Design of Reliable Communication Networks (DRCN) 2003, Banff, Alberta, Canada, October 19-22, 2003

Network Reliability Estimation Using The Tree Cut and Merge Algorithm with Importance Sampling

K.-P. Hui*, N.G. Bean †, M. Kraetzl‡ and D. Kroese§
* IN Division, Defence Science & Technology Organisation, Edinburgh 5111, Australia
Email: Kin-Ping.Hui@dsto.defence.gov.au
† School of Applied Mathematics, University of Adelaide, Adelaide 5005, Australia
Email: nbean@maths.adelaide.edu.au
‡ ISR Division, Defence Science & Technology Organisation, Edinburgh 5111, Australia
Email: Miro.Kraetzl@dsto.defence.gov.au
§ Department of Mathematics, University of Queensland, Brisbane 4072, Australia
Email: kroese@maths.uq.edu.au

***Abstract*— It is well known that the exact calculation of network reliability is a #P-complete problem and that for large networks estimating the reliability using simulation techniques becomes attractive. For highly reliable networks, a Monte Carlo scheme called the *Merge Process* is one of the best performing algorithms, but with a relatively high computational cost per sample. The authors previously proposed a hybrid Monte Carlo scheme called the *Tree Cut and Merge* algorithm which can improve simulation performance by over seven orders of magnitude in some heterogeneous networks. In homogeneous networks, however, the performance of the algorithm may degrade. In this paper, we first analyse the Tree Cut and Merge algorithm and explain why it does not perform well in some networks. Then a modification is proposed that subdivides the problem into smaller problems and introduces the *Importance Sampling* technique to the simulation process. The modified algorithm addresses the slow convergence problem in those hard cases while keeping the performance improvement in heterogeneous networks. Experiments and results are presented with some discussions.**

***Index Terms*— Network Reliability, Monte Carlo, Importance Sampling, Merge Process, Markov Process**

I. Introduction

It is well known that for large networks the exact calculation of network reliability is difficult. Indeed, computing the probability that a graph is connected is a #P-complete problem [4], [16]. Methods like approximation [3], [11] and bounds [1], [2], [9] are available, however, their accuracy and scope are highly dependent on the properties (such as topology and size) of the networks. Hence, for large networks estimating the reliability using simulation techniques becomes attractive. In highly reliable networks such as modern communication networks, the probability of network failure is very low. Direct simulation of such rare events is slow and hence very expensive. Various techniques have been developed to produce better estimates. For example, Kumamoto proposed a very simple technique called *Dagger Sampling* to improve the Crude Monte Carlo simulation [14]. Fishman proposed *Procedure Q* which can provide reliability estimates as well as bounds [10]. Colbourn and Harms proposed a technique that provides progressive bounds that eventually converge to an exact reliability value [5]. Easton and Wong proposed a sequential construction method [6]. Elperin, Gertsbakh and Lomonosov proposed the *Evolution Models* [7], [8]. Hui *et al.* improved the *Evolution Models* by employing the Cross-Entropy technique [13].

For highly reliable networks such as modern communication networks, the Monte Carlo scheme called the *Merge Process* [7], [8] proposed by Elperin *et al.* is one of the best performing algorithms. Instead of sampling in the combinatorial space of functioning edges, the Merge Process models the network's evolution as a Markov Chain and samples from the edge permutation space. The

0-7803-8118-1/03/$17.00 © 2003 IEEE

model avoids the rare event problem in combinatorial sampling and provides low sample variance. However, it has a relatively high computational cost per sample [15]. To combat the high complexity for the Merge Process, Hui *et al.* proposed a hybrid Monte Carlo scheme called the *Tree Cut and Merge* algorithm [12]. It combines combinatorial sampling with permutation sampling together with some bounding techniques. The resultant algorithm can improve simulation performance by over 7 orders of magnitude in some heterogeneous networks. In homogeneous networks, however, the simulation performance may degrade. In some hard cases, the algorithm can take a long time to converge, similar to that of *Crude Monte Carlo* simulations.

In this paper, we first define network reliability in Section II. Then the Merge Process and the Tree Cut and Merge algorithm are briefly reviewed in Section III. The proposed modification to the Tree Cut and Merge algorithm is presented in Section IV followed by the new Importance Sampling scheme in Section V. Experiments and results are presented in Section VI with some discussion.

II. Network Reliability

Consider an undirected graph (or network) $\mathcal{G}(V, E, K)$, where V is the set of n vertices (or nodes), E is the set of m edges, and $K \subseteq V$ is a set of *terminal* nodes, with $|K| \geq 2$. Associated with each edge $e \in E$ is a binary random variable X_e, denoting the *failure state* of the edge. In particular, $\{X_e = 1\}$ is the event that the edge is operational, and $\{X_e = 0\}$ is the event that it is not operational. We label the edges from 1 to m, and call the vector $\boldsymbol{X} = (X_1, \ldots, X_m)$ the (failure) state of the network, or the state of the set E.

