Edge Augmentation with Controllability Constraints in Directed Laplacian Networks

Waseem Abbas, Mudassir Shabbir, Yasin Yazıcıoğlu and Xenofon Koutsoukos

Abstract— In this paper, we study the maximum edge augmentation problem in directed Laplacian networks to improve their robustness while preserving lower bounds on their strong structural controllability (SSC). Since adding edges could adversely impact network controllability, the main objective is to maximally densify a given network by selectively adding missing edges while ensuring that SSC of the network does not deteriorate beyond certain levels specified by the SSC bounds. We consider two widely used bounds: first is based on the notion of zero forcing (ZF), and the second relies on the distances between nodes in a graph. We provide an edge augmentation algorithm that adds the maximum number of edges in a graph while preserving the ZF-based SSC bound, and also derive a closed-form expression for the exact number of edges added to the graph. Then, we examine the edge augmentation problem while preserving the distance-based bound and present a randomized algorithm that guarantees an α–approximate solution with high probability. Finally, we numerically evaluate and compare these edge augmentation solutions.

Index Terms— Edge augmentation, structural controllability, zero forcing, graph distances.

I. INTRODUCTION

In a networked multi-agent system, a frequent approach to improve network connectivity is to systematically increase interconnections between agents. On the one hand, edge augmentation is useful for improving network connectivity, robustness to link failures, and resilience to malicious intrusions, but on the other hand, adding edges could adversely impact network controllability [1]–[4]. In this paper, we study the problem of maximum edge augmentation in a directed network of agents with Laplacian dynamics while preserving the controllability specification. We consider the network's strong structural controllability (SSC), which depends (apart from the set of input nodes) only on the structure of the underlying graph defined by the edge set of the graph. To measure how much of the network is strong structurally controllable with a given set of leader (input) nodes, the concept of the dimension of strong structurally controllable subspace (SSCS) is typically used (Section II-B). The exact computation of the dimension of SSCS is a hard task, so various graph-theoretic bounds on the dimension of SSCS have been proposed in the literature. We utilize two widely used bounds that are based on the ideas of *Zero Forcing (ZF)* [5]–[8] and *distances between nodes* in graphs [9], [10]. We discuss these bounds in detail in Sections III-A and IV-A, respectively. Our main objective is to *add the*

maximum number of edges in a given directed graph while preserving the lower bound (ZF-based or distance-based) on the dimension of SSCS. Our contributions are listed below.

1) We present an optimal edge augmentation algorithm for adding the maximum number of edges in a directed graph while preserving the ZF-based bound on the dimension of SSCS. We analyze the algorithm and provide a closed-form expression for the number of edges added in the graph.

2) We also discuss edge augmentation in graphs that preserves the distance-based bound on the dimension of SSCS. For a given node pair (u, v) in a directed graph, we characterize the optimal solution of the distance preserving edge augmentation problem in which the objective is to add maximum edges in a graph without changing the distance from node u to node v.

3) We then provide a randomized algorithm that adds maximal edges in a directed graph while preserving the distance-based bound on the dimension of SSCS. We also analyze the approximation ratio of the algorithm.

4) Finally, we numerically evaluate and compare various edge augmentation solutions.

We studied the edge augmentation problem while preserving the distance-based bound on SSC in *undirected* networks in [11]. In this paper, we focus on *directed networks* and consider both the ZF-based bound and the distance-based bound on the dimension of SSCS. It is worth emphasizing that the edge augmentation problem differs fundamentally between directed and undirected networks. This is due to the fact that adding an edge between nodes u and v in an undirected network means adding two directed edges (from u to v and v to u) simultaneously. However, no such constraint exists when adding edges to directed networks. This work is also related to [3], which only considers directed networks that are strong structurally controllable and studies the problem of adding edges while retaining their SSC. Our setup is more general because we consider directed networks that are not necessarily strong structurally controllable. We note that such a setup is very relevant to the notion of *target controllability* in linear networks, where the goal is to control only a subset of agents (targets) instead of the entire network. Since controlling the entire network might not be required in certain applications and could be costly, it is desired to control only target nodes (for instance, [6], [12], [13]). Moreover, this "partial controllability" specification could improve other network attributes, which could not be improved otherwise due to complete controllability requirements, for instance, robustness to external perturbations (which could be achieved through link additions).

W. Abbas M Shabbir and X. Koutsoukos are with the Electrical Engineering and Computer Science Department at Vanderbilt University, Nashville, TN, USA (Emails: {waseem.abbas,mudassir.shabbir, xenofon.koutsoukos}@vanderbilt.edu). Y. Yazıcıoğlu is with the Department of Electrical and Computer Engineering at the University of Minnesota, Minneapolis, MN, USA (Email: ayasin@umn.edu).

II. PRELIMINARIES AND PROBLEM DESCRIPTION

We consider a network of agents modeled by a directed graph $G = (V, E)$, where V is the set of agents and E is the set of directed edges. An edge from node u to v is denoted by (u, v), and u is the *in-neighbor*, or simply the *neighbor* of v. We use the terms node and agent interchangeably. The set of all neighbors of u, denoted by $\mathcal{N}_u$, is called the *neighborhood* of u. The *distance* from u to v in G, denoted by $d_G(u, v)$, is the number of edges in the shortest directed path from u to v. Accordingly, $d_G(u, u) = 0$ and $d_G(u, v) = \infty$ if there is no directed path from u to v. We may ignore the subscript G when it is clear from the context. The directed graph is *strongly connected* if there is a path from any node to any other node. The *union* of $G = (V, E)$ and $G' = (V', E')$ is $G \cup G' = (V \cup V', E \cup E')$. The edges in a graph are assigned positive weights by some weighting function:

$$w : E \to \mathbb{R}^+, \tag{1}$$

where $\mathbb{R}^+$ is the set of positive real numbers.

