Estimating Residual Connectivity for Random Graphs

Rohan Shah, Dirk P. Kroese
School of Mathematics and Physics, The University of Queensland, Australia

June 4, 2015

Abstract

Computation of the probability that a random graph is connected is a challenging problem, so it is natural to turn to approximations such as Monte Carlo methods. We describe sequential importance resampling and splitting algorithms for the estimation of these probabilities. The importance sampling steps of these algorithms involve identifying vertices that must be present in order for the random graph to be connected, and conditioning on the corresponding events. We provide numerical results demonstrating the effectiveness of the proposed algorithm.

Notation

\( G = (V, E) \) Graph with vertex set \( V \) and edge set \( E \).
\( G(\mathcal{V}') \) Subgraph of \( G \) induced by the vertex set \( \mathcal{V}' \).
\( \mathcal{P}(\mathcal{V}) \) Set of subgraphs of \( G \) induced by a vertex subset, or the power set of \( \mathcal{V} \).
\( X \) The set of \textit{up} vertices of a network.
\( D_r \) Vertices of \( G \) which are definitely \textit{up}.
\( P_r \) Vertices of \( G \) which are possibly \textit{up}.
\( X_r, \overline{X_r} \) Random subsets and supersets of \( X \).
\( \mathcal{D}_r \) Possible values of \( D_r \).
\( \mathcal{P}_r \) Possible values of \( \overline{X_r} \).
\( F \) Event that \( X \) is connected.
\( F_r \) Event that \( X \) can still be connected conditional on \( D_r \).


1 Introduction

In its broadest sense, network reliability is the study of the performance characteristics of systems that can be modeled by random graphs. The most common application is the study of communication networks (Cancela et al, 2009), but other applications include electricity networks (Chassin and Posse, 2005; Pagani and Aiello, 2013) and air transport networks (Zanin and Lillo, 2013; Wilkinson et al, 2012; Cardillo et al, 2013). Often such systems are highly reliable, and the problem of estimating failure probabilities for these systems is one of estimating a rare-event probability. The most widely studied network reliability model is the K-terminal network reliability model, where the system is operational if a specified set of vertices is connected; see Gertsbakh and Shpungin (2010) for further details.

We consider the problem of estimating the residual connectedness reliability (Lin and Ting, 2015), defined as follows. Let $G = (V, E)$ be an undirected graph. Vertices of $G$ fail independently with probability $1 - p$, and edges are considered failed if either of the vertices fails. The assumption that all vertices fail with equal probability is a notational convenience; generalization to the case where vertices fail with different probabilities is straightforward. Failed vertices are said to be down and functioning vertices are said to be up. The overall network is said to be UP if the up subgraph induced by the up vertices is connected, and DOWN otherwise. The probability that the system is in the UP state is called the residual connectedness reliability. Important applications include Radio Broadcast Networks and Mobile Ad hoc networks (Royer and Toh, 1999; Tanenbaum, 2002).

In practice exact computation of the residual connectedness reliability (RCR) is found to be difficult. The formal statement is that computation of the RCR is NP-hard (Sutner et al, 1991). That is, computing the RCR is at least as difficult as the hardest problem in the computational complexity class NP (Sipser, 1996). For NP-hard problems we generally turn to approximations such as Monte Carlo methods.

Most established Monte Carlo methods for the related K-terminal network reliability problem, such as approximate zero-variance importance sampling (L’Ecuyer et al, 2011), the merge process (Elperin et al, 1991) and generalized splitting (Botev et al, 2013) cannot easily be adapted to the RCR problem.

\[
I_r(d_r) \quad \text{Indicator function of } \mathbb{P}(F \mid D_r = d_r) > 0.
\]

\[
|Z| \quad \text{The number of elements of set } Z.
\]

\[
B(x, r) \quad \text{Closed ball of radius } r \text{ around } x.
\]

\[
B(x, r) \quad \text{Open ball of radius } r \text{ around } x.
\]
These methods generally rely on the network structure function being monotonically decreasing, which is not the case for RCR. One method that can be adapted is the Recursive Variance Reduction (RVR) method (Canela and Urquhart 2002). Another, specifically designed for the RCR model, is the conditional Monte Carlo method in Shah et al (2014). We propose a novel sequential importance resampling algorithm for the estimation of the RCR, which can be interpreted as a “splitting” algorithm combined with importance sampling.

The **splitting method**, first described in Kahn and Harris (1951), is a Monte Carlo technique for the estimation of probabilities of rare events. If the rare event can be written as the intersection of nested events, then the rare-event probability can be written as a product of conditional probabilities. These probabilities will typically be much larger than the rare-event probability and can therefore be estimated more accurately. The key feature of the splitting method is that sample paths of some random process are replicated or split when they hit some “intermediate level of rareness”. Although the estimators of the conditional probabilities are generally not independent, under the splitting method their product is still an unbiased estimator of the rare-event probability. The splitting method and similar sequential importance resampling algorithms such as stochastic enumeration (Villén-Altamirano and Villén-Altamirano 1991, Au and Beck 2001, Liu et al 2001, Del Moral et al 2006, Rubinstein 2010, Vaisman and Kroese 2014) have found numerous applications including network reliability (Héctor et al 2008), queuing theory (Glasserman et al 1999, Garvels 2000), particle transmission (Kahn and Harris 1951) and air traffic control (Jacquemart and Morio 2013).

The rest of this paper is organized as follows. Section 2 introduces the RCR model and describes two existing Monte Carlo estimation methods for the RCR problem. Section 3 introduces two splitting algorithms for the estimation of the RCR. Section 4 gives an overview of the transfer matrix method, which allows the exact computation of the RCR in some cases. Section 5 describes a simulation study performed to compare the different estimation method and discusses the results. A list of proofs is given in Appendix A.

2 The Residual Connectedness Reliability Model

2.1 Definition

Let $G = (V, E)$ be a connected graph in which the state of a vertex $v \in V$ is a binary random variable. If the binary variable takes value 1 then
\(\nu\) is functioning correctly; otherwise the vertex has failed. For notational convenience we assume that these random variables are all independent and identically distributed. This means that each vertex is in the up state with probability \(p\) and the down state with probability \(1 - p\). The subset of \(\mathcal{V}'\) containing the up vertices is denoted by \(X\). We will identify the set \(X\) with the subgraph \(G(X)\) which has vertex set \(X\) and edge set 
\[
\{\{v_1, v_2\} \in E : v_1, v_2 \in X\}.
\]
Let \(P(\mathcal{V})\) be the set of all subgraphs of \(G\) induced by a vertex subset. Define \(\varphi\) as the binary function on \(P(\mathcal{V})\) which takes value 1 for connected graphs and 0 otherwise. When we write \(\varphi(X)\), this will be 1 if \(G(X)\) is connected and 0 otherwise. We are interested in estimating the residual connectedness reliability

\[
\ell(G, p) = P(X \text{ is connected}) = P(\varphi(X) = 1).
\]

It will be convenient to take some enumeration \(v_1, \ldots, v_{|V|}\) of the vertices \(\mathcal{V}\) and use this as a total ordering. This makes it possible to write statements such as \(v_1 < v_2\), and \(\min\{v_1, v_2\} = v_1\). We also use the notation \([v, \infty) = \{w \in \mathcal{V} \mid w \geq v\}\). Note that the ordering used is completely arbitrary.

