6-11-2008

Distributed Topology Control of Dynamic Networks

Michael M. Zavlanos
University of Pennsylvania, zavlanos@seas.upenn.edu

Alireza Tahbaz-Salehi
University of Pennsylvania, atahbaz@seas.upenn.edu

Ali Jadbabaie
University of Pennsylvania, jadbabai@seas.upenn.edu

George J. Pappas
University of Pennsylvania, pappasg@seas.upenn.edu

Distributed Topology Control of Dynamic Networks

Abstract

In this paper, we present a distributed control framework for controlling the topology of dynamic multi-agent networks. Agents are equipped with local sensing and wireless communication capabilities, however, due to power constraints, they are required to switch between two modes of operation, namely active and sleep. The control objective investigated in this paper is to determine distributed coordination protocols that regulate switching between the operation modes of every agent such that the overall network guarantees multi-hop communication links among a subset of so called boundary agents. In the proposed framework, coordination is based on a virtual market where every request to switch off is associated with a bid. Combinations of requests are verified with respect to connectivity and the one corresponding to the highest aggregate bid is finally served. Other than nearest neighbor information, our approach assumes no knowledge of the network topology, while verification of connectivity relies on notions of algebraic graph theory as well as gossip algorithms run over the network. Integration of the individual controllers results in an asynchronous networked control system for which we show that it satisfies the connectivity specification almost surely. We finally illustrate efficiency of our scalable approach in nontrivial computer simulations.

Keywords

distributed parameter networks, graph theory, mobile radio, mobile robots, multi-robot systems, radio links, algebraic graph theory, asynchronous networked control system, distributed coordination protocols, distributed topology control, dynamic multiagent networks, gossip algorithms, multihop communication links, nontrivial computer simulations, power constraints, virtual market, wireless communication capabilities

2008 American Control Conference
Westin Seattle Hotel, Seattle, Washington, USA
June 11-13, 2008

Distributed Topology Control of Dynamic Networks

Michael M. Zavlanos, Alireza Tahbaz-Salehi, Ali Jadbabaie and George J. Pappas

Abstract— **In this paper, we present a distributed control framework for controlling the topology of dynamic multi-agent networks. Agents are equipped with local sensing and wireless communication capabilities, however, due to power constraints, they are required to switch between two modes of operation, namely *active* and *sleep*. The control objective investigated in this paper is to determine distributed coordination protocols that regulate switching between the operation modes of every agent such that the overall network guarantees multi-hop communication links among a subset of so called boundary agents. In the proposed framework, coordination is based on a virtual market where every request to switch off is associated with a bid. Combinations of requests are verified with respect to connectivity and the one corresponding to the highest aggregate bid is finally served. Other than nearest neighbor information, our approach assumes no knowledge of the network topology, while verification of connectivity relies on notions of algebraic graph theory as well as gossip algorithms run over the network. Integration of the individual controllers results in an asynchronous networked control system for which we show that it satisfies the connectivity specification almost surely. We finally illustrate efficiency of our scalable approach in nontrivial computer simulations.**

I. Introduction

Distributed control of networked multi-agent systems has recently received considerable attention. Such systems typically consist of large numbers of inexpensive agents equipped with integrated sensing and wireless communication capabilities. While the agents' primary task is detection of certain physical changes within their proximity, their communication capabilities enable them to share the individually collected data with their peers, in order to achieve a global coordinated objective. Consequently, connectivity of the underlying network is a critical requirement.

In the presence of mobile agents or agents that can switch between *active* and *sleep* operating modes, maintaining connectivity of the underlying network becomes a challenging task due to the continuous changes in the network topology. In the former class of problems belongs [1], where a measure of local connectivity of a network is introduced that under certain conditions is sufficient for global connectivity as well as [2], where a controllability framework for state-dependent dynamic graphs is developed. Distributed maintenance of nearest neighbor links in formation stabilization is addressed in [3], while in [4] topology control of a mobile network is achieved by means of gossip algorithms and market-based coordination. In [5], the authors address the problem of maximizing the second smallest eigenvalue of the graph Laplacian, while a decentralized approach to this problem based on supergradient methods and distributed eigenvector computation is considered in [6]. Network connectivity for double integrator agents is investigated in [7], where existential as well as optimal controller design results are discussed. Equally challenging problems arise when the changes in the network topology are due to power constraints that require agents to occasionally switch off. In this context, duty cycling of sensor networks is investigated in [8]. Similarly, cone based topology control for ad-hoc sensor networks [9] as well as distributed connectivity control algorithms in the absence of exact location information [10] are among other variants of the problem investigated in the literature.

Inspired by the problems of the latter class, in this paper, we propose a distributed strategy for topology control of a network of stationary agents, each one capable of switching between on and off operation modes. In particular, given a subset of so called *boundary* agents, we design local coordination protocols that allow agents to individually switch on and off, while maintaining a connected network among the boundary agents. Our approach assumes no knowledge of the network topology, other than nearest neighbor information, while the proposed coordination scheme depends on the operation status of every agent. In particular, off agents can only switch on if by doing so they do not create clusters of active agents, disconnected from the main network. On the other hand, every request to switch off is associated with a bid and gossip algorithms run over the network allow agents to verify combinations of requests with respect to the connectivity specification and serve the one corresponding the highest aggregate bid. Connectivity is captured by the graph Laplacian matrix and can be checked in a distributed way by comparing the asymptotic values of a randomly initialized consensus run by all active agents in the network that do not request to switch off. Integration of the individual controllers results in a distributed networked multi-agent system for which we show that connectivity of the network involving the boundary agents is guaranteed almost surely.

