2024 European Control Conference (ECC)
June 25-28, 2024. Stockholm, Sweden

A Distributed Algorithm to Establish Strong Connectivity in Spatially Distributed Networks via Estimation of Strongly Connected Components

Made Widhi Surya Atman and Azwirman Gusrialdi

Abstract— **This paper presents a distributed algorithm for ensuring the strong connectivity of spatially distributed networks where the communication network topology depends on both the position and communication range of the nodes. This is achieved by adding new links via adjusting the communication range and/or controlling the position of the nodes. The distributed algorithms rely on the estimation of strongly connected components of a dynamic network topology, accomplished through the utilization of the maximum consensus algorithm. The proposed strategies are scalable and converge in a finite number of steps without requiring information on the overall network topology. Finally, the proposed distributed algorithm is demonstrated through two case studies of ensuring strong connectivity in wireless networks with static and mobile nodes.**

Index Terms— **Distributed algorithms, finite-time, strongly connected digraph, max-consensus, spatially distributed network.**

I. INTRODUCTION

A spatially distributed network (SDN) consists of numerous devices/nodes with limited sensing, data processing, and communication capabilities. SDNs have been widely used to represent wireless sensor networks [1], [2] and robotic networks [3] for estimation, optimization, and control purposes. As SDNs may not have a control center, data processing/decision-making should be designed and implemented at the node level with direct data exchange between the neighboring nodes, also known as distributed estimation/optimization/control [4]. The communication network topology in SDNs is constructed based on the locations of the nodes and their communications range, as a node can only send information to the nodes within its communication range. Since the individual node may have different but limited communication range, the network topology in SDNs can then be modeled as a directed graph.

Strong connectivity of a directed graph associated with the communication network topology of SDNs is a crucial requirement in ensuring the convergence of many distributed algorithms. However, in practice, the communication network topology may not always be strongly connected. Therefore, it is of importance to ensure the strong connectivity of the directed graph, e.g., by adding new links, before executing any distributed estimation/optimization/control algorithms. More importantly, the procedure for ensuring the strong connectivity of the graph should also be done in a distributed manner due to the following reasons: i) the overall network topology is often not available due to geographical constraints; ii) to comply with the feature of distributed algorithms that will be deployed in the SDNs.

This paper focuses on designing a distributed algorithm for establishing strong connectivity of a directed graph associated with the communication network topology in SDNs. This is achieved in practice by adjusting the communication range and/or controlling the position of the nodes. The initial work on this topic dated back to two decades ago, e.g., [5], [6]. The proposed approach requires each node to initially send a neighbor-discovery message and record the acknowledgement from the receiving nodes to establish connection. This process is repeated while increasing its communication range or radius, and a termination condition is presented which ensures that the resulting undirected graph of the communication network is connected. However, in these studies, the connection between any pair of nodes is only considered when the resulting link is undirected.

On the other hand, there have been studies on network topology control whose main objective is to maintain the (strong) connectivity of a graph [7]–[10]. However, these results assume that the network to be controlled is already strongly connected in the first place. A distributed algorithm to verify and ensure strong connectivity of a directed graph was proposed in our prior work [11], [12]. Briefly speaking, the method relies on estimating the paths and the strongly connected components of a graph via the maximum consensus algorithm. However, the setting assumes that each node can add new links to any nodes in the network. Hence, the results are not directly applicable to SDN.

In this paper, we extend our approach in [11], [12] to design a distributed algorithm for establishing strong connectivity in spatially distributed networks. Specifically, the main contribution of this paper is three-fold. First, we improve the distributed strategy for estimating the strongly connected component (SCC) in a directed graph initially introduced in [11], [12], reducing the memory requirement for exchanging information. Moreover, this improved strategy eliminates the necessity of knowing the node's out-neighbors and the exact total number of nodes, as required in [11], [12]. Second, we extend the SCC estimation algorithm to deal with dynamic topology due to the motion of the nodes and the adjustment of the node's communication range. Finally, we demonstrate the distributed link addition algorithm for ensuring strong connectivity of SDNs within two case-studies of wireless network with static and mobile nodes.

This work was supported in part by the Research Council of Finland under the Academy Project decision number 330073 and the Profiling Action on System-on-Chip and Wireless Technology for Intelligent Machines at Tampere University.

M. W. S. Atman and A. Gusrialdi are with Faculty of Engineering and Natural Sciences, Tampere University, Tampere 33014, Finland. Emails: widhi.atman@tuni.fi and azwirman.gusrialdi@tuni.fi

The remainder of this paper is organized as follows. In Section II, we review the basic notions from graph theory and provide the problem settings. Section III presents the distributed algorithm to ensure the strong connectivity in spatially distributed networks. The demonstrations of the proposed algorithm for ensuring strong connectivity in wireless networks are presented in Section IV and followed with concluding remarks in Section V.

II. PROBLEM FORMULATION

In this section, we recall some basic notions of the graph theory and then define the problem settings within this paper.

A. Notation and Graph Theory

Information exchange between nodes in a network can be modeled by a *directed graph (digraph)*. A directed graph is denoted by $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with a set of nodes $\mathcal{V}$ and a set of edges (links) $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$. A graph $\mathcal{G}_1 = (\mathcal{V}_1, \mathcal{E}_1)$ is a *subgraph* of $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ if $\mathcal{V}_1 \subseteq \mathcal{V}$ and $\mathcal{E}_1 \subseteq \mathcal{E}$. The existence of an edge $(i, j) \in \mathcal{E}$ denotes that the node j can obtain information from the node i, or the node i is accessible to the node j. Here, node i is said to be an *in-neighbor* of node j. Within this paper, the set of all in-neighbors of a node i is denoted by $\mathcal{N}_i = \{j \in \mathcal{V} \mid (j, i) \in \mathcal{E}\}$. A *path* is a sequence of nodes $(i_1, i_2, \ldots, i_p), p > 1$, such that i_j is an in-neighbor of i_{j+1} for $j = 1, \ldots, p-1$. An *elementary path*, is a path in which no nodes appear more than once. Node i can *reach* node j, i.e., j is *reachable from* i, if there is a path from i to j.

