Efficient Sampling from Combinatorial Space via Bridging

Dahua Lin
CSAIL, MIT

John Fisher
CSAIL, MIT

Abstract

MCMC sampling has been extensively studied and used in probabilistic inference. Many algorithms rely on local updates to explore the space, often resulting in slow convergence or failure to mix when there is no path from one set of states to another via local changes. We propose an efficient method for sampling from combinatorial spaces that addresses these issues via "bridging states" that facilitate the communication between different parts of the space. Such states can be created dynamically, providing more flexibility than methods relying on specific space structures to design jump proposals. Theoretical analysis of the approach yields bounds on mixing times. Empirical analysis demonstrates the practical utility on two problems: constrained map labeling and inferring partial order of object layers in a video.

1 Introduction

Inference of combinatorial configurations under specific constraints arises as an important problem in numerous areas of artificial intelligence, including structural learning(Friedman and Koller, 2003; Eaton and Murphy, 2007), data mining(Pei et al., 2006), bioinformatics(Vahedi et al., 2009), and circuit verification(Kitchen and Kuehlmann, 2009). In such problems, seeking an optimal estimate is NP-Hard. One example is to infer the underlying partial order from dependent observations. Owing to the complexity, approximate inference techniques are used, among which, Monte Carlo sampling is a prominent choice. Here, we aim to develop a generic method for sampling from constrained combinatorial spaces.

Current sampling methods fall mainly into three categories: (1) *Direct sampling* enumerates all possible samples and evaluates their probabilities. This is usually intractable for combinatorial problems as the sample space grows exponentially with the problem scale. (2) *Rejection sampling* generate samples without enforcing the constraints and rejects those that violate them. This can be very inefficient since the chance of obtaining a valid sample can be extremely low through random sampling from the underlying product space. (3) *Markov Chain Monte Carlo* (MCMC) (Walsh, 2002) is a popular method for Bayesian inference. The idea is to construct an ergodic Markov chain which has the desired distribution as its equilibrium distribution, thus reducing sampling to Markov simulation.

MCMC relies on an ergodic Markov chain with rapid mixing. Devising such a chain over a constrained combinatorial space can be challenging. Gibbs sampling, where each transition updates a single variable of the sample, is one of the most widely used MCMC methods. However, in combinatorial problems (*e.g.* the graph coloring problem, where the color of each node must differ from that of its neighbors), there often exist strong and deterministic relations between variables. Hence, the set of possible values for a variable can be severely restricted by the value of others. At times, no single variable update is possible without violating the constraints, thus rendering the underlying Markov chain non-ergodic.

The Metropolis-Hastings algorithm allows for customized proposal kernels, providing for more flexible moves that may break local traps or jump between different spaces. Duane et al. (1987) proposed Hybrid Monte Carlo, which utilizes Hamiltonian dynamics to drive the evolution of the target state, resulting in larger strides across the space. Swendsen and Wang (1987) proposed an algorithm for efficient simulation of Ising models, which partitions the MRF into clusters, and assign a new spin value for each one at a iteration. Barbu and Zhu (2005) later reformulated it as an M-H algorithm, and extended it to a broader class of posterior segmentation problems. Green (1995, 2003) developed Reversible Jump MCMC, which per-

Appearing in Proceedings of the 15^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2012, La Palma, Canary Islands. Volume XX of JMLR: W&CP XX. Copyright 2012 by the authors.

forms Bayesian model selection, by sampling from a mixture of model spaces with different dimensions, via trans-dimensional jumps. Data-driven strategies that exploit the observed data to generate proposals have received increasing attention, and have been used to solve various problems such as image segmentation (Tu and Zhu, 2002) and Bayesian structure learning (Eaton and Murphy, 2007). These algorithms are difficult to generalize to other contexts as they are tailored to specific models (*e.g.* model selection and MRF labeling).

Here, we develop a generic methodology to efficiently sample from constrained combinatorial spaces without strong assumptions of the space structure. We introduce the notion of *bridging* to connect different regions of the sample space that could otherwise be difficult or even impossible to communicate with each other. Specifically, we augment the state space with a set of *bridging states* and make connections between these bridges and the target states under a *cross-space detailed balance* condition in order to obtain a joint Markov chain. We establish the correctness of this approach and derive bounds on the mixing time using bottleneck analysis. We also show that by hierarchically bridging at multiple levels, one can obtain an ergodic Markov chain in spite of the space structure, while maintaining considerable probability of drawing target samples from the augmented space. Importantly, the approach constructs bridging states dynamically making it more flexible than many previous MCMC methods which exploit complete knowledge of the space structure to derive the proposal kernels.

Previous work suggests sampling to solve constrained combinatorial problems. Wei et al. (2004) proposed WalkSAT that seeks solutions to a boolean satisfiability problem (SAT) via random walks interleaved with simulated annealing steps. Kitchen and Kuehlmann (2009) extended this approach to solve problems with mixed boolean/integer constraints, under the Metropolis-Hastings formulation. This approach allows constraint violation, and drives the state towards satisfying solutions using an energy function that incurs costs for the constraints being violated. Barrett and Simma (2005) proposed an MCMC method that explicitly addresses the disconnected-space issue. The idea is to assign small probability mass to each invalid state, and use occasional random restarts to jump between different regions. Both methods above sample from "smoothed" versions of the target distribution instead of the exact one, mixing slowly when valid solutions are sparse, and increase the probability of falling in an invalid region. Hamze and de Freitas (2010) presented a method to sample from a constrained Ising model through self avoiding walks. It is exact and efficient, but restricted to a specific type of problem.

