*Proceedings of the 2005 Winter Simulation Conference*
*M. E. Kuhl, N. M. Steiger, F. B. Armstrong, and J. A. Joines, eds.*

# Sensitivity analysis of network reliability using Monte Carlo (revision 1, Dec. 17, 2005)

G. Rubino

IRISA-INRIA
Campus Universitaire de Beaulieu
35042 Rennes Cedex, France

## ABSTRACT

We analyze the computation of sensitivities in network reliability analysis. The associated models are graphs whose components are weighted by probabilities (their reliabilities) and they are widely used, for instance, in the design of communication networks. The paper deals with the sensitivities of usual reliability network metrics, with respect to the reliabilities of the components. The importance of sensitivities in this context is discussed and it is shown how to efficiently estmate the vector of sensitivities using Monte Carlo procedures. A first result allows to evaluate sensitivities using the standard Monte Carlo approach. A second method is then presented to deal efficiently with the rare event case. The ideas presented here can be applied to other classes of reliability problems and/or methods.

**Contents**



## 1 INTRODUCTION

### 1.1 The model

Stochastic graphs are widely used models for representing complex multi-component systems subject to component failures. They consist of graphs whose elements (nodes and arcs or edges) are weighted by probabilities. The output usually obtained from this type of structure is the probability of events representing a desired behavior of the system modeled. In this paper we consider the problem of the computation of reliability measures and we focus on a fundamental family of metrics, called $\mathcal{K}$-terminal reliability, in the following version: the input is a stochastic undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V}$ is the set of vertices and $\mathcal{E}$ is the set of edges. We denote by $E$ the number of edges: $E = |\mathcal{E}|$. The graph may represent, for instance, a communication network. In this case, the vertices correspond to nodes sending and receiving information and the edges model communication lines. This application is used as a reference in the paper. For instance, the terms *vertex* and *node* are used as synonymous, and the same with *edge* or *line* or *link*. At a fixed time $\tau$, each line is in one of two states: either working (*operational*) or completely down (*unoperational*, behaving as if it did not exist in the network). To simplify the presentation, we assume that nodes are perfect and that only lines can fail, but this is not critical since we are using Monte Carlo techniques. The graph is also assumed to be undirected, that is, the lines of the modeled communication network are bi-directional (messages can pass in both directions.) Finally, we assume that the graph is connected and without loops.

The state of line or edge $e$ at time $\tau$ is a binary random variable $X_e$ with value 1 if the line is working or 0 if it does not work. A basic assumption now is that $(X_e, e \in \mathcal{E})$ is a family of independent random variables. The $\mathcal{K}$-terminal problem is the following:

Given the probability $r_e$ of the event $\{X_e = 1\}$ for each line $e$ of the network and a subset $\mathcal{K}$ of the set of vertices $\mathcal{V}$, compute the probability $R$ that, at time $\tau$, the nodes can communicate with each other. In other words, if the first graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is fixed, the set $\mathcal{E}'$ of operational lines at time $\tau$ defines a subgraph $\mathcal{G}' = (\mathcal{V}, \mathcal{E}')$ of the previous one.

The number $R$ is the probability that there exists at least one path between every pair of nodes of $\mathcal{K}$ in $\mathcal{G}'$, or, in other words, that the nodes of $\mathcal{K}$ belong to the same connected component of $\mathcal{G}'$. The number $r_e$ is the *reliability* of line $e$. The set $\{r_e,\ e \in \mathcal{E}\}$ is the set of *elementary* reliabilities. In the paper, we give the details for the basic case of $\mathcal{K} = \{s, t\}$, the so-called 2-terminal case, but the methods are valid in the general context, with the appropriate adjustments.

This problem (and several other related reliability problems) has received considerable attention from the research community. Numerous published papers have been devoted to its solution (see, for instance, (Locks and Satyarayana 1986, Ball, Colbourn, and Provan 1995, Rubino 1998) for references) mainly due to two reasons: the general use of these models, in particular in the communication networks area, and the fact that in the general case the problem is in the #P-complete class (Valiant 1979), a family of *NP*-hard problems not known to be in *NP*. Recall that a #P-complete problem is equivalent to counting the number of solutions to an *NP*-complete one. This implies that a #P-complete problem is at least as hard as an *NP*-complete one. This last fact justifies the continuous effort to find more efficient solution methods.

The main goal of this work is to show that sensitivities can be efficiently computed using appropriate Monte Carlo techniques, including in the rare event case, that is, when the network is highly reliable. Before going to the Monte Carlo context, let us recall in next subsection the interest in computing sensitivities.

