Proceedings of the 2012 Winter Simulation Conference
C. Laroque, J. Himmelspach, R. Pasupathy, O. Rose, and A. M. Uhrmacher, eds.

DEPENDENT FAILURES IN HIGHLY RELIABLE STATIC NETWORKS

Zdravko I. Botev

School of Mathematics and Statistics
The University of New South Wales
Sydney, NSW 2052, AUSTRALIA

Pierre L'Ecuyer

DIRO, Université de Montreal
C.P. 6128, Succ. Centre-Ville
Montréal (Québec), H3C 3J7, CANADA

Bruno Tuffin

INRIA Rennes Bretagne Atlantique
Campus Universitaire de Beaulieu
35042 Rennes Cedex, FRANCE

ABSTRACT

Static network reliability models typically assume that the failures of their components are independent. This assumption allows for the design of efficient Monte Carlo algorithms that can estimate the network reliability in settings where it is a rare-event probability. Despite this computational benefit, independent component failures is frequently not a realistic modeling assumption for real-life networks. In this article we show how the splitting methods for rare-event simulation can be used to estimate the reliability of a network model that incorporates a realistic dependence structure via the Marshal-Olkin copula.

1 INTRODUCTION

The problem of static network reliability modeling and estimation has a wide range of applications in communication and transportation (Cancela, El Khadiri, and Rubino 2009; Gertsbakh and Shpungin 2010). The *static reliability* of a network (or graph) is a quantitative measure of the network's ability to provide service. It is defined as the probability that a given set of nodes in the graph are connected by operational links, where each link of the graph is operational with a given probability, called the reliability of the link. Equivalently, network designers are interested in the *unreliability*, defined as the complementary probability.

The exact calculation of network reliability is a #P-complete computational problem (Ball and Provan 1982; Colbourn 1987). This is why for large networks Monte Carlo techniques are indispensable. Also it is well known (Gertsbakh and Shpungin 2010) that in highly reliable networks the Crude Monte Carlo (CMC) method is impractical, because the probability of network failure is a rare-event probability. The search for efficient Monte Carlo algorithms for such graphs has resulted in a number of variance reduction methods. Among the most prominent ones are conditional Monte Carlo approaches (Cancela and El Khadiri 2003; Cancela et al. 2009; Elperin et al. 1991; Gertsbakh and Shpungin 2010; Lomonosov and Shpungin 1999), approximate zero-variance importance sampling (L'Ecuyer et al. 2011), and combinations of these, see Cancela et al. (2010). For a survey of some of these methods see Cancela, El Khadiri, and Rubino (2009).

A salient feature of all of the above Monte Carlo methods is that they assume that the components of the network fail independently. In this paper we consider the important situation where the link failures are dependent.

While there are good algorithms designed for special types of dependent component failures, these can only cope with small networks. These algorithms are typically deterministic (capable of generating a symbolic reliability expression) and inefficient for large highly reliable networks, because they rely on

978-1-4673-4781-5/12/$31.00 ©2012 IEEE

cutsets/pathsets (Fard and Lee 1999; Netes and Filin 1996; Nahman 1992; Lin and Yang 2011) or graph factorizations (Theologou and Carlier 1991; Kuo, Yeh, and Lin 2007; Biegel 1977; Ahmad 1990; Chen and Yuang 1996; Ghosh and Singh 1993) whose numbers grow exponentially with the size of the network.

Our paper contributes to the alleviation of this computational problem by showing how the splitting method for rare-event simulation can estimate efficiently the reliability of large highly reliable networks with realistic copula dependence. The aim is to both use one of the most relevant copula models to account for real-life component failures and at the same time provide an efficient algorithm for reliability estimation in networks driven by such copula models. While there are other possible choices, in this paper we model the dependencies among network components using the realistic Marshal-Olkin shock model. The Marshall-Olkin copula has been used in risk management in finance (Embrechts, Lindskog, and McNeil 2003). Note that other types of realistic dependence include *cascading failures* models (Iyer et al. 2009; Buldyrev et al. 2010), which we do not consider here in the rare-event setting.

Modeling dependence in static networks has been considered previously, but the existing proposals either do not offer a viable algorithm to estimate the reliability in the rare-event setting, or do not capture the dependence in a realistic way (Singpurwalla 2002; Ram and Singh 2009; Walter et al. 2009; Botev et al. 2012). For those proposals that consider the rare-event setting, the dependence is typically modeled using a Gaussian, Pareto, or Weibull copula, which do not easily account for rare shock events that can knock out a multitude of network components simultaneously. As a result, such simple copulas tend to overestimate the real-life reliability of a network. In contrast, it is well known that the more complex Marshall-Olkin copula meets a number of desirable criteria that make it a good candidate for modeling simultaneous component failures due to a shock event.

The rest of the paper is organized as follows. In Section 2.1 we introduce the the graph evolution approach to modeling static networks as advocated by Elperin, Gertsbakh and Lomonosov, see Lomonosov and Shpungin (1999), Elperin, Gertsbakh, and Lomonosov (1991). This is followed by Section 2.2, in which we explain how we combine the Marshall-Olkin copula with the graph evolution approach. Once we have selected a satisfactory copula model, we estimate the corresponding reliability using a modified version of the splitting method of Kahn and Harris (1951). In Section 3 we review the splitting method for rare-event probability estimation and provide implementation details. Finally, in Section 4 we give a numerical example and an application of the copula model to networks in which the nodes as well as the links fail. This is followed by a concluding section discussing possible directions for future research.

2 STATIC NETWORK MODEL

We begin by defining the prototypical mathematical model for a static network. Suppose we are given the graph $\mathscr{G} = (\mathscr{V}, \mathscr{E})$ with a set of nodes/vertexes $\mathscr{V}$ and edges/links $\mathscr{E}$. Associated with each edge i is a Bernoulli random variable X_i denoting whether the edge is operational ($X_i = 1$) or failed ($X_i = 0$). If we label all edges from 1 to $m = |\mathscr{E}|$, then $\mathbf{X} = (X_1, \ldots, X_m)$ represents the configuration of the network, showing which edges are operational and which are failed. Typically, it is assumed that $X_1, \ldots, X_m$ are independent and $\mathbb{P}(X_i = 0) = u_i$, $i = 1, \ldots, m$, where u_i is the unreliability of edge i. A subset of nodes $\mathscr{V}_0 \subset \mathscr{V}$ is selected a priori and the network (or graph) is said to be *operational* if all nodes in $\mathscr{V}_0$ are connected to each other by at least one path or tree comprising of operational edges. Let $\Psi(\mathbf{x}) = 1$ when the network is operational, and $\Psi(\mathbf{x}) = 0$ otherwise. This function Ψ is referred to as the *structure function* of the graph (Barlow and Proschan 1975). An important special case is the *two-terminal network reliability* problem, where $\mathscr{V}_0$ contains only two nodes, $\mathscr{V}_0 = \{v_0, v_1\}$, and $\Psi(\mathbf{x}) = 1$ if and only if there is a path between v_0 and v_1. For example, in the dodecahedron graph on Figure 1 we have $\mathscr{V}_0 = \{1, 20\}$ and it is operational when nodes 1 and 20 are connected by a path of working edges. Another special case of interest is the *all-terminal network reliability* problem, where $\mathscr{V}_0 = \mathscr{V}$, so $\Psi(\mathbf{x}) = 1$ if and only if all nodes are connected.

