The Monte Carlo Method in Science and Engineering

Since 1953, researchers have applied the Monte Carlo method to a wide range of areas. Specialized algorithms have also been developed to extend the method's applicability and efficiency. The author describes some of the algorithms that have been developed to perform Monte Carlo simulations in science and engineering.

Monte Carlo methods are algorithms for solving various kinds of computational problems by using random numbers (or rather, pseudorandom numbers). Monte Carlo simulations play an important role in computational science and engineering, with applications ranging from materials science to biology to quantum physics. They also play an important role in a variety of other fields, including computer imaging, architecture, and economics. Nicholas Metropolis suggested the name "Monte Carlo"—in reference to the famous casino in Monaco—in one of the first applications of the Monte Carlo method in physics.[1] Because of the repetitive nature of a typical Monte Carlo algorithm, as well as the large number of calculations involved, the Monte Carlo method is particularly suited to calculation using a computer.

Monte Carlo methods are particularly useful for problems that involve a large number of degrees of freedom. For example, deterministic methods of numerical integration operate by taking several evenly spaced samples from a function. While this might work well for functions of one variable, such methods can be very inefficient for functions of several variables. For example, to numerically integrate a function of an N-dimensional vector (where $N = 100$) with a grid of 10 points in each dimension would require the evaluation of 10^{100} points, which is far too many to be computed. Monte Carlo methods provide a way out of this exponential time increase: as long as the function is reasonably well behaved, it can be estimated by randomly selecting points in N-dimensional space and then taking an appropriate average of the function values at these points. By the central limit theorem, this method will display $1/\sqrt{N}$ convergence—that is, quadrupling the number of sampled points will halve the error, regardless of the number of dimensions. Another very important application for Monte Carlo simulations is optimization. The traveling salesman problem is an example of an optimization problem that is very difficult to solve using conventional methods, but that might be approximately solved via Monte Carlo methods. A variety of Monte Carlo methods such as stochastic tunneling,[2] simulated annealing,[3] genetic algorithms,[4] and parallel tempering[5] have been developed to handle such optimization problems.

One of the first uses of Monte Carlo simulations is described in the classic article by Nicholas C. Metropolis, Arianna W. Rosenbluth, Marshall N.

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

JACQUES G. AMAR
University of Toledo

Further Reading

Given the vast literature on Monte Carlo simulations, it is virtually impossible to discuss all the methods that have been developed in an article of this length. For fairly recent surveys of the literature, see the books on Monte Carlo methods by M.H. Kalos and P.A. Whitlock,[1] D.P. Landau and K. Binder,[2] and B.A. Berg[3] as well as a recent Los Alamos National Laboratory conference proceedings on the Monte Carlo method in the physical sciences.[4] A fairly recent discussion of extended ensemble methods is provided in a review article by Yukito Iba.[5] For a good description of classical Monte Carlo simulations of fluids, the book by Allen and Tildesley[6] is also recommended.

References

1. M.H. Kalos and P.A. Whitlock, *Monte Carlo Methods*, John Wiley & Sons, 1986.
2. D.P. Landau and K. Binder, *A Guide to Monte Carlo Simulations in Statistical Physics*, Cambridge Univ. Press, 2000.
3. B.A Berg, *Markov Chain Monte Carlo Simulations and Their Statistical Analysis*, World Scientific, 2004.
4. J.E. Gubernatis, ed., "The Monte Carlo Method in the Physical Sciences," *Am. Inst. of Physics Conf. Proc.*, no. 690, Am. Inst. of Physics, 2003.
5. Y. Iba, "Extended Ensemble Monte Carlo," *Int'l J. Modern Physics C*, vol. 12, no. 5, 2001, pp. 623–656.
6. M.P. Allen and D.J. Tildesley, *Computer Simulation of Liquids*, Oxford Univ. Press, 2001.

Rosenbluth, Augusta H. Teller, and Edward Teller.[1] In this work, the general Metropolis algorithm is first described along with its application to the equation of state of fluids. Using the Los Alamos National Laboratory's MANIAC computer (11,000 operations per second), Metropolis and his colleagues obtained results for the equation of state of the hard disk fluid by performing Monte Carlo simulations of 2D systems with 56 and 224 particles. Since then, the computational power of a single processor has increased by approximately a factor of 10^5, and the Monte Carlo algorithm has become increasingly sophisticated.

This article describes some of the algorithms that have been developed to perform both equilibrium and nonequilibrium Monte Carlo simulations of a variety of systems of interest in biology, physics, chemistry, materials science, and engineering.

Metropolis-Hastings Monte Carlo and Detailed Balance

As in the original article by Metropolis and his colleagues, in many scientific applications, researchers use Monte Carlo simulations to sample as accurately as possible the properties of a many-body system within a given statistical distribution or ensemble. An example is the Gibbs canonical ensemble with probability distribution $P_i = \exp(-E_i/k_BT)/Z$, where E_i is the energy of the system in configuration i, k_B is Boltzmann's constant, T is the temperature, and Z is the partition function. Starting with a given initial state i, two steps are usually involved in generating the next Monte Carlo state. First, a possible new state or trial configuration j is selected with a trial selection probability or rate T_{ij}. Then the new configuration is either accepted with probability P_{ij}^{acc}, and the system makes a transition from state i to state j, or it is rejected with probability $1 - P_{ij}^{acc}$. The overall transition rate from state i to state j is thus given by the transition matrix $w_{ij} = T_{ij}P_{ij}^{acc}$.

The sequence of configurations generated in a Monte Carlo simulation is generally referred to as a Markov chain because the transition rate or probability depends on the current state but not on previous states. To generate a Markov chain of states with the desired probability distribution P_i, the overall transition probabilities w_{ij} should satisfy the detailed balance condition

$$w_{ij}P_i = w_{ji}P_j, \tag{1}$$

which implies that the desired distribution P_i is a stationary state. Assuming ergodicity—that is, a nonzero multitransition probability of reaching any allowed state of the system from any other allowed state—this condition further implies that, in such a Monte Carlo simulation, the system will approach the equilibrium ensemble distribution. Formally, the ergodicity requirement also implies that the transition matrix w must satisfy

$$[w^n]_{ij} > 0 \tag{2}$$

for $n > n_{\max}$ for all i,j. Although the ergodicity properties of a particular many-body system are difficult to study, it is believed, in general, that almost any reasonable choice of allowed trial moves will satisfy ergodicity.