A. System Model

Each agent u in the network has a state $x_u \in \mathbb{R}$, and the overall state of the network is $x \in \mathbb{R}^n$, where $n = |V|$. The network dynamics are given by the following equation:

$$\dot{x} = -L_w x + Bu, \tag{2}$$

where L_w is the *weighted Laplacian* matrix of G and defined as $L_w = (\text{Deg} - A_w)$. Here, $A_w \in \mathbb{R}^{n \times n}$ is the *weighted adjacency* matrix of G whose uv^{th} entry is

$$[A_w]_{u,v} = \begin{cases} w(u, v) & \text{if } u \neq v, \text{ and } (u, v) \in E \\ 0 & \text{otherwise,} \end{cases} \tag{3}$$

and $\text{Deg} \in \mathbb{R}^{n \times n}$ is the *degree* matrix defined as

$$[\text{Deg}]_{u,v} = \begin{cases} \sum_{k=1}^{n} [A_w]_{u,k} & \text{if } u = v, \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

In (2), $B \in \mathbb{R}^{n \times m}$ is the input matrix, where m is the number of inputs, which is equal to the number of leader nodes. If $V_\ell = \{\ell_1, \ell_2, \cdots, \ell_m\} \subset V$ is the set of leader nodes, then

$$[B]_{u,v} = \begin{cases} 1 & \text{if node } u \text{ is also a leader } \ell_v \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

B. Strong Structural Controllability (SSC)

A state $x' \in \mathbb{R}^n$ is reachable if there is an input u that can drive the network in (2) from origin (initial state) to x' in a finite amount of time. A network $G = (V, E)$ with edge weights defined by w and leader set V_ℓ is *completely controllable*, that is every point in $\mathbb{R}^n$ is reachable, if and only if the following *controllability matrix* is full rank.

$$\Gamma(L_w, V_\ell) = \begin{bmatrix} B & (-L_w)B & (-L_w)^2 B & \cdots & (-L_w)^{n-1} B \end{bmatrix}.$$

The rank of $\Gamma(L_w, V_\ell)$ defines the *dimension* of the controllable subspace consisting of all the reachable states. A Laplacian network $G = (V, E)$ with a given set of leader nodes is called *strong structurally controllable (SSC)* if it is completely controllable for any choice of w as in (1). At the same time, the *dimension of strong structurally controllable*

subspace (SSCS), denoted by $\gamma(G, V_\ell)$, is the *minimum rank* of the controllability matrix $\Gamma(L_w, B)$ over all feasible w (edge weights), i.e.,

$$\gamma(G, V_\ell) = \min_w \text{ rank}(\Gamma(L_w, V_\ell)). \tag{6}$$

C. Problem Formulation

The main objective in the paper is to identify the maximum number of missing edges in a given network such that the dimension of SSCS of the network is preserved even after adding those edges. Since computing the dimension of SSCS is computationally challenging, we consider its lower bounds, including the zero forcing (ZF) and distance-based bounds (explained in Sections III and IV, respectively). These bounds are tight and have numerous applications [10]. If δ is a (ZF-based or distance-based) lower bound on the dimension of SSCS of the network, then the goal is to maximally densify the graph while maintaining the dimension of SSCS to be at least δ. Formally, we state the problem below.

Problem Let $G = (V, E)$ be a directed network of agents in which $V_\ell \subseteq V$ be a set of leaders and the network dynamics are defined by (2). Let the dimension of SSCS of the network be at least δ. Then, find the maximum size edge set E' such that $E \subseteq E'$ and the dimension of SSCS of the network $G' = (V, E')$ with the same set of leaders V_ℓ is also at least δ, i.e., $\delta \leq \gamma(G', V_\ell)$.

III. ADDING EDGES THROUGH ZERO FORCING BOUND

In this section, we present an edge augmentation algorithm that optimally adds edges in a network while preserving the zero forcing-based bound on the dimension of SSCS.

A. Zero Forcing (ZF) Bound for SSC

First, we explain the notion of *Zero Forcing* process and its relation to the dimension of SSCS [5]–[7].

Definition *(Zero Forcing Process)* Consider a directed graph $G = (V, E)$ such that each node $v \in V$ is initially assigned either a *white* or *black* color. The following coloring defines the zero forcing process: *if a black colored node $v \in V$ has exactly one white in-neighbor u, then change the color of u to black.* We say that v infected u.[1]

Definition *(Derived Set)* Consider a directed graph $G = (V, E)$ where $V' \subseteq V$ is an initial set of black nodes (also called the *input set*), and apply the zero forcing process until no further color changes are possible. The resulting set of black nodes is the derived set, denoted by $\text{dset}(G, V') \subseteq V$. For a given set of input nodes, the derived set is unique [14]. Moreover, an input set V' is called a *zero forcing set* (ZFS) if $\text{dset}(G, V') = V$.

As an example, consider G in Figure 1, where $V' = \{v_1, v_2\}$ is the set of input nodes. As a result of ZF process, v_1 infects v_3, v_2 infects v_4 and the resulting derived set is $\{v_1, v_2, v_3, v_4\}$.

---

[1]Since an edge (u, v) indicates that the state of node u is influenced by the state of v in our system model (as in (2) and (3)), we use in-neighbors in the ZF process for consistency. If an edge (u, v) indicates that node u influences node v's state, then out-neighbors should be used in the ZF process, as done in some other works.
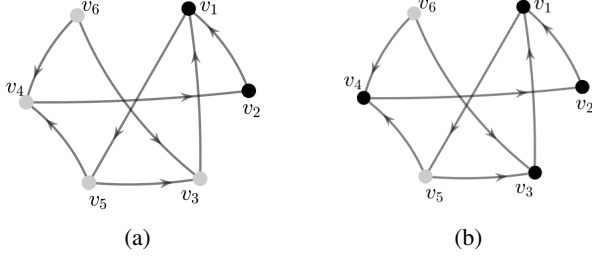
Fig. 1: An example of zero forcing and derived set.

Fig. 2: Example of edge augmentation due to Algorithm 1

.

In the context of SSC, the cardinality of the derived set is significant as it provides a lower bound on the dimension of SSCS, as stated in the following result.

