KYBERNETIKA — VOLUME *56* (2020), NUMBER 3, PAGES 578–597

TOPOLOGY DESIGN FOR GROUP CONSENSUS IN DIRECTED MULTI-AGENT SYSTEMS

Onur Cihan

In this paper, we investigate the grouping behavior of multi-agent systems by exploiting the graph structure. We propose a novel algorithm for designing a network from scratch which yields the desired grouping in a network of agents utilizing a consensus-based algorithm. The proposed algorithm is shown to be optimal in the sense that it consists of the minimum number of links. Furthermore, we examine the effect of adding new vertices and edges to the network on the number of groups formed in the group consensus problem. These results can be further utilized by the network topology designer to restructure the network and achieve the desired grouping. Theoretical results are illustrated with simulation examples.

Keywords: group consensus, topology design, multi-agent agreement

Classification: 93A14, 93C05, 93C85

1. INTRODUCTION

The consensus problem has received considerable critical attention in recent past due to its applicability in different fields including social networks [5, 11], wireless sensor networks [20], computer networks [2], power grids [22], distributed decision making [21], distributed optimization [16] and robotics [3, 15]. In particular, consensus algorithms are widely used in multi-robot systems in formation control [1, 14, 25], orientation control [17], containment control [4], trajectory tracking [26], rendezvous problem [8] and task allocation [6].

In the classical consensus problem, all agents in the multi-agent system are required to agree on the same state, which is generally a physical quantity such as position or velocity. The stability conditions of consensus algorithms can be classified into two groups: i) the parametric conditions, and ii) the conditions related to network topology. While the parametric conditions may differ from algorithm to algorithm, the topological conditions are explicitly defined in the literature. Olfati-Saber and Murray show that an undirected network must be connected and a directed network must have a spanning tree in order to reach consensus [18]. When the network structure fails to satisfy these conditions, the agents do not agree on the same value and multiple groups are formed. In this case, the agents in a group achieve consensus with the agents in the same group

DOI: 10.14736/kyb-2020-3-0578

whereas the agreement values of the groups are different. Although the number of groups is equal to the number of connected components in an undirected network, as we have shown in our previous study, the graph structure must be exploited in order to determine the number of groups in a directed network [10]. Since there are important applications of group consensus in multi-agent systems (e. g., groups with different opinions in social networks, containment control of multi-robot networks), the problem of topology design for group consensus needs to be properly addressed.

In [10], we have introduced the novel concepts of primary and secondary layer subgraphs[1] that are further utilized to express the number of groups to be formed. It is also shown that agents in a particular subgraph achieve consensus with the agents in the same group, i. e., one can determine the total number of groups and the agents in each group in a multi-agent network with directed topology by using these concepts. While there are studies related to the effect of adding edges to an acyclic graph on the consensus performance [24] and the effect of cycles on the $\mathcal{H}_2$ performance of the networks [23], to the best of our knowledge, no study has addressed the question of how to design a directed network from scratch and modify an existing one to achieve the desired number of groups in the multi-agent network and group members in each group.

The main contributions of the paper can be summarized as follows:

1. The minimum number of links required to obtain a network with the desired numbers of primary and secondary layer subgraphs are expressed mathematically.

2. An algorithm, which can be used to design a directed network with the desired number of groups and the desired number of agents in each group from scratch, is proposed. This algorithm is shown to be optimal in the sense that the generated graph consists of the minimum number of links for the given setting.

3. Given an arbitrary graph, the number of links required to obtain a graph with a spanning tree is obtained. In a social network, this result can be utilized to determine the individuals with high strategic importance (the influencers).

4. The effect of creating new links and adding new agents to the multi-agent network with a directed graph is investigated. By this means, the topology designer can determine the required modifications on the existing network to achieve the desired grouping.

The remainder of the paper is organized as follows. In Section 2, we review the concepts of graph theory and give the mathematical formulation of the consensus algorithms. In Section 3, we study the topology design problem and propose an algorithm that can be used to obtain a directed network with the desired number of groups and the desired agents in each group. The analysis of modifications on the network topology is given in Section 4 which is further verified by a numerical example. Finally, we conclude the paper in Section 5.

2. MATHEMATICAL PRELIMINARIES

In this section, we review some important graph theory concepts and introduce the mathematical formulations of well-known consensus protocols in multi-agent networks.

[1]The definitions of the primary and secondary layer subgraphs are given in Section 2.

2.1. Graph theory concepts

The information flow in a multi-agent network is represented by a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V} = \{v_1, \ldots, v_n\}$ is a non-empty finite set of vertices representing the agents where n is the total number of the agents in the network; and $\mathcal{E}$ is a finite set of edges representing the communication between the agents. An edge, denoted by (v_i, v_j) or e_{ij}, shows a directed information flow from v_i to v_j, i. e., v_j receives information of v_i in one step. We denote the adjacency matrix of a graph $\mathcal{G}$ by $A = [a_{ij}]$ where $a_{ij} > 0$ if $e_{ji} \in \mathcal{E}$ and $a_{ij} = 0$ otherwise. Let the set of neighbors of v_i be defined as $\mathcal{N}_i = \{j \mid (v_j, v_i) \in \mathcal{E}\}$. Let the empty set be denoted by $\emptyset$ and the set of vertex indices be denoted by $\mathcal{I} = \{1, 2, \ldots, n\}$.

We say there is a directed path between v_1 and v_n if there exists a finite sequence of ordered edges of the form $(v_1, v_2), (v_2, v_3), \ldots, (v_{n-1}, v_n)$ such that $(v_i, v_{i+1}) \in \mathcal{E}$, $i = 1, \ldots, n-1$. A directed graph has a spanning tree if each vertex has one parent vertex except for one vertex, called the root, which has a directed path to all other vertices in the graph. Note that we use the concepts of agent and vertex interchangeably throughout the paper.

2.2. Mathematical model of consensus algorithms

In this section, we introduce the discrete-time and continuous-time mathematical models of the averaging based consensus algorithm.

2.2.1. Discrete-time consensus algorithm

In a multi-agent network consisting of n agents, the first-order agent dynamics can be expressed in discrete-time as

$$x_i(k+1) = w_{ii}x_i(k) + \sum_{j \in \mathcal{N}_i} w_{ij}x_j(k), \quad i \in \mathcal{I} \tag{1}$$

where $x_i(k) \in R^m$ is the state vector of the ith agent at time step k and w_{ij} is the non-negative averaging coefficient related with the information flow between agents j and i.

Assumption 2.1. The averaging coefficients are assumed to satisfy the conditions below:

(i) $w_{ii} > 0, \forall i \in \mathcal{I}$.

(ii) $w_{ij} \begin{cases} > 0, \text{if } (v_j, v_i) \in \mathcal{E}, \\ = 0, \text{if } (v_j, v_i) \notin \mathcal{E}, \end{cases} \quad i, j \in \mathcal{I}, i \neq j.$

(iii) $\sum_{j=1}^{n} w_{ij} = 1, \forall i \in \mathcal{I}$.

Assumption 2.1(i) requires that the agent uses its own data in its update. Assumption 2.1(ii) states that the information coming from a neighbor should be used with strictly positive weighting. Assumption 2.1(iii) is required to guarantee the stability of the consensus algorithm.

2.2.2. Continuous-time consensus algorithm

In a multi-agent network consisting of n agents, the first-order agent dynamics can be expressed in continuous-time as

$$\dot{x}_i(t) = \sum_{j \in \mathcal{N}_i} a_{ij}(x_j(t) - x_i(t)), \quad i \in \mathcal{I} \tag{2}$$

where $x_i(t) \in R^m$ is the state vector of the ith agent at time t and $a_{ij} \geq 0$ is the (i,j)th component of the weighted adjacency matrix A.

A well-known stability condition of the (discrete-time and continuous-time) consensus algorithms is the existence of a spanning tree in the multi-agent network [18, 19]. When the network graph does not contain a spanning tree, the agents do not agree on the same state and the multi-agent network automatically achieve *group consensus*, whose definition is given as follows.

Definition 2.2. (*Group Consensus*) We say that the network represented by system (1) (or equivalently, (2)) achieves K-group consensus if there exist K distinct constant vectors $c_l \in R^m$, and K non-empty sets S_l, $l = 1, \ldots, K$, such that

$$\bigcup_{l=1}^{K} S_l = \mathcal{V}, \quad S_l \cap S_q = \emptyset, \quad \text{for } l \neq q, \text{ and } l, q = 1, \ldots, K$$

and for the set S_l we have

$$\lim_{k \to \infty} ||x_i(k) - c_l|| = 0, \quad \forall v_i \in S_l, \quad i = 1, \ldots, n$$

for arbitrary initial conditions $x_i(0) \in R^m$ $(i \in \mathcal{I})$ and arbitrary choice of averaging coefficients w_{ij} satisfying Assumption 2.1 (or equivalently, arbitrary choice of non-negative weighted adjacency matrix components a_{ij}).

Definition 2.2 delineates that the multi-agent network forms K groups if the states of the agents asymptotically converge to K distinct equilibria. We would like to emphasize that this grouping behavior is not related to the choice of parameters, but due to the structure of the directed graph as illustrated in the following example.