Several possible forms for the acceptance probabilities P_{ij}^{acc} satisfy the detailed balance condition in Equation 1. The simplest and most commonly used corresponds to the Metropolis-Hastings rule,[6]

$$P_{ij}^{acc} = \min\left(1, \frac{T_{ji}P_j}{T_{ij}P_i}\right). \quad (3)$$

Another option is the "symmetric" Barker expression,[7]

$$P_{ij}^{acc} = \frac{P_j T_{ji}}{P_i T_{ij} + P_j T_{ji}}. \quad (4)$$

Many Monte Carlo simulations, including the ones carried out in the original article by Metropolis and his colleagues,[1] use symmetric trial configuration selection rates $T_{ij} = T_{ji}$. Thus, in a typical Metropolis Monte Carlo simulation of the canonical ensemble with equilibrium distribution $P_i \sim \exp(-E_i/k_BT)$, the acceptance probability P_{ij}^{acc} for a transition from state i to state j can be written as

$$P_{ij}^{acc} = \min(1, \exp(-\beta(E_j - E_i)), \quad (5)$$

where $\beta = 1/k_BT$. An example is the Ising spin-model with "spin-flip" or Glauber dynamics: at each step, a spin is randomly selected from the lattice and then flipped with an appropriate acceptance probability. In Monte Carlo simulations of fluids with velocity-independent interactions,[1] the velocity (momentum) degrees of freedom can be integrated out; only the atomic positions are important. Thus, the energy E_i in Equation 5 can be taken to include only the configuration-dependent portion of the energy. For the hard-disk system studied in the original Metropolis article,[1] the Monte Carlo method is particularly simple. All trial moves involving an overlap are immediately rejected, and all trial moves not involving an overlap are accepted. In this case, a simple way of generating trial configurations is to randomly select a hard disk and then displace it randomly within some radius δ.

Although Monte Carlo simulations can be used to efficiently sample an equilibrium distribution, nonequilibrium or "dynamic" Monte Carlo simulations are also of interest. In these simulations, the trial configuration selection rates T_{ij} are usually assumed to correspond to a fixed attempt rate τ^{-1}, which is the same for all possible allowed transitions. We can obtain the time t for a given transition by noting that the survival probability that the system remains in state i at time t after arrival is given by $P_i^{surv}(t) = e^{-t/\tau}$. The probability distribution $P_i^{tr}(t)$ that the system undergoes a transition from state i at time t is then

$$P_i^{tr}(t) = -\frac{dP_i^{surv}(t)}{dt} = \tau^{-1}e^{-t/\tau}, \quad (6)$$

which implies that the average transition time is given by $\langle t \rangle = \tau$. Such a distribution of transition times t can be generated using the expression

$$t = -\tau \ln(r), \quad (7)$$

where r is a uniform random number between 0 and 1.

Constant-NPT Ensemble

While Equation 5 may be used to perform Metropolis Monte Carlo simulations in the Gibbs canonical ensemble, researchers have extended the Monte Carlo method to a variety of other ensembles. For example, in classical simulations of molecular liquids and gases in the constant NPT ensemble (where N is the number of particles, P is the pressure, and T is the temperature), the configurational average of a quantity $\mathcal{A}$ can be rewritten as[8]

$$\langle A \rangle_{NPT} = Z_{NPT}^{-1} \int_0^\infty dV \exp(-\beta PV) V^N \int_0^1 d^{3N} s A(\mathbf{s}) \exp(-\beta U(\mathbf{s})), \quad (8)$$

where Z_{NPT} is the corresponding partition function, $V = L^3$ is the volume of the system, and $U(\mathbf{s})$ corresponds to the system's configuration-dependent total potential energy. Here, we use a set of scaled coordinates $\mathbf{s} = \mathbf{s}_1, \mathbf{s}_2, \ldots \mathbf{s}_N$, where $\mathbf{s_i} = L^{-1}\mathbf{r_i}$ and $\mathbf{r_i}$ is the coordinate of particle i. Accordingly, the corresponding equilibrium distribution is given by

$$P_i \sim \exp(-\beta H), \quad (9)$$

where $H = PV + U(\mathbf{s}) - \beta^{-1}N\ln(V)$. A trial configuration is generated by randomly displacing a randomly selected molecule (molecule k) or making a volume change from V_i to V_j:

$$\mathbf{s}_k^j = \mathbf{s}_k^i + \delta s_{\max}(2\eta - \mathbf{1})$$

$$V_j = V_i + \delta V_{\max}(2\xi - 1), \quad (10)$$

where ξ is a uniform random number between 0 and 1, η is a 3D vector whose components are also uniform random numbers between 0 and 1, and $\mathbf{1}$ is the vector (1, 1, 1). The quantities $\delta s_{\max}$ and

δV_{max} govern the maximum changes in the scaled coordinates of the particles and in the volume of the simulation box, respectively, and are typically adjusted[8] to produce a Metropolis move acceptance ratio of 35 to 50 percent.[9] Once the new state j is selected, the quantity δH is calculated as

$$\delta H_{ij} = P(V_j - V_i) + \delta U_{ij} - N\beta^{-1}\ln(V_j/V_i), \quad (11)$$

and the transition from state i to state j is accepted with transition probability

$$P_{ij}^{acc} = \min(1, \exp(-\beta\delta H_{ij})). \quad (12)$$

Grand Canonical Monte Carlo

In Monte Carlo simulations of phase transitions and phase equilibria, the grand canonical ensemble corresponding to constant chemical potential μ is particularly useful.[10–13] As an example, in Monte Carlo simulations in the constant-(μ, V, T) ensemble, the energy can fluctuate due to particle displacements. However, fluctuations in the particle number and energy can also occur via particle insertions and deletions, which are selected with equal probability. If deletion is chosen, a trial configuration j in which one of the particles is randomly removed is generated. In this case, the acceptance probability for the new configuration can be written as[11]

$$P_{ij}^{N\to N-1} = \min\left(1, \frac{N\Lambda^3}{zV}\exp(-\beta(E_j - E_i)\right), \quad (13)$$

where $z = \exp(\beta\mu)$, $\Lambda = h/\sqrt{2\pi m k_B T}$ is the thermal wavelength, and N is the number of particles in the system before deletion. If insertion is chosen, then a trial configuration j in which an additional particle is inserted at a randomly chosen location is generated. In this case, the acceptance probability for the new configuration can be written as

$$P_{ij}^{N\to N+1} = \min\left(1, \frac{zV}{(N+1)\Lambda^3}\exp(-\beta(E_j - E_i)\right). \quad (14)$$