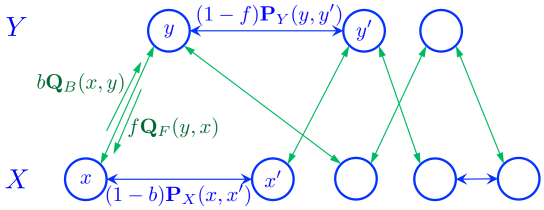

Figure 1: This illustrates how two Markov chains are bridged. In the joint chain over $X \cup Y$, each $x \in X$ has a probability $b\mathbf{Q}_B(x,y)$ to transit to $y \in Y$, and each y has a probability $f\mathbf{Q}_F(y,x)$ to transit to x.

2 Theory

Suppose we wish to sample from distribution $\boldsymbol{\mu}$ over a constrained combinatorial space X. Using local moves, we can derive a Markov chain with transition matrix $\mathbf{P}$, which may have slow mixing or even be non-ergodic. In order to mitigate such issues we suggest the notion of "bridging" as a way to connect different regions of the sample space that are otherwise difficult or even impossible to communicate. Specifically, we introduce a set of "bridging states", denoted by Y. Connecting the states in Y with those in X, we obtain a joint chain over the union space $X \cup Y$. If the joint chain is ergodic and has a stationary distribution in form of $(\alpha\boldsymbol{\mu}_X, (1-\alpha)\boldsymbol{\mu}_Y)$ then sampling from $\boldsymbol{\mu}_X$ is equivalent to drawing samples from $X \cup Y$ via the joint chain and discarding those from Y.

With a goal of constructing a joint chain that is ergodic and mixes rapidly, section 2.1 discusses the generic problem of bridging between two arbitrary finite Markov chains over disjoint state spaces such that the stationary distributions over the respective spaces are preserved. In section 2.2, we then derive bounds of the mixing time, which are influenced by two factors: the bottleneck ratio and laziness.

2.1 Bridging Markov Chains

Consider two finite state spaces X and Y. Suppose we are given two Markov chains: one over X with transition matrix $\mathbf{P}_X$ and stationary distribution $\boldsymbol{\mu}_X$, the other over Y with transition matrix $\mathbf{P}_Y$ and stationary distribution $\boldsymbol{\mu}_Y$. By introducing links that connect between X and Y, as shown in Figure 1, we derive the joint transition matrix, as

$$\mathbf{P}_+ = \begin{bmatrix} (1-b)\mathbf{P}_X & b\mathbf{Q}_B \\ f\mathbf{Q}_F & (1-f)\mathbf{P}_Y \end{bmatrix}. \tag{1}$$

Here, $\mathbf{Q}_B$ is a $|X| \times |Y|$ matrix, $\mathbf{Q}_F$ is a $|Y| \times |X|$ matrix, and each row in these matrices sums to 1. The behavior of the joint chain is described as follows: Starting from some $x \in X$ samples follow the original chain $\mathbf{P}_X$ with probability $1-b$ and jump to Y

with probability b landing at a particular state y with probability $\mathbf{Q}_B(x, y)$. Similarly, starting from $y \in Y$, sampling either stays in Y or jumps to X, respectively with probabilities $1 - f$ and f.

While sampling from the joint chain we wish to preserve the stationary distributions $\boldsymbol{\mu}_X$ and $\boldsymbol{\mu}_Y$ within respective spaces, meaning that $\mathbf{P}_+$ has a stationary distribution over $X \cup Y$, in form of $(\alpha\boldsymbol{\mu}_X, \beta\boldsymbol{\mu}_Y)$ with $\alpha + \beta = 1$. We derive the following lemma, which establishes the conditions under which this is satisfied.

Lemma 1. *The joint transition matrix $\mathbf{P}_+$ given by Eq.(1) has a stationary distribution in form of $(\alpha\boldsymbol{\mu}_X, \beta\boldsymbol{\mu}_Y)$, if and only if*

$$\boldsymbol{\mu}_X \mathbf{Q}_B = \boldsymbol{\mu}_Y, \quad \textit{and} \quad \boldsymbol{\mu}_Y \mathbf{Q}_F = \boldsymbol{\mu}_X. \tag{2}$$

Under this condition, we have $\alpha b = \beta f$. Further, if both $\mathbf{P}_X$ and $\mathbf{P}_Y$ are both reversible, then $\mathbf{P}_+$ is also reversible, if and only if

$$\boldsymbol{\mu}_X(x)\mathbf{Q}_B(x, y) = \boldsymbol{\mu}_Y(y)\mathbf{Q}_F(y, x), \tag{3}$$

for all $x \in X$ and $y \in Y$.

The proofs of this lemma and other lemmas and theorems that we develop are provided in appendix. We name the condition of Eq.(3) as *cross-space detailed balance.* With this construction, the *total probability of cross-space transition* is given by

$$\eta(b, f) \triangleq \alpha b + \beta f = 2\alpha b = 2\beta f = \frac{2bf}{b + f}. \tag{4}$$

The value of $\eta(b, f)$ reflects how frequently X and Y communicate with each other, which, as we shall see, is closely related to the mixing time of the joint chain.

We note that the matrix $\mathbf{Q}_{BF} \triangleq \mathbf{Q}_B\mathbf{Q}_F$ is a stochastic matrix, which actually represents a Markov chain over X, where each transition is via an intermediate state in Y. Intuitively, these chains utilize states in the other space to provide alternative transition routes, which, as stated by the following lemma, also lead to the same stationary distributions.