The rest of this paper is organized as follows. In Section II we define the problem of topology control of networked multi-agent systems and develop the necessary graph theoretic background to capture connectivity. In Section III we develop the proposed distributed coordination protocols and discuss their properties. Finally, nontrivial computer simulations illustrating the efficiency of our approach are presented in Section IV.

This work is partially supported by ARO MURI SWARMS Grants W911NF-05-1-0219 and W911NF-05-1-0381 as well as DARPA DSO SToMP and NSF ECS-0347285.

Michael M. Zavlanos, Alireza Tahbaz-Salehi, Ali Jadbabaie and George J. Pappas are with the Department of Electrical and Systems Engineering, University of Pennsylvania, Philadelphia, PA 19104, USA. {zavlanos,atahbaz,jadbabai,pappasg}@seas.upenn.edu

II. Preliminaries and Problem Formulation

Consider a group of N stationary agents with integrated sensing and wireless communication capabilities, deployed in a p-dimensional space $\mathbb{R}^p$. Let $x_i \in \mathbb{R}^p$ denote the position of agent i and assume that each agent is subject to energy constraints and as a result, has two modes of operation, namely ON (active) and OFF (sleep). When ON, agent i is capable of communicating with other agents located in a disk of radius r_c centered at x_i. When OFF, all communication links with other agents in the network are disabled. Finally, we assume a set of so called *boundary agents*, which are agents that remain permanently ON. Such agents would be placed in key locations where continuous sensing and communication is required.

The setup described so far can be captured using an undirected dynamic graph $\mathbb{G}(t) = (\mathcal{V}_b \cup \mathcal{V}(t), \mathcal{E}(t))$, where $\mathcal{V}_b$ denotes the fixed set of *boundary agents*, $\mathcal{V}(t)$ denotes the set of agents that are ON at time t and $\mathcal{E}(t)$ denotes the set of communication links between all ON agents in $\mathcal{V}_b \cup \mathcal{V}(t)$ at time t. Any pair of ON agents i and j at time t such that $\|x_i - x_j\| < r_c$ are called *neighbors* or *adjacent* and the associated communication link is denoted by $(i,j) \in \mathcal{E}(t)$. In particular, we can define the set of neighbors of agent i at time t by $\mathcal{N}_i(t) = \{j \in \mathcal{V}_b \cup \mathcal{V}(t) \mid (i,j) \in \mathcal{E}(t)\}$. Then, the objective investigated in this paper can be stated as follows.

Problem 1 (Distributed Topology Control): Given a set of N agents, consisting of boundary and non-boundary ones, determine a distributed control framework that regulates the operation status of all non-boundary agents such that a communication path is maintained between any two boundary agents at all times.

The existence of a communication path between any two boundary agents is closely related to the notion of *connectivity* of the graph $\mathbb{G}(t)$. In particular, we say that $\mathbb{G}(t)$ is connected at time t, if there exists a path, i.e., a sequence of distinct nodes such that consecutive nodes are adjacent, between any two nodes in $\mathbb{G}(t)$. Hence, Problem 1 equivalently implies that we want $\mathbb{G}(t)$ to remain connected for al time.[1] The desired connectivity objective can be captured using the algebraic representation of the dynamic graph $\mathbb{G}(t)$. In particular, the structure of any dynamic graph $\mathbb{G}(t)$ can be equivalently represented by a dynamic *Laplacian* matrix,

$$L(\mathbb{G}(t)) = D(\mathbb{G}(t)) - A(\mathbb{G}(t))$$

where $A(\mathbb{G}(t)) = (a_{ij}(t))$ denotes the $|\mathcal{V}_b \cup \mathcal{V}(t)| \times |\mathcal{V}_b \cup \mathcal{V}(t)|$ *adjacency* matrix of the graph $\mathbb{G}(t)$, such that $a_{ii}(t) = 0$ and $a_{ij}(t) = 1$ if and only if $(i,j) \in \mathcal{E}(t)$, and $D(\mathbb{G}(t)) = \mathrm{diag}\big(\sum_{j \in \mathcal{V}_b \cup \mathcal{V}(t)} a_{ij}(t)\big)$ denotes its corresponding *degree* matrix.[2] The following lemma relates graph connectivity to the spectral properties of the Laplacian matrix $L(\mathbb{G}(t))$ [11].

[1]Clearly, communication paths between any two boundary agents may exist even if $\mathbb{G}(t)$ consists of multiple connected components that are disconnected from each other, as long as all boundary agents belong to the same connected component. We will not deal with this case here. Instead we will require that $\mathbb{G}(t)$ consists of a single connected component.

[2]We denote the cardinality of the set $\mathcal{V}$ by $|\mathcal{V}|$.

Lemma 2.1: Let $L(\mathbb{G})$ be the Laplacian matrix corresponding to the graph $\mathbb{G}$ and let $\lambda_1(L(\mathbb{G})) \leq \lambda_2(L(\mathbb{G})) \leq \ldots$ be its ordered eigenvalues. Then, $\lambda_1(L(\mathbb{G})) = 0$ with corresponding eigenvector $\mathbf{1}$, i.e., the vector of all entries equal to 1. Moreover, $\lambda_2(L(\mathbb{G})) > 0$ if and only if $\mathbb{G}$ is connected.

Lemma 2.1 implies that the second smallest eigenvalue, also called the Fiedler eigenvalue, of the positive semi-definite Laplacian matrix $L(\mathbb{G})$ is strictly positive, or equivalently, that $\ker L(\mathbb{G}) = \mathrm{span}\{\mathbf{1}\}$. As a result, we have the following well-known result.

