Network Reliability Optimization via the Cross-Entropy Method

Dirk P. Kroese, Kin-Ping Hui, and Sho Nariai

***Abstract*—Consider a network of unreliable links, each of which comes with a certain price and reliability. Given a fixed budget, which links should be purchased in order to maximize the system's reliability? We introduce a new approach, based on the Cross-Entropy method, which can deal effectively with the constraints and noise (introduced when estimating the reliabilities via simulation) in this difficult combinatorial optimization problem. Numerical results demonstrate the effectiveness of the proposed technique.**

***Index Terms*—Network Reliability, Cross-Entropy Method, Monte Carlo Simulation, Noisy Optimization, Merge Process.**

NOTATION

a_i	purchase probability of link i
c_i	cost of link i
$C_{\max}$	total budget
$m,\ n$	number of [links, nodes]
N	sample size for CE
p_i	reliability of link i
r	network reliability (r^* optimal)
$\boldsymbol{x}$	network topology ($\boldsymbol{x}^*$ optimal, $\boldsymbol{X}$ random)
$\boldsymbol{y}$	system state ($\boldsymbol{Y}$ random)
α	smoothing parameter for CE
$\varphi_{\boldsymbol{x}}$	structure function of topology $\boldsymbol{x}$
ρ	rarity parameter for CE

This research was supported by the Australian Research Council, grant number DP0558957.

D. P. Kroese and S. Nariai are with Department of Mathematics, The University of Queensland, Brisbane, QLD 4072 Australia (e-mail: kroese@maths.uq.edu.au; sho@maths.uq.edu.au).

K.-P. Hui is with the Defence Science and Technology Organisation, Australia (e-mail: Kin-Ping.Hui@dsto.defence.gov.au).

I. INTRODUCTION

RAPID developments and improvements in information and communication technologies in recent years have resulted in increased capacities and higher concentration of traffic in telecommunication networks. Operating failures in such high-capacity networks can affect the quality of service of a large number of consumers. Consequently, the careful planning of a network's infrastructure and the detailed analysis of its reliability become more and more important, in order to ensure that consumers obtain the best service possible.

One of the most basic and useful approaches to network reliability analysis is to represent the network as an undirected graph with unreliable links. The reliability of the network is usually defined as the probability that certain nodes in the graph are connected by functioning links.

This paper is concerned with network *planning*, where the objective is to maximize the network's reliability, subject to a fixed budget. More precisely: given a fixed amount of money, and starting with a non-existent network, the question is which network links should be purchased in order to maximize the reliability of the finished network. Each link carries a pre-specified price and reliability.

There are two reasons why this *Network Planning Problem* (NPP) is difficult to solve. First, the NPP is a constrained integer programming problem known as the Minimum Steiner Problem, which is a 0-1 knapsack problem with non-linear

objective. It is well known that such a problem is NP hard [1], and it is also APX-Complete [2]. For small networks exact methods, such as branch-and-bound, dynamic programming or convexification, may be successful (see for example [3]); but, since the complexity of the problem increases exponentially with the number of links, such methods quickly become infeasible for moderate and large-scale problems.

Second, for large networks the value of the objective function — that is, the network reliability — becomes difficult or impractical to evaluate [4], [5]. A viable option then is to use simulation to estimate the network reliability, for example via the Crude Monte Carlo (CMC) technique. This noisy version of the problem is not even in NP, because the value of a given solution is hard to compute. Moreover, for highly reliable networks — which typically occur in communication networks — CMC requires a very large simulation effort in order to estimate the reliability accurately.

A number of simulation techniques have been developed to address the network reliability estimation problem. For example, Kumamoto *et al.* [6] proposed a simple technique called *Dagger Sampling* to improve the efficiency of CMC simulation. Fishman [7] introduced *Procedure Q*, which can provide reliability estimates as well as bounds. Colbourn and Harms [8] proposed a technique that provides progressive bounds that eventually converge to an exact reliability value. Elperin *et al.* [9], [10] developed *Evolution Models* for estimating the reliability of highly reliable networks. Hui *et al.* [11], [12] proposed a hybrid scheme that provides bounds and can provide a speed-up by several orders of magnitude in certain classes of networks. They also proposed another scheme [13] which employs the Cross-Entropy technique to speed-up the estimation in general classes of networks. Other relevant references on network reliability include [14], [15], [16].

The literature on network planning — rather than reliability estimation — is not extensive, and virtually all studies pertain to networks for which the system reliability can either be evaluated exactly, or sharp reliability bounds can be established. Cancela and Urquhart [17] employed a Simulated Annealing scheme to obtain a more reliable alternative network, given a user-defined network topology. Dengiz *et al.* used a Genetic Algorithm to optimize the design of communication network topologies subject to the minimum reliability requirement [18]. Yeh *et al.* [19] proposed a method based on a Genetic Algorithm to optimize the $k-$node set reliability subject to a specified capacity constraint. Reichelt *et al.* [20] used a Genetic Algorithm in combination with a repair heuristic to minimize the network cost subject to a specified network reliability constraint. Other heuristics can be found in [21].

To our knowledge, no simple algorithm is known that can tackle *at the same time* the combinatorial, constraint and noisy aspects of the NPP, and the purpose of this paper is to introduce such a method, and provide a new and effective approach to network planning. Our approach is based on the *Cross-Entropy* (CE) method [22], which was introduced in [23] as an adaptive technique for estimating probabilities of rare events in complex stochastic networks. It was soon realized [24], [25] that it could be used not only for rare event simulation but for solving difficult combinatorial optimization problems as well. Moreover, (and this is especially relevant for the NPP) the CE method is well-suited to solving *noisy* optimization problems; examples are the Buffer Allocation Problem [26], the Vehicle Routing Problem [27], and the Stochastic Shortest Path Problem [28]. For the NPP we will consider both the deterministic case, where the network reliability can be computed exactly, and the noisy case where it is estimated via simulation. A tutorial on the CE method can be found in [29], which is also available on-line from the CE homepage: `http://www.cemethod.org`.

The rest of the paper is organized as follows: In Section II we formulate the NPP in mathematical terms. In Section III we present the CE approach to the problem. This is further

developed in Section IV for the noisy case, in particular with respect to variance reduction techniques such as Permutation Monte Carlo and the Merge Process. Section V focuses on implementation issues with regard to speeding up the algorithm. We illustrate the effectiveness of the CE approach via a number of numerical experiments in Section VI. Finally, in Section VII we present our conclusion and directions for future work.

II. Problem Description

Consider an undirected graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, with set $\mathcal{V}$ of nodes (vertices), and set $\mathcal{E}$ of links (edges). Suppose the number of links is $|\mathcal{E}| = m$. Without loss of generality we may label the links $1, \ldots, m$. Let $\mathcal{K} \subseteq \mathcal{V}$ be a set of *terminal* nodes. With each of the links is associated a *cost* c_e and *reliability* p_e. The objective is to purchase those links that optimize the reliability of the network — defined as the probability that all the terminal nodes are connected by functioning links — subject to a total budget $C_{\max}$. Let $\boldsymbol{c} = (c_1, \ldots, c_m)$ denote vector of link costs, and $\boldsymbol{p} = (p_1, \ldots, p_m)$ the vector of link reliabilities.

We introduce the following notation. For each link e let x_e be such that

$$x_e = \begin{cases} 1 & \text{if link } e \text{ is purchased,} \\ 0 & \text{otherwise.} \end{cases}$$

We call the vector $\boldsymbol{x} = (x_1, \ldots, x_m)$ the *purchase vector*. The set of all possible purchase vectors is denoted by $\mathcal{X}$.

To identify the operational links, we define for each link e the link *state* by

$$y_e = \begin{cases} 1 & \text{if link } e \text{ is functioning,} \\ 0 & \text{otherwise.} \end{cases}$$

Note that for each link e that is *not* purchased, the state y_e is per definition equal to 0. The vector $\boldsymbol{y} = (y_1, \ldots, y_m)$ is called the *state vector*. For each purchase vector $\boldsymbol{x}$ let $\varphi_{\boldsymbol{x}}$ be the *structure function* of the purchased system. This function assigns to each state vector $\boldsymbol{y}$ the state of the system (working = 1 or failed = 0). That is,

$$\varphi_{\boldsymbol{x}}(\boldsymbol{y}) = \begin{cases} 1 & \text{if all the terminal nodes are connected,} \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

Now, consider the situation with *random* states, where each purchased link e works with probability p_e. Let Y_e be random state of link e, and let $\boldsymbol{Y}$ be the corresponding random state vector. The reliability of the network defined by purchase vector $\boldsymbol{x}$ is given by

$$r(\boldsymbol{x}) = \mathbb{E}[\,\varphi_{\boldsymbol{x}}(\boldsymbol{Y})] = \sum_{\boldsymbol{y}} \varphi_{\boldsymbol{x}}(\boldsymbol{y}) \Pr\{\boldsymbol{Y} = \boldsymbol{y}\} \,. \tag{2}$$

We assume from now on that the links fail s-independently, that is, $\boldsymbol{Y}$ is a vector of s-independent Bernoulli random variables, with success probability p_e for each purchased link e and 0 otherwise. Defining $\boldsymbol{p}_{\boldsymbol{x}} = (x_1 p_1, \ldots, x_m p_m)$ as the vector of probabilities of the components of $\boldsymbol{Y}$, for a given purchase vector $\boldsymbol{x}$, we write $\boldsymbol{Y} \sim \mathsf{Ber}(\boldsymbol{p}_{\boldsymbol{x}})$. It follows that for each $\boldsymbol{x}$, the reliability is computed as

$$r(\boldsymbol{x}) = \sum_{\boldsymbol{y}} \varphi_{\boldsymbol{x}}(\boldsymbol{y}) \prod_{j=1}^{m} (x_j p_j)^{y_j} (1 - x_j p_j)^{1 - y_j}, \tag{3}$$

where $0^0 := 1$. Our main purpose is to determine

$$\max_{\boldsymbol{x} \in \mathcal{X}} r(\boldsymbol{x}) \,, \tag{4}$$

subject to the constraint on the total budget

$$\sum_{e \in \mathcal{E}} x_e c_e \leq C_{\max} \,. \tag{5}$$

Let $r^* := r(\boldsymbol{x}^*)$ denote the optimal reliability of the network, where $\boldsymbol{x}^*$ is the optimal purchase vector.

