Markov Chain Monte Carlo and Image Restoration

Mohammed Sheikh

December 19, 2012

Abstract: The primary purpose of this paper is to illustrate the various concepts involved in Markov Chain Monte Carlo (MCMC), specifically the Metropolis algorithm. By using a process similar to annealing in metals and semiconductors, disordered initial states can be brought into the lowest energy configuration. The hope is that the lowest energy configuration in an image also lowers random distortions, such as noise, in an MCMC image restoration problem. This is due to the there being a phase transition in the image model as temperature is lowered similar to the transition towards a finite magnetization in the Ising model.

1 Introduction

1.1 Markov Chains

The idea of Markov chains is probably one of the most widely applicable and easily understandable ideas in probability. Roughly stated, a Markov chain (MC) is just like a physical system. It describes the evolution of a state. The state of an MC is the basic object of description. A good example of this is the English language. The state in this case would be the current letter. For example, in the sentence:

"A good friend will always stab you in the front."

The initial state of the MC can be taken to be the letter A. The second state is the white space. The third state is the letter g and so on. The basic question that this description asks is whether we can predict the next state given the previous states. In a Markov chain is that the future state of the system depends only on its previous state. Obviously, English is only approximately a Markov dhain. If we know the current state of the system is h, the best we can do is describe the probability that the next letter is an a, an e, or something else. For example, we know that $P(q|h) = 0$ because no word in the English language has q followed by h. Here $P(q|h)$ is the probability of the letter q occurring given the letter h. For a true MC, $P(q|h) = P(q|ah)$ and so on, but for the English language it is an approximation (and a very good one at that, see [1]).

Since the description of MCs involve probabilities, MC states are actually probability distributions. Usually we call the MC distribution $\pi(x)$ (here x is a variable, not the letter), so if we knew our current state was h, we would have $\pi(h) = 1$ and $\pi(x) = 0$ for $x \neq h$. Also, $\sum_x \pi(x) = 1$ and $\pi(x) \geq 0$ $\forall x$.Although MCs can be significantly more complicated in general, for a certain class of MCs, (called time-homogeneous, discrete state, discrete time) the evolution of the MC is governed by a matrix P. So perhaps the fundamental formula for MCs is

$$\pi_n = \pi_0 P^n$$

In this case, the subscript describes the current step of the system (so $\pi_0(A) = 1$ in the example above). P^n denotes the matrix exponent and $\pi_i(x)$ is a row vector, so that

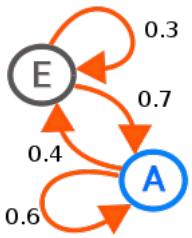

Figure 1: The graphical representation of a Markov Chain. In this case, $P =$

the product $\pi_i P$ is left matrix multiplication. The matrix P can be thought of as the probability of one step transitions between Markov states. So $P_{ij} = P(X_n = j|X_{n-1} = i)$. In general, $P = P(n)$, but for many useful applications P is independent of N.

The reason that MCs are so useful is that for a wide class of MCs, the states of the system all settle on a final, "equilibrium" state, so to speak. For the equilibrium state, we have

$$\pi = \pi P$$

This means that the equilibrium distribution π is the left eigenvector of P, which is what is usually used in practice to find π. The existence of π is guaranteed as long as it is possible for every state of an MC to go to every other state. The convergence of π is guaranteed under the condition that P does not cause any "loops" like $A \to l \to A \to l \ldots$

