Computing Network Reliability Coefficients*

Isabel Beichl and Elizabeth Moseman
National Institute of Standards and Technology
Gaithersburg, MD

Francis Sullivan
Center for Computing Sciences
Bowie, MD

May 4, 2011

Abstract

When a network is modeled by a graph and edges of the graph remain reliable with a given probability p, the probability of the graph remaining connected is called the reliability of the network. One form of the reliability polynomial has as coefficients the number of connected spanning subgraphs of each size in the graph. Since the problem of exact computation is #P-hard, we turn to approximation methods. We have developed two methods for computing these coefficients: one based on sequential importance sampling (SIS) and the other based on Monte Carlo Markov chain (MCMC). MCMC uses a random walk through the sample space while SIS draws a sample directly. There is not much theory available on the SIS method; however, this method is fast. In contrast, MCMC has a great deal of theory associated with it and is thus a more widely used and trusted method.

In order to properly use MCMC, two quantities are needed, the *mixing time*, the parameter which governs how long the algorithm must be run to get independent samples, and the *fugacity*, the parameter which governs the acceptance rates of proposed steps in the random walk. Despite the theory available on MCMC, both of these quantities are very difficult to calculate. As such, it is common practice to guess values for these parameters. This work focuses on the effectiveness of SIS in estimating these MCMC parameters in a given instance of the problem. Thus, we use SIS to speed up MCMC.

The study of network reliability begins with a simple sounding question. A network (such as the power grid) is modeled with a graph, and each edge has a stated probability of remaining connected. We wish to know the probability of

every vertex being connected. Unfortunately, as with many other simple sounding problems, we have yet to find a simple solution. This problem has been shown to be #P-hard[13], roughly meaning that almost everyone has lost hope of finding a simple solution. However, some things are known about reliability.

The formulation of the problem examined here begins with a graph $G = (V, E)$ and a probability, p, that an edge $e \in E$ is reliable. We adopt the convention that $|V| = n$ and $|E| = m$. Then the reliability of the graph is a polynomial in p known as the reliability polynomial. This polynomial may be expressed as

$$R(G; p) = \sum_{k=0}^{m-n+1} f_k \, p^{m-k} (1-p)^k$$

where the numbers f_k for $0 \leq k \leq m - n + 1$ are the number of connected spanning subgraphs $H \subseteq G$ with $m - k$ edges. Since we are only concerned with spanning subgraphs, we will frequently identify a subgraph H with its edge set and take the vertex set to be V. The size of H or $|H|$ represents the number of edges in H, since we know the number of vertices to be n.

For a comprehensive study of the subject of network reliability, see Colbourn[6], or the more recent survey of Chari and Colbourn[5]. This work gives only a brief overview of what is known. Several approximation methods for computing the value of $R(G; p)$ at a given value of p are known, the best known algorithm being due to Karger (see [9], [10]), which bypasses the computation of the coefficients f_k. Here, we focus on this computation, which is of independent interest. The work of Ball and Provan[1],[2] gives polynomial time bounds on these coefficients. For exact computation, Ball and Provan also give a polynomial algorithm to compute f_c, where the graph is $c - 1$-connected. This is improved by Ramanathan and Colbourn [14] to calculate f_{c+k} for any fixed k, but although polynomial this algorithm is still expensive. For the coefficients with higher indices, f_k with k closer to $m - n + 1$, the value f_{m-n+1} has long been known using the Kirchhoff formula for the number of spanning trees, but very little is known about other nearby coefficients. Myrvold [12] gives a polynomial time algorithm for computation of f_{m-n} in the case of planar graphs, but it has limited applicability in more general graphs. With this in mind, we turn to approximation methods.

As with other counting problems, one approach to approximating f_k is via Monte Carlo methods. A direct sampling approach was proposed by Colbourn et al [7]. In this method, a sample is created by selecting a random spanning tree and then adding a random subset of edges. A second Monte Carlo approach is to to create a Monte Carlo Markov Chain (MCMC), and this approach has been considered[4]. In this method, a random walk on the sample space is constructed and samples are taken after a fixed number of steps. The number of steps between samples, known as the mixing time, must be sufficient to achieve a distribution that is close enough to the steady state distribution. For this problem, utilizing the known general approaches to bounding the mixing time on has so far defeated

all theoretical attempts. Another approach to this type of problem is importance sampling[11], but Stockmeyer's results on the variance of this method[16] have drawn attention away from it. However, two of the present authors developed an algorithm for this problem using this method with reasonable results[3].

In this paper, the focus is on combining the results from the importance sampling algorithm with the more tried and true MCMC method to get results that are accepted at a much faster rate than might otherwise be the case. In Section 1 we outline the MCMC algorithm used to get our final results. Section 2 gives the method for calculating the mixing time of the MCMC. Since many of the theoretical aspects of the MCMC are still open, in Section 3 we give our algorithm for calculation of the necessary parameters. We finish in Section 4 with the results of some numerical simulations using the algorithms described in the previous sections.

