Designing an Optimal Network Using the Cross-Entropy Method

Sho Nariai[1], Kin-Ping Hui[2], and Dirk P. Kroese[1]

[1] Department of Mathematics, University of Queensland, Brisbane 4072, Australia
[2] IN Division, DSTO, Edinburgh 5111, Australia

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 bought in order to maximize the system's reliability? We introduce a Cross-Entropy approach to this problem, which can deal effectively with the noise and constraints in this difficult combinatorial optimization problem. Numerical results demonstrate the effectiveness of the proposed technique.

1 Introduction

One of the most basic and useful approaches to network reliability analysis is to represent the network as an undirected graph with unreliable links. Often, the reliability of the network is 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, the question is which links should be purchased, in order to maximize the reliability of the purchased network. Each link has a prespecified price and reliability. This Network Planning Problem (NPP) is difficult to solve, not only because it is a constrained integer programming problem, which complexity grows exponentially in the number of links, but also because for large networks the value of the objective function – that is, the network reliability – becomes difficult or impractical to evaluate [1, 2].

We show that the *Cross-Entropy* (CE) method provides an effective way to solve the NPP. The CE method is a new method for discrete and continuous optimization. It consists of two steps which are iterated:

1. generate random states in the search space according to some specified random mechanism, and
2. update the parameters of this mechanism in order to obtain better scoring states in the next iteration. This last step involves minimizing the distance between two distributions, using the Kullback-Leibler or Cross-Entropy distance; hence the name.

A tutorial introduction can be found in [3], which is also available from the CE homepage `http://www.cemethod.org`. A comprehensive treatment can be found in [4].

The rest of the paper is organized as follows. In Section 2 we formulate the network planning in mathematical terms. In Section 3 we present the CE approach to the problem. In Section 4 we consider a *noisy* version of the CE method [5, 6], where the network reliability is estimated (rather than evaluated) using *graph evolution* techniques [7–9]. We conclude with a numerical experiment in Section 5 that illustrates the effectiveness of our approach.

2 Problem Description

Consider a network represented as 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 $1,\dots,m$. 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 buy those links that optimize the reliability of the network – defined as the probability that the terminal nodes are connected by functioning links – subject to a total budget $C_{\max}$. Let $\boldsymbol{c} = (c_1,\dots,c_m)$ denote vector of link costs, and $\boldsymbol{p} = (p_1,\dots,p_m)$ the vector of link reliabilities.

We introduce the following notation. For each link e let x_e be such that $x_e = 1$ if link e is purchased, and 0 otherwise. We call the vector $\boldsymbol{x} = (x_1,\dots,x_m)$ the *purchase vector* and $\boldsymbol{x}^*$ the *optimal purchase vector*. Similarly, to identify the operational links, we define for each link e the link *state* by $y_e = 1$ if link e is bought and is functioning, and 0 otherwise. The vector $\boldsymbol{y} = (y_1,\dots,y_m)$ 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 = terminal nodes are connected = 1, or failed = 0). Next, 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. Note that for each link e that is *not* bought, the state Y_e is per definition equal to 0. The reliability of the network determined by $\boldsymbol{x}$ is given by

$$r(\boldsymbol{x}) = \mathbb{E}[\,\varphi_{\boldsymbol{x}}(\boldsymbol{Y})] = \sum_{\boldsymbol{y}} \varphi_{\boldsymbol{x}}(\boldsymbol{y})\,\Pr\{\boldsymbol{Y} = \boldsymbol{y}\}\ . \tag{1}$$

We assume from now on that the links fail independently, that is, $\boldsymbol{Y}$ is a vector of independent Bernoulli random variables, with success probability p_e for each purchased link e and 0 otherwise. Defining $\boldsymbol{p}_{\boldsymbol{x}} = (x_1 p_1,\dots,x_m p_m)$, we write $\boldsymbol{Y} \sim \mathsf{Ber}(\boldsymbol{p}_{\boldsymbol{x}})$. Our main purpose is to determine

$$\max_{\boldsymbol{x}} r(\boldsymbol{x})\ , \quad \text{subject to} \quad \sum_{e\in\mathcal{E}} x_e c_e \le C_{\max}\ . \tag{2}$$

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

3 The CE Method

In order to apply the CE method to the optimization problem (2), we need to specify (a) a random mechanism to generate random purchase vectors that satisfy the constraints, and (b) the updating rule for the parameters in that random mechanism.

