2020 IEEE International Conference on Robotics and Automation (ICRA)
31 May - 31 August, 2020. Paris, France

# Behavior Mixing with Minimum Global and Subgroup Connectivity Maintenance for Large-Scale Multi-Robot Systems

Wenhao Luo, Sha Yi, and Katia Sycara

*Abstract*— **In many cases the multi-robot systems are desired to execute simultaneously multiple behaviors with different controllers, and sequences of behaviors in real time, which we call *behavior mixing*. Behavior mixing is accomplished when different subgroups of the overall robot team change their controllers to collectively achieve given tasks while maintaining connectivity within and across subgroups in one connected communication graph. In this paper, we present a provably minimum connectivity maintenance framework to ensure the subgroups and overall robot team stay connected at all times while providing the highest freedom for behavior mixing. In particular, we propose a real-time distributed Minimum Connectivity Constraint Spanning Tree (MCCST) algorithm to select the minimum inter-robot connectivity constraints preserving subgroup and global connectivity that are *least likely to be violated* by the original controllers. With the employed safety and connectivity barrier certificates for the activated connectivity constraints and collision avoidance, the behavior mixing controllers are thus minimally modified from the original controllers. We demonstrate the effectiveness and scalability of our approach via simulations of up to 100 robots with multiple behaviors.**

## I. Introduction

The ability of collaboration in multi-robot systems often relies on the local information sharing and interaction among networked robot members through a connected communication graph, e.g. flocking [1], [2], formation control [3] and leader selection [4], [5]. As robots are often assumed to interact in a proximity-limited manner due to limited communication range, it is necessary to consider connectivity maintenance that ensures robots stay connected as one component. This is often referred as maintaining *global connectivity* [6]–[14] and achieved by constraining inter-robot distance while executing original tasks. In many situations, however, it may be more appropriate and efficient to have the multi-robot systems *simultaneously* performing multiple behaviors in different subgroups while remaining connected. For example, having a robot team split into multiple operating subgroups flocking to multiple task areas at the same time. As subgroups may be formed based on the particular combinations of robots with heterogeneous capabilities, when the robot team spreads out over multiple widely separated task areas, robots in the same subgroup for a designated task area are expected to stay *locally connected by themselves* as one coherent component for efficient local collaboration. Global connectivity is still required to allow for global coordination among different subgroups, e.g. redistribution of robots due to dynamic task reallocation over time. Thus it is also necessary to ensure connectivity within each subgroup and across subgroups as well as global connectivity. We call this ability of multi-robot systems to accommodate different behaviors simultaneously within a single connected robot team while maintaining safety (collision avoidance with other robots and possibly obstacles) and within and across subgroup connectivity *Behavior Mixing*.

*To the best of our knowledge*, there is no existing work on connectivity maintenance that can ensure both global and subgroup connectivity for behavior mixing. Most of the connectivity control methods use either 1) *local methods* that seek to preserve the initial connectivity graph topology over time [2], [15], [16], or 2) *global methods* that aim to preserve the global algebraic connectivity of the communication graph by deriving secondary connectivity controllers for keeping the second smallest eigenvalue of the graph Lapacian positive at all times [6]–[9]. Recent work [17]–[22] have explored the idea of redeploying a certain number of robots to act as communication relays, while aiming to allow the rest of the robots to perform their original tasks. These approaches often has no guarantee that the perturbation from the connectivity controllers is minimum over the original robot task-related controllers. On the other hand, recent advances in permissive control barrier functions [23] have been widely applied to multi-robot systems with guarantee on the forward invariance of desired sets, e.g. robots staying collision-free and connected with predefined communication graph at all times [3], [24], while minimizing revisions to the original robot controllers. However, these capabilities are achieved by predefining the connectivity constraints so as to preserve *fixed predefined* communication topology [3], [24]. It would be desired for the robots to compute in real-time the *optimal* communication topology/connectivity constraints to preserve that provides provably highest freedom for executing the original robot tasks with required network connectivity.

To that end, in this paper we develop a generalized behavior mixing framework with minimum global and subgroup connectivity maintenance. Such framework is based on a bilevel optimization process that 1) incorporates a novel distributed *Minimum Connectivity Constraint Spanning Tree (MCCST)* to compute real-time *minimum connectivity constraints*, and 2) minimizes the revision to the original controllers subject to our invoked connectivity constraints and collision avoidance constraints formulated by the barrier certificates with control barrier functions (CBF) [23], [24].

This work was funded by the DARPA Cooperative Agreement No.: HR00111920012, AFOSR awards FA9550-18-1-0097, FA9550-18-1-0251 and FA9550-15-1-0442.

The authors are with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213, USA. Email: {wenhao, shayi, katia}@cs.cmu.edu.

In particular, MCCST computes the provably optimal set of communication links for the robots to maintain, which (a) has minimum number of links, and (b) invokes the connectivity constraints for global and subgroup connectivity least likely to be violated by the original controllers. Minimum connectivity maintenance is thus achieved by minimally modifying the original controllers to preserve these dynamic *least constraining* communication links and avoid collisions.

Our paper presents the following contributions: (1) a generalized bilevel optimization based behavior mixing framework to enable *simultaneous execution of different behaviors and sequences of behaviors within a single robot team,* while ensuring global and subgroup connectivity and collision avoidance; (2) a novel distributed MCCST method with quantified relationship between original task-related controllers and connectivity constraints to efficiently select *real-time minimum behavior mixing connectivity constraints with provably optimality guarantee*, (3) *computationally efficient* construction of MCCST that is *scalable* to large number of robots and suitable for real-time computation to accommodate dynamic changes in the environment.