III. Cross-Entropy Approach

In this section we show how the CE method can be used to solve the constrained combinatorial optimization problem (4), (5). The CE method consists of two steps which are iterated:

1) generate random purchase vectors $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_N$ according to some specified random mechanism, and
2) update the parameters of this mechanism in order to obtain better system reliabilities in the next iteration.

An efficient method to generate random purchase vectors that satisfy (5) is as follows: First, generate a "uniform" permutation $(e_1, \ldots, e_m)$ of $(1, \ldots, m)$, by s-independently drawing m numbers from the uniform distribution on $[0, 1]$ and letting $e_1, \ldots, e_m$ correspond to the indices of the ordered observations. Second, given such a permutation, flip a coin with success probability a_{e_1} to decide whether to purchase link e_1 or not. If successful and if there is enough money available to purchase link e_1, set $X_{e_1} = 1$, that is, link e_1 is purchased; otherwise set $X_{e_1} = 0$. We repeat the above procedure for links e_2, e_3, etc. For each link e_i we check whether the remaining budget allows us to purchase the link, and if so, we purchase the link with probability a_{e_i}. The main algorithm for generating a random purchase vector using uniform permutation is thus summarized as follows:

Algorithm 1 [Generation Algorithm]**.**

1) *Generate a uniform random permutation* $(e_1, \ldots, e_m)$. *Set* $k = 1$.
2) *Calculate* $C = c_{e_k} + \sum_{i=1}^{k-1} X_{e_i} c_{e_i}$.
3) *If* $C \leq C_{\max}$, *draw* $X_{e_k} \sim \mathsf{Ber}(a_{e_k})$. *Otherwise set* $X_{e_k} = 0$.
4) *If* $k = m$, *then stop; otherwise set* $k = k + 1$ *and reiterate from step 2.*

The usual CE procedure [22] is to construct a sequence of reference vectors $\{\boldsymbol{a}_t, t \geq 0\}$ (i.e., purchase probability vectors), such that $\{\boldsymbol{a}_t, t \geq 0\}$ converges to the degenerate (i.e., binary) probability vector $\boldsymbol{a}^*$ that corresponds to the optimal purchase vector $\boldsymbol{x}^* = \boldsymbol{a}^*$. The sequence of reference vectors is obtained via a two-step procedure, involving an auxiliary sequence of reliability levels $\{\gamma_t, t \geq 0\}$ that tends to the optimal reliability $\gamma^* = r^*$ at the same time as the $\{\boldsymbol{a}_t\}$ tend to $\boldsymbol{a}^*$. At each iteration t, for a given $\boldsymbol{a}_{t-1}$, γ_t is the $(1-\rho)$-quantile of performances (reliabilities). Typically ρ is chosen between 0.01 and 0.1. An estimator $\widehat{\gamma}_t$ of γ_t is the corresponding sample $(1-\rho)$-quantile. That is, generate a random sample $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_N$ using the generation algorithm above; compute the performances $r(\boldsymbol{X}_i),\ i = 1, \ldots, N$ and let

$$\widehat{\gamma}_t = r_{(\lceil (1-\rho)N \rceil)}, \tag{6}$$

where $r_{(1)} \leq \ldots \leq r_{(N)}$ are the order statistics of the performances. The reference vector is updated via CE minimization, which (see [22]) reduces to the following: For a given fixed $\boldsymbol{a}_{t-1}$ and γ_t, let

$$a_{t,j} = \mathbb{E}_{\boldsymbol{a}_{t-1}}[X_j \,|\, r(\boldsymbol{X}) \geq \gamma_t] \,.$$

An estimator $\widehat{\boldsymbol{a}}_t$ of $\boldsymbol{a}_t$ is computed via

$$\widehat{a}_{t,j} = \frac{\sum_{i=1}^N I_{\{r(\boldsymbol{X}_i) \geq \widehat{\gamma}_t\}} X_{ij}}{\sum_{i=1}^N I_{\{r(\boldsymbol{X}_i) \geq \widehat{\gamma}_t\}}}, \quad j = 1, \ldots, m, \tag{7}$$

where we use the *same* random sample as in (6), and where X_{ij} is the j-th coordinate of $\boldsymbol{X}_i$. The main CE algorithm for optimizing (4) using the above generation algorithm is thus summarized as follows:

Algorithm 2 [Main CE Algorithm]**.**

1) *Initialize* $\widehat{\boldsymbol{a}}_0$. *Set t=1 (iteration counter).*
2) *Generate a random sample* $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_N$ *using Algorithm 1, with* $\boldsymbol{a} = \widehat{\boldsymbol{a}}_{t-1}$. *Compute the sample* $(1-\rho)$*-quantile of performances* $\widehat{\gamma}_t$ *using (6).*
3) *Use the* ***same*** *sample to update* $\widehat{\boldsymbol{a}}_t$, *using (7).*
4) *If*

$$\max(\min(\widehat{\boldsymbol{a}}_t, 1 - \widehat{\boldsymbol{a}}_t)) \leq \beta \tag{8}$$

for some small fixed β, *then stop (let* T *be the final iteration); otherwise set* $t = t + 1$ *and reiterate from step 2.*

Note that the cost vector $\boldsymbol{c}$, reliability of links $\boldsymbol{p}$, the initial reference vector $\widehat{\boldsymbol{a}}_0$, the sample size N, total budget $C_{\max}$, the rarity parameter ρ, and the stopping parameter β need to be specified in advance.

Remark 1 [Smoothed Updating]**.** Instead of updating directly using (7), one may choose to use a *smoothed updating* procedure

$$\widehat{\boldsymbol{a}}_t = \alpha\ \widetilde{\boldsymbol{a}}_t + (1 - \alpha)\ \widehat{\boldsymbol{a}}_{t-1} \tag{9}$$

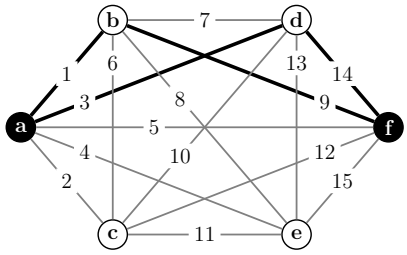


Fig. 1. 6-node complete graph.

TABLE I

LINK COSTS AND RELIABILITIES

i	c_i	p_i	i	c_i	p_i	i	c_i	p_i
1	331	0.9951	6	335	0.9958	11	330	0.9947
2	347	0.9968	7	332	0.9952	12	325	0.9937
3	327	0.9942	8	302	0.9902	13	324	0.9935
4	340	0.9959	9	344	0.9964	14	350	0.9973
5	2000	0.9908	10	315	0.9917	15	312	0.9912

where $\widetilde{\boldsymbol{a}}_t$ is the parameter vector obtained via (7) and α is called the *smoothing parameter*. It is easily seen that for $\alpha = 1$ the original updating procedure is obtained. By setting the smoothing parameter between $0 < \alpha < 1$ we take the past into account when updating the parameter vector.

Remark 2 [Unreliability]**.** In many applications the link and network reliabilities are close to 1. The appropriate quantity to consider is then the network *unreliability* $\bar{r} = 1 - r$. In such case Algorithm 2 can be readily modified to minimize the unreliability, rather than to maximize the reliability. The only differences are that $\hat{\gamma}_t$ now represents the sample ρ-quantile of the unreliabilities, and that $r(\boldsymbol{X}_i) \geq \hat{\gamma}_t$ in (7) is replaced with $\bar{r}(\boldsymbol{X}_i) \leq \hat{\gamma}_t$.

