Computation of the Distance-based Bound on Strong Structural Controllability in Networks

Mudassir Shabbir, Waseem Abbas, *Member, IEEE*, A. Yasin Yazıcıoğlu, *Member, IEEE*, Xenofon Koutsoukos, *Fellow, IEEE*

***Abstract*—In this paper, we study the problem of computing a tight lower bound on the dimension of the strong structurally controllable subspace (SSCS) in networks with Laplacian dynamics. The bound is based on a sequence of vectors containing the distances between leaders (nodes with external inputs) and followers (remaining nodes) in the underlying network graph. Such vectors are referred to as the distance-to-leaders vectors. We give exact and approximate algorithms to compute the longest sequences of distance-to-leaders vectors, which directly provide distance-based bounds on the dimension of SSCS. The distance-based bound is known to outperform the other known bounds (for instance, based on zero-forcing sets), especially when the network is partially strong structurally controllable. Using these results, we discuss an application of the distance-based bound in solving the leader selection problem for strong structural controllability. Further, we characterize strong structural controllability in path and cycle graphs with a given set of leader nodes using sequences of distance-to-leaders vectors. Finally, we numerically evaluate our results on various graphs.**

***Index Terms*—Strong structural controllability, network topology, graph algorithms, dynamic programming.**

I. Introduction

Network controllability has been an important research topic in network science and control. The notion of strong structural controllability accounts for the controllability of all such networks that have the same *structure* of an underlying network graph, that is, networks having the same vertex and edge sets but possibly different (non-zero) edge weights. A network is strong structurally controllable (SSC) with a given set of input (leader) nodes if it is controllable for any choice of (non-zero) edge weights in the underlying network graph. There exist efficient algorithms to verify the strong structural controllability of networks [2]–[5]. If a network is not SSC with a given set of leader nodes, it is of interest to determine how far the network is from becoming SSC, or roughly speaking, how much of the network is always controllable. More formally, this issue is concerned with computing the dimension of the *strong structurally controllable subspace (SSCS)* (Section II-A). The complexity of this problem is unknown to the best of our knowledge. However, we suspect that the problem is NP-hard considering the size and generality of search space and similarity to known NP-hard problems involving minimum rank computation, for instance, [6]–[8].

In this paper, we study the problem of computing a tight lower bound on the dimension of SSCS of networks with Laplacian dynamics. Since the exact computation of is challenging, various bounds have been proposed in the literature [5], [9]–[12]. Here, we consider a tight lower bound proposed in [10], which relates the notion of strong structural controllability to the distances between nodes in the underlying network graph. In [13], we compare this *distance-based* bound with another widely used bound based on the notion of zero-forcing sets [5], [12], [14], [15]. Our analysis in [13] shows that the distance-based bound is typically better than the zero-forcing-based bound, especially when the network is not completely strong structurally controllable. Additionally, the distance-based bound can be applied in exploring the trade-off between controllability and robustness in networks with Laplacian dynamics [16], edge augmentation in networks while preserving their strong structural controllability [17] and designing a leader selection algorithm [10]. It also has applications for target controllability in linear networks, where the goal is to control a subset of agents (targets) instead of the entire network by injecting input through leader nodes [11], [14]. Despite advantages, efficient computation of the distance-based bound has been an issue, especially in large networks.

To compute the distance-based bound on the dimension of SSCS, an algorithm has been presented in [10] that takes $O(m^n)$ time, where n is the total number of nodes in the network and m is the number of leader nodes. Here, we present an algorithm that takes $O(m(n \log n + n^m))$ time to compute the distance-based bound, which is a significant improvement. We note that for a fixed number of leaders, the algorithm is polynomial in the number of nodes. For instance, in the case of two leaders, our algorithm takes $O(n^2)$ time as compared to the $O(2^n)$ runtime of the algorithm in [10]. When the number of leaders is on the order of n, the algorithm will take exponential time. For such cases, we also present a *greedy* algorithm that approximates the distance-based bound and runs in $O(mn \log n)$ time. In our experiments, we observe that the bound returned by the greedy algorithm is very close to the optimal in almost all cases.

The main idea of the distance-based bound is to obtain distances between leaders and other nodes, arrange them in vectors called *distance-to-leaders vectors*, and then construct a sequence of such vectors, called as *Pseudo-monotonically increasing (PMI) sequence*, which satisfies some monotonicity

M. Shabbir is with the Computer Science Department at Information Technology University, Lahore, and the Computer Science Department at the Vanderbilt University, Nashville, TN, USA. E-mail: mudassir.shabbir@vanderbilt.edu. W. Abbas is with the Department of Systems Engineering at the University of Texas at Dallas, Richardson, TX, USA. E-mail: waseem.abbas@utdallas.edu. A. Y. Yazıcıoğlu is with the Department of Electrical and Computer Engineering at the University of Minnesota, Minneapolis, MN, USA. E-mail: ayasin@umn.edu. X. Koutsoukos is with the Computer Science Department at the Vanderbilt University, Nashville, TN, USA. E-mail: xenofon.koutsoukos@vanderbilt.edu.

Some preliminary results appeared in [1].

conditions (as explained in Section II-B). Computing distances between nodes is straightforward; however, constructing an appropriate PMI sequence, whose length provides a bound on the dimension of SSCS, is computationally challenging. We provide efficient algorithms with performance guarantees to compute such sequences.

Contributions: We provide dynamic programming-based exact algorithm that runs in $O(m(n\log n+n^m))$ time to compute an optimal PMI sequence of distance-to-leaders vectors consisting of distances between leaders and other nodes. Here m and n denotes the number of leaders and nodes, respectively. The length of the sequence directly gives a tight lower bound on the dimension of SSCS of networks with Laplacian dynamics. We also propose an approximation algorithm that computes a near-optimal PMI sequence of distance-to-leaders vectors in practice and takes $O(mn\log n)$ time. If there exists a PMI sequence of distance-to-leaders vectors of length n, then the network is strong structurally controllable and the greedy algorithm always returns such a sequence. Further, We analyze PMI sequences of distance-to-leaders vectors in paths and cycles with arbitrary leaders. We also discuss the application of distance-based bound in solving the leader selection problem for strong structural controllability and also provide a numerical evaluation of results.

