Received September 5, 2018, accepted October 2, 2018, date of publication October 12, 2018, date of current version November 9, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2875679

Integrated Topology Management in Flying Ad Hoc Networks: Topology Construction and Adjustment

DO-YUP KIM, (Student Member, IEEE), and JANG-WON LEE, (Senior Member, IEEE)
Department of Electrical and Electronic Engineering, Yonsei University, Seoul 03722, South Korea

Corresponding author: Jang-Won Lee (jangwon@yonsei.ac.kr)

This work was supported in part by the Unmanned Vehicles Advanced Core Technology Research and Development Program through the National Research Foundation of Korea and in part by the Unmanned Vehicle Advanced Research Center through the Ministry of Science and ICT, South Korea, under Grant NRF-2018M1B3A1A02937507.

ABSTRACT Flying ad hoc networks (FANETs) that consist of multiple unmanned aerial vehicles (UAVs) are promising technologies for future networked systems due to the versatility of UAVs. One of the most distinguishing features of FANET is frequent and rapid topological fluctuations due to the high-mobility of UAVs. Hence, the topology management adapting to the movements of UAVs is one of the most critical issues in FANET. In this paper, we study a FANET topology management problem that optimizes the locations and movements of UAVs to maximize the network performance, adapting to the topological changes while UAVs carry out their missions. When formulating the problem, we take into account the routing protocol as an arbitrary function since the network performance is inseparably linked with the routing protocol in use. We first develop two algorithms. One is the topology construction algorithm, which constructs a FANET topology from the scratch without any given initial topology, based on particle swarm optimization. The other is the topology adjustment algorithm, which incrementally adjusts the FANET topology adapting to the movements of UAVs with low-computational costs, based on gradient descent. Then, by defining a logical distance (the so-called topology edit distance) that measures the degree of changes in FANET topology, we develop an integrated topology management algorithm that contains the topology construction and adjustment algorithms. The simulation results show that our algorithm achieves a good network performance with low computational overhead, which is one of the most essential virtues in FANETs with rapidly varying topology.

INDEX TERMS Flying ad hoc network (FANET), gradient descent, particle swarm optimization (PSO), relay deployment, topology edit distance, topology management, unmanned aerial vehicle (UAV), UAV deployment, UAV movement.

I. INTRODUCTION

Unmanned aerial vehicles (UAVs) are flying devices that do not require on-board pilots. Since the employment of UAVs can ensure the safety of pilots, UAVs have been primarily used to perform dangerous missions for military purpose in the beginning [1]–[3]. However, thanks to advances in embedded systems and battery technologies over the past few years, both production costs and physical sizes of UAVs have been significantly reduced and the operational lifetime of UAVs has been considerably extended. Along with these developments, UAVs have begun to be used not only in the military applications but also in public and civilian applications, such as 3-dimensional aerial mapping for earthwork projects [4], management of rangeland [5], and post-disaster rescue [6]. In addition, the advances have paved the way for networks that employ multiple UAVs [7]–[10], called flying ad hoc networks (FANETs).

FANET has received increasing interest from both academia and industry due to its enormous potential, such as high reliability, survivability, and scalability [11]–[14]. For example, since UAVs can be quickly and easily deployed [15], if necessary, not only can the network coverage be easily increased [16], but the network topology can also be more stable and robust to node failures [17].

Moreover, FANET in which ad hoc communications between UAVs is supported can release the massive constraint that UAVs must maintain direct links with any ground control station (GCS) at which a mission commander is located. At the expense of getting these benefits, more considerations are required in FANET, such as aerial communication architecture, topology management, routing protocol, high mobility of UAVs, collision between UAVs, and so forth.

Most FANET applications [18], [19] are intended for some UAVs to perform specific missions (e.g., search and rescue, surveillance and reconnaissance, environmental sensing, etc.), and then report the resultant data to the mission commander. Hence, the reliable communication is necessarily required between each mission-performing UAV (MU) and a GCS at which its corresponding mission commander is located. To this end, relay UAVs (RUs) that play a role in relaying data can be additionally used to support the reliable communications. However, since the mobility of MUs must be highly dependent on their given missions, changes in a FANET topology by their mission-based movements may severely degrade the network performance [20]. Hence, the topology management, which includes deployment of RUs and their mobility control to maintain proper network performance, has been one of the most critical research issues, especially in dynamic networks such as FANET.

However, finding the optimal locations of multiple RUs, i.e., solving the so-called relay node deployment problem, is well known to be one of the most challenging problems, which is proven to be NP-hard [21]. Due to the complexity and difficulty of the relay node deployment problem, over the past decades, many researchers [22]–[28] have attempted to solve the problem in static networks, such as wireless sensor networks, without considering topological changes. They have developed various heuristic algorithms that yield (sub-)optimal solutions under their assumptions. However, such (sub-)optimality may be lost over time in dynamic networks, such as FANETs, mobile ad hoc networks (MANETs), and vehicular ad hoc networks (VANETs). Accordingly, to address such an issue of the (sub-)optimality loss in dynamic networks, the researchers are required upon how network topologies can be dynamically managed to adapt to their topological changes.

Roh and Lee [29] have considered a linear network topology in one-dimensional VANET consisting of controllable and uncontrollable vehicular nodes. By modeling the communication quality as a function of link distance, they have solved a node deployment problem that maximizes the minimum data rate among all links. Roh and Lee [30] have extended their previous work from one-dimensional VANET to two-dimensional MANET. With consideration of the trade-off between mission rewards and overall data rates, they have developed a node placement algorithm that maximizes the weighted sum of the mission rewards and the overall data rates. However, in both [29] and [30] , they have assumed that a set of wireless links of the network is given and fixed. Whereas this assumption may be feasible in dynamic networks with significantly low topological changes, it cannot be apply to FANET in which link creation and destruction are frequent because of the UAVs' high mobility.

In [31]–[34], without assuming that a set of wireless links of the network is given and fixed, the authors have considered UAV-assisted MANET in which UAVs play a role as relay nodes. Han *et al.* [31] have presented four network connectivity metrics: global message, worst case, network bisection, and k-connectivity. Then, they presented gradient-based algorithms that find a UAV's location to maximize the network connectivity. Kim *et al.* [32] have extended the work of [31] into a problem that finds multiple UAVs' locations and their future trajectories to maximize the global message connectivity. Ladosz *et al.* [33] have solved a UAV deployment problem to maximize a new network connectivity metric, which generalizes the global message connectivity and the worst case connectivity. Dengiz *et al.* [34] have presented algorithms that find velocities of relay nodes in each time step to maximize the network connectivity primarily, with secondarily improving the wireless communication quality of the worst link. Even though various problems have been considered in [31]–[34], they all have assumed that any initial network topology is given in advance, and tried to improve the performance by additionally deploying some UAVs. Hence, their solution can provide only marginal improvement over the performance of the given initial topology, and thus the efficiency of their algorithms highly depends upon the given initial topology. Nonetheless, they have not provided any method to find a good initial topology.

Despite such efforts on the topology management in dynamic networks, there have been few works to deploy relay nodes to construct an initial network topology from the scratch, and to manage a network topology adapting to the topological fluctuations. Recently, Magán-Carrión *et al.* [35] have studied the topology management problem in MANET without assuming that any initial topology is given. The authors have attempted to maximize the network connectivity primarily and enhance the wireless communication performance secondarily, by deploying and moving relay nodes. To this end, they have proposed an algorithm that finds the locations of relay nodes for the next time step using the particle swarm optimization (PSO) algorithm with a high computational overhead. Their algorithm requires to run the PSO algorithm one or two times in each time step, and thus its computational overhead should be immensely high. Such high computational overhead can be a fatal drawback for use in FANET with fast and frequent topological changes.