Example 1. On offer is a 6-node fully connected graph given in Figure 1. The two black nodes in the graph represent the terminal nodes. The network is functioning if the two terminal nodes are connected by operational links. The link costs and reliabilities are given in Table I. The total budget $C_{\max}$ is equal to 1500. The optimal purchase vector can be calculated (by total enumeration) to be $\boldsymbol{x}^* = (1,0,1,0,0,0,0,0,1,0,0,0,0,1,0)$, which gives a network unreliability of $\bar{r}^* = 1 - r^* =$ 7.1967e-05. The four black links form the optimal network.

Table II displays the evolution of the purchase probability vector for this problem. We used the following CE parameters: $N = 750,\ \rho = 0.1,\ \alpha = 0.7,\ \beta = 0.05$ and we took $\widehat{\boldsymbol{a}}_0 = (0.5,\ldots,0.5)$. We see that as $t \to \infty,\ \widehat{\gamma}_t$ and $\widehat{\boldsymbol{a}}_t$ quickly approaches $\overline{r}^*$ and $\boldsymbol{x}^*$ respectively.

IV. NOISY OPTIMIZATION

As mentioned in the introduction, for networks involving a large number of links the exact evaluation of the network reliability is in general not feasible, and simulation becomes a viable option. In the corresponding simulation-based optimization problem the objective function (the network reliability) is thus corrupted by noise. In this section we show how the CE method can be easily modified to tackle such *noisy* NPPs.

In order to adapt Algorithm 2 we again, at iteration t, generate a random sample $\boldsymbol{X}_1,\ldots,\boldsymbol{X}_N$ according the $\mathsf{Ber}(\widehat{\boldsymbol{a}}_{t-1})$-distribution. However, the corresponding performances (network reliabilities) are now not computed exactly, but estimated. For example, estimation via CMC involves, for each vector $\boldsymbol{X}_i$, drawing a random sample of state vectors $\boldsymbol{Y}_1,\ldots,\boldsymbol{Y}_K$, each according to a $\mathsf{Ber}(\boldsymbol{p}_{\boldsymbol{X}_i})$-distribution, and estimating the performance as

$$\widehat{r}(\boldsymbol{X}_i) = \frac{1}{K}\sum_{j=1}^{K} \varphi_{\boldsymbol{X}_i}(\boldsymbol{Y}_j), \quad i = 1,\ldots,N. \tag{10}$$

The updating formula is similar to (7). The only difference is that $r(\boldsymbol{X}_i)$ is replaced with $\widehat{r}(\boldsymbol{X}_i)$. Therefore the updating formula at t-th iteration is given by

$$\widehat{a}_{t,j} = \frac{\sum_{i=1}^{N} I_{\{\widehat{r}(\boldsymbol{X}_i)\geq\widehat{\gamma}_t\}} X_{ij}}{\sum_{i=1}^{N} I_{\{\widehat{r}(\boldsymbol{X}_i)\geq\widehat{\gamma}_t\}}}, \quad j = 1,\ldots,m. \tag{11}$$

The main CE algorithm for estimating the reliability of a network is summarized as follows:

Algorithm 3 [Noisy Version of the CE Algorithm]**.**

1) *Initialize* $\widehat{\boldsymbol{a}}_0$. *Set* $t = 1$ *(iteration counter).*
2) *Generate a random sample* $\boldsymbol{X}_1,\ldots,\boldsymbol{X}_N$ *using Algorithm 1. Let* $\widehat{r}_{(1)},\ldots,\widehat{r}_{(N)}$ *be the order statistics of*

TABLE II

THE EVOLUTION OF THE CE ALGORITHM WITH $C_{\max} = 1500,\ N = 750,\ \rho = 0.1,\ \alpha = 0.7$, AND $\beta = 0.05$

t	$\widehat{\gamma}_t$	$\widehat{\boldsymbol{a}}_t$														
0		0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50
1	8.484e-03	0.60	0.29	0.48	0.28	0.15	0.24	0.22	0.27	0.59	0.24	0.31	0.30	0.24	0.49	0.36
2	8.482e-03	0.80	0.18	0.34	0.18	0.05	0.18	0.17	0.18	0.80	0.19	0.18	0.19	0.13	0.36	0.33
3	8.482e-03	0.92	0.14	0.34	0.14	0.01	0.18	0.24	0.11	0.92	0.07	0.07	0.10	0.04	0.35	0.33
4	4.927e-03	0.98	0.07	0.53	0.13	0.00	0.07	0.41	0.05	0.95	0.02	0.02	0.04	0.01	0.53	0.17
5	7.197e-05	0.99	0.02	0.86	0.04	0.00	0.02	0.12	0.02	0.98	0.01	0.01	0.01	0.00	0.86	0.05
6	7.197e-05	1.00	0.01	0.96	0.01	0.00	0.01	0.04	0.00	1.00	0.00	0.00	0.00	0.00	0.96	0.02

$\widehat{r}(\boldsymbol{X}_1), \ldots, \widehat{r}(\boldsymbol{X}_N)$. *Let* $\widehat{\gamma}_t = \widehat{r}_{(\lceil (1-\rho)N \rceil)}$.

3) *Use the* ***same*** *sample to update* $\widehat{\boldsymbol{a}}_t$ *using* (11)*.*
4) *Stop if* (8) *holds for some small fixed* β *(let* T *be the final iteration); otherwise set* $t = t + 1$ *and reiterate from step 2.*

We could take $\widehat{\boldsymbol{a}}_T$ as our final purchase vector, if the latter were binary. Since $\widehat{\boldsymbol{a}}_T$ is close to, but not exactly binary, we round $\widehat{\boldsymbol{a}}_T$ to the nearest binary vector, denote the solution by $\widehat{\boldsymbol{a}}^*$, and take this rounded vector as our solution for the NPP. If, in addition, we wish to obtain an estimate of the optimal reliability r^*, we generate a (larger) sample $\boldsymbol{Y}_1, \ldots, \boldsymbol{Y}_{N_1}$ from $\mathsf{Ber}(\widehat{\boldsymbol{a}}^*)$, and estimate r^* via

$$\widehat{r^*} = \frac{1}{N_1} \sum_{i=1}^{N_1} \varphi_{\widehat{\boldsymbol{a}}^*}(\boldsymbol{Y}_i). \tag{12}$$

Remark 3 [Choice of K]**.** Note that the simulation time can be reduced by choosing a relatively small K in (10), say $K = 100$ in Example 1. However, this corresponds to a relatively large variance of the reliability estimator, which in turn could lead to a suboptimal convergence of the algorithm. On the other hand, choosing a larger K, say $K = 20000$ in Example 1, can increase the accuracy and chance of locating an optimal solution but at a cost of increased simulation time. To overcome this, one may choose K adaptively by setting

$$K = \min\{\lceil K_{\min} \times N / N_{\text{unique}} \rceil, K_{\max}\}\ , \tag{13}$$

where N_{unique} is the number of unique networks generated with no repetitions and $K_{\min}$ and $K_{\max}$ are the minimum and maximum values of K allowed in the simulation. The idea of using (13) is simple: In the early stages of the simulation, when $N_{\text{unique}} \approx N$, we take $K \approx K_{\min}$ to quickly narrow the search space. During the course of the simulation the search space becomes smaller and the algorithm starts to generate networks with similar topologies and performances, so that N_{unique} decreases. When this starts to happen, K is increased automatically to accurately distinguish between networks with similar reliabilities.

It is important to realize that CMC only works satisfactory when the network reliability is neither too small nor too large. For example, consider a highly reliable network. For any given purchase vector $\boldsymbol{x}$ the CMC estimator of the (very small) unreliability $\overline{r}(\boldsymbol{x}) = 1 - r(\boldsymbol{x})$, which is the appropriate quantity to consider here, is given by

$$\widehat{\overline{r}}(\boldsymbol{x}) := \frac{1}{N_1} \sum_{i=1}^{N_1} (1 - \varphi_{\boldsymbol{x}}(\boldsymbol{Y}_i)),$$

and the corresponding s-relative error is

$$\epsilon = \frac{\sqrt{\mathrm{Var}(\widehat{\overline{r}}(\boldsymbol{x}))}}{\mathbb{E}[\widehat{\overline{r}}(\boldsymbol{x})]} = \sqrt{\frac{1 - \overline{r}(\boldsymbol{x})}{\overline{r}(\boldsymbol{x})\, N_1}} \approx \sqrt{\frac{1}{\overline{r}(\boldsymbol{x})\, N_1}}, \tag{14}$$

which shows that for $\epsilon = 0.01$, say, we need a sample size of at least $N_1 \approx 10^4 / \overline{r}(\boldsymbol{x})$, which can be prohibitively large.