A graph is said to be *strongly connected* if there is a path between any pair of distinct nodes, and it is called *weakly connected* if the graph obtained by adding an edge (j, i) for every existing edge $(i, j) \in \mathcal{E}$ in the original graph is strongly connected. A *strongly connected component* (SCC) of a directed graph $\mathcal{G}$ is a subgraph of $\mathcal{G}$ that is strongly connected and maximal, as such, no additional edges or vertices from $\mathcal{G}$ can be included in the subgraph without breaking its property of being strongly connected.

Within this paper, let $\mathbb{R}$ be the set of real numbers and $\mathbb{Z}_{\geq 0}$ be the set of non-negative integers. By $\mathbf{1}_n \in \mathbb{R}^n$ and $\mathbf{0}_n \in \mathbb{R}^n$, we denote the all ones vector and zeros vector in n-dimension, respectively. For a given set $\mathcal{N}$, $|\mathcal{N}|$ denotes the number of elements in this set. Vectors are denoted as boldface lower case and matrices are denoted as capital boldface letters. Finally, the state associated with a node $i \in \mathcal{V}$ is represented by the subscript operator, for example state $\boldsymbol{a} \in \mathbb{R}^b,\ b > 1$ for node i is shown as $\boldsymbol{a_i}$ and the j-th element of vector $\boldsymbol{a_i}$ (with $j \leq b$) is denoted by $a_{i,j}$.

B. Problem Formulation

Consider a network consisting of at maximum $\bar{n}$ number of nodes. We assume that each node is equipped with its own computational resources and is assigned with a unique identifier[1], where each identifier can be mapped into $i \in \{1, \ldots, \bar{n}\}$. The location of each node i is denoted by $\boldsymbol{q}_i \in \mathbb{R}^3$. Additionally, each node i is equipped with a communication device that can transmit information to any other nodes up to a certain radius r_i. Without loss of generality, we consider that each node i can adjust its position and communication radius according to the kinematic model

$$\dot{\boldsymbol{q}}_i = \boldsymbol{u}_i, \quad \dot{r}_i = v_i \tag{1}$$

with $\boldsymbol{u}_i$ and v_i denote the velocity control input for the position and communication radius, respectively. Note that, as we will show later in Section IV, a stationary node is a special case of this model where $\boldsymbol{u}_i := \mathbf{0}_3$ for all time.

We consider the case where not all nodes in the network are *active* and participating in the information exchange. Here, we denote the active nodes as $\mathcal{V} \subseteq \{1, \ldots, \bar{n}\}$ and the information $\bar{n}$ becomes an upper bound of $|\mathcal{V}|$. The communication network between these nodes can then be modeled as a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with $\mathcal{E}$ denotes the set of edges between the active nodes. Here, $(i, j) \in \mathcal{E}$ when $||\boldsymbol{q}_i - \boldsymbol{q}_j|| \leq r_i$ and equivalently $\mathcal{N}_j = \{i \in \mathcal{V} \mid ||\boldsymbol{q}_i - \boldsymbol{q}_j|| \leq r_i\}$. This directed graph is often described as a spatially distributed network (SDN) [14].

In this paper, it is assumed that the information of the overall network topology $\mathcal{G}$ is not available. That is, the available information is only the local information of the individual node given by the following assumption.

Assumption 1: Each node i only knows the information on $\mathcal{N}_i$ and the upper bound of $\bar{n}$.

We restrict the discussion in this paper to the case where initially the graph $\mathcal{G}$ is not disconnected, i.e., it is at least a weakly connected digraph. Furthermore, we consider a discrete-time settings and assume that communication between the nodes occurs in a synchronous manner, where communication instants may either be defined by a clock or by the occurrence of external events. This can be realized, e.g., by allowing the node to have access to global/universal time with predetermined execution timing and interval.

The objective of this paper is to develop a distributed algorithm for solving the following problem:

Problem 1 (Connectivity Augmentation): For a spatially distributed network $\mathcal{G}$, design control input $\boldsymbol{u}_i$ and v_i for each node $i \in \mathcal{V}$ in a distributed manner, such that additional edges $\Delta\mathcal{E}^+ \subseteq (\mathcal{V} \times \mathcal{V}) \setminus \mathcal{E}$ are added to ensure that the resulting graph $\mathcal{G}^* = \{\mathcal{V}, \mathcal{E} \cup \Delta\mathcal{E}^+\}$ is strongly connected.

III. DISTRIBUTED ELECTION OF LINKS FOR ESTABLISHING STRONG CONNECTIVITY

The proposed approach to solve Problem 1 is divided into the following two steps: 1) to identify new links to be augmented resulting in strong connectivity, and 2) to compute the control input $\boldsymbol{u}_i$ and v_i that result in establishment of the new links obtained in step 1 while maintaining the existing edges. In this section, we focus on the first step. As the computation for $\boldsymbol{u}_i$ and v_i largely relies on the considered scenarios and its associated constraints, comprehensive discussions related to the second step will be presented in Section IV.

[1]Note that the unique identifier is a standard assumption commonly used in designing distributed algorithms which can be realized e.g., by using MAC address, see for example [7], [13].

To identify new links to be established, each node first estimates the strongly connected component (SCC) that it belongs to. Here, we first describe a method for distributed estimation of the SCC for a fixed network topology, followed by its extension to the dynamic topology. Once each node finishes estimating its own SCC, the elected links for link addition can then be selected from itself towards its in-neighbors that does not belong to its own SCC. More detailed descriptions are given in the following subsections.