Theorem 2.2: Consider a fixed graph $\mathbb{G}$ on N nodes associated with state variables $\theta_i(t) \in \mathbb{R}$ each, that are updated according to the set of linear differential equations $\dot{\theta}(t) = -L(\mathbb{G})\theta(t)$, where $\theta(t) = \big[\theta_1(t) \cdots \theta_N(t)\big]^T$. Then the network $\mathbb{G}$ is connected if and only if,

$$\lim_{t \to \infty} \theta(t) = \alpha \mathbf{1} \in \mathrm{span}\{\mathbf{1}\}. \tag{1}$$

for all initial conditions $\theta(0) \in \mathbb{R}^N$.

In other words, Theorem 2.2 says that all nodes in $\mathbb{G}$ will eventually reach a consensus on their state values $\theta_i(t)$, for all initial conditions, if and only if the graph $\mathbb{G}$ is connected. This theorem is in fact a special case of the well-known distributed consensus schemes over graphs discussed in [12]–[16]. In the case that $\mathbb{G}$ is disconnected, let $\mathbb{G}_c = (\mathcal{V}_c, \mathcal{E}_c)$ denote its c-th connected component. Then,

$$\ker L(\mathbb{G}) = \mathrm{span}\{\mathbf{1}_{\mathbb{G}_c} \mid \forall\, \mathbb{G}_c\}$$

where $\mathbf{1}_{\mathbb{G}_c}$ is an $N \times 1$ vector with its i-th entry equal to 1 if $i \in \mathcal{V}_c$ and equal to 0, otherwise. Consequently, for random initialization of the states $\theta(0)$, $\lim_{t\to\infty} \theta(t) = \sum_c \alpha_c \mathbf{1}_{\mathbb{G}_c}$ such that $\alpha_c \in \mathbb{R}$ are different for different connected components $\mathbb{G}_c$ almost surely. Therefore, connectivity of a network $\mathbb{G}$ can be verified almost surely by comparing the asymptotic state values (1) of all agents, for any random initialization.

III. Distributed Coordination

The main objective in this section is to derive a distributed coordination mechanism that allows agents to switch ON and OFF without violating the desired connectivity specification. Clearly, agents switching ON can, in general, only increase connectivity of the network, if by doing so they do not create new connected components. On the other hand, switching OFF while preserving connectivity becomes possible by means of a distributed virtual market, where agents that are ON bid in order to switch OFF. Combinations of switching OFF requests can be considered simultaneously. Each such request is verified with respect to the connectivity specification and among the ones that are safe, those corresponding to the highest aggregate bids are eventually processed.

Other than knowledge of the nearest neighbors of every agent, no further information regarding the topology of the network is required. Nevertheless, correctness of our approach relies on knowledge of all agents participating in every auction, which can be obtained in a distributed multi-hop fashion, as well as on some notion of *synchronization*

Algorithm 1 Initialization Phase for Agent i.

Require: $s_{a_i}^{[i]}(i) = 1$, $\text{phase}(i) = 1$;
Require: Self-Bid $b_{a_i}^{[i]}(i) \geq 0$;
1: **if** $\mathcal{I}_{\text{ON}}^{[i]} \cup \mathcal{I}_{\text{OFF}}^{[i]} \neq \mathcal{I}$ **then**
2: Find ON neighbors $\mathcal{N}_{\text{ON}}^{[i]} := \{j \in \mathcal{N}_i \mid s_{a_i}^{[i]}(j) = 1\}$;
3: Exchange status and bids with ON neighbors, i.e., $s_{a_i}^{[i]} := \max_{j \in \mathcal{N}_{\text{ON}}^{[i]}} \{s_{a_i}^{[i]}, s_{a_i}^{[j]}\}$, $b_{a_i}^{[i]} := \max_{j \in \mathcal{N}_{\text{ON}}^{[i]}} \{b_{a_i}^{[i]}, b_{a_i}^{[j]}\}$;
4: Find ON agents, $\mathcal{I}_{\text{ON}}^{[i]} := \{j \in \mathcal{I} \mid s_{a_i}^{[i]}(j) = 1\}$;
5: **else if** $\mathcal{I}_{\text{ON}}^{[i]} \cup \mathcal{I}_{\text{OFF}}^{[i]} = \mathcal{I}$ **then**
6: **if** $\text{wait}(i) = 0$ **then**
7: Set $\text{wait}(i) := \text{wait}(i) + 1$ and repeat steps 2-3;
8: **else if** $\text{wait}(i) = 1$ **then**
9: Set $\mathcal{R}_i := 2^{\{j \in \mathcal{I} \mid j = \text{argmax}_{l \in \mathcal{I}}^{(k)} \{b_{a_i}^{[i]}(l)\}\}}$
10: Set $\mathcal{I}_{\text{OFF}}^{[i]} := \mathcal{I} \backslash \mathcal{I}_{\text{ON}}^{[i]}$;
11: Switch to Verification Phase, i.e., $\text{phase}(i) = 2$;
12: **end if**
13: **end if**

of all agents to the same auction. Since, distributed systems, including the one proposed in this paper, are in general *asynchronous*, the desired synchronization is *event-based* and is obtained by labeling every auction in the set $\{1, 2, 3\}$ and requiring that any information exchange takes place only among neighbors that are in equally labeled auctions. Effectively, all agents that are ON are always synchronized in the sequence of auctions $\{1, 2, 3, 1, 2, 3, \dots\}$. Depending on the status of an agent, the coordination mechanism can be decomposed into an initialization phase, a verification phase and a decision phase for ON agents as well as a switching ON phase for OFF agents.

