Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2013) 21: 309 – 329
© TÜBİTAK
doi:10.3906/elk-1105-50

A new technique for rare-event simulation based on partition of the region

Amir Jalaly BIDGOLY, Mohammad Abdollahi AZGOMI*
School of Computer Engineering, Iran University of Science and Technology, Tehran, Iran

Received: 25.05.2011 • **Accepted:** 26.12.2011 • **Published Online:** 22.03.2013 • **Printed:** 22.04.2013

Abstract: Rare events are an important limitation of discrete-event simulation in some areas of applications. The existing rare-event simulation (RES) techniques, such as importance sampling (IS), splitting, and RESTART, have been used for RES so far. However, these techniques not only have some limitations, but also are not general or easy to use. In this paper, the partition of the region (PoR) method is used as a variance reduction technique in a new RES technique for usage with a wide range of modeling and simulation languages. Three variants of the proposed technique will be introduced: simple PoR, recursive PoR, and advanced PoR. Simple and recursive PoR methods are powerful techniques that have some important features such as bounded relative error; however, they are restricted to some specific models. Advanced PoR is a more general technique that integrates the PoR method and the IS technique. To evaluate the efficiency of the proposed technique and its variants, the comparative simulation results of some illustrative examples using stochastic activity networks will be presented.

Key words: Rare-event simulation, variance reduction technique, partition of the region, simple partition of the region, recursive partition of the region, advanced partition of the region

1. Introduction

The evaluation of useful system measures, such as performance, reliability, and availability, is very effective in the selection of appropriate parameters in the development process of new systems or tuning real systems. Discrete-event simulation is the most important method for the evaluation of a wide range of systems in industry and academia. However, this method has some challenges. In mission- and safety-critical applications and many other areas, the existence of some relevant rare events causes the traditional discrete-event simulation method to be inefficient. High-reliable systems, insurance institutions, and investigation of the statistical properties of physics and chemistry models are some examples of such applications. Standard simulation techniques are not practical in such applications due to the required long simulation time. In such situations, rare-event simulation (RES) techniques are used, which increase the occurrence probability of rare events, usually by utilizing an accelerator.

Importance sampling (IS) [1–3] and splitting [4–6] are 2 famous variance reduction techniques (VRTs) that have widely been used for RES. The following is a brief description of these 2 methods [4]:

- The IS technique increases the probability of rare events by changing the probability laws that drive the evolution of the system. The model is then biased, but the bias is compensated by the introduction of a function called the likelihood ratio, which is the ratio of the sample path in the original model to the altered

*Correspondence: azgomi@iust.ac.ir

model. The major difficulty in applying this method is to figure out how to change the probabilities, in which the IS estimator has much smaller variance than the original one.

- In the splitting method, the probability laws remain unchanged, but an artificial drift toward the rare events is created by terminating with some probability trajectories that seem to go away from it and by splitting (cloning) those that are going in the right direction. A class of splitting is called RESTART [7–9], which is an abbreviation of "REpeative Simulation Trials After Reaching Thresholds". RESTART considers a threshold and kills the trajectories when they cross it.

In 2 previous papers [10,11], we presented the initial ideas and preliminary results of a new RES technique, which uses the partition of the region (PoR) method [12] as a VRT. In this paper, the finalized and extended descriptions of the proposed technique, corresponding simulation algorithms, examples, and results are presented.

The proposed technique is introduced in 3 variants: simple PoR, recursive PoR, and advanced PoR. Simple and recursive PoR methods are powerful techniques that have some important features such as bounded relative error (BRE); however, they are restricted to some specific models. Advanced PoR is a more general technique that integrates the PoR method and the IS technique. To evaluate the efficiency of the technique and its variants, some illustrative examples using stochastic activity networks (SANs) [13] are given. However, the proposed technique may be used for the RES of any stochastic discrete-event system (SDES) model. The examples show a considerable improvement compared to naïve simulation.

The rest of this paper is organized as follows. In Section 2, we review the related works. Section 3 gives some background information. The motivations for this work are mentioned in Section 4. The proposed techniques are introduced in Section 5. In Section 6, we present the simulation algorithms for the proposed techniques. Some illustrative examples are given in Section 7. Finally, the paper ends with conclusions in Section 8.

2. Background

In this section, we present some background concepts that are used in the rest of this paper.

2.1. Monte Carlo method

Very often, scientific problems lead to the computation of integrals or sums. Suppose the goal is finding the answer of the following integral [14]:

$$I = \int_D g(x)dx \tag{1}$$

In most of the cases, the exact analytical calculation of this integral is directly impossible. In these cases, the integral could be computed using the Monte Carlo method [12] as a well-known approximation method. Based on the Monte Carlo method, the above integral can be estimated by generating n independent samples $(g_1(\varepsilon), g_2(\varepsilon), \ldots, g_n(\varepsilon))$ of $g(\varepsilon)$ and taking the estimator as:

$$I \approx \gamma_n = \frac{1}{n}\sum_{i=1}^{n} g(x_i). \tag{2}$$

In rare-event estimating, most of the samples of $g(x_i)$ would be 0, and therefore n would have to be quite large to achieve a reliable result for I. The central limit theorem is useful in developing a confidence interval (CI) for the estimate and can be used to determine the n necessary for accurate estimation [15]. For a large n, an approximate $(1\ -\alpha)$ 100% CI for γ_n is given by:

$$\gamma_n(P) \pm z_{\alpha/2}\sqrt{\frac{\sigma_P^2(I(\varepsilon))}{n}}, \tag{3}$$

where $\sigma^2(I(\varepsilon))$ denotes the variance of $I(\varepsilon)$ and z_x is a number that satisfies the relation $P(N(0,1) \geq z_x) = x$. Here, $N(0,1)$ denotes a normally distributed random variable with mean 0 and variance 1. On the other hand, the variance of $I(\varepsilon)$ is equal to $(1-\gamma)\gamma$. According to the above equation, for a reliable estimator, n must be chosen to make the relative confidence interval sufficiency small [1]:

$$\frac{CI}{\gamma} = z_{\alpha/2}\sqrt{\frac{(1-\gamma)}{n\gamma}}. \tag{4}$$

This implies that as $\gamma \to 0$, $n \to \infty$ to obtain a reasonable level of relative accuracy. Thus, naïve simulation becomes an infeasible proposition for sufficiently rare events. The basic solution for this problem is to use an appropriate VRT, such as IS, which is introduced in the following subsection.

2.2. Importance sampling

IS [1] is one of the most effective and significant methods for reducing the variance of estimators. Let ε denote a rare event in the probability measure P aiming to estimate the probability of this event. Consider the new probability P^* such that $P^*(A)$ >0, where $P(A)$ >0 for $A \subset \varepsilon$. In the IS technique, the probability of ε can be estimated using the new probability measure as follows:

$$P(\varepsilon)=E_P(I(\varepsilon))=\int I(\varepsilon)\,dP=\int I(\varepsilon)\,\frac{dP}{dP^*}dP^*=\int I(\varepsilon)\,LdP^*=E_{P^*}[L.I(\varepsilon)], \tag{5}$$

where $L = dP/dP^*$, which is the quotient of the probability measures and is called the likelihood ratio. This suggests the following procedure [1] for IS simulation for estimating ε: generate n independent samples $(I_1(\varepsilon);\ L_1)$, $(I_2(\varepsilon);\ L_2),\ldots,(I_n(\varepsilon);\ L_n)$ of $(I(\varepsilon);\ L)$. Next, the following equation is used as an unbiased estimator of ε:

$$\gamma_n(P^*)=\frac{1}{n}\sum_{i=1}^{n} I_i(\varepsilon)\,L_i. \tag{6}$$

In particular, the aim of IS is to find a P^* that minimizes the variance of the output $L \times I(\varepsilon)$, or, equivalently, the relative error. In some cases, a new distribution P^* that increases the probability ε is a proper choice, but usually there exist more constraints in selecting a good one. Some choices may not reduce the variances (and thus the simulation time), or may even make them infinite. For more information about the characteristics of a good distribution function for IS, please see [1,16,17].