Next, we assume that the random variables $\{X_e,\ e \in E\}$ are mutually independent. Let p_e and q_e denote the reliability and unreliability of $e \in E$ respectively. That is

$$p_e = \mathbb{P}[X_e = 1],$$
$$q_e = \mathbb{P}[X_e = 0] = 1 - p_e.$$

The reliability $r(\mathcal{G}; \boldsymbol{p})$ of the network is defined as the probability of K being *connected* by operational edges. Further, let $\boldsymbol{p} = (p_1, \ldots, p_m)$. Thus,

$$r(\mathcal{G}; \boldsymbol{p}) = \mathbb{E}[\varphi(\boldsymbol{X})] = \sum_{x \in \mathcal{S}} \varphi(x)\, \mathbb{P}[\boldsymbol{X} = x], \quad (1)$$

where $\mathcal{S}$ is the set of all 2^m possible states of E and

$$\varphi(x) = \begin{cases} 1 & \text{if } K \text{ is connected,} \\ 0 & \text{otherwise.} \end{cases}$$

This is the standard formulation of the reliability of unreliable systems (networks), see for example [1]. The function φ is called the *structure function* of the unreliable system. Note that the reliability of the network is completely determined by the individual edge reliabilities since we do not consider node failures.

In the rest of this paper, when $\mathcal{G}$ and $\boldsymbol{p}$ are assumed to be understood, we write r instead of $r(\mathcal{G}; \boldsymbol{p})$. For highly reliable networks it is sometimes more useful to analyse, or estimate, the system unreliability

$$\overline{r} = 1 - r\ .$$

Let Q be any estimate of $\overline{r}$ obtained through Monte Carlo simulations, an important measure of the "efficiency" of the simulation is its *relative error*

$$re(Q) = \sqrt{\frac{\mathrm{Var}(Q)}{(\mathbb{E}[Q])^2}}.$$

III. Hybrid Sampling

In this section we briefly overview the concepts used in the Merge Process and the Tree Cut and Merge algorithm. An example is given to illustrate the possible slow convergence in the Tree Cut and Merge algorithm with some explanations.

A. Merge Process

The easiest way to estimate r (or $\overline{r}$) is to use *Crude Monte Carlo* (CMC) simulation. However, for small $\overline{r}$ (which is typical in communication networks) a large sample size is needed to estimate $\overline{r}$ accurately. To combat the problem of poor performance of the Crude Monte Carlo sampling in highly reliable networks, Elperin *et al.* [7] proposed a scheme known as the *Merge Process* (MP) to estimate the reliability of highly reliable networks. The idea is to model the network starting with all

edges failed and each edge e has an exponential repair time with repair rate $\lambda(e) = -\log(q_e)$, then the network state becomes a Markov process starting from isolated nodes and terminates with K connected. For a given *trajectory*, the probability of the network having reached the final state at any time t can be calculated by convolutions. At time $t = 1$ the probability of edge e functioning is $(1 - q_e)$, hence the probability of the network having reached the final state at $t = 1$ is the reliability of the network.

The complexity of a single sample in the Merge Process, including generating a trajectory and computing the convolution, is $O(n^2)$, see [7], as compared to almost $O(n)$ in the Crude Monte Carlo. This complexity may be an acceptable price for the low relative error when $\overline{r} \to 0$.

B. Tree Cut and Merge

The Merge Process simulation alway starts with the isolated node state. Hui *et al.* generalized the process to start at any state y and proposed a hybrid sampling scheme called the Tree Cut and Merge algorithm [12]. The edge set was partitioned into a minimum spanning tree T and its complement $\overline{T} = E \setminus T$. The scheme then samples a random state in T and continues with the Merge Process using only the edge set $\overline{T}$. In its intrinsic form, the Tree Cut and Merge scheme will not produce much improvement in performance over the Standard Merge Process. However, a bounding technique was also introduced in [12] which can produce orders of magnitude improvement in certain classes of network such as heterogeneous networks. Since T connects the whole network (assuming $\mathcal{G}$ is connected), K will always be connected if no edges fail in T. Furthermore, if only one edge is failed in T, it will partition the subgraph $\mathcal{G}(V,T)$ into two components. The subgraph can be "merged" into two super-nodes and $\overline{T}$ can be simplified to some self loops plus a single edge connecting the two super-nodes. Therefore the probability of K being connected given one edge is failed in T can be calculated in $O(n^2)$ time. Combined with the case of no failure in T, this give bounds to the network reliability. The Tree Cut and Merge scheme only needs to sample states in T which have two or more failed edges, this is achieved by sequentially sampling the edges in T, while modifying the edge failure probabilities on the fly to preserve the unbiased nature of the sample.

