Auxiliary Variables for Markov Random Fields with Higher Order Interactions

Robin D. Morris[1]

RIACS, NASA Ames Research Center, MS 269-2, Moffett Field, CA 94035 USA**
rdm@ptolemy.arc.nasa.gov
Tel: +1 650 604 0158 Fax: +1 650 604 3594

Abstract. Markov Random Fields are widely used in many image processing applications. Recently the shortcomings of some of the simpler forms of these models have become apparent, and models based on larger neighbourhoods have been developed. When single-site updating methods are used with these models, a large number of iterations are required for convergence. The Swendsen-Wang algorithm and Partial Decoupling have been shown to give potentially enormous speed-up to computation with the simple Ising and Potts models. In this paper we show how the same ideas can be used with binary Markov Random Fields with essentially any support to construct auxiliary variable algorithms. However, because of the complexity and certain characteristics of the models, the computational gains are limited.

1 Introduction

Markov Random Fields (MRFs) were introduced into the image processing literature in 1984 [3], and have since been widely used for many tasks, mainly in low level vision. Despite the increase in computational power that has become available, and the inherent parallelisation that can be applied to the computational algorithms, computation with MRF models and single-site updating algorithms (the usual forms of the Gibbs sampler and Metropolis-Hastings [5] algorithms) is time consuming. In statistical physics applications, where the aim is to simulate large interacting spin systems, the Swendsen-Wang (SW) algorithm [11] was developed to speed up the computation, especially at the critical point, when simulating the Ising [8] or Potts models. The Multi-Level Logistic model of [3] is just the Potts model, and so the SW algorithm is applicable in that case. However, it has become apparent that the Ising or Potts model does not capture the image characteristics that are important in segmentation tasks [10]. This has motivated the development of MRFs with longer range and more complex forms of interaction [1, 12] to model more adequately the structures present in typical segmentation imagery. The application of these models has proved successful, but single-site updating algorithms have proved computationally intensive, especially when there is a requirement to estimate the hyperparameters of these models [4]. Motivated by the success of the

** This work was performed while the author held a National Research Council-NASA Ames Research Associateship. This articles appeared in *Energy Minimization Methods in Computer Vision and Pattern Recognition,* Springer Lecture Notes in Computer Science volume 1654.

SW algorithm to dramatically speed up computation with the Ising or Potts model, in this paper we investigate the use of the auxiliary variable approach applied to MRFs with long range interactions, both in the form used in the SW algorithm, and the partial decoupling approach proposed in [6]. The SW algorithm has most benefit near the critical point of the Ising model. The auxiliary variable methods developed in this paper appear to be of limited benefit for the simulation of models with long-range interactions. This is likely to be because the models are being used well away from any critical regimes. Indeed the presence or absence of critical behaviour in models with long-range interaction has to be demonstrated on a model-by-model basis. Whether the long-range interaction model considered in this paper has a phase transition is currently unknown.

2 Auxiliary Variables for Markov Random Fields

2.1 The Swendsen-Wang algorithm and Partial Decoupling

The idea behind auxiliary variable methods is the following:

It is desired to simulate a distribution $\pi(\mathbf{x})$. Auxiliary variables $\mathbf{u}$ are introduced, with conditional distribution $\pi(\mathbf{u}|\mathbf{x})$. This gives a joint distribution $\pi(\mathbf{x},\mathbf{u}) = \pi(\mathbf{u}|\mathbf{x})\pi(\mathbf{x})$, with the desired marginal distribution for $\mathbf{x}$ of $\pi(\mathbf{x})$. Simulation of this distribution is generally performed by alternately updating $\mathbf{u}$ and $\mathbf{x}$ – the idea being to define $\pi(\mathbf{u}|\mathbf{x})$ such that the updates cause rapid mixing. The realisations of $\mathbf{x}$ are those desired.

The Ising model is defined by

$$\pi(\mathbf{x}) \propto \exp\left(\beta \sum_{i\sim j} \mathrm{I}[x_i = x_j]\right) \tag{1}$$

where $i \sim j$ indicates nearest neighbour pairs.

