Exact sampling from anti-monotone systems

Olle Häggström*† Karin Nelander‡

Abstract

A new approach to Markov chain Monte Carlo simulation was recently proposed by Propp and Wilson. This approach, unlike traditional ones, yields samples which have *exactly* the desired distribution. The Propp–Wilson algorithm requires this distribution to have a certain structure called monotonicity. In this paper an idea of Kendall is applied to show how the algorithm can be extended to the case where monotonicity is replaced by anti-monotonicity. As illustrating examples, simulations of the hard-core model and the random-cluster model are presented.

Keywords: Markov chain Monte Carlo, Propp–Wilson algorithm, hard-core model, random-cluster model.

1. Introduction

In many situations, it is important to be able to sample from some, often very complicated, multivariate probability distribution π. One approach is the Markov chain Monte Carlo method, which originates from statistical mechanics (Metropolis *et al.* 1953), and which currently is very fashionable in statistics (Gilks, Richardson and Spiegelhalter 1996). The idea is to define an ergodic Markov chain with stationary distribution π, to start the chain in some arbitrary state, to run it for a long time, and to output the final state. Ergodicity of the Markov chain guarantees that if it is run for long enough, then the distribution of the final state is close to π. A problem with this approach is that it is often very difficult to determine how long is "long enough". For this reason, much of today's Markov chain Monte Carlo practice lacks rigorous theoretical justification. This is of course highly unsatisfactory, and has even caused some controversy (see e.g. the tandem discussion papers Gelman and Rubin (1992) and Geyer (1992)).

*Mathematical Statistics, Chalmers University of Technology, 412 96 Göteborg, Sweden; olleh@math.chalmers.se

†Research supported by a grant from the Swedish Natural Science Research Council.

‡Mathematical Statistics, Chalmers University of Technology, 412 96 Göteborg, Sweden

During the last few years, some attempts have been made to remedy this situation through the construction of algorithms involving the running of a Markov chain for a certain random amount of time (a stopping time) defined in a clever way so that the output of the algorithm has *exactly* the desired distribution π. The first such algorithm is due to ASMUSSEN *et al.* (1992), and has been followed by other, simpler, algorithms e.g. in LOVÁSZ and WINKLER (1995) (see PROPP and WILSON (1997) for a short survey). These algorithms work for any irreducible finite state Markov chain, but have astronomical running times for all but the very simplest chains (i.e. those with very few states) and are therefore of limited practical value.

In contrast, PROPP and WILSON (1996) recently devised an algorithm for exact sampling which is fast enough to be useful in practice. The price they have to pay for this is that the Markov chain (still with finite state space) has to possess a certain monotonicity structure. This leads to a similar restriction on the set of distributions π that the algorithm is applicable to. This monotonicity structure does hold for many important models, such as the ferromagnetic Ising and Potts models and certain random tilings. The Propp–Wilson algorithm provides the first computationally tractable way of obtaining unbiased samples from these models. In KENDALL (1996) and HÄGGSTRÖM *et al.* (1997), the algorithm has also been successfully adapted to point process settings (this direction is currently being pursued further in KENDALL and MØLLER (1997)). Another direction of generalization was recently considered by MURDOCH and GREEN (1997).

The Propp–Wilson algorithm will be carefully described in Section 2. It is based on so called *coupling-from-the-past*. By a coupling, we here mean the joint construction of two or more realizations of the Markov chain; see LINDVALL (1992) for a broad treatment of coupling methods in probability theory. "From the past" means that we run the chain from a distant point of the past up until the present, and the distance to the past is a kind of "backwards stopping time" which is determined during the running of the algorithm.

Our contribution is to demonstrate how the Propp–Wilson algorithm, via a simple modification, can be used for discrete systems which are anti-monotone rather than monotone. The modification is based on an idea that KENDALL (1996) came up with in a point process context. The anti-monotone setting contains several interesting and important examples, such as the hard-core model and the Ising antiferromagnet. We believe that the need for efficient simulation algorithms may be even greater for anti-monotone than for monotone systems, the reason being the following. For monotone systems, several powerful mathematical tools, such as Holley's Theorem and the FKG inequalities, are available (see Section II.2 of LIGGETT, 1985, for an overview). At present, these tools do not have counterparts for anti-monotone systems. Therefore, it is more difficult to obtain theoretical results for anti-monotone systems, for which one thus has to resort to computer simulations to a larger extent.

After circulation of the preprint for the present paper, MØLLER (1997) has

developed the idea of anti-monotone coupling-from-the-past further. Whereas the state space in our setup is finite, he is able to handle unbounded state spaces. Also worth mentioning is the work of FILL (1997), where an alternative to the Propp–Wilson approach, using "strong stationary times" rather than coupling-from-the-past, is presented. Fill's algorithm works both in monotone and anti-monotone situations.

The examples that we will consider are mainly motivated by problems in statistical mechanics. As we already pointed out, Markov chain Monte Carlo methods are widely used also in statistics. This is particularly true in Bayesian statistics where such methods are used to sample from posterior distributions in order to estimate means etc. (see GILKS *et al.* 1996). It is quite conceivable that coupling-from-the-past methods may in the future become an important tool for getting better such estimates; see MURDOCH and GREEN (1997) for a pioneering application in this direction.

In the remainder of this section, we will define monotonicity and anti-monotonicity, and give some important examples of anti-monotone systems. In Section 2 we present the algorithms: the original Propp–Wilson algorithm, as well as our modification. In Section 3 we discuss the special case of Markov random fields on bipartite graphs. Finally, in Section 4, we present some simulation results.

Our setup involves a finite set V of locations, and a finite set $S \subset \mathbb{R}$ of values attainable at each location. The state space of the whole system is then S^V. We write ξ, η, etc. for fixed elements of S^W, where $W \subseteq V$. We furthermore write $X, X_1, \ldots$ for S^V-valued random variables. We let $\preceq$ denote the natural partial order on S^W, i.e. $\xi \preceq \eta$ if $\xi(v) \leq \eta(v)$ for each $v \in W$.

Definition 1.1. *A probability measure π on S^V is said to be* ***monotone*** *if for each $v \in V$ and each $s \in S$ we have*

$$\pi(X(v) \leq s \,|\, X(V \setminus \{v\}) = \xi) \;\geq\; \pi(X(v) \leq s \,|\, X(V \setminus \{v\}) = \eta)$$

whenever $\xi \preceq \eta$, $\pi(X(V \setminus \{v\}) = \xi) > 0$ and $\pi(X(V \setminus \{v\}) = \eta) > 0$. If on the other hand

$$\pi(X(v) \leq s \,|\, X(V \setminus \{v\}) = \xi) \;\leq\; \pi(X(v) \leq s \,|\, X(V \setminus \{v\}) = \eta)$$

for all such v, s, η and ξ, then π is said to be ***anti-monotone****.*

In other words, π is monotone (resp. anti-monotone) if the conditional distribution of the value at a site v given everything else is stochastically increasing (resp. decreasing) in the configuration on $V \setminus \{v\}$.

We consider three examples of anti-monotone systems. First, we need some graph notation. Let G be a graph with vertex set V and edge set E. The edges are undirected, and loops and parallel edges are not allowed. For $v, w \in V$, we write $v \sim w$ if there is an edge $e \in E$ that has v and w as its end-vertices. In all of the examples, the most common choice of G is a large portion of the cubic lattice $\mathbb{Z}^d$, with edges connecting nearest neighbours.

Example 1.1: The hard-core model. The hard-core model (see e.g. GEORGII (1988) and VAN DEN BERG and STEIF (1994)) was introduced in statistical mechanics as a crude model for a gas whose particles have non-negligible radii. It has also been studied in operations research, where it arises in the study of certain communication networks, see KELLY (1985). Each vertex of the graph G can be in one of two states, 0 and 1. In the gas interpretation of the model, a 1 represents the presence of a particle at a vertex, and a 0 means that the vertex is not occupied by a particle. When two 1's occur on neighbouring sites, we think of the particles as overlapping each other, and therefore such configurations are disallowed. A configuration $\xi \in \{0,1\}^V$ is called **feasible** if no such overlaps occur (i.e. ξ is feasible if $\xi(v)\xi(w) = 0$ whenever $v \sim w$). The hard-core measure π_G^a for G with activity $a > 0$ is the measure on $\{0,1\}^V$ which to each $\xi \in \{0,1\}^V$ assigns probability

$$\pi_G^a(\xi) = \begin{cases} \frac{1}{Z_G^a} \prod_{v \in V} a^{\xi(v)} & \text{if } \xi \text{ is feasible} \\ 0 & \text{otherwise,} \end{cases}$$

where Z_G^a is the appropriate normalizing constant making π_G^a a probability measure. In other words, π_G^a is obtained by letting the vertices independently be in state 1 (resp. 0) with probability $\frac{a}{a+1}$ (resp. $\frac{1}{a+1}$), and then conditioning on the event that the arising configuration is feasible. It is immediate from the definition that if ξ is feasible, then

$$\pi_G^a(X(v) = 1 \,|\, X(V \setminus \{v\}) = \xi) \;=\; \begin{cases} \frac{a}{a+1} & \text{if } \xi(w) = 0 \text{ for all } w \sim v \\ 0 & \text{otherwise.} \end{cases} \qquad (1)$$

Since the right hand side is decreasing in ξ, we have that the hard-core model is anti-monotone.

