Fifty-Fifth Annual Allerton Conference
Allerton House, UIUC, Illinois, USA
October 3-6, 2017

Distributed Leader Selection in Switching Networks of High-order Integrators

Anastasios Tsiamis ⋆ Sérgio Pequito ⋆ George J. Pappas ⋆

***Abstract*— We consider the problem of leader selection in a team of interconnected agents, modeled as higher-order integrators. The leader selection problem aims to determine the minimum number of agents, i.e., the leaders, that are required to receive an external input to ensure a control-theoretical property. The remaining agents, i.e., the followers, update their states based only on local information gathered from the other agents they interact with. In this paper, given the communication topology, our goal is to design distributed protocols that determine the minimum number of leaders and the communication protocol to ensure a desirable property, without any knowledge about the global structure of the network. The main contributions consist of determining the solution to this problem when the controllability and stabilizability are sought. In addition, we extend these results to the case where the communication topology switches over time. Finally, we illustrate the main results by providing a simulation example.**

I. INTRODUCTION

Last decade has unleashed the potential of multi-agent systems in the context of engineering applications. These agents are equipped with local communication capabilities that enable them to interface with each other. In fact, with simple local rules, these can solve complex problems that pertain to distributed control and coordination, flocking, rendezvous and formation [1], [2].

The local rules, or *communication protocols*, often aim to establish control theoretical properties such as *controllability* [3], or *consensus* [4], [5]. These communication protocols have been explored in two distinct scenarios [2]: (i) the *leader-follower*, where the leaders are the agents that receive an external signal, whereas the followers are the remaining agents that only consider local information, i.e., coming from other agents they interact with; and (ii) the *leaderless*, where no exogenous information to the agents is considered. Despite the vast literature on the topic, this has mainly focused on centralized analysis and design of communication protocols, which are distributed in nature. Therefore, if we are ever going to release the full potential of the multi-agent networks, the following three largely unsolved questions need to be addressed:

(i) How does an agent decide if it should be a leader?
(ii) What is the communication protocol it should use?
(iii) Which control strategy should a leader adopt?

This work was supported in part by the TerraSwarm Research Center, one of six centers supported by the STARnet phase of the Focus Center Research Program (FCRP) a Semiconductor Research Corporation program sponsored by MARCO and DARPA. This work was supported in part by the NSF under grants CNS-1302222 and IIS-1447470.

⋆Department of Electrical and Systems Engineering, School of Engineering and Applied Science, University of Pennsylvania

Leader-selection problems address the first question, see, for instance, [6], [7], [8], [9], [10], [11] and references therein. Most of these are solved in a centralized manner, or, when distributively, under restrictive assumptions such as the communication between agents forms a connected graph and/or is known by all agents in the network [12], [13]. The second question is often studied using eigenvalues and eigenvectors of the system matrix, which requires equally restrictive premises, see for instance [3], [14]. Finally, to devise a control law that steers the agents into a target state one often requires knowledge of the overall dynamics of the agents, or the structure of the network [15], which is not achievable in completely distributed scenarios.

In fact, it is easy to agree that in distributed scenarios the three questions need to be answered only considering locally exchanged information between the agents. Notwithstanding, only recently [9] a distributed solution to the above three questions was attained in the case where the agents possess a scalar state, with discrete-time dynamics and the communication topology is fixed over time. To the best of the authors' knowledge, this is the first paper that deals with the *distributed* leader selection problem in the context of multi-dimensional state dynamics and switching network topology, without any global knowledge of the system structure.

The main contributions are threefold: (i) we determine the minimum number of leaders such that the overall system is controllable; (ii) we study the stabilizability-to-input problem and provide convergence guarantees; (iii) we consider the scenario where the communication topology is changing over time.

The rest of the this paper is organized as follows. In Section II, we formally state the problems addressed in this paper. Section III provides some preliminary concepts and results. In Section IV, we present the main technical results, then, in Section V, we discuss the limitations of the proposed solution. Subsequently, we provide a simulation example in Section VI. Conclusions and discussion for further research are presented in Section VII.

II. PROBLEM STATEMENT

Consider n interconnected agents, which are modeled as m-th order integrators with the following linear dynamics:

$$\dot{x}_i(t) = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ & & \vdots & & \\ 0 & 0 & 0 & \cdots & 1 \\ 0 & 0 & 0 & \cdots & 0 \end{bmatrix} x_i(t) + \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix} u_i(t), \quad (1)$$

978-1-5386-3266-6/17/$31.00 ©2017 IEEE

where $x_i(t) \in \mathbb{R}^m$ is the state of each agent $i \in \mathcal{V} = \{1, \ldots, n\}$, and $u_i(t) \in \mathbb{R}$ its input. We note that under similarity transformations, this integrator model can represent any controllable linear system with one input. Thus, the results of this paper also hold for more general controllable linear systems. The interconnection of the agents is represented by a communication graph $\mathcal{D} = (\mathcal{V}, \mathcal{E})$, where a directed edge $(i,j) \in \mathcal{E}$ indicates that agent i is able to transmit information to agent j. We denote the set of all agents that can directly transmit to agent i by $\mathcal{N}_i^- = \{j \in \mathcal{V} : (j,i) \in \mathcal{E}\}$, which we refer to as the in-neighbors of agent i. Each agent knows its own state, which implies that $(i,i) \in \mathcal{E}$, and, consequently, $i \in \mathcal{N}_i^-$. We analogously define the set of out-neighbors $\mathcal{N}_i^+ = \{j \in \mathcal{V} : (i,j) \in \mathcal{E}\}$.