Furthermore, in all of the above works [29]–[35], the routing protocol has not been explicitly considered even though it is closely related to the network performance in practice. Since the links that are not in any active routing path do not affect the network performance or little, if any, the optimal network topology is strongly dependent on the routing protocol in use, and vice versa. Hence, in order to manage the network topology more efficiently considering the network performance, it is important to take into account the routing protocol in use.

In this paper, we study a FANET topology management problem which explicitly reflects the routing protocol. Two types of UAVs are considered: MUs and RUs. Each MU performs its mission assigned by a mission commander and then create resultant data to send to the GCS at which its corresponding mission commander is located, and each RU plays a role in relaying data between adjacent nodes. We assume that the mobility of MUs is determined only by their missions, and thus it cannot be controlled to improve network performance. On the other hand, the mobility of RUs can be freely controlled to achieve high network performance. In our topology management problem, we deal with constructing an initial FANET topology from the scratch and adjusting the constructed topology adapting to the topological changes arising from the unpredictable mission-based movements of MUs. To this end, we first develop the topology construction and adjustment algorithms. We apply the PSO algorithm to the topology construction algorithm since a highly complicated problem should be solved to construct a FANET topology from the scratch. Whereas, we employ the gradient descent method with a low computational complexity to the topology adjustment algorithm since the topology should be quickly adjusted in response to the fast topological changes. In addition, our topology adjustment algorithm can be implemented in a distributed fashion if some conditions are met. Finally, by proposing a topology edit distance that measures the degree of topological changes and using it together with the two developed algorithms, we develop an integrated topology management algorithm that decides adaptively whether to construct or adjust the FANET topology or update the routing paths.

The rest of this paper is organized as follows. In Section II, we introduce the system model and formulate the optimization problem for the FANET topology management. Section III addresses the details of our proposed algorithms. The performance evaluation with simulation results is provided in Section IV, followed by conclusions in Section V.

II. SYSTEM MODEL AND PROBLEM FORMULATION

In this section, we first describe the system model and then formulate the FANET topology management problem. We consider a FANET that consists of a number of GCSs, MUs, and RUs, whose sets are denoted by $\mathcal{G} = \{g_k\}_{k=1}^{|\mathcal{G}|}$, $\mathcal{M} = \{m_k\}_{k=1}^{|\mathcal{M}|}$, and $\mathcal{R} = \{r_k\}_{k=1}^{|\mathcal{R}|}$, respectively, where $|\cdot|$ stands for the set cardinality. We aim to manage the FANET topology so that each MU can communicate well with a GCS which assigns a mission to it. We assume that, for each MU m, its corresponding GCS is given and denoted by $g^{(m)}$. Regarding all the GCSs and the UAVs as wireless nodes in FANET, we denote the location of a node v by $\boldsymbol{x}_v \in \mathbb{R}^3$. The location sets of the GCSs, the MUs, and the RUs are denoted by $\mathcal{X}_\mathcal{G} = \{\boldsymbol{x}_v\}_{v\in\mathcal{G}}$, $\mathcal{X}_\mathcal{M} = \{\boldsymbol{x}_v\}_{v\in\mathcal{M}}$, and $\mathcal{X}_\mathcal{R} = \{\boldsymbol{x}_v\}_{v\in\mathcal{R}}$, respectively. We assume that the locations of the GCSs are given and fixed. We also assume that the location and mobility of each MU are determined only based on its given mission (e.g., an MU will keep moving following a specific mobile object if its mission is monitoring the object), and thus we cannot control them to improve the network performance.

We now consider a routing protocol that gives end-to-end ad hoc routing paths between the MUs and their corresponding GCSs. We assume that the routing protocol in use is known, no matter what routing protocol is adopted, and we treat it as an arbitrary function, which takes the locations of all nodes as input and outputs a set of routing paths between each MU m and its corresponding GCS $g^{(m)}$ for all $m \in \mathcal{M}$. Thus, in this paper, without assuming that any specific routing protocol should be used, we consider a general routing protocol as

$$\rho : \{\mathcal{X}_\mathcal{G}, \mathcal{X}_\mathcal{M}, \mathcal{X}_\mathcal{R}\} \mapsto \{\boldsymbol{\rho}_m\}_{m\in\mathcal{M}}, ^{1} \tag{1}$$

where $\boldsymbol{\rho}_m$ is an ordered set of nodes that represents the routing path between the MU m and the GCS $g^{(m)}$, where the kth element of the routing path $\boldsymbol{\rho}_m$ is denoted by ρ_m^k. Note that the first element, ρ_m^1, and the last element, $\rho_m^{|\boldsymbol{\rho}_m|}$, should be m and $g^{(m)}$, respectively, and the rest elements are intermediate RUs, i.e., $\rho_m^i \in \mathcal{R}$ for $i = 2, \ldots, |\boldsymbol{\rho}_m| - 1$. For example, suppose that an MU m_3 is performing a mission assigned by a GCS g_2, i.e., $g^{(m_3)} = g_2$. In compliance with the routing protocol in use, if the data generated by the MU m_3 are delivered through three intermediate RUs r_3, r_1, and r_5 in turn to its corresponding GCS g_2, then the routing path is given as $\boldsymbol{\rho}_{m_3} = \{m_3, r_3, r_1, r_5, g_2\}$.

In wireless networks, in practice, a signal can be successfully decoded if the received signal-to-noise ratio (SNR) exceeds a certain level. Due to the fact that the average power of the wireless signal decays exponentially with the propagation distance, we consider that two nodes can reliably communicate if they are within a certain communication range, d_{cm}. To ensure reliable end-to-end communication between all MUs and their corresponding GCSs, the lengths of all wireless links in the active routing paths must not be greater than d_{cm}. Hence, we need to consider the following end-to-end communication constraint:

$$\max_{k=1,\ldots,|\boldsymbol{\rho}_m|-1} \delta(\rho_m^k, \rho_m^{k+1}) \le d_{cm}, \quad \forall m \in \mathcal{M}, \tag{2}$$

where $\delta(u, v)$ denotes the geometric distance between two nodes u and v. That is, $\delta(u, v) = \|\boldsymbol{x}_u - \boldsymbol{x}_v\|_2$, where $\|\cdot\|_2$ stands for the ℓ^2-norm, i.e., the Euclidean distance from the origin.

In addition, we need to concern about a risk of UAVs crashing due to the fact that they may move with high speed. Even if the UAVs are not flying at high speeds, some external

[1]For the sake of a simple notation, if there is no confusion, we will use $\{\boldsymbol{\rho}_m\}_{m\in\mathcal{M}}$ and $\{\boldsymbol{\rho}_m\}$ interchangeably.

environments such as strong winds can disturb their hovering at certain locations. Hence, we consider the following safety constraint:

$$\min_{\substack{u,v \in \mathcal{M} \cup \mathcal{R}, \\ u \neq v}} \delta(u,v) \geq d_{sf}, \tag{3}$$

where d_{sf} denotes the minimum safety distance to prevent collision between UAVs. Note that the safety distance d_{sf} must be much smaller than the communication distance d_{cm}. We assume that the safety distance is maintained between MUs since their mobility should be controlled based on their given missions, and the mobility control of MUs is out of the scope of this paper.

