

Article

Autonomous Addition of Agents to an Existing Group Using Genetic Algorithm

Sabyasachi Mondal *  and Antonios Tsourdos 

School of Aerospace, Transport and Manufacturing (SATM), Cranfield University, Cranfield MK430AL, UK;
a.tsourdos@cranfield.ac.uk

* Correspondence: sabyasachi.mondal@cranfield.ac.uk

Received: 21 October 2020; Accepted: 3 December 2020; Published: 5 December 2020



Abstract: This paper presents an idea of how new agents can be added autonomously to a group of existing agents without changing the existing communication topology among them. Autonomous agent addition to existing Multi-Agent Systems (MASs) can give a strategic advantage during the execution of a critical beyond visual line-of-sight (BVLOS) mission. The addition of the agent essentially means that new connections with existing agents are established. It is obvious that the consensus control energy increases as the number of agent increases considering a specific consensus protocol. The objective of this work is to establish the new connections in a way such that the consensus energy increase due to the new agents is minimal. The updated topology, including new connections, must contain a spanning tree to maintain the stability of the MASs network. The updated optimal topology is obtained by solving minimum additional consensus control energy using the Two-Dimensional Genetic Algorithm. The results obtained are convincing.

Keywords: distributed control; optimal topology; consensus; two-dimensional genetic algorithm; autonomous mission; BVLOS

1. Introduction

In recent times, drones have been found to be useful for many applications such as in agriculture, delivery services, transportation, and search and rescue. Many complex operations involved in these applications must be executed by Multi-Agent Systems (MASs) in which the agents perform tasks cooperatively. Irrespective of the application, the agents must achieve a consensus to continue operations cooperatively. A considerable amount of work on the consensus problem has been reported in the literature. Other examples of collective behaviour of cooperative platforms are formation control [1,2], Synchronization [3,4], and consensus tracking [5].

In the majority of applications, drones are usually operated by remote pilots. Therefore, the range of operation is limited to visual line-of-sight (VLOS). However, the demand for beyond VLOS (BVLOS) operation is increasing rapidly. The implementation of autonomy is important to reduce the dependency on the remote pilot and to achieve BVLOS operation. Therefore, autonomous operation of drones is an important area of research. It can be noted that autonomous operation of MASs depends on consensus building and information sharing among the agents, which must be appropriately connected (i.e., have a well defined communication topology). The consensus is achieved by designing various consensus protocols. Generally, these protocols are designed considering the communication topology among agents, agent dynamics, information on the states of neighbours, and communication-related issues such as switching topology, communication delay, and noise. In many research works, the consensus protocols have been designed considering these factors. A few of them are mentioned here. In [6], the distributed consensus tracking problem was addressed for multi-agent systems with Lipschitz-type node dynamics. The consensus tracking problem has been

solved with changing the topology among followers. [5] describes the consensus tracking problem where the agents track a time-varying reference state. The agents have second-order nonlinear dynamics. The communication topology is considered to change with time. [7] addressed the consensus problem for heterogeneous multi-agent systems with first- and second-order dynamics. The agents are subjected to probabilistic link failure. The random nature of these failures is represented using a Bernoulli probability sequence. [8] considers directed and switching communication topology for solving the leaderless consensus problem of multi-agent systems with Lipschitz nonlinear dynamics. In [9], the average consensus problem in distributed multi-agent systems was discussed. The authors considered undirected and connected graphs to represent the communication topology among the agents. They also considered time delays in the communication channel. The work in [10] focused on asynchronous consensus of multiple agents with double-integrator dynamics. It assumed arbitrary sampling intervals and communication delays. In [11], the consensus problem of agents with single integrator dynamics was considered. However, the communication topology was assumed to be fixed. The work in [12] described a mean square consensus protocol for multi-agent systems with linear dynamics. The communication topologies among the agents were fixed, and channel noise was present. [13] describes the consensus of multi-agent systems with communication noise considering linear dynamics. A consensus algorithm for multi-agent communication with noise can be found in [14]. Ref. [15] also reports consensus with communication noise in multi-agent systems (MASs) with stochastic dynamics.

In a real-world scenario, the objectives of an ongoing mission change and more agents need to be involved along with the existing group. To make these additional agents part of the existing network, new connections must be established with them. Moreover, for a BVLOS mission, the connections must be established autonomously without the involvement of a remote pilot. Therefore, the autonomous establishment of new connections between new and existing agents or autonomous agent addition is crucial in the context of BVLOS operations. In this paper, the authors present an approach for the autonomous addition of new agents to an existing MASs in an optimal manner, which is extremely important to design an energy-efficient autonomous UAV platform for BVLOS operations (Figure 1).

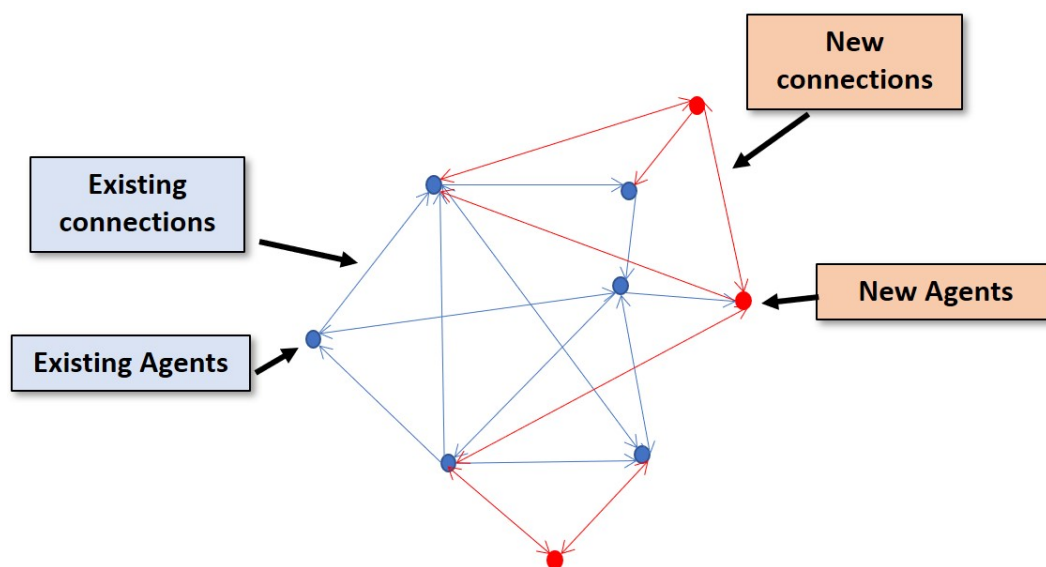


Figure 1. Pictorial representation of the problem. The new agents need to be added by establishing new connections with existing agents while maintaining the current topology.

One of the important advantages of this approach is that the agent addition will not affect the existing topology among the agents, i.e., no need to reconfigure or modify the existing topology. Reconfiguration of the existing topology requires the cancellation of the existing one and new connections are established among all the agents (including new agents) according to a new topology.

This is impossible because cancelling and establishing new connections in the middle of a mission may lead to instability among the agents and loss of agent information. The optimal manner means the addition of agents would be in such a way that the consensus of all the agents (including the new agents) can be achieved by spending the minimum additional cost, i.e., control energy. It has been mentioned earlier that the consensus of the agents depends on the communication topology. This is due to the fact that the consensus protocol is designed considering the communication topology, i.e., the consensus control expression (for example, Equation (11)) contains the communication status among the agents (described by the matrix A). It is important to note that the performance of these consensus protocols is measured by consensus control energy, which is the amount of control energy (a quadratic function of the consensus protocol) that the agents should spend to achieve the consensus. The addition of the agents essentially means that new connections with existing agents are established. It is obvious that the consensus control energy increases as the number of agent increases, considering a specific consensus protocol. The objective of this work is to establish the new connections in such a way that the increase in the consensus energy due to the new agents is minimal. In other words, the new connections will be established in a manner such that the consensus control energy of all the agents (including new agents) increases as little as possible compared to the cost required for the consensus of existing agents.

The contribution in this work is summarized as follows.

- Autonomous agent addition to an existing group of agents is a new idea and has not been reported in the literature (to the best of authors' knowledge). It is an important operation when an ongoing mission demands more agents. The idea helps to build a distributed architecture which is appropriate for BVLOS operations. The agents can operate independently if more agents are required to join the group. There is no need to depend on remote pilots.
- The existing communication topology is not modified. This is important because the existing agents remain connected during the flight. Therefore, network stability is assured. The minimum additional consensus control is required. This is necessary to ensure a minimum number of new connections should be established. In addition, the MASs can operate for a longer duration, which is beneficial for a mission.
- A bio-inspired optimization technique is used to solve the problem which is new in this context. A new crossover and mutation variety is proposed.

The remainder of the paper is organized as follows. The problem description is given in Section 2. In Section 3, preliminaries are presented. The solution method is discussed in Section 4. The results are shown in Section 5. Finally, the conclusion is given in Section 6.

2. Problem Description

The problem can be explained with the help of an example. Let us consider a group of N agents assigned to a mission. They are connected by a communication topology which is shown by an adjacency matrix $A = [a_{ij}]$; $i, j = 1, 2, \dots, N$ is shown in Figure 2. The elements $a_{ij} = 0$ or 1 , $\forall i \neq j$ and $a_{ii} = 0, \forall i = j$.

