Improving Network Connectivity Using Trusted Nodes and Edges

Waseem Abbas, Aron Laszka, Yevgeniy Vorobeychik, and Xenofon Koutsoukos

***Abstract*— Network connectivity is a primary attribute and a characteristic phenomenon of any networked system. A high connectivity is often desired within networks; for instance to increase robustness to failures, and resilience against attacks. A typical approach to increasing network connectivity is to strategically add links; however adding links is not always the most suitable option. In this paper, we propose an alternative approach to improving network connectivity, that is by making a small subset of nodes and edges "trusted," which means that such nodes and edges remain intact at all times and are insusceptible to failures. We then show that by controlling the number of trusted nodes and edges, any desired level of network connectivity can be obtained. Along with characterizing network connectivity with trusted nodes and edges, we present heuristics to compute a small number of such nodes and edges. Finally, we illustrate our results on various networks.**

I. Introduction

Connectedness among various components is a central theme and a characteristic phenomenon in any networked system. It signifies a large number of other network attributes, such as network reliability, robustness to failures, resilience against attacks, and other structural properties of the network. As a result many notions defining connectedness have been proposed. In particular, network connectivity in terms of *vertex* and *edge connectivities* are one of the most widely considered concepts with significant ramifications. In essence, network connectivity captures the ability of the network to retain a connection between any two nodes under a certain number of node or edge failures.

A higher network connectivity is desirable in general to ensure that any two nodes remain connected even under a large number of node or edge failures. Not only for structural robustness, highly connected networks are more suitable to achieve various other network objectives such as information spread, resilient consensus [1], and survivable network designs under node or edge failures [2]. Network connectivity can be improved by strategically adding links (edges) between nodes, i.e., by incorporating *redundancy*, a technique commonly referred to as the *connectivity augmentation* [3], [4]. Although effective, improving network connectivity by adding edges is not always practical due to economic and feasibility issues. Moreover, adding more links also increases the attack surface for the potential attackers to cause edge failures. These concerns inspire us to devise an alternative strategy to improve network connectivity.

Here, we propose an alternative approach that does not involve strategic addition of edges or links to improve network connectivity. The basic idea is to ensure the availability and operational integrity of a very small subset of nodes and edges at all times by protecting them from failures/attacks. We call such a subset of nodes and edges *trusted*. They correspond to the network components that are more reliable owing to more resources and higher security measures. For instance, in terms of protection against physical attacks, defense mechanisms against jamming, protected memory, and sophisticated authentication mechanisms. We then show that the overall network connectivity is significantly improved by making a very small subset of nodes and edges trusted, even if the original network is sparse. Moreover, by controlling the number of trusted nodes, any desired level of connectivity could be obtained. Thus, instead of the idea of *redundancy to improve connectivity*, we exploit the notion of *reliability (trustedness) of a small subnetwork to improve connectivity*.

Our main contributions are as follows: 1) We propose and characterize the notion of network connectivity using trusted nodes and edges, and show that network connectivity can be significantly improved by selecting a small subset of trusted nodes or edges. 2) We provide ways to compute network connectivity in the presence of trusted nodes and edges, and also give heuristics to select the minimum set of trusted nodes and edges to achieve any desired value of network connectivity. 3) Finally, we evaluate our results numerically on various networks including Erdős-Rényi, preferential attachment, and random geometric networks.

The problems related to adding the minimum number of edges to attain the desired network connectivity are referred to as the *connectivity augmentation* problems. The issue was investigated in detail for the first time in [4]. Since then, new techniques have been developed to optimally increase the vertex and edge connectivities (e.g., [5], [6], [7]). For a comprehensive list of papers in the area, we refer readers to an earlier survey by Frank [8] and Chapter 8 in a more recent work of Nagamochi and Ibaraki [3]. A closely related field is the area of *survivable network design*, in which the basic premise is to design and analyze algorithms that exploit structural properties of the network to make them survive under failure of network components (e.g., [2], [9]). To improve structural properties of networks, adding protected resources or defending a subset of network components has been a useful approach. Similar to *trusted nodes*, the notion of *anchor nodes* was used in [10] in the context of maximizing the size of a special structure, known as the k*-core*, in graphs. In a recent work, Dziubiński and Goyal [11] explore trade-offs between the cost of adding links and defending nodes against attacks for network connectivity.

II. Preliminaries

Let $\mathcal{G}(\mathcal{V}, \mathcal{E})$ be an *undirected graph* with a vertex set $\mathcal{V}$ and edge set $\mathcal{E}$. An edge between vertices u and v is denoted by

uv, and vertices u and v are called neighbors of each other. The *neighborhood* of a vertex u, denoted by $\mathcal{N}_u$, is the set of all neighbors of u. The neighborhood of a subset of vertices $U \subseteq \mathcal{V}$ is $\mathcal{N}_U = \bigcup_{u \in U} \mathcal{N}_u$. A *path* $\mathcal{P}$ of length n is a non-empty graph with the vertex set $\mathcal{V} = \{u_0, u_2, \cdots, u_n\}$, and the edge set $\{u_0u_1, u_1u_2, \cdots, u_{n-1}u_n\}$. The vertices u_0 and u_n are the *end vertices*, whereas, all remaining vertices are the *inner vertices* of the path. In a graph $\mathcal{G}$, two paths are *vertex-independent* if they do not have any common inner vertex. Similarly, two paths are *edge-independent* if they do not have any common edge. We use the terms vertex and *node* interchangeably throughout the paper.