For the SW algorithm the distribution $\pi(\mathbf{u}|\mathbf{x})$ is defined such that the u_{ij} are independent, and is

$$p(u_{ij}|\mathbf{x}) \propto \exp\left(-\beta \mathrm{I}[x_i = x_j]\right) \mathrm{I}[0 \leq u_{ij} \leq \exp\left(\beta \mathrm{I}[x_i = x_j]\right)] \tag{2}$$

where u_{ij} can be considered as a continuous 'bond' variable between the pixels x_i and x_j and $\mathrm{I}[\cdot]$ is the indicator function. This results in

$$\pi(\mathbf{x}|\mathbf{u}) \propto \prod_{i\sim j} \mathrm{I}[0 \leq u_{ij} \leq \exp\left(\beta \mathrm{I}[x_i = x_j]\right)] \tag{3}$$

What does this choice of distribution give us? Considering first $p(u_{ij}|\mathbf{x})$, for $u_{ij} > 1$ we must have $\exp\left(\beta \mathrm{I}[x_i = x_j]\right) > 1$, or equivalently, $x_i = x_j$. Thus $u_{ij} > 1$ constrains x_i and x_j to be in the same state. Conversely, if x_i and x_j are in the same state, what is the probability of $u_{ij} > 1$? From the conditional distribution in equation 2 we have that

$$p(u_{ij} > 1 | x_i = x_j) = 1 - \exp(-\beta) \tag{4}$$

Since it is only important whether u_{ij} is greater or less than one we may think of the u_{ij} as binary bond variables. From equation 4 the bond variable is present between two

pixels in the same state with probability $1-\exp(-\beta)$. To sample $\mathbf{u}$ thus involves placing bonds between neighbouring pixels of the same state with probability $1-\exp(-\beta)$ and omitting bonds between neighbouring pixels of differing states.

Once the bonds are in place, the conditional distribution $\pi(\mathbf{x}|\mathbf{u})$ says that all configurations where bonded pixels are of the same state are equally probable. Thus to update $\mathbf{x}$ we form clusters of connected pixels and assign to all pixels of the cluster the same state, chosen uniformly from the allowed states. This scheme allows potentially large clusters of pixels to change state at each iteration, allowing the Markov chain to explore the distribution freely.

In the discussion above we have assumed that the parameter β in the Potts model is positive, inducing clustering. However, the SW algorithm is still applicable if β is negative. In this case a similar argument gives that neighbours in different states are constrained to remain in different states with probability $1-\exp(\beta)$. This forms 'clusters' where neighbouring sites in the cluster must be in different states.

Partial decoupling was introduced [6, 7] to overcome some problems with the basic form of the SW algorithm when simulating systems with data. In this case growing the clusters without taking any account of the data can be unhelpful, as the clusters are unlikely to reflect the structure in the data. A modification to the conditional distribution of the auxiliary variables allows the data to be taken into account when forming the clusters. Later in this paper, we will use this modification to systematically reduce the strength of the constraints introduced by the complexity of the higher order MRF models.

The SW algorithm forms clusters which are coloured independently. The idea behind partial decoupling is to reduce the probability that bonds will be placed, with the consequence that the clusters will not be independent, and so the colouring of the clusters themselves will have to be updated using a Markov chain Monte Carlo (MCMC) scheme [6].

Instead of the definition of $p(u_{ij}|\mathbf{x})$ in equation 2 above, we now define

$$p(u_{ij}|\mathbf{x}) \propto \exp(-\delta\beta \mathrm{I}[x_i = x_j])\mathrm{I}[0 \leq u_{ij} \leq \exp(\delta\beta I[x_i = x_j])] \tag{5}$$

where δ is a fixed constant between zero and one. This results in

$$\pi(\mathbf{x}|\mathbf{u}) \propto \exp\left(\sum_{i\sim j}(1-\delta)\beta \mathrm{I}[x_i = x_j]\right) \times \prod_{i\sim j} \mathrm{I}[0 \leq u_{ij} \leq \exp(\delta\beta \mathrm{I}[x_i = x_j])] \tag{6}$$