### 2.2 Existing Methods

The Recursive Variance Reduction (RVR) method of [Cancela and Urquhart 2002](#) can be adapted to the RCR problem as follows. Let \(U\) be the event that all vertices of \(G\) are up and let \(\ell\) be the probability that the input random graph model is connected. The RVR method begins by writing

\[
\ell = P(\varphi(X) = 1) = \varphi(G) P(U) + P(\varphi(X) = 1 \mid U^c) P(U^c)
\]

\[
= p^{|\mathcal{V}|} + P(\varphi(X) = 1 \mid U^c) \left(1 - p^{|\mathcal{V}|}\right).
\]

The event \(U^c\) is the event that at least one vertex fails. Let \(V_1\) be the first vertex that fails (with respect to the total ordering of vertices). If no vertex fails then we say that \(V_1 = \infty\), but \(U^c\) occurring is equivalent to the condition \(V_1 < \infty\). In this case we have

\[
P(V_1 = v_1 \mid V_1 < \infty) = \frac{p^{v_1-1}(1 - p)}{1 - p^{|\mathcal{V}|}}.
\]

Equation (1) can be rewritten as

\[
\ell = p^{|\mathcal{V}|} + P(\varphi(X) = 1 \mid V_1 < \infty) \left(1 - p^{|\mathcal{V}|}\right).
\]
The problem now is to estimate $P(\varphi(X) = 1 \mid V_1 < \infty)$. But conditional on an observed value of $V_1$, we know that vertex $V_1$ is down, and all vertices smaller (with respect to the total ordering of vertices) than $V_1$ are up. We have therefore fixed the states of some of the vertices. So conditional on $V_1$, we have a random graph where the number of vertices with unknown state is smaller than in the original random graph. Define $\ell^{(1)}(v_1) = P(\varphi(X) = 1 \mid V_1 = v_1)$. Then
\[ \ell = p^{|V'|} + \mathbb{E}[\ell^{(1)}(V_1) \mid V_1 < \infty]\left(1 - p^{|V'|}\right). \quad (2) \]

The RVR method first simulates $V_1 \mid V_1 < \infty$. If $\ell^{(1)}(V_1)$ can be explicitly computed, then an estimator for $\ell$ is
\[ p^{|V'|} + \ell^{(1)}(V_1)\left(1 - p^{|V'|}\right). \]

If $\ell^{(1)}(V_1)$ cannot be explicitly computed then the second down vertex $V_2$ is simulated and the expansion used in (1) is applied recursively to $\ell^{(1)}(V_1)$.

See Cancela and Urquhart (2002) for further details. Another approach is the simple conditional Monte Carlo method suggested in Shah et al (2014). In this method a random order is selected for the vertices. The states of the vertices are then simulated in that order, until the first up vertex is identified, denoted by $\omega$. The connected component for the identified up vertex is then simulated. At this point the states of some vertices are still unknown. The probability that the random graph is connected is the probability that the connected component of $\omega$ is the only connected component. This is the probability that all vertices with unknown state are in fact down, which is a power of $1 - p$. See Shah et al (2014) for further details. We will refer to this method as the conditional Monte Carlo method.

3 Splitting for Residual Connectedness Reliability

We construct a fixed-length Markov chain, denoted by $\{D_r\}_{r=0}^R$. Increasing values of $r$ correspond to increasing information about the random graph $X$. At the last time stage we have complete information about $X$, so in fact $X$ is equal in distribution to $D_R$. In some cases we will be able to determine that $\varphi(X) = 0$ before “time” $R$. We will use this Markov chain to construct a classical splitting algorithm. Note the use of $r$ for the time variable; this is intended to emphasize that our “time” variable corresponds to the radius.
of a certain disk. As in Section 2 we will assume some arbitrary ordering of the vertex set $V'$.

3.1 Notation

We illustrate the ideas in this section using the 5 × 5 grid graph, with $R = 3$ and the value of $X$ shown in Figure 1(a). Recall that the value of $X$ will be the last value of the Markov Chain $\{D_r\}_{r=0}^R$. We assume the vertex ordering shown in Figure 1(b).

We first construct a metric on the set $V'$ of vertices. For vertices $v_i$ and $v_j$, the distance $d(v_i, v_j)$ is defined as one less than the number of vertices in the shortest path in $G$ from $v_i$ to $v_j$. For example, the shortest path from $v_i$ to $v_i$ is $\{v_i\}$, so $d(v_i, v_i) = 0$. If $v_i$ and $v_j$ are connected by an edge then the shortest path is $\{v_i, v_j\}$, so that $d(v_i, v_j) = 1$. As we assumed that $G$ was connected, such a path always exists.

Let $R$ be some positive integer. We now define random sets $X^n_r$, for integers $0 \leq r \leq R$ and $n \geq 1$. Recall that $X$ is the set of up vertices, and define $X^1_r = \min X$. That is, $X^1_r$ is the set containing the smallest up vertex with respect to the partial ordering (which does not depend on $r$). For the value of $X$ given in Figure 1(a), $X^1_r = \{2\}$. For a subset $x$ of vertices, define

$$\text{Next}(x, r) = \min \{x \in X \mid d(x, x) > R - r, x > \max x\}.$$ 

In words, Next $(x, r)$ is the next up vertex, after those in $x$, which is more than distance $R - r$ away from the vertices in $x$. Such a vertex may not exist,
Figure 2: Vertex subsets with $R = 3$ and the value of $X$ shown in Figure 1(a). The red crosshatched vertices are up vertices contained in the specified vertex subset. Solid vertices represent other vertices which will eventually be determined to be up.

in which case $\text{Next} (x, r) = \emptyset$. We now make the inductive definition

$$X_{n+1}^r = X_n^r \cup \text{Next} (X_n^r, r).$$

This means that $X_n^r$ is an increasing sequence of sets of up vertices, each set containing at most $n$ vertices.

Example values of $X_0^1, X_0^2$ and $X_0^3$ are shown in Figures 2(a) – 2(c). As $G$ is a finite graph, at some point this sequence must reach some largest set, and stop increasing. We therefore define $X_r^r = \bigcup_{n=1}^{\infty} X_n^r$. Note that if $r = R$ then in fact $X_R = X$. In words, $X_r$ is the set generated by taking the first up vertex, and then iteratively selecting the next (w.r.t. ordering) up vertex that is at least distance $R - r$ away from those previously selected, until no such vertex can be found. Example values of $X_0^r, X_1^r, X_2^r$ and $X_3^r$ are shown in Figures 2(c) – 2(f). The set of possible values of $X_r$ will be denoted by $\mathcal{X}_r$. We view the generation of $X_r$ as the result of a function $\text{GenerateSubset} (X, R, r)$ defined in Algorithm 1.
Algorithm 1: GenerateSubset \((X, R, r)\)

\[
\text{input: } \text{Set } X \subseteq \mathcal{V}, \text{ maximum radius } R, \text{ radius } r \\
\text{output: } X_r \subseteq \mathcal{V}
\]

1. \(\text{if } X = \emptyset \text{ then} \)
2. \(\text{return } \emptyset \)
3. \(X_r \leftarrow \emptyset \)
4. \(\text{while } \exists s \in X \text{ such that } s > \max X_r \text{ and } d(s, X_r) > R - r \text{ do} \)
5. \(X_r \leftarrow X_r \cup \min \{s \in X \mid s > \max X_r, d(s, X_r) > R - r\} \)
6. \(\text{return } X_r \)

The notation is intended to emphasize that the random sets \(\{X_r\}\) are random subsets of \(X\). The \(\{X_r\}\) are all strongly dependent on \(X\) and on each other. The intermediate random sets \(\{X^n_r\}\) are only needed to construct \(X_r\), and we rarely refer to them past this point.