A. Distributed Estimation of SCCs on Fixed Topology

In the context of SDN, we refer to a fixed topology as a window of time when the relationship between each pair of nodes remains constant. This covers both cases where 1) all nodes' position and communication radius remain constant, and 2) the changes of the nodes' position or communication radius do not result in addition or deletion of any edges.

First, let us introduce the term *information number* of node i as the number of nodes (including itself) that can reach node i. Furthermore, for each node $i \in \mathcal{V}$, we introduce the state $\boldsymbol{x}_i[t] \in \mathbb{R}^{\bar{n}}$ to estimate all nodes that can reach node i, and simultaneously estimate its own and the other node's information number. Each $\boldsymbol{x_i},\ \forall i \in \mathcal{V}$ is initialized as

$$x_{i,j}[t_0 := 0] = \begin{cases} 1, & \text{if } j = i \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

To ease the discussion, let us define an intermediate state $\boldsymbol{z}_i[t] \in \mathbb{R}^{\bar{n}}$. Then, each $\boldsymbol{x_i}$ is updated as follows.

$$z_{i,k}[t] = \max_{j \in \mathcal{N}_i \cup \{i\}} x_{j,k}[t], \qquad \forall k \in \mathcal{V} \tag{3}$$

$$x_{i,j}[t+1] = \begin{cases} \left|\{k \in \mathcal{V} \mid z_{i,k}[t] > 0\}\right|, & \text{if } j = i \\ z_{i,j}[t], & \text{otherwise.} \end{cases} \tag{4}$$

Note that to ensure that the correct information number is computed, the update rule (4) needs to be executed after each node receives the information from all of its in-neighbors and computes (3).

Given the existing shortest path between any pair of nodes in $\mathcal{G}$, let D denotes the maximum length of those paths[2], where $D \leq |\mathcal{V}| - 1$. We then have the following lemma,

Lemma 1: Given a fixed topology of digraph $\mathcal{G}$ and each node executes update rule (3)–(4) starting with each initial values as in (2) simultaneously at the time t_0. Then, the state $\boldsymbol{x_i}[t]$ converges for $t \geq t_0 + 2D$. Furthermore, the convergence value $x_{i,j}[t] \geq 0$ (for $t \geq t_0 + 2D$) equals to the information number of node j that can reach node i.

Proof: The initialization and update rule for state $\boldsymbol{x_i}$ is a condensed version of a two-stage maximum consensus update protocol inspired from [11], [12]. The first stage involves the state $\boldsymbol{a}_i[t] \in \mathbb{R}^{\bar{n}}$, initialized with $a_{i,j}[t_0] = 1$ if $j = i$ and 0 otherwise, and updated via $a_{i,j}[t+1] = \max_{k \in \mathcal{N}_i \cup \{i\}} a_{k,j}[t]$. Note that as the information propagates from one node to another via the maximum consensus protocol, for each existence of elementary path with length l_{ji} from node j to node i, then the initial value of $a_{i,j}[t_0] = 0$ will change to 1 after l_{ji} iteration, i.e., $a_{i,j}[t_0 + l_{ji}] = 1$. As the maximum of existing elementary path is D, then the value of $\boldsymbol{a}_i$ will remain constant after $t_0 + D$. Note that when no path exists from node j to node i, then $a_{i,j}[t_0 + D]$ remains 0. Next, the second stage involves the state $\boldsymbol{c}_i[t] \in \mathbb{R}^{\bar{n}}$, that initialized simultaneously at $t_0 + D$ with $c_{i,j}[t_0 + D] = |\{k \in \mathcal{V} \mid t_{i,k}[t_0 + D] > 0\}|$ if $j = i$ and 0 otherwise, and updated via $c_{i,j}[t+1] = \max_{k \in \mathcal{N}_i \cup \{i\}} c_{k,j}[t]$. With a similar argument as previous, the value of $\boldsymbol{c}_i$ will remain constant after $t_0 + 2D$. The modification for $\boldsymbol{x}_i$ reduced the required memory for communication with identical results to the presented two-stage, i.e., $\boldsymbol{c}_i[t_0 + 2D] = \boldsymbol{x}_i[t_0 + 2D]$. Hence, the state $\boldsymbol{x}_i$ will not change after $2D$ iterations. ■

Lemma 1 provides a guideline for each node $i \in \mathcal{V}$ to evaluate the state $\boldsymbol{x_i}$. Specifically, after $2D$ iterations passed from t_0, node i then has access of node j's information number. Here, $x_{i,j} = 0$ represents the case where node j cannot reach node i. As the value of D is not known to each node, the evaluation can be done via $\boldsymbol{x_i}[t_0 + 2\bar{n}]$ instead, as $\bar{n} \geq D$ and each node has information on $\bar{n}$.

Finally, we now proceed with the distributed identification of the SCC that each node i belongs to by evaluating $\boldsymbol{x_i}[t_0 + 2\bar{n}]$. Let us define node i's estimation of its own SCC as the set $\mathcal{C}_i$. Additionally, the set $\mathcal{P}_i$ refers to node i's estimation of all (predecessor) nodes that can reach all the nodes in $\mathcal{C}_i$. Then, let us recall the following lemmas from [11], [12].

Lemma 2 (Lemma 2 in [12]): If node i is reachable from node j (i.e., $x_{i,j}[t_0 + 2\bar{n}] > 0$) and nodes i and j have the same information number (i.e., $x_{i,j}[t_0 + 2\bar{n}] = x_{i,i}[t_0 + 2\bar{n}]$), then nodes i and j belong to the same SCC (i.e., they are mutually reachable to each other).

Lemma 3 (Lemma 3 in [12]): For each node i, the other nodes in the set $\mathcal{P}_i$ have a smaller (positive) information number compared to the node i (equivalently any nodes in $\mathcal{C}_i$). Specifically, the information number of node i satisfy $x_{i,i}[t_0 + 2\bar{n}] \geq |\mathcal{C}_i| + \max_{j \in \mathcal{P}_i} x_{j,j}[t_0 + 2\bar{n}]$.

