TI 2009-084/4
Tinbergen Institute Discussion Paper

The Cross-Entropy Method With Patching For Rare-Event Simulation Of Large Markov Chains

Bahar Kaynar
Ad Ridder

Dept. of Econometrics & Operations Research, VU University Amsterdam.

Tinbergen Institute

The Tinbergen Institute is the institute for economic research of the Erasmus Universiteit Rotterdam, Universiteit van Amsterdam, and Vrije Universiteit Amsterdam.

Tinbergen Institute Amsterdam
Roetersstraat 31
1018 WB Amsterdam
The Netherlands
Tel.: +31(0)20 551 3500
Fax: +31(0)20 551 3555

Tinbergen Institute Rotterdam
Burg. Oudlaan 50
3062 PA Rotterdam
The Netherlands
Tel.: +31(0)10 408 8900
Fax: +31(0)10 408 9031

The cross-entropy method with patching for rare-event simulation of large Markov chains

Bahar Kaynar*and Ad Ridder
Department Econometrics and Operations Research
Vrije University Amsterdam, Netherlands
email {bkaynar,aridder}@feweb.vu.nl

August 28, 2009

Abstract

There are various importance sampling schemes to estimate rare event probabilities in Markovian systems such as Markovian reliability models and Jackson networks. In this work, we present a general state dependent importance sampling method which partitions the state space and applies the cross-entropy method to each partition. We investigate two versions of our algorithm and apply them to several examples of reliability and queueing models. In all these examples we compare our method with other importance sampling schemes. The performance of the importance sampling schemes is measured by the relative error of the estimator and by the efficiency of the algorithm. The results from experiments show considerable improvements both in running time of the algorithm and the variance of the estimator.

Keywords: Cross-Entropy, Rare Events, Importance Sampling, Large-Scale Markov Chains.

1 Introduction

Markov chain modeling is an important tool that is used heavily in many Applied Probability and Stochastic Operations Research studies of systems such as communication, production, reliability, queueing, inventory, etc (Dayar and Stewart 2000; Stewart 2007). For realistic system modeling these Markov chains become enormously large with state spaces in the order of millions or even more. This large number of states that we observe in Markov chain modeling results from the exponential increase of the number of states with the number of state variables (the 'curse of dimensionality'). The advantage of Markov chain modeling is found in its computable quantities by solving systems of linear equations, for instance for the stationary distribution, or for absorption probabilities. Clearly, when the state space is large enough, direct methods (e.g. Gauss elimination) are inefficient. Other numerical techniques have been developed and analyzed which are based mostly on iterative methods, projection techniques, and aggregation/disaggregation. The applicability and efficiency of a method depend strongly on the structure of the Markov chain under study (Dayar and Stewart 2000; Stewart 1999).

Our interest is the computation of absorption probabilities of extremely small order, say 10^{-9} or less. That is, the set of absorption states is a so-called rare event. Rare events are found typically in applications such

*Research supported by NWO grant 400-06-044.

as reliability systems where the set of states in which the whole system fails, should almost never occur, or in queueing systems that model telecommunication technologies where loss of traffic packets should be unlikely. In this paper we consider estimation of these absorbing probabilities by Monte Carlo simulation because we had the following arguments not to choose one of the numerical techniques that we mentioned earlier.

1. We need the probabilities in a few states only whereas the numerical methods compute these quantities in all states.
2. For large state spaces one needs to implement advanced numerical techniques that fall beyond our scope.
3. We like to have fast and efficient algorithms.
4. We do not know in advance the order of magnitude of the target probabilities, thus this makes the stopping criterion in iterative methods uncertain, or maybe unnecessary small.

Since we deal with rare event probabilities of extremely small order, we execute the simulation under an importance sampling measure. In many problems involving rare events, importance sampling has provided efficient estimators with huge variance reduction and substantial running time improvement, see the recent surveys Bucklew 2004; Juneja and Shahabuddin 2006; Rubino and Tuffin 2009. The challenge of importance sampling is to come up with a biasing scheme or change of measure that indeed gives these improvements. There is substantial literature available on importance sampling for rare event problems in Markov chains, both in a more generic setting, and in specific models. Concerning the generic setting, the basic idea is to approximate the zero-variance change of measure which is theoretically known but practically not implementable (Glynn and Iglehart 1989). We mention (Bucklew et al. 1990) who applied large deviations techniques, (Dupuis and Wang 2005) who considered the importance sampling problem as a stochastic game, (Ahamed et al 2006; Randhawa and Juneja 2004) who considered the rare event problem as an expected average reward in a regenerative Markov chain and applied stochastic approximation techniques. Several adaptive techniques have been proposed which attempt to learn the zero-variance change of measure for Markov chains by an iterative procedure: a number of sample paths is generated and based on the 'scores' of these path the current change of measure is updated (Desai and Glynn 2001; Kollman et al. 1999).

Many more research studies has been carried out to construct importance algorithms that are taylor-made for specific problems, we refer to the surveys (Bucklew 2004; Heidelberger 1995; Juneja and Shahabuddin 2006; Rubino and Tuffin 2009) for some details. Our paper has the intention to propose an importance sampling algorithm that is generally applicable to rare events in multi-dimensional Markov chains, although our examples will be taken from queueing and reliability. Our algorithm builds on the cross-entropy method (Rubinstein and Kroese 2004) which we will describe in more detail in section 3. At this point it suffices to understand that the cross-entropy method updates iteratively the change of measure by maximizing a parameterized nonlinear program, where the parameters are the transition probabilities of the Markov chain. This approach breaks down when there are too many parameters, i.e., when we deal with large Markov chains.

To tackle the problem of too many parameters we propose to partition the state space (and their associated transitions) in smaller 'patches', and then apply the cross-entropy iterations on these smaller sets. We consider two versions of this approach. In the first version we let each transition probability be a parameter that is to be determined optimally by the cross-entropy method. Once the probabilities of a patch are determined, we keep them fixed in the cross-entropy iterations on subsequent patches. The second version assumes a Markov chain with a face-homogeneous structure such as we encounter typically in Jackson queueing networks (Ignatiouk

2005). When we keep this structure in the change of measure, the number of parameters is already reduced to a tractable size. Then we apply our patching algorithm but now we do not keep the parameters fixed while iterating subsequent patches. Other attempts of reducing the parameter space in the cross-entropy method has been reported in Boer and Nicola (2002). The authors use also a grouping idea, however they group states along boundary layers that depend on the contents of the queues.

The rest of the paper is organized as follows. In Section 2 we define the performance measures to compare importance sampling estimators, and we define the type of rare events that we shall consider in this paper. In the first part of Section 3 we describe the classical cross-entropy method for Markov chains, and then we present our patching algorithm that forms the basis of our study. Section 4 contains the face-dependent methods, and Section 5 shows results of our methods applied to reliability models and to Jackson queueing networks.

2 Preliminaries

We assume some underlying probability space $(\Omega, \mathcal{A}, \mathbb{P})$ of the Markov chain such that the chain has stationary transition probabilities and is irreducible over its state space. We assume that a family of rare events $\{A_n\} \subset \mathcal{A}$ is defined and parameterized by n such that $\lim_{n\to\infty} \mathbb{P}(A_n) = 0$. Most often this decaying is exponentially fast $\lim_{n\to\infty} \frac{1}{n} \log \mathbb{P}(A_n) = -\theta$ based on a large deviations asymptotic (Bucklew 2004). Let Y_n be the unbiased estimator of $\mathbb{P}(A_n)$ obtained by averaging i.i.d. replications of $1\{A_n\}$ (the indicator function of the set A_n) which are generated under the original probability $\mathbb{P}$, and let Y_n^* be an unbiased importance sampling estimator obtained by averaging i.i.d. replications of

$$1\{A_n\} \frac{d\mathbb{P}}{d\mathbb{P}_n^*},$$

generated under a change of measure $\mathbb{P}_n^*$, where $d\mathbb{P}/d\mathbb{P}_n^*$ is the likelihood ratio, and where we assume that $1\{A_n\}d\mathbb{P}$ is absolute continuous w.r.t. $d\mathbb{P}_n^*$. The performance of an importance sampling estimator is measured commonly through its relative error, the exponential decay rate of its second moment, and its effort (Asmussen and Rubinstein 1995; Heidelberger 1995). These three performance measures of the importance sampling estimator are

$$\begin{aligned}
\mathrm{RE}[Y_n^*] &= \frac{\sqrt{\mathrm{Var}_n^*[Y_n^*]}}{\mathbb{E}_n^*[Y_n^*]}; \\
\mathrm{RAT}[Y_n^*] &= \frac{\log \mathbb{E}_n^* \left[(Y_n^*)^2\right]}{\log \mathbb{E}_n^* \left[Y_n^*\right]}; \\
\mathrm{EFF}[Y_n^*] &= -\log\left(\mathrm{Var}_n^*[Y_n^*] \times \mathrm{CPU}[Y_n^*]\right),
\end{aligned}$$

where CPU means the running time of the importance sampling algorithm on our computer. Notice that we take here expectations w.r.t. the probability measure $\mathbb{P}_n^*$. Clearly, our target is to obtain small variances (relative to the mean), which is expressed by a small relative error RE. Ideally a zero-variance estimator is associated with the change of measure $\mathbb{P}_n^*(\cdot) = \mathbb{P}(\cdot|A_n)$, but this measure is not implementable because it requires knowledge of the unknown $\mathbb{P}(A_n)$ (Glynn and Iglehart 1989). A 'second best' estimator would have bounded relative error, one says that the estimator is strongly efficient (L'Ecuyer et al. 2008): $\limsup_{n\to\infty} \mathrm{RE}[Y_n^*] < \infty$. It would mean that the number of samples required to achieve a fixed relative error is constant for all n. Finding an importance sampling estimator with this property is in most cases a hard problem, and thus one seeks usually an asymptotically optimal, or logarithmically efficient estimator (L'Ecuyer et al. 2008): $\liminf_{n\to\infty} \mathrm{RAT}[Y_n^*] \geq 2$. It would mean that, assuming that the rare event probabilities $\mathbb{P}(A_n)$ decay exponentially fast, the required sample sizes grow at most polynomially. Finally, we include the EFF-measure which takes into account the computing time of the importance sampling algorithm (the $-\log$ is to get convenient numbers).