The rest of the paper is organized as follows: Section II introduces notations and preliminary concepts. Section III provides dynamic programming-based exact algorithm to compute a distance-based bound on the dimension of SSCS. Section IV presents and analyzes a greedy approximation algorithm. Section V discusses application of the bound to the leader selection problem. Section VI discusses cases of path and cycle graphs. Section VII provides a numerical evaluation of results, and Section VIII concludes the paper.

II. Preliminaries

We consider a network of n dynamical agents represented by a simple (loop-free) undirected graph $G=(V,E)$ where the node set $V=\{v_1,v_2,\dots,v_n\}$ represents agents, and the edge set E represents interconnections between agents.[1] An edge between $v_i,v_j\in V$ is denoted by e_{ij}. The *neighborhood* of node v_i is $\mathcal{N}_i\triangleq\{v_j\in V: e_{ij}\in E\}$. The *distance* between v_i and v_j, denoted by $d(v_i,v_j)$, is simply the number of edges on the shortest path between v_i and v_j. $\mathbb{R}^+$ is the set of positive real numbers. The *weight function*

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

assigns positive weight $w(e_{ij})$ to the edge e_{ij}. These weights define the coupling strength between nodes.

Each agent $v_i\in V$ has a state $x_i(t)\in\mathbb{R}$ at time t and the overall state of the system is $x(t)=\begin{bmatrix} x_1(t) & x_2(t) & \cdots & x_n(t)\end{bmatrix}^T\in\mathbb{R}^n$. The agents update states following the Laplacian dynamics,

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

[1]The results presented can be extended to directed networks in a straightforward manner as the distance-based bound on the dimension of SSCS holds true for directed networks also [10, Remark 3.1].

where $L_w\in\mathbb{R}^{n\times n}$ is the weighted *Laplacian matrix* of G and is defined as $L_w=\Delta-A_w$. Here, $A_w\in\mathbb{R}^{n\times n}$ is the weighted *adjacency matrix* defined as

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

and $\Delta\in\mathbb{R}^{n\times n}$ is the *degree matrix* whose entries are

$$[\Delta]_{ij}=\begin{cases}\sum_{k=1}^n[A_w]_{ik} & \text{if } i=j\\ 0 & \text{otherwise.}\end{cases} \tag{4}$$

The matrix $B\in\mathbb{R}^{n\times m}$ in (2) is an *input matrix*, where m is the number of leaders (inputs), which are the nodes to which external control signals are applied. Let $V_\ell=\{\ell_1,\ell_2,\cdots,\ell_m\}\subseteq V$ be the set of *leaders*, then

$$[B]_{ij}=\begin{cases}1 & \text{if } v_i=\ell_j\\ 0 & \text{otherwise.}\end{cases} \tag{5}$$

A. Strong Structural Controllability

A state $x_f\in\mathbb{R}^n$ is a *reachable state* if there exists an input u that can drive the network in (2) from any initial state x_i to x_f in a finite amount of time. A network $G=(V,E)$ in which edges are assigned weights according to the weight function w in (1), and contains $V_\ell\subseteq V$ leaders is called *completely controllable* if every point in $\mathbb{R}^n$ is reachable. Complete controllability can be checked by computing the rank of the *controllability matrix*, $\Gamma(L_w,B)=\begin{bmatrix} B & (-L_w)B & (-L_w)^2B & \cdots & (-L_w)^{n-1}B\end{bmatrix}$. The network is completely controllable if and only if the rank of $\Gamma(L_w,B)$ is n, and in such case (L_w,B) is called a *controllable pair*. The range space of $\Gamma(L_w,B)$ describes the set of all reachable states, also called the controllable subspace. Thus, the rank of $\Gamma(L_w,B)$ is the dimesnion of the controllable subspace. Note that edges in G define the *structure*—location of zero and non-zero entries in the Laplacian matrix—of the underlying graph. The rank of resulting controllability matrix depends on the weights assigned to edges. For a given graph $G=(V,E)$ and leaders V_ℓ, $\text{rank}(\Gamma(L_w,B))$ could be different from $\text{rank}(\Gamma(L_{w'},B))$, where w and w' are two different choices of weight functions.

A network $G=(V,E)$ with V_ℓ leaders is *strong structurally controllable* if and only if (L_w,B) is a controllable pair for any choice of weight function w, or in other words, $\text{rank}(\Gamma(L_w,B))=n$ for all weight functions w. At the same time, the *dimension of strong structurally controllable subspace (SSCS)*, denoted by $\gamma(G,V_\ell)$, is

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

The minimum is taken over all weight functions w in (1). Thus, $\gamma(G,V_\ell)$ is the minimum dimension of the controllable subspace that can be attained from G with V_ℓ leaders and any choice of feasible edge weights.

B. Distance-based Lower Bound on the dimension of SSCS

We use a tight lower bound on the dimension of SSCS as proposed in [10]. The bound is based on the distances between

nodes in a graph. Assuming m leaders $V_\ell = \{\ell_1, \cdots, \ell_m\}$, we define a vector of non-negative integers called as the *distance-to-leaders* vector for a node $v_i \in V$ as

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

The j^{th} component of D_i, denoted by $[D_i]_j$, is $d(\ell_j, v_i)$, the distance between leader ℓ_j and the node v_i. Next, we define a sequence of distance-to-leaders vectors, called as *pseudo-monotonically increasing* sequence below.

