MCMC With Disconnected State Spaces

Leon Barrett and Aleksandr Simma

May 5, 2005

1 Introduction

Bayes Nets simplify probabilistic models, making it easy to work with these models. Unfortunately, sometimes people devise models that are too complicated to allow calculation of exact probabilities, so they instead use approximate inference, such as Markov Chain Monte Carlo (MCMC). However, MCMC can fail if the Bayes Net has zero-probability states that "disconnect" the state space. In this paper, we attempt to modify MCMC to handle such nets robustly while not losing sight of efficiency.

2 Background and Terminology

2.1 Bayes Nets

Bayes Nets are an increasingly popular tool in many fields, especially machine learning. A *Bayes Net* is a structure for defining a probability distribution on a state space. It is represented by a DAG where each node stands for a random variable with a set of possible states (Ω_j), and at any time, occupies exactly one of those states ($\omega_j \in \Omega_j$). The state of the net is $\omega = (\omega_1, \omega_2 \ldots \omega_n)$, the tuple of the states of all the nodes and the state space for the graph $\Omega = \Omega_1 \times \Omega_2 \times \ldots \Omega_n$.

Each node has a probability distribution over its states dependent on the states of its parents; the probability of a state of the net is the product of the probabilities of all the nodes.

$$P(\omega) \quad = \quad \prod_{j \in V} P(\omega_i | \omega_{\text{parents}_i})$$

Furthermore, using Bayes' Rule, it is simple to calculate the distribution over any variable's states given the other nodes in the graph.

$$P(\omega_i) = \quad \alpha \prod_{p \in \text{parents}_i} P(\omega_i | \omega_p) \prod_{q \in \text{children}_i} P(\omega_q | \text{parents}_q)$$

(Here, α is a normalizing constant.)

Thus, Bayes Nets provide a formalism giving easy calculation of conditional probabilities. However, calculating all conditional probabilities is known to be NP-hard [Coo90]. Furthermore, even in normal conditions, calculating conditional probabilities is often too difficult to do exactly. Instead, approximation methods such as Markov Chain Monte Carlo (MCMC) are used.

For more discussion of Bayes Nets, see [RN03].

2.2 MCMC

Markov Chain Monte Carlo (MCMC)[MRR$^+$53, Has70] is family of computational methods for sampling from a probability distribution π. It works by defining a Markov Chain (a sequence of random states,

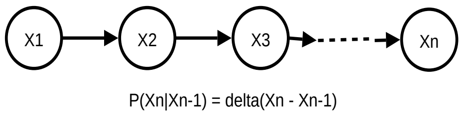


Figure 1: A Chain of Deterministic Variables

each dependent on only the last) over the state space of the distribution, defining transitions between states so that the stationary distribution over the Markov Chain is the same as the distribution to be sampled from. In order for this to be true, *stationarity* must hold when $\pi(\omega) = P(\omega)$. Stationarity means that $\pi(\omega') = \sum_{\omega} \pi(\omega)P(\omega \to \omega')$. *Detailed balance* is a sufficient condition for stationarity, where detailed balance says for any two states ω and ω', $P(\omega \to \omega')\pi(\omega) = P(\omega' \to \omega)\pi(\omega')$. Essentially, detailed balance demands that the total probability mass flow between any two states is zero.

The usual method for performing MCMC on Bayes Nets is *Gibbs Sampling*[GG87], in which the underlying Markov Chain is constructed so that $P(\omega \to \omega') \propto \pi(\omega')$ if ω and ω' differ in the state of at most one node, and 0 otherwise. This allows for a sparse Markov Chain on which the computation of transition probabilities is feasible.

Unfortunately, MCMC works only under the condition of *ergodicity*, which demands that every state be reachable from every other state in a finite number of positive-probability steps. This is unfortunate because sometimes Bayes Nets are *deterministic*, where setting certain random variables to certain values can restrict the distribution of another variable to have only one value with nonzero probability. Specifically, we have a set S of n random variables, where the values of variables $S_{1...n-1}$ restrict the domain of S_n to size 1. If the determinism works both ways, so that knowing the values of $S_{1...m-1,m+1...n}$ determinse the value of S_m, then the state space can become *disconnected*. In a disconnected state space, there is some pair of states, ω and ω' with nonzero probability such that, starting in state ω, no number of MCMC steps will produce ω'.

