

The Cross-Entropy Method for Network Reliability Estimation

K-P. Hui[†], N. Bean[‡], M. Kraetzl^{*}, D.P. Kroese[◊]

[†]IN Division, DSTO, Edinburgh 5111, Australia

[‡]Department of Applied Mathematics, University of Adelaide, Adelaide 5005, Australia

^{*}ISR Division, DSTO, Edinburgh 5111, Australia

[◊]Department of Mathematics, University of Queensland, Brisbane 4072, Australia

Abstract

Consider a network of unreliable links, modelling for example a communication network. Estimating the reliability of the network – expressed as the probability that certain nodes in the network are connected – is a difficult task. In this paper we study how the Cross-Entropy method can be used to obtain more efficient estimation procedures. Two standard techniques for network reliability estimation are considered: Crude Monte Carlo and the more sophisticated Permutation Monte Carlo. We show that the Cross-Entropy method yields a speed-up over both techniques.

1 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 [3, 24]. Hence, for large networks estimating the reliability using simulation techniques becomes necessary. 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 [19]. Fishman proposed *Procedure Q* which can provide reliability estimates as well as bounds [15]. Colbourn and Harms proposed a technique that will provide progressive bounds that will eventually converge to an exact reliability value [4]. Easton and Wong proposed a sequential construction method [12]. Elperin *et al.* proposed *Evolution Models* for estimating reliability of highly reliable networks [13, 14]. Hui *et al.* proposed an hybrid scheme that can provide bounds and speed up by orders of magnitude in certain class of networks [17].

The Cross-Entropy (CE) method originated from Rubinstein [25] as an adaptive algorithm for estimating probabilities of *rare events* in complex stochastic networks. The method can also be used for solving optimisation problems [26, 27]. The Cross-Entropy method has been successfully applied to a wide range of com-

binatorial and continuous optimisation problems [11, 20, 22, 26], including problems in reliability theory [21], buffer allocation [1], telecommunication systems [5, 6, 7, 8], neural computation [11], control and navigation [16, 23], DNA sequence alignment [18], scheduling [10, 22] and Max-Cut and bipartition problems [28]. A short review of the basic ideas behind the Cross-Entropy method is given at the end of this section, but for details we refer to the forthcoming book on Cross-Entropy [29], and the tutorial in [9], which is also available on-line at <http://www.cs.utwente.nl/~ptdeboer/ce/>.

In this paper we investigate the benefit of the Cross-Entropy method to the estimation of network reliability or, equivalently, network *unreliability*. Basically, the Cross-Entropy method provides an iterative procedure to adaptively estimate the optimal Importance Sampling parameters for the quantity of interest, in our case the unreliability. We show that the Cross-Entropy technique indeed can lead to a significant speedup.

The rest of the paper is organised as follows. At the end of this introduction we give a brief review of the most important aspects of the Cross-Entropy method for Monte Carlo simulation. In Section 2 we discuss unreliable networks and give two methods, Crude Monte Carlo (CMC) and Permutation Monte Carlo (PMC) to estimate network (un)reliability. Section 3 deals with the Cross-Entropy modification of the Crude Monte Carlo and Permutation Monte Carlo. In this section we use a simple bridge system as the example. In Section 4 we give numerical results for larger examples, exemplifying what we found in Section 3 for the simple bridge system.

1.1 A Very Short Introduction to CE

It is not our intention to give here a detailed account of the Cross-Entropy method – for this we refer to [9] – but in order to keep this paper fairly self-contained, we mention the main points. Consider the problem of estimating

$$\ell = \mathbb{E}_{\mathbf{u}}[H(\mathbf{Y})] = \int H(\mathbf{y}) f(\mathbf{y}; \mathbf{u}) d\mathbf{y}, \quad (1)$$

where $H(\mathbf{y})$ is some performance function of \mathbf{y} and $\mathbf{Y} = (Y_1, \dots, Y_n)$ is a random vector with (joint) probability density function (pdf) $f(\cdot; \mathbf{u})$ which depends on a *reference parameter* \mathbf{u} . We can estimate ℓ using *Importance Sampling* (IS) as

$$\hat{\ell} = \frac{1}{N_1} \sum_{i=1}^{N_1} H(\mathbf{Y}_i) W(\mathbf{Y}_i; \mathbf{u}, \mathbf{w}), \quad (2)$$

where $\mathbf{Y}_1, \dots, \mathbf{Y}_{N_1}$ is a random sample from $f(\cdot; \mathbf{w})$ – using a *different* reference parameter – and

$$W(\mathbf{Y}; \mathbf{u}, \mathbf{w}) = \frac{f(\mathbf{Y}; \mathbf{u})}{f(\mathbf{Y}; \mathbf{w})}, \quad (3)$$

is the *likelihood ratio*. We can choose *any* reference vector \mathbf{w} in (2), but the one that is optimal in the Cross-Entropy sense is

$$\mathbf{v}^* = \underset{\mathbf{v}}{\operatorname{argmax}} \mathbb{E}_{\mathbf{w}} [H(\mathbf{Y})] W(\mathbf{Y}; \mathbf{u}, \mathbf{w}) \log f(\mathbf{Y}; \mathbf{v}). \quad (4)$$

We can estimate the optimal Cross-Entropy reference vector, as the solution of the iterative procedure

$$\mathbf{v}_t = \underset{\mathbf{v}}{\operatorname{argmax}} \frac{1}{N} \sum_{i=1}^N H(\mathbf{Y}_i) W(\mathbf{Y}_i; \mathbf{u}, \mathbf{v}_{t-1}) \log f(\mathbf{Y}_i; \mathbf{v}), \quad (5)$$

where at each iteration t a random sample from $f(\cdot; \mathbf{v}_{t-1})$ is taken. The solution of (5) can often be determined *analytically*.

