Noname manuscript No.
(will be inserted by the editor)

Bound and Exact Methods for Assessing Link Vulnerability in Complex Networks

T. N. Dinh · M. T. Thai · H. T. Nguyen

Received: date / Accepted: date

Abstract Assessing network systems for failures is critical to mitigate the risk and develop proactive responses. In this paper, we investigate devastating consequences of link failures in networks. We propose an exact algorithm and a spectral lower-bound on the minimum number of removed links to incur a significant level of disruption. Our exact solution can identify optimal solutions in both uniform and weighted networks through solving a well-constructed mixed integer program. Also, our spectral lower-bound derives from the Laplacian eigenvalues an estimation on the vulnerability of large networks that are intractable for exact methods. Through experiments on both synthetic and real-world networks, we demonstrate the efficiency of the proposed methods.

Keywords Vulnerability Assessment · Pairwise connectivity · Integer Programming · Spectral Bound

1 Introduction

Natural disasters and malicious attacks can drastically compromise many communication and transportation systems, impair their quality-of-service (QoS), or cause total network breakdown [5, 22, 26, 29]. It is crucial to assess network vulner-

T. N. Dinh
Department of Comp. Sci., Virginia Commonwealth University
Richmond, VA 23284, USA
E-mail: tndinh@vcu.edu

M. T. Thai
Department of Comp. & Info. Sci. & Engi., University of Florida,
Gainesville, FL 32611, USA
E-mail: mythai@cise.ufl.edu

H. T. Nguyen
Faculty of Info. Tech., Ton Duc Thang University
Ho Chi Minh City, Vietnam
E-mail: hien@tdt.edu.vn

ability and identify the most destructive attack scenarios to either reduce the risk or develop proactive responses.

To measure the effect of failures, existing works in assessing network vulnerability focus mainly on using centrality measurements e.g. degree, betweeness, and closeness centralities [2, 3] to identify critical links or nodes. Unfortunately, these approaches only determine the relative importance of nodes or links and cannot reveal the enormous damage potential caused under *simultaneous* attacks. Other set of works studies links and nodes removal problems that optimize several global graph measures, such as clustering coefficient, network diameter, etc. However, these measures do not cast well for particular kinds of network vulnerability, when the network connectivity is of high priority. To this end, *pairwise connectivity*, the number of node pairs that remain connected, has lent itself as an effective measure to account for the effect of the attacks [4, 9, 16, 18, 26, 27].

In this paper, we assess the network for link vulnerability and identify the set of links that are critical for network connectivity. Specifically, we focus on the *β-edge disruptor* problem [18], which finds *a minimum cost links* whose removal lessen the fraction of connected node pairs to at most a fraction β. Several advantages of β-disruptor assessment frameworks include the ability to identify small subsets of critical elements whose failures lead to network-wide fragmentation and the ability to assess network vulnerability at multiple disruption levels [18]. However, the β-edge disruptor problem is NP-hard, i.e., there is no efficient algorithm for the problem.

First, we present an exact solution for the β-edge disruptor problem, in which we apply branch-and-cut method to solve a new mixed integer programming (MIP) formulation of the β-edge disruptor. The two intriguing aspects of the exact solution are the small size of the MIP formulation and a specialized cutting plane procedure that tightens the bound on the optimal solutions. Second, we provide a lower-bound on the optimal solution via analyzing the Laplacian eigenvalues. Our spectral bound is formulated as an optimization problem of the Laplacian eigenvalues, which are known to contain rich information about the network topology [14].

Related work. Many existing works on network vulnerability assessment mainly focus on the local centrality measurements to differentiate between critical links and nodes and the others, see [12, 18, 24]. Other global graph measures have also been proposed to assess network vulnerability. These measures are mainly functions of graph properties, such as the diameter, global clustering coefficient, etc. [6, 26].

Matisziw and Murray [24] first proposed the pairwise connectivity as an effective measurement and use mathematical programming to solve for exact solutions. Arulselvan et al. later define the Critical Nodes Problem (CNP), in which the main objectives are to identify k nodes whose removal minimize the pairwise connectivity in the residual network. The authors provide an NP-completeness proof and an integer programming formulation that can find exact solutions for networks up to 150 nodes.

Shen et. al. [30] proposes polynomial-time algorithms for special cases of trees and series-parallel graphs. A polynomial-time algorithm for bounded treewidth graphs is given in [1]. In general graphs, it is NP-hard to approximate an optimal solution of CNP within any finite factor (in polynomial time) [1]. In [28], Oosten et. al. provide polyhedral studies that show the triangle inequalities in the LP formulation are facet-defining inequalities. In [15], Suma et. al. proposes an Integer

Programming formulation with a non-polynomial number of constraints whose linear relaxation can be solved in polynomial time. In [34], the authors construct an $1/\theta$-approximation to the optimal objective value of the LP relaxation. However, the proposed algorithm is not an approximation algorithm.

Relatively little work is done on detecting critical links. The β-edge disruptor problem are shown to be NP-hard [18] and the Critical Links Problem (CLP) is shown to be NP-hard even in Unit disk graphs and power-law graphs [32]. The β-edge problem admits a bicriteria approximation algorithm with a factor $O(\log^{3/2} n)$ [18]. The approximation factor is later improved to $O(\sqrt{\log n})$ in a bicriteria algorithm that can identify at the same time both critical nodes and edges [17].

Dinh et. al. [16] provides the first sparse formulation for the problem, which reduces the number of constraints from $\theta(n^3)$ to $\theta(mn)$. The sparse formulation enables the solving for exact solution in networks up to 1,000 nodes. In [35], the authors gives a novel compact formulation for CNP that gives exact solutions for networks up to 1,5000 nodes. The new compact formulation has only $\theta(n^2)$ constraints (in comparison to $\theta(n^3)$ constraints in [4,18,28] and $\theta(mn)$ constraints in [16]). We note that the compact formulation in [35] and the sparse formulation in [16] has roughly the same number of non-zero coefficient. Additionally, the constraints in [16] are all facet-defining [28], while those in [35] are not known to be. Also, the compact formulation in [35] cannot be extended to solve CLP and β-edge disruptor.

Bissias et al. [9, 10] study the problem of bounding the damage under link attacks. However, the provided methods either require solving costly semidefinite programming problem [9] or involving weak bounds due to the presence of partitions with negative sizes [10]. There are also other versions of critical Node/Edge problems for the minimum spanning tree problem [8], shortest path between two nodes [33], and the p-median and p-center location problems [7].

Organization. We briefly present terminologies and problem definitions in Section 2. The new sparse formulation for the β-edge disruptor problem is given in Section 3. In Section 4, we introduce the spectral lower-bound for the the β-disruptor problem together with two methods to compute the lower-bound. Experimental results on different network models and real network instances are obtained in Section 5. We conclude the paper in Section 6.

2 Model and Definitions

Let $G = (V, E)$ be an undirected graph, where the vertices in V are numbered from 1 to $n = |V|$ and E refers to a set of links. Each edge $(i, j) \in E$ is associated with a cost $c_{ij} \geq 0$ (and $c_{ij} = 0$ if $(i, j) \notin E$). For convenience, we also denote the number of nodes and links by n and m, respectively.

We measure network performability using *pairwise connectivity*, defined as the number of connected vertex pairs in G [18]. The pairwise connectivity of G, denoted by $\mathcal{P}(G)$, is maximized at $\binom{n}{2}$ when G is connected. The β-edge disruptor is defined in [18] as follows.

β-edge disruptor. Given $0 \leq \beta \leq 1$, a subset $E_\beta \subset E$ in $G = (V, E)$ is a *β-edge disruptor* if the pairwise connectivity of $G[E \setminus E_\beta]$, obtained by removing E_β from G, is no more than $\beta\binom{n}{2}$. In the *β-edge disruptor problem*, we aim to find a minimum cost β-edge disruptor.