Permutation Monte Carlo

A more efficient way of estimating the network unreliability in highly reliable networks is *Permutation Monte Carlo* (PMC) [9]. The idea is as follows. Consider a network with structure function as in (1), and reliability $r = r(\boldsymbol{x})$ as in (2). Let l be the number of purchased links and let $\mathcal{E}^{\boldsymbol{x}}$ be the set of

purchased links. We assume here that $\varphi_{\boldsymbol{x}}(\boldsymbol{x}) = 1$, so that the purchased network functions if all its links work. Now, observe a *dynamic* network $\mathcal{G}(\mathcal{V}, \mathcal{E})$ in which each purchased link e has an exponential repair time with repair rate $\lambda(e) = -\log(1 - p_e)$. The repair rate for each link e that is not purchased is set to $\lambda(e) = \infty$, so that the link does not become operational in finite time. At time $t = 0$ all links are failed. Assume that all repair times are s-independent of each other. The state of e at time t is denoted by $Y_e(t)$ and the state of the link set $\mathcal{E}$ at time t is given by the vector $\boldsymbol{Y}(t)$, defined in a similar way as before. Then, $(\boldsymbol{Y}(t))$ 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 links are constructed (become operational) in finite time, and let $A_0, A_0 + A_1, \ldots, A_0 + \cdots + A_{l-1}$ be the times at which those links are constructed. Hence the $\{A_i\}$ are *sojourn times* of $(\boldsymbol{Y}(t))$. Π is a random variable which takes values in the space $\{(e_1, \ldots, e_l) \in \{1, \ldots, m\}^l : e_i \neq e_j, j \neq i\}$.

For any such $\pi = (e_1, \ldots, e_l)$ define

$$\begin{aligned} \mathcal{E}_0 &= \mathcal{E}^{\boldsymbol{x}}, \\ \mathcal{E}_i &= \mathcal{E}_{i-1} \setminus \{e_i\}, \qquad 1 \le i \le l-1, \\ \lambda(\mathcal{E}_i) &= \sum_{e \in \mathcal{E}_i} \lambda(e). \end{aligned}$$

Let

$$b(\pi) = \min_i \{\varphi(\mathcal{E}^{\boldsymbol{x}} \setminus \mathcal{E}_i) = 1\}$$

be the *critical number* of π, that is, the number of repairs required to bring the system up in the order specified by π. From the general theory of Markov processes it is not difficult to see that

$$\Pr\{\Pi = \pi\} = \prod_{j=1}^{l} \frac{\lambda(e_j)}{\lambda(\mathcal{E}_{j-1})}\ . \tag{15}$$

Moreover, conditional on $\{\Pi = \pi\}$, the sojourn times $A_0, \ldots, A_{l-1}$ are s-independent and each A_i is exponentially distributed with parameter $\lambda(\mathcal{E}_i)$, $i = 0, \ldots, l-1$.

Recall that each link e has an exponential repair rate $\lambda(e) = -\log(1 - p_e)$. Let T_e be the corresponding repair time. It follows that $\Pr\{T_e > 1\} = \exp\{\log(1 - p_e)\} = 1 - p_e$. Therefore, the probability of link e being operational at time $t = 1$ is p_e, and hence the probability that the network is functioning at time $t = 1$ is precisely the *network reliability*. By conditioning on Π we have

$$\begin{aligned} r &= \mathbb{E}[\varphi(\boldsymbol{Y}(1))] \\ &= \sum_{\pi} \Pr\{\Pi = \pi\} \Pr\{\varphi(\boldsymbol{Y}(1)) = 1 \,|\, \Pi = \pi\}, \end{aligned} \tag{16}$$

and

$$\begin{aligned} \overline{r} &= 1 - r \\ &= \sum_{\pi} \Pr\{\Pi = \pi\} \Pr\{\varphi(\boldsymbol{Y}(1)) = 0 \,|\, \Pi = \pi\}. \end{aligned} \tag{17}$$

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 $t \ge 0$ we have

$$\begin{aligned} &\Pr\{\varphi(\boldsymbol{Y}(t)) = 0 \,|\, \Pi = \pi\} \\ &\qquad = \Pr\{A_0 + \cdots + A_{b(\pi)-1} > t \,|\, \Pi = \pi\} \\ &\qquad = 1 - \underset{0 \le i \le b(\pi)-1}{\mathbf{Conv}} \{1 - \exp[-\lambda(\mathcal{E}_i)\, t]\}. \end{aligned} \tag{18}$$

Let

$$G(\pi) = \Pr\{\varphi(\boldsymbol{Y}(1)) = 0 \,|\, \Pi = \pi\}, \tag{19}$$

as given in (18). Equation (17) can then be rewritten as

$$\overline{r} = \mathbb{E}[G(\Pi)], \tag{20}$$

and this shows how the CP simulation scheme works. Namely, let $\Pi^{(1)}, \ldots, \Pi^{(K)}$ be s-independent identically distributed random vectors, each distributed according to Π. Then

$$\widehat{\overline{r}} = \frac{1}{K} \sum_{i=1}^{K} G(\Pi^{(i)}) \tag{21}$$

is an unbiased estimator for $\overline{r}$, where each $G(\Pi^{(i)})$ can be calculated as the convolution of exponential functions.

Merge Process

A careful study of the evolution of the CP shows that many of the results remain valid when we combine various states to form "super-states" at various stages of the process. The mathematical formulation is as follows (see [9], [30] for more

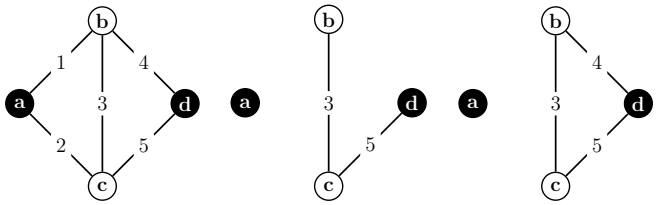


Fig. 2. A bridge network, the proper partition $\sigma = \{\{a\}, \{b, c, d\}\}$, and its corresponding closure $\mathcal{F}_\sigma = \{3, 4, 5\}$.

details): Given the graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ and a subset $\mathcal{F} \subseteq \mathcal{E}$ of edges, a partition $\sigma = \{\mathcal{V}_1, \ldots, \mathcal{V}_k\}$ of $\mathcal{V}$ is said to be *proper* (with respect to $\mathcal{F}$) if each induced subgraph $\mathcal{G}(\mathcal{V}_i)$ of the subgraph $\mathcal{G}(\mathcal{V}, \mathcal{F})$ is connected. Let $\mathcal{F}_i$ be the edge-set of the induced subgraph $\mathcal{G}(\mathcal{V}_i)$. The set $\mathcal{F}_\sigma = \bigcup_{i=1}^{k} \mathcal{F}_i$ of edges is the *closure* of $\mathcal{F}$. The easiest way to visualize the "super-state" σ is to identify it with the graph $\mathcal{G}(\mathcal{V}, \mathcal{F}_\sigma)$, as in Figure 2. Here $\mathcal{F} = \{3, 5\}$, which induces the proper partition $\sigma = \{\{a\}, \{b, c, d\}\}$, so that $\mathcal{F}_1 = \emptyset$ and $\mathcal{F}_2 = \{3, 4, 5\}$ and $\mathcal{F}_\sigma = \{3, 4, 5\}$.

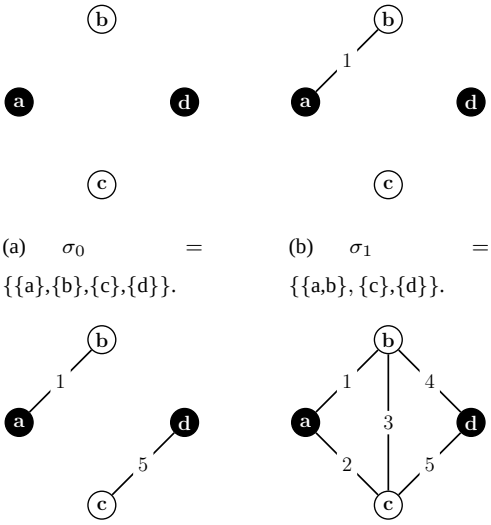
Let $\mathbb{L}(\mathcal{G})$ be the collection of all proper partitions of $\mathcal{V}$, ordered by the relation $\sigma \prec \tau \Leftrightarrow \mathcal{F}_\sigma \subset \mathcal{F}_\tau$, where τ is obtained by merging components of σ.

Recall the CP $(\boldsymbol{Y}(t))$ of the network defined by $\boldsymbol{x}$. The CP induces a Markov process $(\mathbb{Y}(t))$ on $\mathbb{L}(\mathcal{G})$, called the *Merge Process* (MP). Initially, this process starts in the super-state σ_0 in which all nodes are isolated from each other, and it ends in the super-state σ_ω corresponding to the original network with all edges functioning. Furthermore, at any time $t \geq 0$, $\varphi(\boldsymbol{Y}(t)) = \varphi(\mathbb{Y}(t))$.

For each $\sigma \in \mathbb{L}(\mathcal{G})$, the sojourn time in σ has an exponential distribution with parameter $\lambda(\sigma) := \sum_{e \in \mathcal{E}_\sigma} \lambda(e)$, s-independent of other partitions, where $\mathcal{E}_\sigma = \mathcal{E} - \mathcal{F}_\sigma$ and the transition from a current partition σ to one of its successors τ occurs with probability

$$\frac{\lambda(\sigma) - \lambda(\tau)}{\lambda(\sigma)} .$$

We define a *trajectory* of $(\mathbb{Y}(t))$ as a sequence $\theta = (\sigma_0, \ldots, \sigma_{b(\theta)})$, where $b(\theta)$ is equal to the number of transitions required in order for the network to become operational.