So now we form clusters (or enforce the dissimilarity of neighbours, in the case of negative β) by bonding pixels with probability $1-\exp(-\delta\beta)$. However, the clusters thus formed are not independent, and their colouring must be updated conditionally on the pixels neighbouring the cluster. Thus cluster $\mathcal{I}$ takes colour k, conditional on its neighbors, $\mathcal{N}(\mathcal{I})$ with probability

$$\pi(k|\mathcal{N}(\mathcal{I})) \propto \exp\left(\sum_{i\sim j: i\in\mathcal{I}, j\in\mathcal{N}(\mathcal{I})} (1-\delta)\beta \mathrm{I}[k = x_j]\right) \tag{7}$$

and is updated using, for example, the Gibbs sampler or the Metropolis-Hastings algorithm.

In [7] the δ's were chosen to split the lattice up into regular blocks. In [6] the data was used to set the values of δ to encourage clusters supported by the data. In this paper we will use the δ's to avoid overconstraining the possible updates, and to reduce the complexity introduced by negative β's.

2.2 The 'chien' model

In its original formulation the 'chien' model [1] was defined as a binary MRF, where the potential function considered a 5×5 neighbourhood, and 3×3 cliques. For a clique of size 3×3 there are 512 configurations. When symmetries are removed, this reduces down to 51 classes. These classes are shown in figure 1. By considering the energy associated with lines and edges, the 51 parameters, c_1 to c_{51}, in figure 1 were reduced down to functions of three parameters, representing boundary length (e), line length (l) and noise (n). The reader is referred to [1] for full details of this model.

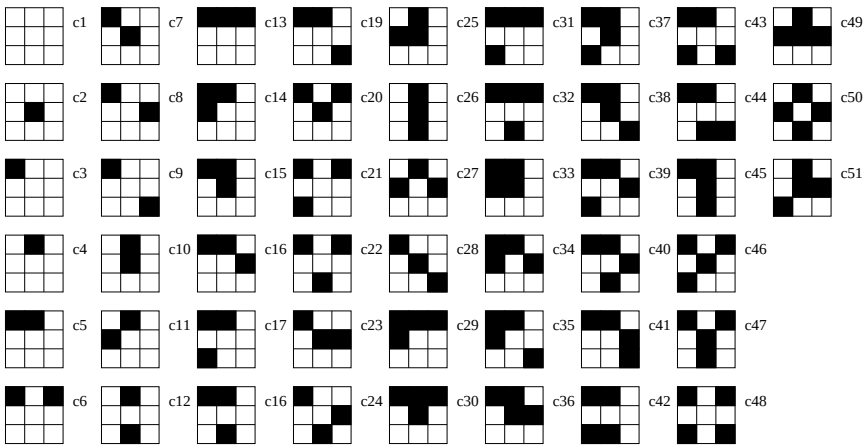

Fig. 1. The 51 classes of clique considered in the 'chien' model

2.3 Auxiliary variables for the 'chien' model

Conventionally, the probability of a configuration of an MRF is written in the following form

$$p(\mathbf{x}) = \frac{1}{Z} \exp\left(\sum_{c} V_c(\mathbf{x}) \right) \tag{8}$$

That is, the total energy of the configuration is made up of a sum over all the cliques of the potentials associated with each clique configuration. In the Ising/Potts model these cliques are the neighbouring pairs, and the configurations the state of homogeneity of

these pairs. In the 'chien' model the cliques are 3×3 blocks, and the configurations are those shown in figure 1. For the Ising/Potts model, writing the energy in the form

$$p(\mathbf{x}) \propto \exp\left(\sum_{i \sim j} \beta_{ij} \mathrm{I}[x_i = x_j]\right) \tag{9}$$

indicates how to introduce auxiliary variables to induce clusters with either reduced or eliminated dependency between the clusters. To introduce similar auxiliary variables for the 'chien' model, we must write the pdf for the 'chien' model in a similar form.