Acceleration Methods and Extended Ensembles

Since the development of the Metropolis algorithm, a variety of Monte Carlo acceleration methods have emerged. Some of these methods involve altering or biasing the trial configurations and selection rates T_{ij} to make them more efficient, whereas others involve performing simulations in different ensembles. We now review some of these methods.

n-Fold Way Algorithm

In some cases, such as at low temperatures when the acceptance probability is low and most transitions are rejected, the standard Metropolis algorithm can become inefficient. To eliminate rejection in discrete Monte Carlo simulations, A.B. Bortz, Mal H. Kalos, and Joel L. Lebowitz developed the rejection-free n-fold way algorithm.[14] The basic idea is to calculate all the possible transition rates $w_{ij} = T_{ij}P_{ij}^{acc}$ for all possible trial configurations j (with $j \neq i$) for a given initial configuration i, and then directly select the new configuration j with a probability proportional to w_{ij}. Once the new configuration is selected, it is always accepted. The penalty for eliminating rejection is the additional overhead for determining all possible transition states and rates, as well as the memory required to keep track of all the transitions.

If N possible new states exist, then the new configuration j can be selected by first calculating the partial sums $S_0^i = 0$ and $S_n^i = \Sigma_{k=1}^n w_{ik}$ for $n = 1$ to N and then generating a uniform random number r between 0 and S_N^i. A search can then be performed to find the value of j such that $S_{j-1}^i < r < S_j^i$, after which a transition to state j is carried out. Such a search can either be performed directly by going through the list of partial sums or more efficiently by using a binary search. However, in many cases, a relatively small number N_c of possible values of the transition probabilities or rates w_α exist in which each value of α corresponds to a different transition class. The main work then involves determining the number n_α^i of possible transitions from state i for each class, and then selecting the type or "class" β of transition with probability

$$\rho_\beta^i = n_\beta^i w_\beta \Big/ \sum_{\alpha=1}^{N_c} n_\alpha^i w_\alpha \ . \quad (15)$$

Once a particular class β is selected, then one of the n_β^i possible transitions in that class is randomly selected from a list of all transitions in that class.

Since in a typical dynamical Metropolis Monte Carlo simulation, the trial configuration selection rate T_{ij} is a constant $1/\tau$ for all possible transitions, for the corresponding n-fold way simulation, we can write $w_\alpha = \tau^{-1}P_\alpha^{acc}$, where P_α^{acc} is the acceptance probability for class α. The average time for a transition then depends on the initial configura-

tion i and is given by $\langle \Delta t_i \rangle = \tau / \Sigma_\alpha n_\alpha^i P_\alpha^{acc}$, while the time for a particular transition is given by $\Delta t_i = -\ln(r)\langle \Delta t_i \rangle$, where r is a uniform random number between 0 and 1. Thus, if the acceptance probabilities P_α^{acc} are low, then the time interval per step for an n-fold way simulation will be much larger than for a standard Metropolis simulation. Although originally developed for Ising spin systems, the n-fold way algorithm has recently been extended to the simulation of continuum systems.[15] However, in this case, the overhead associated with determining the next move is significantly larger than in the discrete case.

Cluster Acceleration

In many cases, the use of Monte Carlo trial moves that correspond to relatively small local changes can be relatively efficient. However, when large-scale fluctuations become important, such as at a critical point, such localized moves become relatively inefficient. On the other hand, the use of arbitrary large moves (such as increasing the maximum displacement in a fluid or moving a large number of particles randomly) can lead to a significant decrease in the acceptance probability. One approach to overcoming these problems has been the development of cluster acceleration methods such as the Sweeny[16] and Swendsen-Wang (SW)[17,18] algorithms. The SW algorithm is based on a mapping of the Ising model to a percolation model by C.M. Fortuin and P.W. Kasteleyn[19] and was originally developed to accelerate the Monte Carlo simulation of Ising and Potts spin models near the critical point. Consider a q-state Potts model Hamiltonian,

$$H = -\mathcal{J} \sum_{i,j} \delta_{s_i, s_j}, \tag{16}$$

where the Potts spins s_i are on a lattice and take on the integer values 1, 2, ... q, and the sum is over all nearest-neighbor spins. The Kronecker delta in Equation 16 corresponds to ferromagnetic coupling, that is, the energy is minimized when nearest-neighbor spins have the same value. In the SW algorithm, bonds are created between all neighboring spins with the same value with probability $p = 1 - \exp(-\mathcal{J}/k_B T)$, thus leading to a set of bond clusters. (An isolated spin with no bonds is also considered to be a single cluster.) In a single Monte Carlo move, the bond clusters are then all "flipped"—that is, for each bond cluster, a new randomly chosen Potts value is selected and assigned to all the spins in that bond cluster. Because the clusters can be arbitrarily large at the critical temperature in this algorithm, the new configuration can differ substantially from the original one. In Monte Carlo simulations of the 2D Ising model performed with this algorithm,[17,18] researchers found that critical slowing down was significantly reduced.

