Reliability Estimation for Stochastic Flow Networks with Dependent Arcs

Hector Cancela, Leslie Murray, Gerardo Rubino

Reliability Estimation for Stochastic Flow Networks with Dependent Arcs

Héctor Cancela
Universidad de la República
Montevideo, Uruguay
cancela@fing.edu.uy

Leslie Murray
FCEIA–Universidad Nacional de Rosario
Rosario. Argentina
lmurray@fceia.unr.edu.ar

Gerardo Rubino
INRIA Rennes – Bretagne Atlantique, Campus de Beaulieu
Rennes, France
Gerardo.Rubino@inria.fr

Abstract

The Creation and the Destruction Processes (CP and DP) are the basis of many efficient Monte Carlo methods for estimating the unreliability of highly reliable networks on both, the static and the stochastic flow network models, for the case of independent components. Some of these methods are based on the Splitting variance reduction techniques. Due to the Splitting basic mechanism, they operate over CP. Here we propose a Splitting–based Monte Carlo method, using the Marshall–Olkin copula for the case of non independent components. This proposal operates on the DP, because the Marshall–Olkin copula model over networks is quite related —and somehow similar— to DP, which is unusual but here necessary. Before addressing the proposal, the article presents a review of the methods based on CP and DP. At the end, a comparative experimental analysis shows the efficiency of the proposed approach.

Index Terms

reliability, Creation Process, Destruction Process, Splitting, dependent arcs.

ACRONYMS[1]

SNM	Static Network Model
SFNM	Stochastic Flow Network Model
SMC	Standard Monte Carlo
CP	Creation Process
DP	Destruction Process
ML–CP	Multi Level Creation Process
ML–DP	Multi Level Destruction Process
PMC	Permutation Monte Carlo
w.p.	with probability

[1]The singular and plural of an acronym are always spelled the same.

NOTATION

G	graph representing a network topology
V	set of n nodes for any type of network model
E	set of m links for SNM
A	set of m arcs for SNFM
C	set of capacities of the m arcs for SNFM
$\mathbf{X}$	random vector $(X_1, X_2, \ldots, X_m)$ indicating the state of links (for SNM) or arcs (for SFNM)
$\mathbf{x}$	a realization of the random vector $\mathbf{X}$
$\phi(\mathbf{X})$	structure function for SNM and SFNM
$F(\mathbf{X})$	total amount of flow, from the set of nodes that produce flow to the set of nodes that consume flow
D	the flow demand in SFNM
ζ	network *unreliability*
$\widehat{\zeta}$	unbiased estimator of ζ
$\widehat{\zeta}_{\mathbf{B}}^{\mathbf{A}}$	unbiased estimator of ζ where: A=*empty* means SMC, A=P means PMC and A=S means Splitting. B is either CP, DP, ML–CP or ML–DP.
N	sample size of a Monte Carlo experiment
N_T	number of thresholds in Splitting
ℓ_i	i–th threshold in both, ML–CP and ML-DP
α_i	probability that a trajectory started at threshold ℓ_i reaches threshold ℓ_{i+1} underneath D, in ML–CP
$\widehat{\alpha}_i$	estimator of α_i
β_i	probability that a trajectory started at threshold ℓ_i reaches threshold ℓ_{i+1} earlier than $t = 1$, in ML–DP
$\widehat{\beta}_i$	estimator of β_i

I. BACKGROUND AND PRELIMINARIES

In reliability settings, a network is usually modeled as an undirected graph $G = (V, E)$, in which V is a set of n nodes and E, a set of m links connecting these nodes. The links and/or the nodes can fail, and, according to this, the network can be either working or failed. The reliability (unreliability) is the probability that the network is working (failed).

A. Static Network Model

In the Static Network Model (SNM), the nodes in V are perfect (they never fail), while the links in E can be in one of two possible states, *operational* (also called *up*) or *failed* (also called *down*). In the *operational* state the links provide a perfect connection, whereas in the *failed* state they perform as if they were removed from the graph.

There is a subset of nodes $K \subset V$, called *terminal nodes*, such that the network is working if, despite links' failures there is a connected subgraph of G containing all the *terminal nodes*.

In SNM the links are modeled by the random vector $\mathbf{X} = (X_1, \ldots, X_m) \in \{0, 1\}^m$, where $X_i = 0$ means that link i is *failed*, and $X_i = 1$ means that link i is *operational*. We say that vector $\mathbf{X}$ is the state of the network and we call *configuration*, denoted $\mathbf{x} = (x_1, \ldots, x_m)$, any possible value of $\mathbf{X}$.

According to the *configuration*, the network assumes one of two possible states, *operational* or *failed*. This information is formalized in the so–called structure function, ϕ, which maps the 2^m possible *configurations* into $\{0, 1\}$ as follows: $\phi(\mathbf{x}) = 1$ if, when removing from G the links which are *failed* in $\mathbf{x}$, all the nodes in K belong to the same connected component of the resulting subgraph, and $\phi(\mathbf{x}) = 0$ otherwise.

Given the probabilities $p_i = \mathbb{P}\{X_i = 1\}$ and $q_i = \mathbb{P}\{X_i = 0\} = 1 - p_i$, $i = 1, \ldots, m$, the unreliability of SNM, ζ, is defined as:

$$\zeta = \mathbb{P}\{\phi(\mathbf{X}) = 0\} = 1 - \mathbb{E}\{\phi(\mathbf{X})\}. \tag{1}$$

Many algorithms have been designed to compute the exact value of these metrics. However, as their computation is NP–hard [1], they only apply to small or, at least, medium sized networks, while the case of large networks can only be tackled by simulation.

Standard or crude Monte Carlo (SMC) simulation is the simplest and most straightforward method for estimating network reliability. However, if networks are highly reliable, the efficiency of SMC estimation falls and simulations must be enhanced, for instance, by means of variance reduction methods.

The SMC estimator, $\widehat{\zeta}$, is based on the following indicator variable:

$$I = \begin{cases} 1 & \text{if } \phi(\mathbf{X}) = 0, \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

Then, given N independent samples of I, $I^{(i)},\ i = 1, \ldots, N$, the SMC unreliability estimator is:

$$\widehat{\zeta} = \frac{1}{N} \sum_{i=1}^{N} I^{(i)}. \tag{3}$$

We immediately have $\mathbb{E}\{\widehat{\zeta}\} = \zeta$ and $\mathbb{V}\{\widehat{\zeta}\} = \zeta(1 - \zeta)/N$. The ratio $\mathbb{V}\{\widehat{\zeta}\}^{1/2}/\mathbb{E}\{\widehat{\zeta}\}$ is interpreted as a statistical bound of the relative error in the estimation. It follows that, as ζ goes to zero, this relative error goes to infinity. In other words, as said before, the SMC unreliability estimator looses precision as the network becomes more reliable ($\mathbb{E}\{\widehat{\zeta}\}$ tends to zero faster than $\mathbb{V}\{\widehat{\zeta}\}^{1/2}$). It is necessary, therefore, to improve the estimation by means of variance reduction methods [2], [3], [4], [5], [6], [7], [8].

B. Stochastic Flow Network

The Stochastic Flow Network Model (SFNM) is based on a directed graph $G = (V, A, C)$, where V is a set of n nodes, A is a set of m arcs connecting these nodes and C the set of the arcs' capacities. The capacity is the maximum amount of flow that can pass through an arc. Nodes are perfect (they never fail) whereas capacities are modeled as a random vector $\mathbf{X} = (X_1, \ldots, X_m)$ in which every X_i is a multi–valued positive random variable.

Recall that a flow on G is a positive function on the arcs, satisfying Kirchoff's law: for each node u having strictly positives indegree and outdegree, the sum of the flow on the arcs entering u is equal to the sum of the flows on the arcs leaving u. In SFNM, we can more generally define two disjoint subsets of nodes, $R \subset V$ and $S \subset V$, such that nodes in R "produce flow", whereas nodes in S "consume flow". If we denote a flow from node x to node y, by $f(x, y)$, then it is easy to prove that:

$$\sum_{u \in R} \sum_{v \notin R} f(u, v) = \sum_{v \in S} \sum_{u \notin S} f(u, v) =: f_{\text{tot}}. \tag{4}$$

Here, f_{tot} is the *total flow* circulating in the network, generated in R and consumed in S. Now a *configuration* of the capacities is $\mathbf{x} = (x_1, \ldots, x_m)$, where x_i is a capacity value for arc i. A flow f is said to be compatible with the network capacities, written here $f \in \mathcal{C}(\mathbf{x})$, iff for all arc ℓ, we have $f(\ell) \leq x_\ell$. We call $F(\mathbf{x})$ the maximum amount of flow compatible with $\mathcal{C}(\mathbf{x})$ that can be sent from R to S, that is,

$$F(\mathbf{x}) = \text{argmax}_{f \in \mathcal{C}(x)} f_{\text{tot}}.$$

Thus, $F(\mathbf{X})$ is a random variable. In this context, we are typically given a threshold $D > 0$ and we want to calculate $\mathbb{P}\{F(\mathbf{X}) < D\}$. The structure function here is $\phi(\mathbf{X}) = 1$ when $F(\mathbf{X}) \geq D$, and $\phi(\mathbf{X}) = 0$ otherwise. Therefore, like in SNM, the SFNM unreliability, ζ, is defined as:

$$\zeta = \mathbb{P}\{\phi(\mathbf{X}) = 0\} = 1 - \mathbb{E}\{\phi(\mathbf{X})\}. \tag{5}$$

The SFNM applies to a wide variety of domains, from data networks, where flows are made up of information packets, to water pipe or electricity distribution networks, to name a few examples. In all these cases reliability is a fundamental parameter, both in performance analysis of existing systems as well as an a target parameter in the design of new ones.

Now, the same drawbacks affecting SNM also affect SFNM. This means that, if the network is large, the exact reliability computation is impossible. Estimation by simulation is an appropriate solution to find an approximate value, being SMC the easiest and most straightforward simulation method at hand. However, if the network is highly reliable, SMC loses efficiency and simulations must be improved by means of variance reduction methods.

II. Introduction

In this section we introduce the two central aspects of the paper, the variance reduction technique that we propose to deal with rarity, and the way we take into account the dependencies between components.

A. Splitting Monte Carlo estimation

One of the most powerful families of techniques to deal with the highly reliable situations is called Splitting, and our paper develops a variant of this class of methods. Splitting is defined on dynamical systems, and the most used models in network reliability, such as our considered SFNM, are static. Our initial step will be transforming our model into a dynamic one, by means of the so-called Creation Process (CP), or the Destruction Process (DP). Both of these stochastic processes have been introduced in [9]. They transform the static SNM into a dynamic process with failures and repairs that occur along an artificial time. In the case of the DP, the guiding idea is as follows: at time $t = 0$ the network components are operational, and they begin to fail randomly in times proportional to the failure probability of each one. Along with one of the individual failures, the network fails at a time which is, ultimately, proportional to its own failure probability. The CP is similar, but the components and the network start failed, and the events of interest are their "repairs" rather than their "failures".

The Multi–Level CP (ML–CP), proposed in [10], is an adaptation of the CP to SFNM, for the case of independent components (arcs). However, it does not directly apply to models with dependent components.

Neither CP nor ML–CP, are variance reduction methods. However, they are the basis of many of them, like [10], [11] and [9], but always on models in which links are independent. In the cases of [10] and [11], the methods are basically applications of Splitting [12], [13], [14], [15], [16], [17], [18] over the corresponding dynamic model (either CP or ML–CP).