Theorem 3.1: [6] For any network $G = (V, E)$ with the leaders $V_\ell \subseteq V$,

$$\zeta(G, V_\ell) \leq \gamma(G, V_\ell), \tag{7}$$

where $\zeta(G, V_\ell) = |\text{dset}(G, V_\ell)|$ is the size of the derived set corresponding to the input set V_ℓ.

B. Edge Augmentation Algorithm Using ZF

We provide an algorithm to add edges to a directed graph with a given set of leaders. The algorithm ensures that the derived set of the graph remains the same after adding edges, thus, preserving the ZF-based bound on the dimension of SSCS. The proposed algorithm is a modification of the ZF process. In summary, we look for a BLACK node with a single WHITE in-neighbor, add edges incident to the BLACK node that do not change change the size of the derived set, change the WHITE node's color, and then repeat the same procedure until there is no BLACK node with a single WHITE in-neighbor. When this process concludes, we add any extra edges that can be added while preserving the derived set. The algorithm is outlined below. We denote the color of the node v by COLOR(v).

Algorithm 1 ZF-based Edge Augmentation

Given $G = (V, E)$, V_ℓ ▷ V_ℓ is a leader set.
Initialize $G' = (V, E')$, $E' \leftarrow E$
For all $v \in V$, COLOR(v) ← WHITE
For all $v \in V_\ell$, COLOR(v) ← BLACK
while $\exists u$ a BLACK node with a single WHITE in-neighbor v **do**
     COLOR(v) ← BLACK
     For all $w \in V$, with COLOR(w) = BLACK,
     set $E' \leftarrow E' \cup \{(w, u)\}$ ▷ Add edges to u from all BLACK nodes.
end while
For all nodes u not considered in the loop above,
set $E' \leftarrow E' \cup \{(w, u) : w \in V\}$ ▷ Add edges to u from all nodes. **return** $G' = (V, E')$.

Figure 2 illustrates an example of edge augmentation due to Algorithm 1. The derived set in G and G' is same. Next, we show that Algorithm 1 is optimal and adds the maximum number of edges while preserving the size of the derived set returned by the ZF process.
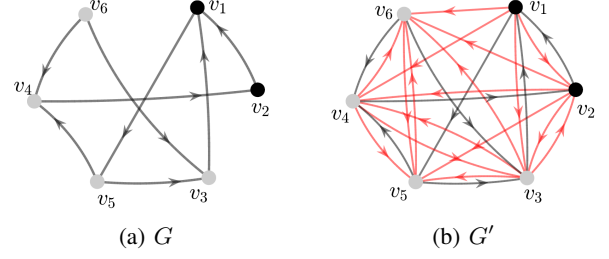
Proposition 3.2: For a directed graph $G = (V, E)$ with a leader set $V_\ell \subseteq V$, let $G' = (V, E')$ be a graph returned by Algorithm 1. Then, $\text{dset}(G, V_\ell) = \text{dset}(G', V_\ell)$.

Proof: Let Δ be the derived set of G with a leader set V_ℓ. In G', all leader nodes in V_ℓ are colored BLACK due to the initial condition of the ZF process. Consider an arbitrary iteration in the ZF process, where a BLACK node u colors its *only* WHITE in-neighbor v BLACK. Algorithm 1 adds edges in G' from (currently) BLACK nodes to u. Since no edge from a currently WHITE node to u is added, v must be the only WHITE in-neighbor of u in G' as well. Thus, the ZF process proceeds by assigning the BLACK color to node v, which is the only WHITE in-neighbor of the BLACK node u in G'. This holds for every iteration in the ZF process. Thus, the ZF process in G and G' will change the colors of nodes exactly the same way. ∎

We can count the number of edges in the directed graph $G' = (V, E')$ returned by Algorithm 1 in terms of $|V|$, $|V_\ell|$ and the size of the derived set.

Proposition 3.3: For a graph $G' = (V, E')$ returned by Algorithm 1 with a leader set V_ℓ and derived set Δ,

$$|E'| \geq \frac{|\Delta|(|\Delta| + 1)}{2} - \frac{m(m+1)}{2} + (m + n - |\Delta|)n - n,$$

where $n = |V|$ and $m = |V_\ell|$.

Proof: In each iteration of the WHILE loop in Algorithm 1, edges from all BLACK nodes to a fixed node are added. There are $|V_\ell| = m$ BLACK nodes when the WHILE loop starts, and this number increases by one in each iteration. Thus, we add $m, m+1, \ldots, |\Delta| - 1$ edges in the WHILE loop. Outside the loop, we add $n - 1$ incoming edges for each of the remaining $n - |\Delta| + m$ nodes. Therefore,

$$|E'| \geq (n - 1)(n - |\Delta| + m) + \sum_{i=m}^{|\Delta|-1} i$$
$$= \frac{|\Delta|(|\Delta| + 1)}{2} - \frac{m(m+1)}{2} + (m + n - |\Delta|)n - n.$$

∎

Theorem 3.4: Let $G = (V, E)$ be a directed graph with $|V| = n$ nodes, leader set V_ℓ and derived set $\Delta = \text{dset}(G, V_\ell)$, then Algorithm 1 returns a graph $G^* = (V, E^*)$ where $E^* \supseteq E$ and $|E^*|$ is maximum while preserving the size of the derived set Δ. Moreover, the number of edges in

the optimal graph is

$$\frac{|\Delta|(|\Delta|+1)}{2} - \frac{m(m+1)}{2} + (m+n-|\Delta|)n - n,$$

where $m = |V_\ell|$.

Proof: Let $G^* = (V, E^*)$ be a graph satisfying the conditions stated in the theorem, and $G' = (V, E')$ be a graph returned by Algorithm 1. We will count the number of edges in G^* and show that $|E^*|$ is upper bound by the expression in Proposition 3.3. Clearly $|E'|$ can not be larger than $|E^*|$, we will get the desired result.