Ulli Wolff[20] has developed a somewhat different cluster algorithm to study continuous spin models. In the Wolff algorithm, a single cluster is built at each step and then flipped using a generalized spin-flip operation. In simulations of the 2D Ising model and continuous spin $O(n)$ models with $n = 2$ (xy model) and $n = 3$ (Heisenberg model) using the Wolff algorithm, the critical slowing down was found to be further reduced. A variety of extensions and modifications of the SW and Wolff algorithms have since been developed, including cluster algorithms for vertex models[21] as well as acceleration algorithms for quantum spin systems.[22]

In many scientific applications, researchers use Monte Carlo simulations to sample as accurately as possible the properties of a many-body system within a given statistical distribution or ensemble.

More recently, researchers have extended cluster acceleration methods to the simulation of continuous systems such as complex fluids.[23–25] In particular, Jiuwen Liu and Erik Luijten[24,25] have developed an elegant generalization of the SW and Wolff algorithms for the simulation of fluids with a pair-potential based on the generalized geometric cluster algorithm developed by C. Dress and W. Krauth.[23] In Liu and Luijten's algorithm, the cluster selection and flipping processes are combined. At the beginning of each Monte Carlo step, a randomly chosen particle i at initial position $\mathbf{r}_i$ is first inverted ("pivoted") about a randomly selected pivot point to a new position $\mathbf{r}_i'$. Using the same pivot point, subsequent particles j are then added to the cluster with probability

$$p_{ij} = \max(1 - \exp(-\beta(V(|\mathbf{r}_i' - \mathbf{r}_j|) - V(|\mathbf{r}_i - \mathbf{r}_j|)), 0), \tag{17}$$

where $V(r)$ is the pair-potential describing the molecular interaction. If accepted as part of the cluster, the particle j is also inverted about the pivot point, so that all the particles belonging to a

cluster maintain their original separation. The process is iterative—that is, once a particle has been added to the cluster, it then also plays the role of particle i in Equation 17 and can recruit new particles to the cluster. The process ends when no more particles can be added to the cluster. As in the discrete cluster algorithms, there is no rejection because at least one particle is always moved. For Monte Carlo simulations of binary mixtures and complex fluids, the simulation efficiency can be orders of magnitude higher using a geometric cluster Monte Carlo than is found in the usual Metropolis Monte Carlo.

Multicanonical Methods

Several interesting methods have been developed in which the ensemble being simulated differs from the actual ensemble for which results are desired. The advantage of these methods, which include umbrella sampling[26] as well as the more recently developed multicanonical method,[27] is that the use of a different ensemble can lead to a better sampling of phase space and provide more information, such as the entropy and partition function. A somewhat related method is the histogram method of Alan Ferrenberg and Robert Swendsen.[28] In this method, information about the density of states $g(E)$ obtained during canonical ensemble Monte Carlo simulations at temperature T_1 is used via "re-weighting" to calculate properties of the system at a nearby temperature T_2.

The Wang-Landau multicanonical algorithm is particularly useful near phase transitions or for disordered systems, and has been applied to a variety of classical and quantum systems.

Recently, Fugao Wang and David Landau have developed a very interesting multicanonical Monte Carlo algorithm.[29,30] The idea of the Wang-Landau algorithm is to sample the energy space uniformly—"generate a flat histogram in energy space" to determine the density of states $g(E)$. Once the density of states is determined, all thermodynamic quantities can be calculated for arbitrary temperatures using the Boltzmann distribution. Typically, trial configurations are randomly chosen, as in the standard Metropolis algorithm, so that the trial configuration selection matrix T_{ij} is symmetric. The acceptance probability for a move from state i to state j is then

$$P_{ij}^{acc} = \min\left(1, \frac{g(E_i)}{g(E_j)}\right), \tag{18}$$

which implies that the probability of configuration i is given by $1/g(E_i)$. Because the density of states is $g(E_i)$, this eventually leads to a flat histogram in energy space—that is, a "random walk" in energy space. Initially, because the density of states isn't known, we have $g(E) = 1$. However, each time an energy level E is visited, the corresponding density of states $g(E)$ is updated by multiplying the existing value by a modification factor $f > 1$, which can be reduced slowly during the course of the simulation to a value just slightly higher than 1. The simulation process stops when the modification factor is smaller than some predefined final value, for example, $f_{final} = 1 + \varepsilon$, where $\varepsilon << 1$. Once this occurs, the density of states $g(E)$ should be converged to the correct value and can be used to calculate thermodynamic quantities such as the free energy,

$$\begin{aligned} F(T) &= -k_B T \ln(Z) \\ &= -k_B T \ln\left(\sum_E g(E) e^{-\beta E}\right). \end{aligned} \tag{19}$$

To accelerate the calculation of the density of states, one can perform multiple simulations with each for a different range of energy and then piece together the results. In each simulation, the random walk in energy space is kept in the selected energy range by rejecting any move out of that range. The Wang-Landau multicanonical algorithm is particularly useful near phase transitions or for disordered systems, and has been applied to a variety of classical and quantum systems.

Parallel Tempering

One method that has proven to be extremely useful in Monte Carlo simulations of complex systems with a rugged energy landscape is parallel tempering. In this method, several independent replicas of a system are simulated simultaneously, but under different conditions (typically different temperatures). Periodically, systems with neighboring temperatures are allowed to interchange configurations. For example, in the canonical ensemble, the probability for a replica at temperature T_1 to accept a trial move from configuration i to configuration j is given by

$$P_{ij}^{acc}(T_1) = \min\left(1, \exp(-\beta_1(E_j - E_i)\right). \quad (20)$$

Similarly, the probability for a replica at temperature T_2 to accept a trial move from configuration j to configuration i is given by

$$P_{ij}^{acc}(T_2) = \min\left(1, \exp(-\beta_2(E_j - E_i)\right). \quad (21)$$

The probability of accepting an interchange of configurations between the two replicas corresponding to a "swap" move is then given by the product of $P_{ij}^{acc}(T_1)$ and $P_{ji}^{acc}(T_2)$, that is,

$$P_{ij}^{swap} = \min\left(1, \exp(-(\beta_1 - \beta_2)(E_j - E_i)\right). \quad (22)$$

Typically, a swap move is attempted between two replicas after both have completed the same number of Monte Carlo steps.