Example 1.2: The Ising antiferromagnet. One of the most studied models in statistical mechanics is the Ising model, in which each vertex v of G can be in state 1 or -1 (see e.g. GEORGII (1988) and LIGGETT (1985) for general discussions). In the original interpretation of the model, the vertices are thought of as atoms, and the two states represent two different spin orientations. A number of other interpretations have since been suggested in various physical and engineering contexts. The Ising measure ν_G^J for G with coupling constant $J \in (-\infty, \infty)$ is the measure on $\{-1,1\}^V$ which to each $\xi \in \{-1,1\}^V$ assigns probability

$$\nu_G^J(\xi) = \frac{1}{Z_G^J} \exp\{-H(\xi)\}$$

where

$$H(\xi) \;=\; 2J \sum_{\substack{v \sim w \\ v,w \in V}} \mathbf{1}_{\{\xi(v) \neq \xi(w)\}} \qquad (2)$$

is the so called energy function and Z_G^J is a normalizing constant. In (2), each pair $\{v, w\}$ with $v \sim w$ is counted only once. It follows that

$$\nu_G^J(X(v) = 1 \,|\, X(V \setminus \{v\} = \xi)) \;=\; \frac{1}{1 + e^{2J(\kappa_- - \kappa_+)}}$$

where κ_+ (resp. κ_-) is the number of neighbours $w \sim v$ for which $\xi(w) = 1$ (resp. $\xi(w) = -1$). This conditional probability is an increasing function of ξ when $J \geq 0$, and decreasing when $J \leq 0$. Hence, the $J \geq 0$ Ising model is monotone whereas the $J \leq 0$ Ising model is anti-monotone. The $J \geq 0$ case, in which neighbouring vertices tend to take the same value, is usually referred to as the Ising ferromagnet, while the anti-monotone case $J \leq 0$ is called the Ising antiferromagnet. The ferromagnetic case has received most of the attention, but also the antiferromagnet has attracted a fair amount of interest (see e.g. VAN DEN BERG (1993) and some of its references).

Example 1.3: The $q < 1$ random-cluster model. Another name for the random-cluster model is the FK model after its inventors FORTUIN and KASTELEYN (1972) (see e.g. GRIMMETT (1995) and HÄGGSTRÖM (1996) for up-to-date discussions). The model lives on the edge set rather than the vertex set of G, and should be thought of as dealing with random subsets of E. Each edge can be in one of the states 0 and 1, where a 1 (resp. 0) represents the presence (resp. absence) of an edge. The random-cluster measure $\mu_G^{p,q}$ for G with parameters $p \in [0, 1]$ and $q > 0$ is the probability measure on the set of subgraphs of G given by

$$\mu_G^{p,q}(\eta) = \frac{1}{Z_G^{p,q}} \left\{ \prod_{e \in E} p^{\eta(e)} (1-p)^{1-\eta(e)} \right\} q^{k(\eta)}$$

for all $\eta \in \{0, 1\}^E$. Here $k(\eta)$ is the number of connected components of η (including isolated vertices), and $Z_G^{p,q}$ is a normalizing constant. Note that omitting the factor $q^{k(\eta)}$, or taking $q = 1$, gives i.i.d. edges with probability p of being present. Other values of q give rise to dependence between edges. The random-cluster model is not only interesting in its own right, as a nice parametric model for random graphs with dependencies between edges, but also because it has proved to be an important and useful tool for studying Ising and Potts models. The definition implies that for any edge $e \in E$, and any configuration $\eta \in \{0, 1\}^{E \setminus \{e\}}$,

$$\mu_G^{p,q}(e \text{ is present} \,|\, \eta) \;=\; \begin{cases} p & \text{if the end-vertices of } e \\ & \text{are connected in } \eta \\ \frac{p}{p+(1-p)q} & \text{otherwise.} \end{cases} \qquad (3)$$

This conditional probability is increasing in η when $q \geq 1$, and decreasing when $q \leq 1$. Hence, the random-cluster model is monotone when $q \geq 1$ and anti-monotone when $q \leq 1$. The random-cluster model is reasonably well understood

in the monotone regime of the parameter space, but much less so in the anti-monotone regime (although see GRIMMETT (1995) and HÄGGSTRÖM (1995) for some results).

2. The algorithms

2.1. The Gibbs sampler

A building block of the Propp–Wilson algorithm, as well as of our modification, is the so called Gibbs sampler which we now go on to describe. It is the most famous and frequently used of all Markov chain Monte Carlo algorithms. Its name was introduced by GEMAN and GEMAN (1984), who used it for analysing Gibbs distributions on lattices. In statistical physics the method is also known as the heat bath algorithm.

As declared in the introduction, we are interested in drawing samples from π, where π is a probability measure on S^V. Let X be a random variable which takes its values in S^V, and which is distributed according to π. Suppose that we know $\pi(X(v) = i \,|\, X(V \setminus \{v\}) = \xi)$ for all $v \in V$, all $i \in S$ and all $\xi \in S^{V \setminus \{v\}}$ with $P(X(V \setminus \{v\}) = \xi) > 0$. (In all three examples of the previous sections these conditional distributions are readily available, as opposed to the joint distributions which involve the computationally intractable task of calculating the normalizing constant.) We can then make use of the Gibbs sampler, which is simply a discrete time Markov chain $\{X_t\}$ with state space S^V and evolution as follows. At each time step, a location to update is chosen from V. If at time t the chosen location was v, the new state of the chain, X_{t+1}, is obtained from the old, X_t, using the following rules:

$$\begin{array}{rcll} X_{t+1}(w) & = & X_t(w) & \text{for } w \neq v \\ X_{t+1}(v) & \stackrel{\mathcal{D}}{=} & \pi(\cdot \,|\, X_t(V \setminus \{v\})). & \end{array}$$

This can be realized concretely by letting U_t be an independent random variable uniformly distributed on $[0, 1]$, and setting

$$X_{t+1}(v) = \max\{s \in S : \pi(X(v) \geq s \,|\, X(V \setminus \{v\}) = \xi) \geq U_t\},$$

with $\xi = X_t(V \setminus \{v\})$. The resulting Markov chain $\{X_t\}$ is ergodic if it satisfies the following conditions:

(i) the set of elements of S^V that have positive π-measure is connected (i.e. can reach each other via successive coordinate changes) and

(ii) the choice of update location is sensible.

By sensible, we mean that all locations are visited every now and then. This can e.g. be accomplished by for each t choosing the location to update at random according to uniform distribution on V, or by sweeping through the set of locations one after the other in a deterministic order. We have chosen the former approach, partly because it simplifies notation as the Markov chain then becomes time homogeneous. It will be evident that the algorithms in the following subsections can be adapted to other updating schemes.

It is obvious that the Markov chain has π as its stationary distribution. Note that condition (i) is satisfied for the three examples mentioned in the introduction, for any choice of G.

2.2. Monotone coupling-from-the-past

We next describe the Propp–Wilson algorithm, which can be used to produce samples with the desired distribution π in cases where π is monotone in the sense described in the introduction.

The two main features, apart from the monotonicity, is the use of couplings and the fact that the chains are run from the past into the present. This is summarized in the term coupling-from-the-past. Each chain is a version of the Gibbs sampler above. How far away in the past they need to be started is determined by the algorithm itself.