Example 2.3. Consider the multi-agent system depicted in Figure 1 where the objective of the agents is to gather at the same location (rendezvous problem). Suppose that the initial positions of the agents ($x_i(0) \in R^2$) are randomly distributed in two-dimensional space as shown in Figure 2a.

When the agents utilize the consensus algorithm (1), the positions of the agents at $k = 10$ and $k = 100$ are illustrated in Figure 2b and 2c, respectively. As can be seen from the figure, the multi-agent network forms 8 groups where the agents in the same group gather at the same location. While it is not straightforward to directly conclude these groups from the structure of the network (see Figure 1), the groups and their members can be determined by utilizing the definitions of primary and secondary layer subgraphs which were first introduced in [10] and restated in Definition 2.5.

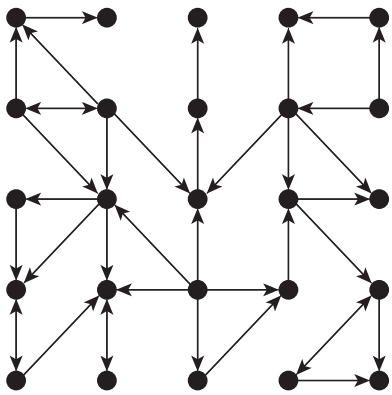


Fig. 1: A directed graph $\mathcal{G}$ consisting of 25 vertices and 38 edges.

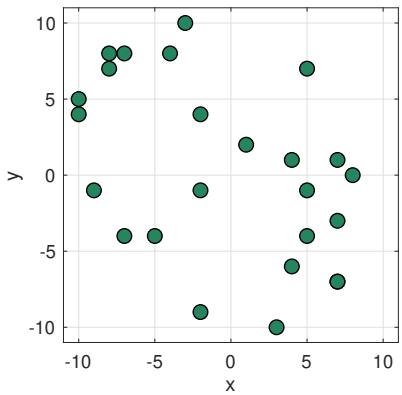
Remark 2.4. Note from Definition 2.2 that the states of the agents in different groups converge to different values and the averaging coefficients w_{ij} are non-negative by Assumption 2.1 in the group consensus problem. On the other hand, in coopetition networks, the weighting coefficients can be positive or negative depending on whether the agents cooperate or not. In such networks, bipartite consensus and fragmentation problems are defined where multiple groups may be formed [12, 13]. Note that the averaging coefficients play a key role in determining the groups in the bipartite consensus and fragmentation problems whereas the groups are solely determined by the network structure in the group consensus problem. Furthermore, unlike the bipartite consensus and fragmentation problems, the existence of a spanning tree is not an assumption in the group consensus problem.

Definition 2.5. (Primary and secondary layer subgraphs Erkan et al. [10]) A directed graph, $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, can be uniquely partitioned into $K = l_p + l_s$ subgraphs such that

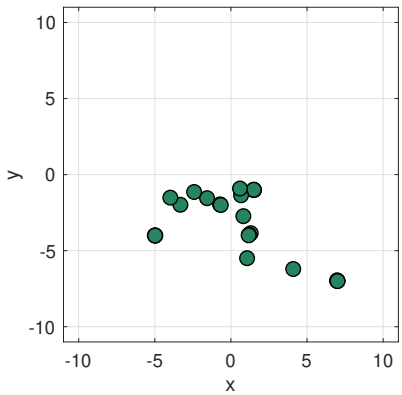
(i) l_p subgraphs, denoted by $\mathcal{G}_{p,i} = (\mathcal{V}_{p,i}, \mathcal{E}_{p,i})$, $i = 1, \ldots, l_p$, consist of a spanning tree with the maximum possible number of vertices where there is no directed path from a vertex in $\mathcal{V} \setminus \mathcal{V}_{p,i}$ to the vertices in $\mathcal{V}_{p,i}$, and

(ii) l_s subgraphs, denoted by $\mathcal{G}_{s,j} = (\mathcal{V}_{s,j}, \mathcal{E}_{s,j})$, $j = 1, \ldots, l_s$, consist of a spanning tree where there exist at least two directed paths from vertices of two different subgraphs to the root of $\mathcal{G}_{s,j}$ which is the only vertex to receive information from the vertices in $\mathcal{V} \setminus \mathcal{V}_{s,j}$.

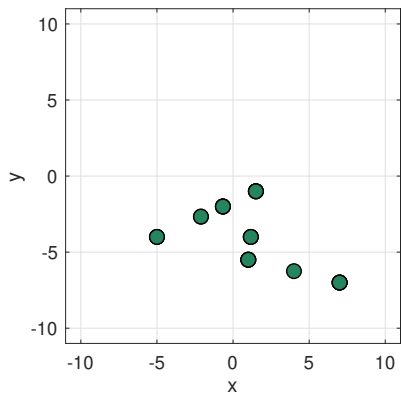
The subgraphs $\mathcal{G}_{p,i}$ $(i = 1, \ldots, l_p)$ and $\mathcal{G}_{s,j}$ $(j = 1, \ldots, l_s)$ are called *the primary layer subgraphs* and *the secondary layer subgraphs* of $\mathcal{G}$, respectively. Note that, there may exist multiple spanning trees with different root vertices for both primary and secondary layer subgraphs. However, for the secondary layer subgraphs, there is a unique root that receives information from the rest of the graph which we call *the principal root* of the subgraph and denote by $r_{\mathcal{G}_{s,j}}$.



(a) Initial positions of the agents.



(b) Positions of the agents at $k = 10$.



(c) Positions of the agents at $k = 100$.

Fig. 2: Evolution of the agent positions in Example 2.3.

Note here that with the above definition, each vertex in $\mathcal{G}$ will either be in a primary or a secondary layer subgraph and these subgraphs can be uniquely determined [9, 10]. Note also that an isolated vertex is a primary layer subgraph itself by definition. These definitions are utilized in the following Lemma which states the conditions on the topology where the network forms K groups.

Lemma 2.6. (Erkan et al. [10], Develer and Akar [7]) Consider a directed graph $\mathcal{G}$ consisting of l_p primary and l_s secondary layer subgraphs. Then the multi-agent network with dynamics (1) (or equivalently (2)) forms $K = l_p + l_s$ groups. Furthermore, the agents in a particular primary (or secondary) layer subgraph achieve consensus with the agents in the same primary (or secondary) layer subgraph.

The proof of Lemma 2.6 can be found in [7, 10]. The relationship between the final values of the agents in the primary and secondary layer subgraphs is stated in the following Lemma.

Lemma 2.7. Let $\mathcal{G}$ denote a directed graph representing a network with agent dynamics (1) (or equivalently (2)). Let $\mathcal{I}_{p,i}$ and $\mathcal{I}_{s,j}$ denote the sets of indices of the vertices in the ith primary and jth secondary layer subgraph of $\mathcal{G}$, respectively. Let $v_i \in \mathcal{R}^m$ denote the consensus values of the agents in the ith primary layer subgraph, i. e., $\lim_{k\to\infty} x_j(k) = v_i$ for all $j \in \mathcal{I}_{p,i}$. Then

$$\lim_{k\to\infty} x_l(k) = \sum_{i=1}^{l_p} \alpha_{ij} v_i \quad \text{for all } l \in \mathcal{I}_{s,j}$$

where $\alpha_{ij} \geq 0$ for all i, j; and $\sum_{i=1}^{l_p} \alpha_{ij} = 1$ for all j.

Remark 2.8. The proof of Lemma 2.7 can be found in [7]. We conclude from Lemma 2.7 that the states of the agents in the secondary layer subgraphs eventually converge to a convex combination of the final values of the agents in the primary layer subgraphs. This result can be verified by Example 2.3 where the positions of the agents in the secondary layer subgraphs are in the convex hull of the positions of the agents in the primary layer subgraphs (see Figure 2c).

Remark 2.9. As stated in Lemma 2.6, the number of groups in the multi-agent network is related to the numbers of primary and secondary layer subgraphs, i. e., the structure of the directed network. In practical applications where a conventional consensus algorithm is used, the topology design arises as an important problem since the groups and the group members need to be determined by the topology designer. For instance, in the containment control problem (which finds applications in dangerous material handling and cooperative transportation), the topology designer must determine the number of leader and follower groups (primary and secondary layer subgraphs), and the agents in each group. Once the topology is designed properly, the agents in the leader groups can be equipped with necessary sensors that can detect hazardous obstacles and consequently the agents in the follower groups (that are not equipped with sensors) will be able to move safely inside the convex hull formed by the leader groups [4].