Let us consider a situation where the mission demands more agents and a group of n new agents (not connected to each other) are required to add to the existing agents. The agents can be added by establishing connections among the existing and new agents, i.e., reconfiguring or modifying the existing communication topology. There are two possible ways to reconfigure the existing topology to accommodate the new agents. The first one is to configure a fresh topology among all the agents. The second one is to connect the new agents to existing agents arbitrarily. In the first way, the existing topology is cancelled, and new connections are established among all the agents (including new agents) according to a new topology. This is not possible because cancelling and establishing new connections may lead to instability of the existing graph topology. This may also cause the loss of agent information.

a_{11}	a_{12}	a_{13}	a_{14}	...	a_{1N}
a_{21}	a_{22}	a_{23}	a_{24}	...	a_{2N}
a_{31}	a_{32}	a_{33}	a_{34}	...	a_{3N}
a_{41}	a_{42}	a_{43}	a_{44}	...	a_{4N}
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
a_{N1}	a_{N2}	a_{N3}	a_{N4}	...	a_{NN}

Figure 2. Adjacency matrix A defining the existing topology.

In the second way, new connections are established among the new and existing agents. The pictorial representation is given in Figure 3.

a_{11}	a_{12}	a_{13}	...	a_{1N}	$a_{1(N+1)}$	...	$a_{1(N+n)}$
a_{21}	a_{22}	a_{23}	...	a_{2N}	$a_{2(N+1)}$	...	$a_{2(N+n)}$
a_{31}	a_{32}	a_{33}	...	a_{3N}	$a_{3(N+1)}$	...	$a_{3(N+n)}$
a_{41}	a_{42}	a_{43}	...	a_{4N}	$a_{4(N+1)}$	...	$a_{4(N+n)}$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
a_{N1}	a_{N2}	a_{N3}	...	a_{NN}	$a_{N(N+1)}$	...	$a_{N(N+n)}$
$a_{(N+1)1}$	$a_{(N+1)2}$	$a_{(N+1)3}$	...	$a_{(N+1)N}$	$a_{(N+1)(N+1)}$	...	$a_{(N+1)(N+n)}$
$\vdots$	$\vdots$	$\vdots$	...	$\vdots$	$\vdots$	$\ddots$	$\vdots$
$a_{(N+n)1}$	$a_{(N+n)2}$	$a_{(N+n)3}$	...	$a_{(N+n)N}$	$a_{(N+n)(N+1)}$	...	$a_{(N+n)(N+n)}$

Figure 3. Modified Adjacency matrix A_{mod} defining the modified topology which includes new agents.

Comparing Figure 2 with Figure 3, it can be observed that the new connections among the new and the existing agents are denoted by the values of the elements a_{ij} , ($i = N + 1, \dots, N + n$; $j = 1, 2, \dots, N + n$) and ($i = 1, \dots, N$; $j = N + 1, \dots, N + n$). These elements are shown in red in Figure 3. The binary values of these elements (except the diagonal elements) are appended to the existing adjacency matrix A to obtain the modified adjacency matrix A_{mod} . However, the assignment of values of the elements (in red colour) corresponding to new agents, cannot be done randomly. The elements should be assigned such that the resulting topology contains a spanning tree, and the consensus can be

achieved. Moreover, the resulting topology may not be optimal. The optimal topology is the topology in which the agents if connected, require minimum control energy to achieve the consensus.

In the context of control energy about the consensus of MASs, it is important to note that the same consensus protocol can produce different control energies for the agents if the topology varies. Therefore, the control energy can be minimized with respect to the topologies. The topology corresponding to the minimum energy is addressed as the optimal topology (see Figure 4 for details). A similar problem has been addressed in [16–18]. In these papers, the Linear Quadratic Regulator (LQR) was used to obtain a complete graph for the homogeneous systems. A similar problem is presented in [19], where heterogeneous agents are considered. The optimal topology obtained is a star graph which shows the direct links between the leader and the followers. In [20], the authors presented the optimal topology problem and solved it using a Two-Dimensional Genetic Algorithm (2D-GA), which relaxes the requirement of direct connection among the leader and the followers.

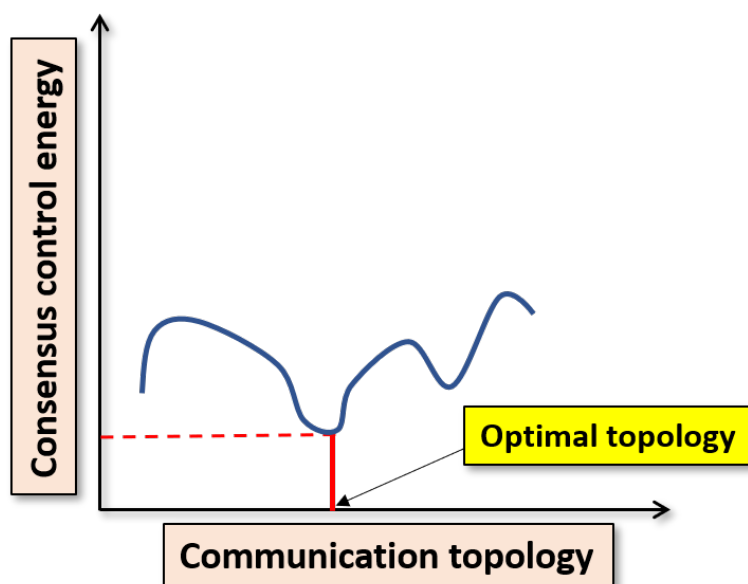


Figure 4. Problem description: Control energy varies with communication topology. (using any consensus protocol).

It is important to note that an increase in the number of agents in a MASs will result in an increase in the consensus control energy. In this problem, let us consider the consensus control of i th agent is u_i , consensus energy of the existing M agents is $J = \sum_{i=1}^M u_i^T u_i$, and the required unknown consensus control energy for the modified MASs (including the new agents) be J_{mod} .

It is obvious that the consensus control energy of the agents will increase with the increase in the number of agents, i.e., $J_{mod} > J$. The objective of this paper is to minimize the difference $\Delta J = J_{mod} - J$ (additional consensus control energy). Since J is constant, minimization of ΔJ is equivalent to minimizing J_{mod} to find the minimum J_{mod} , i.e., J_{mod}^* such that the difference ΔJ becomes minimal, i.e., $\Delta J^* = J_{mod}^* - J$. The problem description is given in Figure 5.

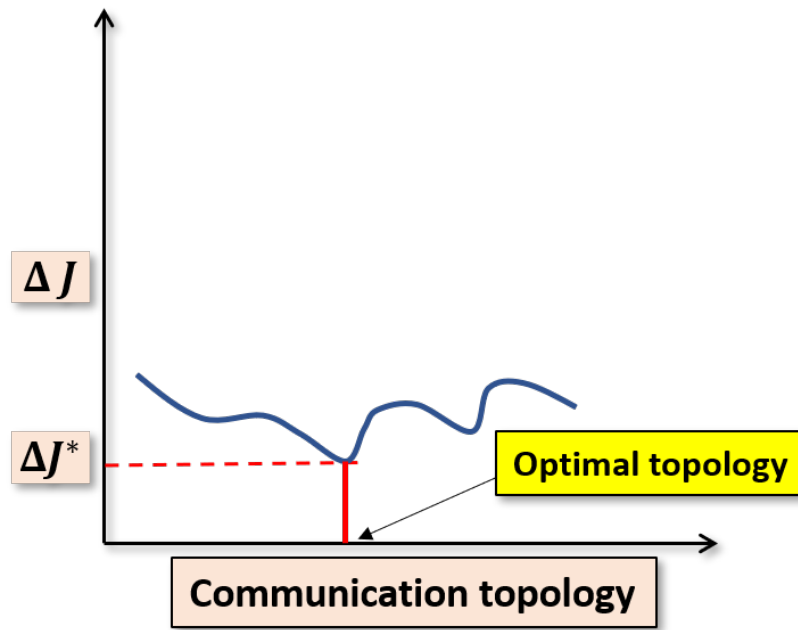


Figure 5. Problem description: Additional consensus control energy ΔJ varies with communication topology.

3. Preliminaries

The preliminaries required for the research work are presented in the following section.

3.1. Consensus of Agents

The consensus of MASs on a communication network is discussed in this section. The definition of the consensus is given as follows.

Definition 1. Let us consider a MASs with N agents, where X_i , ($i = 1, 2, 3, \dots, N$) denotes the states of the i th agent. The MASs will achieve the consensus if $\|X_i - X_j\| \rightarrow 0, \forall i \neq j$ as $t \rightarrow +\infty$.

The primary goal of designing a consensus protocol is to minimize the error in similar states of each individual agent with its neighbour by exchanging information among them through the communication network, which is designed and explained by graph theory.