Each agent i transmits to its out-neighbors a scalar z_i given by a linear combination of its state, i.e.,

$$z_i(t) = c^\intercal x_i(t), \tag{2}$$

for some constant $c \in \mathbb{R}^m$, which is known to all agents offline. Meanwhile, each agent i receives the incoming z_j from its in-neighbors $j \in \mathcal{N}_i^-$ and applies the following control on its state:

$$u_i(t) = -K_i x_i(t) + \sum_{j \in \mathcal{N}_i^-} w_{ij} z_j(t) + b_i v_i(t), \tag{3}$$

where $K_i \in \mathbb{R}^{1\times m}$ is a gain row vector acting on its own state. The constants w_{ij} are nonnegative weights that the agents design in order to integrate their neighbors' information. We refer to the collection of all weights in matrix form $W = [w_{ij}]$, as *communication protocol*. A weight w_{ij} can be positive only if agent j is an in-neighbor of agent i; in other words, if $(j,i) \notin \mathcal{E}$ then $w_{ij} = 0$. Subsequently, we consider the *realization graph*

$$\hat{\mathcal{D}} = \left(\mathcal{V}, \hat{\mathcal{E}}\right), \ \hat{\mathcal{E}} = \{(i,j) \in \mathcal{E} : \ w_{ji} > 0 \text{ or } i = j\}, \tag{4}$$

which is induced by the strictly positive weights on the edges of the communication graph with self-loops included. We define the *design protocol* to be $P = P(\mathcal{D}) = \{W, c, K_i, \ i = 1, \ldots, n\}$.

The signal $v_i \in \mathbb{R}$ models additional actuation capabilities, or equivalently an exogenous signal to the agents, that can be used to drive the network to a specific goal. Only a subset of the agents, the leaders, use this signal. We denote this set of leaders by $\mathcal{J} \subset \mathcal{V}$. If agent i is a leader, i.e. $i \in \mathcal{J}$, then $b_i = 1$ and, as a consequence, the input v_i is used. All the other agents $i \in \mathcal{V} \setminus \mathcal{J}$ are called followers and they do not use the additional input, i.e. $b_i = 0$. In general, we assume that all agents can potentially have access to v_i, but it costs resources to do so. For instance, a robot should use a long-range receptor or GPS to access the input. Hence, this motivates our pursuit of the minimum number of leaders.

With this choice of inputs, the whole network is a linear system that depends on the design protocol P as well as the set of leaders $\mathcal{J}$. Thus, we can rewrite the overall dynamics of the network in matrix form as follows:

$$\dot{x}(t) = A(P)\, x(t) + B(\mathcal{J})\, v(t), \tag{5}$$

where $x(t) = \begin{bmatrix} x_1^\intercal(t) & \cdots & x_n^\intercal(t) \end{bmatrix}^\intercal \in \mathbb{R}^{nm}$ is the overall state vector and $v(t) = \begin{bmatrix} v_1 & \cdots & v_n \end{bmatrix}^\intercal \in \mathbb{R}^n$ is the vector of external inputs. We omit the analytical expressions of $A(P)$, $B(\mathcal{J})$, which are not needed in the subsequent analysis.

The first problem addressed in this paper is the *minimization of the number of leaders* while ensuring controllablity of system (5), in a *fully distributed manner*. Formally, given $\mathcal{D}$, we want to determine, in a distributed fashion, the protocol P^* and the set of leaders $\mathcal{J}^*$ such that:

$$(P^*, \mathcal{J}^*) = \arg\min_{P,\ \mathcal{J}} \quad |\mathcal{J}| \qquad \text{s.t.} \quad (A(P), B(\mathcal{J})) \text{ is controllable}, \tag{6}$$

where the pair $(A(P), B(\mathcal{J}))$ describes system (5).

In practice, we just require the agents of the network to be stabilized to a desired point x_d (consensus); even if controllability is attained, it is not clear how the leaders should select their inputs to perform more general tasks without knowing the global structure/dynamics of the network. Therefore, the second problem we address in this paper is the stabilizability-to-input problem. Our goal is to distributedly select the minimum number of leaders such that all agents can be stabilized to x_d.

Another challenge in the leader selection problem is to address the varying communication topology, possibly due to transmission failures or changes in the communication capabilities of the different agents. We capture this effect with a switching communication graph $\mathcal{D}_{\sigma(t)}$, where $\sigma : \mathbb{R}^+ \to \mathbb{N}$ is the switching signal. The overall dynamics are modeled as follows:

$$\dot{x}(t) = A\left(P_{\sigma(t)}\right) x(t) + B\left(\mathcal{J}_{\sigma(t)}\right) v(t). \tag{7}$$

Therefore, the third problem we address is that of determining strategies that allow the agents to redefine the design protocol to accommodate the communication topology changes, while ensuring that either controllability or stabilizability-to-input hold.

Throughout the paper, we assume that the agents have a synchronized clock. To solve conflicts, we assume that the agents are enumerated, with the number i of the i-th agent being unique. We assume that at most one edge of the communication graph can change each time, and the design protocol is determined between two consecutive changes in the communication topology.

III. PRELIMINARIES AND TERMINOLOGY

First, we introduce the following graph theoretic notions. A *directed path* is a sequence of directed edges where the end-vertex of one edge is the start-vertex of the other. If $(j,i) \in \mathcal{E}$ then j is called a *parent* of i. In addition, j is an *ancestor* of i and i is a *descendant* of j, if there exists a path from j to i. The *distance* from j to i is given by the number of edges in a shortest path from j to i. The distance $l_i^{\mathcal{J}}$ (or l_i when clear from the context) from a set $\mathcal{J}$ to vertex i is the minimum possible distance from $j \in \mathcal{J}$ to i. A *cycle* is a path with the same the end and start-vertices.

A digraph $\mathcal{D}$ is said to be *strongly connected* if there exists a directed path between any two vertices. A subgraph $\mathcal{D}_S$ of $\mathcal{D}$ is a digraph whose vertex and edge sets are subsets of those of $\mathcal{D}$. A *strongly connected component* (SCC) is a maximal subgraph (i.e., there is no other subgraph containing it and satisfying a given property) $\mathcal{D}_S = (\mathcal{V}_S, \mathcal{E}_S)$ of $\mathcal{D}$ such that for every $u, v \in \mathcal{V}_S$ there exists a path from u to v and from v to u. We can create a *directed acyclic graph* (DAG) by visualizing each SCC as a virtual vertex, in which a directed edge between two virtual vertices (SCCs) exists *if and only if* there exists a directed edge between the vertices from the corresponding SCCs in the original digraph $\mathcal{D} = (\mathcal{V}, \mathcal{E})$. A *directed tree* $\mathcal{T} = (\mathcal{V}, \mathcal{E})$ is a directed acyclic graph that is *rooted* in a vertex without incoming edges on it, and where there exists exactly one incoming edge in each of the remaining vertices. A *directed spanning forest* of a digraph $\mathcal{D} = (\mathcal{V}, \mathcal{E})$ is a disjoint union of directed trees $\mathcal{T}_i = (\mathcal{V}_i, \mathcal{E}_i)$, with $i = 1, \ldots, n$, for some $n \in \mathbb{N}$, such that $\cup_{i=1,\ldots,n} \mathcal{V}_i = \mathcal{V}$. In the case $n = 1$, the directed spanning forest is called *directed spanning tree*. At last, the SCCs in the DAG may be further categorized as follows.