Lemma 2. *If the condition given by Eq.(2) holds, then $\boldsymbol{\mu}_X$ and $\boldsymbol{\mu}_Y$ are respectively stationary distributions of $\mathbf{Q}_{BF}$ and $\mathbf{Q}_{FB}$. Moreover, if $\mathbf{P}_+$ is reversible, then both $\mathbf{Q}_{BF}$ and $\mathbf{Q}_{FB}$ are reversible.*

On the other hand, as we will discuss in section 2.2, the ergodicity and the mixing time of the joint chain also depend on the characteristics of $\mathbf{Q}_{BF}$ and $\mathbf{Q}_{FB}$.

2.2 Mixing Time Analysis

The efficiency of a Markov chain is often measured by the *mixing time.* Given an ergodic Markov chain over X, with equilibrium distribution $\boldsymbol{\mu}$, the *mixing time* is

$$t_{mix}(\varepsilon) \triangleq \min\{t : \max_{x \in X} \|\mathbf{P}^t(x, \cdot) - \boldsymbol{\mu}\|_{TV} < \varepsilon\}. \tag{5}$$

We assume that the eigenvalues of $\mathbf{P}$ are $1 = \lambda_1 \geq \cdots \geq \lambda_n \geq -1$. Then, the *absolute spectral gap* of $\mathbf{P}$ is defined to be $\gamma_*(\mathbf{P}) \triangleq \min\{1 - \lambda_2, 1 + \lambda_n\}$. The theorem (Levin et al., 2008) below shows that the mixing time closely relates to this absolute spectral gap.

Theorem 1. *Given a reversible Markov chain with transition matrix $\mathbf{P}$, and $\varepsilon \in (0, 1/2)$, then*

$$\log(1/(2\varepsilon))(\tau - 1) \leq t_{mix}(\varepsilon) \leq \log(1/(\varepsilon\boldsymbol{\mu}_{min}))\tau. \tag{6}$$

Here, τ is called the relaxation time, given by $1/\gamma_*(\mathbf{P})$.

In general, a chain tends to have slow mixing when the absolute spectral gap is small, and when the gap is zero, the chain is non-ergodic and never mixes. The absolute spectral gap depends on two factors, namely the *bottleneck ratio*, which affects the value of $1 - \lambda_2$, *i.e.* the *spectral gap*, and the *laziness of transition*, which influences $1 + \lambda_n$.

2.2.1 Flows and Bottleneck Ratio

Given a Markov chain with transition matrix $\mathbf{P}$, which has a stationary distribution $\boldsymbol{\mu}$. For $x, x' \in X$, we define the *transition flow* (or simply *flow*) from x to x' to be $\mathcal{F}(x, x') \triangleq \boldsymbol{\mu}(x)\mathbf{P}(x, x')$. For a reversible chain, the flows are symmetric, *i.e.* $\mathcal{F}(x, x') = \mathcal{F}(x', x)$. The notion of flow can also be extended to sets. Let A and B be subsets of X, then the flow from A to B is defined to be $\mathcal{F}(A, B) \triangleq \sum_{x \in A} \sum_{x' \in B} \mathcal{F}(x, x')$.

Consider a partition of X into two subsets S and its complement S^c, then the *transition flow ratio* of S is $\Phi(S, S^c; \mathbf{P}) \triangleq \mathcal{F}(S, S^c)/\min\{\boldsymbol{\mu}(S), \boldsymbol{\mu}(S^c)\}$, where $\boldsymbol{\mu}$ is used as a measure, *i.e.* $\boldsymbol{\mu}(S) = \sum_{x \in S} \boldsymbol{\mu}(x)$. Taking the minimum of such ratio values of all partitions, we get the *bottleneck ratio*, which is formally defined as

$$\Phi_*(\mathbf{P}) = \min_{S \subset X} \left\{ \frac{\mathcal{F}(S, S^c)}{\min\{\boldsymbol{\mu}(S), \boldsymbol{\mu}(S^c)\}} : S, S^c \neq \emptyset \right\}. \tag{7}$$

Jerrum and Sinclair (1989) derived the theorem below that establishes both a lower and upper bound of the spectral gap in terms of bottleneck ratio.

Theorem 2. *Let λ_2 be the second largest eigenvalue of a reversible transition matrix $\mathbf{P}$, then*

$$\Phi_*^2(\mathbf{P})/2 \leq 1 - \lambda_2 \leq 2\Phi_*(\mathbf{P}). \tag{8}$$

This theorem shows that increasing the bottleneck ratio tends to expand the spectral gap, and thus reduce the mixing time. Through theoretical study, we found that the bottleneck ratio of the joint chain $\mathbf{P}_+$ given by Eq.(1) depends on both *how frequently the chain jumps between X and Y* and *how well the forward and backward links couple with each other*. The former is controlled by f and b, while the latter is mainly reflected by the spectral structure of the coupled chain: $\mathbf{Q}_{BF}$ and $\mathbf{Q}_{FB}$. We further derived specific bounds that characterize their relations:

Theorem 3. *The reversible transition matrix $\mathbf{P}_+$ as given by Eq.(1) has*

$$\frac{\eta(b,f)}{2} \cdot \frac{\phi}{\phi+1} \leq \Phi_*(\mathbf{P}_+) \leq \max\{b, f\}. \tag{9}$$

Here, $\eta(b,f) = 2\alpha b = 2\beta f$ is the total probability of cross-space transition, $\phi = \min\{\Phi_*(\mathbf{Q}_{BF}), \Phi_*(\mathbf{Q}_{FB})\}$.

