Computers in Physics

Overcoming Critical Slowing Down

Harvey Gould and Jan Tobochnik

Citation: Computers in Physics **3**, 82 (1989); doi: 10.1063/1.4822858
View online: http://dx.doi.org/10.1063/1.4822858
View Table of Contents: http://scitation.aip.org/content/aip/journal/cip/3/4?ver=pdfcov
Published by the AIP Publishing

Overcoming Critical Slowing Down

Harvey Gould and Jan Tobochnik

Our understanding of phase transitions and cooperative behavior has greatly benefited from large-scale computer simulations.[1,2] Since a true phase transition cannot exist in a finite system, there is a strong motivation to simulate the largest possible system in the fastest possible way. However, if the size of the system becomes sufficiently large, a phenomenon known as critical slowing down limits the usefulness of even the fastest supercomputers. Hence the development of more efficient algorithms is also necessary if we are to use our computer resources effectively. In this column we discuss the nature of critical slowing down and several recent algorithms that reduce it. We will focus on a simple thermal system, but similar ideas are relevant to more complex statistical systems and to quantum field theories.[3]

The *Ising model* is a popular choice for studying the transition from paramagnetic to ferromagnetic behavior. Consider a lattice containing N sites and assume that each site i has an associated spin $s_i = \pm 1$. The energy E of a configuration of spins in the absence of an external magnetic field is given by $E = -J\Sigma_{\langle ij\rangle} s_i s_j$, where the summation is over all nearest-neighbor pairs. The exchange constant J, a measure of the strength of the interaction, is assumed to be positive so that the state of lowest energy is ferromagnetic with all spins pointing in the same direction. We assume that the Ising system is in equilibrium with a heat bath at temeprature T. The interesting physical quantities of the Ising model include the magnetization $m = |\Sigma_i s_i|/N$, the specific heat C, and the isothermal susceptibility χ. Note that for convenience we have defined m to be always non-negative.

As we raise T from $T=0$, the increased thermal energy will cause some spins to "flip" and hence $\langle m\rangle$ will decrease but still be nonzero. However, at T_c, the critical temperature, $\langle m\rangle = 0$ and the system becomes paramagnetic for $T > T_c$. In a finite system, a true phase transition cannot exist, since the maximum size of an ordered region is $O(L)$, the linear dimension of the system. Hence a simulation of the Ising model on a finite lattice will lead to finite size effects such as rounded peaks (see Fig. 1).

The large size of the ordered regions near T_c implies that the time τ required for a region to lose its coherence becomes very long. At T_c, τ increases as L^z for L sufficiently large, a phenomenon known as *critical slowing down*. For the conventional Monte Carlo methods,[4] the dynamical critical exponent z is approximately 2, and hence τ can become very large. Since the simulation time must be greater than τ to obtain a sufficient number of independent configurations, critical slowing down limits the maximum value of L that can be considered.

In order to make the above considerations more concrete, we discuss Monte Carlo (MC) simulations of the Ising model at T_c. We assume an $L \times L$ square lattice and adopt periodic boundary conditions. The dependence of E on the spin configuration does not tell us how the spin configurations change with time, and we have to introduce the dynamics separately. The conventional dynamics or Metropolis algorithm involves single spin flips and can be summarized as follows (see Box 1)

0. Establish an initial spin configuration.

1. Choose a spin at random and attempt, to flip it, $s_i \rightarrow -s_i$.

2. Compute ΔE, the change in energy associated with the attempted flip.

3. If $\Delta E \leqslant 0$, accept the change and go to step 5.

4. If $\Delta E > 0$, compare the Boltzmann probability ratio $w = \exp(-\Delta E/k_B T)$ with a random number r between 0 and 1. If $r \leqslant w$, accept the flip; otherwise reject it and retain the previous configuration.