A simple and efficient method to generate the random purchase vectors is as follows: First, generate a "uniform" random permutation $\pi = (e_1, e_2, \ldots, e_m)$ of edges. Then, in the order of the permutation π, flip a coin with success probability a_{e_i} to decide whether to purchase link e_i. If successful and if there is enough money available to purchase link e_i, set $X_{e_i} = 1$, that is, link e_i is purchased; otherwise set $X_{e_i} = 0$. The algorithm is summarized next.

Algorithm 1 (Generation Algorithm)

```
1. Generate a uniform random permutation $\pi = (e_1, \ldots, e_m)$. 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 [4] proceeds by constructing 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 $\boldsymbol{a}^* = \boldsymbol{x}^*$. 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 tend 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 $(1-\rho)$-sample 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 $\widehat{\gamma}_t = r_{(\lceil (1-\rho)N \rceil)}$, 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 [4]) reduces to the following: For a given fixed $\boldsymbol{a}_{t-1}$ and γ_t, let the j-th component of $\boldsymbol{a}_t$ be $a_{t,j} = \mathbb{E}_{\boldsymbol{a}_{t-1}}[X_j \mid r(\boldsymbol{X}) \geq \gamma_t]$. An estimator $\widehat{\boldsymbol{a}}_t$ of $\boldsymbol{a}_t$ is computed via

$$\widehat{a}_{t,j} = \frac{\sum_{i=1}^{N} I_{\{r(\boldsymbol{X}_i) \geq \widehat{\gamma}_t\}} X_{ij}}{\sum_{i=1}^{N} I_{\{r(\boldsymbol{X}_i) \geq \widehat{\gamma}_t\}}}, \quad j = 1, \ldots, m, \tag{3}$$

where we use the *same* random sample $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_N$ and where X_{ij} is the j-th coordinate of $\boldsymbol{X}_i$.

The main CE algorithm for optimizing (2) 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 $(1-\rho)$-sample quantile of performances $\widehat{\gamma}_t$.
3. Use the same sample to update $\widehat{\boldsymbol{a}}_t$, using (3).
4. If some stopping criterion is met then stop; otherwise set $t = t + 1$ and
   reiterate from step 2.
```

4 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 this section we show how the CE method can be easily modified to tackle *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 by means of Monte Carlo simulations. An efficient approach to network reliability estimation is to use *Network Evolution* [7]. This works also well for highly reliable networks. The idea is as follows: Consider a network with structure function φ and reliability r as defined in (1). Assume for simplicity that all the links are bought, that is $\boldsymbol{x} = (1, 1, \ldots, 1)$. Now, observe a *dynamic* version of the network $\mathcal{G}(V, E)$ which starts with all links failed and in which all links are being independently repaired; each link e has an exponential repair time with repair rate $\lambda(e) = -\log(1 - p_e)$. The state of e at time t is denoted by $Y_e(t)$ and, similar to before, the states of all the links is given by the vector $\boldsymbol{Y}(t)$. Then, $(\boldsymbol{Y}(t))$ is a Markov process with state space $\{0, 1\}^m$. Let $\mathit{\Pi}$ denote the *order* in which the links become operational. Note that the probability of link 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 (1). Hence, by conditioning on $\mathit{\Pi}$ we have

$$r = \mathbb{E}[\varphi(\boldsymbol{Y}(1))] = \sum_{\pi} \Pr\{\mathit{\Pi} = \pi\} \Pr\{\varphi(\boldsymbol{Y}(1)) = 1 \,|\, \mathit{\Pi} = \pi\}, \tag{4}$$