The pdf for the 'chien' model can be written as

$$p(\mathbf{x}) \propto \exp\left(\sum_{j} v(a_j, b_j, c_j, d_j, e_j, f_j, g_j, h_j, i_j)\right) \tag{10}$$

where the pixels are labeled as shown in figure 2, the sum over j is over all sites in the image and $v(\cdot)$ is the value given by the classification in figure 1. In subsequent equations we will drop the j subscript to simplify the notation.

a	**b**	**c**
d	**e**	**f**
g	**h**	**i**

Fig. 2. Labeling of the pixels in a clique of the 'chien' model

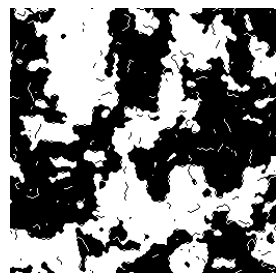

Fig. 3. Sample from the 'chien' model, $e = 0.8, l = 1.5, n = 1.6$

The energy of a clique can be written, using this labeling, as

$$\begin{aligned}
v(a,b,c,d,e,f,g,h,i) = & \\
& \mathrm{I}[a=e]\mathrm{I}[b=e]\mathrm{I}[c=e]\mathrm{I}[d=e]\mathrm{I}[f=e]\mathrm{I}[g=e]\mathrm{I}[h=e]\mathrm{I}[i=e]c_1 \\
& + \mathrm{I}[a=e]\mathrm{I}[b=e]\mathrm{I}[c=e]\mathrm{I}[d=e]\mathrm{I}[f=e]\mathrm{I}[g=e]\mathrm{I}[h=e](1-\mathrm{I}[i=e])c_3 \\
& + \mathrm{I}[a=e]\mathrm{I}[b=e]\mathrm{I}[c=e]\mathrm{I}[d=e]\mathrm{I}[f=e]\mathrm{I}[g=e](1-\mathrm{I}[h=e])\mathrm{I}[i=e]c_4 \\
& + \mathrm{I}[a=e]\mathrm{I}[b=e]\mathrm{I}[c=e]\mathrm{I}[d=e]\mathrm{I}[f=e]\mathrm{I}[g=e](1-\mathrm{I}[h=e])(1-\mathrm{I}[i=e])c_5 \\
& \vdots \\
& + (1-\mathrm{I}[a=e])(1-\mathrm{I}[b=e])\mathrm{I}[c=e](1-d[d=e])(1-\mathrm{I}[f=e]) \\
& \qquad \mathrm{I}[g=e]\mathrm{I}[h=e]\mathrm{I}[i=e]c_{35}
\end{aligned}$$

$$
\begin{aligned}
&\vdots \\
&+ (1-\mathrm{I}[a=e])(1-\mathrm{I}[b=e])(1-\mathrm{I}[c=e])(1-\mathrm{I}[d=e]) \\
&\qquad (1-\mathrm{I}[f=e])(1-\mathrm{I}[g=e])(1-\mathrm{I}[h=e])(1-\mathrm{I}[i=e])c_2 \quad (11)
\end{aligned}
$$

where the c's are the clique parameters from figure 1. A computer algebra package can be used to expand this expression, and then to reduce it into a simple form, where each entry is a product of terms of the form $\mathrm{I}[\cdot = \cdot]$, for example $\mathrm{I}[a = e](c_7 - c_2)$ and $\mathrm{I}[a=e]\mathrm{I}[b=e]\mathrm{I}[c=e]\mathrm{I}[g=e]\mathrm{I}[h=e](-c_2+3c_7+2c_{10}-3c_{15}+c_{18}-2c_{20}-3c_{23}-c_{26}-c_{28}+c_{30}-c_{32}-c_{34}+2c_{37}+2c_{38}-c_{40}-c_{42}-c_{44}+3c_{45}+c_{46}+c_{47})$ etc. In fact, terms involving all pairs, triples, 4-tuples, 5-tuples, 6-tuples, 7-tuples, 8-tuples and the single 9-tuple are present in this representation (some with a coefficient of zero). This enables us to write the energy in the form