Of course, a way to address this problem in MCMC is to increase the connectivity of the state space. Instead of randomly setting one variable at a time (as done in Gibbs sampling), we could set two or more variables simultaneously. Thus, states which were previously unreachable can now be achieved in a single step. The catch is that we do not know the degree of disconnection; a state space of n variables may be disconnected even under the change of $n - 1$ variables! (Consider the case of a chain of deterministic variables, as seen in Fig. 1; here, all n variables must change simultaneously.) Thus, we wish to define an MCMC algorithm that is robust to even such extreme cases as this.

3 Breaking Up the Net

The techniques discussed in the remainder of the paper are intended to yield correct and fairly efficient algorithms for deterministic Bayes Nets with disconnected solution spaces, and for nets with low-probability states which impede slow mixing. In coping with zero-probability states in particular, it may be valuable to partition the net into separate conditionally independent components and sample from each one separately.

3.1 Difficulty of Partitioning

It is of note that the MCMC methods described can be made to work not only on the whole net, but also on any part of the net. That part can be extracted and treated as a new Bayes Net, with prior probabilities defined by the rest of the original net. Then, once a sample is drawn, that sample can be reinserted into the original net, and a different part can be extracted and sampled. As long as every node

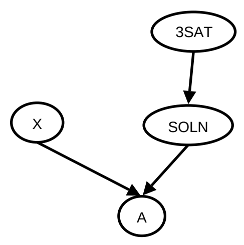


Figure 2: It Is NP-Complete To Determine If The State-Space Is Disconnected.

gets regularly sampled and all the Markov Chains are ergodic, the procedure will produce correct results. This is very similar to the method used in [JKK95].

It is in our interest to partition any net into sub-partitions that are as small as possible, as long as the resulting chain is ergodic. This partitioning requires that the set of zero-probability states of any member of the partition must be independent of the state of any other member of the partition. As shown in §3.2, finding an optimal partition is a NP-hard problem; however, there exist simple algorithms, which will provide a partition that, though not necessarily minimal, satisfies the requirements.

3.2 Complexity Results

It is well-known that probabilistic inference on Bayes Nets is NP-hard [Coo90]. Specifically, one can write a 3SAT problem as a Bayes Net of polynomial size, such that finding a single nonzero-probability state of the net solves the 3SAT problem. Thus, we do not hope to provide a foolproof method for approximate inference on all Bayes Nets.

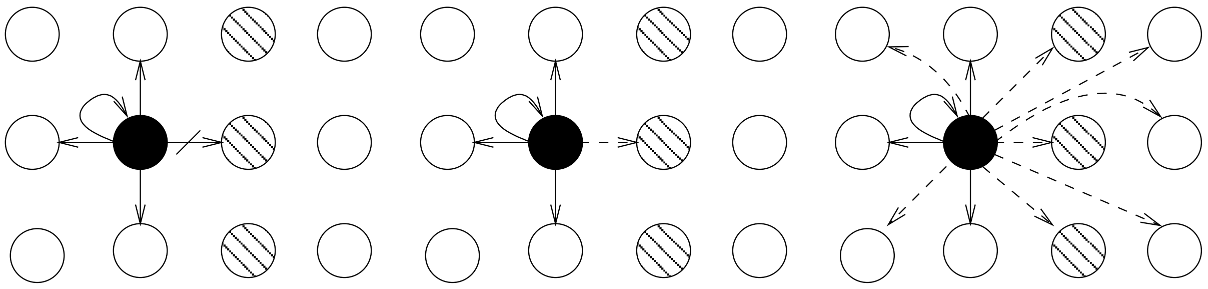
Indeed, even the approximation problem is NP-Complete. To see this, note that for the 3SAT Bayes Net, it is NP-Complete to find a valid state. If the 3SAT problem has only one solution, any approximate solution that fails to find that state will be maximally distant from the true distribution using any distance metric.