The state space S^V contains two particular elements $\hat{0}$ and $\hat{1}$, with $\hat{0} \preceq \xi \preceq \hat{1}$ for every $\xi \in S^V$. For instance, in the Ising ferromagnet case, $\hat{0}$ is the state where all locations have spin -1 and $\hat{1}$ is the state where all locations have spin 1. Let $\{U_t\}_{t=-1,-2,\ldots}$ and $\{W_t\}_{t=-1,-2,\ldots}$ be independent i.i.d. sequences such that U_t is uniformly distributed on $[0,1]$ and W_t is uniformly distributed on V. If the chain at time t is in state $\xi \in S^V$, let the next state be given by $\phi(\xi, U_t, W_t)$, where the (deterministic) function $\phi:\ S^V \times [0,1] \times V \to S^V$ is defined by letting $\phi(\xi, u, w)$ equal ξ at all locations except w, where it gets value

$$\max\{s \in S : \pi(\, X(w) \geq s \,|\, X(V \setminus \{w\}) = \xi(V \setminus \{w\})\,) \geq u\}.$$

This yields transition probabilities identical to those of the Gibbs sampler in the previous subsection. The monotonicity of π now implies that if $\xi \preceq \xi'$, then $\phi(\xi, u, w) \preceq \phi(\xi', u, w)$ for any u and w.

The idea is to start one chain in $\hat{0}$ and one in $\hat{1}$, and to couple them by letting them evolve at each time subject to the same random variables U_t, W_t. For $t_1 < t_2 \leq 0$, define

$$\Phi_{t_1}^{t_2}(\xi, (\boldsymbol{u}, \boldsymbol{w})) = \phi(\phi(...(\phi(\xi, u_{t_1}, w_{t_1}), u_{t_1+1}, w_{t_1+1}), ..., u_{t_2-2}, w_{t_2-2}), u_{t_2-1}, w_{t_2-1})$$

where $(\boldsymbol{u}, \boldsymbol{w})$ is short for $((..., u_{-2}, u_{-1}), (..., w_{-2}, w_{-1}))$. By the monotonicity of π we have that if $\Phi_{t_1}^{t_2}(\hat{0}, (\boldsymbol{u}, \boldsymbol{w})) = \Phi_{t_1}^{t_2}(\hat{1}, (\boldsymbol{u}, \boldsymbol{w}))$, then $\Phi_{t_1}^{t_2}(\xi, (\boldsymbol{u}, \boldsymbol{w}))$ must also equal their common value for all $\xi \in S^V$, and the same thing must then consequently

hold for $\Phi_t^{t_2}(\xi, \boldsymbol{u}, \boldsymbol{w})$ whenever $t \leq t_1$. The Propp–Wilson algorithm consists of running these coupled Markov chains from time $-t$ to time 0 for larger and larger t until the two chains produce the same value at time 0, and to output this common value. PROPP and WILSON (1996) prove that this gives an unbiased sample from π; below, we shall give a slightly different proof of this fact (Theorem 2.1).

At first sight, one might think that the Propp–Wilson algorithm is unnecessarily complicated, and that one could instead run the two chains from time 0 up to the random time T when the two chains coalesce, and to output their common value at time T. Since the value at time T is then the same regardless of the starting value, we have in some sense "reached stationarity". If T were a fixed time, this would indeed yield unbiased samples from π, but since T is a random stopping time, this is in general not the case. See PROPP and WILSON (1996) for further discussion.

Let us now assume that π satisfies (i) and that $\hat{0}$ and $\hat{1}$ both have positive probability, and argue that we can then a.s. find a T (depending on $\boldsymbol{U} = (\ldots, U_{-2}, U_{-1})$ and $\boldsymbol{W} = (\ldots, W_{-2}, W_{-1})$), such that if the two chains are started at time $-T$, then they will have coalesced by time 0. The assumptions on π imply ergodicity of the Gibbs sampler, which in turn implies that there is an L such that for all states ξ and ξ', there is a positive chance of going from ξ to ξ' in L steps. Now we have for each t that $P(\, \Phi_{t-L}^t(\hat{0}, (\boldsymbol{U}, \boldsymbol{W})) = \Phi_{t-L}^t(\hat{1}, (\boldsymbol{U}, \boldsymbol{W}))\,)$ is positive. The events $\{\Phi_{-L}^0(\hat{0}, (\boldsymbol{U}, \boldsymbol{W})) = \Phi_{-L}^0(\hat{1}, (\boldsymbol{U}, \boldsymbol{W}))\}$, $\{\Phi_{-2L}^{-L}(\hat{0}, (\boldsymbol{U}, \boldsymbol{W})) = \Phi_{-2L}^{-L}(\hat{1}, (\boldsymbol{U}, \boldsymbol{W}))\}$, ..., are independent and they all have some positive probability $\epsilon > 0$ of occurring. Hence with probability 1 at least one of them will occur and this implies that a large enough T with the desired property can be found a.s.

We will now describe in more detail how the algorithm dynamically determines the value of T. At first the chains are started at time -1 and they are both updated once. If they are then in the same state, the procedure is ended and the common state is given as the output. If they are not equal, the two chains are started at time -2 and both are updated twice. At time 0 the current states of the chains are compared. If they have not coalesced by then, they are started anew at time $2 \cdot (-2) = -4$. This procedure is continued. Whenever time 0 is reached without the two chains being equal, the starting time is doubled. The procedure is ended when at some time the two chains are found equal. Their common value at time 0 is then given as output. Let T_* denote the smallest T for which $\Phi_{-T}^0(\hat{0}, (\boldsymbol{u}, \boldsymbol{w})) = \Phi_{-T}^0(\hat{1}, (\boldsymbol{u}, \boldsymbol{w}))$. For the sake of obtaining an unbiased sample, it does not matter at what time we start our chains, as long as it is earlier than $-T_*$. However, for the sake of not having to wait too long, one could wonder how well the method of successively trying $T = 1, 2, 4, 8, \ldots$ works. Propp and Wilson argue that this is not too far from optimal: Let us say that we have observed coalescence when starting at $-T$, where $T = 2^k$. The number

of simulation steps needed for one of the chains then equals $1 + 2 + 4 + ... + 2^k$, so the total amount of steps needed are $2(1 + 2 + 4 + ... + 2^k) < 2^{k+2}$. We know that T_* must exceed 2^{k-1}, otherwise we would not have had to increment T to 2^k. By this we see that the number of steps needed to verify that T_* is such that $\Phi^0_{-T_*}(\hat{0}, (\boldsymbol{u}, \boldsymbol{w})) = \Phi^0_{-T_*}(\hat{1}, (\boldsymbol{u}, \boldsymbol{w}))$, is at least $2 \cdot 2^{k-1} = 2^k$. Hence the proposed procedure comes within a factor 4 of the optimal one.

Propp and Wilson give the following pseudocode for the algorithm, which can be found in Figure 1.

$T \leftarrow 1$
repeat
    $upper \leftarrow \hat{1}$
    $lower \leftarrow \hat{0}$
    for $t = -T$ **to** -1
        $upper \leftarrow \phi(upper, u_t, w_t)$
        $lower \leftarrow \phi(lower, u_t, w_t)$
    $T \leftarrow 2T$
until $upper = lower$
return $upper$

Figure 1: Pseudocode for the monotone algorithm.

The random generation of the (u_t, w_t)'s is implicit in the code. An important thing to note is that it is essential that for each t the same choice of (u_t, w_t) is used each time the iterated repeated loop reaches t.

Let us finally give a proof that the algorithm does produce unbiased samples from π.

Theorem 2.1. *Suppose that the probability measure π on S^V is monotone, satisfies (i) above, and assigns positive probabilities to $\hat{0}$ and $\hat{1}$. Then the Propp–Wilson algorithm terminates a.s., and produces an unbiased sample from π.*

Proof. The a.s. termination of the algorithm has already been established, so it only remains to show that $\pi' = \pi$, where π' is defined as the distribution of the output. Recall the concept of distance in total variation $\|\mu - \mu'\|$ between two probability measures μ and μ' on S^V:

$$\|\mu - \mu'\| = \max_{E \subseteq S^V} |\mu(E) - \mu'(E)|.$$

Fix $\epsilon > 0$, and pick t so large that $P(T_* > t) < \epsilon$. Suppose that the random element $X \in S^V$ is picked according to π, independently of $(\boldsymbol{U}, \boldsymbol{W})$. Since π is invariant for the Gibbs sampler, we have that $\Phi^0_{-t}(X, (\boldsymbol{U}, \boldsymbol{W}))$ also has

distribution π. Furthermore, $\Phi^0_{-t}(X,(\boldsymbol{U},\boldsymbol{W})) = \Phi^0_{-T_*}(\hat{1},(\boldsymbol{U},\boldsymbol{W}))$ on the event that $T_* \le t$. Hence,

$$\begin{aligned}\|\pi - \pi'\| &\le P(\Phi^0_{-t}(X,(\boldsymbol{U},\boldsymbol{W})) \neq \Phi^0_{-T_*}(\hat{1},(\boldsymbol{U},\boldsymbol{W})))\\ &\le P(T_* > t)\\ &< \epsilon\end{aligned}$$

and since $\epsilon > 0$ was arbitrary, the proof is complete. ■

2.3. Anti-monotone coupling-from-the-past

In this subsection, we show how the coupling-from-the-past approach of Propp and Wilson can be adapted to the anti-monotone setting. The Propp–Wilson algorithm relies crucially on the fact that the mapping ϕ respects the ordering $\preceq$, i.e. that

$$\phi(\xi,u,w) \preceq \phi(\xi',u,w) \quad \text{whenever} \quad \xi \preceq \xi'.$$

This holds when π is monotone, but not e.g. when π is anti-monotone.

