

Generation of optimal persistent formations for heterogeneous multi-agent systems with a leader constraint

Guo-Qiang Wang(王国强), He Luo(罗贺), Xiao-Xuan Hu(胡笑旋)
Citation: Chin. Phys. B . 2018, 27(2): 028901. **doi:** 10.1088/1674-1056/27/2/028901

Journal homepage: http://cpb.iphy.ac.cn; http://iopscience.iop.org/cpb

What follows is a list of articles you may be interested in

Consensus of heterogeneous multi-agent systems based on sampled data with a small sampling delay

Wang Na, Wu Zhi-Hai, Peng Li
Chin. Phys. B . 2014, 23(10): 108901. **doi:** 10.1088/1674-1056/23/10/108901

Statistical physics of hard combinatorial optimization：Vertex cover problem

Zhao Jin-Hua, Zhou Hai-Jun
Chin. Phys. B . 2014, 23(7): 078901. **doi:** 10.1088/1674-1056/23/7/078901

Output regulation for linear multi-agent systems with unmeasurable nodes

Liang Hong-Jing, Zhang Hua-Guang, Wang Zhan-Shan, Wang Jun-Yi
Chin. Phys. B . 2014, 23(1): 018902. **doi:** 10.1088/1674-1056/23/1/018902

Finite-time consensus of heterogeneous multi-agent systems

Zhu Ya-Kun, Guan Xin-Ping, Luo Xiao-Yuan
Chin. Phys. B . 2013, 22(3): 038901. **doi:** 10.1088/1674-1056/22/3/038901

Consensus protocol for heterogeneous multi-agent systems: A Markov chain approach

Zhu Shan-Ying, Chen Cai-Lian, Guan Xin-Ping
Chin. Phys. B . 2013, 22(1): 018901. **doi:** 10.1088/1674-1056/22/1/018901

Generation of optimal persistent formations for heterogeneous multi-agent systems with a leader constraint*

Guo-Qiang Wang(王国强)[1,2], He Luo(罗贺)[1,2,†], and Xiao-Xuan Hu(胡笑旋)[1,2]

[1] School of Management, Hefei University of Technology, Hefei 230009, China
[2] Key Laboratory of Process Optimization & Intelligent Decision-making, Ministry of Education, Hefei 230009, China

(Received 18 July 2017; revised manuscript received 9 October 2017; published online 5 January 2018)

In this study, we consider the generation of optimal persistent formations for heterogeneous multi-agent systems, with the leader constraint that only specific agents can act as leaders. We analyze three modes to control the optimal persistent formations in two-dimensional space, thereby establishing a model for their constrained generation. Then, we propose an algorithm for generating the optimal persistent formation for heterogeneous multi-agent systems with a leader constraint (LC-HMAS-OPFGA), which is the exact solution algorithm of the model, and we theoretically prove its validity. This algorithm includes two kernel sub-algorithms, which are optimal persistent graph generating algorithm based on a minimum cost arborescence and the shortest path (MCA-SP-OPGGA), and the optimal persistent graph adjusting algorithm based on the shortest path (SP-OPGAA). Under a given agent formation shape and leader constraint, LC-HMAS-OPFGA first generates the network topology and its optimal rigid graph corresponding to this formation shape. Then, LC-HMAS-OPFGA uses MCA-SP-OPGGA to direct the optimal rigid graph to generate the optimal persistent graph. Finally, LC-HMAS-OPFGA uses SP-OPGAA to adjust the optimal persistent graph until it satisfies the leader constraint. We also demonstrate the algorithm, LC-HMAS-OPFGA, with an example and verify its effectiveness.

Keywords: leader constraint, heterogeneous multi-agent system, optimal persistent formation, minimum cost arborescence

PACS: 89.20.Ff, 87.85.St, 89.65.Ef, 02.30.Em **DOI:** 10.1088/1674-1056/27/2/028901

1. Introduction

In recent years, the formation control of multi-agent systems (MAS) has become an important topic, and has found their wide applications in both military and civilian domains. There are many types of agents in MAS, such as mobile robots,[1] satellites,[2] unmanned aerial vehicles (UAVs),[3] and autonomous underwater vehicles (AUVs).[4] The main idea of MAS formation control is that several agents exchange information via communication links and use preset formation control methods to determine their movement, thereby forming and maintaining a specific formation shape.

At present, the set of communication links used by agents to maintain their formation shape can be expressed in multiple manners, including communication topology,[5,6] information exchange topology,[7] connection topology,[8] information structure,[9–11] and information topology.[12,13] In the present study, these links are referred to as communication topology, while the set of all the available communication links between agents are termed network topology. Both communication topology and network topology can be expressed by a weighted directed graph, which, hereafter, is named the directed graph. Furthermore, the communication topology of the agent formation is a subgraph of its network topology, where nodes denote the agents in the formation, arcs represent the communication links between agents, and the weight of each arc refers to the communication cost of the link.

The formation control approaches that an agent formation can be used to maintain its formation shape include the leader–follower-based approach, the virtual structure-based approach, the behavior-based approach, the consensus-based approach, the minimally rigid graph-based approach, and the minimally persistent graph-based approach. The minimally persistent graph-based approach is a formation control method based on distance, with the corresponding agent formation referred to as the minimally persistent formation.[14] The communication topology of the minimally persistent formation must be a minimally persistent graph in its network topology to minimize the number of communication links used by the agent formation to maintain its formation shape. In two-dimensional (2D) space, there must be $2\times|V|-3$ and only $2\times|V|-3$ arcs in a minimally persistent graph with $|V|$ nodes, and the in-degree of each node is less than or equal to 2.[9] Moreover, each agent must maintain a constant distance from the agents which have an outgoing arc to it in the communication topology during the movement of the minimally persistent formation. For the generation of minimally persistent formations, a graph operation method to transform from one minimally persistent formation to another minimally persistent formation was proposed.[9] In

*Project supported by the National Natural Science Foundation of China (Grant Nos. 71671059, 71401048, 71521001, 71690230, 71690235, and 71472058) and the Anhui Provincial Natural Science Foundation, China (Grant No. 1508085MG140).

†Corresponding author. E-mail: luohe@hfut.edu.cn

© 2018 Chinese Physical Society and IOP Publishing Ltd

addition, an algorithm based on the Pebble game and its graph operations to generate minimally persistent formations was proposed.[15] Another algorithm applicable to the generation of minimally persistent formations with a circular structure was proposed.[16]

However, the minimally persistent formation of a given network topology is not unique. The minimally persistent formation with the smallest sum of the costs of the communication links in its communication topology is called the optimal persistent formation.[17] The communication topology of the optimal persistent formation must be an optimal persistent graph of its network topology. For the generation of optimal persistent formations, an algorithm suitable for optimal persistent formations with special structures, and based on the optimal rigid graph and the configuration of the triangle with a leader was proposed.[17] An optimal persistent formation generation algorithm, which is based on the optimal rigid graph and the operation of directed vertex addition and applicable to optimal persistent formations with triangles or quadrangles used as the basic structures, was proposed.[18] Another optimal persistent formation generation algorithm based on the optimal rigid graph with four directed operations was proposed,[19] which is suitable for optimal persistent formations with any structure.