2.1 The Rare Events

We restrict to rare events A_n in Markov chains that are given by hitting times to absorption sets of the state space. To make this clear, we drop for the moment the rarity parameter n, thus, we deal with a discrete-time Markov chain $(X(t))_{t=0}^{\infty}$ on a finite but large state space $\mathcal{X}$, and with a matrix of transition probabilities $P = (p(x,y))_{x,y\in\mathcal{X}}$, i.e., $p(x,y) = \mathbb{P}(X(t+1) = y | X(t) = x)$. The state space is partitioned into three sets: $\mathcal{G}$ is a small set of 'good' states, $\mathcal{F}$ is a small set of 'failed' or 'bad' states, and the other states form the large set of 'internal' states $\mathcal{T} = \mathcal{X} \setminus (\mathcal{G} \cup \mathcal{F})$. For each internal state $x \in \mathcal{T}$ let $\gamma(x)$ be the probability that the Markov chain will hit the failure set before the good set when the chain starts in state x. (Equivalently, $\gamma(x)$ is the probability of absorption in the bad set of a Markov chain $(X^{\text{abs}}(t))$ that is constructed from (X_t) by making the good and bad states absorbing: $p^{\text{abs}}(x,x) = 1$ for all $x \in \mathcal{G} \cup \mathcal{F}$.) Finally, let $\mathcal{I} \subset \mathcal{T}$ be the set of internal states that can be reached from the good states in one transition:

$$\mathcal{I} = \{y \in \mathcal{T} : \exists\ x \in \mathcal{G} \text{ s.t. } p(x,y) > 0\}.$$

Typically we are interested in the probabilities of rare events A that the chain hits a bad state before a good state starting from some $x \in \mathcal{I}$, i.e., $\mathbb{P}(A) = \gamma(x)$. In some applications we let the chain start in a good state while the first transition takes the chain out of the good set, i.e., we consider $\mathbb{P}(A) = \sum_{y\in\mathcal{I}} p(x,y)\gamma(y)$ (for $x \in \mathcal{G}$).

3 State-Dependent Cross-Entropy

In this section we first give a short review of the state-dependent cross-entropy method for Markov chains (Rubinstein and Kroese 2004, Section 3.3), after which in part 3.1 we describe our algorithm.

When we run the standard simulations, we generate sample paths $\boldsymbol{X}$ of the Markov chain by simulating its consecutive states using the matrix P. The sample paths have random length

$$T = \inf\{t > 0 : X(t) \in \mathcal{G} \cup \mathcal{F}\},$$

where we assume for ease of notation that the chain starts at time 0 in some fixed state $x_0 \in \mathcal{I}$. (We still have the rarity parameter n dropped). For importance sampling simulations we restrict the change of measure $\mathbb{P}^*$ so that we retain a Markov chain, say with a matrix of transition probabilities P^*. We restrict even further by demanding for the individual transition probabilities to satisfy $p^*(x,y) > 0$ if and only if $p(x,y) > 0$, which is sufficient for the absolute continuity requirement $1\{A\}d\mathbb{P} \ll d\mathbb{P}^*$. Let $\mathcal{P}$ be the set of all feasible matrices of transition probabilities. The idea of the cross-entropy method is to choose a feasible matrix which minimizes the Kullback-Leibler divergence between the optimal (zero-variance) change of measure and the importance sampling change of measure. After some standard algebraic manipulations, this comes down to solving

$$\sup_{P^*\in\mathcal{P}} \mathbb{E}[1\{X(T) \in \mathcal{F}\} \log d\mathbb{P}^*(\boldsymbol{X})]. \tag{1}$$

Notice that the sample path probability $d\mathbb{P}^*(\boldsymbol{X})$ is a product of transition probabilities:

$$d\mathbb{P}^*(\boldsymbol{X}) = \prod_{t=1}^{T} p^*(X(t-1), X(t)), \tag{2}$$

and notice that we take expectations with respect to the original probability measure $\mathbb{P}$. There are several ways of attacking the cross-entropy optimization program (1), for instance, solving the first order conditions. However, in almost all cases the program is not analytically nor numerically tractable, and thus one resorts

to simulation. Again, several simulation techniques may be applied, for instance stochastic approximation, however, it turns out that sample path optimization is most conveniently, i.e., one considers the stochastic counterpart of (1), see Rubinstein and Kroese (2004). Since the program involves the rare event, we first apply a change of measure:

$$\mathbb{E}[1\{X(T) \in \mathcal{F}\} \log d\mathbb{P}^*(\boldsymbol{X})] = \mathbb{E}^{(0)} \left[\frac{d\mathbb{P}}{d\mathbb{P}^{(0)}}(\boldsymbol{X}) 1\{X(T) \in \mathcal{F}\} \log d\mathbb{P}^*(\boldsymbol{X}) \right]. \tag{3}$$

This is done for a probability measure $\mathbb{P}^{(0)}$ such that (i) the Markov property is retained; (ii) the associated matrix of transition probabilities is feasible $P^{(0)} \in \mathcal{P}$; (iii) the set A is 'not so' rare under $\mathbb{P}^{(0)}$. The program (3) is solved iteratively by estimation: let $\boldsymbol{X}_1, \boldsymbol{X}_2, \ldots, \boldsymbol{X}_k$ be i.i.d. sample paths of the Markov chain generated by simulating the states according to a matrix of transition probabilities $P^{(j)}$ until absorption in the good or bad states, then we calculate for $j = 0, 1, \ldots$

$$P^{(j+1)} = \arg\max_{P^* \in \mathcal{P}} \frac{1}{k} \sum_{i=1}^{k} \frac{d\mathbb{P}}{d\mathbb{P}^{(j)}}(\boldsymbol{X}_i) 1\{X_i(T) \in \mathcal{F}\} \log d\mathbb{P}^*(\boldsymbol{X}_i). \tag{4}$$

We repeat this 'updating' of the change of measure a few times until the difference $P^{(j+1)} - P^{(j)}$ is small enough (in some matrix norm).

It is straightforward to solve the optimization program (4) by considering its first order conditions, and using the product form expression of the sample path probability (2). Denote by $N_i(x, y)$ the number of times that the i-th sample path of the Markov chain makes the transition from state x to state y, and denote by L_i the likelihood ratio of the i-th sample path. Then we get for the individual entries

$$p^{(j+1)}(x, y) = \frac{\sum_{i=1}^{k} L_i 1\{X_i(T) \in \mathcal{F}\} N_i(x, y)}{\sum_{i=1}^{k} L_i 1\{X_i(T) \in \mathcal{F}\} \sum_{y \in \mathcal{X}} N_i(x, y)}. \tag{5}$$

Clearly, we might get a zero update for a transition $x \to y$ which is not observed in the sample but which has originally a positive probability $p(x, y)$. To overcome this problem it is suggested to apply a weighting combination (Rubinstein and Kroese (2004), Section 4.2), i.e., denote by $\hat{p}^{(j+1)}(x, y)$ the expression of (5), then

$$p^{(j+1)}(x, y) = \alpha p(x, y) + (1 - \alpha) \hat{p}^{(j+1)}(x, y), \tag{6}$$

for some $\alpha \in (0, 1)$.

3.1 The Patching Algorithm

The cross-entropy method breaks down when there are too many parameters to update. In this section we consider an adaptation that decreases the number of parameters to update in each iteration. Recall that the parameters in our study are the transition probabilities of a finite-state Markov chain.

The main idea of our so-called patching algorithm is to implement the iteration rule (4) recursively in small portions of the state space. We start with a set of the state space that contains the good states, $\mathcal{G}$, and a small set of internal states some of which we specify as being 'bad'. Being a small subset of the state space we call it a patch. Then we restrict the Markov chain to this patch and apply the the cross-entropy iteration (4) for finding a change of measure for the transition probabilities in the patch. These updated transition probabilities are kept fixed for the rest of the algorithm. Next, we enlarge the patch and apply the same procedure to the parameters which are introduced newly in the expanded patch. We continue enlarging the patch and updating the parameters until the size of the patch matches the original state space. The construction of the patches should be done in such a manner that the Markov chain restricted to a patch remains irreducible.

As a simple example, consider a two-dimensional Markov chain on

$$\mathcal{X} = \{0, 1, ..., 10\}^2 = \{x = (x_1, x_2) : x_1, x_2 \in \{0, 1, ..., 10\}\},$$

with transitions $\pm e_1$ and $\pm e_2$ (whenever possible) where $e_1 = (1, 0)$ and $e_2 = (0, 1)$. The state space is partitioned into the three sets as mentioned in Section 2.1: the good set $\mathcal{G} = \{0\} = \{(0, 0)\}$, the bad set $\mathcal{F} = \{(10, 10)\}$, and the internal set $\mathcal{T}$ of all other states, see Figure 1. The initial state of all simulation paths is $(1, 0)$.