The idea for anti-monotone systems is to have the two "chains" look at each other's configurations on $V\setminus\{w\}$, rather than their own, when updating the value at w. A similar exchange of configurations appears in an algorithm of Kendall (1996) for simulation of a certain class of spatial point processes, and Kendall's work was in fact the main source of inspiration for the present paper. We write "chains" in quotation marks for this variant of coupling-from-the-past, because although the two processes together form a Markov chain, they interact in such a way that the Markov property is lost when one of them is viewed separately.

In order to make this more precise, we first generalize ϕ to the function

$$\tilde{\phi}: S^V \times S^V \times [0,1] \times V \to S^V$$

by letting $\tilde{\phi}(\xi,\xi',u,w)$ equal ξ at all locations except at w, where it gets value

$$\max\{s \in S : \pi(X(w) \ge s \,|\, X(V\setminus\{w\}) = \xi'(V\setminus\{w\})) \ge u\}.$$

Note that $\tilde{\phi}(\xi,\xi,u,w) = \phi(\xi,u,w)$. Define the S^V-valued function $\tilde{\Phi}^{t_2}_{t_1}$ inductively by letting

$$\tilde{\Phi}^{t_1}_{t_1}(\xi,\xi',(\boldsymbol{u},\boldsymbol{w})) = \xi$$

and, for $k = 1,2,\ldots$,

$$\begin{aligned}&\tilde{\Phi}^{t_1+k}_{t_1}(\xi,\xi',(\boldsymbol{u},\boldsymbol{w})) =\\ &\quad\tilde{\phi}\left(\tilde{\Phi}^{t_1+k-1}_{t_1}(\xi,\xi',(\boldsymbol{u},\boldsymbol{w})), \tilde{\Phi}^{t_1+k-1}_{t_1}(\xi',\xi,(\boldsymbol{u},\boldsymbol{w})), u_{t_1+k-1}, w_{t_1+k-1}\right).\end{aligned}$$

The anti-monotone coupling-from-the-past algorithm now consists of generating the (u_t,w_t)'s randomly in the exact same manner as in monotone coupling-from-the-past, and computing $\tilde{\Phi}^0_{-T}(\hat{1},\hat{0},(\boldsymbol{u},\boldsymbol{w}))$ and $\tilde{\Phi}^0_{-T}(\hat{0},\hat{1},(\boldsymbol{u},\boldsymbol{w}))$ for larger and

larger values of T until the two functions give the same answer. Their common value is then given as output. The algorithm can be found in pseudocode in Figure 2.

$T \leftarrow 1$
repeat
    $upper \leftarrow \hat{1}$
    $lower \leftarrow \hat{0}$
    for $t = -T$ **to** -1
        $upper' \leftarrow upper$
        $lower' \leftarrow lower$
        $upper \leftarrow \tilde{\phi}\ (upper',\ lower', u_t, w_t)$
        $lower \leftarrow \tilde{\phi}\ (lower',\ upper', u_t, w_t)$
    $T \leftarrow 2T$
until $upper = lower$
return $upper$

Figure 2: Pseudocode for the anti-monotone algorithm.

One may want to apply anti-monotone coupling-from-the-past to systems where not all elements of S^V have positive π-measure, such as the hard-core model. It is then necessary to define the conditional probability $\pi(\cdot \,|\, X(V \setminus \{v\}) = \xi)$ also for configurations $\xi \in S^{V \setminus \{v\}}$ that have zero probability under π. (For monotone coupling-from-the-past we did not need to worry about this, because each chain is a realization of the Gibbs sampler, which never leaves the set of configurations that have positive π-measure.) For such $\xi \in S^{V \setminus \{v\}}$, we need to define $\pi(\cdot \,|\, X(V \setminus \{v\}) = \xi)$ in such a way that $\pi(X(v) \leq s \,|\, X(V \setminus \{v\}) = \xi)$ still is increasing in ξ, and it seems natural to set

$$\pi(X(v) \leq s \,|\, X(V \setminus \{v\}) = \xi) \;=\; \max_{\xi'} \pi(X(v) \leq s \,|\, X(V \setminus \{v\}) = \xi')$$

where the maximum is taken over all $\xi' \in S^{V \setminus \{v\}}$ which have positive π-probability and satisfy $\xi' \preceq \xi$. For the case of the hard-core model, this means that (1) extends to non-feasible ξ.

Conditions under which we can guarantee convergence of the algorithm are given in the following theorem.

Theorem 2.2. *Suppose that π is a probability measure on S^V which is anti-monotone and which assigns positive probability to at least one of the elements $\hat{0}, \hat{1} \in S^V$. Then the anti-monotone coupling-from-the-past algorithm terminates a.s., and produces an unbiased sample from π.*

To prove this result, we shall make use of the following lemma, which contains the essence of the exploitation of the anti-monotonicity structure. An alternative proof of Theorem 2.2 in a more general situation can be found in MØLLER (1997).

Lemma 2.3. *Let π be anti-monotone, and let $\xi_1, \xi_1', \xi_2, \xi_2' \in S^V$ be such that $\xi_1 \preceq \xi_2$ and $\xi_1' \succeq \xi_2'$. Then, for each t_1, t_2 and $(\boldsymbol{u}, \boldsymbol{w})$,*

$$\tilde{\Phi}_{t_1}^{t_2}(\xi_1, \xi_1', (\boldsymbol{u}, \boldsymbol{w})) \preceq \tilde{\Phi}_{t_1}^{t_2}(\xi_2, \xi_2', (\boldsymbol{u}, \boldsymbol{w})).$$

Proof. Set $k = t_2 - t_1$. For $k = 0$ the result is trivial. For $k = 1$, the assertion is that

$$\tilde{\phi}(\xi_1, \xi_1', u_{t_1}, w_{t_1}) \preceq \tilde{\phi}(\xi_2, \xi_2', u_{t_1}, w_{t_1}).$$

That this domination holds at all locations $v \in V \setminus \{w_{t_1}\}$ is immediate from the assumption that $\xi_1 \preceq \xi_2$, whereas the domination at w_{t_1} follows from the assumptions that $\xi_1' \succeq \xi_2'$ and that π is anti-monotone. This proves the result for $k = 1$. Note also that

$$\tilde{\phi}(\xi_1', \xi_1, u_{t_1}, w_{t_1}) \succeq \tilde{\phi}(\xi_2', \xi_2, u_{t_1}, w_{t_1}).$$

by the same argument. The cases $k = 2, 3, \ldots$ now follow by induction.

■

Proof of Theorem 2.2. Note that $\hat{0}$ and $\hat{1}$ play completely symmetric roles in the algorithm, and that it therefore is sufficient to prove the theorem for the case $\pi(\hat{0}) > 0$. Lemma 2.3 implies that if

$$\tilde{\Phi}_{-T}^{0}(\hat{0}, \hat{1}, (\boldsymbol{u}, \boldsymbol{w})) = \tilde{\Phi}_{-T}^{0}(\hat{1}, \hat{0}, (\boldsymbol{u}, \boldsymbol{w})) \tag{4}$$

then we have for any ξ, ξ' that $\tilde{\Phi}_{-T}^{0}(\xi, \xi', (\boldsymbol{u}, \boldsymbol{w}))$ equals their common value. The same thing consequently holds for $\tilde{\Phi}_{-t}^{0}(\xi, \xi', (\boldsymbol{u}, \boldsymbol{w}))$ whenever $t \geq T$. Hence, the algorithm terminates whenever we can find a T for which (4) holds. Write s_{min} for the minimal element of S. For $W \subseteq V$, write s_{min}^W for the element of S^W where every $v \in W$ has the value s_{min}, so that in particular $\hat{0} = s_{min}^V$. Since $\pi(\hat{0}) > 0$, we have for each $v \in V$ that the conditional π-probability that $X(v) = s_{min}$ given the event that $X(V \setminus \{v\}) = s_{min}^{V \setminus \{v\}}$ is positive. By anti-monotonicity, we thus have

$$\pi(X(v) = s_{min} \mid X(V \setminus \{v\}) = \xi) > 0$$

for every $v \in V$ and $\xi \in S^{V \setminus \{v\}}$. It follows that we can find an L (taking L equal to the cardinality of V suffices) and an $\epsilon > 0$ such that the two processes, starting from arbitrary states, have probability at least ϵ of both reaching state $\hat{0}$ after L time units. The a.s. existence of a T such that (4) holds (and hence the a.s. termination of the algorithm) now follows by the same argument as in the monotone case.