$$
\begin{aligned}
v(a,b,c,d,e,f,g,h,i) = \\
&\mathrm{I}[a=e]\beta_{\{ae\}} + \mathrm{I}[b=e]\beta_{\{be\}} + \dots \\
&+ \mathrm{I}[a=e]\mathrm{I}[b=e]\beta_{\{abe\}} + \mathrm{I}[a=e]\mathrm{I}[c=e]\beta_{\{ace\}} + \dots \\
&+ \mathrm{I}[a=e]\mathrm{I}[b=e]\mathrm{I}[c=e]\beta_{\{abce\}} + \dots \\
&+ \mathrm{I}[a=e]\mathrm{I}[b=e]\mathrm{I}[c=e]\mathrm{I}[d=e]\beta_{\{abcde\}} + \dots \\
&+ \mathrm{I}[a=e]\mathrm{I}[b=e]\mathrm{I}[c=e]\mathrm{I}[d=e]\mathrm{I}[f=e]\beta_{\{abcdef\}} + \dots \\
&+ \mathrm{I}[a=e]\mathrm{I}[b=e]\mathrm{I}[c=e]\mathrm{I}[d=e]\mathrm{I}[f=e]\mathrm{I}[g=e]\beta_{\{abcdefg\}} + \dots \\
&+ \mathrm{I}[a=e]\mathrm{I}[b=e]\mathrm{I}[c=e]\mathrm{I}[d=e]\mathrm{I}[f=e]\mathrm{I}[g=e]\mathrm{I}[h=e]\beta_{\{abcdefgh\}} + \dots \\
&+ \mathrm{I}[a=e]\mathrm{I}[b=e]\mathrm{I}[c=e]\mathrm{I}[d=e]\mathrm{I}[f=e]\mathrm{I}[g=e]\mathrm{I}[h=e]\mathrm{I}[i=e]\beta_{\{abcdefghi\}}
\end{aligned}
\tag{12}
$$

In this representation, the distribution can be written as

$$
\pi(\mathbf{x}) \propto \prod_k \exp\left(\sum_j \mathrm{I}_k[a,b,c,d,e,f,g,h,i]\beta_{\{a,b,c,d,e,f,g,h,i\}}\right) \tag{13}
$$

where $\mathrm{I}_k[\cdot]\beta_{\{\cdot\}}$ are the terms in equation 12, k being the index of the term. This is now in a form which makes application of the extension to the SW algorithm in [2] clear.

We introduce a set of independent auxiliary variables, $\mathbf{u}_k$, corresponding to each of the terms in the representation of equation 12, with conditional distribution

$$
\pi(u_k(j)|\mathbf{x}) = \exp\left(-\mathrm{I}_k[\cdot]\beta_{\{\cdot\}}\right)\mathrm{I}[0 \le u_k(j) \le \exp\left(\mathrm{I}_k[\cdot]\beta_{\{\cdot\}}\right)] \tag{14}
$$

That is, for each term $\mathrm{I}_k[\cdot]\beta_{\{\cdot\}}$ we have a set of auxiliary variables, $\mathbf{u}_k$ as in the standard SW algorithm. Each element of $\mathbf{u}_k$, $u_k(j)$ is independent and drawn from a uniform distribution $U[0, \exp(\mathrm{I}_k[a_j, b_j, \dots]\beta_{\{\cdot\}})]$, resulting in a conditional distribution for $\mathbf{x}$ of

$$
\pi(\mathbf{x}|\mathbf{u}) = \prod_j \prod_k \mathrm{I}[u_k(j) \le \exp(\mathrm{I}_k[\cdot]\beta_{\{\cdot\}})] \tag{15}
$$

So we now have that $\mathbf{x}|\mathbf{u}$ is uniformly distributed, provided that the constraints introduced by the particular realisation of the $\mathbf{u}$ variables are satisfied. These constraints now form 'bonds' between groups of up to 9 pixels at a time, constraining the groups to be in the same state, for $\beta_{\{\cdot\}} > 0$, or groups of up to 9 pixels are constrained to not all be in the same state, for $\beta_{\{\cdot\}} < 0$. An analogous procedure can be performed for any binary MRF.

We can divide the terms into two classes, those where the coefficient $\beta_{\{\cdot\}}$ is greater than zero, and those where it is less than zero. The update procedure is then as follows

1. For each term in the expansion of equation 11 with $\beta_{\{\cdot\}} > 0$, for each site j, if all the pixels concerned are in the same state, constrain them to be in the same state in the next iteration with probability $1 - \exp(-\beta_{\{\cdot\}})$.
2. For each term in the expansion with $\beta_{\{\cdot\}} < 0$, for each site j, if all the pixels concerned are *not* in the same state, constrain them to not all be in the same state in the next iteration with probability $1 - \exp(\beta_{\{\cdot\}})$.
3. Choose any random colouring which satisfies the constraints from steps 1 and 2.

