Scaling, clusters and geometry

Qian, Xiaofeng

Chapter 4

Simulation algorithms for the random-cluster model

We compare the performance of the existing Monte Carlo algorithms for the simulation of the random-cluster representation of the q-state Potts model for continuous values of q. In particular we consider a local bond update method, a statistical reweighting method of percolation configurations, and a cluster algorithm, all of which generate Boltzmann statistics. The dynamic exponent z of the cluster algorithm appears to be quite small, and to assume the values of the Swendsen-Wang algorithm for $q = 2$ and 3. The cluster algorithm appears to be much more efficient than our versions of the other two methods for the simulation of the random-cluster model. The higher efficiency of the cluster method with respect to the local method is primarily due to the fact that the computer time usage of the local method increases more rapidly with system size; the difference between the dynamic exponents is less important. Furthermore, we formulate a new Monte Carlo algorithm for the simulation of the random-cluster model for continuous values of q. It does not use a full cluster decomposition, but flips only one cluster per step, just as the single cluster algorithm for integer q. This algorithm can indeed reduce to the Wolff method for integer q. We compare its performance with the algorithm that involves full cluster decomposition of each configuration. In contrast to the case of integer q, where the single-cluster (Wolff) algorithm is more efficient than the full cluster decomposition (Swendsen-Wang) algorithm, the present single-cluster algorithm for non-integer q has a larger dynamic exponent than the full cluster decomposition algorithm.

4.1 Introduction

The Potts model [1], which can be seen as a generalization of the Ising model, has been the subject of considerable research in recent decades [2]. Many of these investigations make use of the Kasteleyn-Fortuin mapping on the random-cluster model [3]. Remarkably, the symmetry parameter q of the q-state Potts model appears as a continuous parameter in the random-cluster model. Thus, the random-cluster model is a generalization of the Potts

model to non-integer values of q (and on this basis one might even choose to refer to such a model with non-integer q as a Potts model). For the integer $q = 1$, the random-cluster model reduces to the bond-percolation model.

Another mapping, formulated by Baxter, Kelland and Wu [4], leads from the random-cluster model on a planar lattice to the 6-vertex model, which is a limiting case of Baxter's 8-vertex model [5]. This second mapping lends further physical meaning to the random-cluster model.

While many questions concerning the random-cluster model could be answered exactly [6, 7], in many cases, especially in more than two dimensions, numerical approximations are needed. For integer $q > 1$ one can obviously apply a Metropolis-type algorithm to the Potts representation of the model. However, such simulations suffer from the critical-slowing-down phenomenon, which inhibits the investigation of relatively large system sizes. This problem was partly solved by Swendsen and Wang [8]. Their algorithm is non-local in the sense that arbitrarily large groups of Potts variables are flipped at the same time. As a result, critical slowing down, as expressed by the dynamic exponent z, is reduced, though not eliminated. The dynamic exponent is still dependent on the number of states q and the dimensionality d, as reviewed in Ref. [9].

Simulation methods have been developed as well for noninteger q. While the random-cluster model is rich and interesting in its own right, the work on such algorithms for general q may further be justified by fundamental questions such as whether the renormalization scenario [10] for the two-dimensional Potts model does also apply in more than two dimensions. Although this question can also be studied by means of analytical approximations [11, 12], their accuracy is difficult to estimate and numerical tests are thus desirable. Furthermore, some critical exponents, for instance, the so-called backbone exponent of the Potts model is not exactly known, even in two dimensions. It may be determined numerically as a function of q by means of Monte Carlo methods, and then it is natural to include non-integer values of q [13] for a more complete coverage.

A local Monte Carlo algorithm for the non-integer q random-cluster model in two dimensions was formulated by Sweeny [14]. It updates individual bond variables. Although it has been reported that critical slowing down is absent [15], the Li-Sokal method [16] is also applicable here and it follows that $z \geq \alpha/\nu$ where α and ν are the specific-heat and correlation-length exponents respectively. Therefore systems with a positive specific-heat exponent α *must* display critical slowing down, as has been confirmed later [17]. In this algorithm, the transition probabilities depend on non-local information: whether neighbor sites are connected by a percolating path of bond variables. Thus the execution of a bond update may require the exploration of a large percolation cluster. Since the pertinent cluster size is divergent at criticality, the number of operations needed for an update of the system increases faster than the number of sites N in the system. How much faster it increases still depends on the sophistication of the algorithm; the Sweeny [14] algorithm is relatively efficient because it avoids the formation of a whole cluster by following only its perimeter instead.

In a different approach, Hu [18] applied a statistical reweighting procedure to bond percolation configurations in order to sample the $q \neq 1$ random-cluster model. While this

model has no critical slowing down in the sense that it generates uncorrelated configurations, the number of samples needed before a significant weight occurs increases rapidly with the system size [19]. In practice, this effect is similar to critical slowing down in the sense that many simulation steps have to be performed before a meaningful new sample is obtained.

Given the recent simulations [13, 15] that have been performed using the local bond update method, it would be interesting to compare with the performance of a cluster algorithm for continuous q random-cluster models. Indeed the Swendsen-Wang algorithm can be adapted to include non-integer values of q; such an algorithm was described by Chayes and Machta [20]. The resulting cluster algorithm is simple, and requires only of order N operations for an update of the system. But it is applicable only for $q \geq 1$.

In this work we report a comparison between our versions of these three algorithms for non-integer values of q. We illustrate their performance by means of simple applications, and we estimate the dynamic exponent of the cluster algorithm for three values of q. We feel that our findings may be of some use for those planning numerical investigations of the random-cluster model. In Section 4.2 we summarize the algorithms, and we report our results in Section 4.3. Furthermore, in Section 4.4 we formulate a new single-cluster algorithm for the random-cluster model with real $q > 1$. Since for integer q, the Wolff method is generally more efficient than the Swendsen-Wang algorithm, it is of interest to test if the same holds for our single-cluster algorithm in comparison with the full cluster decomposition algorithm for non-integer q. However, after comparing the dynamic exponents of both algorithms, we find that that this is not the case. Remarkably, the single-cluster algorithm formulated in this work represents a new dynamic universality class.