## II. Behavior Mixing

Consider a robotic team $\mathcal{S}$ consisting of $N$ mobile robots in a planar space, with the position and single integrator dynamics of each robot $i \in \{1, \ldots, N\}$ denoted by $x_i \in \mathbb{R}^2$ and $\dot{x}_i = u_i \in \mathbb{R}^2$ respectively. Each robot can connect and communicate directly with other robots within its spatial proximity. The communication graph of the robotic team is defined as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where each node $v \in \mathcal{V}$ represents a robot. If the spatial distance between robots $v_i, v_j \in \mathcal{V}$ is less or equal to the communication radius $R_c$ (i.e. $\|x_i - x_j\| \leq R_c$), then we assume the two can communicate and edge $(v_i, v_j) \in \mathcal{E}$ is undirected (i.e. $(v_i, v_j) \in \mathcal{E} \Leftrightarrow (v_j, v_i) \in \mathcal{E}$).

### A. Safety and Connectivity Constraints using Barrier Certificates

Consider the joint robot states $\mathbf{x} = \{x_1, \ldots, x_N\} \in \mathbb{R}^{2N}$ and define the minimum inter-robot safe distance as $R_s$, for any pair-wise inter-robot collision avoidance constraint between robots $i$ and $j$. We have the following condition defining the safe set of $\mathbf{x}$.

$$\begin{aligned} h^s_{i,j}(\mathbf{x}) &= \|x_i - x_j\|^2 - R_s^2, \qquad \forall i > j \\ \mathcal{H}^s_{i,j} &= \{\mathbf{x} \in \mathbb{R}^{2N} : h^s_{i,j}(\mathbf{x}) \geq 0\} \end{aligned} \tag{1}$$

The set of $\mathcal{H}^s_{i,j}$ indicates the safety set from which robot $i$ and $j$ will never collide. For the entire robotic team, the safety set can be composed as follows.

$$\mathcal{H}^s = \bigcap_{\{v_i, v_j \in \mathcal{V} : i > j\}} \mathcal{H}^s_{i,j} \tag{2}$$

[25] proposed the safety barrier certificates $\mathcal{B}^s(\mathbf{x})$ using control barrier functions $h^s_{i,j}(\cdot)$ that map the constrained safety set (2) of $\mathbf{x}$ to the admissible joint control space $\mathbf{u} \in \mathbb{R}^{2N}$ for ensuring $h^s_{i,j}(\cdot) \geq 0$ at all time. The result is summarized as follows.

$$\mathcal{B}^s(\mathbf{x}) = \{\mathbf{u} \in \mathbb{R}^{2N} : \dot{h}^s_{i,j}(\mathbf{x}) + \gamma h^s_{i,j}(\mathbf{x}) \geq 0, \forall i > j\} \tag{3}$$

where $\gamma$ is a user-defined parameter to confine the available sets. It is proven in [25] that the forward invariance of the safety set $\mathcal{H}^s$ is ensured as long as the joint control input $\mathbf{u}$ stays in set $\mathcal{B}^s(\mathbf{x})$. In other words, the robots will always stay safe if they are initially inter-robot collision free and the control input lies in the set $\mathcal{B}^s(\mathbf{x})$. The constrained control space in (3) corresponds to a class of linear constraints over pair-wise control inputs $u_i$ and $u_j$.

Likewise, if the connectivity constraint is enforced between pair-wise robots $i$ and $j$ to ensure inter-robot distance not larger than communication range $R_c$, we have

$$\begin{aligned} h^c_{i,j}(\mathbf{x}) &= R_c^2 - \|x_i - x_j\|^2 \\ \mathcal{H}^c_{i,j} &= \{\mathbf{x} \in \mathbb{R}^{2N} : h^c_{i,j}(\mathbf{x}) \geq 0\} \end{aligned} \tag{4}$$

The set of $\mathcal{H}^c_{i,j}$ indicates the feasible set on $\mathbf{x}$ from which robot $i$ and $j$ will never lose connectivity. Consider any connectivity spanning graph $\mathcal{G}^c = (\mathcal{V}, \mathcal{E}^c) \subseteq \mathcal{G}$ to enforce, the corresponding constrained set can be composed as follows.

$$\mathcal{H}^c(\mathcal{G}^c) = \bigcap_{\{v_i, v_j \in \mathcal{V} : (v_i, v_j) \in \mathcal{E}^c\}} \mathcal{H}^c_{i,j} \tag{5}$$

Similar to the safety barrier certificates in (3), the connectivity barrier certificates [24] are defined as follows indicating another class of linear constraints over pair-wise control inputs $u_i$ and $u_j$ for $(v_i, v_j) \in \mathcal{E}^c$ at any time point $t$.

$$\mathcal{B}^c(\mathbf{x}, \mathcal{G}^c) = \{\mathbf{u} \in \mathbb{R}^{2N} : \dot{h}^c_{i,j}(\mathbf{x}) + \gamma h^c_{i,j}(\mathbf{x}) \geq 0, \forall (v_i, v_j) \in \mathcal{E}^c\} \tag{6}$$

### B. Bilevel Optimization for Behavior Mixing

In behavior mixing, we assume the robotic team is tasked with $M$ simultaneous behaviors and has been partitioned into $M$ sub-groups $\mathcal{S} = \{\mathcal{S}_1, \ldots, \mathcal{S}_M\}$, with each robot $i$ already assigned to a sub-group $\mathcal{S}_m$ and with original task-related controller $u_i = \hat{u}_i$. To ensure successful behavior mixing, the global connectivity graph $\mathcal{G}$ and the induced subgroup connectivity graph $\mathcal{G}_m = \mathcal{G}[\mathcal{V}_m] \subseteq \mathcal{G}$ where $\mathcal{V}_m \subseteq \mathcal{V}$ containing robots within the same sub-group $\mathcal{S}_m$ for all $m = 1, \ldots, M$ should be connected at all time. We assume these connectivity constraints are satisfied initially. With the defined forms of safety and connectivity constraints in (3) and (6), we formally define the *behavior mixing* problem as a bilevel optimization process at each time step as follows.