In the *rare-event* setting $H(\mathbf{Y})$ is of the form $H(\mathbf{Y}) = I\{S(\mathbf{Y}) \geq \gamma\}$ and

$$\ell = \mathbb{P}[S(\mathbf{Y}) \geq \gamma] \quad (6)$$

is a small probability. The function S is called the *performance function*. For rare-event estimation problems (5) is difficult to carry out: because of the rareness of the event most of the indicators $H(\mathbf{Y}_i)$ will be zero. For these problems a two-phase Cross-Entropy procedure is employed in which apart from \mathbf{v} also the *level* parameter γ is updated, creating a sequence of 2-tuples $\{(v_t, \gamma_t)\}$ with the goal of estimating the optimal Cross-Entropy reference parameter \mathbf{v}^* . Starting with $\mathbf{v}_0 = \mathbf{u}$ (the original or nominal parameter), the updating formulas are as follows:

Given a random sample $\mathbf{Y}_1, \dots, \mathbf{Y}_N$ from $f(\cdot; \mathbf{v}_{t-1})$, let γ_t be the sample $(1 - \rho)$ -quantile of the performances $S(\mathbf{Y}_i), i = 1, \dots, N$, provided the sample quantile is less than γ ; otherwise we set γ_t to γ . In other words,

$$\gamma_t = \min\{\gamma, S_{(\lceil(1-\rho)N\rceil)}\}, \quad (7)$$

where $S_{(j)}$ is the j -th *order-statistic* of the performances. Using the *same sample*, we let

$$\mathbf{v}_t = \operatorname{argmax}_{\mathbf{v}} \frac{1}{N} \sum_{i=1}^N I\{S(\mathbf{Y}_i) \geq \gamma_t\} W(\mathbf{Y}_i; \mathbf{u}, \mathbf{v}_{t-1}) \log f(\mathbf{Y}_i; \mathbf{v}). \quad (8)$$

Again, it is important to understand that in many cases an explicit formula for \mathbf{v}_t can be given; that is, we do not need to “solve” optimisation problem (8).

2 Estimation of Network Reliability

2.1 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 e is operational, and $\{X_e = 0\}$ is the event that it has failed. We label the edges from 1 to m , and call the vector $\mathbf{X} = (X_1, \dots, X_m)$ the state of the network. Let \mathcal{S} be the set of all 2^m possible states of 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

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

The reliability r of the network is defined as the probability of K being *connected* by operational edges. Thus,

$$r = \mathbb{E}[\varphi(\mathbf{X})] = \sum_{\mathbf{x} \in \mathcal{S}} \varphi(\mathbf{x}) \mathbb{P}[\mathbf{X} = \mathbf{x}], \quad (9)$$

where

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

This is the standard formulation of the reliability of unreliable systems, see for example [2]. 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.

For highly reliable networks it is sometimes more useful to analyse or estimate the system unreliability

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

Let Q is an unbiased estimate of \bar{r} obtained through Monte Carlo simulations, an important measure of the “efficiency” of the simulation is its *relative error*

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

Example 1 (Bridge Network)

Consider the simple network in Figure 1, called a *bridge network*. The bridge network will serve as a convenient reference example to which we will return throughout the paper. Here we have 5 unreliable edges, labelled $1, \dots, 5$. The network works if the two terminal nodes A and B are connected by operational edges. It is not difficult to see that the structure function φ is given by

$$\varphi(\mathbf{x}) = 1 - (1 - x_1 x_3 x_5)(1 - x_2 x_3 x_4)(1 - x_1 x_4)(1 - x_2 x_5) .$$

Applying the inclusion-exclusion principle to the mincuts, the system unreliability is found to be equal to

$$\begin{aligned} \bar{r} = & q_1 q_2 + q_2 q_3 q_4 - q_1 q_2 q_3 q_4 + q_1 q_3 q_5 - q_1 q_2 q_3 q_5 \\ & + q_4 q_5 - q_1 q_2 q_4 q_5 - q_1 q_3 q_4 q_5 - q_2 q_3 q_4 q_5 + 2q_1 q_2 q_3 q_4 q_5 . \end{aligned} \tag{10}$$