### 1.2 Motivation for sensitivity analysis

If we consider the reliability $R$ of the network as a function of the individual reliabilities $r_e$ of the lines, it is of interest to know the value of the sensitivity $\sigma_e$ of $R$ with respect to each independent variable $r_e$ which are defined as

$$\sigma_e \stackrel{\text{def}}{=} \frac{\partial R}{\partial r_e}.$$

Several reasons make the computation of this set of derivatives (that is, the vector $\nabla R$) interesting. First, it is involved in optimizations, as pointed out in (Barlow and Proschan 1975), when, for instance, each elementary reliability is a function of some local cost measure $c_e$: $r_e = r_e(c_e)$. Assume that a fixed capital $C$ is available to improve an existing communication network by investing into the lines. If the criterium is to increase the reliability of the network, we have to maximize $R = R(c_e,\ e \in \mathcal{E})$ under the constraint $\sum_e c_e = C$. The rate $\varrho_e = \partial R/\partial c_e$ is the instantaneous variation of $R$ with respect to the variation of the local cost of line $e$. Since only $r_e$ depends on $c_e$, we have

$$\varrho_e = \sigma_e \frac{dr_e}{dc_e}.$$

Once the vector $\nabla R$ is computed, the rates $\varrho_e$ can be obtained using only the local derivates $dr_e/dc_e$, i.e., without explicitly constructing and computing the function $R = R(c_e,\ e \in \mathcal{E})$. This is a rather classical application of sensitivity analysis. It can be mentioned that many authors propose the computation of a symbolic expression of the function $R(r_e,\ e \in \mathcal{E})$ (that is, a formal expression) in order to perform optimizations (see for instance (Hariri and Raghavendra 1987).) Of course, symbolic expressions are not only used for this. When the topology is fixed and the user wants to evaluate its model for different values of the reliabilities of its components, it is very useful to obtain such a symbolic expression first and to perform just variable substitutions by different numerical values. However, the price to pay is a considerably higher computational cost.

We performed some experiments in order to compare these two approaches. We chose an exact combinatorics method (adapting the algorithm proposed in (Cancela, Rubino, and Urquhart 2001) to the 2-terminal case). This algorithm is an improved version of the method proposed in (Ahmad 1982), having the property of using a bounded amount of memory which is known before running. The program written in `C` runs approximately over 100 times faster than the same algorithm coded in the formal system Maple. This ratio varies considerably as a function of the sizes of the coefficients and the formal simplifications that Maple is able to perform.

The computation of $\nabla R$ may also be useful if there is a temporal dimension in the problem. Assume that each individual reliability is considered as a known function of time, that is, $r_e = r_e(\tau)$, $\tau$ measuring the time elapsed since some reference point in the past. The instantaneous variation of $R$ with time can be obtained without considering explicitly $R = R(\tau)$. Applying elementary differentiation rules, we have

$$\frac{dR}{d\tau} = \sum_e \sigma_e \frac{dr_e}{d\tau}.$$

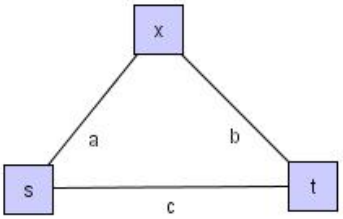


Figure 1: A small "triangle" to illustrate some uses of sensitivities.

If we know how to calculate $\nabla R$ efficiently, this approach is better than the computation of $dR/d\tau$ directly from the function $R = R(\tau)$ since this leads to the computation of $R(s)$ for different values of $s$ in some neighborhood of $\tau$. Moreover, we necessarily get a numerical approximation of $dR/d\tau$, whereas using the previous formula, the (numerically) exact value is obtained.

A more specific and relevant application of the computation of sensitivities in our context is the following. The reliability measure $R$ quantifies the ability of the whole network to transmit messages between the two selected nodes at a fixed point in time (for instance, in steady state.) This number depends, of course, on the precision of input data which also consists of measured (or calculated) reliabilities. The possible (and probable) errors in the input induce an error in the output $R$. Although this numerical output $R$ may not be very precise, the order induced on the set of edges by sensitivities is a much more robust information about the properties of the network. It allows the identification of the most critical parts of the structure, taking into account both the topology and the stochastic behavior of the system (Barlow and Proschan 1975). It is clear that this order is significantly less dependent on the errors arising from each individual reliability $r_e$. As an example, consider the network of Figure 1.

We have $R = r_a r_b + r_c - r_a r_b r_c$, $\sigma_a = r_b(1 - r_c)$, $\sigma_b = r_a(1 - r_c)$ and $\sigma_c = 1 - r_a r_b$. If $r_a = r_b = r_c = 0.9$ then $\sigma_a = \sigma_b = 0.09$ and $\sigma_c = 0.19$, that is, link $c$ is the critical one. It is clear that $c$ will remain the critical link if the "real" values of the elementary reliabilities are "near" 0.9. If we let $r_a = r_b = 0.9$ and we decrease the value of $r_c$, while $r_c > 71/90 \approx 0.7889$ we will have $\sigma_c > \sigma_a = \sigma_b$. For instance, if $r_c = 0.7$, $\sigma_a$ (and $\sigma_b$) changes to 0.27 and $c$ is the least critical link in the network. To illustrate the robustness, the reader can verify for instance that if $r_a < 0.5$, then for any $r_b < r_a$ and for any $r_c$ we have $\sigma_a < \sigma_b < \sigma_c$.