The unreliability of the network $\mathscr{G}$ is defined as the probability that the nodes in $\mathscr{V}_0$ are disconnected, that is,

$$u = \mathbb{P}(\Psi(\mathbf{X}) = 0) \,.$$

In the next section we give a different formulation of the reliability estimation problem, which views the static network as a snapshot of a dynamic one at a given point in time.

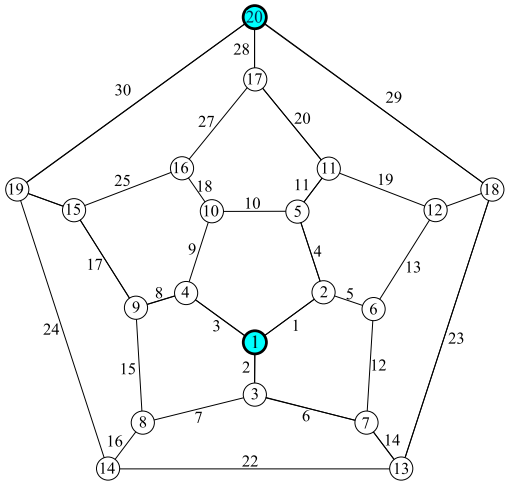

Figure 1: A dodecahedron graph with 20 nodes and 30 links (all labeled). The destination nodes (1 and 20) are shaded.

2.1 Graph Evolution Approach

In our approach we follow the graph evolution formulation in Elperin, Gertsbakh, and Lomonosov (1991), Elperin, Gertsbakh, and Lomonosov (1992), Lomonosov and Shpungin (1999) and assume that the i-th link is operational for y_i units of time before it finally fails. In other words, y_i is the lifetime of the i-th component of the network. For the time being we assume that the nodes are perfect and do not fail. The configuration of the network is thus described by the lifetimes $\mathbf{y} = (y_1, \ldots, y_m)$, where m is the number of edges. We can signify whether the i-th edge is still alive at time γ by keeping track of the binary variable $x_i(\gamma) = \mathbb{I}\{y_i > \gamma\}$, where $\mathbb{I}$ is the indicator function. If $x_i(\gamma) = 1$, then the i-th link is still alive or operational at time γ, and if $x_i(\gamma) = 0$, then the i-th link has failed at time γ. We let $\mathscr{G}(\mathbf{x}(\gamma))$, with $\mathbf{x}(\gamma) = (x_1(\gamma), \ldots, x_m(\gamma))$, denote the subgraph of $\mathscr{G}$ that contains only the edges i which are still alive at time γ, that is, the edges for which $y_i > \gamma$ or $x_i(\gamma) = 1$. The network is said to be *operational* at time γ if $\Psi(\mathbf{x}(\gamma)) = 1$.

The lifetime of link i is modeled as a random variable Y_i with a lifetime distribution $F_i(y) = \mathbb{P}(Y_i \leqslant y)$ such that $F_i(0) = 0$ and, in the case of independent failures, $F_i(1) = u_i$. In other words, the probability that the i-th edge is not alive at time 1 is u_i, and the relationship between the Bernoulli indicator X_i and the lifetime Y_i is $\mathbb{P}(Y_i \leqslant 1) = \mathbb{P}(X_i = 0) = u_i$, where u_i is the unreliability of edge i. In this case, the system's lifetime configuration is described by $\mathbf{Y} = (Y_1, \ldots, Y_m)$. The natural interpretation is that at time 0 all the links are in perfect working condition, then they start to age, and after a working life of Y_i units of time, the i-th edge fails. Gradually, more and more edges fail, until finally there is no path connecting the destination nodes and the network has failed. Elperin, Gertsbakh, and Lomonosov (1991) call this the *destruction process*. In the destruction process the operational state of each link at time γ is a random binary vector $\mathbf{X}(\gamma) = (X_1(\gamma), \ldots, X_m(\gamma))$ with $\mathbf{X}(1) \equiv \mathbf{X}$. Thus, the network unreliability u can be written as:

$$u = \mathbb{P}(\Psi(\mathbf{X}(1)) = 0) = \mathbb{P}(S(\mathbf{Y}) < 1),$$

where $S(\mathbf{Y})$ is the last time the network is operational, that is,

$$S(\mathbf{Y}) = \sup\{\gamma \geqslant 0 : \Psi(\mathbf{X}(\gamma)) = 1\}\,.$$

We evaluate $S(\mathbf{Y})$ for a given $\mathbf{Y}$ using the following straightforward depth first search algorithm.

Algorithm 1 : Evaluating $S(\mathbf{Y})$

Require: lifetimes $\mathbf{Y}$

Let $\pi = (\pi_1, \ldots, \pi_m)$ be the permutation of the edges $1, \ldots, m$ such that

$$Y_{\pi_1} < Y_{\pi_2} < \cdots < Y_{\pi_m} .$$

Let $b = 1$ and consider $\mathscr{G}(\mathbf{X}(Y_{\pi_b}))$, in which edges $\pi_1, \ldots, \pi_b$ are failed and $\pi_{b+1}, \ldots, \pi_m$ are working.
while $\Psi(\mathbf{X}(Y_{\pi_b})) = 1$ (verified using depth first search, for example) **do**
$b \leftarrow b + 1$
return $S(\mathbf{Y}) = Y_{\pi_{b-1}}$ as the last time the network is operational.