Now, we consider the performance metric that evaluates the performance of FANET. Instead of any specific performance metric, we consider a general performance metric that can be defined in a various way in accordance with the purpose of the system. Hence, we define the performance metric function as

$$f : \{\mathcal{X}_{\mathcal{G}}, \mathcal{X}_{\mathcal{M}}, \mathcal{X}_{\mathcal{R}}, \rho(\mathcal{X}_{\mathcal{G}}, \mathcal{X}_{\mathcal{M}}, \mathcal{X}_{\mathcal{R}})^2\} \mapsto y, \tag{4}$$

where the input to the performance metric function is the locations of all nodes in FANET and the routing paths obtained by the routing function with respect to the locations of the nodes, and its output is a real number indicating the network performance of FANET. We assume that the performance metric function is partially differentiable with respect to $\boldsymbol{x}_r$ for each $r \in \mathcal{R}$, and that the smaller its value is, the better the network performance of FANET is. For example, the performance metric function can be defined by the sum of the αth power of the link distances over all active routing paths between MUs and their corresponding GCSs, i.e.,

$$f(\mathcal{X}_{\mathcal{G}}, \mathcal{X}_{\mathcal{M}}, \mathcal{X}_{\mathcal{R}}, \rho) = \sum_{m \in \mathcal{M}} \sum_{k=1}^{|\boldsymbol{\rho}_m|-1} \delta(\rho_m^k, \rho_m^{k+1})^{\alpha}, \tag{5}$$

where the parameter α represents how much the link distance affects its communication quality. As another example, we can consider the performance metric function as the longest link distance among all links belonging to the active routing paths, i.e.,

$$f(\mathcal{X}_{\mathcal{G}}, \mathcal{X}_{\mathcal{M}}, \mathcal{X}_{\mathcal{R}}, \rho) = \max_{m \in \mathcal{M}} \max_{k=1,\ldots,|\boldsymbol{\rho}_m|-1} \delta(\rho_m^k, \rho_m^{k+1}). \tag{6}$$

The function (6) is usually not differentiable due to the maximum functions. However, we can easily convert it into the smooth and differentiable one using the softmax function or the quasimax function [36].

Based on the routing function (1), the end-to-end communication constraint (2), the safety constraint (3), and the performance metric function (4), the problem of the FANET topology management is finally formulated as follows:

$$\begin{aligned} &\underset{\mathcal{X}_{\mathcal{R}} \in \mathcal{S}^{|\mathcal{R}|}}{\text{minimize}} \; f(\mathcal{X}_{\mathcal{G}}, \mathcal{X}_{\mathcal{M}}, \mathcal{X}_{\mathcal{R}}, \rho) \\ &\text{subject to} \max_{k=1,\ldots,|\boldsymbol{\rho}_m|-1} \delta(\rho_m^k, \rho_m^{k+1}) \leq d_{cm}, \quad \forall m \in \mathcal{M}, \\ &\qquad\qquad \min_{\substack{u,v \in \mathcal{M} \cup \mathcal{R}, \\ u \neq v}} \delta(u,v) \geq d_{sf}, \end{aligned} \tag{7}$$

where $\mathcal{S} \subseteq \mathbb{R}^3$ is a 3-dimensional deployable space for UAVs.

III. FANET TOPOLOGY MANAGEMENT ALGORITHMS

In this section, to solve our FANET topology management problem (7), we develop an integrated FANET topology management algorithm that consists of two sub-algorithms. We first develop a FANET topology construction algorithm that finds the optimal locations of RUs to construct an initial topology from the scratch for given MUs' locations. We then develop a FANET topology adjustment algorithm with low computational complexity that controls the movements of RUs adapting to the topological changes arising from the mission-based mobility of MUs. By combining the FANET topology construction and adjustment algorithms and introducing a so-called topology edit distance, we finally develop the integrated FANET topology management algorithm that effectively manages the FANET topology to maintain proper network performance while satisfying the end-to-end communication constraint (2) and the safety constraint (3).

A. FANET TOPOLOGY CONSTRUCTION

Our optimization problem (7) is not only non-convex but also NP-hard [21], [37]. Hence, in order to solve the problem, we use one of the famous nature-inspired metaheuristic algorithms, which is the PSO algorithm [38]. Hereafter, we first explain the rudimentary PSO algorithm and then develop our PSO-based FANET topology construction algorithm.

1) RUDIMENTARY PSO ALGORITHM

Unlike trajectory-based algorithms that use only one agent at a time like a simulated annealing algorithm [39], the PSO algorithm is a population-based stochastic algorithm that employs a population of agents, where the population and the agents are called a swarm and particles, respectively. Each particle is characterized as a tuple of its position and velocity, where the position represents a candidate solution for an optimization problem and the velocity represents a displacement of the particle's position per iteration. In the PSO algorithm, each particle iteratively explores a better position in the search space of the optimization problem, and the best-so-far position at the end of the iterations is chosen as the final solution. Note that there is a trade-off between complexity and diversity according to the size of the swarm, i.e., the number of the particles. As the number of the particles increases, the diversity and the exploration will be greater, but the computational complexity will be also higher. Main

[2]For the sake of a simple notation, in the following, instead of the full notation $\rho(\mathcal{X}_{\mathcal{G}}, \mathcal{X}_{\mathcal{M}}, \mathcal{X}_{\mathcal{R}})$, we will use only ρ if there is no confusion.

Algorithm 1 Particle Swarm Optimization (PSO)

```
Output: $g$
1  Set $N_p$, $w$, $c_1$, and $c_2$.
2  Initialize all particles in $\mathcal{P}$.
3  repeat
4      for each particle $i \in \mathcal{P}$ do
5          Update $v_i$ according to (9).
6          Update $z_i$ according to (10).
7          if $h(z_i) \leq h(p_i)$ then
8              $p_i \leftarrow z_i$.
9          end
10     end
11     $j \leftarrow \arg\min_i h(p_i)$.
12     $g \leftarrow p_j$.
13 until termination conditions are satisfied
```

advantages of the PSO algorithm are high accuracy, easy implementability, and fast convergence.

For convenience of exposition, let us consider the following optimization problem:

$$\underset{z \in \mathbb{R}^n}{\text{minimize}} \quad h(z), \tag{8}$$

where z is the decision variable, $\mathbb{R}^n$ is the search space, and $h : \mathbb{R}^n \to \mathbb{R}$ is the objective function. Denoting a swarm of N_p particles by $\mathcal{P}$, each particle $i \in \mathcal{P}$ is characterized as $(z_i, \boldsymbol{v}_i)$, where z_i and $\boldsymbol{v}_i$ are its position and velocity, respectively. At the beginning of the PSO algorithm, each particle is initialized with the position randomly chosen in the search space and the zero velocity.