$$\mathbf{u}^* = \arg\min_{\mathcal{G}^c, \mathbf{u}} \sum_{i=1}^{N} \|u_i - \hat{u}_i\|^2 \tag{7}$$

$$\text{s.t.} \quad \mathcal{G}^c = (\mathcal{V}^c, \mathcal{E}^c) \subseteq \mathcal{G} \quad \text{is connected}$$

$$\mathcal{G}_m = \mathcal{G}^c[\mathcal{V}_m] \quad \text{is connected} \quad \forall m = 1, \ldots, M \tag{8}$$

$$\mathbf{u} \in \mathcal{B}^s(\mathbf{x}) \bigcap \mathcal{B}^c(\mathbf{x}, \mathcal{G}^c), \quad \|u_i\| \leq \alpha_i, \forall i = 1, \ldots, N \tag{9}$$

This bilevel optimization problem can be solved by two-steps: find 1) the optimal connectivity spanning graph $\mathcal{G}^{c*} \subseteq \mathcal{G}$ to preserve, and 2) the one-step control inputs $\mathbf{u}^* \in \mathbb{R}^{2N}$ bounded by maximum velocities $\{\alpha_i\}$ and minimally deviated from $\hat{u}_i$ subject to constraints in (9) with $\mathcal{G}^c = \mathcal{G}^{c*}$.

## III. Behavior Mixing using Distributed Selection of Minimum Connectivity Constraints

### A. Minimum Connectivity Constraint Spanning Tree (MCCST)

First we consider the sub-problem of selecting optimal connectivity spanning graph $\mathcal{G}^{c*} \subseteq \mathcal{G}$ in Eq. (7) that introduces minimum connectivity constraints. As each edge $(v_i, v_j) \in \mathcal{E}^c$ in a candidate graph $\mathcal{G}^c$ enforces one constraint between robot $i, j$ in (6), the graph $\mathcal{G}^{c*}$ whose edges define the minimum connectivity constraints must exist among the set of all spanning trees $\mathcal{T}$ of $\mathcal{G}$ that have the minimum number of edges (i.e. $N-1$) for $\mathcal{G}^{c*}$ to stay connected.

Hence, the problem boils down to find the optimal spanning tree $\mathcal{G}^{c*} = \mathcal{T}^{c*} \in \mathcal{T}$ of $\mathcal{G}$ whose edges invoke the minimum connectivity constraints in the form of (6) over the robots' controllers. To quantify the strength of connectivity constraint by an edge $(v_i, v_j) \in \mathcal{E}$, we introduce the weight assignment defined as follows.

$$w_{i,j} = \dot{h}^c_{i,j}(\mathbf{x}, \hat{u}_i, \hat{u}_j) + \gamma h^c_{i,j}(\mathbf{x}), \forall (v_i, v_j) \in \mathcal{E} \tag{10}$$

Compared to the connectivity constraint in (6), $w_{i,j}$ indicates the violation of the pair-wise connectivity constraint between the two robots under the original controllers $\hat{u}_i, \hat{u}_j$, with the higher value of $w_{i,j}$ the less violated the connectivity constraint is. This quantifies how likely the existing connectivity link is going to break if no revision made to the controller. It is desired to preserve those links with larger $w_{i,j}$ implying smaller revision needed for the controllers to keep the links connected. With that, each candidate spanning tree $\mathcal{T}^c \in \mathcal{G}$ is redefined as a weighted spanning tree $\mathcal{T}^c_w = (\mathcal{V}, \mathcal{E}^T, \mathcal{W}^T)$ with $\mathcal{W}^T = \{-w_{i,j}\}$. Hence the optimal connectivity graph $\mathcal{G}^{c*}$ with constraints in (8) can be obtained as follows.

$$\begin{aligned} \mathcal{G}^{c*} &= \underset{\mathcal{T}^c_w \in \mathcal{T}}{\arg\max} \sum_{(v_i,v_j)\in\mathcal{E}^T} w_{i,j} = \underset{\mathcal{T}^c_w \in \mathcal{T}}{\arg\min} \sum_{(v_i,v_j)\in\mathcal{E}^T} -w_{i,j} \\ \text{s.t.} \quad & \mathcal{T}_m = \mathcal{T}^c_w[\mathcal{V}_m] \quad \text{is connected} \quad \forall m = 1, \dots, M \end{aligned} \tag{11}$$

The optimal solution of (11) is the Minimal Spanning Tree (MST) weighted by $\{-w_{i,j}\}$ and constrained by sub-group connectivity requirements. We propose to define another class of spanning trees as follows and relate its unconstrained MST to the solution of the constrained MST in (11).

**Definition 1.** *Given a connectivity graph $\mathcal{G}$ and for all edges $(v_i, v_j) \in \mathcal{E}$ on $\mathcal{G}$, redefine their weights by the following.*

$$w'_{i,j} = \begin{cases} \lambda \cdot w_{i,j}, & \textit{if} \quad v_i \textit{ and } v_j \textit{ are in the same sub-group} \\ w_{i,j}, & \textit{if} \quad v_i \textit{ and } v_j \textit{ are in different sub-groups} \end{cases} \tag{12}$$

*where $\lambda \in \{\lambda \gg 1 : \lambda \cdot w_{i,j} \gg w_{i',j'}, \forall v_i, v'_i, v_j, v'_j \in \mathcal{V}\}$ is a unique user-defined constant for the entire graph $\mathcal{G}$. The weight-modified graph is denoted as $\mathcal{G}'$. Then we call the redefined spanning tree $\mathcal{T}^{c'}_w = (\mathcal{V}, \mathcal{E}^T, \mathcal{W}^{T'})$ as the Connectivity Constraint Spanning Tree (CCST).*