The IS technique may change the probability of a model in 2 ways. In the simpler mode, a fixed change of the measure is used throughout the simulation. This method is called the static IS. The second method, which is called the adaptive IS, involves updating and learning an improved change of measure based on the simulated sample paths. However, most published studies have focused on the first method. For more information, please see [7,18].

2.3. Bounded relative error

In the context of RES, the asymptotic robustness of an estimator is qualified with some properties, which is also called asymptotic optimality [16]. BRE is one the most important properties in this regard [3,19].

Consider ε as a rare event and $\gamma(\varepsilon)$ as its estimator. This estimator does not have BRE unless the sample size $n(\varepsilon)$ grows at least at the same rate as $1/\gamma(\varepsilon)$ when $\varepsilon \to 0$. In other words, the estimator has BRE if and only if we have:

$$\frac{\sigma\left(\gamma_n\right)}{\gamma}<\infty, \tag{7}$$

where $\sigma\left(.\right)$ denotes the standard deviation. This feature guarantees that regardless of how small the probability of a rare event is, the computational cost is limited and the simulation is possible. If a simulator has BRE properties, all of the other robustness properties are verifiable [20].

2.4. Partition of the region

Again, consider the integral defined in Eq. (1). In the PoR method [12], the region D will be broken into 2 independent regions, and so Eq. (1) can be redefined as follows:

$$I\approx\int_D g\left(x\right)dx=\int_{D_1} g\left(x\right)dx+\int_{D_2} g\left(x\right)dx. \tag{8}$$

Assume that region D_2 lacks any rare events; thus, the following integral could be estimated by a traditional estimator:

$$I_1=\int_{D_1} g\left(x\right)dx. \tag{9}$$

Let us define a probability density function (PDF) as follows:

$$h\left(x\right)=\left\{\begin{array}{cc}\frac{f_X(x)}{1-P} & x\in D_2\\ 0 & otherwise\end{array}\right., \tag{10}$$

where $f_x(x)$ denotes the PDF, which is used in the sample mean Monte Carlo [12] method, and P is equal to the following probability:

$$P=\int_{D_1} f_X\left(x\right)dx. \tag{11}$$

Let us rewrite Eq. (8) using the above equations:

$$\begin{array}{c}I=I_1+\int_{D_2} g\left(x\right)dx=I_1+\int_{D_2}\frac{g(x)}{h(x)}h\left(x\right)dx=\\ I_1+E\left[\frac{g(x)}{h(x)}\right]=I_1+\left(1-P\right)E\left[\frac{g(x)}{f_X(x)}\right]\end{array}, \tag{12}$$

and an estimator for I could be defined as in Eq. (13), which is called the PoR estimator [12]:

$$\tilde{I}=I_1+\left(1-P\right)\frac{1}{N}\sum_{i=1}^{N}\frac{g\left(x_i\right)}{f_X\left(x_i\right)}. \tag{13}$$

In practice, D_2 will be defined such that it does not have any rare events (i.e. $I_1=0$). Therefore, in simulation, it is not required to simulate this part. We use this method in our proposed techniques, which will be introduced in the rest of this paper.

2.5. Stochastic activity networks

SANs [13,21], as a general and stochastic extension of Petri nets, are powerful and flexible models for concurrent and distributed systems. These models are supported by several powerful modeling tools, such as UltraSAN [22], Möbius [23], and PDETool [24].

The elements of SANs include place, gate, and activity [13]. Places are the same as Petri nets and are shown by circles in the graphical representation. Gates are used to connect places and activities and have 2 types: input and output. Input gates (shown as ⇉▷–) connect one or more places to a single activity and have an enabling predicate and an input gate function. Output gates (shown as –▷⇇) connect an activity to one or more places and have only an output gate function. Moreover, there are 2 types of activities: timed and instantaneous. Instantaneous activity (shown as |) describes events that occur instantaneously. Timed activity (shown as ▌) represents processes that usually take some time to complete. Instantaneous activities model nondeterminacy, while timed activities represent parallelism [21].

An activity is called "enabled" when all of the input gates connected directly to the activity hold true (i.e. the enabling predicates of the input gates hold true) and all of the input places have at least one token. When a timed activity is enabled, it starts to be completed. The activity completion time is taken from the activity's time distribution function. An enabled instantaneous activity completes immediately. After completion of an activity, first the input gate functions and then the output gate functions will be executed and change the marking of the model. For more information about SANs, please see [13].

3. Related work

In the literature, VRTs are known as the main method for the fast simulation of rare events [1,14]. Although in some studies, some other methods were also provided for this purpose, most of these methods require some data produced in numerical analysis, and they cannot be called the absolute simulation method (such as in [25,26]).

Variance reduction can be viewed as a means to use the known information about the problem. In fact, if nothing is known about the process to be simulated, variance reduction cannot be directly achieved. At the other extreme, that is complete knowledge, the variance is equal to 0 and there is no need for simulation [12].

In practice, simulation with 0 variance is impossible because reducing the variance to 0 in each of the existing techniques requires having the answer before running the main simulation. In having the answer, there is no need to run the simulation model.

Among different existing methods, IS and splitting have gained more attention. The main difficulty in the IS method is how to define the new probability laws for the model. A bad choice not only will not reduce the variance, but it may increase or even infinitize the variance, which will increase the simulation time. Therefore, choosing a new distribution for IS is an important open problem, and many studies are still involved with how to figure out an appropriate method for this purpose.

Importance splitting, unlike IS, does not change the probability laws of the model, but an artificial drift toward rare events is created by terminating with some of the probability trajectories that seem to go away from it and by splitting (cloning) those that are going in the right direction. The most important advantage of this method is related to the size of the information required before the simulation. Although the importance splitting method requires some information to run, the data size is considerably less than that of the IS method. This point makes it simpler than IS.

The RESTART technique [8] is a class of splitting that is appropriate for a broader range of models. This method, unlike splitting, can also be used for steady-state simulation.

As mentioned earlier, BRE is an asymptotic optimality property of estimators. The other important property of estimators is called logarithmic efficiency (LE) [19,27]. Both BRE and LE can often show the performance of a RES method, but in many of the existing techniques, proof of them is not always provided. Therefore, some other properties have been introduced in the literature that are generally based on these 2 properties. Bounded normal approximation [28], asymptotic good estimation of the mean [29], vanishing relative centered moment [16], and BRE and LE of order k [16] are some examples of these properties.

4. Motivations and aims

Despite the advantages of the existing RES techniques, they have not attracted enough interest in industrial applications. There are at least 3 technical reasons for this problem that are common among almost all of the RES techniques [30]:

1. We must have enough background knowledge about the model to apply RES techniques. An example is the definition of the importance function and the levels in a splitting setting. The automatic derivation of a good importance function for a given model is still an open problem.
2. RES algorithms have been developed or adapted for some restricted types of models.
3. There is a lack of powerful and easy-to-use simulation software.