For every possible value \(x_r\) of \(X_r\), let

\[
U_{x_r}(x_r) = \bigcup_{x \in x_r} \left( B(x, R - r) \cap [x, \infty) \right). \tag{3}\]

The set \(U_{x_r}(X_r)\) represents all vertices of \(G\) that could possibly be up given that \(X_r = x_r\). Define

\[
\overline{X}_r = U_{x_r}(X_r). \tag{4}\]

Note that the random sets \(\{X_r\}\) are random supersets of \(X\) (see Proposition A.1). The value of \(\overline{X}_r\) completely determines the value of \(X_r\) because of the functional relationship in (4). As \(X_r\) and \(\overline{X}_r\) are subsets and supersets of \(X\) we have that

\[
X_r \subseteq X \subseteq \overline{X}_r, \quad \text{for all } 0 \leq r \leq R. \tag{5}\]

Finally, we take partial intersections and unions to define the supersets and subsets

\[
D_r = \bigcup_{t=0}^{r} X_t, \quad \text{and} \quad P_r = \bigcap_{t=0}^{r} \overline{X}_t. \tag{6}\]

The mnemonics \(D\) and \(P\) are chosen because conditional on known \(X_1, \ldots, X_r\), the set \(D_r\) represents the set of vertices that are definitely up, and \(P_r\) represents the set of vertices that are possibly up. The set of possible values of \(D_r\)
will be denoted by $D_r$. Example values of $D_1$ and $D_2$ are shown in Figure 3(a) and 3(b). Note that the value of $D_0$ is always equal to $X_0$, which is shown in Figure 2(c). The value of $D_R$ is always equal to $X$, which is shown in Figure 1(a).

It is clear from (5) that $X_r = x_r$ implies that $x_r \subseteq X \subseteq U_p(x_r)$. However from Proposition A.5 the converse is also true. So

$$P(X_r = x_r) = P(x_r \subseteq X \subseteq U_p(x_r)) = P(x_r \subseteq X, U_p(x_r)^c \subseteq X^c) = p|x_r|^r (1 - p)^{|U_p(x_r)^c|}.$$ 

From (5) and (6) we see that $D_r$ and $P_r$ are subsets and supersets, so that

$$D_r \subseteq X \subseteq P_r, \quad 0 \leq r \leq R. \quad (7)$$

For every $0 \leq s \leq r$ we know that

$$X_s \subseteq D_r \subseteq X \subseteq P_s \subseteq U_p(x_s). \quad (8)$$

Proposition A.5 and (8) imply that knowledge of $D_r$ completely determines the value of $X_s$ for $0 \leq s \leq r$. Specifically, $X_s = \text{GenerateSubset}(D_r, R, s)$. Consequently $D_r$ also completely determines the values of $D_s$ and $P_s$ for $0 \leq s \leq r$.

This makes $\{D_r\}_{r=0}^R$ a Markov chain, as its state at any time completely determines its sample path up until that time. By contrast, $\{X_r\}_{r=0}^R$ is not
a Markov chain. For example, consider Figure 2(c) and 2(d). Information from $X_0$ shows that vertex 14 of $G$ is up, but this is not reflected in $X_1$. So if $X_{14}$ is the binary variable controlling the final state of vertex 14, we have

$$P(X_{14} = 0 \mid X_0 = x_0) = P(X_{14} = 0 \mid X_0 = x_0, X_1 = x_1) = 1,$$

$$P(X_{14} = 0 \mid X_1 = x_1) \neq 1.$$

This means the Markov property fails to hold.

### 3.2 Splitting Algorithm

Let $F$ be the event that $X$ is connected. Let $F_r$ be the event that $X$ can still be connected, conditional on the observed value of $D_r$. More precisely,

$$F = \{ \varphi (X) = 1 \},$$

$$F_r = \{ \exists x \in P (V') \text{ with } D_r \subseteq x \subseteq P_r, \text{ such that } \varphi (x) = 1 \}

= \{ P (F \mid D_r) > 0 \}.$$

Note that the subsets $D_r$ are increasing and the supersets $P_r$ are decreasing, so that $\{ F_r \}_{r=0}^R$ is a decreasing sequence of events, with $F_R = F$.

For every $0 \leq r \leq R$ and $d_r \in D_r$, define $I_r (d_r) = \mathbb{I} \{ P (F \mid D_r = d_r) > 0 \}$. That is, $I_r$ is the indicator function that takes value 1 if $X$ can still be connected given the observed value of $D_r$, and 0 otherwise. Note that we can evaluate $I_r (D_r)$ by performing a depth-first search of $P_r$, started from only one of the vertices of $D_r$. Let $x_{\text{DFS}}$ be the vertex set traversed by the depth first search. Then $x_{\text{DFS}}$ is a connected graph, and if $x_{\text{DFS}}$ contains $D_r$ we have $D_r \subseteq x_{\text{DFS}} \subseteq P_r$. This makes $x_{\text{DFS}}$ a connected graph which has a non-zero probability of occurring, in which case

$$P (F \mid D_r = d_r) > P (X = x_{\text{DFS}} \mid D_r = d_r) > 0.$$

The nested events $\{ F_r \}_{r=0}^R$ allow us to write $P (F)$ as

$$P (F) = P (F_1 \mid F_0) P (F_2 \mid F_1) \cdots P (F_R \mid F_{R-1}).$$

We can therefore estimate $P (F)$ as a product of estimates of conditional probabilities using a splitting algorithm. This leads to the fixed splitting algorithm given in Algorithm 2. For notational simplicity we take the splitting factors to be integers. Removing this restriction is conceptually easy, see Section 10.6 of Kroese et al (2011) for further details.
Algorithm 2: Splitting algorithm for RCR

\textbf{input}: Connected graph $G$, maximum radius $R$, reliability $p$, integer splitting factors $k_0, \ldots, k_{R-1}$, initial number of particles $N$