It remains to show that $\pi' = \pi$, where π' is the distribution of the output of the algorithm. Write T_* for the smallest T for which (4) holds with $(\boldsymbol{u}, \boldsymbol{w}) = (\boldsymbol{U}, \boldsymbol{W})$. Fix $\epsilon > 0$, and pick t so large that $P(T_* > t) < \epsilon$. Suppose that the random element $X \in S^V$ is picked according to π, independently of $(\boldsymbol{U}, \boldsymbol{W})$. Since $\tilde{\phi}(\xi, \xi, u, w) = \phi(\xi, u, w)$ and π is invariant under the $\phi(\xi, U, W)$ evolution, we have that $\tilde{\Phi}^0_{-t}(X, X, (\boldsymbol{U}, \boldsymbol{W}))$ is distributed according to π. We have already argued that $\tilde{\Phi}^0_{-t}(X, X, (\boldsymbol{U}, \boldsymbol{W})) = \tilde{\Phi}^0_{-T_*}(\hat{0}, \hat{1}, (\boldsymbol{U}, \boldsymbol{W}))$ on the event that $T_* \leq t$, whence

$$\begin{aligned} \|\pi - \pi'\| &\leq P(\tilde{\Phi}^0_{-t}(X, X, (\boldsymbol{U}, \boldsymbol{W})) \neq \tilde{\Phi}^0_{-T_*}(\hat{0}, \hat{1}, (\boldsymbol{U}, \boldsymbol{W}))) \\ &\leq P(T_* > t) \\ &< \epsilon \end{aligned}$$

and since $\epsilon > 0$ was arbitrary we have that $\|\pi - \pi'\| = 0$, and we are done.

■

3. Markov random fields on bipartite graphs

In this section we will show that for a certain class of systems, namely anti-monotone Markov random fields on bipartite graphs, anti-monotone coupling-from-the-past can be viewed as "isomorphic" to monotone coupling-from-the-past for a related class of systems.

A bipartite graph is a graph G whose vertex set V can be partitioned into two sets $(V_{\text{even}}, V_{\text{odd}})$ in such a way that each edge of G has one of its vertices in V_{even} and the other in V_{odd}. We will now give a formal definition of a Markov random field and for that we need the concept of a boundary of a set. For $W \subseteq V$, we define the boundary of W to be the set

$$\partial W = \{v \in V \setminus W \,:\, \exists w \in W \text{ such that } v \sim w\}.$$

Definition 3.1. *Suppose G is a finite graph with vertex set V and that S is a finite set. Let X be a random element taking its values in S^V. Now X is said to be a **Markov random field** on G if for each $W \subseteq V$, the conditional distribution of $X(W)$ given $X(V \setminus W)$ depends on $X(V \setminus W)$ only through its values on ∂W. That is to say, if π is the underlying probability measure, X is a Markov random field if for all $\omega \in S^W$, $\omega' \in S^{V \setminus W}$ and $\omega'' \in S^{\partial W}$ such that ω'' is the restriction of ω' to ∂W and $\pi(X(V \setminus W) = \omega') > 0$, we have*

$$\pi(X(W) = \omega \mid X(V \setminus W) = \omega') \;=\; \pi(X(W) = \omega \mid X(\partial W) = \omega'').$$

We will allow ourselves to identify the system with its underlying probability measure, in saying that π is a Markov random field, when π is the probability measure of a Markov random field.

Let us suppose that the set $S \subseteq \mathbb{R}$ of attainable values is symmetric about zero. We do not lose anything in generality by this assumption, since any finite S can be given this property by a monotone transformation. Let X_a be an S^V-valued random variable and let π_a be the corresponding probability measure. Suppose that π_a is anti-monotone and in addition that X_a is a Markov random field. We introduce a function $\varphi : S^W \to S^W$, where W is any subset of V, with the interpretation that if $\xi' = \varphi(\xi)$, then

$$\xi'(v) = \begin{cases} \xi(v) & v \in V_{\text{even}} \\ -\xi(v) & v \in V_{\text{odd}}. \end{cases}$$

Note that $\varphi(\varphi(\xi)) = \xi$.

Let us construct a new S^V-valued random variable X_m from X_a by letting $X_m = \varphi(X_a)$. We will now show that the distribution π_m of X_m is monotone.

Suppose that $\xi \preceq \eta$ and let v be an even vertex (i.e. $v \in V_{\text{even}}$). We get

$$\begin{aligned} &\pi_m(X_m(v) \le s \,|\, X_m(V \setminus \{v\}) = \xi) \\ &\qquad = \pi_a(X_a(v) \le s \,|\, X_a(V \setminus \{v\}) = \varphi(\xi)) \\ &\qquad \ge \pi_a(X_a(v) \le s \,|\, X_a(V \setminus \{v\}) = \varphi(\eta)) \qquad (5) \\ &\qquad = \pi_m(X_m(v) \le s \,|\, X_m(V \setminus \{v\}) = \eta). \end{aligned}$$

The conditional distribution of the value at location v only depends on the status of the neighbours of v, which all sit in V_{odd}. The fact that π_a is anti-monotone in combination with $\varphi(\eta_{\text{odd}}) \preceq \varphi(\xi_{\text{odd}})$, explain the inequality above. η_{odd} and ξ_{odd} are to be interpreted as the projections on $S^{V_{\text{odd}}}$ of the configurations η and ξ respectively.

Similarly if v is an odd vertex we get

$$\begin{aligned} \pi_m(X_m(v) \le s \,|\, X_m(V \setminus \{v\}) = \xi) &= \pi_a(X_a(v) \ge -s \,|\, X_a(V \setminus \{v\}) = \varphi(\xi)) \\ &= 1 - \pi_a(X_a(v) < -s \,|\, X_a(V \setminus \{v\}) = \varphi(\xi)) \\ &\ge 1 - \pi_a(X_a(v) < -s \,|\, X_a(V \setminus \{v\}) = \varphi(\eta)) \\ &= \pi_m(X_m(v) \le s \,|\, X_m(V \setminus \{v\}) = \eta) \end{aligned}$$

whenever $\xi \preceq \eta$. The inequality is explained by the same argument as above, on noting that $\varphi(\xi_{\text{even}}) \preceq \varphi(\eta_{\text{even}})$. Here ξ_{even} and η_{even} have the obvious meaning. Since s was arbitrary, we have shown that $\pi_m(X_m(v) \le s \,|\, X_m(V \setminus \{v\}) = \xi)$ is decreasing in ξ for all v and all s, and thereby that π_m is monotone.

Examples of anti-monotone Markov random fields on bipartite graphs are the hard-core model and the Ising antiferromagnet on the cubic lattice. An interesting thing to note is that by flipping the values on the odd lattice of an antiferromagnet system, an Ising ferromagnet is obtained.

The fact that π_m is monotone means that if we want to sample from π_a we have the following alternative to anti-monotone coupling-from-the-past: Run

monotone coupling-from-the-past on π_m and expose the output to φ. An obvious question now is which of the procedures is preferable as far as running time is concerned. The answer is that it does not matter which one we choose! We formulate this loosely as follows:

> *Applying the anti-monotone algorithm to an anti-monotone Markov random field π_a on a bipartite graph gives the same result, and takes equally long time, as applying the monotone Propp–Wilson algorithm to the system π_m that is obtained from π_a by flipping the values of the odd vertices.*

To make this more precise, let us couple the two algorithms by at each time step subjecting them to the same random variables U_t and W_t. Let $\tilde{\Phi}$ be defined as in Section 2.3 with π_a as the underlying probability measure. Under the assumptions of Theorem 2.2 we know that there a.s. exists a T_* such that

$$\tilde{\Phi}^0_{-T_*}(\hat{0}, \hat{1}, (\boldsymbol{u}, \boldsymbol{w})) \;=\; \tilde{\Phi}^0_{-T_*}(\hat{1}, \hat{0}, (\boldsymbol{u}, \boldsymbol{w})). \tag{6}$$

Under the assumptions of Theorem 2.1, with Φ defined as in Section 2.2 with respect to π_m, we also know that a.s. a T'_* can be found, for which

$$\Phi^0_{-T'_*}(\hat{0}, (\boldsymbol{u}, \boldsymbol{w})) \;=\; \Phi^0_{-T'_*}(\hat{1}, (\boldsymbol{u}, \boldsymbol{w})) \tag{7}$$

holds. We want to show that $T_* = T'_*$ and that the final state in (6) equals that of (7), when the latter has the values on V_{odd} flipped.