A. Initialization Phase for ON Agents

As discussed above, any new auction $a_i \in \{1, 2, 3\}$ that is initialized requires knowledge of all the agents participating in that auction. This information is encoded in a *status* vector $s_{a_i}^{[i]} \in \{0, 1\}^{1 \times N}$ such that $s_{a_i}^{[i]}(j) = 1$ if agent j is ON and $s_{a_i}^{[i]}(j) = 0$ if agent j is OFF. We further define the sets $\mathcal{I}_{\text{ON}}^{[i]}$ and $\mathcal{I}_{\text{OFF}}^{[i]}$ that consist *estimates* of the ON and OFF agents in the network, respectively. Initially, every agent i is aware of its own operation status only, i.e., $\mathcal{I}_{\text{ON}}^{[i]} = \{i\}$ and $\mathcal{I}_{\text{OFF}}^{[i]} = \emptyset$. On the other hand, the operation status of the whole network is obtained when $\mathcal{I}_{\text{ON}}^{[i]} \cup \mathcal{I}_{\text{OFF}}^{[i]} = \mathcal{I}$. The *initialization phase* for every agent i, denoted by $\text{phase}(i) = 1$, consists of updating $\mathcal{I}_{\text{ON}}^{[i]}$, given an estimate of $\mathcal{I}_{\text{OFF}}^{[i]}$. During this process, agent i also collects bids $b_{a_i}^{[i]}(j) \in \mathbb{R}$ from all agents j that desire to switch OFF and forms the set of requests to be verified with respect to connectivity. The initialization phase is described in Algorithm 1.

For every agent i entering the initialization phase we require that $s_{a_i}^{[i]}(i) = 1$, $s_{a_i}^{[i]}(j) = 0$ for all $j \neq i$, $\mathcal{I}_{\text{ON}}^{[i]} = \{i\}$ as well as a self-bid $b_{a_i}^{[i]}(i) \in \mathbb{R}$, such that $b_{a_i}^{[i]}(i) > 0$, if agent i desires to switch OFF, and $b_{a_i}^{[i]}(i) = 0$, otherwise. Note that $b_{a_i}^{[i]}(i) = 0$ for all *boundary* agents i. While the operation status of the whole network has not yet been obtained (line 1, Algorithm 1), agent i identifies its ON neighbors $\mathcal{N}_{\text{ON}}^{[i]}$ (line 2, Algorithm 1) and updates its status and bids vectors $s_{a_i}^{[i]}$ and $b_{a_i}^{[i]}$, respectively (line 3, Algorithm 1), as well as the set $\mathcal{I}_{\text{ON}}^{[i]}$ (line 4, Algorithm 1). Once the condition $\mathcal{I}_{\text{ON}}^{[i]} \cup \mathcal{I}_{\text{OFF}}^{[i]} = \mathcal{I}$ indicating full knowledge of the network status is satisfied (line 5, Algorithm 1), one more update of the status and bids vectors is required for agent i to also obtain *accurate* knowledge of the status and bids of all agents. This is due to the requirement that OFF agents can only switch ON if they have ON neighbors, in order to avoid clusters of ON agents that are disconnected from the main network. Clearly, collecting the status and bids of the new additions to the network requires no more than one communication cycle once the condition $\mathcal{I}_{\text{ON}}^{[i]} \cup \mathcal{I}_{\text{OFF}}^{[i]} = \mathcal{I}$, indicating that information from their ON neighbors has been received, is satisfied (line 7, Algorithm 1). To model this extra communication cycle, we introduce a dummy variable $\text{wait}(i)$, initialized at 0 and set to 1 once the extra update has been done. The final step of the initialization phase consists of identifying the agents associated with the k largest bids in $b_{a_i}^{[i]}$ and forming the set of requests $\mathcal{R}_i$ consisting of all 2^k combinations of these agents (line 9, Algorithm 1). Having accurate knowledge of the ON agents $\mathcal{I}_{\text{ON}}^{[i]}$, agent i can also update $\mathcal{I}_{\text{OFF}}^{[i]}$ and then switch to the verification phase (lines 11 and 12, Algorithm 1). Note that, since updating of the status and bids vectors of any agent i involves information provided by neighbors that are in the same auction a_i (line 3, Algorithm 1), all agents implementing Algorithm 1 will eventually converge to the same values for $s_{a_i}^{[i]}$ and $b_{a_i}^{[i]}$.

B. Verification Phase for ON Agents

The *verification phase* for every request $r \in \mathcal{R}_i$ consists of every ON agent $i \notin r$ running a consensus update,

$$x_{a_i}^{[i]}(r) := x_{a_i}^{[i]}(r) - \sum_{j \in \mathcal{N}_{\text{ON}}^{[i]} \backslash \{r\}} \left(x_{a_i}^{[i]}(r) - x_{a_i}^{[j]}(r) \right) \qquad (2)$$

where $x_{a_i}^{[i]}(r) \in \mathbb{R}$ is a randomly initialized scalar, on the reduced network obtained by assuming that all agents in r are OFF. If the reduced network is not connected, the associated consensus will converge to equal values $x_{a_i}^{[j]}(r)$ for all agents $j \in \mathcal{I}_{\text{ON}}^{[i]} \backslash \{r\}$ with probability zero. This can be checked in a distributed fashion, by means of a maximum and minimum consensus on the solutions of the corresponding consensus (2). For this, every agent i updates variables $M_{a_i}^{[i]}, m_{a_i}^{[i]} \in \mathbb{R}^{|\mathcal{R}_i|}$ containing the current values for all requests for the maximum and minimum consensus, respectively. The verification phase is described in Algorithm 2.

Running consensus (2) for request r consists of two stages, namely initialization and updating of consensus (2) and determining whether it has converged. With each one of these stages and every agent i we associate variables $u_{a_i}^{[i]}, c_{a_i}^{[i]} \in \{0, 1\}^{N \times |\mathcal{R}_i|}$, respectively, such that $u_{a_i}^{[i]}(j, r) = 1$ (similarly, $c_{a_i}^{[i]}(j, r) = 1$) indicates that agent i is aware that agent j has begun *updating* (similarly, has determined *convergence*) of

Algorithm 2 Verification Phase for Agent i.