3.2. Graph Theory

The communication among the agents is designed using graph theory. The networked MASs is represented by a weighted directed graph written by $G = \{V, E\}$. The vertices $V = \{1, 2, \dots, N\}$ of the graph denotes the agents and the set of edges is denoted by $E \subseteq V \times V$ represents the communication among the agents. e_{ij} denotes the information flow along the edge from j to i . The neighbour of agent i is denoted by $N_i = \{j \in V : (i, j) \in E\}$. The Adjacency matrix $A = [a_{ij}] \in \mathbb{R}^{N \times N}$ denotes the connectivity among the nodes or agents. a_{ij} denotes the elements of A . There is no self loop in the graph. This fact is expressed by selecting the diagonal elements of the adjacency matrix A as zero, i.e., $i \in V$, $a_{ii} = 0$. The off-diagonal elements, i.e., $\forall i \neq j, e_{ij} \in E$, $a_{ij} \in \mathbb{R}^+$ represent the weight associated to edge e_{ij} , while $a_{ij} = 0$ otherwise. The degree matrix is denoted by $D \in \mathbb{R}^{N \times N} = \text{diag}\{d_1 \ d_2 \ \dots \ d_N\}$, where $d_i = \sum_{j \in N_i} a_{ij}$. The Laplacian matrix is written as $L = D - A$. All the matrices describe the connections and properties of the Consensus among MASs.

3.3. Distributed Nonlinear Dynamic Inversion (DNDI) Controller for Consensus of MASs

There exist consensus protocols for nonlinear systems, as discussed earlier. However, in this paper, a new consensus protocol is used, which is designed using Nonlinear Dynamics Inversion

(NDI). The derivation of NDI-based control for MASs, named as DNDI, is presented in this section. In case of consensus of MASs, the reference signal or desired output of each agent is the outputs of its neighbours. The dynamics of the i th agent is given by

$$\dot{X}_i = f(X_i) + g(X_i)U_i \quad (1)$$

$$Y_i = h(X_i) \quad (2)$$

where $X_i \in \mathbb{R}^m$ is state and $Y_i \in \mathbb{R}^n$ is output. The mathematical derivation of DNDI is shown below.

First, the error associated with the scalar output is obtained. The error in output of i th agent can be written as

$$e_i = \sum_{j \in N_i} a_{ij}(Y_i - Y_j) \quad (3)$$

where j denotes the j th agent of i th agent's neighbourhood N_i . The Equation (3) can be simplified as follows

$$e_i = d_i Y_i - a_i Y \quad (4)$$

where

$$d_i = \sum_j a_{ij} \in \mathbb{R}, \quad a_i = [a_{i1} \ a_{i2} \ \dots \ a_{iN}] \in \mathbb{R}^N$$

and

$$Y = \begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_N \end{bmatrix} \in \mathbb{R}^N$$

The error in Equation (4) is written for vector output of i th agent i.e., $Y_i \in \mathbb{R}^n; n > 1$ as

$$\mathbf{e}_i = \bar{d} Y_i - \bar{a} Y \quad (5)$$

where $\bar{d} = (d_i \otimes \mathbf{I}_n) \in \mathbb{R}^{n \times n}$, $\bar{a} = (a_i \otimes \mathbf{I}_n) \in \mathbb{R}^{n \times nN}$, and $Y \in \mathbb{R}^{nN}$. $\mathbf{I}_n$ is $n \times n$ identity matrix. ' $\otimes$ ' denotes the kroneker product. The kroneker product of $A = [a_{ij}]$ and B is given by

$$A \otimes B = [a_{ij}B] \quad \forall i, j$$

We define a Lyapunov function V_i as follows

$$V_i = \frac{1}{2} \mathbf{e}_i^T \mathbf{e}_i \quad (6)$$

Differentiating Equation (6) yields

$$\dot{V}_i = \mathbf{e}_i^T \dot{\mathbf{e}}_i \quad (7)$$

According to the Lyapunov stability theory, let the time derivative of the lyapunov function be

$$\dot{V}_i = -\mathbf{e}_i^T K \mathbf{e}_i \quad (8)$$

where $K \in \mathbb{R}^{n \times n}$ is a positive definite diagonal matrix. The expression of $\dot{V}_i$ in Equations (7) and (8) are equated to obtain

$$\mathbf{e}_i^T \dot{\mathbf{e}}_i = -\mathbf{e}_i^T K \mathbf{e}_i \quad (9)$$

Equation (9) is simplified as follows

$$\dot{\mathbf{e}}_i + K \mathbf{e}_i = 0 \quad (10)$$

Differentiation of Equation (5) yields

$$\begin{aligned}\dot{\mathbf{e}}_i &= \bar{d}\dot{Y}_i - \bar{a}\dot{\mathbf{Y}} \\ &= \bar{d}[f(X_i) + g(X_i)U_i] - \bar{a}\dot{\mathbf{Y}}\end{aligned}\quad (11)$$

Substitution of the expressions for $\mathbf{e}_i$ and $\dot{\mathbf{e}}_i$ in Equation (10) gives

$$\bar{d}[f(X_i) + g(X_i)U_i] - \bar{a}\dot{\mathbf{Y}} + K(\bar{d}Y_i - \bar{a}\mathbf{Y}) = 0 \quad (12)$$

The expression of control U_i for i th agent is obtained by simplifying Equation (12) as follows

$$U_i = (g(X_i))^{-1} \left[-f(X_i) + \bar{d}^{-1}(\bar{a}\dot{\mathbf{Y}} - K(\bar{d}Y_i - \bar{a}\mathbf{Y})) \right] \quad (13)$$

The consensus control energy is given by $J = \sum_{i=1}^N u_i^T u_i$.

It is clear that the control expressions of conventional NDI are different from what is obtained for the consensus of MASs. This control expression in Equation (13) was used to generate the results in this paper.

4. Solution Method: Two-Dimensional Genetic Algorithm (2D-GA)

A genetic algorithm (GA) is a class of bio-inspired algorithms that can produce the optimal or near-optimal solution of complex optimization problems in a reasonable time. It was first proposed by Holland [21], and inspired by Darwin's principle of survival of the fittest. The possible solution of an optimization problem is encoded in a chromosome which consists of an array of bits called genes. The individual chromosome is evaluated by a fitness function. A genetic population consists of a finite number of chromosomes. The chromosomes of the new population are generated by the application of genetic operations such as crossover, mutation, and reproduction on the present population. The new population optimizes the fitness function and thus provides an improved solution. The solution thus approaches the optimal solution over several generations. There exist many applications of GA viz. optimization [22,23], machine learning [24], neural networks [25], fuzzy logic controllers [26], identification [27], fault diagnosis [28], path planning [29], consensus [30], and financial market [31].

The optimization problem mentioned in the previous section cannot be solved by conventional GA. The solution to this problem can be obtained using a two-dimensional genetic algorithm. The chromosome of 2D-GA is a matrix which is appropriate to represent an adjacency matrix [20]. Therefore, in this paper, the adjacency matrix is considered as a chromosome. Depending on the application, the two-dimensional chromosome represents a different solution. For example, the time table or schedule is considered as a 2D chromosome in a flight scheduling problem in [32]. Also, it is used in the packing problem [33], which aims to obtain a high packing density. The two-dimensional chromosome representation is discussed in the following section.

4.1. Two-Dimensional Chromosome Representation

The chromosome for this problem is the modified adjacency matrices A_{mod} , which is discussed in Section 3. In [20], the authors considered the adjacency matrix as a 2D chromosome. An example of such a chromosome is given in Figure 1. As explained in Section 3, the agent addition problem requires the adjacency matrix to be modified to represent the new connections among the new and existing agents. The modified adjacency matrix is shown in Figure 2. The modified adjacency matrix preserves the properties of the adjacency matrix and serves as guess solution or chromosomes for 2D-GA. The population generation is shown in the following section.

4.2. Population Generation

It is clear that the chromosomes are square as the adjacency matrix is square. Let us consider, there are N agents in the existing MASs, and n new agents need to be added. The adjacency matrix or existing communication topology is denoted by $A \in \mathbb{R}^{N \times N}$.

These new connections among existing and new agents are appended to the existing topology A to obtain the modified adjacency matrix or modified 2D chromosome A_{mod} . The k th chromosome A_{mod_k} , $k = 1, 2, \dots, N_p$ can be generated using Algorithm 1, where N_p denotes the population size. The algorithm can be explained with the help of an example. Let us consider, $N = 5$ and $n = 3$ and the existing topology as shown in Figure 6.

The algorithm produces the population Pop whose k th chromosome A_{mod_k} is shown in Figure 7.

The 6–8th rows and 1–8th columns are filled up in a random manner depending on the random value of the variable x . Similarly, the 1–5th rows and 6–8th columns are filled up. The new diagonal elements are zero. The population thus generated is subjected to crossover and mutation operations which are described in the following sections.

Algorithm 1 Initial Population Generation.

```

for  $k = 1$  to  $N_p$  do
  for  $i = N + 1$  to  $N + n$  do
    for  $j = 1$  to  $N + n$  do
       $x \leftarrow$  random number  $x \in (0, 1)$ 
      if  $x > 0.5$  then
         $A(i, j) \leftarrow 1$ 
      else
         $A(i, j) \leftarrow 0$ 
      end if
      if  $i = j$  then
         $A(i, j) \leftarrow 0$ 
      end if
    end for
  end for
  for  $i = 1$  to  $N$  do
    for  $j = N + 1$  to  $N + n$  do
       $x \leftarrow$  random number  $x \in (0, 1)$ 
      if  $x > 0.5$  then
         $A(i, j) \leftarrow 1$ 
      else
         $A(i, j) \leftarrow 0$ 
      end if
    end for
  end for
   $A_{mod_k} \leftarrow A$ 
   $Pop(:, :, k) \leftarrow A_{mod_k}$ 
end for

```

0	1	0	1	1
1	0	1	0	0
1	0	0	1	0
0	1	1	0	1
1	1	0	0	0

Figure 6. Adjacency matrix showing the existing topology.