Definition (*Pseudo-monotonically Increasing (PMI) Sequence)* Let $\mathcal{D}$ be a sequence of distance-to-leaders vectors and $\mathcal{D}_i$ be the i^{th} vector in the sequence. We denote the j^{th} component of the vector $\mathcal{D}_i$ by $[\mathcal{D}_i]_j$. Then, $\mathcal{D}$ is PMI if for every $\mathcal{D}_i$ in the sequence, there exists some $\pi(i) \in \{1, 2, \cdots, m\}$ such that

$$[\mathcal{D}_i]_{\pi(i)} < [\mathcal{D}_j]_{\pi(i)}, \ \ \forall j > i, \tag{7}$$

i.e., the above condition needs to be satisfied for all the subsequent distance-to-leader vectors $\mathcal{D}_j$ appearing after $\mathcal{D}_i$ in the sequence. Here, m is the number of leaders. 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 distance-to-leaders vectors is illustrated in Figure 1. A PMI sequence of length five is

$$\mathcal{D} = \left[\begin{bmatrix} 3 \\ \textcircled{0} \end{bmatrix}, \begin{bmatrix} 2 \\ \textcircled{1} \end{bmatrix}, \begin{bmatrix} \textcircled{0} \\ 3 \end{bmatrix}, \begin{bmatrix} 2 \\ \textcircled{2} \end{bmatrix}, \begin{bmatrix} \textcircled{1} \\ 3 \end{bmatrix} \right]. \tag{8}$$

Indices of circled values in (8) are the coordinates at which the corresponding distance-to-leaders vectors are satisfying the PMI property. The length of the longest PMI sequence of distance-to-leaders vectors is related to the dimension of SSCS as stated in the following result.

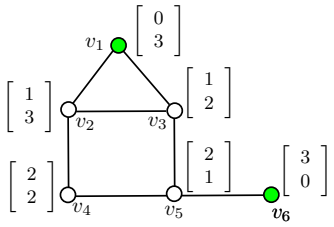


Fig. 1: A network with two leaders $V_\ell = \{\ell_1, \ell_2\} = \{v_1, v_6\}$, along with the distance-to-leaders vectors of nodes. A PMI sequence of length five is $\mathcal{D} = [\mathcal{D}_1 \ \mathcal{D}_2 \ \cdots \ \mathcal{D}_5] = [D_6 \ D_5 \ D_1 \ D_4 \ D_2]$.

Theorem 2.1: [10] If $\delta(G, V_\ell)$ is the length of the longest PMI sequence of distance-to-leaders vectors in a network $G = (V, E)$ with leaders V_ℓ, then

$$\delta(G, V_\ell) \leq \gamma(G, V_\ell). \tag{9}$$

We note that for a given graph $G = (V, E)$ and leader nodes $V_\ell \subseteq V$, the length of the longest PMI sequence describes the minimum dimension of the controllable subspace for any feasible edge weights. In other words, if the length of the longest PMI is $k \leq n$, then the dimension of the controllable subspace of the system is at least k, regardless of the edge weights. Moreover, the bound in (9) is tight, as discussed in [10]. For instance, for path graphs in which one of the end nodes is a leader, and for cycle graphs in which two adjacent nodes are leaders, we have $\delta(G, V_\ell) = \gamma(G, V_\ell)$. the dimension of SSCS and the length of the longest PMI sequence of distance-to-leaders vectors are equal, and hence $\delta(G, V_\ell) = \gamma(G, V_\ell)$. We discuss the length of the longest PMI sequences of distance-to-leaders vectors in path and cycle graphs with arbitrary leaders in Section VI.

Our main goal is to compute a PMI sequence of maximum length, and consequently, a lower bound on the dimension of SSCS. We provide an exact algorithm in Section III and a greedy approximation algorithm in Section IV.

III. Exact Algorithm for the Distance Bound

In this section, we provide a dynamic programming-based exact algorithm to compute a longest PMI sequence of distance-to-leaders vectors and, as a result, a distance-based lower bound on the dimension of SSCS.

We note that each distance-to-leaders vector D_i can be viewed as a point in $\mathbb{Z}^m$, and without loss of generality, we may assume that points D_i are *distinct*. Otherwise, we can throw away multiple copies of the same point since duplicate points can not satisfy the PMI property on any coordinate. The following observation is crucial to our algorithms.

Observation 3.1: Given a set of points $D_1, D_2, \ldots, D_n$, if there exists a point D_i and an index j such that $[D_i]_j < [D_{i'}]_j$ for all $D_i \neq D_{i'}$, then D_i is a unique minimum point in the direction (coordinate) j and there is a longest PMI sequence which starts with D_i. However, it is possible that there is no unique minimum in any direction. This leads us to the definition of a *conflict* and *conflict-partition*.

Definition *(Conflict-partition)* A *conflict* is a set of points $\mathcal{X}$ that can be partitioned into $\mathcal{X}_1, \mathcal{X}_2, \ldots, \mathcal{X}_m$ such that all points $D_p \in \mathcal{X}_j$ have $[D_p]_j = [D_q]_j$ if $D_q \in \mathcal{X}_j$, and $[D_p]_j \leq [D_q]_j$ if $D_q \notin \mathcal{X}_j$. Further, $|\mathcal{X}_j| > 1$ for all j. Such a partition is called *conflict-partition* or *c-partition* for short.[2]

An example of conflict is illustrated in Figure 2.

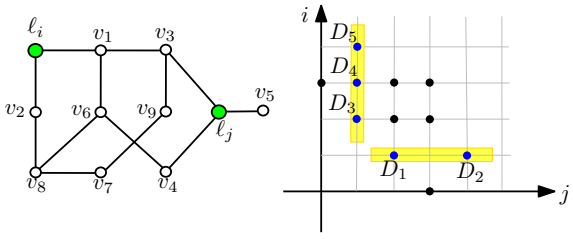


Fig. 2: A graph with two leaders and a plot of distance-to-leaders vectors as points in a plane. Point set $\mathcal{X} = \{D_1, D_2, D_3, D_4, D_5\}$ constitutes a conflict, where $\mathcal{X}_i = \{D_3, D_4, D_5\}$ and $\mathcal{X}_j = \{D_1, D_2\}$.

[2]In general, parts of a partition do not intersect. For the lack of a better term, we are slightly abusing this term in the sense that parts ($\mathcal{X}_i$) intersect at most one element.

It is easy to see that a PMI sequence can not contain all points in a conflict. In fact, we can strictly bound the number of points from a conflict that can be included in a PMI sequence.

Lemma 3.2: Let $\mathcal{X}_1, \mathcal{X}_2, \ldots, \mathcal{X}_m$ be a c-partition of a conflict $\mathcal{X}$ for a given set of points. Then any PMI sequence contains at most $|\mathcal{X}| - \min(|\mathcal{X}_1|, |\mathcal{X}_2|, \ldots, |\mathcal{X}_m|) + 1$ points from $\mathcal{X}$.