This theorem gives both a lower bound and an upper bound of the bottleneck ratio of $\mathbf{P}_+$. We can see that the bottleneck ratio is influenced by two factors: (1) *the frequency of cross-space transition.* Frequent transition between X and Y generally results in high bottleneck ratio; while if the communication between them is inactive, the bottleneck ratio would be very low, leading to slow mixing. (2) *the bottleneck ratio of the collapsed chains.* High bottleneck ratios of the collapsed chains indicate that transition between different regions is made easy with the intermediate states, and thus the joint chain can mix rapidly. More importantly, from this theorem, we get

Corollary 1. *The joint chain $\mathbf{P}_+$ is ergodic when the collapsed chains ($\mathbf{Q}_{BF}$ and $\mathbf{Q}_{FB}$) are both ergodic.*

2.2.2 Laziness

Whereas increasing bottleneck ratio can enlarge the spectral gap, $1-\lambda_2$, the mixing time also depends on $1+\lambda_n$, the distance between λ_n and -1. In general, a reasonable value of $1+\lambda_n$ can be achieved by *laziness*.

Lemma 3. *Let $\mathbf{P}$ be a reversible transition matrix over X, such that $\mathbf{P}(x,x) \geq \xi > 0$ for each $x \in X$ then its smallest eigenvalue λ_n has $\lambda_n \geq 2\xi - 1$.*

This shows that by maintaining a probability $\xi > 0$ for the chain to stay (without transiting to other states), we can keep λ_n away from -1. Given an arbitrary reversible chain with transition matrix $\mathbf{P}$, we can make it lazier by changing $\mathbf{P}$ to $(1-\xi)\mathbf{P} + \xi\mathbf{I}$. However, it is worth noting that increasing the *laziness coefficient* ξ would on the other hand shrink the spectral gap from $1-\lambda_2$ to $(1-\xi)(1-\lambda_2)$. Hence, it is advisable to select a ξ that balances laziness and spectral gap.

3 Algorithms

Based on the theory of bridging Markov chains, we develop practical algorithms to construct the bridges and sample from the joint chain.

3.1 Construction of Bridges

Come back to our original problem of sampling from a distribution $\boldsymbol{\mu}_X$ over X, for which we can get a Markov chain $\mathbf{P}_X$ based on local moves. To improve the mixing, we introduce "bridges" to facilitate nonlocal transition. Specifically, we first choose a collection of state subsets of X: $S_1, \ldots, S_m$, and create a bridging state y_i for each S_i. In this way, we get a set of new states $Y = \{y_1, \ldots, y_m\}$. Suppose each target state in X has been covered by some such subset Next, for each $x \in X$, we set a transition probability $\mathbf{Q}_B(x, y_i) = 1/m(x)$ for each y_i associated with with it, *i.e.* $x \in S_i$, where $m(x)$ is the number of such bridges, and set $\mathbf{Q}_B(x, y_i) = 0$ when $x \notin S_i$. According to Lemma 1, we can construct $\mathbf{Q}_F$, the transition probabilities from Y to X, as follows

$$\mathbf{Q}_F(y_i, x) = \boldsymbol{\mu}_X(x) / \sum_{x' \in S_i} \boldsymbol{\mu}_X(x'). \tag{10}$$

It is not difficult to verify that the matrices $\mathbf{Q}_B$ and $\mathbf{Q}_F$ as above satisfy the cross-space detailed balance in Eq.(3), with $\boldsymbol{\mu}_Y$ given by

$$\boldsymbol{\mu}_Y(y_i) \propto \sum_{x \in S_i} \boldsymbol{\mu}_X(x'). \tag{11}$$

The values of f and b are set empirically. The guideline is to keep a balance between the local updates along the original chain and the transition via bridges.

Discussions: (1) The construction is very flexible. Given a specific problem, one can choose the subsets in any way that they see as best. For a problem where we have a clear perspective of the space structure, we can establish bridges that connect between the samples in different clusters to speed up the transition between them. (2) For problems with huge space, one layer of bridging can be very expensive. For such problems, we devise a novel sampling scheme called *hierarchical bridging* (see section 3.2), which provides a systematic way to derive an ergodic chain.

3.2 Hierarchical Bridging

For many problems, the underlying clustering structure of the sample space is largely unknown, and thus it is difficult to devise the bridges in advance. In the following, we describe a generic approach, which extends the construction above to a hierarchical framework that recursively builds bridges at multiple levels.

Initially, we have the target state space X, where each sample is a discrete vector, in form of $(x_1, \ldots, x_K)$. The target states constitute the 0-th level of the hierarchy. For the first level of bridging, we introduce a set of bridges, denoted by Y_1. Each bridge in Y_1

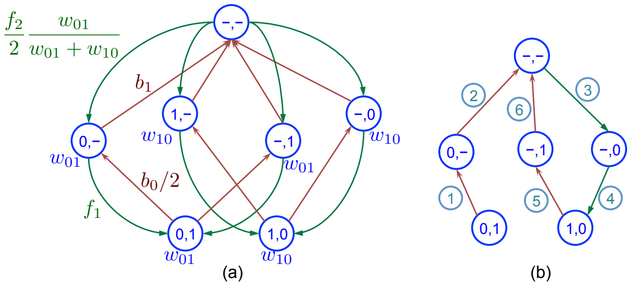

Figure 2: (a) shows the hierarchically bridging Markov chain on a simple problem: $x_1, x_2 \in \{0, 1\}$ with constraint $x_1 \neq x_2$. We use red color for the backward transitions from children to parents, and green for the transitions from parents to children. (b) illustrates a typical transition path. We use numbered circles to indicate the transition order. In this process, the bridges $(0, -)$ and $(-, -)$ are constructed upon the backward transition from a child state. When $(-, -)$ is instantiated, the right branch has not been visited, and the forward probability value for that branch is set with an optimistic estimate, encouraging the chain to visit that branch. Upon seeing $(1, 0)$, the forward probabilities of its parents will be updated accordingly.

corresponds to a partial assignment, *i.e.* a vector with one of the value removed. Take a state space $\{0, 1\}^3$ for example. Consider $(0, 0, 0) \in X$. By removing the middle value, we get a partial vector $(0, -, 0)$, where $-$ indicates a slot at which the value is removed. All vectors in form of $(0, x_2, 0)$, which include $(0, 0, 0)$ and $(0, 1, 0)$ here, are called the *children* of $(0, -, 0)$, and $(0, -, 0)$, in turn, is called the parent of them.

