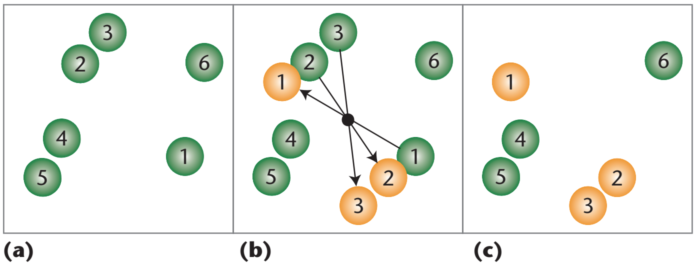

Figure 3. Two-dimensional illustration of the interacting geometric cluster algorithm (GCA). As in Figure 2, green and orange disks denote the particles before and after the geometric operation, respectively, and the small black circle denotes the pivot. However, in the *generalized* GCA, a single cluster is constructed, to which particles are added with an *interaction-dependent* probability. From (a) the original configuration, (b) a cluster is constructed as follows: particle 1 is point-reflected with respect to the pivot; if, in its new position, it has a repulsive interaction with particle 2, the latter has a certain probability to be point-reflected as well, with respect to the same pivot. Assuming an attractive interaction between particles 2 and 3, particle 3 is translated as well, but only with a certain probability. If particles 4 through 6 are not affected by these point reflections, the cluster construction terminates. (c) The new configuration consists of particles 1 through 3 in their new positions and particles 4 through 6 in the original positions. We choose a new pivot, and the procedure repeats. (Reprinted with permission from "Generalized Geometric Cluster Algorithm for Fluid Simulation."[6] Copyright 2005, American Institute of Physics.)

longs to the cluster and a spin j that does not yet belong to the cluster changes if j is *not* added to the cluster. We can discern two different situations. If i and j are initially *antiparallel*, j will never be added to the cluster and only spin i will be inverted, yielding an energy change $\Delta_{ij}^{\mathrm{SW}} = -2J < 0$ that occurs with probability unity. If i and j are initially *parallel*, and j isn't added to the cluster, the resulting change in the pair energy equals $\Delta_{ij}^{\mathrm{SW}} = +2J > 0$. This occurs with a probability $\exp(-2\beta J) < 1$. We can summarize these two scenarios as

$$1 - p_{ij} = \min[\exp(-\beta\Delta_{ij}^{\mathrm{SW}}), 1]\,, \qquad (8)$$

so that we can write the cluster addition probability as $p_{ij} = \max[1 - \exp(-\beta\Delta_{ij}^{\mathrm{SW}}), 0]$. The second step is to phrase the GCA in precisely the same manner, even though it's formulated in continuum space rather than on a lattice. Indeed, the GCA corresponds to a special situation in which either $\Delta_{ij} = 0$ (once particle i has been reflected, it does not overlap with particle j), leading to $p_{ij} = 0$, or $\Delta_{ij} = \infty$ (after point reflection, particle i overlaps with particle j), leading to $p_{ij} = 1$.

Once this connection is made, the generalization of the GCA to arbitrary pair potentials follows in a natural way.[5] All interactions are treated in a unified manner, so there isn't a technical distinction between attractive and repulsive interactions or between hard- and soft-core potentials. To describe a single-cluster variant of the *generalized* GCA, analogous to the Wolff algorithm, let's assume a general pair potential $V_{ij}(\mathbf{r}_{ij})$ that doesn't have to be identical for all pairs i, j (see Figure 3). A single cluster step then proceeds as follows:

1. In a given configuration C, choose a "pivot" at random.
2. Select a particle i at position $\mathbf{r}_i$ as the first particle that belongs to the cluster. This particle moves via a point reflection with respect to the pivot; in its new position, the particle is referred to as i', at position $\mathbf{r}_i'$.
3. Now consider each particle j that interacts with i or i' for addition to the cluster, treating a particle j that interacts with i both in its old and new positions just once. Unlike the first particle, particle j is point-reflected with respect to the pivot only with a probability $p_{ij} = \max[1 - \exp(-\beta\Delta ij), 0]$, where $\Delta_{ij} = V(|\mathbf{r}_i' - \mathbf{r}_j|) - V(|\mathbf{r}_i - \mathbf{r}_j|)$.
4. Each particle j added to the cluster (that is, moved) is also placed on the stack. Once we have considered all particles interacting with i or i', a particle is retrieved from the stack and all of its neighbors that are not yet part of the cluster are considered in turn for inclusion in the cluster as well, following step 3.
5. Steps 3 and 4 repeat iteratively until the stack is empty. The cluster move is now complete.