Crude Monte Carlo estimates u by generating n independent realizations $\mathbf{Y}_1, \ldots, \mathbf{Y}_n$ of $\mathbf{Y}$, and taking the average of the n replicates of $\mathbb{I}\{S(\mathbf{Y}) < 1\} = 1 - \Psi(\mathbf{X}(1))$ as an estimator of u. The square relative error (the relative variance) of this estimator of u is

$$\frac{\mathrm{Var}(\frac{1}{n}\sum_{i=1}^n \mathbb{I}\{S(\mathbf{Y}_i) < 1\})}{u^2} = \frac{\mathrm{Var}(\mathbb{I}\{S(\mathbf{Y}_i) < 1\})}{n\,u^2} = \frac{u - u^2}{n\,u^2} = \frac{r}{n\,u},$$

which increases to infinity when $u \to 0$. For highly reliable networks u is very small so we have a rare-event probability, and n must be very large to get a meaningful estimator. For example, if $u = 10^{-10}$, we need $n > 10^{12}$ to obtain a relative error below 10%. This inefficiency is the reason why the eclectic variance reduction methods surveyed in the introduction have been proposed.

2.2 Lifetime Shock Model

One of the reasons for working within the graph evolution framework above is that since each link is assigned a lifetime, we will be able to use lifetime shock models, which can capture dependencies amongst random lifetimes (Singpurwalla 2006). An example of such a suitable model is the Marshal-Olkin copula.

For example, the bivariate Marshal-Olkin copula with exponential marginals is the only bivariate distribution satisfying the desirable bivariate lack of memory property. If Y_1 and Y_2 denote the exponential lifetimes of the two interdependent components, then these lifetimes can be defined in terms of three independent exponential random variables $Z_1, Z_2, Z_{1,2}$:

$$Y_1 = \min\{Z_1, Z_{1,2}\}, \qquad Y_2 = \min\{Z_2, Z_{1,2}\}$$

with intensities $\lambda_1 = -\ln(1-u_1)$, $\lambda_2 = -\ln(1-u_2)$, and $\lambda_{1,2} = -\ln(1-u_{1,2})$, respectively. Here the parameter $\lambda_{1,2}$ captures the intensity of the interdependence between components 1 and 2 so that the probability of occurrence of the common shock event is $\mathbb{P}(Z_{1,2} < 1) = u_{1,2}$. The joint survival function is

$$\mathbb{P}(Y_1 > y_1, Y_2 > y_2) = \mathbb{P}(Z_1 > y_1)\mathbb{P}(Z_2 > y_2)\mathbb{P}(Z_{1,2} > \max\{y_1, y_2\}) = \mathrm{e}^{-\lambda_1 y_1 - \lambda_2 y_2 - \lambda_{1,2}\max\{y_1, y_2\}} ,$$

from where we can conclude that the joint density of Y_1 and Y_2 is discontinuous, and the marginal distributions of the lifetimes Y_1 and Y_2 are exponential and satisfy the memoryless property:

$$\mathbb{P}(Y_1 > y_1 + a, Y_2 > y_2 + a \,|\, Y_1 > a, Y_2 > a) = \mathbb{P}(Y_1 > y_1, Y_2 > y_2) .$$

For a network with m lifetimes $Y_1, \ldots, Y_m$ the copula model generalizes as follows. Let $\mathscr{S}$ be a subset of $\{1, \ldots, m\}$. Thus, without any restrictions the size of $\mathscr{S}$ is $2^m - 1$, or the set of all subsets. Each subset

$\mathbf{s} \in \mathscr{S}$ represents a collection of components that can be knocked out simultaneously due to a single shock event. For example, in the bivariate case there are three subsets $\mathbf{s}_1 = \{1\}$, $\mathbf{s}_2 = \{2\}$, and $\mathbf{s}_3 = \{1,2\}$, giving $\mathscr{S} = \{\mathbf{s}_1, \mathbf{s}_2, \mathbf{s}_3\}$. The lifetimes of the network can be expressed in terms of $|\mathscr{S}|$ independent exponential random variables:

$$Y_i = \min_{\mathbf{s}: i \in \mathbf{s}} Z_{\mathbf{s}}, \quad i = 1, \ldots, m \,,$$

where $Z_{\mathbf{s}}$ are independent with densities $\lambda_{\mathbf{s}} \mathrm{e}^{-\lambda_{\mathbf{s}} z}$, $z \geqslant 0$ for all $\mathbf{s} \in \mathscr{S}$. Given the set $\mathscr{S}$ and the lifetime intensities $\lambda_{\mathbf{s}} = -\ln(1-u_{\mathbf{s}})$, $\mathbf{s} \in \mathscr{S}$, the crude Monte Carlo estimator of u consists of the average of n replications of the indicator:

$$\mathbb{I}\{S(\mathbf{Y}) < 1\}, \qquad Y_i = \min_{\substack{\mathbf{s}: i \in \mathbf{s} \\ \mathbf{s} \in \mathscr{S}}} Z_{\mathbf{s}}, \text{ for all } i = 1, \ldots, m \,,$$

where the $Z_{\mathbf{s}}$ are independent and exponentially distributed such that $\mathbb{P}(Z_{\mathbf{s}} < 1) = 1 - \mathrm{e}^{-\lambda_{\mathbf{s}}} = 1 - \mathrm{e}^{\ln(1-u_{\mathbf{s}})} = u_{\mathbf{s}}$.

Before continuing with the splitting method for the estimation of u, we switch to working with normally distributed lifetimes, instead of exponentially distributed ones. We do this to make the application of the hit-and-run Markov chain algorithm in Section 3 easier. The hit-and-run sampler is most suitable when sampling from spherically symmetric distributions on a restricted set (Chen and Schmeiser 1996). If the Y_is are expressed in terms of normally distributed (as opposed to exponentially distributed) lifetimes:

$$Y_i = \min_{\substack{\mathbf{s}: i \in \mathbf{s} \\ \mathbf{s} \in \mathscr{S}}} Z_{\mathbf{s}}, \qquad Z_{\mathbf{s}} \sim \mathsf{N}(\mu_{\mathbf{s}}, 1), \quad \mu_{\mathbf{s}} \stackrel{\text{def}}{=} 1 - \Phi^{-1}(u_{\mathbf{s}}), \text{ independently for all } \mathbf{s} \in \mathscr{S} \,, \tag{1}$$

where Φ^{-1} is the inverse of the cdf of the normal distribution, then the distributional properties of $X_i(1) = \mathbb{I}\{Y_i < 1\}$, $i = 1, \ldots, m$ do not change compared with the exponentially distributed case. For example, the probability of occurrence of each of the shock events remains unchanged: $\mathbb{P}(Z_{\mathbf{s}} < 1) = u_{\mathbf{s}}$ for all $\mathbf{s} \in \mathscr{S}$.