1.2 The Metropolis Algorithm

The primary motivation for introducing Markov Chains is for the Metropolis algorithm. The Metropolis algorithm provides a direct link between Markov Chains and statistical physics. The explaination will follow the introduction used in [3]. Imagine a lattice of spins that is relatively small, like $32 \times 32 = 2^{10}$. This is hardly a macroscopic problem. However, the possible number of configurations is $2^{2^{10}} \approx 10^{308}$ for a simple spin-$\frac{1}{2}$ system. We want to find some configuration that has minimum energy. If the spins are not coupled to each other in any way, then this problem is solvable analytically. However, most interesting problems have coupling between spins in some manner. We describe the energy of a system

$$E(\pi) = -\sum_{j \in \mathcal{N}(i)} J_{ij}(s_i, s_j) - \sum_i h_i s_i$$

Here $\mathcal{N}(i)$ denotes some neighborhood of the ith spin, $J_{ij}(s_i, s_j)$ is some coupling function between neighboring spins. The sum is assumed to have no repitition so pairs are only summed over once. The vector π is some vector describing the distributions of the spins. In general, the distribution of spins obeys the Boltzmann distribution

$$P(\omega) = \frac{e^{-\beta E(\omega)}}{Z}$$

where $\beta = \frac{1}{T}$ and Z is the partition function $Z = \sum_\omega P(\omega)$. To find the lowest energy distribution deterministically would require us to sample all 10^{308} distributions and increases exponentially with the number of spins. The Metropolis algorithm is a Monte Carlo algorithm which lets us probabilistically sample the distribution with a random walk.

Consider two distributions A and B, so that

$$\frac{P(A)}{P(B)} = \frac{e^{-\beta E(A)}}{e^{-\beta E(B)}} = e^{-\beta(E(A)-E(B))}$$

Note that the partition function does not appear in the ratio, so it doesn't have to be calculated. This saves a lot of computational effort, as calculating the partition function is as difficult as sampling the Boltzmann distribution. Then the Metropolis algorithm is

1. Start from some random configuration A. Calculate the energy $E(A)$, and consider a configuration B which is "close" to A by some metric (mean square distance or anything else).
2. Compute $E(B)$.
3. If $E(B) < E(A)$, move to B.
4. If $E(B) > E(A)$, move to B with probability $p = e^{-\beta(E(B)-E(A))}$

The relation of the Metropolis algorithm to Markov chains has to do with the transitions between states. If the current state of the system is A, then the future state of the system is determined only by the current state of the system. The future state of the system is simply a the current state of the system multiplied by a matrix (that may or may not be time dependent). Usually the temperature is varied to gain advantages based on the physical process of annealing, which will be described later. For the variation of the Metropolis algorithm described above, the state of the system converges to a Gibbs distribution (which is the stationary distribution of the system). By analogy with a physical system, to find the lowest energy state, the temperature $T \to 0$ as the algorithm evolves. This is closely related the maximum entropy reconstruction of a system given the average energy of the system.

There are two primary problems to solve with this approach. The first is how to choose configurations B that are "close" to A. The second is what to do with the temperature parameter β. These will be dealt with later. In general, a good intuition for why a Markov Chain Monte Carlo (MCMC) like the Metropolis Algorithm might converge is to look at the two limits. In the limit of $T \to 0$ we have that the MCMC will be a greedy algorithm. Given an initial state A, it will look for a state B depending on how we define "closeness." Similarly, in the limit of $T \to \infty$, the algorithm is a random walk over all states (it does not differentiate between energies). So the best intuitive description is that the MCMC is a directed random walk, which allows us to overcome many of the problems associated with typical greedy algorithms. Specifically, the MCMC will be more robust with respect to small local minima depending on the temperature T.

The hope of using the Metropolis algorithm is that at a low enough temperature, an ordered state emerges due to a phase transition similar to the 2D Ising model. This is useful when dealing with noisy data, as the noise can be seen as the random ordering of the Ising lattice at high temperatures. The hope for image restoration is that at low enough temperatures the Ising lattice settles on a state with a small noise level.

2 Simulated Annealing and Image Restoration

2.1 Introduction to Markov Random Fields on graphs

The problem of image restoration that is treated in [2] is the problem of denoising, deblurring, and applying a nonlinear transformation to a function. An image is a set $X = (F, L)$ where F is a matrix of observable pixel intensities and L is an unobservable dual matrix of quantities called edge elements. F is the *intensity process* and L is the *line process*. Essentially, F describes the image and L describes the transitions (i.e. edges or spatial derivatives) between pixels. This point of view for an image is hierarchical, since it considers an image as more than just a set of pixel values (although that is what we are trying to construct).