B. Dependencies between components

As stated before, SFNM are very useful and used models in different areas. They always work from the knowledge of the structure of a multicomponent system and the behavior of each individual component characterized by their probability of being operational. This last information is enough if we assume independence between components, and this was done until very recently, in almost every dependability analysis. It also means that we try to define as components, subsystems of the given system that can be reasonably assumed to work or fail independently of each other. Now, in real life, components' independence is rare (as an example, see [19]). Moreover, in many cases, it is difficult to define components this way, and often, the statistical available data comes for more "naturally" defined components that exhibit clear dependencies. Take as an example the case of natural catastrophes (floods, earthquakes, tsunamis, storms), where when components are geographically close they show a high correlation regarding their states (operational or failed). See, for instance, the analysis in our previous works [20] or [21]. The cost of ignoring dependencies can be high, as shown in previous citations; see also [22].

The most widespread approach to address dependencies in network models is the Marshall–Olkin copula [23], which takes the form of a dynamic process in which failures occur sequentially at random times, just like in DP.

The Splitting–like methods published so far only apply to CP. In fact, Splitting perfectly fits CP because, if we formalize the evolution of the system as trajectories in the state space, the rare event is a region of that space that is "hard to reach" and, therefore, the trajectories that get close to it need to be stimulated or "pushed" towards the rare event. If we attempt to apply Splitting over DP, exactly as it is applied over the CP, the event of interest occurs if the trajectories "do not reach" some particular region of the state space. In other words, it will be necessary to prevent DP trajectories from progress beyond some particular bound, which is exactly the opposite of what Splitting does. A Splitting–like method over DP requires, therefore, a change in the framework, which is solved in this article applying Splitting over DP in a way that is significantly different as it it is applied over CP.

C. Contributions and structure of the paper

The main contribution of this paper is a method based on an application of Splitting over the Multi–Level DP (ML–DP) for SFNM, which operates on models with independent and dependent arcs as well.

Sections III and IV are a review of CP and DP, while Sections V and VI look over the ML–CP and the ML–DP. The case of dependent arcs for the ML–DP is approached in Section VII. Some experimental tests are introduced in Section VIII while conclusions and future work are presented in Section IX. The proposed method is introduced in Section VI and adapted to models with dependent arcs in Section VII.

III. The Creation Process

The *Creation Process* (CP) [9] is the underlying mechanism of many variance reduction methods. The CP transforms a SNM into a dynamic model, making the random vector $\mathbf{X}$ become a stochastic process $\mathbf{X}(t) = (X_1(t), \ldots, X_m(t))$, in which:

$$X_i(t) = \begin{cases} 0 & \text{if } t < \tau_i, \\ 1 & \text{if } t \geq \tau_i, \end{cases} \qquad i = 1, \ldots, m. \tag{6}$$

In short, link i is repaired (switched from *failed* to *operational*) at random time τ_i. The variables $\tau_1, \ldots, \tau_m$ are independent of each other. The key of CP is that, if τ_i is exponentially distributed with rate $\lambda_i = -\ln(q_i)$, then $\mathbb{P}\{X_i(1) = 1\} = p_i$ and $\mathbb{P}\{X_i(1) = 0\} = q_i$. Thus, observing the CP at $t = 1$ is equivalent to observing the original SNM. Therefore, $\zeta = \mathbb{P}\{\phi(\mathbf{X}(1)) = 0\}$.

Recall that the order statistics of the set $\{\tau_1, \ldots, \tau_m\}$ is the vector $S_{\text{CP}} = (\tau_{(1)}, \ldots, \tau_{(m)})$, where $\tau_{(k)}$ is the kth-smallest value in the set. In other words, S_{CP} is the set of all the repairing times, arranged in ascending order.

In S_{CP}, there is one particular time, $\tau_{(c)}$, such that:

$$\phi(\mathbf{X}(t)) = \begin{cases} 0 & \text{if } t < \tau_{(c)}, \\ 1 & \text{if } t \geq \tau_{(c)}. \end{cases} \tag{7}$$

Time $\tau_{(c)}$ is the repair time of the so called *critical link*. Then, $\forall t, t \geq \tau_{(c)}$, there is a connected sub–graph of G that includes the subset K of nodes and all the links that at time $\tau_{(c)}$ are already repaired. This time evinces the way CP and SNM are related, since the event $\{\tau_{(c)} > 1\}$ in the CP, is the same as $\{\phi(\mathbf{X}) = 0\}$ in the corresponding SNM.

Although it is not what it was intended for, the CP can readily be the basis of a SMC estimation. In order to implement this estimation, it is necessary to sample N independent copies of the order statistics $S_{\text{CP}}^{(i)} = (\tau_{(1)}^{(i)}, \ldots, \tau_{(m)}^{(i)}),\ i = 1, \ldots, N$, and to compute the following indicator variable, for each one of them:

$$I_{\text{CP}}^{(i)} = \begin{cases} 1 & \text{if } \tau_{(c)}^{(i)} > 1 \\ 0 & \text{otherwise.} \end{cases} \tag{8}$$

Based on this indicator, the SMC unreliability estimator of ζ, subject to CP, is:

$$\widehat{\zeta}_{\text{CP}} = \frac{1}{N} \sum_{i=1}^{N} I_{\text{CP}}^{(i)}. \tag{9}$$

Again, see that the unreliability estimations introduced so far are based on the following equivalent events: $\{\tau_{(c)} > 1\}$ in this CP–based SMC estimation, and $\{\phi(\mathbf{X}) = 0\}$ in SMC over SNM.

Permutation Monte Carlo (PMC) [9] is the first variance reduction method thought of as to fit CP. It is a variant of Conditional Monte Carlo in which the probability of the event $\{\tau_{(c)} > 1\}$ is conditioned on the ordering in which links are repaired (permutation). Calling Ω the set of all possible permutations, then:

$$\zeta = \mathbb{P}\{\tau_{(c)} > 1\} = \sum_{\omega \in \Omega} \underbrace{\mathbb{P}\{\tau_{(c)} > 1 \mid \Omega = \omega\}}_{\gamma_{\text{CP}}(\omega)} \mathbb{P}\{\Omega = \omega\}. \tag{10}$$

Sampling N independent permutations $\omega^{(i)},\ i = 1, \ldots, N$, and computing (exactly) N times the function $\gamma_{\text{CP}}(\omega^{(i)}) = \mathbb{P}\{\tau_{(c)} > 1 \mid \Omega = \omega^{(i)}\}$, the PMC unreliability estimator, subject to CP, is:

$$\widehat{\zeta}_{\text{CP}}^{\text{P}} = \frac{1}{N} \sum_{i=1}^{N} \gamma_{\text{CP}}(\omega^{(i)}). \tag{11}$$

A well known result, proved in [9], shows that $\mathbb{V}\{\widehat{\zeta}_{\text{CP}}^{\text{P}}\} \leq \mathbb{V}\{\widehat{\zeta}_{\text{CP}}\}$, what makes this PMC estimator more efficient than the corresponding SMC one in (3).

Whether CP is the basis of an SMC or a PMC estimator, it is important to see the ordering sequence of repairing times as a trajectory like the one shown in Figure 1. The simplest and most direct way to build such sequence is: (i) sampling all the repairing times and then (ii) sorting them in ascending order.

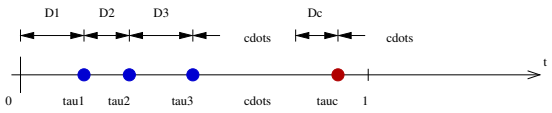

Fig. 1. Ordering sequence of repairing times.

However, based on the properties of the exponential distribution, it is possible to build these sequences much more efficiently, as it will be explained next.

As the target event is $\{\tau_{(c)} > 1\}$, only repairing times up to $\tau_{(c)}$ are of interest. Times between repairs, $\Delta_i,\ i = 1, \ldots, c$, defined as follows, are also exponentially distributed with rates $\Lambda_i,\ i = 1, \ldots, c$. Calling $T = \sum_{i=1}^{m} \lambda_i$, then $\Lambda_1 = T$, $\Lambda_2 = T - \lambda_x$, where link x is the one repaired at time $\tau_{(1)}$, $\Lambda_3 = T - \lambda_x - \lambda_y$, where link y is the one repaired at time $\tau_{(2)}$, and so on. On the other hand, the probability that the link repaired at time $\tau_{(1)}$ is link x is given by λ_x/T, the probability that the link repaired at time $\tau_{(2)}$ is link y is given by $\lambda_y/(T - \lambda_x)$, etc. As a consequence, time $\tau_{(c)} = \sum_{i=1}^{c} \Delta_i$ is the sum of$c$ exponentially distributed random times with rates $\Lambda_i,\ i = 1, \ldots, c$, and its probability density function is, therefore:

$$f_{\Delta_1 + \Delta_2 + \cdots + \Delta_c}(t) = \left(\prod_{i=1}^{c} \Lambda_i \right) \sum_{j=1}^{c} \frac{e^{-\Lambda_j t}}{\prod\limits_{\substack{k=1 \\ k \neq j}}^{c} (\Lambda_k - \Lambda_j)}. \tag{12}$$

In both CP–based estimations considered so far, i.e. the one in (9) and the one in (11), the target value is the probability of the event $\{\tau_{(c)} > 1\}$. The computation in (9) saves and accumulates '1's for every sampled sequence in which the event $\{\tau_{(c)} > 1\}$ occurs, and '0's otherwise. At the end, all the accumulated values are divided by the number of samples. On the other hand, the mechanism in (11) saves and accumulates the probability of the event $\{\tau_{(c)} > 1\}$ for every sampled sequence —computed by means of (12)— and finally all the accumulated values are divided by the number of samples.

Another CP–based variance reduction approach is the one introduced in [10]. It is a novel application of Splitting [12], [13], [14], [15], [16], [17], [18] in which the trajectories of repairing times start at $t = 0$ and progress towards $t = 1$. The method uses a partition of the interval $[0, 1]$ formed by a set of N_T times, namely $0 < \ell_1 < \ldots < \ell_{N_T} < 1$. These time bounds perform as the thresholds of the splitting mechanism. Thus, whenever a repair time exceeds one of the thresholds, the corresponding trajectory actually crosses this threshold. Then, if the *critical link*, $\tau_{(c)}$, is not yet part of such trajectory, many (more than one) new trajectories are started at the crossing point in the expectation that at least one of them reaches the next threshold before $\tau_{(c)}$ is sampled (see [10]). Computing the number of trajectories that, at the end of the process have crossed all the thresholds, including the final bound, $t = 1$, it is possible to obtain an unbiased unreliability estimator, $\widehat{\zeta}^{\text{S}}_{\text{CP}}$, based on Splitting subject to CP [24], as follows:

$$\widehat{\zeta}^{\text{S}}_{\text{CP}} = \prod_{i=0}^{N_T} \widehat{p}_i, \tag{13}$$

where $\widehat{p}_i = n_{i+1}/n_i,\ i = 1, \ldots, N_T$, being n_i the number of crosses through threshold ℓ_i (n_0 is the number of trajectories started at $t = 0$ and n_{N_T+1} is the number of trajectories that reach $t = 1$). The variance of $\widehat{\zeta}^{\text{S}}_{\text{CP}}$ is much smaller than the variance of the crude estimator in (9).

IV. The Destruction Process

The Destruction Process (DP) [9] is another random process that models networks temporal evolution and —as the CP— it is the basis of variance reduction methods. In DP, links fail (switch from *operational* to *failed*) at times τ_1, ..., τ_m, that is:

$$X_i(t) = \begin{cases} 1 & \text{if } t < \tau_i, \\ 0 & \text{if } t \geq \tau_i, \end{cases} \qquad i = 1, \ldots, m. \tag{14}$$

If τ_i is exponentially distributed with rate $\lambda_i = -\ln(p_i)$, then $\mathbb{P}\{X_i(1) = 1\} = p_i$ and $\mathbb{P}\{X_i(1) = 0\} = q_i$, meaning that —just like in CP— observing the DP at $t = 1$ is equivalent than observing the SNM. The unreliability, ζ, is also $\mathbb{P}\{\phi(\mathbf{X}(1)) = 0\}$.