To complete the introduction, let us mention that another pioneering work on the *importance* of a component in a complex system is (Birnbaum 1969). In (Barlow and Proschan 1975), the analysis of sensitivities is considered in the general framework of coherent binary systems. In the case of reliability network theory, the computation of sensitivities for the particular case of series-parallel topologies has been done in (Bobbio and Premoli 1982).

The rest of this paper is devoted to the analysis of the sensitivities of these basic network reliability measures using Monte Carlo. In next section, some general remarks about sensitivities are given. Section 3 contains the main contributions of the paper. In Subsection 3.1 we propose a new estimator allowing evaluating $\nabla R$ as a byproduct of the estimation of the system reliability $R$. In Subsection 3.2, this estimator leads to a specific algorithm designed to work in the rare event case, that is, when $R \approx 1$. Section 4 provide some examples. Finally, Section 5 concludes the paper.

## 2 Definitions and basic properties

Let $Y$ be the binary random variable defined by $Y = 1$ iff there exists at least one path between $s$ and $t$ in which all the links are working. The reliability of the network is $R = \Pr(Y = 1)$, $\Pr()$ being the underlying probability measure. The sensitivity vector is

$$\nabla R \stackrel{\text{def}}{=} \left( \dots \frac{\partial R}{\partial r_e} \dots \right) = ( \dots \sigma_e \dots ) .$$

As this model is a particular case of a *binary monotone system* (see (Rubino 1998), it is well known that for every line $e$ we have $\sigma_e \geq 0$ (see the basic reference (Barlow and Proschan 1975)). This means that we cannot improve the reliability of the whole system by decreasing the (elementary) reliability of one of its components.

The *contraction* of line $e = \{x, y\}$ of $\mathcal{G}$ consists of eliminating line $e$ and "merging" its two extremities $x$ and $y$: the resulting graph has $|\mathcal{V}| - 1$ nodes and $E - 1$ edges and it is denoted by $\mathcal{G}_e^c$. Just deleting line $e$ gives $\mathcal{G}_e^d$ with the same node set and $E - 1$ edges. Let us define

$$R_e^c \stackrel{\text{def}}{=} R|_{r_e=1} \quad \text{and} \quad R_e^d \stackrel{\text{def}}{=} R|_{r_e=0} . \tag{1}$$

The number $R_e^c$ (respectively $R_e^d$) is the reliability of the connection between $s$ and $t$ in $\mathcal{G}_e^c$ (respectively in $\mathcal{G}_e^d$.) If line $e$ is $\{s, t\}$, then $R_e^c$ is defined as 1. Between these quantities the following order relation holds:

$$\text{for all lines } e, \quad 0 \leq R_e^d \leq R \leq R_e^c \leq 1. \tag{2}$$

This is simply a consequence of the fact that we deal with a *monotone* system. Conditioning with respect to the state of line $e$ we obtain the following useful relation:

$$\text{for all lines } e, \quad R = r_e R_e^c + (1 - r_e) R_e^d. \tag{3}$$

Relation (3) is usually called the "factoring theorem" and it is the basis of a powerful family of algorithms designed to compute network reliability exactly (see for instance (Rubino 1998)). Deriving it with respect to $r_e$ gives

$$\text{for all lines } e, \quad \sigma_e = R_e^c - R_e^d. \tag{4}$$

This last relation can be found in (Barlow and Proschan 1975). From relations (2) and (4) we have the following trivial bounds:

$$\text{for all lines } e, \quad \sigma_e \leq R_e^c \quad \text{and} \quad \sigma_e \leq 1 - R_e^d.$$

Now, formula (4) together with (3) give the two following expressions of $\sigma_e$:

$$\text{for all } e \text{ s.t. } 0 < r_e < 1, \quad \sigma_e = \frac{R_e^c - R}{1 - r_e} \tag{5}$$

$$= \frac{R - R_e^d}{r_e}. \tag{6}$$

Relations (5) or (6) allow us to compute $\nabla R$ by means of $|\mathcal{E}| + 1$ reliability computations. They also lead to the two following bounds of sensitivities:

$$\text{for all lines } e, \quad \sigma_e \leq \frac{R}{r_e} \quad \text{and} \quad \sigma_e \leq \frac{1 - R}{1 - r_e}.$$

Observe that

$$\frac{R}{r_e} \lesseqgtr 1 \iff \frac{1 - R}{1 - r_e} \gtreqless 1,$$

so only one of the preceding bounds is really relevant.

For completeness, let us mention another measure of the importance of a component on the reliability of the whole system, proposed in (Fussell 1975). Its definition is

$$\xi_e \stackrel{\text{def}}{=} \Pr(X_e = 0 \mid Y = 0),$$

that is, $\xi_e$ is the probability that line $e$ has failed, given that the network itself has failed. The author proposes this measure as a tool for diagnostic analysis, when the network is observed to fail. After some transformations, we have

$$\xi_e = \frac{(1 - R_e^d)(1 - r_e)}{1 - R},$$

and using (6),

$$\xi_e = \frac{(1 - R + r_e \sigma_e)(1 - r_e)}{1 - R},$$

so that this measure can be directly derived from sensitivities.