\textbf{output}: Estimate of RCR

1. $\mathcal{Y} \leftarrow \emptyset$ // Samples that have hit an intermediate level
2. for $i = 1$ to $N$ do
3. sample $D$ from $D_0$
4. if $I_0(D)$ then add $D$ to $\mathcal{Y}$
5. for $r = 1$ to $R$ do
6. $\mathcal{Z} \leftarrow \emptyset$ // Samples that have hit the next level
7. for $d_{r-1} \in \mathcal{Y}$ do
8. for $i = 1$ to $k_{r-1}$ do // Split each particle $k_{r-1}$ times
9. sample $D$ from $(D_r | D_{r-1} = d_{r-1})$
10. if $I_r(D)$ then add $D$ to $\mathcal{Z}$
11. swap $\mathcal{Y}$ and $\mathcal{Z}$
12. return $|\mathcal{Y}| \left( N \prod_{r=0}^{R-1} k_r \right)^{-1}$

3.3 Importance Sampling and Resampling Algorithms

Experimentally we observe that most particles reach event $F_{R-1}$, but then tend not to reach event $F_R = F$. That is, we have decomposed the rare-event probability into a product of $R$ conditional probabilities, where the last conditional probability is very small and the others are relatively large. Typically in splitting we would solve this by adding events between $F_{R-1}$ and $F_R$. But due to the discrete nature of the constructed Markov chain it is not obvious how this can be done. Fortunately, the particles which reach $F_{R-1}$ have a very special structure.

In graph theory a \textit{cut vertex} or \textit{articulation vertex} is a vertex of a graph which, when removed, increases the number of connected components. Now, consider the set $V_{\text{cut}}$ of cut vertices of $\overline{X}_{R-1}$. If $F_{R-1}$ occurs and some $v \in V_{\text{cut}}$ is not contained in $\overline{X}_{R-1}$, then $v$ must be \textit{up} if $\overline{X}_R = X$ is to be connected. For a proof of this statement see Proposition A.6. Note that this only applies at step $R-1$, and not at any other step. Figure 4 gives a graphical representation of this statement. Figure 4(a) shows $P_2$, which is the set of all vertices that are possible in $X$. Figure 4(b) shows the smaller set $D_2$, which is the set of all vertices that are definitely in $X$. Figure 4(c) shows all the cut vertices of $P_2$, and Figure 4(d) shows the single cut vertex of $P_2$ which is not in $D_2$. If this vertex is \textit{down} then $X$ cannot be connected.
Figure 4: Visual representation of the result described in Proposition A.6 with $R = 3$. Figure 4(d) shows the single cut vertex of $P_2$ that is not in $D_2$. This vertex must be up in order for $X$ to be connected.

Figure 5: Biconnected components of the value of $P_2$ shown in Figure 4(a).
Algorithm 3: Sequential importance sampling algorithm for RCR

input: Connected graph $G$, maximum radius $R$, reliability $p$, integer splitting factors $k_0, \ldots, k_{R-1}$, initial number of particles $N$

output: Estimate of RCR

$Y \leftarrow \emptyset$ // Samples that have hit an intermediate level

for $i = 1$ to $N$ do
  sample $D$ from $D_0$
  if $I_0(D)$ then add $D$ to $Y$

for $r = 1$ to $R-1$ do
  $Z \leftarrow \emptyset$ // Samples that have hit the next level
  for $d_{r-1} \in Y$ do
    for $i = 1$ to $k_{r-1}$ do // Split each particle $k_{r-1}$ times
      sample $D$ from $(D_r | D_{r-1} = d_{r-1})$
      if $I_r(D)$ then add $D$ to $Z$
  swap $Y$ and $Z$

$W \leftarrow \emptyset$ // Importance weights

for $d_{R-1} \in Y$ do
  cut $\leftarrow$ cut vertices of $U_{p_{R-1}}(y)$
  for $i = 1$ to $k_{R-1}$ do // Resample each particle $k_{R-1}$ times
    sample $D \sim (D_R | D_{R-1} = d_{R-1}, \text{cut} \subseteq X)$ // Imp. sampling
    if $I_R(D)$ then add $p|_{\text{cut} \subseteq d_{R-1}}$ to $W$; // Add imp. weight

return $\left(N \prod_{r=0}^{R-1} k_r\right)^{-1} \sum_{w \in W} w$ // Return average of imp. weights

Conditioning on these cut vertices all being up leads to the importance resampling algorithm shown given in Algorithm 3. Note that the conditioning step is a type of importance sampling and leads to an algorithm involving weighted particles. The factor of $p|_{\text{cut} \subseteq d_{R-1}}$ in the final estimator is the importance weight.

The cut vertices of a graph can be used to decompose it into biconnected components. A biconnected graph is a graph which has no cut vertices, and therefore cannot be disconnected by the removal of a single vertex. A biconnected component is a maximal biconnected subgraph of some underlying graph. The value of $P_2$ shown in Figure 4(a) has only one biconnected component with more than two vertices. This subgraph is shown in Figure 5(a) and another component is shown in Figure 5(b). Note that the biconnected components are not disjoint and share vertices. Any vertex contained in more than one biconnected component must be a cut vertex.
Proposition A.7 shows that if $F_{R-1}$ occurs, then $X$ is connected if and only if $V_{\text{cut}} \subseteq X$ and the ‘restriction’ of $X$ to every biconnected component of $P_{R-1}$ is a connected graph. But the connectivity of $X$ restricted to one component is independent of the connectivity of $X$ restricted to another component. We can use this independence to estimate $\mathbb{P}(F \mid D_{R-1} = d_{R-1})$ more efficiently. This leads to Algorithm 4 (SIS).

Algorithm 4: Sequential imp. sampling algorithm for RCR (SIS)

**input**: Connected graph $G$, maximum radius $R$, reliability $p$, integer splitting factors $k_0, \ldots, k_{R-1}$, initial number of particles $N$

**output**: Estimate of RCR

1. Identical to lines 1 - 14 of Algorithm 3
2. for $d_{R-1} \in \mathcal{Y}$ do
3. \hspace{1em} cut $\leftarrow$ cut vertices of $U_{P_{R-1}}(d_{R-1})$
4. \hspace{1em} bi $\leftarrow$ biconnected components of $U_{P_{R-1}}(d_{R-1})$
5. \hspace{1em} count$_1 \leftarrow 0, \ldots, \text{count}_{|bi|} \leftarrow 0, \text{resid} \leftarrow 0$
6. \hspace{1em} for $i = 1$ to $\left\lfloor \sqrt{|bi|/k_{R-1}} \right\rfloor$ do // Simulate most copies efficiently
7. \hspace{2em} sample $D$ from $(D_R \mid D_{R-1} = d_{R-1}, \text{cut} \subseteq X)$
8. \hspace{2em} for $j = 1$ to $|bi|$ do
9. \hspace{3em} if $\varphi(D \cap bi_j)$ then \hspace{1em} count$_j \leftarrow \text{count}_j + 1$
10. \hspace{1em} for $i = \left\lfloor \sqrt{|bi|/k_{R-1}} \right\rfloor + 1$ to $k_{R-1}$ do // Simulate remaining copies
11. \hspace{2em} sample $D$ from $(D_R \mid D_{R-1} = d_{R-1}, \text{cut} \subseteq X)$
12. \hspace{2em} if $\varphi(D)$ then \hspace{1em} resid $\leftarrow$ resid + 1
13. \hspace{2em} add $p^{\text{cut}\setminus d_{R-1}} \left( \prod_{j=1}^{|bi|} \text{count}_j + \text{resid} \right)$ to $\mathcal{Y}$
14. return $\left( N \prod_{r=0}^{R-1} k_r \right)^{-1} \sum_{w \in \mathcal{Y}} w$