The swap moves used in parallel tempering can significantly improve the sampling of configuration space. For example, replicas of a system that are close to a glassy state can exchange their way "up" in temperature so that energy barriers are easier to overcome, and then come back down to low temperatures to yield a new independent uncorrelated configuration. This is particularly useful in performing simulations at low temperatures or when searching for ground states.[31] Recent research has combined the parallel tempering method with multicanonical Monte Carlo, which was found to significantly improve the efficiency in simulations of a Lennard-Jones fluid's coexistence curve.[32]

Quantum Monte Carlo

The correlated motion of electrons plays a crucial role in the aggregation of atoms into molecules and solids, in electronic transport properties, and in many other important physical phenomena. Although ab initio calculations performed with density functional theory[33] have become a vital tool in materials science, condensed-matter physics, and quantum chemistry, quantum Monte Carlo (QMC) calculations are an important complementary alternative. For example, QMC calculations have been used to accurately calculate the exchange-correlation functional for the homogeneous electron gas as a function of density.[34] These results have been parameterized[35] and then used as input to density functional theory calculations in the local density approximation.[35]

A variety of QMC methods have been developed, including the variational Monte Carlo, diffusion Monte Carlo, and path-integral Monte Carlo methods. The variational QMC method was first developed for boson systems by W.L. McMillan,[36] and then generalized to fermions by David Ceperley and colleagues.[37] The diffusion or Green's function Monte Carlo method[33,38] takes advantage of the mapping between the imaginary-time Schrödinger equation and a diffusion process that includes drift due to a trial wavefunction quantum force, and branching (a random walker in configuration space is either duplicated or eliminated) due to the difference between the local trial energy and a reference energy. To take into account the antisymmetry of the wavefunction for fermions, the fixed-node approximation, in which particles are absorbed at the nodes can be used. Path-integral quantum Monte Carlo[39] is based on the Feynman path-integral interpretation of quantum mechanics.[40] Each quantum particle is represented by a "ring polymer" whose elements (beads) are connected by harmonic forces (springs). Interparticle interactions are expressed as interactions between simultaneous beads of the polymers, thus the partition function of a quantum many-body system is reduced to the distribution function of interacting classical ring polymers with harmonic bonds. As for the Green's function Monte Carlo method, the main challenge for fermionic systems comes from the necessity of antisymmetrization.

Quantum Monte Carlo (QMC) calculations are an important alternative to ab initio density functional theory calculations.

Kinetic Monte Carlo

The techniques discussed so far primarily involve equilibrium Monte Carlo, in which the goal is to sample the equilibrium distribution or ensemble as efficiently as possible. However, several important nonequilibrium processes exist in physics, chemistry, and materials science in which the time evolution is of interest. Although molecular dynamics can be used to accurately model the evolution of such systems on the atomic scale, the time scales that can be reached are extremely limited. Accordingly, researchers have used kinetic Monte Carlo (KMC) simulations in many cases. In particular, KMC simulations have been used to efficiently model a wide variety of dynamical processes. A recent review article discussing KMC simulations appears elsewhere.[41]

In KMC simulations, we assume that the transi-

tions from one state to another state are Markovian—that is, the transition rates w_{ij} depend only on the initial state i and the final state j—and that each transition corresponds to a Poisson process (the transition rates w_{ij} are independent of time).[42] This is a good approximation for systems in which the transitions involve infrequent thermally activated events between states separated by relatively large energy barriers. According to transition state theory,[43] the rates w_{ij} are then determined by an activation barrier E_{ij}^b, as well as a prefactor D_{ij}:

$$w_{ij} = D_{ij}\, exp(-E_{ij}^b/k_BT). \quad (23)$$

The prefactors can be calculated more accurately by using transition state theory,[43] but are often set to a typical vibrational frequency. To converge to equilibrium, the detailed balance condition

$$w_{ij}e^{-E_{ij}^b/k_BT} = w_{ji}e^{-E_{ji}^b/k_BT} \quad (24)$$

must be satisfied. Typically, we assume that the energy barrier for a transition from configuration i to configuration j can be written as $E_{ij}^b = E_{ij}^a - E_i$, where E_{ij}^a is the energy of the activated state separating configurations i and j, and that $D_{ij} = D_{ji}$. These assumptions automatically satisfy the detailed balance condition in Equation 24.

Starting from an initial configuration i, the procedure is then similar to that for the n-fold way—the next configuration is chosen with a probability proportional to the transition rate w_{ij}. In particular, the partial sums $S_0^i = 0$ and $S_n^i = \Sigma_{k=1}^n w_{ik}$ are generated for all N possible final configurations j. After generating a uniform random number r between 0 and S_N^i, a search is performed to find the value of j such that $S_{j-1}^i < r < S_j^i$ and a transition to that state is then carried out. If there are several different transition rates w_{ij} for a given initial state i, the search through the list of partial sums can be performed efficiently by using a binary- or K-tree algorithm.[44] However, just as for the n-fold way, in many cases there exist only a small number N_c of distinct transition classes α (with corresponding rates w_α and n_α^i different possible transitions in each class). In this case, the next configuration can be selected more efficiently by first selecting the class of the transition with a probability proportional to $n_\alpha^i w_\alpha$ and then randomly selecting one of the n_α^i possible transitions in the selected class. The time t for the transition is then given by

$$t = -\ln(r)/R_i, \quad (25)$$

where $R_i = \Sigma_{j\neq i} w_{ij} = \Sigma_\alpha n_\alpha^i w_\alpha$ is the total rate for a transition from state i, and r is a uniform random number between 0 and 1.

Many KMC simulations use a precalculated rate catalog that includes a limited number of types of transitions, such as monomer diffusion, and their corresponding rates. However, a new "self-learning" KMC method has recently been developed.[45,46] In this method, a precalculated list of diffusion processes and their associated energetics and transition rates isn't needed—rather, at any time during the simulation, the activation energies for all possible single or multi-atom processes are either computed on-the-fly using a saddle-point search procedure, or retrieved from a database in which previously encountered processes are stored. Initially, this method may require an extensive amount of work to calculate the possible events that might occur in the system. However, as long as the interactions are short-ranged, updating the list of possible events and their rates after each event only requires a local calculation. In addition, this method can lead to a substantial gain in accuracy because of the inclusion of many-particle processes. Recently, self-learning KMC has been used to study the diffusion of Cu clusters on the Cu(111) surface,[45,46] for which there are many possible types of events and energy barriers, and significant differences between the results obtained using self-learning KMC and ordinary KMC with a restricted rate catalog were observed.