Since G^* preserves the size of the derived set, we should be able to run $|\Delta| - |V_\ell|$ iterations of the ZF process in some arbitrary order. When the ZF process starts, there are m BLACK nodes and $(n-m)$ WHITE nodes. At this point, there must exist a BLACK node u which has only one incoming edge from a WHITE in-neighbor. Therefore, at least $n - m - 1$ edges of the form (v, u), where v is a WHITE node, are missing from G^*. In each iteration, the number of WHITE nodes decreases by exactly one. This means that in the second iteration, at least $(n - m - 2)$ edges are missing, and these edges are distinct from previously counted edges because none of these involve the node u. Similarly, in iteration i, there are at least $(n - m - i)$ distinct edges missing. We can upper bound the number of edges in G^* by subtarting the minimum number of missing edges in the graph from the maximum possible $n^2 - n$ edges. Thus,

$$|E^*| \le n^2 - n - \sum_{i=1}^{|\Delta|-m} (n - m - i) = n^2 - n - \sum_{i=n-m-(|\Delta|-m)}^{n-m-1} i$$

$$= n^2 - n - \frac{(n-m-1)(n-m)}{2} + \frac{(n-|\Delta|)(n-|\Delta|-1)}{2}.$$

Using Proposition 3.3, we get $|E^*| \le |E'|$. However, $|E^*|$ is optimal, which means $|E^*| \ge |E'|$. Thus, we deduce that $|E^*| = |E'|$ and conclude the desired statement. ∎

The time complexity of Algorithm 1 is $O(n^2)$ where n is the number of nodes in the graph. Note that the main task in the algorithm is the addition of edges to the edge set E'. If the edges are kept in an adjacency matrix, each addition takes a constant amount of time. We do not add an edge twice to this set, so the time complexity is bounded by the maximum number of possible edges in a directed graph. We also observe that the time complexity is:

$$\Omega(|E'|) = \Omega(\binom{|\Delta|}{2} + (m+n-|\Delta|)n).$$

When the size of derived set is at least $n/2$, the first term is $\Omega(n^2)$, and when it is less than $n/2$, the second term is $\Omega(n^2)$. Thus, the time complexity of Algorithm 1 is $\Theta(n^2)$.

Remark 3.5: An important observation here is that the number of edges that one can add to a directed graph while preserving the derived set is independent of the topology of the given graph. Note, however, that the size of the derived set in an arbitrary graph is not independent of the topology.

IV. Adding Edges through Distance-based Bound

In this section, first, we review a tight lower bound on the dimension SSCS based on the distances of nodes to leaders in a graph [9]. Second, we present a method to add edges while preserving distances between specific node pairs, which also preserves the bound on the dimension of SSCS. The distance-based bound on the dimension of SSCS is typically better than the ZF-based bound, especially when the network is not strong structurally controllable [10].

A. Distance-based Bound for SSC

Given a network with m leaders $V_\ell = \{\ell_1, \cdots, \ell_m\}$, we define the *distance-to-leaders* (DL) vector of each $v_i \in V$ as

$$D_i = \begin{bmatrix} d(v_i, \ell_1) & d(v_i, \ell_2) & \cdots & d(v_i, \ell_m) \end{bmatrix}^T \in \mathbb{Z}^m.$$

The jth component of D_i, denoted by $[D_i]_j$, is equal to the distance of v_i to ℓ_j. Next, we provide the definition of *pseudo-monotonically increasing* sequences of DL vectors.

Definition (*Pseudo-monotonically Increasing (PMI) Sequence*) A sequence of distance-to-leaders vectors $\mathcal{D}$ is PMI if for every i^{th} vector in the sequence, denoted by $\mathcal{D}_i$, there exists some $\pi(i) \in \{1, 2, \cdots, m\}$ such that

$$[\mathcal{D}_i]_{\pi(i)} < [\mathcal{D}_j]_{\pi(i)}, \quad \forall j > i. \tag{8}$$

In other words, (8) needs to be satisfied for all the subsequent distance-to-leader vectors $\mathcal{D}_j$ appearing after $\mathcal{D}_i$ in the sequence. We say that $\mathcal{D}_i$ satisfies the *PMI property* at coordinate $\pi(i)$ whenever $[\mathcal{D}_i]_{\pi(i)} < [\mathcal{D}_j]_{\pi(i)}, \forall j > i$.

An example of DL vectors is illustrated in Fig. 3. A PMI sequence of length five can be constructed as

$$\mathcal{D} = \left[\begin{bmatrix} \mathbf{0} \\ 3 \end{bmatrix}, \begin{bmatrix} 1 \\ \mathbf{0} \end{bmatrix}, \begin{bmatrix} \mathbf{1} \\ 4 \end{bmatrix}, \begin{bmatrix} 2 \\ \mathbf{1} \end{bmatrix}, \begin{bmatrix} \mathbf{2} \\ 2 \end{bmatrix} \right]. \tag{9}$$

Indices of bold values in (9) are the coordinates, $\pi(i)$, at which the corresponding distance-to-leaders vectors are satisfying the PMI property.

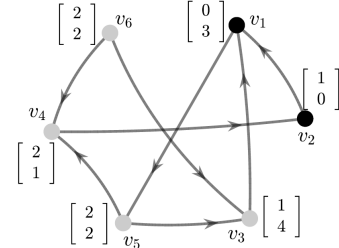
Fig. 3: A network with two leaders, $V_\ell = \{v_1, v_2\}$, and the corresponding distance-to-leaders (DL) vectors.

The longest PMI sequence of DL vectors is related to the dimension of SSCS as stated in the following result.

Theorem 4.1: [9] Consider any network $G = (V, E)$ with the leaders $V_\ell \subseteq V$. Let $\delta(G, V_\ell)$ be the length of the longest PMI sequence of distance-to-leaders vectors with at least one finite entry. Then, $\delta(G, V_\ell) \le \gamma(G, V_\ell)$.