We can also apply the same conditioning ideas at steps other than $R - 1$. This has the additional complication that not every cut vertex of $X_r$ is in fact required to be up in order for $X$ to be connected. However if $v_{\text{cut}}$ splits $X_r$ into components, at least two of which contain vertices of $X_r$, then $v_{\text{cut}}$ must be up. This property can be checked by performing a depth-first search started at all vertices adjacent to $v_{\text{cut}}$. This much more ‘aggressive’ conditioning leads to sample paths with significantly different weights. It is therefore sensible to substitute resampling for splitting, as this tends to automatically remove sample paths with small weights. These ideas lead to Algorithm 5 (SIR).
Algorithm 5: Sequential imp. resampling algorithm for RCR (SIR)

input: Connected graph \( G \), maximum radius \( R \), reliability \( p \), initial number of particles \( N \)

output: Estimate of RCR

1. \( \mathcal{Y} \leftarrow \emptyset \) // Samples that have hit an intermediate level
2. \( \mathcal{W} \leftarrow \emptyset \) // Importance weights
3. for \( i = 1 \) to \( N \) do
   4. sample \( D \) from \( D_0 \)
   5. if \( I_0(D) \) then add \( D \) to \( \mathcal{Y} \), add 1 to \( \mathcal{W} \)
6. for \( r = 1 \) to \( R \) do
   7. averageWeight \( \leftarrow |\mathcal{W}|^{-1} \sum_{w \in \mathcal{W}} \)
6. for \( r = 1 \) to \( R \) do
6. for \( d_{r-1} \in \mathcal{Z} \) do
   11. cut \( \leftarrow \) cut vertices of \( \text{Up}_r(d_{r-1}) \), \( \text{cond} \leftarrow \emptyset \)
   12. for \( c \in \text{cut} \) do
   13. components \( \leftarrow \) connected components of \( \text{Up}_r(d_{r-1}) \) \( \setminus c \)
   14. \( n \leftarrow |\{\text{comp} \in \text{components} : \text{comp} \cap d_{r-1} \neq \emptyset\}|\)
   15. if \( n \geq 2 \) then add \( c \) to \( \text{cond} \)
6. sample \( D \) from \( (D_r \mid D_{r-1} = z, \text{cond} \subseteq X) \) // Imp. sampling
   16. if \( I_r(D) \) then
   17. add \( D \) to \( \mathcal{Y} \), add \( \text{averageWeight} p^{|\text{cond}|} \) to \( \mathcal{W} \)
19. return \( N^{-1} \sum_{w \in \mathcal{W}} w \) // Return average of importance weights

4 The Transfer Matrix Method

In order to compare the different Monte Carlo methods in Section \( \textbf{5} \) it is useful to use graphs that are reasonably large and for which the RCR is exactly known. Assume that for \( 0 \leq i \leq |\mathcal{V}| \) the number of vertex subsets with \( i \) vertices that induce a connected subgraph is \( c_i \). Then

\[
\mathbb{P} (\varphi(X) = 1) = \sum_{i=0}^{\text{|V|}} \mathbb{P} (\varphi(X) = 1 \mid |X| = i) \binom{|\mathcal{V}|}{i} p^i (1-p)^{|\mathcal{V}| - i} = \sum_{i=0}^{\text{|V|}} c_i \binom{|\mathcal{V}|}{i} p^i (1-p)^{|\mathcal{V}| - i} = \sum_{i=0}^{\text{|V|}} c_i p^i (1-p)^{|\mathcal{V}| - i}.
\]

Computation of the values \( c_i \) has been shown to be \( \#P \) complete (Sutner et al., 1991). However for certain classes of regular graphs the transfer matrix
method from statistical physics (Klein et al, 1986; Kloczkowski and Jernigan, 1998; Clisby and Jensen, 2012) can be used to compute the integers $c_i$ relatively quickly. Once the numbers $c_i$ are known, the RCR can be computed quickly for any value of $p$, to arbitrary accuracy.

As an illustration of the transfer matrix method assume that we are interested only in computing the total number of connected subgraphs of the 5 $\times$ 5 grid graph. The graph can be considered as five vertical ‘slices’ consisting of five vertices each. As we move from left to right the ‘state’ of the graph at a particular point consists of the states of those five vertices and information about the paths that connect them which lie to the left. Six of the 52 possible ‘states’ are shown in Figure 6. In the first example state there are three vertices present, which are all connected together by a path lying to the left. In the fourth example state there are three vertices present, and the first and last are connected by a path lying to the left. The middle vertex is not connected to the other two by a path lying to the left. In the third example state all three vertices are connected by a path which lies to the left, although the lower two vertices are necessarily connected as they are adjacent. These 52 states include two ‘empty’ states where no vertices are present; one which occurs before any ‘state’ which contain a vertex, and one which occurs after a ‘state’ which contains a vertex.

When a connected subgraph is decomposed in this manner there are only a certain number of possibilities for the initial and final states. For example the first state in Figure 6 is not possible as an initial state and state 4 as a final state would result in a disconnected subgraph. Further only a limited number of transitions between states are allowed. The number of connected subgraphs can therefore be written as $x^T B^T y$. Here $x$ is a binary vector containing the value 1 if a state is a possible initial state for a connected subgraph, and $y$ is a binary vector containing the value 1 if a state is a possible final state for a connected graph. The $52 \times 52$ binary matrix $B$ contains the value 1 if a transition between two states is possible for a connected subgraph. This approach can be applied in general to regular planar graphs. By keeping a running count of the number of vertices we can determine the number of connected subgraphs for every total number of vertices. Our implementation made use of the Eigen linear algebra library (Guennebaud, Jacob et al, 2010), the MPFR numeric library (Fousse et al, 2007) and Boost C++ libraries.

5 Numerical Experiments

We performed a simulation study to compare four Monte Carlo (MC) methods for the RCR problem. The four methods were conditional MC (Shah
Figure 6: A sample of the six of the 52 possible states for the transfer matrix, for the $5 \times 5$ grid graph.

et al. (2014), the RVR method of Cancela and Urquhart (2002), the sequential importance sampling algorithm given as Algorithm 4 (SIS) and the sequential importance resampling algorithm given as Algorithm 5 (SIR).

We previously assumed that the reliability of every vertex was equal. Not only is this notationally convenient, but it allows us to compute the exact RCR using the transfer matrix method if $G$ is a grid graph. The largest graph for which we applied the transfer matrix method was the $11 \times 11$ grid graph. Although there are $2^{121}$ possible subgraphs, the transfer matrix method allows the computation of the $c_i$ in around an hour. The largest number of connected subgraphs occurred when $i = 75$, for which the number of connected subgraphs was on the order of $10^{31}$. Note that this is well outside the maximum integer value of $2^{64} - 1$ representable on a computer without more specialized software.

