Monte Carlo Splitting Technique for source-terminal Network Reliability Estimation

Leslie Murray[1] and Héctor Cancela[2]

[1] Facultad de Ciencias Exactas, Ingeniería y Agrimensura,
Universidad Nacional de Rosario
Rosario, Argentina
[2] Facultad de Ingeniería, Universidad de la República
Montevideo, Uruguay

Abstract. In this paper we present an adaptation of the Splitting technique (a method used for variance reduction when estimating by Monte Carlo the probability of rare events in Markovian settings) to estimate a topological reliability measure, namely the source-terminal network reliability. The experimental results show that the method has extremely robust performances with respect to the parameters of the network, attaining a few percent relative error with a fixed number of iterations even when the network reliability values change many orders of magnitude; this behavior is not usually obtained with existing methods, and shows the interest of the method for evaluating highly reliable networks.

1 Introduction

Network models are an important tool to represent a large variety of situations and problems arising in real life, covering from transportation infrastructure up to personal interactions in social organization. Among the huge variety of network models that appear in the literature, source-terminal and multi-terminal network reliability models have been widely discussed and employed in applications arising from traditional computer and telecommunication networks, as well as from electrical networks, transportation, space, military applications, general system reliability, etc. [2, 4, 9, 11, 13, 15].

The exact computation of the source-terminal reliability measure is known to be NP-hard [12] (actually #P-hard, as it corresponds to a counting problem), so that no algorithm able to compute this measure exactly in polynomial time is expected to be found. An alternative for evaluation is Monte Carlo simulation. Many works have studied different ways to reduce by the largest possible amount the variance of these simulations (as this variance corresponds to the precision of the method). In general, the precision of the methods is relatively invariant with respect to the number of components of the evaluated network (and the computational complexity grows in most cases linearly on the number of components of the network); but the relative errors grow when the component's reliabilities, and the overall network reliability, go to zero.

XXXIV Conferencia Latinoamericana de Informática (CLEI 2008), pp. 990-999, 2008.

In this paper we look at Splitting, a variance reduction technique which has been proposed and used in another context. Splitting is successfully adapted to the reliability network estimation and its performance is subject to empirical evaluation over three benchmark network topologies. The results show a very stable precision within a relatively small number of Monte Carlo iterations.

The rest of the paper is organized as follows. Section 2 gives a brief review of Splitting, Section 3 introduces network models and Section 4 presents a Splitting application to solve one of the network models introduced in Section 3, implementation details and experimental results are given in Sections 5 and 6 respectively, while conclusions and future work are assessed in Section 7.

2 Splitting – General Setting

Given a Markov process $X(t)$, $t \geq 0$, with state space $\mathcal{X}$, and given two disjoint regions $\mathcal{X}_A$ y $\mathcal{X}_B$ of $\mathcal{X}$, it may be of interest to to find the probability that, evolving from $t = 0$, $X(t)$ comes into $\mathcal{X}_B$ without having entered $\mathcal{X}_A$ before. Calling τ_A the time at which $X(t)$ enters $\mathcal{X}_A$ for the first time (or comes back to $\mathcal{X}_A$ if $X(0) \in \mathcal{X}_A$) and τ_B the time at which $X(t)$ enters $\mathcal{X}_B$ for the first time, the problem is to find the probability of the event $[\tau_B < \tau_A]$. Regions $\mathcal{X}_A$ and $\mathcal{X}_B$ can be expressed in terms of an *importance function* $h : \mathcal{X} \to \mathbb{R}$ as: $A = \{x \in \mathcal{X} : h(x) \leq 0\}$ and $B = \{x \in \mathcal{X} : h(x) \geq \ell\}$ for $\ell \in \mathbb{R}, \ell > 0$.

Splitting [8, 14, 5] makes efficient estimations of the probability of $[\tau_B < \tau_A]$, particularly when $[\tau_B < \tau_A]$ is a rare event. To do this, the state space bounded by thresholds $\{x : h(x) = 0\}$ and $\{x : h(x) = \ell\}$ is divided into m sub–spaces bounded by thresholds $\{x : h(x) = \ell_k\}$, $k = 1, 2, \cdots, m$, such that: $\ell_0 = 0 < \ell_1 < \ell_2 < \cdots < \ell_m = \ell$. Region A is the subspace of the states under threshold $\ell_0 = 0$ and region B the one above threshold $\ell_m = \ell$.