We formulate the problem as an Integer Programming, called IP_d. We use variables x_{ij} to represent the connectivity between a pair of nodes i and j in the *residual network*, after removing edges. That is

$$x_{ij} = \begin{cases} 0 & \text{if } i \text{ and } j \text{ are in the same connected component} \\ 1 & \text{otherwise.} \end{cases}$$

The IP_d formulation is as follows.

$$\begin{aligned}
\text{minimize} \quad & \sum_{(i,j)\in E} c_{ij}x_{ij} && && (1)\\
\text{subject to} \quad & x_{ij} + x_{jk} - x_{ik} \geq 0, && \forall i<j<k && (2)\\
& x_{ij} - x_{jk} + x_{ik} \geq 0, && \forall i<j<k && (3)\\
& -x_{ij} + x_{jk} + x_{ik} \geq 0, && \forall i<j<k && (4)\\
& \sum_{i<j} x_{ij} \geq (1-\beta)\binom{n}{2}, && && (5)\\
& x_{ij} \in \{0,1\}, && i,j = 1..n, && (6)
\end{aligned}$$

The objective minimizes the total costs of the removed edges, i.e., the cost of the edge disruptor. For simplicity, we require $x_{ij} = x_{ji} \quad \forall i \neq j$ and $x_{ii} = 0 \ \forall i$. Constraint (2-4) are the well-known *triangle inequalities* which imply that if i and j are connected, and j and k are connected, then i and k must be connected. Constraint (5) limits the pairwise connectivity in G to be at most $\beta\binom{n}{2}$. We can show that there is an one-to-one correspondence from the optimal solutions of IP_d to the optimal solutions of the β-edge disruptor problem as stated in the following lemma.

Lemma 1 *IP_d is a correct formulation for the β-edge disruptor problem. In addition, the optimal β-edge disruptor is given by $D_x = \{(i,j) \mid (i,j) \in E \text{ and } x_{ij} = 1\}$.*

Unfortunately, the above IP_d formulation shares several drawbacks with the formulations for the Critical Nodes/Links Detection problems in [4, 31, 32]. First, the formulations contains a large number integral variables, $\Theta(n^2)$, that make the selection of branching difficult and significantly increases the depth and size of the search tree. Second, the formulations have an excessive number of constraints, $\Theta(n^3)$ constraints, that cost an extremely large amount of memory and computing time.

3 Mixed Integer Programming Approach

We introduce a lightweight mixed integer programming (MIP) formulation for β-edge disruptor in Subsection 3.1 and show that this new MIP gives the same exact solutions as the IP in [32]. Further, we introduce, in Subsection 3.3, a new class of cutting planes and the separation procedure for the problem.

3.1 Mixed Integer Programming Formulation

We first devise a new Mixed-Integer Programming (MIP) formulation for the β-edge disruptor problem that consists of only $\theta(m+n)$ integer variables and a much smaller number of constraints. The formulation applies the *sparse metric* technique in [16], to remove non-binding constraints from the formulation.

Let d_i be the degree of vertex i, and $N_{\min}(i,j)$ be the set of neighbors of i excluding j if $d(i) < d(j)$, and $N_{\min}(i,j)$ is the set of neighbors of j excluding i, otherwise. Our new MIP formulation for the β-edge disruptor problem (MIP_d) is defined as follows.

$$\text{minimize} \quad \sum_{(i,j)\in E} c_{ij}x_{ij} - \alpha \sum_{(l,k)\notin E} c_{lk}x_{lk} \tag{7}$$

$$\text{subject to} \tag{8}$$

$$x_{ij} + x_{jk} \geq x_{ik}, \qquad k \in N_{\min}(i,j), i < j \tag{9}$$

$$\sum_{i<j} x_{ij} \geq (1-\beta)\binom{n}{2}, \tag{10}$$

$$x_{ij} \in \{0,1\}, \qquad (i,j) \in E, \tag{11}$$

$$0 \leq x_{lk} \leq 1, \qquad (l,k) \notin E, \tag{12}$$

where $\alpha = \frac{1}{n^2}\min_{(i,j)\in E} c_{ij}$, a sufficiently small constant.

The number of constraints is upper-bounded by

$$1 + \sum_{i<j} \min\{d_i, d_j\} \leq \qquad 1 + \frac{1}{2}\sum_{i<j}(d_i + d_j) = 1 + \frac{n-1}{2}\sum_{i=1}^{n} d_i = O(mn).$$

For sparse networks in which $m \approx n$, the number of constraints is only $\Theta(n^2)$. Note that we also drop the integral requirements on x_{ij} $\forall (i,j) \notin E$, replacing $x_{ij} \in \{0,1\}$ with $x_{ij} \in [0,1]$. In comparison to IP_d, the number of integral variables reduces from $\Theta(n^2)$ to $\Theta(m+n)$.

3.2 The Equivalence of MIP_d and IP_d

We show that there is a one-to-one correspondence from the set of optimal solutions of MIP_d to those of IP_d. Since the IP_d is the correct formulation for the β-edge disruptor problem (Lemma 1), the optimal solutions of MIP_d also induce optimal solutions for β-edge disruptor (and vice versa).

We prove the equivalence of the compact formulation MIP_d to IP_d by showing the following facts

- Even without the integrality constraints on x_{ij} $\forall (i,j) \notin E$, all optimal solutions of MIP_d are integral (Proposition 1).
- A feasible solution x is an optimal solution of MIP_d, iff x is also an optimal solution of IP_d (Theorem 1).
- The optimal solution of LP relaxation of IP_d is also an optimal solution for the LP relaxation of MIP_d (Theorem 2).

Proposition 1 *If x is an optimal solution of MIP$_d$, then $x_{ij} \in \{0,1\}\ \forall i,j$.*

Proof We prove by contradiction. Let x be an optimal solution of MIP$_d$. If x is not an integral solution, construct x^I from x by rounding all $0 < x_{ij} < 1$ to 1. We are going to show that x^I is also a feasible solution of MIP$_d$ and x^I gives a strictly higher objective value which contradicts the optimality of x.

First, x^I satisfies the constraints (10) as we rounding up all x_{ij} values. Second, for each triple i,j and $k \in N_{\min}(i,j)$ we have $x_{ij} + x_{jk} \geq x_{ik}$. Thus if $x_{ik} = 0$ then $x^I_{ik} = 0 \leq x^I_{ij} + x^I_{jk}$; and if $x_{ik} > 0$ then either $x_{ij} > 0$ or $x_{jk} > 0$, then $x^I_{ij} + x^I_{jk} \geq 1 \geq x^I_{ik}$. Thus x^I also satisfies the triangle inequality constraints (9) and is a feasible solution of MIP$_d$.

If $0 < x_{ij} < 1$, then $x_{ij} < x^I_{ij} = 1$. The second term in the objective will make the objective given by x^I strictly smaller than the objective given by x. Thus it contradicts the x's optimality. □

Theorem 1 *A feasible solution x is an optimal solution of MIP$_d$, iff x is also an optimal solution of IP$_d$. Moreover, $D_x = \{\ (i,j) \mid x_{ij} = 1$ and $(i,j) \in E\}$ is a minimum cost β-edge disruptor.*

The theorem holds due to the fact that in any optimal solution of MIP$_d$, if nodes i and j are connected, they must be connected through one of their neighbors. This is guarnateed through constraints (9) in MIP$_d$.

Furthermore, we show that the LP relaxation of IP$_d$ and that of MIP$_d$, obtained by removing the integral conditions on the variables, are equivalent.

Theorem 2 *The LP relaxations of IP$_d$ and MIP$_d$ have a same set of optimal solutions.*

Proof Since MIP$_d$ contains only a subset of constraints of IP$_d$, any feasible solution of the relaxation of IP$_d$ is also a feasible solution of the relaxation of MIP$_d$. Thus it is sufficient to show that every optimal solution of the relaxed MIP$_d$ is also a feasible solution of the relaxed IP$_d$.

Let x be an optimal solution of the relaxation of MIP$_d$. In G, we associate each edge $(i,j) \in E$ with a weight x_{ij}, which reflects the length of (i,j). Among all paths between two nodes i and j, we choose a path with the smallest total lengths, and denote the total length of that path by x'_{ij}. Also, let $x^*_{ij} = \min\{x'_{ij}, 1\}$. We prove that x^* is a feasible solution of the relaxed IP$_d$ and that $x^* = x$. It follows that x is a feasible solution of the relaxed IP$_d$.