Now, the order statistics of $\{\tau_1, \ldots, \tau_m\}$ is the ordered set of failing times $S_{\text{DP}} = \{\tau_{(1)}, \ldots, \tau_{(m)}\}$. Time $\tau_{(c)}$ in S_{DP} is the failing time of the so called *critical link*. It is the instant since which the remaining *operational* links —if any— are not enough to guarantee the connection among the nodes in the subset K. See that event $\{\tau_{(c)} < 1\}$ in DP, is the same as event $\{\phi(\mathbf{X}) = 0\}$ in SNM.

DP can also be the basis of an SMC estimation. In order to implement this estimation, it is necessary to sample N independent copies of the order statistics, $S^{(i)}_{\text{DP}} = \{\tau^{(i)}_{(1)}, \ldots, \tau^{(i)}_{(m)}\},\ i = 1, \ldots, N$, and to compute the following indicator variable, for each one of them:

$$I^{(i)}_{\text{DP}} = \begin{cases} 1 & \text{if } \tau^{(i)}_{(c)} < 1, \\ 0 & \text{otherwise.} \end{cases} \tag{15}$$

Finally, the SMC unreliability estimator, subject to DP, is:

$$\widehat{\zeta}_{\text{DP}} = \frac{1}{N} \sum_{i=1}^{N} I^{(i)}_{\text{DP}}. \tag{16}$$

PMC also fits DP, in a similar way as it does to CP. Being Ω the set of all possible permutations (accepting now that permutations come from the ordering sequence of failing times), then:

$$\zeta = \mathbb{P}\{\tau_{(c)} < 1\} = \sum_{\omega} \underbrace{\mathbb{P}\{\tau_{(c)} < 1 \mid \Omega = \omega\}}_{\gamma_{\text{DP}}(\omega)} \mathbb{P}\{\Omega = \omega\}. \tag{17}$$

Sampling N independent permutations, $\omega^{(i)},\ i = 1, \ldots, N$, and computing (exactly) N times the function $\gamma_{\text{DP}}(\omega^{(i)}) = \mathbb{P}\{\tau_{(c)} < 1 \mid \Omega = \omega^{(i)}\}$, the PMC unreliability estimation subject to DP, is:

$$\widehat{\zeta}_{\text{DP}}^{\text{P}} = \frac{1}{N} \sum_{i=1}^{N} \gamma_{\text{DP}}(\omega^{(i)}) \tag{18}$$

As shown in [9], $\mathbb{V}\{\widehat{\zeta}_{\text{DP}}^{\text{P}}\} \leq \mathbb{V}\{\widehat{\zeta}_{\text{DP}}\}$, what makes this PMC estimation more efficient than the corresponding SMC estimation in (3).

V. The Multi–Level CP

CP–based methods are focused on the determination of the probability (extremely low for a highly reliable network) that $\tau_{(c)} > 1$, whereas DP–based methods are focused on the determination of the probability (extremely low for a highly reliable network) that $\tau_{(c)} < 1$. These probabilities are, in both cases, the network unreliability.

Both CP and DP have been designed over networks in which links are modeled as two–valued variables. In [10], the CP has been adapted —and named Multi–Level CP— to Stochastic Flow Network Models (SFNM) in which the variables that model the arcs can take more than two values. The rest of this section contains a complete review of the ML–CP.

As said in Section II, in SFNM the capacities of the m arcs are represented by a random vector $\mathbf{X} = (X_1, \ldots, X_m)$, in which every X_i is a multi–valued random variable. Without loss of generality, let us consider that $X_i, i = 1, \ldots, m$, are identical discrete random variables distributed as follows:

$$X_i = \begin{cases} M_1 & \text{w.p. } p_1, \\ M_2 & \text{w.p. } p_2, \\ \quad\vdots & \qquad\qquad M_1 > M_2 > \ldots > M_n > 0, \\ M_n & \text{w.p. } p_n, \\ 0 & \text{w.p. } p_0, \end{cases}$$

$$i = 1, \ldots, m. \tag{19}$$

Here, $X_i = M_1$ means that arc i is fully *operational*, and $X_i = 0$ means that arc i is completely *failed*. Intermediate values indicate that arc i is *partially operational* (or *partially failed*, according to convenience).

When we have $X_i = M_1\ \forall i$, $F(\mathbf{X})$ assumes its maximum possible value which, if the network has been properly designed, should be greater than the demand, D. If, due to failures, the capacity of one or more arcs —possibly all of them— is different (lower) than M_1, then $F(\mathbf{X})$ may be less than D.

Given the distribution of $\mathbf{X}$, the SFNM reliability is $\mathbb{P}\{\phi(\mathbf{X}) = 1\} = \mathbb{E}\{\phi(\mathbf{X})\}$, and the unreliability, $\mathbb{P}\{\phi(\mathbf{X}) = 0\} = 1 - \mathbb{E}\{\phi(\mathbf{X})\}$. Many algorithms have been designed to compute the exact value of these terms (see, for example, [25] and [26]). However, as their computation is NP–hard [1], they only apply to small or, at least, medium sized networks, whereas the case of large networks can only be estimated by simulation. SMC is the simplest and most straightforward method for obtaining an estimate. However, if networks are highly reliable, the efficiency of SMC estimations fall and simulations must be enhanced by variance reduction methods, like the ones proposed in [10], [27], [28], [29], [30], [31].

In order to build a variance reduction method, the ML–CP [10] transforms SFNM into a dynamic model, making the random vector $\mathbf{X}$ become the stochastic process $\mathbf{X}(t) = (X_1(t), \ldots, X_m(t))$, in which:

$$X_i(t) = \begin{cases} 0 & \text{if } t < \tau_i \\ M_j & \text{if } t \geq \tau_i \end{cases} \qquad i = 1, \ldots, m \quad \text{and}$$

$$j \in \{1, \ldots, n\}. \quad (20)$$

A particular type of "repair" occurs at time τ_i, as the value of $X_i(t)$ switches from 0, which means completely *failed*, to one of the *partially operational* states, namely, to one of the possible $M_j > 0$. The mechanism proposed for deciding which M_j is assigned to $X_i(t)$ at time τ_i, is explained next.

Let us accept, by now, that there exists a set $\{\lambda, t_0 = 0, t_1, t_2, \ldots, t_{n-1}, t_n = 1\}$ such that λ is the rate of the exponentially distributed random time τ_i, and $t_1, t_2, \ldots, t_{n-1}$, a set of fixed times that creates a partition on $[0, 1]$ to let τ_i fall into one of the corresponding intervals, with the probability indicated in Figure 2. According to this, if $t_{j-1} < \tau_i \leq t_j$, which happens w.p. p_j, $j = 1, \ldots, n$, then $X_i(t)$ switches from 0 to M_j, and remains in M_j forever. If $\tau_i > 1$, which happens w.p. p_0, $X_i(t)$ remains in 0 forever. As a result, $X_i(1) = M_j$ w.p. p_j, $j = 1, \ldots, n$, and $X_i(1) = 0$ w.p. p_0, the same distribution as the one proposed in (19).

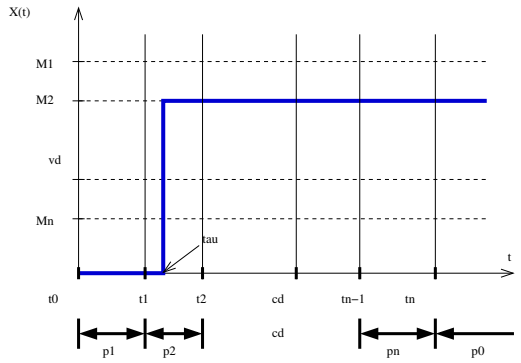

Fig. 2. The ML–CP mechanism for assigning values to $X_i(t)$

In [10] it is proved that the parameters that meet all these requirements are:

$$\begin{cases} \lambda &= -\ln(p_0), \\ t_j &= \dfrac{\ln(1 - p_1 - p_2 - \ldots - p_j)}{\ln(p_0)} \qquad j = 1, \ldots, n-1. \end{cases} \quad (21)$$

Thus, as the arcs are "repaired" as explained, the flow $V(\mathbf{X}(t))$ becomes a constant piece–wise increasing function of time, three replications of which are shown in Figure 3.

Every jump in $V(\mathbf{X}(t))$ corresponds to a repair, whereas not necessarily every arc repair has a jump associated with. Now, the attention must be focused on the evolution of the replications of $V(\mathbf{X}(t))$ — seen as trajectories— to check whether they are below D at $t = 1$, or not. This can be the basis of an SMC unreliability estimation, because $V(\mathbf{X}(1)) < D$ is an event that indicates the network failure, just as $\{\tau_{(c)} > 1\}$ in CP, or $\{\phi(\mathbf{X}) = 0\}$ in SNM.

The value of $V(\mathbf{X}(t))$ can be easily determined at any time, by means of any of the well known algorithms like Ford–Fulkerson [32], Edmonds–Karp [33] or Goldberg–Rao [34].

If N independent copies of the trajectories $V(\mathbf{X}(t))^{(i)},\ t \geq 0,\ i = 1, \ldots, N$, are sampled, it is possible to compute the following indicator variable, for each one of them:

$$I^{(i)}_{\text{ML–CP}} = \begin{cases} 1 & \text{if } V(\mathbf{X}(1))^{(i)} < D, \\ 0 & \text{otherwise.} \end{cases} \quad (22)$$

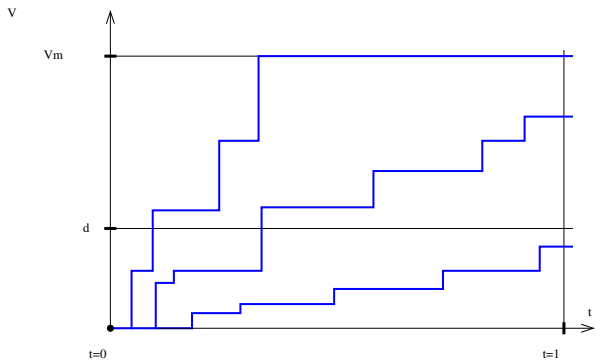

Fig. 3. Increasing of $V(\mathbf{X}(t))$ as the arcs are *repaired*.

Finally, the SMC estimation of ζ, subject to the ML–CP, is:

$$\widehat{\zeta}_{\text{ML–CP}} = \frac{1}{N}\sum_{i=1}^{N} I^{(i)}_{\text{ML–CP}}. \tag{23}$$

As shown in Figure 4, it is not necessary to simulate all the trajectories up to $t = 1$ because, as $V(\mathbf{X}(t))$ is an increasing function of time, whenever a trajectory reaches D at some time $t < 1$, it is guaranteed that such trajectory will not be below D at $t = 1$.

As in the unreliability estimation, the event of interest is the network failure, and considering that here the network failure is modeled by trajectories that at $t = 1$ are still below D (i.e. trajectories associated with points like (3)), we say that these trajectories —for which the event of interest actually occurs— are *successful*. Similarly, we call *unsuccessful* those trajectories for which the event of interest does not occur (for example, those associated with points (1) and (2)), even though they model the operational state of the network.

Thus, the SMC unreliability estimation, subject to the ML–CP, $\widehat{\zeta}_{\text{ML–CP}}$, is the ratio between the number of *successful* trajectories and the total number of trajectories initiated at $t = 0$.

