Improving Network Robustness through Edge Augmentation While Preserving Strong Structural Controllability

Waseem Abbas, Mudassir Shabbir, Hassan Jaleel, and Xenofon Koutsoukos

***Abstract*— In this paper, we consider a network of agents with Laplacian dynamics, and study the problem of improving network robustness by adding maximum number of edges within the network while preserving a lower bound on its strong structural controllability (SSC). Edge augmentation increases network's robustness to noise and structural changes, however, it could also deteriorate network controllability. By exploiting relationship between network controllability and distances between nodes in graphs, we formulate an edge augmentation problem with a constraint to preserve distances between certain node pairs, which in turn guarantees that a lower bound on SSC is maintained even after adding edges. In this direction, first we choose a node pair and maximally add edges while maintaining the distance between selected nodes. We show that an optimal solution belongs to a certain class of graphs called clique chains. Then, we present and analyze two algorithms to add edges while preserving distances between a certain collection of nodes. Finally, we evaluate our results on various networks.**

***Index Terms*— Strong structural controllability, graph distances, edge augmentation, network robustness.**

I. Introduction

Network controllability has been an active research topic in the broad domain of systems and control as well as in network science in recent years [1]. The main goal is to understand how can we manipulate a network of dynamical agents, often represented by a directed or an undirected graph, by controlling only a small subset of agents, referred to as leaders. In a networked dynamical system, the underlying network topology significantly influences its controllability. Therefore, it is crucial to develop a topological view-point of network controllability [2], [3], [4], [5], [6], [7]. Another important aspect here is to consider the effect of weights given by nodes to each other's information, which is typically modeled by assigning edge weights in the graph. In fact, we say that a network is strong structurally controllable if it is possible to control the entire network with an arbitrary choice of non-zero edge weights. This controllability notion, which is independent of edge weights is particularly useful when exact coupling strengths between nodes are unknown.

Along with the network controllability, we also desire to improve other attributes of a networked dynamical system, in particular, its robustness to changes in the underlying network topology as well as to the noisy information. A network can be made robust by adding more links (edges) between nodes. For instance, a widely used measure of network robustness is Kirchhoff index K_f that measures the effect of structural changes in the network topology as well as the effect of noise on the overall dynamics [8], [9], [10]. It is well known that adding edges always improves network's robustness as measured by K_f. In fact, network robustness increases monotonically with edge additions. Adding edges and densifying a graph is effective on the one hand as it improves network's robustness, but on the other hand it can also deteriorate network controllability [11], [12].

In this paper, we study the problem of maximally adding edges in a network with Laplacian dynamics while preserving its strong structural controllability (SSC). It is easy to verify, in fact in a linear time, whether the entire network is strong structurally controllable. If it is not, computing exactly how much of the network is strong structurally controllable is an extremely challenging problem. Thus, lower bounds on the SSC of networks have been studied in the literature [2], [13], [5], [3], [14], [7]. In this work, we utilize a tight lower bound on SSC, which is based on the topological distances between nodes, and propose algorithms to densify the original network while preserving this lower bound.

Our main contributions are: First, we formulate the problem of adding maximal edges in a network while preserving a lower bound on SSC as an edge augmentation problem in which distances between certain node pairs in a graph need to be maintained. Second, we show that for a fixed node pair, optimal solution to the distance preserving edge augmentation problem belongs to a class of graphs known as clique chains. Third, we provide two algorithms, including a randomized algorithm, to add edges in a graph while preserving distances between a collection of node pairs, thus solving the problem of adding edges while ensuring that a lower bound on SSC is maintained. We also show that the proposed randomized algorithm is an approximation algorithm with high probability. Finally, we numerically evaluate our results on various networks.

Numerous results are available in the context of graph sparsification, where the objective is to remove edges while preserving distances between nodes (e.g., see [15]). However, to the best of our knowledge, this paper is novel in considering the opposite problem, that is the densification of graphs while preserving distances between nodes, and then applying it towards preserving SSC in networks. We mention that a recent paper [16] studies the problem of characterizing

This work was supported in part by the National Institute of Standards and Technology under Grant 70NANB18H198.

W. Abbas and X. Koutsoukos are with the Electrical Engineering and Computer Science Department at the Vanderbilt University, Nashville, TN, USA (Emails: waseem.abbas@vanderbilt.edu, xenofon.koutsoukos@vanderbilt.edu). M. Shabbir is with the Computer Science Department at the Information Technology University, Lahore, Pakistan (Email: mudassir@rutgers.edu). H. Jaleel is with the Electrical Engineering Department at Lahore University of Management Sciences, Lahore, Pakistan (Email: hassan.jaleel@lums.edu.pk).

edges, which if added to a directed network preserve its SSC. However, results hold if the entire network is strong structurally controllable. In our case, we solve the maximal edge addition problem in networks even if they are only partially strong structurally controllable.