(a) σ_0 = {{a},{b},{c},{d}}. (b) σ_1 = {{a,b}, {c},{d}}. (c) σ_2 = {{a,b}, {c,d}}. (d) σ_3 = {{a,b,c,d}}.

Fig. 3. A possible transition sequence of the MP $(\mathbb{Y}(t))$.

For example, consider the bridge network given in Figure 2. The network is operational when the two black nodes are connected by operational links. A possible transition sequence from the initial state is shown in Figure 3.

The network becomes operational upon the transition from σ_2 to σ_3. Therefore $b(\theta) = 3$.

Since $\varphi(\boldsymbol{Y}(t)) = \varphi(\mathbb{Y}(t))$, the probability that the network is not functioning at time $t = 1$ is equal to the network unreliability. Thus the network unreliability can be written as

$$\overline{r} = \sum_{\theta} \Pr\{\Theta = \theta\} \Pr\{\varphi(\mathbb{Y}(1)) = 0 \,|\, \Theta = \theta\}. \tag{22}$$

For any $t \geq 0$, we can write the last probability as

$$\Pr\{\varphi(\mathbb{Y}(t)) = 0 \,|\, \Theta = \theta\} = 1 - \underset{0 \leq j \leq b(\theta)-1}{\mathbf{Conv}} \{1 - \exp[-\lambda(\sigma_j)\, t]\}. \tag{23}$$

Let

$$G_{\mathrm{MP}}(\theta) = \Pr\{\varphi\mathbb{Y}(1) = 0 \,|\, \Theta = \theta\}. \tag{24}$$

as given in (23). Equation (22) can be rewritten as

$$\overline{r} = \mathbb{E}[G_{\mathrm{MP}}(\Theta)] \tag{25}$$

where Θ is the random trajectory in $(\mathbb{Y}(t))$. Let $\Theta_1, \ldots, \Theta_K$ be the s-independent and identically distributed random tra-

jectories, each distributed according to Θ. Then

$$\widehat{\overline{r}} = \frac{1}{K}\sum_{i=1}^{K} G_{\mathrm{MP}}(\Theta_i) \tag{26}$$

is an unbiased estimator for $\overline{r}$. A simple way of generating such trajectories is by first generating a random vector Π exactly as in the CP. The sequence of distinct partitions obtained from the order in which the links come up (given by Π) determines a unique trajectory Θ.

V. Implementation Issues

Algorithm 3 is designed to find a single optimal solution. However, there are situations where the purchase probability of a certain link i, $\widehat{a}_{t,i},\ i \in \{1,\ldots,m\}$, could oscillate for a very long time before converging to either 0 or 1, namely when

1) many of the candidate networks have reliabilities that are very close to the optimal network reliability;
2) there are superfluous links in the network, that is, links that do not affect the overall network reliability.

The result of such oscillatory behavior is that the simulation time increases significantly. In this section, we introduce two algorithms, the hybrid CE algorithm and the superfluous link removal algorithm, which can reduce the computational effort.

A. Hybrid CE Method

In the situation where only one optimal network exists but where many networks can be purchased whose network (un)reliabilities are very close to the optimal one, the elements of the purchase probability vector could oscillate, and this would increase the computational effort. In such cases, one may terminate the algorithm once the number of purchase probabilities that lie between $[\beta, 1-\beta]$ falls below a certain threshold, say δ, and then generate *all* candidate networks according to the purchase probability vector. We evaluate the performances of these networks and take the network with the best performance as our final solution to the problem.

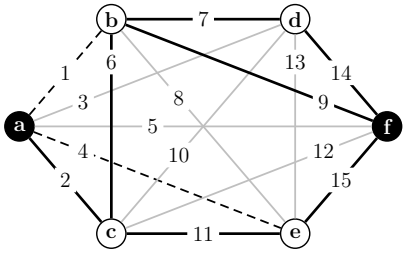
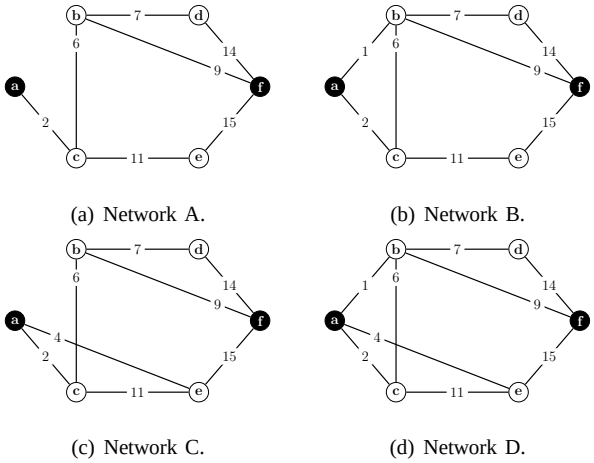


Fig. 4. Example network.



(a) Network A. (b) Network B.

(c) Network C. (d) Network D.

Fig. 5. Candidate networks.

As an example, consider the 6-node fully connected graph given in Figure 4.

The black and gray solid lines indicate that the purchase probabilities of these links lie in the ranges $[1-\beta, 1]$ and $[0, \beta]$ respectively. The two dashed links indicate that the corresponding purchase probabilities lie in the interval $[\beta, 1-\beta]$. In such case, we terminate the main CE loop (steps 2 and 3 of the algorithm) and generate all candidate networks, namely the four networks shown in Figure 5.

We then evaluate the network unreliabilities to determine which network topology is optimal.

The main hybrid CE algorithm for estimating the reliability of a network is thus summarized as follows:

Algorithm 4 [Hybrid CE Method]**.**

1) *Initialize* $\widehat{\boldsymbol{a}}_0$. *Set* $t = 1$.
2) *Generate a random sample* $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_N$ *using Algorithm 1, with* $\boldsymbol{a} = \widehat{\boldsymbol{a}}_{t-1}$. *Compute the sample* $(1-\rho)$*-quantile of performances* $\widehat{\gamma}_t$ *using (6).*

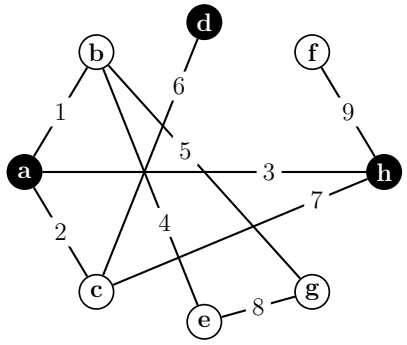


Fig. 6. System with three terminal nodes denoted by black nodes.

3) *Use the **same** sample to update $\widehat{\boldsymbol{a}}_t$, using (7).*
4) *If the number of purchase probabilities in the range $[\beta, 1-\beta]$ is less than or equal to a certain threshold, proceed to the next step (let T be the final iteration); otherwise set $t = t+1$ and reiterate from step 2.*
5) *Generate all candidate networks according to the purchase probability vector $\widehat{\boldsymbol{a}}_t$ and evaluate the network unreliabilities. Output the network with the smallest unreliability as a solution to the problem.*

B. Superfluous Links

We next discuss the role of *superfluous* links in a purchased network. These are links that do not affect the overall network reliability, whether they are functioning or not. As an example, consider the 8-node network with 3 terminal nodes in Figure 6.

In this network, there are five superfluous links in total, namely links 1, 4, 5, 8, and 9. Since these links do not affect the reliability of the network, the CE algorithm could, unnecessarily, have difficulty deciding whether or not to purchase them. In particular, the purchase probabilities of these links, $\widehat{a}_{t,i}$ with $i \in \{1,4,5,8,9\}$, could oscillate for a long time before they converge to either 0 or 1. This increases the computational effort significantly. Note that we only need to consider removing superfluous links when $\varphi_{\boldsymbol{x}}(\boldsymbol{x}) = 1$; that is, if all terminal nodes are connected given that all purchased links are indeed functioning.

Fortunately, the identification of superfluous links can be related to a well-understood problem in graph theory, namely that of identifying the *blocks* in a graph [31]. To illustrate the concepts, consider Figure 7. The open circles in the graph are the so-called *cutvertices*. When any of these is removed, the graph separates into disjoint components. A similar concept is that of a *bridge*: an edge whose deletion separates the graph. A subgraph is called a *block* if it is (a) a bridge including its endvertices, (b) a complete graph with 3 vertices, or (c) a graph with 4 or more vertices such that no single vertex separates it. Examples of each of the cases (a), (b) and (c) are given in the figure as B_1, B_2 and B_3, respectively.

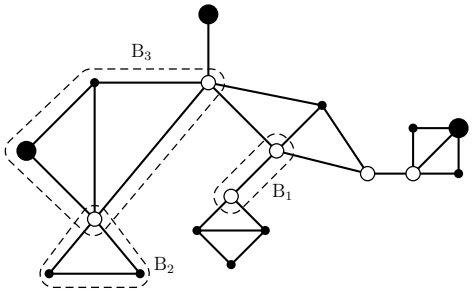


Fig. 7. A graph G with 8 blocks.