The constraints generated in step 1 (type 1 constraints) are easily dealt with – there are only two ways that a group of n pixels can all be homogeneous in a binary MRF. Thus we can use all of the constraints with $\beta_{\{\cdot\}} > 0$, irrespective of how many pixels are included in that term, to form clusters of pixels in a similar manner to the SW algorithm, where all the pixels in a cluster must be in the same state after the update.

The constraints from step 2 (type 2 constraints) are more problematic – there are $2^n - 2$ ways a group of n pixels can not all be in the same state in a binary MRF. The update strategy used was that known as 'generate-and-test' [9]. This heuristic method works well when either the density of the constraints is low (such that there are many configurations that satisfy all the constraints and finding one is relatively easy), or when the density of constraints is high, when there are very few solutions, but local changes that satisfy the constraints will almost always move towards one of the few global solutions.

The 'generate-and-test' approach results in the following algorithm.

1. Start from a random configuration that satisfies the type 1 constraints
2. Go through the list of type 2 constraints until one is not satisfied
3. To satisfy this constraint
   (a) flip the state of one of the clusters involved in this constraint
   (b) if the constraint is still not satisfied, un-flip this cluster, and flip another until the constraint is satisfied
4. Go to step 2 until all the constraints are satisfied

Because the constraints are generated from the current colouring of the pixels, we know that there is at least one colouring which satisfies all the constraints in stage 2. (However, this colouring is not of interest to us; the whole point of constructing the auxiliary variable algorithm is to find a recolouring that is significantly different from the current colouring.) Irreducibility is, however, guaranteed, as there is a non-zero probability of no constraints being placed, resulting in the $\mathbf{x}$ variables being independent, and so any state can be reached in one update.

Figure 3 shows a sample, generated by the single-site Gibbs sampler, from the 'chien' model, with the parameters being $e = 0.8, l = 1.5, n = 1.6$, after 10,000 iterations. These parameters were chosen to give a sample which shows regions together with fine structure. Starting from a random initial image (each pixel is black or white with probability 0.5), figure 4 shows the clusters induced by the type 1 constraints for this set of parameter values – the bond-graph shows how the pixels are constrained, and the cluster map shows how the bonds divide the image up into groups (there are actually 109 regions in this image). Clearly with this density of type 1 constraints finding a colouring which satisfies the type 2 constraints will be easy. It will, however, be very similar to the initial colouring. This motivates the use of the partial decoupling approach – the δ's can be chosen to reduce the density of type 1 constraints.

If the initial state is all of one colour the situation is worse – almost every pixel will be bonded into one region, and the sampler will be essentially immobile.

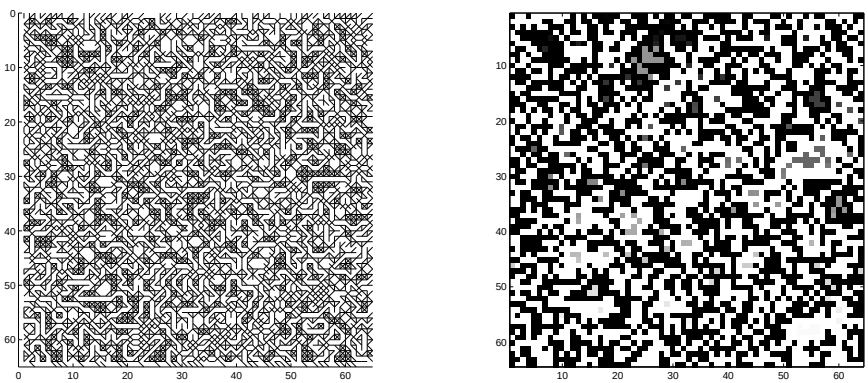

Fig. 4. Bond graph (left) and region map (right) from the type 1 constraints for the SW type algorithm (see text)

2.4 Partial decoupling for the 'chien' model

The density of the type 1 constraints lead to an almost immobile algorithm in the previous subsection. Here we consider how partial decoupling may help the mobility of the sampler by reducing the density of the type 1 constraints and by eliminating the type 2 constraints. This also results in an easier update algorithm.