Definition 1 ([16]): An SCC is said to be linked if it has at least one incoming (respectively, outgoing) edge from (respectively, to) another SCC. In particular, an SCC is *non-top linked* (n-SCC) if it has no incoming edge to its vertices from the vertices in another SCC. $\diamond$

Second, we review some results from structural systems theory [17] and the notion of structural controllability.

Definition 2 ([17]): (Structural controllability) Let $(\bar{A}, \bar{B})$ denote the sparsity (or structural pattern) of the pair $(A, B) \in \mathbb{R}^{n\times n} \times \mathbb{R}^{n\times p}$ describing a linear time-invariant system, with $\bar{A} \in \{0, \star\}^{n\times n}$ and $\bar{B} \in \{0, \star\}^{n\times p}$, where 0 corresponds to zero entries and $\star$ represents an arbitrary parameter. The pair (A, B), or equivalently $(\bar{A}, \bar{B})$, is said to be structurally controllable if there exists a controllable pair (A, B), with zero entries in the same entries as $(\bar{A}, \bar{B})$. $\diamond$

In general, a stronger characterization of structural controllability holds. For a structurally controllable pair $(\bar{A}, \bar{B})$, almost all numerical realizations (A, B) are controllable [18].

Given $(\bar{A}(P), \bar{B}(\mathcal{J}))$, we can define the following digraphs: (i) the *state graph* of (5), denoted by $G(\bar{A}) = (X, E_X)$, where the state variables of the system are the labels of the vertices of the graph described by $X = \{x_{ik},\, i \in \{1, \ldots, n\},\, k \in \{1, \ldots, m\}\}$, and the edges between the state vertices are given by $E_X = \{(x_{ik}, x_{jl}) :\ \bar{A}_{m(j-1)+l, m(i-1)+k}(P) \neq 0\}$; (ii) The *system graph*, denoted by $G(\bar{A}, \bar{B}) = (X \cup V, E_X \cup E_V)$, which is induced by the pair $(\bar{A}(P), \bar{B}(\mathcal{J}))$, where we add vertices $V = \{v_1, \ldots, v_m\}$ to the state graph corresponding to the inputs, and the edge set $E_V = \{(v_i, x_{jl}) :\ \bar{B}_{m(j-1)+l,i}(\mathcal{J}) \neq 0\}$.

To find the conditions under which system (5) is structurally controllable, we will need the following result.

Corollary 1 ([16]): Let $\mathcal{D}(\bar{A})$ be a state digraph spanned by a disjoint union of cycles. Then, the pair $(\bar{A}, \bar{B})$ is structurally controllable if and only if every n-SCC of $\mathcal{D}(\bar{A})$ has an incoming edge from an input vertex in $\mathcal{D}(\bar{A}, \bar{B})$. $\diamond$

IV. MAIN RESULTS

In this section, we present the solution to the problems stated in Section II. For the fixed topology case, the agents distributedly select the realization graph to be a directed spanning forest of the communication graph, with the disjoint trees having the leaders as roots–see Section IV-A. In particular, they keep one tree (and thus, one leader) for each n-SCC of the communication graph. In Section IV-B, we prove that this selection ensures controllability with the minimum number of leaders, while in Section IV-C, we show that this is also true for the stabilizability-to-input problem. Then, in Section IV-D, we introduce reconfiguration strategies that enable the maintenance of the directed spanning tree formation and, thus, the maintenance of both controllability and stabilizability-to-input when the communication topology changes over time.

A. Distributed protocols

In this subsection, we present the distributed process, which leads to the selection of the directed spanning forest of the communication graph. The material of this subsection is adapted from [9]. We break the process in two separate algorithms. First, the agents select one leader per each n-SCC of the communication graph (see Algorithm 1). Second, given the selected leaders, the remaining agents select to follow only the parent closest to a leader, thus, forming a spanning forest (see Algorithm 2).

More specifically, in Algorithm 1, the agents verify in a *distributed way* if they belong to an n-SCC. Next, if they actually belong to an n-SCC, they decide which one agent will become the leader. At first, each agent i learns the list of its ancestors $\mathcal{L}_i$. Next, a min-consensus protocol is run, where the initial state of each agent is $N_i = |\mathcal{L}_i|$. From [9], we know that the min-consensus problem with $y_i[0] = N_i$ is achieved only in the n-SCCs of the communication graph. Ergo, the final value of the min-consensus remains N_i if and only if agent i lies in a n-SCC. Meanwhile, if an agent lies in an n-SCC, it compares its unique id with those in the list of its ancestors. If it has the smallest id, it becomes the leader of the n-SCC.

ALGORITHM 1: Selection of leaders $\mathcal{J}$

1. Initialize $\mathcal{L}_i^0 = \{i\}$;
For $k = 1, \ldots, n$
2. Receive new lists from the in-neighbors $\mathcal{L}_j^{k-1}$;
3. Update the list: $\mathcal{L}_i^k = \bigcup_{j\in\mathcal{N}_i^-} \mathcal{L}_j^{k-1}$;
4. Transmit $\mathcal{L}_i^k$;
endFor
5. Set $\mathcal{L}_i = \mathcal{L}_i^n$; $y_i[0] = |\mathcal{L}|_i$;
For $k = 1, \ldots, n$
6. $y_i[k+1] = \min_{j\in\bar{\mathcal{N}}_i^-} y_j[k]$;
endFor
7. If $y_i[n] == |\mathcal{L}_i|$ and $i == \arg\min \mathcal{L}_i$, then i is a leader.