The probability p_{ij} depends *only* on the change in *pair energy* between i and j that occurs if particle i is point-reflected with respect to the pivot, but particle j is not. This happens with a probability $1 - p_{ij} = \min[\exp(-\beta\Delta_{ij}), 1]$, just as for the SW algorithm in Equation 8. If a particle interacts with multiple other particles that have been added to the cluster, it can thus be considered multiple times for inclusion. The superficial similarity of the expression for the probability to a Metropolis-type acceptance criterion is deceptive, since Δ_{ij} does not represent the *total* energy change resulting from the translation of particle i. Instead, we take other energy changes into account via the algorithm's iterative nature. For the special case of hard-core repulsions, this prescription reduces to the single-cluster variant of the GCA.

It's possible to apply the GCA to lattice models, as Jouke Heringa and Henk Blöte first did for the Ising model;[4] these researchers also devised a way to take into account the nearest-neighbor interactions between spins during cluster construction. Although we can simulate this model with existing algorithms, the approach of Heringa and Blöte permits simulation in the constant-magnetization ensemble. Their procedure is very similar to the generalized GCA, although the presence of a lattice makes it possible to satisfy excluded-volume interactions automatically by exchanging pairs of spins.

The generalized GCA is ergodic, and detailed balance has been proven analytically.[5,6] Conceptually, the most striking point is that this algorithm is rejection free: every cluster constructed is moved (reflected), a fact that might be obscured because in our prescription the cluster is moved during construction, similar to the Wolff algorithm. The central point, however, is that the construction solely involves single-particle energies, whereas a Metropolis-type approach evaluates the total energy change induced by a multiparticle move (and then frequently rejects this move). By contrast, the GCA avoids large energy differences by incorporating "offending" particles into the cluster with a high probability. Rejection-free algorithms were long believed to be rare exceptions, but the generalized GCA shows that we can, in fact, phrase them for large classes of fluids.

Lastly, although the single-cluster approach is more efficient and easier to implement, it's also possible to formulate a multiple-cluster version of the generalized GCA,[6] which serves to illustrate that the generalized GCA is a true off-lattice version of the cluster algorithms described earlier.

The Generalized GCA's Efficiency

Probably the most important feature of the generalized GCA for practical applications is the efficiency with which it generates uncorrelated configurations for size-asymmetric mixtures. This performance directly derives from the nonlocal character of the point reflection employed. In general, the translation of a single particle over large distances has a very low acceptance ratio in conventional Monte Carlo simulations, except in extremely dilute conditions. The situation only deteriorates for multiple-particle moves, unless we select the particles involved in the move in a very specific manner. The generalized GCA makes nonlocal collective moves possible, without any negative consequences for the acceptance ratio. The resulting efficiency gain is illustrated by means of an example taken from elsewhere[5]—namely, a simple binary mixture containing 150 large particles of size σ_{22} (at fixed volume fraction $\phi_2 = 0.1$); N_1 small particles are present as well, also at a fixed volume fraction $\phi_1 = 0.1$. The efficiency is determined through the autocorrelation time (defined below), as a function of size asymmetry. As the size σ_{11} of these small particles varies from $\sigma_{22}/2$ to $\sigma_{22}/15$ (that is, the size ratio $\alpha = \sigma_{22}/\sigma_{11}$ increases from 2 to 15), their number increases from $N_1 = 1{,}200$ to 506,250.

Pairs of small particles and pairs involving a large and a small particle act like hard spheres. If no additional interactions are introduced, the large particles will aggregate because of an entropic phenomenon known as the depletion interaction. Therefore, we also introduce an additional short-range repulsion between the large particles,

$$\beta U_{ll}(r) = \begin{cases} +\infty & r \le \sigma_{22} \\ 3.0\,\dfrac{\sigma_{22}}{r}\,\varepsilon \exp\left[-\dfrac{(r-\sigma_{22})}{\sigma_{11}}\right] & \sigma_{22} < r < 3\sigma_{22} \end{cases} \cdot \tag{9}$$

Because of this additional interaction, the system's energy, indicated by $E(t)$ (the "time" t is counted simply by the number of clusters constructed), fluctuates during the simulation and makes it possible to determine the rate at which the large particles decorrelate. Indeed, the *autocorrelation function* for the energy, defined as

$$C(t) = \frac{\langle E(0)E(t)\rangle - \langle E(0)\rangle^2}{\langle E(0)^2\rangle - \langle E(0)\rangle^2}, \tag{10}$$

decays as $\exp(-t/\tau)$ and the *autocorrelation time* τ indicates how rapidly the system evolves: a large autocorrelation time means a slow evolution.