The Definition 1 introduces a new class of spanning trees (CCST) $\mathcal{T}^{c'}_w$ equivalent to the original spanning trees $\mathcal{T}^c_w$ with inflated weights over the edges connecting robots in the same sub-group. In particular, the designed parameter $\lambda$ in (12) ensures that after inflation the new weights $-w'_{i,j}$ over edges connecting different subgroups are always larger than any edges within all the subgroups for $\mathcal{T}^{c'}_w$. As we will prove by the following Lemma 3 and Theorem 4, this guarantees that the computed MST $\mathcal{T}^{c'}_w$ becomes the solution of constrained MST $\mathcal{T}^c_w$ in (11), namely, the MST $\mathcal{T}^{c'}_w$ contains the MST of each subgroup as well, ensuring that the subgroups are also connected in an optimal way. We review some useful definitions in graph theory [26]:

- *fragment*: a subtree of Minimum Spanning Tree;
- *outgoing edge*: a edge of a fragment if one adjacent node is in the fragment and the other is not.

The first definition describes that a connected set of nodes and edges of the MST is called a fragment. By this definition, a single node is also a fragment by itself. In the following discussion, we focus on *minimum-weight outgoing edge (MWOE)*, which is the edge with minimum weight among all outgoing edge of a fragment.

**Lemma 2.** *Let $e_{min}$ be a minimum-weight outgoing edge (MWOE) of a fragment. Connecting $e_{min}$ and its adjacent node in a different fragment yields another fragment in MST.*

The proof of Lemma 2 can be found in both [26] and [27].

With Lemma 2, the process of constructing MST is as follows [26]:

- Each node starts as a fragment by itself
- Each fragment iteratively connects with MWOE fragment

This process will result in the MST of the given graph.

**Lemma 3.** *By following the process above on $\mathcal{G}'$ in Definition 1, all nodes within the same sub-group will form a MST fragment before connecting to other sub-group.*

*Proof.* We prove by contradiction. Suppose the node $v_i$ from sub-group graph $\mathcal{G}'_i$ connects with node $v_j$ first, which belongs to sub-group graph $\mathcal{G}'_j$, $i \neq j$. From the MST construction process we know that, at each iteration, the edge added is the minimum-weight outgoing edge of the connecting fragment. In this case, the weight $w_{i,j}$ of the edge between $v_i$ and $v_j$ is the minimum of all outgoing edges of $v_i$. Let $v_{i'} \in \mathcal{G}'_i$ where there exists an outgoing edge between $v_i$ and $v_{i'}$, then we know that the weight $w'_{i,j} < w'_{i,i'}$. This contradicts with the property of $\mathcal{G}'$ in Equation 12. □

**Theorem 4.** *Given the redefined Connectivity Constraint Spanning Tree (CCST) $\mathcal{T}^{c'}_w = (\mathcal{V}, \mathcal{E}^T, \mathcal{W}^{T'})$ in Definition 1 and denote minimum weight CCST as $\bar{\mathcal{T}}^{c'}_w = \arg\min_{\mathcal{T}^{c'}_w \in \mathcal{T}} \sum_{(v_i,v_j)\in\mathcal{E}^T} -w'_{i,j}$, we have: $\bar{\mathcal{T}}^{c'}_w = \mathcal{G}^{c*}$ in Equation (11). Namely, the Minimum Spanning Tree $\bar{\mathcal{T}}^{c'}_w$ of $\mathcal{G}'$ is the optimal solution of $\mathcal{G}^{c*}$ in (11) and we call the graph $\bar{\mathcal{T}}^{c'}_w$ as Minimum Connectivity Constraint Spanning Tree (MCCST) of the original connected graph $\mathcal{G}$.*

*Proof.* From Lemma 3, edges *between* sub-groups will be connected only when edges *within* each sub-groups are connected. By definition, the MST of graph $\mathcal{G}'_i$ within subgroup $\mathcal{S}_i$ is optimal with minimum total weight, which means

$$\begin{aligned} \bar{\mathcal{T}}^{c'}_w(i) &= \underset{\mathcal{T}^{c'}_w(i) \in \mathcal{T}(i)}{\arg\min} \sum_{(v_i,v_j)\in\mathcal{E}^T(i)} -w'_{i,j} \\ &= \underset{\mathcal{T}^{c'}_w(i) \in \mathcal{T}(i)}{\arg\min} \lambda \cdot \sum_{(v_i,v_j)\in\mathcal{E}^T(i)} -w_{i,j} \\ &= \underset{\mathcal{T}^{c'}_w(i) \in \mathcal{T}(i)}{\arg\min} \sum_{(v_i,v_j)\in\mathcal{E}^T(i)} -w_{i,j} \end{aligned} \tag{13}$$

The equality holds since $\lambda > 0$. Then we consider $v_i$ and $v_j$ in different subgroups, i.e. $\mathcal{S}(v_i) \neq \mathcal{S}(v_j)$, while $(v_i, v_j)$ is the edge in spanning tree edges $\mathcal{E}^{T(i)}$ connecting two subgroups. Then for the next step, connecting the minimum-weighted outgoing edge between different sub-groups, yields

$$\begin{aligned} \bar{\mathcal{T}}_w^{c'} &= \underset{\mathcal{T}_w^{c'} \in \mathcal{T}}{\arg\min} \sum_{(v_i,v_j)\in\mathcal{E}^{T(i)}} -w'_{i,j}, \quad \mathcal{S}(v_i) \neq \mathcal{S}(v_j) \\ &= \underset{\mathcal{T}_w^{c'} \in \mathcal{T}}{\arg\min} \sum_{(v_i,v_j)\in\mathcal{E}^{T}} -w_{i,j} \end{aligned} \tag{14}$$

With the same form as in (11), this concludes the proof. □

In this way, we relax the constrained MST optimization problem in (11) into unconstrained MST problem with the same optimality guarantee. The connectivity constraints from the obtained MCCST $\bar{\mathcal{T}}_w^{c'}$ are thus minimally violated by the current task-related controllers, implying the least restriction due to global and subgroup connectivity requirements. Such MCCST $\bar{\mathcal{T}}_w^{c'}$ therefore specifies the optimal connectivity graph $\mathcal{G}^{c*} \subseteq \mathcal{G}$ to enforce for behavior mixing in (8). Next, we will present a distributed method for computing MCCST.