First, we prove that x' satisfies the following properties

1. $x'_{ij} = x_{ij}\ \forall (i,j) \in E$
2. $x'_{ij} \geq x_{ij}\ \ \forall i,j$
3. $x'_{ij} = \min_{k=1}^{n}\{x'_{ik} + x'_{kj}\}$.

We prove the first property by contradiction. Assume that there are pair(s) of vertices (i,j) that $x'_{ij} < x_{ij}$, among those pairs we select a pair (i,j) with the minimum number of hops (edges) on the shortest path between i and j. The shortest path between i and j must include one vertex in $N_{\min}(i,j)$, say k. By the definition of the shortest path, we have $x'_{ij} = x'_{ik} + x'_{kj}$.

From the constraint (9), we have $x_{ik} + x_{kj} \geq x_{ij} > x'_{ij} = x'_{ik} + x'_{kj}$. Hence we have either $x_{ik} > x'_{ik}$ or $x_{kj} > x'_{kj}$. W.l.o.g. assume that $x_{ik} > x'_{ik}$. Since the shortest path between i and k has less hops than that of the shortest path

between (i,j), we arrive to a contradiction to the selection of the pair (i,j). Thus $x'_{ij} \geq x_{ij}\ \forall i,j$.

The second property can be shown from the facts that $x'_{ij} \leq x_{ij}$ due to the direct path from i to j and $x'_{ij} \geq x_{ij}$ (the first claim).

The last property is due to the definition of x'_{ij}, the shortest distance between i and j.

Second, we show that x^* is a feasible solution of the relaxed IP_d. Since $x^*_{ik} + x^*_{kj} \geq \min\{x'_{ik}+x'_{kj}, 1\} \geq \min\{x'_{ij}, 1\} = x^*_{ij}$, thus x^* satisfies all triangle inequalities in IP_d. Also, from the definition, x^* are bounded by 0 and 1. Thus x^* is a feasible solution of the relaxed IP_d.

Finally, we show that $x^* = x$. Since $x^*_{ij} = x'_{ij} = x_{ij}\ \forall (i,j) \in E$ and $x^*_{ij} \geq x'_{ij} \geq x_{ij} \forall (i,j) \notin E$. The objective values

$$\sum_{(i,j)\in E} c_{ij}x^*_{ij} - \alpha \sum_{(l,k)\notin E} c_{lk}x^*_{lk} \leq \sum_{(i,j)\in E} c_{ij}x_{ij} - \alpha \sum_{(l,k)\notin E} c_{lk}x_{lk}.$$

If $x^*_{ij} \neq x_{ij}$ for some pair of $(i,j) \notin E$, then we have a contradiction to the optimality of x. Therefore, $x^* = x$.

Therefore any optimal solution x of the relaxed MIP_d is also an optimal solution of the relaxed IP_d (and vice versa). □

3.3 Cutting Planes

We present a class of cutting planes together with the separation procedure to identify those cutting planes. These can be used in conjunction with cutting planes generated automatically by optimization packages to reduce the running time of the branch-and-cut algorithm.

3.3.1 Edge-Connectivity and Invalid Inequalities

One often overlooked characteristic of solutions for clustering and partitioning problems on graph is that clusters must induce connected subgraph. This characteristic is not reflected in either IP_d or MIP_d formulations.

A subset $S \subset E$ is an *edge-cut* for a pair (u,v), if removing S from graph G, disconnects u and v. For an edge-cut S of (u,v), if $\sum_{(i,j)\in S} x_{ij} = |S|$, then x_{uv} must be one. Thus, we have the following inequality

$$\sum_{(i,j)\in S} x_{ij} - x_{uv} \leq |S| - 1$$

This inequality, called EC inequality, is valid for all feasible points inside the polyhedra of MIP_d.

Algorithm 1. Separation procedure for VC inequalities

for each pair $(u, v) \in V \times V$ **do**
  Construct a flow network $G = (V, E)$ as follows
    Assign u and v as source and sink, respectively
    Every edge $(i, j) \in E$ has a capacity $1 - x_{ij}$
    if $(u, v) \in E$, **then** (u, v) has capacity zero.
  Find the maximum-flow (min-cut)
  if maximum-flow is less than $\bar{x}_{uv}$, **then**
    Get S as the set of edges in the min-cut
    Add the EC inequality associated with S to MIP
  end if
end for

3.3.2 Separation Procedure for VC Inequalities

Given a point (fractional solution) $x \in \mathbb{R}^{\binom{n+1}{2}}$, an exact separation algorithm for some class of inequalities either finds a member of the class violated by x, or proves that no such member exists. In many cases, finding such algorithm is intractable (NP-hard problem) and one has to settle for heuristic procedures. Fortunately, there is an exact algorithm for our separation procedure based on finding the max-flow on the network with node capacities.

Let $\bar{x}_{ij} = 1 - x_{ij}$ and $\bar{x}_{uv} = 1 - x_{uv}$. The EC inequality can be rewritten as

$$\sum_{(i,j)\in S} \bar{x}_{ij} - \bar{x}_{uv} \geq 0, \quad \text{S is an edge-cut of } (u, v).$$

Therefore, the point x violates this inequality if and only if $\sum_{(i,j)\in S} \bar{x}_{ij} < \bar{x}_{uv}$. For a given vertex pair (u, v), the most violated inequality is the one that minimize the sum $\sum_{(i,j)\in S} \bar{x}_{ij}$, where S is an edge-cut of (u, v). Thus, the subset S corresponding to the most violated inequalities can be found using a max-flow algorithm as shown in Algorithm 1. If we apply Push-relabel algorithm with dynamic trees [21], the time complexity to find cutting planes for one node pair is $O(mn \log \frac{n^2}{m})$. The total time complexity for the separation procedure will be $O(n^3 m \log \frac{n^2}{m})$. In our implementation, this procedure is called sparingly in order to avoid excessive running time.

4 Spectral Lower-bound for Link Assessment

In this section, we derive a lower-bound on the minimum cost of β-edge disruptor using higher eigenvalues of the Laplacian matrix L. We first formulate the lower-bound as an eigenvalue optimization problem. Then we present an efficient computation method based on Lagrange multiplier strategy.

4.1 Laplacian Matrix and Its Eigenvalues

Let $\boldsymbol{A} = \{c_{ij}\}$ be the weighted *adjacency matrix* and $\boldsymbol{D}$ be the *degree matrix*, defined as the diagonal matrix with the weighted degrees $d_1, d_2, \ldots, d_n$ on the diagonal, where $d_i = \sum_j c_{ij}$.
The unnormalized graph Laplacian matrix [25] is defined as

$$\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{A}$$

The matrix L is symmetric and positive semi-definite, since for every vector $x \in \mathbb{R}^n$ we can verify that

$$x^T \boldsymbol{L} x = \frac{1}{2} \sum_{i,j=1}^{n} c_{ij}(x_i - x_j)^2 \geq 0. \tag{13}$$

A direct consequence is that $\boldsymbol{L}$ has n non-negative, real-valued eigenvalues $\lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$. In addition, the smallest eigenvalue of λ_1 is zero and the corresponding eigenvector is the constant one vector $\mathbb{1}$ [25].

The second smallest eigenvector λ_2 is known as the algebraic connectivity of the graph and can be used to describe many properties of graphs [25]. For example, the graph G is connected if and only if $\lambda_2 > 0$.

Our main result in this section is a spectral lower-bound on OPT_β given by solving the following quadratic programming (QP) optimization problem.

$$\text{minimize} \quad \frac{1}{2} \sum_{i=1}^{n} s_i \lambda_i \tag{14}$$

$$\text{subject to} \quad \sum_{i=1}^{n} s_i = n \tag{15}$$

$$\sum_{i=1}^{n} \tbinom{s_i}{2} \leq \beta \tbinom{n}{2} \tag{16}$$

$$s_i \in \{0, 1, \ldots, n\} \tag{17}$$

Let E^*_β be an optimal β-edge disruptor and let OPT_β be the total cost of the edges in E^*_β. We have the following theorem.