1 Monte Carlo Markov Chain

The idea of a Monte Carlo Markov Chain is to take a random walk on the objects of interest, perturbing the current object just a little in order to get the next object. In this way, after sufficiently many steps, the current object is going to have a known distribution. This allows us to sample one of the objects of interest from a known distribution. There are established methods to translate an algorithm which samples the objects to one which counts the objects[8].

Our goal is to compute $f_k \equiv f_k(G)$, the number of connected, spanning subgraphs of size $m-k$ of the given graph $G=(V,E)$ when $|E|=m$. As such, the objects of interest will be connected, spanning subgraphs of G. We denote the set of all such subgraphs by $\mathcal{S}(G)$ and $\mathcal{S}_k(G)=\{X\in\mathcal{S}(G)\colon |X|=m-k\}$. Our tool is a MCMC so that the stationary distribution takes the form

$$\pi_\mu(X)=\frac{\mu^{m-|X|}}{Z(\mu)}$$

for every $X\in\mathcal{S}(G)$. The value μ is a positive real parameter, the fugacity, which will be specified later and $Z(\mu)$ is the *partition function* defined by

$$Z(\mu)\equiv Z_G(\mu)=\sum_{X\in\mathcal{S}(G)}\mu^{m-|X|}=\sum_{k=0}^{m-n+1}f_k\mu^k.$$

Once we have this MCMC, the coefficient f_k can be computed from the proportion of sample subgraphs which have size $m-k$, so the focus will be on computing $Z(\mu)$ for an arbitrary value of μ.[1] Since $R(G;p)=Z(\frac{1-p}{p})p^m$, computation of $Z(\mu)$ is in a sense more natural than computation of f_k, but the two are linked.

[1]This follows the treatment of monomer-dimer systems in [8]. We therefore skip steps that are completely analogous to that treatment and focus on the differences.

The Markov Chain $\mathcal{M}_S(G, \mu)$ will have transitions from any subgraph X according to the following rule:

Algorithm 1 Advancing in the MCMC walk $\mathcal{M}_S(G\,\mu)$

Step 1: with probability $\frac{1}{2}$, let $X' = X$; otherwise,

Step 2: select an edge $e \in E$ u.a.r. and set

$$X' = \begin{cases} X - \{e\} & \text{if } e \in X \text{ and this is connected;} \\ X + \{e\} & \text{if } e \notin X; \\ X & \text{otherwise;} \end{cases}$$

Step 3: go to X' with probability $\min\{1, \mu^{|X|-|X'|}\}$.

The idea of the transition rule is mostly in Step 2. Here, we select an edge uniformly at random. It may or may not be in our current subgraph. If it isn't, we add the edge, and if it is, we try to remove it. This removal could cause a problem, disconnecting the subgraph, but we won't remove it in this case. Step 3 adds a Metropolis rule to adjust the transition probabilities. With $\mu = 1$, we always move to X' in Step 3, but other values of μ decrease the probability of moving to larger (when $\mu < 1$) or smaller (when $\mu > 1$) subgraphs. It is left to the reader to check that $\mathcal{M}_S(G, \mu)$ is ergodic with stationary distribution π_μ. In order to calculate $Z(\hat{\mu})$, we select a sequence $0 = \mu_0 < \mu_1 < \mu_2 < \cdots < \mu_k \leq \hat{\mu}$. This sequence may be chosen arbitrarily, since the actual values of μ_i will not affect the resulting estimate. We then express $Z(\hat{\mu})$ as the product

$$Z(\hat{\mu}) = \frac{Z(\hat{\mu})}{Z(\mu_k)} \times \frac{Z(\mu_k)}{Z(\mu_{k-1})} \times \frac{Z(\mu_{k-1})}{Z(\mu_{k-2})} \times \cdots \times \frac{Z(\mu_1)}{Z(\mu_0)} \times Z(\mu_0)$$

where $Z(\mu_0) = Z(0) = 1$.

To estimate the ratio $z_k \equiv Z(\mu_{k-1})/Z(\mu_k)$, we express z_k as the expected value of the random variable $E_k = E_k(X) = \left(\frac{\mu_{k-1}}{\mu_k}\right)^{m-|X|}$, where X is a subgraph chosen from the distribution π_{μ_k}. To verify that E_k has the correct

expected value, we calculate

$$\begin{aligned}
\mathrm{E}(E_k) &= \sum_{X \in S(G)} \left(\frac{\mu_{k-1}}{\mu_k}\right)^{m-|X|} \pi_{\mu_k}(X) \\
&= \sum_{X \in S(G)} \left(\frac{\mu_{k-1}}{\mu_k}\right)^{m-|X|} \frac{\mu_k^{m-|X|}}{Z(\mu_k)} \\
&= \sum_{X \in S(G)} \frac{\mu_{k-1}^{m-|X|}}{Z(\mu_k)} \\
&= \frac{1}{Z(\mu_k)} \sum_{i=0}^{m-n+1} f_i \mu_{k-1}^i \\
&= \frac{Z(\mu_{k-1})}{Z(\mu_k)}
\end{aligned}$$

We summarize the algorithm for calculating $Z(\hat{\mu})$ in Algorithm 2.