4.2 The existing algorithms

For the convenience of the reader, we summarize the three algorithms for the simulation of the random-cluster model. To expose the close relation with the discrete-q Potts model, we start from the Potts partition sum

$$Z_\sigma = \Big[\prod_{i=1}^{N} \sum_{\sigma_i=1}^{q} \Big] \prod_{\langle ij \rangle} \exp(K \delta_{\sigma_i \sigma_j}) \,, \tag{4.1}$$

where the σ_i are site variables, and the second product is over all nearest-neighbor pairs $\langle ij \rangle$. The coupling K includes a factor $1/k_B T$ and is restricted to $K \geq 0$. The mapping on the random-cluster model [3] eliminates the site variables $\sigma_i = 1, 2, \cdots, q$ after introducing bond variables $b_{ij} = 0$ or 1 between neighboring sites i and j. Bonds $b_{ij} = 1$ (0) are considered to be present (absent). In terms of the new variables one obtains the random-cluster partition sum

$$Z_\sigma = Z_b \equiv \Big[\prod_{\langle ij \rangle} \sum_{b_{ij}=0}^{1} \Big] q^{n_c} u^{n_b} = \sum_{\{b\}} \prod_{k=1}^{n_c} q u^{n_b^{(k)}} \,, \tag{4.2}$$

where $u \equiv \mathrm{e}^K - 1$, n_b is the number of present bonds, and n_c the number of clusters (or components) formed by these bonds. The sum on $\{b\}$ is shorthand for the sum on all bond variables, and $n_b^{(k)}$ is the number of nonzero bonds in the k-th cluster.

Eq. (4.2) can serve directly to formulate a Metropolis-type importance-sampling algorithm for local updates of the bond variables b_{ij}. A bond ($b_{ij} = 1$) contributes a reduced (i.e., divided by k_BT) energy $\ln(1/u)$ if sites i and j are already connected by some other path of such bonds, or $\ln(q/u)$ if they are not connected. Thus the local update of a bond variable requires the performance of a task that is essentially nonlocal: to determine whether i and j belong to the same cluster. After completion of this task, the energy change due to the bond 'flip' is known, and thereby the transition probabilities. Given the time-consuming nature of the task mentioned, one naturally avoids it if not necessary [15]. The latter possibility arises if the value of the random number used for the bond update is such that the result ($b_{ij} = 0$ or 1) does not depend on whether i and j are connected.

In the statistical reweighting method as formulated by Hu [18] one generates independent configurations of bond variables using the percolation model. This ensemble of configurations can be described by Eq. (4.2) with $q = 1$. The bond probability is $p = u/(u+1)$. Thus, the probability distribution of $\{b\}$ is

$$P_{q=1}(\{b\}) = p^{n_b}(1-p)^{N_b - n_b} = u^{n_b}/Z_{q=1}\,, \tag{4.3}$$

with

$$Z_{q=1} = \Big[\prod_{\langle ij\rangle}\sum_{b_{ij}=0}^{1}\Big]u^{n_b} = (1+u)^{N_b}\,, \tag{4.4}$$

where N_b is the total number of nearest-neighbor bonds in the system. The expectation value of an observable A depending on $\{b\}$ is, in the random-cluster model,

$$\langle A\rangle_{RC} = \Big[\prod_{\langle ij\rangle}\sum_{b_{ij}=0}^{1}\Big]Aq^{n_c}u^{n_b}/Z_b\,. \tag{4.5}$$

This can be rewritten as

$$\langle A\rangle_{RC} = \frac{\big[\prod_{\langle ij\rangle}\sum_{b_{ij}=0}^{1}\big]Aq^{n_c}u^{n_b}}{Z_{q=1}} \times \frac{Z_{q=1}}{Z_b} = \langle Aq^{n_c}\rangle_P/\langle q^{n_c}\rangle_P\,, \tag{4.6}$$

where the subscript P denotes averaging on percolation configurations generated by Eq. (4.3). The advantage of this method is that the relevant quantities Aq^{n_c} and q^{n_c} can be sampled on the basis of percolation configurations which are uncorrelated, and simple to generate. The disadvantage is that the reweighting factor q^{n_c} can vary, in particular for large system sizes, over such a large range that, among the generated configurations $\{b\}$, those which contribute significantly to the $\langle\cdots\rangle_P$ averages become very scarce [19].

The cluster algorithm can conveniently be described in terms of a mapping between the random-cluster model, Eq. (4.2), and a model with site as well as bond variables. To this

purpose one defines auxiliary 'color' variables $\tilde{t}_k$=0 or 1 for each cluster $k = 1, 2, \cdots, n_c$:

$$Z_b = \sum_{\{b\}} \prod_{k=1}^{n_c} \sum_{\tilde{t}_k=0}^{1} u^{n_b^{(k)}(1-\tilde{t}_k)} \Big[(q-1)u^{n_b^{(k)}}\Big]^{\tilde{t}_k} . \tag{4.7}$$

Clusters of color 0 and 1 have weight 1 and $q-1$ respectively. The sum over the colors can be replaced by a sum over N site-color variables $t_i = 0$ or 1 if, at the same time, one includes a factor $\delta_{t_i t_j}^{b_{ij}}$ (with the convention 0^0=1) for each bond variable, so that all sites in one cluster have the same color:

$$Z_b = Z_{tb} \equiv \sum_{\{t\}} \sum_{\{b\}} \prod_{\langle ij \rangle} (u\delta_{t_i t_j})^{b_{ij}} \prod_{k=1}^{n_c} (q-1)^{t_{s(k)}} , \tag{4.8}$$

where $s(k)$ is a site in the k-th cluster. For a given site configuration $\{t\}$ one distinguishes three types of bonds (ij):

$$\begin{aligned} \text{type 0} \quad &: \quad t_i = t_j = 0 \ ; \\ \text{type 1} \quad &: \quad t_i = t_j = 1 \ ; \\ \text{type 2} \quad &: \quad t_i + t_j = 1 \ . \end{aligned}$$