The matricies F and L can naturally be interpreted as graphs, where each vertex correponds to some coordinate (i, j) of the matrix. Usually, what we detect is the matrix $G = \phi(H(F)) + N$ where ϕ is some nonlinear transformation,

$$H(F) = \sum_{(k,l)} H(i-k, j-l)F_{k,l}$$

and N is random white Gaussian noise with mean 0 and variance σ^2. Here $H(F)$ represents blurring, which can be taken as a convolution in the image domain. Since the measured matrix is G, and G depends on the random variable N, we have a graph with random entries at the nodes.

To make use of the Metropolis algorithm in reconstruction of these images, we must make the Markov assumption. By making this assumption we are essentially determining the edges of the graphs of F and L. Edges in the graphs of F, L, or G represent dependencies between two pixels. These types of graphs are called Markov Random Fields (MRFs), with the prototypical example being the Ising Model. For a set for verticies $\mathcal{V} = \{v_1, v_2, \ldots, v_n\}$ we call the neighborhood of vertex v_i to be $\mathcal{N}_i = \{v_j | (i, j) \in E\}$ where E is the set of edges. A *clique* $C \subseteq V$ is the set of verticies such that every pair of verticies are neighbors. $\mathcal{C}$ denotes the set of all cliques. Essentially, cliques generalize the Markov property to include not just nearest neighbors, but any arbitrary neighborhood of a vertex. In the Ising Model, the set of cliques is the set of subsets of all nearest neighbor elements and all singleton elements (so $\{1\}$, $\{2\}$, $\{1, 2\}$, etc).

For the MRF we have that $P(X_s = a | X_r = b_r, r \neq s) = P(X_s = a | X_r = b_r, r \in N_s)$ where $s \in V$ and a and b_r are in Λ, the set of all possible values of an MRF vertex. Then our energy can be written

$$U(\omega) = \sum_{C \in \mathcal{C}} V_C(\omega)$$

We call a random field a *Gibbs ensemble* if and only if for each random variable X_s associated with a vertex $s \in \mathcal{V}$ and $\omega = (b_0, b_1, \ldots, a, \ldots, b_n) \in \Omega = \{\text{set of all possible states of the system}\}$

$$P(X_s = a | X_r = b_r, r \neq s) = Z_s^{-1} exp(-\frac{1}{T} \sum_{C: s \in C} V_C(\omega))$$

$$Z_s = \sum_{x \in \Lambda} exp(-\frac{1}{T} \sum_{C:s \in C} V_C(\omega^{s(x)}))$$

where $\omega^{s(x)} = (b_0, b_1, \ldots, s(x), \ldots, b_n)$ means that the potential is varied over all possible states of the vertex s and Λ is the set of all possible states of the vertex s.

A natural question to ask is what is special about the Boltzmann distribution. The general Metropolis algorithm can be formulated with any arbitrary distribution [1]. The reason to use a Boltzmann distribution in this case is because of the following theorem by Hammersley and Clifford and stated in [2]:

Theorem 1: Let (V, E) be a graph with neighborhood system $\mathcal{G}$. That is, for $\mathcal{N}_i \in \mathcal{G}$, $\mathcal{N}_i$ is the set of neighbors of $v_i \in V$. Then $X = \{X_i\}$ where X is indexed over V is an MRF with respect to $\mathcal{G}$ if and only if $\pi(\omega) = P(X = \omega)$ is a Gibbs distribution with respect to $\mathcal{G}$.