Calling T_k the time at which the process $h(X(t))$ up–crosses ℓ_k for the first time, it is possible to define events $D_k = [T_k < \tau_A]$, $k = 1, 2, \cdots, m$, as the indicator events that process $h(X(t))$ has crossed threshold ℓ_k without having entered the region under threshold $\ell_0 = 0$. As a consequence a set of nested events is defined: $D_m \subset D_{m-1} \subset \cdots \subset D_2 \subset D_1$, being $D_m = [\tau_B < \tau_A]$ the event whose probability γ_m is to be estimated. Then, γ_m may be expressed as:

$$\begin{aligned}\gamma_m &= \mathbb{P}[D_m] \\ &= \underbrace{\mathbb{P}[D_m|D_{m-1}]}_{p_m}\ \underbrace{\mathbb{P}[D_{m-1}|D_{m-2}]}_{p_{m-1}}\ \cdots\ \underbrace{\mathbb{P}[D_2|D_1]}_{p_2}\ \underbrace{\mathbb{P}[D_1]}_{p_1} = \prod_{k=1}^{m} p_k\end{aligned}$$

Up to this point, the exact determination of probability γ_m has been shown. From now on the mechanism to make an estimation $\hat{\gamma}_m$ is introduced. The aim is to make separate estimations $\hat{p}_k$ for every p_k, $k = 1, 2, \cdots, m$. Even though they are not necessarily independent, once all the $\hat{p}_k$ estimations are done, $\hat{\gamma}_m = \prod_{k=1}^{m} \hat{p}_k$ [5] is an unbiased estimator of γ_m.

To determine $\hat{p}_1$, N_0 independent realizations of $X(t)$ are started from $X(0)$. Then, if R_1 realizations of $X(t)$ reach threshold ℓ_1 without falling under ℓ_0,

the estimation of interest results: $\hat{p}_1 = R_1/N_0$. The experiment is equivalent to observing N_0 Bernoulli variables with parameter p_1. Afterwards the state of the R_1 realizations that have reached ℓ_1 is saved (*saved state*). The remaining $m-1$ estimations $\hat{p}_k,\ k = 2, 3, \cdots, m$ are done similarly: N_{k-1} realizations of $X(t)$ are started from the R_{k-1} *saved states* and the number R_k is determined to let $\hat{p}_k = R_k/N_{k-1}$. For practical reasons it is desirable that $N_{k-1} > R_{k-1}$, however the relation between N_{k-1} and R_{k-1} is not analyzed now but delayed for the next sections instead. The consequence of $N_{k-1} > R_{k-1}$ is that from every *saved state* two or more realizations are to be started being then necessary to split or clone them. Finally, the unbiased estimation of $\hat{\gamma}_m$ results: $\hat{\gamma}_m = \frac{R_1}{N_0}\,\frac{R_2}{N_1}\,\frac{R_3}{N_2}\,\cdots\,\frac{R_m}{N_{m-1}}$.

A variance analysis done in [6] reports a value of $\gamma_m^2 m^2(1-\gamma_m^{1/m})/\gamma_m^{1/m}N$ while for [10] the variance is in the order of $m\gamma_m^{2-(1/m)}/N$. Both calculations have been done for a very simple setting in which the values of $p_k,\ k = 1, 2, \cdots, n$ are independent and equal. The implementation used was, in both cases, an N–trajectory *Fixed Effort* (see Section 5).

3 Network Modelling

An undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is a set of n absolutely reliable nodes and $\mathcal{E}$ a set of m independent unreliable links, is a commonly used model for studying communication network reliability. Being $X_i = 1$ when the i^{th} link is *operative* and $X_i = 0$ when the i^{th} link is *failed*, the variable vector $\mathbf{X} = (X_1, X_2, \cdots, X_m)$ depicts the state of the links. According to some probability mass function $f_{\mathbf{X}}(\mathbf{x})$, $r_i = \mathbb{P}[X_i = 1]$ is the single link reliability whereas $q_i = 1 - r_i = \mathbb{P}[X_i = 0]$ the single link unreliability.