If W is a subset of vertices (edges), then $\mathcal{G} \setminus W$ is the subgraph induced by the remaining vertices and edges of $\mathcal{G}$. If $\mathcal{G} \setminus W$ has at least two components, then W *separates* $\mathcal{G}$. Similarly, if u and v belong to two different components, then W *separates* u and v. Such a set W is referred to as the *vertex cut (edge cut)*. Next, we state the definitions of vertex and edge connectivity.

A graph is *k-vertex connected* if there does not exist a set of $k-1$ vertices whose removal disconnects the graph. Likewise, the graph is *k-edge connected* if there is no subset of $k-1$ edges whose removal disconnects the graph. Vertex connectivity of $\mathcal{G}$, denoted by $\kappa(\mathcal{G})$, is the maximum value of k for which $\mathcal{G}$ is k-vertex connected. Similarly, the maximum value of k for which $\mathcal{G}$ is k-edge connected is the edge connectivity of $\mathcal{G}$, denoted by $\lambda(\mathcal{G})$. The vertex connectivity of a complete graph with n nodes is defined to be $n-1$ although no vertex cut exists. A classical theorem of Menger relates the notion of connectivity to the number of vertex and edge-independent paths between any two nodes.

Theorem 2.1: (*Menger's theorem* [12])

- Let u and v be distinct, non-adjacent vertices of $\mathcal{G}$. The minimum size of a vertex cut separating u and v is equal to the maximum number of vertex-independent paths between u and v.
- Let u and v be distinct vertices of $\mathcal{G}$. The minimum size of an edge cut that separates u and v is equal to the maximum number of edge-independent paths between u and v.

As a consequence, for any $k \geq 2$, a graph is k-vertex connected (k-edge connected) if and only if any two vertices have k vertex-independent (k edge-independent) paths between them. Several versions and proofs of this fundamental results have been reported (e.g., see [13]).

III. Connectivity with Trusted Nodes and Edges

In the traditional k-vertex (k-edge) connectivity notion, the idea is to ensure that the graph remains connected if *any* $k-1$ nodes ($k-1$ edges) are removed from the network. By adding more edges between nodes, the vertex and edge connectivities can be improved. However, if we fix a small subset of nodes (edges) such that they cannot be removed from the network, then the minimum number of nodes (edges) from the *remaining* set that are required to disconnect the network also increases. Thus, instead of adding more edges or links, we get an alternative way to improve network connectivity. A merit of this approach is that by making only a very small fraction of the overall nodes (edges) insusceptible to removals, or as we call *trusted*, the overall node (edge) connectivity can be significantly improved. In practice, trustedness can be achieved by making such components more resourceful and highly secure against physical attacks, tampering, and malicious intrusions through sophisticated security mechanisms. Next, we define the new notion of connectivity as follows:

Definition *(k-Vertex Connected with $\mathcal{T}_v$)* – An undirected graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ is said to be k-vertex connected with $\mathcal{T}_v \subseteq \mathcal{V}$, if there does not exist a set of fewer than k vertices in $\mathcal{V} \setminus \mathcal{T}_v$ whose removal disconnects the graph. The maximum value of k for which the graph is k-vertex connected with $\mathcal{T}_v$ is denoted by $\kappa_{\mathcal{T}}(\mathcal{G})$ and is referred to as the *vertex-connectivity with $\mathcal{T}_v$*.

Similarly, we can define the k-edge connectivity with trusted edges $\mathcal{T}_e$.

Definition *(k-Edge Connected with $\mathcal{T}_e$)* – For a given $\mathcal{T}_e \subseteq \mathcal{E}$, $\mathcal{G}(\mathcal{V}, \mathcal{E})$ is said to be k-edge connected with $\mathcal{T}_e$, if there does not exist a set of fewer than k edges in $\mathcal{E} \setminus \mathcal{T}_e$ whose removal disconnects the graph. The maximum value of k for which the graph is k-edge connected with $\mathcal{T}_e$ is called the *edge-connectivity with $\mathcal{T}_e$*, and is denoted by $\lambda_{\mathcal{T}}(\mathcal{G})$.

Analogous to the node-independent paths, we define the notion of *node-independent paths with $\mathcal{T}_v$* as follows: If $\mathcal{T}_v$ is the set of trusted nodes, then two paths are node-independent with $\mathcal{T}_v$ if the common inner vertices are only the trusted nodes. Similarly, for a given set of trusted edges $\mathcal{T}_e$, two paths are *edge-independent with $\mathcal{T}_e$* if their common edges are only the trusted edges. For instance, in Figure 1(a), paths $\{u_1u_2, u_2x, xu_3\}$ and $\{v_1v_2, v_2x, xv_3\}$ are node-independent with $\mathcal{T}_v = \{x\}$, and paths $\{u_1u_2, u_2x, xy, yu_3\}$ and $\{v_1v_2, v_2x, xy, yv_3\}$ in Figure 1(b) are edge-independent with $\mathcal{T}_e = \{xy\}$.