The crucial point is that from the theory of Markov processes it is possible to *calculate* the probability $G(\pi) = \Pr\{\varphi(\boldsymbol{Y}(t)) = 0 \,|\, \mathit{\Pi} = \pi\}$ in terms of convolutions of exponential distribution functions. Hence, we can estimate r by first drawing a random sample $\mathit{\Pi}_1, \ldots, \mathit{\Pi}_N$, each distributed according to $\mathit{\Pi}$, and then estimating r as

$$\widehat{r} = \frac{1}{K} \sum_{i=1}^{K} G(\mathit{\Pi}_i) \; . \tag{5}$$

5 Numerical Experiment

To illustrate the effectiveness of the proposed CE approach, consider the 6-node fully-connected graph with 3 terminal nodes given in Figure 1. The links costs and reliabilities are given in Table 1. Note that the direct links between the terminal nodes have infinite costs. We have deliberately excluded such links to make the problem more difficult to solve. The total budget is set to $C_{\max} = 3000$.

Note that for a typical purchase vector $\boldsymbol{x}$ the network reliability $r(\boldsymbol{x})$ will be high, since all links are quite reliable. Consequently, to obtain an accurate estimate of the network reliability, or better, the network unreliability $\bar{r}(\boldsymbol{x}) = 1 - r(\boldsymbol{x})$, via conventional Monte Carlo methods, would require a large simulation effort. The optimal purchase vector for this problem – which was computed by

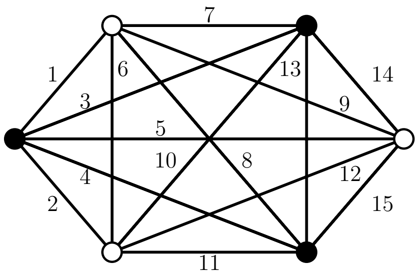

Fig. 1. Network with 3 terminal nodes, denoted by black vertices

Table 1. Link costs and reliabilities

i	c_i	p_i	i	c_i	p_i	i	c_i	p_i
1	382	0.990	6	380	0.998	11	397	0.990
2	392	0.991	7	390	0.997	12	380	0.991
3	∞	0.992	8	395	0.996	13	∞	0.993
4	∞	0.993	9	396	0.995	14	399	0.992
5	320	0.994	10	381	0.999	15	392	0.994

brute force – is equal to $\boldsymbol{x}^* = (1,1,0,0,1,0,1,1,0,1,0,0,0,0,1)$ which yields a minimum network unreliability of $\overline{r}^* = 7.9762 \times 10^{-5}$.

We used the following parameters for our algorithm: the sample size in Step 2 of the CE algorithm $N = 300$; the sample size in (5) $K = 100$; the initial purchase probability $\widehat{\boldsymbol{a}}_0 = (0.5, \ldots, 0.5)$; the rarity parameter $\rho = 0.1$. The algorithm stops when $\max(\min(\widehat{\boldsymbol{a}}_t, 1 - \widehat{\boldsymbol{a}}_t)) \leq \beta = 0.02$, that is, when all elements of $\widehat{\boldsymbol{a}}_t$ are less than β, away from either 0 or 1. Let T denote the final iteration counter. We round $\widehat{\boldsymbol{a}}_T$ to the nearest binary vector and take this as our solution $\widehat{\boldsymbol{a}}^*$ to the problem. As a final step we estimate the optimal system reliability via (5) using a larger sample size of $K = 1000$.