A structure function $\Phi(\mathbf{X})$ that equals 1 when the network is *operative* and 0 when the network is *failed* let us define the network reliability and unreliability respectively as $R = \mathbb{P}[\Phi(\mathbf{X}) = 1]$ and $Q = \mathbb{P}[\Phi(\mathbf{X}) = 0]$. From now on a network is considered *operative* if there is a path of *operative* links between two nodes s and t, and *failed* otherwise (source-terminal network reliability).

Straightforward (also called standard, direct or crude) Monte Carlo estimations of network reliability and unreliability can be computed, respectively, as $\hat{R} = 1/N \sum_{i=1}^{N} \Phi(\mathbf{X}^{(i)})$ and $\hat{Q} = 1/N \sum_{i=1}^{N} (1-\Phi(\mathbf{X}^{(i)})$ where $\mathbf{X}^{(i)},\ i = 1, \cdots, N$ are *i.i.d.* samples taken from $f_{\mathbf{X}}(\mathbf{x})$. For highly reliable networks most of the $X^{(i)}$ samples will equal 1 and, as a consequence, most of the replications $\Phi(\mathbf{X}^{(i)})$ will equal 1 also. For extremely reliable networks, with unreliabilities in the order of 10^{-10} or even less, it will take an average of 10^{10} replications to get $\Phi(\mathbf{X}^{(i)}) = 0$ at least once. In these cases $\Phi(\mathbf{X}) = 0$ is, therefore, a rare event.

Another crude Monte Carlo approach consists of watching the links evolve in time, supposing that at time 0 all of them are *failed* and become *operative* at times $\tau_i,\ i = 1, 2, \cdots, m$, exponentially distributed with parameter λ_i. Variable $\mathbf{X} = (X_1, X_2, \cdots, X_m)$ should now be replaced by $\mathbf{X}(t) = (X_1(t), X_2(t), \cdots, X_m(t))$, a Markov Process called *Construction Process* [3]. Then: $X_i(t) = 1$ if $t \geq \tau_i$ (i^{th} link *operative*) and $X_i(t) = 0$ if $t < \tau_i$ (i^{th} link *failed*).

Since $\mathbb{P}[\tau_i \leq t] = 1 - e^{-\lambda_i t}$ is the probability that the i^{th} link becomes operative at time t or earlier, the choice of $\lambda_i = -\log(q_i)$ makes the probability that the i^{th} link is operative at $t = 1$ be exactly the single link reliability r_i. As a consequence the probability that the whole network is *operative* at $t = 1$ is R, and the probability that the network is *failed* at $t = 1$ is Q. Then $R = \mathbb{E}\{\Phi(\mathbf{X}(1))\}$ and $Q = \mathbb{E}\{1 - \Phi(\mathbf{X}(1))\}$.

The natural way to simulate the last model is (i) to sample a set of times $\tau_i,\ i = 1, 2, \cdots, m$, (ii) to check the value of $\Phi(\mathbf{X}(1)^{(i)})$ every sampled set, (iii) to repeat items (i) and (ii) N times, and finally to average the N values of either $\Phi(\mathbf{X}(1)^{(i)})$ to estimate R or $1 - \Phi(\mathbf{X}(1)^{(i)})$ to estimate Q. From now on the replications index will be omitted and $\mathbf{X}(1)^{(i)}$ will be referred to as $\mathbf{X}(1)$.

The main weakness of these crude estimations comes up when the networks are highly reliable. Particularly the relative precision of the unreliability estimation drops dramatically as Q tends to 0. For this estimation the exact variance is $\mathbb{V}\{\hat{Q}\} = Q(1 - Q)/N$ and the relative error, expressed by the coefficient of variation, $\sqrt{\mathbb{V}\{\hat{Q}\}}/\mathbb{E}\{\hat{Q}\} = \sqrt{(1 - Q)/NQ}$. Hence, for highly reliable networks (Q extremely small), an accurate estimation requires a very large sample size N.

4 Splitting Network Model

Splitting [5, 7, 8, 10, 14] is a variance reduction technique which aims to improve the precision of crude Monte Carlo approaches. Even when this technique has been usually applied to estimate the probability of rare events in a Markovian setting, showing to be particularly efficient to solve unreliability estimations, it has not been yet applied to compute a network reliability measure like the one discussed previously in this paper. The core of this work is the proposal of a Splitting technique that can be applied over the last of the models introduced in Section 3; the main ideas are developed in this section.