Accordingly, superscripts are appended to the pertinent summation and product signs

$$Z_{tb} = \sum_{\{t\}} \Big[\sum_{\{b\}}^{(0)} \prod_{\langle ij \rangle}^{(0)} u^{b_{ij}}\Big] \Big[\sum_{\{b\}}^{(1)} \prod_{\langle ij \rangle}^{(1)} u^{b_{ij}} \prod_{k=1}^{n_c^{(1)}} (q-1)\Big] \Big[\sum_{\{b\}}^{(2)} \prod_{\langle ij \rangle}^{(2)} (1-b_{ij})\Big] , \tag{4.9}$$

where the clusters of color 1 are labeled $1, 2, \cdots, n_c^{(1)}$. Execution of the type 0 and 2 sums, and rewriting the type 1 sum yields the partition sum expressed in site variables, and bond variables only of type 1:

$$Z_{tb} = Z_{tb1} \equiv \sum_{\{t\}} \Big[\prod_{\langle ij \rangle}^{(0)} (1+u)\Big] \sum_{\{b\}}^{(1)} \Big[\prod_{k=1}^{n_c^{(1)}} (q-1) u^{n_b^{(k)}}\Big] . \tag{4.10}$$

Just as Eq. (4.2) describes the probability distribution of bond configurations of the random-cluster model, Eq. (4.10) represents the probability distribution of a system with both site variables $t_i = 0, 1$ and bond variables b_{ij} between nearest-neighbor sites of type 1. The random selection of clusters of color 0 with probability $1/q$ leads to the ensemble of Eq. (4.8) which describes a system of both site and bond variables. A partial summation on bond variables then leads to Eq. (4.10). In a Monte Carlo application of this mapping one makes use of the fact that the terms in the partition sums are proportional to the probability corresponding configuration. Subsequent assignments of values to random variables in the Monte Carlo procedure are decided such that the resulting probability of each configuration is in agreement with the partition function as expressed in the pertinent variables. This guarantees that the equilibrium distribution is always maintained. Thus, starting from a bond configuration drawn from the equilibrium ensemble of Eq. (4.2):

1. Assign color 0 to each cluster with probability $1/q$;
2. Erase all bonds $b_{ij} = 1$ between type-0 sites;
3. Choose new bonds $b_{ij} = 1$ between type-0 sites with probability $u/(u+1)$;
4. Form clusters on the type-0 sites;
5. Erase the color variables.

Here we have stochastically executed the step from Eq. (4.2) to Eq. (4.7), followed by the steps leading to Eq. (4.10); and then, in reverse order, back to Eq. (4.2). This leads to a new bond configuration that again satisfies the equilibrium statistics of Eq. (4.2). We note that these steps resemble the Swendsen-Wang procedure; a difference is that one here uses 2 instead of q colors, and that they are not treated equivalently. The use of a probability $1/q$ restricts the useful range of the algorithm to $q > 1$. The above description of the algorithm is given such as to closely follow the mapping; the actual procedure is even simpler because it is not necessary to keep track of the bond variables b_{ij}. These variables are only needed during the cluster formation process. The information to which cluster a site belongs is stored as an integer that is unique for each cluster.

4.3 Tests and application of the exiting algorithms

We have tested the three algorithms under investigation by comparing their numerical results mutually and, for $q = 2$, with those of conventional algorithms. The results agree within the statistical errors.

4.3.1 Application to specific-heat calculation

To illustrate the use of the Monte Carlo algorithms under consideration, we have calculated the Potts specific heat for the random-cluster model on the square lattice for two non-integer values q. These are $q = 4\cos^2(7\pi/22) = 1.169\cdots$ and $q = 4\cos^2(5\pi/14) = 0.753\cdots$. The dimensionless specific heat (specific heat divided by Nk_B) is here defined on the basis of differentiation of the free energy density $\ln Z$ to the Potts coupling K, with Z e.g. defined as in Eq. (4.2):

$$C \equiv \frac{K^2}{N}\frac{\partial^2 \ln Z}{\partial K^2} \tag{4.11}$$

with $\partial/\partial K = e^K \partial/\partial u$. Since the sum n_b of the bond variables is conjugate to $\ln u$, the specific heat can be obtained from the fluctuations of n_b. It is sufficient to sample the first two moments of n_b:

$$C = \frac{K^2}{N}\Big[\frac{(u+1)^2}{u^2}(\langle n_b^2\rangle - \langle n_b\rangle - \langle n_b\rangle^2) + \frac{u+1}{u}\langle n_b\rangle\Big]. \tag{4.12}$$

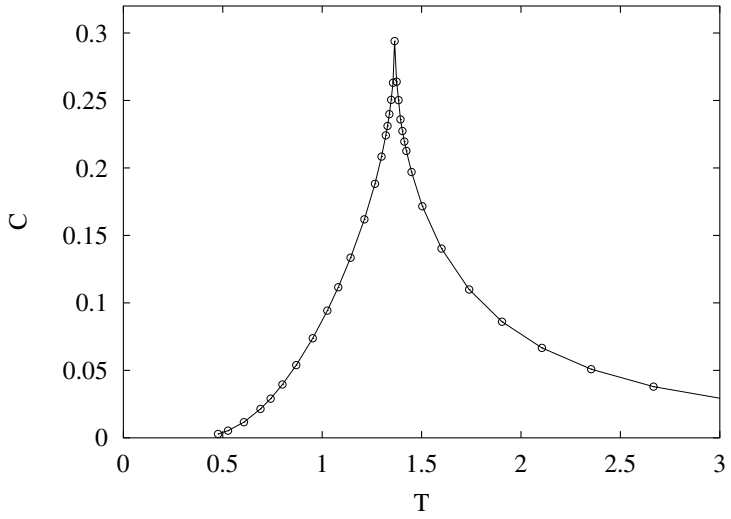

Figure 4.1: Dimensionless Potts specific heat C of the $q = 4\cos^2(7\pi/22) = 1.169\cdots$ random-cluster model versus temperature $T = 1/K$. The statistical errors do not exceed the size of the data points. The data points are extrapolations of finite-size data in the range $6 \leq L \leq 384$, obtained by means of the cluster Monte Carlo algorithm. The finite-size data converge exponentially except at the critical point where power-law behavior occurs. Satisfactory convergence was found for all T except in very narrow ranges ($\Delta T \approx 0.02$) on both sides of the critical point.