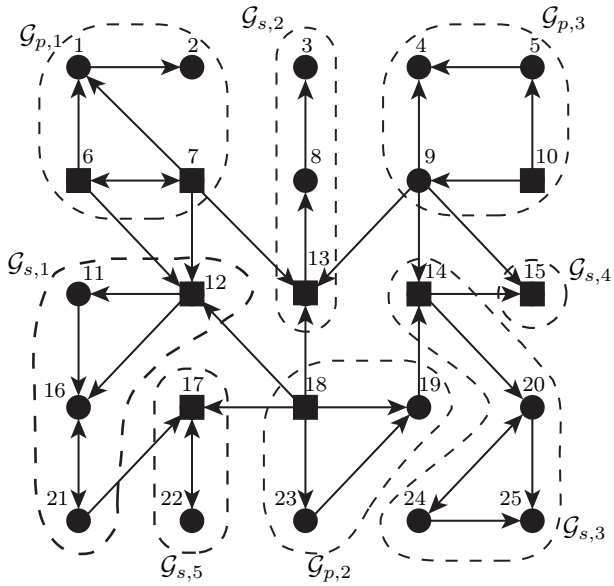


Fig. 3: The primary and the secondary layer subgraphs of $\mathcal{G}$.

Remark 2.10. In previous studies, it is shown that the number of groups formed in a multi-agent network with agent dynamics (1) or (2) is defined as $K = l_p + l_s$, i. e., the sum of the number of primary and secondary layer subgraphs [9, 10]. One of the main objectives of this paper is to design an algorithm that can be used to generate a directed graph with the desired numbers of primary and secondary layer subgraphs, and the desired numbers of agents in each subgraph. By this means, the number of groups and the members in each group can be explicitly determined by the network topology designer. Moreover, we also investigate the effect of adding new agents to the network and building new links between the agents on the number of groups, which can be further used to modify an existing graph in order to satisfy given design criteria.

Remark 2.11. In our previous study, we have proposed two algorithms to determine the primary and secondary layer subgraphs of a given directed graph [10]. While these algorithms are sufficient for detection, they need to be modified as given in the Appendices (see Algorithms 2 and 3) to be further utilized in the topology design.

Example 2.12. Reconsider the multi-agent network consisting of 25 agents as depicted in Figure 1. Algorithms 2 and 3 can be utilized to conclude that there exist $l_p = 3$ primary and $l_s = 5$ secondary layer subgraphs in the network as shown in Figure 3. Note that the roots of the primary layer subgraphs and the principal roots of the secondary layer subgraphs are marked with squares whereas other vertices are marked with circles. Note here also that while there may be more than one vertex which is a root in a primary layer subgraph, the principal roots of the secondary layer subgraphs are unique by Definition 2.5.

3. TOPOLOGY DESIGN IN A MULTI-AGENT NETWORK FROM SCRATCH

In this section, we discuss the topology design problem of an arbitrary directed graph with the desired number of primary and secondary layer subgraphs and the desired number of vertices in each subgraph. To this end, we state the following theorem which will be further utilized in the topology design algorithm.

Theorem 3.1. The minimum number of links required to design a directed graph with l_p primary and l_s secondary layer subgraphs and n vertices is $n - l_p + l_s$.

Proof. Let $n_{p,i}$ $(i = 1, \ldots, l_p)$ and $n_{s,j}$ $(j = 1, \ldots, l_s)$ be the number of vertices in the ith primary and the jth secondary layer subgraph, respectively. Since the simplest graph consisting of a spanning tree is a directed acyclic graph, each primary and secondary layer subgraph must contain at least $n_{p,i} - 1$ and $n_{s,j} - 1$ internal links, respectively. Furthermore, from Definition 2.5, the principal roots of secondary layer subgraphs are linked with at least two different subgraphs and therefore $2l_s$ links are necessary for inter-subgraph connections. Consequently, the minimum number of links required for the design can be computed as

$$\sum_{i=1}^{l_p} (n_{p,i} - 1) + \sum_{j=1}^{l_s} (n_{s,j} - 1) + 2l_s = n - l_p + l_s.$$

□

Given arbitrary feasible numbers of the primary and secondary layer subgraphs[2] (l_p and l_s) and numbers of agents in each subgraph ($n_{p,i}$ and $n_{s,j}$, $i = 1, \ldots, l_p, j = 1, \ldots l_s$), Algorithm 1 can be used to generate a directed graph with the minimum number of edges. The procedure of Algorithm 1 can be summarized as follows. In the first phase, n isolated vertices are added to the graph. In Phase 2, $n_{p,i}$ $(i = 1, \ldots, l_p)$ vertices are connected to each other to form l_p directed acyclic subgraphs. The same method is applied so as to create additional l_s directed acyclic subgraphs in Phase 3. By the end of this phase, the network consists of $l_p + l_s$ acyclic subgraphs without any inter-subgraph connections. In the final phase, the roots of subgraphs generated in Phase 3 are connected to two randomly chosen vertices from two different randomly chosen subgraphs, resulting in a graph topology consisting of l_p primary and l_s secondary layer subgraphs. Note that the algorithm is optimal in the sense of the number of links, i. e., it generates a directed graph with the desired settings by creating the minimum number of links $(n - l_p + l_s)$ as stated in Theorem 3.1.

Remark 3.2. In Algorithm 1, Phase 1 requires time $O(n)$ since it consists of initialization of an array with n elements and assignment of two variables and an empty set. In Phase 2, $\sum_{i=1}^{l_p} n_{p,i} - l_p$ elements are added to the sets of edges and $\sum_{i=1}^{l_p} n_{p,i}$ elements are added to the sets of vertices which yields a time complexity of $O(n)$. Similarly, in Phase 3, $\sum_{j=1}^{l_s} n_{s,j} - l_s$ elements are added to the sets of edges and $\sum_{j=1}^{l_s} n_{s,j}$ elements are added to the sets of vertices which yields a time complexity of $O(n)$. Finally, for each secondary

[2]Based on the partitioning given in Definition 2.5, a directed graph consists of at least one primary layer subgraph ($l_p > 0$). Furthermore, if there is a secondary layer subgraph, there must be at least two primary layer subgraphs.

Algorithm 1 Algorithm to generate a directed graph with a given feasible setting-

Require:
$(l_p \geq 1$ and $l_s = 0)$ OR $(l_p \geq 2$ and $l_s > 0)$,
$n_{p,i} \geq 1$, $n_{s,j} \geq 1$, for all $i = 1, \ldots, l_p$ and $j = 1, \ldots, l_s$

procedure $[\mathcal{G}, \mathcal{G}_{p,1}, \ldots, \mathcal{G}_{p,l_p}, \mathcal{G}_{s,1}, \ldots, \mathcal{G}_{s,l_s}]$=GraphGen$(n_{p,1}, \ldots, n_{p,l_p}, n_{s,1}, \ldots, n_{s,l_s})$
  $n \leftarrow \sum_{i=1}^{l_p} n_{p,i} + \sum_{j=1}^{l_s} n_{s,j}$
  $\mathcal{V} \leftarrow \{v_1, \ldots, v_n\}$ ▷ Phase 1
  $\mathcal{E} \leftarrow \emptyset$
  $k \leftarrow l_p + l_s + 1$
  for $i \leftarrow 1, l_p$ **do** ▷ Phase 2
    $\tilde{\mathcal{V}} \leftarrow \{v_i\}$
    $\tilde{\mathcal{E}} \leftarrow \emptyset$
    if $n_{p,i} > 1$ **then**
      for $j \leftarrow 2, n_{p,i}$ **do**
        $u \leftarrow$ a random element from the set $\tilde{\mathcal{V}}$
        $\tilde{\mathcal{V}} \leftarrow \tilde{\mathcal{V}} \cup \{v_k\}$
        $\tilde{\mathcal{E}} \leftarrow \tilde{\mathcal{E}} \cup \{(u, v_k)\}$
        $k \leftarrow k + 1$
      end for
    end if
    $\mathcal{V}_{p,i} \leftarrow \tilde{\mathcal{V}}$
    $\mathcal{E}_{p,i} \leftarrow \tilde{\mathcal{E}}$
    $\mathcal{G}_{p,i} \leftarrow (\mathcal{V}_{p,i}, \mathcal{E}_{p,i})$
    $\mathcal{E} \leftarrow \mathcal{E} \cup \tilde{\mathcal{E}}$
  end for
  for $i \leftarrow 1, l_s$ **do** ▷ Phase 3
    $\tilde{\mathcal{V}} \leftarrow \{v_{l_p+i}\}$
    $\tilde{\mathcal{E}} \leftarrow \emptyset$
    if $n_{s,i} > 1$ **then**
      for $j \leftarrow 2, n_{s,i}$ **do**
        $u \leftarrow$ a random element from $\tilde{\mathcal{V}}$
        $\tilde{\mathcal{V}} \leftarrow \tilde{\mathcal{V}} \cup \{v_k\}$
        $\tilde{\mathcal{E}} \leftarrow \tilde{\mathcal{E}} \cup \{(u, v_k)\}$
        $k \leftarrow k + 1$
      end for
    end if
    $\mathcal{V}_{s,i} \leftarrow \tilde{\mathcal{V}}$
    $\mathcal{E}_{s,i} \leftarrow \tilde{\mathcal{E}}$
    $\mathcal{G}_{s,i} \leftarrow (\mathcal{V}_{s,i}, \mathcal{E}_{s,i})$
    $\mathcal{E} \leftarrow \mathcal{E} \cup \tilde{\mathcal{E}}$
  end for
  if $l_s > 0$ **then**
    for $i \leftarrow 1, l_s$ **do** ▷ Phase 4
      $g_1, g_2 \leftarrow$ two random elements from $\{1, \ldots, l_p + l_s\} \setminus \{l_p + i, \ldots, l_p + l_s\}$
      for $j \leftarrow 1, 2$ **do**
        if $g_j \leq l_p$ **then**
          $q_j \leftarrow$ random element from $\mathcal{V}_{p,g_j}$
        else
          $q_j \leftarrow$ random element from $\mathcal{V}_{s,g_j - l_p}$
        end if
      end for
      $\mathcal{E} \leftarrow \mathcal{E} \cup \{(q_1, v_{l_p+i}), (q_2, v_{l_p+i})\}$
    end for
  end if
  $\mathcal{G} \leftarrow (\mathcal{V}, \mathcal{E})$
  return $[\mathcal{G}, \mathcal{G}_{p,1}, \ldots, \mathcal{G}_{p,l_p}, \mathcal{G}_{s,1}, \ldots, \mathcal{G}_{s,l_s}]$
end procedure