Our aim has been to propose a method to make RES a general tool and an easy-to-use technique to be used with a wide range of modeling and simulation languages. With this aim, we introduce the PoR family of RES techniques in the next section. These techniques break the model into 2 regions, which are named important and trivial regions, and only the important region will be simulated using a likelihood ratio. Although the proposed technique does not solve all of the above problems, it has some significant characteristics. For example, some variants of the PoR technique can be used with any general simulation tool while it satisfies BRE as an important property. Although this is not always possible and PoR techniques cannot take the place of previous ones, it is an attempt to take a step toward simplicity.

5. The proposed techniques

In this section we introduce PoR techniques as a new family of RES techniques, which is based on the PoR method. We introduce 3 variants of PoR techniques: simple PoR, recursive PoR, and advanced PoR. The simple and recursive PoR techniques do not require any special dedicated simulator and can be used by the existing simulators (e.g., Möbius), which is an important improvement over the existing RES techniques. Moreover, we believe that the required information for the simulation of a model is not as much as in the other techniques, such as IS. Therefore, setting up a meaningful simulation study with PoR techniques is easier. The advanced PoR technique is a more general technique that integrates the PoR and IS techniques. This technique can be used with a wide range of models, while the simple and recursive PoR techniques can be used with certain classes of models.

In the rest of this section, we introduce the definitions and concepts of these techniques.

5.1. The simple PoR technique

Consider a stochastic discrete-event model with the state space Ω, transition (or event) set, and probability measure P. Assume that the aim is to estimate the probability of an event ε, which is a subset of Ω and denotes a rare event. This probability can be defined as follows:

$$P(\varepsilon) = \gamma = \int_{\Omega} I(\varepsilon)\, dP, \tag{14}$$

where $I(\varepsilon)$ is an indicator for ε, whose value is equal to 1 if ε occurs; otherwise, it is equal to 0. Let us define $\acute{I}$ as another indicator function on the state space, which can partition Ω into the following 2 separate sets:

$$\begin{gathered} \acute{I}\colon \Omega \to \{0,1\} \\ \Omega = \Omega_1 \bigcup \Omega_2 | \forall \omega \in \Omega : \begin{cases} \omega \in \Omega_1 & \acute{I}(\omega) = 1 \\ \omega \in \Omega_2 & \acute{I}(\omega) = 0 \end{cases} . \end{gathered} \tag{15}$$

In the above equation, Ω_1 and Ω_2 are respectively called the important region and the trivial (eliminated) region corresponding to D_1 and D_2, respectively, in the PoR method. In the simple PoR technique [10,11], the indicator function is called the partitioner function. As the goal is estimating ε, it must be chosen such that all of the rare events are included in the important region (i.e. $\varepsilon \subset \Omega_1$). Partitioning is generally a heuristic problem, but in most of the models, the partitioner function could simply be chosen such that it puts all of the rare events in the important region and all of the remaining events in the trivial region. For example, in the simulation of an M/M/1/20 queue for estimating the blocking probability, the state space of the model can be partitioned into 2 regions such that the important region includes those states that have more than 15 customers in the queue, and the trivial region will then contain all of the states where there are less than 16 customers in the queue. Another example is evaluating the unreliability of a high-dependable system, where the partitioner function may put all of the fail events in the important region and the repair events in the trivial region.

One may expect that partitioning is always done by putting rare events in one side and the others in another side; more knowledge may be needed about the model to choose a suitable indicator function in complicated models, especially when a rare event needs or depends on an event in the second region. As mentioned earlier, it is still heuristic. After partitioning the model using the partitioner function, the following equations can be used for the estimation of $P(\varepsilon)$:

$$\left.\begin{aligned} P(\varepsilon) = \textstyle\int_{\Omega} I(\varepsilon)dP = (1-\bar{P}) \int_{\Omega_1} I(\varepsilon)dP + \bar{P} \int_{\Omega_2} I(\varepsilon)dP \\ \textstyle\int_{\Omega_2} I(\varepsilon)dP = 0 \end{aligned}\right\} \Rightarrow P(\varepsilon) \tag{16}$$

$$= (1-\bar{P}) \textstyle\int_{\Omega_1} I(\varepsilon)dP.$$

The above equation shows how $P(\varepsilon)$ can be estimated when the trivial region is thrown away. The probability $(1-\bar{P})$ that is used here helps to bias the estimator and is called the simple PoR likelihood ratio. On the other hand, eliminating a large part of the state space causes a meaningful increase in the occurrence of the states in the important region that includes the rare event. $\bar{P}$ is equal to the sum of the states belonging to Ω_2 that are removed from the model and corresponds to the probability P in Eq. (11). In practice, it can be computed using the following indicator function:

$$\begin{gathered} \bar{P} = P\left[l(\omega)\right] \\ l(\omega) = \begin{cases} 1 & \omega \in \Omega_2 \\ 0 & \omega \in \Omega_1 \end{cases} . \end{gathered} \tag{17}$$

Since the second region contains no rare event, the likelihood ratio can be estimated simply using a naïve simulator. Finally, the simple PoR estimator is defined as the following equation using N independent samples of $I_i(\varepsilon)$:

$$\gamma_n = \frac{1}{N}\sum_{i=1}^{N} I_i(\varepsilon)(1-\bar{P}). \tag{18}$$

5.2. The recursive PoR technique

The simple PoR technique provides a powerful tool for modelers to increase the probability of some events of the model by eliminating the trivial events. The increase in the probability has a direct relationship with the total probability space of the trivial state space region. This means that removing more states leads to more of an increase in the occurrence probability of the desired event. Evaluating the probability $\bar{P}$ is an important step in the context of the simple PoR technique. $\bar{P}$ does not contain a rare event, and so there is no need to use a rare-event simulator. However, the precision of its value has an important effect on the main result. For example, if the desired event is a rare event with the probability of 10^{-10}, it is required to increase its probability by at least 10^{6} times to have an appropriate simulation time. This means that the model should be partitioned such that $\bar{P} = 0.999999$. This value is required to be estimated with a confidence interval of about 10^{-7} in order to be reliable enough for simple PoR. In this case, a simple simulation almost leads to 1 as an answer for this value.

We now present the recursive PoR technique to overcome the above issue, which can be considered as an extension of the simple PoR technique.

Again, consider the model defined in the previous section for simple PoR. Let us partition the state space in k separate sets as follows:

$$\begin{matrix} \Omega_{2k} \subset \Omega_{2(k-1)} \ldots \subset \Omega_{22} \subset \Omega_{21} = \Omega_2 \\ \forall i, 0 < i < k : \Omega_{Ri} = \Omega_{2i} - \Omega_{2(i+1)} \\ \Omega_{Rk} = \Omega_{2k} \end{matrix} , \tag{19}$$

where Ω_2 is the trivial region in the simple PoR technique. Each member of the Ω_{Ri} set is called the trivial region in the ith step. In practice, the sum of the probability of states in a trivial region should be neither too close to 1 nor too close to 0. In this case, however, the probability of all of the trivial regions is too close to 1, but breaking it into k sets helps to estimate it accurately via k simple simulations. Let us redefine the partitioner function, i.e. the recursive partitioner function, as follows:

$$\begin{matrix} \acute{I} : \Omega \rightarrow \{0 \ldots, k\} \\ \Omega = \Omega_1 \bigcup \Omega_{R1} \ldots \bigcup \Omega_{Rk} | \forall \omega \in \Omega : \begin{cases} \omega \in \Omega_1 & \acute{I}(\omega) = 0 \\ \omega \in \Omega_{Ri} & \acute{I}(\omega) = i; 0 < i \le k \end{cases} \end{matrix} . \tag{20}$$