Lastly, remark that as in physics, it can be useful to consider the *elasticity* $h_e$ of $R$ with respect to $r_e$, instead of the sensitivity. The elasticity is defined here by

$$h_e \stackrel{\text{def}}{=} \frac{r_e \sigma_e}{R}. \tag{7}$$

From (6), we have that

$$\sigma_e \leq \frac{R}{r_e}, \tag{8}$$

so that we always have $h_e \leq 1$ and it is immediate to verify that $h_e = 1$ if and only if line $e$ is in series with the rest of the network. This quantity reflects the same kind of property as $\sigma_e$ in a normalized way.

## 3 Sensitivity estimation with Monte Carlo

As stated before, network reliability computations are expensive (since we are generally dealing with *NP*-hard problems). This means that, very often, it is not possible to solve exactly the models in reasonable time (this is usually the case in a good workstation if the network has, say, more than 50 or 60 links after possible simplifications such as series-parallel, etc., see (Rubino 1998)). An efficient alternative is the Monte Carlo approach. Instead of getting the exact (numerically speaking) answer, we accept an approximated one but with a (probabilistic) control on the accuracy. The gain is in the size of the models that can be solved.

The naive implementation of the standard Monte Carlo technique consists of sampling from the random variable $\mathcal{G}'$ (with the type "subgraph of $\mathcal{G}$") for, say, $N$ times, checking the criteria defining the operational state of each sample (that is, checking if nodes $s$ and $t$ are connected in $\mathcal{G}'$, in the 2-terminal situation, or if the nodes in $\mathcal{K}$ are in a same connected component of $\mathcal{G}'$ in the general case) and computing the ratio between the number of samples in operational state and the total number of samples $N$ as an estimate of the reliability of the network.

Next, we show that sensitivities can also be obtained by Monte Carlo simulations without using relations (5) or (6), that is, using the standard method and with a small overhead. Then, we propose a specific algorithm applying this idea but designed to operate in the rare event case.

### 3.1 General approach to estimate derivatives

Recall that the random variable $X_e$ has value 1 if line $e$ is working in $\mathcal{G}'$, 0 otherwise, and that Y has value 1 if $s$ and $t$ are connected in $\mathcal{G}'$, 0 otherwise, so that $R = \mathrm{E}(Y)$. A basic Monte Carlo estimation of $R$ consists of building $N$ independent replications of $Y$, which we denote here by $Y^{(1)}, \cdots, Y^{(N)}$, and estimating $R$ by

$$\widetilde{R} = \frac{1}{N}\sum_{i=1}^{N} Y^{(i)}.$$

Now, if sensitivities are to be computed, the first contribution of this paper is to propose an estimator allowing the evaluation of $\nabla R$ as a byproduct of the evaluation of $R$.

For any line $e$, define the r.v.

$$Z_e \stackrel{\text{def}}{=} X_e Y.$$

The $N$ corresponding replications in the considered sample are denoted by $Z_e^{(1)}, \cdots, Z_e^{(N)}$. Observing that

$$Z_e = 1 \Longleftrightarrow X_e = 1 \text{ and } Y = 1$$

we have

$$\begin{aligned} \Pr(Z_e = 1) &= \Pr(X_e = 1, Y = 1) \\ &= \Pr(Y = 1 \mid X_e = 1)\Pr(X_e = 1) \\ &= R_e^c r_e. \end{aligned}$$

We define now

$$S_e \stackrel{\text{def}}{=} \frac{Z_e - r_e Y}{r_e(1-r_e)} = \frac{Y(X_e - r_e)}{r_e(1-r_e)}$$

(and the corresponding replications $S_e^{(1)}, \ldots, S_e^{(N)}$). We have

$$\begin{aligned} \mathrm{E}(S_e) &= \frac{R_e^c r_e - r_e R}{r_e(1-r_e)} \\ &= \frac{R_e^c - R}{1-r_e} \\ &= \sigma_e \qquad \text{(from (5))}. \end{aligned}$$

This means that, to estimate $\sigma_e$, we can estimate the expectation of the random variable $S_e$ by means of the unbiased estimator

$$\widetilde{\sigma}_e \stackrel{\text{def}}{=} \frac{1}{N}\sum_{n=1}^{N} S_e^{(n)}$$

where $S_e^{(n)}$ is the value of $S_e$ in the $n$th replication of $\mathcal{G}'$.

The performance of an estimator is related to the cost in computing it and to its variance. In particular, the size of the confidence intervals is proportional to the square root of its variance. In the case of $\widetilde{R}$, its standard unbiased estimator can be written as

$$\frac{\widetilde{R}(1-\widetilde{R})}{N-1}.$$

For a given significance level of $1-\varepsilon$, the confidence interval for $\widetilde{R}$ is

$$\left]\ \widetilde{R} - \xi(\varepsilon)\sqrt{\frac{\widetilde{R}(1-\widetilde{R})}{N-1}}\ ,\ \widetilde{R} + \xi(\varepsilon)\sqrt{\frac{\widetilde{R}(1-\widetilde{R})}{N-1}}\ \right[$$

where

$$\xi(\varepsilon) = \Phi^{-1}\left(1 - \frac{\varepsilon}{2}\right), \quad \Phi(x) = \frac{1}{\sqrt{2\pi}}\int_0^x e^{-\frac{t^2}{2}}\,dt$$

(for instance, $\xi(0.05) \approx 1.96$.)