### B. Construction of Distributed Minimum Connectivity Constraint Spanning Tree (MCCST)

Here we propose a distributed construction of MCCST of $\mathcal{G}$. For our problem setting, the topology and weights could change over time, thus a time-optimal real-time algorithm is needed. We develop our algorithm based on the work from [26]–[28], but reduce the computation time while sacrificing message optimality. Different from most of the network algorithms such as [27], [28], our algorithm does not require synchronization, which also reduces the total time. Note that MST is unique for a graph with unique edge weights. Therefore the result is the same from centralized and decentralized construction.

A detailed description of the algorithm is as follows:

*1) Overview:* Given a graph $\mathcal{G}' = \{\mathcal{V}, \mathcal{E}\}$ with weights defined in Definition 1 where $|\mathcal{V}| = N$ is the number of robots, the initial state of the system is a singleton graph where each vertex is an individual isolated node without any outgoing edge, and each node is given a distinct id. This gives $N$ fragments and each consists of one vertex. Then each fragment finds the *minimum-weight outgoing edge (MWOE)* and connect with neighboring fragments. Iteratively, the forest of fragments will join as a spanning tree connecting all vertices of the graph, resulting as the MCCST.

As shown in Algorithm 1, each robot takes an input of neighboring edge weights and connectivity information, then outputs the computed MCCST edge list. The incoming message is processed according to whether the node is being initialized or not. The process will reset when there is no new message in the message pool, which implies all the fragments finish updating within themselves and new MWOE need to be connected and a new round begins.

*2) Initial Round:* In the initial round, as shown in Algorithm 2, each fragment initially only contains one vertex. Each node directly connect with the neighbor with MWOE. However, to keep information within a fragment consistent and avoid additional computation, the node with the smallest id is selected as fragment leader. Information keeps updating within the fragment until every node has the same adjacency matrix of its fragment tree.

**Algorithm 1** Distributed MCCST Construction

```
Input: $a$: adjacency edge weight list of the original weighted graph
Output: edge list of MCCST
 1: function CONSTRUCTDISTRIBUTEDMCCST($A$)
 2:     $A \leftarrow$ empty adjacency matrix
 3:     $A \leftarrow$ updated from input $a$
 4:     while $msg \leftarrow$ getNewMessage($msg\_pool$) do
 5:         if not initialized then
 6:             $A \leftarrow$ initialRound($msg$, $A$)
 7:         else
 8:             $A \leftarrow$ processRound($msg$, $A$)
 9:         if isConnected($A$) then
10:             return getEdgeList($A$)
11:     if isEmpty($msg\_pool$) then
12:         resetRound()
```

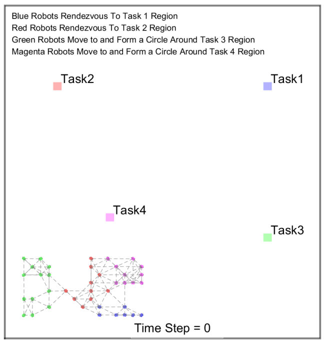
**Algorithm 2** Initial Round of MST Construction

```
Input: $msg$: incoming messages, $A$: current adjacency matrix
Output: $A$: updated adjacency matrix
 1: function INITIALROUND($msg$, $A$)
 2:     connect with neighbor with $MWOE$
 3:     $leader\_id \leftarrow$ min($self\_id$, $neighbor\_id$)
 4:     $A \leftarrow$ update with $msg$
 5:     if no new information in $msg$ then
 6:         finish initial round
 7:     send $init$ message to $msg\_pool$
 8:     return $A$
```

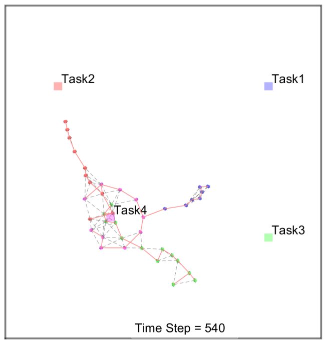
**Algorithm 3** Processing Round of MCCST Construction

```
Input: $msg$: incoming messages, $A$: current adjacency matrix
Output: $A$: updated adjacency matrix
 1: function PROCESSROUND($msg$, $A$)
 2:     if $leader\_id$ is $self\_id$ then
 3:         $MWOE\_cache\_list \leftarrow$ wait($fragment\_node$)
 4:         $MWOE \leftarrow$ min($MWOE\_cache\_list$)
 5:         if all $fragment\_node$ reported then
 6:             inform the one with $MWOE$ to connect
 7:     if not reported to leader then
 8:         if no MWOE info then
 9:             $msg\_pool[MWOE] \leftarrow get\_info$ message
10:         if get information from MWOE neighbor then
11:             $msg\_pool[leader\_id] \leftarrow report$ message
12:     $A \leftarrow$ update with $msg$
13:     return $A$
```