Next, in Algorithm 2, the followers consider the incoming transmission from only one parent p_i. In particular, we

make them choose the parent that minimizes the distance l_i from the leader set $\mathcal{J}$ returned by Algorithm 1. All remaining transmissions are neglected by the agents. Thus, the realization graph takes the form of a directed spanning forest, where the disjoint trees have the leaders as roots.

ALGORITHM 2: Spanning forest creation given $\mathcal{J}$ of Algorithm 1

Let $l_i[k]$ denote the distance from $\mathcal{J}$ to i at time k. Let p_i denote the parent of follower i.
1. Initialize $l_i[0] = 0$ for the leader agents $i \in \mathcal{J}$ and $l_i[k] = \infty$ for the followers.
For $k = 1, \ldots, n$
2. Receive $l_j[k]$ from the in-neighbors $j \in \mathcal{N}_i^-$;
3. Compute $l_i^* = \min_{j \in \mathcal{N}_i^-} \{l_j[k]\}$;
4. If $l_i^* + 1 < l_i[k]$, then $l_i[k] = l_i^* + 1$; $p_i = \arg\min\{l_j[k] : j \in \mathcal{N}_i^-\}$;
5. Transmit $l_i[k]$;
endFor
6. If i is a leader set $w_{ij} = 0$, $\forall j \in \mathcal{N}_i^-$;
7. If follower set $w_{ij} = 0$, $\forall j \in \mathcal{N}_i^- \neq p_i$ and $w_{ip_i} = 1$.

The above algorithms only determine the leaders $\mathcal{J}$ and the weights W. To ensure controllability or stabilizability-to-input, the agents have still to design c in (2) as well as the gains K_i in (3) as described in the following subsections. We note that both algorithms have complexity $\mathcal{O}\left(n^2\right)$ [9].

B. Controllability – Fixed Communication Topology

In this subsection, we present the solution to the first problem stated in (6) (see Theorem 1). We prove that by employing Algorithm 1 and Algorithm 2 (see Section IV-A) and by selecting c in (2) and gains K_i in (3), controllability is achieved with the minimum number of leaders.

In particular, we select

$$c^\intercal = \begin{bmatrix} 1 & 0 & \ldots & 0 \end{bmatrix}, \tag{8}$$

i.e. every agent transmits only its position $z_i = x_{i1}$. Then, it is sufficient for all agents to design the gains K_i, such that the eigenvalues of the internal dynamics (1) and (3), are distinct among agents. One feasible selection is as follows:

$$\sum_{j=1}^{m} K_{ij} s^{j-1} + s^m = (s+i)^m, \tag{9}$$

where i is the number of the i-th agent, assumed to be unique. Now we can state the main result for controllability.

Theorem 1: Let $\mathcal{J}^*$ and W^* be the outputs of Algorithm 1 and Algorithm 2, respectively. Let $P^* = \left\{W^*, \begin{bmatrix} 1 & 0 & \ldots & 0 \end{bmatrix}, K_i^*\right\}$ with K_i^* as in (9). Then, $(P^*, \mathcal{J}^*)$ is a solution to Problem (6). ◇

To show Theorem 1, we first prove some intermediate results. First, based on Corollary 1, we show that a necessary and sufficient condition to ensure *structural controllablity* is to actuate at least one (exactly one included) leader in every n-SCC of the communication graph. Thus, if we actuate exactly one leader per n-SCC (as Algorithm 1 does), we obtain structural controllability with the minimum number of leaders, i.e. we solve the following problem:

$$\begin{aligned} \min_{\mathcal{J}} \quad & |\mathcal{J}| \\ \text{s.t.} \quad & (A(P), B(\mathcal{J})) \text{ is structurally controllable} \end{aligned} \tag{10}$$

This is proved in the following lemma.

Lemma 1: The multi-agent network dynamics described in (5) is structurally controllable if and only if every n-SCC of the *communication graph* has at least one leader. Hence, Algorithm 1 returns a solution to (10) ◇

Although structural controllability guarantees controllability for almost all realizations of the design protocol P, there is still some possibility that the system will be uncontrollable. Thus, we have to carefully select P. We show that given the optimal solution $\mathcal{J}^*$ of problem (10), we can choose W^* as in Algorithm 2 and c, K_i as in (8), (9) to actually guarantee controllability.

Lemma 2: Given the collection of leaders $\mathcal{J}^*$ provided by Algorithm 1, c as in (8), gains K_i as in (9), and weights w_{ij} as in Algorithm 2, then system (5) is controllable. ◇

Therefore, the minimum number of leaders in problem (6) is exactly the number of n-SCCs.

C. Stabilizability-to-Input – Fixed Communication Topology

Next, we address the problem of determining in a distributed fashion the minimum number of leaders and the design protocol such that the system (5) is stabilizable-to-input. In Theorem 1, we obtained a controllable system, yet in order to define a control law, each leader would require global knowledge on $A(P)$ or its structure. In practice this is not feasible in a distributed setting and, therefore, we relax our requirements to stabilizability-to-input.

The agents still have to run Algorithm 1 and Algorithm 2 (section IV-A) to determine the leaders and the realization graph. However, the selection of K_i and $z_i = c^\intercal x_i$ in (2), (3) is different from the controllability problem. We will leverage the results from [19], so the agents transmit the following linear combination of the state:

$$z_i = c^\intercal x_i, \quad i \in \mathcal{V}, \tag{11}$$

where $c_j = \binom{m-1}{j-1}$, $j = 1, \ldots, m$ are binomial coefficients. The followers select K_i such that

$$\dot{x}_{im} = -(z_i - z_{p_i}) - \phi_i, \; i \in \mathcal{V} \setminus \mathcal{J}^* \tag{12}$$

where $\phi_i = \sum_{j=2}^{m} \binom{m-1}{j-2} x_{ij}$. Meanwhile, the leaders select K_i and external input v_i such that

$$\dot{x}_{im} = -(z_i - z_d) - \phi_i, \; i \in \mathcal{J}^*, \tag{13}$$

where $z_d = x_d$ is the desired position.

