Assessing Attack Vulnerability in Networks with Uncertainty

Thang N. Dinh
Dept. of CS, Virginia Commonwealth University
Richmond, VA, USA, 23284. Email: tndinh@vcu.edu

My T. Thai
CISE Dept., University of Florida
Gainesville, FL, USA, 32611, Email:mythai@cise.ufl.edu

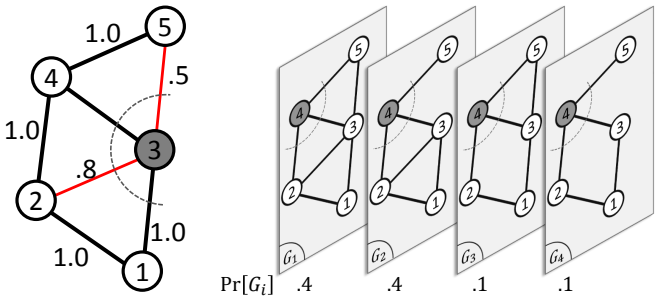
***Abstract*—A considerable amount of research effort has focused on developing metrics and approaches to assess network vulnerability. However, most of them neglect the network uncertainty arisen due to various reasons such as mobility and dynamics of the network, or noise introduced in data collection process. To this end, we introduce a framework to assess vulnerability of networks with uncertainty, modeling such networks as *probabilistic graphs*. We adopt *expected* pairwise connectivity (EPC) as a measure to quantify global connectivity and use it to formulate vulnerability assessment as a stochastic optimization problem. The objective is to identify a few number of critical nodes whose removal minimizes EPC in the residual network. While solutions for stochastic optimization problems are often limited to small networks, we present a practical solution that works for larger networks. The key advantages of our solution include 1) the application of a weighted averaging technique that avoids considering all, exponentially many, possible realizations of probabilistic graphs and 2) a *Fully Polynomial Time Randomized Approximation Scheme* (FPRAS) to efficiently estimate the EPC with any desired accuracy. Extensive experiments demonstrate significant improvement on performance of our solution over other heuristic approaches.**

I. Introduction

Networked systems such as communication networks, electrical grids, and transportation networks are vulnerable to natural disasters and targeted attacks. Even failures of few vital nodes or links can severely compromise the network's ability to meet its quality-of-service (QoS), if not cause total network breakdown [1]. Moreover, there is an increasing concern over such critical systems as targets for (cyber) terrorist attacks [2]. To develop proactive responses and mitigate the risk, it is important to assess network vulnerability, i.e., to identify those crucial nodes and links, beforehand.

Despite of many studies on assessing network vulnerability, little is known about assessment of networks with uncertainty. This uncertainty can arise due to various reasons from mobility and dynamics of networks to data collection process. Particularly, links in technological networks, e. g., the Internet, wireless sensor networks and mobile opportunistic networks, are frequently subject to disruptions. Typical abstraction of networks as static graphs [3], [4], that fails to capture this uncertainty, may lead to serious misjudgement on network vulnerability.

In this paper, we propose a framework to assess vulnerability of networks with uncertainty. We model the network as a *probabilistic graph* $\mathcal{G}$ and formulate the vulnerability assessment as a stochastic optimization problem. The goal is to identify a small set of nodes that removal minimizes *expected value* of network performance. We associate each edge in $\mathcal{G}$ with a probability of existence, representing the fraction of time that the link is in a working state. Additionally, we treat $\mathcal{G}$ as a generative model for deterministic graphs. Each such deterministic graph is a *possible realization/sample* of $\mathcal{G}$ and is also associated with a probability of being generated.



(a) A probabilistic graph (b) Realizations

Fig. 1: A probabilistic graph with existing probabilities on edges. In (a), ad hoc heuristics which target nodes of highest centrality, e.g., degree and betweeness, remove node 3, leaving the residual network intact. In (b), minimizing EPC leads to the removal of node 4. It results in 50% chance of breaking the residual network (in G_2 and G_4) and disrupts effectively 55% network connectivity.

Our basic measure for network performance is pairwise connectivity, defined as the number of connected node pairs in the network. This measure has been recently adopted to account for the impact of attacks in deterministic graphs [3], [4], [5], [6]. It is favored for the strong discrimination in quantifying the network connectivity level, even for disconnected networks. Given a probabilistic graph $\mathcal{G}$, we define k-pCND as the problem of finding k nodes that removal minimizes the *expected* pairwise connectivity (EPC), over all possible realizations of $\mathcal{G}$.

The advantage of our assessment framework over existing approaches is illustrated in Fig. 1. If one node is to remove from the graph according to either degree centrality or betweeness centrality, node 3 will be removed. As shown in Fig. 1a, the residual network remains connected in this case. However, Fig. 1b shows a more destructive attack on node 4 through minimizing EPC. Since most links in the network have existing probabilities one, except for two links $(2, 3)$ and $(3, 5)$, we have four possible realizations of the network, named from G_1 to G_4. Removing node 4 will not only degrade EPC by

55% but also result in a 50% chance of breaking the residual network, as in the cases of G_2 and G_4. More sophisticated assessment methods [4], [5] also fail on this simple example, indicating the importance of considering network uncertainty.

Another advantage of our approach is that we are able to construct an efficient *Fully Polynomial-Time Randomized Approximation Scheme* (FPRAS) to compute EPC. Our FPRAS does not consider all, exponentially many, possible realizations of $\mathcal{G}$, yet it computes EPC with guaranteed accuracy. As exact computation of EPC is #P-complete [7], an FPRAS provides the best theoretical result. Such a result is not known for any other measures for probabilistic graphs except all-terminal-reliability which admits the only known FPRAS in 1995 [8]. Unlike the first FPRAS [8] which is only practical for small networks with equal edge probabilities, our FPRAS is scalable for large networks with heterogeneous edge probabilities.

Last but not least, stochastic optimization problems are extremely difficult to solve. Common techniques for stochastic programming problems such as Bender's decomposition [9] and Sample Average Approximation [10] often do not scale beyond networks with few dozens of nodes. Thus it is critical that we design an efficient solution for the problem.

We summarize our contributions as follows:

- We introduce a *framework to assess attack vulnerability in networks with uncertainty*, formulating it as stochastic optimization problems over probabilistic graphs. Besides EPC, the framework can be integrated with many other reliability and performance measures such as size of largest components and average maximum flow between node pairs [1].
- We formulate k-pCND problem to assess network vulnerability and present a practical solution for the problem which utilizes a *weighted averaging technique* to avoid considering all, exponentially many, possible realizations of probabilistic graphs.
- We propose an FPRAS for computing EPC. The FPRAS is not only of theoretical interest but also practical for large networks. Extending techniques in our FPRAS to other reliability problems, e.g., the two-terminal-reliability and k-terminal-reliability [8], is material for future work.
- We show significant performance improvement of our algorithm over competitor heuristics via experiments. The experiments also reveal that the vulnerability assessment based on deterministic network analysis is too optimistic in real scenarios, as it greatly overestimates how resilient network systems are.