We introduce the random variables $X_t^{m,\,\text{upper}}$, $X_t^{m,\,\text{lower}}$, $X_t^{a,\,\text{upper}}$ and $X_t^{a,\,\text{lower}}$ which will represent, from left to right, the values of the upper and lower elements of the monotone algorithm at time t and the values of the upper and lower elements of the anti-monotone algorithm at time t. We note that at each starting time t the following equalities hold:

$$\begin{aligned} X^{m,\,\text{upper}}_{t,\,\text{even}} &= X^{a,\,\text{upper}}_{t,\,\text{even}} \\ X^{m,\,\text{upper}}_{t,\,\text{odd}} &= \varphi(X^{a,\,\text{lower}}_{t,\,\text{odd}}) \\ X^{m,\,\text{lower}}_{t,\,\text{even}} &= X^{a,\,\text{lower}}_{t,\,\text{even}} \\ X^{m,\,\text{lower}}_{t,\,\text{odd}} &= \varphi(X^{a,\,\text{upper}}_{t,\,\text{odd}}). \end{aligned} \tag{8}$$

The crucial observation is now that if these equalities hold at time t, this must also be the case at time $t+1$. To see this, we look at what happens when the four variables are updated. Each vertex in $X^{m,\,\text{upper}}_{t,\,\text{even}}$ looks at its neighbours, which sit in $X^{m,\,\text{upper}}_{t,\,\text{odd}}$, and each vertex in $X^{a,\,\text{upper}}_{t,\,\text{even}}$ looks at the neighbours of the corresponding vertex in $X_t^{a,\,\text{lower}}$. These neighbours sit in the odd lattice, $X^{a,\,\text{lower}}_{t,\,\text{odd}}$. Now since we have that $X^{m,\,\text{upper}}_{t,\,\text{odd}} = \varphi(X^{a,\,\text{lower}}_{t,\,\text{odd}})$ and we subject Φ and $\tilde{\Phi}$ to the same values of (U_t, W_t), we get that $X^{m,\,\text{upper}}_{t+1,\,\text{even}} = X^{a,\,\text{upper}}_{t+1,\,\text{even}}$. Here we used the first

equality of (5). Similar arguments in the three other cases lead to the conclusion that the equalities of (8) of time t must also be true at time $t+1$.

Having thus convinced ourselves that the equalities remain throughout the evolution of time, we observe that (8) further implies that

$$X_t^{m,\,\mathrm{upper}} = X_t^{m,\,\mathrm{lower}} \quad \text{if and only if} \quad X_t^{a,\,\mathrm{upper}} = X_t^{a,\,\mathrm{lower}},$$

and we are done.

One implication of all this is that in the bipartite Markov random field case, results proved for monotone systems concerning e.g. speed of convergence (see Propp and Wilson (1996)), translate directly to anti-monotone systems.

4. Some simulation results

We will now show how the algorithm performs on some instances of the hard-core model and of the random-cluster model. We have chosen not to include any simulations of the Ising antiferromagnet. Readers interested in this particular example may turn to Propp and Wilson (1996), where the Ising ferromagnet is studied, and translate the results to the antiferromagnetic case using the methods of Section 3.

The implementation of the algorithm was done in the programming language *C*. In all our simulations we have used square portions of $\mathbb{Z}^2$ (with free boundary, i.e. without the torus convention) as the graph G, and we refer to the side-length of the square as the size of the system. Our estimates of mean running time are all based on 20 replicates and we measure running time in number of iterations of the inner loop in the pseudocode needed to reach coalescence.

When our aim has been to investigate the time to coalescence for different parameter values, we have used the naive "coupling-to-the-future" variant of the algorithm, mentioned in Section 2.2. Instead of starting further and further back in time until coalescence is reached at time 0, we start at time 0 and run forward until coalescence occur. Recall that T_* is the smallest t such that $\tilde{\Phi}^0_{-t}(\hat{0},\hat{1},(\boldsymbol{U},\boldsymbol{W})) = \tilde{\Phi}^0_{-t}(\hat{1},\hat{0},(\boldsymbol{U},\boldsymbol{W}))$ holds. Let T^* denote the smallest t such that $\tilde{\Phi}^t_0(\hat{0},\hat{1},(\boldsymbol{U},\boldsymbol{W})) = \tilde{\Phi}^t_0(\hat{1},\hat{0},(\boldsymbol{U},\boldsymbol{W}))$ holds. The actual number of iterations needed in the algorithm comes within a factor 4 of T_*, as argued in Section 2.2. It is easy to see that T_* and T^* are governed by the same probability distribution (see Propp and Wilson (1996)) and hence we can concentrate on T^* when studying running time. The pictures of the various configurations were, however, of course accomplished using the correct coupling-from-the-past protocol.

4.1. Hard-core examples

We have studied the behaviour of our algorithm for the hard-core model on square subsets of $\mathbb{Z}^2$ of various size and with various choices of the activity parameter

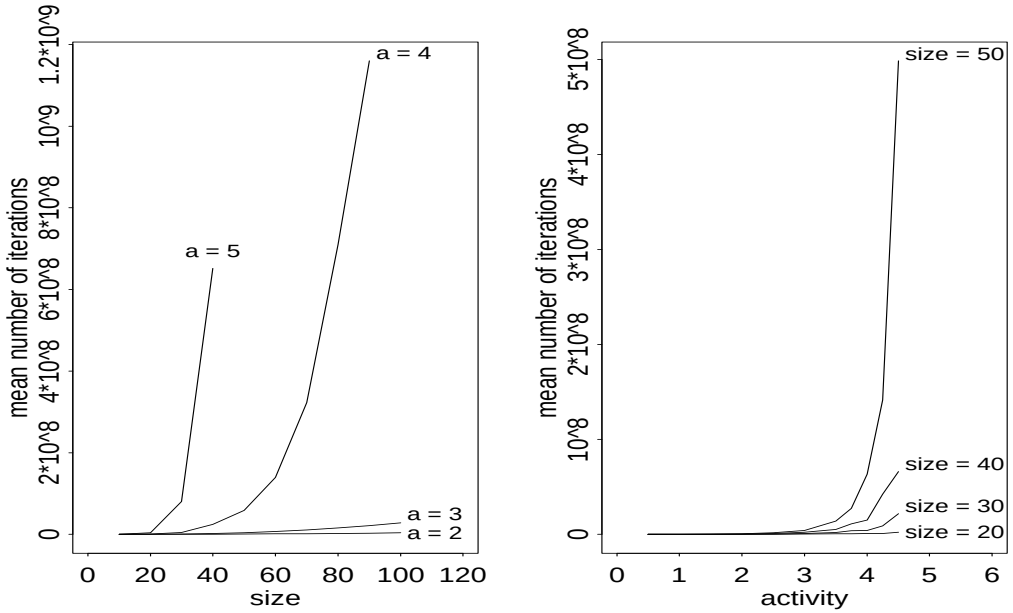

Figure 3: Running times for the 4-neighbour hard-core model. Left diagram: Mean number of iterations versus size. Right diagram: Mean number of iterations versus activity.

a. As the diagrams in Figure 3 clearly demonstrate, the running time of the algorithm depends heavily on both the size of the system and of the value of the activity. That the running time should increase with the size seems obvious. To get some intuition for the dependence on the activity, we can look at formula (1) for the conditional probability of a vertex having value 1 given the values of its neighbours. As the activity gets higher, there is at each vertex a greater probability of having the value 1, given that all neighbours have value 0. This results in a high level of dependence between the values of the vertices — if there is a 1 at a vertex in let us say the even lattice, then all its neighbours, which sit in the odd lattice, of course must have value 0 and moreover it is very likely that there will be 1's at many other vertices in the even lattice even some distance away from the first vertex. Figure 4 demonstrates this phenomenon. The value of the activity is 1 in the left picture and 4 in the right picture. Both simulations were carried out on a 50×50 lattice. The longer running times for large a may also be related to the phase transition phenomenon which occurs for large a, see e.g. Georgii (1988).