Prior to explaining the stochastic update process of the particles, we introduce two special parameters: the pBest $\boldsymbol{p}_i$ and the gBest $\boldsymbol{g}$. The pBest $\boldsymbol{p}_i$ is the individual best-so-far position for each particle i, i.e., the position better than any ones that the particle i has been before, and the gBest $\boldsymbol{g}$ is the global best-so-far position, i.e., the best position thus far in the whole particle swarm. Depending on the pBest $\boldsymbol{p}_i$ and the gBest $\boldsymbol{g}$, the velocity of each particle i is updated as

$$\boldsymbol{v}_i^{l+1} = w\boldsymbol{v}_i^l + c_1\boldsymbol{u}_1 \circ (\boldsymbol{p}_i^l - z_i^l) + c_2\boldsymbol{u}_2 \circ (\boldsymbol{g}^l - z_i^l), \quad \forall i \in \mathcal{P}, \tag{9}$$

where the index l denotes the lth iteration of the PSO algorithm, w is the inertia weight, c_1 and c_2 are cognitive and social parameters, respectively, both $\boldsymbol{u}_1$ and $\boldsymbol{u}_2$ are independent uniformly distributed random vectors on $[0, 1]^n$, and the operator $\circ$ stands for the Hadamard product. The inertia term regulates the drastic change of the velocity, and the cognitive and social terms balance between exploitation (local search) and exploration (global search). After the velocity is updated, its position is updated by adding the updated velocity as

$$z_i^{l+1} = z_i^l + \boldsymbol{v}_i^{l+1}, \quad \forall i \in \mathcal{P}. \tag{10}$$

All of the particles are iteratively updated in the search space according to (9) and (10) until termination conditions are satisfied, where the termination conditions are generally set to the number of iterations, a rate of convergence of the particle swarm, a given time period, a time period of failure to find a better gBest, and so forth. At the moment when the PSO algorithm terminates, the gBest becomes the final solution obtained by the PSO algorithm for the optimization problem (8). The pseudo-code is described in Algorithm 1.

2) FANET TOPOLOGY CONSTRUCTION ALGORITHM

We now explain our FANET topology construction algorithm that solves the FANET topology management problem (7), given the fixed locations of GCSs, the current locations of MUs, and the routing function representing the routing protocol in use. Our FANET topology construction algorithm makes use of the PSO algorithm which works well with unconstrained optimization problems like (8). Hence, we first replace our constrained optimization problem with an unconstrained one using the penalty method [40], which omits the constraints by penalizing the objective function for the constraint violations. Thus, the reformulated FANET topology management problem is defined as

$$\underset{\mathcal{X}_\mathcal{R} \in \mathcal{S}^{|\mathcal{R}|}}{\text{minimize}} \quad \hat{f}(\mathcal{X}_\mathcal{G}, \mathcal{X}_\mathcal{M}, \mathcal{X}_\mathcal{R}, \rho), \tag{11}$$

where $\hat{f}$ is the penalized performance metric function given as

$$\begin{aligned}
&\hat{f}(\mathcal{X}_\mathcal{G}, \mathcal{X}_\mathcal{M}, \mathcal{X}_\mathcal{R}, \rho) \\
&\quad = f(\mathcal{X}_\mathcal{G}, \mathcal{X}_\mathcal{M}, \mathcal{X}_\mathcal{R}, \rho) \\
&\qquad + \sum_{m \in \mathcal{M}} \lambda_m \left(\left[\max_{k=1,\ldots,|\boldsymbol{\rho}_m|-1} \delta(\rho_m^k, \rho_m^{k+1}) - d_{cm} \right]^+ \right)^2 \\
&\qquad + \mu \left(\left[d_{sf} - \min_{\substack{u,v \in \mathcal{M} \cup \mathcal{R}, \\ u \neq v}} \delta(u, v) \right]^+ \right)^2,
\end{aligned} \tag{12}$$

where $\{\lambda_m\}_{m \in \mathcal{M}}$ and μ are penalty coefficients corresponding to the end-to-end communication constraint (2) and the safety constraint (3), respectively, and $[\cdot]^+$ stands for $\max\{0, \cdot\}$. For the sufficiently large penalty coefficients, the solution $\mathcal{X}_\mathcal{R}^*$ that minimizes the penalized performance metric function $\hat{f}$ solves the original FANET topology management problem (7). In our algorithm, each particle i's position and velocity[3] are denoted by $\mathcal{X}_{\mathcal{R},i}$ and $\mathcal{V}_{\mathcal{R},i}$, respectively. We denote a particle i's individual best-so-far position and the global best-so-far position by $\mathcal{X}_{\mathcal{R},i}^*$ and $\mathcal{X}_\mathcal{R}^*$, respectively. The routing paths corresponding to $\mathcal{X}_{\mathcal{R},i}$, $\mathcal{X}_{\mathcal{R},i}^*$, and $\mathcal{X}_\mathcal{R}^*$ are denoted by $\{\boldsymbol{\rho}_m\}_i$, $\{\boldsymbol{\rho}_m\}_i^*$, and $\{\boldsymbol{\rho}_m\}^*$, respectively.

Our proposed FANET topology construction algorithm is described in Algorithm 2. In the preparatory stage (lines 1-8), we first set parameters such as the penalty coefficients

[3]Note that, as in the PSO algorithm, each particle's position is the candidate solution, i.e., the set of RUs' locations. In order to prevent terminological confusion, we differently use two terminologies, "position" and "location." "Position" is used for particles, whereas "location" is used for UAVs. In addition, each particle's velocity is not the actual velocity of UAVs but the displacement of its position per iteration of the PSO algorithm.

Algorithm 2 FANET Topology Construction

```
   Input: $d_{cm}$, $d_{sf}$, $\mathcal{X}_{\mathcal{G}}$, $\mathcal{X}_{\mathcal{M}}$, $\rho$
   Output: $\mathcal{X}^*_{\mathcal{R}}$
1  Set $\{\lambda_m\}_{m\in M}$, $\mu$, $N_p$, $w$, $c_1$, and $c_2$.
2  for each particle $i \in \mathcal{P}$ do
3      Initialize $\mathcal{X}_{\mathcal{R},i}$ and $\mathcal{V}_{\mathcal{R},i}$.
4      $\mathcal{X}^*_{\mathcal{R},i} \leftarrow \mathcal{X}_{\mathcal{R},i}$.
5      $\{\boldsymbol{\rho}_m\}^*_i \leftarrow \rho(\mathcal{X}_{\mathcal{G}}, \mathcal{X}_{\mathcal{M}}, \mathcal{X}^*_{\mathcal{R},i})$.
6  end
7  $j \leftarrow \arg\min_i \hat{f}(\mathcal{X}_{\mathcal{G}}, \mathcal{X}_{\mathcal{M}}, \mathcal{X}^*_{\mathcal{R},i}, \{\boldsymbol{\rho}_m\}^*_i)$.
8  $\mathcal{X}^*_{\mathcal{R}} \leftarrow \mathcal{X}^*_{\mathcal{R},j}$ and $\{\boldsymbol{\rho}_m\}^* \leftarrow \{\boldsymbol{\rho}_m\}^*_j$.
9  repeat
10     for each particle $i \in \mathcal{P}$ do
11         Update $\mathcal{V}_{\mathcal{R},i}$ according to (13) and (14).
12         Update $\mathcal{X}_{\mathcal{R},i}$ according to (15).
13         $\{\boldsymbol{\rho}_m\}_i \leftarrow \rho(\mathcal{X}_{\mathcal{G}}, \mathcal{X}_{\mathcal{M}}, \mathcal{X}_{\mathcal{R},i})$.
14         if $\hat{f}(\mathcal{X}_{\mathcal{G}}, \mathcal{X}_{\mathcal{M}}, \mathcal{X}_{\mathcal{R},i}, \{\boldsymbol{\rho}_m\}_i) <$
           $\hat{f}(\mathcal{X}_{\mathcal{G}}, \mathcal{X}_{\mathcal{M}}, \mathcal{X}^*_{\mathcal{R},i}, \{\boldsymbol{\rho}_m\}^*_i)$ then
15             $\mathcal{X}^*_{\mathcal{R},i} \leftarrow \mathcal{X}_{\mathcal{R},i}$ and $\{\boldsymbol{\rho}_m\}^*_i \leftarrow \{\boldsymbol{\rho}_m\}_i$.
16         end
17     end
18     $j \leftarrow \arg\min_i \hat{f}(\mathcal{X}_{\mathcal{G}}, \mathcal{X}_{\mathcal{M}}, \mathcal{X}^*_{\mathcal{R},i}, \{\boldsymbol{\rho}_m\}^*_i)$.
19     if $\hat{f}(\mathcal{X}_{\mathcal{G}}, \mathcal{X}_{\mathcal{M}}, \mathcal{X}^*_{\mathcal{R},j}, \{\boldsymbol{\rho}_m\}^*_j) <$
       $\hat{f}(\mathcal{X}_{\mathcal{G}}, \mathcal{X}_{\mathcal{M}}, \mathcal{X}^*_{\mathcal{R}}, \{\boldsymbol{\rho}_m\}^*)$ then
20         $\mathcal{X}^*_{\mathcal{R}} \leftarrow \mathcal{X}^*_{\mathcal{R},j}$ and $\{\boldsymbol{\rho}_m\}^* \leftarrow \{\boldsymbol{\rho}_m\}^*_j$.
21     end
22 until termination conditions are satisfied
```