0	1	0	1	1	0	1	1
1	0	1	0	0	1	0	0
1	0	0	1	0	0	0	1
0	1	1	0	1	1	1	0
1	1	0	0	0	0	1	1
0	1	0	0	1	0	0	1
1	0	0	1	0	1	0	0
1	0	1	0	0	1	0	0

Figure 7. Adjacency matrix showing the existing topology.

4.3. Crossover

There are a few Crossover methods that exist in the literature. Some of these methods are Multipoint Crossover [34], Uniform Crossover [35], One-Point Crossover [36], and Substring Crossover [36]. More crossover methods can be found in [37]. The crossover method mentioned in [32] is adopted in this work. These methods are presented in algorithmic form. The crossover algorithm is modified (Algorithm 2) to apply it for the modified chromosome (including new agents), as shown in Figure 3.

Algorithm 2 Crossover().

Generate random integer $r_1 \in (N, N + n)$
Generate random integer $r_2 \in (1, N + n)$
 $Block1_{p1} \leftarrow Parent1(r_1, r_2 : N + n)$
 $Block2_{p1} \leftarrow Parent1(r_1 + 1 : N + n, 1 : N + n)$
 $Block1_{p2} \leftarrow Parent2(r_1, r_2 : N + n)$
 $Block2_{p2} \leftarrow Parent2(r_1 + 1 : N + n, 1 : N + n)$
 $Block1_{p1} \rightleftharpoons Block1_{p2}$ and $Block2_{p1} \rightleftharpoons Block2_{p2}$

The algorithm can be explained using an example. Let us consider $N = 5$ and $n = 3$. A general example of existing topology $A = [a_{ij}]$, $i, j = 1, 2, \dots, 5$ is shown in Figure 8.

a_{11}	a_{12}	a_{13}	a_{14}	a_{15}
a_{21}	a_{22}	a_{23}	a_{24}	a_{25}
a_{31}	a_{32}	a_{33}	a_{34}	a_{35}
a_{41}	a_{42}	a_{43}	a_{44}	a_{45}
a_{51}	a_{52}	a_{53}	a_{54}	a_{55}

Figure 8. Adjacency matrix showing the existing topology.

The modified topology or 2D parent chromosomes are denoted by 'Parent 1', and 'Parent 2'. They are shown in Figure 9.

Parent 1							
a_{11}	a_{12}	a_{13}	a_{14}	a_{15}	a_{16}	a_{17}	a_{18}
a_{21}	a_{22}	a_{23}	a_{24}	a_{25}	a_{26}	a_{27}	a_{28}
a_{31}	a_{32}	a_{33}	a_{34}	a_{35}	a_{36}	a_{37}	a_{38}
a_{41}	a_{42}	a_{43}	a_{44}	a_{45}	a_{46}	a_{47}	a_{48}
a_{51}	a_{52}	a_{53}	a_{54}	a_{55}	a_{56}	a_{57}	a_{58}
a_{61}	a_{62}	a_{63}	a_{64}	a_{65}	a_{66}	a_{67}	a_{68}
a_{71}	a_{72}	a_{73}	a_{74}	a_{75}	a_{76}	a_{77}	a_{78}
a_{81}	a_{82}	a_{83}	a_{84}	a_{85}	a_{86}	a_{87}	a_{88}

Parent 2							
b_{11}	b_{12}	b_{13}	b_{14}	b_{15}	b_{16}	b_{17}	b_{18}
b_{21}	b_{22}	b_{23}	b_{24}	b_{25}	b_{26}	b_{27}	b_{28}
b_{31}	b_{32}	b_{33}	b_{34}	b_{35}	b_{36}	b_{37}	b_{38}
b_{41}	b_{42}	b_{43}	b_{44}	b_{45}	b_{46}	b_{47}	b_{48}
b_{51}	b_{52}	b_{53}	b_{54}	b_{55}	b_{56}	b_{57}	b_{58}
b_{61}	b_{62}	b_{63}	b_{64}	b_{65}	b_{66}	b_{67}	b_{68}
b_{71}	b_{72}	b_{73}	b_{74}	b_{75}	b_{76}	b_{77}	b_{78}
b_{81}	b_{82}	b_{83}	b_{84}	b_{85}	b_{86}	b_{87}	b_{88}

Figure 9. Adjacency matrix showing the modified topology.

According to the Algorithm 2, the random integers r_1 and r_2 are obtained as $r_1 = 7, r_2 = 3$. The selected elements around the crossover point are shown in the shaded area with a green outline in Parent 1 and blue outline in Parent 2. They are shown in Figure 10.

Parent 1								Parent 2							
a_{11}	a_{12}	a_{13}	a_{14}	a_{15}	a_{16}	a_{17}	a_{18}	b_{11}	b_{12}	b_{13}	b_{14}	b_{15}	b_{16}	b_{17}	b_{18}
a_{21}	a_{22}	a_{23}	a_{24}	a_{25}	a_{26}	a_{27}	a_{28}	b_{21}	b_{22}	b_{23}	b_{24}	b_{25}	b_{26}	b_{27}	b_{28}
a_{31}	a_{32}	a_{33}	a_{34}	a_{35}	a_{36}	a_{37}	a_{38}	b_{31}	b_{32}	b_{33}	b_{34}	b_{35}	b_{36}	b_{37}	b_{38}
a_{41}	a_{42}	a_{43}	a_{44}	a_{45}	a_{46}	a_{47}	a_{48}	b_{41}	b_{42}	b_{43}	b_{44}	b_{45}	b_{46}	b_{47}	b_{48}
a_{51}	a_{52}	a_{53}	a_{54}	a_{55}	a_{56}	a_{57}	a_{58}	b_{51}	b_{52}	b_{53}	b_{54}	b_{55}	b_{56}	b_{57}	b_{58}
a_{61}	a_{62}	a_{63}	a_{64}	a_{65}	a_{66}	a_{67}	a_{68}	b_{61}	b_{62}	b_{63}	b_{64}	b_{65}	b_{66}	b_{67}	b_{68}
a_{71}	a_{72}	a_{73}	a_{74}	a_{75}	a_{76}	a_{77}	a_{78}	b_{71}	b_{72}	b_{73}	b_{74}	b_{75}	b_{76}	b_{77}	b_{78}
a_{81}	a_{82}	a_{83}	a_{84}	a_{85}	a_{86}	a_{87}	a_{88}	b_{81}	b_{82}	b_{83}	b_{84}	b_{85}	b_{86}	b_{87}	b_{88}

Figure 10. Parents: horizontal crossover, selected genes are shown in green and blue border.

The elements a_{73} to a_{88} of Parent 1 are exchanged with b_{73} to b_{88} of Parent 2 to generate children denoted by 'Child 1' and 'Child 2', respectively (Figure 11).

Child 1								Child 2							
a_{11}	a_{12}	a_{13}	a_{14}	a_{15}	a_{16}	a_{17}	a_{18}	b_{11}	b_{12}	b_{13}	b_{14}	b_{15}	b_{16}	b_{17}	b_{18}
a_{21}	a_{22}	a_{23}	a_{24}	a_{25}	a_{26}	a_{27}	a_{28}	b_{21}	b_{22}	b_{23}	b_{24}	b_{25}	b_{26}	b_{27}	b_{28}
a_{31}	a_{32}	a_{33}	a_{34}	a_{35}	a_{36}	a_{37}	a_{38}	b_{31}	b_{32}	b_{33}	b_{34}	b_{35}	b_{36}	b_{37}	b_{38}
a_{41}	a_{42}	a_{43}	a_{44}	a_{45}	a_{46}	a_{47}	a_{48}	b_{41}	b_{42}	b_{43}	b_{44}	b_{45}	b_{46}	b_{47}	b_{48}
a_{51}	a_{52}	a_{53}	a_{54}	a_{55}	a_{56}	a_{57}	a_{58}	b_{51}	b_{52}	b_{53}	b_{54}	b_{55}	b_{56}	b_{57}	b_{58}
a_{61}	a_{62}	a_{63}	a_{64}	a_{65}	a_{66}	a_{67}	a_{68}	b_{61}	b_{62}	b_{63}	b_{64}	b_{65}	b_{66}	b_{67}	b_{68}
a_{71}	a_{72}	b_{73}	b_{74}	b_{75}	b_{76}	b_{77}	b_{78}	b_{71}	b_{72}	a_{73}	a_{74}	a_{75}	a_{76}	a_{77}	a_{78}
b_{81}	b_{82}	b_{83}	b_{84}	b_{85}	b_{86}	b_{87}	b_{88}	a_{81}	a_{82}	a_{83}	a_{84}	a_{85}	a_{86}	a_{87}	a_{88}

Figure 11. Children: horizontal crossover, selected genes are exchanged between the parents to obtain the children.