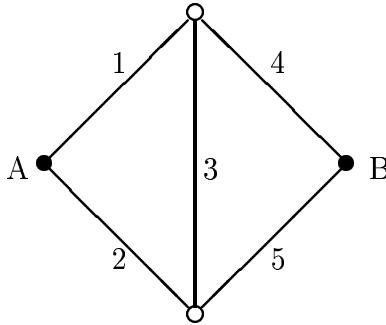


Figure 1: Two-terminal bridge network

2.2 Crude Monte Carlo Simulation

Let us assume the typical situation that the edges are highly reliable, that is, the q_e are close to 0. In that case the appropriate quantity to estimate is the system unreliability \bar{r} which will be close to 0, rather than r which will be close to 1. The easiest way to estimate \bar{r} is to use Crude Monte Carlo (CMC) simulation. Let $\mathbf{X}_{(1)}, \dots, \mathbf{X}_{(N)}$ be independent identically-distributed random vectors with the same distribution as \mathbf{X} . Then

$$Q = \frac{1}{N} \sum_{i=1}^N (1 - \varphi(\mathbf{X}_{(i)})) \quad (11)$$

is an unbiased estimator for \bar{r} and its *relative error* is

$$\text{re}(Q) = \sqrt{\frac{\text{Var}(Q)}{(\mathbb{E}[Q])^2}} = \sqrt{\frac{\bar{r}(1 - \bar{r})/N}{\bar{r}^2}} = \sqrt{\frac{1 - \bar{r}}{N \bar{r}}}.$$

This shows that for small \bar{r} , a large sample size is needed to estimate \bar{r} accurately.

When \bar{r} is small, the event that the terminal nodes are not connected is a rare event.

2.3 Permutation Monte Carlo Simulation

A more accurate way of estimating the network unreliability is *Permutation Monte Carlo* [13]. The idea is as follows. Consider the network $\mathcal{G}(V, E)$ in which each edge e has an exponential repair time with repair rate $\lambda(e) = -\log(q_e)$. At time $t = 0$ all edges are failed. We assume that all repair times are independent of each other. The state of e at time t is denoted by $X_e(t)$ and the state of the edge set E at time t is given by the vector $\mathbf{X}(t)$, defined in a similar way as before. Thus $(\mathbf{X}(t))$ is a Markov process with state space $\{0, 1\}^m$. This process is called the *Construction Process* (CP) of the network.

Let Π denote the *order* in which the edges are constructed (become operational), and let $A_0, A_0 + A_1, \dots, A_0 + \dots + A_{m-1}$ be the times at which those edges are constructed. Hence the A_i are *sojourn times* of $(\mathbf{X}(t))$. Π is a random variable which takes values in the space of permutations of E .

For any permutation $\pi = (e_1, e_2, \dots, e_m)$ define

$$\begin{aligned} E_0 &= E, \\ E_i &= E_{i-1} \setminus \{e_i\}, \quad 1 \leq i \leq m-1, \\ \lambda(E_i) &= \sum_{e \in E_i} \lambda(e), \end{aligned}$$

and let

$$b(\pi) = \min_i \{\varphi(E_i) = 1\}$$

be the so-called *critical number* of π . From the general theory of Markov processes

it is not difficult to see that

$$\mathbb{P}[\Pi = \pi] = \prod_{j=1}^m \frac{\lambda(e_j)}{\lambda(E_{j-1})}. \quad (12)$$

Moreover, conditional on $\{\Pi = \pi\}$, the sojourn times A_0, \dots, A_{m-1} are independent and each A_i is exponentially distributed with parameter $\lambda(E_i)$, $i = 0, \dots, m-1$.

Note that the probability of each edge e being operational at time $t = 1$ is p_e . It follows that the *network* reliability at time $t = 1$ is the same as in (9). Hence, by conditioning on Π we have

$$r = \mathbb{E}[\varphi(\mathbf{X}(1))] = \sum_{\pi} \mathbb{P}[\Pi = \pi] \mathbb{P}[\varphi(\mathbf{X}(1)) = 1 \mid \Pi = \pi], \quad (13)$$

and

$$\bar{r} = 1 - r = \sum_{\pi} \mathbb{P}[\Pi = \pi] \mathbb{P}[\varphi(\mathbf{X}(1)) = 0 \mid \Pi = \pi]. \quad (14)$$

Using the definitions of A_i and $b(\pi)$, we can write the last probability in terms of convolutions of exponential distribution functions. Namely, for any $t \geq 0$ we have

$$\begin{aligned} \mathbb{P}[\varphi(\mathbf{X}(t)) = 0 \mid \Pi = \pi] &= \mathbb{P}[A_0 + \dots + A_{b(\pi)-1} > t \mid \Pi = \pi] \\ &= 1 - \mathbf{Conv}_{0 \leq i < b(\pi)} \{1 - \exp[-\lambda(E_i) t]\}. \end{aligned} \quad (15)$$

Let

$$G(\pi) = \mathbb{P}[\varphi(\mathbf{X}(1)) = 0 \mid \Pi = \pi], \quad (16)$$

as given in (15). Equation (14) can be rewritten as

$$\bar{r} = \mathbb{E}[G(\Pi)], \quad (17)$$

and this shows how the Permutation Monte Carlo simulation scheme works. Namely, let $\Pi_{(1)}, \dots, \Pi_{(N)}$ be independent identically distributed random permutations, each distributed according to Π . Then

$$\hat{r} = \frac{1}{N} \sum_{i=1}^N G(\Pi_{(i)}) \quad (18)$$

is an unbiased estimator for \bar{r} .