Theorem 2: Let w_{ij} be constructed as in Algorithm 2, given the leader indices $\mathcal{J}^*$ from Algorithm 1. In addition, let the signals z_i and vectors K_i be such that (11), (12) and (13) hold. Then, $|\mathcal{J}^*|$ is the minimum number of leaders, such that the system (5) is stabilizable-to-input, i.e., all agents converge to state $\begin{bmatrix} x_d & 0 & \ldots & 0 \end{bmatrix}^\intercal$. ◇

D. Controllability and Stabilizability-to-Input – Extension to Switching Communication Topology

Now, we extend the solution to the problem addressed in Sections IV-B-IV-C to the case where possible communication failures can occur, or, generally, the communication graph varies over time. These changes are described by the switching signal $\sigma(t)$ as described in (7). Upon a switching, it is sufficient for the agents to reconfigure themselves to maintain the spanning forest hierarchy, keeping exactly one leader per n-SCC of the communication graph. In consequence, the system remains controllable at each mode and the switching system (7) is controllable [20].

For the stabilization problem, we need an extra technical assumption. For more details one can refer to [21]. The simulations in Section VI show that in practice this assumption might not be necessary.

Assumption 1: Let the switching times be $t_k, k \in \mathbb{N}$. We assume that $\tau_k = t_k - t_{k+1} \in Y(T)$, where $T \subset \mathbb{R}_+$, $|T| < \infty$ is some finite set of positive numbers and $Y(T)$ is the (infinite) set generated by T, closed under addition and positive integer multiplication. $\diamond$

Under Assumption 1, maintaining the spanning forest hierarchy ensures stabilizability-to-input, by invoking Theorem 1 of [19]. Hence, all the agents converge to the desired position x_d. These are summarized in the following result.

Theorem 3: Suppose $\mathcal{J}_{\sigma(t)}$ is such that every n-SCC of the communication graph has exactly one leader and that the realization graph $\hat{\mathcal{D}}$ is a spanning forest, with trees rooted in the leaders. Then, with $z_i = x_{i1}$, and K_i as in (9), the system (7) is controllable. Moreover, under Assumption 1, with z_i as in (11) and K_i as in (12), (13), the system (7) is stabilizable-to-input, i.e., all agents converge to $\begin{bmatrix} x_d & 0 & \dots & 0 \end{bmatrix}^\mathsf{T}$. $\diamond$

Next, we describe the reconfiguration protocols that the agents need to run to ensure that the conditions of Theorem 3 hold. Towards this goal, notice that Algorithm 1 and Algorithm 2 were executed before, so that the corresponding information is available to the agents. Only two events can make the communication graph change:(a) edge removal; and (b) edge insertion. The reconfiguration protocols consist of two stages. First, there is a flooding stage, where the affected agent floods all its descendants with information describing the event. During this stage the agents might also execute some actions is sequential order, i.e. some initial update of their ancestor lists. Second, all descendants synchronize and re-execute some steps of Algorithms 1, 2 depending on the event's status. This stage is executed synchronously. We list all the possible cases.

Edge removal: Suppose edge (j, i) is deleted from the *communication graph*. Then, agent i removes the list of agent j: $\mathcal{L}_i^{new} = \mathcal{L}_i \setminus \mathcal{L}_j$ and initiates the flooding stage, by sending a *deletion* signal to its out-neighbors as well as the list to be deleted. Any agent k that receives the deletion signal, performs the deletion and resends the signal to its out-neighbors as well as the list to be deleted. Therefore, in at most n time steps, all the agents $k \in \mathcal{V}_i$ have deleted $\mathcal{L}_j$ from their list of ancestors $\mathcal{L}_k$, where $\mathcal{V}_i$ is the set of descendants of agent i. If agent j is the parent of agent i that the latter is following, i.e., $p_i = j$, then during the flooding stage agent i sends also a *repeat* signal. This repeat signal makes the receiving agents $k \in \mathcal{V}_i$ aware that they should re-run all steps of Algorithm 1 and Algorithm 2 in the second stage. Algorithm 1 is re-run to verify whether a new n-SCC is formed and whether a new leader should be selected. In addition, agents $k \in \mathcal{V}_i$ re-run Algorithm 2 to select the minimum distance parents (see–Fig. 1). If $p_i \neq j$, then the descendants $k \in \mathcal{V}_i$ only need to restart the first steps 1-5 of Algorithm 1, to update their lists. The reason is that the lists might still be incomplete; there may still be alternative paths from agents in $\mathcal{L}_j$ to agents in $\mathcal{V}_i$.

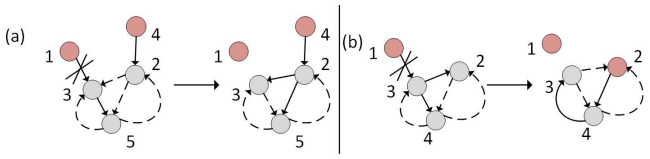

Fig. 1. This figure depicts the reconfiguration, i.e., the redefinition of the design protocol, after an edge is removed (marked by the cross) from the communication graph. The arrows denote the communications used in the realization graph, the dashed arrows represent those information actively neglected by the agents, and the leaders and followers are depicted by red and grey vertices, respectively. Only repeat cases are shown.

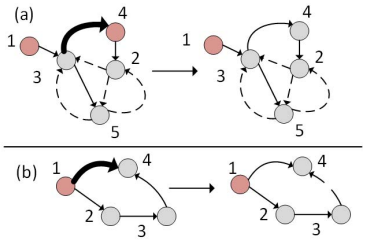

Fig. 2. This figure depicts the reconfiguration, i.e., the redefinition of the design protocol, after an edge is added (depicted by a solid-dark arrow) to the communication graph. The arrows denote the communications used in the realization graph, the dashed arrows represent those information actively neglected by the agents, and the leaders and followers are depicted by red and grey vertices, respectively. More specifically, two scenarios are depicted: (a) shows the ceasing of leadership case; and (b) shows the refollow case, see text for details.

Edge insertion: Suppose edge (j, i) is added to the communication graph. Then, agent i adds the list of agent j to its own $\mathcal{L}_i^{new} = \mathcal{L}_i \cup \mathcal{L}_j$ and initiates the flooding stage by sending an *update* signal to its out-neighbors. Any agent $k \in \mathcal{V}_i$ that receives an update signal, updates its list and retransmits the signal along with $\mathcal{L}_j$ to its out-neighbors. In at most n steps, all the agents $\mathcal{V}_i$ update their ancestor list. Now, there are two cases that require additional action. In any other case, no additional action is needed.