The interpretation of this theorem is relatively simple, although the original proof is quite cumbersome. An elementary proof is provided in [4]. Essentially, the theorem says that the graph (V, E) and the random variable X can only satisfy the Markov property if X has the Gibbs distribution. No other distribution satisfies the Markov property for a random field. This answers the question of why the Gibbs distribution is used in image restoration and related problems.

Generally, the elements of L and the elements of F are connected to each other. The purpose of the line process L is that it serves as a way of connecting the elements of F with each other over long distances without sacrificing the Markov property. Consider an arbitrary MRF X on the graph (V, E). Then the *marginal distribution* of X with respect to X_s is the summation of the distribution of X with respect to the range of X_s. This distribution also defines an MRF because the resulting sum can be put into Gibbs form. This new MRF $\hat{X}$ is related to the original MRF as follows. $x_1, x_2 \in \hat{V}$ are neighbors if either they were neighbors in (V, E) or if they were each a neighbor of x_s. So summing over all line elements in L gives us a graph where all the sites of F are neighbors. In other words, the marginal distribution of X with respect to L is a complete graph F. However, X retains the (useful) Markov property, while F retains only the trivial Markov property. Because of the line process L, long range interactions are still possible within F. Basically, L allows for long range order to be present in F.

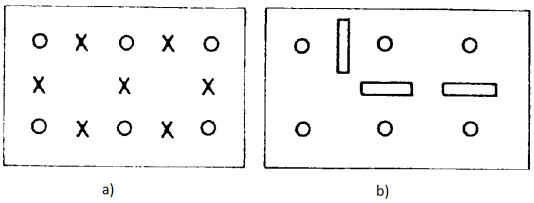

Figure 2: In figure 2(a), the pixel values are represented by circles. The seven x's represent line sites or edge sites. In figure 2(b) the line process is considered to be a binary line process such that each line may or may not be present

2.2 The Gibbs Sampler

The goal of the Metropolis algorithm is to find a state of minimum energy. Because of the importance of the Gibbs distribution in the algorithm, this specialized version of the Metropolis algorithm for image reconstruction is called the Gibbs Sampler by [2]. Besides the choice of distribution, the two primary problems of the Metropolis algorithm remain. There is a choice of a transition algorithm that goes between states in some random walk and the problem of what to do with the temperature.

The direct approach to a transition algorithm is to change the value of the pixels one at a time. The easiest way to do this is a raster scan by going along each row of pixels. Then the sequence of pixels is $n_1, n_2, \ldots$ where at step k we have $n_k \in S$. $S = \{(i,j)|0 \leq i \leq N, 0 \leq j \leq M\}$ for an $N \times M$ image. Call $X(t) = (X_{s_1}(t), \ldots, X_{s_{NM+L}}(t))$ as the random variable of the pixel and line sites. Then $X(t-1)$ and $X(t)$ can differ by at most one coordinate. If $s = n_t$, then we choose $X_s(t)$ randomly from the conditional distribution $P(X_s(t)|X_r(t), r \in \mathcal{N}_t(s))$.

Given this choice of transitioning, we have the following theorem from [2] which has the proof in the appendix:

Theorem 2: Assume that we visit each site infinitely many times. Then for every starting configuration $\eta \in \Omega$

$$\lim_{t\to\infty} P(X(t) = \omega | X(0) = \eta) = \pi(\omega)$$

Here Ω represents the space of all possible configurations of $X(t)$, π represents the Gibbs distribution, and $\omega \in \Omega$. So we are guaranteed that the Markov chain converges to the Gibbs distribution in this case. However, the theorem does not say whether or not the state ω is the lowest energy state. So far the temperature of the system has been fixed. The temperature plays the same role as it does in normal condensed matter systems. As the temperature approaches 0, we hope that the most probable state is the one with the lowest energy.