The rest of the paper is organized as below: Section II introduces preliminaries. Section III presents the distance preserving edge augmentation problem. Section IV provides algorithms to add edges in a network. Section V evaluates our results, and Section VI concludes the paper.

II. Preliminaries and Problem Description

A. Notations

We consider a network of dynamical agents modeled by a simple undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, in which the node set represents agents and the edge set represents interconnections between agents. Nodes[1] a and b are *adjacent* if there is an edge between them. The *neighborhood* of node a, denoted by $\mathcal{N}_a$ is the set of nodes adjacent to a, and the number of nodes in $\mathcal{N}_a$ is the *degree* of a. The distance between nodes a and b in $\mathcal{G}$, denoted by $d_{\mathcal{G}}(a, b)$, is the number of edges in the shortest path between a and b. The maximum distance between any two nodes in a graph is called its *diameter*. Moreover, edges can be weighted by some weight function, $w : \mathcal{E} \mapsto \mathbb{R}_+$. The edge weight represents coupling strength between nodes.

Agent a has a state $x_a \in \mathbb{R}$. Without loss of generality, the overall state of the network is $x \in \mathbb{R}^n$ where $n = |\mathcal{V}|$. Agents follow the Laplacian dynamics given by

$$\dot{x} = -\mathcal{L}_w \; x + \mathcal{B}u. \tag{1}$$

Here, $\mathcal{L}_w \in \mathbb{R}^{n\times n}$ is a weighted Laplacian matrix of the graph $\mathcal{G}$, and is defined as $\mathcal{L}_w = -\mathcal{A} + \Delta$, where $\mathcal{A}$ is the weighted adjacency matrix in which the ij^{th} entry, that is $\mathcal{A}_{ij}$ is simply the weight on the edge between nodes i and j if such an edge exists, and is zero otherwise. Δ is the diagonal matrix whose i^{th} diagonal entry is simply $\sum_{j=1}^{n} \mathcal{A}_{ij}$. At the same time $\mathcal{B} \in \mathbb{R}^{n\times m}$ is an input matrix, where m is the number of inputs, or simply the number of leaders—nodes with an external control signal. Let $\mathcal{V}_\ell = \{\ell_1, \ell_2, \cdots, \ell_m\} \subset \mathcal{V}$ be the set of leaders, then $\mathcal{B}_{i,j} = 1$ if node $i \in \mathcal{V}$ is also a leader ℓ_j, otherwise $\mathcal{B}_{i,j} = 0$.

B. Strong Structural Controllability (SSC)

A state $x_f \in \mathbb{R}^n$ is reachable if an input exists that can drive the network in (1) from an initial state $x_0 = [0\; 0\; \cdots 0]^T \in \mathbb{R}^n$ to x_f in a finite amount of time. A network is *completely controllable* if every point in $\mathbb{R}^n$ is reachable. Complete controllability of a network $\mathcal{G}(\mathcal{V}, \mathcal{E})$ with given edge weights w and leaders $\mathcal{V}_\ell$ can be checked by computing the rank of the following controllability matrix.

$$\Gamma(\mathcal{L}_w, \mathcal{B}) = \begin{bmatrix} \mathcal{B} & (-\mathcal{L}_w)\mathcal{B} & (-\mathcal{L}_w)^2\mathcal{B} & \cdots & (-\mathcal{L}_w)^{n-1}\mathcal{B} \end{bmatrix}.$$

[1]We use terms node and vertex interchangeably throughout the paper.

The network is completely controllable if and only if $\text{rank}(\Gamma(\mathcal{L}_w, \mathcal{B})) = n$, and the pair $(\mathcal{L}_w, \mathcal{B})$ is called the controllable pair. In fact, the rank of Γ defines the dimension of controllable subspace consisting of all the reachable states.

In a network, if we fix the leaders ($\mathcal{B}$) and the edge set ($\mathcal{E}$), the rank of controllability matrix can change with a different choice of edge weights w', that is the $\text{rank}(\Gamma(\mathcal{L}_w, \mathcal{B}))$ might be different from the $\text{rank}(\Gamma(\mathcal{L}_{w'}, \mathcal{B}))$. A network $\mathcal{G} = ((\mathcal{V}, \mathcal{E})$ is called *strong structurally controllable* with a given leader set $\mathcal{V}_\ell$ if it is completely controllable for any choice of edge weights, or in other words $(\mathcal{L}_w, \mathcal{B})$ is a controllable pair for any choice of w. At the same time, the *dimension of strong structurally controllable subspace*, or simply the *dimension of SSC* is the minimum rank of the controllablility matrix $\Gamma(\mathcal{L}_w, \mathcal{B})$ over all possible edge weights w. Roughly, the dimension of SSC quantifies how much of the network can be controlled by a given leader set with arbitrary edge weights.