$\{\lambda_m\}_{m\in\mathcal{M}}$ and μ, the number of particles N_p, the inertia weight w, and the cognitive and social parameters c_1 and c_2 (line 1). Then, denoting a swarm of all particles by $\mathcal{P}$, for each particle $i \in \mathcal{P}$, we initialize its position $\mathcal{X}_{\mathcal{R},i}$ randomly in the search space $\mathcal{S}^{|\mathcal{R}|}$ and its velocity $\mathcal{V}_{\mathcal{R},i}$ to zero. Since the particle i is initialized for the first time, its position automatically becomes its individual best-so-far position $\mathcal{X}^*_{\mathcal{R},i}$, and the routing paths corresponding to $\mathcal{X}^*_{\mathcal{R},i}$ are obtained by the routing function as $\{\boldsymbol{\rho}_m\}^*_i$ (lines 2-6). After completing the initialization of all the particles in $\mathcal{P}$, we find the particle j whose individual best-so-far position and its corresponding routing paths give the smallest penalized performance metric value. Then, the global best-so-far position $\mathcal{X}^*_{\mathcal{R}}$ and its corresponding routing paths $\{\boldsymbol{\rho}_m\}^*$ are set to the individual best-so-far position $\mathcal{X}^*_{\mathcal{R},j}$ of the particle j and its corresponding routing paths $\{\boldsymbol{\rho}_m\}^*_j$, respectively (lines 7-8).

In the iterative stage (lines 9-22), we iteratively update each particle i's velocity $\mathcal{V}_{\mathcal{R},i}$, position $\mathcal{X}_{\mathcal{R},i}$, and its corresponding routing paths $\{\boldsymbol{\rho}_m\}_i$, and then its individual best-so-far position $\mathcal{X}^*_{\mathcal{R},i}$ and its corresponding routing paths $\{\boldsymbol{\rho}_m\}^*_i$ are updated. After updating all the particles, the global best-so-far position $\mathcal{X}^*_{\mathcal{R}}$ and its corresponding routing paths $\{\boldsymbol{\rho}_m\}^*$ are updated to the best ones thus far. In detail, the velocity of each particle $i \in \mathcal{P}$ is updated stochastically depending on both its individual best-so-far position and the global best-so-far position as

$$\mathcal{V}^{l+1}_{\mathcal{R},i} = w\mathcal{V}^{l}_{\mathcal{R},i} + c_1\boldsymbol{u}_1 \circ (\mathcal{X}^{*l}_{\mathcal{R},i} - \mathcal{X}^{l}_{\mathcal{R},i}) + c_2\boldsymbol{u}_2 \circ (\mathcal{X}^{*l}_{\mathcal{R}} - \mathcal{X}^{l}_{\mathcal{R},i}), \quad \forall i \in \mathcal{P}, \tag{13}$$

where the index l denotes the lth iteration of our FANET topology construction algorithm, and both $\boldsymbol{u}_1$ and $\boldsymbol{u}_2$ are independent uniformly distributed random vectors on $[0, 1]^{3\times|\mathcal{R}|}$. We note that the velocity may be exploded to a large value greater than the range of the search space, especially for the particles whose positions are much far from their own individual best-so-far position or the global best-so-far position. Hence, we use a velocity clamping method [41] that prevents the PSO algorithm from diverging. Each particle i's velocity $\mathcal{V}^{l+1}_{\mathcal{R},i}$ is clamped according to

$$v^{l+1}_{i,j} = \begin{cases} v^{l+1}_{i,j} & \text{if } v^{l+1}_{i,j} \in [-V^{max}_j, V^{max}_j], \\ -V^{max}_j & \text{if } v^{l+1}_{i,j} < -V^{max}_j, \\ V^{max}_j & \text{otherwise,} \end{cases} \tag{14}$$

where $v^{l+1}_{i,j}$ is jth element of $\mathcal{V}^{l+1}_{\mathcal{R},i}$, and V^{max}_j is the threshold for the velocity clamping. The velocity clamping threshold V^{max}_j is typically set at about 10 to 20 percent of the range of the search space [42]. After then, the position of each particle i is updated as

$$\mathcal{X}^{l+1}_{\mathcal{R},i} = \mathcal{X}^{l}_{\mathcal{R},i} + \mathcal{V}^{l+1}_{\mathcal{R},i}, \quad \forall i \in \mathcal{P}. \tag{15}$$

The routing paths corresponding to the updated position of the particle i are obtained by the routing function as $\{\boldsymbol{\rho}_m\}_i$. We now update the individual best-so-far position and its corresponding routing paths to the newly updated one if the updated one gives a smaller penalized performance metric value (lines 10-17). After all the particles' individual best-so-far positions and their corresponding routing paths are updated, we find the particle j whose individual best-so-far position and its corresponding routing paths give the smallest penalized performance metric value. If the penalized performance metric value given by the particle j's individual best-so-far position and its corresponding routing paths is smaller than that given by the existing global best-so-far position and its corresponding routing paths, we update the global best-so-far position and its corresponding routing paths to the individual best-so-far position of the particle j and its corresponding routing paths (lines 18-21). This series of updating process is repeated until termination conditions are satisfied, where the termination conditions could be determined in the same manner as the rudimentary PSO algorithm. When the termination conditions are satisfied, the algorithm outputs the global best-so-far position $\mathcal{X}^*_{\mathcal{R}}$, which is the final solution obtained by our FANET topology construction algorithm.