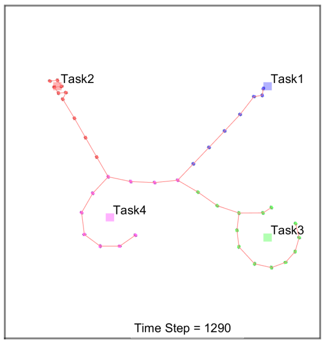
*3) Processing Round:* At each processing round, the fragment leader will determine the minimum-weight outgoing edge (MWOE) in its fragment after receiving all MWOE information from each fragment node (including itself). Since each node in the fragment only has the local knowledge within its own fragment, it will ask the MWOE neighbor for their fragment information, i.e. adjacency matrix, leader id. Whenever a node receives a request to give information, it will reply accordingly. Once each node receives information from MWOE neighbor, it will report to the fragment leader. All connect requests will be accepted and this, by lemma 2, always yields a fragment. When a new connection is made, the two fragments will combine their information and update


Blue Robots Rendezvous To Task 1 Region
Red Robots Rendezvous To Task 2 Region
Green Robots Move to and Form a Circle Around Task 3 Region
Magenta Robots Move to and Form a Circle Around Task 4 Region
Task2
Task1
Task4
Task3
Time Step = 0

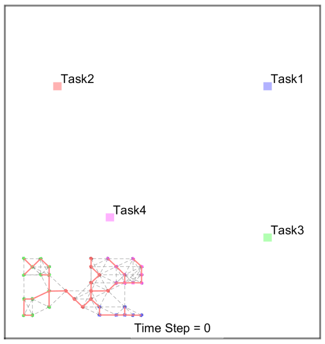
(a) Time Step = 0



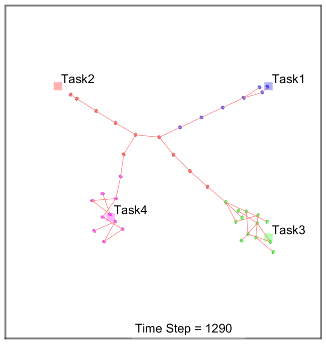
(b) Time Step = 540 (MCCST)



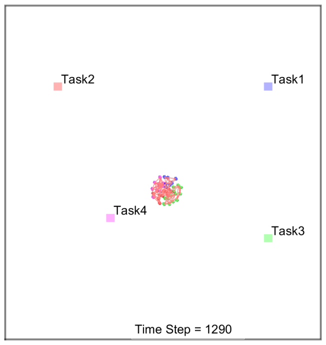
(c) Time Step = 1290 (MCCST, Converged)



(d) Time Step = 0

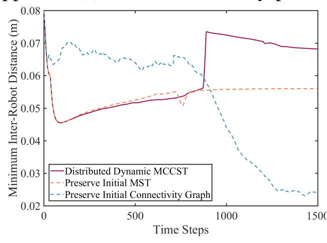


(e) Time Step = 1290 (Fixed Initial MST, Converged)

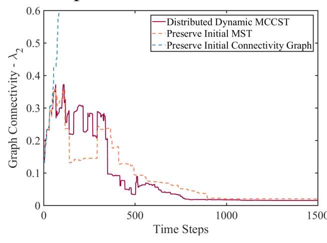


(f) Time Step = 1290 (Fixed Initial Connectivity, Converged)

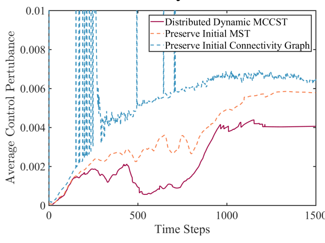
Fig. 1: Simulation example of 40 robots tasked to four different places with behaviour mixing: blue robots and red robots rendezvous to regions of blue task 1 and red task 2 respectively, while green robots and magenta robots move to region of green task 3 and magenta task 4 and form a circle around the regions. Grey dashed lines in (a),(b),(d) denote current connectivity edges and red lines in (a)-(f) denote current active connectivity graph invoking pair-wise connectivity constraints. Compared to fixed inter-robot connectivity constraints from initial MST (e) and initial connectivity graph (f), our proposed MCCST approach (c) enables minimally perturbed task performance due to invoked minimum connectivity constraints on the robots.



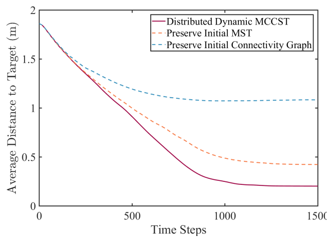
(a) Minimum inter-robot distance



(b) Algebraic connectivity



(c) Average control perturbation



(d) Average distance to target region

Fig. 2: Performance comparison of simulation example in Figure 1 w.r.t. different metrics: (a) Minimum inter-robot distance (safety distance is 0.02m), (b) Algebraic connectivity evaluated by second smallest eigenvalue of mutli-robot laplacian matrix. Positive meaning connectivity ensured, (c) Control perturbation computed by $\frac{1}{N}\sum_{i=1}^{N}\left\|u_i^* - \hat{u}_i\right\|^2$, (d) Average distance between robots to tasked region (the smaller the better).

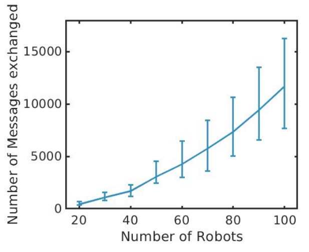
all the nodes within the fragment with the new information. Iteratively, the construction process will end when every node receives the same updated adjacency matrix representing the MST of the graph. Since only the leader of each fragment updates the adjacency matrix within the fragment, eventually when the algorithm terminates, there will be only one fragment, i.e. the MST, with one leader, marking the convergence of the distributed algorithm. The convergence speed of our distributed MCCST algorithm is dependent on the topology of the original communication graph and edge weights, ranging from one iteration to $O(\log N)$ iterations with a worst case time complexity of $O(N \log N)$.

Once the final MCCST is obtained as the optimal connectivity graph $\mathcal{G}^c = \mathcal{G}^{c*}$ in (9), we can specify the safety and connectivity barrier certificates (3) and (6) to invoke a set of linear constraints. Thus the original quadratic programming (QP) problem in (7) could be efficiently solved to get optimal revised robot controllers satisfying safety and global and subgroup connectivity constraints for behavior mixing.

