This article has been accepted for publication in IEEE Transactions on Control of Network Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCNS.2024.3457582

Leaky Forcing in Graphs for Resilient Controllability in Networks

Waseem Abbas

***Abstract*—This paper studies resilient strong structural controllability (SSC) in networks with misbehaving agents and edges. We consider various misbehavior models and identify the set of input agents offering resilience against such disruptions. Our approach leverages a graph-based characterization of SSC, utilizing the concept of zero forcing in graphs. Specifically, we examine three misbehavior models that disrupt the zero forcing process and compromise network SSC. We then characterize a leader set that guarantees SSC despite misbehaving nodes and edges, utilizing the concept of leaky forcing–a variation of zero forcing in graphs. Our main finding reveals that resilience against one misbehavior model inherently provides resilience against others, thus simplifying the design process. Furthermore, we explore combining multiple networks by augmenting edges between their nodes to achieve SSC in the combined network using a reduced leader set compared to the leader sets of individual networks. We analyze the trade-off between added edges and leader set size in the resulting combined graph. Finally, we discuss computational aspects and provide numerical evaluations to demonstrate the effectiveness of our approach.**

I. Introduction

Effective control of networked systems relies heavily on their underlying topology and the choice of leaders, agents through which control signals are injected into the system [1]–[5]. A popular approach to the leader selection problem involves characterizing network controllability from a graph-theoretic perspective, facilitating the formulation of vertex selection problems solvable through various combinatorial methods. Several graph-based characterizations of network controllability have emerged, including those based on graph distances, zero forcing, and matching in graphs (e.g., [6]–[13]). To this end, the notion of zero forcing in graphs is particularly useful for strong structural controllability (SSC) in networks. The zero forcing phenomenon, introduced in [14], is a dynamic coloring of nodes in a graph initiated by a subset of colored nodes called the zero forcing set in a graph. Interestingly, a set of leaders makes a given network strong structurally controllable if the corresponding nodes in the underlying graph constitute a zero forcing set [6], [15], [16].

In this paper, we consider the *resilient strong structural controllability* problem, ensuring that a network remains strongly structurally controllable even in the presence of misbehaving agents or edges. We examine various misbehavior models for agents and edges that hinder SSC and characterize the set of leaders that offer resilience against such disruptions. A single misbehaving agent or an edge–due to a fault or an adversary–can deteriorate the network's controllability. Given the susceptibility of networks to adversarial attacks and failures, achieving resilient SSC poses a significant challenge in the control of networked systems.

We consider three distinct misbehavior models exhibited by nodes (agents) and edges, each posing unique challenges to network SSC. These models include leak nodes, non-forcing edges, and removable edges, all of which disrupt the zero forcing process in the underlying network graph. We then examine the leader set guaranteeing the network SSC despite a given number of such misbehaving nodes/edges. For this, we utilize the idea of *leaky forcing in graphs*, a variant of zero forcing in graphs, recently introduced in [17], and further studied in [18]–[21]. Our main result shows that the resilient leader selection against one misbehavior model extends resilience against the other two, streamlining the design process. Furthermore, we explore a design challenge involving combining multiple networks by adding edges between their nodes. We aim to maintain the resilient SSC of the resulting combined graph with a reduced leader set compared to the combined leader sets of individual networks. This is significant because adding edges can often deteriorate SSC. Our goal is to identify edges that not only sustain the network SSC but also do so with fewer resources–specifically, a reduced number of leaders. Next, we summarize the main contributions below:

- We examine resilient SSC in networks by introducing a framework based on the concept of leaky forcing in graphs. We consider three distinct models of misbehaving nodes and edges, and characterize the leaders that guarantee network SSC despite the presence of ℓ abnormal nodes/edges (Section III).
- We establish the equivalence of resilience against misbehavior across different models. Specifically, we show that a leader set resilient to ℓ misbehaving nodes/edges under one model exhibits the same resilience under other models, streamlining the design process (Section IV).
- We discuss the computational aspects of selecting leaders for the resilient SSC and present a numerical evaluation (Section V).
- We examine combining multiple graphs by augmenting edges between their nodes while maintaining resilient SSC. We examine the tradeoff between the number of added edges and the size of the leader set required to ensure resilient SSC in the resulting combined graph (Section VI).

There have been works discussing the impact of nodes/edges addition and deletion on the network's controllability, for instance, [22]–[27]. Similarly, some researchers have considered

W. Abbas is with the Systems Engineering Department at the University of Texas at Dallas, Richardson, TX (Email: waseem.abbas@utdallas.edu). This work was supported in parts by the National Science Foundation under Grant no. 2325416.

the effects of actuators/input node failures on the overall controllability of systems. Some also propose methods to have a sufficient number of input nodes maintaining network controllability despite such failures [28]–[31]. Our work differs from theirs as we consider node/edge misbehavior models beyond deletion. These misbehaving nodes/edges remain within the network disrupting the underlying graph-theoretic process required to ensure SSC. Our goal is to have leader sets guaranteeing SSC despite the existence of any ℓ such nodes/edges within the network. For this, we utilize the idea of leaky forcing in graphs, which is variation of the well-known zero forcing in graphs. Zero forcing in graphs is a node coloring process where a small subset of initially colored nodes, known as the *zero forcing set*, changes the colors of other nodes according to a specific rule, eventually leading to the entire graph being colored. The concept of zero forcing and its variations in graphs has received significant attention in recent years, with research focusing on upper and lower bounds [32]–[34], algorithmic developments [35]–[38], studies on special graph families [35], [39]–[41], and various applications [6], [16], [41]–[43].

Leaky forcing is a *resilient* variation of the zero forcing phenomenon, where some nodes, termed *leak nodes*, do not adhere to the color-changing rule. Despite the presence of these leak nodes, the goal is to ensure that an initially colored set of nodes, called the *leaky forcing set*, can still manage to color the entire graph. This concept of leaky forcing is relatively new, first introduced in [17], and has since been further explored in [18]–[21], which characterize leaky forcing sets and establish bounds on their sizes within specific graph families. The first part of this article (Section IV) is closely related to [19], which considers various generalizations of 'leaks', or misbehaving nodes and edges in a graph. Specifically, they examine 'leak nodes' (nodes that do not adhere to the color-changing rule of the zero forcing process), 'leak edges' (edges that exist in a graph but prevent the application of the zero forcing process), and 'specified leaks' (where the color-changing rule cannot be applied between a specified pair of nodes). Alameda et al. [19] demonstrate the equivalence of these misbehavior models by showing that a forcing set that guarantees the coloring of the entire node set under one misbehavior model also guarantees the same under other misbehavior models. This article extends these concepts to resilient strong structural controllability in networks. We further demonstrate that a forcing set resilient to ℓ leak nodes is also resilient to the removal of ℓ edges from the graph (not just 'leak edges' but actual edge removal). This distinction emphasizes that the leaky forcing set is also resilient to structural changes in the graph due to edge removals.

A subset of results appeared in preliminary form in [44], but with limited technical details. Specifically, Section IV of this article provides comprehensive results on the equivalence of resilient leaders for various misbehavior models, including complete technical details and proofs. Section V has been expanded to include numerical evaluations and a discussion of the computational aspects of leader selection for resilient SSC. Additionally, Section VI is entirely new to this article and presents findings on graph combinations and their impact on leader selection for SSC and resilient SSC. Moreover, several examples are included to illustrate the main ideas.

II. Preliminaries

We consider a network of agents modeled by an undirected graph $G = (V, E)$. The node set V, and the edge set E, represent agents and interconnections between agents, respectively. The edge between nodes i and j is denoted by an unordered pair (i, j). The *neighborhood* of node i is $\mathcal{N}(i) = \{j \in V | \ (i, j) \in E\}$. Similarly, the *closed neighborhood* of i is $\mathcal{N}[i] = \mathcal{N}(i) \cup \{i\}$. The *distance* between nodes i and j, denoted by $d(i, j)$, is the number of edges in the shortest path between them. For a given graph $G = (V, E)$ with $|V| = n$ nodes, we define a family of symmetric matrices $\mathcal{M}(G)$ as below:

$$\mathcal{M}(G) = \{M \in \mathbb{R}^{n\times n} \mid M = M^\top, \text{ and for } i \neq j, \\ M_{ij} \neq 0 \Leftrightarrow (i, j) \in E\}. \tag{1}$$

Note that the location of zero and non-zero entries in every $M \in \mathcal{M}(G)$ remains the same, and is entirely described by the edge set of G.

A. System Model and Graph Controllability

We consider the following leader-follower system defined over a graph $G = (V, E)$.

$$\dot{x}(t) = Mx(t) + Bu(t), \tag{2}$$

where $x(t) \in \mathbb{R}^n$ is the system state, $u(t) \in \mathbb{R}^m$ is the external input, $M \in \mathcal{M}(G)$ (as in (1)) is the system matrix, and $B \in \mathbb{R}^{n\times m}$ is the input matrix describing the *leader* (input) nodes. If $V = \{v_1, v_2, \cdots, v_n\}$, and $V' = \{\ell_1, \ell_2, \cdots, \ell_m\} \subseteq V$ is a set of leader nodes, then we define B as follows:

$$[B]_{ij} = \begin{cases} 1 & \text{if } v_i = \ell_j, \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

We observe that the family of matrices $\mathcal{M}(G)$ also includes the Laplacian and adjacency matrices of G.

For a given graph G, system matrix $M \in \mathcal{M}(G)$, and input matrix B, the system in (2) is called *controllable* if there exists an input to drive the system from an arbitrary initial state $x(t_0)$ to an arbitrary final state $x(t_f)$. In this case, we say that (M, B) is a *controllable pair*. A pair (M, B) is controllable if and only if the rank of the *controllability matrix* $\Gamma(M, B)$, defined below, is $|V| = n$ (i.e., full rank).

$$\Gamma(M, B) = \begin{bmatrix} B & MB & M^2B & \cdots & M^{n-1}B \end{bmatrix}. \tag{4}$$

Since leader nodes V' define the input matrix B, we sometimes abuse the notation slightly and use (M, V') *is controllable* to denote that (M, B) is a controllable pair.

Definition 1. (Network Strong Structural Controllability) *A network $G = (V, E)$ with a leader set $V' \subseteq V$ is strong structurally controllable if and only if (M, V') is a controllable pair* for all $M \in \mathcal{M}(G)$.

