# Introduction to Cluster Monte Carlo Algorithms

E. Luijten

Department of Materials Science and Engineering, Frederick Seitz Materials Research Laboratory, University of Illinois at Urbana-Champaign, Urbana, Illinois 61801, U.S.A.
luijten@uiuc.edu

Erik Luijten

E. Luijten: *Introduction to Cluster Monte Carlo Algorithms*, Lect. Notes Phys. **703**, 13–38 (2006)
DOI 10.1007/3-540-35273-2_1

This chapter provides an introduction to cluster Monte Carlo algorithms for classical statistical-mechanical systems. A brief review of the conventional Metropolis algorithm is given, followed by a detailed discussion of the lattice cluster algorithm developed by Swendsen and Wang and the single-cluster variant introduced by Wolff. For continuum systems, the geometric cluster algorithm of Dress and Krauth is described. It is shown how their geometric approach can be generalized to incorporate particle interactions beyond hard-core repulsions, thus forging a connection between the lattice and continuum approaches. Several illustrative examples are discussed.

## 1 Introduction

The Monte Carlo method is applied in wide areas of science and engineering. This chapter specifically focuses on its use in equilibrium statistical mechanics of classical systems of particles or spins. To set the stage, I first review the fundamental concepts of Monte Carlo simulation and discuss the importance sampling method introduced by Metropolis et al. However, the main emphasis in these notes lies on so-called cluster methods that have been developed over the last two decades. These are collective-update schemes that are capable of generating independent particle configurations in a very efficient manner. While originally believed to be only applicable to lattice-based systems, they now have been extended to large classes of off-lattice models. Their wide range of applicability is illustrated by means of several examples.

## 2 Local Monte Carlo Simulations

### 2.1 Importance Sampling and the Metropolis Method

It is one of the fundamental results of statistical mechanics that a thermodynamic system is described by its *partition function*,

$$Z = \sum_{\{s\}} \exp(-\beta E_s) \ , \tag{1}$$

where the sum runs over all possible states of the system, $E_s$ denotes the energy of state $s$, and $\beta = 1/(k_{\mathrm{B}}T)$, with $k_{\mathrm{B}}$ Boltzmann's constant and $T$ the absolute temperature. Thermodynamic properties can be computed as ensemble averages,

$$\langle A \rangle = \frac{1}{Z} \sum_{\{s\}} A_s \exp(-\beta E_s) \ , \tag{2}$$

where $\langle A \rangle$ is the expectation value of an observable and $A_s$ is the value of this observable if the system resides in state $s$.

The integrals (summations) in (1) and (2) are taken over phase space, which spans $2dN$ dimensions for $N$ particles in a system of dimensionality $d$. If a conventional numerical integration method would be applied, a prohibitive computational effort would be required to obtain an acceptable accuracy. An alternative approach is *simple sampling*. The integrand is evaluated for a set of randomly chosen states ("samples"), and the mean of these individual evaluations is an estimate of the integral. While this indeed works for smoothly varying functions, the Boltzmann factor $\exp(-\beta E_s)$ is vanishingly small for most samples, making the statistical uncertainty in the integral very large. In general, this problem can be resolved via *importance sampling*, in which the samples are chosen according to a probability distribution. By preferentially sampling states that strongly contribute to the integral in (2) the variance in the estimate of this integral is greatly reduced. Specifically, we desire to 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 cannot compute their *absolute* probability, since we do not know the normalization constant $Z$. It is the accomplishment of Metropolis et al. [1] to have found a way to calculate expectation values (2) *without* evaluating the partition function. The basic idea is to create a Markov chain of states, i.e., a sequence of states in which each state only depends on the state immediately preceding it. One starts from a configuration $s_i$ that has a nonvanishing Boltzmann factor $p_i$. This is the first member of the Markov chain. From this configuration, a new *trial* configuration $s_j$ is created, which has a Boltzmann factor $p_j$. The trial configuration is either accepted or rejected. If it is accepted, it is the next member of the Markov chain. If it is rejected, then the next member of the Markov chain is again $s_i$. This process is repeated iteratively to generate a sequence of configurations. It is emphasized that each trial configuration is created from the previous state in the Markov chain and accepted or rejected only based upon a comparison with this previous state. There is thus a *transition probability* from each state $s_i$ to each state $s_j$, represented by a *transition matrix* $\pi_{ij}$. It is our goal to find a transition matrix that yields the equilibrium distribution $p_j$. Evidently, this matrix must satisfy the condition

$$\sum_i p_i \pi_{ij} = p_j \ . \tag{3}$$

Finding a solution $\pi_{ij}$ of this equation is greatly simplified by imposing the condition of *microscopic reversibility* or *detailed balance*, i.e., 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 $i$.

$$p_i \pi_{ij} = p_j \pi_{ji} \ . \tag{4}$$

Indeed, summation over all states $s_i$ reduces (4) to (3):

$$\sum_i p_i \pi_{ij} = \sum_i p_j \pi_{ji} = p_j \sum_i \pi_{ji} = p_j \ , \tag{5}$$

where we have used that

$$\sum_i \pi_{ji} = 1 \ . \tag{6}$$

The matrix elements $\pi_{ij}$ are the product of two factors, namely an *a priori probability* $\alpha_{ij}$ of generating a trial configuration $s_j$ from a configuration $s_i$ and an *acceptance probability* $P_{ij}$ of accepting the trial configuration as the new state. The detailed balance condition can thus be written as

$$p_i \alpha_{ij} P_{ij} = p_j \alpha_{ji} P_{ji} \ . \tag{7}$$

In the simplest scheme, $\alpha_{ij}$ is symmetric and the condition reduces to

$$p_i P_{ij} = p_j P_{ji} \ , \tag{8}$$

which can be rewritten as

$$\frac{P_{ij}}{P_{ji}} = \exp\left[-\beta(E_j - E_i)\right] \ . \tag{9}$$

The acceptance probability is not uniquely defined by this equation. Metropolis et al. [1] proposed the solution

$$P_{ij} = \begin{cases} \exp\left[-\beta(E_j - E_i)\right] & \text{if } E_j > E_i \\ 1 & \text{if } E_j \leq E_i \end{cases} \ , \tag{10}$$

which is sometimes summarized as

$$P_{ij} = \min\left[\exp\left(-\beta \Delta_{ij}\right), 1\right] \ , \tag{11}$$

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

The trial configuration $s_j$ is generated via a so-called *trial move*. For example, if we consider an assembly of $N$ particles, then a trial move can consist of a small displacement of one particle, in a random direction. If the resulting configuration has an energy that is lower than the original configuration, the new state is always accepted. If the trial configuration has a higher energy, it is only accepted with a probability equal to the ratio of the Boltzmann factor of the new configuration and the Boltzmann factor of the original configuration. In practice, this is realized by generating a random number $0 \leq r < 1$ and accepting the trial configuration only if $r < P_{ij}$.

The expectation value of a thermodynamic property $A$ is calculated as follows. A sequence $\{s_1, \ldots, s_M\}$ of $M$ configurations is generated, and for each configuration $s_n$ the property $A_n$ is sampled. The thermodynamic average (2) is then estimated as a *simple average*,

$$\langle A \rangle \approx \frac{1}{M} \sum_{n=1}^{M} A_n \ . \tag{12}$$

### 2.2 Elementary Moves and Ergodicity

The *trial moves* or *elementary moves* that are used to generate a trial configuration depend on the nature of the system. A simple model for a fluid is the above-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 $\mathbf{r}' = \mathbf{r} + \delta\mathbf{r}$. The displacement $\delta\mathbf{r}$ can be chosen as a randomly oriented vector on a sphere with radius $0 < |\delta\mathbf{r}| < \ell$. However, it is computationally simpler to choose the new position $\mathbf{r}'$ within a cube of linear dimension $\ell$, centered around the original position $\mathbf{r}$. In either case (sphere or cube) the a priori probability $\alpha_{ij}$ is symmetric, i.e., the probability to generate the trial configuration from the original configuration is identical to the probability of the reverse process. As will be discussed in Sect. 2.3, the parameter $\ell$ permits control over the efficiency of the simulation. Monte Carlo algorithms with trial moves that involve small displacements of individual particles are also called *local-update algorithms*.