Organization. We summarize related work in Section II and introduce models, notations, and definition of the k-pCND problem in Section III. Assuming the presence of an efficient oracle to estimate EPC, we present our algorithm for k-pCND in Section IV. We present our FPRAS to estimate EPC, the final piece in our solution, in Section V. We give simulation results on real-world traces to show efficacy of our algorithm and the insights in Section VI.

II. Related Work

To our best knowledge, we are the first to study attack vulnerability assessment for uncertain networks, optimizing directly a reliability measure designed for such networks. However, attack vulnerability in deterministic networks and reliability (in the context of random failures) for networks with uncertainty have motivated many studies in communication and theoretical communities.

A. Vulnerability Assessment in Deterministic Networks.

The function and performance of networks rely on their resilience, defined as the ability to continue functioning under perturbation. To measure robustness and resilient, prior works proposed to monitor different measures including the diameter, size of the largest connected component, [11], connectivity based on minimum node/edge cut, algebraic connectivity, and spectral radius [12], [3], [5]. All these measures capture, in principle, aspects of network *connectivity*.

Many metrics and approaches have been proposed to account for network robustness and vulnerability [12], [3], [6]. While each of these measures has its own emphasis and rationality, they often come with several shortcomings that prevent them from capturing desired characteristics of network connectivity and resilience. For example, measures based on shortest path are rather sensitive to small changes (e.g. removing edges or nodes); algebraic connectivity and diameter are not meaningful for disconnected graphs (all disconnected graphs have the same values); number of connected components and component sizes, arguably, do not fully reflect level of network connectivity.

Pairwise connectivity, defined as the number of node pairs that remain connected, has been proposed as an effective measure for network connectivity [13], [4], [5], [14], [6]. Arulselvan et al. defined in [4] the Critical Node Detection (CND) problem, the deterministic version of k-pCND. The problem seeks k nodes that removal minimize the pairwise connectivity in the residual network. We proposed β-disruptor framework to assess attack vulnerability in terms of pairwise connectivity [5], [14]. We presented $O\left(\log^{1.5} n\right)$ bicriteria approximation algorithms for assessing edge vulnerability, and an $O\left(\log n \log\log n\right)$ bicriteria approximation algorithm for the vertex version of β-disruptor. When both nodes and links in the network are subjected to attacks, we provide an $O(\sqrt{\log n})$ bicriteria approximation algorithm [15] that immediately improve the results in [5], [14].

B. Reliability of Networks with Random Failures.

A significant amount of works has been devoted for reliability of networks when their elements are subjected to random failures [16], [7], [17]. The well studied reliability assessment framework is to calculate the probability that communication can be established among a set of nodes when each node and/or link can fail independently with some probability. The two-terminal-reliability, with two special nodes called source s and destination t, concerns the probability that there exists a path between s and t. More general cases involve k-terminal reliability and all-terminal-reliability. The network reliability

problems were proved to be in #P-hard class, a super class of NP-hard problems [18]. Karger introduced the first FPRAS for all-terminal-reliability problem [8]. To our best knowledge, it is the only known FPRAS for network reliability problems.

In [7], [19], Colbourn introduced network resilience, defined as the average two-terminal-reliability between all node pairs. The exact computation of network resilience was proved to be #P-hard even in planar graphs. Amin et al. [20] proposed pair-connected reliability, the expected number of connected node pairs. We note that our measure EPC is the same as Pair-connected reliability and both can be obtained by multiplying network resilience by $\binom{|V|}{2}$. We use the name expected pairwise connectivity to be consistent with naming of pairwise connectivity in [3], [4], [5]. Recently, Neumayer et al. [21] proposed a polynomial-time algorithm to compute network resilience, which they referred to as average two-terminal reliability (ATTR) [21], [22], for geometric networks when the disaster takes a special form of a straight line. However, efficient methods to compute network resilience in general case is still an open question til this paper. *We note that the reliability literature, however, does not consider targeted attacks, which is the main subject of this paper.*

III. Model and Definitions

In this section, we present the probabilistic network model, and the necessary notations to formulate the vulnerability assessment problem.

A. Probabilistic Network Model

We abstract a network with uncertainty as a probabilistic graph $\mathcal{G} = (V, E, P)$ where vertices in V corresponds to the set of nodes; edges in E corresponds to the set of links in the network; and P that maps each edge $(u, v) \in E$ to a real number in $p_{uv} \in [0, 1]$ that represents the probability that edge (u, v) exists. For each $(u, v) \notin E$, we have $p_{uv} = 0$. An example of probabilistic graphs is Erdos-Renyi random graphs [23] in which all edge probabilities are the same and equal p.

For clarity, we consider only undirected networks and assume independence among edges. However our proposed solution also applies in principle to directed graphs or graphs with edge correlations as long as expected values of edges can be computed.

A probabilistic graph $\mathcal{G}$ can be seen as a generative model for deterministic graphs. A deterministic graph $G = (V, E_s)$ is generated from $\mathcal{G}$ by selecting each edge $(u, v) \in E$, independently, with probability p_{uv}. We refer to G as a *realization* or a *sample* of $\mathcal{G}$ and write $G \sqsubseteq \mathcal{G}$. The probability that G is generated from $\mathcal{G}$ is

$$\mathsf{Pr}[G] = \prod_{e \in E_s} p_e \prod_{e \in E \setminus E_s} (1 - p_e).$$

Let $m = |E|$, there are $W = 2^m$ possible realizations of $\mathcal{G}$. We number those realizations as $G^1 = (V, E^1), G^2 = (V, E^2), \ldots, G^W = (V, E^W)$, where $E^1, E^2, \ldots, E^W$ are all possible subsets of E.

B. Expected pairwise connectivity

Our main measure for the network reliability is *expected pairwise connectivity* (EPC), which is the expected number of connected pairs in the network. Formally, denote by $\mathcal{P}(G)$ the number of connected pairs or *pairwise connectivity* of a deterministic graph G. Then *expected pairwise connectivity* (EPC) of $\mathcal{G}$ is defined as

$$\mathsf{EPC}(\mathcal{G}) = \mathbb{E}\left[\mathcal{P}(\mathcal{G})\right] = \sum_{G \sqsubseteq \mathcal{G}} \mathsf{Pr}[G]\mathcal{P}(G).$$

EPC has a tight connection to two-terminal reliability as stated in the following lemma.

Lemma 1: [7] Let $\mathsf{REL}_{u,v}(\mathcal{G})$ denote the two-terminal-reliability between node u and v in $\mathcal{G}$, i.e. the probability that v is reachable from u. We have

$$\mathsf{EPC}(\mathcal{G}) = \frac{1}{2} \sum_{u,v \in V; u \neq v} \mathsf{REL}_{u,v}(\mathcal{G}). \tag{1}$$

Thus EPC can be computed as the total of two-terminal-reliability between all node pairs. However, this approach is problematic due to the facts that exact computation for two-terminal-reliability is NP-hard, and that even we apply existing heuristics to approximate two-terminal-reliability, computing EPC requires a large number, $\binom{|V|}{2}$, of calls to such heuristics.