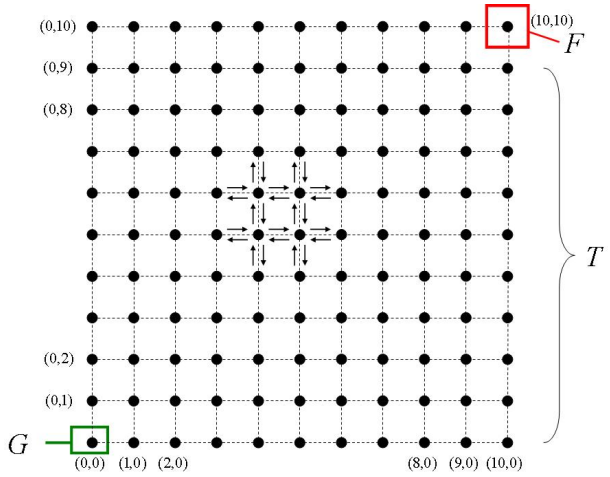


Figure 1: *The state space of the simple example*

Let the first patch be $\text{Patch}_1 = \{0, 1, 2, 3, 4\}^2$, which contains the good set $\mathcal{G}$. Notice that Patch_1 is proportional to the state space $\mathcal{X}$ by the same reduction factor in both dimensions. Thus it is natural to let $\mathcal{F}_1 = \{(4, 4)\}$ be the bad set in Patch_1. Next, we need to define the matrix of transition probabilities P_1 by restricting P to Patch_1. Define the "border states"

$$\mathcal{B}_1 = \{(x_1, 4) \ : \ 0 \le x_1 \le 4\} \cup \{(4, x_2) \ : \ 0 \le x_2 \le 4\}.$$

Clearly $p_1(x, y) = p(x, y)$ for all transitions $x \to y$ when x is not a border state. For the border states we renormalize the probabilities of transitions to states in Patch_1:

$$p_1(x, y) = \frac{p(x, y)}{\sum_{z \in \text{Patch}_1} p(x, z)}, \quad x \in \mathcal{B}_1, \text{ and } y \in \text{Patch}_1.$$

Finally, to start off the cross-entropy iterations with a matrix of transition probabilities $P_1^{(0)}$ such that the bad set $\mathcal{F}_1$ is less rare than simulating with matrix P_1, we take the uniform transition probabilities, i.e.,

$$p_1(x, y) > 0 \ \Leftrightarrow \ p_1^{(0)}(x, y) = \frac{1}{\text{number of transitions out of state } x \text{ in } \text{Patch}_1}.$$

Then by using expressions (5) and (6), we update the transition probabilities for the change of measure a few times. The resulting change of measure $p_1^*(x, y)$ for transitions out off $x \in \text{Patch}_1 \setminus \mathcal{B}_1$ are stored permanently, and will be used both in the subsequent steps of the algorithm, and in the final matrix of transition probabilities P^* on the whole state space. We denote the set of these transitions by $\mathcal{U}$ and we call them permanent transitions.

After dealing with Patch_1, we enlarge it to

$$\text{Patch}_2 = \{x = (x_1, x_2) \ : \ 0 \le x_1 \le 6, 0 \le x_2 \le 6\}.$$

The bad set becomes $\mathcal{F}_2 = \{6, 6\}$. We get the matrix of transition probabilities P_2 on Patch_2 by restricting the original matrix P in the same manner as above: $p_2(x, y) = p(x, y)$ for all transitions $x \to y$ when x is not a border state $(6, x_2)$ or $(x_1, 6)$, and for the border states we renormalize. Now consider the matrix $P_2^{(0)}$ to start off the cross-entropy iterations on Patch_2. We let it be the probabilities determined in Patch_1 for the permanent transitions, and otherwise they are uniform:

$$p_2^{(0)}(x, y) = \begin{cases} p^*(x, y) & \text{for } (x \to y) \in \mathcal{U}, \\ \dfrac{1}{\text{number of transitions out of state } x \text{ in } \text{Patch}_2} & \text{otherwise.} \end{cases}$$

Alternatively, also for the states $x \in \mathcal{B}_1 \setminus \mathcal{F}_1$ in the (previous) border set (except the bad states), we might use the transition probabilities $p_1^*(x, y)$ (where $y \in \text{Patch}_1$) and renormalize these after adding $p(x, y)$ (where $y \in \text{Patch}_2 \setminus \text{Patch}_1$). In our implementations we apply the latter.

Having set the initial matrix $P_2^{(0)}$ and the transitions whose probabilities need to be updated, we apply again the cross-entropy method for finding a change of measure P_2^* on Patch_2. Recall that we keep $p_2^{(j)}(x, y) = p_1^*(x, y) = p^*(x, y)$ for all transitions $(x \to y) \in \mathcal{U}$ during the cross-entropy iterations, thus the updating rule (5) is applied to the other transitions. With the resulting matrix P_2^* we update the permanent set $\mathcal{U}$ by adding the probabilities $p_2^*(x, y)$ of transitions $(x \to y)$ that were not permanent and for which $x \in \text{Patch}_2 \setminus \mathcal{B}_2$ (clearly, $\mathcal{B}_2$ is the set of border states of Patch_2).

This procedure is repeated by enlarging Patch_2 to Patch_3, and later on to Patch_4 which is the whole state space $\mathcal{X}$, see Figure 2. After these steps we have found our change of measure P^* for running the importance sampling simulations and estimating the original rare event probability.

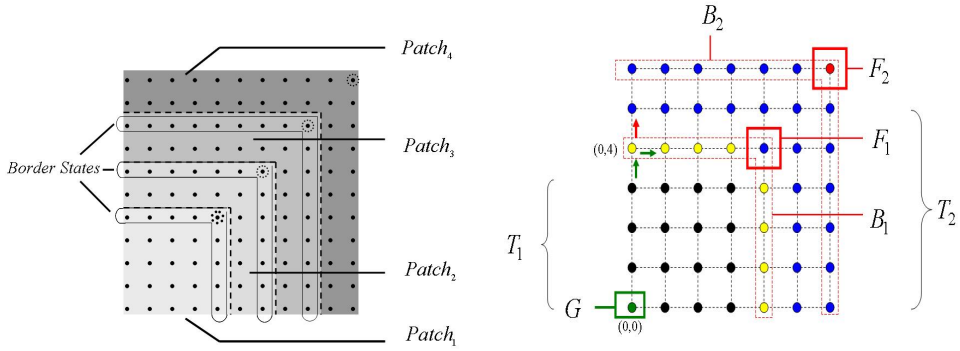


Figure 2: *The steps in the patching algorithm.*

In this small example we chose the patches having the same structure as the original state space, i.e., $\{0, 1, \ldots, m\}^2$ where m increases with jumps. However, there are numerous other ways of defining the patches. Other choices to be made are the number of sample paths in the cross-entropy iterations, the weighting factor α in the updating rule (6), and the number of cross-entropy iterations per patch. The performance of the resulting important sampling estimator may depend on all these choices as we will see in Section 5 with the numerical results. There are no general rules for deciding which choices will give the better results. Hence, our choices are mainly based on "trial and error".

Summarizing, the general cross-entropy patching algorithm reads as follows.

Algorithm 1.

1. Set $i = 1$.

2. *Choose* $Patch_i$.

3. *Form* $P_i^{(0)}$ *for* $Patch_i$.

4. *Apply the updating formulae* (5) *and* (6).

5. *Stop when convergence has been obtained, otherwise set* $j = j + 1$ *and go to step* 3.

6. *Stop when* $Patch_i$ *matches with the original state space, otherwise set* $i = i + 1$ *and go to step* 2.

7. *Estimate the rare event probability* $\mathbb{P}(A)$ *by using importance sampling.*

4 Face-Dependent Cross-Entropy

In the previous section we considered the cross-entropy method for general finite-state (discrete-time) Markov chains, and we described our approach to update the transition probabilities in smaller groups or patches. In this section we shall consider face-homogeneous Markov chains with bounded jumps, to be explained shortly (see for instance also Ignatiouk (2005)). It is even allowed that the state space of the chain is infinite. We shall describe the "standard" cross-entropy method for finding a change of measure for each face, and we shall give a cross-entropy algorithm that combines faces and patches.

We encounter face-homogeneous Markov chains typically in Jackson queueing networks. These networks are modeled by continuous-time Markov chain, but by embedding at the jump times we obtain a discrete-time Markov chain, $(X(t))_{t=0}^{\infty}$, with state space $\mathcal{X} = \mathbb{Z}_+^d$ (assuming d queues). For any $x \in \mathbb{Z}_+^d$, let $\Lambda(x)$ be the set of indices i for which $x_i > 0$. For any subset $\Lambda \subset \{1, 2, ..., d\}$, define face F_Λ by

$$F_\Lambda = \{x \in \mathbb{Z}_+^d \ : \ \Lambda(x) = \Lambda\}.$$

Specifically we have $F_\emptyset = \{0\}$. The face-homogeneity assumption is that the transition probabilities of the chain should satisfy

$$p(x, x + y) = \mathbb{P}\left(X(t + 1) = x + y \mid X(t) = x\right) = p_{\Lambda(x)}(y),$$

for all $x, x + y \in \mathbb{Z}_+^d$. Hence, we can define 2^d independent random variables ξ_Λ on $\mathbb{Z}^d$ with probability densities

$$\mathbb{P}(\xi_\Lambda = y) = p_\Lambda(y) \quad y \in \mathbb{Z}^d.$$

If we suppose that $\xi_\Lambda(1), \xi_\Lambda(2), \ldots$ are i.i.d. copies of ξ_Λ, then the Markov chain evolves according to

$$X(t + 1) = X(t) + \xi_{\Lambda(X(t))}(t + 1).$$

Notice that $(S_\Lambda(n) = \sum_{t=1}^n \xi_\Lambda(t))_{n=1}^{\infty}$ is a (homogeneous) random walk in $\mathbb{Z}^d$, thus the Markov chain $(X(t))$ might be called a face-homogeneous random walk.