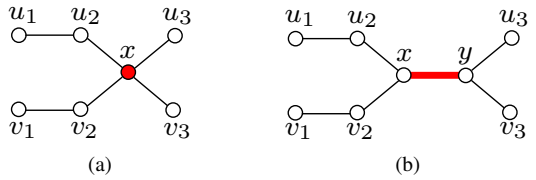


Fig. 1. (a) Node-independent paths with $\mathcal{T}_v$. (b) Edge-independent paths with $\mathcal{T}_e$.

A path with all trusted nodes (edges) is referred to as the *node-trusted* (*edge-trusted*) *path*. If for any pair of nodes, there exists a node trusted path between them, then $\mathcal{G}$ is referred to as *completely vertex-connected with $\mathcal{T}_v$*. Similarly, $\mathcal{G}$ is *completely edge-connected with $\mathcal{T}_e$* if there exists an edge-trusted path between any two nodes. If a graph is *completely vertex connected with $\mathcal{T}_v$*, we define its $\kappa_{\mathcal{T}} = \infty$, and if the graph is *completely edge-connected with $\mathcal{T}_e$*, then we define its $\lambda_{\mathcal{T}} = \infty$.

A. Vertex Connectivity with Trusted Nodes $\mathcal{T}_v$

Next, we compute and relate vertex connectivity with trusted nodes to the traditional notion of vertex connectivity. Let $\mathcal{G}'(\mathcal{V}, \mathcal{E}')$ be a graph obtained from $\mathcal{G}(\mathcal{V}, \mathcal{E})$ as follows: for every non-adjacent pair of nodes u and v, if there exists a trusted node, or a node-trusted path between them, then add an edge $\{uv\}$. An example is shown in Figure 2, where $\{u, v, z\}$ is the set of trusted nodes in $\mathcal{G}$. Since nodes u and v induce a node-trusted path in $\mathcal{G}$, all neighbors of u and v are pair-wise adjacent in $\mathcal{G}'$. Similarly, neighbors of trusted node z are adjacent in $\mathcal{G}'$.

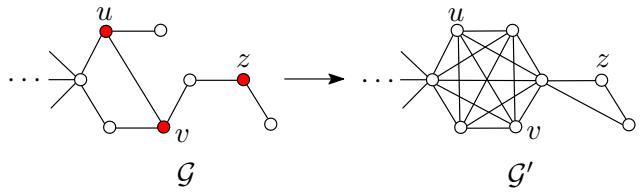


Fig. 2. $\mathcal{G}$ with the set of trusted nodes $\{u, v, z\}$ and the resulting $\mathcal{G}'$.

Proposition 3.1: Let $\mathcal{G}(\mathcal{V}, \mathcal{E})$ be a graph that is not completely vertex connected with $\mathcal{T}_v$, then $\kappa_{\mathcal{T}}(\mathcal{G}) = \kappa(\mathcal{G}')$.

Proof: Let $\kappa_{\mathcal{T}}(\mathcal{G}) = k$. If nodes u and v are connected trough a node-trusted path in $\mathcal{G}$, then u and v are adjacent in $\mathcal{G}'$. Thus, if there is no subset of $k-1$ non-trusted nodes in $\mathcal{G}$ whose removal disconnects the graph, then there is no subset of *any* $k-1$ nodes in $\mathcal{G}'$ whose removal disconnects $\mathcal{G}'$. Similarly, we observe that every vertex-cut in $\mathcal{G}$ consisting of only non-trusted nodes is also a vertex-cut in $\mathcal{G}'$. ■

A direct consequence of the above proposition and Theorem 2.1 is the following Menger's type result.

Corollary 3.2: For a graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ and $\mathcal{T}_v \subseteq \mathcal{V}$, the following statements are equivalent:

1) $\mathcal{G}$ is k-vertex connected with $\mathcal{T}_v$.
2) For any two distinct, non-adjacent vertices $u, v \in \mathcal{V}$; either there exists a node-trusted path between u and v, or there exists at least k paths between u and v that are vertex-independent with $\mathcal{T}_v$.

As an example, consider the 2-vertex connected graph in Figure 3(a), which becomes 4-vertex connected with two trusted nodes $\mathcal{T}_v = \{6, 10\}$ as shown in Figure 3(b).

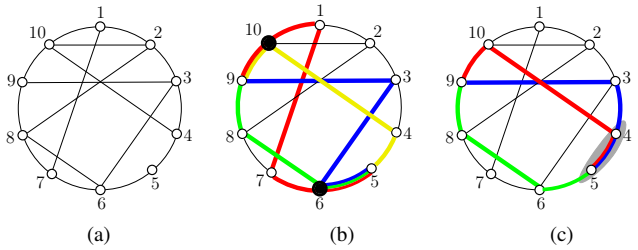


Fig. 3. (a) Graph is 2-vertex and 2-edge connected. (b) The graph becomes 4-vertex connected with $\mathcal{T}_v = \{6, 10\}$. Between nodes 5 and 9, there are four vertex-independent paths with $\mathcal{T}_v$, shown in red, blue, green and yellow. (c) The graph is 3-edge connected with $\mathcal{T}_e = \{45\}$. Three edge-independent paths with $\mathcal{T}_e$ between nodes 5 and 9 are highlighted