The above lemmas provide the foundation that allows each node $i \in \mathcal{V}$ to estimate $\mathcal{C}_i$ and $\mathcal{P}_i$, as summarized in the following proposition.

Proposition 1: Given a fixed topology of digraph $\mathcal{G}$ and each node $i \in \mathcal{V}$ executes update rule (3)–(4) starting with each initial values as in (2) simultaneously at the time t_0. Then, for time $t \geq t_0 + 2\bar{n}$, the SCC that contains the node i is identical to the set

$$\mathcal{C}_i := \{\forall j \in \mathcal{V} \mid x_{i,j}[t] = x_{i,i}[t]\}. \tag{5}$$

Furthermore, the set of all (predecessor) nodes that can reach any node in $\mathcal{C}_i$ is identical to the set

$$\mathcal{P}_i := \{\forall j \in \mathcal{V} \mid 0 < x_{i,j}[t] < x_{i,i}[t]\}. \tag{6}$$

The proof of the Proposition 1 follows directly as a result from Lemma 2 and 3. Note that node i's local estimation of $\mathcal{C}_i$ and $\mathcal{P}_i$ are identical to all the other nodes which belong to the same SCC (i.e., $\mathcal{C}_j = \mathcal{C}_i$ and $\mathcal{P}_j = \mathcal{P}_i$ for all $j \in \mathcal{C}_i$).

Remark 1: In contrast to our prior work [11], [12], the approach to estimate SCC for a fixed topology proposed in this paper uses less memory to exchange the information, i.e., a vector of $\bar{n}$ instead of $2\bar{n}$.

[2]If the graph $\mathcal{G}$ is strongly connected, then D is identical to its *diameter*.

B. Distributed Estimation of SCCs on Dynamic Topology

The connectivity property of an SDN directly depends on the state of the nodes, i.e., its position and communication radius. Accounting for the variation in the edges due to changes in nodes' states or the variation of the active nodes due to joining or leaving nodes, we can generally expect that the modeled directed graph is time varying. To this end, we propose a method for the nodes to reinitialize the distributed estimation of SCCs when the topology changes. Let $\{t_k : k \in \mathbb{Z}_{>0}\}$ denote the time sequence of the changes in topology, i.e., there exists a node i whose $\mathcal{N}_i[t_k] \neq \mathcal{N}_i[t_k - 1]$.

To enable the individual node to record and track the changes in the network, we introduce a new state $m_i \in \mathbb{R}$ whose initial value is set to $m_i[t_0 := 0] = 0$ for all $i \in \mathcal{V}$. Then, let us define an intermediate state $d[t] \in \mathbb{R}$ to compute

$$d_i[t] = \begin{cases} t_k, \text{ if } \mathcal{N}_i[t := t_k] \neq \mathcal{N}_i[t_k - 1] \\ \max_{j \in \mathcal{N}_i \cup \{i\}} m_j[t], \text{ otherwise.} \end{cases} \tag{7}$$

The update rule of the state m_i and the reset mechanism for the state $\boldsymbol{x}_i$ are then given as follows.

$$m_i[t+1] = d_i[t] \tag{8}$$

$$\boldsymbol{x}_i[t+1] = \begin{cases} \text{reset as (2), if } d_i[t] > m_i[t] \\ \text{normal update as (3)–(4), otherwise.} \end{cases} \tag{9}$$

Essentially, the computation (7) updates or compares the time of the last changes t_k in the network that is locally detected by the node i. This information is then stored in (8) and distributed further towards all nodes that node i can reach. Accordingly, (7) needs to be computed prior to (8) and (9).

Let $\mathcal{G}[t_k]$ be the resulting graph after the changes at t_k. Given the existing shortest path between any pair of nodes in $\mathcal{G}[t_k]$, let $D[t_k]$ denotes the maximum length of those paths. We can then extend the results in Lemma 1 for the case of dynamic topology as follows.

Lemma 4: Given a time-varying digraph $\mathcal{G}[t_k]$ where each node executes update rule (3)–(4) and (7)–(9) with their corresponding initialization rules. Each node can correctly estimate its own SCC corresponding to the digraph $\mathcal{G}[t_k]$ if the digraph $\mathcal{G}[t_k]$ remains fixed for at least $3D[t_k]$ iterations, i.e., $t_{k+1} - t_k \geq 3D[t_k]$.

Proof: The proof follows similarly to the proof of Lemma 1. Here, the maximum consensus in (7) is analogous to an additional stage for $D[t_k]$ iterations (prior to the two-stage maximum consensus in the proof of Lemma 1) that propagates the information on topology changes and initiates the reset mechanism. Hence, the total requirement of $3D[t_k]$ iterations. Then, as the topology is fixed since t_k, the correct SCC estimation can be inferred via Proposition 1. ■

Note that as the information on $D[t_k]$ is not easily available, the condition $t_{k+1} - t_k \geq 3D[t_k]$ in the above Lemma can be substituted with $t_{k+1} - t_k \geq 3\bar{n}$. Then, the implication of Lemma 4 is that, in practice, the topology needs to be fixed for at least $3\bar{n}$ to allow a correct estimation of SCC. In practice, the distributed controller of the nodes can be designed to ensure this condition is satisfied, as we will demonstrate in Section IV.

Remark 2: Assume that the change of topology occurs at node i at a time t_k. The changes on the SCCs composition and estimations only affect node i and all nodes that node i can reach. The rest of the nodes and their SCCs remain unchanged. Note that the rest of these nodes will not (and does not need to) be aware of this topology change.

Remark 3: The proposed approaches for distributed estimation of SCCs for both fixed and dynamic topology (subsection III.A and III.B) are applicable to directed graphs in general, i.e., not limited to spatially distributed networks.