Network strong structural controllability is a stronger notion compared to the (weak) structural controllability, which requires the existence of at least one matrix $M \in \mathcal{M}(G)$ for which (M, V') is a controllable pair.

This article has been accepted for publication in IEEE Transactions on Control of Network Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCNS.2024.3457582

B. Network Controllability and Zero Forcing in Graphs

A key concern in network controllability is to compute a minimum leader set $V' \subseteq V$ that makes the network strong structurally controllable (as defined above). The problem is often referred to as the *minimum leader selection* for network controllability. A graph-theoretic characterization of the minimum leader set rendering the network strong structurally controllable is remarkably useful here. Monshizadeh et al. characterized the minimum leader set for network SSC in [6] using the notion of *zero forcing* in graphs, which is related to the dynamic coloring of nodes. We introduce zero forcing ideas below and then state the main result from [6].

Definition 2. *(*Zero forcing*) Given a graph $G = (V, E)$ whose nodes are initially colored either black or white. Consider the following node color changing rule: If $v \in V$ is colored black and has exactly one white neighbor u, change the color of u to black. Zero forcing is the application of the above rule until no further color changes are possible.*

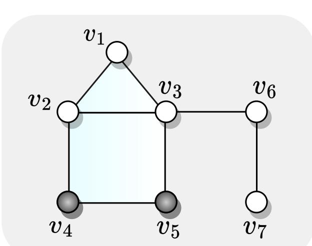
Definition 3. (Force) *A force is an application of the color changing rule due to which the color of a white node u is changed to black by some black node v. We say that v* forced *u and denote it by $v \to u$.*

Definition 4. *(*Input and derived sets*) For a graph $G = (V, E)$ with an initial set of black nodes $V' \subseteq V$, the* derived set *of V', denoted by $\mathcal{D}(V')$, is the set of black nodes obtained after applying the zero forcing rule exhaustively. The initial set of black nodes V' is the set of input nodes.*

We note that for a given input set V', the derived set $\mathcal{D}(V')$ is unique [14].

Definition 5. *(*Zero forcing set (ZFS) and zero forcing number*) For a graph $G = (V, E)$, an input set $V' \subseteq V$ is a ZFS if $\mathcal{D}(V') = V$ (i.e., all nodes are colored black after the exhaustive application of the zero forcing rule). We denote a ZFS by Z_0. The number of nodes in the minimum ZFS is the* zero forcing number *of the graph and denoted by $z_0(G)$.*

Figure 1 illustrates the zero forcing terms defined above.



1. $v_4 \to v_2$
2. $v_5 \to v_3$
3. $v_2 \to v_1$
4. $v_3 \to v_6$
5. $v_6 \to v_7$

Fig. 1: $V' = \{v_4, v_5\}$ is a ZFS of the graph along with a sequence of forces coloring all nodes black.

A leader set for the strong structural controllability is closely related to the notion of ZFS of the network graph. A direct consequence of Theorems IV.4, IV.8, and Proposition IV.9 in [6] is the following result:

Theorem 2.1. *[6] The undirected network $G = (V, E)$ is strong structurally controllable with a leader set $V' \subseteq V$ (as in Definition 1) if and only if V' is a ZFS of G, (i.e., $\mathcal{D}(V') = V$).*

Thus, ZFS in graphs is an important idea from the network controllability perspective and it completely characterizes the leader set for the strong structural controllability of the network. Theorem 2.1 implies that the minimum number of leaders needed for the network strong structural controllability is same as the zero forcing number of the network graph. We note that computing a minimum ZFS and zero forcing number are NP-hrad in general [45]. However, there are several heuristics to compute a small-sized ZFS, for instance, see [36], [38], [40], [43], [46].

III. Resilient Strong Structural Controllability (SSC) in Networks

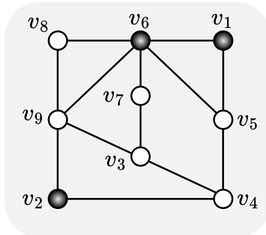
The equivalence between ZFS and leader set for network SSC is significant. According to [6, Corollary IV.3], a network $G = (V, E)$ is strong structurally controllable with a leader set $V' \subset V$ if and only if G is strong structurally controllable with the $\mathcal{D}(V')$, the derived set of V'. Furthermore, [6, Theorem IV.4] indicates that if $\mathcal{D}(V') \neq V$, then the network is not strong structurally controllable. Consequently, if the zero forcing process with a leader set V' is disrupted by misbehaving nodes, edges, or edge failures such that $\mathcal{D}(V') \neq V$, then G is not strong structurally controllable with V' under these misbehaviors. For instance, consider the network in Figure 2 with the leader set $V' = \{v_1, v_2, v_6\}$. If all nodes are normal, the ZF process initiated by the leader set V' will eventually color all nodes in V (i.e., $\mathcal{D}(V') = V$), and the network will be strong structurally controllable with V'. However, if edge (v_3, v_4) is removed, the ZF process will be disrupted, causing $\mathcal{D}(V') \neq V$, implying that the network is not strong structurally controllable with V'. Similarly, if v_5 behaves abnormally in the sense that it does not force any other node, then the zero forcing process will again be hindered and $\mathcal{D}(V') \neq V$ asserting that the network is not strong structurally controllable with V'.

However, if $V' = \{v_1, v_2, v_6, v_8\}$, then the network remains strong structurally controllable despite any single misbehaving node (refusing to force other nodes) or an edge. Thus, the network SSC can be preserved even in the presence of misbehaving nodes or edges through some redundant leader nodes selected carefully. Our goal in the paper is to study:

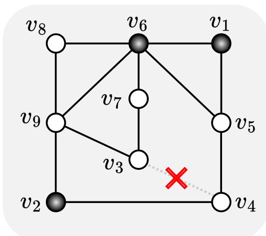
How can we systematically characterize a set of leaders in a network to ensure its SSC in the face of misbehaving nodes and edges? Further, how can we leverage this characterization to compute a resilient set of leaders for the network SSC?

Next, we consider three different misbehaving node and edge behaviors disrupting the zero forcing process. Subsequently, we present leader selections guaranteeing all nodes in the network get colored due to the zero forcing process despite a certain number of misbehaving nodes and edges, thus achieving the resilient network SSC. Our main result (in Section IV) shows that resilience to one type of misbehaving nodes/edges implies resilience to the other kinds of misbehaving nodes/edges.

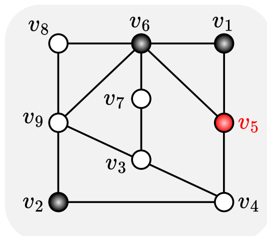
This article has been accepted for publication in IEEE Transactions on Control of Network Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCNS.2024.3457582



(a) G



(b) misbehaving edge



(c) misbehaving node

Fig. 2: (a) $V' = \{v_1, v_2, v_6\}$ is a ZFS of G. (b) (v_3, v_4) is a misbehaving edge. (c) v_5 is a misbehaving node not forcing any other node.

A. Misbehavior Models and Resilience Problems

We consider the following three node and edge misbehaviors that can be caused by the adversarial attack or other abnormality. All of these failures ultimately disrupt the zero forcing process.

1) Leak (non-forcing) nodes: A *leak* is a node $v \in V$ that does not force any other node, i.e., considering v to be a leak node that is colored black and has exactly one white neighbor, then v does not force its white neighbor (which it should in case v was normal). A set of all leaks is the *leak set*, denoted by $L \subseteq V$.

The term 'leak node' is adapted from [17], where such a non-forcing behavior of nodes is introduced. Practically, a leak node can be realized in multiple ways. For instance, if an additional node α, which is not a part of the original network and is colored white, becomes adjacent to exactly one node, say v, in the network G, then v is unable to force any other node in G. Figure 3 illustrates this situation.

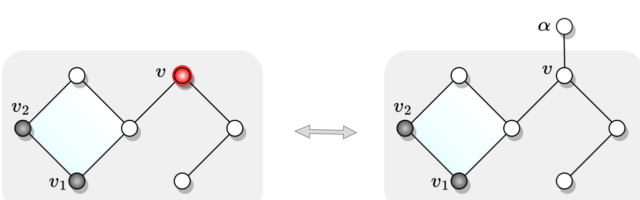


Fig. 3: v is a leak node not forcing any other node. Equivalently, an outside node α becomes adjacent to v and prevents v from forcing any node.

Now the *resilience problem* is to have a (minimal) leader set such that all nodes are colored at the end of the zero forcing process despite ℓ leak nodes, which are unknown. For a given ℓ, computing such a leader set is referred to as the *ℓ-leaky forcing set problem* [17], [18]. We formally define the leaky derived set and leaky forcing set below:

Definition 6. (Leaky derived set) *Given a graph $G = (V, E)$, input set V', and a leak set L, then the set of black nodes obtained after applying the zero forcing rule exhaustively while considering the leaks in L is the leaky derived set, denoted by $\mathcal{D}_L(V')$.*

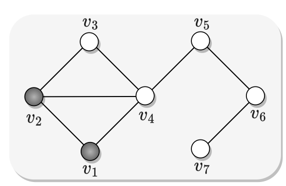
Definition 7. (ℓ-leaky forcing set (ℓ-LFS)) *An input set $V' \subseteq V$ is an ℓ-LFS if for* any *leak set $L \subset V$ with ℓ leaks (i.e., $|L| = \ell$), $\mathcal{D}_L(V') = V$. In other words, starting with V', all nodes are colored black by iteratively applying the zero forcing rule with* any *ℓ leaks. The cardinality of the minimum ℓ-LFS is the* ℓ-forcing number *of G, denoted by $z_\ell(G)$.*

We note that for $\ell = 0$, the ℓ-forcing number is same as the zero forcing number.

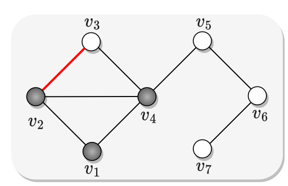
2) Non-forcing edges: Here, we explore edge attacks wherein an edge cannot be utilized by either of its end nodes to force the other end node, categorizing such an edge as a non-forcing edge. Specifically, an edge (u, v) is deemed *non-forcing* if neither u forces v, nor does v force u. In this scenario, the resilience problem involves identifying the ℓ-edge forcing set, as defined below.