5. Determine the values of the desired physical quantities, e.g., m and E.

6. Repeat steps 1–5 to obtain a sufficient number of independent configurations and compute the desired averages.

In order that the measure of time be independent of the number of spins, we define the time unit as N attempted spin flips. This time unit is known as one Monte Carlo step per spin.

Suppose that we wish to compute the mean value of a physical quantity A. In general the computation of A is time consuming, and we do not want to compute it more often than necessary. For example, we would probably not compute A after the flip of only one spin, since the values of A in the two configurations would be almost identical and hence strongly correlated. Instead we wish to compute A at time intervals sufficient for the two configurations to contain new information. One way to determine this time interval is to measure $C(t)$, the time-displaced autocorrelation function of A. For $A = m$ we define $C_m(t)$ as

$$C_m(t) = \frac{\langle m(t+t')m(t')\rangle - \langle m\rangle^2}{\langle m^2\rangle - \langle m\rangle^2} \qquad (1)$$

Harvey Gould is a professor of physics at Clark University, Worcester, MA 01610. Jan Tobochnik is an assistant professor of physics and computer science at Kalamazoo College, Kalamazoo, MI 49007.

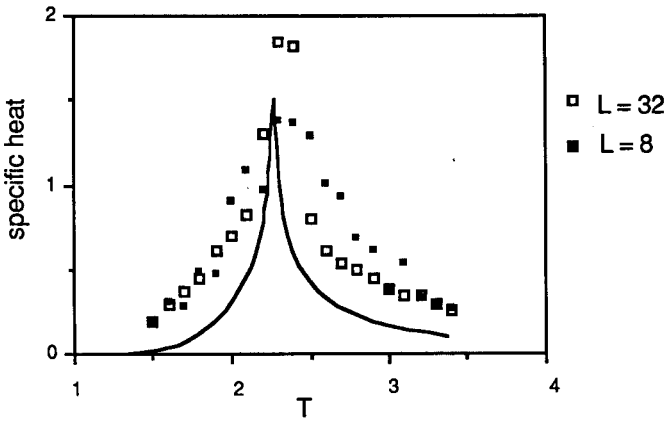

FIG. 1. Temperature dependence of the specific heat (per spin) on an $L \times L$ square lattice with periodic boundary conditions. Note that although C diverges at $T_c = 2.269$ in the limit $L \to \infty$, C exhibits rounded peaks that are shifted from T_c for finite systems. The Metropolis algorithm was used and one thousand Monte Carlo steps per spin were obtained for each value of the temperature. The continuous line represents the behavior of C for $L \to \infty$.

where

$$\langle m(t+t')m(t')\rangle = \frac{1}{(t_s - t)} \sum_{t'=0}^{t_s - t} m(t+t')m(t') \tag{2}$$

and t_s is the number of MC samples or configurations. In equilibrium the choice of t' in (2) is arbitrary and hence $\langle m(t+t')\rangle$ depends on the time difference t rather than $t+t'$ and t' separately. Also, $C_m(t=0) = 1$ and $\lim_{t\to\infty} \langle m(t)m(0)\rangle = \langle m(t)\rangle\langle m(0)\rangle = \langle m^2\rangle$, and hence $C_m(t \to \infty) = 0$.

Consider the calculation of $\langle m(t+t')m(t')\rangle$ for the time difference $t = 2$ and $t_s = 10$. According to (2), $\langle m(2+t')m(t')\rangle = \frac{1}{8}[m(3)m(1) + m(4)m(2) + \cdots + m(10m)(8)]$. We see that in order to use (2) directly for all t, we would have to store the t_s values of m in an array. In most cases it is necessary to consider only $t < t_m$, where t_m is usually chosen to be the time difference for which $C_m(t > t_m) \approx 0$. A FORTRAN subroutine that implements this method for computing $\langle m(t+t')m(t')\rangle$ is given in Box 2. The subroutine stores t_m values of m in the array ms and updates this array each time the contribution to $\langle m(t+t')m(t')\rangle$ is computed. The total number of contributions to $\langle m(t+t')m(t')\rangle$ is $t_s - t_m$. In order to understand this procedure, we suggest that you do an explicit example for small t_m.