The Tree Cut and Merge scheme works well with heterogeneous network (particularly if $q_e \ll 0.5$) because it takes advantage of the fact that T has a higher chance of having no failed edges. For modern communication networks, the chance of T having two or more simultaneous failures is fairly small. As a result, the bounds produced by the algorithm are likely to be close to each other and so this reduces the sample variance of the Monte Carlo sampling. Another characteristic of the algorithm is that a smaller number of cuts in T leads to a smaller number of components, and hence the Merge Process will conclude more quickly. Again in most communication networks, the distribution of the number of cuts in T is heavily skewed towards a small number of link failures. Therefore the algorithm can take full advantage of such behavior.

As successful as the Tree Cut and Merge scheme is, there is one weakness built into the algorithm. The first step of the sampling scheme, the "Tree Cut 2+" step, is essentially a Crude Monte Carlo sampling on a reduced population (those states having two or more failures in T). As long as there is a Crude Monte Carlo component in a scheme, there is a risk of poor performance in some extreme distributions. The following is one example of such a case. Consider the ALL-terminal reliability of a dodecahedron network with all links having a failure probability equal to 10^{-6} (see Figure 1). Let $\boldsymbol{X}^T$ be the random state of the edges in T and

$$P_k = \mathbb{P}[|\boldsymbol{X}^T| = k + 1]$$

be the probability of T having k cuts, and let

$$\overline{r_k} = \mathbb{P}[\varphi(\boldsymbol{X}) = 0 \big| |\boldsymbol{X}^T| = k + 1]$$

be the system's failure probability given there are k cuts in T. Table I shows for the different number of cuts (k) in T, its corresponding probabilities P_k, $\overline{r_k}$ and their products. Notice that the ratios $P_1 : P_2$ and $P_2 : P_3$ are about 10^5 but all their contributions to the network failure probability ($P_k \times \overline{r_k}$) are roughly the same. Given that the Tree Cut and Merge algorithm starts sampling from $k = 2$, the events that T has three cuts are important but rare events.

TABLE I
THE FIRST 6 CUT PROBABILITIES.

k	P_k	$\overline{r_k}$	$P_k \times \overline{r_k}$
0	0.999981	0	0
1	1.89997e-05	3.68421e-13	6.99988e-18
2	1.70997e-10	4.67844e-08	8e-18
3	9.68984e-16	0.00516011	5.00007e-18
4	3.87594e-21	0.0216722	8.4e-23
5	1.16278e-26	0.0565021	6.56998e-28

TABLE II
SLOW CONVERGENCE IN AN EXTREME CASE.

Sample size	Estimated $\overline{r}$	relative error
10^6	1.494e-17	2.75e-03
10^7	1.500e-17	8.68e-04
10^8	1.842e-17	1.31e-01
10^9	1.962e-17	4.53e-02
True value	2.000e-17	

Table II lists some typical simulation results of the Tree-Merge (2+) scheme with different numbers of samples taken. Notice that the estimated relative errors are quite low in 10^6 and 10^7 samples but their estimated $\overline{r}$ are in fact not close to the true value. With a million samples, one can only expect a few of them to have two or more cuts. Therefore it takes much more than a million samples to accurately estimate $\overline{r}$ and in this case, something like $10^8 - 10^9$ samples are required.

IV. DIVIDE AND CONQUER

In this section we present a simple method of mitigating the above weakness in the Tree Cut and Merge algorithm. It starts with modifying the Tree Cut 2+ step in the original scheme. Instead of randomly sampling states of T having two or more cuts, it samples states of T having exactly k cuts. By sampling T with $2, \ldots, n-1$ cuts separately, one can improve the sample variance by other means such as the one described in Section V-A. Now we start with the detailed description of the Tree Cut step of the improved scheme.

A. Tree Cutting

If there are k failed links in T, they will partition the subgraph $\mathcal{G}(V,T)$ into exactly $k+1$ components. Let $\boldsymbol{X}$ be the random state of all edges, and $\boldsymbol{X}^T$ be the random state of the edges in T. The network's failure probability $\overline{r}$ can be expressed as:

$$\overline{r} = \sum_{k=0}^{n-1} \mathbb{P}[\varphi(\boldsymbol{X}) = 0, |\boldsymbol{X}^T| = k+1] \quad (2)$$

$$= \sum_{k=0}^{n-1} P_k \overline{r_k} \,. \quad (3)$$

Since $P_0 \ldots P_{n-1}$ can be calculated together in $O(n^2)$ time, the problem of estimating the network failure probability $\overline{r}$ is subdivided into n separate sub-problems of estimating the $\overline{r_k}$. For $k \geq 1$, we can estimate $\overline{r_k}$ as follows:

1) *Tree k Cut:* An outcome of the random state x^T given there are exactly k cuts is generated by sequentially cutting the edges $e \in T$ as follows, see the Appendix for the proof. For the i^{th} edge in T:
If there are l edges failed before the i^{th} edge, modify its failure probability to
$$q_i' = \frac{q_i \, C_{k-l-1,n-1-i}}{C_{k-l,n-i}}$$
where C is a triangular matrix of cut probability with
$$C_{a,b} = \mathbb{P}[a \text{ cuts in the last } b \text{ edges}]$$
$$= \begin{cases} 0 & a > b, \\ \prod_{j>m-a} p_j & a = 0, \\ \prod_{j>m-a} q_j & a = b > 0, \\ q_{m-b+1} C_{a-1,b-1} + p_{m-b+1} C_{a,b-1} & \text{otherwise.} \end{cases} \quad (4)$$
This also gives a corresponding outcome σ of the initial state of the Merge Process. The complexity of this step is $O(n-1)$.
2) *Tree Merge:* Let $\mathbb{L}$ be the lattice of all *proper partitions* of $\mathcal{G}(V,E)$ as described in [12]. For each initial state of the Merge Process σ and the edge set $\overline{T}$, we define the sub-lattice $\mathbb{L}_\sigma^{\overline{T}}$ of $\mathbb{L}$ as the set of all successors of σ that can be obtained by merging *only* the edges in $\overline{T}$. Next we generate a random network state trajectory $\Theta_\sigma = (\sigma_0, \sigma_1, \ldots, \sigma_b)$ in $\mathbb{L}_\sigma^{\overline{T}}$, starting with state $\sigma_0 = \sigma$ until it becomes operational at state σ_b. Given an outcome θ_σ of Θ_σ, the probability of the network coming up after time t is given by the following convolution equation:
$$g(\theta_\sigma; t) = 1 - \underset{0 \leq i < b}{\mathbf{Conv}} \{1 - \exp[-\lambda(\sigma_i)\, t]\} \quad (5)$$

with

$$\lambda(\sigma_i) = \sum_{e \in E(\sigma_i)} -\log(q_e)$$

where $E(\sigma_i)$ denotes the set of inter-component edges in σ_i. Please refer to [12] for the derivation of $g(\theta_\sigma; t)$. At time 1, $g(\theta_\sigma; 1)$ is the network failure probability given trajectory θ_σ.

Let $z = g(\theta_\sigma; 1)$ be the outcome of each simulation run. Then z is the outcome of the random variable $Z = \mathbb{P}[\varphi(X) = 0 | |X^T| = k+1]$.

If we take N independent samples $Z^{(1)}, \ldots, Z^{(N)}$ from Z, then

$$\widehat{\overline{r_k}} = \frac{1}{N} \sum_{i=1}^{N} Z^{(i)}$$

is an unbiased estimator of $\overline{r_k}$.

B. Reliability Bounds

Since $\mathcal{G}$ is connected and T connects $\mathcal{G}$, therefore $\overline{r_0} = 0$ and hence

$$\overline{r} = \sum_{i=1}^{n-1} P_i \overline{r_i}$$

and

$$0 \leq \overline{r} \leq 1 - P_0 \, .$$

There are only $n-1$ edges in T and therefore $n-1$ single cut states; as a consequence P_1 can be calculated in $O(n)$ time. For each such state there are only two components and therefore it takes at most one state jump to bring the network up. $P_1\overline{r_1}$ can easily be evaluated without convolution and the complexity is $O(n^2)$. Combining $\overline{r_1}$ with $\overline{r_0}$, an improved bound,

$$P_1\overline{r_1} \leq \overline{r} \leq P_1\overline{r_1} + (1 - P_0 - P_1) \, ,$$

can be calculated in time $O(n^2)$.

In fact, it is possible to further compress the sample variance by evaluating $\overline{r_1}, \ldots, \overline{r_k}$ and estimating only the $\overline{r_{k+1}}, \ldots, \overline{r_{n-1}}$. Since there are $\mathsf{Bin}(n-1, k)$ ways to make a k-cut to the tree which results in a $k+1$ component network, it takes $O\left(\frac{(n-k)(n-k-1)}{2} + 2^k k^2\right)$ time to perform such a k-cut and exhaust all the associated states. Therefore $\overline{r_k}$ can be computed in $O\left(\mathsf{Bin}(n-1,k)\left(\frac{(n-k)(n-k-1)}{2} + 2^k k^2\right)\right)$ time. Then the reliability bounds are

$$\sum_{i=1}^{k} P_i \overline{r_i} \leq \overline{r} \leq \sum_{i=1}^{k} P_i \overline{r_i} + \sum_{i=k+1}^{n-1} P_i \, .$$

V. Variance Reduction

Since the problem of estimating $\overline{r}$ has been subdivided into estimating $\overline{r_2}, \ldots, \overline{r_{n-1}}$, we have the flexibility of allocating the number of samples to each subproblem. In this section we discuss techniques to harness this flexibility in order to reduce the sample variance of the $\overline{r}$ estimate. Let N_i be the number of samples taken to estimate $\overline{r_i}$, then

$$N = \sum_i N_i$$

is the total number of samples taken to estimate $\overline{r}$. Alternatively, let $\boldsymbol{w}$ be the *allocation vector* where

$$w_i = \frac{N_i}{N}$$

describes the proportion of samples used in estimating $\overline{r_i}$. The vector $\boldsymbol{w}$ can also be interpreted as a discrete probability distribution.