Definition 8. (ℓ-Edge forcing set (ℓ-EFS)) *For a given $G = (V, E)$ and a positive integer ℓ, let $E_\ell \subseteq E$ be an* arbitrary *subset of at most ℓ non-forcing edges (i.e., $|E_\ell| \le \ell$). An input set $V' \subseteq V$ is an ℓ-EFS if there is a zero forcing process that colors all nodes in V without using the edges in E_ℓ to force nodes.*

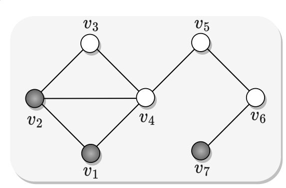
Figure 4 illustrates the non-forcing edge and 1-EFS. $V' = \{v_1, v_2\}$ is a ZFS of G in Figure 4(a). If the edge between v_2 and v_3 is non-forcing, then the derived set consists of only three nodes $\{v_1, v_2, v_4\}$ at the end of the zero forcing process. However, if the leader set is $V' = \{v_1, v_2, v_7\}$, then all nodes are colored as a result of the zero forcing process despite any single non-forcing edge.



(a)



(b)



(c)

Fig. 4: (a) $\{v_1, v_2\}$ is a ZFS given all edges are normal. (b) (v_2, v_3) is a non-forcing edge. (c) $\{v_1, v_2, v_7\}$ is a 1-EFS.

3) Removable edges: The third failure model we consider is the one where a maximum of ℓ edges are removed from the

This article has been accepted for publication in IEEE Transactions on Control of Network Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCNS.2024.3457582

graph to disrupt the zero forcing process. The corresponding resilience problem is to have enough leaders to guarantee that despite ℓ edge removals, the network remains strong structurally controllable, or equivalently all nodes are colored as a result of the zero forcing. In other words, the goal is to find a minimum size *ℓ-forcing set with removable edges* defined below:

Definition 9. (ℓ-forcing set with removable edges (ℓ-FSR)) *For a given $G = (V, E)$ and ℓ, consider a subgraph $G' = (V, E')$, where $E' \subseteq E$ and $|E| - |E'| \leq \ell$. Then, $V' \subseteq V$ is an ℓ-FSR of G if V' is a ZFS of* every *such G'. Note that an ℓ-FSR must also be a ZFS of G.*

We observe that making an edge non-forcing can be different from removing the edge. For instance, unlike a non-forcing edge, removing an edge can sometimes be useful, as Figure 5 illustrates. If edge (v_4, v_7) in G in Figure 5(a) is removed, then $V' = \{v_1, v_2\}$ is a ZFS of the resulting graph (Figure 5(b)), thus, making the network controllable. However, if (v_4, v_7) is a non-forcing edge (Figure 5(c)), then $\{v_1, v_2\}$ is no longer a ZFS of the network.

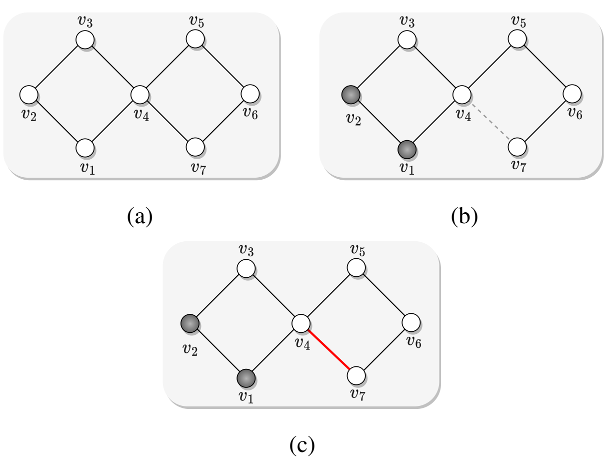


Fig. 5: (a) A graph G. (b) $\{v_1, v_2\}$ becomes a ZFS of G after removing the edge (v_4, v_7). (c) If (v_4, v_7) is a non-forcing edge, then $\{v_1, v_2\}$ is a not a ZFS of G.

Next, we show that a leader set resilient to one misbehavior model is also resilient to the other models.

IV. Equivalence of Resilience for Various Misbehavior Models

Here, for a given $G = (V, E)$ and ℓ, we show the equivalence between ℓ-LFS, ℓ-EFS and ℓ-FSR. As a result, we show that a leader set $V' \subseteq V$ ensures resilient controllability against one misbehavior model if and only if it extends resilience to the other two models (discussed above). For instance, a leader set V' that is resilient to ℓ non-forcing nodes must also be resilient to ℓ non-forcing edges and ℓ removable edges simultaneously. We introduce the following terms as in [18], [42].

Definition 10. *Consider a graph $G = (V, E)$, input set $V' \subseteq V$ and the corresponding derived set $\mathcal{D}(V')$, then we define the following terms:*

- *A **chronological list of forces** is a list of forces recorded in the order in which they are performed to construct the derived set.*
- *A **forcing process** F is a set of forces containing a chronological ordering of forces through which all nodes in V are colored black (i.e., $\mathcal{D}(V') = V$).*
- *A **forcing chain** is a sequence of forces $v_i \rightarrow v_{i+1}$, for $i = 1, 2, \cdots, k-1$. We denote such a forcing sequence by $v_1 \rightarrow v_2 \rightarrow \cdots \rightarrow v_{k-1} \rightarrow v_k$.*
- *A **maximal forcing chain** $v_1 \rightarrow v_2 \rightarrow \cdots \rightarrow v_{k-1} \rightarrow v_k$ is a forcing chain such that $v_1 \in V'$ and v_k does not force any other node in G. The terminal node of the maximal forcing chain v_k, is referred to as the **free node**.*
- *A **total forcing set** of V', denoted by $\mathcal{F}(V')$, is a set of all possible forces given an input V', i.e., $v_i \rightarrow v_j \in \mathcal{F}(V')$ if there is a forcing process in G containing $v_i \rightarrow v_j$.*
- *A **total forcing set with leaks** L and input set V', denoted by $\mathcal{F}_L(V')$, is a set of all possible forces given an input set V' and leaks L. In other words, if $v_i \rightarrow v_j \in \mathcal{F}_L(V')$, then $v_i \notin L$ and there is a forcing process containing the force $v_i \rightarrow v_j$.*

Next, consider $G = (V, E)$, a ZFS $V' \subset V$, and a forcing process F with V'. Then, for some $S \subseteq V$, we define the following notations:

$$F(S) = \{x \rightarrow y \in F : \; y \notin S\}. \tag{5}$$

$F(S)$ represents the set of forces in the forcing process F that do not lead to the forcing of nodes within S. Similarly,

$$F \backslash F(S) = \{x \rightarrow y \in F : \; y \in S\}. \tag{6}$$

$F \backslash F(S)$ consists of the forces within the forcing process F that result in the nodes within S being forced, or in other words, colored black.

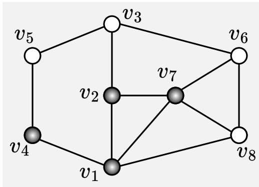
We now state some results from [18] that will be used later.

Lemma 4.1. *[18] Consider $G = (V, E)$ with a ZFS V'. Let F and F' be forcing processes of G with V'. Then, $(F \backslash F(\tilde{V})) \cup F'(\tilde{V})$ is a forcing process with V' for any $\tilde{V}$ obtained from V' using F.*

Lemma 4.1 explains the process of combining two forcing processes to obtain a new forcing process. For example, consider the network G depicted in Figure 6(a), where $V' = \{v_1, v_2, v_4, v_7\}$ forms a ZFS. Figure 6(b) illustrates two forcing processes, F and F', with respect to V'. Now, consider $\tilde{V} = \{v_5, v_3\}$, a subset of black vertices obtained through F. Note that $F \backslash F(\tilde{V}) = \{v_4 \rightarrow v_5, v_5 \rightarrow v_3\}$, and $F'(\tilde{V}) = \{v_1 \rightarrow v_8, v_7 \rightarrow v_6\}$. By Lemma 4.1, combining these two subsets of forces, i.e., $F \backslash F(\tilde{V})) \cup F'(\tilde{V})$, results into another forcing process, as depicted in the last column of Figure 6(b).

Lemma 4.2. *[18] A set V' is a 1-LFS if and only if $\forall v \in V \setminus V'$, there exists $x \rightarrow v \in \mathcal{F}(V')$, $y \rightarrow v \in \mathcal{F}(V')$, where $y \notin x$.*

Theorem 4.3. *[18] A set V' is an ℓ-LFS if and only if V' is an $(\ell - 1)$-LFS for every set of $\ell - 1$ leaks L and for every $v \in V \setminus V'$, there exists $x \rightarrow v \in \mathcal{F}_L(V')$, $y \rightarrow v \in \mathcal{F}_L(V')$, where $y \notin x$.*



(a) G

F	F'	$(F\backslash F(\tilde{V}))\cup F'(\tilde{V})$
$v_4 \rightarrow v_5$	$v_1 \rightarrow v_8$	$v_4 \rightarrow v_5$
$v_5 \rightarrow v_3$	$v_2 \rightarrow v_3$	$v_5 \rightarrow v_3$
$v_3 \rightarrow v_6$	$v_3 \rightarrow v_5$	$v_1 \rightarrow v_8$
$v_7 \rightarrow v_8$	$v_7 \rightarrow v_6$	$v_7 \rightarrow v_6$

(b) Zero forcing processes.

Fig. 6: $V' = \{v_1, v_2, v_4, v_7\}$ is a ZFS of G. Forcing processes F and F' are combined into another forcing processes $F\backslash F(\tilde{V}) \cup F'(\tilde{V})$, where $\tilde{V} = \{v_3, v_5\}$.

A. Main Result

Our main result here is to show the following:

Theorem 4.4. *Given a graph $G = (V, E)$, input set $V' \subseteq V$, and a positive integer $\ell \leq |V|$, the following statements are equivalent:*

1) *V' is an ℓ-LFS.*
2) *V' is an ℓ-EFS.*
3) *V' is an ℓ-FSR.*

We recall that notions of ℓ-LFS, ℓ-EFS, and ℓ-FSR are explained in Definitions 7, 8, and 9, respectively. To prove Theorem 4.4, we need some intermediate results.