Given $b_0, f_1 < 1$, the transition between X and Y is described as follows. Starting from a target state $x \in X$, with probability $1 - b_0$, the chain stays in X, and with probability b_0, it transits to the parent of x in Y_1, by randomly removing a value. Note that a vector of length K has K different parents, and thus the transition probability from x to any particular parent is b_0/K. Starting from a bridge $y \in Y_1$, with probability $1 - f_1$, it stays at y, and with probability f_1, it transits back to X. In particular, the transition probability from $y = (x_1, \dots, -, \dots, x_K)$ to $x = (x_1, \dots, x_i, \dots, x_K)$ is proportional to $\boldsymbol{\mu}(x)$. To calculate this probability, one only have to evaluate of $\boldsymbol{\mu}(x)$ up to a scale. This is a useful property, as the normalization constant of a distribution is often difficult to evaluate in practical problems.

The construction of the hierarchy can be completed by recursively adding levels up to the root (the K-th level). Each bridge at the k-th level (denoted by Y_k) is a partially assigned vector with k entries removed. Starting from $y \in Y_k$, the probability of transiting to the upper level Y_{k+1} is b_k. Specifically, each $y \in Y_k$ has $K - k$ assigned values, and thus it has a probability $b_k/(K - k)$ to transit to any of its parent by randomly removing one of the assigned values. The chain also has a total probability f_k to transit to the lower level Y_{k-1}. To accomplish such a transition, we randomly pick one of the k unassigned slots (say the j-th entry), and draws a value for x_j, resulting a child state y'. The forward transition probability from y to y' is proportional to $\boldsymbol{\mu}_{k-1}(y')$. For any bridge state $y \in Y_k$, the value $\boldsymbol{\mu}_k(y)$ is defined via the recursive formula below

$$\boldsymbol{\mu}_k(y) \propto \sum_{y' \in Ch(y)} \boldsymbol{\mu}_{k-1}(y'). \tag{12}$$

When $k = 0$, $\boldsymbol{\mu}_0(x) \triangleq \boldsymbol{\mu}(x)$ for $x \in X$. Here, $Ch(y)$ is the set of y's children in Y_{k-1}. Through this construction, we obtain a joint chain over $X \cup Y_1 \cup \cdots \cup Y_K$, which we call the *hierarchically bridging Markov chain*, as illustrated in Figure 2(a). We derive the theorem below that characterizes this chain:

Theorem 4. *The hierarchically bridging Markov chain with $b_k < 1$ for $k = 0, \dots, K - 1$, and $f_k < 1$ for $k = 1, \dots, K$ is ergodic. If we write the equilibrium distribution in form of $(\alpha\boldsymbol{\mu}_0, \beta_1\boldsymbol{\mu}_1, \dots, \beta_K\boldsymbol{\mu}_K)$, then (S1) $\boldsymbol{\mu}_0$ equals the target distribution $\boldsymbol{\mu}$; (S2) for each $k \geq 1$, and $y \in Y_k$, $\boldsymbol{\mu}_k(y)$ is proportional to the total probability of its descendant target states (the target states derived by filling all its placeholders); (S3) α, the probability of being at the target level, is given by $\alpha^{-1} = 1 + \sum_{k=1}^{K}(b_0 \cdots b_{k-1})/(f_1 \cdots f_k)$.*

Here, we briefly explain the statements. (S1), together with the proved ergodicity, establishes the correctness of the construction, *i.e.* drawing samples from the joint chain and retaining only those from X amounts to directly sampling from $\boldsymbol{\mu}$. (S2) characterizes the distribution within other levels. (S3) gives the probability that a state drawn from the joint chain is a target state. From this statement, we derive

Corollary 2. *If $b_k/f_{k+1} \leq \kappa < 1$ for each $k = 1, \dots, K$, then $\alpha > 1 - \kappa$.*

This lower bound of α is independent from the number of levels K. Consequently, despite the problem scale, one can maintain a considerable chance of drawing a target state from the joint chain by keeping the backward/forward ratio below 1.

3.3 Dynamic Construction

Whereas the total number of bridges can be huge generally for a moderate problem, which however need not be explicitly instantiated prior to sampling. Instead, we can *build the chain progressively along with the sampling procedure.* As shown in Figure 2(b), except for the initial state that we start with, each state is instantiated only upon the first transition to it. In addition, we maintain references from each state to all its parents and children, to facilitate the transition from one state to another.

When a bridge state is constructed, one needs to determine the forward transition probabilities from this state to its immediate children. Exact evaluation of these probabilities requires complete knowledge of the distribution of all its descendants, which is generally unavailable upon the construction. A natural idea is to obtain such information by recursively visiting all the descendants. However, the complexity of this method can grow exponentially as we travel up along the hierarchy, making it infeasible in practice.