C. Problem Description

Our goal is to add a maximum number of edges within the network while preserving the dimension of its strong structurally controllable subspace at the same time. However, computing the exact dimension of SSC with a given set of leaders is a computationally hard problem. As a result, finding good lower bounds on the dimension of such a subspace has been an active research topic. Our approach is to select a tight lower bound, and then add maximal edges within the network while preserving the lower bound on the dimension of SSC. We discuss the lower bound used in this work in the next subsection.

Problem Consider a network of agents $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with the dynamics in (1). Let $\mathcal{V}_\ell \subset \mathcal{V}$ be the leaders and the dimension of SSC of $\mathcal{G}$ with $\mathcal{V}_\ell$ is at least δ. Then, our task is to find a maximum size edge set $\mathcal{E}'$ such that $\mathcal{E} \subseteq \mathcal{E}'$ and the dimension of SSC of the network induced by $\mathcal{V}$ and $\mathcal{E}'$, say $\mathcal{H} = (\mathcal{V}, \mathcal{E}')$, is also at least δ with the same leaders $\mathcal{V}_\ell$.

D. A Tight Lower Bound on SSC Based on Graph Distances

We utilize a lower bound proposed in [5] that is based on the distances between nodes in a graph. Assuming m leaders $\mathcal{V}_\ell = \{\ell_1, \cdots, \ell_m\}$, we define a *distance-to-leader* vector for each node $a \in \mathcal{V}$ in $\mathcal{G}$ as below,

$$D_a = \begin{bmatrix} d_{\mathcal{G}}(\ell_1, a) & d_{\mathcal{G}}(\ell_2, a) & \cdots & d_{\mathcal{G}}(\ell_m, a) \end{bmatrix}^T \in \mathbb{Z}_+^m.$$

The j^{th} component of D_a, denoted by $D_{a,j}$, is the distance between node a and leader j, that is $D_{a,j} = d_{\mathcal{G}}(\ell_j, a)$. Next, we define a sequence of such vectors, called as *pseudo-monotonically increasing* sequence as below.

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

$$\mathcal{D}_{i,\alpha(i)} < \mathcal{D}_{j,\alpha(i)}, \;\; \forall j > i. \tag{2}$$

An example of distance-to-leader vectors is illustrated in Figure 1(a). A PMI sequence of length five is

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

Note that for each vector in the sequence, there is an index—of the circled value—such that the values of all the subsequent vectors at the corresponding index are strictly greater than the circled value, satisfying the condition in (2).

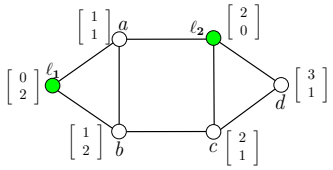


Fig. 1: A network with two leaders $\mathcal{V}_\ell = \{\ell_1, \ell_2\}$. The distance-to-leader vectors of all nodes are also shown. A PMI sequence of length five is $\mathcal{D} = [\mathcal{D}_1 \ \mathcal{D}_2 \ \cdots \ \mathcal{D}_5] = [D_{\ell_1} \ D_{\ell_2} \ D_b \ D_c \ D_d]$.

The length of PMI sequence of distance-to-leader vectors is related to strong structural controllability as below.

Theorem 2.1: [5] If δ is the length of a longest PMI sequence of distance-to-leader vectors in a network, then the dimension of SSC of the network is at least δ.

In [17], we presented a dynamic programming based exact algorithm and an approximate greedy algorithm that returns near optimal PMI sequence of distance-to-leader vectors in $O(mn \log n)$ time where m is the number of leaders and n is the total number of nodes in a graph.

III. Edge Augmentation While Preserving Distances in Graphs

Our approach is to add a maximal edge set to the original graph while ensuring that the maximum lengths of PMI sequences[2] of the resulting and the original graphs remain the same. As a result, we preserve a lower bound on the dimension of SSC in the original graph even after adding edges to it. We note that the bound based on the maximum length of PMI sequence (Theorem 2.1) is tight and numerical evaluation in [17] shows that it generally performs better than the other known bounds, such as the one based on zero forcing sets [13]. Moreover, we provide algorithms to efficiently compute maximum length PMI sequence in [17].