Instead, EPC can be computed efficiently as shown in Section V. This is an important advantage of EPC over other reliability measures and the reason for the adoption of EPC.

C. Vulnerability Assessment in Probabilistic Networks

We formulate vulnerability assessment as the following stochastic optimization problem.

Probabilistic Critical Nodes Detection (k-pCND). Given a probabilistic network $\mathcal{G} = (V, E, p)$ and an integer $0 \leq k \leq n$, find a k nodes subset $S \subset V$ that removal minimizes EPC in the residual network.

When all edge probabilities are one, we obtain the CND problem in [24]. Since the CND problem is NP-hard, k-pCND, generalizing CND, is also NP-hard.

IV. Vulnerability Assessment in Probabilistic Networks

In this section, we investigate the Probabilistic Critical Nodes Problem (k-pCND). We formulate the problem as a two-stage stochastic program in Subsection IV-A; and devise efficient approaches to overcome the difficulty of having an exponential number of constraints in the mathematical formulation in Subsection IV-B. Our solution assume the presence of an efficient oracle to compute EPC, which we present later in Section V.

A. Two-stage Stochastic Linear Program

Stochastic programming has been a common approach for optimization under uncertainty when the probability distribution governs the data is given. A comprehensive introduction to stochastic programming can be found in reference [25].

Given a probabilistic graph $\mathcal{G} = (V, E, p)$ and an integer $0 < k < n$, we first use integer variables s_i to represent whether or not node i is removed, i.e, $s_i = 1$ if node i is

removed, and 0, otherwise. Here $n = |V|$ is the number of nodes and we assume nodes are numbered from 1 to n. We impose on s the constraint $\sum_{i=1}^{n} s_i \leq k$ to guarantee no more than k nodes are removed. Variables s are known as first stage variables. The values of s are to be decided before the actual realization of the uncertain parameters in $\mathcal{G}$.

We associate with each node pair (u, v) a random Bernouli variable ξ_{uv} satisfying $\mathsf{Pr}[\xi_{uv} = 1] = p_{uv}$ and $\mathsf{Pr}[\xi_{uv} = 0] = 1 - p_{uv}$. For each realization of $\mathcal{G}$, the values of ξ_{uv} are revealed to be either 0 or 1 and we can compute the pairwise connectivity in the residual graph after removing k nodes indicated by s. To do so, we define integer variables x_{ij} to be the "disconnectivity" between a node pair i and j in the residual network, i.e., $x_{ij} = 1$ if i and j are still connected and 0, otherwise. Pairwise connectivity in the residual graph can be computed using a second stage integer programming, denoted by $P(s, x, \xi)$ as follows.

$$P(s, x, \xi) = \min \sum_{i<j} (1 - x_{ij}) \tag{2}$$

$$\text{s. t. } x_{ij} \leq s_i + s_j + 1 - \xi_{\mathbf{ij}},\ (i, j) \in E, \tag{3}$$

$$x_{ij} + x_{jk} \geq x_{ik}, (i, j) \in E, k = 1..n \tag{4}$$

$$s_i \in \{0, 1\}, x_{ij} \in \{0, 1\} \tag{5}$$

This second stage programming formulation is essentially the same with the formulation for the CND problem in [4] (except for the cardinality constraint on s). Indeed, we adopt the improved formulation of CND in [14]. This formulation reduces the number of constraints from $\theta(n^3)$ to $O(mn)$ and shorten the solving time substantially.

The two-stage stochastic linear formulation for the k-pCND problem is as follows.

$$\min_{s \in \{0,1\}^n} \quad \mathbb{E}\left[P(s, x, \xi)\right] \tag{6}$$

$$\text{s. t. } \sum_{i=1}^{n} s_i \leq k \tag{7}$$

$$\text{where } P(s, x, \xi) \text{ is given in (2)-(5)} \tag{8}$$

The objective is to minimize the expected connectivity in the residual network $\mathbb{E}\left[P(s, x, \xi)\right]$, where $P(s, x, \xi)$ is the optimal value of the second-stage problem. This stochastic programming problem is, however, not yet ready to be solved with linear algebra solver.

Discretization. To solve a two-stage stochastic problem, one often need to discretize the problem into a single (very large) linear programming problem. That is we need to consider all possible realizations $G^l \sqsubseteq \mathcal{G}$ and their probability masses $\mathsf{Pr}[G^l]$. Denote by $\{\xi^l\}_{ij}$ the adjacency matrix of the realization $G^l = (V, E^l)$, i.e., $\xi^l_{ij} = 1$, if $(i, j) \in E^l$, and 0, otherwise. Since the objective involves only the *expected cost* of the *second stage* variables x_{ij}, the two-stage stochastic program can be discretized into a mixed integer programming, denoted by MIP_F as follows.

$$\min \sum_{l=1}^{W} \mathsf{Pr}[G^l] \sum_{i<j} (1 - x^l_{ij}) \tag{9}$$

$$\text{s. t. } \sum_{i=1}^{n} s_i \leq k \tag{10}$$

$$x^l_{ij} \leq s_i + s_j + 1 - \xi^l_{ij}, (i, j) \in E, l = 1..W \tag{11}$$

$$x^l_{ij} + x^l_{jk} \geq x^l_{ik}, \quad (i, j) \in E, k = 1..n, l = 1..W \tag{12}$$

$$x^l_{ij} = x^l_{ji}, \quad i, j = 1..n, l = 1..W \tag{13}$$

$$s \in \{0, 1\}^n, x^l \in [0, 1]^{n^2},\ l = 1..W \tag{14}$$

The major challenge in solving this discretized form is that there is an exponential number of variables and constraints. Thus, solving MIP_F is intractable even for very small instances of $\mathcal{G}$. To overcome this difficulty, we present in next subsection a compact relaxation of MIP_F. Solving this polynomial size relaxation leads to high quality solutions for k-pCND problem, as we will show in the experimental section.

B. Algorithm

We present our solution, named REGA, for the stochastic optimization problem in Algorithm 1. First, the algorithm constructs a linear relaxation of the exponential size formula MIP_F and select k vertices via a iterative rounding procedure. The result is a subset D of cardinality k. The algorithm follows by a local search procedure that improves D via swapping vertices. A vertex $u \in D$ and a vertex $v \notin D$ are swapped places if doing so reduces the EPC. The key of the local search is to compute EPC quickly and accurately. This is done with the CSP algorithm (presented later in Algorithm 2). The local search stops when no more swaps can reduce the EPC.