Remark 4.2: While the bound in Theorem 4.1 was presented for connected undirected graphs in [9, Theorem 3.2], it also holds for any choice of leaders on strongly connected directed graphs as shown in [9, Remark 3.1]. Such connectivity properties already ensure that all DL vectors have only finite entries. The bound can be extended easily

to directed graphs without strong connectivity by excluding the DL vectors with all ∞ entries.

B. Adding Edges While Preserving Node Distances

Let $G = (V, E)$ be a graph with a leader set V_ℓ, $\mathcal{D}$ be a PMI sequence of length δ, and $\tilde{V} \subseteq V$ be the set of nodes whose DL vectors are included in $\mathcal{D}$. If we add edges in G to obtain a new graph $G' = (V, E')$ such that the DL vectors of nodes in $\tilde{V}$ remain the same in G', then $\mathcal{D}$ will also be a PMI sequence of G' and $\delta \leq \gamma(G', V_\ell)$. Therefore, one approach to augment edges in a graph while preserving a bound on the dimension of SSCS is to ensure that the distances from a certain set of nodes to leaders do not change due to edge additions. In this direction, we first need to study the maximal edge augmentation in a graph while preserving the distance from a given node a to another node b.

Definition (*Distance Preserving Edge Augmentation (DPEA) Problem*) Given a directed graph $G = (V, E)$ and nodes $a, b \in V$ such that $d_G(a, b) = k$, find a graph $G'(V, E')$ with the (same) node set V and an edge set $E' \supseteq E$ such that $d_{G'}(a, b) = k$ and $|E'|$ is maximized.

Next, we characterize optimal solutions of the DPEA problem for a given node pair (a, b) in G. We show that the optimal solution belongs to a special class of graphs, which is obtained by the union of *clique chains* and *modified clique chains* described below.

Definition (*Directed clique chain*) A directed graph $\mathcal{C}_k = (V, E)$ is a directed clique chain if the node set V can be partitioned into sets $V_0, V_1, V_2, \ldots, V_k$, such that there is an edge from every node in V_i to every node in $V_{i-1} \cup V_i \cup V_{i+1}$ for all $1 \leq i \leq k-1$. Moreover, nodes in each of V_0 and V_k induce cliques.[2]

Definition (*Directed modified clique chain*) A directed graph $\mathcal{M}_k = (V, E)$ is a directed modified clique chain if the node set V can be partitioned into sets $V_0, V_1, V_2, \ldots, V_k$, such that there is an edge from every node in V_i to every node in V_j for all $j \leq i$.

Examples of directed clique chain and directed modified clique chain are shown in Figure 4.
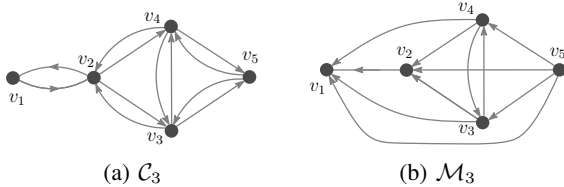

(a) $\mathcal{C}_3$ (b) $\mathcal{M}_3$

Fig. 4: (a) Directed clique chain and (b) directed modified clique chain. The node set V is partitioned into four subsets: $V_0 = \{v_1\}$, $V_1 = \{v_2\}$, $V_2 = \{v_3, v_4\}$ and $V_3 = \{v_5\}$.

Theorem 4.3: Let $G = (V, E)$ be a directed graph and let a, b be two fixed vertices in G with $d_G(a, b) = k$. Then, an augmented graph $G' = (V, E'), E' \supseteq E$, that preserves the distance from a to b, and contains the maximum

[2]In a clique, there is an edge between every pair of nodes.

number of edges is a union of a directed clique chain $\mathcal{C}_k$ and a directed modified clique chain $\mathcal{M}_k$ for some partition $V_0 = \{a\}, V_1, V_2, \ldots, V_k = \{b\}$ of the node set V.

Proof: Let $d_G(a, b) = k$. Then, it is clear that for all vertices $v \in V$, $d_{G'}(a, v) \leq k$ because otherwise we can add an edge from an arbitrary node at distance $\leq k-1$ to v. Also, an arbitrary vertex v in G' at distance i from a, we may assume that $d_{G'}(v, b) = k-i$. Clearly, $d_{G'}(v, b)$ can not be less than $k-i$, and if it is more, we can add an edge from v to a vertex w where $d_{G'}(w, b) = k - i - 1$. Thus, every vertex in G' lies on a shortest path from a to b. Let u, v be two vertices such that $d_{G'}(a, u) = i$, and $d_{G'}(a, v) = j$. We have the following cases:

1) $i = j$ or $i = j-1$: an edge from u to v doesn't change the distance from a to b. Thus, we may assume that all such edges exist in G'.
2) $i < j - 1$: since both of u, v lie on a shortest path from a to b, we know that $d_{G'}(v, b) = k - j$. So an edge u, v will create a path from a to b of distance $\leq j - 2 + 1 + k - j = k - 1$. This is a contradiction to the fact that G' preserves a, b distance. Therefore, G' doesn't contain any edge of the form u, v.
3) $i > j$: in this case, an edge from u to v doesn't create any new shortest paths from a to b. Thus, we may assume that all such edges exist in G'.

Let us define $V_0, V_1, \ldots, V_k$ a partition of V where $V_i = \{v \in V : d_{G'}(a, v) = i\}$. It is clear that with this partition, the graph G is a union of a directed clique chain and a directed modified clique chain. This provides a complete characterization of edges in G' and completes the proof. ∎

From Theorem 4.3, we obtain a simple way to greedily construct a maximally dense graph that preserves the distance from node a to node b in a given graph G as follows:

Algorithm 2 Distance Preserving Edge Augmentation (DPEA)

1: **Given** $G = (V, E)$, $a, b \in V$.
2: **Initialize** $E' \leftarrow E$, $G' \leftarrow (V, E')$
3: **While** $\exists\, (u, v) \in V \times V$ with $d_{G'}(a, u) \geq d_{G'}(a, v) - 1$
4:    $E' \leftarrow E' \cup \{(u, v)\}$, $G' \leftarrow (V, E')$.
5: **End While**
6: **Return** $G' = (V, E')$

An example of the DPEA is shown in Figure 5 in which edges are augmented to preserve the distance from node $a = v_1$ to node $b = v_5$.