4.4. Mutation

The mutation is an important operation to preserve the genetic diversity of a population of chromosomes in every generation. The mutation is performed by exchanging one or more genes of the chromosomes. Generally, a certain percentage of the population is allowed to undergo mutation. The mutation may change the solution considerably from the previous solution. Hence, GA can arrive at a better solution by using mutations. The process for this mutation is given in Algorithm 3.

Algorithm 3 Mutation().

```

 $m_1 \leftarrow$  random integer  $[N + 1, N + m]$ 
 $m_2 (\neq m_1) \leftarrow$  random integer  $[N + 1, N + m]$ 
if  $m_1 \neq m_2$  then
    Swap  $m_1^{th}$  and  $m_2^{th}$  rows of a chromosome
end if

```

$Mutation()$ function is given in Algorithm 3. It swaps m_1^{th} and m_2^{th} rows of a chromosome. The pictorial representation of the operation is given in Figure 12. The algorithm is explained with the help of an example. Let us consider, $m_1 = 6$, and $m_2 = 8$. The selected rows are shown in green and blue lines, respectively. According to the algorithm, the sixth and eighth rows are swapped, as shown in the figure.



Figure 12. Horizontal swapping mutation.

5. Results

In this study, an example scenario is considered. The existing MASs consist of ten agents, and they are assigned a mission. Depending on the requirement, more agents need to be added to the existing group of agents. All the agents have the same dynamics, i.e., they are homogeneous in nature. The nonlinear dynamics considered for the i th agents is given by

$$\dot{X}_{i1} = X_{i2} \sin(2X_{i1}) + U_{i1} \quad (14)$$

$$\dot{X}_{i2} = X_{i1} \cos(3X_{i2}) + U_{i2} \quad (15)$$

where $X_i = [X_{i1} \ X_{i2}]^T$ and $U_i = [U_{i1} \ U_{i2}]^T$ are states and control variables, respectively. The simulation study is presented in two parts. In the first part, we show the details of the existing topology. In the second part, we will show how more agents are added to the existing agents without changing the existing topology.

5.1. Part I: Existing Topology

The existing topology in which the group of agents are connected is considered to be the optimal one [20]. The optimal topology is obtained using 2D-GA, as described in this paper. The cost (consensus control energy) generated is shown in Figure 13.

The optimal topology obtained is shown in Figure 14. This is the existing topology among the agents. The topology contains a spanning tree, which is evident from the eigenvalues shown in Figure 15. There is only one zero eigenvalue. Others have a positive real part.

The consensus is achieved by using the NDI-based controller. The control signals U_1 and U_2 are shown in Figures 16 and 17, respectively.

The trajectories generated are shown in Figures 18 and 19.

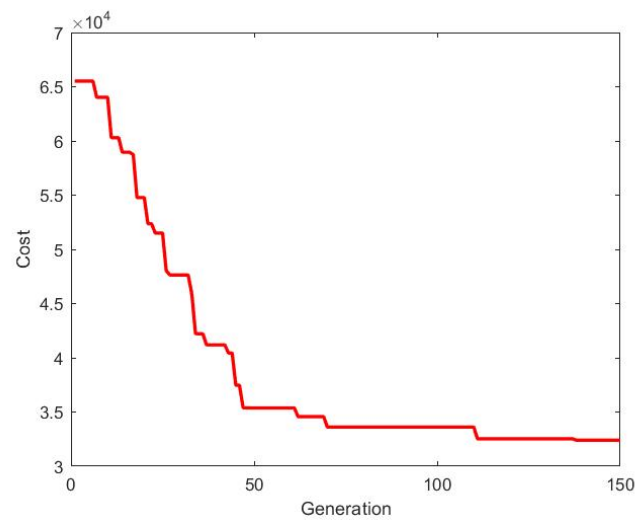


Figure 13. Cost for building initial optimal topology.

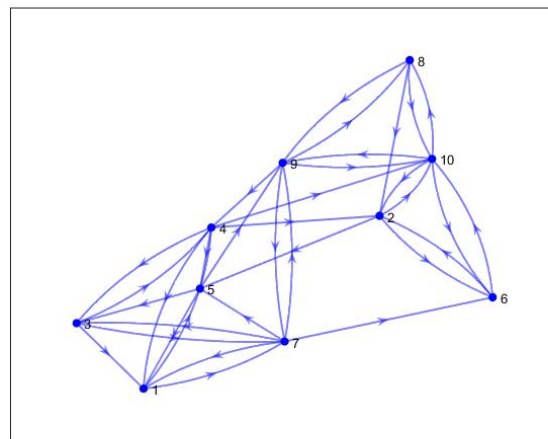


Figure 14. Optimal topology.

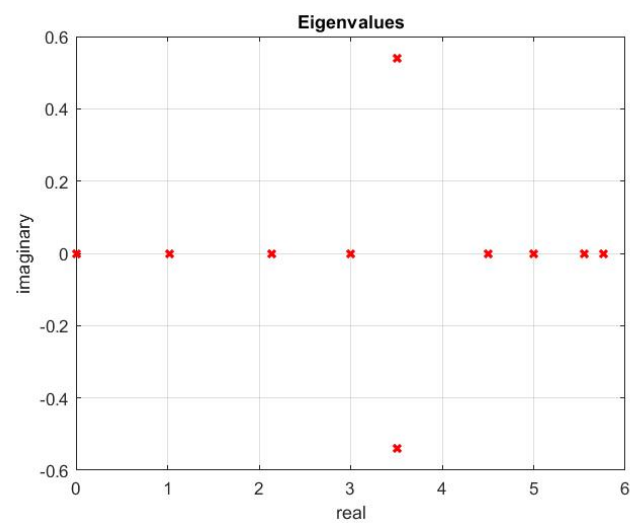


Figure 15. Eigenvalues.

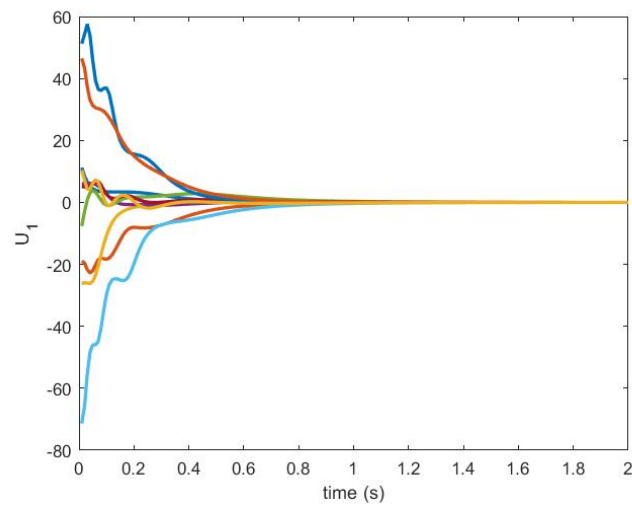


Figure 16. Control U_1 produced by an NDI-based controller.

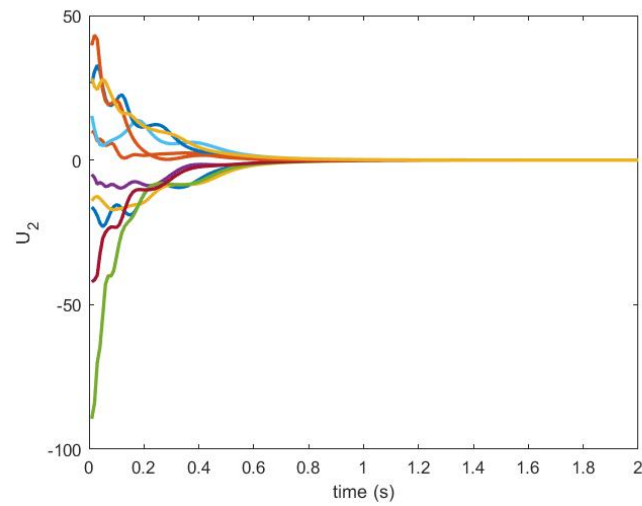


Figure 17. Control U_2 produced by an NDI-based controller.

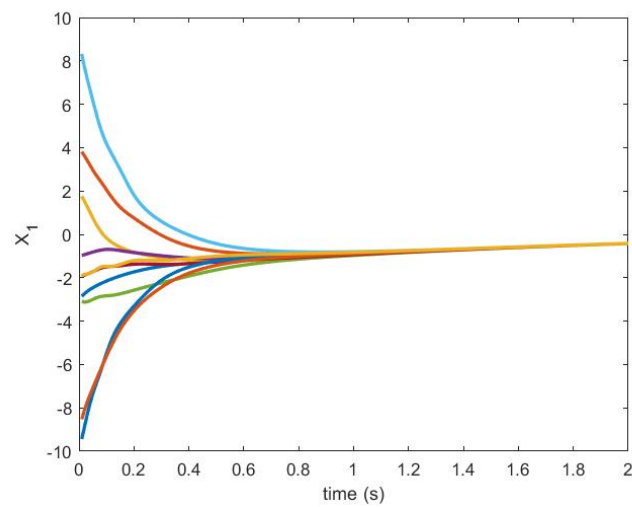


Figure 18. State trajectory X_1 .