Finally, if vector $\mathbf{Z} = (Z_{\mathbf{s}_1}, Z_{\mathbf{s}_2}, \ldots)$ collects all the variables $\{Z_{\mathbf{s}}, \mathbf{s} \in \mathscr{S}\}$ and $\mathbf{Y}$ is completely determined from $\mathbf{Z}$ via (1), then we can introduce the shorthand notation $S^*(\mathbf{Z}) \equiv S(\mathbf{Y})$ and simply write the crude Monte Carlo estimator as

$$\frac{1}{n} \sum_{k=1}^{n} \mathbb{I}\{S^*(\mathbf{Z}_k) < 1\}, \qquad \mathbf{Z}_1, \ldots, \mathbf{Z}_n \overset{\text{iid}}{\sim} \mathsf{N}(\boldsymbol{\mu}, I) \,,$$

where $\boldsymbol{\mu} \stackrel{\text{def}}{=} (\mu_{\mathbf{s}_1}, \mu_{\mathbf{s}_2}, \ldots)$ collects all the parameters of the shock random variables, and I is the $n \times n$ identity covariance matrix.

3 GENERALIZED SPLITTING FOR RELIABILITY ESTIMATION

To estimate the reliability of a network under the Marshall-Olkin copula we use the Generalized Splitting (GS) method described in Botev et al. (2012) (which is an adaptation of the splitting method of Kahn and Harris). Here we use the splitting procedure with the following important difference. In Botev et al. (2012) we use the construction process, in which all components are initially failed and $\mathbf{Y}$ is a vector of repair times indicating the time at which each component of the network is repaired and becomes operational. The network is thus gradually "constructed" over time and we wish to estimate $u = \mathbb{P}(S(\mathbf{Y}) > 1)$, where $S(\mathbf{Y})$ is interpreted as the first time the network becomes operational, given the repair times $\mathbf{Y}$. In contrast, in this article we use the destruction process, in which all components are initially operational and $\mathbf{Y}$ is a vector of lifetimes indicating the time at which each component of the network fails. The network is hence gradually "destroyed" over time and we wish to estimate $u = \mathbb{P}(S(\mathbf{Y}) < 1)$, where, as mentioned previously, $S(\mathbf{Y})$ is the last time the network is operational, given the destruction over time. The reason for using the destruction process instead of the creation process is so that the network model fits the joint failures from the Marshal-Olkin copula.

With this modification, the GS algorithm will read as follows. We select an integer $s \geqslant 2$, called the *splitting factor* and another integer $\tau > 0$. Then we select intermediate levels $\infty = \gamma_0 > \gamma_1 > \gamma_2 > \cdots > \gamma_\tau = 1$, for which

$$\begin{aligned} \rho_t \stackrel{\text{def}}{=} \mathbb{P}(\Psi(\mathbf{X}(\gamma_t)) = 0 \,|\, \Psi(\mathbf{X}(\gamma_{t-1})) = 0) &= \mathbb{P}(S(\mathbf{Y}) < \gamma_t \,|\, S(\mathbf{Y}) < \gamma_{t-1}) \\ &= \mathbb{P}(S^*(\mathbf{Z}) < \gamma_t \,|\, S^*(\mathbf{Z}) < \gamma_{t-1}) \approx 1/s \end{aligned} \tag{2}$$

for $t = 1, \ldots, \tau$, except for ρ_τ, which can be larger than $1/s$. These γ_t represent the levels of the splitting algorithm and τ is the number of levels. Good values for τ and $\{\gamma_t\}$ can be estimated by an (independent) adaptive pilot algorithm, as explained in the Appendix. Botev et al. (2012) argue that $s = 2$ is a good choice that yields satisfactory empirical results. Thus, we will use $s = 2$ for the rest of the paper.

For each level γ_t, we run a hit-and-run (Kroese, Taimre, and Botev 2011, Page 240) Markov chain $\{\mathbf{Z}_{t,j},\, j \geq 0\}$ having a stationary density equal to the density of $\mathbf{Z}$ conditional on $S^*(\mathbf{Z}) < \gamma_t$. We can write this stationary density as

$$f_t(\mathbf{z}) \propto \mathbb{I}\{S^*(\mathbf{z}) < \gamma_t\} \prod_{\mathbf{s} \in \mathscr{S}} \mathrm{e}^{-\frac{1}{2}(z_\mathbf{s} - \mu_\mathbf{s})^2}, \quad t = 0, \ldots, \tau, \tag{3}$$

where by convention f_0 is the unconditional density of $\mathbf{Z}$. The transition kernel density of the hit-and-run Markov chain, which is the density of the next state $\mathbf{Z}_{t,j}$ conditional on the current state $\mathbf{Z}_{t,j-1}$, is denoted by $\kappa_t(\cdot \,|\, \mathbf{Z}_{t,j-1})$ and defined implicitly via the following algorithm.

Algorithm 2 : Transition density $\kappa_t(\cdot \,|\, \mathbf{z}_{t,j-1})$ defined via hit-and-run sampling

Require: Initial state $\mathbf{Z}_{t,j-1}$ such that $S^*(\mathbf{Z}_{t,j-1}) < \gamma_t$ and a positive integer β.

```
Z_0 ← Z_{t,j-1}
for i = 1, ..., β do
    generate a vector d uniformly distributed over the m-dimensional unit sphere
    generate a random scale Λ ~ N((μ − Z_{i-1}) · d, 1)    // here · denotes the vector dot product
    if S*(Z_{i-1} + Λd) < γ_t then
        Z_i ← Z_{i-1} + Λd
    else
        Z_i ← Z_{i-1}
return Z_{t,j} ← Z_β.
```

The indentation in the algorithm above demarcates the scope of the **if**, **else**, and **for** statements. Note that to evaluate $S^*(\mathbf{Z})$ we simply determine $\mathbf{Y}$ from $\mathbf{Z}$ and use Algorithm 1 to compute $S(\mathbf{Y}) = S^*(\mathbf{Z})$. In the algorithm above β is a positive integer that can be 1. However, a higher value for β reduces the Markov chain dependence between the input state $\mathbf{Z}_{t,j-1}$ and the output state $\mathbf{Z}_{t,j}$. In our simulations we use $\beta = 30$. Note that while there are many possibilities for constructing a Markov chain with stationary density (3), here we use the hit-and-run Markov chain, because it yields simple updating rules, regardless of the Marshall-Olkin dependence structure.