We note that the time complexity of Algorithm 2 is $O(n^2 \log n)$ because we can implement this using runs of Dijkstra's shortest path algorithm. We can use DPEA to add edges in a graph $G(V, E)$ with a leader set V_ℓ while preserving a bound on the dimension of SSCS. Let $\mathcal{D}$ be a PMI sequence of length δ and $\tilde{V} \subseteq V$ be the set of nodes whose DL vectors are included in $\mathcal{D}$. By solving the DPEA problem for the node pair (v, l), where $v \in \tilde{V}$ and $l \in V_\ell$, we can obtain edges, say $E_{v,l}$, whose addition to the graph will preserve the distance from node v to leader l. By solving DPEA problem for all node pairs (v, l), where $v \in \tilde{V}$ and $l \in V_\ell$, we can obtain edges that are common in all solutions, that is, $E_{comm} = \cap_{v \in \tilde{V}, l \in V_\ell} E_{v,l}$. By adding
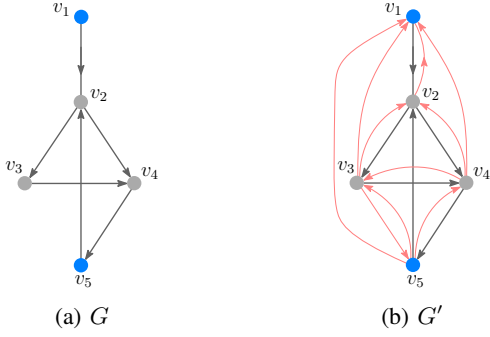
(a) G (b) G'

Fig. 5: Distance preserving edge augmentation for the node pair (v_1, v_5). In (b), red edges are the ones that are added as a result of Algorithm 2.

these common edges in the given graph G, we obtain a new graph $G' = (V, E \cup E_{comm})$ such that $d_G(v, l) = d_{G'}(v, l)$, $\forall v \in \tilde{V}$ and $\forall l \in V_\ell$. Consequently, $\mathcal{D}$ will also be PMI sequence of G', which means that the dimension of SSCS in the augmented graph G' will also be at least δ. Thus, by solving multiple instances of DPEA problem, we can determine edges whose addition to the graph will preserve the distance-based bound on the dimension of SSCS.

Since the problem of finding an optimal set of edges while preserving the distance-based bound is computationally challenging, in the next subsection, we present and analyze a *randomized* edge augmentation algorithm that is simple to understand, easy to implement, and offers good numerical results in our experiments.

C. Randomized Edge Augmentation Algorithm Preserving the Distance-based Bound

First, we discuss that for a given G, if $\mathcal{D}$ is a PMI sequence of length δ containing DL vectors of nodes $\tilde{V} \subseteq V$, then we can add edges in G to obtain an augmented graph G' with the same leader set V_ℓ. Moreover G' will also have a PMI sequence of length δ even if $d_G(v, l) \neq d_{G'}(v, l)$, $\forall v \in \tilde{V}, \forall l \in V_\ell$. In particular, for every node pair (v, l), where $v \in \tilde{V}$, $l \in V_\ell$, there exists an integer $\epsilon_{v,l} \in \mathbb{Z}$ such that if $\epsilon_{v,l} < d_{G'}(v, l) \leq d_G(v, l)$, then G' will have a PMI sequence consisting of DL vectors of nodes in $\tilde{V}$ and having a length δ. We will then provide a randomized edge augmentation algorithm satisfying conditions to preserve the distance-based bound on the dimension of SSCS. Finally, we will analyze the performance of the algorithm.

Let $\mathcal{D}_i$ be the ith vector in the PMI sequence $\mathcal{D}$ of the given graph G. Then by the definition of PMI sequence, for each element $[\mathcal{D}_i]_j$, there is an integer, say $\epsilon_{i,j}$, such that $[\mathcal{D}_i]_j > \epsilon_{i,j}$. We denote the maximum possible value of $\epsilon_{i,j}$ by $\epsilon_{i,j}^*$ and define $\epsilon_i^* := \begin{bmatrix} \epsilon_{i,1}^* & \epsilon_{i,2}^* & \cdots & \epsilon_{i,|V_\ell|}^* \end{bmatrix}^T$. For instance, consider the PMI sequence in (9). The vectors $\mathcal{D}_i$ in the sequence and the corresponding ϵ_i^* are given below.

$$\mathcal{D}_1 = \begin{bmatrix} 0 \\ 3 \end{bmatrix}, \; \epsilon_1^* = \begin{bmatrix} -1 \\ -1 \end{bmatrix}; \quad \mathcal{D}_2 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \; \epsilon_2^* = \begin{bmatrix} 0 \\ -1 \end{bmatrix};$$

$$\mathcal{D}_3 = \begin{bmatrix} 1 \\ 4 \end{bmatrix}, \; \epsilon_3^* = \begin{bmatrix} 0 \\ 0 \end{bmatrix}; \quad \mathcal{D}_4 = \begin{bmatrix} 2 \\ 1 \end{bmatrix}, \; \epsilon_4^* = \begin{bmatrix} 1 \\ 0 \end{bmatrix};$$

$$\mathcal{D}_5 = \begin{bmatrix} 2 \\ 2 \end{bmatrix}, \; \epsilon_5^* = \begin{bmatrix} 1 \\ 1 \end{bmatrix};$$

We can think of $\epsilon_{i,j}^*$ as a strict lower bound on $[\mathcal{D}_i]_j$. From a given PMI sequence of length δ, we can always obtain ϵ_i^* for all $i \in \{1, 2, \cdots, \delta\}$.

Observation 4.4: In a PMI sequence $\mathcal{D}$, if we replace $[\mathcal{D}_i]_j$ by some integer x, where $\epsilon_{i,j}^* < x \leq [\mathcal{D}_i]_j$, then the resulting sequence will still be a PMI sequence.