The reason for choosing the state with the lowest energy deals with the phase transition that happens in the 2D Ising model. In the 2D Ising model, it has been proven that as the temperature approaches 0, the overall magnetization becomes nonzero. We hope that as the temperature is lowered to 0, a similar effect will happen with the Metropolis algorithm. The random white Gaussian noise represents an analog to the thermal fluctuations that cause the Ising model to have 0 magnetization at termpatures above the Curie temperature. As the temperature is lowered, we hope that the defects due to the noise will disappear. If the temperature is lowered too quickly, disorder in the original state is frozen into the $T = 0$ state. The following theorem gives a soft upper bound on how quickly the temperature may be lowered.

Theorem 3: Let $\Omega_0 = \{\omega \in \Omega | U(\omega) = \min_\eta U(\eta)\}$ *and* π_0 *be the uniform distribution on* Ω_0. *Let* $U^* = \max_\omega U(\omega)$, $U_* = \min_\omega U(\omega)$, *and* $\Delta = U^* - U_*$. *For* $T(t)$ *with*

$$a) \lim_{t\to\infty} T(t) \to 0$$

$$b) T(t) \geq \frac{N\Delta}{\log t}$$

for $t \geq t_0$ with $t_0 \geq 2$, we have that $\forall \eta \in \Omega$ and $\forall \omega \in \Omega$,

$$\lim_{t \to \infty} P(X(t) = \omega | X(0) = \eta) = \pi_0(\omega)$$

In other words, the theorem says that if the temperature is lowered at a slow enough rate (slower than the rate of $\frac{C}{\log t}$), then the Gibbs Sampler converges to a minimum energy state. In practice, the constant is usually much too large (as it involves N, which is the total number of pixels). However, the general form $\frac{C}{\log t}$ serves as a good guideline. In fact, the entire idea of the Metropolis algorithm can be understood in terms of *annealing*, and the method used here is called *simulated annealing*. Annealing refers to the process of heating and then slowly cooling a metal or semiconductor to get rid of impurities. Generally, if the temperature is lowered too quickly, the impurities remain. The process of simulated annealing is analogous to the annealing of metals or semiconductors. Here, the impurities corrupt the images. In practice the constant $C \approx 3$ and $T(k) = \frac{C}{log(1+k)}$. Thus the process starts as a uniform distribution $T = \infty$ and cools after about 1000 iterations to $T = 0.5$.

2.3 Experimental Results

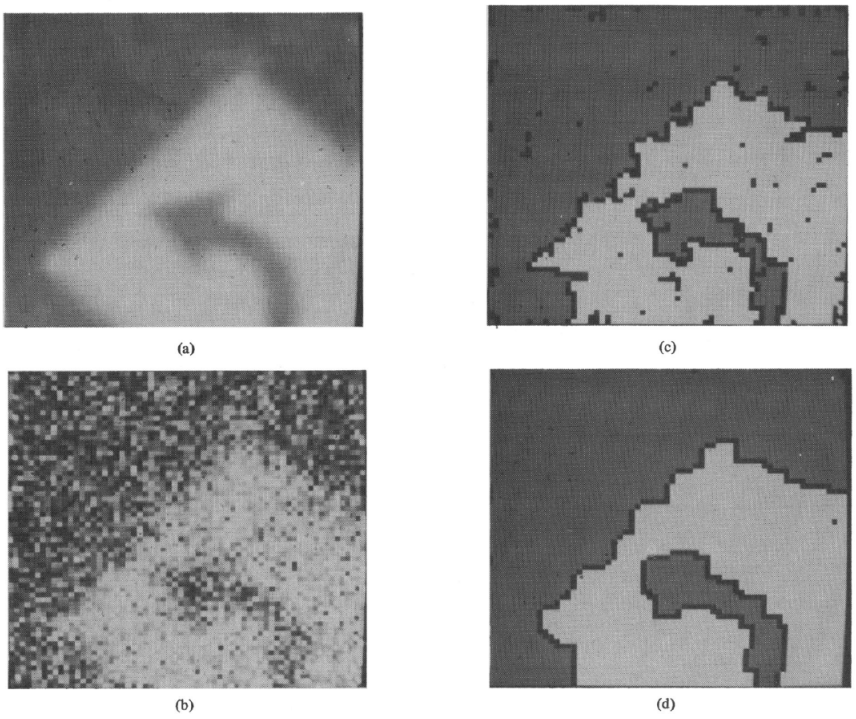

Figure 3: Figure 3(a) shows the image of a roadside sign that has been blurred. The image was then degraded with additive noise in 3(b). 3(c) shows the progression of the restoration after 100 iterations. 3(d) shows the restoration after 1000 iterations