For homogeneous agent formations, where each agent can act as a leader, the optimal persistent formation can be generated by simply directing the optimal rigid graph to one arbitrary optimal persistent graph. However, for a heterogeneous agent formation, only specific agents can act as the leader, and this leader constraint must be considered when directing the optimal rigid graph to the optimal persistent graph. To solve this problem, in this study are introduced the following innovations: (i) we propose the algorithm MCA-SP-OPGGA, which directs any optimal rigid graph to obtain an optimal persistent graph and we then prove the validity of this algorithm; (ii) we propose the algorithm SP-OPGAA, which adjusts any optimal persistent graph to satisfy the leader constraint and we then prove this algorithm validity; and (iii) we propose the algorithm LC-HMAS-OPFGA based on MCA-SP-OPGGA and SP-OPGAA, which generates the optimal persistent formation for heterogeneous multi-agent system with the leader constraint.

The rest of this study is organized in the following manner. In Section 2, we analyze the problem and establish the model that describes it. In Section 3, we propose an exact solution algorithm to solve the stated problem and prove its validity. In Section 4, we illustrate this algorithm by a simulation example and verify the validity of the algorithm. In Section 5, we draw some conclusions from this study and point out future directions of research in this field.

2. Problem analysis and modeling

2.1. Formation shape of agent formation

The $AGENT = \{agent_i\}$ represents the set of n agents, where the agents that can act as the leaders are represented by $AGENT^*$, with $AGENT^* \subset AGENT$. Matrix $\boldsymbol{S}$ represents the formation shape constituted by n agents as shown in Eq. (1) below.

$$\boldsymbol{S} = \begin{bmatrix} x_{1F} & y_{1F} & z_{1F} \\ \cdots & \cdots & \cdots \\ x_{iF} & y_{iF} & z_{iF} \\ \cdots & \cdots & \cdots \\ x_{nF} & y_{nF} & z_{nF} \end{bmatrix}. \tag{1}$$

In the above expression, F denotes the formation reference point (which is the position of one specific agent in the formation shape) and (x_{iF}, y_{iF}, z_{iF}) represents the position coordinates of the i-th agent in the coordinate system, with F used as the origin. For example, figure 1 shows the formation shape composed of five agents in the two-dimensional (2D) space, with their positions numbered as $\{1,2,3,4,5\}$, respectively. Their relative positions in the space are shown in the left portion in Fig. 1. If position four is regarded as the origin, then the coordinates of all the positions are shown in the right portion in Fig. 1.

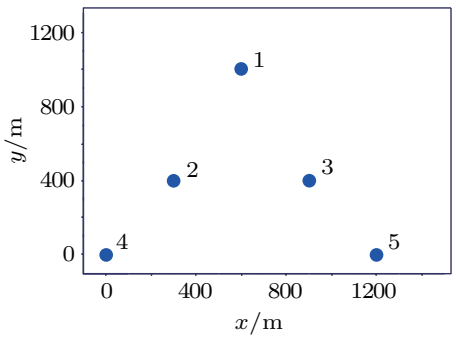


Fig. 1. (color online) Formation shape composed of five agents and its mathematical description.

2.2. Network topology and communication topology of optimal persistent formation

The network topology of the agent formation composed of n agents can be expressed by the directed graph $D = (V, A, W)$, where $V = \{v_i\}, 1 \le i \le n$ denotes the set of nodes in the graph, node v_i denotes $agent_i$, $A = \{a_{ij}\} \subset V \times V, 1 \le i \ne j \le n$ is the set of arcs in the graph, arc $a_{ij} = (v_i, v_j)$ represents the communication link from v_i to v_j, $W = \{w(a_{ij})\}, a_{ij} \in A$ is the set of arc weights in the graph, and arc weight, $w(a_{ij})$, denotes the cost of the communication link from v_i to v_j, which is equivalent to the communication distance from v_i to v_j in this study. For example, figure 2 shows the network topology of five agents to keep the formation shape shown in Fig. 1.

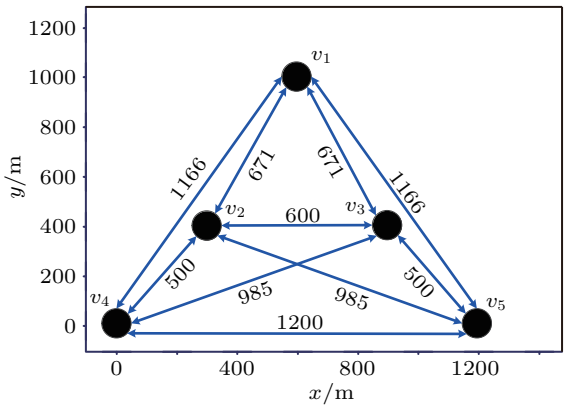


Fig. 2. (color online) Network topology of the agent formation composed of five agents.

The communication topology of the above agent formation for maintaining its formation shape can be expressed by the directed graph $T = (V, A^*, W^*)$, $A^* \subseteq A, W^* \subseteq W$, where T is a subgraph of $D = (V, A, W)$. The function, $w(T)$, is the sum of the weights of all the arcs in T and denotes the total communication cost when the agent formation maintains its formation shape. It is called the formation communication cost and shown in Eq. (2) as follows:

$$w(T) = \sum_{a_{ij} \in A^*} w(a_{ij}). \tag{2}$$

For a minimally persistent formation, the communication topology T is a minimally persistent graph in its network topology D, and $a_{ij} = (v_i, v_j) \in A^*$ means that v_i needs to send its own information to v_j when the formation tries to keep its formation shape. Then v_j can adjust its movement to keep a constant distance from v_i. In other words, there is a distance constraint between v_j and v_i. The function $d(j)$, the in-degree of v_j, denotes the number of distance constraints of v_j with other agents. For example, under the network topology shown in Fig. 2, the communication topology of a minimally persistent formation is shown in Fig. 3. The number of communication links is $2 \times |V| - 3 = 2 \times 5 - 3 = 7$, the in-degree of each node is less than or equal to 2, and its formation communication cost is 5588.