For instance, consider the sequence $\bar{\mathcal{D}}$ below, which is obtained from $\mathcal{D}$ in (9) by replacing $[\mathcal{D}_1]_2 = 3$ and $[\mathcal{D}_3]_2 = 4$ by $[\bar{\mathcal{D}}_1]_2 = 1$ and $[\bar{\mathcal{D}}_3]_2 = 1$, respectively. Since $\epsilon_{1,2}^* = -1$ and $\epsilon_{3,2}^* = 0$, the resulting sequence $\bar{\mathcal{D}}$ is a PMI sequence.

$$\bar{\mathcal{D}} = \begin{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \begin{bmatrix} 2 \\ 1 \end{bmatrix}, \begin{bmatrix} 2 \\ 2 \end{bmatrix} \end{bmatrix}. \quad (10)$$

Next, we present a randomized algorithm to add edges in a graph G with a leader set V_ℓ. G has a PMI sequence $\mathcal{D}$ of length δ consisting of DL vectors of nodes $\tilde{V} \subseteq V$. Moreover, let $S = \begin{bmatrix} s_1 & s_2 & \cdots & s_\delta \end{bmatrix}$, where s_i is the index of the node whose DL vector is the ith element (vector) in $\mathcal{D}$. For instance, $\mathcal{D}$ in (9) is a PMI sequence of the graph in Figure 3. The corresponding $\tilde{V}$ is $\{v_1, v_2, \cdots, v_5\}$ and S is $\begin{bmatrix} 1 & 2 & 3 & 4 & 5 \end{bmatrix}$. For every i^{th} vector in the sequence, the algorithm first computes the corresponding vector ϵ_i^*, which basically provides minimum distances that need to be maintained from node v_{s_i} to all leaders during the edge augmentation. The algorithm then selects a missing edge randomly and augments it to the graph if its addition does not violate distance conditions (line 8 in Algorithm 3), otherwise discards it. This process is repeated until all missing edges from the original graph are either added to the graph, or discarded. The details are outlined in Algorithm 3.

Algorithm 3 Randomized Algorithm for the Distance-based Edge Augmentation

---

1: **Given** $G = (V, E)$, $V_\ell = \{\ell_1, \cdots, \ell_m\}$, $\mathcal{D}$, S.
2: **Initialize** $E' \leftarrow E$
3: Compute $\epsilon_{i,j}^*$ for each element $[\mathcal{D}_i]_j$ in $\mathcal{D}$.
4: Compute E^c (set of all missing edges).
5: **While** $E^c \neq \emptyset$
6:    Randomly select $e \in E^c$, and obtain $H = (V, E' \cup \{e\})$.
7:    Compute $d_H(v_{s_i}, \ell_j)$ for all $j \in \{1, \cdots, m\}$ and for all $i \in \{1, \cdots, |S|\}$.
8:    If $(\epsilon_{i,j}^* < d_H(v_{s_i}, \ell_j) \leq [\mathcal{D}_i]_j$ for all $j \in \{1, \cdots, m\}$ and for all $i \in \{1, \cdots, |S|\}$, then $E' \leftarrow E' \cup \{e\}$.
9:    Update $E^c \leftarrow E^c \setminus \{e\}$.
10: **End While**
11: **Return** E'

Figure 6(a) illustrates an example of edge augmentation due to Algorithm 3 with the graph G in Figure 3 and a PMI sequence of length 4 as inputs. The red edges indicate the augmented edges. We observe that the graph G' in

Figure 6(a) contains a total of 29 edges compared to the graph in Figure 2(b), which contains a total of 25 edges after the ZF-based edge augmentation.
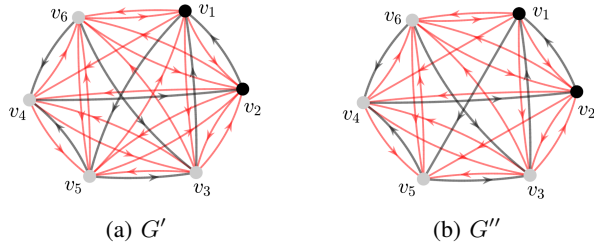

(a) G' (b) G''

Fig. 6: Distance-based edge augmentation due to Algorithm 3. (a) G' is the augmented graph when PMI sequence of length 4 consisting of DL vectors of nodes $v_1, \cdots, v_4$ is the input to the algorithm. (b) G'' is the augmented graph with a PMI sequence of length 5 (given in (9)) as the input.

Remark 4.5: We note that the distance-based bound on the dimension of SSCS is typically better than the ZF-based bound, especially when the graph is not SSC [10]. Thus, edge augmentation using Algorithm 3 allows to add edges while preserving a better bound on the dimension of SSCS. For instance, the distance-based bound on the dimension of SSCS in G is 5, which is better than the ZF-based bound. We see that using Algorithm 3, we can have a total of 27 edges in the augmented graph G'' (as illustrated in Figure 6(b)) while maintaining the dimension of SSCS to be 5, which is not possible by the edge augmentation using the ZF-based bound.

Next, we analyze the performance of Algorithm 3 by the following result.

Proposition 4.6: If we repeat Algorithm 3 a constant c number of times and then return the best graph among these iterations, then the returned graph is an α–approximation with probability at least $1 - e^{-c\left(\frac{OPT}{\beta}\right)^{\alpha \times OPT}}$, where OPT is the optimal number of edges and β is the number of edges that can each be added without changing the PMI.

We omit the proof of this proposition, because it follows the same arguments given in [11, Proposition 4.2] for a similar result on the undirected graphs. Moreover, the time complexity of Algorithm 3 is $O(|E| \times |E^c| \times m \log n)$, where $|E^c|$ is the number of edges not in the input graph. The term $O(|E| \times m \log n)$ is the cost of running Dijkstra's shortest path algorithm for all leader nodes and we do it for all of the missing edges E^c in the randomized algorithm.