In the systems studied so far, every interior vertex had 4 neighbours (north, south, east and west). This makes the graph bipartite, so these 4-neighbour systems can be translated into monotone systems by the arguments in Section 3. To

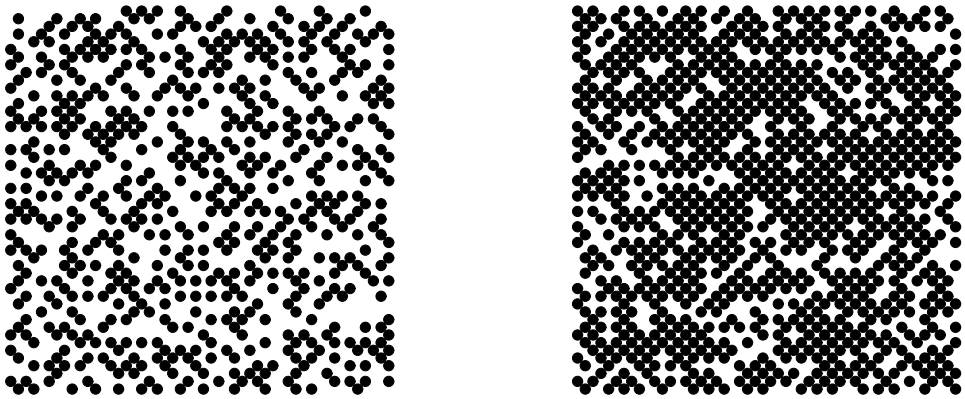

Figure 4: The 4-neighbour hard-core model on a 50×50 lattice. The value of the activity is 1 (left diagram) and 4 (right diagram).

see what happens in a genuinely anti-monotone situation we have also studied systems where diagonal edges are included so that every vertex in the interior of the set have 8 neighbours; the bipartite structure of the lattice is then removed. This turned out to be much more demanding as far as running time is concerned. We only managed to simulate 20×20 lattices and at relatively low values of activity, see Figure 5. That it should take longer for the algorithm to terminate in the case of 8 neighbours seems natural enough, since the graph in this case has a higher degree of connectivity.

4.2. Random-cluster examples

Another example of a genuinely anti-monotone system is the $q < 1$ random-cluster model, since it lacks the Markov random field property necessary for the arguments in Section 3. We have chosen to simulate the random-cluster model for values of p and q on the so called self-dual curve $p = \frac{\sqrt{q}}{1+\sqrt{q}}$, see Welsh (1993). If we let $p' = \frac{p}{p+(1-p)q}$, we have that $p + p' = 1$ for these values of p and q. Recall from (3) that p' is the conditional probability of an edge being present given that its two end-vertices are not connected in the configuration and that p is the conditional probability of an edge being present given that the two end-vertices are connected. By means of the self-duality of the quadratic lattice (well known in percolation theory; see Grimmett (1989)), one can argue intuitively that approximately half of the edges will be present in these graphs, although with different dependence structures for the different values of p and q.

Let us now study Figure 6. It contains four examples of random-cluster models

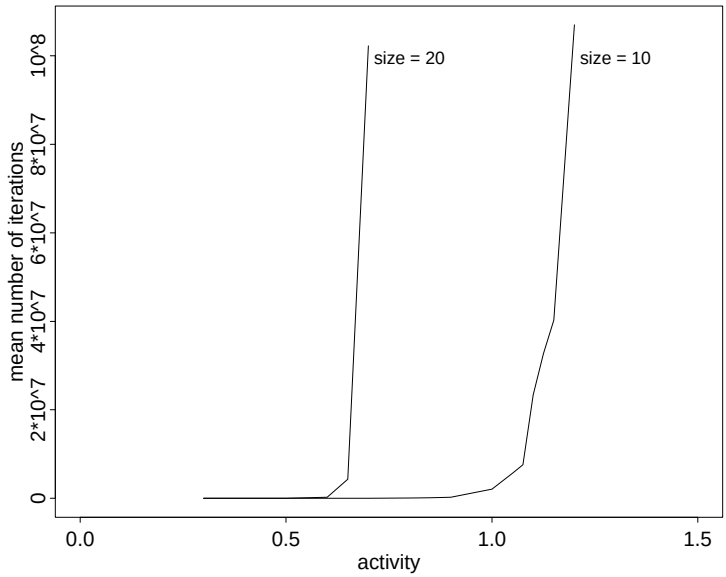

Figure 5: Mean number of iterations versus activity for the 8-neighbour hard-core model on lattices of size 20 and 10 respectively.

on a 70×70 lattice. The upper left configuration is an instance of a monotone random-cluster model with $q = 4$ and $p = \frac{2}{3}$. It shows clear signs of clustering. This is explained by the fact that values of q greater than 1 favour configurations with a high number of connected components, or alternatively by the fact that the conditional probability in (3) that an edge e is present tends to be larger the more edges around e that are present. In the upper right configuration the edges are present independently of each other, the parameter values being $q = 1$ and $p = \frac{1}{2}$. This is simply independent bond percolation with bond-probability $\frac{1}{2}$. The lower left configuration is anti-monotone with $q = 0.6$ and $p = \frac{\sqrt{0.6}}{1+\sqrt{0.6}}$. This was the lowest value of q we managed to simulate on the given size of lattice. The structure is notably more ordered than in the two configurations above. There is even more order in the last of the four pictures. It is a realization from the uniform spanning tree measure, which arises in the limit of $q \to 0$ on the self-dual curve (see HÄGGSTRÖM (1995)). The configuration was produced using the famous random walk algorithm described e.g. in PEMANTLE (1995) and implemented by SEGERBERG (1994).

As opposed to the case of the hard-core model, the evaluation of the conditional probabilities in the random-cluster model is computationally non-trivial. We have used a slightly simplified version of an algorithm by SWEENY (1983) to determine the connectivity necessary to evaluate (3). The algorithm relies on the

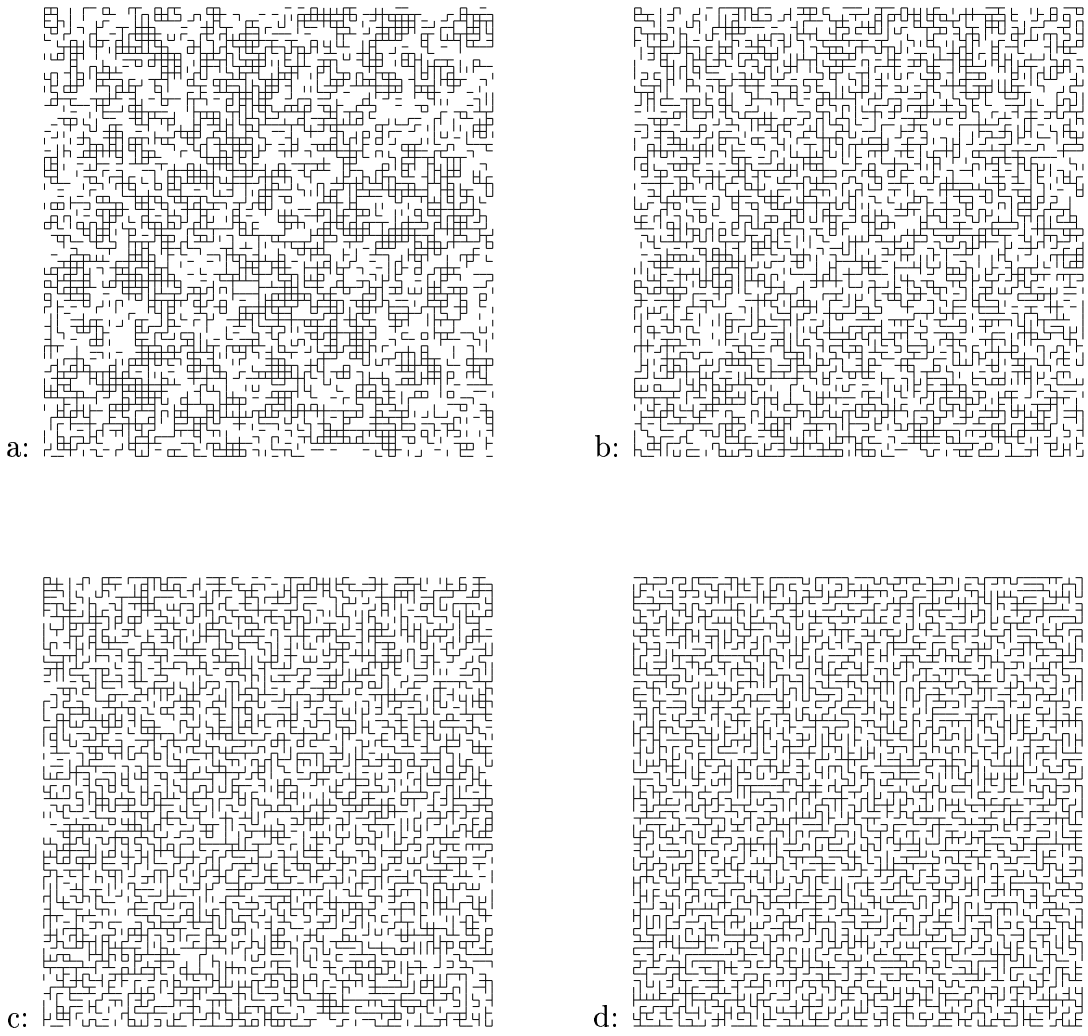

Figure 6: Random-cluster configurations for different values of q and $p = \frac{\sqrt{q}}{1+\sqrt{q}}$. a: $q = 4$, b: $q = 1$, c: $q = 0.6$, d: $q \to 0$.

fact that we are dealing with $\mathbb{Z}^2$, since it uses the dual lattice. Each connected component in the ordinary lattice is surrounded by a loop in the dual lattice and each of these loops is given a name. To determine if the two end-vertices of an edge chosen to update are connected, only a comparison of names is needed. However, if the algorithm decides to change the presence of the edge, new loops may develop or disappear and new names will be needed. The process of changing names takes different amounts of time depending on the length of the loops in the dual lattice, that is the size of the components in the ordinary lattice. Thus the time used for each iteration of the loop may differ. Nevertheless, we have only recorded the number of iterations used for the algorithm to terminate, and not the actual time used. This is because we feel that only the number of iterations of the loop has general relevance to the Propp–Wilson algorithm.