A valid Monte Carlo scheme must not only obey detailed balance, but must also be ergodic. This means that there is a path 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, then the estimator for a thermodynamic observable can differ severely from the correct expectation value.

### 2.3 Efficiency Considerations

The statistical quality of the estimate (12) depends on the number of *independent* samples in the sequence of configurations, and it is therefore the objective of a simulation to maximize the rate at which independent configurations are generated. If a trial configuration is generated via a small change to the previous configuration, the energy difference $\Delta_{ij}$ will typically be small and the acceptance ratio will be large. However, many configurations may have to be generated before an *independent* configuration results. Conversely, if a trial configuration is generated via a big change to the previous configuration then the sequence of configurations would decorrelate quickly, were it not that the typical energy difference will be large and the acceptance probability thus very small. For the elementary moves described in Sect. 2.2, the parameter $\ell$ controls the maximum displacement of a particle and thus the acceptance ratio.

It follows from these considerations that it is not always desirable to sample the property $A_n$ in (12) in each successive configuration, in particular not if the calculation of $A_n$ is computationally expensive. However, it is crucial that the sampling takes place at a *regular* interval. A typical mistake in Monte Carlo calculations is that a sample only is taken after a fixed number of trial configurations have been *accepted*. This is wrong, as can also easily be seen from the following example: At low temperatures a configuration with a low

energy is very unlikely to make a transition to a state with a higher energy. But this precisely borne out by (2): Low-energy states contribute more strongly to the expectation value and hence should be sampled more frequently than high-energy states.

In addition to variation of the maximum single-particle displacement, one may also attempt 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 is less efficient than a sequence of single-particle moves [2].

### 2.4 Ising Model

In order to describe the collective-update schemes that are the focus of this chapter, it is necessary to introduce the Ising model. This model is defined on a $d$-dimensional lattice of linear size $L$ (a square lattice in $d = 2$ and a 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. This system is described by the Hamiltonian,

$$\mathcal{H}_{\text{Ising}} = -J \sum_{\langle ij \rangle} s_i s_j \ . \tag{13}$$

The spins $s$ take values $\pm 1$. The sum runs over all pairs of nearest neighbors, which are coupled via a ferromagnetic coupling with strength $J > 0$. The Metropolis algorithm can be applied directly to this system. Local trial moves amount to the inversion of a single spin and are accepted or rejected on the basis of the change in coupling energy.

## 3 Lattice Cluster Algorithms

### 3.1 Swendsen–Wang Algorithm

In 1987, Swendsen and Wang (SW) [3] introduced a new Monte Carlo algorithm for the Ising spin model, which constituted a radical departure from the Metropolis or "single-spin flip" method used until then. Since the "recipe" is relatively straightforward, it is instructive to begin with a description of this algorithm.

Starting from a given configuration of spins, the SW algorithm proceeds as follows:

1. A "bond" is formed between every pair of nearest neighbors that are aligned, with a probability $p_{ij} = 1 - \exp(-2\beta J)$, where $J$ is the coupling constant [cf. (13)].
2. All spins that are connected, directly or indirectly, via bonds belong to a single cluster. Thus, the bond assignment procedure divides the system into clusters of parallel spins (a so-called *cluster decomposition*). Note how the bond probability (and hence the typical cluster size) grows with

increasing coupling strength $\beta J$ (decreasing temperature). For finite $\beta J$, $p_{ij} < 1$ and hence a cluster is generally a *subset* of all spins of a given sign – in other words, two spins of the same sign need not belong to the same cluster, even if these spins are adjacent on the lattice.
3. All spins in each cluster are flipped *collectively* with a probability $\frac{1}{2}$. I.e., for each cluster of spins a spin value $\pm 1$ is chosen and this value is assigned to all spins that belong to the cluster.
4. All bonds are erased and the "cluster move" is complete; a new spin configuration has been created. The algorithm restarts at step (1).

Step (3) is the crucial one. It is made possible by the so-called Fortuin–Kasteleyn mapping [4, 5] of the Ising model on the random-cluster model.[1] This mathematical result essentially shows that the partition function of the Potts model can be written as a sum over all possible clusters, or "graphs," on the lattice. Consequently, all spins in a cluster (a "connected component" in the random-cluster model) are uncorrelated with all spins that belong to other clusters and can be independently assigned a new spin value. Here, we use the word *cluster* in a general sense: A cluster may also consist of a single, isolated spin. The independence of clusters also implies that the cluster-flip probability in step (3) can be chosen at will. Evidently, if this probability is very small the configuration does not change much. On the other hand, flipping the spins in *all* clusters amounts to an inversion of the entire sample and therefore does not accomplish anything. Thus, a probability halfway these two extremes is typically chosen in order to maximize the rate at which the system evolves.

One remarkable aspect of this algorithm is that it is *rejection free*.[2] Indeed, the assignment of bonds involves specific probabilities, but once the clusters have been formed each of them can be flipped independently without imposing an acceptance criterion that involves the energy change induced by such a collective spin-reversal operation. We note that the absence of an acceptance criterion does *not* imply that a cluster flip does not entail an energy difference! Indeed, there is nothing in the algorithm that would guarantee this property (which would require that the boundary of the cluster cuts through an equal number of parallel and antiparallel pairs of interacting spins). Furthermore, this peculiar property would cause the system to move over a constant-energy surface in phase space, which is certainly not what one desires for a simulation that operates in the canonical ensemble. In contrast, the cluster flips *do* result in a sequence of configurations with different energies, in such a way that they appear exactly according to the Boltzmann distribution.

The aspect that made the SW algorithm very popular is its ability to strongly suppress dynamic slowing down near a critical point. This can be

[1] Strictly, the Fortuin–Kasteleyn mapping applies to the $q$-state Potts model, but the Ising model is equivalent to a Potts model with $q = 2$.

[2] The selection of clusters that are flipped and clusters that are not flipped is considered as part of the move, and not as a "rejection."

explained by a brief digression into the field of critical phenomena. For a substance near a critical point (a continuous phase transition), the relaxation time of thermodynamic properties depends as a power law on the correlation length $\xi$,

$$\tau \propto \xi^z , \tag{14}$$

where $z \approx 2$ is the so-called dynamical critical exponent [6]. The correlation length itself diverges as a power law of the difference between the temperature $T$ of the substance and the critical temperature $T_c$,

$$\xi \propto |T - T_c|^{-\nu} , \tag{15}$$

where $\nu$ is a positive exponent. In simulations of finite systems, e.g., a $d$-dimensional "hypercube" of volume $L^d$, the correlation length is bounded by the linear system size $L$. Thus, if the temperature approaches $T_c$, $\xi$ grows according to (15) until it reaches a maximum value $\xi_{max} \propto L$, and for temperatures sufficiently close to the critical temperature, (14) is replaced by

$$\tau \propto L^z . \tag{16}$$

We thus encounter a phenomenon called *critical slowing down.* If a system becomes larger, the correlation time grows very rapidly and it becomes increasingly difficult to generate statistically independent configurations. However, the clusters created in the Swendsen–Wang algorithm have a structure that is very efficient at destroying nonlocal correlations. As a result, the dynamical critical exponent $z$ is lowered to a much smaller value and independent configurations can be generated at a much faster rate than with a single-spin flip algorithm. This advantage only holds in the vicinity of the critical temperature, which also happens to be the most interesting point in the study of lattice spin models such as the Ising model.