Remark 4: In the special case where all the nodes are active, i.e., $\mathcal{V} = \{1, \ldots, \bar{n}\}$, each node can distributively verify the strong connectivity of the spatially distributed network if its estimation of SCC consists of all nodes, i.e., $\mathcal{C}_i = \mathcal{V}$ for all nodes $i \in \mathcal{V}$. This is due to the fact that the only SCC of a strongly connected graph is the graph itself. The readers are referred to Corollary 1 in [12] for more detailed discussion.

C. Distributed Election of Augmented Links

Finally, we present the distributed approach to elect augmented links to establish strong connectivity of the SDN. As described in Section II-B, we limit our discussion to the case where the graph is weakly connected. The following theorem is the main result of this paper.

Theorem 1: Given a weakly connected digraph $\mathcal{G}$, adding new links from each node i towards the set of nodes $\mathcal{N}_i \cap \mathcal{P}_i$ results in a strongly connected digraph.

Proof: We show the proof through condensation of $\mathcal{G}$ into a directed acyclic graph $\hat{\mathcal{G}}$ as described in [15]. Specifically, each node in $\hat{\mathcal{G}}$ represents a strongly connected component (SCC) in $\mathcal{G}$. Additionally, each edge in $\hat{\mathcal{G}}$ represents an edge in $\mathcal{G}$ from a node in one SCC to a node in the other, i.e., (j, i) for all $j \in \mathcal{N}_i \cap \mathcal{P}_i$ for each $i \in \mathcal{V}$. Due to this property, the resulting $\hat{\mathcal{G}}$ is also a weakly connected graph. Then, by the definition of a weakly connected graph, the addition of an edge (i, j) for every existing edge (j, i) in $\hat{\mathcal{G}}$ results in a strongly connected graph. These edges correspond to adding new links from each node i towards the set of nodes $\mathcal{N}_i \cap \mathcal{P}_i$. By the result in [15], given the set of edges that strongly connects $\hat{\mathcal{G}}$, then the corresponding set of edges also strongly connects the original graph $\mathcal{G}$. ■

Theorem 1 provides a prescription on a set of links that each node can pursue to establish the strong connectivity, namely, each node i can approach any links within $\mathcal{N}_i \cap \mathcal{P}_i$. A more detailed discussion on how to establish these links largely depends on the considered scenario and the corresponding constraints, as we illustrate in the subsequent section via two case studies.

Remark 5: Note that in the case where the graph $\mathcal{G}$ originally consists of disjoint subgraphs, there is no means for any node in one subgraph to exchange information with the other to allow the proposed distributed approach to work. One strategy is to rely on an advisory system that knows the information from both subgraphs to provide a direction on the elected links to connect the disjoint subgraphs. We leave the discussion on this issue for our future work.

IV. CASE STUDIES ON THE DISTRIBUTED LINK ADDITION ALGORITHM

In this section, we present the computation of the control input $\boldsymbol{u}_i$ and v_i that result in the establishment of the new links elected in the previous section while maintaining the existing edges. Particularly, we consider two case studies, namely the stationary nodes scenario where each node i can only control its communication radius, i.e., $\boldsymbol{u}_i = 0$, and the mobile nodes scenario where each node can manipulate its position as well.

In general, the resulting control framework is a hybrid system which combines the estimation of the SCCs and election of the augmented links in the discrete domain, and the controller for each node's position and communication range in the continuous domain. To ease the remaining discussion, we consider the discretized dynamics of each node i's position and communication radius in (1) as follows

$$\begin{aligned}\boldsymbol{q}_i[t+1] &= \boldsymbol{q}_i[t] + u_i[t]T_s,\\ r_i[t+1] &= r_i[t] + v_i[t]T_s.\end{aligned}$$

Without loss of generality, we assume the same time-sampling/update duration T_s for the node's discrete dynamics as well as the distributed estimation algorithm in Section III.

The proposed strategy for each scenario will be demonstrated via numerical simulations. The implementation is done via Python programming language, where we set the update time $T_s = 50$ms (to 20Hz update rate). The initial configuration of the node's position and communication range for the simulation are shown in Fig. 1a. The $\bar{n}$ is set as 10, where the active nodes are $\mathcal{V} = \{1, \ldots, \bar{n}\} \setminus \{5, 10\}$.

A. Stationary Nodes Scenario

In this scenario, the nodes are anchored to a fixed location, e.g., base transceiver stations for a communication network. In other words, $\boldsymbol{u}_i := \boldsymbol{0}_3$ for all $i \in \mathcal{V}$ and for all $t \geq 0$. Based on the result presented in Theorem 1, the control input for the communication range is then designed as

$$v_i[t] = \begin{cases} c, & \text{if } \mathcal{N}_i[t] \cap \mathcal{P}_i[t] \neq \emptyset \\ 0, & \text{otherwise.} \end{cases} \tag{10}$$

where c is the constant speed in increasing the communication range. Note that the estimation value of $\mathcal{P}_i$ is conducted via the distributed estimation of SCCs for dynamic topology presented in Section III-B. The value of c can be selected in accordance with the node's capability to increase its communication range. However, as c is applied continuously at each time-window between the two sampling times of the SCC estimation, it should be selected accordingly to avoid too large overshoot. The pseudocode of the distributed algorithm is given in Algorithm 1. Note that when the graph becomes strongly connected, the set $\mathcal{P}_i$ is empty for all $i \in \mathcal{V}$. Hence, from (10), we have $v_i[t] = 0$ for all nodes.