Theorem 3 *Let Q_β be the optimal objective of the QP problem (14-17) and OPT_β be the minimum β-edge disruptor of graph $G = (V, E)$. Then, $Q_\beta \leq \mathrm{OPT}_\beta$ for $\beta \in [0, 1]$. Moreover, the equality holds when $\beta = 0$ or $\beta = 1$*

Proof Let $s^*_1 \geq s^*_2 \geq \ldots \geq s^*_n$ be the sizes of the connected components after removing E^*_β from the network. Here we allow dummy subsets of size zero and assume w.l.o.g. that $l = n$. Note that $s^*_1, \ldots, s^*_n$ are not known without finding E^*_β. First, we have $\mathrm{OPT}_\beta = |E^*_\beta| \geq \frac{1}{2} \sum_{i=1}^{n} s^*_i \lambda_i$ from the following lemma.

Lemma 2 *[10] Let a k-partition of a graph be a division of the vertices into l disjoint subsets containing $s_1 \geq s_2 \geq \ldots \geq s_l$ vertices. Let E_{cut} be the total weights of edges whose two ends belong to different subsets. Let $\lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_l$, be the l smallest*

eigenvalues of the Laplacian matrix plus any diagonal matrix U such that the sum of all the elements of U is zero. Then

$$E_{\text{cut}} \geq \frac{1}{2}\sum_{i=1}^{l} s_i\lambda_i.$$

This generalizes the result in [19] for unweighted graphs. For completeness, the proof of this lemma is provided in the Appendix.

Since the above QP consider all possible values of $\{s_1, \ldots, s_n\}$ which infer network partitions of pairwise connectivity at most $\beta\binom{n}{2}$, it gets the minimum of the sum $\frac{1}{2}\sum_{i=1}^{n} s_i\lambda_i$ as a lower-bound on OPT_β. In addition, the sizes of connected components after removing optimal β-edge disruptor satisfy all constraints (15-17). Hence, $Q_\beta \leq \text{OPT}_\beta$ for all $\beta \in [0, 1]$.

We continue with the tightness of the bound at extreme cases.

Case $\beta = 0$: All subsets are of size one. Hence, $Q_1 = \frac{1}{2}\sum_{i=1}^{n}\lambda_i = \frac{1}{2}\text{Trace}(\boldsymbol{L}) = \frac{1}{2}(2|E|) = |E|$. The only way to cut all pairs in the network is to cut all edges. In other words, $Q_0 = \text{OPT}_0 = |E|$.

Case $\beta = 1$: To achieve the maximum connectivity $\binom{n}{2}$, there must be a single partition in the network and the optimal disruptor cutting no edges. That is $s_1 = n$ and $s_i = 0\ \forall i > 1$. Since $\lambda_1 = 0$, it follows that $Q_1 = 0 = \text{OPT}_1$. □

The bound given by QP can be computed by using a dynamic programming algorithm, presented in Appendix A.

Theorem 4 *Optimal solutions of* QP *(14-17) can be found in $O(n^4)$ time and $O(n^3)$ space.*

While the spectral bound can be computed in polynomial time, the high time complexity of the dynamic programming algorithm prevents the method from being applied to large networks. Moreover, the dynamic programming algorithm requires computing the whole set of eigenvalues of the networks, which is both time and memory consuming. We continue with an approximation of the spectral bound that achieves (almost) the same lower-bound quality in significantly less time.

4.2 Lagrange Multipliers Method

We relax the integral conditions on s_i to obtain the following relaxation of the QP, rewritten in vector notation.

$$\begin{aligned}
\text{minimize} \quad & \frac{1}{2}\boldsymbol{s}^T\boldsymbol{\lambda} && (18)\\
\text{subject to} \quad & \|\boldsymbol{s}\|_1 - n = 0, && (19)\\
& \|\boldsymbol{s}\|_2^2 - \Delta_\beta \leq 0, && (20)\\
& \boldsymbol{s} \geq 0, && (21)
\end{aligned}$$

where $\Delta_\beta = \beta n(n-1) + n$ and $\|.\|_p$ denotes the L^p norm.

The Lagrange multiplier is then

$$\mathcal{L}(\boldsymbol{s}, \chi, \psi, \boldsymbol{\omega}) = \frac{1}{2}\boldsymbol{s}^T\boldsymbol{\lambda} + \chi(\|\boldsymbol{s}\|_1 - n) + \psi(\|\boldsymbol{s}\|_2^2 - \Delta_\beta) - \boldsymbol{\omega}^T\boldsymbol{s}$$

where $\boldsymbol{\omega} = (\omega_1, \ldots, \omega_n) \geq 0$ is a positive multiplier vector.

Notice that the problem is a convex optimization problem with differentiable objective and constraint functions and it satisfies the Slater's condition with $\boldsymbol{s} = (1, 1, \ldots, 1)^T$ [13]. Hence, the following Karush–Kuhn–Tucker (KKT) conditions provide the necessary and sufficient conditions for optimality

$$\begin{aligned}
\nabla_{\boldsymbol{s}}\mathcal{L} = \frac{1}{2}\boldsymbol{\lambda} + \chi + 2\psi\boldsymbol{s} - \boldsymbol{\omega} &= 0 \\
\nabla_{\chi}\mathcal{L} = \|\boldsymbol{s}\|_1 - n &= 0 \\
\nabla_{\psi}\mathcal{L} = \|\boldsymbol{s}\|_2^2 - \Delta_\beta &= 0 \\
\boldsymbol{\omega}^T\boldsymbol{s} &= 0 \\
\boldsymbol{s}, \psi, \boldsymbol{\omega} &\geq 0
\end{aligned}$$

Algorithm 2: LMB(G, β)

1: $t = \lceil 2/\beta \rceil$, $\quad \Delta_\beta \leftarrow \lfloor \beta n(n-1) + n \rfloor$
2: Compute $\lambda_1, \ldots, \lambda_t$
3: **for** $k = 1$ **to** n
4: **if** $k > t$ **then**
5: $t = \min\{2t, n\}$
6: Compute $\lambda_1, \ldots, \lambda_t$
7: Compute ψ as in Eq. 31.
8: Compute $\mathcal{D}_\beta^{(k)}$, and $\mathcal{C}_\beta^{(k)}$ as in Eqs. 32, and 33
9: **if** ($\psi \geq 0$ and $\mathcal{C}_\beta^{(k)} \geq 0$) or ($k = n$) **then**
10: return $\lceil \mathcal{D}_\beta^{(k)} \rceil$
11: **end for**

Moreover, the magnitude of eigenvalues define the order of component sizes as shown in the following lemma.

Lemma 3 *There exists an optimal solution* $\boldsymbol{s}^*$ *of* QP*(18-21) such that* $s_1^* \geq s_2^* \geq \ldots \geq s_n^*$.

Proof Let $\boldsymbol{s}^* = \{s_1^*, s_2^*, \ldots, s_n^*\}$ be an optimal solution of QP(18-21). Denote inv$(\boldsymbol{s}^*)$ the number of inversions of m^* i.e. such pairs of indices (i, j) that $i < j$ such that $s_i^* > s_j^*$. If inv$(\boldsymbol{s}^*) = 0$, then $s_1^* \geq s_2^*, \geq \ldots \geq s_n^*$, otherwise there exists a pair $i < j$ and $s_i^* > s_j^*$. Construct $\boldsymbol{s}'$ by swapping s_i^* and s_j^* inside $\boldsymbol{s}^*$. Then, $\boldsymbol{s}'$ is a feasible solution of QP(18-21) and the objective increases an amount $s_i^*\lambda_j + s_j^*\lambda_i - (s_i^*\lambda_i + s_j^*\lambda_j) = (s_i^* - s_j^*)(\lambda_j - \lambda_i) \geq 0$. Thus, we obtain a new optimal solution with less the number of inversions. Repeat the process at most $\binom{n}{2}$, that is the maximum number of inversions in $\boldsymbol{s}^*$, we finally obtain an optimal solution with no inversions. That optimal solution shall satisfy the lemma's condition. □

Let $l = \max\{i \mid s_i > 0\}$. By Lemma 3 and the complementary slackness $\boldsymbol{\omega}^T \boldsymbol{s} = 0$, we have $s_i > 0$ for $i \leq k$, thus, $s_i = 0\ \forall i > l$ and $\omega_j = 0\ \forall j \leq l$.