To estimate the variance of $\widetilde{\sigma}_e$ we use

$$\widetilde{\nu}_e \stackrel{\text{def}}{=} \frac{1}{N-1}\sum_{n=1}^{N}(S_e^{(n)} - \widetilde{\sigma}_e)^2$$

which can be written after some algebra

$$\widetilde{\nu_e} = \frac{N}{N-1}\left[\frac{(1-2r_e)\widetilde{Z}_e + r_e^2\widetilde{R}}{r_e^2(1-r_e)^2} - \widetilde{\sigma}_e^2\right]$$

where

$$\widetilde{Z}_e \stackrel{\text{def}}{=} \frac{1}{N}\sum_{n=1}^{N} Z_e^{(n)}.$$

The corresponding confidence interval (level $1-\varepsilon$) for $\sigma_e$ is

$$\left]\ \widetilde{\sigma_e} - \xi(\varepsilon)\sqrt{\frac{\widetilde{\nu_e}}{N}}\ ,\ \widetilde{\sigma_e} + \xi(\varepsilon)\sqrt{\frac{\widetilde{\nu_e}}{N}}\ \right[.$$

### 3.2 A specific algorithm in case of rare events

Assume now the usual situation in which $R \approx 1$. In this case, a naive implementation of the method described in the previous subsection can be difficult or impossible to apply because of its computational cost. Consider instead the following technique, whose inspiration lies in previous work (Rubino 1998, Section 11.5).

Imagine a naive implementation of the previously described approach working the following way. Since the estimators are built from sequences $(Y^{(n)})_{n=1\cdots N}$ and $(X_e^{(n)})_{n=1\cdots N}$ for all edges $e$, or equivalently, from sequences $(Y^{(n)})_{n=1\cdots N}$ and $(Z_e^{(n)})_{n=1\cdots N}$, assume we fill a (huge) table having $N$ rows and $E+1$ columns, the first $E$ of which are associated with the edges of the graph. At row $n$, the value of the column associated with edge $e$ is $X_e^{(n)}$, and the last value of the row at column $E+1$ has value $Y^{(n)}$. Imagine we first fill the table row by row, and then compute from it $\widetilde{R}$ and $\widetilde{\sigma}_e$ for all edge $e$. Now, look at the column associated with edge $e$: this is a Bernoulli sequence of 1s and 0s having length $N$, with a "majority" of 1s (in the rare case, the reliability of line $e$, $r_e$, is supposed to be close to 1). Now, to simplify the presentation, consider $N=\infty$ (an infinite table), and denote by $F_e$ the random variable "first row of the table with value 0 at column $e$". Variable $F_e$ is geometric:

$$\Pr(F_e = f) = r_e^{f-1}(1-r_e), \quad f \geq 1.$$

Denote by $M$ the minimum of the $F_e$s: $M = \min\{F_e,\ e \in \mathcal{E}\}$. Variable $M$ is also geometric, with distribution

$$\Pr(M = m) = r^{m-1}(1-r), \quad m \geq 1,$$

where $r = \prod_{e\in\mathcal{E}} r_e$. The mean of $M$ is $\mathrm{E}(M) = (1-r)^{-1}$.

The proposed method works as follows. First, we sample the $E$ random variables $(F_e)$. Call $f_e$ the value obtained for $F_e$, and call $m = \min\{f_e\}$, the corresponding realization of $M$. Assume that $m > 1$. This means that in rows 1 to $m-1$ we have only 1s in the table, and it implies that without any supplementary sampling (and *a fortiori*, of computation), we have $m-1$ values of the network state and of the $E$ lines for free. At row $m$, we must just perform a DFS (Depth First Search) to see if the network works. Of course, if $m=1$ we come directly to this step. Now, for next step, assume that $F_{e_1} = F_{e_2} = \cdots = F_{e_k} = m < F_e$ if $e \notin \{e_1, e_2, \cdots, e_k\} = \mathcal{E}'$ for some $k$, $1 \leq k \leq E$. For these edges $e_1, e_2, \cdots, e_k$ we sample again from the respective geometric distributions, getting $f'_{e_1}, f'_{e_2}, \cdots, f'_{e_k}$, in order to have, as at the beginning, a new set of values containing for each edge $e$, the next row in the table where that edge is down (that is, value $f_e$ if $e \notin \mathcal{E}'$ and $f_e + f'_e$ for all $e \in \mathcal{E}'$). Then, again the minimum $m'$ of these $E$ values is computed ($m'$ is new line where there is at least one 0 in the first $E$ columns) and the process is repeated. A detailed presentation of the algorithm together with formal proofs can be found in (Rubino 2005).