Table 2 displays a typical evolution of the CE method. Here, t denotes the iteration counter, $\hat{\gamma}_t$ the $1-\rho$ quantile of the estimated unreliabilities, and $\widehat{\boldsymbol{a}}_t$ the

Table 2. A typical evolution of the CE algorithm with $N = 300$, $K = 100$ $\rho = 0.1$, and $\beta = 0.02$

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	4.0e-03	0.66	0.69	0.15	0.15	0.62	0.48	0.59	0.64	0.38	0.62	0.52	0.38	0.15	0.41	0.62
2	2.6e-04	0.69	0.63	0.05	0.05	0.72	0.21	0.88	0.71	0.33	0.75	0.58	0.26	0.05	0.38	0.77
3	1.4e-04	0.67	0.75	0.01	0.01	0.78	0.11	0.89	0.89	0.12	0.76	0.57	0.22	0.01	0.44	0.77
4	1.0e-04	0.76	0.76	0.00	0.00	0.89	0.03	0.97	0.90	0.06	0.83	0.43	0.11	0.00	0.41	0.84
5	8.1e-05	0.79	0.88	0.00	0.00	0.97	0.01	0.99	0.97	0.02	0.90	0.15	0.03	0.00	0.33	0.95
6	6.7e-05	0.94	0.96	0.00	0.00	0.97	0.00	1.00	0.99	0.01	0.97	0.07	0.01	0.00	0.10	0.99
7	6.3e-05	0.98	0.99	0.00	0.00	0.99	0.00	1.00	1.00	0.00	0.99	0.02	0.00	0.00	0.03	1.00
8	5.8e-05	0.99	1.00	0.00	0.00	1.00	0.00	1.00	1.00	0.00	1.00	0.01	0.00	0.00	0.01	1.00

purchase probability vector, at iteration t. The important thing to notice is that $\widehat{\boldsymbol{a}}_t$ quickly converges to the optimal degenerate vector $\boldsymbol{a}^* = \boldsymbol{x}^*$. The estimated network unreliability was found to be 8.496×10^{-5} with relative error of 0.0682. The simulation time was 154 seconds on a 3.0GHz computer using a Matlab implementation.

In repeated experiments, the proposed CE algorithm performed effectively and reliably in solving the noisy NPP, which constantly obtained the optimal purchase vector. Moreover, the algorithm only required on average 9 iterations with a CPU time of 180 seconds.

Acknowledgement

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

References

1. Colbourn, C.J.: The Combinatorics of Network Reliability. Oxford University Press (1987)
2. Provan, J.S., Ball, M.O.: The complexity of counting cuts and of computing the probability that a graph is connected. SIAM Journal of Computing **12** (1982) 777–787
3. de Boer, P.T., Kroese, D.P., Mannor, S., Rubinstein, R.Y.: A tutorial on the cross-entropy method. Annals of Operations Research **134** (2005) 19 – 67
4. Rubinstein, R.Y., Kroese, D.P.: The Cross-Entropy Method: A unified approach to Combinatorial Optimization, Monte Carlo Simulation and Machine Learning. Springer Verlag, New York (2004)
5. Alon, G., Kroese, D.P., Raviv, T., Rubinstein, R.Y.: Application of the buffer allocation problem in simulation-based environment. Annals of Operations Research **134** (2005) 137 – 151
6. Chepuri, K., Homem de Mello, T.: Solving the vehicle routing problem with stochastic demands using the cross-entropy method. Annals of Operations Research **134** (2005) 153 – 181
7. Elperin, T., Gertsbakh, I.B., Lomonosov, M.: Estimation of network reliability using graph evolution models. IEEE Transactions on Reliability **40** (1991) 572–581
8. Hui, K.P., Bean, N., Kraetzl, M., Kroese, D.P.: The tree cut and merge algorithm for estimation of network reliability. Probability in the Engineering and Informational Sciences **17** (2003) 24–45
9. Hui, K.P., Bean, N., Kraetzl, M., Kroese, D.P.: Network reliability estimation using the tree cut and merge algorithm with importance sampling. Proceedings. Fourth International Workshop on Design of Reliable Communication Networks (2003) 254–262