The value of the temperature exponent of the Potts model is known as a function of q. This expression was first conjectured by den Nijs [21]; see also [7]. For the specific-heat exponent α this expression leads to $\alpha = 4/3 - 2/[3 - 6\arccos(\sqrt{q}/2)/\pi]$. This formula allows us to select the value of q corresponding to a given value of α.

We first simulated the $q = 4\cos^2(7\pi/22)$ Potts model, which has a specific-heat exponent $\alpha = -1/2$. We have calculated the Potts specific heat C of the square-lattice model in a suitable temperature range, and obtained the curve shown in Fig. 4.1, which does indeed display a square-root type cusp as implied by $\alpha = -1/2$. During these simulations we found that the cluster algorithm was the most efficient one, i.e., produced a more accurate result in a given computer time. The results in Fig. 4.1 are those generated by the cluster algorithm.

Next, we simulated the square-lattice $q = 4\cos^2(5\pi/14)$ Potts model, which has a specific-heat exponent $\alpha = -1$. We have calculated the Potts specific heat and obtained the curve shown in Fig. 4.2, which does indeed display a pronounced kink as implied by $\alpha = -1$. The results in Fig. 4.2 are those generated by the local bond update algorithm, because it became clear that it was more efficient than the reweighting algorithm.

These two figures, together with the well-known logarithmic divergence of the specific heat for $q = 2$, illustrate that the critical singularity becomes less strong when q decreases.

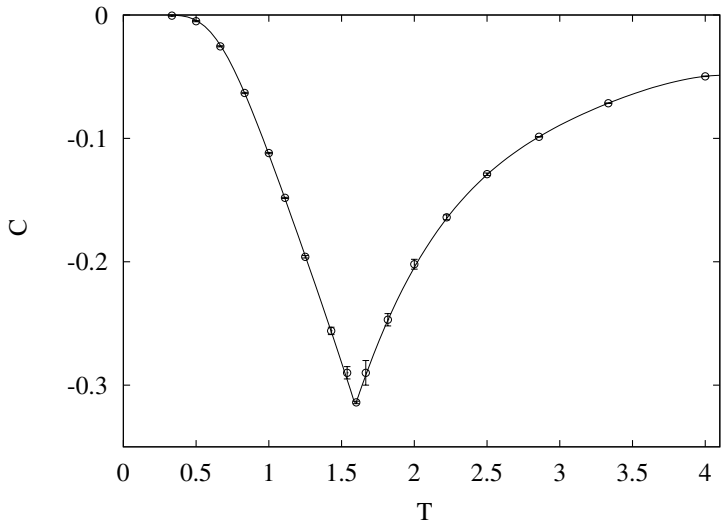

Figure 4.2: Dimensionless Potts specific heat C of the $q = 4\cos^2(5\pi/14) = 0.753\cdots$ random-cluster model versus temperature $T = 1/K$. The statistical errors are larger than in the preceding figure, in some cases they exceed the symbol size. The data points are extrapolations of finite-size data in the range $4 \leq L \leq 40$, obtained by means of the local bond-update Monte Carlo method. The finite-size data converge exponentially except at the critical point where power-law behavior occurs. Satisfactory convergence was found for all T except in narrow ranges ($\Delta T \approx 0.2$) on both sides of the critical point.

The use of K as the temperature parameter facilitates a comparison with the results for the integer-q Potts model. The negative specific heat for $q = 4\cos^2(5\pi/14) < 1$ reflects the fact that the Potts energy per bond decreases with temperature: the reduced energy is K in the ordered state and K/q in the disordered one. This illustrates the unphysical nature of the Potts model for $q < 1$. We note however that the random-cluster model *is physical*. For instance, its energy (not reduced) $E_{RC} = -k_B T\langle n_c \ln q + n_b \ln u\rangle$ is a well-behaved, increasing function of the temperature T when the non-reduced parameters $k_B T \ln q$ and $k_B T \ln u$ are kept constant.

4.3.2 Efficiency of the algorithms

While the dynamical exponent is an important factor in the efficiency of an algorithm, it is not the only one. The degree of overlap between generated and target distributions is crucial in reweighting methods, and furthermore, the computer time per spin update may depend strongly on the system size. From a practical point of view one may be interested in the computer time needed to reach a result with a given statistical accuracy. Thus, to compare the performance of the three algorithms in a quantitative way, we have simulated the two-dimensional random-cluster model on the square lattice, and determined