### 3.3 Complexity analysis

Let us now look at the gain of this method with respect to the algorithm given in previous subsection. First of all, observe that running a direct implementation of the standard Monte Carlo technique described at the beginning of Section 3 for estimating $R$ alone from $N$ samples has average computational complexity proportional to $EN$. This comes from the fact that the sampling cost is proportional to $EN$ and the cost of a DFS is linear in the number of edges. This gives a mean cost proportional to $EN$ of the direct implementation of the standard Monte Carlo technique of estimating $R$. If we write this cost as $\alpha EN$, then using (4) for estimating $\nabla R$ leads to a mean cost $\approx 2\alpha E^2 N$, and using relations (5) or (6) we get a mean cost $\approx \alpha E(E+1)N$.

Implementing the result presented in Subsection 3.1 for estimating the sensitivities, the average computational complexity is simply proportional to $EN$; we can write it as $\approx \beta EN$, with $\beta > \alpha$ for accounting for the overhead with respect to the estimation of $R$ alone.

Finally, consider the technique proposed in Subsection 3.2. The gain, with respect to the previous approach, lies in the fact that (i) the pseudo-random number generator is used only for the geometric distributions and (ii) the DFS is performed just in a (hopefully small) part of the potential set of $N$ samples.

The average cost in sampling the states of the $E$ components is proportional to $\sum_e N/\mathrm{E}(F_e)$, that is, to $N\sum_e(1-r_e)/r_e$. The average number of calls to the DFS procedure is $N/\mathrm{E}(M) = N(1-r)/r$. This is to be compared to $EN$ (which is proportional to both the sampling and DFS costs in the basic algorithm of Subsection 3.2). To be more specific, let us consider a model for which $r_e = 1-\varepsilon$ for all edge $e$, where $\varepsilon \ll 1$ (we are in the rare event case). Then $r \approx 1 - E\varepsilon$. We have $N\sum_e(1-r_e)/r_e = NE\varepsilon/(1-\varepsilon)$ and $N(1-r)/r = N\varepsilon E/(1-\varepsilon E)$. The resulting average complexity of the algorithm is thus proportional to $\varepsilon E^2 N$.

Let us compare the cost of the technique in Subsection 3.2 to the cost of the direct implementation of the result in Subsection 3.1. Dividing the latter by the former, we get a ratio $\approx \gamma(\varepsilon E)^{-1}$. For instance, in the example given in next section, we have $E=18$. If the elementary reliabilities of the edges are $r_e = 0.9999$ for all edges $e$, we have $\varepsilon = 10^{-4}$ and the theoretical analysis says that the second algorithm runs $\approx 500\gamma$ times faster *for the same accuracy* (since both are implementations of the standard estimator, and thus the variances are the same!) See next section for some illustrations on this, showing some cases where $\gamma > 1$.

## 4 Example

Consider the model shown in Figure 2, having 14 nodes and 18 edges.

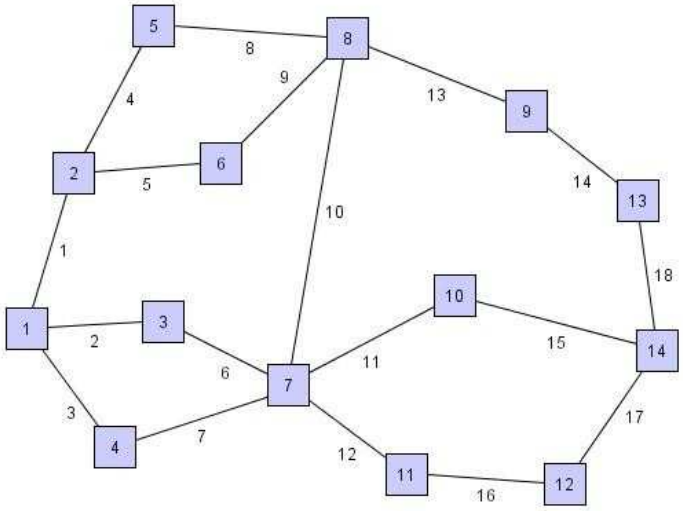


Figure 2: An example for illustrating sensitivity analysis. The source is 1 and the terminal is 14.

The example has been chosen because when $s = 1$ and $t = 14$, the reliability of the model can be easily computed by conditioning with respect to the state of line 10 (that is, by applying the factoring theorem with $r_{10}$ as the *pivot*): the graphs obtained by contracting (respectively by deleting) line 10 in the given model are both $s, t$-series-paralel, which implies that the computation of their reliabilities is straightforward and can even be done Formally. This allows the reader to check the numerical values given by the simulators. In (Rubino 1998) there are some examples having sizes slightly larger and with no hope of even a numerical solution (using any of the numerous combinatorial algorithms available). This is because of the combinatorial explosion in the number of possible vectors $X = (\dots X_e \dots)$.