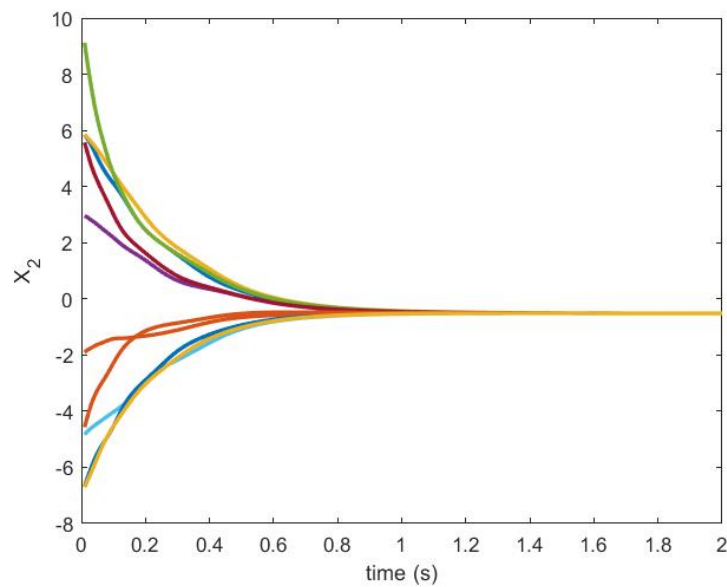


Figure 19. State trajectory X_2 .

The consensus among the agents is achieved successfully on this topology.

5.2. Part II: Agent Addition, No Change in Existing Topology

In this part, the new agents are added to the existing group of agents. The existing topology is kept unchanged. The updated topology is obtained by minimizing the consensus control energy of all the agents (including new agents) using 2D-GA. The cost obtained is shown in Figure 20.

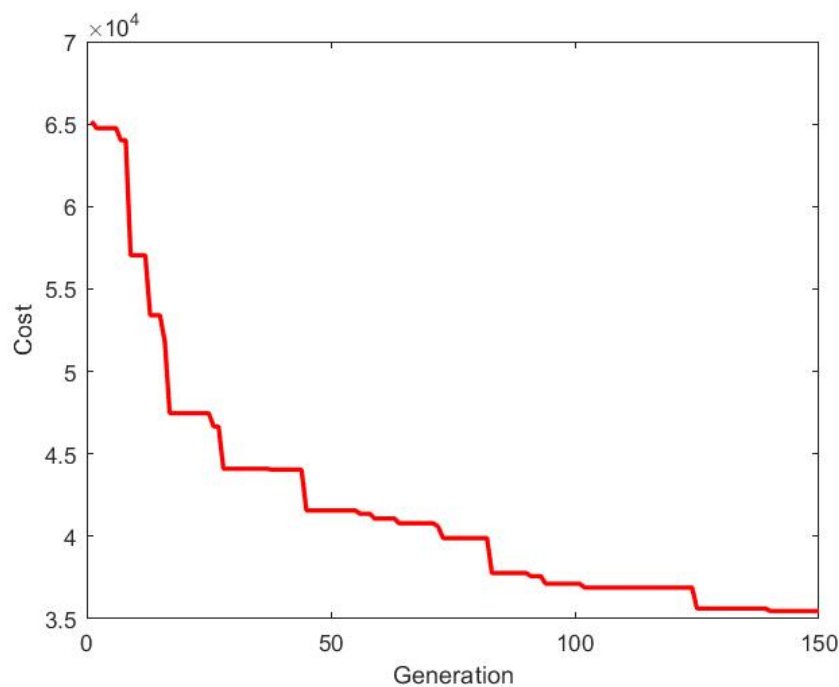


Figure 20. Updated cost.

It is obvious that the addition of new agents result in an increase in the consensus control energy. The objective of this paper is to minimize the additional control energy. The cost produced by

2D-GA to obtain the updated topology is compared with that of the existing topology in Figure 21. Also, the additional cost required to obtain the updated topology is shown in Figure 22.

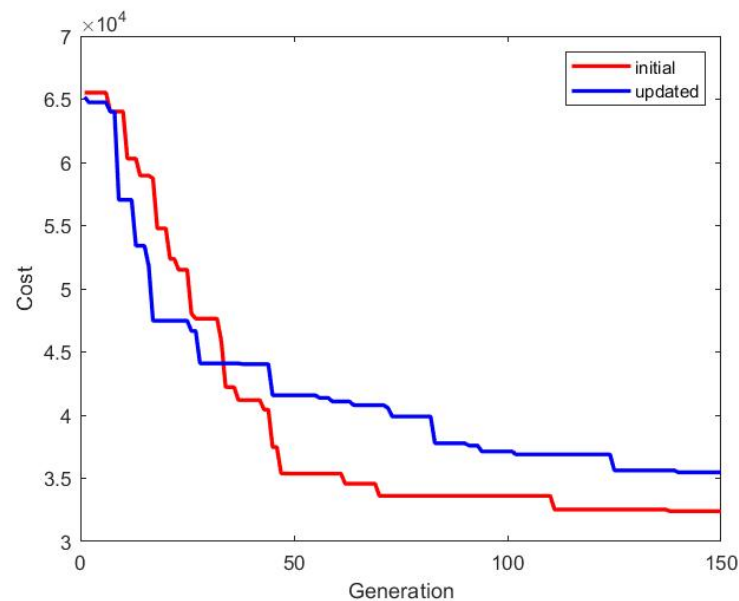


Figure 21. Comparison of costs between existing and updated topology.

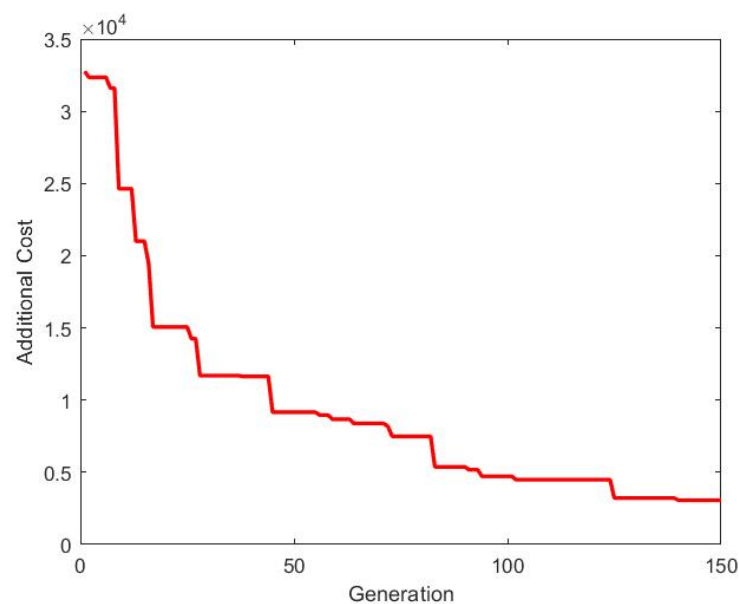


Figure 22. Additional cost.

It can be observed that the additional cost is minimized over the generations, i.e., the consensus of all the agents (including the new agents) can be achieved by spending the minimum amount of additional control energy. The updated topology is shown in Figure 23. The agents 11–13 are added to the existing agents in the updated topology. It is important to note that the existing topology (Figure 16) is unchanged.

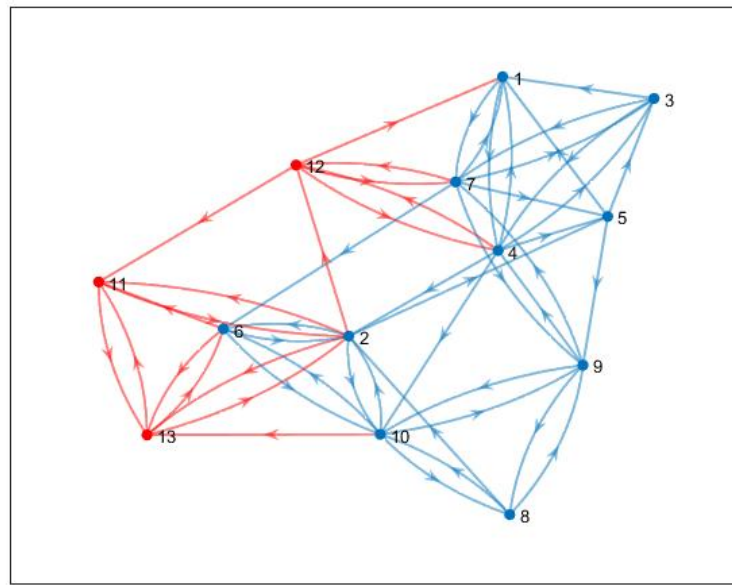


Figure 23. Updated topology, new agents are connected to existing agents via red edges.

The new connections are shown by red edges. The stability of this is assured by the eigenvalues shown in Figure 24, which explains the presence of a spanning tree in the topology. The eigenvalues of the updated topology are different from those corresponding to existing topology (Figure 16).