We proceed by letting $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and $\mathcal{H} = (\mathcal{V}, \mathcal{E}')$ respectively denote the original graph and the graph obtained after adding edges, where $\mathcal{E} \subseteq \mathcal{E}'$. At the same time, consider $\mathcal{D}$ to be a PMI sequence of maximum length in $\mathcal{G}$ with $\mathcal{V}_\ell$ leaders. If $\mathcal{D}$ is of length δ, then by Theorem 2.1, the dimension of SSC of $\mathcal{G}$ is at least δ. At the same time, if $\mathcal{H}$ is such that $d_{\mathcal{G}}(\ell, a) = d_{\mathcal{H}}(\ell, a)$, where a is any such node whose distance-to-leader vector is included in $\mathcal{D}$ and ℓ is an arbitrary leader in $\mathcal{V}_\ell$, then $\mathcal{D}$ is also a PMI sequence in $\mathcal{H}$. Consequently, the dimension of SSC of $\mathcal{H}$ is also lower bounded by δ. Thus, to add edges while preserving a lower bound on the dimension of SSC, our approach is to add edges while ensuring that the distance between leaders and a certain subset of nodes (whose distance-to-leader vectors are included in a maximum length PMI sequence of $\mathcal{G}$) are preserved. In fact, if there are m leaders, and the maximum length PMI sequence is $|\mathcal{D}| = \delta$, then we need to preserve distances between $\frac{m(m-1)}{2} + m(\delta - m)$ node pairs. Thus, the problem of edge addition while preserving strong structural controllability becomes the edge augmentation problem in networks while preserving distances between nodes.

A. Adding Edges While Preserving Distance Between Two Nodes

We proceed by fixing a node pair $a, b \in \mathcal{V}$ and finding a maximal edge set, which if added to the original graph, preserves the distance between a and b. We call this as the *Distance Preserving Edge Augmentation (DPEA) problem*, formally stated below. We solve the DPEA problem for all the node pairs v, ℓ where ℓ is a leader and v is a node whose distance-to-leader vector is included in a maximum length PMI sequence $\mathcal{D}$. Taking an intersection of solutions to all the instances of DPEA problem then gives a maximal edge set that can be added to the original graph $\mathcal{G}$ to obtain a new graph $\mathcal{H}$ in which distances between leaders and nodes in the PMI sequence $\mathcal{D}$ are preserved. As a result, $\mathcal{D}$ is also a PMI sequence in $\mathcal{H}$ and a lower bound on the dimension of SSC of $\mathcal{G}$ also holds for $\mathcal{H}$. Next, we formulate and solve the DPEA problem.

Distance Preserving Edge Augmentation (DPEA) Given an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and $a, b \in \mathcal{V}$ such that $d_{\mathcal{G}}(a, b) = k$, we are interested in $\mathcal{H} = (\mathcal{V}, \mathcal{E}')$ where $\mathcal{E} \subseteq \mathcal{E}'$, such that $d_{\mathcal{G}}(a, b) = d_{\mathcal{H}}(a, b) = k$. Our goal is to find $\mathcal{H} = (\mathcal{V}, \mathcal{E}')$ with the maximum $|\mathcal{E}'|$.

We show that an optimal solution to the DPEA problem for a given pair of nodes belongs to a class of graphs known as *clique chains* [9], which we define below.

Definition (*Clique chain)* Let $n_0, n_1, \cdots, n_k \in \mathbb{Z}_+$ and $n = \sum_{i=0}^{k} n_i$, then a clique chain of n nodes is a graph obtained from a path graph of length k by replacing each node with a clique[3] of size n_i such that the vertices in distinct cliques are adjacent if and only if the corresponding vertices in the path graph are adjacent. We denote such a clique chain by $\mathcal{G}_k(n_0, \cdots, n_k)$.

An example of a clique chain is illustrated in Figure 2. Note that the diameter of $\mathcal{G}_k(n_0, \cdots, n_k)$ is k.

Theorem 3.1: For a given $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, and $a, b \in \mathcal{V}$ where $d_{\mathcal{G}}(a, b) > 1$, optimal solution to the DPEA problem is a clique chain of the form $\mathcal{G}_k(n_0 = 1, n_1, \cdots, n_{k-1}, n_k = 1)$, where $\sum_{i=0}^{k} n_i = |\mathcal{V}|$.

Proof: See [18].

Note that when $d_{\mathcal{G}}(a, b) = 1$, optimal solution to the DPEA problem is a complete graph. Next, we construct a

[2]For brevity, we use the term 'PMI sequence' instead of 'PMI sequence of distance-to-leader vectors'.

[3]All vertices in a clique are pair-wise adjacent.

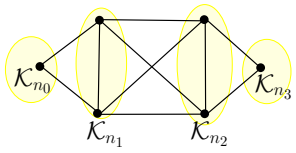


Fig. 2: A clique chain $\mathcal{G}_3(1,2,2,1)$ with $n_0 = 1$, $n_1 = 2$, $n_2 = 2$ and $n_3 = 1$.

clique chain for a given pair of nodes a, b. Note that the original graph $\mathcal{G}$ must be a subgraph of this clique chain.