A. Importance Sampling

The sample variance of the network failure probability estimate is given by

$$\begin{aligned} V &= \mathrm{Var}\left(\widehat{\overline{r}}\right) \\ &= \sum_i \frac{P_i^2}{N_i} v_i \\ &= \frac{1}{N} \sum_i \frac{P_i^2 v_i}{w_i} \end{aligned}$$

where v_i is the variance of $\overline{r_i}$ using the Tree Cut and Merge sampling. Then the following optimization program will minimize the sample variance of $\widehat{\overline{r}}$.

$$\min_{\boldsymbol{w}} \sum_i \frac{P_i^2 v_i}{w_i} \qquad (6)$$

$$\text{subject to } w_i \geq 0, \sum w_i = 1 \, .$$

The solution of the Program (6) is called the *Importance Sampling density* and is given by

$$w_i^* = \frac{P_i^2 v_i}{\sum_i P_i^2 v_i} . \quad (7)$$

Unfortunately, implementation of the Importance Sampling density $\boldsymbol{w}^*$ as per Equation (7) is problematic. The main difficulty lies in the prerequisite of the unknown parameters v_i. The estimated sample variance is an important indicator of the accuracy of an estimate, and so it is useful to estimate the variances v_i in order to calculate the estimated sample variance $\widehat{V}$. Fortunately, the simulation procedures can easily be modified to estimate v_i as well as $\overline{r_i}$. An adaptive pilot simulation can be used to find the initial estimate of the parameters v_i, then the estimated Importance Sampling density can be constructed by

$$\widehat{w_i^*} = \frac{P_i^2 \widehat{v_i}}{\sum_{i=2}^{n-1} P_i^2 \widehat{v_i}} \quad (8)$$

where $\widehat{v_i}$ is the estimate of v_i. The density $\widehat{\boldsymbol{w}^*}$ is *dynamically* updated during the main simulations. The scheme can be implemented as follows:

Importance Sampling Scheme:

1) *Initialization*
   Run the adaptive pilot simulation (see below) to obtain the initial estimates of $\overline{r_i}$ and v_i. Then initialize the Importance Sampling density using Equation (8).
2) *Sampling*
   According to the probability density $\widehat{\boldsymbol{w}^*}$, take an additional batch of
   $$N_i^+ = \lceil N^+ \widehat{w_i^*} \rceil$$
   samples to refine the estimates $\widehat{\overline{r_i}}$ and $\widehat{v_i}$, where N^+ is the batch constant for the minimum number of samples in each run.
3) *Update*
   Update $\widehat{\boldsymbol{w}^*}$ with the new estimates $\widehat{v_i}$. Then repeat step 2 until the total number of samples N reaches the predefined value or the estimated sample variance
   $$\widehat{V} = \sum_{i=2}^{n-1} \frac{P_i^2}{N_i} \widehat{v_i} \quad (9)$$
   drops below the predefined level.

As the total number of samples $N \to \infty$, $\widehat{\boldsymbol{w}^*} \to \boldsymbol{w}^*$ and $\widehat{\overline{r}} \to \overline{r}$. However, the rate of convergence is sensitive to the initial value of $\widehat{\boldsymbol{w}^*}$. In the extreme case where the initial estimate $P_i^2 \widehat{v_i}$ is near zero but the true value of $P_i^2 v_i$ is not, it may result in an inaccurate estimate of $\overline{r}$. To avoid such a problem, one can use the estimated relative error ($\widehat{re_i}$) as a guide to make sure the initial $\widehat{v_i}$ are reasonably close to v_i. The following adaptive pilot simulation is proposed to constrain the $\widehat{re_i}$ to small values:

Pilot Simulation: For each i where $\overline{r_i}$ is to be estimated:

1) Take an additional batch of n (say 100) simulation to estimate $\overline{r_i}$ and v_i.
2) Calculate the estimated relative error of $\widehat{\overline{r_i}}$,
   $$\widehat{re_i} = \sqrt{\frac{\widehat{v_i}}{N_i \widehat{\overline{r_i}}^2}} .$$
3) Repeat from step 1 if $\widehat{re_i} > \mu$, where μ is a constant. A value between 0.1 and 0.3 is usually sufficient for μ.