A typical result for $C_m(t)$ is given in Fig. 2 for $L = 32$. All our simulations are performed at the critical temperature of the infinite system, i.e., $T_c = 2/\ln(1+\sqrt{2}) \approx 2.269$. Note that C_m appears to decay exponentially, at leat for intermediate times. Since the analysis of $C_m(t)$ involves nonlinear least-squares fits to a sum of exponential decays,[5] we will obtain only qualitative results for τ assuming that the simple form $C_m(t) = Ae^{-t/\tau}$ gives an acceptable fit. In Table I we show our results for τ as a function of L using the Metropolis or single spin flip algorithm. Our estimate obtained from the slope of the log–log plot of $\tau(L)$ yields $z_{sf} = 2.2$, consistent with the best estimate[5] $z_{sf} = 2.125$.

On a serial computer the CPU time needed to obtain t_s configurations increases as L^2 for $d = 2$. However, since $z_{sf} \neq 0$, the time needed to obtain t_s *independent* configurations is approximately $\tau L^2 \propto L^{2+z} \approx L^4$, if we use a single spin flip algorithm. Hence an increase in L by a factor of 10 requires 10^4 more computing time. For this reason there has been considerable effort to develop algorithms that reduce z and hence lead to more efficient simulations of large systems at T_c.

Box 1

```
      subroutine metro(L,n,spin,m,e,iseed,w,imcs)
         Metropolis algorithm for 1 Monte Carlo step per spin (imcs) for L x L lattice
         L linear dimension, n = L^2, iseed for random numbers. Assume L = 16.
      integer L,n,spin(16,16),m,e,iseed,imcs
      real*8 w(9)
      integer isum,isite
         w = Boltzmann probability ratios; index of w is sum of four neighbors + 5.
         w(1) = w(9) = e8, w(3) = w(7) = e4 with e4 = exp(-4/T), e8 = e4*e4; temperature
         T in units of J/k_B.
      do 100 isite = 1,n
         i = int(L*ran(iseed)) + 1          Randomly select location of spin
         j = int(L*ran(iseed)) + 1          ran pseudorandom number, 0 ≤ ran < 1
         Use periodic boundary conditions and return in isum the value of the sum of
         the spins of 4 nearest neighbors
         call pbc(L,spin,i,j,isum)
         if ( spin(i,j)*isum .le. 0 ) then
            call accept(spin,m,e,i,j,isum)
         elseif ( ran(iseed) .lt. w(isum + 5) ) then
            call accept(spin,m,e,i,j,isum)
         endif
100   continue
      imcs = imcs + 1
      end

      subroutine accept(spin,m,e,i,j,isum)
         Update lattice.
      integer spin(16,16),m,e
      integer i,j,isum
      spin(i,j) = -spin(i,j)
      m = m + 2*spin(i,j)
      e = e - 2*spin(i,j)*isum
      end
```

Box 2

```
      subroutine correl(m,ms,c,imcs,itmax,mcum,m2cum)
         Compute correlation function C(t) = <m(t+t')m(t')> for time differences idiff
         <= itmax; itmax configurations saved in array ms before computing C(t).
         Initialize C(t), mcum, m2cum before accumulating data. At end of program
         normalize C, obtain <m> and <m^2> from mcum and m2cum, and calculate correlation
         function using (2) in text.
      integer m,ms(itmax),imcs,itmax,mcum,m2cum
      real*8 c(itmax)
      integer mag,idiff
      mag = abs(m)                                   m = net number of up spins
      if (imcs .gt. itmax) then
         do 100 idiff = 1,itmax
            c(idiff) = c(idiff) + mag*ms(itmax + 1 - idiff)
100      continue
         do 200 idiff = 1,itmax - 1          Update array ms
            ms(idiff) = ms(idiff+1)
200      continue
         ms(itmax) = mag                Store new configuration
      else                              Store configurations if imcs <= itmax
         ms(imcs) = mag
      endif
      mcum = mcum + mag                       Accumulate values of |m|, m^2
      m2cum = m2cum + m*m
      end
```