(i) agent i belonged to an n-SCC and $i \notin \mathcal{L}_j$ (see Fig. 2a). Then, this means that the n-SCC became a top-linked SCC. The leader in this previously n-SCC should cease its leadership. Thus, during the flooding stage the affected agent sends an additional *termination* signal to its descendants. The agent that was the leader in the original n-SCC, will

in due course receive the termination signal and cease the leadership. After the flooding stage, the descendants repeat the steps of Algorithm 2 to follow the parents that lead to the shortest path to a leader.

(ii) if $l_j^* + 1 < l_i^*$ (see Fig. 2b). Then agent i selects j as its new parent $p_i^{new} = j$, updates its distance $l_i^{*,new} = l_j^* + 1$ and sends a *refollow* signal to its descendants during the flooding stage. Each agent $k \in \mathcal{V}_i$ that receives the *refollow* signal, compares the in-neighbors' distances and reselects its parent, as in steps 2-7 of Algorithm 2.

Notice that the reconfiguration protocols are only run locally, i.e. only by the descendants $\mathcal{V}_i$ of the affected agent i. The remaining agents $\mathcal{V} \setminus \mathcal{V}_i$ are unaffected by the edge change and the reconfiguration actions.

Remark 1: The triggering of the reconfiguration actions is event-based; they start running only when an edge fails or is created, i.e., when the incoming signal is lost or when an agent receives from a new agent, respectively. The flooding stage is executed sequentially in the network. However, the subsequent actions require the agents to start steps of the Algorithm 1 or Algorithm 2 in *a synchronized way*. One way to synchronize the agents $k \in \mathcal{V}_i$ is to force each one to wait $n - l_{ik}$ time steps after it executes the flooding action, where l_{ik} is the distance from i to k. $\diamond$

Remark 2: The case where agents leave or join the network can be dealt with using the proposed strategy. For example, when an agent i leaves, all its out-neighbors can start the correction protocol at the same time. This is similar to agent i losing its connection with $j = p_i$. The only difference is that the list to be deleted is $\mathcal{L}_i$. If agent i joins the network, then it can send an insert signal along with the repeat signal. The repeat signal is needed to make sure the list of ancestors of the joining agent is correctly updated. $\diamond$

V. Discussion of Results

We note that controllability might not be guaranteed in the stabilizability-to-input problem and vice-versa. Although the directed spanning forest hierarchy of the realization graph (induced by the outputs of Algorithm 1 and Algorithm 2) is exactly the same, the constant vector c and the gains K_i are different in the design protocols associated with the two problems. However, in a practical scenario, the agents could have both designs available, selecting between controllability and stabilization without changing the leaders.

Notice that the protocols, i.e. Algorithm 1, require the agents to start execution at the same time. This is why we need them to have perfectly synchronized clocks, which might be challenging to achieve in practice. Another problem in distributed networks with directed communication links is that the transmitter may not know when the receiver will need its state. Hence, in our scenario, all agents $i \in \mathcal{V}$ should periodically transmit their lists $\mathcal{L}_i$ and distances l_i, resulting in increased communication bandwidth.

Finally, the condition that the communication graph should have a leader in each n-SCC at each time might actually be conservative in the case of switching topologies. A switching system can be controllable even if it is not controllable at every mode. Similarly, stabilizability-to-input can be guaranteed even if the communication graph is not spanned by a forest every time (see [19]). Thus, the optimality of the number of leaders may not hold for some switching sequences. Still, the proposed strategy is robust, since it works for every possible switching sequence.

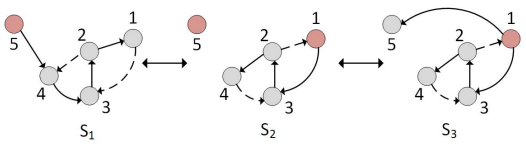

Fig. 3. This figure depicts the network switches between three modes. The transitions are such that only one edge changes at each time. The solid arrows denote the communications used in the realization graph, the dashed arrows represent those information actively neglected by the agents, and the leaders and followers are depicted by red and grey vertices, respectively.

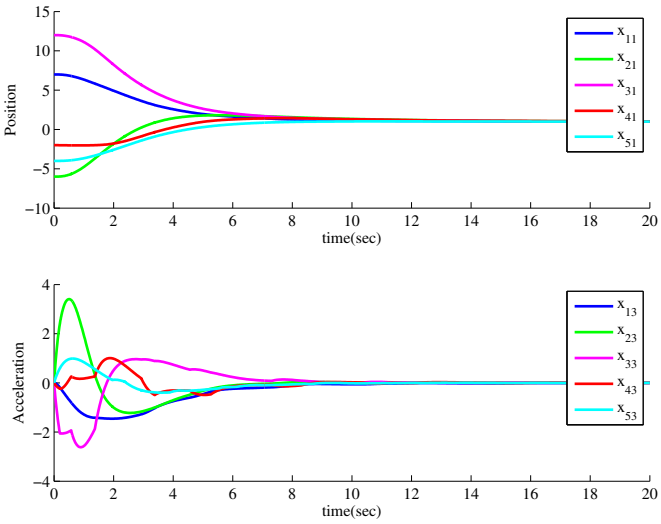

Fig. 4. The position and acceleration over time for the 5 agents interaction as described in Fig. (3). Notice that all agents' positions converge to $x_d = 1$.

VI. Simulation

We consider a scenario of five agents as shown in Fig. 3, where communication with agent 5 changes over time. More specifically, three communication topology modes will occur: (i) it can transmit to agent 1; (ii) it is isolated from the rest of the network; and (iii) it is only receiving from agent 1. We model the time interval between two switchings as a uniform random variable with unit variance and mean value 0.6 sec. When the system is in mode (ii), it switches to mode (i) with probability $\frac{1}{2}$ and to mode (iii) also with probability $\frac{1}{2}$. As discussed in Section IV-D, the agents reconfigure themselves, maintaining the directed spanning forest hierarchy in every mode. In particular, we consider the stabilizability-to-input problem, where the reference $x_d = 1$ is provided to the leaders. The agents have initial positions $\begin{bmatrix} 7 & -6 & 12 & -2 & -4 \end{bmatrix}^\intercal$ and zero initial velocities and accelerations. They converge to the desired position, while the velocity and acceleration converge to zero (see Fig. 4).