The result from the numerical simulation is shown in Figure 1b with $c = 0.1$ m/s. The final configuration of the adjustment of the communication range results in a strongly connected network. Figure 2a shows the cardinality of each node i's SCC estimation, i.e., $|\mathcal{C}_i|$ over time. Here, the strong connectivity of the graph is achieved when all the nodes' SCC contain every active node in the network, i.e., $\mathcal{C}_i = \mathcal{V}$ or simply $|\mathcal{C}_i| = |\mathcal{V}|$, as shown at the time 15s. It is worth noting that $\mathcal{V}$ (or $|\mathcal{V}|$) is unknown to the individual node. The proposed Algorithm 1 is simple, as it does not require the knowledge of other nodes' position or communication radius. Furthermore, the execution of Algorithm 1 relies solely on the incoming information of the estimation state, i.e., $\boldsymbol{x}_i$ and m_i, and the upper-bound of the active network, i.e., $\bar{n}$ to determine the duration of the estimation.

Algorithm 1 Distributed Link Addition for Stationary Nodes

Require: weakly connected SDN $\mathcal{G}[t_0 = 0]$, upper bound of network size $\bar{n}$, in-neighbor $\mathcal{N}_i[t_0]$
Ensure: establish strong connectivity of graph $\mathcal{G}[t_k]$
1: Initialize $\boldsymbol{x}_i[0]$ and $m_i[0]$ with $t = 0$
2: **while** true **do** {run infinitely to accommodate changes}
3: $t = t + 1$
4: Update $\boldsymbol{x}_i[t]$ and $m_i[t]$ via (3)–(4), and (7)–(9)
5: **if** $t \geq m_i[t] + 3\bar{n}$ **then**
6: estimate $\mathcal{C}_i[t]$ and $\mathcal{P}_i[t]$ via (5) and (6)
7: **else**
8: $\mathcal{C}_i[t] = \emptyset$, $\mathcal{P}_i[t] = \emptyset$
9: **end if**
10: Compute and apply $v_i[t]$ via (10)
11: **end while**

B. Mobile Nodes Scenario

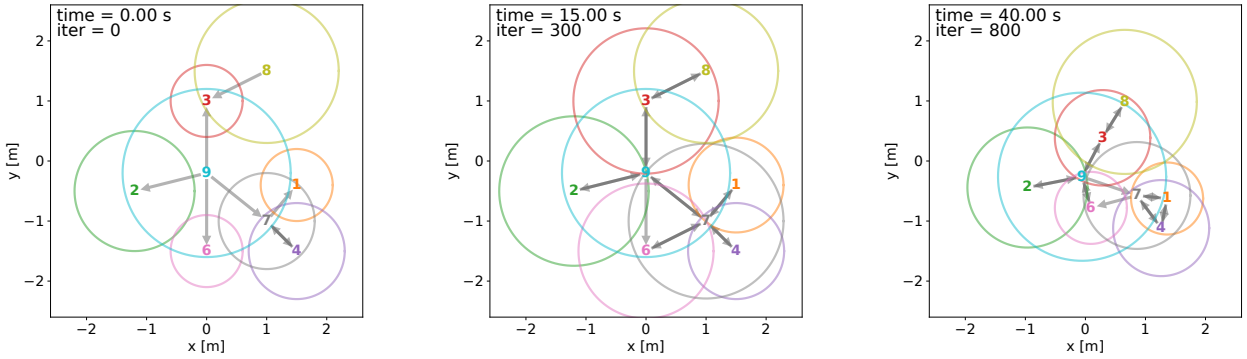
Next, we consider a scenario where each node can manipulate both their position and communication radius, e.g., mobile sensor network. Furthermore, we assume that each node has access to its in-neighbor's position and communication range, e.g., via continuous update from its in-neighbors. This information is required for the node to compute its velocity vector for reducing its distance with a targeted node, and thus establishing a new link.

The proposed distributed controller is given as follows. Note that the proposed strategy is designed under the premise that prioritizing movement over increasing communication range is more beneficial in terms of energy cost. First, each node $i \in \mathcal{V}$ selects its nearest in-neighbor which does not belong to its own SCC, if exists, denoted by l_i^* according to

$$l_i^* = \underset{j \in \mathcal{N}_i[t] \cap \mathcal{P}_i[t]}{\arg\min} \; d_{ij}. \tag{11}$$

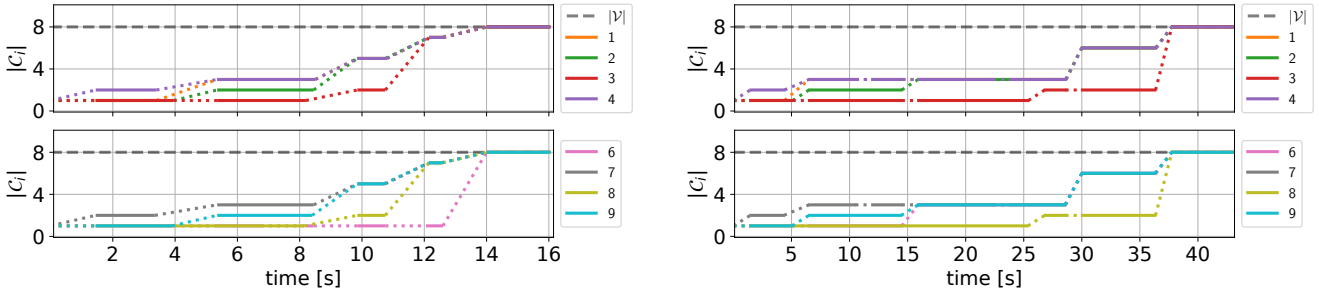
where $d_{ij} := \|\boldsymbol{q}_i - \boldsymbol{q}_j\|$ denotes the distance between a pair of nodes i and j. To establish new links, node i computes a velocity input to reduce the distance towards node l_i^* according to

$$\begin{aligned}\boldsymbol{u}_i^{\text{la}} &= \begin{cases} k^{\text{la}}(\boldsymbol{q}_{l^*} - \boldsymbol{q}_i), & \text{if } \|\boldsymbol{q}_i - \boldsymbol{q}_{l^*}\| > r_i - \epsilon^{\text{la}} \\ \boldsymbol{0}_3, & \text{otherwise.} \end{cases}\\ k^{\text{la}} &= \frac{v_{\max}^{\text{la}}}{d_{ij}} \left(1 - \exp(-\beta d_{ij})\right)\end{aligned} \tag{12}$$



(a) Initial configuration (b) Final configuration for stationary nodes (c) Final configuration for mobile nodes

Fig. 1: Numerical results for distributed link addition to establish strong connectivity for stationary and mobile nodes. The colored circles represent the communication range of the nodes. The tail and head of each gray arrow denotes node i and j for the established link (i, j). The video of the simulation can be viewed in `https://youtu.be/_9uSeqm2JZw`.