Generating $\mathbf{Z}$ conditional on $S^*(\mathbf{Z}) < \gamma_0$ is the same as generating it via (1). If a generated state $\mathbf{Z}$ satisfies $S^*(\mathbf{Z}) < \gamma_1$, then its distribution is obviously the distribution of $\mathbf{Z}$ conditional on $S^*(\mathbf{Z}) < \gamma_1$, so that the underlying $\mathbf{Z}$ has density f_1. At the t-th stage, if the Markov chain starts from a state having density f_{t-1} and evolves according to the kernel $\kappa_{t-1}(\cdot \,|\, \mathbf{Z}_{t-1,j-1})$, then each visited state also has density f_{t-1} — the stationary density for the kernel κ_{t-1}. With this in mind, the GS algorithm reads as follows.

Algorithm 3 : generalized splitting algorithm; returns W, an unbiased estimate of u

Require: $s, \tau, \gamma_1, \ldots, \gamma_\tau, \mathscr{S}, \boldsymbol{\mu}$ // use the adaptive algorithm in Appendix to get splitting levels

$\mathscr{Z}_1 \leftarrow \emptyset$

for $j = 1 \ldots, s$ **do**

Let $\mathbf{Z}_j = (Z_{\mathbf{s}_1}, Z_{\mathbf{s}_2}, \ldots)$, where $Z_{\mathbf{s}} \sim \mathsf{N}(\mu_{\mathbf{s}}, 1)$ for all $\mathbf{s} \in \mathscr{S}$ independently.

if $S^*(\mathbf{Z}_j) < \gamma_1$ **then**

add $\mathbf{Z}_j$ to $\mathscr{Z}_1$ // Set of states $\mathbf{Z}$ that have reached level γ_1

for $t = 2, \ldots, \tau$ **do**

$\mathscr{Z}_t \leftarrow \emptyset$ // set of states $\mathbf{Z}$ that have reached the level γ_t; initially empty

for all $\mathbf{Z}_0 \in \mathscr{Z}_{t-1}$ **do**

for $j = 1, \ldots, s$ **do**

sample $\mathbf{Z}_j$ from the density $\kappa_{t-1}(\cdot \mid \mathbf{Z}_{j-1})$

if $S^*(\mathbf{Z}_j) < \gamma_t$ **then**

add $\mathbf{Z}_j$ to $\mathscr{Z}_t$

return $W \leftarrow |\mathscr{Z}_\tau| / s^\tau$ as an unbiased estimate of the unreliability u.

In the script above, $\mathscr{Z}_t$ stands for the collection of vectors $\mathbf{Z}$, which yield lifetimes $\mathbf{Y}$ that have reached the level γ_t. Note that Algorithm 3 states the procedure with a single starting chain (trajectory) and gives the unbiased estimator W of the unreliability u. In practice, we run n times Algorithm 3 to obtain n independent realizations $W_1, \ldots, W_n$ and deliver the estimator

$$\widehat{u} = \frac{1}{n} \sum_{i=1}^{n} W_i, \tag{4}$$

with estimated relative error $\widehat{\sigma}_n / (\widehat{u}\sqrt{n})$, where $\widehat{\sigma}_n^2 = \frac{1}{n}\sum_{i=1}^n (W_i - \widehat{u})^2$. In the Appendix we show that under two idealizing assumptions, this estimator is logarithmically efficient as the unreliability u goes to zero.

4 NUMERICAL EXPERIMENTS

Now that we have a computational tool for estimating the reliability under the Marshal-Olkin copula, we can consider the types of dependencies that we wish to model. First note that if $\mathscr{S}$ is unrestricted and we consider all possible interactions among the component failures, then the number of parameters in the Marshal-Olkin copula grows exponentially with the number of components. For a relatively small network with a mere 100 components the number of possible interactions is already above $2^{100} - 1 \approx 10^{30}$, and thus computationally unmanageable. Thus, to make the model scalable, one has to restrict the size of $\mathscr{S}$ and consider only subsets of possible dependencies. Which subsets are most suitable depends on the particular modeling requirements and will in practice be determined on a case by case basis. We now give an example of such specific modeling.

One of the common assumptions in static network models is that the nodes do not fail and hence all failures are (independent) link failures (Aggarwal, Gupta, and Misra 1975; L'Ecuyer, Saggadi, and Tuffin 2011; L'Ecuyer and Tuffin 2011). In reality nodes also fail, exacerbating the reliability of the network, and we must take their fallibility into account. Node failure can be elegantly accounted for by observing that the failure of a node is equivalent to the simultaneous failure of the links *incident* to that node (Aggarwal, Gupta, and Misra 1975; Fard and Lee 1999; Nahman 1992).

For example, consider the dodecahedron network, a popular benchmark problem, in Figure 1. The failure of node 1 has the same effect as the simultaneous (due to a shock event) failure of links $1, 2, 3$. Thus, by making the links incident to each node dependent on a common shock event, we can account for the node failures and avoid overestimating the reliability of the network.

As a numerical example consider the dodecahedron network with fallible nodes. Since the degree of connectivity of this network is 3 (meaning that there are 3 links emanating from each of the 20 nodes),

the copula model has 20 parameters describing the joint failure of all triplets of components incident to a node. For example, some of these triplets in Figure 1 are $(2,7,6),(6,12,14),(22,14,23)$, corresponding to nodes $3,7$, and 13, respectively. Taking into account the failure of each of the 30 links individually, we get that the size of $\mathscr{S}$ is $20+30=50$. Thus in this setting the number of parameters in the copula is always under control. Table 1 shows the reliability of the dodecahedron network under the perfect and imperfect nodes assumption.

Table 1: Reliability of dodecahedron network with and without node failures. Here $\mathscr{V}_0 = \{1,20\}$ and $n = 10^4$.

	imperfect nodes			perfect nodes	
$u_{\mathbf{s}}$	$\widehat{u}$	estimated rel. error	CPU time min.	$\widehat{u}$	estimated rel. error
10^{-1}	0.15	1.1%	5	0.0028	1.3%
10^{-2}	0.0014	2.1%	25	2.05×10^{-6}	1.9%
10^{-3}	5.40×10^{-5}	3.1%	41	2.02×10^{-9}	2.4%
10^{-4}	4.72×10^{-6}	3.6%	47	2.01×10^{-12}	2.8%
10^{-5}	4.52×10^{-7}	4%	64	1.98×10^{-15}	3.1%
10^{-6}	4.59×10^{-8}	3.5%	65	1.96×10^{-18}	3.5%

While in the imperfect node case $|\mathscr{S}| = 50$, in the perfect node case $\mathscr{S} = \{1,2,3,\ldots,30\}$ (the copula reduces to the independent component failure case). In both cases $u_{\mathbf{s}} = \mathbb{P}(Z_{\mathbf{s}} < 1)$ is the same for all $\mathbf{s} \in \mathscr{S}$, (which means that the node failure probability is the same as the link failure probability) and $\widehat{u}$ and its estimated relative error are obtained via (4) with $n = 10^4$. Note that the relative error grows slowly when the network becomes more and more reliable.