Using the above partitioner, Eq. (14) can be rewritten as follows:

$$\begin{aligned} P(\varepsilon) &= \int_{\Omega} I(\varepsilon)\,dP{=}\left(1{-}\bar{P}_1\right)\int_{\Omega_1+\Omega_{22}} I(\varepsilon)\,dP{+}\bar{P}_1\int_{\Omega_{R1}} I(\varepsilon)\,dP \\ &= \left(1{-}\bar{P}_2\right)\left(1{-}\bar{P}_1\right)\int_{\Omega_1+\Omega_{23}} I(\varepsilon)\,dP{+}\bar{P}_2\int_{\Omega_{R2}} I(\varepsilon)\,dP \\ &= \left(1{-}\bar{P}_k\right)\prod_{i=1}^{k-1}\left(1{-}\bar{P}_i\right)\int_{\Omega_1} I(\varepsilon)\,dP{+}\bar{P}_k\int_{\Omega_{Rk}} I(\varepsilon)\,dP{=}\prod_{i=1}^{k}\left(1{-}\bar{P}_i\right)\int_{\Omega_1} I(\varepsilon)\,dP\int_{\Omega_{Ri}} I(\varepsilon)\,dP{=}0, \end{aligned} \tag{21}$$

where each of the $\bar{P}_i$ can be computed using the following equation:

$$\begin{gathered} \bar{P}_i{=}P_{[\Omega_1+\Omega_{2i}]}\left[l_i(\omega)\right]=\textstyle\int_{\Omega_1+\Omega_{2i}} l_i(\omega)\,dP \\ l_i(\omega)=\begin{cases} 1 & \omega{\in}\Omega_{Ri} \\ 0 & otherwise \end{cases} \quad . \end{gathered} \tag{22}$$

In the above equation, the expression $\prod_{i=1}^{k}\left(1-\bar{P}_i\right)$ is called the likelihood ratio of the recursive PoR technique.

5.3. Variance reduction in the simple/recursive PoR technique

In this subsection, the variance reduction as well as the time reduction of the simple PoR technique will be discussed. Moreover, we discuss the robustness property of this technique. Since both simple and recursive PoR are in the same class, the relationship defined for one of them is true for the other. In the following text, the simple PoR technique is considered.

Consider a rare event $\varepsilon$and its indicator function $I(\varepsilon)$, where $\gamma = P(\varepsilon)$. The variance of a naïve estimator is equal to:

$$\sigma^2\left[\gamma_n\right]{=}E\left[I^2(\varepsilon)\right]-E^2\left[I(\varepsilon)\right]=\gamma{-}\gamma^2 \xrightarrow{\gamma\to 0} \gamma. \tag{23}$$

Assume that there is another estimator, γ_n^{PoR}, which uses the simple PoR technique. The variance of this estimator is as follows:

$$\sigma^2\left[\gamma_n^{PoR}\right]{=}E\left[L^2.I^2_{PoR}(\varepsilon)\right]-E^2\left[L.I^2_{PoR}(\varepsilon)\right]{=}L^2\gamma{-}\gamma^2 \xrightarrow{\gamma\to 0} L^2\gamma, \tag{24}$$

where$L$is the likelihood ratio that is defined for simple PoR [see Eqs. (16) and (22)]. Since the likelihood ratio is constant during the simulation, the following equation is true:

$$\frac{\sigma^2\left[\gamma_n^{PoR}\right]}{\sigma^2\left[\gamma_n\right]}{=}\frac{L^2.\gamma}{\gamma}{=}L^2{=}\left\{\begin{array}{ll} \left(1{-}\bar{P}\right)^2 & \text{Simple } PoR \\ \prod_{i=1}^{k}\left(1{-}\bar{P}_i\right)^2 & \text{Recursive } PoR \end{array}\right\}=\left[1{-}P\left(\Omega_2\right)\right]^2=\left[P(\Omega_1)\right]^2 \ . \tag{25}$$

Hence, the guaranteed variance reduction by a factor of $\left[P(\Omega_1)\right]^2$ is achieved. Therefore, the central limit theorem can be used to prove the reduction in the simulation time with the order of at least $\left[P(\Omega_1)\right]^2$.

If the state space of the model is large enough and the simulation result is reliable enough, we can claim that the occurrence of the rare event can be increased to any desired value. To do so, the trivial region should be large enough. The probability of the rare event would be increased by the factor of $1/(1 - P(\Omega_2))$. Therefore, each model can be partitioned such that:

$$P\left[I_{PoR}(\varepsilon)\right] \geq \beta > \tag{26}$$

and

$$\frac{\sigma_{PoR}(\gamma_n^{PoR})}{\gamma} \leq \frac{\sqrt{E\left[L^2.I_{PoR}(\varepsilon)\right]}}{\gamma} \leq \frac{1}{\sqrt{\beta}} \tag{27}$$

Regarding the above relationship, the simple PoR technique satisfies the BRE property, which is the most powerful among the robustness properties. Other properties can be proven using this property. This is an important advantage for a RES technique. In other techniques, the likelihood ratio cannot be considered constant during the simulation; hence, their robustness property could not be proven simply.

5.4. Advantages and restrictions of the simple/recursive PoR techniques

In comparison with the other RES techniques, the simple PoR technique has some important advantages, which can be summarized as follows:

1. It is simple and easy to use.
2. It does not need any special tools for the simulation of models based on this method.
3. It satisfies the BRE property in the simulation of models.

The most attractive feature of simple PoR is that it is not dedicated to any special simulator and can be used by the existing simulators. Using the other techniques, such as IS, splitting, or RESTART, it is necessary to adapt the simulator by the rules in accordance with the defined relationships.

On the other hand, the lack of a powerful tool is one of the issues in RES. The algorithms presented for simple PoR in the next section show that this method uses only 2 naïve simulators. This feature can be considered as the most important advantage of this technique, which makes it possible for it to be used with any available tool. Moreover, finding a part of the model to remove does not require as much information about the evaluated model as other RES techniques (such as IS) require. Therefore, it is easier to set up a meaningful simulation study. In addition, as we mentioned in the previous section, the BRE property is always proven in simple PoR, whereas in the other techniques, the proof of this property is sometimes impossible.

However, as mentioned earlier, simple PoR could take the place of others where this technique has a major limitation as well. There are 2 conditions that must be considered in selecting the trivial region. First, the trivial region must be free of rare events (i.e. $\Omega_2 \cap \varepsilon = \emptyset$). Second, the important region should be independent of the trivial region. The former is easy to achieve, but the latter is a major limitation for this technique. This limitation and the term of independency in this context are described in the following text.

Consider a model with state space Ω, which is partitioned into Ω_1 and Ω_2. Assume $\bar{P}$ is the likelihood ratio defined for simple PoR. To use this technique, the model should have the following preconditions:

$$\forall \omega \in \Omega : P_{Orig}(\omega) = P_{Altered}\left(\omega * \left(1 - \bar{P}\right)\right), \tag{28}$$

where *Orig* and *Altered* refer to the model before and after partitioning, respectively (i.e. Ω and Ω_1). In the worst case, the above expression should at least be true for the rare event (i.e. ε) and the model should have the following condition:

$$\forall \omega \in \varepsilon : P_{Orig}\left(\omega\right) = P_{Altered}\left(\omega * \left(1 - \bar{P}\right)\right). \tag{29}$$

It seems that some systems (for example, queuing systems or high-reliable systems without any specific characteristics, such as error propagation or group repair) satisfy the above preconditions; however, the above relationship is not generally true, even in some simple models. In the following text, a model is studied that cannot be run with simple PoR, and then we introduce the advanced PoR technique to overcome this problem.