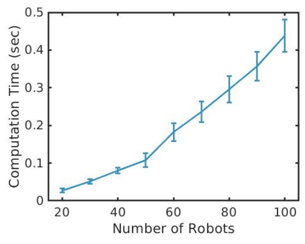
## IV. Results

### A. Simulation Example

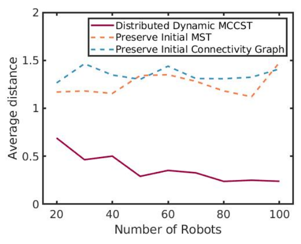
The first set of experiments are performed on a team of $N = 40$ mobile robots with unicycle dynamics as shown in Figure 1. The robot team is divided into $M = 4$ subgroups with different colors and is tasked with 4 parallel behaviors. In the figures, robots in blue subgroup 1 and red subgroup 2 execute biased rendezvous behaviors towards the blue task site 1 and red task site 2 respectively, while robots in green subgroup 3 and magenta subgroup 4



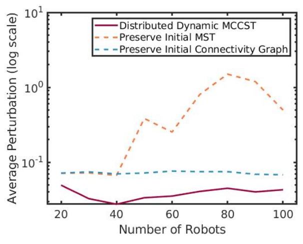
(a) Number of messages exchanged



(b) Computation Time



(c) Average Distance to targets



(d) Average Control Perturbation

Fig. 3: Quantitative results summary. (a)-(b) are results from our proposed Distributed MCCST approach. (c)-(d) are comparison results with ours and the other two approaches with static connectivity graph but the same controller (7). (a) Number of messages exchanged during the distribute MCCST construction. The error bar shows the maximum and minimum number of messages exchanged. (b) Computation time of constructing the distributed MCCST. The error bar shows the standard deviation. (c) Average distance from robot to target location after converged. (d) Average control perturbation.

perform circle formation behaviors around the green task site 3 and magenta task site 4 respectively. For our MCCST method, we apply the minimally revised controllers from (7) with single-integrator dynamics to the robots with unicycle dynamics using kinematics mapping from [25]. As shown in Figure 1a-c, our distributed MCCST approach is able to generate real-time minimum connectivity graph (red edges) from the present connectivity graph (grey edges) so that the invoked connectivity constraints are minimally restrictive to the original behavior controllers. Most of the target behavior configurations have been accomplished as shown in Figure 1c. The communication relays connecting different subgroups are implicitly formed to provide greater flexibility for the rest of the robots without the need of explicit robots roles assignment as done in [17], [19]. This is because our algorithm enforces provably minimum connectivity graph that is least restrictive to the robots.

In comparison, we present converged results of other two methods with static connectivity graph in Figure 1e and Figure 1f respectively: i) always preserving communication edges in the initial MST (red) depicted in Figure 1d, and ii) always preserving edges in initial connectivity graph (grey) in Figure 1a as done in [2]. Since the invoked connectivity graph is fixed as the robots move, they can hardly achieve circle formation (Figure 1e) or could fall into deadlock (Figure 1f). Numerical results are provided in Figure 2 showing our method ensures safety and connectivity, while having minimal control perturbation due to connectivity and maximum task performance (very close to designated target area as shown from Figure 2d). Note that in Figure 1e the provided comparison method of preserving initial MST from our MCCST without updating in real-time is already better than other barrier certificate based connectivity controllers [3], [25] that impose predefined fixed connectivity graph not necessarily as optimal for the tasks.

## B. Quantitative Results

For validating the computation efficiency and scalabiltiy of our algorithm, we run experiments with up to 100 robots and 4 parallel behaviors (four robots subgroups simultaneously rendezvous to four different places with safety and connectivity constraints). For Figure 3a and 3b, the experiment is done by computing the distributed MCCST 1500 to 3000 times, depending on the iterations for the system to converge, which varies with the number of robots and graph topology. The complexity of the worst case for both message and time is $O(N \log N)$. However, the average case, as shown in the figure, is better than $O(N \log N)$. Figure 3b shows the time duration for computing the distributed MCCST, which shows that computing distributed MCCST could be done in real time with large number of robots.

The average distance to target region and perturbation after convergence is calculated from 10 runs for each batch of robots with up to 100 robots. In Figure 3c, the average distance to target with MCCST is significantly smaller (closer to target region) than with static connectivity graph. The distance also decreases as the number of robots increases, since only a limited number of robots are needed to maintain connectivity, which enables more robots to rendezvous to the target locations. However, for the other two methods with imposed static connectivity topology, the average distance increases with the number of robots. Figure 3d shows the result of average perturbation. Our method gives much smaller perturbation on average. Note that the result from preserving initial MST gives much worse result than the other two, because the initial MST edges could give huge deviation from the optimal control outputs as behaviors progressed, while the full connectivity graph gives larger number of constrain edges to keep, so that some are canceled out with each other. Nevertheless, our distributed MCCST method always computes the minimum connectivity constraints, thus outperforming the other two methods significantly.

## V. Conclusion

In this paper, we developed the bilevel optimization based minimum connectivity maintenance framework for behavior mixing. We proposed a distributed Minimum Connectivity Constraint Spanning Tree (MCCST) algorithm to compute provably minimum global and subgroup connectivity constraints in real-time. By formulating the invoked connectivity constraints and safety constraints using safety and connectivity barrier certificates, the robots controllers are minimally modified from the original controllers with dynamic and possibly discontinuous communication topology. Experimental results show that our method is scalable and computation efficient to large number of robots. Future work includes the incremental computation of MCCST to more efficiently handle the robots joining or leaving the team dynamically.

## References

[1] R. Olfati-Saber, "Flocking for multi-agent dynamic systems: Algorithms and theory," *IEEE Transactions on automatic control*, vol. 51, no. 3, pp. 401–420, 2006.