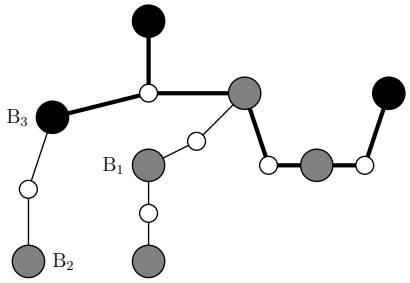


Fig. 8. The block-cutvertex tree of G.

The next step in the analysis is to form a *block-cutvertex tree* whose vertices are identified with the blocks and cutvertices of the graph, and whose edges join cutvertices to blocks. Figure 8 gives the block-cutvertex of the graph G of Figure 7. The large black and grey vertices correspond to blocks; the small white vertices correspond to the cutvertices. The three black vertices are the *terminal* blocks which need to be connected in order for the system to function.

The procedure for removing superfluous links now becomes clear: If a block is either (a) an isolated block (a block of

degree 0), (b) an end block which contains no terminal node, or (c) an end node which contains exactly one terminal node and its adjacent block contains the same terminal node, a block can be removed. We repeat this procedure until all superfluous blocks are removed from the block-cutvertex tree. There exist efficient algorithms for finding all the blocks in a graph, e.g., Algorithm 3.2 of [32].

The following algorithm can be applied to remove superfluous links from the graph.

Algorithm 5 [Superfluous Link Removal]**.**

1) *Find all isolated blocks and remove from the block-cutvertex tree.*
2) *Find all single degree blocks that do not contain any terminal node and remove from the block-cutvertex tree.*
3) *Let G_1 be an unmarked single degree block in the block-cutvertex tree that contains exactly one terminal node and proceed to step 4. Terminate if no such block exists.*
4) *Let G_2 be the block to which G_1 is connected. If G_2 contains the same terminal node as G_1, prune G_1; otherwise mark G_1 to keep in the block-cutvertex tree. Return to step 2.*

Note that a superfluous link in a network defined by $\boldsymbol{x}$ does not mean that the link is also superfluous in $\boldsymbol{x}'$. By removing a superfluous link with probability 1, the computational effort decreases, because the corresponding purchase probability no longer oscillates. However, like in the hybrid CE method, the accuracy could be negatively affected if the purchase probability converges too quickly to 0, which might lead to a suboptimal solution. To overcome this difficulty, one could choose to remove the link with probability delP, $0 \leq \text{delP} \leq 1$. In Section VI, we investigate at what probability superfluous link should be removed in order to obtain the maximum performance of the algorithms with minimum computational effort.

VI. Numerical Experiments

In this section, we present three test cases. For all test cases we compare the performance of the deterministic CE algorithm (CE-DET) and the hybrid CE algorithm (CE-HYBRID) with those of the noisy versions using CMC simulation (CE-CMC) and MP simulation (CE-MP). Thus, in the former case the system reliabilities for each purchase vector are estimated via Crude Monte Carlo simulation, and in the latter case via the Merge Process. In step 5 of the hybrid CE algorithm, we rank the candidate networks using either a deterministic approach (CE-HYBRID-DET) or MP simulation (CE-HYBRID-MP). All experiments were repeated 15 times in order to assess the variability and accuracy of the statistics.

Test Case 1: For the first experiment we return to Example 1, where the optimal purchase vector is given by $\boldsymbol{x}^* = (1,0,1,0,0,0,0,0,1,0,0,0,0,1,0)$, which gives a minimum network unreliability of $\bar{r}^* = 7.1967\text{e-}05$. The corresponding optimal network is depicted in Figure 1.

We take the same CE parameters as in Example 1 and set $K_{\min} = 1000$, $K_{\max} = 2000$, and $\delta = 4$. In CE-HYBRID-MP, 20000 samples were used to rank candidate solutions. Tables III and IV show typical evolutions of CE-CMC and CE-MP respectively. As was the case with the deterministic version (see Table II), both methods found the optimal purchase vector very quickly. The entry "0.000" in Table III means that the corresponding network unreliability was estimated (via CMC) as 0. Note that this can occur when the sample size K is relatively small.

Table V summarizes the statistics over 15 independent replications. Here "Method" represents the method used in simulation; "worst" is the worst network unreliability obtained over the 15 replications; "best" is the best unreliability obtained in simulation; "mean(T)" is the average number of iterations; and "how often" is the number of times the method obtained the optimal solution.

Each method performed exceptionally well, obtaining the

TABLE III

A TYPICAL EVOLUTION OF CE-CMC FOR TEST PROBLEM 1

t	$\widehat{\gamma}_t$	$\widehat{\boldsymbol{a}}_t$														
0		0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50
1	7.386e-03	0.44	0.47	0.45	0.31	0.15	0.36	0.22	0.21	0.50	0.22	0.22	0.43	0.26	0.50	0.32
2	7.062e-03	0.50	0.47	0.50	0.21	0.05	0.23	0.17	0.11	0.49	0.13	0.08	0.36	0.17	0.59	0.18
3	5.236e-03	0.62	0.46	0.54	0.20	0.01	0.13	0.18	0.06	0.60	0.08	0.02	0.35	0.08	0.61	0.08
4	4.000e-03	0.72	0.43	0.69	0.15	0.00	0.08	0.12	0.02	0.72	0.04	0.01	0.30	0.02	0.65	0.02
5	0.000	0.90	0.25	0.80	0.04	0.00	0.02	0.04	0.01	0.90	0.01	0.00	0.21	0.01	0.79	0.01
6	1.000e-03	0.96	0.14	0.88	0.01	0.00	0.01	0.01	0.00	0.96	0.00	0.00	0.13	0.00	0.88	0.00
7	0.000	0.99	0.04	0.97	0.00	0.00	0.00	0.00	0.00	0.99	0.00	0.00	0.04	0.00	0.96	0.00
8	0.000	1.00	0.01	0.99	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.01	0.00	0.99	0.00

TABLE IV

A TYPICAL EVOLUTION OF CE-MP FOR TEST PROBLEM 1

t	$\widehat{\gamma}_t$	$\widehat{\boldsymbol{a}}_t$														
0		0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50
1	8.593e-03	0.36	0.28	0.64	0.37	0.15	0.28	0.28	0.25	0.36	0.30	0.23	0.36	0.24	0.63	0.32
2	8.415e-03	0.39	0.28	0.69	0.22	0.05	0.09	0.21	0.11	0.40	0.19	0.14	0.25	0.13	0.70	0.25
3	8.330e-03	0.42	0.15	0.89	0.09	0.01	0.04	0.16	0.03	0.42	0.17	0.05	0.31	0.15	0.88	0.10
4	5.747e-03	0.61	0.12	0.92	0.06	0.00	0.01	0.28	0.01	0.60	0.08	0.02	0.14	0.10	0.94	0.07
5	6.968e-05	0.88	0.04	0.98	0.02	0.00	0.00	0.08	0.00	0.88	0.02	0.00	0.04	0.03	0.98	0.02
6	7.285e-05	0.97	0.01	0.99	0.01	0.00	0.00	0.03	0.00	0.96	0.01	0.00	0.01	0.01	0.99	0.01

TABLE V

NUMERICAL RESULTS FOR TEST CASE 1

Method	worst	best	mean(T)	how often
CE-DET	7.1967e-05	7.1967e-05	6.93	15
CE-CMC	8.0411e-01	7.1967e-05	6.87	14
CE-MP	7.1967e-05	7.1967e-05	5.80	15
CE-HYBRID-DET	8.0430e-05	7.1967e-05	7.40	13
CE-HYBRID-MP	8.0430e-05	7.1967e-05	6.93	14

optimal purchase vector in most of replications. This suggests that the proposed noisy CE algorithms using the CMC simulation and MP simulation and the hybrid CE algorithm can perform as effectively and reliably as the deterministic version. The s-relative error in the reliability estimation (for $K = 1500$) was around 30 to 70% for CMC and 2% for CE-MP. Thus even with this amount of uncertainty all algorithms can effectively locate the optimal network topology for this problem.

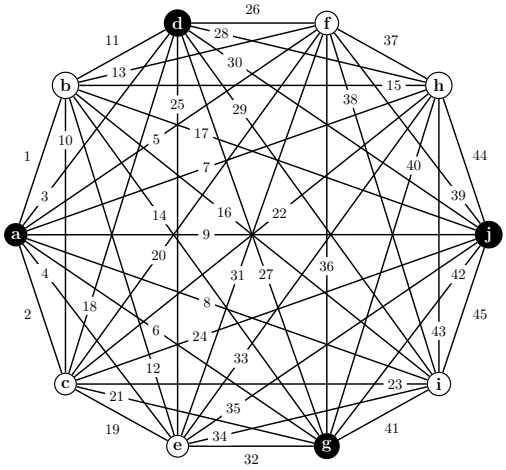


Fig. 9. Network topology for test case 2.