Again we assume some rare event probability $\mathbb{P}(A)$ given by absorption in the bad set before absorption in the good set. We apply the cross-entropy method for finding a change of measure (or transition probability matrix) P^* under which the importance sampling simulations will be executed. We restrict the class of feasible transition matrices $\mathcal{P}$ in the optimization program (1) to those for which the face-homogeneity is retained. Hence, the sample path probability (2) can be expressed as

$$d\mathbb{P}^*(\boldsymbol{X}) = \prod_{t=1}^{T} {p^*}_{\Lambda(X(t))}(X(t + 1) - X(t)).$$

Therefore, if we let M_Λ denote the number of jumps of the jump variable ξ_Λ in face F_Λ, we get a total of $\sum_{\Lambda\subset\{1,2,\ldots,d\}} M_\Lambda$ parameters to update during the cross-entropy iterations, whatever the size of the state space of the Markov chain is. If we again denote by $N_i(\Lambda, y)$ the number of times that the i-th sample path of the Markov chain makes the the jump $\xi_\Lambda = y$, i.e., the associated transition $(x \to x+y)$ for some $x \in \Lambda$, and denote by L_i the likelihood ratio of the i-th sample path, then the update rule (5) in the j-th cross-entropy iteration becomes

$$p_\Lambda^{(j+1)}(y) = \frac{\sum_{i=1}^k L_i 1\{X_i(T) \in \mathcal{F}\} N_i(\Lambda, y)}{\sum_{i=1}^k L_i 1\{X_i(T) \in \mathcal{F}\} \sum_y N_i(\Lambda, y)} \tag{7}$$

The weighting expression (6) is still used with some parameter $\alpha \in (0, 1)$.

A similar idea of reducing the parameter space has been studied in Boer and Nicola (2002) for state-dependent tilting under the method called "Boundary Layers". The fundamental assumption this method is based on is that when a queue's content becomes large, the transition probabilities (in an optimal change of measure) will hardly depend on that queue's content. Therefore, it is suggested to assign the same set of transition probabilities to a group of such states. Thus, the grouping is based on the levels of the queues. We, on the other hand, consider the intrinsic homogeneous face structure of the Markov chain which we retain when implementing the cross-entropy method. Specifically for queueing applications, the fundamental observation for face-dependent cross-entropy method is that it is independent of the level of the queues.

4.1 The Patching Algorithm

As mentioned in Section 3.1, the main idea of the patching algorithm is to update the change of measures for the whole state space in small portions. However, what matters in case of face-dependency is the size of the total number of transitions $\sum_{\Lambda\subset\{1,2,\ldots,d\}} M_\Lambda$ of the d random walks constituting the Markov chain. To explain the patching idea, we return to the original setting in Section 2 and denote the rare event by A_n where the rarity parameter $n \to \infty$. For instance in the Jackson network applications, the rarity parameter n will be an overflow level of one or more queues.

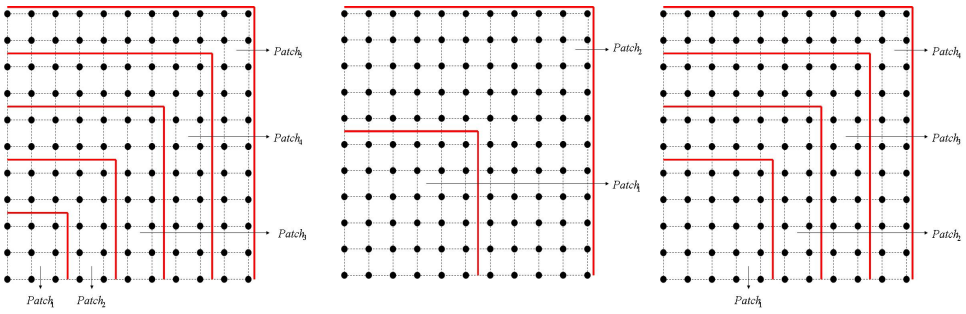
For the patching version of our algorithm we apply the same technique as the standard cross-entropy method for level-crossing probabilities (Rubinstein and Kroese (2004)), and let the rarity parameter n increase as the iterations go on. For the first Patch we set n_0 to some low value. We update the change of measures according to this level. The parameters updated in Patch_i form a good initial probability matrix for Patch_{i+1}. By gradually increasing the value of the parameter n, the sample paths are more and more pushed to the rare event set. However, as will be seen in Section 5, the starting and the increasing value of the parameter n play a crucial role in the reduction of the variance.

5 Numerical examples

In this section we will discuss the performance of our algorithms mentioned in previous sections in two different models: highly reliable Markovian systems and Jackson networks. These systems differ in terms of rare event setting as mentioned in Heidelberger (1995). In reliability models there are a few events happening that create the notion of rare events whereas in Jackson networks rare events happen as a combination of large number of events. It is well known in the literature that different kinds of importance sampling schemes should be applied to both type of problems. Since, our aim is to come up with a general algorithm which is applicable to any type of Markov chain, we will discuss the performance of the algorithms for both models.

CE-D and Patch-D refer to the state-dependent cross-entropy algorithms in section 3 without and with patching, respectively. Similarly, CE-F and Patch-F notify that we have implemented the face-dependent cross-

entropy methods described in section 4. In the notation CE-D(n;k;α), n refers to the number of iterations the cross-entropy is run, k refers to the number of sample paths produced in each iteration and α refers to the weighting factor in equation (6). The same notation is valid also for the Patch-D, CE-F and Patch-F. For the state-dependent and face-dependent patching algorithms we also add a number at the end of the usual notation to indicate the form of clusters we take in every patch. That means, as the number decreases, we will have more patches. In Figure 3 we give a few examples: Patch-F(n;k;α)-2 (clusters of 2), Patch-F(n;k;α)-5 (clusters of 5), and Patch-F(n;k;α)$-2(4)$ (clusters of 2 but starting with a group of 4).



Patch-F(n;k;α)-2 Patch-F(n;k;α)-5 Patch-F(n;k;α)$-2(4)$

Figure 3: *Examples of notation for face-dependent cross-entropy with patches.*

The results for each example contain the probability estimation and the three performance measures discussed in Section 2. We run the cross-entropy method with a predefined number of iterations with a predefined sample size which refers to a fixed budget for running the algorithms. Then, we implement the importance sampling to estimate the rare event probability and performance measures. We do not check whether the cross-entropy method converged or not in every iteration. Our aim is to show the performance of the algorithms with a fixed budget.

5.1 Reliability

Our first example is a model of highly reliable Markovian system which has been studied largely in relation to importance sampling estimation of system failure and mean time to failure, for instance Alexopoulos and Shultes (2001); L'Ecuyer and Tuffin (2007); Goyal et al. (1992); Juneja and Shahabuddin (2001a, 2001b); Shahabuddin (1994a, 1994b). In this reliability system there are d different component types, where type i contains K_i identical components. Each component fails, independently of other events, after an exponentially distributed failure time, with failure rate λ_i for type i components. Failed components may be repaired by one or more service men. The repair times of type i components are exponentially distributed with rate μ_i. We make a few system assumptions.

Assumption 1.
(i) There is no failure propagation.
(ii) There is no group repair.
(iii) The failure rates are functions of a parameter ε according to

$$\lambda_i = \varepsilon^{r_i}, \tag{8}$$

where $\varepsilon > 0$ small and $r_i \geq 1$.
(iv) The repair rates are several orders of magnitude larger: $\mu_i \geq \mu_{\min} = 1$.

However, we do allow priority of service. The system is modeled by a continuous-time Markov chain with states $x = (x_1, \ldots, x_d)$ representing the number of failed components per type. Thus the state space is

$$\mathcal{X} = \prod_{i=1}^{d} \{0, 1, \ldots, K_i\},$$

with cardinality $|\mathcal{X}| = \prod_{i=1}^{d}(K_i + 1)$. Under assumptions 1 (i) and (ii) the Markov chain has jumps of size one, viz. an operational component fails, or a failed component is repaired. It is easy to see that the chain is irreducible over its state space. We might generalize to allow failure propagation, group repair and other transition structures based on state information, but at the moment we consider the most simple models.

The associated discrete-time Markov chain $(X(t))_{t=0}^{\infty}$ is constructed by embedding the continuous-time Markov chain at its jump times. The good set $\mathcal{G}$ consists of the single 'perfect' state $0 = (0, 0 \ldots, 0)$ with all components operational, and the 'bad' or failure set $\mathcal{F}$ represents all states in which the system is down, for instance $\mathcal{F}$ is the single state $(K_1, K_2, \ldots, K_d)$ with all components of all types failed, or the set of states in which all components of at least one type have failed. Let $\mathcal{I}$ be the states with one failed component, in other words, these are the states that can be reached from 0 by one transition. Recall that $\gamma(x)$ is the absorption probability in the failure set when the chain starts in an internal state $x \in \mathcal{T}$, and that we are interested in these absorption probabilities specifically for $x \in \mathcal{I}$. It is more convenient to have a single target. Therefore, we define the failure probability (FP)

$$\gamma(0) = \sum_{x \in \mathcal{I}} p(0, x)\gamma(x),$$

which we wish to estimate by simulation. Besides its own interest, the FP is used for determining the system unavailability and the mean time to failure (Goyal et al. 1992).

The rarity parameter in this reliability model is $n = [1/\varepsilon]$ (see (8) for ϵ), thus we might denote the FP by $\gamma_n(0)$. Under the conditions of Assumption 1 it has been shown in Juneja and Shahabuddin (2001a) that $\gamma_n(0) \to 0$ polynomially as $n \to \infty$, viz., let $r = \min\{r_i : i = 1, 2, \ldots, d\}$, then

$$\lim_{n \to \infty} \frac{1}{\log n} \gamma_n(0) = -r.$$

Several importance sampling schemes have been proposed to give bounded relative error (in certain cases) or asymptotically optimal estimators, notably: simple failure biasing (SFB) (Shahabuddin 1994a), balanced failure biasing (BFB) (Shahabuddin 1994a), balanced likelihood ratio method (BLR) (Alexopoulos and Shultes 2001), and more recently a zero-variance approximation method (ZVA) (L'Ecuyer and Tuffin 2007).