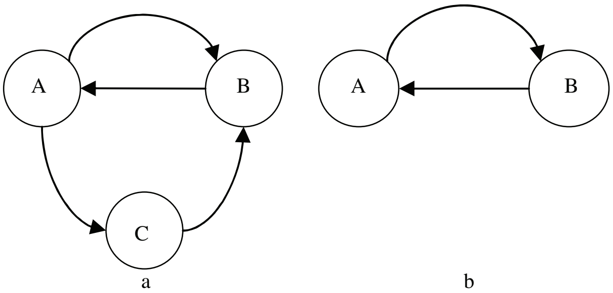

Figure 1. A simple CTMC model: a) before and b) after using simple PoR.

A simple continuous-time Markov chain (CTMC) model is shown in Figure 1a. This model has 3 states, in which all of the output transitions for each state have the rate of 1. The model in Figure 1b is partitioned into 2 areas $\Omega_1 = \{A, B\}$ and $\Omega_2 = \{C\}$. In the original model, the steady-state probability of the state C is equal to 0.25, and so the likelihood ratio of simple PoR is equal to $\bar{P} = P\left(C\right) = 0.25$. If the aim of this model is finding the steady-state probability of state A, using the simple PoR technique, the result will be as follows:

$$\left.\begin{array}{c} P_{PoR}\left(A\right) = P_{Altered}\left(A\right) * \left(1 - \bar{P}\right) = 0.5 * 0.75 = 0.375 \\ P_{Org}\left(A\right) = 0.25 \end{array}\right\} \Rightarrow P_{PoR}\left(A\right) \neq P_{Org}(A). \tag{30}$$

$P_{Org}(A)$ and $P_{Altered}\left(A\right)$ are the steady-state probability of state A in the original and partitioned models, respectively, and $P_{PoR}\left(A\right)$ is the result using simple PoR. Therefore, the result obtained by simple PoR is not the same as the real answer. This example clearly shows that the simple PoR technique could not be used with all models. In the next subsection, this problem will be solved by a modified method that integrates PoR and the IS technique and is called advanced PoR.

5.5. The advanced PoR technique

As mentioned above, despite numerous advantages of the simple PoR technique, it has some limitations and drawbacks that make it not usable for all models. Here we introduce the third variant of PoR techniques, called advanced PoR, to remove the restrictions of simple and recursive PoR by changing the way of partitioning models.

Consider a stochastic model with the state space Ω, transition set, and rare event ε, as presented in the previous section. Suppose there are 2 separate regions, Ω_1 and Ω_2 (such that $\Omega_1 \cup\ \Omega_2 = \Omega$), obtained by defining a partitioner function $\acute{I}$. Let us define a new partitioning in the transition set using the following existing regions:

$$\forall\pi \in \Pi,\omega_i,\omega_j\in \Omega :\omega_i \xrightarrow{\pi} \omega_j \begin{cases} \pi \in\Pi_2 & \omega_i\in\Omega_1,\omega_j\in\Omega_2 \\ \pi \in\Pi_1 & otherwise \end{cases} \\ \Pi_1\cup\Pi_2= \Pi \tag{31}$$

where Π_1 and Π_2 are called important and trivial events, respectively, and are equivalent to the important and trivial regions in simple PoR. Generally, it is not mandatory to divide events by a partitioner defined in simple PoR. To extend the method, we define a new partitioner function called the advanced PoR partitioner function, as follows:

$$\ddot{I}: \Pi \to \{0,1\} \\ \forall\pi \in \Pi : \begin{cases} \pi \in \Pi_1 & \acute{I}\,(\pi)= 1 \\ \pi \in\Pi_2 & \acute{I}\,(\pi)= 0 \end{cases} \tag{32}$$

Contrary to simple and recursive PoR techniques, which try to partition the state space directly, the advanced PoR technique partitions the transition set of the model for this purpose. As before, it is required to have enough knowledge about the model to define the advanced PoR partitioner. In RES, trivial events are usually events that make the simulation path banish rare events. For example, in high-reliable systems, the repair events can be chosen as trivial events.

Let us define a new probability measure P^* for the model, in which the rate of each event is modified as follows:

$$\forall\pi \in \Pi :f^*\,(.,\pi)= \begin{cases} f(.,\pi) & \pi \in \Pi_1 \\ \mu : \mu \to 0 & \pi \in\Pi_2 \end{cases} \tag{33}$$

Here, $f(.,\pi)$and $f^*\,(.,\pi)$ denote the PDF of event π in the original and modified model, respectively. In the above equation, $f^*\,(.,\pi) = \mu : \mu \to 0$ means that $f^*\,(.,\pi)$ converges to 0, which causes the event π to occur only in infinity (e.g., in exponential distribution with the rate λ when $\lambda \to 0$). In practice, it can be replaced by a very small value, and so the events that have such a probability measure will never occur. By defining this probability measure, the model could be considered as shown in Figure 2. By regarding the region containing the initial state of the model, we can claim that Ω_2 is eliminated from the model, or at least that its probability is decreased to an extremely small value.

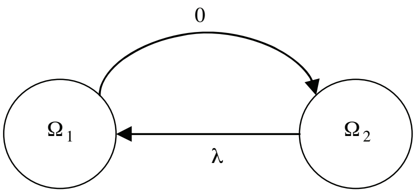

Figure 2. Applying advanced PoR to a model.

By considering the partitioner function, in addition to the mentioned probability measure, the rest of the theory of advanced PoR is same as the principle of the IS technique. Let us denote P^* to be the new probability distribution. The following equation shows the basic principle in this regard:

$$P\,(\varepsilon) =E_P\,(I\,(\varepsilon))= \int I\,(\varepsilon)\,dP= \int I\,(\varepsilon)\,\frac{dP}{dP^*}dP^*= \int I\,(\varepsilon)\,LdP^*=E_{P^*}\,[L.I\,(\varepsilon)] \tag{34}$$

$L = dP/dP^*$, the quotient of probability measures, is called the IS likelihood ratio. For more details about IS and likelihood ratio computation, please see [1,16,17].

To estimate $I(\varepsilon)$ under the probability measure P^*, we have to compensate the bias by actually simulating $I(\varepsilon)L$. An estimator of $E_{P*}(I(\varepsilon))$ is then equal to:

$$\gamma_n\left(P^*\right)=\frac{1}{n}\sum_{i=1}^{n} I_i\left(\varepsilon\right)L_i. \tag{35}$$

Unlike simple PoR, advanced PoR can be used for a wide range of models. However, because of this point, some interesting properties of simple PoR, such as BRE or other robustness properties, are no longer true with advanced PoR. This is because the likelihood ratio of this technique is not constant during the simulation, and so Eq. (27) cannot easily be proven as earlier. In addition, advanced PoR, unlike simple PoR, requires a tool with special capabilities (i.e. IS simulation). In fact, advanced PoR and IS have the same features, except for the way in which they increase the occurrence probability of rare events. The IS method accelerates the occurrence of rare events by changing the probability law. This could be done by increasing their distribution rates. In this approach, the whole model must be simulated and no part is removed from the model. On the other hand, the advanced PoR technique decreases the probability of other events to 0 instead of increasing the probability of rare events, and so they never occur in simulation. Therefore, the probability of the occurrence of rare events increases because of the absence of competitive events beside them. Advanced PoR may be better for the models with large state spaces.

Considering the above points, the first choice for a model should be simple PoR, followed by advanced PoR. Advanced PoR can be used if the model does not have the prerequisite of simple PoR, which can be considered as the only interesting property of advanced PoR.