To address this issue, we adopt a dynamic programming strategy. Consider a bridge y at the k-th level with a set of children $Ch(y)$. Recall that for each child state $y' \in Ch(y)$, the forward transition probability from y to y' is proportional to $\boldsymbol{\mu}_{k-1}(y')$. If y' has been visited, then $\boldsymbol{\mu}_{k-1}(y')$ is immediately available. Otherwise, we initially use a quick estimate of $\boldsymbol{\mu}_{k-1}(y')$ and update it when y' and its descendants are visited. In general, one can overestimate the forwarding probability of an unvisited branch, thereby encouraging exploration of unknown regions. The initial value need not be accurate, as it is updated as the branch below y' is visited. A possible way to this quick estimation is to assume all assignments in that branch are valid (*i.e.* satisfying all constraints). For both applications described in next section, we employ this way, which results in an estimate as the product of the marginal probabilities of the available values.

In this scheme, the transition probabilities can change dynamically, resulting in time-inhomogeneity. In practice, such changes to the chain happen primarily during burn-in, and thus have negligible effect on asymptotic behavior. It is also worth noting that while the total number of states in X can be tremendous even for a problem with moderate size, our algorithm generally only visit those states with non-negligible probabilities. Though just constituting a small fraction of the entire space, they still provide a close approximation of the target distribution.

4 Application and Experiments

We assess the effectiveness of the proposed method on two problems: (1) constrained binary labeling and (2) partial order inference. Despite their different origins, both problems require sampling from a constrained combinatorial space, to which our method can be applied. Moreover, to demonstrate its practical utility, we also test the method in a real application, namely, inferring the partial order of objects in a video.

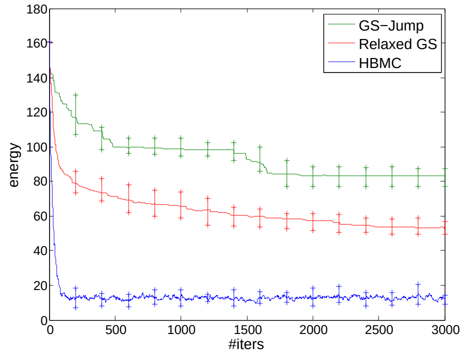

Figure 3: Each curve shows the mean energy values ($-\log p(x)$) as a function of elapsed iterations. Since Relaxed-GS and HBMC may yield states that are not in Ω, we use the energy of the last valid state as the energy value for an iteration. This also shows bars at 10% and 90% quantiles for 100 repeated runs.

4.1 Constrained Binary Labeling

Given a graph with n nodes and m edges, we are to set a binary label $x_i \in \{0,1\}$ to each node. Here, each edge is associated with a constraint on the labels of its two ends (*e.g.* $x_i \neq x_j$). We use an n-dimensional vector $x \in \{0,1\}^n$ to represent a label configuration, and use Ω to denote the set of all configurations that satisfy the constraints. In addition, each node has a preference function $w_i : \{0,1\} \to \mathbb{R}^+$. Then, we get a distribution over Ω, given by $p(x) \propto \prod_{i=1}^n w_i(x_i)$. While the probabilities are in a product form, the labels are not independent as they are related to each other via the constraints. This formulation actually stems from real world problems, such as circuit design, scheduling, and object placement.

We first consider a 4-connected graph with 5×5 nodes. Though the graph might seem small, it is sufficient to generate a large enough state space (up to 2^{25}), where the differences of algorithm behaviors can be clearly seen. Importantly, with this scale, it is feasible to evaluate the entire distribution through enumeration, enabling direct comparison between the sample distribution and the true one. To obtain a constrained problem, we randomly draw a constraint for each edge from a set of constraints ($x_i = x_j$, $x_i \neq x_j$, $x_i = 1$ or $x_j = 1$, etc). In this way, we generate a set of 20 constrained labeling problems as a testbed.

On these problems, we compare three algorithms: (1) *Gibbs sampling with long jump (GS-Jump)*: a method adapted from the one proposed by Barrett and Simma (2005). At each iteration, we update all variables by Gibbs sampling, and then propose a jump to arbitrary configuration drawn from the product distribution, accepting it if the result is valid. (2) *Relaxed Gibbs sampling (Relaxed-GS)*: similar to Walk-

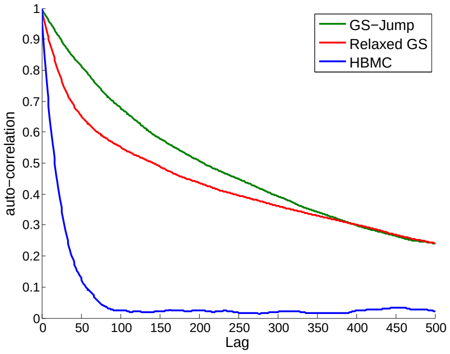

Figure 4: The energy auto-correlation function.

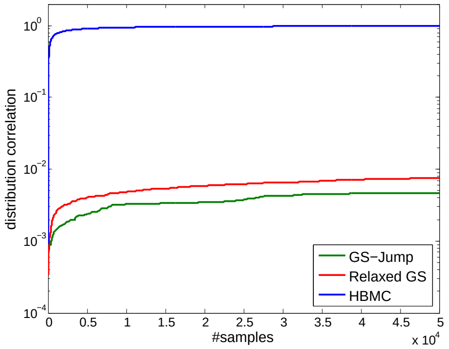

Figure 5: The correlations between the empirical distributions of the collected samples and the true distribution. Note that the y-axis is at log-scale.

SAT (Wei et al., 2004; Kitchen and Kuehlmann, 2009), we modulate the probability with a factor $\exp(-c \cdot \#\{\text{violated constraints}\})$, and turn the constrained model into an unconstrained MRF, upon which Gibbs sampling is applied. Here, c is empirically set to balance approximation accuracy and sampling efficiency, (3) *Hierarchical Bridging Markov Chain (HBMC)*: this is our approach. Here, we set $b_0 = 0.5$, meaning that starting from a target state, the chain performs a Gibbs update with 50% chance, and transits to upper level with 50% chance. For all other levels, we set $b_k = 0.4$ and $f_k = 0.6$. Each iteration consists of 25 walks, just like the other methods in comparison.