### 3.2 Wolff or Single-Cluster Algorithm

Soon after the SW algorithm described in Sect. 3.1 had been developed, Wolff [7] introduced a so-called single-cluster variant of this algorithm. In the SW algorithm, small and large clusters are created. While the destruction of critical correlations is predominantly due to the large clusters, a considerable amount of effort is spent on constructing the smaller clusters. In Wolff's implementation, no decomposition of the entire spin configuration into clusters takes place. Instead, 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 by means of the large clusters in the SW algorithm, without the effort of creating the smaller clusters that fill up the remainder of the system. If the Wolff cluster turns out to be small, then not much is gained, but also not much computational effort is required. As a result, critical slowing down is suppressed even more strongly than in the SW algorithm, and the dynamical critical exponent $z$ [see (16)] is even smaller.

A convenient side effect, which has certainly contributed to the popularity of the Wolff algorithm, is that it is exceedingly simple to implement. The prescription is as follows:

1. A spin $i$ is selected at random.
2. All nearest neighbors $j$ of this spin are added 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 $j$ has not been considered before.
3. Each spin $j$ that is indeed added to the cluster is also placed on the stack. Once all neighbors of $i$ have been considered for inclusion in the cluster, a spin is retrieved from the stack and all its neighbors are considered in turn for inclusion in the cluster as well, following step (2).
4. Steps (2) and (3) are repeated iteratively until the stack is empty.
5. Once the cluster has been completed, all spins that belong to the cluster are inverted.

Again, this is a rejection-free algorithm, in the sense that the cluster is always flipped. Just as in the SW algorithm, the cluster-construction process is probabilistic, but the probabilities $p_{ij}$ involve energies of individual spin pairs in contrast with an acceptance criterion that involves the *total* energy change induced by a cluster flip. The implementation can be simplified by a small trick: In step (2), each spin $j$ that is added to the cluster can immediately be inverted. This guarantees that a spin is never added twice. Step (5) can then be eliminated.[3]

### 3.3 Cluster Algorithms for Other Lattice Models

The algorithms described here are not restricted to the Ising model, but can be applied to a number of other problems. (i) For multicomponent spins (such as the XY or planar model), Wolff [7] replaced the spin-inversion operation by a reflection operation in which only the component of a spin is reversed that is orthogonal to a randomly oriented plane. For each cluster, a new orientation of the plane is chosen. (ii) The original Fortuin–Kasteleyn mapping is valid for $q$-state Potts models in which each lattice site corresponds to a variable that can take $q$ different, equivalent states. In the SW algorithm, after the cluster decomposition, one of these $q$ values is assigned with probability $1/q$ to each cluster and all variables in the cluster take this new value. (iii) The Fortuin–Kasteleyn mapping can also be applied to systems in which each spin interacts not only with its nearest neighbors, but also with other spins [8]. In particular, the coupling strength can be different for different spin pairs, leading to a probability $p_{ij}$ that is, e.g., dependent on the separation between spins $i$ and $j$. While this permits the direct formulation of a cluster Monte

[3] To formulate this more precisely, also spin $i$ should be inverted upon selection in step (1) and in step (2) the spins $j$ must be considered for addition to the cluster if their sign is the same as the *original* sign of spin $i$.

Carlo algorithm for these systems, the cluster addition probability becomes very small if the interaction is long-ranged. As shown by Luijten and Blöte [9], this can be resolved by reformulating the algorithm, allowing the study of spin systems with medium- [10] and long-range [11, 12] ferromagnetic interactions, as well as dipolar [13] interactions.

## 4 Cluster Algorithms for Continuum Systems

### 4.1 Geometric Cluster Algorithm for Hard-Sphere Mixtures

The advantages brought by the cluster algorithms described in Sect. 3, in particular the suppression of critical slowing down, made it a widely pursued goal to generalize the SW and Wolff algorithms to fluid systems in which particles are not confined to lattice sites but can take arbitrary positions in continuum space. Unfortunately, the absence of a lattice structure breaks a fundamental symmetry, rendering such attempts largely unsuccessful. An Ising model can be interpreted as a so-called *lattice gas*, where 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 off-lattice (continuum) systems. While a particle in a fluid configuration can straightforwardly be deleted, there is no unambiguous prescription on how to transform empty space into a particle. More precisely, in the lattice cluster algorithms the operation performed on every spin is self-inverse. This requirement is not fulfilled for off-lattice fluids.

Independently of these efforts, in 1995 Dress and Krauth [14] 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 have no interaction as long as they do not overlap. Because of the *hard-core repulsion*, a Monte Carlo algorithm involving local moves is relatively inefficient, since any move that generates a particle overlap is rejected. Instead, the *geometric cluster algorithm* (GCA) [14] is designed to avoid such overlaps while generating a new configuration, by proceeding as follows (cf. Fig. 1).

1. In a given configuration $C$ of particles, a "pivot" is chosen at random.
2. A configuration $\tilde{C}$ is now generated by carrying out a point reflection for all particles in $C$ with respect to the pivot.[4]
3. The configuration $C$ and its transformed counterpart $\tilde{C}$ are superimposed, which leads to groups of overlapping particles. The groups generally come

[4] In the original algorithm, a $\pi$ rotation with respect to the pivot was performed for all particles. In the two-dimensional example of Fig. 1 this is equivalent to a point reflection, but the reflection is more suitable for generalization to higher dimensions.

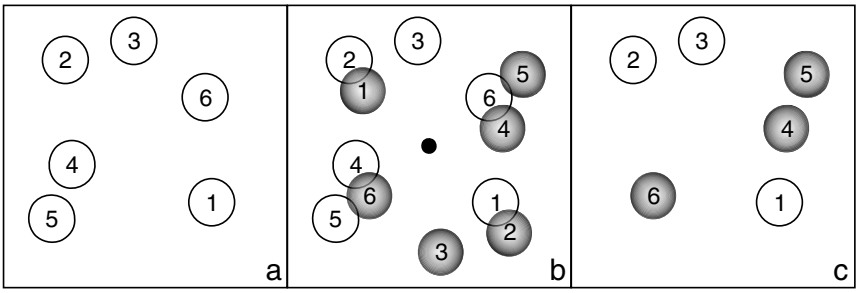


**Fig. 1.** Illustration of the geometric cluster algorithm for hard disks [14]. (**a**) Original configuration. (**b**) A new configuration (*shaded circles*) is created by means of a point reflection of all particles with respect to a randomly chosen pivot point (*small filled disk*). The superposition of the original and the new configuration leads to groups of overlapping particles. In this example, there are three *pairs* of groups ($\{1,2\}$, $\{3\}$, $\{4,5,6\}$). Each pair is denoted a *cluster*. The particles in any one of these clusters can be point-reflected with respect to the pivot without affecting the other two clusters. This can be used to carry out the point reflection for every cluster with a pre-set probability. (**c**) Final configuration that results if, starting from the original configuration, only the particles in the third cluster $\{4,5,6\}$ are point-reflected. This approach guarantees that every generated configuration will be free of overlaps. Note that the pivot will generally not be placed in the center of the cell, and that the periodic boundary conditions indeed permit any position. Reprinted figure with permission from [16]. Copyright 2005 by the American Physical Society

in pairs, except possibly for a single group that is symmetric with respect to the pivot. Each pair is denoted a "cluster."