To compute the vertex connectivity of $\mathcal{G}$ with $\mathcal{T}_v$, we first obtain $\mathcal{G}'$ as above, and then can use any algorithm to compute the vertex connectivity of $\mathcal{G}'$. There is an extensive literature on such algorithms [14]. A typical approach is to utilize the *max-flow-min-cut* theorem (e.g., see [3]). The idea is to compute the *local* vertex connectivity between any two non-adjacent nodes $s, t \in \mathcal{V}$ in $\mathcal{G}'$, which is the minimum number of nodes that need to be removed to disconnect s and t. The local vertex connectivity is equal to the maximum flow between s and t in a particular directed network $\mathcal{H}_{\mathcal{G}'}$ obtained from $\mathcal{G}'$ (e.g., see [15]). The vertex connectivity of $\mathcal{G}'$, and hence the vertex connectivity of $\mathcal{G}$ with $\mathcal{T}_v$, is simply the minimum of the local vertex connectivities between any two non-adjacent nodes in $\mathcal{G}'$.

The directed network $\mathcal{H}_{\mathcal{G}'}$ required to compute local vertex connectivity between non-adjacent nodes s and t is obtained from $\mathcal{G}'$ as follows [15]: For each undirected edge uv, create two directed edges; one from u to v, and the other from v to u. Remove all edges incoming to s and outgoing from t. For each node $u \in \mathcal{V} \setminus \{s, t\}$, create two nodes u_i and u_o, and add a directed edge from u_i to u_o. Next, make all incoming edges to u as the incoming edges to u_i, and make all edges leaving from u as the outgoing edges from u_o. Finally, assign a *unit* weight (capacity) to each directed edge and compute the maximum flow between s and t. In summary, to compute the vertex connectivity of $\mathcal{G}$ with a given $\mathcal{T}_v$, following steps are performed,

$$\mathcal{G} \longrightarrow \mathcal{G}' \xrightarrow{s,t \in \mathcal{V}} \mathcal{H}_{\mathcal{G}'} \longrightarrow \texttt{max\_flow(s,t)}. \qquad (1)$$

$\kappa_{\mathcal{T}}(\mathcal{G})$ is the minimum of the maximum flow between any two non-adjacent nodes s, t in $\mathcal{G}'$.

In fact, the vertex connectivity of $\mathcal{G}$ with $\mathcal{T}_v$ can be computed by directly obtaining a directed network $\mathcal{H}_{\mathcal{G}}$ from $\mathcal{G}$ as described above. In other words, the $\mathcal{G} \rightarrow \mathcal{G}'$ transformation in (1) can be avoided. The only difference in the construction of $\mathcal{H}_{\mathcal{G}}$ from above would be that the edge weights (capacities) of all the edges incoming to and outgoing from the nodes corresponding to the trusted nodes in $\mathcal{G}$ would be *infinite* (i.e., very large), rather than one. The edge weights of all other edges would remain to be one. To compute the vertex connectivity with $\mathcal{T}_v$, maximum flow between all non adjacent nodes in $\mathcal{G}$ is then computed over $\mathcal{H}_{\mathcal{G}}$. An example is shown in Figure 4.

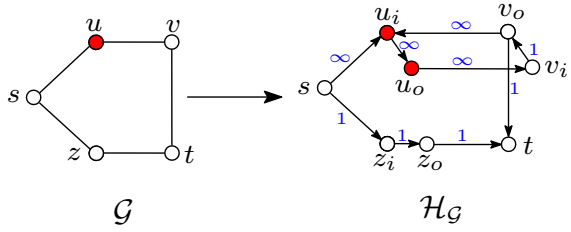


Fig. 4. For a given $\mathcal{G}$, $\mathcal{T}_v = \{u\}$, and non-adjacent nodes s, t, the construction of a directed network $\mathcal{H}_{\mathcal{G}}$

B. Edge Connectivity with Trusted Edges $\mathcal{T}_e$

Here, we discuss the case of edge-connectivity with trusted edges, and ways to compute it. Let $\mathcal{G}(\mathcal{V}, \mathcal{E})$ be a graph with

trusted edges $\mathcal{T}_e \subseteq \mathcal{E}$, then we define $\tilde{\mathcal{G}}(\tilde{\mathcal{V}}, \tilde{\mathcal{E}})$ be the graph obtained by identifying the end nodes of each trusted edge. Here, identifying two nodes, say u_1 and u_2, means replacing them by a single node u such that an edge exists between some node x and u in the new graph if and only if an edge exists between x and u_1 or between x and u_2 in the original graph. We obtain $\tilde{\mathcal{G}}$ from $\mathcal{G}$ by identifying the end nodes of all trusted edges one by one. Note that $\tilde{\mathcal{G}}$ might have multiple edges between nodes, and hence $\tilde{\mathcal{G}}$ is a *multigraph*. All self loops are ignored. An example is illustrated in Figure 5.