VII. Conclusion and Future Research

By maintaining a spanning forest hierarchy with trees rooted in the leaders, we can achieve both controllability and stabilizability-to-input with the minimum number of leaders. This holds for a static networks of multi-dimensional integrators. The results can be extended to the case of switching networks if we employ reconfiguration strategies that preserve this spanning forest hierarchy. However, our conditions for the results in the switching system case may be conservative, so it would be interesting to explore how could we relax them. Finally, it would be interesting to apply the protocols to a team of robotic agents, to explore how we could deal with the increased synchronization requirements.

Appendix

Proof of Lemma 1: From the internal dynamics (1) we deduce that

$$\left\{\left(x_{im}, x_{i(m-1)}\right), \ldots, \left(x_{i2}, x_{i1}\right)\right\} \subset E_X, i = 1, \ldots, n \quad (14)$$

From the communication protocol (3), all states x_{ik} are connected to x_{im} through K_i. Moreover, if agents (i, j) are connected in the communication graph then through w_{ji} all states x_{ik} are connected to x_{jm}. Thus,

$$\left\{\left(x_{ik}, x_{jm}\right),\ k = 1, \ldots, m\right\} \subset E_X \text{ if } (i, j) \in \mathcal{E}. \quad (15)$$

Finally, if an agent i is a leader then $(v_i, x_{im}) \in E_V$. This is illustrated in Fig. 5.

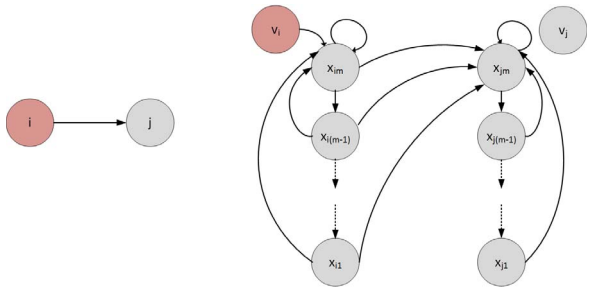

Fig. 5. A directed edge between two agents in the communication graph (left) and in the system graph (right). All the agents have self loops in the communication graph, but we hide them for simplicity. The red agent i is a leader. Thus the edge (v_i, x_{im}) exists. Notice that agent j has incoming links from all the states of agent i. This is due to the scalar z_i, which is a linear combination of the states of agent i.

Notice that each agent has a cycle covering all its states, which we denote by $C_i = \{x_{im} \to \ldots x_{i1} \to x_{im}\}$ (see–Fig. 5). Hence, the state graph of (5) can be covered by the disjoint union of cycles C_i. From Corollary 1, it is necessary and sufficient to actuate one state in each n-SCC of the state graph, associated with system (5).

From (14), (15), it follows that every n-SCC of the state graph is in a one-to-one relationship to an n-SCC in the communication graph. Thus, there is also a one-to-one relationship between the n-SCCs in the communication graph and n-SCCs in the state graph. Moreover, selecting a leader j is equivalent to actuating x_{jm} by (3). Thus, selecting a leader j in an n-SCC the communication graph is equivalent to actuating x_{jm} in an n-SCC of the state graph. Hence, the proof of the first part follows from Corollary 1.

To prove the second part, we invoke Theorem 1 in [9], which proves that Algorithm 1 returns exactly one leader per n-SCC of the communication graph. ■

Proof of Lemma 2: From the proof of Theorem 2 in [9], we get that Algorithm 2 produces a realization graph, which is a directed spanning forest of the communication graph, with the trees rooted in the leaders (self-loops included). Thus, the matrix $A(P^*)$ is block upper-triangular up to a permutation. The only elements which do not belong to the diagonal blocks are those corresponding to edges $(x_{p_i 1}, x_{im})$ for $i \in \mathcal{V} \setminus \mathcal{J}$. The diagonal blocks, related to the self-loops, are in the canonical controllable form, and their eigenvalues are determined by the characteristic polynomial (9). Since the eigenvalues of the agents are distinct, controllability of system (5) follows by invoking the Popov-Hataus-Belovich (PHB) criterion [22].

Since the eigenvalues of the agents are unique, for fixed λ, matrix $\lambda\mathbb{I} - A(P^*)$ can lose at most one column rank. Since each block is in controllable canonical form, this is true even if within one block we have repeated eigenvalues as in (9). The loss of rank occurs at the columns of the agent, for which λ is an eigenvalue, i.e. at agent k. The remaining columns of A have full rank. Consider the path from agent k to the respective leader $j \in \mathcal{J}$ in the realization graph, i.e. the path $k, p_k, p_{p_k}, \ldots, j$. Denote by $(\lambda\mathbb{I} - A(P^*))_{k\to j}$, the columns of $\lambda\mathbb{I} - A(P^*)$ that correspond to the agents in this path. Notice that the matrix $\left[\ (\lambda\mathbb{I} - A(P^*))_{k\to j} \ \middle|\ -B(\mathcal{J}^*)\ \right]$ has full column rank as its super-diagonal consists of -1 elements. Thus, the whole matrix $\left[\ \lambda\mathbb{I} - A(P^*) \ \middle|\ -B(\mathcal{J}^*)\ \right]$ has full column rank and the PHB criterion holds.