Figure 4 compares τ for a conventional (Metropolis) MC algorithm and the generalized GCA. To avoid arbitrariness resulting from the computational cost involved with a single sweep or a cluster's construction, we assume that both methodologies have been programmed efficiently and express τ in actual CPU time. Furthermore, τ is normalized by the total number of particles in the system, to account for the variation in N_1 as the size ratio α increases. The autocorrelation time for the conventional MC calculations, τ_{MC}, rapidly increases with increasing α because the small particles tend to trap the large ones. Indeed, already for $\alpha > 7$ it isn't feasible to obtain an accurate estimate for τ_{MC}. By contrast, τ_{GCA} exhibits a very different dependence on α. At $\alpha = 2$, both algorithms require virtually identical simulation time, which establishes that the GCA doesn't involve considerable overhead compared to standard algorithms (if any,

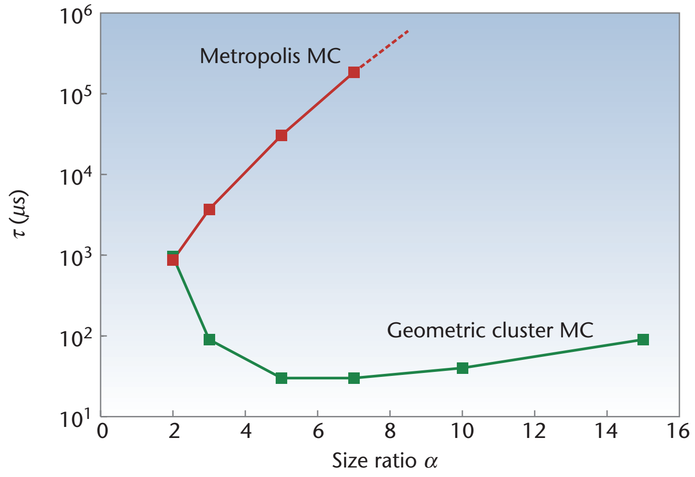

Figure 4. Efficiency comparison between a conventional local update algorithm (red squares) and the generalized geometric cluster algorithm (GCA; blue squares), for a binary mixture with size ratio α. Whereas the autocorrelation time per particle (expressed in μs of CPU time per particle move) rapidly increases with size ratio, the GCA features only a weak dependence on α. (Reprinted with permission from "Rejection-Free Geometric Cluster Algorithm for Complex Fluids."[5] Copyright 2004, American Institute of Physics.)

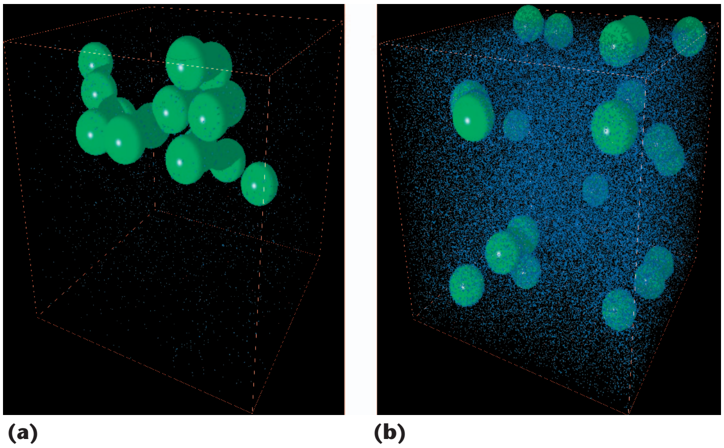

Figure 5. Illustration of colloidal stabilization by means of charged nanoparticles. (a) Twenty colloids (diameter 1μm, packing fraction 0.01) aggregate through their van der Waals attraction. The presence of several thousands of nanoparticles (diameter 25 nm) does not affect this aggregation. (b) The number of nanoparticles has increased to 128,000, but they still occupy only a tiny volume fraction of the system and are weakly attracted by the colloids, preventing their aggregation.[15]

it's mitigated by the fact that all moves are accepted). Upon increasing α, τ_{GCA} initially *decreases* until it starts to increase weakly. The nonmonotonic variation of τ_{GCA} results from the changing ratio N_2/N_1, which causes the cluster composition to vary with α. The main points to note are, first, that the GCA greatly suppresses the autocorrelation time, $\tau_{GCA} << \tau_{MC}$ for $\alpha > 2$, with an efficiency increase that amounts to more than three orders of magnitude already for $\alpha = 7$, and second, that the increase of the autocorrelation time with α is much slower for the GCA than for a local-move MC algorithm, making the GCA increasingly advantageous with increasing size asymmetry.

Sample Application: Nanoparticle Haloing

The field of colloidal stabilization offers an interesting sample application of the generalized GCA. Suspensions of colloidal particles, which have a diameter in the nanometer to micrometer range, find widespread application as precursors for a variety of materials, including advanced coatings and drug carriers. An important goal is controlling the interactions between the colloidal particles, either to keep them suspended in solution or to induce their aggregation.