Test Case 2: Test case 2 is concerned with the 10-node fully connected network with 4 terminal nodes depicted in Figure 9. This test case is a lot harder than the previous one, not only because there are 2^{45} candidate networks, but also because many of these candidate networks have unreliabilities that are very close to the best found solution (less than 10^{-10} away).

TABLE VI

THE PSEUDO CODE FOR GENERATING THE COST AND PROBABILITY VECTORS FOR TEST CASE 2

$i \leftarrow 0,\ j \leftarrow 1$

while $j \leq m$

do
- $i \leftarrow i + 1$
- $b \leftarrow 1 - 7654321 \pmod{987 + i}/10^5$
- **if** $b > 0.99$ **and** $b \neq p_k,$ for all $k = 1, \ldots, j-1$
  - **then**
    - $p_j \leftarrow b$
    - $c_j \leftarrow 20/\exp(8\sqrt{1-b})$
    - $j \leftarrow j + 1$

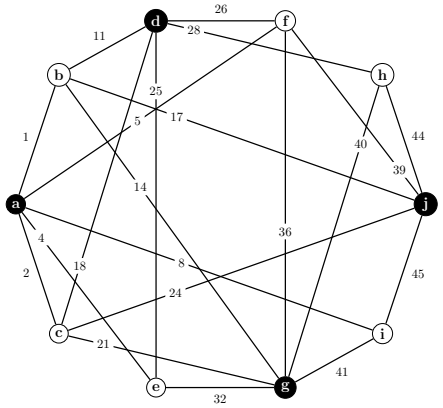


Fig. 10. The best found network for test case 2.

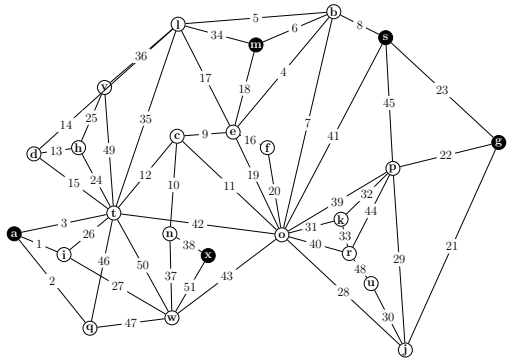


Fig. 11. Network with 24 nodes and 51 links.

TABLE VIII

PSEUDO CODE FOR GENERATING THE LINK COST AND RELIABILITIES FOR CASE 3

$i \leftarrow 0,\ j \leftarrow 1$

while $j \leq m$

do
- $i \leftarrow i + 1$
- $b \leftarrow 1 - 765432 \pmod{9876 + i}/10^5$
- **if** $b > 0.99$ **and** $b \neq p_k,$ for all $k = 1, \ldots, j-1$
  - **then**
    - $p_j \leftarrow b$
    - $c_j \leftarrow 20/\exp(8\sqrt{1-b})$
    - $j \leftarrow j + 1$

The total budget is $C_{\max} = 250$, and we take the following CE parameters: $N = 2250,\ K_{\min} = 3000,\ K_{\max} = 10000,\ \rho = 0.05,\ \alpha = 0.7,\ \beta = 0.05$, and $\delta = 4$. In CE-HYBRID-MP, 50000 samples were used to rank candidate solutions. Table VI gives the pseudo code for generating the link costs and reliabilities.

Based on 15 independent trials, the best found solution for test case 2 is given in Figure 10, with a network unreliability of 8.7396e-13.

The results of 15 replications for each method are summarized in Table VII, with the exception of CE-CMC, which failed to produce a solution — the reason is that CMC simulation required too large a computational effort (that is, a large sample size) to accurately estimate the very small network unreliability; cf. (14).

It is encouraging that CE-HYBRID-DET obtained the best found network in all trials. CE-MP and CE-HYBRID-MP performed reliably, obtaining 12 and 11 times respectively. We found that CE-DET did not perform as well as other methods. One reason is that the speed of convergence was too fast. Experiments showed that by choosing smaller α, say $\alpha = 0.5$, CE-DET performed as well as its noisy counterpart.

Test Case 3: The network topology for test case 3, taken from the survivable fixed telecommunication network design library website `http://sndlib.zib.de/`, comprises 24 nodes, 51 links and 5 terminal nodes, see Figure 11.

In this problem, we investigate how superfluous link removal (Algorithm 5) affects the overall performance of the methods, using different values for delP. For this test case, we use the following CE parameters: $N = 3000,\ K_{\min} = 1000,\ K_{\max} = 10000,\ \rho = 0.05,\ \alpha = 0.7,\ \beta = 0.05$, and $\delta = 4$. The total budget $C_{\max} = 5000$. In CE-HYBRID-MP, 10^5 samples were used to rank candidate solutions. The pseudo code for generating $\boldsymbol{c}$ and $\boldsymbol{p}$ is given below.

The best found solution for test case 3 is given in Figure 12,

TABLE VII

NUMERICAL RESULTS FOR TEST CASE 2

Method	worst	best	mean(T)	how often
CE-DET	1.31706557e-12	8.73960309e-13	16.14	7
CE-MP	1.18106167e-12	8.73960309e-13	18.33	12
CE-HYBRID-DET	8.73960310e-13	8.73960310e-13	14.93	15
CE-HYBRID-MP	1.16903365e-12	8.73960309e-13	14.36	11

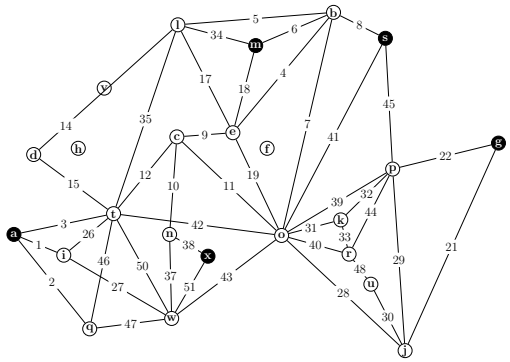


Fig. 12. The best found network for test case 3.

with a network unreliability of 1.58635e-05.

Table IX shows the numerical results for test case 3, averaged over 15 independent replications for different values of delP. Here "how often" is the number of times the method obtained the best found solution or near-best solutions, that is, solutions whose network unreliabilities are less than 1% away from the best found solution. "CPU" denotes the average simulation time in seconds.

We see that applying Algorithm 5 achieves significant reduction in computational time. However for values of delP ≥ 0.8, all four methods failed to obtain the best found or near-best solutions. The reason is that some of the links in the network may have been considered as superfluous links in early iterations. This caused the purchase probabilities to converge to 0 very quickly. This can be avoided by taking smaller delP. It is interesting to note that CE-MP performed worse with superfluous link removal than without. A likely reason is that the s-relative error in the network reliability estimator was quite large, so that the networks could not be accurately ranked. However, to decrease the relative error to about 0.01 a sample size of at least $K = 50000$ would be required, which would significantly increase the computational effort.

In contrast to CE-MP, the other three methods performed as well with $0.2 \leq$ delP ≤ 0.5 as with delP = 0, obtaining the best found or near-best solutions around the same number of times. This shows that by removing superfluous links with probability $0.2 \leq$ delP ≤ 0.5, we were able to reduce the computational time up to 75% without affecting the accuracy of the methods.

Moreover, both hybrid CE algorithms, with or without superfluous link removal, significantly reduced the computation time, without affecting significantly the accuracy.

VII. CONCLUSIONS AND FUTURE RESEARCH

In this paper, we proposed a CE approach for solving the Network Planning Problem, which is a difficult constrained combinatorial optimization problem with a noisy objective function. This noise is introduced when the objective function (the system reliability) is estimated rather than evaluated exactly. We reviewed several techniques for reducing this noise. We explained how the deterministic version of the CE algorithm (when the objective function is known) could be readily modified to handle the noisy situation. The numerical results showed that the noisy CE algorithm can perform as effectively and reliably as its deterministic counterpart. We proposed two techniques to speed up the convergence of the algorithm: probabilistic superfluous links removal and hybrid CE. Experiments showed that with an appropriate choice of parameters these techniques can speed up the optimization