layer subgraph, two random vertices are picked from two different sets and a link between these vertices is added to the edge set which requires time $O(l_s)$ in Phase 4. Therefore, the overall time complexity of Algorithm 1 is $O(n)+O(n)+O(n)+O(l_s) = O(n)$. Since a directed graph consisting of n vertices and $n - l_p + l_s$ links cannot be generated by an algorithm faster than $O(n)$, we conclude that Algorithm 1 is optimal and the time required to execute the algorithm will increase linearly with the network size n.

We provide the following example to demonstrate Algorithm 1.

Example 3.3. Consider the problem of generating a directed network with $l_p = 4$ primary and $l_s = 5$ secondary layer subgraphs where the numbers of agents in each subgraph are desired to be as follows: $n_{p,1} = 1, n_{p,2} = 2, n_{p,3} = 3, n_{p,4} = 4, n_{s,1} = 4, n_{s,2} = 2, n_{s,3} = 5, n_{s,4} = 1$ and $n_{s,5} = 2$. Figure 4 illustrates a possible topology with the desired settings generated by Algorithm 1. The obtained graph consists of $n = \sum_{i=1}^{l_p} n_{p,i} + \sum_{i=1}^{l_s} n_{s,i} = 24$ vertices and $n - l_p + l_s = 25$ edges which verifies the efficiency of Algorithm 1.

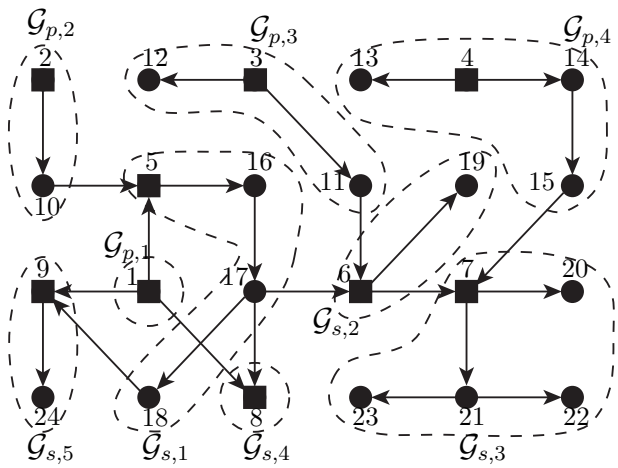


Fig. 4: A directed graph generated by Algorithm 1.

4. MODIFICATION OF THE GRAPH STRUCTURE

In this section, we investigate the effect of adding new vertices and edges to a given graph on the number of groups formed in the multi-agent system. Let the modified graph be denoted by $\bar{\mathcal{G}}$, and the numbers of primary and secondary layer subgraphs of $\bar{\mathcal{G}}$ be denoted by $\bar{l}_p$ and $\bar{l}_s$, respectively.

4.1. Adding new vertices

Given an arbitrary graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, the following lemma states the effect of adding a new vertex on the topological properties of the graph.

Lemma 4.1. Suppose a vertex v_k is added to a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ resulting in the modified graph $\bar{\mathcal{G}} = (\mathcal{V} \cup \{v_k\}, \mathcal{E})$. Then $\bar{l}_p = l_p + 1$ and $\bar{l}_s = l_s$.

P r o o f . Note from Definition 2.5 that an isolated vertex is a primary layer subgraph itself and therefore the number of primary layer subgraphs will be increased by 1. Furthermore, since an isolated vertex does not interact with other vertices, the rest of the subgraphs will remain unchanged. □

4.2. Creating new links

Let v_k, v_l be two vertices in $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. In this section, we investigate the effect of creating a link between v_k and v_l (i. e., $\bar{\mathcal{G}} = (\mathcal{V}, \mathcal{E} \cup \{(v_k, v_l)\})$ on the topology of the graph in terms of the graph partitioning given in Definition 2.5. To this end, let $\Upsilon_{\mathcal{G}_{s,j}}$ denote the set of primary layer subgraphs whose root vertex can access the vertices in $\mathcal{G}_{s,j}$. Let $\mathcal{S}_{\mathcal{G},v}$ denote the set of vertices that are reachable from vertex v in $\mathcal{G}$. For a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ consisting of a spanning tree, let $\mathcal{R}_{\mathcal{G}}$ denote the set of roots of $\mathcal{G}$, i. e., for all $v \in \mathcal{R}_{\mathcal{G}}$, we have $\mathcal{S}_{\mathcal{G},v} = \mathcal{V}$.

The following lemma considers the case where a new link between two vertices in the same subgraph is created.

Lemma 4.2. Suppose a new link (v_k, v_l) is created between two vertices in the same subgraph. Then $\bar{l}_p = l_p$ and $\bar{l}_s = l_s$.

P r o o f . Recall from Definition 2.5 that each subgraph consists of a spanning tree. Therefore, the subgraph with the new link also consists of a spanning tree. Furthermore, since the link (v_k, v_l) is not connecting two different subgraphs of $\mathcal{G}$, it follows from Definition 2.5 that not only the type of the subgraph consisting of v_k and v_l , but also the rest of the subgraphs of $\mathcal{G}$ remains unchanged. Hence, we conclude that $\mathcal{G}$ and $\bar{\mathcal{G}}$ have the same partitioning. □

The effect of creating a link between two vertices that are in different primary layer subgraphs is described in the following lemma.

Lemma 4.3. Suppose that $\mathcal{G}_{p,i}$ and $\mathcal{G}_{p,j}$ are two distinct primary layer subgraphs of $\mathcal{G}$ and a new link between $v_k \in \mathcal{G}_{p,i}$ and $v_l \in \mathcal{G}_{p,j}$ is created. Let $\mathbb{G}_p = \{\hat{\mathcal{G}} \mid \Upsilon_{\hat{\mathcal{G}}} = \{\mathcal{G}_{p,i}, \mathcal{G}_{p,j}\}\}$ be the set of (secondary layer) subgraphs of $\mathcal{G}$ whose vertices are accessible from the roots of $\mathcal{G}_{p,i}$ and $\mathcal{G}_{p,j}$; but not accessible from the roots of other primary layer subgraphs. Then,

(i) If $v_l \in \mathcal{R}_{\mathcal{G}_{p,j}}$, then $\bar{l}_p = l_p - 1$ and $\bar{l}_s = l_s - |\mathbb{G}_p|$, and

(ii) If $v_l \notin \mathcal{R}_{\mathcal{G}_{p,j}}$, then $\bar{l}_p = l_p$ and $\bar{l}_s = l_s + \hat{l}_s$

where $\hat{l}_s$ denotes the number of secondary layer subgraphs of $\hat{\mathcal{G}} = (\mathcal{V}_{p,j} \cup \{v_k\}, \mathcal{E}_{p,j} \cup \{(v_k, v_l)\})$ and $|\mathbb{G}_p|$ denotes the cardinality of the set $\mathbb{G}_p$.

Proof. We first consider (i). In the modified graph $\bar{\mathcal{G}}$, the vertices in $\mathcal{G}_{p,i} \cup \mathcal{G}_{p,j} \cup (\bigcup \mathbb{G}_p)$ are reachable from the root vertex of $\mathcal{G}_{p,i}$. This corresponds to an augmented primary layer subgraph consisting of the vertices of two primary layer subgraphs and $|\mathbb{G}_p|$ secondary layer subgraphs of $\mathcal{G}$. The remaining secondary layer subgraphs remains unchanged, and therefore the number of primary layer subgraphs is decreased by 1 and the number of secondary layer subgraphs is decreased by $|\mathbb{G}_p|$.