Lemma 4.5. *If V' is a 1-LFS of $G = (V, E)$, then for every edge $e = (u, v) \in E$, there exists a forcing process F_e that does not use e, i.e., $u \rightarrow v \notin F_e$ and $v \rightarrow u \notin F_e$.*

Proof. Consider a forcing process F with V' containing $u \rightarrow v$. Suppose $\tilde{V}$ represents the set of black vertices obtained from F until the point where $u \rightarrow v$ is a valid force in F, but v is not yet included in $\tilde{V}$. At this stage, u and all its neighbors, except for v, are colored black through F, and the edge (u, v) remains unused by either u or v to force each other. According to Lemma 4.2, another forcing process F' exists where a different node, denoted as $x \neq u$, forces v. Now, let us introduce another forcing process denoted as F_e, obtained by combining F and F', i.e., $F_e = (F\backslash F(\tilde{V})) \cup F'(\tilde{V})$. By Lemma 4.1, F_e constitutes a valid forcing process with input V'. Note that node v is forced by a node $x \neq u$ in F_e, implying that the edge (u, v) is not utilized by either node u or v in the forcing process F_e, thereby confirming the desired claim. ∎

Lemma 4.6. *Let V' be an $(\ell-1)$-EFS, E_ℓ be a set of ℓ non-forcing edges, and $\mathcal{D}(V')$ be the derived set after forcing. If $e = (u, v) \in E_\ell$, then u and v can not be white simultaneously. Moreover, if exactly one end node of e, say u, is black, then $N[u] \setminus \{v\} \subseteq \mathcal{D}(V')$.*

Proof. If both end nodes of $e \in E_\ell$ are white, then none of the end nodes can force the other end node. Thus, zero forcing behavior of V' does not change even if e is not a non-forcing edge. So, if we consider $E_\ell \backslash e$ as the set of non-forcing edges, $\mathcal{D}(V') \neq V$, implying that V' is not an $(\ell-1)$-EFS, which is a contradiction. Similarly, let $e = (u, v) \in E_\ell$ be such that $u \in \mathcal{D}(V')$ and $v \notin \mathcal{D}(V')$. Assume that $x \in N(u)$ is white and $x \neq v$. Since u has two white neighbors, u can not force any node (including v) even if e is not a non-forcing edge. Again, considering $E_\ell \backslash e$ as the set of non-forcing edges will give $\mathcal{D}(V') \neq V$. It means V' is not an $(\ell-1)$-EFS, which is a contradiction. ∎

Lemma 4.7. *Let V' be an $(\ell-1)$-LFS and L be a set of ℓ leaks. Then $L \subseteq \mathcal{D}(V')$, where $\mathcal{D}(V')$ is a derived set with leaks. Also, each $v \in L$ has at most one white neighbor.*

Proof. Assume $v \in L$ is white, i.e., $v \notin \mathcal{D}(V')$. A white leak node does not change the zero forcing behavior of the black nodes. So, we consider $L' = L \setminus \{v\}$. Since V' is an $(\ell-1)$-LFS, so all nodes should be black for any $(\ell-1)$ leaks. However, all nodes are not black, which is a contradiction. For the second part, assume that there is leak node $v \in L$ that is colored black and has two white neighbors. A black node with two white neighbors can not force any node, so we consider $L' = L \backslash \{v\}$ as a set of $(\ell-1)$ leaks. Since V' is $(\ell-1)$-LFS, it should color all nodes black with leaks in L', which is not the case. Hence, a contradiction, proving the desired claim. ∎

Next, we show the equivalence between ℓ-LFS and ℓ-EFS.

Theorem 4.8. *For a given ℓ and $G = (V, E)$, $V' \subseteq V$ is an ℓ-LFS if and only if V' is an ℓ-EFS.*

Proof. See Appendix. ∎

Lemma 4.9. *If V' is 1-FSR then it is 1-EFS.*

Proof. By contraposition, let V' be a ZFS that is not a 1-EFS. It means there is an edge, say (u, v), such that every forcing process must use the edge, i.e., (u, v) must be a forcing edge in any forcing process. Without the loss of generality, we assume u forces v. It implies that all the black neighbors of v have at least two white neighbors and u has only one white neighbor v. Thus, by removing edge (u, v), v can not be forced. Hence, V' is not a 1-FSR, which is the desired claim. ∎

In the following, we show the equivalence of ℓ-EFS and ℓ-FSR.

Theorem 4.10. *V' is ℓ-EFS if and only if V' is ℓ-FSR.*

Proof. (ℓ-EFS $\rightarrow$ ℓ-FSR) We first note that if F is a ZFP with V' as a ZFS, then there are at most $n-1$ edges used in F. If we remove edges not used in F to get G', then V' will still be a ZFS of G'. Thus, if V' is an ℓ-EFS, it means for any edge set $E_\ell \subseteq E$, where $|E_\ell| \leq \ell$, there exist a forcing process with V', say F_ℓ, coloring all nodes black without using edges in E_ℓ. Since edges in E_ℓ are not used in F_ℓ, we can remove them from G while maintaining V' to be a ZFS of $G' = (V, E\backslash E_\ell)$, i.e., F_ℓ is a forcing process of G' with V', implying V' is an ℓ-FSR of G.

(ℓ-FSR $\rightarrow$ ℓ-EFS) We will prove using induction on ℓ. For $\ell = 1$, if V' is 1-FSR, then it is 1-EFS by Lemma 4.9. Thus, we make the induction hypothesis, if V' is $(\ell-1)$-FSR, then it is $(\ell-1)$-EFS. Assuming V' to be ℓ-FSR, we need to show that V' is ℓ-EFS. Let E_ℓ be a set of ℓ non-forcing edges. Since V' is $(\ell-1)$-EFS (by the induction hypothesis), we apply forces such that each edge $e = (u, v) \in E_\ell$ satisfies one

This article has been accepted for publication in IEEE Transactions on Control of Network Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCNS.2024.3457582

of the two conditions (by Lemma 4.6): (i) Both end nodes of e are colored black. (ii) If one node, say u, is black, then all nodes in $N[u] \setminus \{v\}$ are also colored black. Note that the removal of edges in both cases (i) and (ii) will not change the zero forcing behavior of the black nodes. Thus, we remove these ℓ edges. Since V' is ℓ-FSR, it means that all nodes will be colored black at the end of the forcing process. Thus, all nodes are colored black in spite of ℓ non-forcing edges, i.e., V' is ℓ-EFS, which is the desired result. ∎

A direct corollary of Theorems 4.8 and 4.10 is Theorem 4.4 entailing that resilience to one type of misbehaving nodes/edges implies resilience to other kinds of misbehaving nodes/edges. To illustrate the equivalence of the three leaky forcing variations, consider a graph $G = (V, E)$ in Figure 7(a), where $V' = \{v_1, v_2, v_3\}$ serves as a 1-LFS, 1-EFS, and 1-FSR simultaneously. V' ensures that all nodes become colored through some zero forcing process despite the presence of any single non-forcing node, non-forcing edge, or removable edge. For instance, if we have a non-forcing node v_7, as in Figure 7(b), the zero forcing process, such as $v_1 \to v_4$, $v_3 \to v_5$, $v_4 \to v_6$, $v_6 \to v_7$, guarantees coloring for all nodes. It is important to note that since v_7 is a leak node, it should not force any other node in this process, which is indeed the case here. Similarly, in the presence of a removable edge, for instance, (v_4, v_6) as in Figure 7(c), the zero forcing process, $v_1 \to v_4$, $v_3 \to v_5$, $v_5 \to v_7$, $v_7 \to v_6$, ensures coloring for all nodes without utilizing the edge (v_4, v_6). Lastly, if we have a single non-forcing edge, say (v_6, v_7), as in Figure 7(d), a zero forcing process, $v_1 \to v_4$, $v_3 \to v_5$, $v_4 \to v_6$, $v_5 \to v_7$, ensures coloring of all the nodes without involving (v_6, v_7).

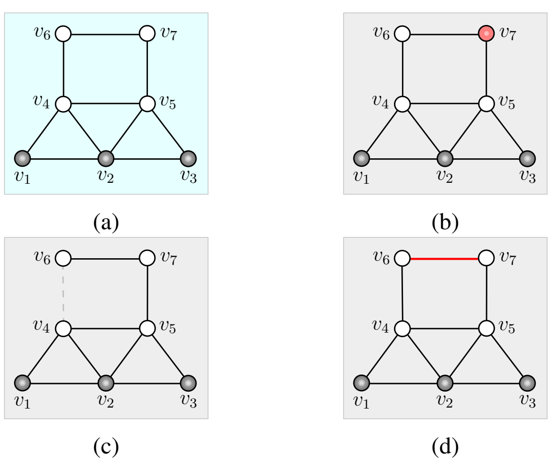


Fig. 7: (a) A graph showing $V' = \{v_1, v_2, v_3\}$, which serves as a 1-LFS, 1-EFS, and 1-FSR simultaneously. (b) v_7 is a leak node. (c) (v_6, v_7) is a removable edge. (d) (v_6, v_7) is a non-forcing edge.

We note that [19] considers other variations of 'leaks' and demonstrates their equivalence, including *specified leaks* and *mixed leaks*. A forcing of node v through node u is a specified leak if $u \to v$ is prohibited. Thus, if u is a leak node (as defined in Section III-A), it represents a set of specified leaks $\{u \to v : \; v \in \mathcal{N}(u)\}$ [19]. Similarly, the notion of mixed leaks represents scenarios where a network contains a combination of different leaks, such as a leak node, a non-forcing edge, and a specified leak. If a set of initially colored nodes guarantees the forcing of the entire node set despite one type of leak, it also guarantees the same in the presence of mixed leaks, i.e., a combination of different leaks [19].

V. Computation and Numerical Illustration

Computing a minimum ℓ-LFS, and therefore, ℓ-EFS and ℓ-FSR, are NP hard problems (since minimum ZFS is NP-hard [38], [45]). Here, we present a greedy algorithm to compute a small-sized 1-LFS and numerically evaluate it. The choice of a greedy algorithm stems from its practical efficiency and effectiveness in generating near-optimal solutions. A greedy algorithm for any ℓ can be designed using a similar approach and utilizing the characterization in Theorem 4.3. For a given input set V', we define $\mathcal{Q}(V')$ to be the set of non-input nodes, each of which can be forced by at least two distinct nodes. More precisely,

$$\begin{aligned}\mathcal{Q}(V') = \{v \in V \setminus V' : \; &\exists\, x \to v \in \mathcal{F}(V') \text{ and} \\ &y \to v \in \mathcal{F}(V'), \text{ and } x \neq y\}.\end{aligned} \tag{7}$$

By Theorem 4.2, V' is 1-LFS if and only if $\mathcal{Q}(V') = V \setminus V'$. Figure 10 also provides an illustration, where the set $\{v_1, v_2, v_4, v_7\}$ is a 1-LFS of G. For each $v \in V \setminus V'$, there exist two zero forcing processes wherein v is forced by distinct nodes (as depicted in Figure 10(b)).