- SFB with parameter $\theta \in (0, 1)$: increase the total failure-transition probability at any state to a constant θ while keeping their original mutual proportionalities; similarly, the repair-transition probabilities are reduced to $1 - \theta$.
- BFB with parameter $\theta \in (0, 1)$: similar to SFB(θ), however it distributes the total probability θ equally to the failure-transitions.
- BLR: the crucial observation is that each repair must be preceded by a failure. Then one can force the product of respective pairs of event likelihood ratios to be one, and thus, the likelihood ratio of a sample path to be bounded above by one.

- ZVA: in the optimal change of measure the associated importance sampling estimator would have zero variance. It is given by transition probability matrix (L'Ecuyer and Tuffin 2007)

$$p^{\text{opt}}(x,y) = p(x,y)\frac{\gamma(y)}{\gamma(x)}, \quad x,y \in \mathcal{T},$$

where $\gamma(x)$ (and $\gamma(y)$) is the probability of absorption in the bad set starting from internal state x (respectively y). The idea is to approximate $p^{\text{opt}}(x,y)$ by

$$p^*(x,y) = \frac{p(x,y)\gamma^{\text{app}}(y)}{\sum_{y\in\mathcal{X}} p(x,y)\gamma^{\text{app}}(y)}. \tag{9}$$

In L'Ecuyer and Tuffin (2007) it is suggested to let $\gamma^{\text{app}}(y) = v_0(y)$, where $v_0(y)$ is the maximal probability of paths of states $y = y_0 \to y_1 \to \cdots \to y_k \in \mathcal{F}$ going from state y to the bad set $\mathcal{F}$. Notice that the denominator in the approximation (9) is simply the normalizing constant. Also we have considered a refinement suggested in L'Ecuyer and Tuffin (2007) to replace $v_0(y)$ by $v_1(y) = v_0(y)^\alpha$, where α is determined by first executing an importance sampling procedure to get a rough estimate $\hat{\gamma}$ of the failure probability $\gamma(0)$, and then matching $v_0(0)^\alpha = \hat{\gamma}$.

We have implemented the SFB, BFB and ZVA schemes to compare with our method in both statistical and computational performance. We give three examples: the first example is a small test example with $1,331$ states and $7,260$ nonzero transition matrix entries (parameters). The second example is from L'Ecuyer and Tuffin (2007) with $2,197$ states and $12,168$ nonzero matrix entries, and the third example has a large state space of $759,375$ states and $4,303,124$ parameters. To assess the estimations we computed numerically the absorption probabilities in the two smaller examples by applying the Gauss-Seidel iteration using software with arbitrary-precision arithmetic.

Example 1

There are 3 types, each with 10 components, with failure rates $\lambda_1 = \varepsilon$, $\lambda_2 = 5\varepsilon$ and $\lambda_3 = 8\varepsilon$ (all failure rates are of the same order as functions of ε: this is called "balanced") and repair rates $\mu_i = 1$ for all i. The rarity parameter ε is taken to be 0.01. There is a single repairman but no priorities are applied between types. All the repairs are made one by one as soon as one has failed, no group repairs are allowed. Finally, the system breaks down when all components of all types have failed, which corresponds to $\mathcal{F} = \{10, 10, 10\}$. The associated failure probability is found by the Gauss-Seidel method to be $1.057E-24$.

The results for all the models can be found in Table 1. All results are averages of 5 repetitions, where the sample size in the (final) importance simulations for estimating the rare event probability is $1,000,000$ in all experiments.

Recall the three performance measures of the importance sampling estimators introduced in Section 2: RE (relative error), RAT (logarithmic ratio) and EFF (efficiency). The EFF measure takes into account the total running time on the computer which includes the execution of the cross-entropy iterations and the final importance sampling simulation. Better performance is obtained by a smaller RE, higher RAT ,and larger EFF.

Method	Estimation	RE	RAT	EFF
SFB(0.7)	8.15E-25	0.3477	1.7950	48.7823
SFB(0.8)	7.20E-25	0.3167	1.7973	49.0323
SFB(0.9)	3.92E-25	0.2971	1.8022	49.6496
BFB(0.7)	8.19E-25	0.1735	1.8179	49.2005
BFB(0.8)	8.11E-25	0.1734	1.8201	49.4110
BFB(0.85)	7.39E-25	0.2141	1.8144	49.3780
ZVA(v_0)	7.27E-26	0.4043	1.7964	50.9142
ZVA(v_1)	2.95E-25	0.3298	1.8002	49.7402
CE-D(7;20000;0.15)	5.90E-29	0.5433	1.0649	*
CE-D(10;30000;0.15)	3.77E-29	0.5544	1.0867	*
CE-D(15;80000;0.15)	3.29E-27	0.5798	1.8029	*
Patch-D(10;20000;0.05)-2	3.06E-25	0.0776	1.8476	49.2907
Patch-D(10;20000;0.15)-2	5.11E-25	0.1564	1.8252	48.3192
Patch-D(10;20000;0.05)-2(4)	4.08E-25	0.0861	1.8424	48.9142
Patch-D(10;20000;0.15)-2(4)	5.07E-25	0.0916	1.8408	48.7110
Patch-D(10;40000;0.05)-2	5.53E-25	0.0527	1.8612	48.7816
Patch-D(10;40000;0.15)-2	7.16E-25	0.0815	1.8446	48.2375
Patch-D(10;40000;0.05)-2(4)	6.42E-25	0.0718	1.8495	48.4291
Patch-D(10;40000;0.15)-2(4)	7.09E-25	0.0449	1.8663	48.7651

Table 1: *Comparison of performances for Example 1.*

Although the number of states is not very high, we noticed from the experiments that this kind of Markovian reliability model with only one failure state is a hard problem for every approach. We experimented with a large range of θ's in the SFB(θ) and BFB(θ) schemes. Although they provide fast estimations, the relative error is not very small (relative error of 17% or worse). Expectedly, BFB(θ) gives better results compared to SFB(θ), but still poor. There is also not much difference in performance between ZVA and the biasing schemes. ZVA(v_1) gives slightly good results compared to ZVA(v_0). However, none of them gives relative error smaller than 10%.

Also, we experiment it with various CE-D($n; k; \alpha$) and Patch-D($n; k; \alpha$) schemes. For CE-D($n; k; \alpha$), we could not get good performance (nor good estimates!) even not after increasing the number of sample paths in each iteration. Consequently, the EFF performance measure for CE-D($n; k; \alpha$) is not compatible with other methods. Though, Patch-D($n; k; \alpha$) schemes gave excellent performance when the number k of iteration sample paths was sufficiently large. Clearly, Patch-D($k; n; \alpha$) schemes outperform the best bias schemes.

We observe the effects of different ways of clustering and weight factor α used in Equation (6) in the algorithms in the last part of Table 1. We report a few combinations with clustering in groups of 2 starting either with 2 or 4, because larger clusters showed worse results. Increasing the weight factor α from 0.05 to 0.15 gives sometimes worse results, but not always. As expected, if we increase the number of sample paths in each iteration, the relative errors decrease. As a conclusion, the best combination of number of sample paths, clustering and α factor should be found out by trial and error. Although the patching algorithm consumes more computing time, the efficiency is about the same as the other algorithms due to its variance improvements.

Example 2

This is an example from L'Ecuyer and Tuffin (2007). There are again 3 types, this time each with 12 components, with failure rates $\lambda_1 = \varepsilon$, $\lambda_2 = 1.5\varepsilon$ and $\lambda_3 = 2\varepsilon^2$ (unbalanced) and repair rates $\mu_i = 1$ for all i. We only give results for $\varepsilon = 0.001$. There are ample repairmen who work simultaneously on all failed components. There is

no priority or group repair. The system breaks down as soon as at least one component type has less than 2 operational units, i.e.,

$$\mathcal{F} = \{x = (x_1, x_2, x_3) : \exists\ i\ x_i \geq 11\}.$$

The associated failure probability is found by the Gauss-Seidel method to be $3.892E-28$. The results for all the models can be found in Table 2. All results are averages of 5 repetitions, where the sample size in the (final) importance simulations for estimating the rare event probability is $1,000,000$ in all experiments.

Method	**Estimation**	**RE**	**RAT**	**EFF**
BFB(0.7)	3.75E-28	0.6725	1.7597	54.0421
BFB(0.8)	4.43E-28	0.3749	1.7765	54.1473
BFB(0.9)	3.71E-28	0.3746	1.7766	54.3167
ZVA(v_0)	3.88E-28	0.2563	1.8246	55.6483
ZVA(v_1)	3.87E-28	0.0135	1.9306	58.7978
CE-D(7; 5000;0.15)	2.24E-28	0.3228	1.8824	60.1135
CE-D(7;10000;0.15)	2.05E-28	0.1801	1.9180	59.1779
CE-D(7;20000;0.15)	3.42E-28	0.1703	1.9089	57.7000
CE-D(7;50000;0.15)	3.83E-28	0.0048	1.9642	58.0076
Patch-D(2; 300;0.15)-2	3.24E-28	0.0554	1.9454	58.7695
Patch-D(3; 500;0.15)-2	3.84E-28	0.0086	1.9664	58.9652
Patch-D(1;1000;0.15)-2	3.90E-28	0.0140	1.9578	59.7262

Table 2: *Comparison of performances for Example 2.*

Observations in terms of comparison of methods are generally similar to Example 1. ZVA(v_1) provided better estimation, with a relative error around %1.3. Cross-entropy (without patching) showed small relative error when we increased the sample size to $50,000$ in each iteration, whereas Patch-D($n; k; \alpha$) obtained the same relative error with much less sample paths. All the patching results are given in clusters of 2 and for fixed $\alpha = 0.15$.