The network reliability $R$ for $s = 1$ and $t = 14$ is given by the following expressions:

$$\begin{aligned}
\alpha &= r_1(r_4 r_8 + r_5 r_9 - r_4 r_5 r_8 r_9),\\
\beta &= r_{13} r_{14} r_{18},\\
\gamma &= r_2 r_6 + r_3 r_7 - r_2 r_3 r_6 r_7,\\
\delta &= r_{11} r_{15} + r_{12} r_{16} r_{17} - r_{11} r_{15} r_{12} r_{16} r_{17},\\
R &= r_{10} R_c + (1 - r_{10}) R_d, \text{ where}\\
R_c &= (\alpha + \gamma - \alpha\gamma)(\beta + \delta - \beta\delta) \text{ and}\\
R_d &= \alpha\beta + \gamma\delta - \alpha\beta\gamma\delta.
\end{aligned}$$

| $\varepsilon$ | $1 - R$ | $22\,\varepsilon^3$ |
|---|---|---|
| $10^{-2}$ | $0.21846\ 10^{-4}$ | $0.22\ 10^{-4}$ |
| $10^{-3}$ | $0.21985\ 10^{-7}$ | $0.22\ 10^{-7}$ |
| $10^{-4}$ | $0.21998\ 10^{-10}$ | $0.22\ 10^{-10}$ |

Table 1: Some values of the unreliability for the symmetric case in which for all edge $e$ we have $r_e = 1 - \varepsilon$. Third column gives an approximation of $1 - R$ using the concept Of *mincut*. The graph has 22 mincuts with 3 edges each, and any other mincut has at least 4 edges.

| $e$ | $\sigma_e$ | $e$ | $\sigma_e$ | $e$ | $\sigma_e$ |
|---|---|---|---|---|---|
| 1 | 401.460 | 2 | 210.516 | 3 | 210.516 |
| 4 | 7.83316 | 5 | 7.83316 | 6 | 21.0516 |
| 7 | 210.516 | 8 | 7.83316 | 9 | 7.83316 |
| 10 | 17.5902 | 11 | 876.285 | 12 | 581.249 |
| 13 | 583.064 | 14 | 583.064 | 15 | 876.285 |
| 16 | 581.249 | 17 | 581.249 | 18 | 583.064 |

Table 2: Exact numerical values of the sensitivities (multiplied by $10^6$), when for all edge $e$ we have $r_e = 0.99$. Observe the strong impact of the topology on the sensitivities, in spite of the fact that lines are identical. The reliability of the system is $R = 0.99997815$.

For instance, Table 1 gives some (exact) values of $1-R$, the *unreliability* of the system, in the case $r_e = 1-\varepsilon$ for all edge $e$, as a function of $\varepsilon$. Using a well-known approximation arising in this case where all lines have the same reliability, we have $1 - R = \nu_c \varepsilon^c + o(\varepsilon^c)$ where $c$ is the minimal size of a mincut in the graph ($c$ is called the *breadth* of the graph) and $\nu_c$ is the number of mincuts with $c$ edges (see (Rubino 1998, Section 11.6) for details). The third column in Table 1 gives the value of $\nu_c \varepsilon^c$ (in the example, $c = 3$ and $\nu_3 = 22$). This example is intended to illustrate that there are other possible techniques in particular cases, and because this parameter $c$ will be used in a further improvement of the method.

From the polynomial expression of $R$ previously given, the computation of $\nabla R$ is immediate and can be done formally. In Tables 2 and 3 we list the exact values of the sensitivities in two cases. In Table 2, $r_e = 0.99$ for all edge $e$. We see the huge differences that can appear on the sensitivities. For instance, $\sigma_{11}$ is over 100 times bigger than $\sigma_3$. In spite of the fact that the lines are identical (same reliabilities), the topology induces large differences on the importance of each line in the network.

In Table 3 we started from the same graph as in Table 2 and we changed the reliabilities of the two most critical components in that system, lines 11 and 15,

| $e$ | $\sigma_e$ | $e$ | $\sigma_e$ | $e$ | $\sigma_e$ |
|---|---|---|---|---|---|
| 1 | 450.382 | 2 | 210.446 | 3 | 210.446 |
| 4 | 8.78773 | 5 | 8.78773 | 6 | 210.446 |
| 7 | 210.446 | 8 | 8.78773 | 9 | 8.78773 |
| 10 | 68.4540 | 11 | 796.622 | 12 | 5549.62 |
| 13 | 5534.11 | 14 | 5534.11 | 15 | 796.622 |
| 16 | 5549.62 | 17 | 5549.62 | 18 | 5534.11 |

Table 3: Exact numerical values of the sensitivities (multiplied by $10^6$), when $r_e = 0.99$ for all edge $e$ except for lines 11 and 15, where $r_{11} = r_{15} = 0.9$. Observe the strong changes in the sensitivities with respect to the values shown in Table 2. The reliability of the system is $R = 0.99982759$.

from 0.99 to 0.90. In the results, we now see the significant changes that this modification of the elementary reliabilities induces In the relative importance of the lines.