Proof: Let $k_j = |\mathcal{X}_j|$ for all $1 \leq j \leq m$ for the partition defined in the statement, then for the sake of contradiction, let's assume that there is a sequence $\mathcal{D}'$ that contains more points from $\mathcal{X}$. Let $D_p \in \mathcal{X}_j$ be a point that appears first in $\mathcal{D}'$. If D_p satisfies PMI property on j^{th} coordinate then the remaining $k_j - 1$ points with the same minimum j^{th} coordinate in $\mathcal{X}_j$ can not be included in $\mathcal{D}'$. So D_p must satisfy PMI property on some j'^{th} coordinate for the following points in the sequence, where $j' \neq j$. But then $\mathcal{D}'$ must miss at least $k_{j'}$ points that have smaller or equal j' coordinate by the definition of conflict, which is a contradiction. Thus, the claim follows. ■

As an example, consider a set of points $\mathcal{X} = \{D_1, D_2, D_3, D_4, D_5\}$ in Figure 2. There are two points with the minimum j^{th} coordinate and three points with the minimum j'^{th} coordinate (where $j' = i$). If D_1 is picked as first point (among this set), we must either drop D_2 or all D_3, D_4, D_5 for future consideration in the PMI sequence. Similarly if D_3 is picked before everyone else, we can not pick either of D_4, D_5, or any of D_1, D_2 for future consideration regardless of the other points. Note that the bound in Lemma 3.2 is tight: if we remove $|\mathcal{X}_j| - 1$ points from the smallest part of a c-partition, all remaining points can satisfy the PMI property on coordinate j unless some of these remaining points are included in any other conflict.

In the following, we use the following notations:

- $Ł^j$ denotes a list of points ordered by the non-decreasing j^{th} coordinate.
- $Ł^j_i$ denotes the i^{th} point in the list $Ł^j$, and
- $Ł^j_{i,k}$ is the (integer) value of k^{th} coordinate of $Ł^j_i$.[3]

Let D be a set of n points in $\mathbb{Z}^m$. We can sort all points with respect to all coordinates beforehand, so our algorithm will get m lists $\{Ł^1, Ł^2, \ldots, Ł^m\}$ of n points each as input. Next, we design an algorithm that is based on dynamic programming to compute the lower bound $\delta(G, V_\ell)$ in polynomial runtime when the number of leaders is fixed. Let $\{c_1, c_2, \ldots, c_m\}$ be a set of non-negative integers and $\mathcal{D}^{[c_1, c_2, \ldots, c_m]}$ be a longest PMI sequence in which the value at j^{th} coordinate of any point is at least c_j. Let $\alpha^{[c_1, c_2, \ldots, c_m]}$ be the length of such a sequence. Our algorithm will memoize on $\alpha^{[c_1, c_2, \ldots, c_m]}$.

In the absence of conflict, Observation 3.1 guarantees that we can start our sequence with any point with the unique minimum value in some fixed coordinate. However, as suggested by Lemma 3.2, in case of a conflict, we cannot include all points to PMI. Thus, we need to include some of the points and exclude others. The longest PMI sequence can be found by computing m subsequences corresponding to m coordinates and taking the maximum. We conclude that $\alpha^{[c_1, c_2, \ldots, c_m]}$ can be obtained by the following recurrence:

$$\alpha^{[c_1, c_2, \ldots, c_m]} = \max_{1 \leq j \leq m} (\alpha^{[c_1, c_2, \ldots, c_j + 1, \ldots, c_m]} + \mathbf{1}_{c_j}), \quad (10)$$

where

$$\mathbf{1}_{c_j} = \begin{cases} 1 & \text{if } \exists\ D_p \text{ s.t. } [D_p]_j = c_j \text{ and } [D_p]'_j \geq c_{j'}, \forall j' \neq j. \\ 0 & \text{otherwise.} \end{cases} \quad (11)$$

We plan to pre-compute and memoize all *required* values of $\alpha^{[c_1, \ldots c_d]}$ in a table. Clearly there are infinitely many possible values for c_j; however, we observe the following:

Observation 3.3: Let $Ł^j_{i,j}$ and $Ł^j_{i,j+1}$ be the j^{th} coordinate values of two consecutive points in $Ł^j$, then

$$\alpha^{[c_1, c_2, \ldots, x, \ldots, c_m]} = \alpha^{[c_1, c_2, \ldots, Ł^j_{i,j+1}, \ldots, c_m]},$$

for all x, such that $Ł^j_{i,j} < x \leq Ł^j_{i,j+1}$.

Observation 3.3 implies that there are at most n different values for each variable c_j, which gives at most n unique values for $\alpha^{[c_1, c_2, \ldots, c_m]}$. Thus, we only keep a table of size n^m for computation and storage of solutions to all sub-problems.

Algorithm 1 PMI - Dynamic Program

```
procedure PMI-DP (Ł^1, Ł^2, ..., Ł^m)
    z_j be number of unique values of j^th coordinate among all points.
    z = max(z_1, z_2, ..., z_m)
    Define a m-dimensional array A with dimensions (z+1) × (z+1) × ... (z+1)
    Let A_{c_1,c_2,...,c_m}, i.e. value of A at index set c_1, c_2, ..., c_m represents α^{[c_1,c_2,...,c_m]} as in (10).
    for k from 1 to m do
        A_{c_1,c_2,...,c_m} ← 0 for c_k = z, c_{k'} ≤ z, k' ≠ k.
    end for
    for j from z − 1 to 0 do
        for k from 1 to m do
            Compute A_{c_1,c_2,...,c_m} for c_k = j, c_{k'} ≤ j, k' ≠ k using (10).
        end for
    end for
    return A_{0,0,...,0}
end procedure
```

We now state and prove the main result of this section:

Theorem 3.4: Given a graph G on n vertices, and m leaders, Algorithm 1 returns a longest PMI sequence of distance-to-leaders vectors in $O(m(n \log n + n^m))$ time.