Considering the commutation times defined so far, i.e. the exponentially distributed times at which the links goes from *failed* to *operative*, define a trajectory as: $\mathcal{T} = \{\tau^1, \tau^2, \cdots, \tau^m\}$, $\tau^1 \leq \tau^2 \leq \cdots \leq \tau^m$. A more explicit notation for τ^i would be τ^i_j, meaning that link j goes from *failed* to *operative* in time τ^i. However, depending on the context, it might be enough with only one index. In correspondence with trajectory $\mathcal{T}$, and under the same index considerations, it is possible to define $\Lambda = \{\lambda^1, \lambda^2, \cdots, \lambda^m\}$.

For any given trajectory $\mathcal{T}$, event $E = [\Phi(\mathbf{X}(1)) = 0]$ occurs if the network becomes operative after $t = 1$. Being $Q = \mathbb{P}[E]$, for a considerable number of trajectories, the estimation $\hat{Q}$ is the proportion of them for which event E occurs. For highly reliable networks E will be a rare event. In order to make the estimations of Q more accurate, it is useful to make event E occur "more often".

If trajectories $\mathcal{T}$ are seen as the replications of process $X(t)$ introduced in Section 2, the occurrence of event E, i.e. trajectory $\mathcal{T}$ entering region $t > 1$ with the network still *failed*, is equivalent to process $X(t)$ entering region $\mathcal{X}_B$ without having entered region $\mathcal{X}_A$. On the other side, a trajectory $\mathcal{T}$ making the network

become *operative* in $t \leq 1$ is equivalent to process $X(t)$ entering region $\mathcal{X}_A$. This observation leads to the suggestion of applying Splitting to the trajectories $\mathcal{T}$.

Let thresholds $u_0 = 0 < u_1 < u_2 < \cdots < u_n = 1$ define a partition of the interval $[0, 1]$ and let's define events $E_k = [\Phi(\mathbf{X}(u_k)) = 0],\ k = 1, 2, \cdots, n$. Then:

$$Q = \mathbb{P}[E_n]$$
$$= \underbrace{\mathbb{P}[E_n|E_{n-1}]}_{p_n}\ \underbrace{\mathbb{P}[E_{n-1}|E_{n-2}]}_{p_{n-1}}\ \cdots\ \underbrace{\mathbb{P}[E_2|E_1]}_{p_2}\ \underbrace{\mathbb{P}[E_1]}_{p_1}\ =\ \prod_{k=1}^{n} p_k$$

and therefore $\hat{Q} = \prod_{k=1}^{n}\ \hat{p}_k$. It is necessary then to estimate the single values of $\hat{p}_k$. These estimations are done as part of the whole Splitting mechanism that for the current problem can be summarized as follows: start one or more trajectories $\mathcal{T}_1, \mathcal{T}_2, \cdots$ from threshold $u_0 = 0$; keep checking everyone of them and (i) cancel the ones for which event E_1 does not occur and (ii) clone the ones for which event E_1 occurs. Proceed the same way with all the new trajectories started from u_1, i.e. cancel or clone at the cross of threshold u_2. Keep repeating the mechanism until threshold $u_n = 1$ is reached. In other words, cancel trajectories as soon as the network becomes *operative* and clone trajectories at every threshold cross, whenever the network is still *failed* at the cross. The target is to finally have some trajectories for which the network becomes operative beyond $t = 1$. If this is achieved, the following estimations are possible: $\hat{p}_k = R_k/R_{k-1},\ k = 1, 2, \cdots, n$, where R_k is the total number of trajectories that have crossed threshold u_k and R_0 the number of trajectories $\mathcal{T}_1, \mathcal{T}_2, \cdots$ started from threshold $u_0 = 0$.

Two issues mentioned in the prior paragraphs deserve further explanation: how to generate trajectories and how to clone them.