Parallel Monte Carlo

Although independent or quasi-independent parallel simulations can be used in equilibrium Monte Carlo to increase statistics and accelerate simulation speed, the standard dynamic Monte Carlo algorithm is inherently serial because only one event can occur at each step. However, in some cases it is desirable to perform Monte Carlo simulations over very long times or for very large system sizes. In such cases, rigorous parallel dynamical Monte Carlo simulations—in which the system is decomposed into different parts that are then assigned to different processors via domain decomposition—are of interest.

Parallel Metropolis Monte Carlo

In Metropolis Monte Carlo, the attempt times are typically independent of system configuration (see Equation 7), so parallel simulations can be carried out using an asynchronous "conservative" algorithm.[47–50] In this algorithm, each processor keeps track of the times of the events performed in its domain, but must check the next attempt times of the appropriate neighboring processors when performing a boundary move. Thus, although Monte Carlo

events that occur away from the processor boundary (that is, whose acceptance probabilities aren't affected by another processor's configuration) are allowed to proceed independently, boundary moves can only be accepted or rejected if the "local time condition" that the processor's current attempt time is less than the corresponding neighboring processors' next attempt time is satisfied. This leads to an asynchronous dynamics in which all processors or domains have different local times at any stage of the simulation, but communication is required whenever a boundary move is attempted. Even in the extreme case in which all moves are boundary moves, such as in the spin-flip Ising model,[49] such an algorithm doesn't lead to deadlock because at any point in time the processor whose local time is a minimum can always proceed. For efficiency, a modified version of this algorithm can also be used, in which *n*-fold way simulations are performed in the interior of each processor, while Metropolis simulations are performed at the boundary.[48,51]

Recently, it has been shown[49] that the conservative asynchronous (CA) algorithm scales (that is, the overall simulation speed in Monte Carlo steps per second is proportional to the number of processors) because a finite fraction of all processors can proceed at any given time. However, the CA algorithm also leads to a surface or "time horizon" (corresponding to the next attempt times of all processors) that roughens with time and whose evolution can be described[49] by the Kardar-Parisi-Zhang (KPZ) equation.[52] The roughening of the time horizon can lead to problems with data-taking and memory storage because data can only be taken using configurations in different processors that correspond to the same local time.[49] For models in which a processor's "boundary moves" can affect not only the acceptance probabilities of a neighboring processor, but also that processor's boundary configuration, this can lead to more subtle problems.[53] However, research has recently shown[50] that by enhancing the local time restriction to include a small fraction of randomly chosen processors that are arbitrarily far away, thus creating a small-world network, the time horizon's roughness can be significantly reduced without sacrificing computational efficiency. As a result, the memory storage requirements associated with data-taking can be significantly reduced.

Parallel Kinetic Monte Carlo

While the CA algorithm is appropriate for parallel Metropolis simulations, it cannot be directly applied to parallel kinetic Monte Carlo simulations, since in KMC the event time depends on the system configuration. In particular, because fast events can "propagate" across processors, the time for an event already executed by a processor can change due to earlier events in nearby processors, thus leading to an incorrect evolution. If all possible KMC moves and event rates are known in advance, the CA algorithm can still be used by mapping the event rates to Metropolis probabilities.[53] However, for problems with a wide range of different possible event rates, the resulting parallel simulations are typically very inefficient.[53]

A more efficient approach to parallel KMC simulations is the synchronous relaxation (SR) algorithm.[54,55] S.G. Eick and colleagues[54] originally used this algorithm to simulate large circuit-switched communication networks, and it has also been discussed recently by Boris Lubachevsky and Alan Weiss[55] in the context of Ising model simulations. In this approach, all processors remain synchronized at the beginning and end of a time interval T, and an iterative relaxation method is used to correct errors due to boundary events between processors. The SR algorithm has the advantage of generality (for example, it isn't necessary to know the types or rates of all possible events in advance) and flexibility because the cycle length T can be dynamically tuned[56] to optimize parallel efficiency. However, due to fluctuations, which increase logarithmically[56] with the number of processors N_p, as well as the requirement of global communications at the end of each cycle (the global communications time also increases logarithmically with N_p), the computational speedup as a function of N_p is sublinear for a fixed processor size.

While the conservative asynchronous algorithm is appropriate for parallel Metropolis simulations, it cannot be directly applied to parallel kinetic Monte Carlo simulations, since in KMC the event time depends on the system configuration.

Recently, a more efficient but semirigorous synchronous sublattice (SL) algorithm has been developed.[57] The SL algorithm is useful for a variety of parallel KMC simulations and has recently been applied to simulations of epitaxial Cu/Cu(100) growth.[58] Because the SL algorithm requires only local communications, the parallel efficiency is independent of the number of processors in the large

processor limit, thus leading to linear scaling. In addition, research has shown[57] that by using certain reasonable assumptions on the cycle length and processor size, the results obtained are identical to those obtained in serial simulations. Nevertheless, the development of rigorous efficient parallel algorithms for KMC simulations remains a challenging problem.

Since the original article by Metropolis and his colleagues[1] first appeared in 1953, researchers have applied the Monte Carlo method to a wide range of nonequilibrium and equilibrium processes as well as to a variety of complex problems. Moreover, various specialized algorithms have been developed to extend its applicability and make it increasingly sophisticated and efficient. Coupled with the rapid increase in computer power, this evolution has greatly expanded the complexity and size of systems to which the Monte Carlo method can be applied. Although the Monte Carlo method's effectiveness in solving complex problems is due in part to the central-limit theorem, the development of improved algorithms and sampling methods will continue to play an important role in expanding the accuracy and applicability of Monte Carlo simulations in the sciences and engineering. CiSE

Acknowledgments

This work was supported by the US National Science Foundation through grant numbers DMR-0219328 and CCF-0428826, as well as grants of computer time from the Ohio Supercomputer Center (grant number PJS0245) and the Pittsburgh Supercomputer Center.