In an interesting corollary, it can be shown that determining whether a Bayes Net has a disconnected solution space is also NP-hard. Consider the network in Fig. 2. Here, $3SAT$ represents a 3SAT problem defined as a Bayes Net, and the variable $SOLN$ indicates whether or not the 3SAT problem has a solution. X can be either true or false, and $A = SOLN \wedge X$. That is, A is true if and only if both of X and $SOLN$ are true. If the 3SAT problem has no solution, then there is only one state for A: it must be false, no matter the value of X, so the state space is connected. However, if there is a solution, then $A = X$, so X and A must change at the same time; the solution space is disconnected. Determining connectedness requires solving the 3SAT problem, so it is NP-hard to determine whether the state space is connected.

Fortunately, we can still put an upper bound on the size of a deterministic region. We know that the disconnected region must consist of variables which restrict each others' domains. Then, a variable X which cannot possibly restrict the domain of another variable Y must not be deterministic on Y, and vice versa. So, we can find a deterministic region by starting with one variable, X. Then, we find all variables that restrict X's domain, or which have domains restricted by X. We add these variables to the deterministic region, and we expand the region in the same way until there are no variables outside it deterministic with variables inside it. This may well find regions which are too large, but we are sure that it will never find regions that are too small.

3.3 Benefits of Partitioning

Let $\Omega^+ = \{\omega \in \Omega : P(\omega) > 0\}$, the positive-probability states, and $\Omega^0 = \{\omega \in \Omega : P(\omega) = 0\}$, the zero-probability states. Consider n Bayesian Nets that have state spaces $\Omega_1, \Omega_2 \ldots \Omega_n$. Now, suppose we



(a) Normal Gibbs only allows transitions to non-zero probability states and so the states on the right side are unreachable.

(b) Smoothed Gibbs will allow the chain to transition into a zero-probability state, but only with low probability.

(c) Random restart will allow a transition to any state with a low probability.

Figure 3: Comparison of the three MCMC methods. The black state represents the current state, the shaded states have zero-probability. Solid lines indicate the normal Gibbs transitions; dashed-lines indicate low-probability added transitions.

combine the n nets into a larger Bayes Net such that there are no deterministic interactions between the subnets. Then, $\Omega = \Omega_1 \times \Omega_2 \times \Omega_3 ... \times \Omega_n$ and $\Omega^+ = \Omega_1^+ \times \Omega_2^+ \times \Omega_3^+ .. \times \Omega_n^+$ because in order for a state to have a positive probability in the combined net, each of the subnets must have a non-zero probability.

For the sampling algorithms described in the remainder of this paper, the *density*$=|\Omega^+|/|\Omega|$ is important for efficiency. Note that

$$density(\Omega) \quad = \quad \prod_{i=1}^{n} density(\Omega_i)$$

If we combine m nets of density at most d and any number of nets with any density, the resulting net will have a density of at most d^m. That is, the inverse density of the combined graph increases exponentially. Thus, if we can reverse this construction by decomposing a net into partitions which exert no deterministic influence over each other, the density of each partition will be exponentially greater than the density of the entire net, and the sampling can be more efficient.

4 Giving Probability Mass to Disallowed States

One approach to resolving the problem of the non-ergodic Markov Chain is to transform the problem into one where the Chain is ergodic and then apply the inverse transformation to the samples. This can be done by altering the probability distribution we sample from, and then weighting the resulting samples.

4.1 Motivation

A fairly intuitive solution to the problem of the state space having disjoint components is to introduce some way to allow the Markov Chain to traverse the zero-probability regions. One idea is that if a state is next to a zero-probability region, it has a certain chance of entering that region. Once there, the chain can explore that zero-probability region, and at some point exit into a positive probability state. But can this be made to give a correct answer?

An alternative way of thinking about it is that if we adjust the net to give the zero-probability regions some tiny ϵ probability, the potential state space will no longer be disjoint. Then, we can ignore all "impossible" samples the chain produces, and hopefully get a correct distribution as a result.

While this idea has some merit, consider the situation when two disjoint solutions are separated by a region with very low, but non-zero probability. This method will not help bridge that gap; crossing the low-probability region will still be very, very unlikely. The Markov Chain will have a very long mixing time. However, an alternative is to adjust the probabilities of all states, run the Markov Chain on a smoothed probability distribution, and then extract the original distribution from the chain. This is very similar to the method used by [dJWD].