To check empirically the asymptotic efficiency of our Patch-D($n; k; \alpha$) algorithm for this specific problem setting, the experiments are repeated for increasing number of components (where the failure set remains to be a layer of two), and for decreasing ε values. All the simulation results for asymptotic efficiency can be found in Table 3. It is clear from the table that as ε decreases, meaning as the event gets rarer, the method provides better solutions. In all of the cases we see that the relative error grows linearly with a lower speed where as the probability decreases fast. The last column with RAT values also approves the asymptotic efficiency observation. As the number of components increase, RAT values approach to 2 as in the definition of asymptotic efficiency.

Model	Number Components	Estimation	RE	RAT
$\varepsilon = 0.05$	8	1.14E-06	0.0033	1.8224
3 iterations	12	6.59E-11	0.0081	1.8229
10^3 replications	20	1.56E-19	0.0354	1.8372
	40	1.39E-42	0.3844	1.8788
	60	3.88E-68	0.7191	1.9190
$\varepsilon = 0.01$	8	5.43E-11	0.0025	1.9182
3 iterations	12	4.27E-18	0.0034	1.9371
10^3 replications	20	2.00E-32	0.0068	1.9473
	40	1.53E-68	0.0401	1.9532
	60	4.59E-105	0.1204	1.9603
$\varepsilon = 0.001$	8	5.09E-17	0.0036	1.9396
3 iterations	12	3.86E-28	0.0029	1.9647
10^3 replications	20	1.69E-50	0.0070	1.9662
	40	1.12E-106	0.0320	1.9718

Table 3: *Test of asymptotic efficiency of Patch-D for Example 2.*

Example 3

There are 5 types, with 14 components each, with unbalanced failure rates; 2.5ε, 3ε, ε^2, 6ε, $2\varepsilon^2$ and repair rates $\mu_i = 1$ for all i. The rarity parameter ε is taken to be 0.001. There is a single repairman who applies preemptive priority according to $1 > 2 > 3 > 4 > 5$. Type 1 has the highest priority. All types are repaired one by one as soon as one has failed. The system breaks down as soon as all components of at least one type have failed. All results are averages of 5 repetitions. For results of Example 3, please refer to Table 4.

Method	Estimation	RE	RAT	EFF
BFB(0.7)	6.38E-24	0.5830	1.6753	*
BFB(0.8)	1.99E-21	0.5861	1.6419	*
BFB(0.9)	1.26E-21	0.7865	1.6299	*
BFB(0.95)	8.87E-21	0.8561	1.6588	*
ZRV(v_0)	4.00E-18	0.0965	1.7737	36.8293
ZRV(v_1)	3.98E-18	0.0999	1.7744	36.4315
CE-D(7;10000;0.15)	3.21E-20	0.4006	1.7330	35.8264
Patch-D(7; 500;0.15)-2	8.61E-19	0.1897	1.7529	34.8303
Patch-D(7;1500;0.15)-2	1.84E-18	0.1139	1.7719	33.7477
Patch-D(7;2000;0.15)-2	2.22E-18	0.1373	1.7637	33.3438
Patch-D(7;3000;0.15)-2	2.73E-18	0.1263	1.7606	34.0231

Table 4: *Comparison of performances for Example 3.*

The BFB algorithms gave quite different estimates than all other algorithms with relative errors changing between 50% and 85%, even after having increased the number of sample paths (for BFB only) form 10^6 to 10^8, and thus we conjecture that the correct estimates are given by these other algorithms. The best and the fastest performance among these four methods is given by ZRV(v_0) and ZRV(v_1) with relative error slightly lower than %10. Also, the patching algorithm with only 1500 sample paths gives close performance to ZRV method with

little bit higher relative error. However, since it takes longer time with patching algorithm, the efficiency is lower compared to ZRV.

5.2 Jackson Network

Our second example comprises the well-known product-form Jackson queueing networks. There is ample literature on importance sampling simulation for estimating excessive backlogs (or buffer cq. population overflows) in networks with more or less general network topologies, by algorithms with proven complexity or by heuristic algorithms. We mention here a few.

- Tandem queue and total population overflow: Parekh and Walrand (1989) gave a state-independent heuristic algorithm based on exchanging the arrival rate with the service rate of the bottleneck server. Later, this heuristic has been analyzed in Boer (2006) and Glasserman and Kou (1995) to show that asymptotic optimality is obtained only in certain specific parameter regions. In Dupuis et al. (2007) a state-dependent change of measure has been developed and proven to be asymptotically optimal. The method is based on connections between importance sampling, stochastic games and optimal control.
- Tandem queue and buffer overflow at the second queue: using a Markov additive process representation (Kroese and Nicola 2002) construct a state-dependent change of measure (dependent on the content at the first queue) and prove asymptotic optimality.
- This has been generalized to Jackson networks and buffer overflow at a single target queue in Juneja and Nicola (2005). The resulting change of measure is obtained as a solution to a set of conditions and proven to be asymptotically optimal.
- General topologies and population overflow: Frater et al. (1991) apply the large deviations heuristic of Parekh and Walrand (1989) to come up with a set of nonlinear equations whose (unique) solution serves as a state-independent change of measure. In Nicola and Zaburnenko (2007) heuristic state-dependent algorithms are constructed based on the observation that the state-dependency of the optimal (zero-variance) change of measure is strong along the boundaries and weak and diminishes in the interior. The proposed solution is a state-dependent combination of matrix solutions.
- General Markovian queueing models and rare events: Boer and Nicola (2002) and Boer et al. (2004) report cross-entropy based adaptive state-dependent algorithms. In Boer and Nicola (2002) the algorithm uses boundary layers as we explained in Section 4. The method in Boer et al. (2004) is a three stage adaptive importance sampling algorithm which uses cross entropy to update the arrival and departure parameters together with routing probabilities. First, they estimate the minimum cross-entropy tilting parameter for a small buffer level; next, they use this as the starting tilting parameter value for the estimation of the optimal tilting parameter for the actual buffer level. Lastly, the tilting parameter obtained is used to estimate the population overflow probability.

We will compare our algorithms with the heuristic methods of Boer et al. (2004) (in all our examples) and Nicola and Zaburnenko (2006) (in example 6).

Suppose that the network consists of d queues, Poisson arrivals with rates λ_i $(i = 1, \ldots, d)$, single servers at the queues and exponentially distributed service times with rates μ_i $(i = 1, \ldots, d)$, routing probabilities r_{ij} $(i, j = 1, \ldots, d)$ such that the routing matrix R is substochastic. We make the usual stability assumption: let $(\nu_i)_{i=1}^d$ be the traffic intensities determined as the solution to the traffic equations

$$\nu_j = \lambda_j + \sum_{i=1}^{d} \nu_i r_{ij}, \quad j = 1, \ldots, d.$$

Assumption 2. *For any $i = 1, \ldots, d$ the load $\rho_i = \nu_i/\mu_i < 1$.*

The network is modeled by a continuous-time Markov chain with states $x = (x_1, \ldots, x_d)$ representing the number of jobs present at the queues (including the servers). Again we consider its associated discrete-time Markov chain $(X(t))_{t=0}^{\infty}$ by embedding at the jump times. The state space of the chain has infinite size, being the subspace of all nonnegative integer vectors, $\mathbb{Z}_{\geq 0}^d$. However, we consider only the following two bad sets:

1. Total population exceeds n, i.e.,
$$\mathcal{F} = \{x : \sum_{i=1}^{d} x_i \geq n\}.$$
2. At least one queue exceeds n, i.e.,
$$\mathcal{F} = \{x : \max_{i=1,\ldots,d} x_i \geq n\}.$$

In both cases we can reduce the infinite state space to a finite state space $\mathcal{X}$. The good set consists of the single empty state: $0 = (0, \ldots, 0)$. Thus, similarly as in the reliability example we consider the failure probability (FP)
$$\gamma(0) = \sum_{x \in \mathcal{I}} p_{0x} \gamma(x),$$
where $\gamma(x)$ is the probability of hitting the bad set $\mathcal{F}$ before hitting 0, when the chain would start in state x.

In this example the rarity parameter is the buffer or the population overflow level n. It has been shown in Glasserman and Kou (1995) that
$$\lim_{n \to \infty} \frac{1}{n} \log \gamma_n(0) = -\theta.$$

Example 4

This example of a five-node Jackson network is taken from Boer and Nicola (2002); Boer et al. (2004). We have implemented the three stage method of Boer et al. (2004) (described above), which we denote by CE-3-Stage$(n; k; l_0)$: n is the number of iterations where CE is implemented, k denotes the number of repetitions used in each iteration whereas l_0 refers to the small buffer level to obtain the starting tilting parameters.

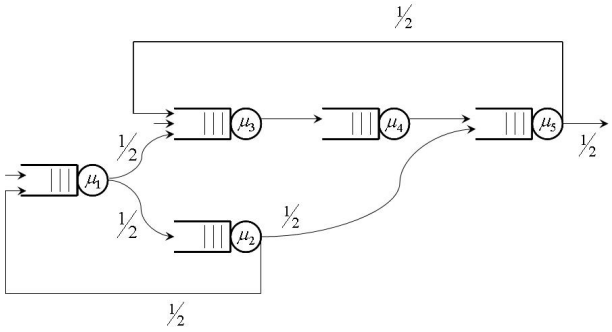


Figure 4: *Jackson Network with 5 queues.*

Jobs arrive at the first queue according to a Poisson process with rate $\lambda_1 = 3$. There are single servers at each queue along with exponentially distributed service times with rates $\mu_1 = 40$, $\mu_2 = 20$, $\mu_3 = 25$, $\mu_4 = 50$ and $\mu_5 = 60$ with all routing probabilities equal to 0.5. Other than server 3, which is the bottleneck queue, all the loads are equal to 0.1. Consider the estimation of the overflow probability that total population exceeds level $l = 80$.