For the second case, while the new link does not affect the subgraph $\mathcal{G}_{p,i}$, the subgraph $\mathcal{G}_{p,j}$ shrinks to a smaller primary layer subgraph with the vertex set $\mathcal{V}_{p,j} \setminus \mathcal{S}_{\mathcal{G}_{p,i},v_l}$. Furthermore, when the link (v_k, v_l) is created, the vertices in $\mathcal{S}_{\mathcal{G}_{p,j},v_l}$ form $\hat{l}_s$ secondary layer subgraphs in $\bar{\mathcal{G}}$. Since the rest of the graph $\mathcal{G}$ remains unchanged, we conclude that creating a link between v_k and v_l does not change the number of primary layer subgraphs whereas the number of secondary layer subgraphs is increased by $\hat{l}_s$. □

Remark 4.4. Given an arbitrary directed graph, creating a link between a vertex in a primary layer subgraph and a root vertex of another primary layer subgraph always decreases the total number of subgraphs. This result is extended in the following theorem to state the conditions on the minimum number of links to be created under which the modified graph has a spanning tree.

Theorem 4.5. Let $\mathcal{G}$ be a directed graph consisting of l_p primary and l_s secondary layer subgraphs. The minimum number of links that must be created in order to obtain a graph containing a spanning tree is $l_p - 1$.

Proof. Let $v^*_{p,i}$, $i \in \mathcal{I}_p$, denote a root vertex of the ith primary layer subgraph, i. e., $\mathcal{S}_{\mathcal{G}_{p,i},v^*_{p,i}} = \mathcal{V}_{p,i}$ for all $i \in \mathcal{I}_p$. Then, we conclude from Lemma 4.3 that creating the links $(v^*_{p,i}, v^*_{p,j})$, $j \in \mathcal{I}_p \setminus \{i\}$, results in a single primary layer subgraph and consequently no secondary layer subgraphs. Since a graph with a single primary layer subgraph consists of a spanning tree, $l_p - 1$ links are sufficient for a graph to have a spanning tree. Furthermore, since the vertices of a primary layer subgraph do not receive information from the vertices in other subgraphs, it is not possible to obtain a graph with a spanning tree by creating less than $l_p - 1$ links. Hence, $l_p - 1$ is the minimum number of links that must be created in order to build a directed graph with a spanning tree. □

Remark 4.6. In a multi-agent system, it is well-known that the existence of a spanning tree is necessary to achieve consensus on a single equilibrium state [18, 19]. Theorem 4.5 states the minimum number of links that must be created for the agents in a directed network with K subgraphs to agree on a single equilibrium state.

Remark 4.7. In social networks, consensus algorithms are widely used to express the evaluation of the opinions of the individuals [5, 11]. We conclude from Theorem 4.5 that it is theoretically possible to influence all individuals by connecting $l_p - 1$ individuals together. The individuals with high strategic importance are the roots of the primary layer subgraphs of the social network.

The following lemma states the effect of creating a link between a vertex in a primary layer subgraph and a vertex in a secondary layer subgraph.

Lemma 4.8. Suppose that $\mathcal{G}_{p,i}$ and $\mathcal{G}_{s,j}$ are primary and secondary layer subgraphs of $\mathcal{G}$, respectively. When a new link between $v_k \in \mathcal{G}_{p,i}$ and $v_l \in \mathcal{G}_{s,j}$ is created,

i) If v_l is the principal root of $\mathcal{G}_{s,j}$, then $\bar{l}_p = l_p$ and $\bar{l}_s = l_s$, and

ii) If v_l is not the principal root of $\mathcal{G}_{s,j}$, then $\bar{l}_p = l_p$ and $\bar{l}_s = l_s + \hat{l}_s$

where $\hat{l}_s$ denotes the number of secondary layer subgraphs of $\hat{\mathcal{G}} = (\mathcal{V}_{s,j} \cup \{v_k\}, \mathcal{E}_{s,j} \cup \{(v_k, v_l)\})$.

P r o o f. For case (i), the link (v_k, v_l) does not affect the structure of the overall graph since v_l already has at least two neighbors from at least two different subgraphs by Definition 2.5. For case (ii), while the new link does not affect the subgraph $\mathcal{G}_{p,i}$, the subgraph $\mathcal{G}_{s,j}$ shrinks to a smaller secondary layer subgraph with the vertex set $\mathcal{V}_{s,j} \setminus \mathcal{S}_{\mathcal{G}_{s,j},v_l}$. Furthermore, when the link (v_k, v_l) is created, the vertices in $\mathcal{S}_{\mathcal{G}_{s,j},v_l}$ form $\hat{l}_s$ secondary layer subgraphs in $\bar{\mathcal{G}}$. Since the rest of the graph $\mathcal{G}$ remains unchanged, we conclude that creating a link between v_k and v_l does not change the number of primary layer subgraphs whereas the number of secondary layer subgraphs is increased by $\hat{l}_s$. □

The effect of building a link between the vertices of a secondary and a primary layer subgraph is stated in the following lemma.

Lemma 4.9. Suppose that $\mathcal{G}_{p,i}$ and $\mathcal{G}_{s,j}$ are primary and secondary layer subgraphs of $\mathcal{G}$, respectively. Let $\mathbb{G} = \{\mathcal{G}_{s,k} \mid \Upsilon_{\mathcal{G}_{s,k}} = \Upsilon_{\mathcal{G}_{s,j}}\}$ be the set of (secondary layer) subgraphs of $\mathcal{G}$ which receive information from the same primary layer subgraphs as $\mathcal{G}_{s,j}$. When a new link between $v_k \in \mathcal{G}_{s,j}$ and $v_l \in \mathcal{G}_{p,i}$ is created,

(i) If $v_l \in \mathcal{R}_{\mathcal{G}_{p,i}}$, then $\bar{l}_p = l_p - 1$. Furthermore, $\bar{l}_s = l_s - |\mathbb{G}|$ if $|\Upsilon_{\mathcal{G}_{s,j}} \setminus \mathcal{G}_{p,i}| = 1$; and $\bar{l}_s = l_s$ otherwise.

(ii) If $v_l \notin \mathcal{R}_{\mathcal{G}_{p,i}}$, then $\bar{l}_p = l_p$ and $\bar{l}_s = l_s + \hat{l}_s$

where $\hat{l}_s$ denotes the number of secondary layer subgraphs of the graph $\hat{\mathcal{G}} = (\mathcal{V}_{p,i} \cup \{v_k\}, \mathcal{E}_{p,i} \cup \{(v_k, v_l)\})$ and $|\mathbb{G}|$ denotes the cardinality of $\mathbb{G}$.

P r o o f. We first consider (i). In the modified graph, the vertices in $\mathcal{G}_{p,i}$ are reachable from the root of $\mathcal{G}_{s,j}$ and consequently $\mathcal{G}_{p,i}$ will no longer be a primary layer subgraph. Hence we have $\bar{l}_p = l_p - 1$. Moreover, if $|\Upsilon_{\mathcal{G}_{s,j}} \setminus \mathcal{G}_{p,i}| = 1$ all subgraphs in $\mathbb{G}$ will join to the primary layer subgraph consisting the former subgraphs $\Upsilon_{\mathcal{G}_{s,j}} \setminus \mathcal{G}_{p,i}$. Therefore the number of secondary layer subgraphs will be decreased by $|\mathbb{G}|$. On the other hand, if $|\Upsilon_{\mathcal{G}_{s,j}} \setminus \mathcal{G}_{p,i}| > 1$, the subgraphs $\mathcal{G}_{s,j}$ and $\mathcal{G}_{p,i}$ merge and become a larger secondary layer subgraph, i. e., the number of secondary layer subgraphs will remain unchanged.

For the second case, while the new link does not affect the subgraph $\mathcal{G}_{s,j}$, the subgraph $\mathcal{G}_{p,i}$ shrinks to a smaller primary layer subgraph with the vertex set $\mathcal{V}_{p,i} \setminus \mathcal{S}_{\mathcal{G}_{p,i},v_l}$. Furthermore, when the link (v_k, v_l) is created, the vertices in $\mathcal{S}_{\mathcal{G}_{p,i},v_l}$ form $\hat{l}_s$ secondary layer subgraphs in $\bar{\mathcal{G}}$. Since the rest of the graph $\mathcal{G}$ remains unchanged, we conclude that creating a link between v_k and v_l does not change the number of primary layer subgraphs whereas the number of secondary layer subgraphs is increased by $\hat{l}_s$. □

The effect of building a link between the vertices of two distinct secondary layer subgraphs is given in the following lemma.

Lemma 4.10. Suppose that $\mathcal{G}_{s,i}$ and $\mathcal{G}_{s,j}$ are two distinct secondary layer subgraphs of $\mathcal{G}$. When a new link between $v_k \in \mathcal{G}_{s,i}$ and $v_l \in \mathcal{G}_{s,j}$ is created,

(i) If v_l is the principal root of $\mathcal{G}_{s,j}$, then $\bar{l}_p = l_p$ and $\bar{l}_s = l_s$.

(ii) If v_l is not the principal root of $\mathcal{G}_{s,j}$, then $\bar{l}_p = l_p$ and $\bar{l}_s = l_s + \hat{l}_s$

where $\hat{l}_s$ denotes the number of secondary layer subgraphs of $\hat{\mathcal{G}} = (\mathcal{V}_{s,j} \cup \{v_k\}, \mathcal{E}_{s,j} \cup \{(v_k, v_l)\})$.

P r o o f . For case (i), all subgraphs will remain unchanged since there is no modification in the primary layer subgraphs and the principal roots of the secondary layer subgraphs will continue receiving information from at least two different subgraphs.