We compared the four methods on square grid graphs with length 8, 9, 10 and 11. The parameter $p$ was varied between 0.1 and 0.65 in steps of 0.05. At larger and smaller values of $p$ the target event was no longer rare and we do not consider these cases interesting. In addition we also applied all methods to the $14 \times 14$ grid graph, for which it is not possible to compute the unreliability exactly.

The methods were compared on the basis of their estimated relative error (RE) and work normalized relative variance (WNRV). For an estimator $\hat{p}$ of a probability $p$ which takes on average time $T$ to compute, these quantities are defined as

$$\text{RE} (\hat{p}) = \sqrt{\frac{\text{Var} (\hat{p})}{p}},$$

and

$$\text{WNRV} (\hat{p}) = \frac{T \text{Var} (\hat{p})}{p^2}.$$ 

In order to estimate the RE and WNRV, each method was run 100 times.
Sample sizes for every method were $10^6$ per run. For SIS the splitting factors are a required input to the algorithm. A separate run with sample size $10^6$ and splitting factors identically 1 was performed in order to estimate the splitting factors. These estimated values were then used for all the subsequent 100 runs. The time required to estimate these splitting factors is not included in the work normalized results.

The simulation study shows that the RVR method is not competitive with the other three methods tested. For the $8 \times 8$ grid graph the average result from the RVR method is numerically accurate for the exactly computed value, although the estimated RE (Figure 7) and WNRV (Figure 8) show that the RVR method is inferior to the other methods. The shading in Figure 7 shows 95% confidence intervals obtained by bias-corrected and accelerated (BCA) bootstrapping (Davison and Hinkley, 1997) using the R package boot (Canty and Ripley, 2014). In the case of the $11 \times 11$ grid graph the RVR method also performs poorly, with the added problem that for $p$ between 0.2 and 0.35 the RVR method is not close to the exact probability (Figure 9).

In every case either SIR or conditional MC has the lowest relative error. Which of these performs best depends on $p$. In general for $p < 0.25$ conditional MC performs best but for $p > 0.25$ SIR performs best. A similar pattern is observed for the work normalized relative variance, although SIS
Figure 8: Work normalized relative variance 8 × 8 grid graph.

Figure 9: The absolute empirical bias as a proportion of the true value, over 100 replications for the 11 × 11 grid graph.
and SIR and conditional MC are somewhat closer in terms of performance. The value of $p = 0.25$ separating these behaviors is believed to be an artifact of our choice of grid values for $p$. The value of 0.25 is the value that is closest to the value $p^*$ of $p$ that solves

$$p = \mathbb{E}[|X| |Y|^{-1} | F].$$

That is, the threshold value $p^*$ is the value of $p$ for which the expected proportion of up vertices under the conditional distribution is exactly $p$. For example, Figure 10 shows the expected proportion of up vertices for the $11 \times 11$ grid graph, with a vertical line at $p^* = 0.2454$. Note that in Figure 9 for $p < p^*$ SIS and SIR do not always converge numerically to the true value using 100 replications, even on a log scale. Similarly, for $p > p^*$ conditional MC does not always converge to the true value using 100 replications.

![Figure 10](image)

Figure 10: Expected proportion of up vertices for the $11 \times 11$ grid graph, under the conditional distribution. The blue line represents $y = x$, the red line represents $p = 0.2454$.

The simulation results suggest that very different strategies must be employed to estimate the RCR, depending on the value of $p$. At some value of $p^*$ (which we assume is unique) the expected fraction of up vertices under the conditional distribution is equal to $p^*$. For values of $p$ slightly smaller than $p^*$ the expected fraction of up vertices is much smaller than $p$, and for values
of \( p \) slightly larger than \( p^* \) the expected fraction of \( up \) vertices is much larger than \( p \). This occurs because for \( p \) smaller than \( p^* \) the ‘easiest’ way to obtain a connected graph is to delete all but one of the connected components obtained under the unconditional measure. However if \( p \) is large then the easiest way to obtain a connected graph is to add extra vertices, connecting up the components generated under the unconditional measure into a single component. SIS and SIR take the approach of adding extra vertices, and are therefore only useful when \( p > p^* \). SIR is more aggressive than SIS in adding vertices, and the difference in performance between the two cases is therefore more pronounced. Conditional MC can be interpreted as ‘deleting’ vertices, and is therefore efficient for \( p < p^* \).

Figure 11 gives the relative errors for the four tested algorithms for the \( 11 \times 11 \) grid graph. The shaded regions indicate 95% confidence intervals for the RE for the SIR and conditional MC methods, obtained using BCA bootstrapping. The relative errors of SIR and conditional MC differ by a factor of up to 100, and which method performs better depends on whether \( p < p^* \) or \( p > p^* \). On a work normalized basis (Figure 12) conditional MC outperforms SIR by a factor of up to \( 10^5 \) for \( p < p^* \), but SIR outperforms conditional MC by a factor of up to \( 10^2 \) for \( p > p^* \).

![Figure 11: Relative error results for the 11×11 grid graph. Shading represents bootstrapped 95% confidence intervals for Conditional and SIR.](image-url)
Figure 12: Work normalized relative variance for the 11 × 11 grid graph.

Figure 13 shows the relative errors of the tested algorithms for the 14 × 14 grid graph. The shaded regions indicate 95% confidence intervals for the RE for the SIR and conditional MC algorithms, obtained using BCA bootstrapping. We do not show results from the RVR method as the estimated relative errors were orders of magnitude higher than those for the conditional, SIR and SIS algorithms. For the purposes of calculating the relative errors we used the estimates given by conditional MC for $p \leq 0.25$ and the estimates given by the SIR method for $p > 0.25$.

Note that for $p < 0.25$ the SIR method appears to have lower relative error. However the relative error estimates are believed to be inaccurate in this case, similar to the situation for the RVR method on the 11 × 11 grid graph (Figure 11). The true RE for the SIR method is believed to be orders of magnitude larger than the RE for conditional MC for $p < 0.25$. The opposite situation occurs for $p \geq 0.25$; the relative error of conditional MC appears to decrease as $p$ increases to 0.35, but this is believed to be because the RE is poorly estimated in this case. The true RE for conditional MC is believed to be orders of magnitude higher than the RE for the SIR method, for $0.3 < p < 0.45$. The relative errors are believed to be well estimated for all three methods for $0.45 \leq p \leq 0.6$, and in this region the relative error of the SIR method is a factor of 100 smaller than the RE for conditional MC. For the cases where the relative errors are poorly estimated, it is believed to
be computationally infeasible to run the relevant algorithm enough times to obtain an accurate estimate.

Figure 13: Relative error results for the $14 \times 14$ grid graph. Shading represents bootstrapped 95% confidence intervals for the SIR and conditional MC algorithms.