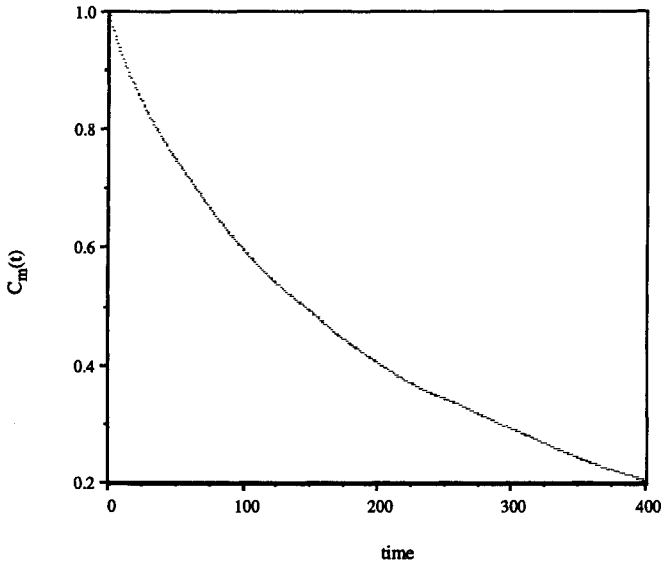

FIG. 2. Typical behavior of the time-dependent correlation function $C_m(t)$.

The new Monte Carlo algorithms[6–9] involve flips of *clusters* of spins rather than single spins. A naive definition of a cluster might be a domain of parallel spins connected by nearest-neighbor bonds. However, the appropriate clusters are obtained by placing bonds between parallel spins with bond probability p less than unity. This probability can be determined by requiring that the percolation transition[10] is equivalent to the Ising transition, i.e., a spanning cluster should be associated with T_c and the critical exponents of all analogous quantities should be identical. The requirement yields $p = 1 - \exp(-2J/k_B T)$. An example of the clusters determined by these bonds is shown in Fig. 3.

The various cluster flip algorithms differ in part by the way that the clusters are weighted. The Swendsen–Wang MC cycle generates all the possible clusters and is summarized by this algorithm.

TABLE I. Estimates for the L dependence of the correlation time τ_{sf} obtained from the single flip (Metropolis) algorithm and τ_w obtained from the Wolff algorithm. Also shown is $\langle c \rangle$, the mean mass of the clusters generated using the Wolff algorithm. The units of τ are Monte Carlo steps per spin and cluster flips, respectively. The typical number of configurations is about 10^5 for each L. More configurations are necessary to obtain quantitative results.

	Metropolis		Wolff
L	τ_{sf}	τ_w	$\langle c \rangle$
4	3.5	...	...
8	6.2	...	...
16	31.7	2.58	140
32	240.0	3.14	471
64	560.0	5.21	1576

1. Search the entire lattice. If two nearest-neighbor spins are parallel, generate a random number r. If $r \leqslant p = 1 - \exp(-2J/k_B T)$, introduce a bond between the spins.

2. Assign each spin to a cluster. The smallest possible cluster contains a single spin.

3. Flip each cluster with probability 1/2.

This cycle is repeated until a sufficient number of independent configurations has been obtained. Because it is nontrivial to determine all the possible clusters for a given configuration, we only quote the Swendsen-Wang result, $z_{sw} = 0.35$ for $d = 2$.

Let us instead explore an algorithm due to Wolff[9] that flips single clusters. The Wolff MC cycle can be summarized by the following algorithm.

1. Choose a spin at random. Its four neighbors (on the square lattice) represent the perimeter spins (see Fig. 4). Form an ordered array corresponding to the perimeter spins that are parallel to the spin and define a counter for the total number of perimeter sites.