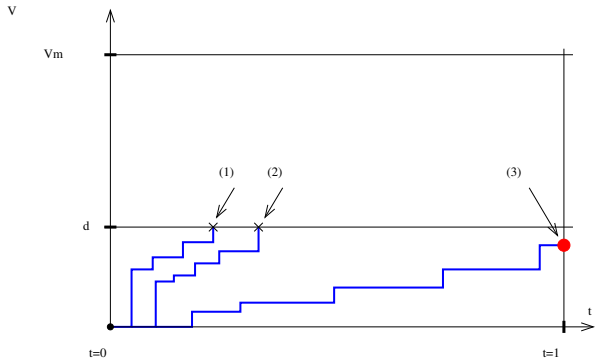

Fig. 4. *Successful* and *unsuccessful* trajectories in ML–CP

For highly reliable networks, most of the trajectories —almost all of them— will be *unsuccessful*. A possible interpretation of this fact is as follows: as highly reliable networks are "repaired very soon", trajectories rise up immediately, being extremely rare to find them below D at $t = 1$. The unreliability estimation in (23) needs, therefore, to be improved by a variance reduction technique. Otherwise, when the true value of the unreliability is very small, but not zero, the outcome of the estimations will always be zero.

The solution proposed in [10] consists of the use of Splitting as shown in Figure 5. As the points of interest are the ones associated with trajectories crossing the bound $t = 1$ below D (the red ones), the candidates are those trajectories that, at any $t < 1$, are still underneath D. So, trajectories for which $V(\mathbf{X}(t)) < D$ at times like $\ell_1, \ell_2, \ldots$, are cloned, attempting to finally increase the number of red points. This application of Splitting over the ML–CP [10], with thresholds $\ell_1, \ell_2, \ldots$, achieves an important variance reduction compared to the crude estimation in (23).

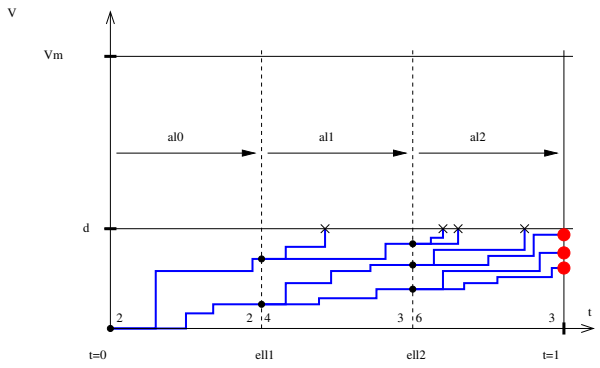

Fig. 5. Splitting over the ML–CP

In a setting with N_T thresholds (out of which, $\ell_0 = 0$ and $\ell_{N_T+1} = 1$ are implicit), if α_i is the probability that a trajectory started at threshold ℓ_i reaches threshold ℓ_{i+1} underneath D, $i = 0, \ldots, N_T$, then:

$$\zeta = \prod_{i=0}^{N_T} \alpha_i. \tag{24}$$

An estimator per stage, $\widehat{\alpha}_i$, can be computed as the ratio between the number of trajectories reaching ℓ_i below D, and the number of trajectories started from ℓ_{i-1}. Once the simulation is complete, and one estimator per stage is computed, the Splitting network unreliability estimation, subject to the ML–CP, is:

$$\widehat{\zeta}^{\mathrm{S}}_{\text{ML–CP}} = \prod_{i=0}^{N_T} \widehat{\alpha}_i. \tag{25}$$

Some experiments in [10] show that the variance reduction achieved by Splitting over the ML–CP, produce estimations whose relative errors are less than 1% for unreliabilities in the order of 2.00×10^{-12}. The duration of the simulations is, approximately, 1.5 million times less than the time that would take to perform SMC simulations, over the same network model, to achieve the same variance and, consequently, the same relative error.

VI. The Multi–Level DP

The method to be introduced in this section, named Multi–Level DP, is the core of this article and also one of its main contributions.

Without any loss of generality, let the arcs' capacities be modeled by the following set of discrete random variables:

$$X_i = \begin{cases} M_n & \text{w.p. } p_n, \\ M_{n-1} & \text{w.p. } p_{n-1}, \\ \quad\vdots & \qquad M_n > \ldots > M_0 = 0, \\ M_1 & \text{w.p. } p_1, \\ M_0 & \text{w.p. } p_0, \end{cases}$$

$$i = 1, \ldots, m. \quad (26)$$

The ML–DP transforms the random vector $\mathbf{X}$ into the stochastic process $\mathbf{X}(t) = (X_1(t), \ldots, X_m(t))$, in which:

$$X_i(t) = \begin{cases} M_n & \text{if } t < \tau_i, \\ M_j & \text{if } t \geq \tau_i, \end{cases} \qquad i = 1, \ldots, m \quad \text{and}$$

$$0 \leq j \leq n-1. \quad (27)$$

A particular type of "failure" occurs at time τ_i, as the value of $X_i(t)$ switches from M_n, which means completely *operational*, to one of the *partially failed* states, namely, to one of the possible $M_j < M_n$. The mechanism proposed for deciding which M_j is assigned to $X_i(t)$ at time τ_i, is explained next.

Let us accept, by now, that there exists a set $\{\lambda, t_0 = 0, t_1, t_2, \ldots, t_{n-1}, t_n = 1\}$ such that λ is the rate of the exponentially distributed random time τ_i, and $t_1, t_2, \ldots, t_{n-1}$, a set of fixed times that creates a partition on $[0, 1]$ to let τ_i fall into one of the corresponding intervals, with the probability indicated in Figure 6. According to this, if $t_j < \tau_i \leq t_{j+1}$, which happens w.p. p_j, $X_i(t)$ switches from M_n to M_j, and remains in M_j forever. If $\tau_i > 1$, which happens w.p. p_n, $X_i(t)$ remains in M_n forever. As a result, $X_i(1) = M_j$ w.p. p_j, $j = 0, \ldots, n$, the same distribution as the one proposed in (26).

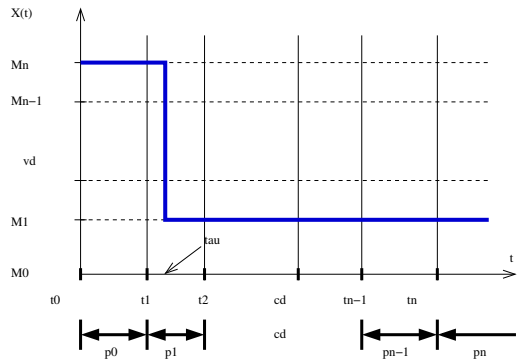

Fig. 6. The ML–DP mechanism for assigning values to $X_i(t)$

The parameters that meet all these requirements are:

$$\begin{cases} \lambda = -\ln(p_n), \\ t_j = \dfrac{\ln(1 - p_0 - p_1 - \ldots - p_{j-1})}{\ln(p_n)}, \quad j = 1, \ldots, n-1. \end{cases} \quad (28)$$

Thus, as the arcs "fail" as explained, the flow $V(\mathbf{X}(t))$ becomes a constant piece–wise decreasing function of time, three replications of which are shown in Figure 7.

Every jump in $V(\mathbf{X}(t))$ corresponds to some failure, whereas not necessarily every arc failure has a jump associated with. Now, the attention must be focused on the evolution of the replications of $V(\mathbf{X}(t))$, to see whether they are below D at $t = 1$, or not.

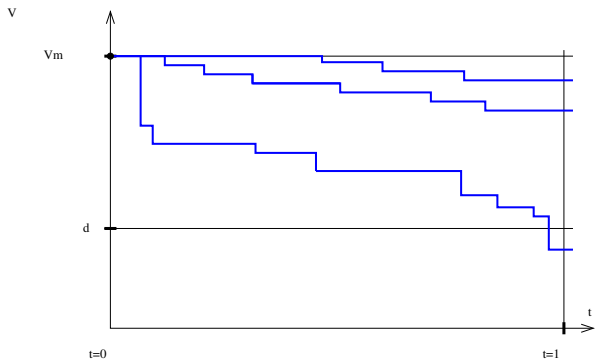

Fig. 7. Decreasing of $V(\mathbf{X}(t))$ as the arcs fail.

Here, $V(\mathbf{X}(1)) < D$ is an event that indicates the network failure, just as $\{\tau_{(c)} < 1\}$ in DP, $\{\phi(\mathbf{X}) = 0\}$ in SNM or $\{F(\mathbf{X}) < D\}$ in SFNM. Then, based on the ML–DP model, a SMC unreliability estimation can be obtained as follows.

Given N independent copies of the trajectories $V(\mathbf{X}(t))^{(i)}$, $t \geq 0$, $i = 1, \ldots, N$, it is possible to compute the following indicator variable, for each one of them:

$$I^{(i)}_{\text{ML-DP}} = \begin{cases} 1 & \text{if } V(\mathbf{X}(1))^{(i)} < D \\ 0 & \text{otherwise.} \end{cases} \tag{29}$$

Finally, the SMC estimation of ζ, subject to the ML–DP, is:

$$\widehat{\zeta}_{\text{ML–DP}} = \frac{1}{N} \sum_{i=1}^{N} I^{(i)}_{\text{ML–DP}}. \tag{30}$$

As shown in Figure 8, it is not necessary to simulate all the trajectories until $t = 1$ because, as $V(\mathbf{X}(t))$ is a decreasing function of time, whenever $V(\mathbf{X}(t)) < D$ for some $t < 1$, it is guaranteed that such trajectory will be below D at $t = 1$.

Here, trajectories are classified exactly as in section V. According to this, the ones associated with points like (3) are *successful*, whereas the ones associated with points like (1) and (2) are *unsuccessful*.

Thus, the SMC unreliability estimation, $\widehat{\zeta}_{\text{ML–DP}}$, is the ratio between the number of *successful* trajectories and the total number of trajectories initiated at $t = 0$.

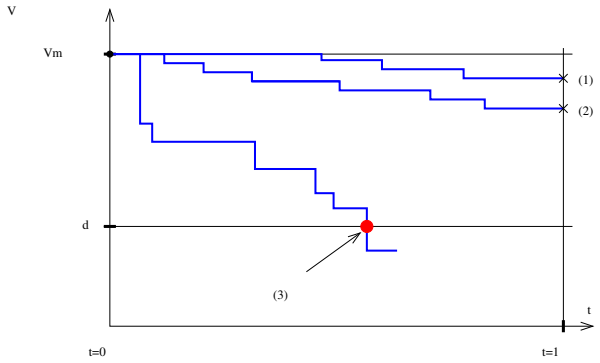

Fig. 8. *Successful* and *unsuccessful* trajectories in ML–DP

For highly reliable networks, most of the trajectories —almost all of them— will be *unsuccessful*. A possible interpretation of this fact is as follows: as highly reliable networks "almost never fail", trajectories are most of the time close to V_{max}, being extremely rare to find them below D, at $t = 1$.

Thus, if the networks under analysis are highly reliable, the estimation in (30) needs to be improved by a variance reduction technique, otherwise, when the true value of the unreliability is very small, but not zero, the outcome of the estimations will always be zero.

The solution proposed in this article consists of an application of Splitting over the ML–DP, as shown in Figure 9.

Keep in mind that for highly reliable networks, the arcs are most of the time at their maximum capacity, M_n, or at least at capacity values that are close to M_n. Then, if all —or almost all— the arcs are most of the time close to their maximum capacity, trajectories tend to be all the time close to V_{max}, and they rarely fall down.

As we are interested in the probability that the trajectories reach the bound D earlier than $t = 1$, it is necessary to stimulate trajectories that tend to deviate from V_{max}, "pushing" them down to D. Splitting can do this, and it does it by cloning recursively the trajectories that crosses the thresholds located between V_{max} and D. Formally speaking, the candidates are those trajectories for which $V(\mathbf{X}(t)) < V_{max}$, at times $t < 1$. Then, trajectories down–crossing the thresholds $\ell_1, \ell_2, \ldots$, earlier than $t = 1$ are cloned, finally increasing the number of red points.