In this example we have $614,918,628$ states, which is quite high to implement the state-dependent change of measure. That is why, here we will only present the results for the face-dependent cross-entropy methods. We used $100,000$ replications in the final estimation by importance sampling. In all Patch-F implementations we started with the lower level value of 10 and increased it by 10 in every patch until we hit the actual overflow level of 80. All results in Table 5 are averages of 5 repetitions.

Method	**Estimation**	**RE**	**RAT**	**EFF**
CE-3Stage(7;100000;5)	7.60E-55	0.0170	1.9576	107.6620
CE-F(30;100000;0.001)	7.72E-55	0.0110	1.9793	108.9570
CE-F(30;400000;0.001)	7.74E-55	0.0096	1.9813	108.5420
Patch-F(10;10000 ;0.001)	7.38E-55	0.0744	1.9498	108.5244
Patch-F(10;50000 ;0.001)	7.77E-55	0.0626	1.9521	107.6283
Patch-F(10;100000;0.001)	7.69E-55	0.0710	1.9499	107.3333

Table 5: *Comparison of performances for Example 4.*

The most remarkable result we see in Table 5 is that CE-F gives estimations with much lower relative error than Patch-F. We also observe that increasing the number of samples used in cross-entropy iteration from $10,000$ to $100,000$ does not improve the estimation in a considerable amount. If we want to get better estimations by Patch-F with lower relative error, we should either increase the number of replications in the (final) importance sampling implementation, or increase the number of replications within patches.

Equal Loads

Next, we simulate the same network, but with all servers having an equal load of 0.1, with $\lambda = 3$, $\mu_1 = 40$, $\mu_2 = 20$, $\mu_3 = 50$, $\mu_4 = 50$ and $\mu_5 = 60$, and all routing probabilities equal to 0.1. For the final estimation by importance sampling we used $1,000,000$ samples and all results are average of 5 repetitions.

The simulation results are presented in Table 6. We found that the number of replications per patch could be less compared to the previous bottleneck problem. The same observation is also true for CE-F. If we compare the results in Boer et al. (2004) with CE-F and Patch-F, we see that CE-F outperforms for this problem.

Method	**Estimation**	**RE**	**RAT**	**EFF**
CE-3Stage(7;1,0000,000;4)	7.68E-16	0.0132	-	-
CE-F(20;100,000 ;0.001)	7.78E-16	0.0058	1.8988	32.1707
CE-F(20;2,000,000;0.001)	7.75E-16	0.0049	1.9081	31.4587
Patch-F(15;100000;0.001)	7.82E-16	0.0204	1.8265	30.7792
Patch-F(15;50000 ;0.001)	7.74E-16	0.0221	1.8226	30.9362
Patch-F(15;10000 ;0.001)	7.63E-16	0.0392	1.8281	31.3089
Patch-F(15;5000 ;0.001)	7.55E-16	0.0329	1.8075	31.0556

Table 6: *Comparison of performances for Example 4 with equal loads.*

Example 5

In this example there are three single server queues. The jobs arrive at the first queue according to a Poisson process of rate $\lambda = 1$. There are single servers at the queues each and exponentially distributed service times with rates $\mu_1 = 4$, $\mu_2 = 3$ and $\mu_3 = 5$. The routing matrix R of the system is as follows:

$$R = \begin{bmatrix} 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & 1 \\ \frac{1}{3} & 0 & 0 \end{bmatrix}$$

Each job can only go out of the system after receiving service at the third station. The corresponding Jackson network is depicted in Figure 5.

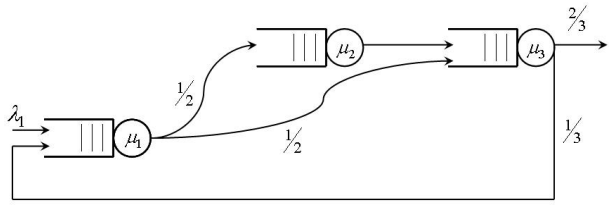


Figure 5: *Jackson Network with 3 queues.*

The traffic intensities are $\nu_1 = 1.5$, $\nu_2 = 0.75$ and $\nu_3 = 1.5$ with corresponding loads $\rho_1 = 0.375$, $\rho_2 = 0.25$ and $\rho_3 = 0.30$. Hence, the system is stable. The failure set is reached whenever at least one of the queues exceeds $n = 30$. Hence, we have 29791 states and $157,500$ transitions with nonzero probability. For the final estimation by importance sampling we used $1,000,000$ samples. We compare the estimations with the algorithm CE-3-Stage$(n; k; l_0)$ mentioned in the previous example. All results in Table 7 are averages of 5 repetitions.

Method	**Estimation**	**RE**	**RAT**	**EFF**
CE-3-Stage(7;100000 ;5)	7.34E-13	0.0329	1.8174	25.1203
CE-3-Stage(7;1000000;7)	7.32E-13	0.0159	1.7837	24.7847
CE-D(15;25000;0.05)	8.74E-16	0.4853	1.6670	*
CE-D(15;25000;0.10)	5.17E-15	0.3685	1.6597	*
CE-D(15;25000;0.25)	3.87E-15	0.6077	1.6394	*
CE-D(10;100000;0.10)	6.71E-16	0.1797	1.7028	*
Patch-D(15;25000;0.10)-10	2.22E-13	0.3954	1.5994	22.7849
Patch-D(15;25000;0.25)-10	4.08E-13	0.1165	1.6730	23.3132
Patch-D(15;10000;0.25)-5	1.72E-13	0.1695	1.6569	23.8863
Patch-D(8;25000;0.25)-5	2.87E-13	0.1074	1.6771	23.6915
CE-F(15; 500;0.05)	6.60E-13	0.0036	1.9073	27.3627
CE-F(15; 500;0.15)	6.60E-13	0.0033	1.9129	27.3359
CE-F(15; 500;0.001)	5.90E-13	0.0262	1.7790	25.8970
CE-F(7;5000;0.15)	6.59E-13	0.0028	1.9214	27.4325
Patch-F(15; 500;0.05)-10	6.60E-13	0.0048	1.8869	26.7790
Patch-F(15; 500;0.05)-15	6.59E-13	0.0036	1.9085	27.2608
Patch-F(15; 500;0.15)-15	6.64E-13	0.0135	1.8580	26.5549
Patch-F(15; 500;0.001)-5	6.64E-13	0.0216	1.7810	24.9873
Patch-F(5;5000;0.05)-15	6.60E-13	0.0031	1.9167	27.4015

Table 7: *Comparison of performances for Example 5.*

If we compare the probability estimations in the second column of Table 7, we can see that the rare event probability estimated by state-dependent CE is quite lower than the probabilities estimated by the other four algorithms. By the "Underestimation Theorem" discussed in Devetsikiotis and Towsend (1993), we can say

that state-dependent CE does not converge even after 10 iterations with 10^5 sample paths. Hence, the efficiency results of CE-D is not comparable with the other efficiency results.

CE-3-Stage, the algorithm introduced in Boer et al. (2004) gives better results, relative error around %1, compared to CE-D and Patch-D. However, for CE-F, sample size of 500 for 15 iterations is even enough to get relative error between %0.3, %0.4. The same number of sample paths and number of iterations are run for different values of weight factor α, which determines the dependency on the original transition probabilities. For this example $\alpha = 0.15$ gives the best answer for both runs with 500 and 5000 sample paths. As we decrease the α value, decreasing the effect of original transition probabilities, we start to get higher relative errors. In the Patch algorithms along with the weight factor α, the number of patches also plays an important role. For instance, if we compare the RE results for patches in clusters of 5 with the rest, we see that they are considerably higher. The best is obtained when we take patches with clusters of 15. That means, as we increase the number of patches, meaning as we get smaller and smaller clusters, the algorithm gets more and more distracted which results in increasing the variance of the estimation.

Example 6

This example of a four node feed-forward network is taken from Nicola and Zaburnenko (2006). We compare the results of our algorithms to those obtained by the methods of Parekh and Walrand (1989) (indicated by PW), Boer and Nicola (2002) (SDA), Nicola and Zaburnenko (2006) (SDH), and Boer et al. (2004) (CE-3-stage). The results of the PW, SDA and SDH methods are given in Nicola and Zaburnenko (2006).

The network can be depicted in Figure 6. Jobs arrive at the first queue according to a Poisson Process with rate $\lambda = 0.074$. There are single servers at each queue along with exponentially distributed service times with rate $\mu_1 = 0.617$, $\mu_2 = 0.024$, $\mu_3 = 0.135$ and $\mu_4 = 0.15$, with the routing probability in the first queue equal to 0.1. We are interested in the total-population overflow probability for level of 25. The parameters chosen in this problem are chosen from the region where PW heuristic does not work. This is shown empirically in Nicola and Zaburnenko (2006). All the results given in Table 8 are the average of 5 repetitions. In our methods, in the final implementation of importance sampling to estimate the overflow probability we use $1,000,000$ sample paths.