4. For each cluster, all particles can be exchanged between $C$ and $\tilde{C}$ without affecting particles belonging to other clusters. This exchange is performed for each cluster independently with a probability $\frac{1}{2}$. Thus, if the superposition of $C$ and $\tilde{C}$ is decomposed into $N$ clusters, there are $2^N$ possible new configurations. The configurations that are actually realized are denoted $C'$ and $\tilde{C}'$, i.e., the original configuration $C$ is transformed into $C'$ and its point-reflected counterpart $\tilde{C}$ is transformed into $\tilde{C}'$.
5. The configuration $\tilde{C}'$ is discarded and $C'$ is the new configuration, serving as the starting point for the next iteration of the algorithm. Note that a new pivot is chosen in every iteration.

Observe that periodic boundary conditions must be employed, such that an arbitrary placement of the pivot is possible. Other self-inverse operations are permissible, such as a reflection in a plane [15], in which case various orientations of the plane must be chosen in order to satisfy ergodicity.

Comparison to the lattice cluster algorithms of Sect. 3 shows that the SW and Wolff algorithms operate in the grand-canonical ensemble, in which the cluster moves do not conserve the magnetization (or the number of particles, in the lattice-gas interpretation), whereas the geometric cluster algorithm

operates in the canonical ensemble. Nevertheless, this prescription bears a remarkable resemblance to the SW algorithm. The original configuration is decomposed 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 and in the geometric cluster algorithm it is a geometric symmetry operation. Subsequently, a new configuration is created by moving each cluster independently with a certain probability.

This approach is very general. For example, it is not restricted to monodisperse systems, and Krauth and co-workers have applied it successfully to binary [17] and polydisperse [18] 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 geometric cluster algorithm particles are moved in a nonlocal fashion, yet overlaps are avoided.

The most important limitation of the GCA is the fact that the average cluster size increases very rapidly for systems with a density that exceeds the percolation threshold of the combined system containing the superposition of the configurations $C$ and $\tilde{C}$. Once the clusters span the entire system, the algorithm is clearly no longer ergodic.

In order to emphasize the analogy with the lattice cluster algorithms, we can also formulate a single-cluster (Wolff) variant of the geometric cluster algorithm [15, 19].

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

As in the SW-like prescription, a new pivot is chosen for each cluster that is constructed.

### 4.2 Generalized Geometric Cluster Algorithm for Interacting Particles

The geometric cluster algorithm described in the previous section is formulated for particles that interact via hard-core repulsions only. Clearly, in order to make this approach widely applicable, a generalization to other types of pair potentials must be found. Thus, Dress and Krauth [14] suggested to impose a Metropolis-type acceptance criterion, based upon the energy difference induced by the cluster move. Indeed, if a pair potential consists of a hard-core contribution supplemented by an attractive or repulsive tail, such as a

Yukawa potential, the cluster-construction procedure takes into account the excluded-volume contribution, guaranteeing that no overlaps are generated, and the acceptance criterion takes into account the tail of the interactions. For "soft-core" potentials, such as a Lennard-Jones interaction, the situation becomes already somewhat more complicated, since an arbitrary excluded-volume distance must be chosen in the cluster construction. As the algorithm will not generate configurations in which the separation (center-to-center distance) between a pair of particles is less than this distance (i.e., the particle "diameter," in the case of monodisperse systems), it must be set to a value that is smaller than any separation that would typically occur. It is important to recognize that both for hard-core particles with an additional tail and for soft-core particles the clusters are now constructed on the basis of only a part of the pair potential, and the evaluation of a part of the energy change resulting from a cluster move is deferred until the acceptance step. As a result, the computational efficiency can decrease significantly, since – similar to the situation for regular multiple-particle moves – rejection of a cluster move is quite likely. To make things worse, every rejection leads to a considerable waste of computational effort spent on the construction of the cluster and the evaluation of the corresponding energy change. Nevertheless, this approach certainly works in principle, as shown by a study of Yukawa mixtures with moderate size asymmetry (diameter ratio $\leq 5$) [20].

However, an entirely different approach is possible, by carrying the analogy with the lattice cluster algorithms further. The probability $p_{ij}$ to add a spin $j$ (which is neighboring a spin $i$) to the cluster in the SW algorithm can be phrased in terms of the corresponding energy difference. Two different situations can be discerned that lead to a change in the relative energy $\Delta_{ij}^{\mathrm{SW}}$ between a spin $i$ that belongs to the cluster and a spin $j$ that does not yet belong to the cluster. 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 $j$ is not added to the cluster, the resulting change in the pair energy equals $\Delta_{ij}^{\mathrm{SW}} = +2J > 0$. This occurs with a probability $\exp(-2\beta J) < 1$. These two situations can be summarized as

$$1 - p_{ij} = \min\left[\exp(-\beta\Delta_{ij}^{\mathrm{SW}}), 1\right] \ , \tag{17}$$

so that the probability of adding spin $j$ to the cluster can be written as $p_{ij} = \max[1 - \exp(-\beta\Delta_{ij}^{\mathrm{SW}}), 0]$. The GCA, although formulated in continuum space rather than on a lattice, can now be interpreted as special situation in which either $\Delta_{ij} = 0$ (after reflection of particle $i$, there is no overlap between particles $i$ and $j$), leading to $p_{ij} = 0$, or $\Delta_{ij} = \infty$ (after point reflection, particle $i$ overlaps with particle $j$), leading to $p_{ij} = 1$.

A generalization of the GCA to general pair potentials then follows in a natural way [19]. All interactions are treated in a unified manner, so that there is no technical distinction between attractive and repulsive interactions or between hard-core and soft-core potentials. This *generalized GCA* is most

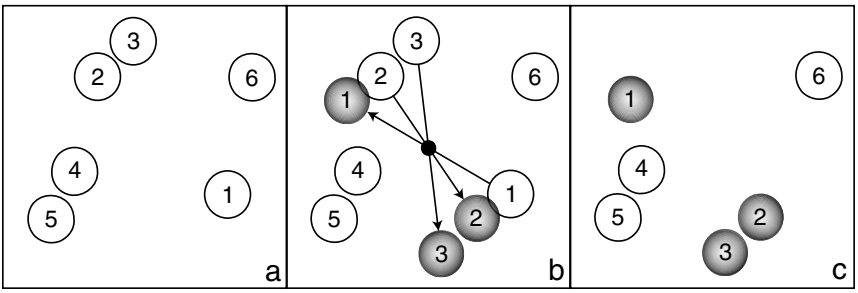


**Fig. 2.** Two-dimensional illustration of the interacting geometric cluster algorithm. Like in Fig. 1, open and shaded disks denote the particles before and after the geometric operation, respectively, and the small disk denotes the pivot. However, in the *generalized* GCA a single cluster is constructed, to which particles are added with an interaction-dependent probability. (**a**) 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 4–6 are not affected by these point reflections, the cluster construction terminates. (**c**) The new configuration consists of particles 1–3 in their new positions and particles 4–6 in the original positions. A new pivot is chosen and the procedure is repeated. Reprinted figure with permission from [16]. Copyright 2005 by the American Physical Society

easily described as a combination of the single-cluster methods formulated in Sect. 3.2 and Sect. 4.1. We assume a general pair potential $V_{ij}(\mathbf{r}_{ij})$ that does not have to be identical for all pairs $(i, j)$ (see Fig. 2).