Denote $\boldsymbol{s}^{(l)} = \{s_1, s_2, \ldots, s_l\}$ and $\boldsymbol{\lambda}^{(l)} = \{\lambda_1, \lambda_2, \ldots, \lambda_l\}$, the KKT condition can be simplified to

$$\nabla_{\boldsymbol{s}^{(l)}} \mathcal{L} = \frac{1}{2}\boldsymbol{\lambda}^{(l)} + \chi + 2\psi \boldsymbol{s}^{(l)} = 0, \qquad i \leq l \tag{22}$$

$$\nabla_{s_i} \mathcal{L} = \frac{1}{2}\lambda_i + \chi - \omega_i = 0, \qquad i > l \tag{23}$$

$$\nabla_{\chi} \mathcal{L} = \|\boldsymbol{s}^{(l)}\|_1 - n = 0, \tag{24}$$

$$\nabla_{\psi} \mathcal{L} = \|\boldsymbol{s}^{(l)}\|_2^2 - \Delta_\beta = 0, \tag{25}$$

$$\boldsymbol{s}^{(l)} > 0, \psi > 0, \boldsymbol{\omega}^{(l)} = 0 \tag{26}$$

For each value of l, we can solve for values of s_i and check if all $s_i \geq 0$. The other unknowns can be found as follows. First, substitute the constraint (24) into the sum of the constraints (22) to obtain χ in terms of ψ.

$$\chi = -2\frac{n}{l}\psi - \frac{\|\boldsymbol{\lambda}^{(l)}\|_1}{2l} \tag{27}$$

Therefore, we can derive $\boldsymbol{s}^{(l)}$ from (22) as

$$\boldsymbol{s}^{(l)} = \frac{n}{l} + \left(\frac{\|\boldsymbol{\lambda}^{(l)}\|_1}{4l} - \frac{\boldsymbol{\lambda}^{(l)}}{4}\right)\frac{1}{\psi} \tag{28}$$

Substituting the above equation into the condition (25) and solving for ψ, we have

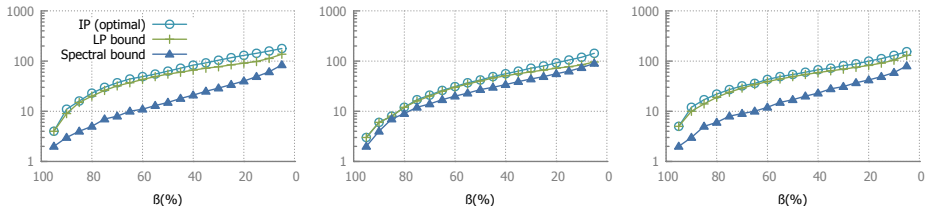
$$\begin{aligned}
&\|\boldsymbol{s}^{(l)}\|_2^2 - \Delta_\beta = 0 \\
\Leftrightarrow &\sum_{i=1}^{l}\left(\frac{n}{l} + \left(\frac{\|\boldsymbol{\lambda}^{(l)}\|_1}{4l} - \frac{\lambda_i}{4}\right)\frac{1}{\psi}\right)^2 = \Delta_\beta && (29)\\
\Leftrightarrow &\frac{n^2}{l} + 2\frac{n}{l\psi}\sum_{i=1}^{l}\left(\frac{\|\boldsymbol{\lambda}^{(l)}\|_1}{4l} - \frac{\lambda_i}{4}\right) + \sum_{i=1}^{l}\left(\frac{\|\boldsymbol{\lambda}^{(l)}\|_1}{4l} - \frac{\lambda_i}{4}\right)^2\frac{1}{\psi^2} = \Delta_\beta && (30)\\
\Leftrightarrow &\left(\frac{\|\boldsymbol{\lambda}^{(l)}\|_2^2}{16} - \frac{\|\boldsymbol{\lambda}^{(l)}\|_1^2}{16l}\right)\frac{1}{\psi^2} = \Delta_\beta - \frac{n^2}{l} \\
\Leftrightarrow &\psi = \frac{1}{4}\left(\frac{\|\boldsymbol{\lambda}^{(l)}\|_2^2 - \|\boldsymbol{\lambda}^{(l)}\|_1^2/l}{\Delta_\beta - \frac{n^2}{l}}\right)^{1/2} && (31)
\end{aligned}$$

The objective is then

$$\begin{aligned}
\mathcal{D}_\beta^{(l)} &= \frac{1}{2}\boldsymbol{s}^{(l)T}\boldsymbol{\lambda}^{(l)} = n\frac{\|\boldsymbol{\lambda}^{(l)}\|_1}{2l} + \left(\frac{\|\boldsymbol{\lambda}^{(l)}\|_1^2}{4l} - \frac{\|\boldsymbol{\lambda}^{(l)}\|_2^2}{4}\right)\frac{1}{2\psi} \\
&= n\frac{\|\boldsymbol{\lambda}^{(l)}\|_1}{2l} - \frac{1}{2}\left(\|\boldsymbol{\lambda}^{(l)}\|_2^2 - \frac{\|\boldsymbol{\lambda}^{(l)}\|_1^2}{l}\right)^{1/2}\left(\Delta_\beta - \frac{n^2}{l}\right)^{1/2}
\end{aligned} \tag{32}$$

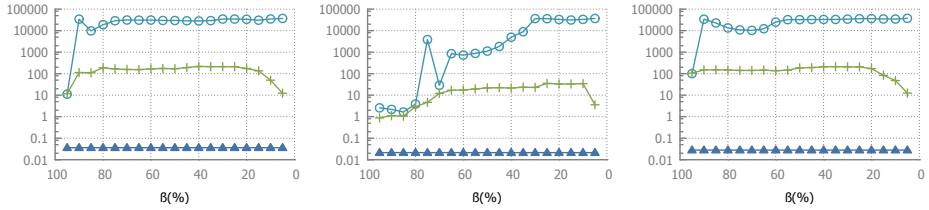
Since $\lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$, Eq. 28 implies that $s_1^{(l)} \geq s_2^{(l)} \geq \ldots \geq s_k^{(l)}$. Hence, in order to satisfy $\boldsymbol{s}^{(l)} > 0$, it is sufficient that

$$\mathcal{C}_\beta^{(l)} = s_k^{(l)} = \frac{n}{l} + \left(\frac{\|\boldsymbol{\lambda}^{(l)}\|_1}{4l} - \frac{\boldsymbol{\lambda}_l}{4}\right)\frac{1}{\psi} \geq 0. \tag{33}$$



(a) Erdos-Reyni network (b) Barabasi network (c) Watts-Strogatz network

Fig. 1: Minimum cost and lower-bounds for β-disruptor on the synthetic networks



(a) Erdos-Reyni network (b) Barabasi network (c) Watts-Strogatz network

Fig. 2: Running time on the synthetic networks

Theorem 5 *The size of a β-edge disruptor is lower-bounded by*

$$\mathcal{D}_\beta = \min_{n \geq l \geq n^2/\Delta_\beta} \left\{ \mathcal{D}_\beta^{(l)} \mid \mathcal{C}_\beta^{(l)} > 0 \right\},$$

where $\mathcal{D}_\beta^{(l)}$ and $\mathcal{C}_\beta^{(l)}$ are given by Eqs. 32 and 33.

The steps to solve the relaxation of the QP is summarized in the Algorithm 2 (LMB Algorithm).