The most immediate way to sample $\mathcal{T} = \{\tau^1, \tau^2, \cdots, \tau^m\}$ where $\tau^1 \leq \tau^2 \leq \cdots \leq \tau^m$, is to sample separate and independently every element as $\tau^i = -\log U/\lambda_i$ where $U \sim unif(0, 1]$, and once they are all sampled, arrange them by increasing order of time. The main limitation of this option is that it can only sample a trajectory as a whole, being unable to do it in stages. If at certain time a sub–set of the m times is already sampled, say $\mathcal{T}' = \{\tau^1, \tau^2, \cdots, \tau^q\}$ with $q < m$, and some simulation process makes use of $\mathcal{T}'$, it's impossible to come back to it in further time to sample the remaining links because some of the new times might need to be inserted among the existing ones (that have already been used!).

Some properties of the *Construction Process* [3], due to the exponential distributions, make it possible to sample trajectories in a different way. Prior to introducing this sampling mechanism we define sub–trajectories:

$\mathcal{T}_-^i = \{\tau^1, \tau^2, \cdots, \tau^j\}$ is the subset of elements of $\mathcal{T}$ formed by the times that are earlier or equal to i: $\tau^j \leq i$ and $\tau^{j+1} > i$. In correspondence with $\mathcal{T}_-^i$ it is possible to define: $\Lambda_-^i = \{\lambda^1, \lambda^2, \cdots, \lambda^j\}$.

$\mathcal{T}_+^i = \{\tau^j, \tau^{j+1}, \cdots, \tau^m\}$ is the subset of elements of $\mathcal{T}$ formed by the times grater than i: $\tau^j > i$ and $\tau^{j-1} \leq i$. In correspondence with $\mathcal{T}_+^i$ it is possible to define: $\Lambda_+^i = \{\lambda^j, \lambda^{j+1}, \cdots, \lambda^m\}$.

Upon these definitions, trajectories can now be sampled link by link:

Sampling the next link Let the set of links whose commutation time belong to $\mathcal{T}_+^i$ in time i be $S_Z = \{e_r, e_s, \cdots, e_t\}$, then λ_r, $\lambda_s \cdots$ and λ_t belong to Λ_+^i. The first element of $\mathcal{T}_+^i$ can be sampled from the discrete random variable Z with domain S_Z and $\mathbb{P}[e_w] = \lambda_w / \sum_{\lambda_v \in \Lambda_+^i} \lambda_v,\ w = r, s, \cdots, t$.

Sampling the time for the next link The periods between commutation times of a trajectory: $\Delta^t = \tau^t - \tau^{t-1},\ t = 1, 2, \cdots, m$ (accepting $\tau^0 = 0$), are exponentially distributed with parameter $\sum_{\lambda_s \in \Lambda_t^+} \lambda_s$. Then for any time i, being $\mathcal{T}_i^-$ already sampled, it is possible to sample the time τ^{i+1} for the first position of $\mathcal{T}_+^i$ by adding Δ^{i+1} to τ^i: $\tau^{i+1} = \tau^i + \Delta^{i+1}$.

So, the times $\tau_i,\ i = 1, 2, \cdots, m$ can be sampled *on demand*, being the resulting trajectory *under construction* all the time. Conditions may be checked whenever necessary for all the existing trajectories and then, at any simulation stage, more times can be added to anyone in order to check further conditions.

The final issue to consider concerns the way to clone trajectories every time a threshold is crossed. Suppose that trajectory $\mathcal{T}$ is *under construction*, being the last two commutations sampled times τ^x and τ^{x+1} such that $\tau^x < u_k < \tau^{x+1}$. Suppose also that the network is still *failed* at time τ^{x+1} meaning that event E_k has actually occurred. Trajectory $\mathcal{T}$ must therefore be cloned at $t = u_k$. The value of τ^{x+1} has once been obtained as $\tau^{x+1} = \tau^x + \Delta^{x+1}$ where, as stated, $\tau^{x+1} > u_k$. The *memoryless* property applied to the time Δ^{x+1} (exponentially distributed) means that we can resample a new statistically equivalent value for τ^{x+1} as $\tau^{x+1} = u_k + \Delta^{x+1}$ rather than using the original one $\tau^x + \Delta^{x+1}$. If trajectory $\mathcal{T}$ has to be cloned n_k times, n_k new values can be obtained for τ^{x+1}.