4.2 Smoothing the distribution

If our Bayes Net defines a probability distribution P, define a measure M by $M(\omega) = P(\omega) + \epsilon$ for $\omega \in \Omega$. M is now a smoothed version of P, but M is no longer a probability distribution because $M(\Omega)=1+\epsilon|\Omega|$. However $P'(\cdot) = M(\cdot)/M(\Omega)$ is a valid probability distribution, so we can run the Gibbs Sampler on it. However, because P' is a different distribution, we have to weight observations with a scaling factor. If we weight each sample ω by $P(\omega)/P'(\omega)$, the resulting distribution will be correct.

4.3 Correctness

The Markov Chain induced by the Gibbs Sampler on P' is ergodic, because every pair of states is linked by a positive-probability path. If we let H' be the distribution produced by the Gibbs Sampler on P', then $H' \to P'$ in the sense that for all ω, $H'(\omega) - P'(\omega)$ approaches 0 with probability 1 as the number of samples goes to infinity. So, if we define $H(\omega) = H'(\omega)P(\omega)/P'(\omega)$, then $H \to P$.

Note that if we set $\epsilon = 0$, we recover the original Gibbs Sampler.

4.4 Efficiency

Even though the approach is guaranteed to produce correct results – that is, the sampled distribution will asymptotically converge to the true distribution – the rate at which convergence occurs depends on the choice of parameters. Indeed, a good question to ask is what meaning of the term "rate" is reasonable. The usual probability definition for the distance between two distributions in the context of convergence is

$$D(P,H) \quad = \quad \sup_{A \subset \Omega} |P(A) - H(A)|$$

or the maximum disparity between the two distributions on the probability of some set of states. This is the definition used for discussing convergence of Markov Chains, but for this problem, it is of dubious worth. As our state space Ω is huge (100^2? 1000^{10}?), the vast, vast majority of states will not be visited, and so will have an estimated probability of 0. So, if $A = \{\omega : \omega \textit{ not visited by the sampler}\}$ then $|P(A) - H(A)| = P(A)$ will remain large unless all the probability density is concentrated in a very small number of states.

Nevertheless, we can use the above definition of distance to observe why large or small values of ϵ result in inefficient algorithms. Consider a pathological case – a chain of n binary nodes and a probability distribution of $P[\omega = (1,1,0,0,\ldots 0)] = P[\omega = (0,0,0\ldots 0)] = 1/2$ (that is, the first two variables must be the same, and the rest must be zero). Suppose the chain starts in $(0,0,0\ldots 0)$. If ϵ is low, the chance of the chain ever leaving this state during the duration of the sampler is low. If ϵ is very high, the chain will quickly leave the start state, but will be liable to continue exploring the zero-probability state space without ever entering $(1,1,0,0,\ldots 0)$. This is because each state in the zero-probability space has an adjusted probability not much lower than that of $(1, 1, 0 \ldots 0)$, but there are many more zero-probability states.

Fundamentally, the choice of ϵ determines a trade-off between fast mixing times (high ϵ) and sampling from a distribution which is close to the original Bayes Net (instead of spending a lot of time in zero or low-probability states). For any $A \subset \Omega$, the chain will on average spend $P'(A) = \frac{P(A)+\epsilon|A|}{1+\epsilon|\Omega|}$ of its time in states A. In particular, if $A = \{\omega : P(\omega) = 0\} \equiv \Omega^0$, the zero probability space, then $\frac{\epsilon|\Omega^0|}{1+\epsilon|\Omega|}$ time will be spent exploring zero-probability states, which is only useful insofar it discovers new high-probability states. As $|\Omega^0|/|\Omega|$ is often close to one, unless $\epsilon = O(|\Omega|^{-1})$, the amount of time spent exploring the positive-probability states will quickly go to 0.

Recall, from §3.3, that *density* $\equiv d = \frac{|\Omega^+|}{|\Omega|}$. Then, writing the efficiency in terms of d, we see

$$\text{efficiency} = 1 - \text{time spent in } \Omega^0 = \frac{\epsilon d}{|\Omega|^{-1} + \epsilon}$$

Our efficiency increases linearly with d, so subdivision, which grows d exponentially, is very useful.