To derive the partial decoupling algorithm, the same representation of the energy as in equation 12 is used. However, the conditional distribution of the $\mathbf{u}_k$'s is now

$$\pi(u_k(j)|\mathbf{x}) \propto \exp\left(\mathrm{I}_k[\cdot]\delta_{\{\cdot\}}\beta_{\{\cdot\}}\right) \mathrm{I}[0 \leq u_k(j) \leq \exp\left(\mathrm{I}_k[\cdot]\delta_{\{\cdot\}}\beta_{\{\cdot\}}\right)] \qquad (16)$$

where $\delta_{\{\cdot\}}$ is the factor associated with each of the auxiliary variables. This enables the influence in the clustering and anti-clustering of each term in equation 12 to be controlled. In practise this enables the complexity of the update algorithm caused by

the terms with $\beta_{\{\cdot\}} < 0$ to be eliminated, by choosing $\delta_{\{\cdot\}} = 0$ for these terms. In the experiments described below, the same value of δ was used for all the terms with $\beta_{\{\cdot\}} > 0$.

With this set of auxiliary variables, the conditional distribution of $\mathbf{x}|\mathbf{u}$ is now

$$\pi(\mathbf{x}|\mathbf{u}) \propto \prod_j \left[\prod_{k:\beta_{\{\cdot\}}<0} \exp\left(\mathrm{I}_k[\cdot]\beta_{\{\cdot\}}\right) \times \prod_{k:\beta_{\{\cdot\}}>0} \exp\left(\mathrm{I}_k[\cdot](1-\delta_{\{\cdot\}})\beta_{\{\cdot\}}\right) \right.$$
$$\left. \times \prod_{k:\beta_{\{\cdot\}}>0} \mathrm{I}[u_k(j) \leq \exp(\mathrm{I}_k[\cdot]\delta_{\{\cdot\}}\beta_{\{\cdot\}})] \right] \quad (17)$$

The final term of this equation is the cluster constraints – when updating the $\mathbf{u}_k : \beta_{\{\cdot\}} > 0$, the pixels are bonded with probability $1 - \exp(-\delta_{\{\cdot\}}\beta_{\{\cdot\}})$, and this terms says that all the pixels in a cluster must be the same colour.

The first two terms give the distribution of the colours of the clusters. From them we can easily derive the conditional distribution for the colour of each cluster, given its neighbours. For computational purposes, it is convenient to transform the representation back into the form of equation 10, except that now the potentials of the configurations have been modified by the inclusion of the $(1 - \delta_{\{\cdot\}})$ terms. This allows simpler computation when computing the probabilities of the allowed colours for the regions when implementing the Gibbs sampler.

This results in a form of block update algorithm. The weakened cluster constraints form regions, the size and shape of which is a function of the model. These regions are then recoloured with probabilities which reflect the cluster formation process and the model.

Figure 5 shows the bond-graph and the corresponding cluster map for the application of the partial decoupling algorithm to an initial random image. Clearly the density of the bonds is much reduced from figure 4, so the algorithm should be more mobile. The colouring of the clusters is updated conditionally on the cluster's neigbours, using the Gibbs sampler. Figure 6 shows the initial image and the result after one iteration. The algorithm clearly moves rapidly towards the equilibrium distribution. However, further updates using the Partial Decoupling algorithm rapidly move towards an almost uniform image – even with reduced bonding strength, the number of possible ways the pixels in a uniform region can be bonded results in most of them being joined into one cluster, and then the conditional update will preferentially re-colour the regions to eliminate edges. The algorithm is thus of limited applicability – it moves rapidly towards the equilibrium distribution, but moves slowly once it has converged.

3 Conclusions

We have shown how to construct an auxiliary variable algorithm for the MRF known as the 'chien' model, and explained how this method may be used with an essentially arbitrary binary MRF. Because of the strength of interaction in the model, however, introducing a full set of auxiliary variables results in a sampler which moves very slowly.