The relaxation of MIP_F is constructed by applying a weighted-averaging of all constraints in MIP_F. Constraints involving the realization G^l are given weights $\mathsf{Pr}[G^l]$. Thus constraints (11) are reduced to a single constraint

$$x_{ij} \leq s_i + s_j + 1 - \sum_{G^l \sqsubseteq G} \mathsf{Pr}[G^l] \xi^l_{ij},$$

which can be further simplified into $x_{ij} \leq s_i + s_j + 1 - p_{ij}$. The other constraints can be "averaged" in the same way, giving us the following relaxation of MIP_F.

$$\min \quad \sum_{i<j} (1 - x_{ij}) \tag{15}$$

$$\text{s. t. } \quad \sum_{i=1}^{n} s_i \leq k \tag{16}$$

$$x_{ij} \leq s_i + s_j + 1 - p_{ij}, \quad (i, j) \in E \tag{17}$$

$$x_{ij} + x_{jk} \geq x_{ik}, (i, j) \in E, k = 1..n \tag{18}$$

$$s_i \in \{0, 1\}, x_{ij} \in [0, 1], \quad i, j = 1..n \tag{19}$$

We shall refer to this relaxation of MIP_F as MIP_R.

Note that the non-integrality of x_{ij} is essential for the above relaxation, denoted by MIP_R. If we restrict x_{ij} to $\{0, 1\}$ the constraint $x_{ij} \leq s_i + s_j - \overline{\xi}_{\mathbf{ij}}$ is equivalent to $x_{ij} \leq s_i + s_j + 1$.

Then the constraint holds trivially for any values of x and s. Thus the information encoded in the edge probabilities is not integerated in the formulation.

Algorithm 1. Rounding the Expected Graph Algorithm (REGA)

1) Obtain an LP relaxation of MIP$_R$ with the relaxed constraints $s \in [0,1]^n$.
2) Initialize the set of selected nodes $D = \emptyset$.
3) Repeat k times the following steps
   - Solve the LP relaxation
   - Select $u = \arg\max_{i \in V \setminus D} s_i$.
   - Add u to D and set $s_u \leftarrow 1$
4) Repeat
5) For each pair $(u,v) \in D \times (V \setminus D)$
6) Estimate the EPC after removing $D - \{u\} + \{v\}$ using CSP
7) Update $D = D - \{u\} + \{v\}$, if the new EPC is lower
8) Until no possible update
9) Output D.

Lower-bound. One of the nice feature of MIP$_R$ is that its optimal objective provides a *lower-bound* on the optimal objective of MIP$_F$. This provides a useful tool to assess the quality of proposed algorithms, especially when finding the optimal solutions of MIP$_F$ is likely intractable. We prove the objective lower-bound in the following lemma.

Lemma 2: The optimal objective value of the MIP$_R$ is a lower-bound on the optimal objective value of MIP$_F$.

Proof: To show that the the objective of the MIP$_R$ is a lower-bound on that of the MIP$_F$, we construct a feasible solution $(\tilde{s}, \tilde{x})$ of MIP$_R$ that gives an objective equal to the optimal objective of MIP$_F$.

Let $\left(\hat{s}, \hat{x}^1, \ldots, \hat{x}^W\right)$ be an optimal solution of the MIP$_F$. Construct a solution $\left(\tilde{s} = \hat{s}, \tilde{x} = \sum_{l=1}^{W} \mathsf{Pr}[G^l]\hat{x}^l\right)$. The objective value of MIP$_R$ given by that solution is

$$\begin{aligned}\sum_{i<j}(1-\tilde{x}_{ij}) &= \sum_{i<j}(1-\sum_{l=1}^{W}\mathsf{Pr}[G^l]\hat{x}^l_{ij}) \\ &= \sum_{i<j}\sum_{l=1}^{W}\mathsf{Pr}[G^l](1-\hat{x}^l_{ij})\end{aligned}$$

which is exactly the optimal objective of MIP$_F$. The last equality holds because the probabilities $\mathsf{Pr}[G^l]$ add up to one.

The rest is to show that $(\tilde{s}, \tilde{x})$ is a feasible solution of MIP$_R$. Clearly, $\tilde{s}$ satisfy (16) and the integral constraints. Also since $\tilde{x}$ is a convex combination of $\hat{x}^l, l = 1..W$ with the masses $\mathsf{Pr}[G^l]$, $\tilde{x}$ satisfy the constraints (18), (17),& (19) as they can be inferred from the same convex combination of the constraints from (11) to (14). ∎

We note that due to the high similarity in programming formulations of critical elements detection problems, our REGA algorithm can be easily modified to solve extensions of other vulnerability assessment problems to networks with uncertainty. Examples include the Critical Edge Detection [26], β-vertex disruptor, and β-edge disruptor [5], [26].

V. Computing EPC

This section focuses on efficient methods to compute EPC of probabilistic graph, the final but important piece of the REGA algorithm. Since it is intractable to compute the exact value of EPC [7], we present efficient methods to approximate EPC with any desired accuracy.

A. Component Sampling Procedure to Approximate EPC

We develop a Monte Carlo method to approximate the EPC within an arbitrary small error with a high probability. We also reveal why the naive Monte Carlo method cannot guarantee a polynomial time complexity.

Given a pair of $\epsilon, \delta > 0$, our Monte Carlo method returns an estimation of $\mathsf{EPC}(\mathcal{G})$ accurate to within a relative error of ϵ with a probability at least $1-\delta$. Mathematically, our proposed method is an (ϵ, δ)-approximation of EPC, which is defined as follows.

Definition 1 ((ϵ, δ)-approximation): A function $\hat{F}(G)$ is an (ϵ, δ)-approximation for the expected pairwise connectivity $\mathsf{EPC}(\mathcal{G})$ if

$$\mathsf{Pr}\left[(1-\epsilon)\mathsf{EPC}(\mathcal{G}) \leq \hat{F}(G) \leq (1+\epsilon)\mathsf{EPC}(\mathcal{G})\right] > 1-\delta.$$

An (ϵ, δ)-approximation is called a *fully polynomial randomized approximation scheme* (FPRAS) if its running time is bounded by a polynomial in terms of $1/\epsilon, \log(1/\delta)$, and the input size. In general, an FPRAS is the best theoretical result one can hope for a #P-hard computational problem.

We present our *Component Sampling Procedure* (CSP) with two important advantages over the naive Monte Carlo method. First, it has a polynomial time complexity and is, thus, an FPRAS for the $\mathsf{EPC}(\mathcal{G})$ problem. Second, it has a smaller average time complexity, and is up to n times faster than naive Monte Carlo methods.

CSP is summarized in Algorithm 2. The algorithm computes the sum of edge probabilities $P_E = \sum_{e \in E} p_e$. If P_E is sufficiently small (at most $\frac{\epsilon}{2}n^{-2}$), the algorithm returns P_E as an unbiased estimator of $\mathsf{EPC}(\mathcal{G})$. Otherwise, it performs an importance sampling method to estimate $\mathsf{EPC}(\mathcal{G})$ in steps 4 to 6. In the importance sampling method, we select a node $u \in V$ uniformly and perform a Bread-First Search procedure from u, until reaching all nodes in the connected component that contains u. The algorithm then computes the average of the size of the component that contains u less one, and multiply the result by n to obtain an unbiased estimator $\mathcal{E}_2$.

One advantage of CSP over direct Monte-Carlo approaches is that it avoids generating too many graph samples when the EPC is predicted to be small. This is the key to guarantee that the algorithm is polynomial-time. Further, the algorithm does not generate the whole sample graph at once, but only reveal the availability of edges along the Bread-First Search procedure. This characteristic substantially reduces CSP's average running time, as analyzed later in Theorem 3.