Proof: The correctness of Algorithm 1 follows from Observation 3.1, Observation 3.3, and Lemma 3.2 so all that remains is to prove the time complexity. Computing sorted lists $Ł^1, Ł^2, \ldots, Ł^m$ takes $O(mn \log n)$ time. Each value of $A_{c_1, c_2, \ldots, c_m}$ can be computed by taking the maximum of m known values previously computed, and saved in multi-dimensional array A. The value of $\mathbf{1}_{c_j}$ can be computed in constant time by checking whether element at the last index of $Ł^j$ has j^{th} coordinate equal to c_j as defined in (11). The multi-dimensional array A contains at most n^m values at the

[3] We recommend to use linked priority queues or similar data structure for these lists so that one could easily delete a point from lists while maintaining respective orders in logarithmic time.

completion each of which takes constant amount of time to compute. Therefore running time of this algorithm is bounded by $O(m(n\log n + n^m))$. ∎

An example illustrating the algorithm is provided in [18].

Remark 3.5: We note that an exact algorithm to compute the longest PMI sequence in $O(m^n)$ was proposed in [10]. Since m is much smaller than n typically, the dynamic programming solution in Algorithm 1 computes the longest PMI sequence in a much lesser $O(m(n\log n + n^m))$ time.

IV. Linearithmic Time Approximation Algorithm for the Distance-based Bound

In this section, we discuss a greedy algorithm that takes linearithmic time to approximate the lower bound $\delta(G, V_\ell)$. The algorithm gives very close approximations in practice, as illustrated numerically in Section VII. We also discuss the approximation guarantees of the algorithm.

The main idea behind the greedy algorithm is to make *locally* optimal choices when faced with the situation in Lemma 3.2, that is, when including a point in PMI results in discarding a subset of points from possible future consideration. In this case, the best thing to do locally is to pick a point that results in the loss of the minimum number of other points. The details are provided in Algorithm 2. An illustration of this algorithm is given in [18].

Algorithm 2 PMI-Greedy Algorithm

procedure PMI-Greedy($Ł^1, Ł^2, \ldots, Ł^m$)
  $\mathcal{D} \leftarrow \emptyset$ ▷ Initially empty sequence
  while $Ł^1 \neq \emptyset$ **do**
    $X_j \leftarrow \{Ł_i^j : Ł_{i,j}^j = L_{1,j}^j\}$ for all j.
    if $\exists j$ such that $|X_j| = 1$ **then** ▷ Unique min.
      $\mathcal{D} \leftarrow [\mathcal{D}\ \ X_j]$
      Remove X_j from all lists.
    else
      Let $j' \leftarrow \arg\min_j |X_j|$ ▷ Get smallest X_j
      $\mathcal{D} \leftarrow [\mathcal{D}\ \ Ł_1^{j'}]$
      Remove all points in $X_{j'}$ from all lists.
    end if
  end while
  return $\mathcal{D}$
end procedure

Proposition 4.1: Algorithm 2 computes a PMI sequence in $O(mn\log n)$ time. The length of the PMI sequence returned by the algorithm is a $\min(m, \frac{n}{m})$-approximation to the optimal length, where m is the number of leaders and n is the total number of nodes. Further, the approximation ratio of PMI lengths is $\log n$ if $m \leq \log n$ or $m \geq \frac{n}{\log n}$.

Proof: Regarding the time complexity, computing sorted lists $Ł^1, Ł^2, \ldots, Ł^m$ takes $O(m \times n\log n)$ time. Once we have m sorted lists, we can keep the indices and count of points with the minimum coordinate value in $(m+1)$ Min-Priority queues (m queues to maintain lists $Ł^1, Ł^2, \ldots, Ł^m$ and one queue for $X_1, X_2, \ldots, X_m$). Cost of one *update*, or *delete* operation is $O(\log n)$ in a Priority queue. Since we will *update* and/or *delete* at most n points from m queues. In total we will perform at most $m \times n$ deletions and $m \times n$ updates, thus, the overall time complexity is $O(mn\log n)$.

Regarding the m-approximation ratio, we observe that there are at least $\frac{n}{m}$ different values in at least one coordinate. Otherwise, we may assume that we have at most $\frac{n}{m} - 1$ unique values in each coordinate. This would imply that there are at most $(\frac{n}{m} - 1) \times m$ distinct points by the pigeonhole principle, which contradicts that we have n unique points. As Algorithm 2 picks all distinct values in any coordinate, the returned PMI sequence has a size of at least $\frac{n}{m}$. Note that there is at least one unique minimum point (corresponding to the leader itself) in each of m directions, so the algorithm is $(\frac{n}{m})$-approximation as well.

It is evident that when m is at most $\log n$, there are at least $\frac{n}{\log n}$ different values in at least one coordinate by the same argument. To see why $\log n$-approximation ratio holds when m is large, note that there is at least one unique minimum point (corresponding to the leader itself) in each of m directions so when $m \geq \frac{n}{\log n}$, the algorithm will include all of those unique points in the returned PMI sequence. Thus, the approximation ratio follows. ∎

If there exists a PMI sequence of length n, then the network is strong structurally controllable with a given set of leaders. The greedy algorithm presented above always returns a PMI sequence of length n if there exists one.

Lemma 4.2: If there exists a PMI sequence of length n, then Algorithm 2 always returns an optimal PMI.

Proof: We observe that if there exists a PMI sequence of length n, then by Lemma 3.2, we can not have a conflict as defined in Section III. In the absence of any conflict, we can always find a unique minimum point along some coordinate. Consequently, in Algorithm 2, statements in **else** will never be executed and algorithm will return a PMI sequence of length n. ∎

While in many cases, Algorithm 2 achieves a solution close to optimum, we observe that examples can be constructed for which a greedy solution may not be globally optimal [18]. Moreover, the greedy algorithm approach is significantly different from the one in Algorithm 1, even if greedy returns an optimal solution.