Now, if we run the algorithms derived from both subsections in Section 3 on the graph associated with Table 3, the gain of the second method is slightly larger than the factor given in the complexity analysis. When the reliabilities verify $r_e = 1 - \varepsilon$ for all edge $e$, the gain is a little bit larger than $(18\varepsilon)^{-1}$. For instance, we got speedups higher than 600 for $\varepsilon = 0.0001$. One of the explanations of this is that

the implemented algorithm corresponding to the description given in Subsection 3.2 has a supplementary improvement. Before calling the DFS, the program checks if the number of down lines is less than $c$, the breadth of the graph. In that case, there is no need to run a DFS since, by definition of mincut, the system is necessarily up. Recall that computing the breadth is polynomial in the size of the graph.

As a last illustrative element, consider the model evaluated in Table 3. The elementary reliabilities are not very close to 1 (0.99 or 0.90), leading to a value of $R$ allowing the reader to easily run the algorithm in Subsection 3.1, and even the brute force approaches given at the beginning of Section 3. The gain of the last specific algorithm over the one in Subsection 3.1 is about 6 for $N = 10^7$ samples and about 7 for $N = 10^8$. The respective execution times of the former technique, using a Pentium 4 PC, are of 17s if $N = 10^7$ and 2m30s for $N = 10^8$.

## 5 Conclusions

In this paper, we pointed out the practical importance of sensitivity analysis in network reliability computations and we analyzed their estimations following a Monte Carlo approach. More specifically, we provided a new estimator allowing obtaining the sensitivities as a byproduct of the estimation of the system reliability, and a second result showing that in the rare event case, we can still use the standard estimator with a smarter implementation, which allows to obtain the same accuracy but much more efficiently.

The exploration of other reliability metrics is obviously an immediate possible direction for further research. The basic result presented in Subsection 3.1 can also be extended to other algorithms such as the one proposed in (Khadiri and Rubino 1996) and used in (Cancela, Rubino, and Tuffin 2005) for illustrating new propositions in the analysis of the properties of rare events estimators, so that a detailed analysis of those extensions is the object of current efforts.

## REFERENCES


Ahmad, S. 1982, Apr.. A simple technique for computing network reliability. *IEEE Trans. Reliab.* R-31 (1).

Ball, M., C. Colbourn, and S. Provan. 1995. Chapter 11: Network reliability. In *Handbooks in OR & MS 7*, ed. M. Ball, T. Magnanti, C. Monma, and G. Nemhauser, 673–762. Elsevier Science.

Barlow, B., and F. Proschan. 1975. *Statistical theory of reliability and life testing*. New York: Holt, Rinehart & Winston.

Birnbaum, Z. 1969. On the importance of different components in a multicomponent system. In *Multivariate Analysis II*, ed. P.R.Krishnaiah, 581–592. New York: Academic Press.

Bobbio, A., and A. Premoli. 1982. Fast algorithm for unavailability and sensitivity analysis of series-parallel systems. *IEEE Trans. Reliab.* R-31 (4).

Cancela, H., G. Rubino, and B. Tuffin. 2005. New measures of robustness in rare event simulation. In *Proceedings of the 2005 Winter Simulation Conference*.

Cancela, H., G. Rubino, and M. Urquhart. 2001. An algorithm to compute the all-terminal reliability measure. *OpSearch* 38 (6): 567–579.

Fussell, J. 1975. How to hand-calculate system reliability characteristics. *IEEE Trans. Reliab.* R-24 (3).

Hariri, S., and C. Raghavendra. 1987. Syrel: A symbolic reliability algorithm based on path and cutset methods. *IEEE Trans. on Comput.* C-36 (10): 1224–1232.

Khadiri, M. E., and G. Rubino. 1996. An efficient formulation of the standard Monte Carlo simulation of binary systems reliability. In *2nd International Conference on Monte Carlo and Quasi-Monte Carlo Methods in Scientific Computing*.

Locks, M., and A. Satyarayana. 1986. Network reliability – the state of the art. *IEEE Trans. Reliab.* R-35 (3).

Rubino, G. 1998. Network reliability evaluation. In *State-of-the art in performance modeling and simulation*, ed. K. Bagchi and J. Walrand. Gordon and Breach Books.

Rubino, G. 2005. Sensitivity Analysis in Network Reliability with Monte Carlo. Technical report, INRIA.

Valiant, L. 1979. The complexity of enumeration and reliability problems. *SIAM J. Comput.* 8:410–421.

## AUTHOR BIOGRAPHY

**GERARDO RUBINO** is a senior researcher at INRIA (the French National Institute for Research in Computer Science and Control) where he is the leader of the ARMOR (Architectures and Models of Networks) team at INRIA Unit of Rennes. His research interests are in the quantitative analysis of computer and communication systems, mainly using probabilistic models. He also works on the quantitative evaluation of perceptual quality of multimedia communications over the Internet. He is a member of the IFIP WG 7.3. His e-mail address is <rubino@irisa.fr>, and his Web page is <www.irisa.fr/armor/lesmembres/Rubino/Rubino_en.htm>.