2. Choose the first spin in the ordered perimeter array. Remove it from the array and replace it by the last spin in the array. Generate a random number r. A bond exists and the spin is added to the cluster if $r \leqslant p$, where p is the same as in the Swendsen–Wang algorithm.

3. If the spin is added to the cluster, inspect its parallel perimeter spins. If such a spin is not already part of the cluster, add it to the end of the array of perimeter spins.

4. Repeat steps 2 and 3 until no perimeter spins remain.

5. Flip all the spins in the single cluster.

The natural unit of MC time in Wolff dynamics is the number of cluster flips, t_{cf}. In one time unit, the number of sites visited is proportional to the mean mass $\langle c \rangle$ of the cluster, a number less than a sweep of the entire lattice. In order to compare the values of z obtained from the Metropolis, Swendsen–Wang, and Wolff algorithms, we must rescale the time in the latter. We can relate the measured Wolff correlation time τ_{cf} to the corresponding time $\tau_{\rm mcs}$ measured in terms of lattice sweeps by the scaling relation $\tau_{\rm mcs} \propto (\langle c \rangle / N) \tau_{cf}$. We assume $\tau_{cf} \propto L^{z_{cf}}$ and $\tau_{\rm mcs} \propto L^{z_w}$, where z_{cf} is the measured dynamical exponent and z_w is the derived exponent that can be directly compared to the values of z obtained from the lattice sweep algorithms. We use the scaling relation[10] $\langle c \rangle \propto L^{\gamma/\nu}$, and obtain $\tau_{\rm mcs} \propto L^{z_w} \propto L^{\gamma/\nu} L^{-d} L^{z_{cf}}$. Hence z_w is related to the measured value of z_{cf} by

$$z_w = z_{cf} - (d - \gamma/\nu). \tag{3}$$

Suggestions for Further Study

1. Our preliminary results for τ_{cf} and $\langle c \rangle$ are shown in Table I. Use this data to make log–log plots of $\tau_{cf}(L)$ and $\langle c(L) \rangle$ and estimate z_{cf} and γ/ν. Assume that $\nu = 1$ and compare this estimated value of γ to the exact value $\gamma = 1.75$ for $d = 2$. The best estimate[11] for z_w is consistent with $z_w = z_{sf}$ in $d = 2$.

2. Write programs that implement the Metropolis and Wolff algorithms. If a sufficient number of readers send us their results for $C_m(t)$, we can obtain a more precise estimate for z_{sf} and z_w. (Be sure to send us your value of t_m and the number of Monte Carlo steps per spin and/or the number of cluster flips.) It is important to allow the system to equilibrate before accumulating data for C_m. Also, use a previously saved spin configuration as the starting point for a new simulation, and compute quantities such as m and E as the spins are flipped or added to the cluster.

3. The number of independent spin configurations is proportional to t/τ, where t is the number of MC cycles (lattice sweeps or cluster flips). Estimate the actual CPU time needed to obtain 100 independent spin configurations using the various algorithms. For what value of L does your implementation of the cluster flip algorithm become more efficient?

4. Use the Wolff algorithm to compute $\langle m \rangle$ and estimate the critical exponent β using the finite-size scaling relation $\langle m \rangle \propto L^{-\beta/\nu}$. Also compute C for different values of L using the relation $C = (1/k_B T^2)(\langle E^2 \rangle - \langle E \rangle^2)$. Make a log–log plot of $C(L)$. If your plot shows curvature, assume that $C \propto C_0 \ln L$ and determine the constant C_0.

5. Develop a Wolff cluster dynamics program for a simple cubic lattice. (Assume $T_c = 4.5108$.) Estimate z_{cf}, α/ν, β/ν, and γ/ν. Use (3) to estimate z_w and comapre it to $z_{sw} = 0.75$. How do your results for the static exponents compare to that obtained in Ref. 12 using a single spin flip algorithm?