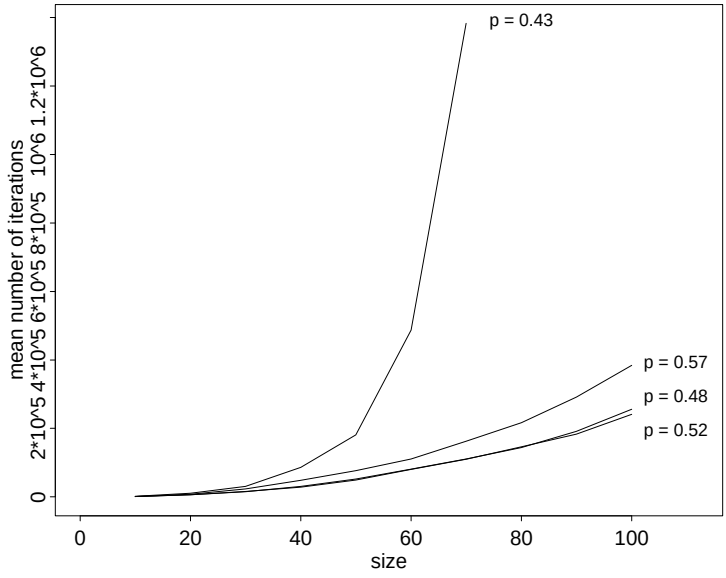

Figure 7: Plot of mean number of iterations versus the size for different parameter values on the self-dual curve $p = \frac{\sqrt{q}}{1+\sqrt{q}}$.

Figure 7 plots observed mean running "time" versus size for four pairs of p and p' values: *1a)* $p = 0.43$ and $p' = 0.57$, *1b)* $p = 0.57$ and $p' = 0.43$, *2a)* $p = 0.48$ and $p' = 0.52$ and *2b)* $p = 0.52$ and $p' = 0.48$. The systems *1a)* and *2a)* are anti-monotone and the other two are monotone. *A priori*, one might think that *1a)* and *1b)* would be about equally difficult to simulate, as would *2a)* and *2b)*. However, as Figure 7 shows, the anti-monotone case is more

demanding, the effect being hardly noticeable for p and p' taking values 0.48 and 0.52, but increasing drastically as p and p' move away from 0.5. We do not know whether this reflects some general fact about anti-monotone simulation being more difficult than monotone, or if it is just a particular feature of the random-cluster model.

4.3. Concluding remarks

There is almost certainly room for refinement of our methods. In most cases a very large number of iterations were needed to reach coalescence. One question that arises is if the running time can be reduced by using some other method of choosing the location to update, e.g. to use a deterministic sweeping scheme rather than selecting the location at random. Another potential speedup of the algorithm might be to use the ideas of MØLLER (1997), where the upper process is started in some "dominating chain" rather than in the maximal state.

References

ASMUSSEN, S., P.W. GLYNN and H. THORISSON (1992), Stationary detection in the initial transient problem, *ACM Transactions on Modeling and Computer Simulation* **2**, 130–157.

VAN DEN BERG, J. (1993), A uniqueness condition for Gibbs measures, with application to the 2-dimensional Ising antiferromagnet, *Communications in Mathematical Physics* **152**, 161–166.

VAN DEN BERG, J. and J. STEIF (1994), Percolation and the hard core lattice gas model, *Stochastic Processes and their Applications* **49**, 179–197.

FILL, J.A. (1997) An interruptible algorithm for perfect sampling via Markov chains, *Annals of Applied Probability*, to appear.

FORTUIN, C.M. and P.W. KASTELEYN (1972), On the random-cluster model. I. Introduction and relation to other models, *Physica* **57**, 536–564.

GELMAN, A. and D.B. RUBIN (1992), Inference from iterative simulation using multiple sequences (with discussion), *Statistical Science* **7**, 457–472 and 483–511.

GEMAN, S. and D. GEMAN (1984), Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **6**, 721–741.

GEORGII, H-O. (1988), *Gibbs Measures and Phase Transitions*, de Gruyter, New York.

GEYER, C.J. (1992), Practical Markov chain Monte Carlo (with discussion), *Statistical Science* **7**, 473–511.

GILKS, W.R., S. RICHARDSON and D.J. SPIEGELHALTER (1996), *Markov Chain Monte Carlo in Practice*, Chapman and Hall, London.

GRIMMETT, G. (1989), *Percolation*, Springer, New York.

GRIMMETT, G. (1995), The stochastic random-cluster process, and the uniqueness of random-cluster measures, *Annals of Probability* **23**, 1461–1510.

HÄGGSTRÖM, O. (1995), Random-cluster measures and uniform spanning trees, *Stochastic Processes and their Applications* **59**, 267–275.

HÄGGSTRÖM, O. (1996), Random-cluster representations in the study of phase transitions, preprint.

HÄGGSTRÖM, O., M.N.M. VAN LIESHOUT and J. MØLLER (1997), Characterisation results and Markov chain Monte Carlo algorithms including exact simulation for some spatial point processes, *Bernoulli*, to appear.

KELLY, F.P. (1985), Stochastic models of computer communication systems, *Journal of the Royal Statistical Society, Series B* **47**, 379–395.

KENDALL, W. (1996), Perfect simulation for the area-interaction point process, *Proceedings of the Symposium on Probability towards the year 2000*, (L. Accardi and C. Heyde, eds), Springer, to appear.

KENDALL, W. and J. MØLLER (1997), Perfect Metropolis–Hastings simulation of locally stable spatial point processes, in preparation.

LIGGETT, T.M. (1985), *Interacting Particle Systems*, Springer, New York.

LINDVALL, T. (1992), *Lectures on the Coupling Method*, Wiley, New York.

LOVÁSZ, L. and P. WINKLER (1995), Exact mixing in an unknown Markov chain, *Electronic Journal of Combinatorics* **2**, paper #R15.

METROPOLIS, N., A.W. ROSENBLUTH, M.N. ROSENBLUTH, A.H. TELLER and E. TELLER (1953), Equations of state calculations by fast computing machine, *Journal of Chemical Physics* **21**, 1087–1091.

MURDOCH, D.J. and P.J. GREEN (1997), Exact Sampling from a Continuous State Space, *Scandinavian Journal of Statistics*, to appear.

MØLLER, J. (1997), A note on perfect simulation of conditionally specified models. Research Report R-97-2006, Departement of Mathematics, Aalborg University.

PEMANTLE, R. (1995), Uniform random spanning trees, *Topics in Contemporary Probability and its Applications* (ed. J.L. Snell), 1–54, CRC, Boca Raton.

PROPP, J.G. and D.B. WILSON (1996), Exact sampling with coupled Markov chains and applications to statistical mechanics, *Random Structures and Algorithms* **9**, 223–252.

PROPP, J.G. and D.B. WILSON (1997), How to get a perfectly random sample from a generic Markov chain and generate a random spanning tree of a directed graph, *Journal of Algorithms*, to appear.

SEGERBERG, U. (1994), Generating uniform spanning trees from random walks, Master's thesis, Department of Mathematics, Chalmers University of Technology.

SWEENY, M. (1983), Monte Carlo study of weighted percolation clusters relevant to the Potts model, *Physical Review B* **27**, 4445–4455.

WELSH, D.J.A. (1993), Percolation in the random cluster process and Q-state Potts model, *Journal of Physics A: Mathematical and General* **26**, 2471–2483.