Algorithm 2 Calculating $Z(\hat{\mu})$

Step 1: Select a sequence $0 = \mu_0 < \mu_1 < \mu_2 < \cdots < \mu_k \leq \hat{\mu}$.

Step 2: For each value $\mu = \mu_1, \mu_2, \ldots, \mu_k, \hat{\mu}$, compute as estimate Z_i of the ratio z_i by performing S_i independent simulations of the Markov Chain $\mathcal{M}_S(G)$, each of length m_i, to obtain an independent sample of size S_i from close to the distribution π_{μ_i}. The estimate Z_i is the sample mean of the quantity $E_i(X) = \left(\frac{\mu_{i-1}}{\mu_i}\right)^{m-|X|}$.

Step 3: Output the estimate $Z = \prod_{i=1}^{k+1} Z_i^{-1}$.

The number of steps m_i to reach the distribution π_{μ_i} will be discussed in Section 2, so the focus of the remainder of the section is on S_i, the sample size. We assert the following proposition is immediate from the considerations in the appendix to [8]:

Proposition 1.1. *Given a bound* $\mathrm{Var}[E_i]/(\mathrm{E}[E_i])^2 < B$ *on the relative variance of* E_i*, if the simulation length* m_i *is sufficient to ensure that the variation distance of* $\mathcal{M}_S(G, \mu_i)$ *from its stationary distribution is at most* $\epsilon/5Bk$*, then a sample size of* $S = \lceil 130B\epsilon^{-2}k \rceil$ *is sufficient to ensure the output variable* Z *satisfies*

$$\Pr\left((1-\epsilon)Z(\hat{\mu}) \leq Z \leq (1+\epsilon)Z(\hat{\mu})\right) \geq \frac{3}{4}.$$

Some remarks on the bound B are appropriate at this point. Since E_i only takes on values between 0 and 1, one choice to bound the relative variance is $\mathrm{Var}[E_i]/(\mathrm{E}[E_i])^2 \leq Z(\mu_i)/Z(\mu_{i-1}) = z_i^{-1}$. The inequality here is very generous, so for our algorithm we use an estimate of z_i^{-1} obtained from the SIS method as outlined in Section 3. For theoretical purposes, however, it is nice to have a general bound on this value. We assume that the coefficients f_k form a log-concave sequence[2] and that we choose the μ_i to fit the criteria $f_{i-1}/f_i \leq \mu_i < f_i/f_{i+1}$, so that $Z(\mu_i) \leq (m-n+1)f_i\mu_i^i$. This gives the bound

$$z_i^{-1} \leq (m-n+1)\left(\frac{\mu_i}{\mu_{i-1}}\right)^i.$$

In the numerical simulations, we selected $\mu_i \approx f_i/f_{i-1}$. A bridge is an edge whose removal will disconnect the graph. Define $\langle g_{i+1}\rangle$ as the mean number of edges in a subgraph of size $m-i$ which are not bridges. In Section 2, we show $(i+1)f_{i+1} = f_i\langle g_{i+1}\rangle$, so our choice of μ_i gives

$$\frac{\mu_{i+1}}{\mu_i} \approx \frac{\langle g_i\rangle}{\langle g_{i+1}\rangle}\left(1+\frac{1}{i}\right).$$

Lemma 1.2. *If $m > 2n$ and $\langle g_k\rangle/\langle g_{k+1}\rangle > T$ then $k > m(1-T/2(T-1))$.*

Proof. Suppose $\langle g_k\rangle/\langle g_{k+1}\rangle > T$. Then

$$m-k > \langle g_k\rangle > T\langle g_{k+1}\rangle > T(m-n+1-k-1)$$

The first inequality says that k edges have been removed by step k and the last inequality says that to get to a spanning tree, $m-n+1$ edges must be removed (of which $k+1$ have already been removed). Re-arranging and using the assumption $n < m/2$ gives

$$(1-T)(m-k) > -Tn$$

and so

$$m-k < (T/(T-1))n < (T/(T-1))(m/2)$$

hence

$$m[1-(1/2)(T/(T-1))] < k.$$

□

This gives us that the value μ_k/μ_{k+1} will be bounded for early k. For theoretical purposes, this bound can be used, but as we shall see in Section 4, even the smaller bound used in the simulations blows up for later k.

[2]The sequence $f_0, f_1, \ldots, f_{m-n+1}$ is conjectured to be log-concave, and there is strong evidence to support this.

2 Aggregated Monte Carlo Markov Chain

One technique for simplifying the mixing time analysis of a Markov chain is to combine the states. This process is called aggregation. Although the MCMC with combined states may not have the same mixing time, the mixing time of the new MCMC is a lower bound on the mixing time of the original. With reasonably chosen states, the transition probabilities of the aggregated MCMC may be computed.