5 Implementation

There are two main options in a Splitting implementation: *Fixed Splitting* and *Fixed Effort*. If at every threshold cross the trajectories are cloned or split in a fixed number of copies, the number of trajectories started at every threshold is a random variable. This option is *Fixed Splitting*. Accepting that the whole simulation effort is proportional to the number of started trajectories, some sort of control on such a number would permit some control on the simulation times. To do this, the number of copies should be controlled at every Splitting point. *Fixed Effort* proposes to clone as much as necessary to let the total number of trajectories that start from every threshold be a fixed number. There is no general agreement about the variance reduction benefits of every option, however *Fixed Effort* provides a closer control on the execution times.

A crucial issue in a Splitting implementation is how to select the number of thresholds. There is no recipe for such determination, only some guidelines derived from the analysis of very particular problems. These guidelines suggest to set this number to $-(\log \hat{Q})/2$ where $\hat{Q}$ is the probability to estimate. This calculation makes it necessary to dedicate some previous runs to set the number of thresholds. The factor $-(\log \hat{Q})/2$ has been obtained separately and after

different analysis by [6, 5] and [10] for the particular case in which the values of $p_k,\ k = 1, 2, \cdots, n$ are independent and equal. For these cases both analysis prove that $-(\log \hat{Q})/2$ produces the maximum variance reduction. The use of this calculation is recommended anyway for any kind of problem, at least as starting value for further iterations.

An important problem arising in typical Splitting implementations is the considerable computational effort that might be necessary to simulate the process since any threshold is crossed until the trajectory eventually "dies". In the most general case, after any threshold cross, the process may follow an *up* and *down* evolution before the final condition is reached (i.e. before it enters $\mathcal{X}_A$, in terms of Section 2). In the network model that we propose to solve by Splitting, this problem does not arise. The only wasted effort for any "dying" trajectory is the one devoted to take it closer to the next threshold but, if the next threshold is not reached, the trajectory "dies" suddenly with no additional effort, actually with the same effort that could have necessary to take it to the next threshold in the successful case.

6 Numerical Results

The experimental part of this report has been conducted on the three network topologies shown in Figure 1, known as the bridge, the dodecahedron, and the Arpanet (1972 topology). We take equi-reliable links, studying the cases where single link reliabilities are `0.9`,`0.99`,`0.9999` and `0.999999`, resulting in source-terminal unreliabilities between `2.00e-02` and `2.00e-18`. The Splitting implementation selected was *Fixed Effort* and the total number of trajectories started from every threshold was set to `2,000` in all the experiments. In order to estimate the variance of the unreliability estimations every experiment was repeated `100` times, and out of these sequences of `100` experiments, the final estimation of the network unreliability and its associated variance were determined as $\hat{Q} = \frac{1}{N}\sum_{i=1}^{N} \hat{Q}_i$ and $\mathbb{V}\{\hat{Q}\} = \frac{\sum_{i=1}^{N} \hat{Q}_i^2}{N(N-1)} - \frac{\hat{Q}^2}{N-1}$, being $N =$`100` and $\hat{Q}_i$ every single estimation done by means of a `2,000` *Fixed Effort* trajectories.

The relative error (RE), expressed by an estimation of the coefficient of variation $\sqrt{\hat{\mathbb{V}}\{\hat{Q}\}}/\hat{Q}$, shows a very high independence with respect to the value of $\hat{Q}$. The following is a summary for all the experiments:

Dodecahedron:	`2.83e-03`$< \hat{Q} <$`2.04e-18`	$\rightarrow$	`1.02`%<RE<`2.70`%
Bridge:	`2.16e-02`$< \hat{Q} <$`2.00e-12`	$\rightarrow$	`0.59`%<RE<`1.53`%
Arpanet:	`9.54e-02`$< \hat{Q} <$`6.02e-12`	$\rightarrow$	`0.44`%<RE<`1.76`%

Besides the estimations of unreliability $\hat{Q}$, two values have been collected: the execution times t_S and the variances of the estimations $\mathbb{V}_S$. Both of these values must be taken into account to assess the simulations performance. We also performed experiments by means of a Crude Monte Carlo simulation, to estimate the corresponding t_C and $\mathbb{V}_C$ values. In order to compare the methods taking into account all factors, we assessed the single measure $(t_C \times \mathbb{V}_C)/(t_S \times \mathbb{V}_S)$,