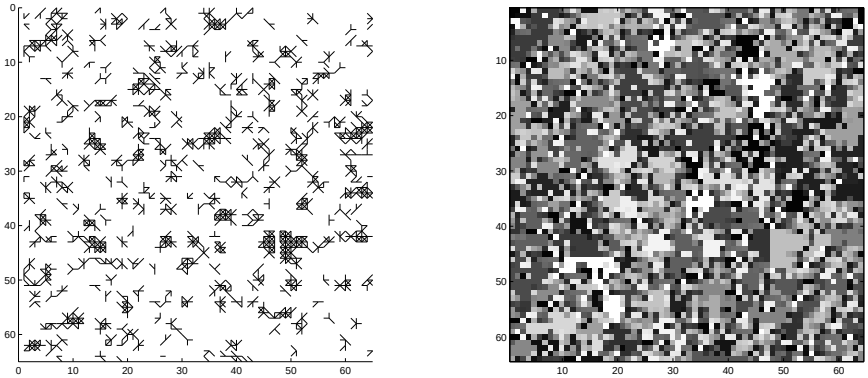

Fig. 5. Bond graph (left) and region map (right) for the Partial Decoupling algorithm ($\delta = 0.0067$)

Fig. 6. Initial random image (left) and image after one iteration of the Partial Decoupling algorithm (right)

Reducing the set of auxiliary variables, and reducing the influence of those included, enables an algorithm to be constructed which is a form of block-update algorithm. This moves rapidly from a random start point towards the equilibrium distribution, but then moves into one of the modes of the distribution, and becomes immobile.

The SW algorithm for the Ising model shows most spectacular improvement at the critical point, when the correlation length becomes infinite. The 'chien' model with parameters corresponding to the characteristics of real images does not seem to exhibit this behaviour, and so auxiliary variables are of less benefit. Whether the 'chien' and other higher order models do show critical behaviour for some parameter values is an open problem. If they do, then the algorithm described in this paper should be very useful for simulating the models in those behaviour regimes.

References

1. X. Descombes, J.F. Mangin, E. Pechersky, and M. Sigelle. Fine structures preserving model for image processing. In *Proc. 9th SCIA 95 Uppsala, Sweden*, pages 349–356, 1995.
2. R.G. Edwards and A.D. Sokal. Generalization of the Fortuin-Kasteleyn-Swendsen-Wang representation and Monte Carlo algorithm. *Physical Review Letters*, pages 2009–2012, 1988.
3. S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images. *IEEE Trans PAMI*, 6(6):721–741, November 1984.
4. C. J. Geyer and E.E. Thompson. Constrained Monte Carlo Maximum Likelihood for dependent data. *JRSS - B*, 54(3):657–699, 1992.
5. W.K. Hastings. Monte Carlo sampling methods using Markov Chains and their applications. *Biometrika*, 57:97–109, 1970.
6. D.M. Higdon. Auxiliary variable methods for Markov chain Monte Carlo with applications. *Journal of the American Statistical Association*, 93:585-595, June 1998
7. M. Hurn. Difficulties in the use of auxiliary variables in Markov chain Monte Carlo methods. *Statistics and Computing*, 7:35–44, 1997.
8. E. Ising. . *Zeitschrift Physik*, 31:253, 1925.
9. S. Minton, M.D. Johnston, A.B. Phillips, and P. Laird. Solving large-scale constraint satisfaction and scheduling problems using a heuristic repair method. In *Proceedings of AAAI*, 1990.
10. R.D. Morris, X. Descombes, and J. Zerubia. The Ising/Potts model is not well suited to segmentation tasks. In *Proceedings of the IEEE Digital Signal Processing Workshop*, September 1996.
11. R.H. Swendsen and J-S. Wang. Nonuniversal critical dynamics in Monte Carlo simulations. *Physical Review Letters*, 58:86–88, 1987.
12. H. Tjelmeland and J. Besag. Markov Random Fields with higher order interactions. *Scandinavian Journal of Statistics*, 25:415-433, 1998.