As expected from Table 1 we can see that the reliability of the network is much lower when the nodes are considered imperfect and having the same reliability as a link. This example shows that node failures are a special case of the more general Marshall-Olkin copula for dependent component failures.

5 CONCLUSIONS

Network reliability computation requires smart time-saving Monte Carlo simulation strategies even under the simplifying assumption of independent component failures. When the independence assumption is relaxed the complexity of reliability estimation becomes more challenging. In this article we have shown how the (rare-event) probability of failure of highly reliable static binary networks governed by a Marshall-Olkin copula model can be estimated via the GS algorithm — a static version of the splitting method of Kahn and Harris.

As future research we intend to consider the Network Planning Problem (NPP). The objective of the NPP is to optimally purchase a collection of links, subject to a fixed budget, so as to maximize the network reliability. We intend to consider this NP-hard integer optimization problem under the additional complication of dependent link failures.

The dependence model considered here also makes analysis of stochastic flow networks with rare-event effects more challenging. In stochastic flow networks one is interested in the (rare-event) probability that the network capacity exceeds a given network demand, where instead of having the i-th link assigned a random lifetime, each link has a (discrete or continuous) random flow capacity. The splitting approach presented here may also allow practitioners to handle stochastic flow networks with dependent link capacities.

ACKNOWLEDGMENTS

This work has been supported by an NSERC-Canada Discovery Grant and a Canada Research Chair to the second author, and INRIAs associated team MOCQUASIN to second and third authors.

APPENDIX

Here we briefly present the pilot splitting algorithm used to determine τ and $\gamma_1,\ldots,\gamma_\tau$. This algorithm is an modification of the one given in Botev et al. (2012). We have tailored the algorithm to the destruction process, instead of the construction process.

Suppose we are given the splitting factor $s=2$. Initially, we generate $m\times s$ independent states $\mathbf{Z}$ from $\mathsf{N}(\boldsymbol{\mu},I)$, and determine a threshold parameter γ_1 so that exactly m of them have $S^*(\mathbf{Z})\geqslant\gamma_1$. Then at each step t, for $t=2,3,\ldots$, we run for s steps the hit-and-run Markov chain in Algorithm 2 with stationary density (3) from each of those m states $\mathbf{Z}$ for which $S^*(\mathbf{Z})\geqslant\gamma_{t-1}$. This gives another $m\times s$ states and we select a parameter γ_t so that exactly m of them have $S^*(\mathbf{Z})\geqslant\gamma_t$. This is done until $\gamma_t\leq 1$ for some t. Then τ is set to this t and we put $\gamma_\tau=1$. This iterative procedure is summarized in the following Algorithm 4.

Algorithm 4 Adaptive splitting sampler.

Require: $\mathscr{S},\boldsymbol{\mu}$ and splitting factor $s=2$
  $\mathscr{Z}_1\leftarrow\emptyset$
  for $i=1$ to $m\times s$ **do**
    generate a vector $\mathbf{Z}\sim\mathsf{N}(\boldsymbol{\mu},I)$ and add it to $\mathscr{Z}_1$
  sort the elements of $\mathscr{Z}_1$ by decreasing order of $S^*(\mathbf{Z})=S(\mathbf{Y})$, say $\mathbf{Z}_{(1)},\ldots,\mathbf{Z}_{(m\times s)}$
  $\gamma_1\leftarrow[S^*(\mathbf{Z}_{(m)})+S^*(\mathbf{Z}_{(m+1)})]/2$
  $t\leftarrow 1$
  while $\gamma_t\geqslant 1$ **do**
    $t\leftarrow t+1$
    $\mathscr{Z}_{t-1}\leftarrow\{\mathbf{Z}_{(1)},\ldots,\mathbf{Z}_{(m)}\}$ // retain only the best performing m elements from $\mathscr{Z}_{t-1}$
    $\mathscr{Z}_t\leftarrow\emptyset$
    for all $\mathbf{Z}_0\in\mathscr{Z}_{t-1}$ **do**
      for $j=1$ to s **do**
        sample $\mathbf{Z}_j$ from the hit-and-run sampler in Algorithm 2 and add it to $\mathscr{Z}_t$
    sort the elements of $\mathscr{Z}_t$ by decreasing order of $S^*(\mathbf{Z})$, say $\mathbf{Z}_{(1)},\ldots,\mathbf{Z}_{(m\times s)}$
    $\gamma_t\leftarrow\max\{[S^*(\mathbf{Z}_{(m)})+S^*(\mathbf{Z}_{(m+1)})]/2,1\}$
    $\tau\leftarrow t$
  return $\tau,\gamma_1,\ldots,\gamma_\tau$

In this algorithm, $\mathscr{Z}_t$ denotes a set of vectors $\mathbf{Z}$ for which $S^*(\mathbf{Z})\geqslant\gamma_{t-1}$. When this set contains $m\times s$ elements, we sort it to retain the m vectors having the smallest value of $S^*(\mathbf{Z})$, and we remove the other vectors from this set. The threshold parameter γ_t is placed midway between the m-th and the $(m+1)$-th smallest values of $S^*(\mathbf{Z})$.

A IDEAL CASE ANALYSIS OF GS ALGORITHM

We present an analysis of the asymptotic performance of the GS algorithm under two idealizing assumptions, which hold only approximately in practice. The first assumption is that the hit-and-run Markov chain in Algorithm 2 mixes perfectly. In other words, $\mathbf{Z}_{t,j}$ and $\mathbf{Z}_{t,j-1}$ are independent of each other and $\mathbf{Z}_{t,j}$ follows the conditional density (3) exactly. In practice, $\mathbf{Z}_{t,j}$ and $\mathbf{Z}_{t,j-1}$ are dependent and this dependence is attenuated as β is increased. Note that this simplifying assumption is standard in analyzing similar splitting algorithms (Guyader et al. 2011). The second assumption is that the pilot algorithm selects the levels $\{\gamma_t\}$ so that the conditional probabilities ρ_t in (2) are exactly (as opposed to approximately) equal to $1/s$ for all t.