Figure 3 compares the energy trajectories obtained from 100 independent runs on a constrained problem as described above. We can see that GS-Jump gets stuck locally before a long jump is accepted, which rarely happens (once per over 1000 iterations on average). By allowing violation of constraints with cost, Relaxed-GS escapes from local traps, though rather slowly. HBMC significantly outperforms the other methods. Initially, encouraged by the optimistic weights set for unseen branches, the HBMC sampler quickly travels over the sample space, and at the same time builds bridges at different levels. In this process, the forward probabilities will be updated, with small values set to the branches leading to unlikely regions. Consequently, the chain rapidly gets to the states with high probabilities and rarely travels away.

Using the energy trajectories, we compute the *autocorrelation function*, averaged over all runs on all problem sets (in total 2000 runs for each algorithm). The results are shown in Figure 4. For HBMC, the correlation decreases to 0.1 after 50 iterations, and samples obtained with an interval of 80 can be considered as independent. Significant correlation remains for the other two methods even after 500 iterations, indicating that the underlying chains mix slowly.

We also investigate how many samples are needed to approximate the underlying distribution. For this study, we choose a constrained problem of which the number of distinct samples is about $10,000$, and collect $50,000$ samples for each algorithm, each per 200 iterations. We compute the correlation between the empirical distribution $\tilde{\mathbf{p}}$ and the true distribution $\mathbf{p}$, *i.e.* $\mathbf{p}^T\tilde{\mathbf{p}}/\sqrt{\|\mathbf{p}\|\|\tilde{\mathbf{p}}\|}$. The results are shown in Figure 5. The sample distribution obtained via HBMC is significantly closer to the true distribution as compared to the other methods, obtaining a correlation of 0.9 after only $5,000$ samplers. The other two methods exhibit drastically slower behavior. After 10 times greater samples, they remain stuck in a low-probability region with the correlation below 0.01.

4.2 Partial Order Inference

Partial order, namely a binary relation that is *reflexive*, *anti-symmetric*, and *transitive*, plays a significant role in various problems. Here, we consider the layered video model in computer vision, which has a partial order of layers at its core. Layered video modeling, initially proposed by Wang and Adelson (Wang and Adelson, 1994) and followed by a significant series of improvements (Weiss, 1997; Weiss and Adelson, 2006; Sun et al., 2010), is a popular approach to decomposing videos. Moving objects within a scene are assigned to layers and each image frame in the video is generated by composing these layers according to their *Z-order*(*i.e.* depth order).

Consider a video with n foreground layers and a background layer. There is a depth order, denoted by R^t, among them. The i-th layer is associated with a covering domain D_i^t and an appearance template A_i^t, where $A_i^t(x)$ is the pixel value at location x for this layer at time t. If x is covered by multiple overlapping domains, the pixel value observed at time t, denoted by $I^t(x)$, is from the top layer. We use an indicator map L^t to maintain the association between pixels and lay-

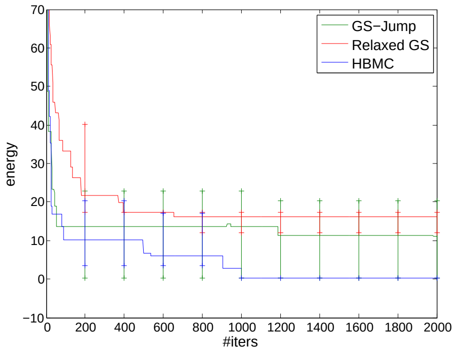

Figure 6: The mean energy as a function of elapsed iterations, with bars at 10% and 90% quantiles.

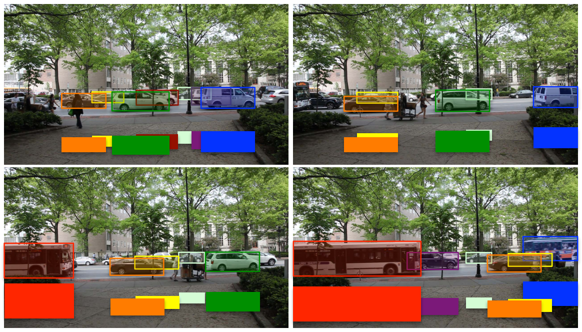

Figure 7: The inferred partial orders of vehicles in 4 frames of a video (interval = 3 sec). Vehicles are marked with transparent rectangles in different colors. Below them are opaque blocks that illustrate their Z-orders.

ers. Specifically, for each location x, $L^t(x)$ is the maximum of the layers that cover x with respect to R^t, *i.e.* top layer at x. Then, the frame at time t is generated as $I^t(x) = A_{L^t(x)}(x) + \varepsilon_t(x)$. Here, $\varepsilon_t(x)$ is a noise term. Whereas many previous models assume a total order between layers, it is, however, both unnecessary (one need not consider the ordering of disjoint layers) and inefficient. A natural idea to is to treat R^t as a partial order, and thereby dispense with unnecessary comparisons. Interested reader can refer to Kumar et al. (2005) or Weiss and Adelson (2006) for more details about layered video models.

Here, we are interested in inferring the partial order given the observed image frames, namely sampling from the posterior distribution $p(R^t|I^t; A^t)$. The first experiment here is to infer the partial Z-order of 6 rectangle objects from a noisy synthetic image. These objects together with their intersects divide the image into 18 sub-regions, each covered by the same set of objects. Hence, the problem can be reformulated as inferring the top layers of these regions in a consistent way (*e.g.* if a is the maximum (top) of the set $\{a, b, c\}$, then the maximum of $\{a, b, d\}$ can not be b). This is a constrained combinatorial problem.