With $\mathcal{M}_S(G, \mu)$, it is natural to aggregate into exactly $m-n+2$ states, each of which consists of all connected subgraphs of a specified size. Additionally, since we only worry about the size of the sample subgraphs in calculating the random variables E_k, this is the chain we really care about. This allows us to say that after waiting for the mixing time in the aggregated MCMC, the distribution of samples subgraph sizes is close enough to the stationary distribution, so this mixing time is sufficient.

We denote the set of subgraphs of size $m-i$ by $S_i(G)$. Then we wish to compute the transition probabilities, assuming that we are in the steady state. (This assumption is necessary so that the transition probabilities are constant.) By observing the MCMC algorithm, we know that $P(S_i(G), S_j(G)) = 0$ if $|i-j| > 1$.

Algorithm 3

Input: A graph G, a value μ and a total variation $\epsilon > 0$. Output: The mixing time m of the aggregated MCMC $\mathcal{M}_S(G, \mu)$.

Step 1: Estimate the values f_i for $0 \leq i \leq m - n + 1$. Call these estimates F_i.

Step 2: Use the estimates F_i to set up the aggregated transition matrix M_μ of $\mathcal{M}_S(G, \mu)$.

Step 3: Calculate the eigenvalues λ of M_μ. Define $\lambda_\mu = \max\{|\lambda| \colon |\lambda| < 1\}$.

Step 4: Output

$$m = \lceil (1 - \lambda_\mu)^{-1} (\ln m + \ln \epsilon^{-1}) \rceil$$

For the probabilities $P(S_i(G), S_j(G))$ with $|i-j|=1$, we have

$$\begin{aligned}
P(S_i(G), S_{i-1}(G)) &= \frac{1}{f_i} \sum_{X \in S_i(G)} \sum_{e \notin X} \frac{1}{2m} \min\{1, 1/\mu\} \\
&= \frac{i}{2m} \min\{1, 1/\mu\} \\
P(S_i(G), S_{i+1}(G)) &= \frac{1}{f_i} \sum_{X \in S_i(G)} \sum_{e \in X \text{not bridge}} \frac{1}{2m} \min\{1, \mu\} \\
&= \frac{1}{f_i} \frac{1}{2m} \min\{1, \mu\} \sum_{X \in S_i(G)} |\{e \in X \mid e \text{ not bridge}\}| \\
&= \frac{1}{2m} \min\{1, \mu\} \langle g_{i+1} \rangle
\end{aligned}$$

where $\langle g_{i+1} \rangle$ is the mean number of edges in a subgraph $X \in S_i(G)$ that are not bridges. The mean number of non-bridge edges in a graph of a given size seems difficult to count, but we may analyze it combinatorially. Beginning with the set $S_{i+1}(G) \times E$, consider the subset $\{(X, e) \mid e \notin X\}$. The size of this subset is exactly $(i+1)f_{i+1}$ because every subgraph of size $m-(i+1)$ has $(i+1)$ edges that are not in the set. We map $(X, e) \mapsto X \cup \{e\}$ and we may also measure the size of our domain by counting the multiplicity of a graph $Y \in S_i(G)$ in the resulting image. Since $(Y-e, e)$ is in the set for every e that is not a bridge, Y will have the same multiplicity as the number of edges in Y that are not bridges and the average multiplicity will be the expected number of edges that are not bridges. This gives the size of the subset as exactly $(i+1)f_{i+1} = f_i \langle g_{i+1} \rangle$ so

$$P(S_i(G), S_{i+1}(G)) = \frac{1}{2m} \min\{1, \mu\}(i+1)\frac{f_{i+1}}{f_i}.$$

The aggregated transition matrix is then given by

$$M_\mu = \begin{bmatrix}
1 - A_0 - B_0 & A_1 & 0 & \cdots & 0 \\
B_0 & 1 - A_1 - B_1 & A_2 & \cdots & 0 \\
0 & B_1 & 1 - A_2 - B_2 & \cdots & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & 0 & \cdots & 1 - A_\ell - B_\ell
\end{bmatrix} \tag{1}$$

where $\ell = m - n + 1$, $A_i = i/(2m) \min\{1, 1/\mu\}$, and $B_i = \frac{i\, f_i}{2m\, f_{i-1}} \min\{1, \mu\}$ for $1 \le i \le \ell$. Additionally, $A_0 = B_\ell = 0$. We compute the spectral gap, $1 - \lambda_M$, where λ_M is the second eigenvalue of the transition matrix. Then the mixing time is at most

$$(1 - \lambda_M)^{-1}(\ln(\pi(X))^{-1} + \ln \epsilon^{-1})$$

where $\pi(X)$ is the equilibrium probability of the initial state [15]. We will need to initialize $\mathcal{M}_S(G, \mu_i)$ in order to assure that each subgraph of a given size is