TABLE IX

NUMERICAL RESULTS FOR TEST CASE 3 WITH DIFFERENT VALUES FOR DELP

delP	Method	worst	best	mean(T)	how often	CPU
0.00	CE-DET	1.58635e-05	1.58635e-05	14.87	15	21680
	CE-MP	1.61361e-05	1.58644e-05	165.33	11	13086
	CE-HYBRID-DET	1.61025e-05	1.58635e-05	31.13	13	3062
	CE-HYBRID-MP	1.61023e-05	1.58635e-05	28.47	8	2829
0.20	CE-DET	1.58635e-05	1.58635e-05	15.93	15	14254
	CE-MP	1.61337e-05	1.58664e-05	159.27	2	9579
	CE-HYBRID-DET	1.61021e-05	1.58635e-05	21.80	14	4113
	CE-HYBRID-MP	1.61024e-05	1.58635e-05	29.33	9	2868
0.40	CE-DET	1.58635e-05	1.58635e-05	16.27	15	8206
	CE-MP	1.61327e-05	1.58669e-05	123.47	1	6602
	CE-HYBRID-DET	1.61021e-05	1.58635e-05	21.00	12	1976
	CE-HYBRID-MP	1.61025e-05	1.58635e-05	20.07	11	1886
0.50	CE-DET	1.58635e-05	1.58635e-05	17.47	15	5176
	CE-MP	1.61335e-05	1.61278e-05	103.33	0	5505
	CE-HYBRID-DET	1.61021e-05	1.58635e-05	19.47	10	1730
	CE-HYBRID-MP	1.61025e-05	1.58635e-05	17.33	8	1527
0.60	CE-DET	1.61024e-05	1.58635e-05	16.40	12	1817
	CE-MP	1.61371e-05	1.61272e-05	111.93	0	4759
	CE-HYBRID-DET	1.61025e-05	1.58635e-05	16.60	9	1236
	CE-HYBRID-MP	1.61272e-05	1.58635e-05	14.60	7	1106
0.80	CE-DET	1.61272e-05	1.61024e-05	10.27	0	360
	CE-MP	1.61385e-05	1.61272e-05	87.07	0	3006
	CE-HYBRID-DET	1.61272e-05	1.61021e-05	9.53	0	430
	CE-HYBRID-MP	1.61291e-05	1.61023e-05	9.80	0	515
1.00	CE-DET	1.61272e-05	1.61272e-05	9.00	0	299
	CE-MP	1.61330e-05	1.61272e-05	73.87	0	2418
	CE-HYBRID-DET	1.61272e-05	1.61272e-05	8.00	0	349
	CE-HYBRID-MP	1.61280e-05	1.61272e-05	7.87	0	360

process by over 300% with very little impact on the accuracy of the algorithm.

We have taken a pragmatic approach regarding convergence, observing that numerically the CE algorithms tends to converge in a global rather than a local sense. Although several CE convergence results are discussed in [33], [34], this topic remains an interesting challenge for future research. Another possible direction is to investigate whether *reliability ranking* [35] can further improve the performance of the algorithms. Here only the relative rankings of the networks are important, rather than the exact values of their reliabilities. In some cases the former are easier to estimate than the latter. This could lead to an improvement of the performance of the CE algorithm. A further improvement could be achieved by examining at what stage in the simulation one should start using superfluous link removal, and what value of delP should be taken. Finally, developing an efficient rare event paradigm that can reduce the noise in the network reliability estimation with relatively low computational effort is another important area for future research.

REFERENCES

[1] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. San Francisco: Freeman, 1979.

[2] M. Bern and P. Plassman, "The steiner problem with edge lengths 1 and 2," *Information Processing Letters*, vol. 32, pp. 171–176, 1989.

[3] D. Li, X. L. Sun, and K. McKinnon, "An exact solution method for reliability optimization in complex systems," *Annals of Operations Research*, vol. 133, pp. 129 – 144, 2005.

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

[5] 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*, vol. 12, pp. 777–787, 1982.

[6] H. Kumamoto, K. Tanaka, K. Inoue, and E. J. Henley, "Dagger sampling Monte Carlo for system unavailability evaluation." *IEEE Transactions on Reliability*, vol. R-29, no. 2, pp. 376–380, 1980.

[7] G. Fishman, "A Monte Carlo sampling plan for estimating network reliability," *Operations Research*, vol. 34, no. 4, pp. 581–594, Jul-Aug 1986.

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

[9] T. Elperin, I. B. Gertsbakh, and M. Lomonosov, "Estimation of network reliability using graph evolution models," *IEEE Transactions on Reliability*, vol. 40, no. 5, pp. 572–581, Dec 1991.

[10] ——, "An evolution model for Monte Carlo estimation of equilibrium network renewal parameters," *Probability in the Engineering and Informational Sciences*, vol. 6, pp. 457–469, 1992.

[11] K.-P. Hui, N. Bean, M. Kraetzl, and D. P. Kroese, "The tree cut and merge algorithm for estimation of network reliability," *Probability in the Engineering and Informational Sciences*, vol. 17, no. 1, pp. 24–45, 2003.

[12] ——, "Network reliability estimation using the tree cut and merge algorithm with importance sampling," *Proceedings. Fourth International Workshop on Design of Reliable Communication Networks*, pp. 254–262, 2003.

[13] ——, "The cross-entropy method for network reliability estimation," *Annals of Operations Research*, vol. 134, no. 1, pp. 101–118, 2005.

[14] W.-C. Yeh, "A new Monte Carlo method for the network reliability," *Proceedings of First International Conference on Information Technologies and Applications(ICITA2002)*, November 2002.

[15] C. Srivaree-ratana and A. E. Smith, "Estimating all-terminal network reliability using a neural network," *Proceedings of the 1998 IEEE International Conference on Systems, Man, and Cybernetics*, vol. 5, pp. 4734–4740, October 1998.

[16] I. B. Gertsbakh, *Statistical Reliability Theory*. New York: Marcel Dekker, Inc., 1989.

[17] H. Cancela and M. E. Urquhart, "Simulated annealing for communication network reliability improvements," in *Proceedings of the XXI Latin American Conference On Informatics*. CLEI-SBC, July 1995.

[18] B. Dengiz, F. Altiparmak, and A. E. Smith, "Local search genetic algorithm for optimal design of reliable networks," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 3, pp. 179–188, September 1997.

[19] Y.-S. Yeh, C. C. Chiu, and R.-S. Chen, "A genetic algorithm for k-node set reliability optimization with capacity constraint of a distributed system," *Proc. Natl. Sci. Counc. ROC(A)*, vol. 25, no. 1, pp. 27–34, 2001.

[20] D. Reichelt, F. Rothlauf, and P. Bmilkowsky, "Designing reliable communication networks with a genetic algorithm using a repair heuristic," in *Evolutionary Computation in Combinatorial Optimization*. Springer-Verlag Heidelberg, 2004.

[21] W. Kuo and V. Prasad, "An annotated overview of system–reliability optimization," *IEEE Trans. Reliability*, pp. 176–187, 2000.

[22] R. Y. Rubinstein and D. P. Kroese, *The Cross-Entropy Method: A unified approach to Combinatorial Optimization, Monte Carlo Simulation and Machine Learning*. New York: Springer Verlag, 2004.

[23] R. Y. Rubinstein, "Optimization of computer simulation models with rare events," *European Journal of Operations Research*, vol. 99, pp. 89–112, 1997.

[24] ——, "The simulated entropy method for combinatorial and continuous optimization," *Methodology and Computing in Applied Probability*, vol. 2, pp. 127–190, 1999.

[25] ——, "Combinatorial optimisation, cross-entropy, ants and rare events," in *Stochastic Optimization: Algorithms and Applications*, S. Uryasev and P. M. Pardalos, Eds., Kluwer, 2001, pp. 304–358.

[26] G. Alon, D. P. Kroese, T. Raviv, and R. Y. Rubinstein, "Application of the buffer allocation problem in simulation-based environment," *Annals of Operations Research*, vol. 134, no. 1, pp. 137 – 151, 2005.

[27] K. Chepuri and T. Homem de Mello, "Solving the vehicle routing problem with stochastic demands using the cross-entropy method," *Annals of Operations Research*, vol. 134, no. 1, pp. 153 – 181, 2005.

[28] D. P. Bertsekas and J. N. Tsitsiklis, "An analysis of stochastic shortest path problems," *Mathematics of Operations Research*, vol. 16, pp. 580–595, 1991.

[29] P. T. de Boer, D. P. Kroese, S. Mannor, and R. Y. Rubinstein, "A tutorial on the cross-entropy method," *Annals of Operations Research*, vol. 134, no. 1, pp. 19 – 67, 2005.

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

[31] B. Bollobás, *Graph Theory: An Introductory Course*. Springer-Verlag, 1979.

[32] G. Chartrand and O. R. Oellermann, *Applied and Algorithmic Graph Theory*. McGraw-Hill, 1993.

[33] L. Margolin, “On the convergence of the cross-entropy method,” *Annals of Operations Research*, vol. 134, no. 1, pp. 201 – 214, 2005.

[34] A. Costa, O. Jones, and D. P. Kroese, “Convergence properties of the cross-entropy method for discrete optimization,” *Operations Research Letters*, 2007, to appear.

[35] K.-P. Hui, “Monte Carlo network reliability ranking estimation,” *IEEE Transactions on Reliability*, April 2007.

Dirk P. Kroese has a wide range of publications in applied probability and simulation. He is a pioneer of the well-known *Cross-Entropy* method and co-author (with R.Y. Rubinstein) of the first monograph on this method. He is associate editor of *Methodology and Computing in Applied Probability* and guest editor of *Annals of Operations Research*. He has held research and teaching positions at Princeton University and the University of Melbourne, and is currently working at the Department of Mathematics of the University of Queensland.

Kin-Ping Hui is currently a research engineer at the Defence Science and Technology Organization in Australia. His research interests include network reliability estimation, network optimization, network survivability, and network recovery.

Sho Nariai is a PhD student in the Department of Mathematics at the University of Queensland. His research interests include applications of the Cross-Entropy method to network design problems, continuous optimization, and noisy optimization.