3 Estimating Network Reliability using the Cross-Entropy Method

3.1 Cross-Entropy and Crude Monte Carlo

If we use the Construction Process idea of Section 2.3 in the Crude Monte Carlo framework of Section 2.2, the Cross-Entropy method fits naturally. That is, instead of sampling the up/down state of individual edges, we sample the up time of each edge. Then we see if the network is functioning at time $t = 1$ and this probability is the network reliability.

In other words, translate the original problem (estimating \bar{r}), which involves independent Bernoulli random variables X_1, \dots, X_m , into an estimation problem involving independent exponential random variables Y_1, \dots, Y_m . Specifically, imagine that we have a time-dependent system in which at time 0 all edges have failed and are under repair, and let Y_1, \dots, Y_m , with $Y_i \sim \text{Exp}(u_i^{-1})$ and $u_i = 1/\lambda(i) = -1/\log q_i$ be the independent *repair times* of the edges. Note that,

by definition

$$\mathbb{P}[Y_i \geq 1] = e^{-1/u_i} = q_i \quad i = 1, \dots, m.$$

Now, for each $\mathbf{Y} = (Y_1, \dots, Y_m)$ let $S(\mathbf{Y})$ be the (random) time at which the system “comes up” (the terminal nodes become connected). Then, we can write

$$\bar{\tau} = \mathbb{P}[S(\mathbf{Y}) \geq 1].$$

Hence, we have written the estimation of $\bar{\tau}$ in the standard rare event formulation of (6) and thus can directly apply the Cross-Entropy method from [9].

Instead of sampling independently for each i from $\text{Exp}(u_i^{-1})$ we sample from $\text{Exp}(v_i^{-1})$. The vector $\mathbf{v} = (v_1, \dots, v_m)$ is thus our reference parameter. We now construct a sequence of 2-tuples $\{\mathbf{v}_t, \gamma_t\}$ such that \mathbf{v}_t converges to a reference vector close to the optimal Cross-Entropy reference parameter and γ_t eventually reaches 1. Starting with $\mathbf{v}_0 = \mathbf{u} = (u_1, \dots, u_m)$, we draw at each iteration t a random sample from $\mathbf{Y}_{(1)}, \dots, \mathbf{Y}_{(N)}$ from the pdf $f(\cdot; \mathbf{v}_{t-1})$ of \mathbf{Y} and update the level parameter using (7) and the reference parameter using (8), which in this case has the analytical solution

$$v_{t,j} = \frac{\sum_{i=1}^N I\{S(\mathbf{Y}_{(i)}) \geq \gamma_t\} W(\mathbf{Y}_{(i)}; \mathbf{u}, \mathbf{v}_{t-1}) Y_{(i)j}}{\sum_{i=1}^N I\{S(\mathbf{Y}_{(i)}) \geq \gamma_t\} W(\mathbf{Y}_{(i)}; \mathbf{u}, \mathbf{v}_{t-1})}, \quad (19)$$

where W is the likelihood ratio

$$W(\mathbf{y}; \mathbf{u}, \mathbf{v}) = \frac{f(\mathbf{y}; \mathbf{u})}{f(\mathbf{y}; \mathbf{v})} = \exp\left(-\sum_{j=1}^m y_j \left(\frac{1}{u_j} - \frac{1}{v_j}\right)\right) \prod_{j=1}^m \frac{v_j}{u_j}. \quad (20)$$

For instance after iteration T when γ_T reaches 1, we estimate $\bar{\tau}$ using Impor-

tance Sampling as

$$\widehat{r} = \frac{1}{N_1} \sum_{i=1}^{N_1} I\{S(\mathbf{Y}_{(i)}) \geq 1\} W(\mathbf{Y}_{(i)}; \mathbf{u}, \mathbf{v}_T) .$$

Example 2 (Bridge Network, CE and CMC)

Consider the bridge network of Example 1. Since we have a 2-terminal network, the function S can be expressed in terms of the maximal paths of the networks. In particular, we have

$$S(\mathbf{Y}) = \min_i \max_{e \in \mathcal{P}_i} X_e .$$

where $\mathcal{P}_1 = \{1, 4\}$, $\mathcal{P}_2 = \{2, 5\}$, $\mathcal{P}_3 = \{2, 3, 4\}$ and $\mathcal{P}_4 = \{1, 3, 5\}$ are the maximal path sets of the system, see [2, 21]. Suppose the “nominal” parameter vector is $\mathbf{u} = (0.3, 0.1, 0.8, 0.1, 0.2)$. From (10) the exact unreliability is $\bar{r} = 7.07868\text{e-}05$. A typical result of the simulations is given in Table 1. The Cross-Entropy parameters used were: (initial) sample size $N = 2000$ and rarity parameter $\rho = 0.01$. In both CMC and CE-CMC a final sample size of 10^6 was used.