6. RES using PoR techniques

As mentioned earlier, one of the motivations of this work has been to remove the gap between academic research and the industrial applications of RES. In this regard, we present the required simulation algorithms for implementing PoR techniques. In the rest of this section, the simulation algorithms of the simple, recursive, and advanced PoR techniques are presented. In all of the algorithms, it is assumed that the user has access to a tool for discrete-event simulation and has enough knowledge about it.

6.1. The simulation algorithm for simple PoR

In this subsection, we present the simple PoR simulation algorithm. The algorithm is shown in Figure 3.

Before the implementation of the algorithm, it is necessary to define the partitioner function by the modeler. Algorithm 1 can be divided into 4 main steps:

1. The first step (line 1) tries to compute the probability, $\bar{P}$, using a naïve simulator.
2. In the second step (line 2), the model will be partitioned into important and trivial regions.
3. In the third step (lines 3 to 5), the algorithm again uses a naïve simulator to evaluate the partitioned model.
4. In the fourth step (line 6), the reward values will be estimated using the likelihood ratio as defined in Eq. (17).

Algorithm 1. The simple PoR simulation algorithm.

```
1.    Simulate the model to estimate the probability of $l(w)$ [as defined in Eq. (17)].
2.    Remove $\Omega_2$ from the model.
3.    Initialize the model's variables.
4.    Set the likelihood ratio to $P[l(w)]$.
5.    While the specified confidence level is not achieved do:
5.1     Determine the set of enabled events in the current state.
5.2     Generate the execution time for the newly enabled events.
5.3     Apply the corresponding enabling rates to the remaining time of the enabled events.
5.4     By considering the remaining time of all of the enabled events, select the events with the least remaining time for completion.
5.5     If more than one event is enabled, compute the rank of each event and select the events with the highest rank.
5.6     If still more than one event is selected, compute the weight of each event and select one of them probabilistically. The selection probability of each event is as follows:
          $P\left(a\right)=\frac{w(a)}{\sum\limits_{a'\in SE} w(a')}$,
        where $SE$ is the set of selected events and $w(a)$ is the weight of the event $a$.
5.7     Observe the reward variables for the completion time of the selected event.
5.8     Update the confidence mean of the reward variables.
5.9     Run the action method for the selected event.
5.10    Advance the simulation clock to the completion time of the selected event.
6.    Weight the reward values by multiplying by the likelihood ratio.
```

Figure 3. The simple PoR simulation algorithm.

The main advantage of Algorithm 1 is that there is no need to have a special simulation tool. It only needs to use a naïve simulator in 2 steps (in the first and second steps). Therefore, the simple PoR technique can be run with a traditional simulator. In Section 7, we present an example model with some rare events, which is simulated using simple PoR and the Möbius modeling tool.

6.2. The simulation algorithm for recursive PoR

In this subsection, we present the recursive PoR simulation algorithm. The main difference between simple and recursive PoR is in the likelihood ratio computation. Thus, first, it is necessary to introduce a function for this purpose. The algorithm of this function is shown in Figure 4, which gets $i$as the input and returns the likelihood ratio of the ith level of recursive PoR as the output.

Using the above function, the simulation algorithm of recursive PoR is shown in Figure 5. As mentioned before, the principle of recursive PoR is the same as that of simple PoR. Therefore, recursive PoR has all of the advantages of simple PoR and there is no need to overwrite the existing simulation tools to adapt this technique.

6.3. The advanced PoR algorithm

In this subsection, we present the advanced PoR simulation algorithm. As mentioned earlier, the advanced PoR technique uses the principle and relations of the IS technique. Therefore, in Algorithm 4, shown in Figure 6, it is assumed that the modeler has access to a tool with the ability of the IS simulation (e.g., PDETool). The fourth step in the proposed algorithm is similar to the IS simulation algorithm in [17]. As before, the advanced PoR partitioner function must be defined before starting the simulation.

```
Algorithm 2. The likelihood ratio computation function, ComputeLI.
Input: $i$ (the level).
Output: The likelihood ratio of the $i$th level.
1.  If $i>k$, then
        Return 1.
2.  Remove all states of the model, except for $\Omega_1$ and $\Omega_{2i}$.
3.  Simulate the model to estimate the probability of $l_i(w)$ [as defined in Eq. (22)].
4.  Set $P_i = P[l_i(w)]$.
5.  Set $L_{i+1} = ComputeLI(i+1)$.
6.  Return $P_i * L_{i+1}$.
```

Figure 4. The algorithm of the likelihood ratio computation function.

7. Some illustrative examples

This section provides 3 models using SAN formalism in order to evaluate the efficiency of the PoR techniques. All of the models are simulated and run with the confidence interval 0.1 and the confidence level 95%, on a 2.0 GHz Intel Core2Duo laptop with 1.5 GB RAM and Microsoft Windows XP. Moreover, all of the times are expressed in seconds, except those having explicit units. Because some simulations take a very long time, the maximum time defined for naïve simulation is considered as 5000 s. For those that need more time than this, the result and time are filled with ”—” and ” $\approx x$”, where x is an approximate time of the simulation estimated using the central limit theorem.

7.1. An M/M/1 queue

The first example examined in this section is an M/M/1 queue. The SAN model corresponding to an M/M/1 queue is shown in Figure 7. In this model, the timed activity *Job_Income* models the arrival process and *Process* models the server. There is a place, named *Queue*, that models the queue's waiting line. The interarrival and service times are exponentially distributed with the rate 1.0 and 2.0, respectively. The aim of the simulation is to compute the probability of having more than n customers in the queue. The simulation has been executed for $n =$ 20, 30, and 45 (i.e. the probability of there being more than 20, 30, and 45 tokens in the queue). For applying the simple PoR technique, the following partitioner function is defined, which is changed according to the values of n:

$$\acute{I}=\begin{cases} 1 & Queue.Mark\geq n' \\ 0 & otherwise \end{cases}$$
$$n^{'}=\begin{cases} 12 & n=20 \\ 15 & n=30 \\ 23 & n=45 \end{cases} \quad . \tag{36}$$

```
Algorithm 3. The recursive PoR simulation algorithm.
1.     Set the likelihood ratio to ComputeLI (1).
2.     Remove $\Omega_2$ from the model.
3.     Initialize the model's variables.
4.     While the specified confidence level is not achieved do:
4.1       Determine the set of enabled events in the current state.
4.2       Generate the execution time for the newly enabled events.
4.3       Apply the corresponding enabling rates on the remaining time of the enabled events.
4.4       By considering the remaining time of all of the enabled events, select the events with
          the least remaining time for completion.
4.5       If more than one event is enabled, compute the rank of each event and select the
          events with the highest rank.
4.6       If still more than one event is selected, compute the weight of each event and select
          one of them probabilistically. The selection probability of each event is as follows:

              $P\left(a\right)=\frac{w(a)}{\sum_{a'\in SE} w(a')}$

          where $SE$ is the set of selected events and $w(a)$ is the weight of the event $a$.
4.7       Observe the reward variables for the completion time of the selected event.
4.8       Update the confidence mean of the reward variables.
4.9       Run the action method for the selected event.
4.10      Advance the simulation clock to the completion time of the selected event.
5.     Weight the reward values by multiplying by the likelihood ratio.
```

Figure 5. The recursive PoR simulation algorithm.

This partitioner forces the model to continue the simulation while at least n' tokens exist in the place *Queue*. This leads to an increase in the existence probability of more than n tokens in this place, which in turn accelerates the reaching of the result. To apply this partitioning, we have used an input gate *ig1* for the activity *Process*; its enabling predicate and input function are shown in Table 1.

Table 1. $ig1$ input gate properties.