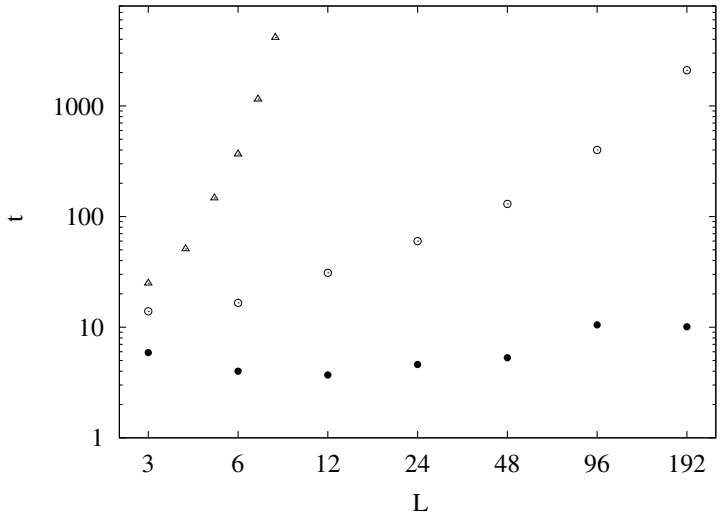

Figure 4.3: Computer time usage t of the three algorithms for the simulation of the continuous-q random-cluster model, versus linear system size L. The time t is determined as the computer time in seconds per lattice site required to reach an accuracy of 10^{-4} in the dimensionless ratio Q of the random-cluster representation of the critical $q = 2$ Potts model. The data points apply to the statistical reweighting method ($\triangle$), to the local bond update method ($\circ$) and to cluster updates ($\bullet$).

a dimensionless ratio Q similar to the Binder cumulant [22]. To this purpose we sampled powers of the cluster sizes

$$S^{(m)} \equiv \sum_{k=1}^{n_c} c_k^m \,, \tag{4.13}$$

where c_k is the size of the k-th cluster, for $m = 2$ and 4. Then the ratio Q is defined as

$$Q \equiv \frac{\langle S^{(2)} \rangle^2}{\langle 3(S^{(2)})^2 - 2S^{(4)} \rangle} \,, \tag{4.14}$$

which, for the case $q = 2$, reduces to the ratio of magnetization moments $\langle m^2 \rangle^2 / \langle m^4 \rangle$. The computer time t per lattice site needed to reach a statistical accuracy of 10^{-4} in Q serves as an inverse measure of the efficiency. The results in terms of $t = (10^4 \delta Q / L)^2 t_R$, where t_R is the CPU time of a run in seconds, δQ the statistical error in Q and L the linear system size, are shown in Fig. 4.3. These results indicate that the cluster algorithm is more efficient than the other two, increasingly so for larger system sizes. The reweighting method appears to become rapidly inefficient with increasing system sizes. Here one may remark that a simple statistical analysis of the probability that the Monte Carlo algorithm generates a state in the center of the target distribution yields factors in which N appears *in the exponent*. This argument thus indicates that the data for t obtained by the reweighting

method increase exponentially with L^2. The interpretation of the results in Fig. 4.3 still requires some reservation. First, the reweighting method naturally becomes more efficient when q approaches 1. Nevertheless, the data shown are clear enough to indicate that the useful range of q is quite narrow for the reweighting method. Second, our version of the local bond-update algorithm is relatively simple and forms clusters, instead of tracing their perimeter as in Sweeny's version. Since the fractal dimension of the perimeter is smaller than that of the cluster itself, Sweeny's version is expected to be more efficient than ours for sufficiently large system sizes, but at the expense of a more complicated code. Given the simplicity and efficiency of the cluster algorithm, we consider it the best choice for the investigation of $q > 1$ models.

4.3.3 Dynamic exponent of the cluster algorithm

As mentioned in the Introduction, the reweighting method does not suffer, at least formally, from any critical slowing down, and thus its dynamic exponent is $z = 0$. The dynamic exponent of the local bond-update method has recently been investigated by Wang et al. [17] for $q = 2$ and 3. Their analysis, apparently more accurate than earlier investigations [14, 15], reported nonzero but still rather small values of z that are, depending on the value of q, comparable with, or even somewhat smaller than those of the Swendsen-Wang algorithm. For $q \to 1$ one expects $z \to 0$ because the bond variables become independent. To evaluate the dynamic universality class of the cluster algorithm for continuous q, we have determined the dynamic exponent z for three different values of q, on the basis of simulations of $L \times L$ square lattices with sizes $L = 6, 12, ..., 160$. We sampled the energy and determined its autocorrelation times τ_L, in units of cluster steps as described in Sec. 4.2, from least-squares fits to the exponentially decaying autocorrelation function. The results are shown in Fig. 4.4. We have analyzed their L-dependence as $\tau_L \simeq L^z$ by means of least-squares fits. We obtain $z = 0.08\,(1)$ for $q = 4\cos^2(7\pi/22)$, and $z = 0.551\,(8)$ for $q = 3$. The result for $q = 3$ is consistent with an existing result for the Swendsen-Wang algorithm, namely $z = 0.56\,(1)$ [17], but it is larger than the result of a more detailed study [23], using system sizes up to $L = 1024$, which is $z = 0.515\,(6)$. The question thus arises whether the difference with the continuous-q cluster algorithm, which amounts to a few standard deviations, indicates that the dynamic universality classes of the two algorithms are different. We do not consider the numerical evidence to be sufficient to reach such a conclusion: it was noted in Ref. [23] that the largest system sizes ($L \geq 128$) led to a significantly smaller result in comparison with the smaller system sizes. This suggest the presence of slowly converging correction terms in the autocorrelation times; such corrections could also be present in our results for the continuous-q cluster algorithm, and thus also explain the difference with our value of z. Especially for $q = 2$ the autocorrelation times are not well described by a single power law; the fits suggest the presence of a second term proportional to $L^{z'}$ with $z' \approx -0.4$. Allowing for such a contribution we obtain $z = 0.254\,(10)$ for $q = 2$, which is consistent with a result $z = 0.25\,(1)$ [24] for the Swendsen-Wang algorithm, but larger than a more recent determination [9, 25] which led to $z = 0.222\,(7)$ on the basis of system sizes up to $L = 512$. We found that our $q = 2$ result