	\widehat{r}	\widehat{r}_e
CMC	5.9e-05	0.1302
CE-CMC	7.00611e-05	0.0127

Table 1: Results for CMC and CE-CMC. Exact unreliability $\bar{r} = 7.07868\text{e-}05$

By using the Cross-Entropy method we have achieved, with minimal effort, 90% reduction in relative error (a 100-fold speedup) compared to the Crude Monte Carlo method. The algorithm stopped after two iterations, as is illustrated in

Table 2. Notice that the algorithm tilted the parameters of the *mincut* elements $\{1, 3, 5\}$ to higher values while leaving the rest unimportant.

t	$\hat{\gamma}_t$	$\hat{\mathbf{v}}_t$				
0	–	0.3	0.1	0.8	0.1	0.2
1	0.507	0.964833	0.216927	1.20908	0.0892952	0.567551
2	1.000	1.19792	0.120166	1.57409	0.0630103	1.15137

Table 2: Convergence of the parameters

3.2 Cross-Entropy and Permutation Monte Carlo Simulation

We now wish to apply the Cross-Entropy method to the Permutation Monte Carlo simulation of Section 2.3. Instead of estimating \bar{r} using (18), we estimate it using Importance Sampling, where we apply a change of measure – determined by the Cross-Entropy method – to the distribution of the random permutation Π . There are many ways to define a distribution on the space of permutations, see also Remark 3.1 below. However, note that the original distribution of Π is determined by the exponential distribution of \mathbf{Y} . In fact, Π can be viewed as a function of \mathbf{Y} . Namely, we generate Y_1, \dots, Y_m independently according to $Y_i \sim \text{Exp}(u_i^{-1})$ and order the Y_i 's such that $Y_{\Pi_1} \leq Y_{\Pi_2} \leq \dots \leq Y_{\Pi_m}$. Then take

$\Pi(\mathbf{Y}) = (\Pi_1, \dots, \Pi_m)$ as our random permutation. We may thus write (17) as

$$\bar{r} = \mathbb{E}_{\mathbf{u}}[G(\Pi(\mathbf{Y}))] = \mathbb{E}_{\mathbf{u}}[S(\mathbf{Y})], \quad (21)$$

where we *redefine* $S(\mathbf{Y})$ as $G(\Pi(\mathbf{Y}))$, with G as in (16). A natural way of defining a change of measure is thus to choose different parameters v_i (instead of the nominal u_i) for the exponential distributions of the edge lifetimes, in a similar way to Section 3.1. Thus $\mathbf{v} = (v_1, \dots, v_m)$ is still the vector of mean “repair” times. However, we have a slightly different situation from Section 3.1, because instead of having to estimate a rare event probability $\mathbb{P}[S(\mathbf{Y}) \geq 1]$ we now have to estimate the (small) expectation $\mathbb{E}[S(\mathbf{Y})]$. We can no longer use a *two-phase* procedure (updating γ and \mathbf{v}) but instead use a one-phase procedure (5) in which we only update \mathbf{v}_t . The analytic solution to (5) for the i -th component of \mathbf{v}_t is

$$v_{t,i} = \frac{\sum_{k=1}^N S(\mathbf{Y}_{(k)})W(\mathbf{Y}_{(k)}; \mathbf{u}, \mathbf{v}_{t-1})Y_{(k)i}}{\sum_{k=1}^N S(\mathbf{Y}_{(k)})W(\mathbf{Y}_{(k)}; \mathbf{u}, \mathbf{v}_{t-1})},$$

where $Y_{(k)i}$ is the i -th component of $\mathbf{Y}_{(k)}$. To improve convergence in random sampling situations, sometimes it is beneficial to use a smoothing parameter α to blend the old with the new estimates. That is we take

$$\mathbf{v}'_t = \alpha \mathbf{v}_t + (1 - \alpha) \mathbf{v}_{t-1}$$

as the new parameter vector for the next iteration.

Example 3 (Bridge Network, CE and PMC)

We return to the bridge network of Example 2. Table 3 lists the results for

the standard Permutation Monte Carlo simulation, compared with the Permutation Monte Carlo simulation with Importance Sampling in which the reference parameter is determined by the Cross-Entropy method. The nominal reference parameter is again $\mathbf{u} = (0.3, 0.1, 0.8, 0.1, 0.2)$, and we use the Cross-Entropy parameters $\alpha = 0.7$ and $N = 2000$. The final sample size is $N_1 = 10^6$ in both the original and Cross-Entropy simulation.

	$\hat{\bar{r}}$	re
PMC	7.07706e-05	0.0013249
CE-PMC	7.07402e-05	0.0011772
CE-CMC	7.00611e-05	0.0127

Table 3: Results for permutation sampling with exponential repair times. Exact unreliability $\bar{r} = 7.07868e-05$

We have repeated this experiment various times and have consistently found an improvement in relative error of about 10%. Compared to the CE-CMC simulation, the Cross-Entropy with Crude Monte Carlo still has about 10 times the relative error of that in PMC or CE-PMC simulations. This shows that no matter how much you modify the Crude Monte Carlo with smart sampling techniques, it still cannot compare to the simple Permutation Monte Carlo sampling. It is well known that *conditional Monte Carlo* methods such as the PMC always yield a variance reduction over CMC, see for example Section 4.4 of [30]. However, Table 3 also suggests that even though the Cross-Entropy method does not pro-

vide the same percentage improvement as in Crude Monte Carlo scheme, it is still worthwhile to apply the technique to the “better” Permutation Monte Carlo sampling scheme.