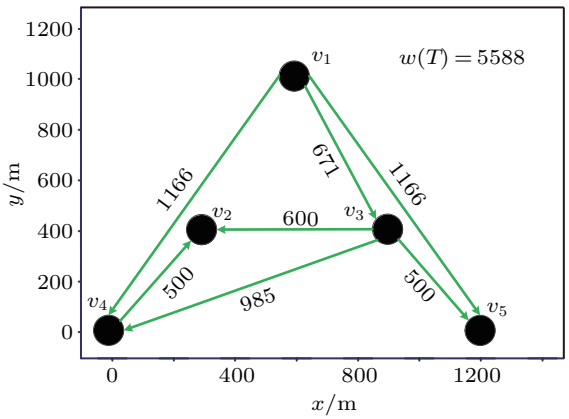


Fig. 3. (color online) Communication topology of a minimally persistent formation composed of five agents.

Furthermore, under the same network topology, the minimally persistent formation with the smallest formation communication cost is defined as the optimal persistent formation. Figure 4 shows the communication topology of an optimal persistent formation, whose formation communication cost is the smallest in all possible minimally persistent formations under the network topology shown in Fig. 2.

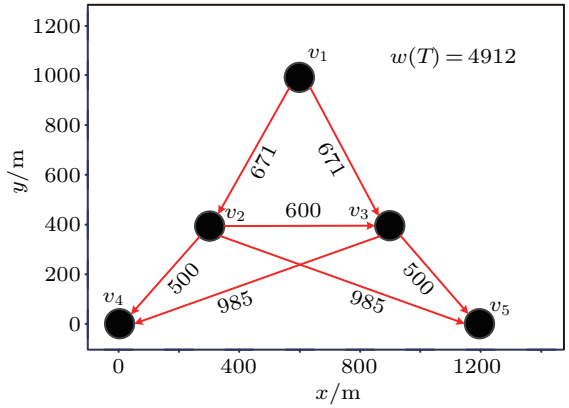


Fig. 4. (color online) Communication topology of an optimal persistent formation composed of five agents.

2.3. Problem modeling

The optimal persistent formation can be determined by the following three control modes[20] in 2D space.

i) Leader-first-follower (LFF): An agent, as a leader, can move freely in 2D space (i.e., it has 2 degrees of freedom). Another agent, as the first follower, needs to maintain a constant distance from the leader during movement (i.e., it has 1 degree of freedom). Each of the remaining agents, as ordinary followers, must maintain a constant distance from any other two agents during the movement (i.e., it has no degree of freedom). In this control mode, the in-degrees of the nodes corresponding to the leader, the first follower, and the ordinary followers in the communication topology of the formation are 0, 1, and 2 respectively.

ii) Leader-remote-follower (LRF): An agent, as a leader, can move freely in 2D space (i.e., it has 2 degrees of freedom). Another agent, as the remote follower, needs to maintain a constant distance from an agent apart from the leader during movement (i.e., it has 1 degree of freedom). Each of the remaining agents, as ordinary followers, must maintain a constant distance from any other two agents during the movement (i.e., it has no degree of freedom). In this control mode, the in-degrees of the nodes corresponding to the leader, the remote follower, and the ordinary followers in the communication topology of the formation are 0, 1, and 2 respectively.

iii) Co-leader: Three agents act as the co-leaders, each of whom needs to maintain a constant distance from any other agent during motion (i.e., each co-leader has one degree of freedom). Each agent, other than co-leaders, as an ordinary follower should maintain a constant distance from any other

two agents during movement (i.e., it has no degree of freedom). In this control mode, the in-degrees of corresponding nodes of the co-leaders and the ordinary followers in the communication topology of the formation are 1 and 2, respectively.

The leader constraint of the optimal persistent formation for heterogeneous multi-agent systems can be described in this way: among the agents that can act as the leaders, there is one and only one agent whose in-degree of its corresponding node in the communication topology is 0, or else there are three and only three agents whose in-degree of their corresponding nodes in the communication topology is 1.

In this study, the generation of the optimal persistent formation for heterogeneous multi-agent systems with the leader constraint is formulated as the following optimization problem.

Among $AGENT = \{agent_i\}, 1 \leq i \leq n$, the set of the n heterogeneous agents, $AGENT^* \subset AGENT$ is the subset of agents that can act as the leaders. To make sure the n agents comprise the optimal persistent formation and keep the formation shape $\boldsymbol{S}$, an optimal persistent graph $T = (V, A^*, W^*)$ must be generated as the communication topology for this formation. This formation must minimize $w(T)$, the formation communication cost corresponding to this communication topology, and the optimal persistent graph, T, must satisfy the leader constraint, namely one of the following conditions:

1) $\exists! v_i \in V : d(i) = 0$, $agent_i \in AGENT^*$;

2) $\exists! (v_i, v_j, v_k \in V)$: $i \neq j \neq k$, $d(i) = d(j) = d(k) = 1$, $agent_i$, $agent_j$, $agent_k \in AGENT^*$.

3. Algorithm

We propose the algorithm LC-HMAS-OPFGA to address the model named above. This algorithm includes two kernel sub-algorithms. The first sub-algorithm, MCA-SP-OPGGA, is used to direct an optimal rigid graph to an optimal persistent graph, and the second sub-algorithm, SP-OPGAA, is used to adjust an optimal persistent graph until it satisfies the leader constraint. Given the formation shape and the set of agents that can act as the leaders, LC-HMAS-OPFGA can generate the optimal persistent formation for heterogeneous multi-agent system under the leader constraint.

3.1. Algorithm for the generation of optimal persistent graph

In this subsection, we propose an optimal persistent graph generating algorithm based on minimum cost arborescence (MCA)[21] and shortest path (SP),[22] referred to as MCA-SP-OPGGA. By adding a virtual node and virtual arcs, generating and combining minimum cost arborescence, adding arcs, and reversing paths, this algorithm can direct any optimal rigid graph to obtain an optimal persistent graph. The pseudo-codes of this algorithm are shown in Table 1.

We will also verify the feasibility of MCA-SP-OPGGA theoretically. By proving the existence of MCA, the combinatorial properties of MCA, and the characteristics of adding arcs and reversing paths, it is proven that MCA-SP-OPGGA can generate an optimal persistent graph that satisfies the judging criterion shown in Lemma 2.

Lemma 1[23] The rigid graph with the number of nodes being $|V|$ and the number of edges being $2 \times |V| - 3$ is the minimally rigid graph. If the sum of the weight of each edge of a minimally rigid graph is the smallest in all the rigid graphs with the same nodes, then this minimally rigid graph is the optimal rigid graph.

Lemma 2[9] In 2D space, if the in-degree $d(i)$ of each node, v_i, in a directed graph is less than or equal to 2 and its corresponding undirected graph is the optimal rigid graph, then this directed graph is the optimal persistent graph.

Theorem 1 (Existence of MCA): T_1' and T_2', the minimum cost arborescences in MCA-SP-OPGGA, must exist and their root nodes are v_0 and the number of their arcs is $|V|$.