Note that the SIR and SIS algorithms are not well suited to the situation where $G$ is a large subset of a complete graph. In that case every vertex is adjacent to every other vertex. So $D_0, \ldots, D_{R-1}$ represent knowledge of only the first up vertex, and $D_R$ represents knowledge of the state of every vertex. In this case the SIR and SIS algorithms we describe are similar to crude MC or the RVR method. Further work is required to find methods that behave well in this situation. Another direction for further work is to find algorithms that efficiently estimate the RCR in the regime where $p$ is small. Although conditional MC works acceptably well, it is somewhat unsophisticated. For the special value $p = p^*$ (which appears to be around 0.25 in our examples) none of the algorithms we tested work well. As shown in Figure 11, the SIS, SIR and conditional MC algorithms all had a relative error close to 1 in that case. This case appears to be particularly difficult as methods based around increasing or decreasing the number of up vertices will not be effective.
6 Acknowledgments

This work was supported by the Australian Research Council Centre of Excellence for Mathematical & Statistical Frontiers, under grant number CE140100049.

A Proofs

Proposition A.1. The random variable $\overline{X}_r$ is a superset of $X$.

Proof. Recall that $\overline{X}_r \subseteq X_r$. Assume that there is a vertex $x'$ of $X$ which is not in $\overline{X}_r$ and therefore not in $X_r$. If $x' > \max X_r$, then we must have $d(x', X_r) > R - r$ to avoid having $x' \in X_r$. So Next $(X_r, r) = x'$. This implies that $x' \in X_r$, which is a contradiction.

Assume that $x' < \max X_r$, and define the set $N = \{ n \in \mathbb{Z}^+ \mid x' > \max X^{n}_r \}$.

There is a maximum element $l$ of this set, implying that $x' > \max X^l_r$ and $x' < \max X^{l+1}_r$. As $x'$ is not in $X_r$ we must have $d(x', X_r) > R - r$. This means that Next $(X^l_r, r) = x'$, in which case we would have $x' \in X_r$, which is a contradiction.

Therefore every vertex of $X$ must be contained in $\overline{X}_r$. $\square$

Proposition A.2. Take any $0 \leq r \leq R$ and $x_r \in \mathcal{Z}_r$. Then for any $x$ so that $x_r \subseteq x \subseteq \text{Up}_r (x_r)$, it holds that $\min x_r = \min x$.

Proof. Clearly $\min x_r = \min \text{Up}_r (x_r)$ by the definition of $\text{Up}_r$ in (3). But $x_r \subseteq x \subseteq \text{Up}_r (x_r)$ implies that $\min x_r \geq \min x \geq \min \text{Up}_r (x_r)$.

So $\min x_r = \min x$. $\square$

Proposition A.3. Assume that $x = \{ x_1, \ldots, x_n \}$ is an ordered sequence of vertices so that $d (x_k, \{ x_1, \ldots, x_{k-1} \} ) > R - r$. Then $\mathcal{Z}_r$ consists exactly of all such sequences $x$.

Proof. It is clear that $\text{GenerateSubset} (x, R, r, \cdot =, x)$ for any such $x$. But the definition of $\text{GenerateSubset}$ in Algorithm 1 will always generate vertices with the stated property. $\square$

Corollary A.4. If $x \in \mathcal{Z}_r$, then the value obtained by removing any number of points is still in $\mathcal{Z}_r$. 

24
**Proposition A.5.** Take any $0 \leq r \leq R$ and $x_r \in \mathcal{R}_r$. Then for any $x$ so that $x_r \subseteq x \subseteq U_{p_r}(x_r)$,

$$\text{GenerateSubset}(x, R, r) = x_r.$$  

**Proof.** Let $y = \text{GenerateSubset}(x, R, r)$ and let $x_r, \ldots, x_{n_y}$ be an enumeration of $x_r$ according to the arbitrary ordering. Similarly let $y^1, \ldots, y^{n_y}$ be an enumeration of the set $y$. From Proposition A.2 we know that $y^1 = x_r$. All the subsequent points in $y$ are more than distance $R - r$ from $y^1 = x_r$. Let $A = (y^1, \infty) = (x^1, \infty)$. Then

$$y \cap A \subseteq x \subseteq U_{p_r}(y \cap A), \quad x_r \cap A \subseteq x \subseteq U_{p_r}(x_r \cap A).$$

Taking the intersection with $A$ removes only the first point of both $y$ and $x_r$, so by Corollary A.4 these are still in $\mathcal{R}_r$. Applying Proposition A.2 again shows that $y^2 = x^2$. Applying this argument inductively shows that $n_y = n_x$ and $y = x_r$.  

**Proposition A.6.** Assume that $d_{R-1}$ is a value of $D_{R-1}$ which implies that $F_{R-1}$ occurs and let $p_{R-1}$ be the associated value for $P_{R-1}$. If $v$ is a cut vertex of $p_{R-1}$ then

$$F \cap \{D_{R-1} = d_{R-1}\} \subseteq \{D_{R-1} = d_{R-1}\} \cap \{v \in X\}.$$  

**Proof.** If $v \in D_{R-1}$ the statement is trivial as $D_{R-1} \subseteq X$. So assume that $v \notin D_{R-1}$.

As $v$ is a cut vertex, it separates $p_{R-1}$ into two non-empty components, denoted by $C_1$ and $C_2$. Assume that component $C_1$ does not contain any vertices of $d_{R-1}$ and let $w$ be a vertex in $C_1$. By the definition of $p_{R-1}$ we must have $w \in x_{R-1}$. This implies that there is a vertex of $x_{R-1}$ contained in $B(w, 1) \subseteq C_1 \cup v$. As we assumed that $v \notin x_{R-1}$, we must have a vertex of $x_{R-1}$ contained in $C_1$.

This implies that $C_1$ contains a vertex of $d_{R-1}$, and by an identical argument so does $C_2$. So there are vertices of $X$ in both $C_1$ and $C_2$, and if $v \notin X$ these vertices will lie in different components of $X$.  

**Proposition A.7.** Assume that $d_{R-1}$ is a value of $D_{R-1}$ for which $F_{R-1}$ occurs and let $p_{R-1}$ be the associated value for $P_{R-1}$. Let $V_{\text{cut}}$ be the set of cut vertices of $p_{R-1}$, and let $B = \{B_i\}_{i=1}^{|B|}$ be the set of all biconnected components of $p_{R-1}$. Define the events

$$A = \{D_{R-1} = d_{R-1}\}, \quad B = \{V_{\text{cut}} \subseteq X\}, \quad C = \bigcap_{i=1}^{|B|} \{\varphi(X \cap B_i) = 1\}.$$  

The notation $X \cap B_i$ refers to the subgraph of $X$ induced by the vertex set $B_i$. Then $F \cap A = A \cap B \cap C$.  

25
Proof. Let $H$ be the graph with vertices $B$, and edges between $B_i$ and $B_j$ if $B_i \cap B_j \neq \emptyset$. Note that as $p_{R-1}$ was connected, $H$ must be too.

Assume now that events $A, B$ and $C$ all occur. Take any vertices $B_i$ and $B_j$ of $H$ which are connected by an edge, and pick arbitrary vertices $v_i \in B_i \cap X$ and $v_j \in B_j \cap X$. Let $c$ be some vertex contained in $B_i \cap B_j$. Note that the event $B$ implies that in fact $B_i \cap B_j \subseteq X$, as all vertices in the intersection are cut vertices.