With the CE-PMC sampling, there is no γ parameter to indicate when to stop the Cross-Entropy parameter tuning, therefore we need to use other strategies. In order to optimize the function (4), we have to resort to simulation to evaluate that function at each point. Since we do not want to spend too long on the CE parameter estimation effort, compared to the real simulation, we have imprecise knowledge of the function. As a result, we cannot use classic convergence criteria such as: stop when two consecutive vectors are ϵ close in some norm. Fortunately, however, permutation sampling depends on the relative weight of each edge and hence it is fairly insensitive to the precise values of the Importance Sampling parameter \mathbf{v}_t . Therefore we only require a vector that is in the “right” region.

Table 4 displays the evolution of the reference parameters, where we stopped the Cross-Entropy algorithm after only 3 iterations, when the estimates have “stabilised” (the values stop fluctuating). Notice again that the algorithm allocated more attention to the *mincut* elements $\{1, 3, 5\}$ and treated the rest as unimportant.

Remark 3.1 (Zero-variance IS Permutation Distribution)

For a general network, it is not difficult to find the *ideal* Importance Sampling distribution on the space of all permutations. It is simply given by the probability

t	\mathbf{v}_t				
0	0.3	0.1	0.8	0.1	0.2
1	0.435003	0.911117	2.96817	0.276884	0.195018
2	0.369376	0.0626414	0.900118	0.0594029	0.27086
3	0.383463	0.0642839	0.861578	0.0587683	0.279763

Table 4: Evolution of the reference parameters

mass function (pmf)

$$g^*(\pi) = \frac{G(\Pi)f(\pi)}{\bar{r}},$$

where $f(\pi)$ is the probability of the permutation $\pi = (e_1, \dots, e_m)$ occurring under the original measure, that is (12) with $\lambda(e_j)$ replaced with $1/u_{e_j}$. Under g^* the Importance Sampling estimator has zero variance. Although this has little practical value for large systems (for which we do not know \bar{r}), it can be of help to construct and test good Cross-Entropy sampling strategies.

4 Numerical Experiments

In this section we give a few slightly larger examples that might be found in communication networks. Figure 2 shows a 3×3 and a 6×6 grid network, each network has 4 terminals which are marked with solid nodes at the corners. All the links have the same failure probability. All the experiments use a final sample size of 10^6 and their Cross-Entropy tuning batch sample size of 5000. In the

tables, T denotes the CE tuning iterations taken and α denotes the smoothing parameter. The CE-CMC had a rarity parameter $\rho = 0.02$.

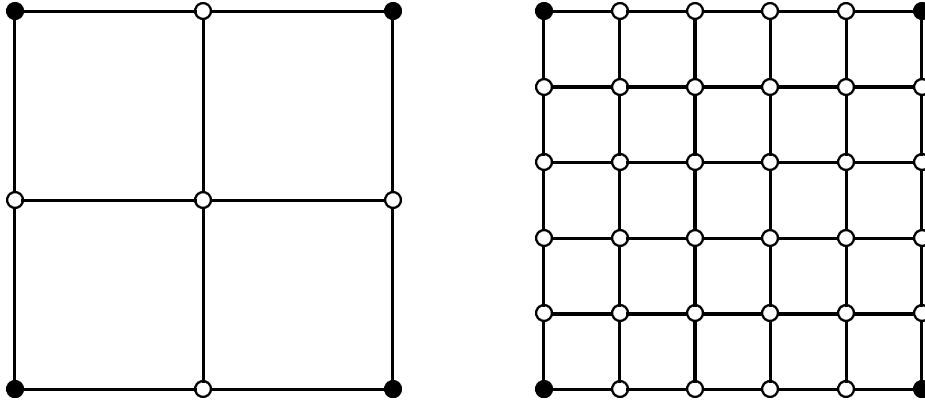


Figure 2: A 3×3 and a 6×6 grid network

Example 4 (3×3 unreliable grid)

In this example, all the links of the 3×3 grid network have the same failure probability $q = 10^{-3}$. A typical result of the simulation is given in Table 5.

	T	α	\hat{r}	\hat{r}_e
CMC	-	-	4e-06	0.499999
CE-CMC	4	1	3.08281e-06	0.335240
PMC	-	-	4.01609e-06	0.003895
CE-PMC	10	0.1	4.01057e-06	0.003420

Table 5: Simulation results for the 3×3 unreliable grid network with $q = 10^{-3}$.

Exact unreliability $\bar{r} = 4.01199\text{e-}06$.