Proof

Since there is one outgoing arc from v_0 to each of all other nodes in D_R', there is at least one spanning tree in D_R' with v_0 used as the root node. For example, one of these spanning trees is composed of v_0, all other nodes, and all outgoing arcs from v_0 to each of all other nodes. Meanwhile, the minimum cost arborescence is the minimum spanning tree of a weighted directed graph. Thus, the minimum cost arborescence, T_1' of D_R', must exist.

Since T_1 is obtained by deleting v_0 from T_1' (namely, v_0 and its outgoing arcs are not included in T_1), the graph D_R'' obtained by deleting all the arcs and their reverse arcs in T_1 from D_R' must contain v_0 and one outgoing arc from v_0 to each of all other nodes. Thus, there is at least one spanning tree in D_R'' with v_0 as the root node, which means that the minimum cost arborescence, T_2' of D_R'', must also exist.

Since T_1' is a spanning tree of D_R', T_1' must contain all the nodes in D_R' (namely, v_0 and the other $|V|$ nodes), and the number of arcs of T_1' is the number of nodes of D_R' minus 1, which is $|V|$. Meanwhile, since there are only outgoing arcs for v_0 in D_R', the root node of T_1' must be v_0, otherwise, v_0 must have an incoming arc based on the conditions of spanning tree supposing that v_0 is not a root node of T_1'.

Similarly, the root node of T_2' must be v_0, and the number of arcs of T_2' is the number of nodes of D_R'' minus 1, which is also $|V|$.

Theorem 2 (Combinatorial properties of MCA): The in-degree of each node in the directed graph, T, obtained by the combination of T_1 and T_2 in MCA-SP-OPGGA, is less than or equal to 2, and the number of arcs of T must be less than or equal to $2 \times |V| - 3$.

Table 1. Pseudo-codes of MCA-SP-OPGGA.

Input: Optimal rigid graph $R=(V,E,W)$.
Output: Optimal persistent graph $T=(V, A^*, W^*)$
/* Operations of adding virtual node and virtual arcs. */
D_R = GenerateDirectedGraph (R); /* Converting each edge of R into 2 arcs with the same weight but opposite directions, and obtaining a directed graph $D_R=(V,A_R,W_R)$. */
D'_R =AddVirtualNode(D_R); // Adding virtual node v_0 in D_R.
D'_R =AddVirtualArcs(D'_R , v_0); /* Adding an outgoing arc from v_0 to each other node. */
D'_R =SetWeightOfVirtualArc(D'_R , v_0 , $w(D_R)+1$); /* Setting the weight of each outgoing arc of v_0 into w_0=$w(D_R)+1$, which is 1 greater than the sum of the weights of all arcs in D_R. */
/* Operations of generating and combining minimum cost arborescence. */
T'_1 =GenerateMCA(D'_R); /* Generating minimum cost arborescence T'_1 of D'_R. */
T_1 =DeleteVirtualNode(T'_1); // Deleting v_0 in T'_1 to obtain T_1.
D''_R =DeleteArcsAndReversedArcs(D'_R , T_1); /* For each arc in T_1, deleting it and its reverse arc in D'_R, and obtaining the directed graph D''_R. */
T'_2 =GenerateMCA(D''_R); /* Generating minimum cost arborescence T'_2 of D''_R. */
T_2 =DeleteVirtualNode(T'_2); // Deleting v_0 in T'_2 to obtain T_2.
T =MergeDirectedGraphs(T_1 , T_2); // Combining T_1 and T_2 to obtain T.
/* Judgement of the condition of optimal persistent graph. */
m =GetNumberOfArcs(T); // Getting the number of arcs in T.
if $m == 2\times|V|-3$
  return T ; // T is an optimal persistent graph.
end if
/* T is currently not an optimal persistent graph, thus performing arc addition and path reversal operations until T satisfies the criteria of optimal persistent graph. */
for $l=1:|V|$
  if (The arc a_{ij} or a_{ji}, which corresponds to the lth edge $e_{ij}(i<j)$ in R, is in T.)
    continue;
  end if
  if $d(j)<2$ // The in-degree of v_j in T is less than 2.
    T =AddArc(T , a_{ij}); // Adding the arc a_{ij} to T.
  else if $d(i)<2$ // The in-degree of v_i in T is less than 2.
    T =AddArc(T , a_{ji}); // Adding the arc a_{ji} to T.
  else /* The in-degrees of v_j and v_i in T are both equal to 2. */
    T =AddArc(T , a_{ij}); // Adding the arc a_{ij} to T.
    k =FindNode(T , j ,2); /* Finding node v_k in T which satisfies that there is a path from v_k to v_j and $d(k)<2$. */
    P = FindShortestPath(T , k , j); /* Finding the shortest path P from v_k to v_j in T. */
    T = RevertPath(T , P); // Reversing all arcs on P in T.
  end if
  $m=m+1$; // Updating the number of arcs in T.
  if $m == 2\times|V|-3$ /* Judging whether T satisfies the criteria of optimal persistent graph. */
    return T ; // When the criteria is satisfied, return T.
  end if
end for

Proof

Since T'_1 is the minimum cost arborescence of D'_R (namely, the minimum spanning tree of D'_R), the in-degrees of all nodes in T'_1 are less than or equal to 1 and T'_1 is a subgraph of D'_R. Since T'_2 is the minimum cost arborescence of D''_R, T'_2 is a subgraph of D''_R and in-degrees of all nodes in T'_2 are less than or equal to 1. D''_R is a subgraph of D'_R because D''_R is obtained by deleting all the arcs and their reverse arcs in T_1 from D'_R, thus T'_2 is also a subgraph of D'_R.

Since T_1 and T_2 are gained by deleting v_0 in T'_1 and T'_2, and the in-degrees of all nodes in T'_1 and T'_2 are less than or equal to 1, the in-degrees of all nodes in T_1 and T_2 are also less than or equal to 1. Furthermore, because T is obtained by the combination of T_1 and T_2, the in-degrees of all nodes in T

must be less than or equal to 2.

Since both T_1' and T_2' are subgraphs of D_R', T_1 and T_2 are gained by deleting v_0 in T_1' and T_2', and D_R is obtained after v_0 in D_R' has been deleted, thus both T_1 and T_2 are subgraphs of D_R. Therefore the directed graph, T, gained by the combination of T_1 and T_2, must be a subgraph of D_R.