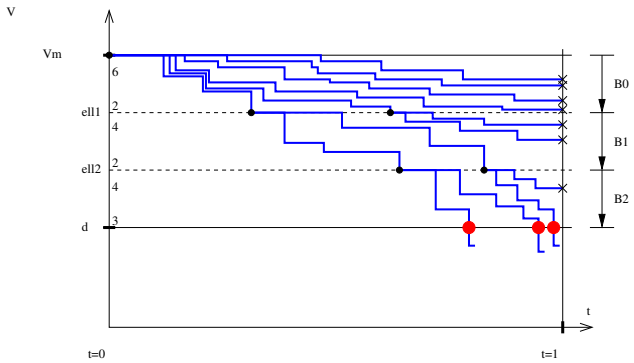

Fig. 9. Splitting over the ML–DP

In a general setting we have N_T thresholds (out of which, $\ell_0 = V_{max}$ and $\ell_{N_T+1} = D$ are implicit). Calling β_i the probability that a trajectory started at threshold ℓ_i reaches threshold ℓ_{i+1} earlier than $t = 1$, and accepting that ζ is the probability that a trajectory that starts from V_{max} at $t = 0$, reaches D earlier than $t = 1$, then:

$$\zeta = \prod_{i=0}^{N_T} \beta_i. \tag{31}$$

As we are interested in an estimator for ζ,we need a set of estimators for β_i, $i = 0, \ldots, N_T$. The estimators $\widehat{\beta}_i$ can be computed as the ratio between the number of trajectories reaching ℓ_i earlier than $t = 1$, and the number of trajectories started from ℓ_{i-1}.

Finally, the Splitting network unreliability estimation, subject to the ML–DP, is:

$$\widehat{\zeta}^{\mathrm{S}}_{\text{ML–DP}} = \prod_{i=0}^{N_T} \widehat{\beta}_i. \tag{32}$$

In the example shown in Figure 9 six trajectories are launched at $t = 0$, all of them with capacity $V(\mathbf{X}(0)) = V_{max}$. They evolve and two of them reach ℓ_1 earlier that $t = 1$, therefore, $\widehat{\beta}_0 = 2/6 = 0.33$. As a consequence, four new trajectories are launched from ℓ_1, two from every crossing point (we consider a splitting factor of 2). These four trajectories evolve and two of them reach ℓ_2 producing $\widehat{\beta}_1 = 2/4 = 0.50$. Finally, $\widehat{\beta}_2 = 3/4 = 0.75$, what makes $\widehat{\zeta}^{\mathrm{S}}_{\text{ML–DP}} = 0.33 \times 0.50 \times 0.75 = 0.12$.

VII. SFNM WITH NON–INDEPENDENT ARC FAILURES

In all the models used so far in this article, a failure affecting a single component (either a link in SNM or an arc in SFNM) is independent from failures affecting the other components. This hypothesis, though an important mathematical simplification, is often quite far from real networks behavior.

Systems with dependencies have been studied in recent times, with a particular emphasis in the last few years. Works like [20], [21], [35] and [36] address models with dependent failures and show that the dependency has a high impact in the reliability value.

Think for example in the very simple SFNM shown in Figure 10. According to the capacities, X_1 and X_2, there is a maximum amount of flow generated in node s that can reach node t. This maximum is simply $X_1 + X_2$. Suppose that the service requirement is such that $D = 0$, meaning that any amount of flow from s to t above zero is enough and, therefore, the only failure condition corresponds to the event $X_1 + X_2 = 0$. The unreliability of this network is then, $\zeta = \mathbb{P}\{X_1 + X_2 = 0\}$ or, considering that

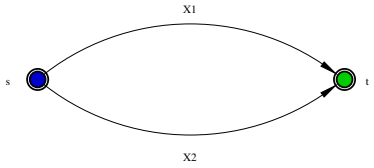

Fig. 10. Simple Stochastic Flow Network

$X_1, X_2 \geq 0$:

$$\zeta = \mathbb{P}\{X_1 = 0 \wedge X_2 = 0\} \tag{33}$$

Then, when both arcs are independent, the unreliability, called now ζ^i, is:

$$\zeta^i = \mathbb{P}\{X_1 = 0\} \times \mathbb{P}\{X_2 = 0\} \tag{34}$$

This setting, in which the behavior of every component is independent from those of the rest, is the background idea of all the models considered so far in this article.

Suppose now that, due to any reason, the behavior of both arcs is positive correlated in the following sense: if the event $X_2 = 0$ has actually occurred, the probability of event $X_1 = 0$ is increased with respect to the corresponding marginal, that is $\mathbb{P}\{X_1 = 0 \mid X_2 = 0\} > \mathbb{P}\{X_1 = 0\}$. Suppose also that failures in arc 1 influence arc 2 in the same way, that is $\mathbb{P}\{X_2 = 0 \mid X_1 = 0\} > \mathbb{P}\{X_2 = 0\}$. Then, the expressions above can be easily adapted to show that now the network unreliability, called ζ^d, and ζ^i are related as follows:

$$\zeta^d = \mathbb{P}\{X_1 = 0 \wedge X_2 = 0\} > \mathbb{P}\{X_1 = 0\} \times \mathbb{P}\{X_2 = 0\} = \zeta^i. \tag{35}$$

The opposite ($\zeta^d < \zeta^i$) occurs if the behavior of the arcs is negative correlated. Thus, to get closer to a realistic value of the unreliability (either as and estimate or an exact computation) it is important to find the way to consider dependencies among the network components.

Think now of an SFNM, in which the arcs capacity distribution is not necessarily the same for every arc. In the ML–DP introduced earlier in this article, the arcs fail independently from each other at times determined by the exponentially distributed random variables $\tau_i,\ i = 1, \ldots, m$. No matter which $M_j < M_n$ is the capacity after τ_i, the failure actually occurs at time τ_i, whose probability density function is $\lambda_i \, \mathrm{e}^{-\lambda_i \, x}$ over $[0, \infty)$. Thus:

$$\mathbb{P}\{\tau_i > x\} = \mathrm{e}^{-\lambda_i \, x}, \quad i = 1, \ldots, m. \tag{36}$$

This distribution satisfies the lack of memory property so, given the times $x, y \geq 0$,

$$\mathbb{P}\{\tau_i > x + y \mid X > y\} = \mathbb{P}\{\tau_i > x\}, \quad i = 1, \ldots, m, \tag{37}$$

indicating that the residual lifetime of each arc is independent of its age. For single continuous random variables, the only probability distribution that satisfies the lack of memory property is the exponential distribution. The lack of memory property can be extended to the random vector $(\tau_1, \ldots, \tau_m)$, over $[0, \infty)^m$ if, given the times $x_1, \ldots, x_m$, and $y \geq 0$:

$$\mathbb{P}\{\tau_1 > x_1 + y, \ldots, \tau_m > x_m + y \mid \tau_1 > y, \ldots, \tau_m > y\} = \mathbb{P}\{\tau_1 > x_1, \ldots, \tau_m > x_m\}. \quad (38)$$

This implies that the residual lifetime of the arcs is independent of the age of the whole set of arcs. If $m = 1$, as stated above, τ_1 is exponentially distributed. For the case of $m = 2$, Marshall and Olkin have shown in [23] that, if τ_1 and τ_2 are both exponentially distributed with parameters $\lambda_1, \lambda_2 > 0$, there exists a third parameter $\lambda_{1,2} \geq 0$, such that the following bivariate distribution:

$$\mathbb{P}\{\tau_1 > x_1, \tau_2 > x_2\} = \mathrm{e}^{(-\lambda_1 x_1 - \lambda_2 x_2 - \lambda_{1,2} \max\{x_1, x_2\})} \quad (39)$$

satisfies the lack of memory property.

The distributions referred to in (36) and (39) verify the following two properties: they are exponentials, so that every τ_i is useful to model the lifetime of an arc, and they satisfy the lack of memory property. It is worth noting that, in the case of the bivariate distribution, the parameter $\lambda_{1,2}$ is responsible for a dependence between τ_1 and τ_2. As said in [37], expression (39) makes the marginal distributions of τ_1 and τ_2 be coupled correctly in order to let the dependence structure guarantee the lack of memory property. See also that, if $\lambda_{1,2} = 0$, the dependency is extinguished and τ_1 and τ_2 become independent.

Marshall and Olkin also proved in [23] that for $m > 2$, it is possible to find a distribution with the lack of memory property in which the marginals are exponentials, if there is a distribution with these two properties for all the $(m-1)$–subvectors of $(\tau_1, \ldots, \tau_m)$. As these distributions exist for $m = 1$ and $m = 2$, it follows inductively that they exist $\forall\, m$. The general case of this distribution ($m \geq 1$), called Marshall–Olkin distribution, is going to be introduced next.

Let $I \subset V = \{1, \ldots, m\}$ be a non empty subset of the set of arcs, and let C be the collection of all the possible sets I, except for the empty set. A random vector $(\tau_1, \ldots, \tau_m)$ has the Marshall–Olkin distribution if:

$$\mathbb{P}\{\tau_1 > x_1, \ldots, \tau_m > x_m\} = \mathrm{e}^{\left(-\sum\limits_{I \in C} \lambda_I \max\limits_{i \in I}\{x_i\}\right)}, \quad (40)$$

where $\lambda_I \geq 0$. As there are 2^m different sets in V, but there is no need to consider the empty set, $\mid C \mid = 2^m - 1$.

If, for example, $m = 3$, $V = \{1, 2, 3\}$ and $C = \{\{1\}, \{2\}, \{3\}, \{1, 2\}, \{1, 3\}, \{2, 3\}, \{1, 2, 3\}\}$, then (τ_1, τ_2, τ_3) has the Marshall–Olkin distribution if $\mathbb{P}\{\tau_1 > x_1, \tau_2 > x_2, \tau_3 > x_3\} = \mathrm{e}^{\psi}$, where $\psi = -\lambda_1 x_1 - \lambda_2 x_2 - \lambda_3 x_3 - \lambda_{1,2} \max\{x_1, x_2\} - \lambda_{1,3} \max\{x_1, x_3\} - \lambda_{2,3} \max\{x_2, x_3\} - \lambda_{1,2,3} \max\{x_1, x_2, x_3\}$.

When the arcs are independent, their expected lifetimes are $1/\lambda_i,\ i = 1, \ldots, m$, and this is all what a simulation needs for sampling the failing times. When the hypothesis of independence does not hold, and one attempts to model the network in terms of the Marshall–Olkin distribution, some of the parameters —precisely those who measure the degree of dependency— may not be easy to determine. To overcome this difficulty, the model can be expressed in terms of a different set of variables, which is more explicitly related to the behavior of the components. These new variables, called here $E_I,\ I \in C$, are independent and exponentially distributed with parameter λ_I. These variables are the components of the so–called Marshall–Olkin copula and they are related to the vector $(\tau_1, \ldots, \tau_m)$, as follows:

$$\tau_k = \min\{E_I \mid I \in C,\ k \in I\}, \quad k = 1, \ldots, m \quad (41)$$

where

$$\sum_{I : k \in I} \lambda_I > 0 \quad \forall\, k = 1, \ldots, m. \quad (42)$$

Expression (42) shows that every arc must belong to, at least, one I with $\lambda_I > 0$.

Now, we can analyze the model in terms —only— of the set of variables E_I, with the following interpretation. All the arcs in I start to operate at time $t = 0$, and they fail simultaneously, as the consequence of a *shock* occurring at time E_I. Any arc of the network fails when the first *shock* including it occurs, just as pointed out in (41).