The CMC method gives a poor relative error as expected. The CE-CMC shows a 33% reduction but is still too high to make it very useful. The PMC method gives a much smaller relative error and the Cross-Entropy method achieved an average 8-10% further reduction, making the CE-PMC a much more desirable method to use.

Example 5 (3×3 reliable grid)

This example is the same as last one except the link failure probability is $q = 10^{-6}$.

A typical result of the simulation is given in Table 6.

	T	α	\hat{r}	\hat{r}_e
CMC	-	-	0	NaN
CE-CMC	5	1	1.34033e-12	0.212152
PMC	-	-	3.99354e-12	0.003940
CE-PMC	10	0.1	4.00797e-12	0.003455

Table 6: Simulation results for the 3×3 reliable grid network with $q = 10^{-6}$.

Exact reliability $\bar{r} = 4.00001e-12$.

Clearly the failure probability in the order of 10^{-12} is well beyond the ability of the CMC method with only 10^6 samples. The Cross-Entropy method managed to help and produce some meaningful results, however, the relative error is still rather high. On the other hand the PMC method gives accurate estimates and the Cross-Entropy method consistently improves it by around 12%.

Example 6 (6×6 unreliable grid)

This is a larger example network consisting of 36 nodes and 60 edges with equal link failure probability $q = 10^{-3}$. A typical result of the simulation is given in Table 7. Again the CMC and CE-CMC methods cannot provide accurate

	T	α	$\widehat{\bar{r}}$	\widehat{re}
CMC	-	-	3e-06	0.577349
CE-CMC	4	1	3.20052e-06	0.747044
PMC	-	-	3.95377e-06	0.020306
CE-PMC	10	0.1	4.02088e-06	0.011778

Table 7: Simulation results for the 6×6 unreliable grid network with exact unreliability $\bar{r}=4.00800e-06$.

estimates with 10^6 samples while the PMC and CE-PMC give pretty good results. Notice that in this larger example, the Cross-Entropy method reduces the relative error of PMC by nearly half. Compared to the previous examples, this network has a much larger population size and variance. For instance, the 3×3 network has about 4.8×10^8 edge permutations while the 6×6 network has about 8.3×10^{81} ! Here, the Cross-Entropy method demonstrated its ability to find excellent Importance Sampling parameter vectors to reduce the sample variance.

Example 7 (6×6 reliable grid)

This example is the same as the last one except the link failure probability is

$q = 10^{-6}$. A typical result of the simulation is given in Table 8. It has very similar

	T	α	\widehat{r}	\widehat{re}
CMC	-	-	0	NaN
CE-CMC	5	1	8.36512e-14	0.362529
PMC	-	-	4.00552e-12	0.020997
CE-PMC	10	0.1	4.00455e-12	0.012755

Table 8: Simulation results for the 6×6 reliable grid network with exact unreliability $\bar{r}=4.00001e-12$.

findings to the last example: The CMC and CE-CMC methods cannot handle such a low probability with 10^6 samples. The PMC provides good estimates and yet the Cross-Entropy method improves it further by about 40%.

4.1 Summary of results

With a better “sampling structure” and smart conditioning, the Permutation Monte Carlo scheme is superior than the Crude Monte Carlo schemes. The Cross-Entropy technique further improves the performance of the PMC scheme, the degree of improvement becomes more prominent as the network size grows. Close inspection of the Importance Sampling parameter \mathbf{v}_T reveals that all the *bottleneck-cut* edges have been allocated a higher importance than the rest.

Another point to note is the smoothing parameter α . If we keep $\alpha = 0.7$ as in the bridge example, the Importance Sampling parameters \mathbf{v} might oscillate

instead of converge to the optimal \mathbf{v}^* and so give poor estimates. We found that in larger networks, a smaller smoothing parameter such as $\alpha = 0.1$ is much more robust and always give good results in our examples. Numerical experience suggests that an increase in the tuning sample size N can alleviate the need to reduce the smoothing parameter α in larger problems. Of course this means more effort has to be spent estimating the optimal Importance Sampling parameter \mathbf{v}^* . However, if we leave α very small, then more iterations would be required to approach the \mathbf{v}^* . This raises the question of the most efficient way to allocate effort in estimating \mathbf{v}^* .

5 Conclusions

The Cross-Entropy technique gives substantial improvement over the Crude Monte Carlo estimation of network reliability. However, no matter how much we improve the Crude Monte Carlo sampling, it still cannot match the simple Permutation Monte Carlo sampling. Therefore it is still better to apply the Cross-Entropy technique to the Permutation Monte Carlo schemes. In this paper, we show that the Cross-Entropy technique can be applied to further improve the Permutation Monte Carlo scheme. Furthermore, the examples suggested that the improvement can grow with the network size as the technique has the ability to “home in” to those important edges quickly.