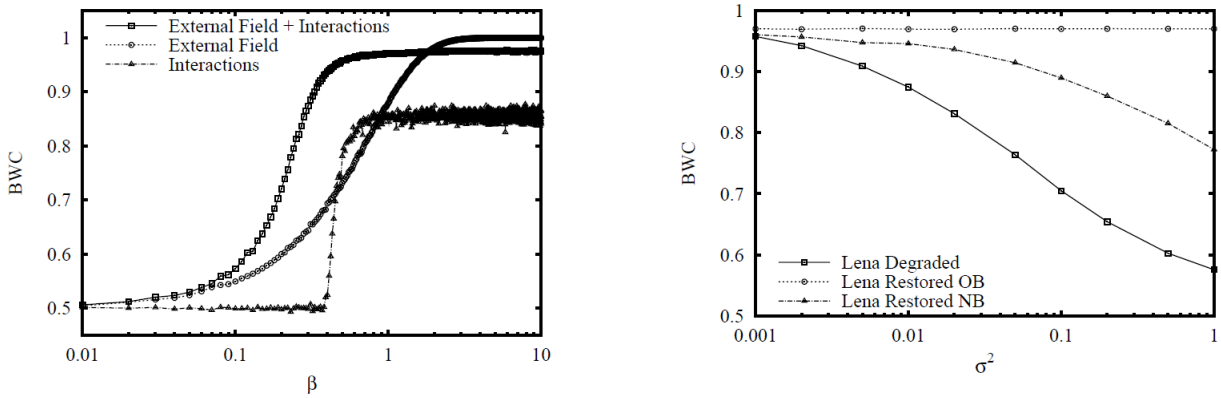

Figure 4: Figure 4(a): The vertical axis represents the fraction of bits correct to the total fraction of bits. As β is increased, the temperature approaches 0. 4(b) shows the fraction of bits correct to the total fraction of bits as a function of the additive Gaussian noise. Lena Restored OB uses the original image as a bias field so it is not surprising that it always converges.

The algorithm was run in [2] on various test images. It was found that the algorithm worked particularly well for edge detection, and this is not surprising given that the line processes were included explicitly in the image model. Specifically, the model is fairly simple and does not include any interaction between the intensity process. Intensities are limited to binary intensities so that the range of F is $\{0, 1\}$. The noise added to the image in figure 3 has standard deviation $\sigma = .5$, so that the signal-to-noise ratio is essentially 0 dB. Any pixel to the left or above an edge in the line process is colored black.

The algorithm is fairly successful at finding the edges in the noisy version of the blurred image. The energy function was explicitly made to find edges in this case. This is because edges obey the Markov property better than other parts of the image. Because of the Bayesian nature of the restoration, it is possible to incorporate other constraints and restore the image more accurately. This is not done in [2]. The *a posteriori* probability distribution is difficult to derive in the general case. In other words, an accurate energy function is difficult to find to restore images fully. The best ways to find *a posteriori* distributions is through evolutionary algorithms, and combining them with the simulated annealing provides a powerful image restoration tool. See [5] for the full description. Here we include a restored MRI picture using a combination of evolutionary and simulated annealing algorithms.