(a) Stationary nodes scenario (b) Mobile nodes scenario

Fig. 2: Time series data for the cardinality of each node's SCC. Dashed colored lines denote the time window without valid estimation of SCCs, due to changes in the network topology, where all nodes wait until valid estimation can be determined.

where k^{la} is the time-varying proportional gain with $v_{\max}^{\text{la}}$ defines the prescribed maximum magnitude of $\boldsymbol{u}_i^{\text{la}}$ and β defines the slope of the transition between 0 and the maximum value. As a result, the node i will keep moving towards l_i^*, despite any changes in the network, until the node l_i^* is within a distance $r_i - \epsilon^{\text{la}}$, with ϵ^{la} is a small margin to ensure that the established link is maintained for the next several iterations.

Next, to maintain the existing network connectivity during the motion of the nodes, we propose the following connectivity maintenance algorithm based on potential field

$$\boldsymbol{u}_i^{\text{cm}} = \sum_{j \in \mathcal{N}_i} k^{\text{cm}}(d_{ij})(\boldsymbol{q}_j - \boldsymbol{q}_i) \tag{13}$$

$$k^{\text{cm}} = \begin{cases} v_{\max}^{\text{cm}} \left[1 - \exp\left(- \left[\frac{d_{ij} - r_j^{\text{L}}}{r_j - r_j^{\text{L}}} \right]^2 \right) \right] & \text{if } r_j^{\text{L}} < d_{ij} < r_j \\ 0, & \text{otherwise.} \end{cases}$$

with $r_j^{\text{L}} = r_j - \epsilon^{\text{cm}}$ and for all node $j \in \mathcal{N}_i$. Here, the gain remains $k^{\text{cm}} > 0$ when the node i is within a certain band of ϵ^{cm} from the edge of its in-neighbor's communication range. This results in a velocity vector that pushes node i towards node j, and reduces the distance up to r_j^{L}, where its value is the highest on the edge with a magnitude of $v_{\max}^{\text{cm}}$ and decreases near 0 at r_j^{L}. Note that the robot i can freely move when $d_{ij} \leq r_j^{\text{L}}$.

The overall control input $\boldsymbol{u}_i[t]$ can then be computed as

$$\boldsymbol{u}_i[t] = \boldsymbol{u}_i^{\text{la}} + \boldsymbol{u}_i^{\text{cm}}. \tag{14}$$

Note that if the control input $\boldsymbol{u}_i^{\text{la}}$ is much larger than $\boldsymbol{u}_i^{\text{cm}}$, the node i will go beyond its in-neighbor's communication range, i.e., $d_{ij} \geq r_j$. This behavior is undesired as the existing edge (j, i) might be disconnected and as a result, node i can no longer receive information from node j. This situation can be avoided by setting $v_{\max}^{\text{la}} \ll v_{\max}^{\text{cm}}$.

Finally, the increase in the communication range is considered when it is not possible for a node to move closer to the targeted node l_i^*, e.g., due to constraint on connectivity maintenance. This can be detected by a small value of the resulting u_i, e.g., when its value is below a certain threshold φ. The control input for enlarging the communication range is designed as follows

$$v_i[t] = \begin{cases} c, & \text{if } l_i^* \neq \emptyset \text{ and } ||\boldsymbol{u}_i[t]|| < \varphi \\ 0, & \text{otherwise.} \end{cases} \tag{15}$$

The pseudocode of the distributed algorithm for mobile nodes is given in Algorithm 2.

The simulation result for the mobile node scenario is shown in Figure 1c. The parameters are chosen as $v_{\max}^{\text{la}} = 0.1$ m/s, $\beta = 1$, $v_{\max}^{\text{cm}} = 0.4$ m/s, $c = 0.1$ m/s, $\epsilon^{\text{la}} =$

Algorithm 2 Distributed Link Addition for Mobile Nodes

Require: weakly connected SDN $\mathcal{G}[t_0 = 0]$, upper bound of network size $\bar{n}$, in-neighbor $\mathcal{N}_i[t_0]$
Ensure: establish strong connectivity of graph $\mathcal{G}[t_k]$
1: Initialize $\boldsymbol{x}_i[0]$ and $m_i[0]$ with $t = 0$ and $l_i^* = \emptyset$
2: **while** true **do** {run infinitely to accommodate changes}
3: $t = t + 1$
4: Receive in-neighbors' information ($\boldsymbol{q}_j$, r_j, $\boldsymbol{x}_j$, and m_i for all $j \in \mathcal{N}_i[t]$) and compute $\mathcal{C}_i[t]$ and $\mathcal{P}_i[t]$ by following step 4–9 in Algorithm 1
5: **if** $l_i^* \neq \emptyset$ and $\|\boldsymbol{q}_i - \boldsymbol{q}_{l^*}\| \leq r_i - \epsilon^{\text{la}}$ **then**
6: Node l_i^* is within communication range, set $l_i^* = \emptyset$
7: **end if**
8: **if** $l_i^* = \emptyset$ and $\mathcal{N}_i[t] \cap \mathcal{P}_i[t] \neq \emptyset$ **then**
9: select l_i^* via (11)
10: **end if**
11: Compute link addition control input $\boldsymbol{u}_i^{\text{la}}$ via (12) if $l_i^* \neq \emptyset$, otherwise $\boldsymbol{u}_i^{\text{la}} = \boldsymbol{0}_n$
12: Compute connectivity maintenance control input $\boldsymbol{u}_i^{\text{cm}}$ via (13)
13: Compute and apply $\boldsymbol{u}_i[t]$ and $v_i[t]$ via (14)–(15)
14: **end while**

$\epsilon^{\text{cm}} = 0.1$ m, and $\varphi = 0.01$ m/s. It can be observed from Figure 1c that the final configuration of the nodes' position and communication range results in a strongly connected digraph, as verified by the time series of $|\mathcal{C}_i|$ in Figure 2b.