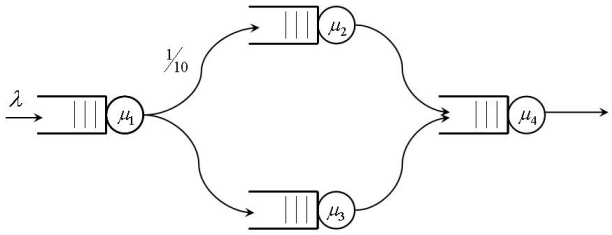


Figure 6: *Jackson Network with 4 queues.*

Method	Estimation	RE	RAT	EFF
PW	1.61E-6	0.0808	-	-
SDA	1.70E-6	0.0011	-	-
SDH	1.71E-6	0.0041	-	-
CE-3-Stage(7;1000000;5)	1.71E-6	0.0173	1.5354	12.2764
CE-F(30;1000 ;0.25)	1.70E-6	0.0082	1.6826	13.3578
CE-F(30;3000 ;0.25)	1.68E-6	0.0074	1.6983	13.4429
Patch-F(10;50000 ;0.001)-2	1.69E-6	0.0087	1.6731	12.5455
Patch-F(20;10000 ;0.05)-2	1.68E-6	0.0090	1.6676	12.6939
Patch-F(20;10000 ;0.15)-2	1.68E-6	0.0097	1.6579	12.5621

Table 8: *Comparison of performances for Example 6*

Experimental results suggest that CE-F gives the best performance among the four approaches we introduce in this paper for this example. CE-D and Patch-D do not converge even with $500,000$ sample paths for 20 iterations. Implementations with more than $500,000$ sample paths cause the EFF performance measure to decrease due to the fact that the relative error is not low and the running time increases considerably. We tried Patch-F with many combinations of cross entropy iterations, number of sample paths, forms of clustering and different weight factors. The best 3 estimations are provided in the table. As observed in the previous Jackson network examples, CE-F gives more stable estimations than Patch-F. Both CE-F and Patch-F provides better estimations than PW. There is also not much difference in the performance between CE-3-Stage and Patch-F, Patch-F giving slightly better results. However, Patch-F, CE-F and CE-3-Stage are worse compared to SDA and SDH. While comparing these methods we have to keep in mind that the results provided in Nicola and Zaburnenko (2006) are obtained by using 10^6 replications in each iteration, where the number of iterations is unknown.

In all of the three examples studying Jackson networks we saw that Patch-F does not provide any better estimation if CE-F already converges. Although we do not face convergence problems with Patch-F, the lowest relative error we can get is higher than %1.

6 Summary & Conclusions

In the previous sections we considered state-dependent approach for two types of problems; reliability models and Jackson networks. We observed that for Jackson networks with higher dimensions the state-dependent cross-entropy method does not give good results. However, if we look at these problems in a more detailed way, it is well known that the transition probabilities in reliability models depend on the state whereas in Jackson networks the transition probabilities are the same for specific types of states. That is the main notion we try to use in Section 4. As we have observed unnecessary usage of state dependency may cause

- enormous increment in the number of decision variables,
- increase in variance and computation time.

Another general observation is that the patching algorithms for both state and face-dependent methods do not provide any improvement when state or face-dependent CE works. However, for Markovian systems where standard cross-entropy is facing some problems with converging, then patching algorithms help cross-entropy method to converge. For every system, there is an optimal way of choosing the patches. If we use less patches than needed, then patching algorithms do not converge, or to put it in another way, we still need more sample

paths in each iteration in order for the patching algorithm to converge. Similarly, if we use more patches than needed, then the change of measure gets so distracted that the variance becomes high. Nonetheless, optimal clustering can only be found through experimentation.

References

[1] Ahamed I., Borkar, V.S., and Juneja, S. Adaptive importance sampling for Markov chains using stochastic approximation. Operations Research 2006; 54, 489-504.

[2] Alexopoulos, C. and Shultes, B.C. Estimating reliability measures for highly-dependable Markov systems using balanced likelihood ratios. IEEE Transactions on Reliability 2001; 50; 265-280.

[3] Asmussen, S. and Rubinstein, R.Y. 1995. Steady-state rare-events simulation in queueing models and its complexity properties. In: J. Dshalalow (Ed.), Advances in Queueing: Models, Methods and Problems, CRC Press; Boca Raton, 1995. p. 429-466.

[4] Boer, P.T. de, and Nicola, V.F. Adaptive state-dependent importance sampling simulation of Markovian queueing networks. European Transactionson Telecommunications 2002; 13; 303-315.

[5] Boer, P.T. de, Kroese, D.P. and Rubinstein, R.Y. Estimating buffer overflows in queueing networks. Management Science 2004; 50(7); 883-895.

[6] Boer, P.T. de. Analysis of state-independent importance sampling measures for the two-node tandem queue. ACM Transactions on Modeling and Computer Simulation 2006; 16; 225-250.

[7] Bucklew, J.A., Ney, P., and Sadowsky, J.S. Monte Carlo simulation and large deviations theory for uniformly recurrent Markov chains. Journal of Applied Probability 1990; 27; 44-59.

[8] Bucklew, J.A. Introduction to Rare Event Simulation. Springer: New York; 2004.

[9] Dayar, T. and Stewart, W.J. Comparison of partitioning techniques for two-level iterative solvers on large, sparse Markov chains. SIAM Journal on Scientific Computing 2000; 21; 1691-1705.

[10] Desai, P. and Glynn, P.W. 2001. A Markov chain perspective on adaptive Monte Carlo algorithms. In: B. Peters, J. Smith, D. Medeiros, and M. Rohrer (Eds.), Proceedings of 2001 Winter Simulation Conference. IEEE Press; 2001. p. 379-384.

[11] Devetsikiotis, M. and Towsend, J.K. An algorithmic approach to the optimization of importance sampling parameters in digital communication system simulation. IEEE Transactions on Communications 1993; 41; 1464-1473.

[12] Dupuis, P. and Wang, H. Dynamic importance sampling for uniformly recurrent Markov chains. Annals of Applied Probability 2005; 15; 1-38.

[13] Dupuis, P., Sezer, D., and Wang, H. Dynamic importance sampling for queueing networks. Annals of Applied Probability 2007; 17; 1306-1346.

[14] L'Ecuyer, P. and Tuffin B. 2007. Effective approximation of zero-variance simulation in a reliability setting. In: J. Sklenar, C. Bertelle and G. Fortino (Eds.), Proceedings of the 2007 European Simulation and Modeling Conference. p. 48-54.

[15] L'Ecuyer, P., Blanchet, J.H., Tuffin, B., and Glynn, P.W. Asymptotic robustness of estimators in rare-event simulation. ACM Transactions on Modeling and Computer Simulation 2008. To appear.

[16] Frater, M., Lennon, T., and Anderson, B. Optimally efficient estimation of the statistics of rare events in queueing networks. IEEE Transactions on Automatic Control 1991; 36; 1395-1405.

[17] Glasserman, P. and Kou, S.-G. Analysis of an importance sampling estimator for tandem queues. ACM Transactions on Modeling and Computer Simulation 1995; 5; 22-42.

[18] Glynn, P.W. and Iglehart, D.L. Importance sampling for stochastic simulations. Management Science 1989; 35; 1337-1392.

[19] Goyal, A., Shahabuddin, P., Heidelberger, P., Nicola, V.F., and Glynn, P.W. A unified framework for simulating Markovian models of highly dependable systems. IEEE Transactions on Computers 1992; 41; 36-51.

[20] Heidelberger, P. Fast simulation of rare events in queueing and reliability models. ACM Transactions on Modelling and Computer Simulation 1995; 5; 43-85.

[21] Ignatiouk-Robert, I. Large deviaitions for processes with discontinuous statistics. Annals of Probability 2005; 33; 1292-1329.

[22] Juneja, S. and Nicola, V.F. Efficient simulation of buffer overflow probabilities in Jackson networks with feedback. ACM Transactions on Modeling and Computer Simulation 2005; 15; 281-315.

[23] Juneja, S. and Shahabuddin, P. Fast simulation of Markov chains with small transition probabilities. Management Science 2001; 47; 547-562.

[24] Juneja, S. and Shahabuddin, P. Splitting based importance sampling algorithm for fast simulation of Markov reliability models with general repair policies. IEEE Transactions on Reliability 2001; 50; 235-245.

[25] Juneja, S. and Shahabuddin, P. 2006. Rare-event simulation techniques: an introduction and recent advances. In: S. Henderson and B. Nelson (Eds.), Handbooks in Operations Research and Management Science, Vol. 13: Simulation. Elsevier; Amsterdam; 2006. p. 291-350.

[26] Kollman, C., Baggerly, K., Cox, D., and Picard, R. Adaptive importance sampling on discrete markov chains. Annals of Applied Probability 1999; 9; 391-412.

[27] Kroese, D.P. and Nicola, V.F. Efficient simulation of a tandem Jackson network. ACM Transactions on Modeling and Computer Simulation 2002; 12; 119-141.

[28] Nicola, V.F. and Zaburnenko, T.S. Efficient importance sampling heuristics for the simulation of population overflow in Jackson networks. ACM Transactions on Modeling and Computer Simulation 2007; 17; issue 2.

[29] Parekh, S. and Walrand, J. A quick simulation method for excessive backlogs in networks of queues. IEEE Transactions on Automatic Control 1989; 34; 54-66.

[30] Randhawa, R. and S. Juneja, S. Combining importance sampling and temporal difference control variates to simulate Markov chains. ACM Transactions on Modelling and Computer Simulation 2004; 14; 1-30.

[31] Rubino G. and Tuffin, B. (Eds.). 2009. *Rare event simulation using Monte Carlo methods*. Wiley.

[32] Rubinstein, R.Y. and Kroese, D.P. 2004. *The cross-entropy method: a unified approach to combinatorial optimization, Monte-Carlo simulation and machine learning*, Springer.

[33] Shahabuddin, P. Importance sampling for simulation of highly reliable Markovian systems. Management Science 1994a; 40; 333-352.

[34] Shahabuddin, P. Fast transient simulation of Markovian models of highly dependable systems. Performance Evaluation 1994b; 20; 267-286.

[35] Stewart, W.J. 1999. Numerical methods for computing stationary distributions of finite irreducible Markov chains. Chapter 3 in *Advances in Computational Probability*. W. Grassmann (Ed.), Kluwer.

[36] Stewart, W.J. 2007. Performance modeling and Markov chains. In *Formal Methods for performance Evaluation*. M. Bernardo and J. Hillston (Eds.): SFM 2007, LNCS 4486, Springer, pp. 1-33.