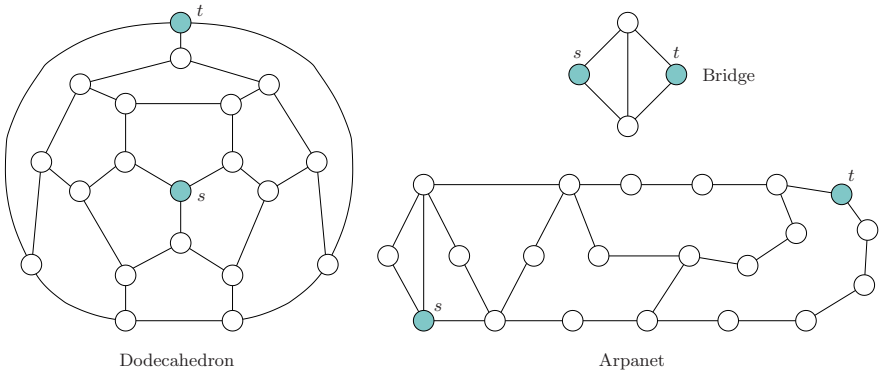

Fig. 1. Three network topologies

usually labelled as Speedup (or also Precision Gain, as it shows the precision improvement of the incumbent method S over the standard crude Monte Carlo C, given a fixed computational time[1]). The Speedup values are recorded in Table 1. The values obtained with a number of $-(\log \hat{Q})/2$ thresholds are square bracket emphasized while the best performance values are underlined. The ideal number of thresholds seems to be slightly higher than $-(\log \hat{Q})/2$. However the method is robust in this feature and does not seem to change the performance considerably at a variation on the number of thresholds around $-(\log \hat{Q})/2$. As the number of thresholds grow higher (beyond $-(\log \hat{Q})/2$) the performance falls. The Speedup grows as the network reliability grows.

7 Conclusions

A well known Monte Carlo technique called Splitting, which was successfully employed to solve a variety of problems, most of them over Markovian models, was customized in this paper to solve the source-terminal network reliability estimation. The performance over a set of numerical experiments reveal a very high independence with respect to the estimated values. For the tested networks the relative error is almost invariant and the precision gain is huge compared to the pure Crude Monte Carlo technique.

There are many open problems derived from this paper proposal. One of them is to study in more detail performance vs. number of thresholds and try to find a relation between this number and the network parameters. It will be interesting also to evaluate different network topologies. Another future work is to implement and evaluate a *Fixed Splitting* variant.

Finally, as the results obtained in this work are very promising, specially by the precision robustness in the case of highly reliable networks, an in-depth comparative study should be made. As many other variance reduction schemes proposed in the literature show good performance in this arena, it is therefore worthwhile to study the conditions where each method may be preferred.

Table 1. Precision Gain

Single Link Reliability	Thresholds	**Dodecahedron**	**Bridge**	**Arpanet**
0.9	1	5.05e-01	2.97e-01	[6.47e-01]
	2	1.09e+00	[5.03e-01]	5.59e-01
	3	[1.76e+00]	7.22e-01	4.00e-01
	5	2.95e+00	5.86e-01	3.62e-01
	7	2.41e+00	3.84e-01	4.07e-01
	9	1.78e+00	2.79e-01	2.69e-01
	11	1.38e+00	2.33e-01	1.84e-01
	13	1.51e+00	1.30e-01	9.71e-02
	15	8.86e+01	9.31e-02	4.65e-02
0.99	2	---	4.51e+00	3.23e+00
	4	3.00e+02	[9.08e+00]	[9.85e+00]
	6	4.80e+02	1.21e+01	1.05e+01
	7	[4.88e+02]	1.72e+01	1.08e+01
	10	5.48e+02	1.14e+01	1.08e+01
	13	8.30e+02	1.00e+01	6.21e+00
	16	5.84e+02	9.22e+00	4.76e+00
	19	3.75e+02	6.54e+00	2.45e+00
	22	3.66e+02	3.50e+00	9.62e-01
0.9999	3	---	---	---
	6	---	3.03e+04	1.96e+04
	8	8.33e+07	3.98e+04	[2.62e+04]
	9	1.21e+08	[3.27e+04]	3.39e+04
	13	[3.06e+08]	4.33e+04	2.51e+04
	18	2.32e+08	3.08e+04	3.68e+04
	23	2.05e+08	3.24e+04	1.97e+04
	28	1.61e+08	2.73e+04	2.04e+04
	33	1.30e+08	1.70e+04	1.26e+04
0.999999	4	---	---	---
	8	---	8.15e+07	9.30e+07
	12	4.67e+13	1.21e+08	9.66e+07
	13	7.58e+13	[1.45e+08]	[1.53e+08]
	16	7.96e+13	1.82e+08	1.95e+08
	20	[1.32e+14]	1.50e+08	1.52e+08
	30	1.06e+14	1.36e+08	1.42e+08
	40	1.04e+14	1.35e+08	7.50e+07
	50	6.76e+13	9.01e+07	5.20e+07