Let $N_t=|\mathscr{Z}_t|$ be the random number of points or states that have reached level γ_t at the t-th iteration of the GS Algorithm 3. Initially we have $N_0=1$, because Algorithm 3 runs a single trajectory. In this setting and under the above assumptions, each state $\mathbf{Z}_j$ can either yield s offspring points or zero points

and all states yield s offspring with the same probability. If we denote the number of offspring of state j in the t-th iteration by $Q_{j,t}$, then we have the branching process recursion:

$$N_{t+1} = Q_{1,t} + Q_{2,t} + \cdots + Q_{N_t,t},$$

where $\mathbb{P}(Q_{j,t} = s) = \rho = 1/s$, $\mathbb{P}(Q_{j,t} = 0) = 1 - \rho$. Thus, $\mathbb{E}[Q_{j,t}] = s\rho = 1$, $\mathrm{Var}(Q_{j,t}) = s - 1$,and we have via standard branching process arguments (Harris 1989, Page 6) that $\mathbb{E}[N_t] = 1$, $\mathrm{Var}(N_t) = t(s-1)$. Hence, for the unbiased estimator $W = N_\tau / s^\tau$ in Algorithm 3 we obtain $\mathrm{Var}(W) = \tau(s-1)/s^{2\tau}$ with $\tau = \lfloor -\ln_s(u) \rfloor = -\ln_s(u)$ from Assumption 2.

Recall that an estimator $\widehat{u}$ of u is *logarithmically efficient* (Kroese et al. 2011, Page 382) if the following condition holds:

$$\limsup_{u \downarrow 0} \left| \frac{\ln(\mathrm{Var}(\widehat{u}))}{\ln(u^2)} \right| \geqslant 1 .$$

For the logarithmic efficiency criterion we thus obtain

$$\lim_{u \downarrow 0} \left| \frac{\ln(\mathrm{Var}(W))}{\ln(u^2)} \right| = \lim_{u \downarrow 0} \left| \frac{\ln(\tau(s-1)) - 2\tau \ln(s)}{-2\tau \ln(s)} \right| = \lim_{\tau \uparrow \infty} \left| \frac{\ln(\tau(s-1)) - 2\tau \ln(s)}{-2\tau \ln(s)} \right| = 1 .$$

Therefore, under the two idealized assumptions the estimator W and hence $\widehat{u}$ in (4) is logarithmically efficient. Next, note that the total simulation effort (proportional to computing time) in Algorithm 3 is the random variable $s(N_0 + \cdots + N_{\tau-1})$ with expected value $s\tau$. Hence, the expected *relative time variance product* is given by

$$\frac{\mathrm{Var}(W)}{u^2} \times (s\tau) = \tau(s-1) \times (s\tau) = \tau^2 s(s-1) = [\ln_s(u)]^2 s(s-1) = \frac{[\ln(u)]^2 s(s-1)}{[\ln(s)]^2},$$

which is minimized as a function of s for $s > 1$ at $s = 1.9036969...$ ($s = 2$ when constrained on the integers). This theoretical finding agrees with empirical results that the best performance of the GS occurs when the splitting factor $s = 2$.

REFERENCES

Aggarwal, K. K., J. S. Gupta, and K. B. Misra. 1975. "A Simple Method for Reliability Evaluation of a Communication System". *IEEE Transactions on Communication*:563–566.

Ahmad, S. H. 1990. "Enumeration of minimal cutsets of an undirected graph". *Microelectronics Reliability* 30 (1): 39–42.

Ball, M. O., and J. S. Provan. 1982. "Bounds on the Reliability Polynomial for Shellable Independence Systems". *SIAM Journal on Algebraic and Discrete Methods* 3:166–181.

Barlow, R., and F. Proschan. 1975. *Statistical Theory of Reliability and Life Testing*. New York: Holt, Rinehart and Wilson.

Biegel, J. E. 1977. "Determination of Tie Sets and Cutsets for a System without Feedback". *IEEE Transactions on Reliability* 26:39–42.

Botev, Z. I., P. L'Ecuyer, G. Rubino, R. Simard, and B. Tuffin. 2012. "Static Network Reliability Estimation Via Generalized Splitting". *INFORMS Journal of Computing*. doi: 10.1287/ijoc.1110.0493.

Buldyrev, S. V., R. Parshani, G. Paul, H. E. Stanley, and S. Havlin. 2010. "Catastrophic Cascade of Failures in Interdependent Networks". *Nature* 464:1025–1028.

Cancela, H., and M. El Khadiri. 2003. "On the RVR simulation algorithm for network reliability evaluation". *IEEE Transactions on Reliability* 52 (2): 207–212.

Cancela, H., M. El Khadiri, and G. Rubino. 2009. "Rare Event Analysis by Monte Carlo Techniques in Static Models". In *Rare Event Simulation Using Monte Carlo Methods*, edited by G. Rubino and B. Tuffin, 145–170. Wiley. Chapter 7.

Cancela, H., P. L'Ecuyer, M. Lee, G. Rubino, and B. Tuffin. 2009. "Analysis and Improvements of Path-Based Methods for Monte Carlo Reliability Evaluation of Static Models". In *Simulation Methods for Reliability and Availability of Complex Systems*, edited by J. Faulin, A. A. Juan, S. Martorell, and E. Ramirez-Marquez, 65–84. Springer Verlag.

Cancela, H., P. L'Ecuyer, G. Rubino, and B. Tuffin. 2010, December. "Combination of Conditional Monte Carlo and Approximate Zero-Variance Importance Sampling for Network Reliability Estimation". In *Proceedings of the 2010 Winter Simulation Conference*, edited by B. Johansson, S. Jain, J. Montoya-Torres, J. Hugan, and E. Yücesan, 1263–1274. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Chen, M.-H., and B. W. Schmeiser. 1996. "General Hit-and-Run Monte Carlo sampling for evaluating multidimensional integrals". *Operations Research Letters* 19 (4): 161–169.

Chen, Y. G., and M. C. Yuang. 1996. "A cut-based method for terminal-pair reliability". *IEEE Transactions on Reliability* 45:413–416.

Colbourn, C. J. 1987. *The Combinatorics of Network Reliability*. New York: Oxford University Press.

Elperin, T., I. B. Gertsbakh, and M. Lomonosov. 1991. "Estimation of Network Reliability Using Graph Evolution Models". *IEEE Transactions on Reliability* 40 (5): 572–581.

Elperin, T., I. B. Gertsbakh, and M. Lomonosov. 1992. "An Evolution Model for Monte Carlo Estimation of Equilibrium Network Renewal Parameters". *Probability in the Engineering and Informational Sciences* 6:457–469.