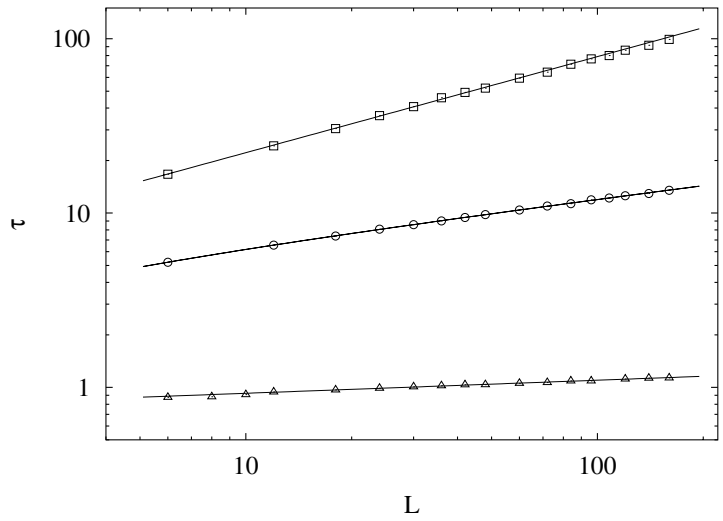

Figure 4.4: Autocorrelation times τ versus system size L for three critical Potts models: $q = 1.169\cdots$ ($\triangle$), $q = 2$ ($\circ$), and $q = 3$ ($\square$) on logarithmic scales. These results were obtained using the continuous-q cluster algorithm. The statistical errors do not exceed the size of the data points.

for z depends considerably on the choice of the fit formula and the range of L. Instead of adding a second term, on can choose to skip the smallest system sizes in order to obtain an acceptable value of the residual χ^2. For systems sizes in the range $60 \leq L \leq 160$ we thus find $z = 0.265\,(5)$. This number, as well as χ^2, increases when the lower limit of the L-range is decreased (these numbers agree better with earlier determinations [8, 26] which are close to $z = 0.3$). Under these circumstances, we do not assign much significance to the differences between the reported values of z for the Swendsen-Wang algorithm and the continuous-q cluster algorithm. These differences are of the same order as those between different results reported for the Swendsen-Wang algorithm and may be attributed to unresolved corrections to scaling.

4.4 Single-cluster algorithm

Each term in the second sum in Eq. (4.10) specifies a cluster decomposition $\mathcal{D}(\{b\})$ of the sublattice formed by sites k with $t_k = 1$. Different sets of bond variables $\{b\}$ may still correspond with the same cluster decomposition. Thus, if we replace the sum on $\{b\}$ by a sum on all cluster decompositions of the color-1 regions, we have to insert a summation

on all $\{b\}$ that are consistent with $\mathcal{D}$:

$$Z_{tb} = Z_{t\mathcal{D}} \equiv \sum_{\{t\}} \prod_{\langle ij \rangle}^{(0)} (1+u) \sum_{\{\mathcal{D}\}}^{(1)} \sum_{\{b\}|\mathcal{D}} \Big[\prod_{k=1}^{n_c^{(1)}} (q-1) u^{n_b^{(k)}} \Big] . \tag{4.15}$$

Eq. (4.15) can serve as the basis on which a single-cluster Monte Carlo algorithm can be constructed. This algorithm is applied as follows to a mixed configuration specified by the site variables t_i and a cluster decomposition $\mathcal{D}$ of the color-1 sites.

Choose a random site i. The action taken by the algorithm depends on the color variable t_i. If

1. $t_i = 0$, do with probability $p_1 = (q-1)/q$ the following: form a random cluster around site i with bond probability $p = u/(u+1)$ between sites of color 0. The sites j in the newly formed cluster are assigned color 1 (i.e. $t_j = 1$) and the number $n_c^{(1)}$ of clusters of color 1 is thus increased by 1.

2. $t_i = 1$, do with probability $p_2 = 1/q$ the following: assign color 0 to all sites of the cluster containing site i, and thus decrease the number of clusters of color 1 by 1.

4.4.1 Proof of detailed balance

The proof of detailed balance can be formulated as follows. Consider two mixed configurations S_1 and S_2, which differ only in a region $\mathcal{C}$ whose sites j have color $t_j = 0$ in S_1, and whose sites belong to a single cluster in S_2, and thus have color $t_j = 1$. The probability to move from S_1 to S_2 is

$$T(2,1) = \frac{(q-1)N_c}{qN} \sum_{\{b\}|\mathcal{C}} \left(\frac{u}{u+1} \right)^{n_b} \left(\frac{1}{u+1} \right)^{n_p + n_{nn} - n_b} , \tag{4.16}$$

where N_c is the number of sites in region $\mathcal{C}$; N is the total number of sites in the system; $\{b\}$ stands for the n_{nn} bond variables on the edges between nearest-neighbor sites in $\mathcal{C}$; the combination on $\{b\}|\mathcal{C}$ indicates the sum on all configurations $\{b\}$ that connect all sites in $\mathcal{C}$ into a single cluster; n_b denotes the number of nonzero bond variables in $\{b\}$; n_p is the number of bond variables connecting sites inside $\mathcal{C}$ with those outside $\mathcal{C}$ of color 0 (i.e., the number of bonds along the boundary of $\mathcal{C}$ that is broken when the color of $\mathcal{C}$ is changed). The prefactor $(q-1)N_c/qN$ describes the probability that the cluster formation starts within $\mathcal{C}$. Each of the n_b 'present' bonds contributes a factor $u/(u+1)$, and each of the $n_{nn} - n_b$ 'absent' bonds a factor $1/(u+1)$. Also each 'broken' bond along the perimeter of $\mathcal{C}$ contributes a factor $1/(u+1)$.