As mentioned above, D_R'' is obtained by deleting all the arcs and their reverse arcs in T_1 from D_R', T_2' is the minimum cost arborescence of D_R'', and T_2 is gained by deleting v_0 in T_2'. Therefore, the edges corresponding to arcs in T_1 and T_2 do not overlap each other, the arcs in T_1 and T_2 do not overlap each other, and the number of arcs in the directed graph, T, gained by the combination of T_1 and T_2, must be equal to the number of edges in the undirected graph corresponding to T. Since T is a subgraph of D_R, the corresponding undirected graph of T is a subgraph of R, which is the corresponding undirected graph of D_R. Thus, the number of arcs in T must be less than or equal to the number of edges in R. In addition, R is the optimal rigid graph, as indicated by Lemma 1, so the number of edges of R is $2\times|V|-3$, which means that the number of arcs of T must be less than or equal to $2\times|V|-3$.

Theorem 3 (Characteristics of adding arcs and reversing paths): When the in-degree of each node in T is less than or equal to 2, if both a_{ij} $(i<j)$ and a_{ji} do not belong to T, then the in-degrees of all the nodes will still be less than or equal to 2, even after the a_{ij} $(i<j)$ or a_{ji} has been added to T and the corresponding path has been reversed for MCA-SP-OPGGA.

Proof

Since the in-degree of each node in T is less than or equal to 2 before a_{ij} or a_{ji} is added to T, namely $d(i)\leq 2$ and $d(j)\leq 2$, this theorem can be in three different cases.

(I) When $d(i)\leq 2$ and $d(j)<2$. After a_{ij} is added to T for MCA-SP-OPGGA, $d(j)$ will be increased by 1, but it will still be less than or equal to 2, and the in-degrees of the remaining nodes will remain unchanged, namely less than or equal to 2.

(II) When $d(i)<2$ and $d(j)\leq 2$. After a_{ji} is added to T for MCA-SP-OPGGA, $d(i)$ will be increased by 1, but it will still be less than or equal to 2, and the in-degrees of the remaining nodes will remain unchanged, namely less than or equal to 2.

(III) When $d(i)=2$ and $d(j)=2$. First, a_{ij} is added to T for MCA-SP-OPGGA and then another node, v_k in T is found, which satisfies that the in-degree of v_k in T is less than 2, and there is a path from v_k to v_j in T. Then the algorithm reverses all the arcs on this path.

First, we prove that the node v_k must exist as follows. Since the in-degree of v_i in T is equal to 2, T is gained by the combination of T_1 and T_2, and the in-degrees of all nodes in T_1 and T_2 are less than or equal to 1 as indicated by the proof of Theorem 2, thus the in-degrees of v_i in both T_1 and T_2 are 1. Since T_1 is obtained after deleting v_0 in T_1', and T_1' is the spanning tree of D_R' with v_0 as the root node, thus there must be a path, $(v_0,v_k,\ldots,v_i)$, from v_0 to v_i in T_1' and the in-degree of v_k in T_1' is 1. Meanwhile, T_1 is obtained by deleting v_0 in T_1', thus there must be a path, $(v_k,\ldots,v_i)$, from v_k to v_i in T_1 and the in-degree of v_k in T_1 is 0. Since T_1 is a subgraph of T, there must be a path, $(v_k,\ldots,v_i)$, from v_k to v_i in T. Furthermore, since a_{ij} has been added to T, there must be a path, $(v_k,\ldots,v_i,v_j)$, from v_k to v_j in T. Since the in-degrees of all nodes in T_2 are less than or equal to 1 as indicated by the proof of Theorem 2, the in-degree of v_k in T_2 is less than or equal to 1. Since the in-degree of v_k in T_1 is 0 and T is gained by the combination of T_1 and T_2, the in-degree of v_k in T must be less than 2.

Second, since $d(j)=2$ before a_{ij} is added to T, the in-degree of v_j plus 1 equals 3 after a_{ij} has been added into T. After all the arcs on the path $(v_k,\ldots,v_i,v_j)$ are reversed, the in-degree of v_k plus 1 is still less than or equal to 2. The in-degree of v_j minus 1 equals 2 and the in-degrees of the other nodes remain unchanged, namely less than or equal to 2.

In conclusion, after a_{ij} $(i<j)$ or a_{ji} has been added to T and the corresponding path has been reversed for MCA-SP-OPGGA in all the above three different cases, the in-degrees of all the nodes will still be less than or equal to 2.

Theorem 4 The directed graph, T, output by MCA-SP-OPGGA, is an optimal persistent graph of R.

Proof As indicated by Theorem 2, the in-degree of each node in the directed graph, T, obtained by the combination of T_1 and T_2 is less than or equal to 2, and the number of arcs of T must be less than or equal to $2\times|V|-3$. The proof will be continued in two cases.

I) The number of arcs in T is equal to $2\times|V|-3$. Since the arcs in T are generated by directing the edges in the optimal rigid graph R, the number of edges in the undirected graph of T is also equal to $2\times|V|-3$. As indicated by the proof of Theorem 2, the undirected graph of T is a subgraph of R. In addition, as indicated by Lemma 1, the number of edges in R is equal to $2\times|V|-3$. Therefore, the undirected graph corresponding to T must be R. As indicated by Theorem 2, the in-degree of each node in T is less than or equal to 2. As a result, T meets the conditions in Lemma 2 and is an optimal persistent graph of R.

II) The number of arcs in T is less than $2\times|V|-3$. Since the undirected graph of T is a subgraph of R as indicated by proving Theorem 2, neither of the two reverse arcs corresponding to one or more edges $\{e_{ij}\ (i<j)\}$ in R is in T. As indicated by Theorem 3, after the corresponding arcs a_{ij} $(i<j)$ of each edge e_{ij} in $\{e_{ij}\ (i<j)\}$ or a_{ji} are successively added to T, the in-degrees of all the nodes in T are still less than or equal to 2 and the number of edges of the undirected graph corresponding to T is finally also equal to $2\times|V|-3$. Therefore, T meets the conditions in Lemma 2, namely T is an optimal persistent

graph of R.

3.2. Adjusting algorithm for optimal persistent graph to satisfy leader constraint

When the optimal persistent graph does not satisfy the leader constraint, the optimal persistent graph adjusting algorithm based on shortest path[22] (SP-OPGAA), proposed below, can solve this problem. The optimal persistent graph that satisfies the leader constraint can be obtained using a path reversal operation, whose pseudo-code is shown in Table 2.

The validity of SP-OPGAA will be proven theoretically.

Lemma 3[14] In an optimal persistent graph, if and only if $d(j) \leq 1$ and $d(i) \geq 1$, there is a directed path between node v_j and node v_i.

Table. 2. Pseudo-codes of SP-OPGAA.