Remark 6: The selection of the simple model of the system is intentional due to the space limitation, namely to focus the discussion on demonstrating how the main result in Theorem 1 can be adopted to a given scenario. Note that in the real practice where each node is limited in its maximum velocity and communication range, the success of the algorithm will also rely on the initial configuration of all nodes as well as the proposed control approach. A more detailed discussion on this limitation and rigorous guarantee of the distributed control algorithm will be pursued in our future work.

Remark 7: In contrast to [5], [6], our proposed distributed link addition algorithms via Algorithm 1 and 2 do not impose the communication between each connected nodes to be bidirectional, as can be seen in the pairing nodes (9,6) in Figure 1b and the pairings (9,7) and (7,6) in Figure 1c.

Remark 8 (Privacy Preservation): Via the information retrieved via Algorithm 1 and 2, each node may directly retrieve their in-neighbors's information on position and communication range. However, each node's knowledge regarding the communication network is limited to the existence of path from other nodes to itself, as well as the information number of other nodes (state $\boldsymbol{x}_i$). Hence, Algorithm 1 and 2 do not reveal the overall network topology.

V. CONCLUSION

This paper proposes a distributed algorithm for each node in a spatially distributed network that allows estimation of a strongly connected component (SCC) each node belongs to and then elects a set of nodes that collectively establish strongly connectivity. Then, distributed controllers are proposed for two scenarios of ensuring strong connectivity in wireless network comprises static and mobile nodes. The proposed framework is demonstrated via numerical simulations. The proposed distributed strategies provide the solutions without requiring knowledge of the overall network topology, and further preserve the privacy in terms of the overall network's topology. Future work will aim towards extending the framework without the need to maintain the existing links, as well as the consideration of maximum velocity and communication range.

REFERENCES

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Computer networks*, vol. 38, no. 4, pp. 393–422, 2002.
[2] Q. Wu, Y. Zeng, and R. Zhang, "Joint trajectory and communication design for multi-uav enabled wireless networks," *IEEE Transactions on Wireless Communications*, vol. 17, no. 3, pp. 2109–2121, 2018.
[3] L. E. Parker, D. Rus, and G. S. Sukhatme, "Multiple mobile robot systems," *Springer handbook of robotics*, pp. 1335–1384, 2016.
[4] F. Bullo, *Lectures on Network Systems*, 1.6 ed. Kindle Direct Publishing, 2022. [Online]. Available: https://fbullo.github.io/lns
[5] V. Rodoplu and T. Meng, "Minimum energy mobile wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 17, no. 8, pp. 1333–1344, Aug. 1999.
[6] R. Wattenhofer, L. Li, P. Bahl, and Y.-M. Wang, "Distributed topology control for power efficient operation in multihop wireless ad hoc networks," in *Proceedings IEEE INFOCOM 2001. Conference on Computer Communications. Twentieth Annual Joint Conference of the IEEE Computer and Communications Society (Cat. No.01CH37213)*, vol. 3, Apr. 2001, pp. 1388–1397 vol.3.
[7] L. Sabattini, C. Secchi, and N. Chopra, "Decentralized estimation and control for preserving the strong connectivity of directed graphs," *IEEE Transactions on Cybernetics*, vol. 45, no. 10, pp. 2273–2286, 2015.
[8] B. Capelli, H. Fouad, G. Beltrame, and L. Sabattini, "Decentralized Connectivity Maintenance with Time Delays using Control Barrier Functions," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, May 2021, pp. 1586–1592.
[9] H. A. Poonawala and M. W. Spong, "Preserving Strong Connectivity in Directed Proximity Graphs," *IEEE Transactions on Automatic Control*, vol. 62, no. 9, pp. 4392–4404, Sep. 2017.
[10] A. Gusrialdi, "Connectivity-preserving distributed algorithms for removing links in directed networks," *Network Science*, vol. 10, no. 3, pp. 215–233, 2022.
[11] M. W. S. Atman and A. Gusrialdi, "Distributed Algorithms for Verifying and Ensuring Strong Connectivity of Directed Networks," in *2021 60th IEEE Conference on Decision and Control (CDC)*, Dec. 2021, pp. 4798–4803.
[12] ——, "Finite-Time Distributed Algorithms for Verifying and Ensuring Strong Connectivity of Directed Networks," *IEEE Transactions on Network Science and Engineering*, vol. 9, no. 6, pp. 4379–4392, Nov. 2022.
[13] T. Charalambous, M. G. Rabbat, M. Johansson, and C. N. Hadjicostis, "Distributed finite-time computation of digraph parameters: Left-eigenvector, out-degree and spectrum," *IEEE Transactions on Control of Network Systems*, vol. 3, no. 2, pp. 137–148, Jun. 2016.
[14] N. Emirov, C. Cheng, Q. Sun, and Z. Qu, "Distributed algorithms to determine eigenvectors of matrices on spatially distributed networks," *Signal Processing*, vol. 196, p. 108530, Jul. 2022.
[15] K. P. Eswaran and R. E. Tarjan, "Augmentation problems," *SIAM Journal on Computing*, vol. 5, no. 4, pp. 653–665, Dec. 1976.