1. In a given configuration $C$, a "pivot" is chosen at random.
2. A particle $i$ at position $\mathbf{r}_i$ is selected as the first particle that belongs to the cluster. This particle is moved 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. Each particle $j$ that interacts with $i$ or $i'$ is now considered for addition to the cluster. A particle $j$ that interacts with $i$ both in its old and in its new position is nevertheless treated 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 $j$ that is indeed added to the cluster (i.e., moved) is also placed on the stack. Once all particles interacting with $i$ or $i'$ have been considered, a particle is retrieved from the stack and all 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 are repeated iteratively until the stack is empty. The cluster move is now complete.

If a particle interacts with multiple other particles that have been added to the cluster, it can thus be considered multiple times for inclusion. However, once it has been added to the cluster, it cannot be removed. This is an important point in practice, since particles undergo a point reflection already during the cluster construction process (and thus need to be tagged, in order to prevent them from being returned to their original position by a second point reflection). A crucial aspect is that the probability $p_{ij}$ *only* depends 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* added to the cluster (and hence not point-reflected). This happens with a probability $1-p_{ij} = \min[\exp(-\beta\Delta_{ij}), 1]$, just as we found for the SW algorithm in (17). The similarity of this probability to the Metropolis acceptance criterion is deceptive (and merely reflects the fact that both algorithms aim to generate configurations according to the Boltzmann distribution), since $\Delta_{ij}$ does not represent the *total* energy change resulting from the translation of particle $i$. Instead, other energy changes are taken into account via the iterative nature of the algorithm.

It is interesting to note that the GCA can also be applied to lattice-based models. This was first done for the Ising model by Heringa and Blöte [21,22], who also devised a way to take into account the nearest-neighbor interactions between spins already during the cluster construction. While this lattice model can obviously be simulated by the SW and Wolff algorithms, their approach permits simulation in the constant-magnetization ensemble. Since the geometric operations employed map the spin lattice onto itself, excluded-volume conditions are satisfied automatically: Every spin move amounts to an *exchange* of spins. For every spin pair $(i, i')$ that is exchanged, each of its nearest-neighbor pairs $(k, k')$ is exchanged with a probability that depends on the change in pair energy, $\Delta = (E_{ik} + E_{i'k'}) - (E_{ik'} + E_{i'k})$. This procedure is then again performed iteratively for the neighbors of all spin pairs that are exchanged. This is similar to the generalized GCA discussed above, although, in the absence of a lattice, particles are added to the cluster on an individual basis rather than in pairs.

In order to establish the correctness of the generalized GCA, we need to prove ergodicity as well as detailed balance. The ergodicity of this algorithm follows from the fact that there is a nonvanishing probability that a cluster consists of only one particle, which can be moved over an arbitrarily small distance, since the location of the pivot is chosen at random. This obviously requires that not all particles are part of the cluster, a condition that is violated at high packing fractions and, depending on the nature of the interactions, at very strong coupling strengths.

Detailed balance is proven as follows. We consider a configuration $X$ that is transformed into a configuration $Y$ by means of a cluster move. All particles included in the cluster maintain their relative separation; as noted above, an energy change arises if a particle is *not* included in the cluster, but interacts with a particle that does belong to the cluster. Following Wolff [7] we denote each of these interactions as a "broken bond." A broken bond $k$ that

corresponds to an energy change $\Delta_k$ occurs with a probability $1 - p_k = 1$ if $\Delta_k \leq 0$ and a probability $1 - p_k = \exp(-\beta\Delta_k)$ if $\Delta_k > 0$. The formation of an entire cluster corresponds to the breaking of a set $\{k\}$ of bonds, which has a probability $P$. This set is comprised of the subset $\{l\}$ of broken bonds $l$ that lead to an increase in pair energy and the subset $\{m\}$ of broken bonds that lead to a decrease in pair energy, such that

$$P = \prod_k (1 - p_k) = \exp\left[-\beta \sum_l \Delta_l\right] . \tag{18}$$

The transition probability from configuration $X$ to configuration $Y$ is proportional to the cluster formation probability,

$$T(X \to Y) = C \exp\left[-\beta \sum_l \Delta_l\right] , \tag{19}$$

where the factor $C$ accounts for the fact that various arrangements of bonds within the cluster ("internal bonds") correspond to the same set of broken bonds. In addition, it incorporates the probability of choosing a particular pivot and a specific particle as the starting point for the cluster.

If we now consider the reverse transition $Y \to X$, we observe that this again involves the set $\{k\}$, but all the energy differences change sign compared to the forward move. Consequently, the subset $\{l\}$ in (19) is replaced by its complement $\{m\}$ and the transition probability is given by

$$T(Y \to X) = C \exp\left[+\beta \sum_m \Delta_m\right] , \tag{20}$$

where the factor $C$ is identical to the prefactor in (19). Since we require the geometric operation to be self-inverse we thus find that the cluster move satisfies detailed balance at an acceptance ratio of unity,

$$\begin{aligned} \frac{T(X \to Y)}{T(Y \to X)} &= \frac{\exp\left[-\beta \sum_l \Delta_l\right]}{\exp\left[+\beta \sum_m \Delta_m\right]} = \exp\left[-\beta \sum_k \Delta_k\right] \\ &= \exp\left[-\beta (E_Y - E_X)\right] = \frac{\exp(-\beta E_Y)}{\exp(-\beta E_X)} , \end{aligned} \tag{21}$$

where $E_X$ and $E_Y$ are the internal energies of configurations $X$ and $Y$, respectively. That is, the ratio of the forward and reverse transition probabilities is equal to the inverse ratio of the Boltzmann factors, so that we indeed have created a rejection-free algorithm. This is obscured to some extent by the fact that in our prescription the cluster is moved while it is being constructed, similar to the Wolff algorithm in Sect. 3.2. The central point, however, is that the construction solely involves single-particle energies, whereas a Metropolis-type approach only evaluates the total energy change induced by a multi-particle

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 (i.e., strong bonds are unlikely to be broken).

### 4.3 Generalized Geometric Cluster Algorithm: Full Cluster Decomposition

It is instructive to also formulate a SW version of the generalized GCA, based upon the single-cluster version described in the previous section. This demonstrates that the generalized GCA is a true off-lattice counterpart of the cluster algorithms of Sect. 3. Furthermore, it is of conceptual interest, as this algorithm decomposes a continuum fluid configuration into *stochastically independent clusters*. This implies an interesting and remarkable analogy with the Ising model. As observed by Coniglio and Klein [23] for the two-dimensional Ising model at its critical point, the clusters created according to the prescription in Sect. 3 are just the so-called "Fisher droplets" [24]. Indeed, these "Coniglio–Klein clusters" are implied by the Fortuin–Kasteleyn mapping of the Potts model onto the random-cluster model [5], which in turn constitutes the basis for the Swendsen–Wang approach [3]. The clusters generated by the GCA do not have an immediate physical interpretation, as they typically consist of two spatially disconnected parts. However, just like the Ising clusters can be inverted at random, each cluster of fluid particles can be moved independently with respect to the remainder of the system. As such, the generalized GCA can be viewed as a continuum version of the Fortuin–Kasteleyn mapping.