According to the rule given above, the probability of the inverse move from S_2 to S_1 is simply

$$T(1,2) = \frac{N_c}{qN} . \tag{4.17}$$

The condition of detailed balance requires that the transition probabilities $T(2,1)$ and $T(1,2)$ are related to the equilibrium probabilities $P(1)$ and $P(2)$ of configurations 1 and 2 respectively:

$$T(2,1)/T(1,2) = P(2)/P(1)\,. \tag{4.18}$$

Since the probabilities $P(1)$ and $P(2)$ are proportional to the weights specified by Eq. (4.10), we may write

$$P(2)/P(1) = W(2)/W(1)\,, \tag{4.19}$$

where the weights associated with region $\mathcal{C}$ in Eq. (4.10) are

$$W(1) = (1+u)^{n_p+n_{nn}}\,, \tag{4.20}$$

and

$$W(2) = (q-1)\sum_{\{b\}|\mathcal{C}} u^{n_b}\,. \tag{4.21}$$

From Eqs. (4.16) and (4.17), and from Eqs. (4.20) and (4.21), we conclude that

$$T(2,1)/T(1,2) = (q-1)(1+u)^{-n_p-n_{nn}}\sum_{\{b\}|\mathcal{C}} u^{n_b} = W(2)/W(1)\,, \tag{4.22}$$

which shows that the condition of detailed balance indeed is satisfied.

4.4.2 Other versions

The probabilities p_1 and p_2 can be chosen differently, depending on the value of q. For $1 < q < 2$ we may take $p_1 = q-1$ and $p_2 = 1$. For $q > 2$, this is not possible but other possibilities arise. One can generalize the algorithm by allowing more than two values of the color variables t_i. The most obvious way is to allow $n \equiv [q]$ (the integer part of q) values with weight one, and one special value with weight $q-n$. Sites of the latter color are divided in clusters (just as before); sites of the n remaining colors are not. A cluster step starting from a randomly chosen site can then be specified as follows: if that site belongs to a cluster (thus, of the special color), then the cluster is erased and its sites are given a random color $1, 2, \cdots, n$ with probability $1/n$ each. If the cluster step starts from a randomly chosen site of color $1, 2, \cdots, n$, then a single cluster is formed. Its sites receive one of the $n-1$ other weight-1 colors with probability $(2n-q)/(n(n-1))$ each, and the cluster receives the special color with probability $(q-n)/n$. This choice of probabilities satisfies detailed balance and maximizes the probability of a cluster flip. For integer q it reduces to the Wolff algorithm.

4.4.3 Test and dynamic exponent

Test of the algorithm

We tested the single-cluster algorithm for the $q = 2$, 3, and 4 Potts model, by comparing their numerical results with those of the Wolff algorithm. We let $n = q-1$ and the special

value be $q - n = 1$. Simulations were performed on the $L \times L$ square lattice with periodic boundary conditions. We sampled various quantities, including the second moment of the magnetization $m_2 = \langle m^2 \rangle$ and the average size s of the clusters formed by the algorithm. The quantity m_2 is defined as

$$m_2 = \frac{1}{q-1} \left\langle \sum_{i=1}^{q-1} \sum_{j=i+1}^{q} (\rho_i - \rho_j)^2 \right\rangle , \tag{4.23}$$

where ρ_i is the density of sites in state i. The cluster size s is counted as the total number of lattice sizes in the cluster. It can be easily shown that, within the statistical error margins, the two quantities m_2 and s are equal to each other. As expected, for the same model with the same values of q, K, L, the Wolff and the present algorithm indeed yielded identical results for m_2 and s; the values of m_2 and s also agree with one another. For an illustration, the m_2 and s data for the critical Ising model are shown in Table 4.1. Since

Table 4.1: Data for m_2 and s in the critical Ising model, as obtained by the Wolff (W) and the present single-cluster algorithm (S). The parameter L specifies the linear system size. The number of samples per system size is 4×10^6 for each simulation, and the number of clusters formed between subsequent samples is 2 for $L \leq 24$ and 3 for $L = 32$. The numbers in the brackets represent the statistical error margins in the last two decimal places.

L	m_2 (W)	m_2 (S)	s (W)	s (S)
8	0.64693(18)	0.6478(6)	0.64666(18)	0.6470(6)
12	0.58581(18)	0.5861(8)	0.58581(18)	0.5860(8)
16	0.54537(17)	0.5442(9)	0.54544(17)	0.5441(9)
20	0.51584(16)	0.5164(10)	0.51594(16)	0.5165(10)
24	0.49305(17)	0.4932(13)	0.49311(17)	0.4932(13)
32	0.45874(14)	0.4610(12)	0.45878(14)	0.4610(12)

both simulations involve the same number of samples, the statistical error uncertainties, shown in the brackets in Table 4.1, reflect the relative efficiency of the Wolff and the single-cluster algorithm. For size $L = 8$, the Wolff method is about 10 times as efficient as the present algorithm, while for $L = 32$ it is about 100 times. Therefore, the two algorithms have different dynamic exponents.

Dynamic exponent of the algorithm

To study the dynamic universality class of the present single-cluster algorithm, we simulated the $q = 4, 3, 2$, and $4\cos^2(7\pi/22)$ Potts models on a square lattice with periodic boundary conditions. Similar to what we did in Sec. 4.3.2, we sampled the energy and determined its autocorrelation times τ_L for 10 different system sizes L with $8 \leq L \leq 48$. The data of τ_L are shown in Fig. 4.5. We fitted the finite-size data by

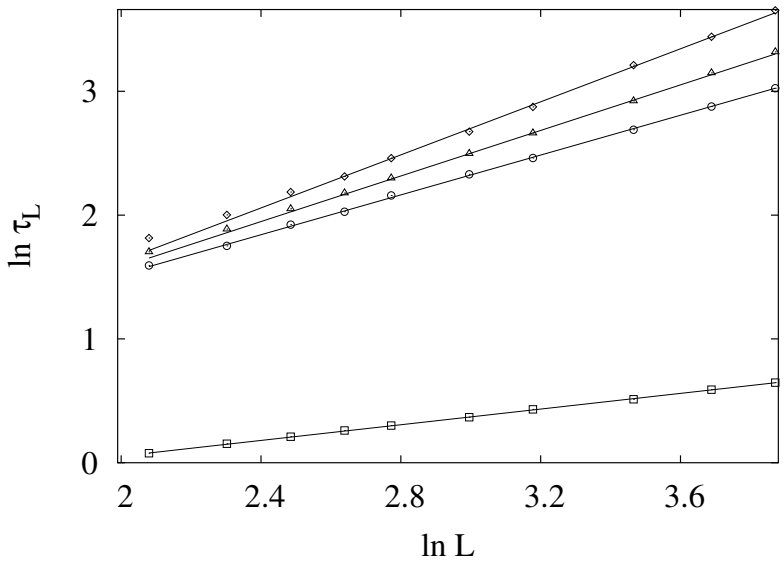

Figure 4.5: Autocorrelation time τ_L for several critical Potts models, shown as $\ln \tau_L$ versus $\ln L$. The data for $q = 4\cos^2(7\pi/22)$, 2, 3, and 4 are represented by symbols $\square$,$\bigcirc$, $\triangle$, and $\Diamond$, respectively.