Require: $s_{a_i}^{[i]}(i) = 1$, $\text{phase}(i) = 2$;
Require: $\min_{j \in \mathcal{I}_{\text{ON}}^{[i]}, r \in \mathcal{R}_i} c_{a_i}^{[i]}(j, r) = 0$;
1: Find ON neighbors $\mathcal{N}_{\text{ON}}^{[i]} := \{j \in \mathcal{N}_i \mid s_{a_i}^{[i]}(j) = 1\}$;
2: Exchange information with ON neighbors, i.e.,
$u_{a_i}^{[i]} := \max_{j \in \mathcal{N}_{\text{ON}}^{[i]}} \{u_{a_i}^{[i]}, u_{a_i}^{[j]}\}$,
$c_{a_i}^{[i]} := \max_{j \in \mathcal{N}_{\text{ON}}^{[i]}} \{c_{a_i}^{[i]}, c_{a_i}^{[j]}\}$,
$M_{a_i}^{[i]} := \max_{j \in \mathcal{N}_{\text{ON}}^{[i]}} \{M_{a_i}^{[i]}, M_{a_i}^{[j]}\}$,
$m_{a_i}^{[i]} := \min_{j \in \mathcal{N}_{\text{ON}}^{[i]}} \{m_{a_i}^{[i]}, m_{a_i}^{[j]}\}$;
3: Find set of requests that are not being updated,
$\mathcal{R}_{\neg u} := \{r \in \mathcal{R}_i \mid u_{a_i}^{[i]}(i, r) = 0\}$;
4: **for** all requests $r \in \mathcal{R}_{\neg u}$ **do**
5: Set $u_{a_i}^{[i]}(i, r) = 1$;
6: Randomly initialize a scalar $x_{a_i}^{[i]}(r)$;
7: **end for**
8: Find set of requests that are being updated but have not converged,
$\mathcal{R}_{\neg c}^{u} := \{r \in \mathcal{R}_i \mid u_{a_i}^{[i]}(i, r) = 1,\ c_{a_i}^{[i]}(i, r) = 0\}$;
9: **for** all requests $r \in \mathcal{R}_{\neg c}^{u}$ **do**
10: **if** $i \notin r$ **then**
11: Update $x_{a_i}^{[i]}(r)$ according to (2);
12: **if** $\{x_{a_i}^{[i]}(r)\} \uparrow$ and $\min_{j \in \mathcal{I}_{\text{ON}}^{[i]}} u_{a_i}^{[i]}(j, r) = 1$ **then**
13: Set $c_{a_i}^{[i]}(i, r) := 1$,
$M_{a_i}^{[i]}(r) := \max\{x_{a_i}^{[i]}(r), M_{a_i}^{[i]}(r)\}$,
$m_{a_i}^{[i]}(r) := \min\{x_{a_i}^{[i]}(r), m_{a_i}^{[i]}(r)\}$;
14: **end if**
15: **end if**
16: **end for**

consensus (2) for request r. During the initialization stage of the verification process, every agent i identifies requests $\mathcal{R}_{\neg u}$ that are not being updated yet (line 3, Algorithm 2) as well as requests $\mathcal{R}_{\neg c}^{u}$ that are being updated but the corresponding consensus (2) has not yet converged (line 8, Algorithm 2). For all requests $r \in \mathcal{R}_{\neg u}$, agent i initializes consensus (2) by setting $u_{a_i}^{[i]}(i, r) = 1$ and randomly initializing a scalar $x_{a_i}^{[i]}(r)$ (lines 5 and 6, Algorithm 2). On the other hand, for every request $r \in \mathcal{R}_{\neg c}^{u}$, if agent $i \notin r$, it updates $x_{a_i}^{[i]}(r)$ according to consensus (2) (line 15, Algorithm 2). Determining whether consensus (2) for request $r \in \mathcal{R}_{\neg c}^{u}$ has converged depends not only on convergence of the sequence $\{x_{a_i}^{[i]}(r)\}$, but also on the condition that all other ON agents $\mathcal{I}_{\text{ON}}^{[i]}$ have initialized the corresponding consensus (line 12, Algorithm 2). In this way, false convergence alarms due to delays in information propagation in the network, are avoided. When consensus (2) for a request $r \in \mathcal{R}_{\neg c}^{u}$ has converged, agent i sets $c_{a_i}^{[i]}(i, r) = 1$ and performs a maximum and minimum update on the variables $M_{a_i}^{[i]}(r)$ and $m_{a_i}^{[i]}(r)$, respectively (line 13, Algorithm 2). The verification phase lasts as long as there exist requests $r \in \mathcal{R}_i$ for which agent i is awaiting a convergence message $c_{a_i}^{[i]}(j, r)$ by ON agents $j \in \mathcal{I}_{\text{ON}}^{[i]}$, which translates to the requirement that $\min_{j \in \mathcal{I}_{\text{ON}}^{[i]}, r \in \mathcal{R}_i} c_{a_i}^{[i]}(j, r) = 0$. Note that, with every cycle of Algorithm 2, the variables $u_{a_i}^{[i]}, c_{a_i}^{[i]}, M_{a_i}^{[i]}, m_{a_i}^{[i]}$ are locally updated with information from agent's i ON neighbors $\mathcal{N}_{\text{ON}}^{[i]}$ that are in the same auction a_i (line 2, Algorithm 2), which guarantees that eventually all variables converge to the same values for all ON agents.

Algorithm 3 Decision Phase for Agent i.