Algorithm 1 presents a *greedy* heuristic to compute a 1-LFS. The main idea is to iteratively include nodes in the leader set V' to maximize the size of $\mathcal{Q}(V')$ (as in (7)) until $\mathcal{Q}(V') = V \setminus V'$. The algorithm initializes V' with a ZFS, leveraging the fact that every 1-LFS is also a ZFS. As a result of the greedy selection, V' might contain some redundant nodes. In Algorithm 1, lines $9 - 14$ remove such redundant nodes to reduce the size of 1-LFS.

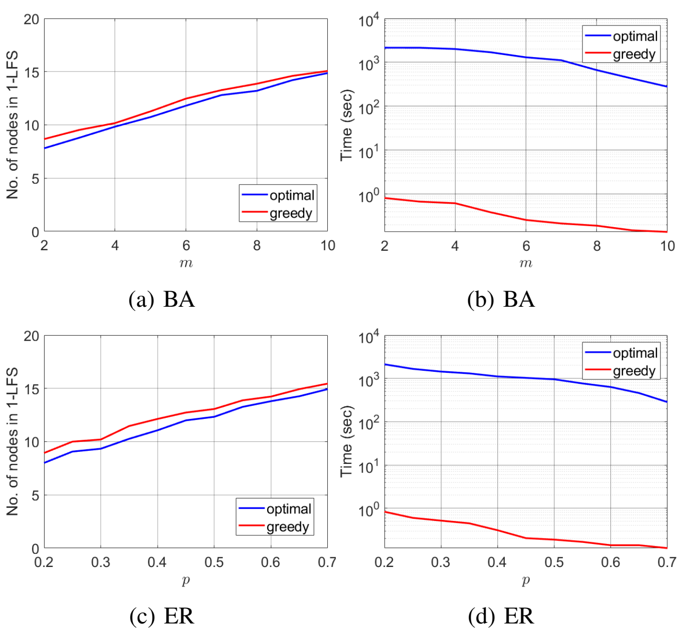
Algorithm 1: Greedy Heuristics for 1-LFS

1 : **given:** $G = (V, E)$, $|V| = n$
2 : **initialization:** $V' = \emptyset$
3 : Compute ZFS $Z_{\mathbf{0}}$, and assign $V' = Z_{\mathbf{0}}$
4 : Compute $\mathcal{Q}(V')$
5 : **while** $|\mathcal{Q}(V')| < n - |V'|$
6 : $\quad v^* = \underset{v \in V \setminus (V' \cup \mathcal{Q}(V'))}{\arg\max} \mathcal{Q}(V' \cup \{v\})$
7 : $\quad V' = V' \cup \{v^*\}$
8 : **end while**
`-------- removing redundancies --------`
9 : $Z = V'$
10 : **for** all $v \in Z$
11 : $\quad$ **if** $|\mathcal{Q}(V' \setminus \{v\})| = n - |V'|$ - 1
12 : $\quad\quad V' = V' \setminus \{v\}$
13 : $\quad$ **end if**
14 : **end for**
15 : **return** V'

Figure 8 compares the greedy heuristic with the optimal solution for Erdös-Rényi (ER) and Barabási-Albert (BA) random graphs with $n = 20$ nodes. In ER graphs, any two nodes

This article has been accepted for publication in IEEE Transactions on Control of Network Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCNS.2024.3457582

are adjacent with a probability p. BA graphs are obtained by attaching a new node (one at a time) to an existing graph through m edges using a preferential attachment model. For computing an optimal solution, we employ an exhaustive search approach. First, we determine the zero forcing number of the graph, say z_0, using the wavefront algorithm [38]. Then, we exhaustively check subsets of nodes to find a minimum 1-leaky forcing set of the graph. Starting with subsets of size of $i = z_0$, we iteratively increment i until finding an optimal 1-leaky forcing set. Each point on the plots averages 15 randomly generated instances. Figures 8(a) and (c) plot the size of 1-LFS as a function of m in BA graphs, and as a function of p in ER graphs, respectively. Figures 8(b) and (d) plot the time taken by the greedy and optimal solutions to compute 1-LFS in BA and ER graphs, respectively.Our results indicate that the greedy and optimal solutions closely align, with differences diminishing as graph density increases. However, the computational time required for an optimal solution, using exhaustive search, is orders of magnitude higher, even for small-sized graphs. Note that in Figure 8(b) and (d), the time axis is presented on a logarithmic scale. We can design a similar greedy heuristic to compute an ℓ-LFS for $\ell > 1$; however, the time complexity will increase significantly with increasing ℓ, and even the greedy heuristic will become inefficient for higher ℓ. Thus, more efficient heuristics are needed to compute ℓ-LFS for large ℓ values.



(a) BA (b) BA

(c) ER (d) ER

Fig. 8: Comparison of optimal (exhaustive search) and greedy heuristic (Algorithm 1) for the computation of 1-LFS in Erdös-Rényi (ER) and Barabási-Albert (BA) random graphs.

VI. Network Design using Graph Combination

In the previous section, we examined an analysis problem focused on characterizing leader sets ensuring resilient SSC in networks. In this section, we focus on a design problem involving graph composition. Our aim is twofold: to seamlessly combine given graphs while guaranteeing SSC in the combined graph, and to achieve this with fewer leaders. For this, we must ensure that the leader sets constitute a ZFS for controllability or ℓ-LFS for resilient controllability. So, we study the following problem:

How to combine multiple graphs by establishing connections between their vertices while considering the trade-off between the number of added edges and the size of the resulting combined graph's leader set, ensuring strong structural controllability of the combined graph.

In [25], we addressed the problem of densifying a given graph while preserving its Zero Forcing Set (ZFS). In [40], we identified edges whose removal not only preserves but also reduces the size of the ZFS in a given graph. The current work is distinct as it focuses on the composition of graphs. In the subsequent subsections, we investigate this graph combination within the context of ZFS and 1-LFS.

A. Graph Combination and ZFS

First,we explore the impact of adding edges between graphs on the size of the Zero Forcing Set (ZFS) of the resulting combined graph. We introduce the concept of a 'patch,' which refers to the set of edges inserted between graphs to create the combined graph. Our observation reveals that the ZFS size of the combined graph typically increases with the size of the patch. In particular, consider two graphs G and H with ZFS Z_g and Z_h, respectively. Also, $|Z_g| = \zeta_g$ and $|Z_h| = \zeta_h$, where $\zeta_g \geq \zeta_h$ (without the loss of generality). For any positive integer $0 \leq \kappa \leq \zeta_h$, we design a maximal patch to combine G and H to obtain $\mathcal{G}$ such that the ZFS of $\mathcal{G}$ is of size $\zeta_g + \kappa$. Thus, we attain $\mathcal{G}$ with a ZFS of the size of any integer value in the interval $[\zeta_g, \quad (\zeta_g + \zeta_h)]$. We note that if ζ_g is the zero forcing number of G, then the size of a ZFS of $\mathcal{G}$ must be at least ζ_g for any patch. For a higher κ, both the ZFS and patch sizes will be larger, and vice versa. We state the result in Theorem 6.1 and illustrate in Figure 9. First, we define the *patch* formally.

Definition 11. *(*Patch and Combined Graph*) Given graphs $G = (V_g, E_g)$ and $H = (V_h, E_h)$, a* patch *is a set of edges $E_p = \{(u, v) \mid u \in V_g,\ v \in V_h\}$ combining G and H to get a* combined graph $\mathcal{G} = (V_g \cup V_h, E_g \cup E_h \cup E_p)$.

Also, recall the concept of *free node*, as in Definition 10. A free node is simply a terminal node of the maximal zero forcing chain and possesses the property that it does not force any other node in the graph. If Z_g is a ZFS of G of size ζ_g, and $R(Z_g)$ represents the corresponding set of free nodes, then $|R(Z_g)| = \zeta_g$. Interestingly, $R(Z_g)$ also forms a ZFS of G [42]. With this understanding, we state the following result.

Theorem 6.1. *Let $G = (V_g, E_g)$ and $H = (V_h, E_h)$ be two graphs with zero forcing sets Z_g and Z_h, respectively. Let $|Z_g| = \zeta_g$, $|Z_h| = \zeta_h$, and $\zeta_g \geq \zeta_h$ (without the loss of generality). If κ is a positive integer, where $0 \leq \kappa \leq \zeta_h$, then G and H can be combined through a patch E_p to get $\mathcal{G} = (V_g \cup V_h, E_g \cup E_h \cup E_p)$, such that $\mathcal{G}$ has a ZFS of size*

$(\zeta_g + \kappa)$, and

$$|E_p| = \frac{(\zeta_h - \kappa)(\zeta_h - \kappa + 1)}{2} + \kappa|V_g| + (\zeta_g - \zeta_h + \kappa)(|V_h| - \kappa). \quad (8)$$

Moreover, E_p is maximal.

Proof. Let $R(Z_g)$ be the set of free nodes in G corresponding to Z_g, and $R(Z_g) = \{u_1, u_2, \cdots, u_{\zeta_h-\kappa}, u_{\zeta_h-\kappa+1} \cdots, u_{\zeta_h}, u_{\zeta_h+1} \cdots, u_{\zeta_g}\}$. Also, let $Z_h = \{v_1, v_2, \cdots, v_{\zeta_h-\kappa}, v_{\zeta_h-\kappa+1}, \cdots, v_{\zeta_h}\}$ be a ZFS of H. First, we construct a patch E_p between G and H as following:

(i) If $\kappa < \zeta_h$, then for each $i \in \{1, 2, \cdots, \zeta_h - \kappa\}$, $u_i \in R(Z_g)$ is adjacent to $v_j \in V_h$ for all $j \leq i$.

(ii) If $\kappa > 0$, then for each $i \in \{\zeta_h - \kappa + 1, \cdots, \zeta_h\}$, $v_i \in V_h$ is adjacent to all nodes in V_g.