V. Application: Leader Selection for Strong Structural Controllability

In this section, we briefly discuss an application of computing the distance-based bound in approximately solving a leader selection problem for strong structural controllability, which is intractable to solve exactly [2], [8], [11]. The problem of finding the minimum number of leaders to make a network strong structurally controllable is known to be NP-complete [2], [11]. Here, we consider the problem of finding a set V_ℓ of m leaders that maximizes the dimension of SSCS, i.e.,

$$\underset{V_\ell \subseteq V}{\text{maximize}} \quad \gamma(G, V_\ell); \quad \text{subject to} \quad |V_\ell| = m. \tag{12}$$

In light of (9), the distance-based bound $\delta(G, V_\ell)$ can be used to obtain an approximate solution to such a leader selection problem by solving

$$\underset{V_\ell \subseteq V}{\text{maximize}} \quad \delta(G, V_\ell); \quad \text{subject to} \quad |V_\ell| = m. \tag{13}$$

Any solution to the problem in (13), V_ℓ^*, ensures that the resulting dimension of SSCS is at least $\delta(G, V_\ell^*)$. While the problem in (13) is still hard to solve due to its combinatorial nature, an approximate solution can be obtained by utilizing an algorithm for computing $\delta(G, V_\ell)$. We present a simple greedy heuristic for leader selection for strong structural controllablity using PMI sequences of distance-to-leaders vectors in a graph. Given a network G and the number of leaders m as inputs, the main idea is to iteratively select leaders that maximally increase the length of resulting PMI sequences. The outline of the heuristic is in Algorithm 3.

Algorithm 3 Greedy Leader Selection Algorithm

```
procedure LEADER-SELECT(G, m)
    V_ℓ ← ∅, V' ← V
    for i ← 1 to m do
        for each v ∈ V' do
            Compute PMI sequence with V_ℓ ∪ {v} leaders.
        end for
        Choose v' ∈ V' that gives a PMI sequence of maximum length with V_ℓ ∪ {v'} leaders.
        V_ℓ ← V_ℓ ∪ {v'}.
        V' ← V' \ {v'}.
    end for
end procedure
```

The time complexity of Algorithm 3 depends on the complexity of computing a PMI sequence with a given set of leaders. Using the algorithm in [10] to compute PMI sequences, the complexity of Algorithm 3 is $O(n \times m^n)$, which means the algorithm is practically infeasible even for $m = 2$. Using Algorithm 1 (dynamic programming) to compute PMI sequences, the time complexity of Algorithm 3 is $O(n^2 \log n + m \times n^{m+1})$, which is a significant improvement. The first term in this expression is the cost of sorting the distance lists and the second term is the cost of computing n PMI sequences when the leader set includes m nodes (the iteration when $i = m$) as this dominates the cost of all previous iterations. However, leader selection in Algorithm 3 can be achieved in $O(m^2 n^2 \log n)$ time if we use Algorithm 2 to compute PMI sequences. Section VII-C provides numerical evaluation of the leader selection algorithm (Algorithm 3) that uses exact/greedy algorithm to compute $\delta(G, V_\ell)$ to approximately solve (13).

VI. Bounds in Paths and Cycles

In this section, we explore connections between graph-theoretic properties and the length of the longest PMI in path and cycle graphs. As a result, we show interesting topological bounds on the dimension of SSCS in such graphs with a given set of leader nodes. We note that our results differ from previous works in this direction in two aspects: first, we specifically study the strong structural controllability of such graphs; second, instead of focusing on complete controllability, we provide tight bounds on the dimension of SSCS even when the graph is not strong structurally controllable with a given set of leaders (e.g., [19], [20]).

Recall that a node with a single neighbor is called *leaf*. Moreover, given $G = (V, E)$ and $V' \subset V$, then the subgraph of G *induced* on V' is the graph whose vertex set is V', and the edge set consists of all of the edges in E that have both endpoints in V'. We start with the following obvious fact.

Fact 6.1: A path graph in which a leaf is a leader has a PMI sequence of length n.

Theorem 6.2: Let G be a path graph on n nodes, let V_ℓ be a set of $m \leq n$ leader nodes, and let G^{-V_ℓ} denote the subgraph of G induced on vertices $V \setminus V_\ell$. Then the following holds:

(i) If the number of connected components in G^{-V_ℓ} is less than $m+1$, then the longest PMI sequence induced by V_ℓ has length n.

(ii) If the number of connected components in G^{-V_ℓ} is $m+1$, then V_ℓ induces a PMI sequence of length $n - a$, where a is the size of the smallest connected component in G^{-V_ℓ}.

Proof: (i) Removal of a node from a path results in at most two connected components. Hence, G^{-V_ℓ} has at most $m+1$ such components. If the number of components is less than $m+1$, either one of the leader nodes is a leaf, or at least two leaders are adjacent. If a leaf x is chosen as a leader, then by Fact 6.1, we can get a PMI sequence of length n. Assuming none of the leaders is a leaf node, let v_i and v_{i+1} be adjacent leader nodes; further assume that $i < n/2$ without loss of generality. We will construct a PMI sequence of length n based on these two leaders as follows.

$$\left[\begin{bmatrix} 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 2 \end{bmatrix}, \begin{bmatrix} 2 \\ 1 \end{bmatrix}, \cdots, \begin{bmatrix} i-1 \\ i \end{bmatrix}, \begin{bmatrix} i \\ i-1 \end{bmatrix}, \begin{bmatrix} i+1 \\ i \end{bmatrix}, \begin{bmatrix} i+2 \\ i+1 \end{bmatrix}, \cdots, \begin{bmatrix} n-i \\ n-i-1 \end{bmatrix} \right]$$

(ii) If the smallest connected component X contains either of the leaf nodes, then G^{-X} has a leaf leader node and thus has a PMI sequence of length $n - |X|$ by Fact 6.1. If X doesn't contain leaf nodes, then there exist two leader nodes v_i, v_j are adjacent to some nodes in X. Also, assume that v_i is not farther away from a leaf node than v_j is. Then, the following sequence of distance-to-leaders vectors defines a PMI sequence of claimed length.

$$\left[\begin{bmatrix} 0 \\ a+1 \end{bmatrix}, \begin{bmatrix} a+1 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ a+2 \end{bmatrix}, \begin{bmatrix} a+2 \\ 1 \end{bmatrix}, \cdots, \begin{bmatrix} i-1 \\ a+i \end{bmatrix}, \begin{bmatrix} a+i \\ i-1 \end{bmatrix}, \begin{bmatrix} a+i+1 \\ i \end{bmatrix}, \begin{bmatrix} a+i+2 \\ i+1 \end{bmatrix}, \cdots, \begin{bmatrix} n-i \\ n-i-a-1 \end{bmatrix} \right]$$

■

Theorem 6.3: Let G be a cycle on n nodes, let V_ℓ be a set of $2 \leq m \leq n$ leader nodes, and let G^{-V_ℓ} denote the subgraph of G induced on vertices $V \setminus V_\ell$. Then, the following holds:

(i) If the number of connected components in G^{-V_ℓ} is less than m, then the longest PMI sequence induced by V_ℓ has length n.