equally likely, but after that this mixing time is sufficient between samples to assure independence. One way to initialize is to repeatedly sample from $\mathcal{M}_S(G, \mu_{i-1})$ until obtaining a sample subgraph of size $m - k + 1$ (an event which occurs with probability at least $1/m$, so we do not expect to have to take many samples before this occurs) and use this as our initial subgraph. For $\mathcal{M}_S(G, \mu_1)$, we may initialize at G, since that is the only subgraph of size m. We then begin in the state $S_{i-1}(G)$ and $\pi(S_{i-1}(G)) = f_{i-1}\mu_i^{i-1}/Z(\mu_i) \geq 1/(m - n + 1)$. Algorithm 3 summarizes this method of calculating m_i, the mixing time of $\mathcal{M}_S(G)$ used in Step 2 of Algorithm 2.

Careful examination of Algorithms 2 and 3 reveals a conundrum: to calculate $Z(\mu_i)$, we need the mixing times m_i. However, to calculate m_i, we need the values f_i which are the coefficients of $Z(\mu_i)$. Which do we do first?

3 Sequential Importance Sampling and How it Helps

To answer the question of the previous section, we will estimate f_i first, using a different method. The method used here is SIS, and is explored in the context of this problem in [3]. The SIS algorithm is given in Algorithm 4.

Algorithm 4

Step 1: Set $a_0 = f_0 = 1$.

Step 2: Build a sequence of subgraphs $G = G_0 \supset G_1 \supset G_2 \supset \cdots \supset G_{m-n+1}$ so that $|G_i| = m - i$. To form G_i, let a_i be the number of edges in G_{i-1} which are not bridges. Select e uniformly at random from this set and let $G_i = G_{i-1} - \{e\}$.

Step 3: For $0 < k \leq m - n + 1$, output the estimates

$$f_k \approx \prod_{0<i\leq k} \frac{a_i}{k!}$$

The advantage of this algorithm as opposed to a MCMC algorithm is that it is very fast. Each time we run through this algorithm, we get a rough estimate of f_k for every k, which is much more information than one sample from the MCMC will give us, and this enables us to quickly approximate the matrix M_μ.

If the coefficients f_k are log-concave, the distribution π_μ will output a unimodal distribution of subgraph sizes. The peak of this distribution will be with subgraphs of size $m - i$ where $f_{i+1}/f_i < \mu < f_{i+2}/f_{i+1}$. With the SIS algorithm, the ratios of the coefficients will be reasonably well approximated, so we can quickly tell where any distribution peaks. In addition, if $\mu = \frac{1-p}{p}$, then we

know that the terms $f_k\mu^k$ with $k > i+1$ are decreasing, so getting good approximations of the later coefficients is decreasingly important in computing the reliability.

We may also use SIS to approximate the bound B to be used in calculation of the sample size. Defining $\hat{z}(\mu) = \sum_k F_k\mu^k$, we have that $Z(\mu) \approx \hat{z}(\mu)$. Although we do not have that $Z(\mu_{i+1})/Z(\mu_i) \leq \hat{z}(\mu_{i+1})\hat{z}(\mu_i)$, using this as a bound for the relative variance of the random variable Z_i is reasonable because there was already some leeway introduced in calculating that $\mathrm{Var}[E_i]/\mathrm{E}[E_i]^2 \leq Z(\mu_{i+1})/Z(\mu_i)$.[3] We summarize the complete process used to calculate f_k in Algorithm 5. This is the algorithm used for the numerical simulations.

Algorithm 5 A complete algorithm for computing an estimate of f_k, the number of connected spanning subgraphs of size $m-k$ in a given graph G.

Step 1: Use Algorithm 4 to obtain m samples $F_k^{(1)}, F_k^{(2)}, \ldots, F_k^{(m)}$ of f_k for $0 \leq k \leq m-n+1$. Let $F_k = m^{-1}\sum_{j=1}^{m} F_k^{(j)}$ be our first approximation of f_k.

Step 2: Define $\mu_k = F_{k-1}/F_k$. Use F_i to approximate f_i in M_{μ_k} (See Equation 1) and λ_k as the second eigenvalue of the resulting matrix.

Step 3: Define $\zeta(\mu) = \sum_{k=0}^{\ell} F_k\mu^k$. For each k between 1 and $m-n+1$ (inclusive), define

$$\begin{aligned} m_k &= \left\lceil (1-\lambda_k)^{-1}(m + \ln(5m^2\mathrm{e}/\epsilon)) \right\rceil, \text{ and} \\ s_k &= \left\lceil 130\epsilon^{-2} m \frac{\zeta(\mu_i)}{\zeta(\mu_{i-1})} \right\rceil. \end{aligned}$$

Perform s_k independent simulations of $\mathcal{M}_S(G, \mu_k)$, each of length m_k, to obtain a sample $\mathcal{S}_k$ of size s_k. Then set Z_k as the sample mean of $E_k = (\mu_{k-1}/\mu_k)^{m-|X|}$ over $X \in \mathcal{S}_k$ and $I_k = |\{X \in \mathcal{S}_k \colon |X| = m-k\}|/s_k$.