5 Randomly Restarting

5.1 Motivation

The random restart algorithm attempts to solve the connectedness problem by connecting every state to *every other state* with at least some small probability. Thus, there can be no discontinuities at all in the state space. However, the probability of such a very large jump should be kept small, so as to keep the advantages of MCMC.

5.2 Method

The random restart algorithm works by altering the Gibbs Sampler so that, with probability ρ, it instead transitions to a random state chosen from a completely uniform distribution. So, ρ becomes a tunable parameter determining how likely the sampler is to make huge jumps. When $\rho = 0$, the algorithm reduces to Gibbs sampling, and when $\rho = 1$, it simplifies to likelihood-weighted sampling.

It is easy to select a new Bayes Net state from a completely uniform distribution. We can simply select each variable's state from a uniform distribution, and the product of many independent uniform distributions is also a uniform distribution. However, to obtain the correct distribution, we must also *weight* the samples taken from this distribution. This weight must cancel out the uniform distribution to give the correct distribution, so it is $\text{Weight}(\omega) = P(\omega)/|\Omega|$.

The resulting algorithm is shown in Algorithm 1.

5.3 Proof of Correctness

First, we prove that weighted uniform sampling provides the right probability distribution. Let Ω represent the state space, with states ω, each of which has the true probability value $P(\omega)$, which is easy to calculate given a Bayes Net. Then, uniform sampling randomly selects a state ω with probability $1/|\Omega|$. If we then assign a weight of $P(\omega)|\Omega|$ to this state, it will on average receive $P(\omega)$ weight. By the law of large numbers, as we take the number of random samples to ∞, the average weight for each state will converge to the expectation value, $P(\omega)$.

Finally, we prove that the randomly-restarted algorithm is correct. At each step, in some state ω, we do one of two things. We may take a Gibbs step (choose a variable uniformly, and then choose a new value for it out of a distribution proportional to $P(v', \omega - v)$). Alternatively, we could randomly jump to

Algorithm 1 RandomRestart(net, samples, ρ)

```
  Choose state $\omega$ uniformly from net
  $a \leftarrow P(\omega)/|\Omega|$
  for samples iterations do
    Choose $r$ uniformly from $[0, 1]$
    if $r > \rho$ then
      $\omega \leftarrow \text{GibbsStep}(\text{net}, \omega)$
    else
      Choose a new state $\omega$ uniformly from net
      $a \leftarrow P(\omega)/|\Omega|$
    end if
    stats $\leftarrow$ stats $+ a\omega$
  end for
```

a new state ω' and assign it a weight as in the weighted sampling method. To show that this is a correct sampling algorithm, we must first show that the method is *ergodic* (every state is reachable from every other state). It is trivial to show that this sampling method is ergodic; because every state can choose a successor uniformly, any successor state is possible.

It turns out that if ergodicity holds, there is only one stationary distribution, and this is what MCMC converges to [RN03]. So, we must also show that the distribution produced is *stationary* when $\pi(\omega) = P(\omega)$, meaning that

$$P(\omega') = \pi(\omega') = \sum_{\omega} \text{Mass}(\omega \to \omega')$$

The two transition possibilities (a Gibbs step or a random restart) give very different mass transfers.

Let $A(\omega, \omega')$ be an indicator, so that

$$A(\omega, \omega') = \begin{cases} 1 & \text{if } \omega \text{ is adjacent to } \omega' \\ 0 & \textit{else} \end{cases}$$

Also, let $\bar{\omega}_i$ be the state of all variables except the i^{th} one. Then, the Gibbs step provides

$$\begin{aligned} \text{GibbsMass}(\omega \to \omega') = & \quad (1-\rho)\pi(\omega)A(\omega,\omega')P(\text{selecting var. } i)P(\text{setting var. } i \text{ to the value in } \omega') \\ = & \quad (1-\rho)P(\omega)A(\omega,\omega')P(i)P(\omega'_i|\bar{\omega}_i, i) \end{aligned}$$