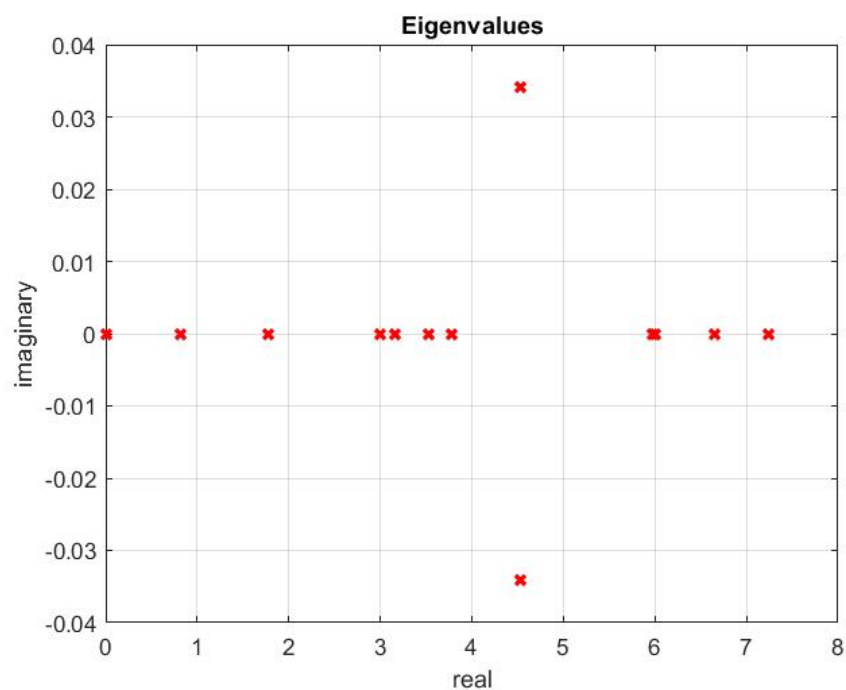


Figure 24. Updated topology

The control signals are produced by an NDI-based controller and they are shown in Figures 25 and 26.

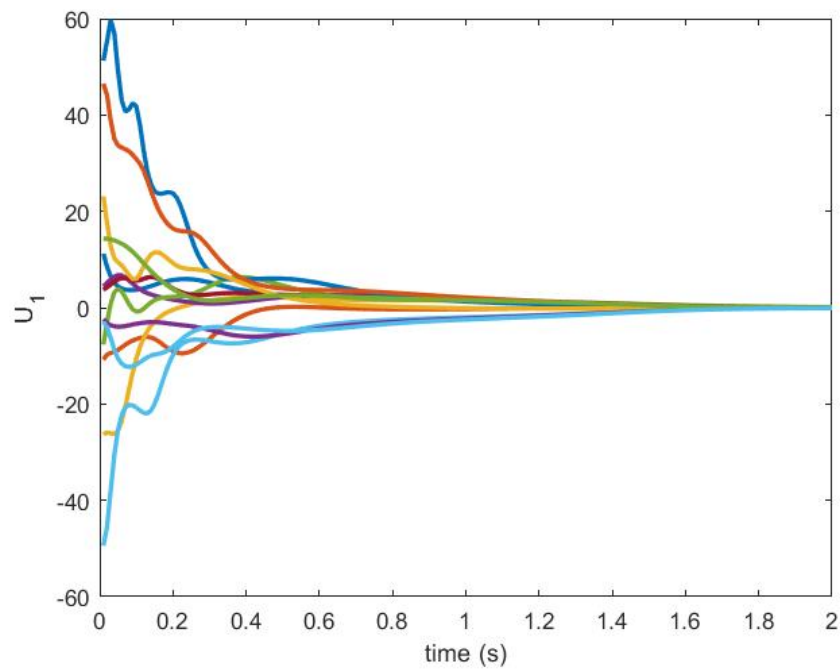


Figure 25. Control U_1 produced by an NDI-based controller.

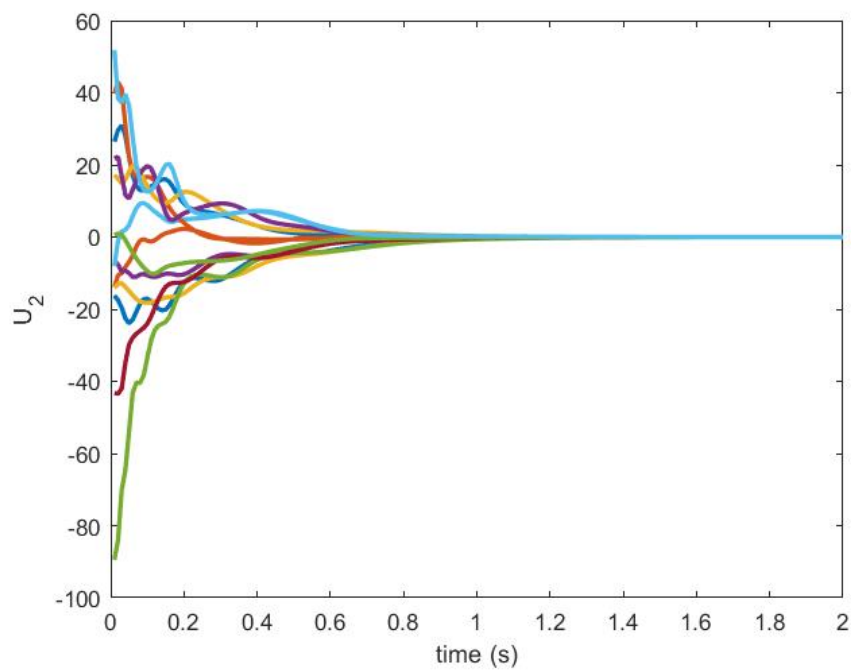
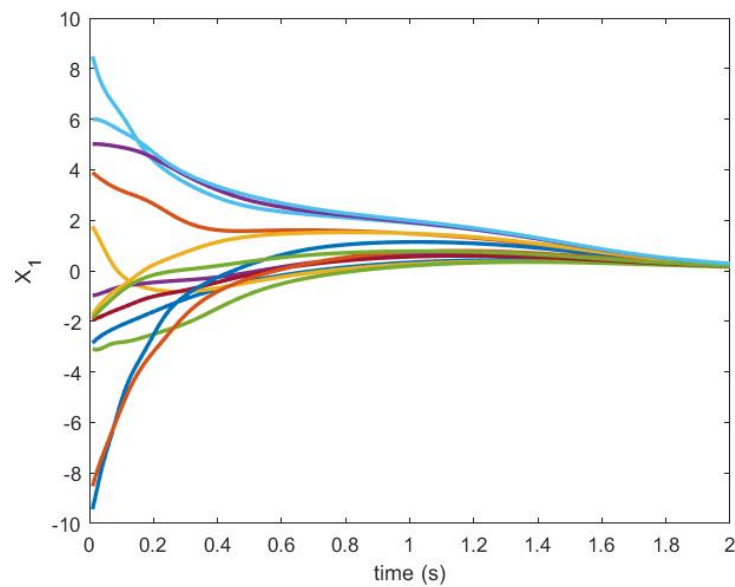
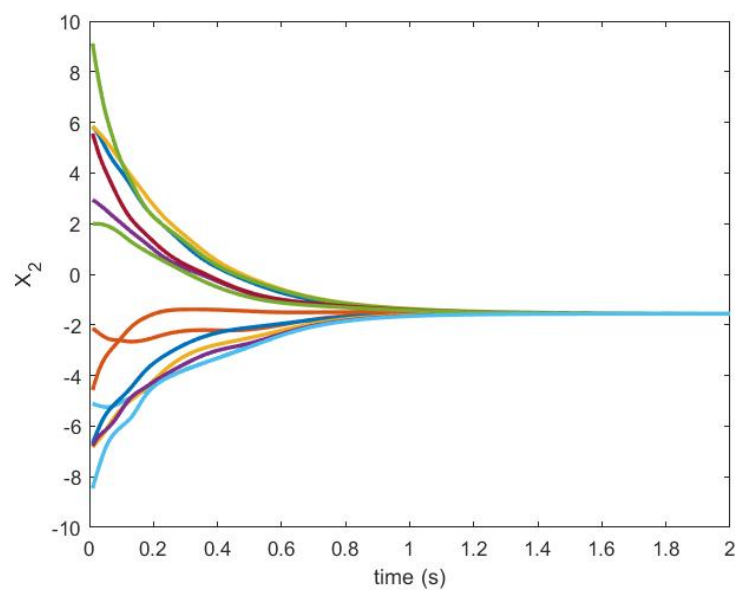


Figure 26. Control U_2 produced by NDI based controller.

The state trajectories of the agents are shown in Figures 27 and 28.

Figure 27. State trajectory X_1 .Figure 28. State trajectory X_2 .

6. Conclusions

The addition of agents is achieved successfully using the proposed idea. This idea will help to build an autonomous drone platform, which is an essential requirement for BVLOS operation. The problem is addressed using a bio-inspired optimization technique which is new in the context of designing topology related to MASs. The controller based on NDI worked well for nonlinear agents. Overall, the results obtained are convincing, and they can be used for executing autonomous operations in many critical missions.