Embrechts, P., F. Lindskog, and A. McNeil. 2003. "Modelling Dependence with Copulas and Applications to Risk Management". In *Handbook of heavy tailed distributions in finance*, edited by S. T. Rachev, 329–384. North Holland: Elsevier.

Fard, N. S., and T.-H. Lee. 1999. "Cutset Enumeration of Network Systems with link and node failures". *Reliability Engineering and System Safety* 65:141–146.

Gertsbakh, I. B., and Y. Shpungin. 2010. *Models of Network Reliability: Analysis, Combinatorics, and Monte Carlo*. Boca Raton, FL: CRC Press.

Ghosh, S. K., and B. Singh. 1993. "Enumeration of minimal cutsets from matrix representation of directed/undirected network". *Microelectronics Reliability* 33 (7): 947–949.

Guyader, A., N. Hengartner, and E. Matzner-Lber. 2011. "Simulation and Estimation of Extreme Quantiles and Extreme Probabilities". *Applied Mathematics and Optimization* 64 (2): 171–196.

Harris, T. E. 1989. *The Theory of Branching Processes*. New York: Dover Publications.

Iyer, S. M., M. K. Nakayama, and A. V. Gerbessiotis. 2009. "A Markovian Dependability Model with Cascading Failures". *IEEE Transactions on Computers* 58 (9): 1238–1249.

Kahn, H., and T. E. Harris. 1951. "Estimation of particle transmission by random sampling". *National Bureau Standards Appl. Math. Ser.* 12:2730.

Kroese, D. P., T. Taimre, and Z. I. Botev. 2011. *Handbook of Monte Carlo methods*. New York: John Wiley & Sons.

Kuo, S., F. Yeh, and H.-Y. Lin. 2007. "Efficient and Exact Reliability Evaluation for Networks with Imperfect Vertices". *IEEE Transactions on Reliability* 51 (2): 288–300.

L'Ecuyer, P., G. Rubino, S. Saggadi, and B. Tuffin. 2011. "Approximate Zero-Variance Importance Sampling for Static Network Reliability Estimation". *IEEE Transactions on Reliability* 8 (4): 590–604.

L'Ecuyer, P., S. Saggadi, and B. Tuffin. 2011, December. "Graph Reductions to Speed Up Importance Sampling-Based Static Reliability Estimation". In *Proceedings of the 2011 Winter Simulation Conference*, edited by S. Jain, R. R. Creasey, J. Himmelspach, K. P. White, and M. Fu, 429–438. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

L'Ecuyer, P., and B. Tuffin. 2011. "Approximating Zero-Variance Importance Sampling in a Reliability Setting". *Annals of Operations Research* 189:277–297.

Lin, C.-H., and W.-N. Yang. 2011. "A simple and efficient importance sampling scheme for stochastic network unreliability estimation". *Simulation Modelling Practice and Theory* 19:924–935.

Lomonosov, M., and Y. Shpungin. 1999. "Combinatorics and Reliability Monte Carlo". *Random Structures and Algorithms* 14 (4): 329–343.

Nahman, J. M. 1992. "Minimal paths and cuts of network exposes for common-cause failures". *IEEE Transactions on Reliability* 41 (1): 76–80.

Netes, V. A., and B. P. Filin. 1996. "Consideration of node failures in network-reliability calculation". *IEEE Transactions on Reliability* 45 (1): 127–128.

Ram, M., and S. B. Singh. 2009. "Analysis of Reliability Characteristics of a Complex Engineering System Under Copula". *Journal of Reliability and Statistical Studies* 2 (1): 91–102.

Singpurwalla, N. D. 2002. "Dependence in network reliability". In *Proceedings of the Fifth International Conference on Information Fusion, vol II*, 981–985: IEEE.

Singpurwalla, N. D. 2006. *Reliability and risk: a Bayesian perspective*. illustrated, reprint ed. Wiley series in probability and mathematical statistics. New York: John Wiley & Sons.

Theologou, O. R., and J. G. Carlier. 1991. "Factoring & reductions for networks with imperfect nodes". *IEEE Transactions on Reliability* 40 (2): 210–217.

Walter, M., S. Esch, and P. Limbourg. 2009. "COBAREA: The COpula-BAsed REliability and Availability modeling environment". In *Proceedings of the 6-th International Conference on Quantitative Evaluation of Systems*, 119–120.

AUTHOR BIOGRAPHIES

ZDRAVKO I. BOTEV is a Lecturer at the School of Mathematics and Statistics at the University of New South Wales in Sydney, Australia. He obtained his Ph.D. in Mathematics from The University of Queensland, Australia, in 2010. His research interests include splitting and adaptive importance sampling methods for rare-event simulation. He has written jointly with D. P. Kroese and T. Taimre a *Handbook of Monte Carlo Methods* published by John Wiley & Sons in 2011.

PIERRE L'ECUYER is Professor in the Département d'Informatique et de Recherche Opérationnelle, at the Université de Montréal, Canada. He holds the Canada Research Chair in Stochastic Simulation and Optimization. He is a member of the CIRRELT and GERAD research centers. His main research interests are random number generation, quasi-Monte Carlo methods, efficiency improvement via variance reduction, sensitivity analysis and optimization of discrete-event stochastic systems, and discrete-event simulation in general. He is currently Editor-in-Chief for *ACM Transactions on Modeling and Computer Simulation*, and Associate/Area Editor for *ACM Transactions on Mathematical Software*, *Statistics and Computing*, *International Transactions in Operational Research*, and *Cryptography and Communications*. More information and his recent research articles are available on-line from his web page: http://www.iro.umontreal.ca/~lecuyer.

BRUNO TUFFIN received his PhD degree in applied mathematics from the University of Rennes 1 (France) in 1997. Since then, he has been with INRIA in Rennes. He spent eight months as a postdoc at Duke University in 1999. His research interests include developing Monte Carlo and quasi-Monte Carlo simulation techniques for the performance evaluation of telecommunication systems, and developing new Internet-pricing schemes and telecommunication-related economical models. He is currently Associate Editor for *INFORMS Journal on Computing*, *ACM Transactions on Modeling and Computer Simulation* and *Mathematical Methods of Operations Research*. He has written or co-written two books devoted to simulation: *Rare event simulation using Monte Carlo methods* published by John Wiley & Sons in 2009, and *La simulation de Monte Carlo* (in French), published by Hermes Editions in 2010. His email address is bruno.tuffin@inria.fr.