Algorithm 2. (ϵ, δ) **Component Sampling Procedure to Approximate** $\mathsf{EPC}(\mathcal{G})$

1) Let $P_E = \sum_{e \in E} p_e$
2) **if** $P_E < \frac{\epsilon}{2} n^{-2}$ **then**
3) return $\mathcal{E}_2 = P_E$.
4) $C_2 \leftarrow 0$.
5) **for** $i = 1$ to $N(\epsilon, \delta)$ **do**
   - Select a node $u \in V$ uniformly.
   - Start a Breath-First Search from u. For each encountered edge (v, w), flip a coin of bias p_{vw} to determine its availability.
   - Let S_i be the number of visited nodes, including node u.
   - $C_2 = C_2 + (S_i - 1)$.
6) Return $\mathcal{E}_2 = \frac{nC_2}{2N}$ as an unbiased estimator of $\mathsf{EPC}(\mathcal{G})$.

B. Correctness

We determine whether or not the value EPC is too small based on the value of $P_E = \sum_{e \in E} p_e$. This is based on an observation that EPC is sandwiched between two functions of P_E, as shown in the following proposition.

Proposition 1: Let $\mathcal{G} = (V, E, p)$ be an probabilistic graph and $P_E = \sum_{e \in E} p_e$, the following inequality holds

$$P_E \le \mathsf{EPC}(\mathcal{G}) \le \left(1 + \frac{P_E}{m}\right)^m. \tag{20}$$

The bounds in Proposition 1 are asymptotic tight in the sense that there are arbitrary large graphs in which the bounds are only different from the actual values of $\mathsf{EPC}(\mathcal{G})$ by a factor of two. For example, consider $\mathcal{G}$ as a star graph of size n that consists of one center vertex and $n-1$ leaves. All $n-1$ edges are assigned the same probability $1/(n-1)$. One can verify that the lower-bound, $\mathsf{EPC}(\mathcal{G})$, and the upper bound are $1, \frac{3}{2} - \frac{1}{2(n-1)}$, and $\left(1 + \frac{1}{n-1}\right)^{n-1} < e$, respectively.

Further, we state several inequalities needed for (ϵ, δ)-approximation proof in the following proposition.

Proposition 2: Let $\mathcal{G} = (V, E, p)$ be an probabilistic graph and q_i be the probability that $\mathcal{G}$ has exactly i edges, for $i = 0..|E|$. If $P_E = \sum_{e \in E} p_e < 1/2$, the following inequalities hold

$$1 - P_E \le q_0 \le \exp(-P_E), \tag{21}$$

$$q_0 P_E \le q_1 \le q_0 \frac{P_E}{1 - P_E}, \tag{22}$$

$$\sum_{k=2}^{m} q_k \le P_E^2. \tag{23}$$

For the sake of completeness, we present the proofs of Proposition 1 and 2 in the Appendix.

We now derive $N(\epsilon, \delta)$, the number of necessary samples to be drawn using the following Generalized Zero-One Estimator Theorem introduced by Dagum et al. [27].

Theorem 1: (Generalized Zero-One Estimator [27]) Let $X_1, X_2, \ldots, X_N$ be independent identically distributed random variables taking values in $[0, 1]$, with mean $\mu > 0$. If $0 < \epsilon < 1$ and $N \ge 4(e-2)\ln(2/\sigma)1/(\epsilon^2 \mu)$, where $e \approx 2.718$ is Euler's number, then

$$\mathsf{Pr}\left[(1-\epsilon)\mu \le \frac{1}{N}\sum_{i=1}^{N} X_i \le (1+\epsilon)\mu\right] > 1 - \delta.$$

The required number of samples to obtain an (ϵ, δ) approximation is

$$N(\epsilon, \delta) = 4(e-2)\ln\frac{2}{\sigma}\frac{n(n-1)}{\epsilon^2 \mathsf{EPC}(\mathcal{G})},$$

as proved in the following lemma.

Lemma 3: If $N(\epsilon, \delta) \ge 4(e-2)\ln\frac{2}{\sigma}\frac{n(n-1)}{\epsilon^2 \mathsf{EPC}(\mathcal{G})}$, then $\mathcal{E}_2$, the output of CSP, is an (ϵ, δ)-approximation for $\mathsf{EPC}(\mathcal{G})$.

Proof: We consider two cases of P_E.

Case $P_E < \frac{\epsilon}{2} n^{-2}$: CSP returns P_E (step 2). We show that P_E is indeed an (ϵ, δ)-approximation for $\mathsf{EPC}(\mathcal{G})$ by proving the following inequalities.

$$P_E \le \mathsf{EPC}(\mathcal{G}) \le (1+\epsilon) P_E.$$

From Lemma 1, we already have $P_E \le \mathsf{EPC}(\mathcal{G})$. Thus we only need to show

$$\mathsf{EPC}(\mathcal{G}) \le (1+\epsilon) P_E.$$

Denote by $q_k, k = 0..m$, the probability that $\mathcal{G}$ has exactly k edges. Since in any (deterministic) graph with m edges and n vertices, the pairwise connectivity cannot exceeds $\binom{\min\{m-1, n\}}{2}$, we obtain the following inequality.

$$\mathsf{EPC}(\mathcal{G}) \le \tbinom{1}{2} \cdot q_0 + \tbinom{2}{2} \cdot q_1 + \tbinom{n}{2} \sum_{k=2}^{m} q_k \tag{24}$$

Apply inequalities (22), and (23)

$$\mathsf{EPC}(\mathcal{G}) \le q_0 \frac{p_E}{1 - p_E} + \tbinom{n}{2} P_E^2.$$

Using inequality (21), we arrive

$$\mathsf{EPC}(\mathcal{G}) \le \exp(-P_E)\frac{P_E}{1 - P_E} + \tbinom{n}{2} P_E^2.$$

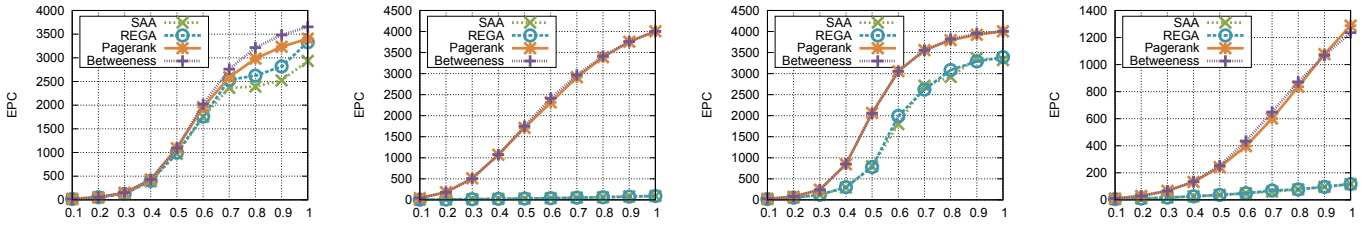
Since $P_E \le \frac{\epsilon}{2} n^{-2} < 1/2$, we apply the inequality $\frac{\exp(-x)}{1-x} \le 1 + x, x \in (0, \frac{1}{2})$ to yield

$$\mathsf{EPC}(\mathcal{G}) \le \left(1 + P_E + \tbinom{n}{2} P_E\right) P_E \le (1+\epsilon) P_E. \tag{25}$$

This completes the proof of P_E being (ϵ, δ)-approximation of $\mathsf{EPC}(\mathcal{G})$ when $P_E < \frac{\epsilon}{2} n^{-2}$.