Although the optimal locations of RUs can be obtained by our FANET topology construction algorithm, the optimality of their locations will be lost over time since MUs keep moving according to their given missions. Hence, whenever the optimality is broken, we need to relocate the RUs to the locations newly obtained by Algorithm 2 in response to the new locations of the MUs. However, the procedure of deriving the optimal locations of the RUs by Algorithm 2 is extremely time-consuming since the PSO algorithm generally requires a large number of iterations. Hence, in a FANET where nodes (e.g., MUs) keep moving fast, it is impractical to manage the topology with this algorithm alone to adapt to the continuous movements of nodes. For this reason, in the following subsection, we propose an algorithm that has much less computational overhead.

B. FANET TOPOLOGY ADJUSTMENT

In this subsection, we develop a FANET topology adjustment algorithm with little computational overhead, which maintains the network performance as good as possible by incrementally adjusting the locations of RUs adapting to the movements of MUs. Unlike Algorithm 2 in which the routing paths are jointly updated according to the locations of nodes, in this algorithm, we consider fixed routing paths since updating the routing paths is also a time consuming procedure. This algorithm is performed at each time step adaptively, where the length of the time-step interval is determined small enough to follow the movements of the MUs effectively. We denote the location of a node v at time step t by $\boldsymbol{x}_v(t)$, and similarly denote the location sets of MUs and RUs at time step t by $\mathcal{X}_{\mathcal{M}}(t)$ and $\mathcal{X}_{\mathcal{R}}(t)$, respectively.

Our FANET topology adjustment algorithm employs the gradient descent method [37]. The gradient of the performance metric function at time step t with respect to the location of each RU r can be expressed as

$$\nabla_{\boldsymbol{x}_r} f(\mathcal{X}_{\mathcal{G}}, \mathcal{X}_{\mathcal{M}}(t), \mathcal{X}_{\mathcal{R}}(t), \{\boldsymbol{\rho}_m\}),^{4} \tag{16}$$

which can be easily calculated since $\mathcal{X}_{\mathcal{G}}$, $\mathcal{X}_{\mathcal{M}}(t)$, $\mathcal{X}_{\mathcal{R}}(t)$, and $\{\boldsymbol{\rho}_m\}$ are known. In order for the performance metric value to be decreased, i.e., to enhance the network performance of FANET, each RU should be headed in the opposite direction of the gradient. In addition, when determining the travel distance of each RU, we should consider its speed limitation. Hence, the location where each RU r should be located at the next time step $t+1$ is defined as

$$\boldsymbol{x}_r(t+1) = \begin{cases} \boldsymbol{x}_r(t) - \gamma \cdot \nabla_{\boldsymbol{x}_r} f(t), & \text{if } \|\gamma \cdot \nabla_{\boldsymbol{x}_r} f(t)\|_2 \le \gamma_r, \\ \boldsymbol{x}_r(t) - \gamma_r \cdot \dfrac{\nabla_{\boldsymbol{x}_r} f(t)}{\|\nabla_{\boldsymbol{x}_r} f(t)\|_2}, & \text{otherwise,} \end{cases} \tag{17}$$

where γ is the positive step size that scales the travel distance, and γ_r is the maximum travel distance of RU r per time-step interval. Although the optimal locations of the RUs will be constantly varying depending on the movements of the MUs, this algorithm keeps adjusting the locations of the RUs in the direction of improving the network performance. The pseudo-code of the algorithm is described in Algorithm 3.

[4]For the sake of a simple notation, if there is no confusion, we will use $\nabla_{\boldsymbol{x}_r} f(\mathcal{X}_{\mathcal{G}}, \mathcal{X}_{\mathcal{M}}(t), \mathcal{X}_{\mathcal{R}}(t), \{\boldsymbol{\rho}_m\})$ and $\nabla_{\boldsymbol{x}_r} f(t)$ interchangeably.

Algorithm 3 FANET Topology Adjustment

```
Input: $\mathcal{X}_{\mathcal{G}}$, $\mathcal{X}_{\mathcal{M}}(t)$, $\mathcal{X}_{\mathcal{R}}(t)$, $\{\boldsymbol{\rho}_m\}_{m \in M}$
Output: $\mathcal{X}_{\mathcal{R}}(t+1)$
1 for each RU $r \in R$ do
2     Find $\boldsymbol{x}_r(t+1)$ according to (17).
3 end
```

In addition, unlike Algorithm 2 which is implemented only in a centralized manner and requires a great deal of execution time, not only is this FANET topology adjustment algorithm significantly low in computational complexity, but it can also be implemented in a distributed way if some conditions are satisfied. In other words, each RU can find its location for the next time step based only on the location information of its neighbor nodes which can directly communicate with it.

Remark 1: Algorithm 3 is implemented in a distributed fashion as long as the performance metric function is defined so that there is no term in which the location of each RU and those of its non-neighbor UAVs are coupled together, due to the fact that all terms involving the locations of the non-neighbor UAVs become zero when differentiated.

For example, when the performance metric function is defined as in (5), its gradient with respect to the location of each RU r at time step t is obtained as (18), where $\mathcal{N}_r(t)$ is a set of neighbor nodes of the RU r belonging to the active routing paths at time step t. Hence, the gradient can be calculated only with the locations of itself and its neighbor nodes, and thus Algorithm 3 can be implemented in a distributed manner.

Compared to the FANET topology construction algorithm, the FANET topology adjustment algorithm requires a very simple computation. Also, it is sufficient to take only a few movements of RUs when the degree of topological changes is small. Accordingly, it is little burden to run this FANET topology adjustment algorithm in each time step. However, if the topological changes are accumulated for long time steps, the optimality of the routing paths, which are not updated by the topology adjustment algorithm, might be lost. That is, there might be new routing paths that provide better network performance than the current routing paths. Also, the cumulative effect of the topological changes might create situations in which the constraints are no longer satisfied, and it might not be resolved by only incremental location updates with the FANET topology adjustment algorithm. Hence, to address the cumulative changes in the FANET topology, in the following subsection, we develop an algorithm that effectively manages the FANET topology by integrating the FANET topology construction and adjustment algorithm.

C. INTEGRATED FANET TOPOLOGY MANAGEMENT

In this subsection, we develop an integrated FANET topology management algorithm using both Algorithms 2 and 3. Our basic philosophy in this algorithm is to avoid frequently updating the routing paths and reconstructing the FANET topology for the sake of low computational overhead, while maintaining the network performance as good as possible. To this end, we update the routing paths or reconstruct the topology only if the topology has changed over certain levels, or if either the end-to-end communication constraint (2) or the safety constraint (3) is likely to be violated. We will measure how much the topology of FANET has changed and the constraints are tightened (i.e., how close the topology is to the bounds of the constraints) through a new proposed logical distance, named topology edit distance.

Prior to introducing the topology edit distance, we briefly explain the graph edit distance [43] that indicates how dissimilar two graphs are. In graph theory, a graph $\mathscr{G}_1$ can be transformed into the other graph $\mathscr{G}_2$ by using a multiple number of graph edit operations, such as insertion and deletion of a node and of an edge. An ordered set of graph edit operations transforming $\mathscr{G}_1$ into $\mathscr{G}_2$ is called a graph edit path from $\mathscr{G}_1$ to $\mathscr{G}_2$. Given the cost of each graph edit operation, the cost of a graph edit path can be defined by the sum of the costs of all graph edit operations in the path. In this case, the graph edit distance between $\mathscr{G}_1$ and $\mathscr{G}_2$ is defined as the smallest cost of the graph edit path from $\mathscr{G}_1$ to $\mathscr{G}_2$. As a specific example, if the cost of each graph edit operation is one, the graph edit distance between two graphs can be simplified as the minimum required number of the graph edit operations to transform the one graph into the other.