(ii) If the number of connected components in G^{-V_ℓ} is exactly m, then V_ℓ induces a PMI sequence of length $n - a$, where a is the size of the smallest connected component in G^{-V_ℓ}.

Proof: (i) Removing a single node from a cycle does not affect the number of connected components. However, the removal of every subsequent node will result in at most one extra

component. Thus, the total number of connected components is at most m after the removal of m nodes. If the number of components is less than m in G^{-V_ℓ}, then at least two nodes in V_ℓ are neighbors in G. Let v_1 and v_2 be an arbitrary adjacent pair in V_ℓ. We will construct a PMI sequence of length n based on these two leaders. Consider the nodes in G with the following distance-to-leaders vectors:

$$\left[\begin{bmatrix} 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 2 \end{bmatrix}, \begin{bmatrix} 2 \\ 1 \end{bmatrix}, \cdots, \begin{bmatrix} \frac{n}{2}-1 \\ \frac{n}{2} \end{bmatrix}, \begin{bmatrix} \frac{n}{2} \\ \frac{n}{2}-1 \end{bmatrix} \right]$$

when n is even, and

$$\left[\begin{bmatrix} 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 2 \end{bmatrix}, \begin{bmatrix} 2 \\ 1 \end{bmatrix}, \cdots, \begin{bmatrix} \lfloor \frac{n}{2} \rfloor \\ \lfloor \frac{n}{2} \rfloor \end{bmatrix} \right]$$

when n is odd. This defines a PMI sequence of length n.

(ii) An argument identical to proof of Theorem 6.2(ii) can be used here to prove (ii) as well. ∎

Theorems 6.2 and 6.3 imply graph-theoretic bounds on the dimension of SSCS for path and cycle graphs. A path (cycle) graph is strong structurally controllable with V_ℓ leaders if G^{-V_ℓ} has $m+1$ components (m components in a cycle). Another direct implication of the above results is as follows.

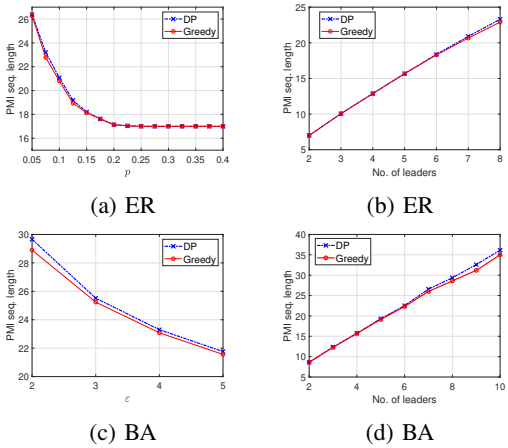
Corollary 6.4: Let G be a path or cycle graph and let V_ℓ be a set of leaders, then the dimension of SSCS is at least $n-a$, where a is the smallest distance between any two leader nodes.

VII. Numerical Evaluation

In this section, we numerically evaluate our results on Erdös-Rényi (ER) and Barabási-Albert (BA) graphs. In ER graphs, any two nodes are adjacent with a probability p. BA graphs are obtained by adding nodes to an existing graph one at a time. Each new node is connected to ε existing nodes with probabilities proportional to the degrees of those nodes.

A. Comparison of Algorithms

First, we compare the performance of the exact dynamic programming algorithm (Algorithm 1) and the approximate greedy algorithm (Algorithm 2) for computing the maximum-length PMI sequences. For simulations, we consider graphs with $n = 200$ nodes. For ER graphs, we first plot the length of PMI sequences computed by using Algorithms 1 and 2 as a function of p while fixing the number of leaders (selected randomly) to be eight (Figure 3(a)). Second, we fix $p = 0.075$, and plot the length of PMI sequences as a function of the number of leaders selected randomly (Figure 3(b)). We repeat similar plots for BA graphs in Figures 3(c) and 3(d). We fix the number of leaders to be eight in Figure 3(c) and set $\varepsilon = 2$ in Figure 3(d). Each point in the plots in Figure 3 corresponds to the average of 50 randomly generated instances. From the plots, it is clear that the greedy algorithm, which is much faster than the DP algorithm, performs almost as well as the DP algorithm in terms of the length of PMI sequences returned. From the theoretical analysis and the numerical evaluation, we conclude that the Algorithm 1 should be used to compute the exact solution when the number of nodes is at most few thousand, and the number of leaders is small. Moreover, whenever the number of nodes is in millions, or the number of leaders is large, Algorithm 2, which computes an approximate solution, should be preferred.



(a) ER (b) ER

(c) BA (d) BA

Fig. 3: Comparison of Algo. 1 (dynamic programming) and Algo. 2 (greedy) for computing the distance-based bound.

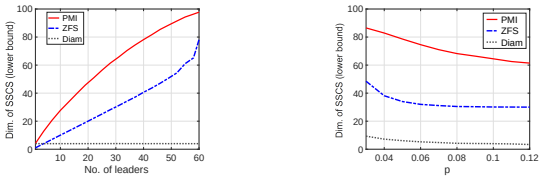
B. Comparison of Bounds

Next, we numerically compare our distance-based bound with another well known bound on the dimension of SSCS based on the notion of zero-forcing sets (ZFS) [5], [14]. First, we explain the notion of ZFS. Given a graph $G = (V, E)$ in which each node is colored either *white* or *black*, we repeatedly apply the following coloring rule: *If $v \in V$ is colored black and has one white neighbor u, then the color of u is changed to black.* Now, given an initial set of black nodes (called *input set*) in G, *derived set* $V' \subseteq V$ is the set of all black nodes obtained after repeated application of the coloring rule until no color changes are possible. It is easy to see that for a given input set, the resulting derived set is unique. The input set is called a ZFS if the corresponding derived set contains all nodes in V. It is shown in [5], [14] that for a given set of leader nodes as input set, the size of the corresponding derived set is a lower bound on the dimension of SSCS.