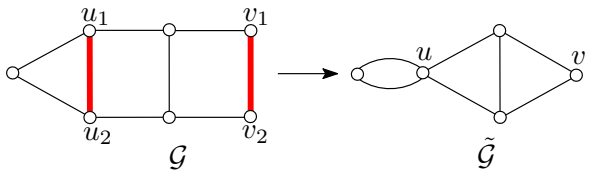


Fig. 5. The end nodes of trusted edges $\{u_1u_2\}$ and $\{v_1v_2\}$ in $\mathcal{G}$ are identified as u and v respectively into $\tilde{\mathcal{G}}$.

We observe that edge-trusted paths exist between nodes in $\mathcal{G}$ that are identified as a single node in $\tilde{\mathcal{G}}$. Thus, if $|\tilde{\mathcal{V}}| = 1$, it implies that $\mathcal{G}$ is *completely edge-connected with* $\mathcal{T}_e$. Since the end nodes of all trusted edges are identified in $\tilde{\mathcal{G}}$, thus, if there is no set of $k-1$ non-trusted edges that disconnects $\mathcal{G}$, then there is no set of *any* $k-1$ edges in $\tilde{\mathcal{G}}$ whose removal disconnects $\tilde{\mathcal{G}}$. Similarly, it is easy to observe that any edge-cut of size k in $\mathcal{G}$ is also an edge-cut in $\tilde{\mathcal{G}}$. Thus, we get the following:

Proposition 3.3: Let $\mathcal{G}(\mathcal{V}, \mathcal{E})$ be a graph that is not completely edge-connected with $\mathcal{T}_e$, then $\lambda_{\mathcal{T}}(\mathcal{G}) = \lambda(\tilde{\mathcal{G}})$.

Given a multigraph $\tilde{\mathcal{G}}$, if we replace each (edge in a) multiple edge by a path of length two, then there is a one-to-one relation between the paths in $\tilde{\mathcal{G}}$ and the resulting graph. Thus, the result in the Menger's theorem is immediately applicable to the resulting graph. We can state the following.

Proposition 3.4: The edge-version of the Menger's theorem holds for multigraphs.

As a result, for the notion of edge connectivity with $\mathcal{T}_e$, the following Menger's type result is directly obtained.

Corollary 3.5: For a graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ and $\mathcal{T}_e \subseteq \mathcal{E}$, the following statements are equivalent:

1) $\mathcal{G}$ is k-edge connected with $\mathcal{T}_e$.
2) For any two distinct vertices u and v in $\mathcal{G}$; either there exists an edge-trusted path between them, or there are at least k paths between them that are edge-independent with $\mathcal{T}_e$.

As an example, consider the 2-edge connected graph in Figure 3(a), which becomes 3-edge connected with a single trusted edge as shown in Figure 3(c).

To compute the edge connectivity of $\mathcal{G}$ with $\mathcal{T}_e$, we can first obtain the multigraph $\tilde{\mathcal{G}}$ and can then compute the edge connectivity of $\tilde{\mathcal{G}}$ using known methods, for instance [3], [16], [17]. At the same time, as with the vertex connectivity with $\mathcal{T}_v$, we can directly compute the edge connectivity of $\mathcal{G}$ with $\mathcal{T}_e$ by first obtaining a directed graph $\mathcal{D}_{\mathcal{G}}$ with appropriate edge weights. $\mathcal{D}_{\mathcal{G}}$ is obtained from $\mathcal{G}$ by replacing each undirected edge uv in $\mathcal{G}$ by two directed edges, one from u to v and the other from v to u. All the directed edges corresponding to the trusted edges in $\mathcal{G}$ are assigned infinite weights (capacities), whereas all remaining directed edges are assigned unit weight. The edge connectivity of $\mathcal{G}$ with $\mathcal{T}_e$ is simply the minimum of the maximum flow between any two nodes in $\mathcal{D}_{\mathcal{G}}$. It is sufficient to compute the minimum of the maximum flow between a fixed node u (randomly picked) and all other nodes [14].

IV. Computing Trusted Nodes and Edges

In this section, we present heuristics to select minimum set of trusted nodes $\mathcal{T}_v$ (trusted edges $\mathcal{T}_e$) to achieve the desired vertex connectivity with $\mathcal{T}_v$ (edge connectivity with $\mathcal{T}_e$).

A. Problem Complexity

First, we show that finding a minimum set of trusted nodes achieving a certain connectivity is a computationally hard problem. We begin by formulating this as a decision problem.

Definition (*Trusted Node-Connectivity Augmentation Problem (TNCAP)*) Given a graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, a desired connectivity k', and the number of trusted nodes T, determine if there exists a set of trusted nodes $\mathcal{T}_v$ of cardinality T such that $\mathcal{G}$ is k'-connected with $\mathcal{T}_v$.

Theorem 4.1: TNCAP is NP-hard.

We show that TNCAP is NP-hard using a reduction from a well-known NP-hard problem, the Set Cover Problem (SCP).

Set Cover Problem (SCP) Given a base set U, a family $\mathcal{F}$ of subsets of U, and a threshold size t, determine if there exists a subfamily $\mathcal{C} \subseteq \mathcal{F}$ of cardinality t whose union is U.