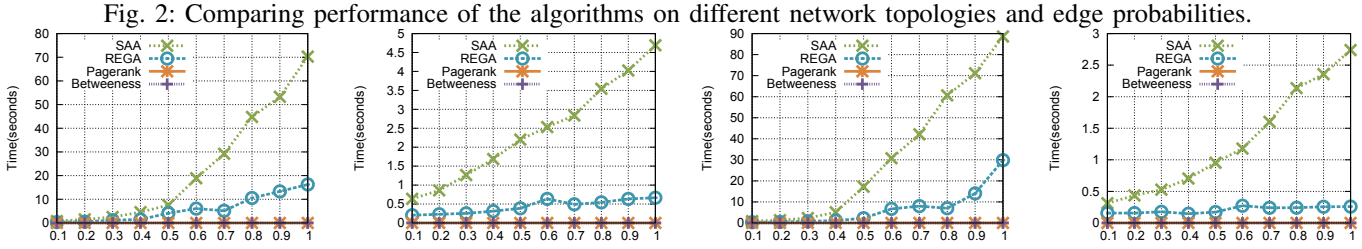
Case $P_E \ge \frac{\epsilon}{2} n^{-2}$: The importance sampling is carried out in steps 4 to 6 in Algorithm 2. Within the loop in Step 5, we can compute S_i with the following equivalent procedure: 1) Draw a sample graph G^i; and 2) Select a node u in G^i uniformly and compute S_i as the size of connected component that contains u. Assume that there are t connected components with sizes $s_1, s_2, \ldots, s_t$ in G^i. We have

$$\mathbb{E}[S_i - 1 | \mathcal{G} = G^i] = \frac{\sum_{i=1}^{k} s_i(s_i - 1)}{\sum_{i=1}^{k} s_i} = 2\frac{\mathcal{P}(G^i)}{n}.$$



(a) Erdos-Renyi network (b) Barabasi-Albert Network (c) Small-world Network (d) US backbone network

Fig. 2: Comparing performance of the algorithms on different network topologies and edge probabilities.



(a) Erdos-Renyi network (b) Barabasi-Albert Network (c) Small-world Network (d) US backbone network

Fig. 3: Comparing running time of the algorithms (y-axis) with different edge probabilities (x-axis).

Hence

$$\mathbb{E}[S_i - 1] = 2\mathsf{EPC}(\mathcal{G})/n. \tag{26}$$

By applying Theorem 1 to i.i.d. random variables $Y_i = (S_i - 1)/(n-1)$ with mean $\mu = \mathsf{EPC}(\mathcal{G})/\binom{n}{2}$, it follows that $\mathcal{E}_2$ is an (ϵ, δ)-approximation of $\mathsf{EPC}(\mathcal{G})$. ■

C. Time Complexity Analysis

Lemma 4: CSP has a time complexity $O(mn^4\epsilon^{-3})$.

Proof: If $P_E < \frac{\epsilon}{2}n^{-2}$, the algorithm takes an $O(m)$ time, as we only need to compute P_E.

Otherwise, the algorithm performs $N(\epsilon, \delta)$ times the BFS algorithm in step 5. Since the BFS algorithm takes a time at most $O(m+n)$, the total time taken by Algorithm 2 is upper bounded by

$$\theta(m+n)4(\exp -2)\ln\frac{2}{\delta}\frac{n(n-1)}{\epsilon^2\mathsf{EPC}(\mathcal{G})}$$

By Proposition 1, $\mathsf{EPC}(\mathcal{G}) \geq P_E \geq \frac{\epsilon}{2}n^{-2}$. Thus the worst-case time complexity is at most $O(mn^4\epsilon^{-3})$. ■

Lemmas 3 and 4 immediately lead to our main result for approximating EPC, stated in the following theorem.

Theorem 2 (Main theorem): CSP is an FPRAS for the EPC computation problem that outputs an (ϵ, δ)-approximation of EPC in an $O(mn^4\epsilon^{-3})$ time.

CSP not only has a polynomial running time, it's also faster than naive Monte Carlo methods. In general, the time needed for the BFS procedure in CSP is often less than the time to generate a sample graph. The reason is that the BFS procedure only needs to be aware about the surrounding of the selected vertex u, while generating a graph sample might involve all edges and vertices in the graph. To formally prove this observation, we give the expected running time of CSP in the following theorem.

Theorem 3: CSP has an expected time complexity $O\left(\epsilon^{-2}\min\{n^2 + \frac{mn}{\mathsf{EPC}(\mathcal{G})}, \frac{mn^2}{\mathsf{EPC}(\mathcal{G})}\}\right)$.

The proof of this theorem can be found in the Appendix.

VI. Experiments

We demonstrate through our experiments the efficiency of our proposed algorithm and the need of new assessment methods for networks with uncertainty.

A. Experiment Setup

Dataset. We analyze the performance our proposed algorithm through experiments on different network models and a real communication network, as described below.

- *Erdos-Reyni*: A random graph of 100 vertices and 200 edges following the Erdos-Reyni model [23].
- *Barabasi-Albert*: A random graph of 100 vertices and 200 edges. The graph follows power-law model using preferential attachment mechanism [28].
- *Watts–Strogatz*: A random graph that is generated from the small-world model [29] with the dimension of the lattice 2 and the rewiring probability 0.3 [29].
- *US Backbone network*: The US backbone cabling network of XO company [30] with 78 nodes and 91 links.

Compared Methods. We compare the performance of REGA, Algorithm 1, with the following methods

- *SAA*: Sample Average Approximation method [10], a common technique to solve stochastic optimization problem. The solution is furthered optimize using the same local search procedure in REGA. The number of samples to optimize is $T = 30$.
- *Betweeness*, a greedy algorithm that removes the nodes with the highest betweeness centrality values.
- *Pagerank*, another heuristic that removes the edge with the highest Pagerank values. The damping factor is 0.85.

The number of samples drawn in the local search procedure in both SAA and REGA are 1000. During our experiments, we observe that the local search procedure are quite insensitive to the number of sampling times. The final EPC in each network is, however, measured by setting the number of sample times to 100,000 to guarantee high accurate estimation of EPC.

Enviroment. All algorithms are implemented in C++ and compiled with GCC 4.4 compiler on a 64 bit Window machine with an i7 3.4Ghz processor and 16 GB memory. The mathematical optimization package to solve linear programming formulation is GUROBI 5.5.