It remains to be proved that the failing times determined by the variables $E_I,\ I \in C$, in the Marshall–Olkin copula model, are such that the vector $(\tau_1, \ldots, \tau_m)$ follows the Marshall–Olkin distribution.

For simplicity, let us first analyze the case of networks composed of only two arcs, for which the life times are τ_1 and τ_2. Here, the surviving condition is:

$$\left\{ \begin{array}{ll} \text{For arc 1:} & \tau_1 > x_1, \\ \text{For arc 2:} & \tau_2 > x_2, \end{array} \right. \quad \rightarrow \quad (\tau_1 > x_1 \wedge \tau_2 > x_2),$$

In terms of the Marshall–Olkin copula, this condition becomes:

$$\left\{ \begin{array}{ll} \text{For arc 1:} & E_1 > x_1 \wedge E_{1,2} > x_1, \\ \text{For arc 2:} & E_2 > x_2 \wedge E_{1,2} > x_2, \end{array} \right. \quad \rightarrow$$

$$(E_1 > x_1 \wedge E_2 > x_2 \wedge E_{1,2} > \max\{x_1, x_2\}). \quad (43)$$

Let us now go back to the general case of a network composed of m arcs. Then, expression (43) takes the form:

$$(E_I > \max_{i \in I}\{x_i\}, \quad \forall I \in C)$$

Finally:

$$\mathbb{P}\{\tau_1 > x_1, \ldots, \tau_m > x_m\} = \mathbb{P}\{E_I > \max_{i \in I}\{x_i\}, \quad \forall I \in C\} \quad (44)$$

$$= \prod_{I \in C} \mathrm{e}^{-\lambda_I \max_{i \in I}\{x_i\}} \quad (45)$$

$$= \mathrm{e}^{\left(- \sum_{I \in C} \lambda_I \max_{i \in I}\{x_i\} \right)}. \quad (46)$$

This proves that, when the failing times are determined by the variables $E_I,\ I \in C$, vector $(\tau_1, \ldots, \tau_m)$ follows the Marshall–Olkin distribution. As a result, in order to build a network model under the Marshall–Olkin distribution, one has to focus on the failing times, and proceed as follows:

1) Select the arcs for which one knows the mean time of failures that depend only on the arc and that are not related with the rest of the network. With this information, set properly the parameters λ_I of the variables E_I for which I contains only the selected arc.
2) Select sets of arcs for which one knows the mean time of the simultaneous failure of the set (*shock*), for failures that depend only on the selected set and that are not related with the rest of the network. With this information, set the parameters λ_I of the variables E_I for which I contains the elements of the selected set.

Every $I \in C$ not considered in any of these two items is implicitly associated with variables E_I with $\lambda_I = 0$.

VIII. Experimental tests

This section shows the results of some experiments in which the SFNM unreliability is estimated by means of Splitting over the ML–DP, for the case of non–independent arc failures.

Let us first point out how Splitting over the ML–DP fits the model introduced in Section VII. The time to failure in sets composed of only one arc are modeled by exponentially distributed random variables E_I, $I \in \{\{1\}, \{2\}, \ldots, \{m\}\}$, and it is reasonable to consider all the sets, because each one of these variables

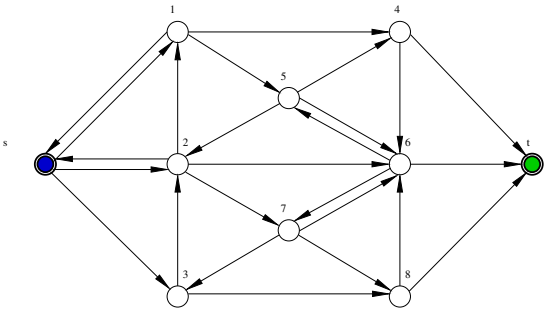

Fig. 11. *Fishman* Network

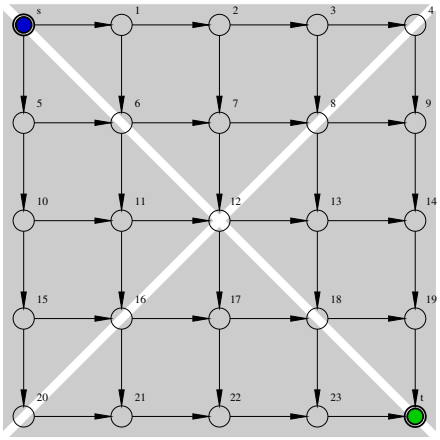

Fig. 12. Lattice 5×5 Network

models the individual behavior of every arc, i.e. the behavior without consideration of the dependencies, which is always the first description at hand in the analysis of systems' components. Besides, the sets composed of more than one arc are modeled in terms of variables E_I, where I is a reference to a set of arcs. For example, if $I = \{1, 3, 12\}$, the variable E_I models the simultaneous failure of arcs 1, 3 and 12. It is usual, therefore, to consider only a few number of sets, out of all the possible ones, because to observe the joint behavior of a number of arcs, that may not even be close to each other, is not an easy task or, at least, is not as easy as observing only one arc.

All the experiments are based on the networks shown in Figures 11 and 12. These benchmark topologies are some of those usually employed in the literature; and we experiment with a wide range of arc failure probability values, as this parameter has a larger effect in the performance of the methods than the topology itself. In both of them all the sets composed of one arc have the same probability distribution, thus, we will refer to them as I_s (the s indicates *single*). The sets composed of more than one arc also share the same probability distribution; we will refer to them as I_m (the m indicates *multiple*). In other words, I_s is a reference to *any* of the sets composed of one arc, whereas I_m a reference to *any* of the sets composed of more than one arc. In the case of the *Fishman* Network, the sets I_m are used to model failures in nodes, because they consider the simultaneous failure of the arcs that *touches* the same node (for both, incoming and outgoing arcs). There is one set I_m per node, except for $s = 0$ and $t = 9$. In the case of the Lattice 5×5 Network there are four sets I_m clearly indicated (shaded) in Figure 12.

The dynamics of the sets I_s is simple and clear: the arcs start at their maximum capacity value M_n and, at a time determined by the variable E_{I_s} they fail, what in terms of the mechanism introduced in Section VI, means that their capacities drop to some $M_j < M_n$. The dynamics of the sets I_m is quite similar, with the only difference of a simultaneous change in the capacity value of all the arcs that belong to the set. Thus, at a time determined by the variable E_{I_m}, the capacity of all the arcs in the set drop, but not necessarily all of them to the same value. The ML–DP is flexible enough to permit the definition of simultaneous failures in which the arcs drop to capacity values that are not necessarily equal. See also that the capacity value after a failure of an arc as a single set, does not necessarily need to be the same as the capacity after the failure of the same arc, as part of a set. And the same holds for one arc as part of different sets.

For both of the networks used in the experiments, we propose the capacity distributions shown in Table I, Type A and Type B. These sets of capacities were selected with the purpose of testing Splitting over the ML–DP over two different network models. See that, for the sets I_s, in the Type B the capacity values are a bit more scattered than in Type A, thus, the variance for Type B is larger that the variance for Type A. For the sets I_m, the distribution, and therefore the variance, is the same for both types. Assuming that, in real models, a failure affecting any arc as part of a multiple set should be rarer than a failure affecting

the same arc itself, we proposed that the probability that the capacity of a single arc drops to zero is ten times larger than the probability that the capacity of all the arcs of a multiple set drop to zero.

All the probabilities in Table I are based on p_0 —the probability that any arc is completely *failed*, without consideration of dependencies—. This value is used as a parameter to identify the experiments, as it is a reference to the different instances of the network models. According to the explanations given in Sections VI and VII, the rates λ_{I_s} of the variables E_{I_s} are $-\ln(p_8)$ for Type A and $-\ln(p_{10})$ for Type B, whereas the rates λ_{I_m} of the variables E_{I_m} are $-\ln(p'_8)$ for Type A and $-\ln(p'_{10})$ for Type B.

TABLE I
CAPACITY DISTRIBUTION OF THE NETWORKS USED IN THE EXPERIMENTAL TESTS

	Type A			Type B		
I_s	8	w.p.	$p_8 = 1 - p_0 - p_2 - p_4$	10	w.p.	$p_{10} = (1 - p_1 - p_0) \times 2/3$
	4	w.p.	$p_4 = p_0/100$	9	w.p.	$p_9 = (1 - p_1 - p_0) \times 1/3$
	2	w.p.	$p_2 = p_0/10$	1	w.p.	$p_1 = p_0/2$
	0	w.p.	p_0	0	w.p.	p_0
I_m	8	w.p.	$p'_8 = 1 - p'_0$	10	w.p.	$p'_{10} = 1 - p'_0$
	0	w.p.	$p'_0 = p_0/10$	0	w.p.	$p'_0 = p_0/10$

Tables II and III show some results in which the efficiency of Splitting over the ML–DP is evaluated in terms of the *Speedup*, which is defined as follows:

$$\textit{Speedup} = \frac{t_{\text{ML–DP}} \times \mathbb{V}_{\text{ML–DP}}}{t^{\text{S}}_{\text{ML–DP}} \times \mathbb{V}^{\text{S}}_{\text{ML–DP}}}, \tag{47}$$

where the numerator refers to the execution time and the variance of SMC estimation of ζ, subject to the ML–DP (see (30)), and the denominator refers to the same parameters for the method under test, i.e. Splitting over the ML–DP. Assuming that the most efficient method is the one referred to in the denominator, the *Speedup* is an indication of the improvement in the execution time to achieve the same variance as the worst method, or, the variance reduction achieved by the best method when the execution times of both methods are equal.

The splitting implementation in all the experiments is Fixed–Splitting, with a splitting factor of 2. The number N of trajectories launched was selected in order to obtain a relative error (RE) of, approximately, 3%. As expected, the *Speedup* increases as the network becomes more reliable. The method under test seems to be more efficient for networks of Type B, in which the variance of the capacity distribution is higher.

TABLE II
EFFICIENCY EVALUATION OF THE UNRELIABILITY OVER NETWORKS TYPE A

Network	p_0	N	N_T	$\widehat{\zeta}^{\text{S}}_{\text{ML–DP}}$	RE	$t^{\text{S}}_{\text{ML–DP}}$	*Speedup*
Fishman $D = 15$	0.001	8.00E+03	4	1.52E−02	2.98%	0.20 sec.	1
	0.0001	4.50E+05	7	1.44E−04	3.03%	1.68 sec.	9
	0.00001	6.00E+06	13	1.37E−06	3.00%	15.87 sec.	98
	0.000001	1.60E+08	18	1.40E−08	2.88%	6.46 min.	400
	0.0000001	2.10E+09	24	1.38E−10	2.82%	123.29 min.	2,186
Lattice 5×5 $D = 7$	0.001	1.00E+05	6	4.58E−03	1.17%	22.93 sec.	1
	0.0001	1.00E+06	9	4.70E−05	2.55%	17.01 sec.	14
	0.00001	1.00E+07	15	4.78E−07	2.81%	87.82 sec.	193
	0.000001	1.00E+08	22	4.71E−09	3.14%	16.97 min.	1,411
	0.0000001	1.00E+09	27	4.74E−11	4.96%	186.67 min.	5,097

In the second set of experiments the objective was to observe the influence of the dependency on the unreliability estimations. To do this, the distribution of the sets I_s was fixed to the one corresponding