(iii) If $\kappa > 0$ or $\zeta_g > \zeta_h$, then for each $i \in \{\zeta_h - \kappa + 1, \cdots, \zeta_g\}$, $u_i \in V_g$ is adjacent to all nodes in V_h.

It is easy to see that the number of edges due to (i) is $(1/2)(\zeta_h - \kappa)(\zeta_h - \kappa + 1)$. Similarly, the number of edges due to (ii) is $\kappa|V_g|$, and due to (iii) is $(\zeta_g - \zeta_h + \kappa)|V_h|$. Since $(\zeta_g - \zeta_h + \kappa)\kappa$ of such edges have already been included due to (ii), we subtract them from the total count. Adding all these edges results in $|E_p|$ as given in (8).

Next, we show that $Z_{\mathcal{G}} = Z_g \cup \{v_{\zeta_h-\kappa+1}, \cdots, v_{\zeta_h}\}$ is a ZFS of the combined graph $\mathcal{G} = (V_g \cup V_h, E_g \cup E_h \cup E_p)$. For this, first, we show that if nodes in $Z_{\mathcal{G}}$ are initially black, then nodes V_g in $\mathcal{G}$ are also colored black. Since Z_g is a ZFS of G, the only reason for our claim to be not true is that edges in E_p might prevent nodes in V_g from getting black. There are three types of edges in E_p, as explained above in (i), (ii), and (iii). Each edge in E_p has one end node from V_g and the other end node from V_h. Considering edges in (i) and (iii), the end nodes of edges from V_g are the free nodes and do not force any node in V_g. Hence, they do not affect the coloring of V_g by Z_g. For all edges in (ii), the end node from V_h is black. In a zero forcing process, if a white node becomes black, then adjoining the white node to a black node does not affect the color changing of the white node. Thus, edges in (ii) do not prevent V_g from getting colored black.

Next, we show that nodes in V_h in $\mathcal{G}$ are also colored black due to $Z_{\mathcal{G}}$. Note that $\{v_1, v_2, \cdots, v_{\zeta_h-\kappa}\} \subseteq V_h$. As a result of edges in (i) above, for every $v_i \in \{v_1, v_2, \cdots, v_{\zeta_h-\kappa}\}$, there is a node $u_j \in R(Z_g)$ such that v_i is the only white neighbor of u_j. Thus, each node in $\{v_1, v_2, \cdots, v_{\zeta_h-\kappa}\}$ gets colored black. Thus, all nodes in the set $\{v_1, v_2, \cdots, v_{\zeta_h-\kappa}, v_{\zeta_h-\kappa+1}, \cdots, v_{\zeta_h}\}$ are black. Also, these nodes constitute a ZFS of H, i.e., Z_h. Now for every edge in (ii) and (iii), one of the end nodes that is in V_g is black. Since adjoining a white node to black node does not prevent the white node from becoming black in a zero forcing process, all nodes in V_h gets black due to Z_h. Thus, $Z_{\mathcal{G}} = Z_g \cup \{v_{\zeta_h-\kappa+1}, \cdots, v_{\zeta_h}\}$ is indeed a ZFS of $\mathcal{G}$. ∎

For illustration, consider G and H in Figure 9. Z_g (dark colored nodes) and $Z_h = \{v_1, v_2, v_3\}$ are zero forcing sets of G and H, respectively. Also, $\zeta_g = 4$, $\zeta_h = 3$, and the free nodes corresponding to Z_g are $R(Z_g) = \{u_1, u_2, u_3, u_4\}$. In Figure 9(a), $\kappa = 0$, thus, $Z_{\mathcal{G}} = Z_g$ is a ZFS of the combined graph $\mathcal{G}$ obtained by adding a patch (red colored edges) of size $\zeta_h(\zeta_h+1)/2 + |V_h| = 6 + |V_h|$. In Figure 9(b), $\kappa = 1$ and $Z_{\mathcal{G}} = Z_g \cup \{v_3\}$ is a ZFS of $\mathcal{G}$. In this case, a patch of larger size is obtained. Similarly, in Figure 9(c), $Z_{\mathcal{G}} = Z_g \cup Z_h$ is a ZFS of $\mathcal{G}$ and a patch of size $|E_p| = \zeta_h|V_g| + \zeta_g(|V_h| - \zeta_h)$ is obtained, which is of the largest size among all the cases.

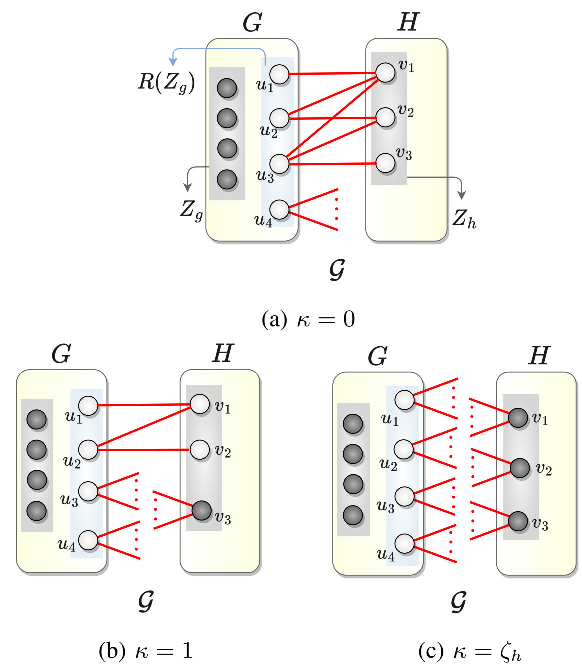


Fig. 9: Illustration of Theorem 6.1 for various κ values.

Next, we consider graph combinations in the context of 1-LFS in graphs ensuring that the input set remains a ZFS despite a single misbehaving node.

B. Graph Combination and 1*-LFS*

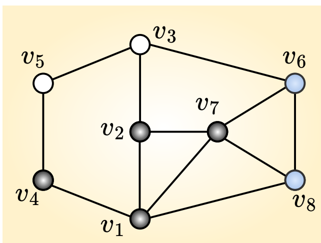
Our objective in this subsection is to combine two graphs, denoted as G and H, by introducing edges between their vertices to minimize the size of the resulting 1-LFS of the combined graph $\mathcal{G}$. Specifically, if Z'_g and Z'_h represent the 1-LFS of graphs G and H respectively, we aim to identify edges between vertices in G and H such that $\mathcal{G}$ possesses a 1-LFS that is a subset of $Z'_g \cup Z'_h$. It is important to note that the addition of edges typically leads to an increase in the size of the 1-LFS (and ZFS). We proceed by defining the notion of 1-free nodes in a graph.

Definition 12. (1-free nodes) *Consider Z'_g to be a 1-LFS of $G = (V, E)$. Let $F_a(v_i)$ and $F_b(v_i)$ represent two distinct forcing processes, where $v_i \in V_g \setminus V$ is forced by different nodes in each process. A node is a 1-free node if it is not a forcer in any force included in $\bigcup_{v_i \in V_g \setminus V} (F_a(v_i) \cup F_b(v_i))$. We denote the set of 1-free nodes for Z'_g by $R_1(Z'_g)$*

In simple words, if Z'_g is a 1-LFS of G and v is a 1-free node, then all the nodes in G are colored black despite a single leak and importantly, this can be achieved without requiring the free node v to force any other node in the process.

This article has been accepted for publication in IEEE Transactions on Control of Network Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCNS.2024.3457582

To illustrate 1-free nodes, consider the example graph $G = (V_g, E_g)$ in Figure 10. Here, $Z'_g = \{v_1, v_2, v_4, v_7\}$ is a 1-LFS. The corresponding 1-free nodes are $R_1(Z'_g) = \{v_6, v_8\}$. To see this, let us examine the zero forcing processes depicted in Figure 10(b). For each node $v \in V_g \setminus Z'_g$, there exists two forcing processes (FPs), where v is forced by two distinct nodes. For instance, node v_3 is forced by v_2 in the FP 1, and by v_5 in FP 3. Similarly, node v_5 is forced by v_4 in FP 1 and by v_3 in FP 4. Notably, nodes v_6 and v_8 do not force any other nodes in any of the mentioned zero forcing processes, thereby qualifying as 1-free nodes corresponding to the given 1-LFS.



(a) G

FP 1	FP 2
$v_1 \to v_8$	v_1
$v_2 \to v_3$	$v_2 \to v_3 \to v_6$
$v_4 \to v_5$	$v_4 \to v_5$
$v_7 \to v_6$	$v_7 \to v_8$
FP 3	**FP 4**
$v_1 \to v_8$	$v_1 \to v_8$
v_2	$v_2 \to v_3 \to v_5$
$v_4 \to v_5 \to v_3$	v_4
$v_7 \to v_6$	$v_7 \to v_6$

(b) Zero forcing processes.

Fig. 10: $Z'_g = \{v_1, v_2, v_4, v_7\}$ is a 1-LFS and $\{v_6, v_8\}$ are the corresponding 1-free nodes determined through the analysis of forcing processes illustrated in (b).

Another interpretation of $R_1(Z'_g)$ is that if $v \in R_1(Z'_g)$, then Z'_g is a ZFS of $G = (V_g, E_g)$ despite the set of leak nodes given by $L = R_1(Z'_g) \cup \{u\}$, where u is an arbitrary node in $V_g \setminus R_1(Z'_g)$. For instance, consider the leak set $L = \{v_6, v_8\} \cup \{v_1\}$ in Figure 10(a). Despite this leak set, $Z'_g = \{v_1, v_2, v_4, v_7\}$ remains a ZFS of G. In this scenario, $R_1(Z'_g) = \{v_6, v_8\}$ represents the set of 1-free nodes. We also note that verifying whether a given set of nodes constitutes a set of 1-free nodes is computationally straightforward. Next, we state the result in Proposition 6.2 and illustrate in Figure 11.