To visualize the above argument, assume that $m = 3$, $n = 2$ agent 1 is a follower and $p_1 = 2$ is a leader. Then the matrix $\left[\ \lambda\mathbb{I} - A(P^*) \ \middle|\ -B(\mathcal{J}^*)\ \right]$ has the form:

$$\left[\begin{array}{ccc|ccc|c} \lambda & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & \lambda & -1 & 0 & 0 & 0 & 0 \\ \star & \star & \star+\lambda & -1 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & \lambda & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & \lambda & -1 & 0 \\ 0 & 0 & 0 & \star & \star & \star+\lambda & -1 \end{array}\right].$$

The only element outside of the two diagonal blocks is the -1 corresponding to edge $(x_{p_i 1}, x_{im})$. Hence, the super-diagonal consists only of -1 elements. This implies that the rank of the above matrix is always 6. ■

Proof of Theorem 1: Feasibility follows by construction and Lemma 2. Optimality follows by Lemma 1, where $|\mathcal{J}^*|$ leaders are required to ensure structural controllability, which is a necessary condition for controllability. ■

Proof of Theorem 2: Looking at the equation of each leader we observe that it behaves like a follower, following a virtual static agent, say an agent with id 0, with state $x_0(t) = \left[\ x_d \quad 0 \quad \ldots \quad 0\ \right]^\intercal$ and no incoming connections. In additon, Algorithm 2 produces a realization graph, which is a directed spanning forest rooted in the leaders $\mathcal{J}^*$ [9]. Hence, the realization graph augmented with the virtual static

agent is a leaderless directed spanning tree for all $t \geq 0$ and we can apply Theorem 1 in [19]. All agents, including the agent 0, achieve consensus at a state $\begin{bmatrix} \xi & 0 & \dots & 0 \end{bmatrix}^\intercal$, for some constant ξ. Since agent 0 has no incoming link and it is static, $\xi = x_d$.

The number of leaders is still the minimum possible in the stabilizability-to-input problem, since if an n-SCC of the communication graph does not have a leader, then the communication graph augmented with the static agent does not have a spanning tree for all $t \geq 0$. Thus, by invoking the converse of Theorem 1 in [19], it follows that having one leader per n-SCC is a necessary condition. ■

REFERENCES

[1] Y. Chen, J. Lu, X. Yu, and D. J. Hill, "Multi-agent systems with dynamical topologies: Consensus and applications," *IEEE Circuits and Systems Magazine*, vol. 13, no. 3, pp. 21–34, 2013.

[2] Y. Cao, W. Yu, W. Ren, and G. Chen, "An overview of recent progress in the study of distributed multi-agent coordination," *IEEE Transactions on Industrial Informatics*, vol. 9, no. 1, pp. 427–438, 2013.

[3] H. Tanner, "On the controllability of nearest neighbor interconnections," in *Proceedings of the IEEE Conference on Decision and Control*, vol. 3, 2004, pp. 2467–2472.

[4] R. Olfati-Saber, J. A. Fax, and R. M. Murray, "Consensus and cooperation in networked multi-agent systems," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 215–233, 2007.

[5] A. Jadbabaie, J. Lin, and A. S. Morse, "Coordination of groups of mobile autonomous agents using nearest neighbor rules," *IEEE Transactions on Automatic Control*, vol. 48, no. 6, pp. 988–1001, 2003.

[6] S. Patterson, N. McGlohon, and K. Dyagilev, "Efficient, optimal k-leader selection for coherent, one-dimensional formations," in *Proceedings of the IEEE European Control Conference*, 2015, pp. 1908–1913.

[7] F. Lin, M. Fardad, and M. Jovanovic, "Algorithms for leader selection in stochastically forced consensus networks," *IEEE Transactions on Automatic Control*, vol. 59, no. 7, pp. 1789–1802, 2014.

[8] S. Pequito, S. Kar, and A. P. Aguiar, "On the complexity of the constrained input selection problem for structural linear systems," *Automatica*, vol. 62, pp. 193 – 199, 2015.

[9] S. Pequito, V. Preciado, and G. Pappas, "Distributed leader selection," in *Proceedings of the IEEE Conference on Decision and Control*, 2015, pp. 962 – 967.

[10] A. Clark and R. Poovendran, "A submodular optimization framework for leader selection in linear multi-agent systems," in *Proceedings of the IEEE Conference on Decision and Control and European Control Conference (CDC-ECC)*, 2011, pp. 3614–3621.

[11] S. Jafari, A. Ajorlou, and A. G. Aghdam, "Leader localization in multi-agent systems subject to failure: A graph-theoretic approach," *Automatica*, vol. 47, no. 8, pp. 1744–1750, 2011.

[12] I. Shames, A. Teixeira, H. Sandberg, and K. H. Johansson, "Distributed leader selection without direct inter-agent communication," in *Proceedings of the 2nd IFAC Workshop on Distributed Estimation and Control in Networked Systems (NecSys'10)*, ser. IFAC Proceedings Volumes, 2010, pp. 221–226.

[13] A. Franchi, H. Bülthoff, and P. Giordano, "Distributed online leader selection in the bilateral teleoperation of multiple uavs," in *Proceedings of the IEEE Conference on Decision and Control*, 2011, pp. 3559–3565.

[14] L. Wang, G. Chen, X. Wang, and W. K. S. Tang, "Controllability of networked mimo systems," *ArXiv e-prints*, May 2015.

[15] M. Mesbahi and M. Egerstedt, *Graph theoretic methods in multiagent networks*. Princeton University Press, 2010.

[16] S. Pequito, S. Kar, and A. P. Aguiar, "A framework for structural input/output and control configuration selection in large-scale systems," *IEEE Transactions on Automatic Control*, vol. 61, no. 2, pp. 303–318, 2016.

[17] J. Dion, C. Commault, and J. V. der Woude, "Generic properties and control of linear structured systems: a survey," *Automatica*, vol. 39, no. 7, pp. 1125–1144, 2003.

[18] K. J. Reinschke, *Multivariable control: a graph theoretic approach*, ser. Lect. Notes in Control and Information Sciences. Springer-Verlag, 1988, vol. 108.

[19] S. Su and Z. Lin, "Distributed consensus control of multi-agent systems with higher order agent dynamics and dynamically changing directed interaction topologies," *IEEE Transactions on Automatic Control*, vol. 61, no. 2, pp. 515–519, 2016.

[20] G. Xie, D. Zheng, and L. Wang, "Controllability of switched linear systems," *IEEE Transactions on Automatic Control*, vol. 47, no. 8, pp. 1401–1405, 2002.

[21] W. Ren and R. W. Beard, "Consensus seeking in multiagent systems under dynamically changing interaction topologies," *IEEE Transactions on Automatic Control*, vol. 50, no. 5, pp. 655–661, 2005.

[22] J. P. Hespanha, *Linear Systems Theory*. Princeton Press, 2009.