References

1. N.C. Metropolis et al., "Equation of State Calculations by Fast Computing Machines," *J. Chemical Physics*, vol. 21, 1953, pp. 1087–1092.
2. W. Wenzel and K. Hamacher, "Stochastic Tunneling Approach for Global Minimization of Complex Potential Energy Landscapes," *Physical Rev. Letters*, vol. 82, no. 15, 1999, pp. 3003–3007.
3. S. Kirkpatrick, C.D. Gelatt, and M.P. Vecchi, "Optimization by Simulated Annealing," *Science*, vol. 220, no. 4598, 1983, pp. 671–680.
4. J.R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, MIT Press, 1992.
5. E. Marinari and G. Parisi, "Simulated Tempering: A New Monte Carlo Scheme," *Europhysics Letters*, vol. 19, 1992, pp. 451–458.
6. W.K. Hastings, "Monte Carlo Sampling Methods Using Markov Chains and Their Applications," *Biometrika*, vol. 57, no. 1, 1970, pp. 97–109.
7. A.A. Barker, "Monte Carlo Calculations of Radial Distribution Functions for a Proton-Electron Plasma," *Australian J. Physics*, vol. 18, no. 2, 1965, pp. 119–128.
8. I.R. McDonald, "NpT-Ensemble Monte Carlo Calculations for Binary Liquid Mixtures," *Molecular Physics*, vol. 23, 1972, pp. 41–58.
9. M.P. Allen and D.J. Tildesley, *Computer Simulation of Liquids*, Oxford Univ. Press, 2001.
10. G.E. Norman and V.S. Filinov, "Investigation of Phase Transitions by a Monte Carlo Method," *High Temperature* (USSR), vol. 7, 1969, pp. 216–222.
11. N.B. Wilding, "Computer Simulation of Fluid Phase Transitions," *Am. J. Physics*, vol. 69, no. 11, 2001, pp. 1147–1155.
12. D. Nicholson and N. Parsonage, *Computer Simulation and the Statistical Mechanics of Adsorption*, Academic Press, 1982.
13. A.Z. Panagiotopoulos et al., "Phase Equilibria by Simulation in the Gibbs Ensemble, Alternative Derivation, Generalisation and Application to Mixture and Membrane Equilibria," *Molecular Physics*, vol. 63, no. 4, 1988, pp. 527–538.
14. A.B. Bortz, M.H. Kalos, and J.L. Lebowitz, "A New Algorithm for Monte Carlo Simulation of Ising Spin Systems," *J. Computational Physics*, vol. 17, 1975, pp. 10–18.
15. J.D. Munoz, M.A. Novotny, and S.J. Mitchell, "Rejection-Free Monte Carlo Algorithms for Models with Continuous Degrees of Freedom," *Physical Rev. E*, vol. 67, no. 2, 2003, pp. 026101-1–026101-4.
16. M. Sweeny, "Monte Carlo Study of Weighted Percolation Clusters Relevant to the Potts Models," *Physical Rev. B*, vol. 27, no. 7, 1983, pp. 4445–4455.
17. 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.
18. J.-S. Wang, O. Kozan, and R.H. Swendsen, "Sweeny and Gliozzi Dynamics for Simulations of Potts Models in the Fortuin-Kasteleyn Representation," *Physical Rev. E*, vol. 66, 2002, pp. 057101-1–057101-4.
19. 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.
20. U. Wolff, "Collective Monte Carlo Updating for Spin Systems," *Physical Rev. Letters*, vol. 62, no. 4, 1989, pp. 361–364.
21. H.G. Evertz, G. Lana, and M. Marcu, "Cluster Algorithm for Vertex Models," *Physical Rev. Letters*, vol. 70, no. 7, 1993, pp. 875–878.
22. M. Troyer, S. Wessel, and F. Alet, "Flat Histogram Methods for Quantum Systems: Algorithms to Overcome Tunneling Problems and Calculate the Free Energy," *Physical Rev. Letters*, vol. 90, 2003, pp. 120201-1–120201-4.
23. C. Dress and W. Krauth, "Cluster Algorithm for Hard Spheres and Related Systems," *J. Physics A: Mathematics and General*, vol. 28, 1995, pp. L597–L601.
24. J. Liu and R. Luijten, "Rejection-Free Geometric Cluster Algorithm for Complex Fluids," *Physical Rev. Letters*, vol. 92, no. 3, 2004, pp. 035504-1–035504-4.
25. J. Liu and R. Luijten, "Generalized Geometric Cluster Algorithm for Fluid Simulation," *Physical Rev. E*, vol. 71, no. 12, 2004, pp. 066701-1–066701-12.
26. G.M. Torrie and J.P. Valleau, "Nonphysical Sampling Distributions in Monte Carlo Free-Energy Estimation: Umbrella Sampling," *J. Computational Physics*, vol. 23, no. 2, 1977, pp. 187–199.
27. G.R. Smith and A.D. Bruce, "A Study of the Multi-Canonical Monte Carlo Method," *J. Physics A: Mathematical and General*, vol. 28, 1995, pp. 6623–6643.
28. A.M. Ferrenberg and R.H. Swendsen, "New Monte Carlo Technique for Studying Phase Transitions," *Physical Rev. Letters*, vol. 61, no. 23, 1988, pp. 2635–2638.
29. F. Wang and D.P. Landau, "Efficient, Multiple-Range Random Walk Algorithm to Calculate the Density of States," *Physical Rev. Letters*, vol. 86, no. 10, 2001, pp. 2050–2053.
30. F. Wang and D.P. Landau, "Determining the Density of States for Classical Statistical Models: A Random Walk Algorithm to

Produce a Flat Histogram," *Physical Rev. E*, vol. 64, no. 5, 2001, pp. 056101-1–056101-16.