Time complexity. The LMB algorithm spends its major time on computing the eigenvalues. This can be done with Implicitly Restarted Lanczos Method which has worst-case time complexity $O(mKh + nK^2h + K^3h)$ where K is the number of eigenvalues to be computed, and h is the number of iterations for the eigenvalue algorithm to converge [37]. Given the eigenvalues, the rest of LMB takes only $O(n)$ time in the worst-case.

The number of required eigenvalues K is small in our algorithm. At beginning, the algorithm computes $t = \lceil 2/\beta \rceil$ smallest eigenvalues and the number of computed eigenvalues is double each time if necessary. In our experiments, the number of needed eigenvalues is $2/\beta$ in most cases. For example, to bound the number of necessary links whose removal disrupts 90% pairwise connectivity we only need to compute about 20 smallest eigenvalues of the Laplacian matrix (instead of computing approximately all 30,000 eigenvalues). We found the LMB algorithm to be scalable, taking *linear time* with respect to the number of nodes and edges.

5 Experimental Results

We run the exact solution and compute the spectral lower-bound for both synthetic and real-world networks.

5.1 Synthetic Networks

We generate the synthetic networks following well-known complex network models. All networks have 100 nodes and around 300 edges. The details of those networks are as follows.

- **Erdos-Reyni**: A random graph (the Erdos-Reyni model) [20].
- **Barabasi-Albert**: A power-law model using preferential attachment mechanism [6].
- **Small world**: A random graph following Watts and Strogatz model [36]. The dimension of the lattice is set to be 3 and the rewiring probability is 0.3.

We run our experiments on a PC with Intel Xeon 2.93 Ghz processor and 12 GB memory. The integer programming (IP) and the linear programming (LP) are solved with the mathematical optimization package GUROBI 4.5.

Table 1: Comparing MIP_d and the original IP in [4]. The first value in each cell associates with the MIP_d and the second associates with the IP.

	Erdos-Reyni	**Barabasi-Albert**	**Watts-Strogatz**
#Constraint	23K/485K	12K/485K	24K/485K
Time(s), $\beta = 0.95$	11.1/1091	2.6 / 421	100.6/2231

Comparing MIP_d with the IP. We compare the performance of our new MIP_d formulation with the original IP in [4]. The number of constraints (regardless of the value of β) are shown in Table 1. The MIP_d size is more than 20 times smaller than that of the IP. As a result of excessive large size, the IP cannot find the optimal solutions within 20000 seconds in the most cases with the exceptions of $\beta = 0.95$. The running time for $\beta = 0.95$, the last line in Table 1, show that the MIP_d is from 10 to 160 times faster than the IP. Therefore, our new MIP_d outperforms the existing IP formulation in [31, 32].

Comparing the lower-bounds with the optimal objectives. We compare the lower bounds given by our spectral technique and the LP with the optimal objective values. The optimal objective are found by solving our new MIP_d formulation with the cutting planes in Section 3.3. The results produced by ILB (the dynamic programming algorithm in Appendix A) and LMB algorithms (the Lagrange multiplier method in Section 4.2) are identical (after rounded up) and plotted under the same name "spectral bound".

The minimum number of links whose removal causes certain level of disruption, are shown in Fig. 1. For all three different networks, solving LP gives close lower-bounds on the minimum number of links to remove. The spectral bounds have smaller values than the LP bounds; however, the spectral bound approximates

closely the curvature of the LP bounds and the optimal solution, especially in power-law networks.

As shown in Fig. 2, there is a big gap between the running time of the spectral bound and those of LP and IP. Note that all the spectral bound are computed at once, i.e., the provided running time is the total running time over all different values of β. Even though the running time of the spectral bound is still thousand of times faster than LP and IP.

Overall, while IP is best used for small networks, and LP can be used for medium networks of few thousand nodes, the only feasible method to compute the lower-bound in large networks is the spectral bound. One of the attractive aspect of the LMB spectral bound, described in the Alg. 2, is that the algorithm can be easily implemented in a distributed manner. The most time-consuming part of the algorithm is to compute the few smallest eigenvalues. This can be done distributedly with the existing mathematical software [11].

Table 2: Sizes of the investigated networks and the corresponding running time to compute the lower-bound

	CAIDA AS	**Oregon AS**	**P2P Gnutella**
Vertices	8,020	11,174	22,663
Edges	36,406	23,410	109, 386
Time (s)	1530.1	321.0	207.9

5.2 Real-world Datasets

We compute the spectral lower-bounds for real networks and show the results in Fig. 3. Neither LP nor IP can run on these networks due to both time and memory limits. The studied networks are

- **Gnutella P2P**: Gnutella peer-to-peer network from from Aug. 25, 2002 [23]. Nodes represents hosts in the network and edges are the connections between the Gnutella hosts.
- **Oregon AS**: AS peering information inferred from Oregon route-views between Mar. 31 and May 26, 2001 [23].
- **CAIDA AS**: The CAIDA AS Relationships Datasets, from September 17, 2007 [23].

The lower-bounds in Fig. 3 indicates that it is difficult to destroy major connectivity in communication networks. For examples, even after removing 369 links at least 50% node pairs in the CAIDA AS network stay connected; and to bring down the connectivity level in the Gnutella P2P network to 15% one has to destroy at least 960 links. Due to low edge density, the Oregon AS network tends to be more vulnerable than the other two networks. Nevertheless, uterly disrupting the connectivity in the network to 5% level would require removing more than 763 links.

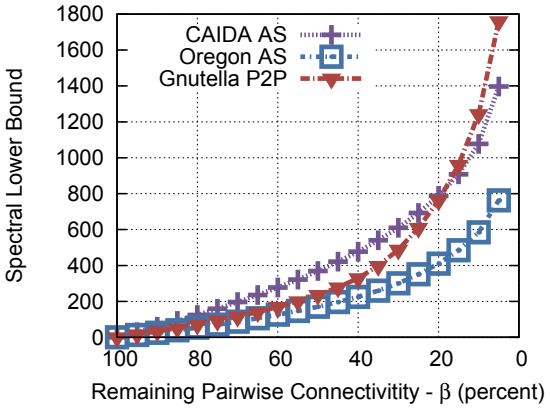


Fig. 3: Lower bounds on the number of link-attack for real networks found with the LMB algorithm.

6 Concluding Remarks

Assessing network for topological vulnerabilities is an important and challenging problem. We present in this paper an exact algorithm and a spectral lower-bound method for the link vulnerability assessment problem, called β-edge disruptor. The exact algorithm is several magnitude more efficient than the straightforward mathematical formulation; and the new lower-bound method is useful in both comparing the vulnerability of different networks and providing guarantees for other heuristics assessment methods. In addition, the Lagrange multiplier method to compute the lower-bound requires only a portion of the eigenvalues and is applicable for large-scale networks.

We also propose an efficient branch-and-cut algorithm to find exact solution for β-edge disruptor problem. The techniques used in the paper are general and can be easily adapted to solve many graph clustering and partitioning problems. Our future work will be combining the column generation technique (branch and price) to find exact solutions for even larger instances of networks.

Acknowledgment

This work is partially supported by NSF CAREER Award #0953284.