Proof: Given an instance of SCP, we construct an instance of TNCAP as follows:

- For each element $u \in U$, create a node u. Similarly, for each member F of the family $\mathcal{F}$, create a node F.
- For each $u \in U$ and $F \in \mathcal{F}$, create an edge (u, F) if $u \in F$.
- For each $F_1, F_2 \in \mathcal{F}$, $F_1 \neq F_2$, create an edge (F_1, F_2).
- Let number of trusted nodes be $T = t$, and let the desired connectivity be $k' = |\mathcal{F}|$.

It is clear that the reduction can be performed in polynomial time. As a consequence, we only need to show that TNCAP has a solution if and only if SCP does.

First, let us suppose that there exists a set cover $\mathcal{C}$ of cardinality t. Then, let the set of trusted nodes be $\mathcal{T}_v = \mathcal{C}$. Since $\mathcal{C}$ is a set cover of U, every node corresponding to an element of U is connected to a trusted node in $\mathcal{T}_v$. Further, every node corresponding to a member of $\mathcal{F} \setminus \mathcal{C}$ is also connected to a trusted node in $\mathcal{T}_v$, and the trusted nodes are connected to each other. Consequently, $\mathcal{G}$ cannot be separated by the removal of any set of non-trusted nodes, which proves that $\mathcal{T}_v = \mathcal{C}$ is a solution for TNCAP.

Second, let us suppose that there does not exist a set cover of cardinality t. Now, we will show that the graph $\mathcal{G}$ cannot be k'-connected with any set of trusted nodes $\mathcal{T}_v$ of cardinality $T = t$. Let $\mathcal{T}_v$ be an arbitrary set of trusted

nodes of cardinality T, and consider the removal of all non-trusted nodes corresponding to members of $\mathcal{F}$. Since $\mathcal{T}_v \cap \mathcal{F}$ cannot be a set cover due to our supposition, there exists an element $u \in U$ that is connected only to non-trusted nodes. Consequently, the removal of the non-trusted nodes corresponding to members of $\mathcal{F}$ separates u from the remainder of the graph, which proves that $\mathcal{T}_v$ cannot be a solution for TNCAP. ■

B. Heuristics

In a graph, complete connectivity with $\mathcal{T}_v$ is obtained whenever any two nodes are connected through a node-trusted path between them. A node trusted path exists between any pair of nodes if and only if $\mathcal{T}_v$ is a *connected dominating set*, which is defined as

Definition (*Connected Dominating Set*) In a graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, a subset of nodes $\Gamma \subseteq \mathcal{V}$ is a connected dominating set if and only if

(i) for every $u \in \mathcal{V}$, either $u \in \Gamma$, or u is adjacent to some $v \in \Gamma$, and
(ii) the nodes in Γ induce a connected subgraph in $\mathcal{G}$.

The cardinality of the smallest connected dominating set is known as the *connected domination number*, denoted by $\gamma_{\mathcal{G}}$.

For any k, the minimum size of trusted nodes required to achieve desired k-connectivity with $\mathcal{T}_v$ is bounded by $\gamma_{\mathcal{G}}$,

$$|\mathcal{T}_v| \leq \gamma_{\mathcal{G}}. \tag{2}$$

Thus, starting with a $\mathcal{T}_v = \Gamma$, we can iteratively reduce the size of $\mathcal{T}_v$ to get a minimal set of trusted nodes with which the graph remains k-vertex connected with $\mathcal{T}_v$. The notion of connected dominating set in graphs has been extensively studied in both graph theory and sensor network literature (e.g., see [18], [19]), wherein a wide variety of applications along with various distributed algorithms for constructing small sized connected dominating sets have been reported.

Let $\texttt{V\_Conn\_Trust}(\mathcal{G}, \mathcal{T}_v)$ be the procedure to determine the vertex connectivity with a given $\mathcal{T}_v$, as outlined in Section III-A, and $\texttt{Conn\_Dom\_Set}(\mathcal{G})$ be the procedure to determine the minimal connected dominating set. Then, a minimal $\mathcal{T}_v$ required to achieve the desired vertex connectivity k' with trusted nodes can be obtained using Algorithm 1 given below. Starting with a $\mathcal{T}_v = \Gamma$, in each iteration, a node is removed if the resulting connectivity is at least equal to the desired vertex connectivity with $\mathcal{T}_v$.

Note that in Algorithm 1, $O(|\Gamma|)$ calls are made to the sub-routine that computes vertex connectivity with trusted nodes. Next, we present a heuristic to compute a minimal set of trusted edges $\mathcal{T}_e$ to achieve the desired edge connectivity with $\mathcal{T}_e$. As with the vertex connectivity case, complete edge connectivity with $\mathcal{T}_e$ is obtained if and only if there exists an edge-trusted path between every pair of nodes, which is true if $\mathcal{T}_e$ consists of edges in a *spanning tree*. Thus, to achieve complete edge connectivity with $\mathcal{T}_e$, we need $|\mathcal{T}_e| = n - 1$, where n is the number of nodes in $\mathcal{G}$. At the same time, we get an upper bound on the minimum size of $\mathcal{T}_e$ to achieve desired k-edge connectivity with trusted edges for any k,