31. F.C. Chuang et al., "Structure of Si(114) Determined by Global Optimization Methods," *Surface Science*, vol. 578, 2005, pp. 183–195.
32. R. Faller, Q. Yan, and J.J. de Pablo, "Multicanonical Parallel Tempering," *J. Chemical Physics*, vol. 116, no. 13, 2002, pp. 5419–5423.
33. W. Kohn and L.J. Sham, "Self-Consistent Equations Including Exchange and Correlation Effects," *Physical Rev. A*, vol. 140, no. 4A, 1965, pp. A1133–A1138.
34. D.M. Ceperley and B.J. Alder, "Ground State of the Electron Gas by a Stochastic Method," *Physical Rev. Letters*, vol. 45, no. 7, 1980, pp. 566–569.
35. J.P. Perdew and A. Zunger, "Self-Interaction Correction to Density-Functional Approximations for Many-Electron Systems," *Physical Rev. B*, vol. 23, no. 10, 1981, pp. 5048–5079.
36. W.L. McMillan, "Ground State of Liquid He^4," *Physical Rev.*, vol. 138, no. 2A, 1965, pp. A442–A451.
37. D.M. Ceperley, G.V. Chester, and M.H. Kalos, "Monte Carlo Simulation of a Many-Fermion Study," *Physical Rev. B*, vol. 16, no. 7, 1977, pp. 3081–3099.
38. M.H. Kalos, D. Levesque, and L. Verlet, "Helium at Zero Temperature with Hard-Sphere and other Forces," *Physical Rev. A*, vol. 9, no. 5, 1974, pp. 2178–2195.
39. S.D. Ivanov, A.P. Lyubartsev, and A. Laaksonen, "Bead-Fourier Path Integral Molecular Dynamics," *Physical Rev. E*, vol. 67, 2003, pp. 066710-1–066710-11.
40. R.P. Feynman and A.R. Hibbs, *Quantum Mechanics and Path Integrals*, McGraw Hill, 1965.
41. A.F. Voter, "Introduction to the Kinetic Monte Carlo Method," to be published in *Radiation Effects in Solids*, K.E. Sickafus and E.A. Kotomin, eds., Springer, 2005.
42. K.A. Fichthorn and W.H. Weinberg, "Theoretical Foundations of Dynamical Monte Carlo Simulations," *J. Chemical Physics*, vol. 95, no. 2, 1991, pp. 1090–1096.
43. G.H. Vineyard, "Frequency Factors and Isotope Effects in Solid State Rate Processes," *J. Physics and Chemistry of Solids*, vol. 3, 1957, pp. 121–127.
44. J.L. Blue, I. Beichl, and F. Sullivan, "Faster Monte Carlo Simulations," *Physical Rev. E*, vol. 51, no. 2, 1995, pp. R867–R868.
45. T.S. Rahman et al., "Atomistic Studies of Thin Film Growth," *Proc. SPIE*, vol. 5009, 2004, pp. 1–14.
46. O. Trushin et al., "Self-Learning Kinetic Monte Carlo Method: Application to Cu(111)," *Physical Rev. B*, vol. 72, 2005, pp. 115401-1–115401-9.
47. K.M. Chandy and J. Misra, "Distributed Simulation: A Case Study in Design and Verification of Distributed Programs," *IEEE Trans. Software Eng.*, vol. 5, no. 9, 1979, pp. 440–452.
48. B.D. Lubachevsky, "Efficient Parallel Simulations of Dynamic Ising Spin Systems," *J. Computational Physics*, vol. 75, no. 1, 1988, pp. 103–122.
49. G. Korniss et al., "From Massively Parallel Algorithms and Fluctuating Time Horizons to Nonequilibrium Surface Growth," *Physical Rev. Letters*, vol. 84, no. 6, 2000, pp. 1351–1354.
50. G. Korniss et al., "Suppressing Roughness of Virtual Times in Parallel Discrete-Event Simulations," *Science*, vol. 299, 31 Jan. 2003, pp. 677–679.
51. G. Korniss, M.A. Novotny, and P.A. Rikvold, "Parallelization of a Dynamic Monte Carlo Algorithm: A Partially Rejection-Free Conservative Approach," *J. Computational Physics*, vol. 153, no. 2, 1999, pp. 488–508.
52. M. Kardar, G. Parisi, and Y.C. Zhang, "Dynamic Scaling of Growing Interfaces," *Physical Rev. Letters*, vol. 56, no. 9, 1986, pp. 889–892.
53. Y. Shim and J.G. Amar, "Hybrid Asynchronous Algorithm for Parallel Kinetic Monte Carlo Simulations of Thin Film Growth," *J. Computational Physics*, vol. 212, no. 1, 2006, pp. 305–317.
54. S.G. Eick et al., "Synchronous Relaxation for Parallel Simulations with Applications to Circuit-Switched Networks," *ACM Trans. Modeling and Computer Simulation*, vol. 3, no. 4, 1993, pp. 287–314.
55. B.D. Lubachevsky and A. Weiss, "Synchronous Relaxation for Parallel Ising Spin Simulations," *Proc. 15th Workshop Parallel and Distributed Simulation* (PADS'01), IEEE CS Press, 2001; http://lanl.arXiv.org/abs/cs.DC/0405053.
56. Y. Shim and J.G. Amar, "Rigorous Synchronous Relaxation Algorithm for Parallel Kinetic Monte Carlo Simulations of Thin Film Growth," *Physical Rev. B*, vol. 71, 2005, pp. 115436-1–115436-12.
57. Y. Shim and J.G. Amar, "Semirigorous Synchronous Sublattice Algorithm for Parallel Kinetic Monte Carlo Simulations of Thin Film Growth," *Physical Rev. B*, vol. 71, 2005, pp. 125432-1–125432-13.
58. Y. Shim and J.G. Amar, "Growth Instability in Cu Multilayer Films Due to Fast Edge/Corner Diffusion," to be published in *Physical Rev. B*, 2006.

Jacques G. Amar *is an associate professor in the Department of Physics and Astronomy at the University of Toledo. His research interests include the theory of condensed-matter systems far from equilibrium, surface physics, and epitaxial thin-film growth, as well as computational and statistical physics. Amar has a PhD in physics from Temple University. He is a member of the APS and the Materials Research Society. Contact him at jamar@physics.utoledo.edu.*