Require: $s_{a_i}^{[i]}(i) = 1$, $\text{phase}(i) = 2$;
Require: $\min_{j \in \mathcal{I}_{\text{ON}}^{[i]}, r \in \mathcal{R}_i} c_{a_i}^{[i]}(j, r) = 1$;
1: Reset $s_{a_i^{old}}^{[i]}, b_{a_i^{old}}^{[i]} := \mathbf{0}_{1 \times N}$, $u_{a_i^{old}}^{[i]}, c_{a_i^{old}}^{[i]} := \mathbf{0}_{N \times |\mathcal{R}_i|}$ and $M_{a_i^{old}}^{[i]} := -\mathbf{10}_{1 \times |\mathcal{R}_i|}^{3}$, $m_{a_i^{old}}^{[i]} := \mathbf{10}_{1 \times |\mathcal{R}_i|}^{3}$, where $a_i^{old} = a_i - 1 \pmod 3$;
2: Find safe requests, $\mathcal{S}_i := \left\{r \in \mathcal{R}_i \mid \|M_{a_i}^{[i]}(r) - m_{a_i}^{[i]}(r)\| < \epsilon\right\}$
3: **if** $\mathcal{S}_i \neq \emptyset$ and $i \in \text{argmax}_{r \in \mathcal{S}_i} \left\{\sum_{j \in r} b_{a_i}^{[i]}(j)\right\}$ **then**
4: Set $a_i := a_i$ and $s_{a_i}^{[i]}(i) := 0$;
5: **else**
6: Set $a_i := a_i + 1 \pmod 3$;
7: Set $s_{a_i}^{[i]}(i) := 1$, $\mathcal{I}_{\text{ON}}^{[i]} := \{i\}$ and $\text{phase}(i) := 1$;
8: Set $\mathcal{I}_{\text{OFF}}^{[i]} := \mathcal{I}_{\text{OFF}}^{[i]} \cup \left\{\text{argmax}_{r \in \mathcal{S}_i} \left\{\sum_{j \in r} b_{a_i}^{[i]}(j)\right\}\right\}$;
9: **end if**

C. Decision Phase for ON Agents

Once agent i has obtained convergence messages for all requests from all ON agents, i.e., once $\min_{j \in \mathcal{I}_{\text{ON}}^{[i]}, r \in \mathcal{R}_i} c_{a_i}^{[i]}(j, r) = 1$, it enters the *decision phase* (Algorithm 3). Upon entering the decision phase both variables $M_{a_i}^{[i]}, m_{a_i}^{[i]}$ have converged to their global minimum and maximum values, respectively, over the whole network, which is due to simultaneous updating of all $c_{a_i}^{[i]}, M_{a_i}^{[i]}, m_{a_i}^{[i]}$ (line 2, Algorithm 2).

The decision phase consists of comparing the values of $M_{a_i}^{[i]}(r)$ and $m_{a_i}^{[i]}(r)$ for every request $r \in \mathcal{R}_i$ and deciding safety depending on whether these values are equal or not. In particular, every agent i identifies the set of safe requests $\mathcal{S}_i$ (line 2, Algorithm 3) and with every request $r \in \mathcal{S}_i$, it associates a cost corresponding to the aggregate bid values $\sum_{j \in r} b_{a_i}^{[i]}(j)$ of all agents participating in this request.[3] If requests that are safe with respect to connectivity exist, i.e., if $\mathcal{S}_i \neq \emptyset$, then the one associated with the highest cost is finally served. In other words, if agent i belongs to the request with the highest cost (line 3, Algorithm 3) it switches OFF (line 4, Algorithm 3). Otherwise, it initializes a new auction (line 6, Algorithm 3) and updates its set of OFF agents $\mathcal{I}_{\text{OFF}}^{[i]}$ by adding the agents in the highest-cost request that is to be served (line 8, Algorithm 3). This new set of OFF agents $\mathcal{I}_{\text{OFF}}^{[i]}$ is to be used in the following initialization phase (Algorithm 1). Whether agent i stays ON or switches OFF, it resets all variables corresponding to the previous auction

[3]Other cost functions could also be considered.

Algorithm 4 Agent i OFF switching ON

Require: $s_{a_i}^{[i]}(i) = 0$;

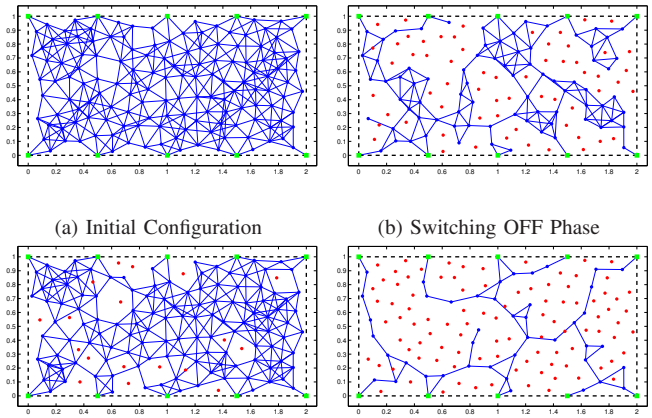
1: **if** $\text{pend}(i) = 0$ and agent i "wakes up" **then**
2: Find ON neighbors $\mathcal{N}_{\text{ON}}^{[i]} := \{j \in \mathcal{N}_i \mid s_{a_j}^{[j]}(j) = 1\}$;
3: Set $a_i := \max_{j \in \mathcal{N}_{\text{ON}}^{[i]}} \{a_j\} + 1 \pmod 3$;
4: Set $\text{pend}(i) = 1$;
5: **else if** $\text{pend}(i) = 1$ **then**
6: Find ON neighbors $\mathcal{N}_{\text{ON}}^{[i]} := \{j \in \mathcal{N}_i \mid s_{a_j}^{[j]}(j) = 1\}$;
7: **if** $\mathcal{N}_{\text{ON}}^{[i]} \neq \emptyset$ and $\max_{j \in \mathcal{N}_{\text{ON}}^{[i]}} \{a_j\} = a_i$ **then**
8: Set $s_{a_i}^{[i]}(i) := 1$ and $\text{phase}(i) := 1$;
9: Set $\mathcal{I}_{\text{ON}}^{[i]} := \{i\}$ and $\mathcal{I}_{\text{OFF}}^{[i]} := \mathcal{I}_{\text{OFF}}^{[l]} \backslash \{i\}$, where $l := \text{argmax}_{j \in \mathcal{N}_{\text{ON}}^{[i]}} \{a_j\}$;
10: **else if** $\mathcal{N}_{\text{ON}}^{[i]} = \emptyset$ **then**
11: Set $\text{pend}(i) = 0$;
12: **end if**
13: **end if**