For case (ii), while the new link does not affect the subgraph $\mathcal{G}_{s,i}$, the subgraph $\mathcal{G}_{s,j}$ shrinks to a smaller secondary layer subgraph with the vertex set $\mathcal{V}_{s,j} \setminus \mathcal{S}_{\mathcal{G}_{s,j},v_l}$. Furthermore, when the link (v_k, v_l) is created, the vertices in $\mathcal{S}_{\mathcal{G}_{s,j},v_l}$ form $\hat{l}_s$ secondary layer subgraphs in $\bar{\mathcal{G}}$. Since the rest of the graph $\mathcal{G}$ remains unchanged, we conclude that creating a link between v_k and v_l does not change the number of primary layer subgraphs whereas the number of secondary layer subgraphs is increased by $\hat{l}_s$. □

The following comprehensive example is provided to demonstrate Lemmas 4.1-4.10 and Theorem 4.5.

Example 4.11. Reconsider the multi-agent network shown in Figure 1 which consists of 3 primary and 5 secondary layer subgraphs with the sets of vertices $\mathcal{V}_{p,1} = \{v_1, v_2, v_6, v_7\}$, $\mathcal{V}_{p,2} = \{v_{18}, v_{19}, v_{23}\}$, $\mathcal{V}_{p,3} = \{v_4, v_5, v_9, v_{10}\}$, $\mathcal{V}_{s,1} = \{v_{11}, v_{12}, v_{16}, v_{21}\}$, $\mathcal{V}_{s,2} = \{v_3, v_8, v_{13}\}$, $\mathcal{V}_{s,3} = \{v_{14}, v_{20}, v_{24}, v_{25}\}$, $\mathcal{V}_{s,4} = \{v_{15}\}$, $\mathcal{V}_{s,5} = \{v_{17}, v_{22}\}$. The principal roots of secondary layer subgraphs and the set of primary layer subgraphs associated with the secondary layer subgraphs are given in Table 1.

Subgraph	$r_{\mathcal{G}}$	$\Upsilon_{\mathcal{G}}$
$\mathcal{G}_{s,1}$	v_{12}	$\mathcal{G}_{p,1}, \mathcal{G}_{p,2}$
$\mathcal{G}_{s,2}$	v_{13}	$\mathcal{G}_{p,1}, \mathcal{G}_{p,2}, \mathcal{G}_{p,3}$
$\mathcal{G}_{s,3}$	v_{14}	$\mathcal{G}_{p,2}, \mathcal{G}_{p,3}$
$\mathcal{G}_{s,4}$	v_{15}	$\mathcal{G}_{p,2}, \mathcal{G}_{p,3}$
$\mathcal{G}_{s,5}$	v_{17}	$\mathcal{G}_{p,1}, \mathcal{G}_{p,2}$

Tab. 1: Secondary layer subgraphs, their principal roots and related primary layer subgraphs.

The number of primary and secondary layer subgraphs and the vertices in each subgraph are summarized in Table 2 after the modifications in the network which verifies the theoretical results given in Section 4. Note that these theoretical results can be utilized by the topology designer to obtain the desired grouping without redetecting the primary and secondary layer subgraphs of the modified graph by an algorithm.

Graph	K	$\bar{l}_p$	$\bar{l}_s$	Primary layer subgraphs	Secondary layer subgraphs
$\bar{\mathcal{G}} = (\mathcal{V} \cup \{v_{26}\}, \mathcal{E})$	9	4	5	$\bar{\mathcal{V}}_{p,1} = \{v_1, v_2, v_6, v_7\}$ $\bar{\mathcal{V}}_{p,2} = \{v_{18}, v_{19}, v_{23}\}$ $\bar{\mathcal{V}}_{p,3} = \{v_4, v_5, v_9, v_{10}\}$ $\bar{\mathcal{V}}_{p,4} = \{v_{26}\}$	$\bar{\mathcal{V}}_{s,1} = \{v_{11}, v_{12}, v_{16}, v_{21}\}$ $\bar{\mathcal{V}}_{s,2} = \{v_3, v_8, v_{13}\}$ $\bar{\mathcal{V}}_{s,3} = \{v_{14}, v_{20}, v_{24}, v_{25}\}$ $\bar{\mathcal{V}}_{s,4} = \{v_{15}\}$ $\bar{\mathcal{V}}_{s,5} = \{v_{17}, v_{22}\}$
$\bar{\mathcal{G}} = (\mathcal{V}, \mathcal{E} \cup \{(v_{20}, v_{14})\})$	8	3	5	$\bar{\mathcal{V}}_{p,1} = \{v_1, v_2, v_6, v_7\}$ $\bar{\mathcal{V}}_{p,2} = \{v_{18}, v_{19}, v_{23}\}$ $\bar{\mathcal{V}}_{p,3} = \{v_4, v_5, v_9, v_{10}\}$ $\bar{\mathcal{V}}_{p,4} = \{v_{26}\}$	$\bar{\mathcal{V}}_{s,1} = \{v_{11}, v_{12}, v_{16}, v_{21}\}$ $\bar{\mathcal{V}}_{s,2} = \{v_3, v_8, v_{13}\}$ $\bar{\mathcal{V}}_{s,3} = \{v_{14}, v_{20}, v_{24}, v_{25}\}$ $\bar{\mathcal{V}}_{s,4} = \{v_{15}\}$ $\bar{\mathcal{V}}_{s,5} = \{v_{17}, v_{22}\}$
$\bar{\mathcal{G}} = (\mathcal{V}, \mathcal{E} \cup \{(v_{23}, v_7)\})$	5	2	3	$\bar{\mathcal{V}}_{p,1} = \{v_1, v_2, v_6, v_7, v_{11}, v_{12}, v_{16}, v_{17}, v_{18}, v_{19}, v_{21}, v_{22}, v_{23}\}$ $\bar{\mathcal{V}}_{p,2} = \{v_4, v_5, v_9, v_{10}\}$	$\bar{\mathcal{V}}_{s,1} = \{v_3, v_8, v_{13}\}$ $\bar{\mathcal{V}}_{s,2} = \{v_{14}, v_{20}, v_{24}, v_{25}\}$ $\bar{\mathcal{V}}_{s,3} = \{v_{15}\}$
$\bar{\mathcal{G}} = (\mathcal{V}, \mathcal{E} \cup \{(v_7, v_{23})\})$	10	3	7	$\bar{\mathcal{V}}_{p,1} = \{v_1, v_2, v_6, v_7\}$ $\bar{\mathcal{V}}_{p,2} = \{v_4, v_5, v_9, v_{10}\}$ $\bar{\mathcal{V}}_{p,3} = \{v_{18}\}$	$\bar{\mathcal{V}}_{s,1} = \{v_{11}, v_{12}, v_{16}, v_{21}\}$ $\bar{\mathcal{V}}_{s,2} = \{v_3, v_8, v_{13}\}$ $\bar{\mathcal{V}}_{s,3} = \{v_{14}, v_{20}, v_{24}, v_{25}\}$ $\bar{\mathcal{V}}_{s,4} = \{v_{15}\}$ $\bar{\mathcal{V}}_{s,5} = \{v_{17}, v_{22}\}$ $\bar{\mathcal{V}}_{s,6} = \{v_{19}\}$ $\bar{\mathcal{V}}_{s,7} = \{v_{23}\}$
$\bar{\mathcal{G}} = (\mathcal{V}, \mathcal{E} \cup \{(v_7, v_{10}), (v_7, v_{23})\})$	1	1	0	$\bar{\mathcal{V}}_{p,1} = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9, v_{10}, v_{11}, v_{12}, v_{13}, v_{14}, v_{15}, v_{16}, v_{17}, v_{18}, v_{19}, v_{20}, v_{21}, v_{22}, v_{23}, v_{24}, v_{25}\}$	-
$\bar{\mathcal{G}} = (\mathcal{V}, \mathcal{E} \cup \{(v_9, v_{17})\})$	8	3	5	$\bar{\mathcal{V}}_{p,1} = \{v_1, v_2, v_6, v_7\}$ $\bar{\mathcal{V}}_{p,2} = \{v_{18}, v_{19}, v_{23}\}$ $\bar{\mathcal{V}}_{p,3} = \{v_4, v_5, v_9, v_{10}\}$	$\bar{\mathcal{V}}_{s,1} = \{v_{11}, v_{12}, v_{16}, v_{21}\}$ $\bar{\mathcal{V}}_{s,2} = \{v_3, v_8, v_{13}\}$ $\bar{\mathcal{V}}_{s,3} = \{v_{14}, v_{20}, v_{24}, v_{25}\}$ $\bar{\mathcal{V}}_{s,4} = \{v_{15}\}$ $\bar{\mathcal{V}}_{s,5} = \{v_{17}, v_{22}\}$
$\bar{\mathcal{G}} = (\mathcal{V}, \mathcal{E} \cup \{(v_6, v_{11})\})$	10	3	7	$\bar{\mathcal{V}}_{p,1} = \{v_1, v_2, v_6, v_7\}$ $\bar{\mathcal{V}}_{p,2} = \{v_{18}, v_{19}, v_{23}\}$ $\bar{\mathcal{V}}_{p,3} = \{v_4, v_5, v_9, v_{10}\}$	$\bar{\mathcal{V}}_{s,1} = \{v_3, v_8, v_{13}\}$ $\bar{\mathcal{V}}_{s,2} = \{v_{14}, v_{20}, v_{24}, v_{25}\}$ $\bar{\mathcal{V}}_{s,3} = \{v_{15}\}$ $\bar{\mathcal{V}}_{s,4} = \{v_{17}, v_{22}\}$ $\bar{\mathcal{V}}_{s,5} = \{v_{11}\}$ $\bar{\mathcal{V}}_{s,6} = \{v_{12}\}$ $\bar{\mathcal{V}}_{s,7} = \{v_{16}, v_{21}\}$
$\bar{\mathcal{G}} = (\mathcal{V}, \mathcal{E} \cup \{(v_{11}, v_6)\})$	5	2	3	$\bar{\mathcal{V}}_{p,1} = \{v_1, v_2, v_6, v_7, v_{11}, v_{12}, v_{16}, v_{17}, v_{18}, v_{19}, v_{21}, v_{22}, v_{23}\}$ $\bar{\mathcal{V}}_{p,2} = \{v_4, v_5, v_9, v_{10}\}$	$\bar{\mathcal{V}}_{s,1} = \{v_3, v_8, v_{13}\}$ $\bar{\mathcal{V}}_{s,2} = \{v_{14}, v_{20}, v_{24}, v_{25}\}$ $\bar{\mathcal{V}}_{s,3} = \{v_{15}\}$
$\bar{\mathcal{G}} = (\mathcal{V}, \mathcal{E} \cup \{(v_{13}, v_{18})\})$	7	2	5	$\bar{\mathcal{V}}_{p,1} = \{v_1, v_2, v_6, v_7\}$ $\bar{\mathcal{V}}_{p,2} = \{v_4, v_5, v_9, v_{10}\}$	$\bar{\mathcal{V}}_{s,1} = \{v_3, v_8, v_{13}, v_{18}, v_{19}, v_{23}\}$ $\bar{\mathcal{V}}_{s,2} = \{v_{14}, v_{20}, v_{24}, v_{25}\}$ $\bar{\mathcal{V}}_{s,3} = \{v_{15}\}$ $\bar{\mathcal{V}}_{s,4} = \{v_{17}, v_{22}\}$ $\bar{\mathcal{V}}_{s,5} = \{v_{11}, v_{12}, v_{16}, v_{21}\}$
$\bar{\mathcal{G}} = (\mathcal{V}, \mathcal{E} \cup \{(v_8, v_1)\})$	9	3	6	$\bar{\mathcal{V}}_{p,1} = \{v_6, v_7\}$ $\bar{\mathcal{V}}_{p,2} = \{v_4, v_5, v_9, v_{10}\}$ $\bar{\mathcal{V}}_{p,3} = \{v_{18}, v_{19}, v_{23}\}$	$\bar{\mathcal{V}}_{s,1} = \{v_3, v_8, v_{13}\}$ $\bar{\mathcal{V}}_{s,2} = \{v_{14}, v_{20}, v_{24}, v_{25}\}$ $\bar{\mathcal{V}}_{s,3} = \{v_{15}\}$ $\bar{\mathcal{V}}_{s,4} = \{v_1, v_2\}$ $\bar{\mathcal{V}}_{s,5} = \{v_{17}, v_{22}\}$ $\bar{\mathcal{V}}_{s,6} = \{v_{11}, v_{12}, v_{16}, v_{21}\}$
$\bar{\mathcal{G}} = (\mathcal{V}, \mathcal{E} \cup \{(v_{13}, v_{12})\})$	8	3	5	$\bar{\mathcal{V}}_{p,1} = \{v_1, v_2, v_6, v_7\}$ $\bar{\mathcal{V}}_{p,2} = \{v_4, v_5, v_9, v_{10}\}$ $\bar{\mathcal{V}}_{p,3} = \{v_{18}, v_{19}, v_{23}\}$	$\bar{\mathcal{V}}_{s,1} = \{v_3, v_8, v_{13}\}$ $\bar{\mathcal{V}}_{s,2} = \{v_{14}, v_{20}, v_{24}, v_{25}\}$ $\bar{\mathcal{V}}_{s,3} = \{v_{15}\}$ $\bar{\mathcal{V}}_{s,4} = \{v_{17}, v_{22}\}$ $\bar{\mathcal{V}}_{s,5} = \{v_{11}, v_{12}, v_{16}, v_{21}\}$
$\bar{\mathcal{G}} = (\mathcal{V}, \mathcal{E} \cup \{(v_{17}, v_{16})\})$	9	3	6	$\bar{\mathcal{V}}_{p,1} = \{v_1, v_2, v_6, v_7\}$ $\bar{\mathcal{V}}_{p,2} = \{v_4, v_5, v_9, v_{10}\}$ $\bar{\mathcal{V}}_{p,3} = \{v_{18}, v_{19}, v_{23}\}$	$\bar{\mathcal{V}}_{s,1} = \{v_3, v_8, v_{13}\}$ $\bar{\mathcal{V}}_{s,2} = \{v_{14}, v_{20}, v_{24}, v_{25}\}$ $\bar{\mathcal{V}}_{s,3} = \{v_{15}\}$ $\bar{\mathcal{V}}_{s,4} = \{v_{17}, v_{22}\}$ $\bar{\mathcal{V}}_{s,5} = \{v_{11}, v_{12}\}$ $\bar{\mathcal{V}}_{s,6} = \{v_{16}, v_{21}\}$