In the posterior distribution, a small number of partial Z-orders actually get most of the probability mass. However, there remain some ambiguities due to appearance similarity and noises. To make it challenging, we use "the most wrong" sample (derived by reversing the true order) for each sampler to start with, such that they have to go through a long way to get it right. All three algorithms (GS-Jump, Relaxed-GS, and HBMC) are tested, each run for 100 times, with their settings tuned via multiple trials to get the best performance. The energy trajectories are shown in Figure 6. Again, HBMC consistently outperform the others, reaching the high-probability part of the sampling space within 100 iterations, and settling at the correct answer within 1000 iterations. GS-Jump can find the correct answer with random long jumps when lucky. But we can see a large variance of its performance, implying that it fails in a great portion of runs. Relaxed-GS gets stuck after 600 iterations, and have a trouble figuring out the path towards the right answer.

To assess its practical utility, we applied our method to solve a real world problem, namely inferring the partial Z-order of cars in a 10-minute long video of a busy avenue. The focus here is on sampling partial orders, rather than developing a full-fledged video model, and therefore we employ simple approaches for motion and appearance modeling. Specifically, we treat each car as an object layer, with a rectangular domain, and use Kalman filtering to update the positions of the cars and their templates. The Z-order is re-inferred each time based on the updates, using the previous Z-order as a prior. Part of the results are shown in Figure 7, which shows that our method performs very well in inferring the partial Z-orders, despite the simplicity of the motion and appearance models.

5 Conclusion

We proposed a general approach to sampling from constrained combinatorial spaces, via bridging, in order to address the issue of poor mixing (or even non-ergodicity) due to strong dependencies between variables caused by constraints. We performed both theoretical and empirical analysis of the proposed method, deriving bounds of the mixing time comparitive simulations with other methods. The results obtained on both constrained binary labeling and inference of partial Z-order of object layers clearly show that the proposed method, utilizing dynamically constructed bridging states, achieves remarkably better mixing performance than other methods in comparison.

References

Adrian Barbu and Song-Chun Zhu. Generalizing Swendsen-Wang to Sampling Arbitrary Posterior Probabilities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8), 2005.

Leon Barrett and Aleksandr Simma. MCMC With Disconnected State Spaces, 2005.

S. Duane, A.D. Kennedy, B.J. Pendleton, and D. Roweth. Hybrid monte carlo. *Physics Letters B*, 195(2), 1987.

Daniel Eaton and Kevin Murphy. Bayesian Structure Learning using Dynamic Programming and MCMC. In *Proc. of UAI'07*, 2007.

N. Friedman and D. Koller. Being Bayesian about network structure - A Bayesian approach to structure discovery in Bayesian networks. *Machine learning*, 50(1):95–125, 2003.

Peter J. Green. Reversible Jump Markov Chain Monte Carlo Computation and Bayesian Model Determination. *Biometrika*, 82(4), 1995.

P.J. Green. Trans-dimensional Markov chain Monte Carlo. *Highly structured stochastic systems*, 27, 2003.

Firas Hamze and N. de Freitas. Intracluster Moves for Constrained Discrete-Space MCMC. In *Proc. of UAI'10*, 2010.

M. R. Jerrum and A. J. Sinclair. Approximating the permanent. *SIAM Journal on Computing*, 18:1149–1178, 1989.

Nathan Kitchen and Andreas Kuehlmann. A Markov Chain Monte Carlo Sampler for Mixed Boolean/Integer Constraints. In *Computer Aided Verification*, pages 446–461, 2009.

M. Pawan Kumar, P.H.S. Torr, and A. Zisserman. Learning Layered Motion Segmentations of Video. In *Proc. of ICCV'05*, 2005.

David A. Levin, Yuval Peres, and Elizabeth L. Wilmer. *Markov Chains and Mixing Times.* American Mathematical Society, 2008.

Jian Pei, Haixun Wang, Jian Liu, Ke Wang, Jianyong Wang, and Philip S. Yu. Discovering frequent closed partial orders from strings. *IEEE Trans. on Knowledge and Data Engineering*, 18(11), 2006.

Deqing Sun, Erik B Sudderth, and Michael J Black. Layered Image Motion with Explicit Occlusions, Temporal Consistency, and Depth Ordering. In *Proc. of NIPS'10*, 2010.

R.H. Swendsen and J. Wang. Nonuniversal critical dynamics in monte carlo simulation. *Physics Review Letters*, 58(2), 1987.

Zhuowen Tu and Song-Chun Zhu. Image Segmentation by Data-Driven Markov Chain Monte Carlo. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5), 2002.

G Vahedi, I V Ivanov, and E R Dougherty. Inference of Boolean networks under constraint on bidirectional gene relationships. *IET systems biology*, 3(3):191–202, 2009.

B. Walsh. Markov Chain Monte Carlo and Gibbs Sampling, 2002.

John Y. Wang and Edward H. Adelson. Representing Moving Images with Layers. *IEEE Transactions on Image Processing*, 3(5):625–38, 1994.

Wei Wei, Jordan Erenrich, and Bart Selman. Towards Efficient Sampling : Exploiting Random Walk Strategies. In *Proc. of AAAI'04*, volume 000, 2004.

Yair Weiss. Smoothness in Layers: Motion Segmentation using Nonparametric Mixture Estimation. In *Proc. of CVPR'97*, 1997.

Yair Weiss and Edward H. Adelson. A Unified Mixture Framework for Motion Segmentation: Incorporating Spatial Coherence and Estimating the Number of Models. In *Proc. of CVPR'06*, 2006.