Step 4: Output the estimates

$$f_k \approx \frac{1}{\mu_k^k} \frac{I_k}{\prod_{i=1}^{k} Z_i}$$

for each $1 \leq k \leq m-n+1$.

---

[3] In fact, the exact value of the relative variance is $\frac{\mathrm{Var}[E_k]}{(\mathrm{E}[E_k])^2} = \frac{Z(\mu_{k-1}^2/\mu_k)}{Z(\mu_{k-1})} \frac{Z(\mu_k)}{Z(\mu_{k-1})} - 1$.

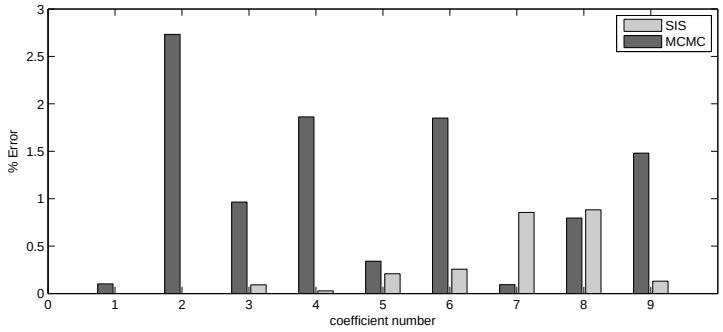

Figure 1: The percentage error of the coefficients for G_1 when estimated using SIS and MCMC methods.

4 Numerical Simulation

The numerical simulations here serve as a proof of concept. In particular, we are hoping to validate the claims we make about the SIS calculations for the MCMC parameters. To do this, we need to analyze the fugacities, sample sizes and mixing times selected. The SIS and MCMC algorithms were programmed in C++ and initially tried on small graphs. All trials were performed with $\epsilon = 1$.

We first ran the simulation with G_1, a graph of 7 vertices and 15 edges. This graph is small enough that we were also able to compute exact coefficients f_k for every k. All three methods of computing coefficients, exact, SIS, and MCMC, were nearly indistinguishable when the vertical axis was logarithmic (necessary with the large value changes in the f_k), so Figure 1 plots the percentage error in the SIS and MCMC estimates for each k. The values computed are shown in Table 1.

The value of the coefficients and the error in the coefficients serves as an indication that the algorithm works on the graph G_1. However, we wish to more closely evaluate other indicators of the effectiveness of the algorithm. The first indicator will be the distribution of sample subgraph sizes for each fugacity μ_i. Our selection of $\mu_i = F_{i-1}/F_i$ is intended to produce a high proportion of sample subgraphs with $m - i$ or $m - i + 1$ edges, and other subgraph sizes will occur less frequently. The distribution of subgraph sizes is shown in Figure 2. From the figure, we see that our selection does give the desired distribution of subgraph sizes for every μ_i.

The second parameter calculated is the mixing time. Figure 2 also serves to validate the mixing time selection since we see no artifacts that we would expect to see if the samples were not independent. In addition, for this graph, we were able to compute exact coefficient values, and from those we could compute expected distributions for each value of μ_i. By considering each proportion of

Table 1: The resulting coefficients, using SIS, MCMC, and an exact algorithm, along with the percentage error using each approximation method.

Index	Actual	SIS	% Error	MCMC	% Error
0	1	1	0.00	1	0.00
1	15	15	0.00	15	0.10
2	105	105	0.00	102	2.73
3	454	454	0.09	450	0.96
4	1350	1350	0.03	1325	1.86
5	2900	2906	0.21	2890	0.34
6	4578	4590	0.26	4493	1.85
7	5245	5200	0.86	5250	0.09
8	4092	4056	0.88	4060	0.80
9	1728	1726	0.13	1702	1.48

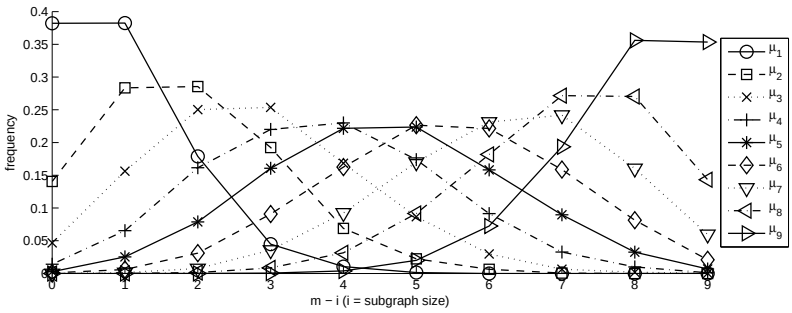

Figure 2: For the graph G_1, observed subgraph size frequencies for μ_i where i varies between 1 and 9. Each μ_i was given $130 * m * \hat{z}(\mu_i)/\hat{z}(\mu_{i-1})$ samples where $\hat{z}(\mu) = \sum F_i \mu^{m-i}$ and $\mu_i = F_{i-1}/F_i$ and the F_i are taken from SIS.