Tab. 2: Effect of modifications on the number of groups in the given network.

5. CONCLUSIONS

In this paper, we have investigated the topology design problem for group consensus in a multi-agent network. We have derived the minimum number of links required to attain a directed network with a spanning tree which is one less than the number of primary layer subgraphs in the network. Furthermore, we have proposed an algorithm to design a directed network with any feasible setting from scratch and it is shown that the algorithm is optimal in the sense that the generated graph has the minimum number of links. We have also examined the effect of adding new agents to a multi-agent network and creating new links between agents on the number of groups formed when a conventional consensus-based algorithm is used. These results can be utilized to modify an existing network and obtain the desired grouping without redetecting the primary and secondary layer subgraphs of the modified graph by an algorithm. Numerical simulations are provided to illustrate the effectiveness of the proposed algorithms and the theoretical results of the paper.

While we have extensively investigated the effect of adding vertices and edges to the graph on the number of groups in the network, the effect of their removal remains an important issue for future research. We hope that such research will pave the way to fully understand the grouping behavior of multi-agent systems utilizing consensus-based algorithms.

APPENDICES

Algorithm 2 Algorithm to determine primary layer subgraphs.

procedure $[\mathcal{G}_{p,1}, \ldots, \mathcal{G}_{p,l_p}] = \textsc{PrimaryLayer}(\mathcal{G}(\mathcal{V}, \mathcal{E}))$
  $n \leftarrow$ the number of vertices in $\mathcal{G}$
  for $i \leftarrow 1, n$ **do**
    $\mathcal{S}_{\mathcal{G},v_i} \leftarrow$ the set of vertices reachable from v_i in $\mathcal{G}$
  end for
  $\mathcal{L}_p \leftarrow \{v_1, \ldots, v_n\}$
  for $i \leftarrow 1, n$ **do**
    for $j \leftarrow 1, n$ **do**
      if $i \neq j$ and $\mathcal{S}_{\mathcal{G},v_i} \subseteq \mathcal{S}_{\mathcal{G},v_j}$ **then**
        $\mathcal{L}_p \leftarrow \mathcal{L}_p \setminus \{v_i\}$
      end if
    end for
  end for
  $k \leftarrow 0$
  for all $v_i \in \mathcal{L}_p$ **do**
    $k \leftarrow k + 1$
    $\mathcal{V}_{p,k} \leftarrow \mathcal{S}_{\mathcal{G},v_i} \setminus \bigcup_{j \neq i} \mathcal{S}_{\mathcal{G},v_j}$
    $\mathcal{E}_{p,k} \leftarrow \mathcal{E} \cap (\mathcal{V}_{p,k} \times \mathcal{V}_{p,k})$
    $\mathcal{G}_{p,k} \leftarrow (\mathcal{V}_{p,k}, \mathcal{E}_{p,k})$
    $r_{\mathcal{G}_{p,k}} \leftarrow v_i$
  end for
  $l_p \leftarrow k$
  $r \leftarrow [r_{\mathcal{G}_{p,1}}, \ldots, r_{\mathcal{G}_{p,l_p}}]$
  return $[\mathcal{G}_{p,1}, \ldots, \mathcal{G}_{p,l_p}, r]$
end procedure

Algorithm 3 Algorithm to determine secondary layer subgraphs.