B. Experimental results.

The solution quality, i.e., the expected pairwise connectivity (EPC) in the residual networks are shown in Figure 2. The lower EPC value, the better the algorithm performs. Thus both SAA and REGA are much better than the adhoc heuristics based on centrality. The results of SAA and REGA are highly similar, except for the largest test cases, where SAA shows a slight advantage over REGA.

The running time of the algorithms are shown in Figure 3 in log-scale. There is no doubt that heuristics based on centrality takes only a fraction of second to complete and is much faster than REGA and SAA. SAA runs much slower than REGA, up to 10 times slower. This expected behavior is due to the larger size of the linear program that SAA has to cope with.

Overall, REGA turns out to be the best choice in terms of both quality and running time. It runs much faster than SAA, and also provide much better solution quality than the naive centrality-based heuristics.

VII. Conclusion

Assessing vulnerability in networks with uncertainty is a challenging problem. While the NP-hardness of exact computation for network reliability measures is a significant obstacle, such obstacle can be overcome with efficient computational methods, e.g., the FPRAS to compute EPC. In future, we aim to investigate efficient methods to compute other network reliability measures as well as design more efficient solutions for the vulnerability assesment in forms of optimization problems.

VIII. Acknowledgement

This work is partially supported by the NSF CAREER Award 0953284 and by the DTRA grant HDTRA1-14-1-0055.

Appendix

A. Proof of Proposition 1

Proof: We prove the lower and upper bounds separately.

Lower bound: By Lemma 1, we have

$$\begin{aligned}\mathsf{EPC}(\mathcal{G}) &= \frac{1}{2}\sum_{u,v\in V;u\neq v}\mathsf{REL}_{uv}(\mathcal{G})\\ &\geq \sum_{(u,v)\in E}\mathsf{REL}_{uv}(\mathcal{G}) \geq \sum_{(u,v)\in E} p_{uv}\end{aligned}$$

Upper bound: First, we show that $\mathsf{EPC}(\mathcal{G}) \leq \prod_{e\in E}(1+p_e)$. Then we can apply the inequality of arithmetic and geometric means for positive numbers $(1+p_e)\ \forall e \in E$ to obtain

$$\mathsf{EPC}(\mathcal{G}) \leq \prod_{e\in E}(1+p_e) \leq \left(1+\frac{1}{m}\sum_{e\in E}p_e\right)^m.$$

We prove $\mathsf{EPC}(\mathcal{G}) \leq \prod_{e\in E}(1+p_e)$ by induction on μ_E the number of *undetermined* edges (those with probabilities strictly less than one).

Basis: If $\mu_E = 0$, we have a deterministic graph with $m = |E|$ edges. Since, the size of the largest component cannot exceed $m+1$, the pairwise connectivity is at most $1/2n(m+1) < 1/2m(m+1) < 2^m\ \forall m \geq 0$. Thus, the inequality holds for $\mu_E = 0$.

Induction step: Assume that the inequality holds for $\mu_E = t \geq 0$, we show that the inequality also holds when $\mu_E = t+1$. Assume that $\mu_E = t+1$, select an arbitrary undetermined edge $(u,v) \in E$ and perform a branching procedure on (u,v) we have

$$\mathsf{EPC}(\mathcal{G}) = p_{uv}\mathsf{EPC}(\mathcal{G}^+) + (1-p_{uv})\mathsf{EPC}(\mathcal{G}^-),$$

where $\mathcal{G}^+$ is obtained from $\mathcal{G}$ by setting the (u,v)'s probability to one and $\mathcal{G}^-$ is obtained from $\mathcal{G}$ by removing (u,v). Since, both $\mathcal{G}^+$ and $\mathcal{G}^-$ have exactly μ_E undetermined edges, we can apply the induction hypothesis to obtain

$$\begin{aligned}\mathsf{EPC}(\mathcal{G}) &\leq p_{uv}(1+1)\prod_{e\neq(u,v)}(1+p_e)\\ &\quad + (1-p_{uv})\prod_{e\neq(u,v)}(1+p_e)\\ &= (1+p_{uv})\prod_{e\neq(u,v)}(1+p_e) = \prod_{e\in E}(1+p_e).\end{aligned}$$

Thus, the inequality holds for all $\mu_E \geq 0$. ■

B. Proof of Proposition 2

Inequalities on q_0: On one hand

$$q_0 = \prod_{e\in E}(1-p_e) \geq 1 - \sum_{e\in E}p_e = 1 - P_E. \tag{27}$$

On the other hand, we have

$$\begin{aligned}q_0 &\leq (1-\frac{\sum_{e\in E}p_e}{m})^m \qquad \text{(AM-GM inequality)}\\ &= \left((1-\frac{P_E}{m})^{m/P_E}\right)^{P_E} < \exp(-P_E).\end{aligned} \tag{28}$$

The last step holds due to the inequality

$$(1-x)^{1/x} \leq \exp(-x), x \in (0,1),$$

. **Inequalities on q_1:** Iterate through all edges in E, we have

$$q_1 = \sum_{e\in E}p_e\prod_{e'\neq e}(1-p_{e'}) = q_0\sum_{e\in E}\frac{p_e}{1-p_e}.$$

Since $0 \leq p_e \leq P_E < 1$, it follows that

$$\sum_{e\in E}p_e \leq \sum_{e\in E}\frac{p_e}{1-p_e} \leq \frac{\sum_{e\in E}p_e}{1-P_E}.$$

Hence

$$q_0P_E \leq q_1 \leq q_0\frac{P_E}{1-P_E}.$$

Inequalities on $\sum_{k=2}^m q_k$: Apply (22) and then (21), we obtain

$$\begin{aligned}\sum_{k=2}^m q_k &= 1 - q_0 - q_1 \leq 1 - q_0(1+P_E)\\ &\leq 1 - (1-P_E)(1+P_E) = P_E^2.\end{aligned}$$

C. Proof of Theorem 3

Proof: In step 5 of CSP, each vertex u is chosen uniformly with a probability $1/n$. The BFS procedure starts from u and gradually reveals the availability of edges when needed. For any vertex v visited by the BFS procedure (including the starting vertex u), it takes $O(d_v)$ times to check the availability of the incident edges.