As $B_i \cap X$ is a connected graph there exists a path $P_i$ in $B_i \cap X$ from $v_i$ to $c$ and another path $P_j$ in $B_j \cap X$ from $c$ to $v_j$. This gives a path in $X$ connecting $v_i$ and $v_j$. The connectivity of $H$ means this argument can be extended to the case where $B_i$ and $B_j$ are not adjacent in $H$. This proves that, given our assumptions, $X$ is connected. We have therefore proved that $A \cap B \cap C \subseteq F \cap A$.

Now assume that $F$ and $A$ occur. Proposition A.6 implies that $F \cap A \subseteq A \cap B$, so event $B$ must occur. Now pick any biconnected component $B_i$ and any two vertices $v_1, v_2 \in B_i \cap X$. As $X$ is connected there exists a simple path between $v_1$ and $v_2$. If there is such a simple path $P$ that leaves $B_i \cap X$ then the subgraph $P \cup B_i$ of $P_{R-1}$ is a biconnected component strictly larger than $B_i$. This would contradict $B_i$ being maximal, so $P$ must stay within $B_i \cap X$, making $B_i \cap X$ connected. Therefore event $C$ must also occur.

This proves that $F \cap A = A \cap B \cap C$. 

References

Au SK, Beck JL (2001) Estimation of small failure probabilities in high dimensions by subset simulation. Probabilistic Engineering Mechanics 16(4):263–277

Botev ZI, L’Ecuyer P, Rubino G, Simard R, Tuffin B (2013) Static network reliability estimation via generalized splitting. INFORMS Journal on Computing 25(1):56–71

Cancela H, Urquhart M (2002) Adapting RVR simulation techniques for residual connectedness network reliability models. IEEE Transactions on Computers 51(4):439–443

Cancela H, El Khadiri M, Rubino G (2009) Rare Event Analysis by Monte Carlo Techniques in Static Models, John Wiley & Sons, Ltd, pp 145–170

Canty A, Ripley BD (2014) boot: Bootstrap R (S-Plus) Functions. R package version 1.3-11

26
Cardillo A, Zanin M, Gmez-Gardees J, Romance M, Garca del Amo A, Boccaletti S (2013) Modeling the multi-layer nature of the european air transport network: Resilience and passengers re-scheduling under random failures. The European Physical Journal Special Topics 215(1):23–33

Chassin DP, Posse C (2005) Evaluating North American electric grid reliability using the Barabási-Albert network model. Physica A: Statistical Mechanics and its Applications 355(24):667 – 677

Clisby N, Jensen I (2012) A new transfer-matrix algorithm for exact enumerations: self-avoiding polygons on the square lattice. Journal of Physics A: Mathematical and Theoretical 45(11)

Davison AC, Hinkley DV (1997) Bootstrap Methods and Their Applications. Cambridge University Press, Cambridge

Del Moral P, Doucet A, Jasra A (2006) Sequential Monte Carlo samplers. Journal of the Royal Statistical Society: Series B (Statistical Methodology) 68(3):411–436

Elperin TI, Gertsbakh I, Lomonosov M (1991) Estimation of network reliability using graph evolution models. IEEE Transactions on Reliability 40(5):572–581

Fousse L, Hanrot G, Lefèvre V, Péllissier P, Zimmermann P (2007) MPFR: A multiple-precision binary floating-point library with correct rounding. ACM Transactions on Mathematical Software 33(2)

Garvels MJJ (2000) The splitting method in rare event simulation. PhD thesis, Universiteit Twente

Gertsbakh I, Shpungin Y (2010) Models of Network Reliability. CRC Press, Boca Raton

Glasserman P, Heidelberger P, Shahabuddin P, Zajic T (1999) Multilevel splitting for estimating rare event probabilities. Operations Research 47(4):585–600

Guennebaud G, Jacob B, et al (2010) Eigen v3. http://eigen.tuxfamily.org

Héctor C, Murray L, Gerado R (2008) Splitting in source-terminal network reliability estimation. In: Proceedings of the 7th International Workshop on Rare Event Simulation (RESIM 2008, France)
Jacquemart D, Morio J (2013) Conflict probability estimation between aircraft with dynamic importance splitting. Safety Science 51(1):94–100

Kahn H, Harris TE (1951) Estimation of particle transmission by random sampling. National Bureau of Standards Applied Mathematics Series 12:27–30

Klein DJ, Hite GE, Schmalz TG (1986) Transfer-matrix method for subgraph enumeration: Applications to polyyrene fusenes. Journal of Computational Chemistry 7(4):443–456

Kloczkowski A, Jernigan RL (1998) Transfer matrix method for enumeration and generation of compact self-avoiding walks. I. Square lattices. The Journal of Chemical Physics 109(12):5134–5146

Kroese DP, Taimre T, Botev ZI (2011) Handbook of Monte Carlo Methods. J. Wiley & Sons, New York

L’Ecuyer P, Rubino G, Saggadi S, Tuffin B (2011) Approximate zero-variance importance sampling for static network reliability estimation. IEEE Transactions on Reliability 60(3):590–604

Lin MS, Ting CC (2015) A polynomial-time algorithm for computing K-terminal residual reliability of d-trapezoid graphs. Information Processing Letters 115(2):371 – 376

Liu JS, Chen R, Logvinenko T (2001) A theoretical framework for sequential importance sampling with resampling. In: Doucet A, de Freitas N, Gordon N (eds) Sequential Monte Carlo Methods in Practice, Statistics for Engineering and Information Science, Springer New York, pp 225–246

Pagani GA, Aiello M (2013) The power grid as a complex network: A survey. Physica A: Statistical Mechanics and its Applications 392(11):2688 – 2700

Royer E, Toh CK (1999) A review of current routing protocols for ad hoc mobile wireless networks. Personal Communications, IEEE 6(2):46–55

Rubinstein R (2010) Randomized algorithms with splitting: Why the classic randomized algorithms do not work and how to make them work. Methodology and Computing in Applied Probability 12(1):1–50

Shah R, Hirsch C, Kroese DP, Schmidt V (2014) Rare event probability estimation for connectivity of large random graphs. In: Tolks A, Diallo SD, Ryzhov IO, Yilmaz L, Buckley S, Miller JA (eds) Proceedings of the
Sipser M (1996) Introduction to the Theory of Computation, 1st edn. International Thomson Publishing

Sutner K, Satyanarayana A, Suffel C (1991) The complexity of the residual node connectedness reliability problem. SIAM Journal on Computing 20(1):149–155

Tanenbaum A (2002) Computer Networks, 4th edn. Prentice Hall Professional Technical Reference

Vaisman R, Kroese DP (2014) Stochastic enumeration method for counting trees, submitted

Villén-Altamirano M, Villén-Altamirano J (1991) Restart: A method for accelerating rare events simulations. In: Proceedings of the 13th International Teletraffic Congress, North-Holland, pp 71–76

Wilkinson S, Dunn S, Ma S (2012) The vulnerability of the European air traffic network to spatial hazards. Natural Hazards 60(3):1027–1036

Zanin M, Lillo F (2013) Modelling the air transport with complex networks: A short review. The European Physical Journal Special Topics 215(1):5–21