6. Which of the three algorithms that we have discussed is closest to the dynamics of a real magnet near a

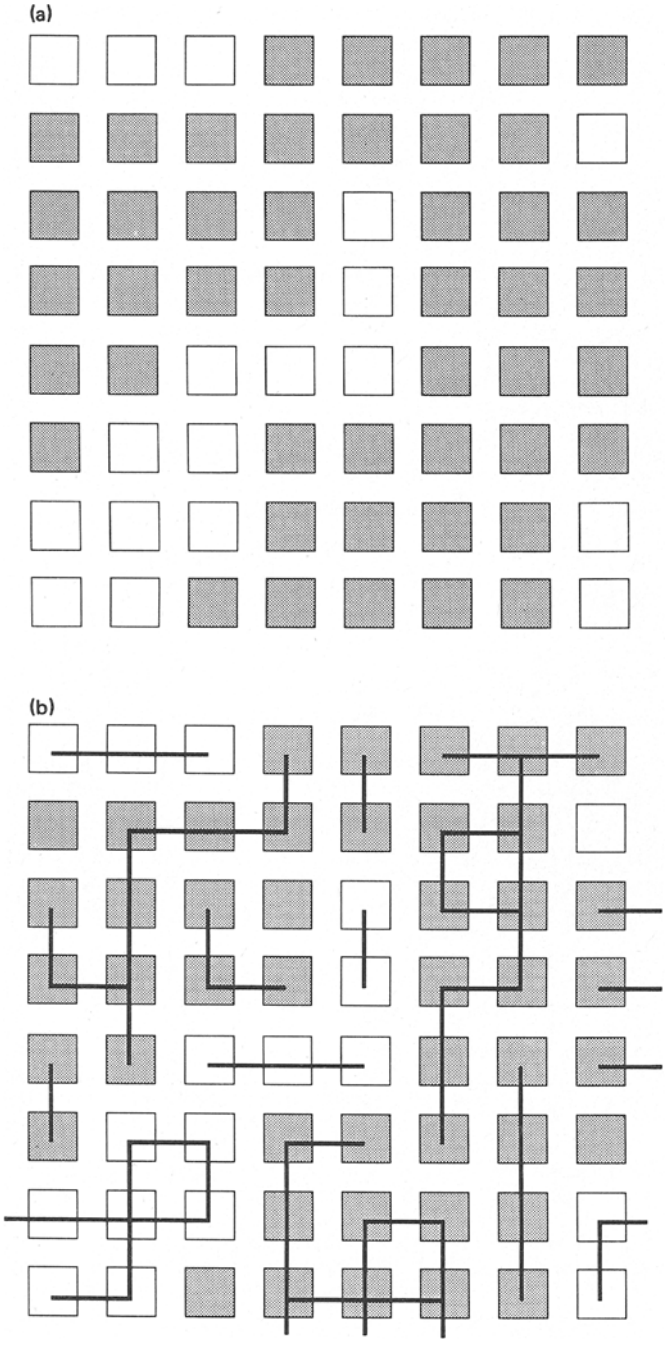

FIG. 3. (a) A typical spin configuration. (b) A typical configuration of clusters formed by bonds placed with probability $p = 1 - \exp(-2J/k_B T)$ between parallel spins. Note that domains of parallel spins are split into smaller clusters, since bonds do not occur between all parallel spins. No bonds are introduced between antiparallel spins and periodic boundary conditions are used.