B. Clique Chain Construction

To construct a clique chain, we define S_i^a to be the set of nodes that are at a distance i from a and similarly define S_i^b. If $d_{\mathcal{G}}(a,b) = k$ is odd, then the sets S_i^v where $i \in \{0,1,\cdots,\lfloor k/2 \rfloor\}$ and $v \in \{a,b\}$ are all non-empty and are pairwise disjoint. Similarly, for even k, the sets S_i^a where $i \in \{0,1,\cdots,k/2\}$ and S_i^b where $i \in \{0,1,\cdots,\frac{k}{2}-1\}$ are all pairwise disjoint and non-empty. Next, we define free and fixed nodes as below:

Definition (*Fixed and free nodes)* For a given pair of nodes $a, b \in \mathcal{V}$, all such nodes that are included in some *shortest* path between a and b are referred to as *fixed* nodes, while the remaining nodes are called *free* nodes.

We note that every fixed node must lie in S_i^v, where $v \in \{a,b\}$. However, there could be free nodes that might not be included in any S_i^a or S_i^b. For instance, if k is even, consider a node x with $d_{\mathcal{G}}(x,a) > k/2$ and $d_{\mathcal{G}}(x,b) > \frac{k}{2}-1$; and if k is odd, consider x such that $d_{\mathcal{G}}(x,a) > \lfloor k/2 \rfloor$ and $d_{\mathcal{G}}(x,b) > \lfloor k/2 \rfloor$. We can always include such a free node x into the set $S^a_{\lfloor k/2 \rfloor}$ by adding an edge between x and y for some $y \in S^a_{\lfloor k/2 \rfloor - 1}$ while preserving the distance k between nodes a and b. Furthermore, if a free node is already included in some S_i^a (or S_i^b), it might be possible to place it in some S_j^a (or S_j^b) for some $j \neq i$ by creating an edge between x and some $y \in S^a_{j-1}$ (or $y \in S^b_{j-1}$) as long as the distance between a and b is maintained. However, if x is a fixed node in some S_i^a (or S_i^b), then it can never be placed in S_j^a (or S_j^b) for any $j \neq i$ without changing the distance between a and b. Moreover, each of the S_i^a and S_i^b contains at least one fixed node. As a result, for given a and b in $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, we always have a partition of $\mathcal{V}$ into $k+1$ subsets given by,

$$\mathcal{S} = \{S_0^a, S_1^a, \cdots, S^a_{\lfloor \frac{k}{2} \rfloor}, S^b_{\lfloor \frac{k}{2} \rfloor}, \cdots, S_1^b, S_0^b\}, \quad \text{(odd } k\text{)}; \quad (3)$$

and

$$\mathcal{S} = \{S_0^a, S_1^a, \cdots, S^a_{k/2}, S^b_{\frac{k}{2}-1}, \cdots, S_1^b, S_0^b\} \quad \text{(even } k\text{)}, \quad (4)$$

where $S_0^a = \{a\}$ and $S_0^b = \{b\}$.

Figure 3 illustrates the above notions. Next, we induce a clique chain over subsets in $\mathcal{S}$, that is $\mathcal{G}_k(1, |S_1^a|, |S_2^a|, \cdots, |S_2^b|, |S_1^b|, 1)$. The distance between a and b in this clique chain is k, and it contains the original graph $\mathcal{G}$ as a subgraph. We illustrate this in Figure 3(d).

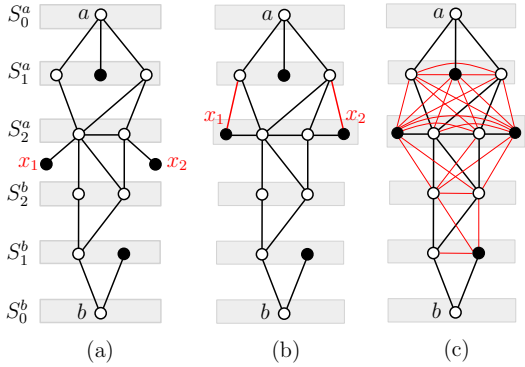


Fig. 3: All white nodes are *fixed* as each one of them lies on some shortest path between a and b, whereas black nodes do not lie on any shortest path between a and b and are *free* nodes. (b) Nodes are partitioned into sets S_i^v where $v \in \{a,b\}$ and $i \in \{0,1,2\}$. Note that free nodes x_1 and x_2 are not included in any S_i^a or S_i^b. (c) Both x_1 and x_2 are included in S_2^a by creating an edge between x_1 and some node in S_1^a, and similarly by an edge between x_2 and a node in S_1^a. The distance between a and b does not change by adding edges. (d) Clique chain is induced over all S_i^a and S_i^b.

As for the time complexity of constructing such a clique chain, a breadth-first search (BFS) tree starting at a (similarly starting at b) gives us distances $d_{\mathcal{G}}(a,v)$ (similarly $d_{\mathcal{G}}(b,v)$) for all vertices v. Once these two distances are available for each node v, we can determine the subset in $\mathcal{S}$ to which v belongs to. Further, we can check if the node is free or fixed by verifying the equation: $d_{\mathcal{G}}(a,b) = d_{\mathcal{G}}(a,v) + d_{\mathcal{G}}(b,v)$. Consequently, for a given graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and a pair of nodes a, b, we can construct the desired clique chain in $O(|\mathcal{V}| + |\mathcal{E}|)$ time.