The cluster decomposition of a configuration proceeds as follows. First, a cluster is constructed according to the single-cluster algorithm of Sect. 4.2, with the exception that the cluster is only *identified*; particles belonging to the cluster are marked but not actually moved. The pivot employed will also be used for the construction of all subsequent clusters in this decomposition. These subsequent clusters are built just like the first cluster, except that particles that are already part of an earlier cluster will never be considered for a new cluster. Once each particle is part of exactly one cluster the decomposition is completed. Like in the SW algorithm, every cluster can then be moved (i.e., all particles belonging to it are translated via a point reflection) independently, e.g., with a probability $f$. Despite the fact that all clusters except the first are built in a restricted fashion, each individual cluster is constructed according to the rules of the Wolff formulation of Sect. 4.2. The exclusion of particles that are already part of another cluster simply corresponds to the fact that every bond should be considered only once. If a bond is broken during the construction of an earlier cluster it should not be re-established during the construction of a subsequent cluster. The cluster decomposition thus obtained is not unique, as it depends on the placement of the pivot and the choice of the first particle. Evidently, this also holds for the SW algorithm.

In order to establish that this prescription is a true equivalent of the SW algorithm, we prove that each cluster can be moved (reflected) independently while preserving detailed balance. If only a single cluster is actually moved, this essentially corresponds to the Wolff version of the GCA, since each cluster is built according to the GCA prescription. The same holds true if several clusters are moved and no interactions are present between particles that belong to different clusters (the hard-sphere algorithm is a particular realization of this situation). If two or more clusters are moved and *broken* bonds exist between these clusters, i.e., a nonvanishing interaction exists between particles that belong to disparate (moving) clusters, then the shared broken bonds are actually preserved and the proof of detailed balance provided in the previous section no longer applies in its original form. However, since these bonds are identical in the forward and the reverse move, the corresponding factors cancel out. This is illustrated for the situation of two clusters whose construction involves, respectively, two sets of broken bonds $\{k_1\}$ and $\{k_2\}$. Each set comprises bonds $l$ ($\{l_1\}$ and $\{l_2\}$, respectively) that lead to an *increase* in pair energy and bonds $m$ ($\{m_1\}$ and $\{m_2\}$, respectively) that lead to a *decrease* in pair energy. We further subdivide these sets into *external* bonds that connect cluster 1 or 2 with the remainder of the system and *joint* bonds that connect cluster 1 with cluster 2. Accordingly, the probability of creating cluster 1 is given by

$$C_1 \prod_{i\in\{k_1\}} (1-p_i) = C_1 \prod_{i\in\{l_1\}} (1-p_i) = C_1 \prod_{i\in\left\{l_1^{\mathrm{ext}}\right\}} (1-p_i) \prod_{j\in\left\{l_1^{\mathrm{joint}}\right\}} (1-p_j)\ . \quad (22)$$

Upon construction of the first cluster, the creation of the second cluster has a probability

$$C_2 \prod_{i\in\left\{l_2^{\mathrm{ext}}\right\}} (1-p_i)\ , \quad (23)$$

since all joint bonds in $\{l_2^{\mathrm{joint}}\} = \{l_1^{\mathrm{joint}}\}$ already have been broken. The factors $C_1$ and $C_2$ refer to the probability of realizing a particular arrangement of internal bonds in clusters 1 and 2, respectively (cf. Sect. 4.2). Hence, the total transition probability of moving *both* clusters is given by

$$T_{12}(X\to Y) = C_1C_2 \exp\left[-\beta \sum_{i\in\left\{l_1^{\mathrm{ext}}\right\}} \Delta_i - \beta \sum_{j\in\left\{l_2^{\mathrm{ext}}\right\}} \Delta_j - \beta \sum_{n\in\left\{l_1^{\mathrm{joint}}\right\}} \Delta_n\right]\ . \quad (24)$$

In the reverse move, the energy differences for all external broken bonds have changed sign, but the energy differences for the joint bonds connecting cluster 1 and 2 are the same as in the forward move. Thus, cluster 1 is created with probability

$$C_1 \prod_{i\in\{m_1^{\rm ext}\}} (1-\bar{p}_i) \prod_{j\in\{l_1^{\rm joint}\}} (1-p_j) = C_1 \prod_{i\in\{m_1^{\rm ext}\}} \exp[+\beta\Delta_i] \prod_{j\in\{l_1^{\rm joint}\}} \exp[-\beta\Delta_j] \,, \tag{25}$$

where the $\bar{p}_i$ reflects the sign change of the energy differences compared to the forward move and the product over the external bonds involves the complement of the set $\{l_1^{\rm ext}\}$. The creation probability for the second cluster is

$$C_2 \prod_{i\in\{m_2^{\rm ext}\}} (1-\bar{p}_i) = C_2 \prod_{i\in\{m_2^{\rm ext}\}} \exp[+\beta\Delta_i] \tag{26}$$

and the total transition probability for the reverse move is

$$T_{12}(Y\to X) = C_1 C_2 \exp\left[+\beta \sum_{i\in\{m_1^{\rm ext}\}} \Delta_i + \beta \sum_{j\in\{m_2^{\rm ext}\}} \Delta_j - \beta \sum_{n\in\{l_1^{\rm joint}\}} \Delta_n\right] . \tag{27}$$

Accordingly, detailed balance is still fulfilled with an acceptance ratio of unity,

$$\frac{T_{12}(X\to Y)}{T_{12}(Y\to X)} = \exp\left[-\beta \sum_{i\in\{k_1^{\rm ext}\}} \Delta_i - \beta \sum_{j\in\{k_2^{\rm ext}\}} \Delta_j\right] = \exp\left[-\beta(E_Y - E_X)\right] \,, \tag{28}$$

in which $\{k_1^{\rm ext}\} = \{l_1^{\rm ext}\}\cup\{m_1^{\rm ext}\}$ and $\{k_2^{\rm ext}\} = \{l_2^{\rm ext}\}\cup\{m_2^{\rm ext}\}$ and $E_X$ and $E_Y$ refer to the internal energy of the system before and after the move, respectively. This treatment applies to any simultaneous move of clusters, so that *each cluster in the decomposition indeed can be moved independently* without violating detailed balance. This completes the proof of the multiple-cluster version of the GCA. It is noteworthy that the probabilities for breaking joint bonds in the forward and reverse moves cancel only because the probability in the cluster construction factorizes into individual probabilities.

### 4.4 Implementation Issues

The actual implementation of the generalized GCA involves a variety of issues. The point reflection with respect to the pivot requires careful consideration of the periodic boundary conditions. Furthermore, as mentioned above, particles that have been translated via a point reflection must not be translated again within the same cluster move, and particles that interact with a given cluster particle both before and after the translation of that cluster particle must be considered only once, on the basis of the difference in pair potential. One way to account for all interacting pairs in an efficient manner is the use of the cell index method [25]. For mixtures with large size asymmetries (the situation where the generalized GCA excels), it is natural to set up different cell structures, with cell lengths based upon the cutoffs of the various particle

interactions. For example, in the case of a binary mixture of two species with very different sizes and cutoff radii ($r_{\rm cut}^{\rm large}$ and $r_{\rm cut}^{\rm small}$, respectively), the use of a single cell structure with a cell size that is determined by the large particles would be highly inefficient for the smaller particles. Thus, two cell structures are constructed in this case (with cell sizes $l_{\rm large}$ and $l_{\rm small}$, respectively) and each particle is stored in the appropriate cell of the structure belonging to its species, and incorporated in the corresponding linked list, following the standard approach [25]. However, in order to efficiently deal with interactions between unlike species (which have a cutoff $r_{\rm cut}^{\rm ls}$), a mapping between the two cell structures is required. If all small particles that interact with a given large particle must be located, one proceeds as follows. First, the small cell $\mathbf{c}$ is identified in which the center of the large particle resides. Subsequently, the interacting particles are located by scanning over all small cells within a cubic box with linear size $2r_{\rm cut}^{\rm ls}$, centered around $\mathbf{c}$. This set of cells is predetermined at the beginning of a run and their indices are stored in an array. Each set contains approximately $N_{\rm cell} = (2r_{\rm cut}^{\rm ls}/l_{\rm small})^3$ members. In an efficient implementation, $l_{\rm small}$ is not much larger than $r_{\rm cut}^{\rm small}$, which for short-range interactions is of the order of the size of a small particle. Likewise, $r_{\rm cut}^{\rm ls}$ is typically of the order of the size of the large particle, so that $N_{\rm cell} = \mathcal{O}(\alpha^3)$, where $\alpha > 1$ denotes the size asymmetry between the two species. Since $N_{\rm cell}$ indices must be stored for each large cell, the memory requirements become very large for cases with large size asymmetry, such as the suspension of colloids and nanoparticles (size asymmetry $\alpha = 100$) studied in [26].