(line 1, Algorithm 3), so that this *old* information can not be used in future equally labeled auctions.

D. OFF Agents Switching ON

Switching ON needs to satisfy two main objectives. First, no clusters of agents that are disconnected from the main network should be created. Second, all agents switching ON should synchronize themselves with the initialization phase of the current auction of their ON neighbors. The first objective is achieved by requiring that no OFF agent can switch ON if it has no ON neighbors. The second, on the other hand, relies on observing the sequence of auctions of the ON neighbors and joining when possible. In particular, when an OFF agent i "wakes up", it identifies its ON neighbors $\mathcal{N}_{\text{ON}}^{[i]}$ (line 2, Algorithm 4) and, if such neighbors exist, it prepares to join the auction followed by its neighbors' current auction by appropriately initializing a_i (line 3, Algorithm 4). Note that, due to synchronization of all ON agents to the same auction, at the time when agent i "wakes up", all its ON neighbors are either in the same auction or in two consecutive ones. The latter case occurs if "waking up" of agent i coincides with a transition of its ON neighbors to a new auction. Hence, the update in line 3 of Algorithm 4 is well defined. Once agent i has identified the auction it plans to join, it enters a *pending* phase denoted by $\text{pend}(i) = 1$ (line 4, Algorithm 4). During this phase, it keeps observing the auctions of its ON neighbors $\mathcal{N}_{\text{ON}}^{[i]}$ and as soon as one of its neighbors enters the auction agent i plans to join (line 7, Algorithm 4),[4] it switches ON and prepares to join the initialization phase by updating the variables $\mathcal{I}_{\text{ON}}^{[i]}$, $\mathcal{I}_{\text{OFF}}^{[i]}$ and $s_{a_i}^{[i]}(i)$ (lines 8 and 9, Algorithm 4). Note that, if during the time that agent i is pending, all its ON neighbors decide to switch OFF, then agent i remains OFF and waits for its neighbors to switch back ON (lines 10 and 11, Algorithm 4). Note also that, in this latter case none of its neighbors will switch to the auction that agent i wishes to join due to the update in line 4 of Algorithm 3, which guarantees that the condition in line 7 of Algorithm 4 will not be enabled.

[4]The maximum here is taken modulo 3.



(a) Initial Configuration (b) Switching OFF Phase

(c) Switching ON/OFF Phase (d) Final Configuration

Fig. 1. Distributed Topology Control for $N = 150$ agents.

E. Correctness of Distributed Coordination

Correctness of the proposed distributed coordination framework is obtained by construction and is discussed in details in the previous subsections. Those ideas are summarized in the following result.[5]

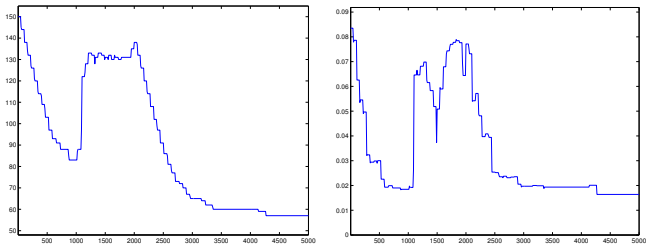
Proposition 3.1 (Correctness): Assume N agents, initially forming a connected network, each one of which is able to switch ON or OFF according to Algorithms 1-4. Then, a connected network including all boundary agents is guaranteed almost always.

IV. Simulations

In this section we illustrate the proposed topology control algorithm in a nontrivial connectivity task and show that it has the desired liveness, safety and scalability properties. In particular, we consider ten *boundary* agents symmetrically positioned on the upper and lower faces of a rectangle region and 140 other agents randomly distributed in the interior of the region such that all $N = 150$ agents initially form a connected network (Fig. 1(a)). Boundary agents are denoted with green squares, while the remaining agents with blue or red dots, depending on whether they are ON or OFF, respectively. The communication range is taken $r_c = .2$ and neighboring relations are denoted with lines drawn between them. The goal of this task is to let agents switch ON and OFF, always maintaining a connected network among the boundary agents.

To best illustrate our approach, we decompose the proposed connectivity task in three stages. First we only allow agents to switch OFF aiming at introducing some sparseness in the network (Fig. 1(b)). Then, we enable agents to also switch ON (with probability .01) and study how they are able to synchronize with the main network (Fig. 1(c)).

[5]Due to space limitations, the proof of this result is omitted.



(a) Number of ON Agents (b) Fiedler Eigenvalue

Fig. 2. Performance of Distributed Topology Control for $N = 150$ agents.

The final stage consists of pushing our algorithm to its limit and checking whether it is able to always maintain a connected network even if we never allow agents to switch ON again. In particular, we see that the final network is almost a tree structure (Fig. 1(d)). The requests verified with respect to connectivity during every auction consist of all 2^k combinations of the $k = 6$ highest bids.