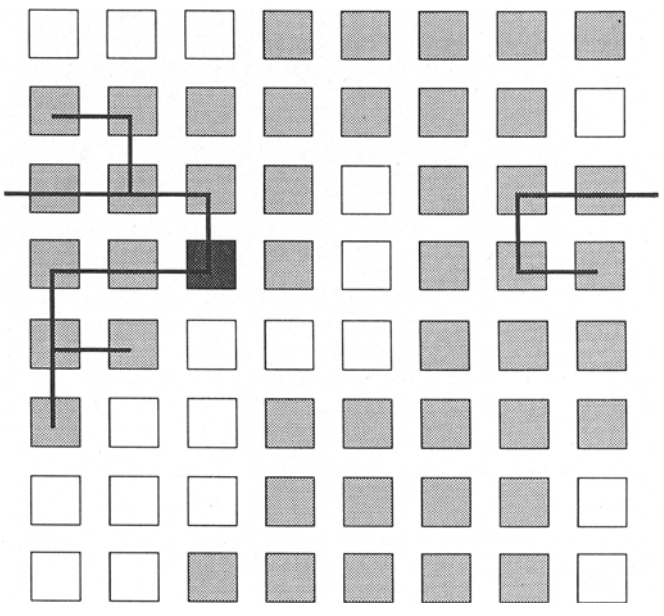

FIG. 4. Example of a cluster using the Wolff algorithms. The seed spin is darkened. Note that although the spin configuration is identical to Fig. 3, the bonds are different since they are generated randomly.

critical point? Since each algorithm yields the same values of the static quantities, which one would you choose to determine static exponents? Which method would you use at other temperatures?

Current areas of research include the applications of cluster dynamics to more complex spin systems and to systems for which the magnetization is conserved and "spin-exchange" dynamics is used. In addition there are many fundamental questions to be answered.[13] For example, why do the cluster dynamics algorithms not eliminate critical slowing down altogether?

The success of this column depends on reader input. Please send us your results, comments, and suggestions for future columns. Results, comments, and requests for FORTRAN programs that implement the spin flip and Wolff algorithms can be sent to us via email to hgould@clarku or tobochnik%heyl.dnet@gw.wmich.edu. ■

Acknowledgments

We wish to thank Richard Brower, Rosco Giles, Bill Klein, Andrew Mel'čuk, Tane Ray, Pablo Tamayo, and Ju-xing Yang for many helpful discussions.

References

1. H. Gould and J. Tobochnik, *Introduction to Computer Simulation Methods* (Addison–Wesley, Reading, MA, 1988). Further discussions of percolation, the Ising model, and finite-size scaling can be found in Part 2.
2. K. Binder and D. W. Heermann, *Monto Carlo Simulation in Statistical Physics* (Springer, New York, 1988). A detailed discussion is given of finite-size effects for the Ising model and percolation.
3. R. C. Brower and P. Tamayo, Phys. Rev. Lett. **62**, 1087 (1989).
4. E. Stoll, K. Binder, and T. Schneider, Phys. Rev. B **8**, 3266 (1973). This pioneering paper is an example of the state of Monte Carlo estimates of z in the early 1970s. Can you obtain a better estimate of z on a desktop computer?
5. S. Tang and D. P. Landau, Phys. Rev. B **36**, 567 (1987).
6. R. H. Swendsen and J.-S. Wang, Phys. Rev. Lett. **58**, 86 (1987).
7. D. Kandel, E. Domany, D. Ron, A. Brandt, and E. Loh, Phys. Rev. Lett. **60**, 1591 (1988).
8. F. Niedermayer, Phys. Rev. Lett. **61**, 2026 (1988).
9. U. Wolff, Phys. Rev. Lett. **62**, 361 (1989). Another important aspect of this paper not described here is an algorithm for reducing critical slowing down for more complex spin systems.
10. D. Stauffer, *Introduction to Percolation Theory* (Taylor and Francis, London, 1985).
11. P. Tamayo, R. C. Brower, and W. Klein, "A Study of Wolff's Collective Monte Carlo Dynamics," Boston University preprint.
12. D. P. Landau, Phys. Rev. B **14**, 255 (1976).
13. See, for example, W. Klein, T. Ray, and P. Tamayo, Phys. Rev. Lett. **62**, 163 (1989). According to their ansatz, z_{sw} can be related to z_{sf} by static exponents.

In our next column: What is the probability density that a many-body system has energy E? (An introduction to optimized Monte Carlo data analysis.)