Predicate	Function

If (Queue.Mark>n')
return true;
else return false;
	Queue.Mark–;

In Table 1, n' will be replaced by the corresponding value according to Eq. (36). The results of the simple PoR simulation, as well as the naïve simulation, were compared with the analytical results and are shown in Table 2. To evaluate the model, we have used the Möbius tool. This comparison is done for 3 different values of n. The results show an improvement of up to 3000 times in the simulation time; however, the efficiency of the simple PoR technique is reduced when the likelihood ratio is very close to 1.

Algorithm 4. The advanced PoR simulation algorithm.

```
1.    Define a new probability measure $P^*$ for the model according to Eq. (33).
2.    Initialize the model's variables.
3.    Initialize the likelihood ratio.
4.    While the specified confidence level is not achieved do:
4.1   Determine the set of enabled events in the current state.
4.2   Generate the execution time for the newly enabled events.
4.3   Apply the corresponding enabling rates on the remaining time of the enabled events.
4.4   By considering the remaining time of all of the enabled events, select the events with the least
      remaining time for completion.
4.5   If more than one event is enabled, compute the rank of each event and select the events with the
      highest rank.
4.6   If still more than one event is selected, compute the weight of each event and select one of them
      probabilistically. The selection probability of each event is as follows:

          $P(a) = \frac{w(a)}{\sum_{a' \in SE} w(a')}$

      where $SE$ is the set of the selected events and $w(a)$ is the weight of the event $a$.
4.7   Observe the reward variables for completion time of the selected event.
4.8   Weight the observation by multiplying by the likelihood ratio.
4.9   Update the confidence mean of the reward variables.
4.10  Run the action method for the selected event.
4.11  Advance the simulation clock to the completion time of the selected event.
4.12  Update the likelihood ratio and sample path.
```

Figure 6. The advanced PoR simulation algorithm.

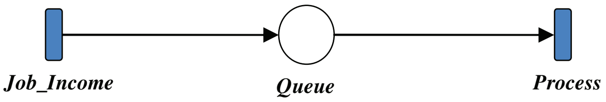

Figure 7. The SAN model of an M/M/1 queue.

Table 2. The results for naïve simulation vs. simple PoR for an M/M/1 queue.

		$n = 20$	$n = 30$	$n = 45$
Analytic solution	Result	4.76×10^{-7}	4.65×10^{-10}	1.42×10^{-14}
Naïve simulation	Result	4.59×10^{-7}	---	---
	Time	3291	≈337,000	$\approx 1.1 \times 10^{10}$ ≈350 years
Simple PoR	$\overline{\text{P}}$	0.99976	0.999973	---
	Result	4.80×10^{-7}	4.21×10^{-10}	---
	Time	1.42	131	≈100,000

7.2. An example for recursive PoR

In this subsection, we will again study the model of Figure 7. It can help us to compare simple and recursive PoR. The M/M/1 queue is simulated by recursive PoR using the same conditions. To have a better comparison, we reduce the probability of the desired rare event, and then the queue is studied while $n =$ 30, 45, and 90. Hence, the following function is used as the recursive partitioner function:

$$\acute{I}\colon \begin{cases} \frac{Queue.Mark}{3}+1 & Queue.Mark \leq n-3 \\ 0 & otherwise \end{cases} . \tag{37}$$

The result of the recursive PoR simulation of the model in comparison with simple PoR and the naïve simulation is shown in Table 3. The results show an improvement of up to 10^9 times compared with simple PoR. On the other hand, the results show a considerable improvement by the order of 10^{22} over the time of the naïve simulation.

Table 3. The simulation results for the recursive PoR example.

		$n = 30$	$n = 45$	$n = 90$
Analytic solution	Result	4.65×10^{-10}	1.42×10^{-14}	4.03×10^{-28}
Naïve simulation	Result	---	---	---
	Time	≈337,000	$\approx 1.1 \times 10^{10}$ ≈350 years	$\approx 3.8 \times 10^{23}$ $\approx 1.23 \times 10^{16}$ years
Simple PoR	Result	4.21×10^{-10}	---	---
	Time	131	≈100,000	$\approx 1.5 \times 10^{20}$ $\approx 4.73 \times 10^{12}$ years
Recursive PoR	Result	4.72×10^{-10}	1.45×10^{-14}	4.32×10^{-28}
	Time	23	36	75

7.3. An example of a high-reliable system

In this subsection, the simulation results of a sample SAN model using the advanced PoR technique will be presented. The model, shown in Figure 8, is a machine repairman system (used in several other studies, such as [17,31]) that uses the group repair policy. There are 2 types of components in the system with the same failure rates. The places labeled as *type_1* and *type_2* are used to model the 2 types of working components. In this case, there are 15 components of each type. The timed activities *fail_1* and *fail_2* model the failures for each component. The markings of the places *failed_1* and *failed_2* represent the number of components of each type that have failed. There are 2 repairmen in the system. The repairmen begin to repair when at least one component has failed, and when the repair is completed, all of the components of that type will be as good as a new one. Repairing the first type of component is given preemptive priority over repairing the second type. This repair policy is implemented in the input gate named *policy.* All of the activities are exponential with the rate of 1.0. If all of the components of both types fail, the system fails, and all of the repair activity halts; the failed state is an absorbing state. The aim of the simulation is to compute the unreliability during the interval [0, 100]. Due to the large number of redundant components and the group repair policy, this property is very small and can be considered as a rare event.

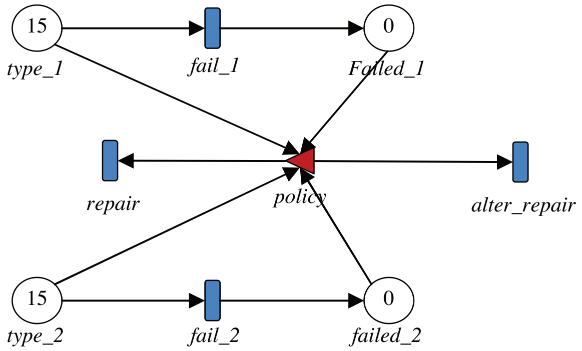

Figure 8. A sample machine repairman model.

For applying advanced PoR, the following PDF is chosen for the *repair* and *alter_repair* activities, which means that no component will be repaired if at least 7 components of each type have failed:

$$f\,(.,fail) = \begin{cases} \exp(\lambda)\,|\lambda\to 0 & \begin{array}{c} failed\_1.Mark > 7 \\ \text{and failed\_2.Mark } > 7 \end{array} \\ \exp(1) & otherwise \end{cases} \quad . \tag{38}$$

Therefore, the unreliability of the model will be increased and there will be more chance of running in the failed state.

From another point of view, the model shown in Figure 8 is also simulated using the IS technique. For this purpose, the model is simulated again using the likelihood ratio, while the rate of fail events is increased to 4.

Table 4 shows the simulation results using the advanced PoR technique and naïve simulation. The maximum time for naïve and IS simulation is considered as 50,000 and 10,000 s, respectively. The simulations have been run with relative half-widths of 95% confidence intervals using PDETool, which supports IS simulation. For comparison purposes, the model is also solved numerically using the transient solver of the Möbius tool. Because the IS method does not follow standard simulation method rules, it is not possible to get an approximation time for the simulation.

Table 4. The simulation results using advanced POR vs. naïve and IS simulation for the model of Figure 8.