VI. Numerical Experiments

In this section, different variants of the *Tree Cut and Merge* algorithm are compared with the Standard Merge Process. The *Relative Time Variance* product (RTV) is used as a metric to compare different algorithms. It is defined as the simulation time (in seconds) multiplied by the (estimated) squared relative error. Since for a large number of iterations N, the simulation time is proportional to N and the relative error is inversely proportional to $\sqrt{N}$, the RTV is a number that depends on the network and the performance of the algorithm being studied rather than on N. The smaller the RTV value, the more efficient is the simulation algorithm.

An exact algorithm using exhaustive search to calculate $\overline{r_i}$ in Equation (3) is also implemented to confirm that the simulations produce accurate estimates. A dodecahedron network (Figure 1) with different link reliabilities is used to test the different schemes. In each experiment, all the Monte Carlo algorithms were run for 10^6 iterations and the parameter μ in the pilot runs was set to 0.1. Table III lists the meaning of the labels used in the experimental results.

Fig. 1. Dodecahedron network & its Minimum Spanning Tree

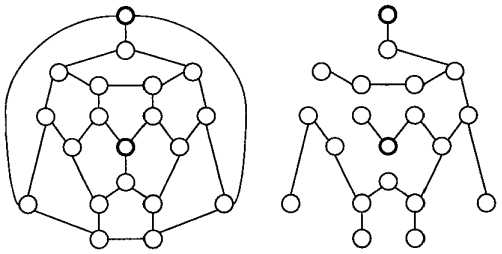

TABLE III

DESCRIPTIONS OF DIFFERENT LABELS USED.

Label	Explanation
MP	Standard Merge Process algorithm
TM (2+)	Tree Cut & Merge algorithm with two or more cuts in each sample
TM (IS)	Tree Cut & Merge algorithm with Importance Sampling
t	Simulation time for 10^6 samples in sec
Q	Estimated network failure probability
$\bar{r}$	The true network failure probability
$\widehat{re}$	Estimated relative error of Q
RTV	Relative Time Variance product
bounds	Bounds on the network unreliability calculated by the algorithm

Experiment 1: ALL-terminal reliability of a heterogeneous unreliable network: In the first experiment, there are two groups of links: the first group is the minimum spanning tree (resembling a backbone network), and the second group consists of the remaining links with slightly lower reliability (resembling wireless backup links). In particular, the backbone links have failure probabilities of 0.1%, and the backup links have failure probabilities of 1%. The ALL-terminal network failure probability is to be estimated and the results are listed in Table IV. The best performing algorithm in this experiment is the Tree-Merge IS scheme. It shows that the Tree-Merge IS scheme is able to further reduce the sample variance compared to the Tree-Merge (2+) scheme.

Experiment 2: TWO-terminal reliability of a heterogeneous unreliable network: The second experiment uses the same network as in the first experiment. This time we are estimating the TWO-terminal network failure probability (the two terminal nodes are marked by thick circles in Figure 1). The results are listed in Table V and both Tree-Merge schemes perform substantially better than that of the Standard Merge scheme. The Tree-Merge (2+) scheme performs slightly better but the differences are not significant.

TABLE IV

ALL-TERMINAL RELIABILITY OF A HETEROGENEOUS UNRELIABLE NETWORK.

Scheme	MP	TM (2+)	TM (IS)
t	815	80	85
Q	7.908e-7	7.915e-7	7.893e-7
$\bar{r}$	7.902e-7	7.902e-7	7.902e-7
$\widehat{re}$	1.49e-3	1.32e-3	6.23e-4
RTV	1.80e-3	1.40e-4	3.31e-5
bounds	n/a	[6.915e-7,1.698e-4]	

TABLE V

TWO-TERMINAL RELIABILITY OF A HETEROGENEOUS UNRELIABLE NETWORK.

Scheme	MP	TM (2+)	TM (IS)
t	755	45	54
Q	1.123e-7	1.124e-7	1.120e-7
$\bar{r}$	1.123e-7	1.123e-7	1.123e-7
$\widehat{re}$	4.72e-3	1.42e-3	1.48e-3
RTV	1.69e-2	9.04e-5	1.19e-4
bounds	n/a	[1.002e-7,1.692e-4]	

Experiment 3: ALL-terminal reliability of a heterogeneous reliable network: The third experiment is the same as the first experiment except the backbone network is much more reliable, the link failure probability is 10^{-6} which is more realistic in shielded cable networks. The results are listed in Table VI and it has the same pattern as in Experiment 2, that is both Tree-Merge schemes have very similar performance.

Experiment 4: TWO-terminal reliability of a homogeneous reliable network: In the fourth experiment, the network is homogeneous and reliable. Each link has the same failure probability of 10^{-6}, and the TWO-terminal network failure probability is to be estimated. The results are listed in Table VII. In

TABLE VI

ALL-TERMINAL RELIABILITY OF A HETEROGENEOUS RELIABLE NETWORK.