TABLE III
EFFICIENCY EVALUATION OF THE UNRELIABILITY OVER NETWORKS TYPE B

Network	p_0	N	N_T	$\widehat{\zeta}^{\mathrm{S}}_{\mathrm{ML-DP}}$	RE	$t^{\mathrm{S}}_{\mathrm{ML-DP}}$	*Speedup*
	0.01	1.00E+04	8	5.80E−03	2.48%	0.64 sec.	2
	0.001	5.00E+05	14	5.82E−05	1.13%	25.93 sec.	17
Fishman	0.0001	1.00E+06	18	5.41E−07	2.42%	76.28 sec.	156
$D = 15$	0.00001	5.00E+06	26	5.98E−09	2.56%	11.58 min.	1,223
	0.000001	2.50E+07	30	5.91E−11	4.03%	93.35 min.	5,800
	0.01	1.50E+04	8	2.00E−03	2.66%	6.37 sec.	2
	0.001	1.00E+05	15	1.85E−05	2.62%	58.48 sec.	20
Lattice 5×5	0.0001	1.00E+06	19	1.87E−07	2.90%	10.11 min.	118
$D = 7$	0.00001	6.00E+06	26	1.91E−09	2.96%	116.06 min.	1,083
	0.000001	2.50E+07	27	1.90E−11	5.65%	386.32 min.	9,321

to $p_0 = 0.00001$. Table IV reports the results. The values of the probability p'_0 were proposed, and the corresponding values of p'_8, for Type A, and p'_{10}, for Type B, were set accordingly. For extremely low values of p'_0, the failures of the sets I_m have no influence and the unreliability estimated is the same as the corresponding network model in which the arcs are all independent from each other. But when the values of p'_0 become larger —and extremely larger— the unreliability is mostly determined by the shocks affecting sets of multiple arcs. See that when the difference in the probabilities is large in favor of the sets I_m, making that the arcs almost never fail individually, the behavior of Lattice 5×5 Network is only determined by the four shaded sets of arcs, each of which fails with probability p'_0. In this case it is easy to see that the unreliability becomes ${p'_0}^4 + 4{p'_0}^3(1 - p'_0) + 2{p'_0}^2(1 - p'_0)^2$. For $p'_0 = 0.001$, $p'_0 = 0.01$ and $p'_0 = 0.1$ this last expression returns, respectively, the values 2.00×10^{-06}, 2.00×10^{-04} and 1.99×10^{-02} which are almost identical to the values reported in the last three lines of Table IV.

TABLE IV
UNRELIABILITY EVOLUTION, CHANGING THE PROBABILITY OF THE SHOCKS

Network	p'_0	N	N_T	$\widehat{\zeta}^{\mathrm{S}}_{\mathrm{ML-DP}}$	RE
	0.0000001	5.00E+06	26	5.56E−09	2.60%
	0.000001	5.00E+06	26	5.98E−09	2.56%
	0.00001	1.00E+07	21	9.58E−09	2.40%
Fishman	0.0001	2.00E+06	19	1.07E−07	2.56%
$D = 15$	0.001	1.00E+06	12	6.75E−06	1.89%
	0.01	5.00E+04	9	6.57E−04	2.07%
	0.1	1.00E+04	4	6.14E−02	1.34%
	0.0000001	6.00E+06	26	1.93E−09	2.96%
	0.000001	6.00E+06	26	1.91E−09	2.96%
	0.00001	1.00E+07	21	3.29E−09	2.74%
Lattice 5×5	0.0001	2.00E+06	20	3.21E−08	2.66%
$D = 7$	0.001	1.00E+06	12	2.06E−06	2.46%
	0.01	1.00E+05	9	1.87E−04	1.88%
	0.1	1.00E+04	5	1.99E−02	1.77%

It remains to explain the way trajectories are cloned at the cross of a threshold. Let us suppose that, as shown in Figure 13 (a), arc a fails at time t_k making $V(\mathbf{X}(t))$ drop from v_1 to v_2. If the splitting factor is 2, two new trajectories should start at point (1), both of them with the state of the original trajectory at this point. The state of the original trajectory is composed of two elements: (i) the set of arcs that have already failed at time ${t_k}^-$ and (ii) the arc a failing and moving $V(\mathbf{X}(t))$ from v_1 to v_2. We must consider, then, that two trajectories that verify (i) and (ii) must emerge from point (1), as shown in Figure 13 (b).

In order to continue these two new trajectories, for each one of them, a new arc will be sampled at time $t > t_k$ and, eventually $V(\mathbf{X}(t))$ will drop down to some $v < v_2$, as shown in Figure 13 (c). In summary, at the crossing in point (1), restart two new trajectories at t_k, both with $V(\mathbf{X}(t)) = v_2$ and accept that, at point (1), there is one incoming and two outgoing trajectories.

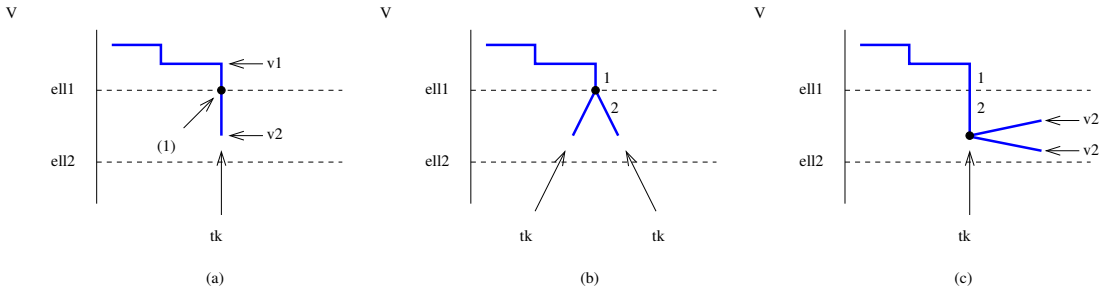

Fig. 13. Cloning a trajectory after crossing a single threshold

It is possible that, as shown in Figure 14 (a), due to the failure of some arc a, a trajectory crosses more than one threshold simultaneously. The mechanism is essentially the same, the new trajectories emerging at every crossing point must retain the same state. Trajectories are sequentially cloned, as shown in Figure 14 (b) and, at the end, it is only necessary to restart many new trajectories at t_k (eight, in this example), all of them with $V(\mathbf{X}(t)) = v_2$, and to make account of the incoming and outgoing trajectories at all the crossing points, as shown in Figure 14 (c).

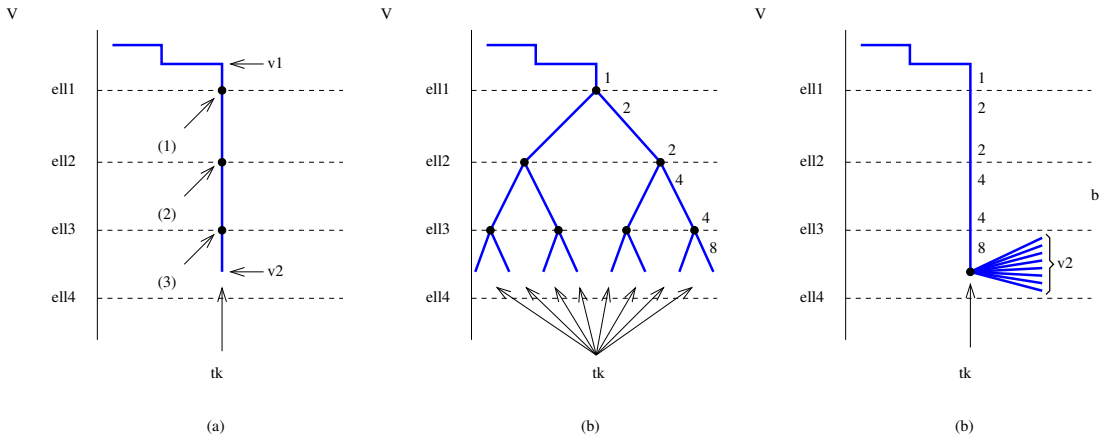

Fig. 14. Cloning a trajectory after crossing multiple thresholds

The optimal number of thresholds —which is referred to as N_T in Tables II, III and IV— is selected after a few pilot runs. It is interesting to point out that, in many cases, this optimal number is such that simulations are plenty of crosses of multiple thresholds simultaneously. Let us analyze, for example, the Lattice 5×5 Network, Type A. In this case, due to the possible state of the arcs, $V(\mathbf{X}(t)) \in \{0, 2, 4, 6, 8, 10, 12, 16\}$. The thresholds will, therefore, partition the space $[V_{max}, D]$, which, for the selected model, is $[16, 7]$. The minimum size of the step to traverse this space, for trajectories that start at V_{max} and end at D, is 2. This suggests that the only condition under which there will not be crosses of multiple thresholds simultaneously is $N_T \leq 4$ and only in the extremely fortuitous event in which *all* the steps are of size 2. Looking at the values of N_T in all the tables, it is clear that in most simulations multiple crosses occur.

The experiments were conducted on a DELL Inspiron 15 machine, equipped with a processor Intel(R) Core(TM) i5-8250U, operating system Linux, kernel version 5.4.0-89. All the implementations were programmed in C language, compiler gcc version 7.5.0.

The SFNM is a particular case of a Multi Component System Model (MCSM). In a typical MCSM there is a set of individual components that cooperate with each other, so that the whole system provides some type of service. The individual behavior of the components is known, and the way they interact with each other is also known. But the global response of a MCSM, as a function of the set of components, is usually unknown and sometimes very difficult to determine. The reliability of an MCSM is defined exactly as we have defined the reliability of an SFNM, that is, the probability that the system can deliver service as expected.

For any of these two models, the exact computation of the reliability is impossible or almost impossible when the size of the system, measured by the number of components (arcs in the SFNM), is large. The individual features and properties of the components is not a crucial issue as long as the way they interact with each other is clearly defined and known. The parameter that usually makes the reliability determination extremely difficult (the one that gives the problem its NP-hard character) is the number of components.

In the problem addressed in the paper, the components whose individual behavior is known are the arcs, and the way they cooperate with each other, so that the network can transport a certain amount of flow, is given by its topology. Here, the parameter whose value makes it necessary to improve simulation by variance reduction methods is the number of arcs. It is possible —and relatively simple— to transform the problem and to make the same evaluation considering failures in the nodes, or even considering different sets of nodes generating and consuming flow. But the difficulties in developing variance reduction methods, whatever form the network takes, will be about the same, and the resulting methods will be quite similar, as long as the number of components remain high or extremely high.

These are the reasons why, sometimes, important simplifications are made on the benchmark models used in the development of variance reduction methods, like for example, the use of only two terminal nodes or the fact that nodes are perfect and never fail. However, the change in difficulty resulting from disregarding these simplifications is minimal compared to the intrinsic complexity of the variance reduction methods.

For these types of simplified models, a complexity analysis made in [38] claims that, as a function of a rarity parameter, the complexity of the standard simulation methods is exponential in the number of components of the system, whereas the complexity of the Splitting–like methods is polynomial (quadratic). Similar results can be found, for instance, in Chapter 3 of [39].

IX. Conclusions

The main proposal of this article is the introduction of a Monte Carlo method designed to estimate the unreliability of highly reliable stochastic flow networks. It is based on an application of Splitting over the well-known Destruction Process, a model in which the components fail at exponentially distributed times. This makes the new procedure, called Splitting over the ML–DP, suitable to work on networks in which the components are non–independent, in terms of the Marshall–Olkin copula.

The paper also proposes a review of techniques based on CP and DP [9], which are graph evolution models originally conceived to work on SNM. The CP was adapted in [10], and the DP was adapted here in this article, to SFNM. Due to the fundamentals of the Marshall–Olkin copula model, that refers to failures that take place in exponentially distributed times, it was inevitable to prefer DP over CP to build the new method. And the only way Splitting can be applied to enhance the DP is by partitioning –by means of thresholds— the domain of $V(\mathbf{X}(t))$ and not the temporal domain.

The Marshall–Olkin copula model, so far the most accepted model to work with networks in which the components are not independent, is firstly mentioned in this article over a general setting, and then it is adapted to Splitting over the ML–DP with a high level of detail and providing many implementation guidelines.