Input: Optimal persistent graph $T=(V,A^*,W^*)$ and the set $AGENT^*$ of agents that can act as the leaders.
Output: Optimal persistent graph T that meets the leader constraint.
i = FindLeaderNode (T , $AGENT^*$); /* Finding node v_i in T which satisfies $agent_i \in AGENT^*$. */
/* When the in-degree of v_i is greater than 0, performing the path reversal operation until $d(i)==0$. */
while $(d(i)>0)$
  j = FindNode(T , i ,2); /* Finding node v_j in T which satisfies that there is a path from v_j to v_i and $d(j)<2$. */
  P = FindShortestPath(T , j , i); /* Calculating the shortest path P from v_j to v_i in T. */
  T = RevertPath(T , P); // Reversing all arcs on the shortest path P in T.
end while
return T **;** /* T is optimal persistent graph that meets the leader constraint, and return T. */

For the paths satisfying Lemma 3, Theorem 5 proves that SP-OPGAA can produce the optimal persistent graph satisfying the leader constraint through several path reversal operations.

Theorem 5 The optimal persistent graph, T, that meets the leader constraint can be obtained by SP-OPGAA.

Proof

When the optimal persistent graph, T, does not satisfy the leader constraint, none of the in-degrees of corresponding nodes of $AGENT^*$ in T is 0. Thus, there must be a node, v_i in T, that satisfies $agent_i \in AGENT^*$, but has an in-degree that is not 0, rather $1 \leq d(i) \leq 2$.

Assume that the in-degrees of $|V|-1$ nodes in T except v_i are equal to 2. Since one in-degree of a node corresponds to one arc, the sum of arcs of the $|V|-1$ nodes is $2\times(|V|-1)$. Also, since $1 \leq d(i) \leq 2$, the number of arcs of v_i is greater than or equal to 1. Therefore, the total number of arcs in T must be greater than or equal to $2\times(|V|-1)+1$, which contradicts the statement that the number of arcs in the optimal persistent graph T equals $2\times|V|-3$ as stipulated in Lemma 1 and Lemma 2. Therefore, the assumption does not hold true, which means that there must be a node, v_j, with an in-degree less than 2 in T apart from v_i, namely $d(j) \leq 1$. In this case, as indicated by Lemma 3, there is a path from v_j to v_i in T.

After all the arcs on the path from v_j to v_i are reversed, the in-degree of v_j will be increased by 1, so that $d(j) \leq 2$. Meanwhile, the in-degree of v_i will be reduced by 1, so that $0 \leq d(i) \leq 1$. However, the in-degrees of the other nodes, except v_j and v_i, on the path from v_j to v_i remain unchanged, namely less than or equal to 2. Thus, the path reversal can reduce the in-degree of v_i by 1 and this proves that T meets the conditions in Lemma 2, showing that T is still an optimal persistent graph of R.

To sum up, after several operations of path reversal, the in-degree of v_i will become 0 and T will remain an optimal persistent graph of R. Also, $agent_i \in AGENT^*$, making T an optimal persistent graph that satisfies the leader constraint.

3.3. Algorithm for the generation of optimal persistent formation for heterogeneous multi-agent system under leader constraint

Based on MCA-SP-OPGGA, SP-OPGAA, and the optimal rigid graph generating algorithm, an optimal persistent formation generating algorithm for heterogeneous multi-agent systems under the leader constraint (LC-HMAS-OPFGA) is here proposed. It can generate the optimal persistent formation for any formation shape in 2D space under any leader constraint, and its pseudo-code is shown in Table 3.

Table. 3. Pseudo-codes of LC-HMAS-OPFGA.

Input: Formation shape $\boldsymbol{S}$ and the set $AGENT^*$ of agents that can act as the leaders.
Output: Communication Topology T of optimal persistent formation under the leader constraint.
D = GenerateNetworkTopology ($\boldsymbol{S}$); /* Generating network topology $D=(V,A,W)$ according to the formation shape $\boldsymbol{S}$. */
G = GenerateUndirectedGraph (D); /* Converting the arcs in D to edges, and obtaining an undirected graph $G=(V,E,W)$. */
R = GenerateOptimalRigidGraph (G); /* Generating the optimal rigid graph $R=(V,E^*,W^*)$ of G . */
T = MCA-SP-OPGGA(R); /* Generating an optimal persistent graph $T=(V,A^*,W^*)$ corresponding to R . */

/* Judging whether T meets the leader constraint. */
for $i=1:|V|$
  if (($agent_i \in AGENT^*$ **and** $d(i)==0$) **or**
    ($agent_i, agent_j, agent_k \in AGENT^*, i\neq j\neq k$ **and** $d(i)==d(j)==d(k)==1$))
    return T ; // T meets the leader constraint, and return T .
  end if
end for

T = SP-OPGAA (T); /* T doesn't meet the leader constraint, thus adjusting T until it satisfies the leader constraint. */

return T ;

4. Calculation examples

Now, we assume that one heterogeneous multi-agent persistent formation consists of 16 agents, namely $AGENT = \{agent_i\}, 1 \le i \le 16$, also assume that the communication cost of links between agents equals the Euclidean distance between them. They need to form and maintain a formation shape as shown in Fig. 5.

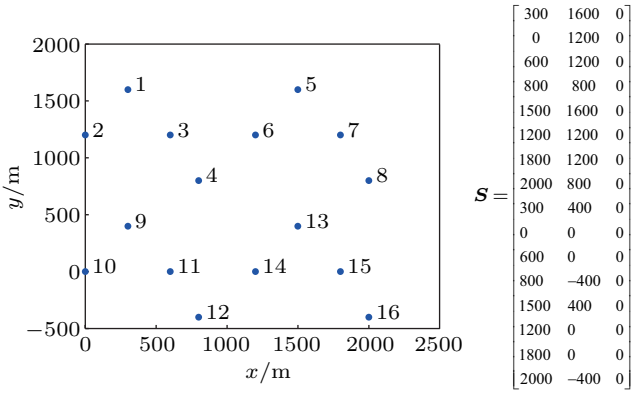


Fig. 5. (color online) Formation shape composed of 16 agents and its mathematical description.

The agents' positions in the formation shape are numbered as 1, 2, ...,16, respectively. Their relative positions in space are shown in the left part in Fig. 5 and their coordinates are shown in the right part in Fig. 5. The leader constraint is $AGENT^* = \{agent_2, agent_6, agent_{10}, agent_{14}\}$.

The process of generating the optimal persistent formation for the above heterogeneous multi-agent system using LC-HMAS-OPFGA is described as follows. The optimal rigid graph, R, is calculated by the algorithm mentioned in Ref. [17], and shown in Fig. 6. Then, the optimal persistent graph, T, corresponding to the above optimal rigid graph, R, is generated by MCA-SP-OPGGA, with the concrete steps as follows.