Algorithm 1 Trusted Nodes for Vertex Connectivity

Input: $\mathcal{G}(\mathcal{V}, \mathcal{E})$, k'
Output: $\mathcal{T}_v \subseteq \mathcal{V}$
$\Gamma \leftarrow \texttt{Conn\_Dom\_Set}(\mathcal{G})$
$\mathcal{T}_v \leftarrow \Gamma$
for $i = 1$ to $|\Gamma|$ **do**
    $v \leftarrow \texttt{V\_Conn\_Trust}(\mathcal{G}, \mathcal{T}_v \setminus \{\Gamma(i)\})$
    if $v \geq k'$ **do**
        $\mathcal{T}_v \leftarrow \mathcal{T}_v \setminus \{\Gamma(i)\}$
    end if
end for

$$|\mathcal{T}_e| \leq n - 1. \tag{3}$$

Thus, to obtain a minimal $\mathcal{T}_e$ to achieve desired edge connectivity with trusted edges, say k', we start with a $\mathcal{T}_e$ consisting of edges in a spanning tree, and then iteratively reduce the size of $\mathcal{T}_e$. If $\texttt{E\_Conn\_Trust}(\mathcal{G}, \mathcal{T}_e)$ is the routine to compute edge-connectivity with $\mathcal{T}_e$, as discussed in Section III-B, and $\texttt{Min\_Span\_Tree}(\mathcal{G})$ computes edges in a spanning tree, then Algorithm 2 computes a minimal $\mathcal{T}_e$ to achieve the desired edge connectivity with trusted edges.

Algorithm 2 Trusted Edges for Edge Connectivity

Input: $\mathcal{G}(\mathcal{V}, \mathcal{E})$, k'
Output: $\mathcal{T}_e \subseteq \mathcal{E}$
$\mathcal{E}' \leftarrow \texttt{Min\_Span\_Tree}(\mathcal{G})$
$\mathcal{T}_e \leftarrow \mathcal{E}'$
for $i = 1$ to $|\mathcal{E}'|$ **do**
    $e \leftarrow \texttt{E\_Conn\_Trust}(\mathcal{G}, \mathcal{T}_e \setminus \{\mathcal{E}'(i)\})$
    if $e \geq k'$ **do**
        $\mathcal{T}_e \leftarrow \mathcal{T}_e \setminus \{\mathcal{E}'(i)\}$
    end if
end for

If n is the number of nodes in the graph, then, in Algorithm 2, $O(n)$ calls are made to the routine that computes edge connectivity with trusted edges. Next, we present a numerical evaluation of both the algorithms.

V. Numerical Evaluation

We evaluate our results for three different types networks, including *Preferential attachment networks*, *Erdős-Rényi networks*, and *Random geometric networks*. These network are frequently used to model various networking phenomenon existing in nature and also for various engineering applications. The details of networks considered for our simulations are stated below.

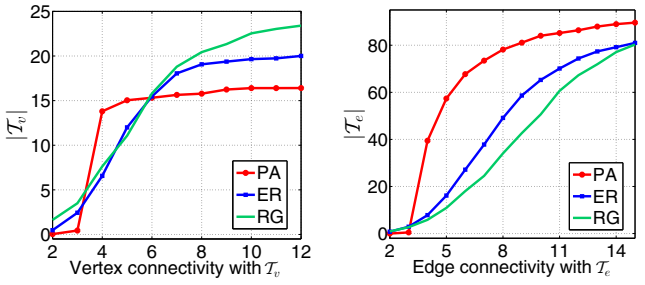
- *Preferential attachment (PA) networks* with $n = 100$ nodes obtained by adding a node to an existing network one at a time. Each new node is connected to $m = 3$ existing nodes with the probability proportional to the degree of the nodes.

- *Erdős-Rényi (ER) networks* consisting of $n = 100$ nodes in which the probability of an edge between any two nodes is $p = 0.07$.
- *Random geometric (RG) networks* consisting of $n = 100$ nodes that are distributed uniformly at random in the unit area. An edge exists between any two nodes if the Euclidean distance between them is at most 0.18.

Every single point in the plots in Figures 6 (a) and (b) is an average of thirty randomly generated instances. The minimum number of trusted nodes (computed by the Algorithm 1) sufficient to achieve the desired vertex connectivity with $\mathcal{T}_v$, and the minimum number of trusted edges (computed by the Algorithm 2) sufficient to achieve the desired edge connectivity with $\mathcal{T}_e$, are plotted in Figures 6 (a) and (b) respectively.

In the case of the preferential attachment networks, the vertex connectivity with no trusted nodes is 3. To increase the vertex connectivity from 3 to 4, we observe a big jump in $|\mathcal{T}_v|$, which is almost equal to the size of the minimum connected dominating set. In the case of our preferential attachment networks, vertex connectivity with $\mathcal{T}_v$ is exhibited as an 'all-or-nothing' type phenomenon, i.e., to increase the vertex connectivity even by one, the number of trusted nodes needed are sufficient to make the network completely vertex connected with $\mathcal{T}_v$. However, in the cases of Erdős-Rényi and random geometric networks, we observe rather a continuous increase in $|\mathcal{T}_v|$. The plot of $\mathcal{T}_v$ is plateaued once $|\mathcal{T}_v|$ is equal to the size of the connected dominating set.