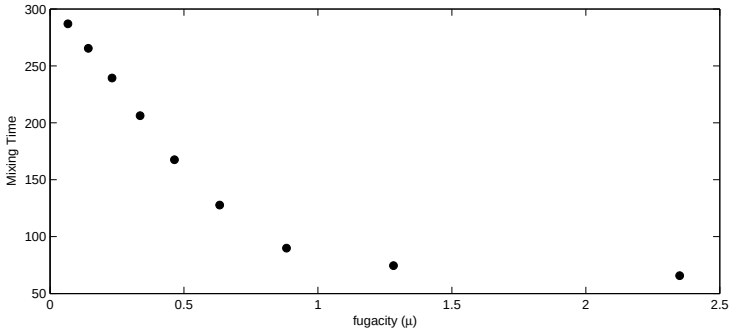

Figure 3: Mixing time against fugacity for G_1.

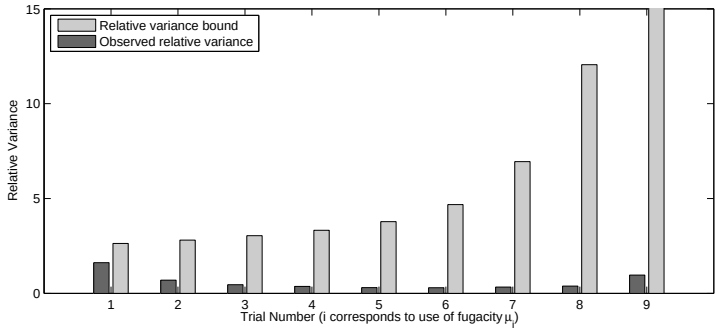

Figure 4: The relative variance of the random variable E_i, both the observed value and the bound calculated from SIS. The bound for the relative variance of E_9 was 96.

sample subgraph sizes as a sequence of Bernoulli trials, we calculate variance and standard deviation as well. In this case (this run on the graph G_1) we had 71% of the observed proportions within one standard deviation of the expected proportion. The mixing time depends on both the fugacity (μ) and the underlying graph (G_1). The values computed are shown in Figure 3. These were also compared to the mixing time if the aggregated transition matrix were computed exactly, but in all case there was less than 1% error, so we chose not to display both on the graph.

The third parameter calculated is the sample size. This is sufficient when the observed relative variance of E_i is less than the calculated bound. A comparison of these values is shown in Figure 4.

In general, we are very happy with the results on the graph G_1 and wished to continue our trials with larger graphs about which we had less information. The

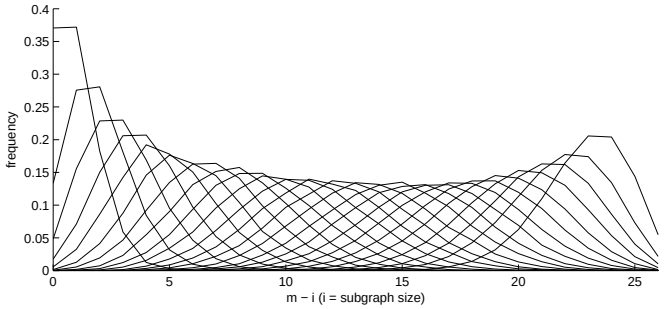

Figure 5: The observed subgraph size distribution on a 5 by 5 toroidal lattice when subgraphs are selected using MCMC with fugacity μ_i. Peaks correspond to increasing values of i from left to right. 24 different fugacities are represented.

distributions of sample sizes from running on a 5×5 toroidal lattice ($n = 25$ and $m = 50$) and relative variance are shown in Figure 5 and 6, respectively. In Figure 5, we again see the progression of the peaks of the distribution as i increases, so that the distribution π_{μ_i} peaks around subgraphs of size $m - i$, and there are no artifacts that would indicate the samples are not independent. In Figure 6, we see that the relative variance bounds were again much higher than the observed relative variance, so our sample sizes were sufficient. Based on the estimated sample sizes, we determined not to run the MCMC with the fugacities μ_{25} and μ_{26} due to time constraints, but the behavior in the remaining fugacities and the calculation of the coefficients proceeded as before.

Further trial graphs were selected, with similar results, but these results are omitted in the interests of space.

5 Conclusion

We have outlined a MCMC method for calculating the coefficients of the reliability polynomial. While there is a great deal of room for improvement on the theory for this algorithm, we make use of an existing method of calculating these coefficients in order to solve empirically the problems that do not yet have theoretical solutions. In doing so, we have greatly reduced the running time of the algorithm to something that is manageable for small graphs, achieving improvements in estimation of fugacities, mixing times, and sample sizes.