References

1. Addis, B., Summa, M.D., Grosso, A.: Identifying critical nodes in undirected graphs: Complexity results and polynomial algorithms for the case of bounded treewidth. Discrete Applied Mathematics **161**(1617), 2349 – 2360 (2013). DOI http://dx.doi.org/10.1016/j.dam.2013.03.021. URL `http://www.sciencedirect.com/science/article/pii/S0166218X13001686`
2. Albert, R., Albert, I., Nakarado, G.L.: Structural vulnerability of the north american power grid. Phys. Rev. E **69**(2), 10 (2004)

3. Albert, R., Jeong, H., Barabasi, A.: Error and attack tolerance of complex networks. Nature **406**(6794), 378–382 (2000). DOI 10.1038/35019019. URL http://dx.doi.org/10.1038/35019019
4. Arulselvan, A., Commander, C.W., Elefteriadou, L., Pardalos, P.M.: Detecting critical nodes in sparse graphs. Computers and Operations Research **36**(7) (2009). DOI http://dx.doi.org/10.1016/j.cor.2008.08.016
5. Banerjee, S., Shirazipourazad, S., Sen, A.: Design and analysis of networks with large components in presence of region-based faults. In: Communications (ICC), 2011 IEEE International Conference on, pp. 1–6 (2011). DOI 10.1109/icc.2011.5962427
6. Barabasi, A., Albert, R., Jeong, H.: Scale-free characteristics of random networks: the topology of the world-wide web. Physica A **281** (2000)
7. Bazgan, C., Toubaline, S., Vanderpooten, D.: Complexity of determining the most vital elements for the p-median and p-center location problems. Journal of Combinatorial Optimization **25**(2), 191–207 (2013). DOI 10.1007/s10878-012-9469-8. URL http://dx.doi.org/10.1007/s10878-012-9469-8
8. Bazgan, C., Toubaline, S., Vanderpooten, D.: Critical edges/nodes for the minimum spanning tree problem: complexity and approximation. Journal of Combinatorial Optimization **26**(1), 178–189 (2013). DOI 10.1007/s10878-011-9449-4. URL http://dx.doi.org/10.1007/s10878-011-9449-4
9. Bissias, G., Levine, B.N., Rosenberg, A.L.: Bounding Damage From Link Destruction with Application to the Internet (extended abstract). In: Proc. ACM SIGMETRICS, pp. 367—368 (2007). URL http://prisms.cs.umass.edu/brian/pubs/bissias.sigmetrics.abstract.2007.pdf
10. Bissias, G.D.: Bounds on service quality for networks subject to augmentation and attack. Ph.D. thesis, University of Massachusett Amherst (2010)
11. Blackford, L.S., Choi, J., Cleary, A., D'Azeuedo, E., Demmel, J., Dhillon, I., Hammarling, S., Henry, G., Petitet, A., Stanley, K., Walker, D., Whaley, R.C.: ScaLAPACK user's guide. SIAM (1997)
12. Borgatti, S.P.: Identifying sets of key players in a social network. Computational & Mathematical Organization Theory **12**(1), 21–34 (2006). DOI http://dx.doi.org/10.1007/s10588-006-7084-x
13. Boyd, S., Vandenberghe, L.: Convex optimization. Cambridge University Press (2004). URL http://books.google.com/books?id=mYm0bLd3fcoC
14. Chung, F.R.K.: Spectral Graph Theory (CBMS Regional Conference Series in Mathematics, No. 92). American Mathematical Society (1997). URL http://www.amazon.com/exec/obidos/redirect?tag=citeulike07-20\&path=ASIN/0821803158
15. Di Summa, M., Grosso, A., Locatelli, M.: Branch and cut algorithms for detecting critical nodes in undirected graphs. Computational Optimization and Applications **53**(3), 649–680 (2012). DOI 10.1007/s10589-012-9458-y. URL http://dx.doi.org/10.1007/s10589-012-9458-y
16. Dinh, T.N., Thai, M.T.: Precise structural vulnerability assessment via mathematical programming. In: Proc. of IEEE MILCOM (2011)
17. Dinh, T.N., Thai, M.T.: Network under joint node and link attacks: Vulnerability assessment methods and analysis. IEEE/ACM Trans. Netw. **99**, 00–12 (2014). DOI 10.1109/TNET.2011.2170849
18. Dinh, T.N., Xuan, Y., Thai, M.T., Park, E.K., Znati, T.: On approximation of new optimization methods for assessing network vulnerability. In: INFOCOM, pp. 2678–2686. IEEE Press, Piscataway, NJ, USA (2010)
19. Donath, W.E., Hoffman, A.J.: Lower bounds for the partitioning of graphs. IBM J. Res. Dev. **17** (1973). DOI http://dx.doi.org/10.1147/rd.175.0420. URL http://dx.doi.org/10.1147/rd.175.0420
20. Erdos, P., Renyi, A.: On the evolution of random graphs. Publ. Math. Inst. Hungary. Acad. Sci. **5**, 17–61 (1960)
21. Goldberg, A.V., Tarjan, R.E.: A new approach to the maximum flow problem. In: Proceedings of the eighteenth annual ACM symposium on Theory of computing, STOC '86, pp. 136–146. ACM, New York, NY, USA (1986). DOI http://doi.acm.org/10.1145/12130.12144. URL http://doi.acm.org/10.1145/12130.12144
22. Grubesic, T.H., Matisziw, T.C., Murray, A.T., Snediker, D.: Comparative approaches for assessing network vulnerability. Inter. Regional Sci. Review **31** (2008). DOI 10.1177/0160017607308679. URL http://dx.doi.org/10.1177/0160017607308679

23. Leskovec, J., Kleinberg, J., Faloutsos, C.: Graphs over time: densification laws, shrinking diameters and possible explanations. In: KDD, pp. 177–187. ACM (2005). DOI 10.1145/1081870.1081893
24. Matisziw, T.C., Murray, A.T.: Modeling s-t path availability to support disaster vulnerability assessment of network infrastructure. Comput. Oper. Res. **36**, 16–26 (2009). DOI 10.1016/j.cor.2007.09.004
25. Mohar, Poljak: Eigenvalue in combinatorial optimization. Combinatorial and Graph-Theoretical Problems in Linear Algebra (1992)
26. Murray, A.T., Matisziw, T.C., Grubesic, T.H.: A methodological overview of network vulnerability analysis. Growth and Change **39**(4), 573–592 (2008). DOI 10.1111/j.1468-2257.2008.00447.x. URL http://dx.doi.org/10.1111/j.1468-2257.2008.00447.x
27. Neumayer, S., Zussman, G., Cohen, R., Modiano, E.: Assessing the vulnerability of the fiber infrastructure to disasters. IEEE/ACM Trans. Netw. pp. 1610–1623 (2011)
28. Oosten, M., Rutten, J.H.G.C., Spieksma, F.C.R.: Disconnecting graphs by removing vertices: a polyhedral approach. Statistica Neerlandica **61**(1), 35–60 (2007). DOI 10.1111/j.1467-9574.2007.00350.x. URL http://dx.doi.org/10.1111/j.1467-9574.2007.00350.x
29. Sen, A., Murthy, S., Banerjee, S.: Region-based connectivity - a new paradigm for design of fault-tolerant networks. In: HPSR (2009)
30. Shen, S., Smith, J.C.: Polynomial-time algorithms for solving a class of critical node problems on trees and series-parallel graphs. Networks **60**(2), 103–119 (2012). DOI 10.1002/net.20464. URL http://dx.doi.org/10.1002/net.20464
31. Shen, Y., Dinh, T., Thai, M.: Adaptive algorithms for detecting critical links and nodes in dynamic networks. In: MILITARY COMMUNICATIONS CONFERENCE, 2012 - MILCOM 2012, pp. 1–6 (2012). DOI 10.1109/MILCOM.2012.6415629
32. Shen, Y., Nguyen, N.P., Xuan, Y., Thai, M.T.: On the discovery of critical links and nodes for assessing network vulnerability. IEEE/ACM Trans. Netw. **21**(3), 963–973 (2013). DOI 10.1109/TNET.2012.2215882. URL http://dx.doi.org/10.1109/TNET.2012.2215882
33. Su, B., Xu, Q., Xiao, P.: Finding the anti-block vital edge of a shortest path between two nodes. Journal of Combinatorial Optimization **16**(2), 173–181 (2008). DOI 10.1007/s10878-007-9120-2. URL http://dx.doi.org/10.1007/s10878-007-9120-2
34. Ventresca, M., Aleman, D.: A derandomized approximation algorithm for the critical node detection problem. Computers & Operations Research **43**(0), 261 – 270 (2014). DOI http://dx.doi.org/10.1016/j.cor.2013.09.012. URL http://www.sciencedirect.com/science/article/pii/S0305054813002827
35. Veremyev, A., Boginski, V., Pasiliao, E.: Exact identification of critical nodes in sparse networks via new compact formulations. Optimization Letters **8**(4), 1245–1259 (2014). DOI 10.1007/s11590-013-0666-x. URL http://dx.doi.org/10.1007/s11590-013-0666-x
36. Watts, D.J., Strogatz, S.H.: Collective dynamics of 'small-world' networks. Nature **393**(6684), 440–442 (1998). DOI 10.1038/30918. URL http://dx.doi.org/10.1038/30918
37. White, S., Smyth, P.: A spectral clustering approach to finding communities in graph. In: SDM (2005)

A Appendix

A.1 Proof of Lemma 2 from [10]

Let $\boldsymbol{H}$ be an $n \times k$ indicator matrix where

$$H_{ij} = \begin{cases} 1, \text{ vertex } i \text{ in } j^{th}\text{subset (or component)} \\ 0, \qquad\qquad\qquad \text{otherwise} \end{cases}$$

and $\boldsymbol{h_j}$ denote the j^{th} column of $\boldsymbol{H}$, i.e., the membership vector for the j^{th} subset.