The random restart provides

$$\begin{aligned} \text{RestartMass}(\omega \to \omega') = & \quad \rho P(\text{being in } \omega)P(\text{going to } \omega')(\text{weight of } \omega') \\ = & \quad \rho P(\text{being in } \omega)\tfrac{1}{|\Omega|}\left(P(\omega')|\Omega|\right) \\ = & \quad \rho P(\text{being in } \omega)P(\omega') \end{aligned}$$

So,

$$\begin{aligned} \sum_{\omega} \text{Mass}(\omega \to \omega') = & \quad \textstyle\sum_{\omega}(1-\rho)P(\omega)A(\omega,\omega')P(i)P(\omega'_i|\bar{\omega}_i, i) + \rho P(\text{being in } \omega)P(\omega') \\ = & \quad \rho P(\omega') + (1-\rho)\textstyle\sum_{\omega} P(\omega)A(\omega,\omega')P(i)P(\omega'_i|\bar{\omega}_i, i) \\ = & \quad \rho P(\omega') + (1-\rho)\textstyle\sum_{\omega} P(\omega_i|\bar{\omega}_i, i)P(\bar{\omega}_i)A(\omega,\omega')P(i)P(\omega'_i|\bar{\omega}_i, i) \\ = & \quad \rho P(\omega') + (1-\rho)P(\omega')\textstyle\sum_{\omega} A(\omega,\omega')P(i)P(\omega_i|\bar{\omega}_i, i) \end{aligned}$$

Note that the contents of the sum give the probability of transitioning from ω' to some ω under the Gibbs Sampler. This must sum to 1, because ω' must always transition to some other state.

$$\begin{aligned}\sum_{\omega} \text{Mass}(\omega \to \omega') = &\quad \rho P(\omega') + (1-\rho)P(\omega') \\ = &\quad P(\omega')\end{aligned}$$

So, stationarity occurs when $\pi(\omega) = P(\omega)$, and sampling via this algorithm will produce probability mass approaching the correct distribution.

5.4 Efficiency

The source of inefficiency for the RandomRestart algorithm is that it sometimes spends time visiting states with $P(\omega) = 0$. Since this can never happen with Gibbs sampling, it only happens at a random restart. Let $\Omega^0 = \{\omega : P(\omega) = 0\}$ (i.e. the "disallowed states"). Then, the probability of going to a disallowed state is $\rho\frac{|\Omega^0|}{|\Omega|}$. Clearly, higher ρ leads to higher connectivity and faster exploring of disconnected regions, but only at the cost of increased inefficiency.

In fact, we can find a relation between ρ and ϵ, the tunable parameter from §4. When both have the same efficiency, we have

$$\begin{aligned}\rho\frac{|\Omega^0|}{|\Omega|} &= \text{inefficiency} = \frac{\epsilon|\Omega^0|}{1+\epsilon|\Omega|} \\ \rho &= \frac{\epsilon|\Omega|}{1+\epsilon|\Omega|}\end{aligned}$$

Because of this relation, this algorithm performs exactly like the add-ϵ method in terms of density, so subdividing the Bayes Net is again very valuable.

6 Conclusions

The Gibbs Sampler *can* be altered to work on Bayes Nets with disconnected state spaces. In this paper, we outlined two such modified algorithms. While perhaps less efficient than the Gibbs Sampler on graphs with quick mixing times, they are robust in the case of graphs that mix poorly.

Though the problem has high complexity, when combined with graph subdivision, the algorithms described perform as well as can reasonably be expected.

References

[Coo90] Gregory F. Cooper. The computational complexity of probabilistic inference using bayesian belief networks (research note). *Artif. Intell.*, 42(2-3):393–405, 1990.

[dJWD] Edwin D. de Jong, Marco A. Wiering, and Madalina M. Drugan. Post-processing for mcmc.

[GG87] Stuart Geman and Donald Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. pages 564–584, 1987.

[Has70] W. K. Hastings. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57:97–109, 1970.

[JKK95] C. Jensen, A. Kong, and U. Kjaerulff. Blocking gibbs sampling in very large probabilistic expert systems, 1995.

[MRR$^+$53] N. Metropolis, A. Rosenbluth, M. Rosenbluth, M. Teller, and Teller E. Equations of state calculations by fast computing machines. *J. Chem. Phys*, 21:1087–1092, 1953.

[RN03] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Pearson Education, Inc., 2 edition, 2003.