IV. Edge Augmentation While Preserving a Lower Bound on SSC

In this section, we present two algorithms to add maximal edges in a graph while preserving a (PMI based) lower bound on the dimension of SSC. For a graph $\mathcal{G}$, our algorithms take PMI sequence of distance-to-leader vectors $\mathcal{D}$ as input and return edges whose addition to $\mathcal{G}$ does not change distance-to-leader vectors of nodes included in $\mathcal{D}$. As a result, $\mathcal{D}$ is also a PMI sequence of the graph even after adding edges.

A. Intersection Algorithm

If $\mathcal{D}$ is a PMI sequence and $\bar{\mathcal{V}} \subseteq \mathcal{V}$ is the set of nodes whose distance-to-leader vectors are included in $\mathcal{D}$, then our goal is to add edges that do not change the distance between nodes in the node pair (v, ℓ), where $v \in \bar{\mathcal{V}}$ and $\ell \in \mathcal{V}_\ell$. One way of achieving this is to solve DPEA problem (as discussed in Section III-A) for each such node pair, and then select edges that are common in solutions of all such DPEA instances. These common edges, if added to the graph, will preserve the distance between any leader node and $v \in \bar{\mathcal{V}}$. We summarize this scheme in Algorithm 1.

Proposition 4.1: If δ is a distance-based lower bound on the dimension of SSC of $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with leaders $\mathcal{V}_\ell \subset \mathcal{V}$,

Algorithm 1 Intersection Algorithm

1: Consider a PMI sequence $\mathcal{D}$ in $\mathcal{G}$ with leaders in $\mathcal{V}_\ell$.
2: Identify nodes whose distance-to-leader vectors are included in $\mathcal{D}$, and denote them by $\bar{\mathcal{V}} \subseteq \mathcal{V}$.
3: For each node pair (ℓ, v) where $\ell \in \mathcal{V}_\ell$ and $v \in \bar{\mathcal{V}}$, solve the DPEA problem. Let $\mathcal{E}_{\ell,v}$ be a solution.
4: Compute

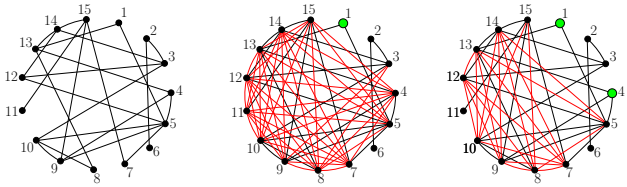
$$\mathcal{E}' = \bigcap_{\ell \in \mathcal{V}_\ell;\ v \in \bar{\mathcal{V}}} \mathcal{E}_{\ell,v}. \tag{5}$$

then δ is also a lower bound on the dimension of SSC of a graph $\mathcal{H} = (\mathcal{V}, \mathcal{E}')$, where $\mathcal{E}'$ is given in (5).

Proof Let $\mathcal{D}$ be a PMI sequence of length δ in $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ containing distance-to-leader vectors of nodes in $\bar{\mathcal{V}} \subseteq \mathcal{V}$. Using the above scheme, we compute $\mathcal{E}'$ and obtain a graph $\mathcal{H} = (\mathcal{V}, \mathcal{E}')$. Note that the distances between leaders and nodes in $\bar{\mathcal{V}}$ is exactly same in $\mathcal{G}$ and $\mathcal{H}$. Consequently, $\mathcal{D}$ is also a PMI sequence in $\mathcal{H}$, and hence δ is also a lower bound on the dimension of SSC of $\mathcal{H}$. ∎

Since there are at most $|\mathcal{V}_\ell| \times |\mathcal{D}|$ instances of the DPEA problem, and each such instance takes $O(|\mathcal{V}| + |\mathcal{E}|)$ time, Algorithm 1 runs in $O(|\mathcal{V}_\ell| \times |\mathcal{D}| \times (|\mathcal{V}| + |\mathcal{E}|))$ time.

Example: As an example, consider the network in Figure 4(a) with 15 nodes and 23 edges. With a single leader ($\mathcal{V}_\ell = \{v_1\}$), the dimension of SSC is at least 6, as the maximum length of a PMI sequence is 6. We can add 41 extra edges (shown in Figure 4(b)), while ensuring that the distance between the leader node v_1 and each of the node in $\bar{\mathcal{V}} = \{v_2, v_3, v_4, v_5, v_6\}$ is preserved in the new graph. Similarly, with two leaders $\mathcal{V}_\ell = \{v_1, v_4\}$, the dimension of SSC is at least 10. We can add 21 extra edges while ensuring that the maximum length of a PMI sequence is at least 10 in the new graph (Figure 4(c)).