Figs. 2(a) and 2(b) illustrate the number of agents that are ON and the Fiedler eigenvalue of the overall network, respectively. Note the three stages of our task as well as the fact that our approach succeeds in maintaining a connected network among the boundary agents. Fig. 3 shows a snapshot of the sequence of auctions during the transition time from the switching OFF stage of the task to the switching ON/OFF stage. Vertical lines indicate transitions from one auction to another. Horizontal lines, on the other hand, indicate either the last auction of OFF agents before they switched OFF (thin blue lines) or the time spent in every auction by ON agents (thick red lines). Observe that the sequence of auctions is of the form $\{1, 2, 3, 1, 2, 3, \dots\}$ as predicted, as well as that all ON agents are synchronized in equally labeled auctions. On the other hand, OFF agents that switch ON are always synchronized with the network in the desired auction (arrows and numbers indicating the auction). Finally, note that in the presence of more ON agents (ON/OFF stage) the time spent in every auction is slightly shorter than before due to the fact that nearest neighbor consensus updates converge faster in denser networks.

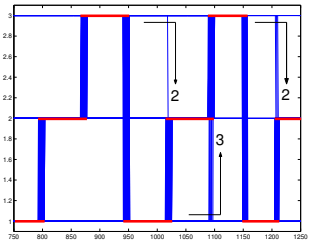


Fig. 3. Sequence of Auctions

V. Conclusions

In this paper, we presented a distributed control framework to regulate switching between the ON and OFF operation modes of a group of agents such that the overall network guaranteed multi-hop communication links among a set of boundary agents. Coordination was based on a virtual market where requests to switch off were associated with bids. Gossip algorithms were used to verify safety of combinations of these requests with respect to connectivity. Among the safe requests, the one corresponding to the highest aggregate bid was finally served. Connectivity was captured by the graph Laplacian matrix and was verified in a distributed way by comparing the asymptotic values of a randomly initialized consensus run by all active agents in the network. Other than nearest neighbor information, our approach required no knowledge of the network topology. We showed that the integrated networked control system satisfies the connectivity specification almost surely. We finally illustrated efficiency of our scalable approach by nontrivial computer simulations.

References

[1] D. P. Spanos and R. M. Murray, "Robust connectivity of networked vehicles," in *Proceedings of the 43rd IEEE Conference on Decision and Control*, Bahamas, Dec. 2004, pp. 2893–2898.

[2] M. Mesbahi, "On state-dependent dynamic graphs and their controllability properties," *IEEE Transactions on Automatic Control*, vol. 50, no. 3, pp. 387–392, 2005.

[3] M. Ji and M. Egerstedt, "Distributed formation control while preserving connectedness," in *Proceedings of the 45th IEEE Conference on Decision and Control*, San Diego, CA, Dec. 2006, pp. 5962–5967.

[4] M. M. Zavlanos and G. J. Pappas, "Distributed connectivity control of mobile networks," in *Proceedings of the 46th IEEE Conference on Decision and Control*, New Orlens, LA, pp. 3591–3596.

[5] Y. Kim and M. Mesbahi, "On maximizing the second smallest eigenvalue of a state-dependent graph laplacian," *IEEE Transactions on Automatic Control*, vol. 51, no. 1, pp. 116–120, Jan. 2006.

[6] M. C. DeGennaro and A. Jadbabaie, "Decentralized control of connectivity for multi-agent systems," in *Proceeding of the 45th IEEE Conference on Decision and Control*, San Diego, CA, Dec. 2006, pp. 3628–3633.

[7] G. Notarstefano, K. Savla, F. Bullo, and A. Jadbabaie, "Maintaining limited-range connectivity among second-order agents," in *Proceedings of American Control Conference*, Minneapolis, MN, 2006, pp. 2124–2129.

[8] C. F. Hsin and M. Liu, "Network coverage using low duty-cycled sensors: random & coordinated sleep algorithms," in *IPSN '04: Proceedings of the third international symposium on Information processing in sensor networks*, Berkeley, CA, 2004, pp. 433–442.

[9] L. Li, J. Y. Halpern, P. Bahl, Y. M. Wang, and R. Wattenhofer, "A cone-based distributed topology-control algorithm for wireless multi-hop networks," *IEEE/ACM Transactions on Networking*, vol. 13, no. 1, pp. 147–159, Feb. 2005.

[10] R. Wattenhofer and A. Zollinger, "XTC: A practical topology control algorithm for ad-hoc networks," in *Proceedings of the 18th IEEE International Parallel and Distributed Processing Symposium*, Santa Fe, NM, 2004, pp. 216–223.

[11] N. Biggs, *Algebraic Graph Theory*. Cambridge University press, 1993.

[12] 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.

[13] R. Olfati-Saber and R. M. Murray, "Consensus problems in networks of agents with switching topology and time-delays," *IEEE Transactions on Automatic Control*, vol. 49, no. 9, pp. 1520–1533, Sept. 2004.

[14] H. G. Tanner, A. Jadbabaie, and G. J. Pappas, "Flocking in fixed and switching networks," *IEEE Transactions on Automatic Control*, vol. 52, no. 5, pp. 863–868, May 2007.

[15] W. Ren and R. Beard, "Consensus of information under dynamically changing interaction topologies," in *Proceedings of the American Control Conference*, 2004, pp. 4939–4944.

[16] J. Cortes, S. Martinez, and F. Bullo, "Robust rendezvous for mobile autonomous agents via proximity graphs in arbitrary dimensions," *IEEE Transactions on Automatic Control*, vol. 51, no. 8, pp. 1289–1298, Aug. 2006.