$$\tau_L = L^z(a + bL^{y_i})\,, \tag{4.24}$$

where a correction term is introduced and the correction exponent was taken [27] as $y_i = -28/15$ for $q = 4\cos^2(7\pi/22)$, $y_i = -2$ for $q = 2$, and $y_i = -4/5$ for $q = 3$. For $q = 4$, in addition to the term with exponent $y_i = -2$, we included $c/\ln L$ in the brackets of Eq. (4.24). We obtain $z = 0.27(2)$ for $q = 4\cos^2(7\pi/22)$, $z = 0.80(2)$ for $q = 2$, $z = 0.94(2)$ for $q = 3$, and $z = 1.17(2)$ for $q = 4$. A comparison of these dynamic exponents and those of the full-cluster decomposition algorithm described in Sec. 4.3.2 (see also Ref. [20]) indicates they are in different dynamic universality classes.

4.5 Discussion

The single-cluster algorithm described above is obviously related to the Wolff [28] algorithm as defined for discrete-q Potts models; it can reduce to the Wolff method if q is an integer. On the other hand, it is different in the sense that the single-cluster algorithm acts on a mixed configuration of site variables and random-cluster variables. In contrast to the fact that the Wolff method is generally more efficient than the full-cluster decomposition algorithm for integer q, the present method is much less efficient in comparison with the full-cluster decomposition algorithm. This can be understood as follows. In the single-cluster algorithm, the cluster decompositions of the special state $q - [q]$ is stored in computer memory, while direct spin values are assigned to lattice sites for the other $[q]$ Potts states.

As described in Sec. 4.4.2, the formed clusters for the special state will be flipped back into the integer spin states. Thus, it seems that one gains little by forming a cluster for the special state from one of the $[q]$ Potts states. More importantly, the autocorrelation function for the cluster decompositions has a long time "tail" during simulations, because the smaller a cluster is, the less probable it is to be visited. This long "tail" is mainly responsible for the relatively large dynamic exponent of the single-cluster method.

Bibliography

[1] R. B. Potts, Proc. Cambridge Philos. Soc. **48**, 106 (1952).

[2] F. Y. Wu, Rev. Mod. Phys. **54**, 235 (1982).

[3] P. W. Kasteleyn and C. M. Fortuin, J. Phys. Soc. Japan. (Suppl.) **26**, 11 (1969).

[4] R. J. Baxter, S. B. Kelland and F. Y. Wu, J. Phys. A **9**, 397 (1976).

[5] R. J. Baxter, Phys. Rev. Lett. **26**, 178 (1971).

[6] R. J. Baxter, J. Phys. C **6**, L445 (1973).

[7] B. Nienhuis, in *Phase Transitions and Critical Phenomena*, edited by C. Domb and J. L. Lebowitz (Academic Press, London, 1987), Vol. **11**, p.1.

[8] R. H. Swendsen and J. S. Wang, Phys. Rev. Lett. **58**, 86 (1987).

[9] G. Ossola and A. D. Sokal, Nucl. Phys. B **691**, 259 (2004).

[10] B. Nienhuis, A. N. Berker, E. K. Riedel and M. Schick, Phys. Rev. Lett. **43**, 737 (1979).

[11] B. Nienhuis, E. K. Riedel and M. Schick, Phys. Rev. B **23**, 6055 (1981).

[12] S. Grollau, M. L. Rosinberg and G. Tarjus, Physica A **296**, 460 (2001).

[13] Y. Deng, H. W. J. Blöte and B. Nienhuis, Phys. Rev. E **69**, 026114 (2004).

[14] M. Sweeny, Phys. Rev. B **27**, 4445 (1983).

[15] F. Gliozzi, Phys. Rev. E **66**, 016115 (2002).

[16] X. J. Li and A. D. Sokal, Phys. Rev. Lett. **63**, 827 (1989).

[17] J. -S. Wang, O. Kozan and R. H. Swendsen, Phys. Rev. E **66**, 057101 (2002).

[18] C. -K. Hu, Phys. Rev. Lett. **69**, 2739 (1992).

[19] J. R. Heringa and H. W. J. Blöte, Phys. Rev. Lett. **70**, 2044 (1993).

[20] L. Chayes and J. Machta, Physica A **254**, 477 (1998).

[21] M. P. M. den Nijs, J. Phys. A **12**, 1857 (1979).

[22] K. Binder, Z. Phys. B**43**, 119 (1981).

[23] J. Salas and A. D. Sokal, J. Stat. Phys. **87**, 1 (1997).

[24] P. D. Coddington and C. F. Baillie, Phys. Rev. Lett. **68**, 962 (1992).

[25] J. Salas and A. D. Sokal, e-print cond-mat/9904038v1, Sec. 6.

[26] C. F. Baillie and P. D. Coddington, Phys. Rev. B **43**, 10617 (1991).

[27] J. L. Cardy, in *Phase Transitions and Critical Phenomena*, edited by C. Domb and J. L. Lebowitz (Academic Press, London, 1987), Vol. 11, p.55.

[28] U. Wolff, Phys. Rev. Lett. **62**, 361 (1989).