(a) (b) $\mathcal{V}_\ell = \{v_1\}$ (c) $\mathcal{V}_\ell = \{v_1, v_4\}$

Fig. 4: Extra edges that can be added to the original graph with one and two leaders while preserving a lower bound on the dimension of SSC. Edges that are added are colored red.

B. Randomized Algorithm

Next, we present a randomized algorithm to add edges while preserving distances between certain node pairs in a graph. Let $\mathcal{E}^c$ be the set of edges not included in the original graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. In other words, $\mathcal{E}^c \cup \mathcal{E}$ induces a complete graph. The main idea is to randomly select an edge in $\mathcal{E}^c$ and add it to the existing graph if its addition does not decreases distances between nodes in the desired node pairs. We outline the algorithm below. As previously, $\mathcal{D}$ is a PMI sequence and $\bar{\mathcal{V}} \subseteq \mathcal{V}$ is the set of nodes whose distance-to-leader vectors are included in $\mathcal{D}$.

Algorithm 2 Randomized Algorithm

1: **Given** $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, $\mathcal{V}_\ell$, $\mathcal{D}$, $\bar{\mathcal{V}}$
2: **Initialize** $\mathcal{E}' \leftarrow \mathcal{E}$
3: Compute $\mathcal{E}^c$.
4: **While** $\mathcal{E}^c \neq \emptyset$
5: Randomly select $e \in \mathcal{E}^c$, and obtain $\mathcal{H} = (\mathcal{V}, \mathcal{E}' \cup \{e\})$.
6: Compute $d_{\mathcal{H}}(\ell, v)$ for all $\ell \in \mathcal{V}_\ell$ and for all $v \in \bar{\mathcal{V}}$.
7: If $(d_{\mathcal{H}}(\ell, v) = d_{\mathcal{G}}(\ell, v))$ for all $\ell \in \mathcal{V}_\ell$, and for all $v \in \bar{\mathcal{V}}$, then

$$\mathcal{E}' \leftarrow \mathcal{E}' \cup \{e\}.$$

8: Update $\mathcal{E}^c \leftarrow \mathcal{E}^c \setminus \{e\}$.
9: **End While**
10: **Return** $\mathcal{E}'$

1) Analysis: In this subsection we analyze the performance of the randomized algorithm. Let $T \leq |\mathcal{E}^c|$ be the number of edges that are (individually) legal to add to the input graph and let $\tau \leq T$ be the size of an (unknown) optimal solution that is, the maximum number of edges from the legal set that can actually be added to the graph. Algorithm 2 randomly picks $\tau' \leq \tau$ legal edges, one at a time, to add to the graph. Next, we show that if the randomized algorithm is repeated c times, then with a certain probability we get a solution that is within a factor of $\alpha < 1$ of the optimal solution.

Proposition 4.2: Algorithm 2 returns an α-approximate solution with probability at least $\left(1 - e^{-c\left(\frac{\tau}{T}\right)^{\alpha\tau}}\right)$, when repeated c times.

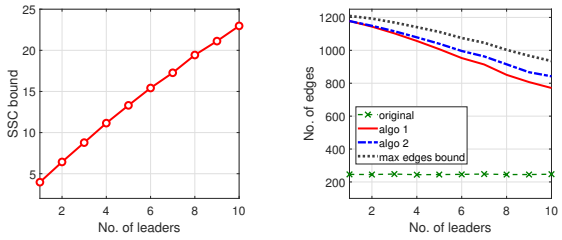
Proof: See [18].

We note that Proposition 4.2 provides a rather loose bound on the probability of success of finding an α-approximate solution when we repeat randomized algorithm c times. We expect many suboptimal solutions of required size to exist that are not subsets of one fixed optimal solution. In any case, this bound is still helpful in guessing a reasonable value of c. For instance, if there are $T = 100$ individually legal edges, τ is $0.92T$, and we are interested in a solution that is at least $\alpha = 3/4$ times the size of optimal solution, then setting $c > 500$ will give us a good chance (success probability of at least 0.8) of finding a desired solution. Regarding the runtime of Algorithm 2, note that in each iteration we need to compute distances from each leader to all nodes in the PMI sequence. Since there are at most $|\mathcal{E}|$ iterations, Algorithm 2 runs in $O(|\mathcal{E}| \times |\mathcal{V}_\ell| \times (|\mathcal{V}| + |\mathcal{E}|))$ time.

V. Numerical Evaluation

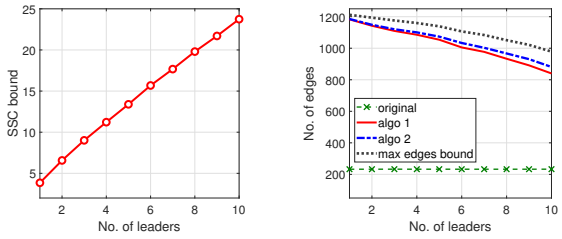
Here, we evaluate our algorithms on Erdős-Rényi (ER) networks in which any two nodes are adjacent with probability p, and Barabási-Albert (BA) networks in which each new node is adjacent to γ existing nodes through a preferential attachment strategy. For both ER and BA models, we consider networks of 50 nodes.