With the fugacity, we give an algorithm for selecting fugacities so that no more than $n - m$ fugacities will completely explore the sample space. The choice of the fugacities is shown to fill the requirements with no guessing involved.

With the mixing time, we empirically determine the mixing time for each

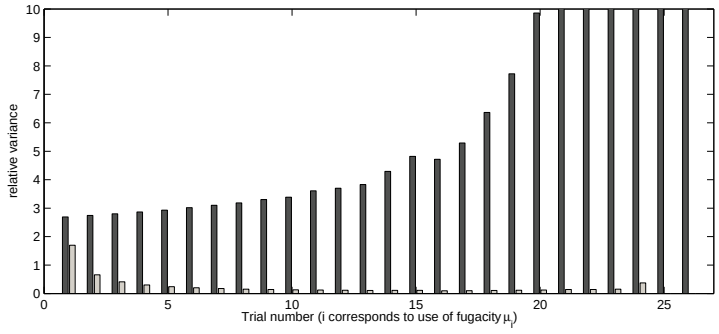

Figure 6: The observed relative variance and calculated bound for the same MCMC runs.

graph and fugacity, insuring that the minimum possible mixing time is used. The mixing times selected are demonstrated to be sufficient for the instances studied.

With the sample size, we select a sample size based on a variance bound, but there is still room for improvement.

Altogether, we feel that this method of accelerating MCMC methods could be very applicable to other problems. Further research will be focused on developing SIS methods for other graph theory problems and incorporating the results into new or existing MCMC algorithms.

References

[1] Michael O. Ball and J. Scott Provan. Bounds on the reliability polynomial for shellable independence systems. *SIAM J. Algebraic Discrete Methods*, 3(2):166–181, 1982.

[2] Michael O. Ball and J. Scott Provan. Calculating bounds on reachability and connectedness in stochastic networks. *Networks*, 13(2):253–278, 1983.

[3] Isabel Beichl, Brian Cloteaux, and Francis Sullivan. An approximation algorithm for the coefficients of the reliability polynomial. *Congr. Numer.*, 197:143–151, 2009. Proceedings of the Fortieth Southeastern International Conference on Combinatorics, Graph Theory and Computing.

[4] Adam L. Buchsbaum and Milena Mihail. Monte Carlo and Markov chain techniques for network reliability and sampling. In *Computational support for discrete mathematics (Piscataway, NJ, 1992)*, volume 15 of *DIMACS Ser.*

Discrete Math. Theoret. Comput. Sci., pages 199–222. Amer. Math. Soc., Providence, RI, 1994.

[5] Manoj Chari and Charles J. Colbourn. Reliability polynomials: a survey. *J. Combin. Inform. System Sci.*, 22(3-4):177–193, 1997.

[6] Charles J. Colbourn. *The combinatorics of network reliability*. International Series of Monographs on Computer Science. The Clarendon Press Oxford University Press, New York, 1987.

[7] Charles J. Colbourn, Bradley M. Debroni, and Wendy J. Myrvold. Estimating the coefficients of the reliability polynomial. *Congr. Numer.*, 62:217–223, 1988. Seventeenth Manitoba Conference on Numerical Mathematics and Computing (Winnipeg, MB, 1987).

[8] Mark Jerrum and Alistair Sinclair. Chapter 12: The markov chain monte carlo method: an approach to approximate counting and integration. In Dorit Hochbaum, editor, *Approximation Algorithms for NP-Hard Problems*. Course Technology, 1 edition, July 1996.

[9] David R. Karger. A randomized fully polynomial time approximation scheme for the all-terminal network reliability problem. *SIAM J. Comput.*, 29(2):492–514 (electronic), 1999.

[10] David R. Karger and Ray P. Tai. Implementing a fully polynomial time approximation scheme for all terminal network reliability. In *Proceedings of the Eighth Annual ACM-SIAM Symposium on Discrete Algorithms (New Orleans, LA, 1997)*, pages 334–343, New York, 1997. ACM.

[11] Donald E. Knuth. Estimating the efficiency of backtrack programs. *Math. Comp.*, 29:122–136, 1975. Collection of articles dedicated to Derrick Henry Lehmer on the occasion of his seventieth birthday.

[12] Wendy Myrvold. Counting k-component forests of a graph. *Networks. An International Journal*, 22(7):647–652, 1992.

[13] J. Scott Provan and Michael O. Ball. The complexity of counting cuts and of computing the probability that a graph is connected. *SIAM J. Comput.*, 12(4):777–788, 1983.

[14] Aparna Ramanathan and Charles J Colbourn. Counting almost minimum cutsets with reliability applications. *Mathematical Programming*, 39(3):253–261, 1987.

[15] Alistair Sinclair. Improved bounds for mixing rates of Markov chains and multicommodity flow. *Combin. Probab. Comput.*, 1(4):351–370, 1992.

[16] Larry Stockmeyer. On approximation algorithms for #P. *SIAM J. Comput.*, 14(4):849–861, 1985.