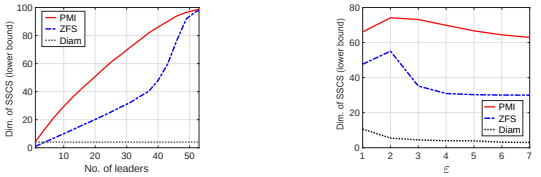
In our simulations in Figure 4, for both ER and BA models, we consider graphs with $n = 100$ nodes. In Figure 4(a), we plot these bounds for ER graphs as a function of the number of leaders, which are selected randomly, while fixing $p = 0.1$. Next, we fix the number of leaders to be 30 in Figure 4(b) and plot bounds as a function of p. As previously, each point in the plots is an average of 50 randomly generated instances. Similar results are obtained in the case of BA graphs, where we fix $\varepsilon = 4$ in Figure 4(c), and select the number of leaders to be 30 in Figure 4(d). In all the plots, lengths of PMI sequences are greater than the derived sets indicating that the distance-based bound on the dimension of SSCS performs better than the one based on the derived sets.

C. Leader Selection

We implement Algorithm 3 to illustrate the application of proposed algorithms to the leader selection problem given in (13). Again, the networks were generated for both ER and BA models with $n = 60$ nodes. In order to compute the length of the longest PMI sequence, we use the bounds returned by the dynamic programming solution and the greedy algorithm. We compare the respective bounds on the dimension



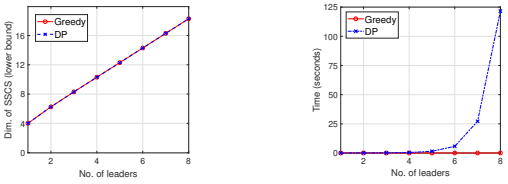
(a) ER ($p = 0.1$) (b) ER ($|V_\ell| = 30$)



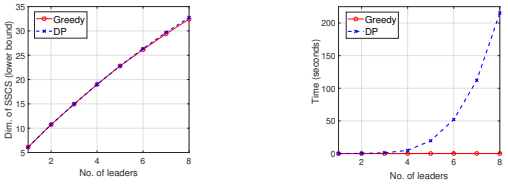
(c) BA ($\varepsilon = 4$) (d) BA ($|V_\ell| = 30$)

Fig. 4: Comparison of the distance-based and ZFS-based lower bounds on the dimension of SSCS in ER and BA graphs. The total number of nodes in all graphs is 100. The diameters of graphs (denoted by Diam) are also plotted.

of SSCS and the computation times of the two algorithms. The results of our experiments are shown in Figure 5. The resulting bounds are plotted against the number of leaders in Figures 5(a) and 5(c), and the running times are plotted in Figures 5(b) and 5(d). For both graph families, the bounds computed by the two algorithms are almost identical, but the greedy algorithm has the advantage of superior runtime that becomes more pronounced as the number of leaders increases.



(a) ER ($p = 0.3$) (b) ER ($p = 0.3$)



(c) BA ($\varepsilon = 2$) (d) BA ($\varepsilon = 2$)

Fig. 5: Comparison of the leader selection algorithm with the greedy computation and the exact computation (dynamic programming (DP)) of the distance-based bound. The total number of nodes in all graphs is 60.

VIII. Conclusion

We studied the computational aspects of a lower bound on the dimension of SSCS in networks with Laplacian dynamics. The bound is based on a sequence of distance-to-leaders vectors and has several applications. We proposed an algorithm that runs in $O(n^m)$ time (compared to $O(m^n)$ runtime of the algorithm in [10]) to compute the bound. We also presented a linearithmic approximation algorithm to compute the bound, which provided near-optimal solutions in practice. Further, we explored connections between the graph-theoretic properties and the distance-based bound in path and cycle graphs using the results. We plan to use these results to explore further the trade-offs between controllability and other desirable network properties, such as robustness and resilience to perturbations.

References

[1] M. Shabbir, W. Abbas, and Y. Yazıcıoğlu, "On the computation of a lower bound on strong structural controllability in networks," in *58th IEEE Conference on Decision and Control*, 2019, pp. 5468–5473.

[2] 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.

[3] J. C. Jarczyk, F. Svaricek, and B. Alt, "Strong structural controllability of linear systems revisited," in *50th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC)*, 2011.

[4] A. Weber, G. Reissig, and F. Svaricek, "A linear time algorithm to verify strong structural controllability," in *53rd IEEE Conference on Decision and Control (CDC)*, 2014, pp. 5574–5580.

[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] M. Fazel, H. Hindi, and S. Boyd, "Rank minimization and applications in system theory," in *Proceedings of the American control conference (ACC)*, vol. 4, 2004, pp. 3273–3278.

[7] S. M. Fallat and L. Hogben, "The minimum rank of symmetric matrices described by a graph: a survey," *Linear Algebra and its Applications*, vol. 426, no. 2-3, pp. 558–582, 2007.

[8] A. Bhangale and S. Kopparty, "The complexity of computing the minimum rank of a sign pattern matrix," *arXiv:1503.04486*, 2015.

[9] 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, 2014.

[10] 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.

[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] 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.

[13] 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, pp. 566–571.

[14] 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.

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

[16] W. Abbas, M. Shabbir, A. Y. Yazıcıoğlu, and A. Akber, "Trade-off between controllability and robustness in diffusively coupled networks," *IEEE Transactions on Control of Network Systems*, 2020.

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

[18] M. Shabbir, W. Abbas, Y. Yazıcıoğlu, and X. Koutsoukos, "Computation of the distance-based bound on strong structural controllability in networks," *arXiv:1909.03565*, 2021.

[19] G. Parlangeli and G. Notarstefano, "On the reachability and observability of path and cycle graphs," *IEEE Transactions on Automatic Control*, vol. 57, no. 3, pp. 743–748, 2012.

[20] X. Liu and Z. Ji, "Controllability of multiagent systems based on path and cycle graphs," *International Journal of Robust and Nonlinear Control*, vol. 28, no. 1, pp. 296–309, 2018.