Now, using the principle of the graph edit distance, we will first define our topology edit operations, and then define the topology edit distance as the weighted sum of the topology edit operations. To define our topology edit operations, we first model a FANET topology at time step t as a graph $\mathcal{T}(t)$ with node set $\mathcal{N}$, edge set $\mathcal{E}(t)$, and location set $\mathcal{X}_{\mathcal{N}}(t)$. The node set $\mathcal{N}$ is defined as a set of all wireless nodes in FANET, the edge set $\mathcal{E}(t)$ as a set of unordered pairs of nodes whose lengths are not greater than d_{cm}, and the location set $\mathcal{X}_{\mathcal{N}}(t)$ as a set of the locations of all nodes in $\mathcal{N}$.

In our system, we need to measure how much the FANET topology has changed from time step t to time step τ, where $t < \tau$. In the FANET topology at time steps t and τ, their node sets are the same as $\mathcal{G} \cup \mathcal{M} \cup \mathcal{R}$,[5] but their edge sets may differ due to the changes in the locations of the nodes. Hence, we exclude the node insertions and deletions, and treat the minimum required number of the edge insertions and deletions to transform $\mathcal{T}(t)$ into $\mathcal{T}(\tau)$ as our first two topology edit operations, respectively, i.e.,

$$e_1(t,\tau) = |\mathcal{E}(\tau) \setminus \mathcal{E}(t)|, \quad e_2(t,\tau) = |\mathcal{E}(t) \setminus \mathcal{E}(\tau)|. \tag{19}$$

[5]In this paper, we do not consider the variation of the number of UAVs.

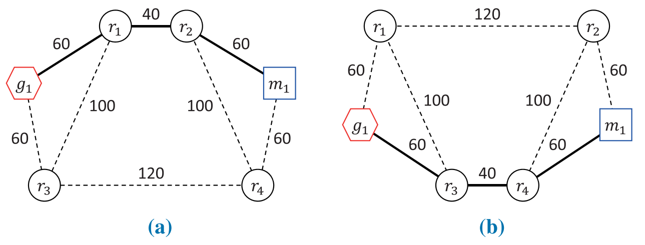


FIGURE 1. Example of the change in the FANET topology at different time steps t and τ. (The hexagon, square, and circles represent the GCS, the MU, and the RUs, respectively. The dotted and solid lines represent the available wireless links and the active routing path between the MU and the GCS, respectively. The numbers next to the edges represent the lengths of the edges in meters.). (a) At time step t, $\mathcal{T}(t)$. (b) At time step τ, $\mathcal{T}(\tau)$.

In addition, we need to consider the changes in edge lengths because each edge length is closely related to its wireless communication quality. Let us consider a simple example in which an MU m_1 sends data to a GCS g_1 in a FANET topology at two different time steps t and τ, as shown in Fig. 1. Assuming that the shortest path routing protocol is in use, where the cost of each edge is set to be proportional to its length, the optimal routing path at time steps t and τ are given as $\{m_1, r_2, r_1, g_1\}$ and $\{m_1, r_4, r_3, g_1\}$, respectively, as shown in Figs. 1a and 1b. Even though there is no change in the edge set, i.e., $\mathcal{E}(t) = \mathcal{E}(\tau)$, the optimal routing path can be different because of the changes in the lengths of the edges (r_1, r_2) and (r_3, r_4). Hence, we treat the total amount of changes in edge lengths as the third topology edit operation, i.e.,

$$e_3(t,\tau) = \sum_{(u,v)\in\mathcal{E}(t)\cap\mathcal{E}(\tau)} |\delta(u,v,t) - \delta(u,v,\tau)|, \tag{20}$$

where $\delta(u,v,t)$ stands for the distance between two nodes u and v at time step t, i.e., $\delta(u,v,t) = \|\boldsymbol{x}_u(t) - \boldsymbol{x}_v(t)\|_2$.

We lastly consider how much the end-to-end communication constraint and the safety constraint are likely to be violated in the changed FANET topology $\mathcal{T}(\tau)$. As either the end-to-end communication constraint or the safety constraint becomes more and more tightened, the routing paths update and the topology reconstruction are urged in order to avoid violating the constraints. Hence, we define two additional topology edit operations whose values become larger as the end-to-end communication and safety constraints become more tightened, respectively, i.e.,

$$e_4(t,\tau) = \exp\left[\psi_1\left(\max_{\substack{k=1,\ldots,|\boldsymbol{\rho}_m|-1,\\ m\in\mathcal{M}}} \delta(\rho_m^k, \rho_m^{k+1}, \tau) - d_{cm}\right)\right], \tag{21}$$

$$\nabla_{\boldsymbol{x}_r} f(\mathcal{X}_{\mathcal{G}}, \mathcal{X}_{\mathcal{M}}(t), \mathcal{X}_{\mathcal{R}}(t), \{\boldsymbol{\rho}_m\}) = \sum_{m\in M} \sum_{v\in\mathcal{N}_r(t)} \left[\alpha \cdot \|\boldsymbol{x}_r(t) - \boldsymbol{x}_v(t)\|_2^{\alpha-2} \cdot (\boldsymbol{x}_r(t) - \boldsymbol{x}_v(t))\right], \tag{18}$$

$$e_5(t,\tau) = \exp\left[\psi_2\left(d_{sf} - \min_{\substack{u,v\in\mathcal{M}\cup\mathcal{R},\\ u\neq v}} \delta(u,v,\tau)\right)\right], \tag{22}$$

where ψ_1 and ψ_2 are the sensitivity parameters. The topology edit operation e_4 is exponentially increased as the longest link distance in all active routing paths gets larger. Likewise, the topology edit operation e_5 is exponentially increased as the shortest inter-UAV distance gets smaller. Hence, we can judge how much the end-to-end communication and safety constraints have been tightened via these topology edit operations e_4 and e_5, respectively. In addition, we can detect that the topology is not satisfying the end-to-end communication and safety constraints if the topology edit operations e_4 and e_5 are greater than one, respectively.

Finally, the topology edit distance of a FANET topology between at different time steps t and τ is defined as

$$\delta_{ted}(\mathcal{T}(t), \mathcal{T}(\tau), \{\boldsymbol{\rho}_m\}) = \sum_{i=1}^{5} w_i \cdot e_i(t,\tau), \tag{23}$$

where w_i is the weight parameter for the topology edit operation e_i, and $\{\boldsymbol{\rho}_m\}$ is the set of the currently active routing paths. A large topology edit distance implies either that the FANET topology has changed much or that any constraint has become tightened. Hence, based on the topology edit distance, we can make a decision whether to update the routing paths and whether to reconstruct the FANET topology. To this end, we introduce two threshold levels ϵ_1 and ϵ_2, where $\epsilon_1 < \epsilon_2$. We regard that the FANET topology has changed enough to update the routing paths if the topology edit distance is greater than ϵ_1. Likewise, we regard that it has changed enough to be reconstructed if the topology edit distance is greater than ϵ_2. We note that the weights w_4 and w_5 and the threshold ϵ_2 should be carefully determined to prevent situations in which any constraint is violated. According to (21) and (22), the topology edit operations e_4 and e_5 become one when the longest link distance over all active routing paths and the shortest inter-UAV distance are equal to d_{cm} and d_{sf}, respectively. Hence, in order for the topology of FANET to be reconstructed before any constraint is violated, we have to determine w_4, w_5, and ϵ_2 such that $\epsilon_2 < \min\{w_4, w_5\}$. The larger the difference between ϵ_2 and $\min\{w_4, w_5\}$, the more likely it is that the longest link distance becomes small and the shortest inter-UAV distance becomes large. However, the computational overhead will also increase due to more frequent topology reconstruction.