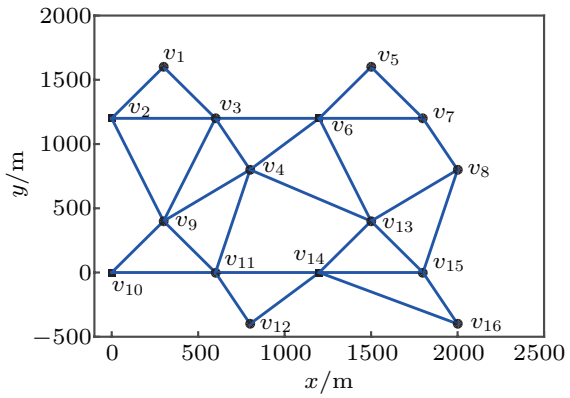


Fig. 6. (color online) Optimal rigid graph R.

After the optimal rigid graph, R, is transformed into a directed graph D_R, T_1', and T_2', the two minimum cost arborescence (shown in Figs. 7(a) and 7(b)), are obtained respectively with the assistance of the virtual node v_0. Then, the directed graph, T, is obtained by combining the two directed graphs, after v_0 has been deleted as shown in Fig. 7(c).

However, neither of $a_{8,15}$ and $a_{15,8}$, the two arcs corresponding to the edge, $e_{8,15}$, in the optimal rigid graph, R, is in T, and the in-degrees of both v_8 and v_{15} are both 2. Therefore, the operation of adding arcs is carried out. Arc, $a_{8,15}$, is added to T as shown in Fig. 7(d), then the node, v_5, with an in-degree less than 2 is found and all the arcs on the path from v_5 to v_{15}

are reversed, so that the in-degrees of all nodes are less than or equal to 2. The directed graph, T, obtained is shown in Fig. 7(e). Similarly, arc a_{39} is added to the directed graph T, and then all the arcs on the path from v_2 to v_9 are reversed to gain the directed graph T as shown in Figs. 7(f) and 7(g). As a result, the in-degrees of all nodes in the directed graph, T, are less than or equal to 2, and both the number of all the arcs in T and that of all the edges in the optimal rigid graph, R, are 29, which meets the conditions of an optimal persistent graph as stated in Lemma 2. Therefore, T is an optimal persistent graph of R.

There is only one node, that is v_1, with an in-degree of 0 in the optimal persistent graph T, but it cannot act as the leader. Thus, T is further adjusted by SP-OPGAA as proposed earlier in this study, with the concrete process as shown in the following.

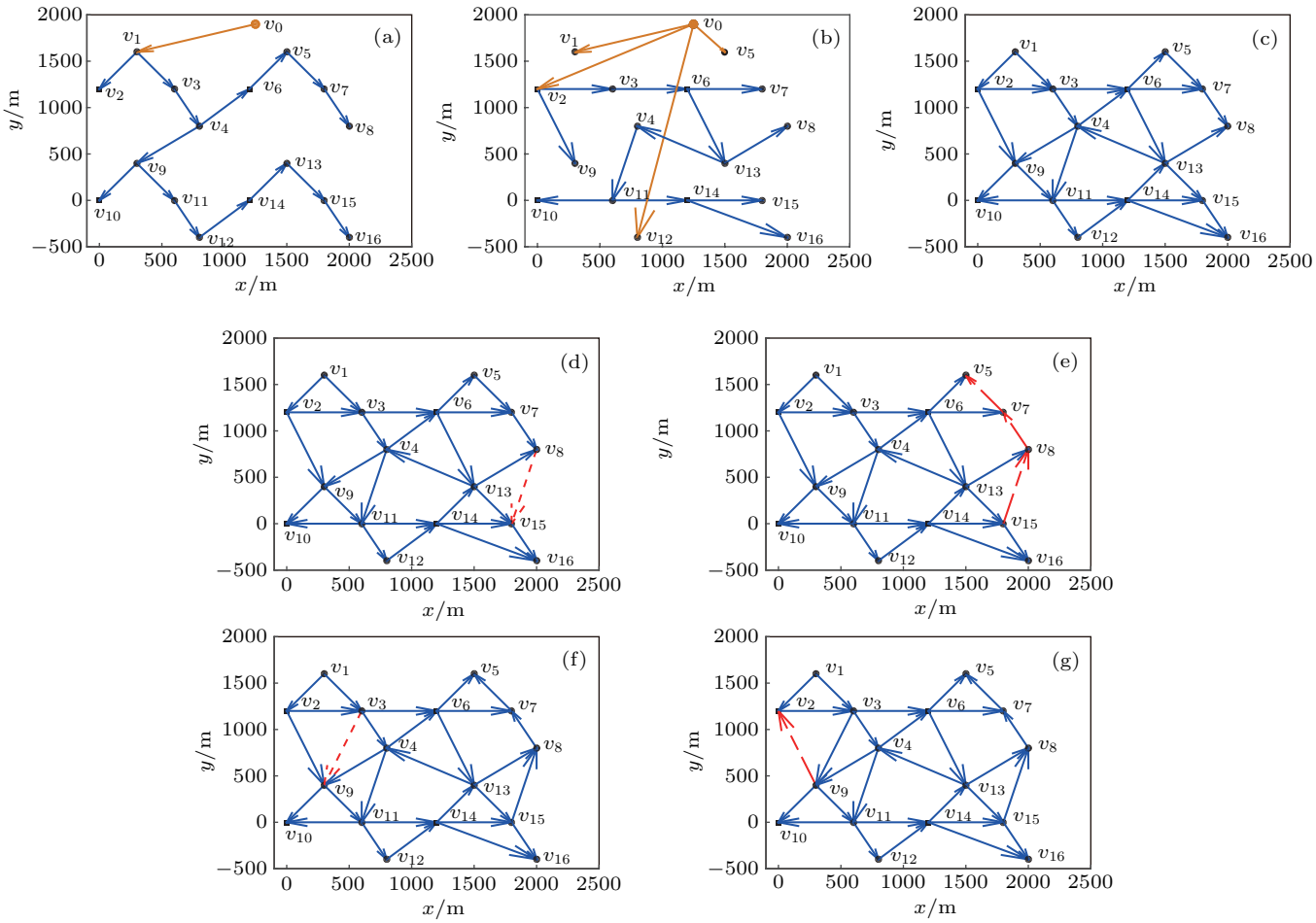


Fig. 7. (color online) Process for MCA-SP-OPGGA to generate the optimal persistent graph, T. (a) The first minimum cost arborescence, T_1' and (b) the second minimum cost arborescence, T_2', (c) directed graph, T, obtained by the combination of T_1 and T_2, (d) directed graph, T, after arc $a_{8,15}$ has been added, (e) directed graph, T, after the path, (v_5, v_7, v_8, v_{15}), has been reversed from v_5 to v_{15}, (f) directed graph, T, after arc a_{39} has been added; (g) directed graph, T, after path, (v_2, v_9), has been reversed from v_2 to v_9.