Proposition 6.2. *Let $G = (Vg, E_g)$ and $H = (V_h, E_h)$ be two graphs. Let Z'_g be a 1-LFS of G and $R_1(Z'_g)$ be the set of 1-free nodes. Similarly, consider Z'_h to be a 1-LFS of H. If Z'_h can be partitioned as $X \cup Y$ such that $X \cup (Y \setminus \{y\})$ is a ZFS of H, for all $y \in Y$, then G and H can be combined through a patch E_p to get a graph $\mathcal{G} = (V_g \cup V_h, E_g \cup E_h \cup E_p)$ whose 1-LFS is of size $|Z'_g| + |Z'_h| - \min(|R_1(Z'_g)|, |Y|)$.*

Proof. First, assume that $|R_1(Z'_g)| \leq |Y|$, $R_1(Z'_g) = \{r_1, \cdots, r_m\}$ and $Y = \{y_1, \cdots, y_m, \cdots, y_k\}$. Combine G and H to get a graph $\mathcal{G}$ by adding edges $\{(r_i, y_i) : r_i \in R_1,\ y_i \in Y\}$, $\forall i \in \{1, \cdots, m\}$. Note that $Z'_{\mathcal{G}} = Z'_g \cup X \cup \{y_{m+1}, \cdots, y_k\}$ is of size $|Z'_g| + |Z'_h| - \min(R_1(Z'_g), |Y|)$. We will show that $Z'_{\mathcal{G}}$ is a 1-LFS of the combined graph $\mathcal{G}$. Let u be a leak node in $\mathcal{G}$.

Case a: $u \in V_g$. The only nodes in V_g that are adjacent to any node in V_h are $R_1(Z'_g)$. Also, by the definition of 1-free nodes, none of the nodes in $R_1(Z'_g)$ force any node in V_g. Thus, each node in V_g will be forced by Z'_g, which is a 1-LFS of G, despite a leak $u \in V_g$. Also, since $u \in V_g$, it means none of the nodes in V_h can be a leak. Thus, if black nodes in V_h constitute a ZFS, all nodes in V_h will be forced. Now, we note that at least $m-1$ nodes in $\{y_1, \cdots, y_m\} \subset Z'_h$ will be forced by $R_1(Z'_g)$. Since $\{y_{m+1}, \cdots, y_k\}$ are colored black initially. Thus, there is at most one node in Y that is not black and remaining are colored black. Since $Y \setminus \{y\}$ is a ZFS of V_h for any $y \in Y$ (as given), all nodes in V_h will be forced. As a result all nodes in $V_g \cup V_h$ will be forced by $Z'_{\mathcal{G}}$ despite a leak u.

Case b: $u \in V_h$. It means none of the nodes in V_g is a leak, and all nodes in V_g will be colored black due to Z'_g. Thus, 1-free nodes in $R_1(Z'_{\mathcal{G}})$ will force the nodes in $\{y_1, \cdots, y_m\}$. Since $X \cup \{y_{m+1}, \cdots, y_k\}$ is initially black, all nodes in $X \cup Y$ will be black. Also, $X \cup Y$ is a 1-LFS of V_h, which means all nodes in V_h will be forced eventually despite a leak $u \in V_h$. Thus, all nodes in $V_g \cup V_h$ will be forced by $Z'_{\mathcal{G}} = Z'_g \cup X \cup \{y_{m+1}, \cdots, y_k\}$, which is the desired result.

The case where $|R_1(Z'_g)| > |Y|$ is proven analogously using the same details and is omitted. ■

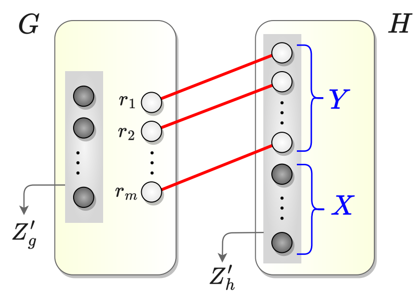
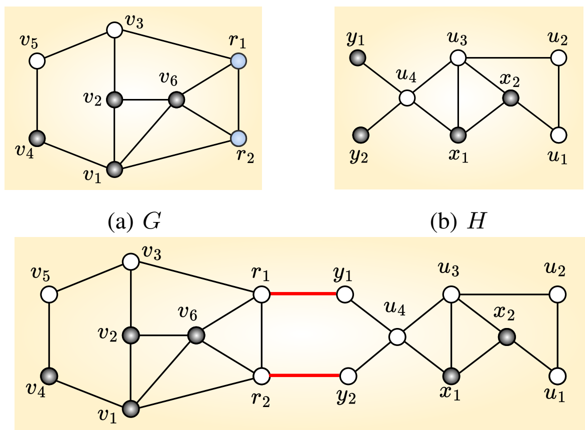


Fig. 11: Illustration of Proposition 6.2.

As an example, consider graphs G and H in Figures 12(a) and (b), respectively. In G, the set $Z'_g = \{v_1, v_2, v_4, v_6\}$ forms a 1-LFS, with the corresponding set of 1-free nodes $R_1(Z'_g) = \{r_1, r_2\}$. Similarly, in H, the set $Z'_h = \{x_1, x_2, y_1, y_2\}$ represents a 1-LFS. Note that $Z'_h \setminus \{y_i\}$, where $y_i \in Y = \{y_1, y_2\}$, is a ZFS of H. Using Proposition 6.2, we combine G and H by adding edges between nodes in $R_1(Z'_g)$ and Y to get a graph $\mathcal{G}$, as shown in Figure 12(c). The set of nodes $\{v_1, v_2, v_4, v_6, x_1, x_2\}$ constitutes a 1-LFS of $\mathcal{G}$ and is a proper subset of $Z'_g \cup Z'_h$.

VII. Conclusion

We studied the problem of maintaining SSC in networks despite misbehaving nodes and edges. We investigated various models of misbehavior aimed at disrupting the zero forcing process in graphs, which in turn impacts the network SSC. One key finding is the equivalence in resilience across different misbehavior models. We demonstrated that a network's resilience to one type of misbehaving node or edge extends to resilience against other threat types. This insight simplifies the leader selection process, facilitating the identification of effective strategies for ensuring SSC in the face of diverse adversities. Moreover, we discussed leader selection strategies to guarantee resilient SSC and investigated the integration of multiple graphs while maintaining SSC with a reduced leader



(a) G (b) H

(c) $\mathcal{G}$

Fig. 12: Combining G and H to obtain $\mathcal{G}$. The added edges are highlighted in red. Notably, 1-LFS of $\mathcal{G}$ is a proper subset of the union of the 1-LFS of G and H.

set compared to individual networks. However, we acknowledge the inherent challenge of resilience, as it often demands a significant number of additional leaders to counteract the impact of a few misbehaving nodes and edges.

In the future, we aim to enhance network resilience against faults and adversarial attacks by safeguarding selected nodes and edges. The selection of 'trusted' elements (nodes/links) within the network and analyzing their impact, especially on the number of leaders required for resilient SSC, will be an interesting direction. Additionally, designing efficient heuristics for computing leaders guaranteeing resilient SSC remains an important concern for further exploration.

Acknowledgement

We thank Joseph Alameda, who pointed out that the equivalence between ℓ-LFS and ℓ-EFS (Theorem 4.8) has appeared in [19] independent of our work.

Appendix

Here, we give a proof of Theorem 4.8. First, we present the following lemma.

Lemma A.1. *If V' is a* 1*-EFS, then V' is a* 1*-LFS.*

Proof. Let F be a forcing process with V' as input nodes. Consider $u \in V$ to be be an arbitrary fixed leak. If u is an end node of a forcing chain, then u does not force any node. So, we assume that $u \to v \in F$ for some v. Since V' is 1-EFS, there exists another forcing process F' such that v is not forced by u. Let $x \to v$, for some $x \neq u$. Now, consider F till the point $u \to v$ becomes valid but v is not colored black. Let $\tilde{V}$ be the set of black nodes till this point. Note that $v \notin \tilde{V}$. Note that all nodes in $\mathcal{N}[u] \setminus \{v\}$ are colored black and v is the only node that u can force. Consider $F_u = (F \setminus F(\tilde{V})) \cup F'(\tilde{V})$. By Lemma 4.1, F_u is a valid forcing process with input nodes V' coloring all nodes black. Note that u is not forcing v in F_u, and hence u is not forcing any node in F_u. Thus, V' is a 1-LFS, which is the desired claim. ■

A. Proof of Theorem 4.8

Proof. (ℓ-LFS $\to$ ℓ-EFS) We will prove using induction on ℓ. From Lemma 4.5, if V' is 1-LFS, then it is 1-EFS. So, our induction hypothesis is, if V' is $(\ell-1)$-LFS, then V' is $(\ell-1)$-EFS. Now, assume that V' is ℓ-LFS. Thus, V' must be $(\ell-1)$-LFS implying that it is also $(\ell-1)$-EFS (by our induction hypothesis). Let E_ℓ be a set of ℓ non-forcing edges. By Lemma 4.6, there is a forcing process F such that for each $e = (u, v) \in E_\ell$, either both end nodes u and v are black, or one end node, say u, is black with $N[u] \setminus \{v\}$ also colored black. Let $\tilde{V}$ be the set of black nodes with the forcing process F. Next, for each $e \in E_\ell$, we consider one of its black end node as a leak, and denote the set of leaks by L. There will be at most ℓ leaks. We observe that a black colored leak node can be ignored and deleted without altering the zero forcing behavior of the other black nodes. So, we consider $G' = G \setminus \{L\}$. Since G is ℓ-LFS with V', $\tilde{V} \setminus L$ is a ZFS of G'. As a result, all nodes in V are colored black while considering E_ℓ as non-forcing edges, implying V' is an ℓ-EFS.

(ℓ-EFS $\to$ ℓ-LFS) Again, we will use induction on ℓ. For $\ell = 1$, if V' is 1-EFS, then it is 1-LFS (by Lemma A.1). Our induction hypothesis is that V' is $(\ell-1)$-EFS implies it to be $(\ell-1)$-LFS. Now assume V' to be ℓ-EFS. It means V' is $(\ell-1)$-EFS and hence, $(\ell-1)$-LFS (by the induction hypothesis). Consider L to be a set of ℓ leaks. By Lemma 4.7, these leaks are colored black and each of them has at most one white neighbor. Next, we consider the edge between the leak and its white neighbor as a non-forcing edge. There will be at most ℓ such non-forcing edges, which we denote by E_ℓ. Since one end node (leak) of each non-forcing edge is colored black and a leak has at most one white neighbor (which is the other end node of the non-forcing edge), we can safely delete the non-forcing edge without affecting the zero forcing behavior of the black node. Thus, we get $G' = G \setminus E_\ell$. Since V' is ℓ-EFS, it means V' is a ZFS of G'. Thus, all nodes will be colored black despite ℓ leaks. Hence, V' is ℓ-LFS, which proves the desired claim. ■

References

[1] F. Pasqualetti, S. Zampieri, and F. Bullo, "Controllability metrics, limitations and algorithms for complex networks," *IEEE Transactions on Control of Network Systems*, vol. 1, no. 1, pp. 40–52, 2014.