Author Contributions: Conceptualization, S.M. and A.T.; methodology, S.M.; validation, S.M. and A.T.; writing—original draft preparation, S.M.; writing—review and editing, S.M. and A.T.; supervision, A.T.; project administration, A.T.; funding acquisition, A.T. All authors have read and agreed to the published version of the manuscript.

Funding: This research was partially funded by an Engineering and Physical Sciences Research Council (EPSRC) project CASCADE (EP/R009953/1).

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

References

1. Liu, X.; Ji, Z.; Hou, T.; Yu, H. Decentralized stabilizability and formation control of multi-agent systems with antagonistic interactions. *ISA Trans.* **2019**, *89*, 58–66. [[CrossRef](#)] [[PubMed](#)]
2. Nguyen, T.; La, H.M.; Le, T.D.; Jafari, M. Formation control and obstacle avoidance of multiple rectangular agents with limited communication ranges. *IEEE Trans. Control. Netw. Syst.* **2016**, *4*, 680–691. [[CrossRef](#)]
3. Cao, J.; Chen, G.; Li, P. Global synchronization in an array of delayed neural networks with hybrid coupling. *IEEE Trans. Syst. Man Cybern. Part B* **2008**, *38*, 488–498.
4. Yu, W.; Cao, J.; Chen, G.; Lu, J.; Han, J.; Wei, W. Local synchronization of a complex network model. *IEEE Trans. Syst. Man Cybern. Part B* **2008**, *39*, 230–241.
5. Wen, G.; Yu, Y.; Peng, Z.; Rahmani, A. Consensus tracking for second-order nonlinear multi-agent systems with switching topologies and a time-varying reference state. *Int. J. Control* **2016**, *89*, 2096–2106. [[CrossRef](#)]
6. Wen, G.; Duan, Z.; Chen, G.; Yu, W. Consensus tracking of multi-agent systems with Lipschitz-type node dynamics and switching topologies. *IEEE Trans. Circ. Syst. I Regul. Pap.* **2013**, *61*, 499–511. [[CrossRef](#)]
7. Kim, J.M.; Park, J.B.; Choi, Y.H. Leaderless and leader-following consensus for heterogeneous multi-agent systems with random link failures. *IET Control Theory Appl.* **2014**, *8*, 51–60. [[CrossRef](#)]
8. Liu, W.; Zhou, S.; Qi, Y.; Wu, X. Leaderless consensus of multi-agent systems with Lipschitz nonlinear dynamics and switching topologies. *Neurocomputing* **2016**, *173*, 1322–1329. [[CrossRef](#)]
9. Wang, A. Event-based consensus control for single-integrator networks with communication time delays. *Neurocomputing* **2016**, *173*, 1715–1719. [[CrossRef](#)]
10. Zhan, J.; Li, X. Asynchronous consensus of multiple double-integrator agents with arbitrary sampling intervals and communication delays. *IEEE Trans. Circ. Syst. I Regul. Pap.* **2015**, *62*, 2301–2311. [[CrossRef](#)]
11. Xu, X.; Liu, L.; Feng, G. Consensus of single integrator multi-agent systems with unbounded transmission delays. *J. Syst. Sci. Complex.* **2019**, *32*, 778–788. [[CrossRef](#)]
12. Cheng, L.; Hou, Z.G.; Tan, M. A mean square consensus protocol for linear multi-agent systems with communication noises and fixed topologies. *IEEE Trans. Autom. Control* **2013**, *59*, 261–267. [[CrossRef](#)]
13. Wang, Y.; Cheng, L.; Hou, Z.G.; Tan, M.; Zhou, C.; Wang, M. Consensus seeking in a network of discrete-time linear agents with communication noises. *Int. J. Syst. Sci.* **2015**, *46*, 1874–1888. [[CrossRef](#)]
14. Morita, R.; Wada, T.; Masubuchi, I.; Asai, T.; Fujisaki, Y. Multiagent consensus with noisy communication: Stopping rules based on network graphs. *IEEE Trans. Control. Netw. Syst.* **2015**, *3*, 358–365. [[CrossRef](#)]
15. Liu, J.; Ming, P.; Li, S. Consensus gain conditions of stochastic multi-agent system with communication noise. *Int. J. Control Autom. Syst.* **2016**, *14*, 1223–1230. [[CrossRef](#)]
16. Cao, Y.; Ren, W. Optimal linear-consensus algorithms: An LQR perspective. *IEEE Trans. Syst. Man Cybern. Part B* **2009**, *40*, 819–830.
17. Ma, J.; Zheng, Y.; Wang, L. LQR-based optimal topology of leader-following consensus. *Int. J. Robust Nonlinear Control* **2015**, *25*, 3404–3421. [[CrossRef](#)]
18. Ma, J.; Zheng, Y.; Wu, B.; Wang, L. Equilibrium topology of multi-agent systems with two leaders: A zero-sum game perspective. *Automatica* **2016**, *73*, 200–206. [[CrossRef](#)]
19. Wang, H.; Ma, J. Optimal topology for consensus of heterogeneous multi-agent systems. *Neurocomputing* **2016**, *177*, 594–599. [[CrossRef](#)]
20. Mondal, S.; Tsourdos, A. Optimal Topology for Consensus using Genetic Algorithm. *Neurocomputing* **2020**, *404*, 41–49. [[CrossRef](#)]
21. Holland, J.H. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*; MIT Press: Cambridge, MA, USA, 1992.
22. Filipič, B.; Juričič, D. An interactive genetic algorithm for controller parameter optimization. In *Artificial Neural Nets and Genetic Algorithms*; Springer, Vienna: Berlin/Heidelberg, Germany, 1993; pp. 458–462.

23. Grefenstette, J.J. Optimization of control parameters for genetic algorithms. *IEEE Trans. Syst. Man Cybern.* **1986**, *16*, 122–128. [[CrossRef](#)]
24. Goldberg, D.E.; Holland, J.H. *Genetic Algorithms and Machine Learning*; Machine Learning 3; Kluwer Academic Publishers: Dordrecht, The Netherlands, 1988; pp. 95–99.
25. Niculescu, S.P. Artificial neural networks and genetic algorithms in QSAR. *J. Mol. Struct.* **2003**, *622*, 71–83. [[CrossRef](#)]
26. Karr, C.L. Design of an adaptive fuzzy logic controller using genetic algorithm. In Proceedings of the 4th International Conference on Genetic Algorithms, San Diego, CA, USA, 13–16 July 1991; pp. 450–457.
27. Oh, S.K.; Pedrycz, W.; Park, H.S. Hybrid identification in fuzzy-neural networks. *Fuzzy Sets Syst.* **2003**, *138*, 399–426. [[CrossRef](#)]
28. Shi, X.C.; Xie, C.L.; Wang, Y.H. Nuclear power plant fault diagnosis based on genetic-RBF neural network. *J. Mar. Sci. Appl.* **2006**, *5*, 57–62. [[CrossRef](#)]
29. Hao, K.; Zhao, J.; Yu, K.; Li, C.; Wang, C. Path Planning of Mobile Robots Based on a Multi-Population Migration Genetic Algorithm. *Sensors* **2020**, *20*, 5873. [[CrossRef](#)] [[PubMed](#)]
30. Najm, A.A.; Ibraheem, I.K.; Azar, A.T.; Humaidi, A.J. Genetic Optimization-Based Consensus Control of Multi-Agent 6-DoF UAV System. *Sensors* **2020**, *20*, 3576. [[CrossRef](#)]
31. Phua, P.K.H.; Ming, D.; Lin, W. Neural network with genetically evolved algorithms for stocks prediction. *Asia Pac. J. Oper. Res.* **2001**, *18*, 103.
32. Tsai, M.W.; Hong, T.P.; Lin, W.T. A two-dimensional genetic algorithm and its application to aircraft scheduling problem. *Math. Probl. Eng.* **2015**. [[CrossRef](#)]
33. Jain, S.; Gea, H.C. Two-dimensional packing problems using genetic algorithms. *Eng. Comput.* **1998**, *14*, 206–213. [[CrossRef](#)]
34. De Jong, K.A.; Spears, W.M. A formal analysis of the role of multi-point crossover in genetic algorithms. *Ann. Math. Artif. Intell.* **1992**, *5*, 1–26. [[CrossRef](#)]
35. Syswerda, G. Uniform crossover in genetic algorithms. In *Proceedings of the Third International Conference on Genetic Algorithms*; Morgan Kaufmann Publishers: San Francisco, CA, USA, 1989; pp. 2–9.
36. Booker, L.B.; Goldberg, D.E.; Holland, J.H. Classifier systems and genetic algorithms. *Artif. Intell.* **1989**, *40*, 235–282. [[CrossRef](#)]
37. Umbarkar, A.; Sheth, P. Crossover operators in genetic algorithms: A review. *ICTACT J. Soft Comput.* **2015**, *6*, 1083–1092.

Publisher’s Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



© 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).



DSPACE

<https://dspace.org/>

Autonomous addition of agents to an existing group using genetic algorithm

Mondal, Sabyasachi; Tsourdos, Antonios

2020-12-05

MDPI

<https://doi.org/10.3390/s20236953>

Mondal S, Tsourdos A. (2020) Autonomous addition of agents to an existing group using genetic algorithm. Sensors, Volume 20, Issue 23, 2020, Article number 6953