Scheme	MP	TM (2+)	TM (IS)
t	814	80	82
Q	7.041e-10	7.041e-10	7.041e-10
$\bar{r}$	7.041e-10	7.041e-10	7.041e-10
$\widehat{re}$	1.38e-3	6.08e-7	6.16e-7
RTV	1.55e-3	2.95e-11	3.10e-11
bounds	n/a	[7.040e-10,8.750e-10]	

TABLE VII
TWO-TERMINAL RELIABILITY OF A HOMOGENEOUS RELIABLE NETWORK.

Scheme	MP	TM (2+)	TM (IS)
t	752	43	66
Q	1.984e-18	2.982e-24	1.997e-18
$\bar{r}$	2.000e-18	2.000e-18	2.000e-18
$\widehat{re}$	4.36e-3	1.01e-2	2.25e-2
RTV	1.43e-2	4.43e-3	3.32e-2
bounds	n/a	[1.000e-30,1.710e-10]	

TABLE VIII
IMPROVED PERFORMANCE BY EMPLOYING MORE EXHAUSTIVE SEARCH.

Scheme	TM (IS) 4+
t	89
Q	2.000e-18
$\bar{r}$	2.000e-18
$\widehat{re}$	1.83e-7
RTV	2.99e-12
bounds	[2.000e-18,2.536e-16]

this experiment, the best performing scheme is the Standard Merge. Notice that despite the fact that the RTV value of the Tree-Merge (2+) scheme is the lowest, its estimate has not converged to the true value yet. It shows that in this network, the Crude Monte-Carlo component in the Tree-Merge (2+) scheme is taking its toll. It would take much more than 10^6 samples for this scheme to produce an accurate estimate for this network. On the other hand, the variance reduction schemes performed reasonably well.

The Tree-Merge IS scheme quickly calculates the reliability bounds and fully utilises their associated knowledge. Furthermore, it allows us to easily choose where to stop exhaustive search, hence it is possible to further compress the sample variance by spending more time on the search. For instance, if we choose to use exhaustive search for up to $k = 3$, it produces exceptional performance as listed in the Table VIII. However, such a tactic may not work in large networks as the time needed for the exhaustive search grows quickly with k.

Conclusions of the experiments: In heterogeneous networks where the original Tree Cut and Merge algorithm is good, the variance reduction schemes introduced in this paper have similar performance to the original scheme. In some cases the overhead slightly slows down the simulation without reducing the variance significantly while in another case the variance reduction overcame the speed overhead. In homogeneous networks where the Tree-Merge (2+) scheme does not perform as well as the standard Merge Process, the variance reduction schemes improve the performance to a comparable level (further experiments are not shown in this paper). If one is willing to spend more time on exhaustive search in homogeneous networks, the variance reduction schemes can significantly outperform the standard Merge Process, as in Experiment 4. The most important contribution of the variance reduction schemes is that they vastly improve the robustness of the Tree Cut and Merge algorithm. In some harsh cases, such as the homogeneous reliable networks in Example 4, the Tree-Merge (2+) scheme may take billions of samples to converge whereas the variance reduction schemes bring it down to around a hundred thousand.

VII. CONCLUSIONS

The Tree Cut and Merge algorithm is a hybrid sampling technique developed to speed up simulations by taking advantage of the topological knowledge in heterogeneous networks. The original Tree-Merge (2+) scheme works as designed and produces reliability bounds in a very short time. However, the scheme can have problems in some difficult cases where it can take billions of samples to converge. The reason for the slow convergence is due to the Crude Monte Carlo component in the Tree-Cut step of the scheme. In an effort to reduce the sample variance, this paper introduced the Importance Sampling technique to mitigate this weakness in the Tree-Merge (2+) scheme. The resultant Tree-Merge IS scheme successfully improves the Tree-Merge (2+) scheme in hard cases without impeding the performance in other cases. The key benefits of the new scheme are:

- It improves the robustness of the Tree Cut and Merge algorithms and provides comparable performance to the standard Merge Process in homogeneous networks.
- It keeps the speed benefit of the Tree Cut and Merge algorithms in heterogeneous networks.
- By dividing $\bar{r}$ into $\overline{r_0}, \ldots, \overline{r_{n-1}}$, tighter bounds are readily available to the new scheme if higher levels of exhaustive search are used.

The Tree-Merge IS scheme requires a pilot simulation to obtain the initial allocation vector, the information learned in this stage may be further utilized. One possible application is to determine whether the exhaustive search should extend to higher levels. For example, if the estimated number of samples needed to estimate $\overline{r_k}$ is comparable to the state count for exhaustive search, one might want to use exhaustive search instead.

REFERENCES

[1] R. Barlow and F. Proschan. *Statistical Theory of Reliability and Life Testing*. Holt, Rinehart & Wilson, 1975.