Figure 5 illustrates results for ER graphs. For a selected p, first we plot distance-based lower bound on the dimension of SSC as a function of the number of leaders selected randomly (Figure 5(a)). Then, using Algorithms 1 and 2, we add edges to networks while preserving lower bounds on their dimensions of SSC. We also compare results against an upper bound on the optimal number of edges that can be added without changing PMI sequence. The bound is obtained by observing that an edge cannot be added between two such nodes that lie on a shortest path between a leader and a node whose distance-to-leader vector is included in a given PMI sequence. This observation gives an upper bound on the maximum number of edges that can be added in a graph while preserving a PMI sequence. Figure 5(b) illustrates that new networks obtained after applying Algorithms 1 and 2 contain significantly more edges as compared to the original network. Algorithm 2 performs slightly better than the Algorithm 1, especially when the number of leaders increases. However, our experiments show that as the value of p increases, the difference between two algorithms is negligible. To obtain results of Algorithm 2, we perform 150 repetitions (that is $c = 150$) and then select the best solution. We also mention that Algorithm 1 takes significantly less time as compared to the Algorithm 2 (based on the choice of c). Similar results are obtained for BA networks as illustrated in Figure 6. In all the plots, value at each point is an average of 100 randomly generated instances.



(a) $p = 0.2$ (b) $p = 0.2$

Fig. 5: *ER–Networks:* (a) Lower bound on the dimension of SSC. (b) A comparison of Algorithms 1 and 2.



(a) $\gamma = 5$ (b) $\gamma = 5$

Fig. 6: *BA–Networks:* (a) Lower bound on the dimension of SSC. (b) A comparison of Algorithms 1 and 2.

VI. Conclusion

Adding extra links between nodes improves network's robustness to noisy information and to structural changes in the underlying network topology, but at the same time, these extra links can deteriorate network controllability. We introduced the problem of improving network robustness by adding edges while maintaining a lower bound on the dimension of SSC. By exploring the relationship between strong structural controllability and distances between nodes in a graph, we showed that the above problem can be formulated as an edge augmentation problem with the constraint of maintaining distances between a certain pair of nodes. We believe that characterizing densest graphs that preserve distances between certain node pairs is an interesting problem in its own respect. We aim to study this problem further and design efficient exact algorithms to solve it.

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

[3] 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.

[4] C. O. Aguilar and B. Gharesifard, "Graph controllability classes for the Laplacian leader-follower dynamics," *IEEE Transactions on Automatic Control*, vol. 60, no. 6, pp. 1611–1623, 2015.

[5] 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.

[6] S. S. Mousavi and M. Haeri, "Controllability analysis of networks through their topologies," in *2016 IEEE 55th Conference on Decision and Control (CDC)*. IEEE, 2016, pp. 4346–4351.

[7] 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.

[8] G. F. Young, L. Scardovi, and N. E. Leonard, "Robustness of noisy consensus dynamics with directed communication," in *Proceedings of the American Control Conference*, 2010, pp. 6312–6317.

[9] W. Ellens, F. Spieksma, P. Van Mieghem, A. Jamakovic, and R. Kooij, "Effective graph resistance," *Linear Algebra and its Applications*, vol. 435, no. 10, pp. 2491–2506, 2011.

[10] W. Abbas and M. Egerstedt, "Robust graph topologies for networked systems," in *3rd IFAC Workshop on Distributed Estimation and Control in Netwoked Systems*, 2012, pp. 85–90.

[11] W. Abbas, M. Shabbir, A. Yazıcıoğlu, and A. Akbar, "On the trade-off between controllability and robustness in networks of diffusively coupled agents," in *American Control Conference (ACC)*, 2019.

[12] 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.

[13] 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.

[14] 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.

[15] B. Bollobás, D. Coppersmith, and M. Elkin, "Sparse distance preservers and additive spanners," *SIAM Journal on Discrete Mathematics*, vol. 19, no. 4, pp. 1029–1055, 2005.

[16] 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.

[17] M. Shabbir, W. Abbas, and Y. Yazicioglu, "On the computation of a lower bound on strong structural controllability in networks," in *58th IEEE Conference on Decision and Control*, 2019. [Online]. Available: https://arxiv.org/abs/1909.03565

[18] W. Abbas, M. Shabbir, H. Jaleel, and X. Koutsoukos, "Improving network robustness through edge augmentation while preserving strong structural controllability," 2020. [Online]. Available: https://arxiv.org/abs/2003.05490