The meaning of --- is that some threshold has not been reached by any trajectory. As a consequence, the Splitting estimations are undefined.

References

1. H. Cancela and M. El Khadiri. A recursive variance-reduction algorithm for estimating communication-network reliability. *IEEE Transactions on Reliability*, 44(4):595–602, December 1995.
2. Jason L. Cook and Jose Emmanuel Ramirez-Marquez. Two-terminal reliability analyses for a mobile ad hoc wireless network. *Reliability engineering & systems safety*, 92(6):821–829, 2007.
3. T. Elperin, I. B. Gertsbakh, and M. Lomonosov. Estimation of network reliability using graph evolution models. *IEEE Transactions on Reliability*, 40(5):572–581, Dec 1991.
4. Susana Duarte Flores, Benjamín Barán Cegla, and Diana Benítez Cáceres. Telecommunication network design with parallel multi-objective evolutionary algorithms. In *Proceedings of the 2003 IFIP/ACM Latin America conference*, pages 1 – 11. ACM, 2003.
5. M. J. J. Garvels. *The Splitting Method in Rare Event Simulation*. PhD thesis, Faculty of mathematical Science, University of Twente, The Netherlands, 2000.
6. M. J. J. Garvels and D. P. Kroese. A comparison of RESTART implementations. In *Proceedings of the 1998 Winter Simulation Conference*, pages 601–609. IEEE Press, 1998.
7. M. J. J. Garvels, D. P. Kroese, and J.-K. C. W. Van Ommeren. On the importance function in splitting simulation. *European Transactions on Telecommunications*, 13(4):363–371, 2002.
8. P. Glasserman, P. Heidelberger, P. Shahabuddin, and T. Zajic. Splitting for rare event simulation: Analysis of simple cases. In *Proceedings of the 1996 Winter Simulation Conference*. IEEE Press, 1996.
9. Dilek Günnec and F. Sibel Salman. Assessing the reliability and the expected performance of a network under disaster risk. In *Proceedings of the International Network Optimization Conference, INOC 2007*, Spa, Belgium, 2007.
10. P. L'Ecuyer, V. Demers, and B. Tuffin. Rare-events, splitting, and quasi-Monte Carlo. *ACM Transactions on Modeling and Computer Simulation*, 17(2):Article 9, 2007.
11. Nobuoto Nojima. Prioritization in upgrading seismic performance of road network based on system reliability analysis. In *Third China-Japan-US Trilateral Symposium on Lifeline Earthquake Engineering*, Kunming, China, 1998.
12. J.S. Provan and M.O. Ball. The complexity of counting cuts and of computing the probability that a graph is connected. *SIAM Journal on Computing*, 12:777–788, 1983.
13. Heidi A. Taboada, Fatema Baheranwala, and David W. Coit. Practical solutions for multi-objective optimization: An application to system reliability design problems. *Reliability Engineering and System Safety*, 92:314–322, 2007.
14. M. Villén-Altamirano and J. Villén-Altamirano. Restart: A method for accelerating rare events simulations. In *Proceedings of the 13th International Teletraffic Congress*, pages 71–76. North-Holland, 1991.
15. Hiroshi Wakabayashi. Network reliability improvement: Probability importance and criticality importance. In *Proceedings of the 2nd Symposium on Transportation Network Reliability (INSTR 2004)*, pages vol.3, pp. 204–210, Christchurch, New Zealand, 2004.