procedure $[\mathcal{G}_{s,1},\ldots,\mathcal{G}_{s,l_s},\Upsilon_{\mathcal{G}_{s,1}},\ldots,\Upsilon_{\mathcal{G}_{s,l_s}}]$ = SecondaryLayer$(\mathcal{G}(\mathcal{V},\mathcal{E}),\mathcal{G}_{p,1},\ldots,\mathcal{G}_{p,l_p},r)$
  $l \leftarrow 0$
  $\mathcal{V}_{s,1} \leftarrow \emptyset$
  $\bar{\mathcal{V}} = \bigcup \mathcal{V}_{p,i}$
  while $\bar{\mathcal{V}} \neq \mathcal{V}$ **do**
    $\mathcal{L}_s \leftarrow \{v_k : (v_j,v_k) \in \mathcal{E}, \forall v_j \in \bar{\mathcal{V}} \text{ and } \forall v_k \notin \bar{\mathcal{V}}\}$
    for all $v_i \in \mathcal{L}_s$ **do**
      $\tilde{\mathcal{G}}_{v_i} = (\tilde{\mathcal{V}}_{v_i},\tilde{\mathcal{E}}_{v_i}) \leftarrow$ the graph obtained by removing all edges associated with the vertices $\mathcal{L}_s \setminus v_i$
    end for
    for all $v_i \in \mathcal{L}_s$ **do**
      $\mathcal{S}_{\tilde{\mathcal{G}}_{v_i},v_i} \leftarrow$, the set of vertices reachable from v_i in $\tilde{\mathcal{G}}_{v_i}$
    end for
    for all $v_i \in \mathcal{L}_s$ **do**
      $l \leftarrow l+1$
      $\mathcal{V}_{s,l} \leftarrow \mathcal{S}_{\tilde{\mathcal{G}}_{v_i},v_i} \setminus \bigcup_{i\neq j} \mathcal{S}_{\tilde{\mathcal{G}}_{v_j},v_j}$
      $\mathcal{E}_{s,l} \leftarrow \mathcal{E} \cap (\mathcal{V}_{s,l} \times \mathcal{V}_{s,l})$
      $\mathcal{G}_{s,l} \leftarrow (\mathcal{V}_{s,l},\mathcal{E}_{s,l})$
      $\bar{\mathcal{V}} \leftarrow \bar{\mathcal{V}} \cup \mathcal{V}_{s,l}$
      $\Upsilon_{\mathcal{G}_{s,l}} \leftarrow \{\mathcal{G}_{p,i} : v_i \text{ is reachable from } r_{\mathcal{G}_{p,i}} \text{ in } \mathcal{G}\}$
    end for
  end while
  $l_s \leftarrow l$
  return $[\mathcal{G}_{s,1},\ldots,\mathcal{G}_{s,l_s},\Upsilon_{\mathcal{G}_{s,1}},\ldots,\Upsilon_{\mathcal{G}_{s,l_s}}]$
end procedure

ACKNOWLEDGEMENT

This work was sponsored by Scientific and Technological Research Council of Turkey (TUBITAK) under Grant 117E204.

(Received December 24, 2019)

REFERENCES

[1] J. Alonso-Mora, E. Montijano, T. Nägeli, O. Hilliges, M. Schwager, and D. Rus: Distributed multi-robot formation control in dynamic environments. Auton. Robot. *43* (2018), 1079–1100. DOI:10.1007/s10514-018-9783-9

[2] N. Amelina, A. Fradkov, Y. Jiang, and D. J. Vergados: Approximate consensus in stochastic networks with application to load balancing. IEEE Trans. Inform. Theory *61* (2015), 1739–1752. DOI:10.1109/tit.2015.2406323

[3] R. Aragues, J. Cortes, and C. Sagues: Distributed consensus on robot networks for dynamically merging feature-based maps. IEEE Trans. Robot. *28* (2012), 840–854. DOI:10.1109/tro.2012.2192012

[4] Y. Cao, D. Stuart, W. Ren, and Z. Meng: Distributed containment control for multiple autonomous vehicles with double-integrator dynamics: Algorithms and experiments. IEEE Trans. Control Syst. Technol. *19*, (2011), 929–938. DOI:10.1109/tcst.2010.2053542

[5] Z. Chen, Y. Xing, and H. Qin: Multiagent opinion dynamics influenced by individual susceptibility and anchoring effect. Kybernetika *55* (2019), 714–726. DOI:10.14736/kyb-2019-4-0714

[6] H.-L. Choi, L. Brune, and J. How: Consensus-based decentralized auctions for robust task allocation. IEEE Trans. Robot. *25* (2009), 912–926. DOI:10.1109/tro.2009.2022423

[7] Ü. Develer and M. Akar: Cluster consensus in first and second-order continuous-time networks with input and communication delays. Int. J. Control (2019).

[8] D. V. Dimarogonas and K. J. Kyriakopoulos: On the rendezvous problem for multiple nonholonomic agents. IEEE Trans. Automat. Control *52* (2007), 916–922. DOI:10.1109/tac.2007.895897

[9] Ö. F. Erkan, O. Cihan, and M. Akar: Distributed consensus with multi-equilibria in directed networks. In: 2017 American Control Conference, Seattle 2017. DOI:10.23919/acc.2017.7963678

[10] Ö. F. Erkan, O. Cihan, and M. Akar: Analysis of distributed consensus protocols with multi-equilibria under time-delays. J. Franklin Inst. *355* (2018), 332–360. DOI:10.1016/j.jfranklin.2017.10.028

[11] R. Hegselmann and U. Krause: Opinion dynamics and bounded confidence: Models, analysis and simulation. J. Artif. Soc. Soc. Simul. *5* (2002).

[12] J. Hu: Bipartite consensus control of multiagent systems on coopetition networks. Abstr. Appl. Anal. *Article ID: 689070* (2014), 1–9. DOI:10.1155/2014/689070

[13] J. Hu and W.-X. Zheng: Emergent collective behaviors on coopetition networks. Phys. Lett. A *378* (2014), 1787–1796. DOI:10.1016/j.physleta.2014.04.070

[14] J. Jin and N. Gans: Collision-free formation and heading consensus of nonholonomic robots as a pose regulation problem. Rob. Auton. Syst. *95* (2017), 25–36. DOI:10.1016/j.robot.2017.05.008

[15] M. Mirzaei, H. Atrianfar, N. Mehdipour, and F. Abdollahi: Asynchronous consensus of continuous-time lagrangian systems with switching topology and non-uniform time delay. Rob. Auton. Syst. *83* (2016), 106–114. DOI:10.1016/j.robot.2016.05.014

[16] S. Mou, J. Liu, and A. S. Morse: A distributed algorithm for solving a linear algebraic equation. IEEE Trans. Automat. Control *60* (2015), 2863–2878. DOI:10.1109/tac.2015.2414771

[17] I. Navarro and F. Matía: Distributed orientation agreement in a group of robots. Auton. Robot. *33* (2012), 445–465. DOI:10.1007/s10514-012-9300-5

[18] R. Olfati-Saber and R. M. Murray: Consensus problems in networks of agents with switching topology and time-delays. IEEE Trans. Automat. Control *49* (2004), 1520–1533. DOI:10.1109/tac.2004.834113

[19] W. Ren and R. Beard: Consensus seeking in multiagent systems under dynamically changing interaction topologies. IEEE Trans. Automat. Control *50* (2005), 655–661. DOI:10.1109/tac.2005.846556

[20] L. Schenato and F. Fiorentin: Average TimeSynch: A consensus-based protocol for clock synchronization in wireless sensor networks. Automatica *47* (2011), 1878–1886. DOI:10.1016/j.automatica.2011.06.012

[21] Z. Xu and X. Cai: Group consensus algorithms based on preference relations. Inform. Sci. *181* (2011), 150–162. DOI:10.1016/j.ins.2010.08.002

[22] S. Yang, S. Tan, and J.-X. Xu: Consensus based approach for economic dispatch problem in a smart grid. IEEE Trans. Power Syst. *28* (2013), 4416–4426. DOI:10.1109/tpwrs.2013.2271640

[23] D. Zelazo, S. Schuler, and F. Allgöwer: Performance and design of cycles in consensus networks. Syst. Control. Lett. *62* (2013), 85–96. DOI:10.1016/j.sysconle.2012.10.014

[24] H.-T. Zhang, Z. Chen, and X. Mo: Effect of adding edges to consensus networks with directed acyclic graphs. IEEE Trans. Automat. Control *62* (2017), 4891–4897. DOI:10.1109/tac.2017.2692527

[25] X. Zhang, Z. Peng, S. Yang, G. Wen, and A. Rahmani: Distributed fixed-time consensus-based formation tracking for multiple nonholonomic wheeled mobile robots under directed topology. Int. J. Control (2019). DOI:10.1080/00207179.2019.1590646

[26] Q. Zhu, X. Wang, and Q. Lin: Consensus-based impact-time-control guidance law for cooperative attack of multiple missiles. Kybernetika *53* (2017), 563–577. DOI:10.14736/kyb-2017-4-0563

Onur Cihan, Department of Electrical and Electronics Engineering, Faculty of Engineering, Marmara University, 34722, Istanbul. Turkey.
e-mail: onur.cihan@marmara.edu.tr