[2] M. M. Zavlanos, A. Jadbabaie, and G. J. Pappas, "Flocking while preserving network connectivity," in *46th IEEE Conference on Decision and Control*. IEEE, 2007, pp. 2919–2924.

[3] A. Li, L. Wang, P. Pierpaoli, and M. Egerstedt, "Formally correct composition of coordinated behaviors using control barrier certificates," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 3723–3729.

[4] W. Luo, S. S. Khatib, S. Nagavalli, N. Chakraborty, and K. Sycara, "Asynchronous distributed information leader selection in robotic swarms," in *IEEE International Conference on Automation Science and Engineering (CASE)*. IEEE, 2015, pp. 606–611.

[5] ——, "Distributed knowledge leader selection for multi-robot environmental sampling under bandwidth constraints," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 5751–5757.

[6] L. Sabattini, N. Chopra, and C. Secchi, "Decentralized connectivity maintenance for cooperative control of mobile robotic systems," *The International Journal of Robotics Research*, vol. 32, no. 12, pp. 1411–1423, 2013.

[7] P. R. Giordano, A. Franchi, C. Secchi, and H. H. Bülthoff, "Bilateral teleoperation of groups of uavs with decentralized connectivity maintenance." in *Robotics: Science and Systems*. Citeseer, 2011.

[8] P. Yang, R. A. Freeman, G. J. Gordon, K. M. Lynch, S. S. Srinivasa, and R. Sukthankar, "Decentralized estimation and control of graph connectivity for mobile sensor networks," *Automatica*, vol. 46, no. 2, pp. 390–396, 2010.

[9] R. K. Williams, A. Gasparri, G. S. Sukhatme, and G. Ulivi, "Global connectivity control for spatially interacting multi-robot systems with unicycle kinematics," in *IEEE international conference on robotics and automation (ICRA)*. IEEE, 2015, pp. 1255–1261.

[10] D. Tateo, J. Banfi, A. Riva, F. Amigoni, and A. Bonarini, "Multiagent connected path planning: Pspace-completeness and how to deal with it," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[11] T. Charrier, A. Queffelec, O. Sankur, and F. Schwarzentruber, "Reachability and coverage planning for connected agents," in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*. International Joint Conferences on Artificial Intelligence Organization, 7 2019, pp. 144–150. [Online]. Available: https://doi.org/10.24963/ijcai.2019/21

[12] K. Khateri, M. Pourgholi, M. Montazeri, and L. Sabattini, "A comparison between decentralized local and global methods for connectivity maintenance of multi-robot networks," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 633–640, 2019.

[13] W. Luo and K. Sycara, "Voronoi-based coverage control with connectivity maintenance for robotic sensor networks," in *International Symposium on Multi-Robot and Multi-Agent Systems (MRS)*. IEEE, 2019, pp. 148–154.

[14] ——, "Minimum k-connectivity maintenance for robust multi-robot systems," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 7370–7377.

[15] D. V. Dimarogonas and K. H. Johansson, "Decentralized connectivity maintenance in mobile networks with bounded inputs," in *IEEE International Conference on Robotics and Automation*. IEEE, 2008, pp. 1507–1512.

[16] M. Ji and M. Egerstedt, "Distributed coordination control of multiagent systems while preserving connectedness," *IEEE Transactions on Robotics*, vol. 23, no. 4, pp. 693–703, 2007.

[17] J. Banfi, N. Basilico, and S. Carpin, "Optimal redeployment of multirobot teams for communication maintenance," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 3757–3764.

[18] M. Chandarana, W. Luo, M. Lewis, K. Sycara, and S. Scherer, "Decentralized method for sub-swarm deployment and rejoining," in *IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 2018, pp. 1209–1214.

[19] N. Majcherczyk, A. Jayabalan, G. Beltrame, and C. Pinciroli, "Decentralized connectivity-preserving deployment of large-scale robot swarms," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 4295–4302.

[20] L. Guerrero-Bonilla, D. Saldana, and V. Kumar, "Design guarantees for resilient robot formations on lattices," *IEEE Robotics and Automation Letters*, vol. 4, no. 1, pp. 89–96, 2018.

[21] A. Li, W. Luo, S. Nagavalli, and K. Sycara, "Decentralized coordinated motion for a large team of robots preserving connectivity and avoiding collisions," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 1505–1511.

[22] R. Aragues, C. Sagues, and Y. Mezouar, "Triggered minimum spanning tree for distributed coverage with connectivity maintenance," in *European Control Conference (ECC)*. IEEE, 2014, pp. 1881–1887.

[23] A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada, "Control barrier functions: Theory and applications," in *18th European Control Conference (ECC)*. IEEE, 2019, pp. 3420–3431.

[24] L. Wang, A. D. Ames, and M. Egerstedt, "Multi-objective compositions for collision-free connectivity maintenance in teams of mobile robots," in *IEEE 55th Conference on Decision and Control (CDC)*. IEEE, 2016, pp. 2659–2664.

[25] ——, "Safety barrier certificates for collisions-free multirobot systems," *IEEE Transactions on Robotics*, vol. 33, no. 3, pp. 661–674, 2017.

[26] R. G. Gallager, P. A. Humblet, and P. M. Spira, "A distributed algorithm for minimum-weight spanning trees," *ACM Transactions on Programming Languages and systems (TOPLAS)*, vol. 5, no. 1, pp. 66–77, 1983.

[27] D. Peleg, "Distributed computing," *SIAM Monographs on discrete mathematics and applications*, vol. 5, 2000.

[28] G. Pandurangan, P. Robinson, and M. Scquizzato, "A time-and message-optimal distributed algorithm for minimum spanning trees," in *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*. ACM, 2017, pp. 743–756.