Similar patterns are observed in the cases of edge connectivity with trusted edges. In the case of preferential attachment networks considered, to increase the edge connectivity from 3 to 4, the number of trusted edges needed increases from 0 to almost $n/2$. For the cases of Erdős-Rényi and random geometric networks, $|\mathcal{T}_e|$ increases in a more continuous manner to achieve higher values of edge connectivity with $\mathcal{T}_e$.



(a) (b)

Fig. 6. (a) Number of trusted nodes $\mathcal{T}_v$ as a function of the vertex connectivity with trusted nodes. (b) Number of trusted edges $\mathcal{T}_e$ as a function of the edge connectivity with trusted edges.

VI. Conclusions

A typical approach to improving network connectivity is to add links in a strategic manner. We adapted a different approach to achieve any desired value of vertex and edge connectivity. The basic idea was to make a small subset of nodes and edges trusted, i.e., insusceptible to failures. We then showed that existence of such nodes and edges has an effect of having a higher network connectivity. We also presented heuristics to select a small subset of trusted nodes and edges to achieve any desired value of network connectivity. Using this approach, even the sparse networks can be made highly connected without adding edges. As opposed to traditional way of improving connectivity by "redundancy," our approach relies on making a portion of network more reliable and trusted, and thereby, achieving the desired connectivity. In future, we aim to combine both approaches to devise a more efficient strategy to improve connectivity. Moreover, we would like to generalize the notion of trustedness in the context of practical networks.

Acknowledgments

This work was supported in part by the National Science Foundation (CNS-1238959) and by the Air Force Research Laboratory (FA 8750-14-2-0180).

References

[1] F. Pasqualetti, A. Bicchi, and F. Bullo, "Consensus computation in unreliable networks: A system theoretic approach," *IEEE Transactions on Automatic Control*, vol. 57, no. 1, pp. 90–104, 2012.
[2] H. Kerivin and A. R. Mahjoub, "Design of survivable networks: A survey," *Networks*, vol. 46, no. 1, pp. 1–21, 2005.
[3] H. Nagamochi and T. Ibaraki, *Algorithmic Aspects of Graph Connectivity*. Cambridge University Press New York, 2008, vol. 123.
[4] K. P. Eswaran and R. E. Tarjan, "Augmentation problems," *SIAM Journal on Computing*, vol. 5, no. 4, pp. 653–665, 1976.
[5] J. Bang-Jensen, H. N. Gabow, T. Jordán, and Z. Szigeti, "Edge-connectivity augmentation with partition constraints," *SIAM Journal on Discrete Mathematics*, vol. 12, no. 2, pp. 160–207, 1999.
[6] B. Jackson and T. Jordán, "A near optimal algorithm for vertex connectivity augmentation," in *International Symposium on Algorithms and Computation*. Springer, 2000, pp. 313–325.
[7] Z. Nutov, "Approximating node-connectivity augmentation problems," *Algorithmica*, vol. 63, no. 1-2, pp. 398–410, 2012.
[8] A. Frank, *Mathematical Programming: State of the Art 1994*. Ann Arbor, MI: University of Michigan, 1994, ch. Connectivity augmentation problems in network design, pp. 34–63.
[9] G. Kortsarz, R. Krauthgamer, and J. R. Lee, "Hardness of approximation for vertex-connectivity network design problems," *SIAM Journal on Computing*, vol. 33, no. 3, pp. 704–720, 2004.
[10] K. Bhawalkar, J. Kleinberg, K. Lewi, T. Roughgarden, and A. Sharma, "Preventing unraveling in social networks: the anchored k-core problem," *SIAM Journal on Discrete Mathematics*, vol. 29, no. 3, 2015.
[11] M. Dziubiński and S. Goyal, "Network design and defence," *Games and Economic Behavior*, vol. 79, pp. 30–43, 2013.
[12] K. Menger, "Zur allgemeinen kurventheorie," *Fundamenta Mathematicae*, vol. 10, no. 1, pp. 96–115, 1927.
[13] O. R. Oellermann, "Menger's theorem," in *Topics in Structural Graph Theory*, L. W. Beineke and R. J. Wilson, Eds. Cambridge University Press, 2013, pp. 13–39.
[14] A.-H. Esfahanian, "Connectivity algorithms," in *Topics in Structural Graph Theory*, L. W. Beineke and R. J. Wilson, Eds. Cambridge University Press, 2013, pp. 268–281.
[15] S. Even, *Graph Algorithms*. Computer Science Press, 1979.
[16] H. Nagamochi and T. Ibaraki, "Computing edge-connectivity in multigraphs and capacitated graphs," *SIAM Journal on Discrete Mathematics*, vol. 5, no. 1, pp. 54–66, 1992.
[17] D. R. Karger, "Minimum cuts in near-linear time," *Journal of the ACM*, vol. 47, no. 1, pp. 46–76, 2000.
[18] S. Guha and S. Khuller, "Approximation algorithms for connected dominating sets," *Algorithmica*, vol. 20, no. 4, pp. 374–387, 1998.
[19] P.-J. Wan, K. M. Alzoubi, and O. Frieder, "Distributed construction of connected dominating set in wireless ad hoc networks," *Mobile Networks and Applications*, vol. 9, no. 2, pp. 141–149, 2004.