The experimental tests show a high level of efficiency, reflected in the order of the estimated relative error bounds reached in very low simulation times. The performance improvement is also reflected in

the values of the *Speedup*, indicating, as expected for any efficient variance reduction method, that the improvements are more significant as the networks become more reliable. While the experiments were performed on two benchmark topologies, and other network configurations could result in other values, the tests employed a wide variety of probabilities for the different capacity distribution functions. This is a most important factor for the method performance, and in this sense we think the experimental results give a clear indication of the performance of the method.

It is interesting and somehow unexpected that the most efficient parameter settings are those that cause crosses of multiple thresholds simultaneously.

Given that the article introduces a new proposal, it is expected that many of its features will require further analysis. One of the topics to be considered as part of our future work is a theoretical analysis of robustness, for example, in terms of the measures discussed in [40].

Although the variance of the Splitting estimator is extremely difficult to compute exactly in these contexts, another interesting subject to consider, at least experimentally, is to check if there are conditions under which the model verifies the Bounded Relative Error property. The design of techniques to determine —or, at least, to approximate— the optimal number of thresholds and the multiplying Splitting factor are also of interest.

The capacity distributions that we worked with in this article are, in all cases, discrete. However, it is possible, and this is another interesting subject for future work, to adapt Splitting over the ML–DP to the case of continuous capacity distributions.

ACKNOWLEDGMENT

This work was partially supported by the Math AmSud project "RareDep" (Rare events analysis in multi–component systems with dependent components), 2019–2020–2021.

REFERENCES

[1] J. S. Provan and M. O. Ball, "The complexity of counting cuts and of computing the probability that a graph is connected," *SIAM J. Comput.*, vol. 12, no. 4, pp. 777–788, 1983. [Online]. Available: https://doi.org/10.1137/0212053

[2] H. Cancela and M. El Khadiri, "The recursive variance–reduction simulation algorithm for network reliability evaluation," *IEEE Transactions on Reliability*, vol. 52, no. 2, pp. 207–212, June 2003.

[3] H. Cancela, M. El Khadiri, and G. Rubino, "Rare events analysis by Monte Carlo techniques in static models," in *Rare Event Simulation using Monte Carlo Methods*, G. Rubino and B. Tuffin, Eds. John Wiley & Sons, 2009, ch. 7, pp. 145–170.

[4] M. C. Easton and C. K. Wong, "Sequential destruction method for Monte Carlo evaluation of system reliability," *IEEE Transactions on Reliability*, vol. R–29, no. 1, April 1980.

[5] G. S. Fishman, "A Monte Carlo Sampling Plan for Estimating Network Reliability," *Operations Research*, vol. 34, no. 4, 1986. [Online]. Available: http://dx.doi.org/10.2307/170604

[6] K. P. Hui, N. G. Bean, M. Kraetzl, and D. P. Kroese, "The cross–entropy method for network reliability estimation," *Annals of Operations Research*, vol. 134, pp. 101–118(18), February 2005.

[7] M. Lomonosov, "On Monte Carlo estimates in network reliability," *Probability in the Engineering and Informational Sciences*, vol. 8, pp. 245–264, 1994.

[8] H. Cancela, L. Murray, F. Robledo, P. Romero, and P. Sartor, "On the reliability estimation of stochastic binary systems," *International Transactions in Operational Research*, 2021. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/itor.13034

[9] T. Elperin, I. B. Gertsbakh, and M. Lomonosov, "Estimation of network reliability using graph evolution models," *IEEE Transactions on Reliability*, vol. 40, no. 5, pp. 572–581, Dec 1991.

[10] H. Cancela, L. Murray, and G. Rubino, "Efficient estimation of stochastic flow network reliability," *IEEE Transactions on Reliability*, vol. 68, no. 3, pp. 954–970, Sep. 2019.

[11] L. Murray, H. Cancela, and G. Rubino, "A Splitting algorithm for network reliability estimation," *IIE Transactions*, vol. 45, no. 2, pp. 177–189, 2013.

[12] P. Glasserman, P. Heidelberger, P. Shahabuddin, and T. Zajic, "Splitting for rare event simulation: Analysis of simple cases," in *Proceedings of the 28th conference on Winter simulation, WSC 1996, Coronado, CA, USA, December 8-11, 1996*, 1996, pp. 302–308. [Online]. Available: http://doi.ieeecomputersociety.org/10.1109/WSC.1996.873293

[13] P. L'Ecuyer, V. Demers, and B. Tuffin, "Rare events, splitting and Quasi-Monte Carlo," *ACM Trans. Model. Comput. Simul.*, vol. 17, no. 2, Apr. 2007. [Online]. Available: http://doi.acm.org/10.1145/1225275.1225280

[14] P. L'Ecuyer, F. Le Gland, P. Lezaud, and B. Tuffin, "Splitting techniques," in *Rare Event Simulation using Monte Carlo Methods*, G. Rubino and B. Tuffin, Eds. John Wiley & Sons, 2009, ch. 3, pp. 39–61.

[15] M. Villén-Altamirano and J. Villén-Altamirano, "Analysis of restart simulation: Theoretical basis and sensitivity study," *European Transactions on Telecommunications*, vol. 13, no. 4, pp. 373–385, 2002. [Online]. Available: https://doi.org/10.1002/ett.4460130409

[16] A. Borodina and V. Tishenko, "On splitting scenarios for effective simulation of reliability models," in *Distributed Computer and Communication Networks: Control, Computation, Communications*, V. M. Vishnevskiy, K. E. Samouylov, and D. V. Kozyrev, Eds. Cham: Springer International Publishing, 2020, pp. 353–366.

[17] R. Vaisman, D. P. Kroese, and I. B. Gertsbakh, "Splitting sequential monte carlo for efficient unreliability estimation of highly reliable networks," *Structural Safety*, vol. 63, pp. 1–10, 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167473016300236

[18] R. Vaisman, "Sequential stratified splitting for efficient monte carlo integration," *Sequential Analysis*, vol. 40, no. 3, pp. 314–335, 2021.

[19] A. Gonzalez, B. Helvik, J. Hellan, and P. Kuusela, "Analysis of dependencies between failures in the uninett ip backbone networks," in *Proceedings of the IEEE 16th Pacific Rim International Symposium on Dependable Computing (PRDC'10)*, 2010, pp. 149–156. [Online]. Available: http://dx.doi.org/10.1109/PRDC.2010.12

[20] J. Barrera, H. Cancela, and E. Moreno, "Topological optimization of reliable networks under dependent failures," *Operations Research Letters*, vol. 43, no. 2, pp. 132 – 136, 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167637714001771

[21] O. Matus, J. Barrera, E. Moreno, and G. Rubino, "On the Marshall–Olkin copula model for network reliability under dependent failures," *IEEE Transactions on Reliability*, vol. 68, no. 2, pp. 451–461, 2019.

[22] B. M. O. Lapcevic, M. Lackovic, "Impact of dependent failures on the availability of the optical network," in *Proceedings of the 6th International Symposium on Communication Systems, Networks and Digital Signal Processing (CNSDSP'08)*, 2008, pp. 423—-427. [Online]. Available: http://dx.doi.org/10.1109/CSNDSP.2008.4610799

[23] A. W. Marshall and I. Olkin, "A multivariate exponential distribution," *Journal of the American Statistical Association*, vol. 62, no. 317, pp. 30–44, 1967. [Online]. Available: https://www.tandfonline.com/doi/abs/10.1080/01621459.1967.10482885

[24] M. Amrein and H. R. Künsch, "A variant of importance splitting for rare event estimation: Fixed number of successes," *ACM Trans. Model. Comput. Simul.*, vol. 21, no. 2, pp. 13:1–13:20, Feb. 2011. [Online]. Available: http://doi.acm.org/10.1145/1899396.1899401

[25] S. Soh and S. Rai, "An efficient cutset approach for evaluating communication-network reliability with heterogeneous link-capacities," *IEEE Transactions on Reliability*, vol. 54, no. 1, pp. 133–144, March 2005.

[26] W. C. Yeh, "An improved sum-of-disjoint-products technique for symbolic multi-state flow network reliability," *IEEE Transactions on Reliability*, vol. 64, no. 4, pp. 1185–1193, Dec 2015.

[27] Z. I. Botev, R. Vaisman, R. Y. Rubinstein, and P. L'Ecuyer, "Reliability of stochastic flow networks with continuous link capacities," in *Proceedings of the 2014 Winter Simulation Conference*, ser. WSC '14, Piscataway, NJ, USA, 2014, pp. 543–552. [Online]. Available: http://dl.acm.org/citation.cfm?id=2693848.2693926

[28] S. Bulteau and M. El Khadiri, "A Monte Carlo simulation of the flow network reliability using importance and stratified sampling," *RAIRO - Operations Research - Recherche Opérationnelle*, vol. 32, no. 3, pp. 271–287, 1998. [Online]. Available: http://eudml.org/doc/105171

[29] H. Cancela, L. Murray, and G. Rubino, "Highly reliable stochastic flow network reliability estimation," in *XLII Latin American Computing Conference, CLEI 2016, Valparaíso, Chile, October 10-14, 2016*, 2016, pp. 1–10. [Online]. Available: https://doi.org/10.1109/CLEI.2016.7833343

[30] G. S. Fishman and T. Y. D. Shaw, "Evaluating reliability of stochastic flow networks," *Probability in the Engineering and Informational Sciences*, vol. 3, pp. 493–509, 10 1989.

[31] I. B. Gertsbakh, R. Y. Rubinstein, Y. Shpungin, and R. Vaisman, "Permutational methods for performance analysis of stochastic flow networks," *Probability in the Engineering and Informational Sciences*, vol. 28, no. 1, pp. 21–38, 2014.

[32] L. R. Ford and D. R. Fulkerson, *Flows in Networks*. Princeton University Press, Princeton, New Jersey, 1962.

[33] J. Edmonds and R. M. Karp, "Theoretical improvements in algorithmic efficiency for network flow problems," *J. ACM*, vol. 19, no. 2, pp. 248–264, Apr. 1972. [Online]. Available: http://doi.acm.org/10.1145/321694.321699

[34] A. Goldberg and S. Rao, "Beyond the flow decomposition barrier," *J. ACM*, vol. 45, no. 5, pp. 783–797, Sep. 1998. [Online]. Available: http://doi.acm.org/10.1145/290179.290181

[35] Z. I. Botev, P. L'Ecuyer, R. Simard, and B. Tuffin, "Static network reliability estimation under the Marshall–Olkin copula," *ACM Trans. Model. Comput. Simul.*, vol. 26, no. 2, pp. 14:1–14:28, Jan. 2016. [Online]. Available: http://doi.acm.org/10.1145/2775106

[36] Y. K. Lin, P. C. Chang, and L. Fiondella, "Quantifying the impact of correlated failures on stochastic flow network reliability," *IEEE Transactions on Reliability*, vol. 61, no. 3, pp. 692–701, Sept 2012.

[37] J. F. Mai and M. Scherer, "Sampling exchangeable and hierarchical Marshall–Olkin distributions," *Communications in Statistics - Theory and Methods*, vol. 42, no. 4, pp. 619–632, 2013. [Online]. Available: https://doi.org/10.1080/03610926.2011.615437

[38] M. J. J. Garvels, "The splitting method in rare event simulation," Ph.D. dissertation, Faculty of mathematical Science, University of Twente, The Netherlands, 2000.

[39] G. Rubino and B. Tuffin, *Rare event simulation using Monte Carlo methods*. Wiley, 2009.

[40] H. Cancela, G. Rubino, and B. Tuffin, "New measures of robustness in rare event simulation," in *Proceedings of the 2005 Winter Simulation Conference*, F. A. M.E. Kuhl, N.M. Steiger and J. Joines, Eds., 2005, pp. 519–527.