### 4.5 Illustration 1: Efficiency of the Generalized Geometric Cluster Algorithm

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 the particles involved in the move are selected in a very specific manner. The generalized GCA makes nonlocal collective moves possible, without any negative consequences in the acceptance ratio. The resulting efficiency gain is illustrated by means of an example taken from [19], namely a simple binary mixture containing 150 large particles of size $\sigma_{22}$, at fixed volume fraction $\phi_2 = 0.1$, and $N_1$ small particles, also at fixed volume fraction $\phi_1 = 0.1$. The efficiency is determined through the autocorrelation time, as a function of size asymmetry. As the size $\sigma_{11}$ of these small particles is varied from $\sigma_{22}/2$ to $\sigma_{22}/15$ (i.e., the size ratio $\alpha = \sigma_{22}/\sigma_{11}$ is increased from 2 to 15), their number increases from $N_1 = 1\,200$ to $506\,250$.

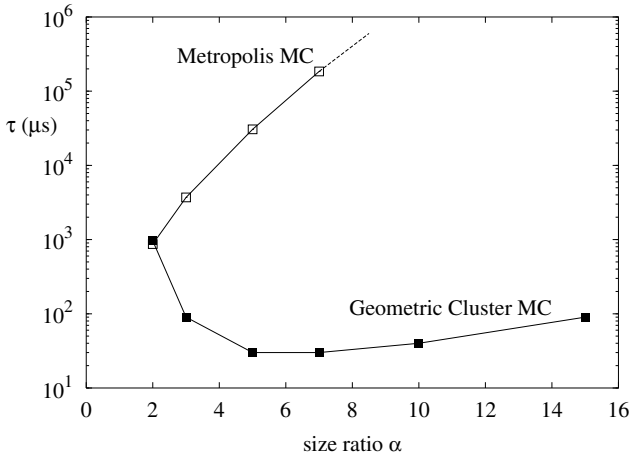


**Fig. 3.** Efficiency comparison between a conventional local update algorithm (*open symbols*) and the generalized geometric cluster algorithm (*closed symbols*), for a binary mixture (see text) with size ratio $\alpha$. Whereas the autocorrelation time per particle (expressed in μs of CPU time per particle move) rapidly increases with size ratio, the GCA features only a weak dependence on $\alpha$. Reprinted figure with permission from [19]. Copyright 2004 by the American Physical Society

Pairs of small particles and pairs involving a large and a small particle act like hard spheres. However, in order to prevent depletion-driven aggregation of the large particles [27], a short-ranged Yukawa repulsion is introduced,

$$U_{22}(r) = \begin{cases} +\infty & r \le \sigma_{22} \\ \frac{\sigma_{22}}{r}\varepsilon \exp[-\kappa(r-\sigma_{22})] & r > \sigma_{22} \end{cases} , \tag{29}$$

where $\beta\varepsilon = 3.0$ and the screening length $\kappa^{-1} = \sigma_{11}$. In the simulation, the exponential tail is cut off at $3\sigma_{22}$.

The additional Yukawa interactions also lead to a fluctuating internal energy $E(t)$ that makes it possible to determine the rate at which the large (and slower) particles decorrelate. We consider the integrated autocorrelation time $\tau$ obtained from the energy autocorrelation function [28],

$$C(t) = \frac{\langle E(0)E(t)\rangle - \langle E(0)\rangle^2}{\langle E(0)^2\rangle - \langle E(0)\rangle^2} , \tag{30}$$

and compare $\tau$ for a conventional (Metropolis) MC algorithm and the generalized GCA, see Fig. 3. In order to avoid arbitrariness resulting from the computational cost involved with a single sweep or the construction of a cluster, we assume that both methodologies have been programmed in an efficient manner and express $\tau$ in actual CPU time. Furthermore, $\tau$ is normalized by the total number of particles in the system, to account for the variation in $N_1$ as the size ratio $\alpha$ is increased. The autocorrelation time for the conventional

MC calculations, $\tau_{\mathrm{MC}}$, rapidly increases with increasing $\alpha$, because the large particles tend to get trapped by the small particles. Indeed, already for $\alpha > 7$ it is not feasible to obtain an accurate estimate for $\tau_{\mathrm{MC}}$. By contrast, $\tau_{\mathrm{GCA}}$ exhibits a very different dependence on $\alpha$. At $\alpha = 2$ both algorithms require virtually identical simulation time, which establishes that the GCA does not involve considerable overhead compared to standard algorithms (if any, it is mitigated by the fact that all moves are accepted). Upon increase of $\alpha$, $\tau_{\mathrm{GCA}}$ initially *decreases* until it starts to increase weakly. The nonmonotonic variation of $\tau_{\mathrm{GCA}}$ results from the changing ratio $N_2/N_1$ which causes the cluster composition to vary with $\alpha$. The main points to note are: (i) the GCA greatly suppresses the autocorrelation time, $\tau_{\mathrm{GCA}} \ll \tau_{\mathrm{MC}}$ for $\alpha > 2$, with an efficiency increase that amounts to more than three orders of magnitude already for $\alpha = 7$; (ii) the increase of the autocorrelation time with $\alpha$ is much slower for the GCA than for a local-move MC algorithm, making the GCA increasingly advantageous with increasing size asymmetry.

### 4.6 Illustration 2: Temperature and Cluster Size

The cluster size clearly has a crucial influence on the performance of the GCA. If a cluster contains more than 50% of all particles, an equivalent change to the system could have been made by moving its complement; unfortunately it is unclear how to determine this complement without constructing the cluster. Nevertheless, it is found that the algorithm can operate in a comparatively efficient manner for average relative cluster sizes as large as 90% or more. Once the total packing fraction of the system exceeds a certain value, the original hard-core GCA breaks down because each cluster occupies the entire system. The same phenomenon occurs in the generalized GCA, but in addition the cluster size can saturate because of strong interactions. Thus, the maximum accessible volume fraction depends on a considerable number of parameters, including the range of the potentials and the temperature. For multi-component mixtures, size asymmetry and relative abundance of the components are of importance as well, and the situation can be complicated further by the presence of competing interactions.

As an illustration, we consider the cluster-size distribution for a monodisperse Lennard-Jones fluid (particle diameter $\sigma$, interaction cut-off $2.5\sigma$) at a rather arbitrary density $0.16\sigma^{-3}$, for a range of temperatures, see Fig. 4. Whereas the distribution is a monotonously decreasing function of cluster size at high temperatures, it becomes bimodal at temperatures around 25% above the critical temperature. The bimodal form is indicative of the formation of large clusters.

