Structural Safety 63 (2016) 1–10

Contents lists available at ScienceDirect

Structural Safety

journal homepage: www.elsevier.com/locate/strusafe

Splitting sequential Monte Carlo for efficient unreliability estimation of highly reliable networks

Radislav Vaisman [a,*], Dirk P. Kroese [a], Ilya B. Gertsbakh [b]

[a] School of Mathematics and Physics, The University of Queensland, Australia
[b] Department of Mathematics, Ben-Gurion University, Beer-Sheva 84105, Israel

ARTICLE INFO

Article history:
Received 23 April 2015
Received in revised form 5 July 2016
Accepted 5 July 2016

Keywords:
Terminal network reliability
Permutation Monte Carlo
Multilevel splitting
Rare events

ABSTRACT

Assessing the reliability of complex technological systems such as communication networks, transportation grids, and bridge networks is a difficult task. From a mathematical point of view, the problem of estimating network reliability belongs to the #P complexity class. As a consequence, no analytical solution for solving this problem in a reasonable time is known to exist and one has to rely on approximation techniques. In this paper we focus on a well-known sequential Monte Carlo algorithm — Lomonosov's turnip method. Despite the fact that this method was shown to be efficient under some mild conditions, it is known to be inadequate for a stable estimation of the network reliability in a rare-event setting. To overcome this obstacle, we suggest a quite general combination of sequential Monte Carlo and multilevel splitting. The proposed method is shown to bring a significant variance reduction as compared to the turnip algorithm, is easy to implement and parallelize, and has a proven performance guarantee for certain network topologies.

© 2016 Elsevier Ltd. All rights reserved.

1. Introduction

Nowadays it is hard to underestimate the importance of networks in our life and, as a consequence, a natural question of their reliability arises [1–5]. Many engineering applications, such as computer and transportation networks, water distribution and gas supply systems, can be modelled via a graph structure, whose components (nodes and edges), are subject to failure. Such networks are often used to model a delivery of some resource or commodity, so one will be generally concerned with the reliability of the entire system. Consequentially, we adopt the following definition of the network reliability problem [6]. Let $G = G(V, E, \mathcal{K})$ be an undirected graph, where V and E are the vertex and edge sets, respectively, and $\mathcal{K} \subseteq V$ is a set of "terminal" nodes. We assume that the vertices never fail, but that the edges are subject to failure. In particular, every $e \in E$ has a corresponding failure probability $0 \leqslant q_e \leqslant 1$. An edge can be in an *up* or *down* state with probabilities $p_e = 1 - q_e$ and q_e, respectively. Under the above framework we wish to assess the network unreliability, defined as the probability that the terminal set $\mathcal{K}$ is disconnected [7].

The exact solution to the $\mathcal{K}$-terminal network reliability problem is hard to obtain within reasonable computation time, since this problem belongs to the #P complexity class [8,9]. This complexity class, introduced by Valiant [10], consists of the set of counting problems that are associated with a decision problem in NP (non–deterministic polynomial time). For example, #SAT is the problem of counting the number of feasible solutions to a satisfiability formula (SAT).

For some #P-complete problems there are known efficient approximations. For example, Karp and Luby [11] introduced a fully polynomial randomized approximation scheme (FPRAS) for counting the solutions of satisfiability formulas in disjunctive normal form (DNF). The DNF counting algorithm allows an efficient solution to the $\mathcal{K}$-terminal network reliability problem, provided that the list of $\mathcal{K}$-separating cuts is available [12]; however, the latter is generally expensive to obtain. For the *all-terminal* network reliability case ($\mathcal{K} = V$), an FPRAS was developed by Karger [8]. However, to the best of our knowledge, there exists no FPRAS for estimating the general $\mathcal{K}$-terminal network reliability case. The current state-of-the-art can deal only with specific graph topologies such as series–parallel and directed acyclic networks [13,14], or with small-sized graphs. We refer to [7] for further details.

Due to the problem's importance, various approximation techniques were proposed [7,15–19]. For more recent advances in cut based, matrix-based, and linear programming methods, we refer to [20–23], respectively. In the stochastic simulation area, see the works of Shafieezadeh and Ellingwood [15], the multilevel splitting algorithms of Botev et al. [24,25], Walter [26], the similar

* Corresponding author.
E-mail addresses: r.vaisman@uq.edu.au (R. Vaisman), kroese@maths.uq.edu.au (D.P. Kroese), elyager@bezeqint.net (I.B. Gertsbakh).

http://dx.doi.org/10.1016/j.strusafe.2016.07.001
0167-4730/© 2016 Elsevier Ltd. All rights reserved.

subset simulation approach of Zuev et al. [27], and the sequential Importance Sampling (SIS) method of L'Ecuyer et al. [28]. The latter generates the link states in a sequential manner, while introducing a smart sampling scheme that approximates a corresponding zero-variance importance sampling distribution. In this paper we focus on Lomonosov's turnip (LT) algorithm [17]. This method is an improvement of the Permutation Monte Carlo (PMC) scheme which was shown to be efficient under some mild conditions. In particular, it brings a significant variance reduction as compared to LT, and has a proven performance guarantee for some network topologies.