Thus the expected number of edges that are incident at v and visited by the BFS procedure at u will be

$$\mathsf{REL}_{u,v}(\mathcal{G})d_v.$$

And the expected number of visited edges by the BFS procedure at u will be $\sum_{v\in V}\mathsf{REL}_{u,v}(\mathcal{G})d_v$. Since u is chosen uniformly, the expected number of visited edges by the BFS procedure is

$$\frac{1}{n}\sum_{u\in V}\sum_{v\in V}\mathsf{REL}_{u,v}d_v = \frac{1}{n}\sum_{u\in V}\left(d_u\sum_{v\neq u}\mathsf{REL}_{u,v}\right)+\frac{m}{n}$$

From (26), the expected number vertices visited by the BFS procedure is

$$2\mathsf{EPC}(\mathcal{G})/n+1.$$

Thus the expected time complexity of the BFS procedure is

$$O\left(\frac{1}{n}\sum_{u\in V}\left(d_u\sum_{v\neq u}\mathsf{REL}_{u,v}\right)+\frac{m}{n}+\frac{\mathsf{EPC}(\mathcal{G})}{n}\right)$$

Apply the inequality $d_u \leq n$, the expected time complexity of the BFS procedure can be simplified to

$$O\left(\mathsf{EPC}(\mathcal{G})+\frac{m}{n}\right).$$

Multiply the above with $N(\epsilon,\delta)$ gives us the expected time complexity of CSP

$$O\left(\epsilon^{-2}(n^2+\frac{mn}{\mathsf{EPC}(\mathcal{G})})\right).$$

Since the BFS procedure takes at most $O(m+n)$ time, another upper-bound for the expected time complexity of CSP is

$$O(N(\epsilon,\delta)(m+n)).$$

The combination of the above two complexity forms of CSP gives us the $O\left(\epsilon^{-2}\min\{n^2+\frac{mn}{\mathsf{EPC}(\mathcal{G})},\frac{mn^2}{\mathsf{EPC}(\mathcal{G})}\}\right)$ expected time complexity of CSP. ∎

REFERENCES

[1] S. Neumayer, G. Zussman, R. Cohen, and E. Modiano, "Assessing the vulnerability of the fiber infrastructure to disasters," in *Proc. of IEEE INFOCOM*, 2009.

[2] A. Pinar, J. Meza, V. Donde, and B. Lesieutre, "Optimization strategies for the vulnerability analysis of the electric power grid," *SIAM J. on Optimization*, vol. 20, 2010.

[3] A. Murray, T. Matisziw, and T. Grubesic, "Multimethodological approaches to network vulnerability analysis," *Growth Change*, 2008.

[4] A. Arulselvan, C. W. Commander, L. Elefteriadou, and P. M. Pardalos, "Detecting critical nodes in sparse graphs," *Computers and Operations Research*, vol. 36, no. 7, 2009.

[5] T. N. Dinh, Y. X., M. T. Thai, E. Park, and T. Znati, "On approximation of new optimization methods for assessing network vulnerability," in *Proc. of IEEE INFOCOM*, 2010.

[6] S. Neumayer, G. Zussman, R. Cohen, and E. Modiano, "Assessing the vulnerability of the fiber infrastructure to disasters," *IEEE/ACM Trans. Netw.*, pp. 1610–1623, 2011.

[7] C. Colbourn, *The combinatorics of network reliability*, ser. International Series of Monographs on Computer Science Series. Oxford University Press, Incorporated, 1987.

[8] D. R. Karger, "A randomized fully polynomial time approximation scheme for the all terminal network reliability problem," in *SIAM J. COMPUT*, 1996, pp. 11–17.

[9] J. F. Benders, "Partitioning procedures for solving mixed-variables programming problems," *Num. Math.*, vol. 4, pp. 238–252, 1962.

[10] A. Kleywegt, A. Shapiro, and T. Homem-de Mello, "The sample average approximation method for stochastic discrete optimization," *SIAM Journal on Optimization*, vol. 12, no. 2, pp. 479–502, 2002.

[11] R. Albert, H. Jeong, and A. Barabasi, "Error and attack tolerance of complex networks," *Nature*, vol. 406, no. 6794, p. 14, 2000.

[12] T. H. Grubesic, T. C. Matisziw, A. T. Murray, and D. Snediker, "Comparative approaches for assessing network vulnerability," *Inter. Regional Sci. Review*, 2008.

[13] T. C. Matisziw and A. T. Murray, "Modeling s-t path availability to support disaster vulnerability assessment of network infrastructure," *Comput. Oper. Res.*, vol. 36, pp. 16–26, January 2009.

[14] T. N. Dinh and M. T. Thai, "Precise structural vulnerability assessment via mathematical programming," in *Proc. of IEEE MILCOM*, 2011.

[15] T. Dinh and M. Thai, "Network under joint node and link attacks: Vulnerability assessment methods and analysis," *Networking, IEEE/ACM Transactions on*, 2015.

[16] K. Aggarwal, J. S. Gupta, and K. Misra, "A simple method for reliability evaluation of a communication system," *Communications, IEEE Transactions on*, vol. 23, no. 5, pp. 563–566, 1975.

[17] N. S. Fard and T.-H. Lee, "Cutset enumeration of network systems with link and node failures," *Reliability Engineering & System Safety*, vol. 65, no. 2, pp. 141 – 146, 1999.

[18] J. Provan and M. Ball, "The complexity of counting cuts and of computing the probability that a graph is connected," *SIAM Journal on Computing*, vol. 12, no. 4, pp. 777–788, 1983.

[19] C. J. Colbourn, "Analysis and synthesis problems for network resilience," *Mathematical and Computer Modelling*, vol. 17, no. 11, pp. 43 – 48, 1993.

[20] A. T. Amin, K. T. Siegrist, and P. J. Slater, "On the nonexistence of uniformly optimal graphs for pair-connected reliability," *Networks*, vol. 21, no. 3, pp. 359–368, 1991.

[21] S. Neumayer and E. Modiano, "Network reliability with geographically correlated failures," in *INFOCOM, 2010 Proceedings IEEE*, March 2010, pp. 1–9.

[22] P. K. Agarwal, A. Efrat, S. K. Ganjugunte, D. Hay, S. Sankararaman, and G. Zussman, "The resilience of wdm networks to probabilistic geographical failures," *Networking, IEEE/ACM Transactions on*, vol. 21, no. 5, pp. 1525 – 1538, 2013.

[23] P. Erdos and A. Renyi, "On the evolution of random graphs," *Publ. Math. Inst. Hungary. Acad. Sci.*, vol. 5, pp. 17–61, 1960.

[24] A. Arulselvan, C. W. Commander, L. Elefteriadou, and P. M. Pardalos, "Detecting critical nodes in sparse graphs," *Computers & Operations Research*, vol. 36, no. 7, pp. 2193–2200, 2009.

[25] A. Shapiro, D. Dentcheva, and A. Ruszczyński, *Lectures on Stochastic Programming: Modeling and Theory*, ser. MPS-SIAM Series on Optimization Series, 2009.

[26] Y. Shen, N. P. Nguyen, Y. Xuan, and M. T. Thai, "On the discovery of critical links and nodes for assessing network vulnerability," *IEEE/ACM Transactions on Networking (TON)*, vol. 21, no. 3, pp. 963–973, 2013.

[27] P. Dagum, R. Karp, M. Luby, and S. Ross, "An optimal algorithm for monte carlo estimation," *SIAM Journal on Computing*, vol. 29, no. 5, pp. 1484–1496, 2000.

[28] A. Barabasi, R. Albert, and H. Jeong, "Scale-free characteristics of random networks: the topology of the world-wide web," *Phy. A*, 2000.

[29] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world' networks." *Nature*, vol. 393, no. 6684, 1998.

[30] "US IP Backbone network XO company," http://www.xo.com/about/network/Pages/overview.aspx.