It turns out to be possible to influence the cluster-size distribution by placing the pivot in a biased manner. Rather than first choosing the pivot location, a particle is selected that will become the first member of the cluster. Subsequently, the pivot is placed at random within a cubic box of linear size $\delta$, centered around the position of this particle. By decreasing $\delta$, the

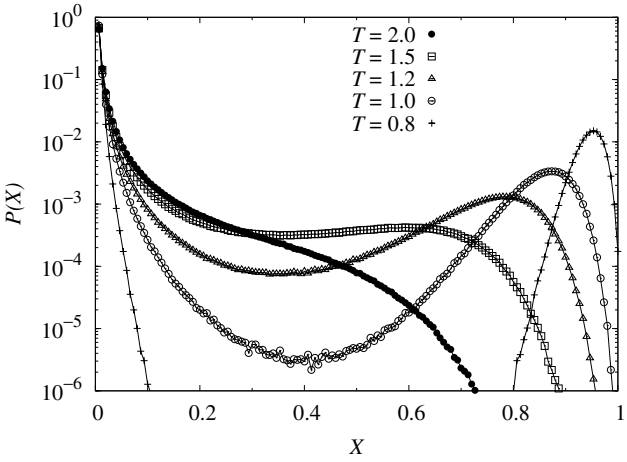


**Fig. 4.** Cluster-size distributions as a function of relative cluster size $X$, for a monodisperse Lennard-Jones fluid. The number density $\rho = 0.16\sigma^{-3}$ is set to 50% of the critical density. For low temperatures, the cluster-size distribution becomes bimodal. At higher temperatures, it decreases monotonically. All temperatures are indicated in units of $\varepsilon/k_{\mathrm{B}}$. Reprinted figure with permission from [16]. Copyright 2005 by the American Physical Society

displacement of the first particle is decreased, as well as the number of other particles affected by this displacement. As a consequence, the average cluster size decreases, and higher volume fractions can be reached. Ultimately, the cluster size will still occupy the entire system (making the algorithm no longer ergodic), but it has been found that the maximum accessible volume fraction can be increased from approximately 0.23 to a value close to 0.34. This value indeed corresponds to the percolation threshold for hard spheres. Note that the proof of detailed balance is not affected by this modification.

## Summary and Conclusions

In this chapter, I have presented a detailed discussion of cluster Monte Carlo algorithms for off-lattice systems. In order to emphasize the efficiency of these algorithms as well as their connection to methods developed earlier, I have first introduced the basic ingredients of a conventional (Metropolis-type) Monte Carlo algorithm. Subsequently, the Swendsen–Wang and Wolff algorithms for Ising and $q$-state Potts models have been discussed, which are striking because of their rejection-free character. The off-lattice cluster algorithms, which are based upon geometric symmetry operations, are a direct generalization of these lattice cluster methods, and illustrate that rejection-free algorithms are by no means rare exceptions, but can actually be phrased for large classes of systems.

## Acknowledgments

This work is supported by the National Science Foundation under CAREER Award No. DMR-0346914 and by the U.S. Department of Energy, Division of Materials Sciences under Award No. DEFG02-91ER45439, through the Frederick Seitz Materials Research Laboratory at the University of Illinois at Urbana-Champaign.

## References

1. N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller (1953) Equation of state calculations by fast computing machines. *J. Chem. Phys.* **21**, pp. 1087–1092
2. D. Frenkel and B. Smit (2002) *Understanding Molecular Simulation*. San Diego: Academic, 2nd ed.
3. R. H. Swendsen and J.-S. Wang (1987) Nonuniversal critical dynamics in Monte Carlo simulations. *Phys. Rev. Lett.* **58**(2), pp. 86–88
4. P. W. Kasteleyn and C. M. Fortuin (1969) Phase transitions in lattice systems with random local properties. *J. Phys. Soc. Jpn. Suppl.* **26s**, pp. 11–14
5. C. M. Fortuin and P. W. Kasteleyn (1972) On the random-cluster model. I. Introduction and relation to other models. *Physica* **57**, pp. 536–564
6. S.-K. Ma (1976) *Modern Theory of Critical Phenomena*. Redwood City, Calif.: Addison-Wesley
7. U. Wolff (1989) Collective Monte Carlo updating for spin systems. *Phys. Rev. Lett.* **62**(4), pp. 361–364
8. R. J. Baxter, S. B. Kelland, and F. Y. Wu (1976) Equivalence of the Potts model or Whitney polynomial with an ice-type model. *J. Phys. A* **9**, pp. 397–406
9. E. Luijten and H. W. J. Blöte (1995) Monte Carlo method for spin models with long-range interactions. *Int. J. Mod. Phys. C* **6**, pp. 359–370
10. E. Luijten, H. W. J. Blöte, and K. Binder (1996) Crossover scaling in two dimensions. *Phys. Rev. E* **54**, pp. 4626–4636
11. E. Luijten and H. W. J. Blöte (1997) Classical critical behavior of spin models with long-range interactions. *Phys. Rev. B* **56**, pp. 8945–8958
12. E. Luijten and H. Meßingfeld (2001) Criticality in one dimension with inverse square-law potentials. *Phys. Rev. Lett.* **86**, pp. 5305–5308
13. A. Hucht (2002) On the symmetry of universal finite-size scaling functions in anisotropic systems. *J. Phys. A* **35**, pp. L481–L487
14. C. Dress and W. Krauth (1995) Cluster algorithm for hard spheres and related systems. *J. Phys. A* **28**, pp. L597–L601
15. J. R. Heringa and H. W. J. Blöte (1996) The simple-cubic lattice gas with nearest-neighbour exclusion: Ising universality. *Physica A* **232**, pp. 369–374
16. J. Liu and E. Luijten (2005) Generalized geometric cluster algorithm for fluid simulation. *Phys. Rev. E* **71**, p. 066701
17. A. Buhot and W. Krauth (1998) Numerical solution of hard-core mixtures. *Phys. Rev. Lett.* **80**, pp. 3787–3790
18. L. Santen and W. Krauth (2000) Absence of thermodynamic phase transition in a model glass former. *Nature* **405**, pp. 550–551

19. J. Liu and E. Luijten (2004) Rejection-free geometric cluster algorithm for complex fluids. *Phys. Rev. Lett.* **92**(3), p. 035504
20. J. G. Malherbe and S. Amokrane (1999) Asymmetric mixture of hard particles with Yukawa attraction between unlike ones: a cluster algorithm simulation study. *Mol. Phys.* **97**, pp. 677–683
21. J. R. Heringa and H. W. J. Blöte (1998) Geometric cluster Monte Carlo simulation. *Phys. Rev. E* **57**(5), pp. 4976–4978
22. J. R. Heringa and H. W. J. Blöte (1998) Geometric symmetries and cluster simulations. *Physica A* **254**, pp. 156–163
23. A. Coniglio and W. Klein (1980) Clusters and Ising critical droplets: a renormalisation group approach. *J. Phys. A* **13**, pp. 2775–2780
24. M. E. Fisher (1967) The theory of condensation and the critical point. *Physics* **3**, pp. 255–283
25. M. P. Allen and D. J. Tildesley (1987) *Computer Simulation of Liquids.* Oxford: Clarendon
26. J. Liu and E. Luijten (2004) Stabilization of colloidal suspensions by means of highly charged nanoparticles. *Phys. Rev. Lett.* **93**, p. 247802
27. S. Asakura and F. Oosawa (1954) On interaction between two bodies immersed in a solution of macromolecules. *J. Chem. Phys.* **22**, pp. 1255–1256
28. K. Binder and E. Luijten (2001) Monte Carlo tests of renormalization-group predictions for critical phenomena in Ising models. *Phys. Rep.* **344**, pp. 179–253