The algorithm was also run on an image in [6]. The transition between the ordered and disordered states is shown in figure (4). In [6] an external field was applied to bias the transition to happen in a certain direction, although this was not present in [2]. The image used was a binary image. BWC is the fraction of bits that are equal in the processed and noiseless image to the total number of bits. There were three models used in this case. A simple model used a bias field (singleton interactions in the cliques)

that was closely related to the image. Two other models were also used. One involved only interactions and the other involved both interactions and a bias field. From the picture, it can be seen that as $T \to 0$, a sharp transition occurs at $T \approx 3$. The quality of the image increases sharply at this point regardless of the model used. In this case, an analogy can be drawn between the BWC term and the magnetization in the Ising model. The reason that the model cannot perfectly reconstruct the image is the same as above. Essentially the model for the interactions introduces some bias in the reconstruction that causes it to deviate from the actual picture because the interactions are an imperfect description of the image. Note that the value of β is a log scale corresponding to the annealing schedule. From figure 4(b), we can also see that as the noise level is increased, the quality of the restored image decreases.

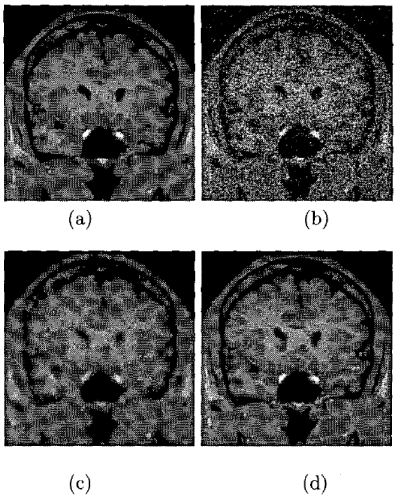

Figure 5: Figure 5(a) shows the an MRI of the head. Figure 5(b) shows the same image corrupted with white Gaussian noise. Figure 5(c) shows a restoration based on the Wiener filter. Figure 5(d) shows a restoration based on an evolutionary MRF algorithm in [5]. Note that the Wiener filter is linear but the MRF algorithm in [5] is nonlinear.

3 Other uses of the Markov Chain Monte Carlo

Another example of MCMC, and specifically the Gibbs sampler, is in a probabilistic solution to the traveling salesman problem. As one of the prototypes of NP-hard problems, the traveling salesman problems is the problem of finding the shortest path that visits every vertex (just think of the verticies as cities to visit) on a graph. The fastest known algorithms solve the problem in exponential time, much like the problem of finding the maxima of various probability distributions. In [3], the Gibbs sampler is used to give an approximate solution. In this case, the energy function is just the length of a certain path. Transitions between states are formed by swapping the destinations between two pairs of cities at random and reconnecting the graph. The MCMC solution is not

guaranteed to give the optimal solution, but approaches a relatively good solution in polynomial time.

MCMC comes with almost no guarantees in the general application of the algorithm. The Gibbs sampler is a special case when there is an established proof of convergence, but even then the rate of convergence is not known. Probably the most important open problems in the field relate to finding rates of convergence for the various MCMC algorithms in existence. However, for most practical applications of the algorithm, it seems that the rate of convergence is polynomial, or even approaching $O(n \log n)$.

References

[1] P. Diaconis, *The Markov Chain Monte Carlo Revolution*, American Mathematical Society **46** (2009) 179-208

[2] S. Geman, *Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images*, IEEE Pattern Analysis and Machine Intelligence **6** (1984) 721-741

[3] P. N. Saeta, *The Metropolis Algorithm*, available at `http://saeta.physics.hmc.edu/courses/p170/Metropolis.pdf`

[4] G. R. Grimmett, *A Theorem About Random Fields*, Bull. London Math. Soc. **5** (1973) 81-84

[5] T. Jiang, *3D MR image restoration by combining local genetic algorithm with adaptive pre-conditioning*, 15th International Conference on Pattern Recognition **3** (2000) 298-301

[6] M. Kandes, *Statistical Image Restoration via the Ising Model*, available at `http://www-rohan.sdsu.edu/~kandes/math336/project/paper.pdf`