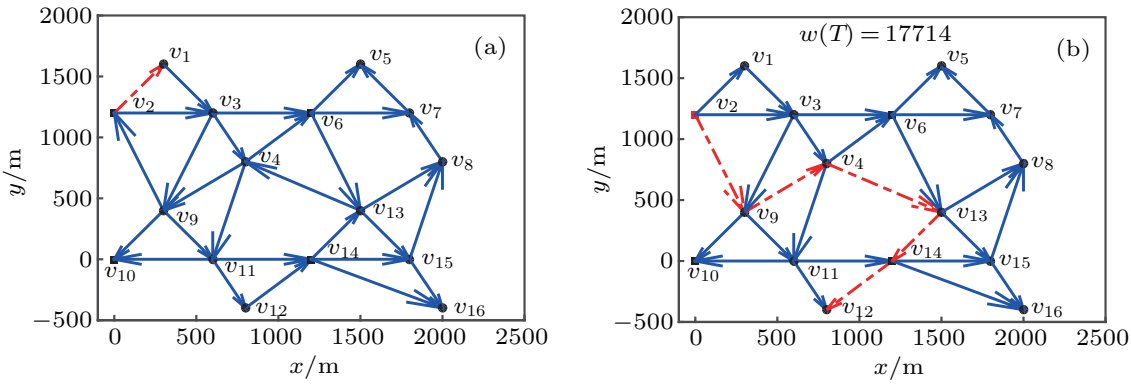


Fig. 8. (color online) Process for SP-OPGAA to generate the optimal persistent graph, T, that satisfies the leader constraint. (a) The optimal persistent graph, T, after reversing the arcs on the path, (v_1, v_2), from v_1 to v_2; (b) The optimal persistent graph, T, after reversing the arcs on the path, $(v_{12}, v_{14}, v_{13}, v_4, v_9, v_2)$, from v_{12} to v_2.

In T, the node v_2 with an in-degree greater than 0 that can act as the leader is chosen and node v_1 with an in-degree less than 2 is found so that there could be a path, (v_1, v_2), from v_1 to v_2. Then all the arcs on this path are reversed to obtain the adjusted optimal persistent graph, T, as shown in Fig. 8(a). Nevertheless, the in-degree of v_2 in T is still greater than 0, which does not meet the leader constraint. Therefore, the above process needs to be repeated to find node v_{12} and reverse all the arcs on the path, $(v_{12}, v_{14}, v_{13}, v_4, v_9, v_2)$, from v_{12} to v_2 to obtain the adjusted optimal persistent graph, T, as shown in Fig. 8(b). As a result, the in-degree of v_2 in T is 0, which meant T is the optimal persistent graph that satisfies the leader constraint, namely the optimal communication topology of this heterogeneous multi-agent persistent formation. $agent_2$ acts as the leader of this formation, and the formation communication cost is 17714.

5. Conclusions

When heterogeneous multiple agents comprise a formation shape, not every agent can act as the leader. In view of this leader constraint, in this study we propose an algorithm to generate the optimal persistent formation for heterogeneous multi-agent systems. First, we generate an optimal persistent graph of the formation network topology by adding a virtual node and virtual arcs, generating and combining minimum cost arborescence, adding arcs, and reversing paths. Then, the optimal persistent graph is adjusted by using path reversal to satisfy the leader constraint. This algorithm supports the generation of the optimal persistent formation for any formation shape in 2D space under any leader constraint. For future research, further consideration will be given to communication failures that may occur in the process when the heterogeneous multi-agent formation keeps its formation shape. In addition, the dynamic optimization of persistent formation for leader-constrained heterogeneous multi-agent systems in the case of communication failure will be investigated.

References

[1] Barrientos A G, Lopez J L, Espinoza E S, Hoyo J and Valencia G 2016 *IEEE Latin America Transactions* **14** 1184

[2] Guo H, Chen S, Bao A M, Behrangi A, Hong Y, Ndayisaba F, Hu J J and Stepanian P M 2016 *Atmospheric Research* **176** 121

[3] Nigam N, Bieniawski S, Kroo I and Vian J 2012 *IEEE T Control Syst. T* **20** 1236

[4] Cao X, Zhu D Q and Yang S X 2016 *IEEE Transactions on Neural Networks and Learning Systems* **27** 2364

[5] Giulietti F, Pollini L and Innocenti M 2000 *IEEE Control Systems* **20** 34

[6] Pollini L, Giulietti F and Innocenti M 2002 *American Control Conference*, May 8–10, 2002, Hilton Anchorage and Egan Convention Center, Anchorage, Alaska, USA, p. 2860

[7] Ren W 2007 *IET Control Theory & Applications* **1** 505

[8] Yang H, Jiang B and Zhang Y M 2014 *International Journal of Control Automation and Systems* **12** 29

[9] Hendrickx J M, Fidan B, Changbin Y, Anderson B D O and Blondel V D 2008 *IEEE T Automat. Control* **53** 968

[10] Yu C and Anderson B D O 2009 *Int. J. Robust Nonlin.* **19** 1427

[11] Motevallian S A, Yu C and Anderson B D O 2015 *Int. J. Robust Nonlin.* **25** 1654

[12] Tousi M M and Khorasani K 2012 *Automatica* **48** 410

[13] Tousi M M and Khorasani K 2015 *Int. J. Control* **88** 90

[14] Hendrickx J M, Fidan B, Yu C, Anderson B D O and Blondel V D 2006 *the 17th International Symposium on Mathematical Theory of Networks and Systems (MTNS*2006), July 24–28, Kyoto, Japan, p. 859

[15] Smith B S, Egerstedt M and Howard A 2009 *Mobile Networks & Applications* **14** 322

[16] Luo X Y, Shao S K, Zhang Y Y, Li S B, Guan X P and Liu Z X 2014 *Chin. Phys. B* **23** 028901

[17] Luo X Y, Li S B and Guan X P 2009 *Chin. Phys. B* **18** 3104

[18] Luo X Y, Shao S K, Guan X P and Zhao Y J 2013 *Acta Automatica Sinica* **39** 1431 (in Chinese)

[19] Luo X Y, Yang F, Li S B and Guan X P 2014 *Acta Automatica Sinica* **40** 1311 (in Chinese)

[20] Summers T H, Yu C B, Dasgupta S and Anderson B D O 2011 *IEEE T. Automat. Control* **56** 2772

[21] Gabow H N, Galil Z, Spencer T and Tarjan R E 1986 *Combinatorica* **6** 109

[22] Dijkstra E W 1959 *Numerische Mathematics* **1** 269

[23] Ren R, Zhang Y Y, Luo X Y and Li S B 2010 *International Journal of Automation and Computing* **7** 557