We have

$$\boldsymbol{h_j}^T \times \boldsymbol{h_j} = s_j,$$

and the total weights of edges going out of the j^{th} subset is given by

$$\boldsymbol{h_j}^T \boldsymbol{L} \boldsymbol{h_j}.$$

Therefore

$$E_{cut} = \frac{1}{2}\sum_{j=1}^{k} \boldsymbol{h_j}^T \boldsymbol{L}\boldsymbol{h_j}.$$

Since $\boldsymbol{L}$ is symmetric and positive semidefinite, it has n real non-negative eigenvalues $\lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$,and the corresponding eigenvectors $u_1, u_2, \ldots, u_n$ form a complete orthonormal basis, i.e., $\boldsymbol{L} = \sum_{i=1}^{n} \lambda_i \boldsymbol{u_i}\boldsymbol{u_i}^T$.

Hence

$$E_{cut} = \frac{1}{2}\sum_{j=1}^{k} \boldsymbol{h_j}^T \left(\sum_{i=1}^{n} \lambda_i \boldsymbol{u_i}\boldsymbol{u_i}^T\right) \boldsymbol{h_j} \tag{34}$$

$$= \frac{1}{2}\sum_{j=1}^{k}\sum_{i=1}^{n} \lambda_i \left(\boldsymbol{u_i}^T \boldsymbol{h_j}\right)^2 \tag{35}$$

Let $x_{ij} = \left(\boldsymbol{u_i}^T \boldsymbol{h_j}\right)^2 / m_j$. Substituting x_{ij}, we have

$$E_{cut} = \frac{1}{2}\sum_{j=1}^{k} m_j \sum_{i=1}^{n} \lambda_i x_{ij} \tag{36}$$

We can verify that

$$\sum_{i=1}^{n} x_{ij} = 1 \text{ and } \sum_{j=1}^{k} x_{ij} \leq 1$$

Since λ_1 is the smallest and m_1 is the largest, the equation (36) is minimized when $x_{ii} = 1$ and $x_{ij} = 0\ \forall i \neq j$, i.e.,

$$E_{cut} = \frac{1}{2}\sum_{j=1}^{k}\sum_{i=1}^{n} \lambda_i \left(\boldsymbol{u_i}^T \boldsymbol{h_j}\right)^2 \geq \frac{1}{2}\sum_{j=1}^{k} \lambda_j m_j.$$

A.2 Dynamic Programming Algorithm (ILB)

We present a dynamic programming algorithm to compute the exact solution of the QP(14-17).

For $k \leq l \leq n$ and $p \leq \beta\binom{n}{2}$, define $\mathcal{L}_k(l,p)$ to be the minimum spectral bound obtained by first k subsets that the total sizes is l and the total pairwise connectivity is at most p. That is

$$\mathcal{L}_k(l,p) = \min_{\boldsymbol{s}^{(k)} \in \mathbb{N}^k} \left\{ \boldsymbol{s}^{(k)T} \boldsymbol{\lambda}^{(k)} \ : \ \|\boldsymbol{s}^{(k)}\|_1 = l,\ \sum_{i=1}^{k} \binom{s_i}{2} \leq p \right\},$$

Then the optimal objective value QP(14-17) shall be given by $Q_\beta = \mathcal{L}_n(n, \beta\binom{n}{2})$.

By Lemma 3, we pay attention only to partitions satisfying $s_1 \geq s_2 \geq \ldots \geq s_n$. We now derive the recursive formula for $\mathcal{L}_p(l,k)$ based on the sub-optimal structure of the QP problem. Consider two possible cases of s_k

Algorithm 3: ILB(G, β)

Compute $\lambda_1, \ldots, \lambda_n$

$\mathcal{L}_k(l,p) = \begin{cases} +\infty, & \text{if } p < p_{\min}(l,k) \\ \lambda_1 l = 0, & \text{if } p \geq p_{\max}(l,k) \end{cases}$

for $k = 1$ **to** n

  for $l = 1$ **to** n

    for $p = p_{\min}(l,k)$ **to** $\min\left\{\beta\binom{n}{2}, p_{\max}(l,k)\right\}$

      $\mathcal{L}_k(l,p) = \min\left\{\begin{array}{l} \mathcal{L}_{k-1}(l,p), \\ \mathcal{L}_k(l-k, p-l+k) + \sum_{i=1}^{k} \lambda_i \end{array}\right\}$

  if $\mathcal{L}_{k-1}\left(n, \beta\binom{n}{2}\right) = \mathcal{L}_k\left(n, \beta\binom{n}{2}\right)$

    return $\lceil \mathcal{L}_k\left(n, \beta\binom{n}{2}\right)\rceil$

return $\lceil \mathcal{L}_n\left(n, \beta\binom{n}{2}\right)\rceil$

– $s_k = 0$: There are at most $k-1$ partitions whose sizes sum up to l. Hence, for this case $\mathcal{L}(l,k) = \mathcal{L}_{k-1}(l,p)$.

– $s_k > 0$: Since $s_1 \geq s_2 \geq \ldots \geq s_k > 0$. Let $\tilde{s}_i = s_i - 1 \geq 0$, the vector $\tilde{s} = \{\tilde{s}_1, \tilde{s}_2, \ldots, \tilde{s}_k\}$ satisfies simultaneously the following

$$\begin{aligned} \sum_{i=1}^{k} \lambda_i \tilde{s}_i &= \sum_{i=1}^{k} \lambda_i s_i - \sum_{i=1}^{k} \lambda_i \\ \sum_{i=1}^{k} \tilde{s}_i &= \sum_{i=1}^{k} s_i - k = l - k \\ \sum_{i=1}^{k} \binom{\tilde{s}_i}{2} &= \sum_{i=1}^{k} \left[\binom{s_i}{2} - s_i + 1\right] \\ &= \sum_{i=1}^{k} \binom{s_i}{2} - l + k \leq p - l + k \end{aligned}$$

Therefore, in this case $\mathcal{L}_k(l,p) = \mathcal{L}_k(l-k, p-l+k) + \sum_{i=1}^{k} \lambda_i$

In summary, we have

$$\mathcal{L}_k(l,p) = \min\left\{\begin{array}{l} \mathcal{L}_{k-1}(l,p), \\ \mathcal{L}_k(l-k, p-l+k) + \sum_{i=1}^{k} \lambda_i \end{array}\right\}$$

We compute value of $\mathcal{L}_p(l,k)$ in increasing order of p and l but in decreasing order of k. The base cases for $\mathcal{L}_p(l,k)$ are as follow.

$$\mathcal{L}_k(l,p) = \begin{cases} +\infty, & \text{if } p < p_{\min}(l,k) \\ \lambda_1 l = 0, & \text{if } p \geq p_{\max}(l,k) \end{cases} \tag{37}$$

where $p_{\min}(l,k) = \binom{\lceil l/k \rceil}{2}(l \bmod k) + \binom{\lfloor l/k \rfloor}{2}(k - l \bmod k)$ and $p_{\max}(l,k) = \binom{l}{2}$ that are the minimum and maximum pairwise connectivity of a graph with l vertices and k connected components, respectively.