We give a brief introduction to PMC and LT in Section 2. Despite the fact that PMC and LT are designed to deal with quite hard network instances, it was shown in [24] that these methods can be very inefficient in a rare-event setting. To overcome the rare-event complication, Botev et al. [24] formulated the network reliability problem as a static rare-event probability estimation problem and employed the Generalized Splitting (GS) algorithm [6, Chapter 14].

The multilevel splitting framework was first used by Kahn and Harris [29] to estimate rare-event probabilities. The main idea is to partition the state space in such a way that the problem becomes one of estimating conditional probabilities that are not rare. The GS algorithm of Botev and Kroese [30] generalizes this to a method able to evaluate a wide range of rare-event estimation problems. For a survey of the general methodology we refer to [31–33, Chapter 4].

Inspired by the successful approach of Botev et al. [24], we put the LT method into a sequential Monte Carlo (SMC) framework combined with multilevel splitting [30,32,33]. In particular, we propose to combine the very general splitting idea of Kahn and Harris [29] with the LT procedure. Unlike Botev's GS, we do not reformulate the reliability problem, but rather equip the LT algorithm with the corresponding splitting mechanism, thus exploiting the strengths of both methods. The resulting algorithm introduces a significant variance reduction as compared to the basic LT method and has a proven performance guarantee for some networks. Namely, we prove that our method is an FPRAS for special families of graphs. See Section 3 for details.

The rest the paper is organized as follows. In Section 2 we give a brief introduction to the PMC and LT algorithms and show a simple family of networks for which LT's performance is inefficient. In Section 3 we put LT into a quite general SMC framework combined with multilevel splitting. We show that the resulting algorithm can be used to deliver highly accurate estimators and provide an explanation for its efficiency. In Section 4 we present various numerical examples to demonstrate the advantage of the proposed method. Finally, in Section 5 we summarize our findings and discuss possible directions for future research.

2. Permutation Monte Carlo

Below we describe the PMC algorithm of Michael Lomonosov, also called the network evolution process. This method was designed to estimate the reliability of networks with independent components having different failure probabilities. For detailed explanations, see [17] and [18, Chapter 9].

Our setting is as follows. Given a network $G = G(V, E, \mathcal{K})$ where V is the node set, E is the edge set and $\mathcal{K} \subseteq V$ is the terminal set. The edges states are binary; that is, edge e can be in the *up* or *down* state with probabilities p_e and $q_e = 1 - p_e$, respectively. The network *UP* state is defined as the presence of connectivity of all terminal nodes.

The basic idea of PMC is to associate with each edge $e \in E$ an exponentially distributed random "birth time" $\mathcal{T}(e)$ with parameter $\lambda(e)$, such that $\mathbb{P}(\mathcal{T}(e) \leqslant \tau) = 1 - \mathrm{e}^{-\lambda(e)\tau} = p_e$ holds for all $e \in E$ and for an arbitrary chosen time value τ. Let us assume that all the edges are in the *down* state at time zero. Then, an edge e is born at time $\mathcal{T}(e)$; that is, at the time $\mathcal{T}(e)$ it enters the *up* state and stays there "forever". The probability that e will be "alive" at time τ is $\mathbb{P}(\mathcal{T}(e) \leqslant \tau) = p_e$. The value of τ can be arbitrary, so for simplicity we put $\tau = 1$ and it follows that $\lambda(e) = -\ln q_e$. If we take a "snapshot" of the state of all edges at time instant $\tau = 1$, we will see the network in the state which is stochastically equivalent to the static picture in which edge e is *up* or *down* with probability p_e or q_e, respectively.

Suppose that $|E| = n$ and consider the ordering (permutation) of the edges $\boldsymbol{\pi} = (e_1, \ldots, e_n)$, according to their birth times sorted in increasing order. Since the birth times are exponentially distributed, it holds that

$$\mathbb{P}(\boldsymbol{\Pi} = \boldsymbol{\pi}) = \prod_{t=1}^{n} \frac{\lambda(e_t)}{\Lambda(E_{t-1})}, \tag{1}$$

where $E_t = E \setminus \{e_1, \ldots, e_t\}$ for $1 \leqslant t \leqslant n-1$, and $\Lambda(E_t) = \sum_{e \in E_t} \lambda(e)$ [17,34].