[2] A. Clark, B. Alomair, L. Bushnell, and R. Poovendran, "Submodularity in input node selection for networked linear systems: Efficient algorithms for performance and controllability," *IEEE Control Systems Magazine*, vol. 37, no. 6, pp. 52–74, 2017.

[3] T. H. Summers, F. L. Cortesi, and J. Lygeros, "On submodularity and controllability in complex dynamical networks," *IEEE Transactions on Control of Network Systems*, vol. 3, no. 1, pp. 91–101, 2015.

[4] J. Li, X. Chen, S. Pequito, G. J. Pappas, and V. M. Preciado, "On the structural target controllability of undirected networks," *IEEE Transactions on Automatic Control*, vol. 66, no. 10, pp. 4836–4843, 2020.

[5] J. Ruths and D. Ruths, "Control profiles of complex networks," *Science*, vol. 343, no. 6177, pp. 1373–1376, 2014.

[6] N. Monshizadeh, S. Zhang, and M. K. Camlibel, "Zero forcing sets and controllability of dynamical systems defined on graphs," *IEEE Transactions on Automatic Control*, vol. 59, pp. 2562–2567, 2014.

[7] A. Chapman and M. Mesbahi, "On strong structural controllability of networked systems: A constrained matching approach." in *American Control Conference (ACC)*, 2013, pp. 6126–6131.

[8] S. Zhang, M. Cao, and M. K. Camlibel, "Upper and lower bounds for controllable subspaces of networks of diffusively coupled agents," *IEEE Transactions on Automatic control*, vol. 59, pp. 745–750, 2013.

[9] J. Jia, H. L. Trentelman, W. Baar, and M. K. Camlibel, "Strong structural controllability of systems on colored graphs," *IEEE Transactions on Automatic Control*, vol. 65, no. 10, pp. 3977–3990, 2019.

[10] A. Yazıcıoğlu, W. Abbas, and M. Egerstedt, "Graph distances and controllability of networks," *IEEE Transactions on Automatic Control*, vol. 61, no. 12, pp. 4125–4130, 2016.

[11] H. J. Van Waarde, M. K. Camlibel, and H. L. Trentelman, "A distance-based approach to strong target control of dynamical networks," *IEEE Transactions on Automatic Control*, vol. 62, pp. 6266–6277, 2017.

[12] J. Jia, H. J. Van Waarde, H. L. Trentelman, and M. K. Camlibel, "A unifying framework for strong structural controllability," *IEEE Transactions on Automatic Control*, vol. 66, no. 1, pp. 391–398, 2020.

[13] G. Joseph, S. Moothedath, and J. Lin, "Minimal input structural modifications for strongly structural controllability," *arXiv preprint arXiv:2311.05653*, 2023.

[14] AIM Minimum Rank Special Graphs Work Group, "Zero forcing sets and the minimum rank of graphs," *Linear Algebra and its Applications*, vol. 428, no. 7, pp. 1628–1648, 2008.

[15] S. S. Mousavi, M. Haeri, and M. Mesbahi, "On the structural and strong structural controllability of undirected networks," *IEEE Transactions on Automatic Control*, vol. 63, no. 7, pp. 2234–2241, 2017.

[16] M. Trefois and J.-C. Delvenne, "Zero forcing number, constrained matchings and strong structural controllability," *Linear Algebra and its Applications*, vol. 484, pp. 199–218, 2015.

[17] S. Dillman and F. Kenter, "Leaky forcing: a new variation of zero forcing," *arXiv preprint arXiv:1910.00168*, 2019.

[18] J. S. Alameda, J. Kritschgau, N. Warnberg, and M. Young, "On leaky forcing and resilience," *Discrete Applied Mathematics*, vol. 306, pp. 32–45, 2022.

[19] J. Alameda, J. Kritschgau, and M. Young, "Generalizations of leaky forcing," *Journal of Combinatorics*, vol. 14, no. 4, 2023.

[20] J. S. Alameda, S. Dillman, and F. Kenter, "Leaky forcing: a new variation of zero forcing," *Australasian Journal of Combinatorics*, vol. 88, no. 3, pp. 308–326, 2024.

[21] R. Herrman, "The $(d-2)$-leaky forcing number of Q_d and ℓ-leaky forcing number of $GP(n,1)$," *Discrete Optimization*, vol. 46, 2022.

[22] M. Pirani, A. Mitra, and S. Sundaram, "A survey of graph-theoretic approaches for analyzing the resilience of networked control systems," *arXiv preprint arXiv:2205.12498*, 2022.

[23] S. S. Mousavi, M. Haeri, and M. Mesbahi, "Robust strong structural controllability of networks with respect to edge additions and deletions," in *American Control Conference (ACC)*, 2017, pp. 5007–5012.

[24] Y. Zhang and T. Zhou, "Minimal structural perturbations for controllability of a networked system: Complexities and approximations," *International Journal of Robust and Nonlinear Control*, vol. 29, no. 12, pp. 4191–4208, 2019.

[25] W. Abbas, M. Shabbir, Y. Yazıcıoğlu, and X. Koutsoukos, "Edge augmentation with controllability constraints in directed Laplacian networks," *IEEE Control Systems Letters*, vol. 6, pp. 1106–1111, 2021.

[26] S. Moothedath, P. Chaporkar, and M. N. Belur, "Optimal network topology design in composite systems for structural controllability," *IEEE Transactions on Control of Network Systems*, vol. 7, no. 3, pp. 1164–1175, 2020.

[27] N. Popli, S. Pequito, S. Kar, A. P. Aguiar, and M. Ilić, "Selective strong structural minimum-cost resilient co-design for regular descriptor linear systems," *Automatica*, vol. 102, pp. 80–85, 2019.

[28] S. Pequito, G. Ramos, S. Kar, A. P. Aguiar, and J. Ramos, "The robust minimal controllability problem," *Automatica*, vol. 82, pp. 261–268, 2017.

[29] C.-L. Pu, W.-J. Pei, and A. Michaelson, "Robustness analysis of network controllability," *Physica A: Statistical Mechanics and its Applications*, vol. 391, no. 18, pp. 4420–4425, 2012.

[30] G. Ramos, D. Silvestre, and C. Silvestre, "The robust minimal controllability and observability problem," *International Journal of Robust and Nonlinear Control*, vol. 31, no. 10, pp. 5033–5044, 2021.

[31] Y. Zhang, "Controllability robustness under actuator failures: Complexities and algorithms," *arXiv preprint arXiv:1812.07745*, 2018.

[32] Y.-P. Liang, J. Li, and S.-J. Xu, "On extremal graphs for zero forcing number," *Graphs and Combinatorics*, vol. 38, no. 6, p. 185, 2022.

[33] T. Kalinowski, N. Kamcev, and B. Sudakov, "The zero forcing number of graphs," *SIAM Journal on Discrete Mathematics*, vol. 33, no. 1, pp. 95–115, 2019.

[34] M. Gentner and D. Rautenbach, "Some bounds on the zero forcing number of a graph," *Discrete Applied Mathematics*, vol. 236, pp. 203–213, 2018.

[35] G. Hernández, J. Ranilla, and S. Ranilla-Cortina, "Zero forcing in triangulations," *Journal of Computational and Applied Mathematics*, vol. 354, pp. 123–130, 2019.

[36] B. Brimkov, D. Mikesell, and I. V. Hicks, "Improved computational approaches and heuristics for zero forcing," *INFORMS Journal on Computing*, 2021.

[37] B. Brimkov and I. V. Hicks, "Complexity and computation of connected zero forcing," *Discrete Applied Mathematics*, vol. 229, pp. 31–45, 2017.

[38] B. Brimkov, C. C. Fast, and I. V. Hicks, "Computational approaches for zero forcing and related problems," *European Journal of Operational Research*, vol. 273, no. 3, pp. 889–903, 2019.

[39] M. Gentner, L. D. Penso, D. Rautenbach, and U. S. Souza, "Extremal values and bounds for the zero forcing number," *Discrete Applied Mathematics*, vol. 214, pp. 196–200, 2016.

[40] W. Abbas, M. Shabbir, Y. Yazıcıoğlu, and X. Koutsoukos, "On zero forcing sets and network controllability–computation and edge augmentation," *IEEE Transactions on Control of Network Systems*, 2023.

[41] S. Li and W. Sun, "On the zero forcing number of a graph involving some classical parameters," *Journal of Combinatorial Optimization*, vol. 39, no. 2, pp. 365–384, 2020.

[42] F. Barioli, W. Barrett, S. M. Fallat, H. T. Hall, L. Hogben, B. Shader, P. Van Den Driessche, and H. Van Der Holst, "Zero forcing parameters and minimum rank problems," *Linear Algebra and its Applications*, vol. 433, no. 2, pp. 401–411, 2010.

[43] I. V. Hicks and B. Brimkov, "The many face(t)s of zero forcing," *Notices of the American Mathematical Society*, vol. 71, no. 2, 2024.

[44] W. Abbas, "Resilient strong structural controllability in networks using leaky forcing in graphs," in *American Control Conference (ACC)*, 2023.

[45] A. Aazami, "Hardness results and approximation algorithms for some problems on graphs," Ph.D. dissertation, University of Waterloo, 2008.

[46] A. Agra, J. O. Cerdeira, and C. Requejo, "A computational comparison of compact MILP formulations for the zero forcing number," *Discrete Applied Mathematics*, vol. 269, pp. 169–183, 2019.

Waseem Abbas is an Assistant Professor in the System Engineering Department at the University of Texas at Dallas, TX, USA. Previously, he was a Research Assistant Professor at the Vanderbilt University, Nashville, TN, USA. He received Ph.D. (2013) and M.Sc. (2010) degrees, both in Electrical and Computer Engineering, from Georgia Institute of Technology, Atlanta, GA, and was a Fulbright scholar from 2009 to 2013. His research interests are in the areas of control of networked systems, resilience and robustness in networks, graph machine learning, and and graph methods in complex networks.