V. Numerical Evaluation

We illustrate and compare the ZF-based and the distance-based edge augmentation algorithms on random directed networks with $N = 100$ nodes in which edge (i, j) exists with probability $p = 0.075$, $\forall i \neq j$. Each point in plots in Figure 7 is an average of 30 randomly generated instances.

In Figure 7(a), we plot ZF-based and distance-based bounds on the dimension of SSCS as a function of number of leaders, which are chosen randomly. The distance-based bound is better than the ZF-based, especially for a smaller number of leaders (as discussed in [10]). Figure 7(b) plots the
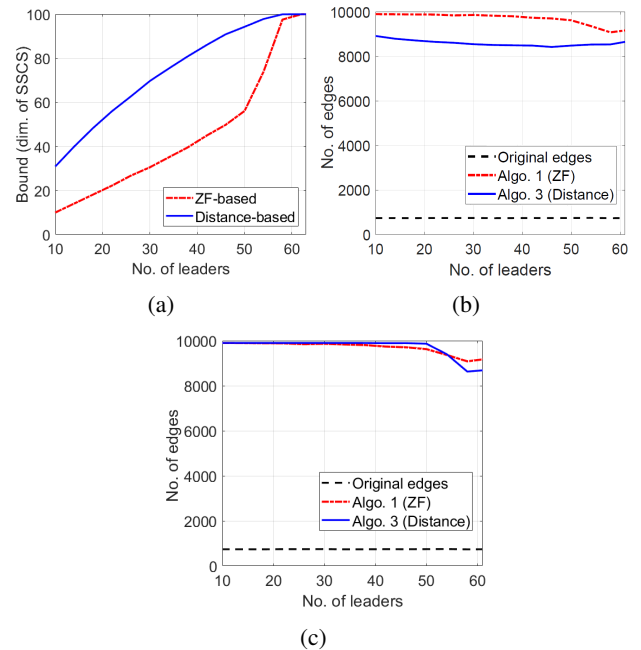

(a)   (b)

(c)

Fig. 7: (a) Lower bounds on the dimension of SSCS. (b) Number of edges added by ZF-based and distance-based augmentation algorithms while preserving their respective bounds. (c) Number of edges added by Algorithms 1 and 3 while preserving the same (ZF-based) bound.

number of edges added in graphs as a function of number of (randomly selected) leaders. We note that Algorithms 1 and 3 augment edges while preserving the ZF-based and distance-based bounds on the dimension of SSCS, respectively. For the same number of leaders, the number of edges augmented by Algorithm 1 is greater than the Algorithm 3 because the controllability bound preserved by the ZF-based augmentation (Algorithm 1) is smaller than the distance-based augmentation. The number of edges in the original graphs are also shown. Figure 7(c) illustrates the result when we augment edges using Algorithms 1 and 3 while preserving the same controllability bound, which is ZFS-based.

VI. Conclusion

Network connectivity and robustness can be improved by augmenting extra links between nodes. However, adding new links may degrade the network's controllability. In this paper, we presented edge augmentation algorithms to add the maximum number of edges in a network while preserving the ZF-based bound and the distance-based bounds on the dimension of SSCS. When the bound on the dimension of SSCS to be preserved is smaller, a large number of edges can be augmented. Though we considered networks with Laplacian dynamic (2), both the distance-based and the ZF-based methods are applicable to more generalized dynamics in the form $\dot{x} = Px + Bu$ (e.g., [5], [12]). In particular, the ZF-based bound holds for any such linear dynamics on a network where an edge (v_i, v_j) denotes that the corresponding entry in the system matrix, P_{ij}, is non-zero. In comparison, the distance-based method requires

the system matrix, P, to be in a class of matrices called the *distance-information-preserving matrices* (as explained in [12]), which contain the graph Laplacian as a special case. We aim to further explore the relation between network controllability and edge density to co-optimize robustness and controllability in the future.

REFERENCES

[1] W. Abbas, M. Shabbir, Y. Yazıcıoğlu, and A. Akber, "Trade-off between controllability and robustness in diffusively coupled networks," *IEEE Transactions on Control of Network Systems*, vol. 7, no. 4, 2020.

[2] F. Pasqualetti, C. Favaretto, S. Zhao, and S. Zampieri, "Fragility and controllability tradeoff in complex networks," in *American Control Conference (ACC)*, 2018, pp. 216–221.

[3] S. S. Mousavi, M. Haeri, and M. Mesbahi, "Strong structural controllability of networks under time-invariant and time-varying topological perturbations," *IEEE Transactions on Automatic Control*, 2020.

[4] G. Lindmark and C. Altafini, "Investigating the effect of edge modifications on networked control systems," *arXiv preprint arXiv:2007.13713*, 2020.

[5] 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.

[6] N. Monshizadeh, K. Camlibel, and H. Trentelman, "Strong targeted controllability of dynamical networks," in *54th IEEE Conference on Decision and Control (CDC)*, 2015, pp. 4782–4787.

[7] 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.

[8] 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, 2018.

[9] A. Y. 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.

[10] A. Y. Yazıcıoğlu, M. Shabbir, W. Abbas, and X. Koutsoukos, "Strong structural controllability of diffusively coupled networks: Comparison of bounds based on distances and Zero Forcing," in *IEEE Conference on Decision and Control (CDC)*, 2020.

[11] W. Abbas, M. Shabbir, H. Jaleel, and X. Koutsoukos, "Improving network robustness through edge augmentation while preserving strong structural controllability," in *American Control Conference (ACC)*, 2020, pp. 2544–2549.

[12] 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, no. 12, 2017.

[13] J. Gao, Y.-Y. Liu, R. M. D'souza, and A.-L. Barabási, "Target control of complex networks," *Nature Communications*, vol. 5, 2014.

[14] A. M. R.-S. G. W. Group, "Zero forcing sets and the minimum rank of graphs," *Linear Algebra and its Applications*, vol. 428, no. 7, pp. 1628–1648, 2008.