References

- [1] G. Alon, D. Kroese, T. Raviv, and R.Y. Rubinstein. Application of the Cross Entropy Method for Buffer Allocation Problem in Simulation Based Environment. Submitted.
- [2] R.E. Barlow and F. Proschan. *Statistical Theory of Reliability and Life Testing*. Holt, Rinehart and Wilson., 1975.
- [3] C.J. Colbourn. *The Combinatorics of Network Reliability*. Oxford University Press, 1987.
- [4] C.J. Colbourn and D.D. Harms. Evaluating performability: Most probable states and bounds. *Telecommunication Systems*, 2:275–300, 1994.
- [5] P. T. de Boer. *Analysis and efficient simulation of queueing models of telecommunication systems*. PhD thesis, University of Twente, 2000.
- [6] P. T. de Boer, D. P. Kroese, and R. Y. Rubinstein. Estimating buffer overflows in three stages using cross-entropy. In *Proceedings of the 2002 Winter Simulation Conference, San Diego*, pages 301–309, 2002.
- [7] P. T. de Boer and V. F. Nicola. Adaptive state-dependent importance sampling simulation of markovian queueing networks. *European Transactions on Telecommunications*, 13(4):303–315, 2002.

- [8] P. T. de Boer, V. F. Nicola, and R. Y. Rubinstein. Adaptive importance sampling simulation of queueing networks. In *Proceedings of the 2000 Winter Simulation Conference, Orlando, Florida*, pages 646–655, 2000.
- [9] P.T de Boer, D.P. Kroese, S. Manor, and R.Y. Rubinstein. A tutorial on the cross-entropy method. *The Annals of Operations Research*, 2003. Submitted. <http://www.cs.utwente.nl/~ptdeboer/ce/>.
- [10] T. Homem de Mello and R.Y. Rubinstein. Rare event probability estimation for static models via cross-entropy and importance sampling. Submitted.
- [11] U. Dubin. The cross-entropy method for combinatorial optimization with applications. Master’s thesis, The Technion, Israel Institute of Technology, Haifa, June 2002.
- [12] M.C. Easton and C.K. Wong. Sequential destruction method for Monte Carlo evaluation of system reliability. *IEEE Trans. Reliability*, R-29:27–32, 1980.
- [13] T. Elperin, I. Gertsbakh, and M. Lomonosov. Estimation of network reliability using graph evolution models. *IEEE Transactions on Reliability*, 40(5):572–581, 1991.
- [14] 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.

- [15] G.S. Fishman. A Monte Carlo sampling plan for estimating network reliability. *Operation Research*, 34(4):122–125, 1986.
- [16] B. E. Helvik and O. Wittner. Using the cross-entropy method to guide/govern mobile agent’s path finding in networks. In *3rd International Workshop on Mobile Agents for Telecommunication Applications - MATA ’01*, 2001.
- [17] 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.
- [18] J. Keith and D.P. Kroese. Sequence Alignment By Rare Event Simulation. In *Proceedings of the 2002 Winter Simulation Conference*, pages 320–327, San Diego, 2002.
- [19] H. Kumamoto, K. Tanaka, K. Inoue, and E. J. Henley. Dagger sampling Monte Carlo for system unavailability evaluation. *IEEE Trans. Reliability*, R-29(2):376–380, 1980.
- [20] D. Lieber. *Rare-events estimation via cross-entropy and importance sampling*. PhD thesis, William Davidson Faculty of Industrial Engineering and Management, Technion, Haifa, Israel, 1998.
- [21] D. Lieber, R. Y. Rubinstein, and D. Elmakis. Quick estimation of rare events in stochastic networks. *IEEE Transaction on Reliability*, 46:254–265, 1997.

- [22] L. Margolin. Cross-entropy method for combinatorial optimization. Master's thesis, The Technion, Israel Institute of Technology, Haifa, July 2002.
- [23] O. Wittner and B. E. Helvik. Cross Entropy Guided Ant-like Agents Finding Dependable Primary/Backup Path Patterns in Networks. In *Proceedings of Congress on Evolutionary Computation (CEC2002)*, Honolulu, Hawaii, May 12-17th 2002. IEEE.
- [24] J.S. Provan and M.O. Ball. The complexity of counting cuts and of computing the probability that a graph is connected. *SIAM Journal of Computing*, 12:777–787, 1982.
- [25] R. Y. Rubinstein. Optimization of computer simulation models with rare events. *European Journal of Operations Research*, 99:89–112, 1997.
- [26] R. Y. Rubinstein. The cross-entropy method for combinatorial and continuous optimization. *Methodology and Computing in Applied Probability*, 2:127–190, 1999.
- [27] R. Y. Rubinstein. Combinatorial optimization, cross-entropy, ants and rare events. In S. Uryasev and P. M. Pardalos, editors, *Stochastic Optimization: Algorithms and Applications*, pages 304–358. Kluwer, 2001.
- [28] R. Y. Rubinstein. The cross-entropy method and rare-events for maximal cut and bipartition problems. *ACM Transactions on Modelling and Computer Simulation*, 12(1):303–315, 2002.

- [29] R.Y. Rubinstein and D.P. Kroese. *The Cross-Entropy Method: A unified approach to combinatorial optimisation, Monte-Carlo simulation and neural computation*. Wiley, 2003. In Preparation.
- [30] R.Y. Rubinstein and B. Melamed. *Modern simulation and modeling*. Wiley series in probability and Statistics, 1998.