[2] R.E. Barlow and A. Marshall. Bounds for distributions with monotone hazard rate, I and II. *Ann. Maths. Statist.*, 35:1234–1274, 1964.

[3] Yu Burtin and B. Pittle. Asymptotic estimates of the reliability of a complex system. *Engrg. Cybern.*, 10(3):445 – 451, 1972.

[4] C. Colbourn. *The combinatorics of Network Reliability*. Oxford University Press, 1987.

[5] C. Colbourn and D. Harms. Evaluating performability: Most probable states and bounds. *Telecommunication Systems*, 2:275–300, 1994.

[6] M.C. Easton and C.K. Wong. Sequential destruction method for Monte Carlo evaluation of system reliability. *IEEE Transactions on Reliability*, R-29:27–32, 1980.

[7] T. Elperin, I Gertsbakh, and M. Lomonosov. Estimation of network reliability using graph evolution models. *IEEE Transactions on Reliability*, 40(5):572–581, Dec 1991.

[8] T. Elperin, I Gertsbakh, and M. Lomonosov. An evolution model for Monte Carlo estimation of equilibrium network renewal parameters. *Probability in the Engineering and Informational Sciences*, 6:457–469, 1992.

[9] J.D Esary, F. Proschan, and D.W. Walkup. Association of random variables, with applications. *Ann. math. Statist.*, 38:1466–1474, 1967.

[10] G. Fishman. A Monte Carlo sampling plan for estimating network reliability. *Operations Research*, 34(4):581–594, Jul-Aug 1986.

[11] I Gertsbakh. *Reliability theory with Application to Preventive maintenance*. Springer, 2000.

[12] K-P. Hui, N. Bean, M. Kraetzl, and D. Kroese. The tree cut and merge algorithm for estimation of network reliability. *Probability in the Engineering and Informational Sciences*, 17(1):24–45, 2003.

[13] K-P. Hui, N. Bean, M. Kraetzl, and D. Kroese. The cross-entropy method for network reliability estimation. *To appear in the Annals of Operations Research*, 2004.

[14] H. Kumamoto, K. Tanaka, K. Inoue, and E. J. Henley. Dagger sampling Monte Carlo for system unavailability evaluation. *IEEE Transactions on Reliability*, R-29(2):376–380, 1980.

[15] M. Lomonosov. On Monte Carlo estimates in network reliability. *Probability in the Engineering and Informational Sciences*, 8:245–264, 1994.

[16] J. Provan and M. Ball. The comoplexity of counting cuts and of computing the probability that a graph is connected. *SIAM Journal of computing*, 12:777–787, 1982.

APPENDIX

Consider a graph with edge set $E = \{e_1, \ldots, e_m\}$ where edge e_i has a failure probability of q_i (or a functioning probability of $p_i = 1 - q_i$). Let Y_i be the binary random variable associated with edge e_i. In particular, $\{Y_i = 1\}$ is the event that the edge is failed, and $\{Y_i = 0\}$ is the event that the edge is operational.

We want to unbiasedly sample the network state given there are exactly k failed links, that is, $\sum Y_i = k$. This can be achieved through the sequential sampling technique, sampling edge states one by one and modifying the failure probability according to the states of previous edges.

Consider the i^{th} edge in E, if there are l edges failed before the i^{th} edge, the probability of e_i having failed given that exactly $k \geq l$ edges have failed in E is

$$
\begin{aligned}
q_i' &= \mathbb{P}\left[Y_i = 1 \,\middle|\, \sum_j Y_j = k, \sum_{j<i} Y_j = l \right] \\
&= \mathbb{P}\left[Y_i = 1 \,\middle|\, \sum_{j\geq i} Y_j = k-l, \sum_{j<i} Y_j = l \right] \\
&= \frac{\mathbb{P}\left[Y_i = 1, \sum_{j\geq i} Y_j = k-l, \sum_{j<i} Y_j = l \right]}{\mathbb{P}\left[\sum_{j\geq i} Y_j = k-l, \sum_{j<i} Y_j = l \right]} \\
&= \frac{\mathbb{P}\left[Y_i = 1, \sum_{j\geq i+1} Y_j = k-l-1, \sum_{j<i} Y_j = l \right]}{\mathbb{P}\left[\sum_{j\geq i} Y_j = k-l, \sum_{j<i} Y_j = l \right]}.
\end{aligned}
$$

Since Y_i, Y_j are independent if $i \neq j$, therefore

$$
\begin{aligned}
q_i' &= \frac{\mathbb{P}[Y_i = 1]\mathbb{P}\left[\sum_{j\geq i+1} Y_j = k-l-1 \right]}{\mathbb{P}\left[\sum_{j\geq i} Y_j = k-l \right]} \\
&= \frac{q_i \, C_{k-l-1,m-i}}{C_{k-l,m-i+1}},
\end{aligned}
$$

where C is the cut probabilities matrix defined in Equation (4).