	Analytical	Naïve	IS	Advanced PoR
Result	1.00×10^{-7}	–	–	1.09×10^{-7}
Time	–	$\approx 40,000$	>>10,000	1536
Error	–	–	–	9%

8. Conclusions

In this paper, with the aim of developing a practical RES technique, PoR techniques have been introduced. These techniques are based on a VRT with the same name. Three variants of PoR techniques were introduced. The first variant, which is called simple PoR, is the result of direct implementation of the PoR method. In this case, without any specific simulation tool, a modeler is able to use the technique. The simple PoR technique simulates models by breaking the simulation into 2 separate steps using the information of the first step in order to reduce the simulation time in the second step. The second variant, which is called recursive PoR, is an extension of simple PoR that breaks the simulation into a few steps, instead of only 2 steps. Simple and recursive PoR are in the same class and both are the result of the direct implementation of the PoR method.

The results of some illustrative examples showed a reduction in the simulation time of up to 10^{22}, which confirms the ability of these 2 techniques. The simple PoR technique is not applicable to all kinds of models. Therefore, the third variant, which is called the advanced PoR technique, is introduced to eliminate this limitation. In this technique, the model is not directly partitioned, but with the laws of probability and getting the benefit of the principles of IS, the trivial region of the model is provided with an implicit partitioning. For this purpose, the probability of the desired region will be reduced to 0. This means that although no part of the model is removed, the simulated path never leads to the trivial region. The results show a meaningful improvement in the simulation time.

As mentioned before, partitioning in PoR methods (i.e. choosing important and trivial regions) is generally a heuristic problem and needs an expert user. In the future, we intend to propose an automated method, or at least minimize the need for an expert for partitioning. In addition, we are going to implement PoR methods in SimGine [24,32], which is a simulation engine for SDESs that we have used in PDETool.

References

[1] S. Juneja, P. Shahabuddin, “Rare-event simulation techniques: an introduction and recent advances”, in: Handbook of Simulation, Handbooks in Operations Research and Management Science, Amsterdam, Elsevier, Vol. 13, pp. 291–350, 2006.

[2] P. Shahabuddin, “Importance sampling for the simulation of highly reliable Markovian systems”, Journal of Management Science, Vol. 40, pp. 333–352, 1994.

[3] P.W. Glynn, D.L. Iglehart, “Importance sampling for stochastic simulations”, Journal of Management Science, Vol. 35, pp. 1367–1392, 1989.

[4] P. L’Ecuyer, V. Demers, B. Tuffin, “Splitting for rare-event simulation”, Proceedings of the 2006 Winter Simulation Conference, pp. 137–148, 2006.

[5] H. Kahn, T.E. Harris, “Estimation of particle transmission by random sampling”, National Bureau of Standards Applied Mathematics Series, Vol. 12, pp. 27–30, 1951.

[6] P. Heidelberger, P. Glasserman, P. Shahabuddin, T. Zajic, “Splitting for rare event simulation: analysis of simple cases”, Proceedings of the 1996 Simulation Conference, pp. 302–308, 1996.

[7] P.T. de Boer, V.F. Nicola, R.Y. Rubinstein, “Adaptive importance sampling simulation of queueing networks”, Proceedings of 2000 the Winter Simulation Conference Vol. 1, pp. 646–655, 2000.

[8] M. Villèn-Altamirano, J. Villèn-Altamirano, “RESTART: a method for accelerating rare event simulations”, Queueing Performance and Control in ATM, pp. 71–76, 1991.

[9] M. Villèn-Altamirano, J. Villèn-Altamirano, “RESTART: a straightforward method for fast simulation of rare events”, Proceedings of the 1994 Winter Simulation Conference, pp. 282–289, 1994.

[10] A.J. Bidgoly, M.A. Azgomi, “Rare event simulation of stochastic activity networks using partition of the region technique”, Proceedings of the 24th UK Performance Engineering Workshop, pp. 107–122, 2008.

[11] A.J. Bidgoly, M.A. Azgomi, A. Khalili, “An advanced technique for rare-event simulation based on partition of the region”, Proceedings of the 10th International Middle Eastern Simulation Multiconference, pp. 5–10, 2009.

[12] R.Y. Rubinstein, Simulation and Monte Carlo Method, New York, Wiley, 2009.

[13] W.H. Sanders, J.F. Meyer, “Stochastic activity networks: formal definitions and concepts”, Lectures on Formal Methods and Performance Analysis, pp. 315–343, 2001.

[14] G. Rubino, B. Tuffin, Rare Event Simulation Using Monte Carlo Methods, New York, Wiley, 2009.

[15] D.R. Cox, D.V. Hinkley, Theoretical Statistics, London, Chapman & Hall, 1974.

[16] V.F. Nicola, M.K. Nakayama, P. Heidelberger, A. Goyal, "Fast simulation of highly dependable systems with general failure and repair processes", IEEE Transactions on Computers, Vol. 40, pp. 1440–1452, 1993.

[17] W.D. Obal 2nd, "Importance sampling simulation of SAN-based reward models", MSc, University of Arizona, USA, 1993.

[18] C. Kollman, K. Baggerly, D. Cox, R. Picard, "Adaptive importance sampling on discrete Markov chains", Annals of Applied Probability, pp. 391–412, 1999.

[19] P. Heidelberger, "Fast simulation of rare events in queueing and reliability models", ACM Transactions on Modeling and Computer Simulation, Vol. 5, p. 43–85, 1995.

[20] P. L'Ecuyer, J.H. Blanchet, B. Tuffin, P.W. Glynn, "Asymptotic robustness of estimators in rare-event simulation", ACM Transactions on Modeling and Computer Simulation, Vol. 20, pp. 6:1–6:41, 2010.

[21] A. Movaghar, J.F. Meyer, "Performability modeling with stochastic activity networks", Proceedings of the 1984 Real-Time Systems Symposium, pp. 215–224, 1984.

[22] W.H. Sanders, W.D. Obal 2nd, M.A. Qureshia, F.K. Widjanarko, "The UltraSAN modeling environment", Performance Evaluation, Vol. 24, pp. 89–115, 1995.

[23] D.D. Deavours, G. Clark, T. Courtney, D. Daly, S. Derisavi, J.M. Doyle, W.H. Sanders, P.G. Webster, "The Mobius framework and its implementation", IEEE Transactions on Software Engineering, Vol. 28, pp. 956–969, 2002.

[24] A. Khalili, A.J. Bidgoly, M.A. Azgomi, "PDETool: A multi-formalism modeling tool for discrete-event systems based on SDES description", Proceedings of the 30th International Conference on Application and Theory of Petri Nets and Other Models of Concurrency, Vol. 5606, pp. 343–352, 2009.

[25] S. Lazarova-Molnar, "The proxel-based method: formalisation, analysis and applications", PhD, der Otto-von-Guericke-University, Germany, 2005.

[26] C. Isensee, G. Horton, "Fast simulation without randomness: a simulation tool combining proxels and discrete phases", Proceedings of the 18th Symposium Simulationstechnik, 2005.

[27] P. Glasserman, Monte Carlo Methods in Financial Engineering, Berlin, Heidelberg, Springer-Verlag, 2004.

[28] B. Tuffin, "Bounded normal approximation in simulations of highly reliable Markovian systems", Journal of Applied Probability, Vol. 36, pp. 974–986, 1999.

[29] B. Tuffin, "On numerical problems in simulations of highly reliable Markovian systems", Proceedings of the 1st International Conference on the Quantitative Evaluation of Systems, pp. 156–164, 2004.

[30] A. Zimmermann, "RESTART simulation of colored stochastic Petri nets", Proceedings of the 7th International Workshop on Rare Event Simulation, pp. 143–152, 2008.

[31] S.R. Agnihothri, "Interrelationships between performance measures for the machine-repairman problem", Naval Research Logistics, Vol. 36, pp. 265–271, 1989.

[32] SimGine homepage, URL: http://pdel.iust.ac.ir/projects/SimGine.html.