The first index $1 \leqslant \mathrm{a}(\boldsymbol{\pi}) \leqslant n$ of the edge permutation $\boldsymbol{\pi}$ for which the sub-graph of G defined by $G(V, (e_1, \ldots, e_{\mathrm{a}(\boldsymbol{\pi})}), \mathcal{K})$ is in the *UP* state, is called an anchor of $\boldsymbol{\pi}$. That is, $\mathrm{a}(\boldsymbol{\pi}) = \min\{t : G(V, (e_1, \ldots, e_t), \mathcal{K}) \text{ is } UP\}$. Let $\xi_1 + \cdots + \xi_t$ be the birth time of edge e_t in $\boldsymbol{\pi}$ for $1 \leqslant t \leqslant n$. Then, given the edge permutation $\boldsymbol{\Pi} = \boldsymbol{\pi}$, the probability that the network is in the *UP* state is given by

$$\mathbb{P}\left(\sum_{t=1}^{\mathrm{a}(\boldsymbol{\pi})} \xi_t \leqslant 1 \,|\, \boldsymbol{\Pi} = \boldsymbol{\pi}\right) = \mathrm{Conv}_{1 \leqslant t \leqslant \mathrm{a}(\boldsymbol{\pi})}\left\{1 - \mathrm{e}^{-\Lambda(E_t)}\right\},$$

where Conv stands for exponential convolution. The network *DOWN* and *UP* probabilities denoted by $\bar{r}$ and r, respectively, can be expressed as

$$\bar{r} = \sum_{\boldsymbol{\pi}} \mathbb{P}(\boldsymbol{\Pi} = \boldsymbol{\pi}) \cdot \mathbb{P}\left(\sum_{t=1}^{\mathrm{a}(\boldsymbol{\pi})} \xi_t > 1 \,|\, \boldsymbol{\Pi} = \boldsymbol{\pi}\right), \tag{2}$$

and

$$r = \sum_{\boldsymbol{\pi}} \mathbb{P}(\boldsymbol{\Pi} = \boldsymbol{\pi}) \cdot \mathbb{P}\left(\sum_{t=1}^{\mathrm{a}(\boldsymbol{\pi})} \xi_t \leqslant 1 \,|\, \boldsymbol{\Pi} = \boldsymbol{\pi}\right),$$

respectively, where the summation is over all permutations $\boldsymbol{\pi}$. Since the network unreliability and reliability in (2) is expressed as an expectation, it can be estimated without bias as the sample average of conditional probabilities, $\mathbb{P}(\xi_1 + \xi_2 + \cdots + \xi_{\mathrm{a}(\boldsymbol{\Pi})} > 1 | \boldsymbol{\Pi})$ over an independent sample of trajectories $\{\boldsymbol{\Pi}^{(1)}, \boldsymbol{\Pi}^{(2)}, \ldots, \boldsymbol{\Pi}^{(N)}\}$. This procedure is summarized in Algorithm 2.1.

Algorithm 2.1 (*PMC Algorithm For unreliability estimation*). Given a network $G = G(V, E, \mathcal{K})$, edge failure probabilities $(q_e, e \in E)$, and sample size N, execute the following steps.

1. **(Initialization)** Set $S \leftarrow 0$. For each edge $e \in E$, set $\lambda(e) \leftarrow -\ln(q_e)$ and $k \leftarrow 0$.
2. **(Permutation Generation)** Set $k \leftarrow k + 1$ and sample $\boldsymbol{\Pi}^{(k)} = \left(e_1^{(k)}, \ldots, e_n^{(k)}\right)$ using (1).
3. **(Find the Anchor)** Calculate
$$\mathrm{a}\left(\boldsymbol{\Pi}^{(k)}\right) = \min\left\{t : G\left(V, \left(e_1^{(k)}, \ldots, e_t^{(k)}\right), \mathcal{K}\right) \text{ is UP}\right\}.$$
4. **(Calculation of Convolution)** Set:
$$R^{(k)} \leftarrow 1 - \mathrm{Conv}_{1 \leqslant t \leqslant \mathrm{a}\left(\boldsymbol{\Pi}^{(k)}\right)}\left\{1 - \mathrm{e}^{-\Lambda\left(E_t^{(k)}\right)}\right\},$$

Download English Version:

https://daneshyari.com/en/article/307425

Download Persian Version:

https://daneshyari.com/article/307425

Daneshyari.com