In recent years, a new technique has emerged[14] in which an effective repulsive interaction between colloids arises from the presence of highly charged nanoparticles. A simple model to describe this system must contain both the colloids and the nanoparticles, which can differ by up to a factor 100 in diameter. This is a prototypical example of a size-asymmetric mixture that is impossible to attack via conventional Monte Carlo or MD simulations. However, the generalized GCA is exquisitely suited to address this problem. Recently, we've been able to computationally reproduce the observed stabilization in a model system and explain its origin.[15] As Figure 5 shows, the addition of a small amount of nanoparticles prevents aggregation of the colloidal particles. An important technological advantage offered by these model calculations is the ability to easily vary system parameters, such as nanoparticle size and charge, and to predict the resulting changes in the suspension's stability.

The generalized GCA is the first general rejection-free cluster algorithm for off-lattice systems. Its most significant property is the fact that it can greatly accelerate the simulation of fluids in which the constituents exhibit a large asymmetry.

In the near future, look for a variety of extensions to this algorithm. The incorporation of electrostatic interactions and the treatment of phase transitions are two areas in which work is currently in progress.

Acknowledgments

The work described in this article was performed with Jiwen Liu and supported by the US National Science Foundation under CAREER award number DMR-0346914 and grant number CTS-0120978, as well as by the US Department of Energy, Division of Materials Sciences, under award number DEFG02-91ER45439, through the Frederick Seitz Materials Research Laboratory at the University of Illinois at Urbana-Champaign.

References

1. R.H. Swendsen and J.-S. Wang, "Nonuniversal Critical Dynamics in Monte Carlo Simulations," *Physical Rev. Letters*, vol. 58, no. 2, 1987, pp. 86–88.
2. U. Wolff, "Collective Monte Carlo Updating for Spin Systems," *Physical Rev. Letters*, vol. 62, no. 4, 1989, pp. 361–364.
3. C. Dress and W. Krauth, "Cluster Algorithm for Hard Spheres and Related Systems," *J. Physics A*, vol. 28, no. 23, 1995, pp. L597–L601.
4. J.R. Heringa and H.W.J. Blöte, "Geometric Cluster Monte Carlo Simulation," *Physical Rev. E*, vol. 57, no. 5, 1998, pp. 4976–4978.
5. J. Liu and E. Luijten, "Rejection-Free Geometric Cluster Algorithm for Complex Fluids," *Physical Rev. Letters*, vol. 92, no. 3, 2004, article no. 035504.
6. J. Liu and E. Luijten, "Generalized Geometric Cluster Algorithm for Fluid Simulation," *Physical Rev. E*, vol. 71, no. 6, 2005, article no. 066701.
7. C. Day, "New Algorithm Speeds up Computer Simulations of Complex Fluids," *Physics Today*, vol. 57, no. 3, 2004, pp. 25–27.
8. N. Metropolis et al., "Equation of State Calculations by Fast Computing Machines," *J. Chemical Physics*, vol. 21, no. 6, 1953, pp. 1087–1092.
9. D. Frenkel and B. Smit, *Understanding Molecular Simulation*, 2nd ed., Academic Press, 2002.
10. C.M. Fortuin and P.W. Kasteleyn, "On the Random-Cluster Model, I: Introduction and Relation to Other Models," *Physica*, vol. 57, no. 4, 1972, pp. 536–564.
11. J.R. Heringa and H.W.J. Blöte, "The Simple-Cubic Lattice Gas with Nearest-Neighbour Exclusion: Ising Universality," *Physica A*, vol. 232, nos. 1–2, 1996, pp. 369–374.
12. A. Buhot and W. Krauth, "Numerical Solution of Hard-Core Mixtures," *Physical Rev. Letters*, vol. 80, no. 17, 1998, pp. 3787–3790.
13. L. Santen and W. Krauth, "Absence of Thermodynamic Phase Transition in a Model Glass Former," *Nature*, vol. 405, no. 6786, 2000, pp. 550–551.
14. V. Tohver et al., "Nanoparticle Halos: A New Colloid Stabilization Mechanism," *Proc. Nat'l Academy Science USA*, vol. 98, no. 16, 2001, pp. 8950–8954.
15. J. Liu and E. Luijten, "Stabilization of Colloidal Suspensions by Means of Highly Charged Nanoparticles," *Physical Rev. Letters*, vol. 93, no. 24, 2004, article no. 247802.

***Erik Luijten** is an assistant professor of materials science and engineering at the University of Illinois at Urbana-Champaign. His research interests include soft condensed matter, statistical mechanics, and algorithm development. Luijten has degrees in physics from Utrecht University and the Delft University of Technology. Contact him at luijten@uiuc.edu.*