We now explain our integrated FANET topology management algorithm in detail. Its pseudo-code is described in Algorithm 4. In the preparatory stage (lines 1-3), with given locations of GCSs and MUs, we first construct the initial FANET topology and determine the initial routing paths by Algorithm 2. We set the constructed topology to a reference topology $\mathcal{T}_{ref}$.

In the iterative stage (lines 4-14), we adjust the FANET topology by Algorithm 3 to adapt to the movements of the MUs, and then compare the newly adapted FANET topology with the reference topology $\mathcal{T}_{ref}$ in order to detect how much the FANET topology has changed from the reference topology and how well the constraints are satisfied. If the topology edit distance between them does not exceed the threshold ϵ_1, we decide that the cumulative changes in the FANET topology are not large, so continue to adjust the FANET topology by Algorithm 3 to adapt to the movements of the MUs (lines 4-5). On the other hand, if the topology edit distance is greater than ϵ_1, we decide that the FANET topology has changed enough to update the routing paths or reconstruct the topology. Since the topology reconstruction by Algorithm 2 requires a significant computational overhead, we try to resolve this situation by the routing paths update without the topology reconstruction first, and if it is no use, then we reconstruct the topology of FANET by Algorithm 2. Hence, once the topology edit distance is greater than ϵ_1, we primarily update the routing paths (lines 6-7). Then, despite updating the routing paths, if the topology edit distance is still greater than ϵ_2, we reconstruct the FANET topology (lines 8-11). Additionally, we update the reference topology $\mathcal{T}_{ref}$ to the current FANET topology (line 12).

Algorithm 4 Integrated FANET Topology Management

```
1  Construct an initial FANET topology by Alg. 2.
2  Initialize routing paths.
3  Set the reference topology, $\mathcal{T}_{ref}$.
4  for each time step $t$ do
5      Adjust the FANET topology by Alg. 3.
6      if $\delta_{ted}(\mathcal{T}_{ref}, \mathcal{T}(t), \{\boldsymbol{\rho}_m\}) > \epsilon_1$ then
7          Update the routing paths.
8          if $\delta_{ted}(\mathcal{T}_{ref}, \mathcal{T}(t), \{\boldsymbol{\rho}_m\}) > \epsilon_2$ then
9              Reconstruct the FANET topology by Alg. 2.
10             Update the routing paths.
11         end
12         Update the reference topology, $\mathcal{T}_{ref}$.
13     end
14 end
```

IV. PERFORMANCE EVALUATION

In this section, we present simulation results to validate the effectiveness of our proposed algorithms. We first show the performance of our PSO-based FANET topology construction algorithm that constructs a FANET topology from the scratch with consideration of the static environment in which the locations of MUs are fixed. After then, considering the dynamic environment in which the MUs keep moving based on their given missions, we show the performance of our integrated FANET topology management algorithm that involves the FANET topology construction and adjustment algorithms.

In the following simulations, we consider a FANET that consists of one GCS and four MUs, and we take the function in (5) with $\alpha = 2$ as the network performance metric function, which is the sum of the squared link distances over all active routing paths. As the routing protocol in use, we consider

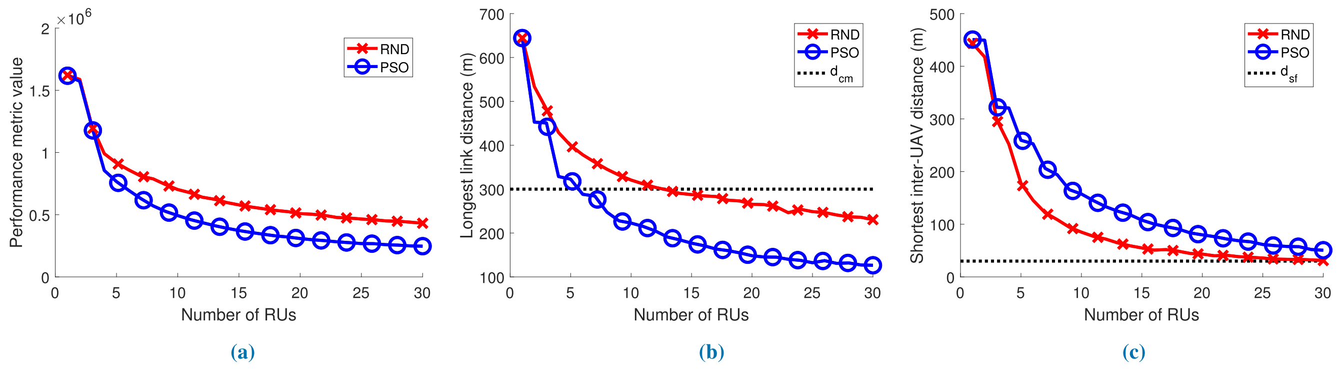


FIGURE 2. Comparison of our PSO-based FANET topology construction algorithm (PSO) with the random-based FANET topology construction algorithm (RND) varying the number of RUs. (a) Performance metric value. (b) Longest link distance. (c) Shortest inter-UAV distance.

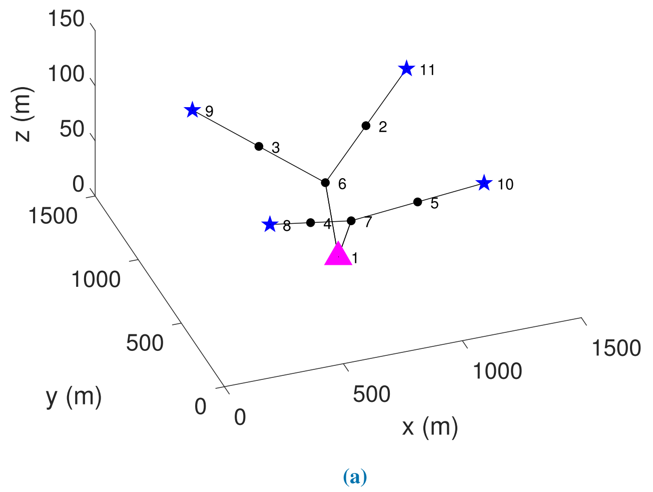
the shortest path routing algorithm in which the link cost is defined by the squared link distance. We consider a rectangular parallelepiped region [0, 1500] × [0, 1500] × [50, 150] as the 3-dimensional deployable space $\mathcal{S}$ for UAVs. The threshold distances for the wireless communication d_{cm} and the safety maintenance d_{sf} are set to be 300 meters and 30 meters, respectively.

A. PERFORMANCE OF THE FANET TOPOLOGY CONSTRUCTION

In this subsection, we evaluate Algorithm 2 that constructs a topology of FANET. We consider a simulation scenario in which the GCS is located at the fixed point (750, 750, 0), and the four MUs are deployed at the points (300, 300, 100), (300, 1200, 100), (1200, 300, 100), and (1200, 1200, 100), respectively. The parameters used in Algorithm 2 are set as follows. The penalty coefficients λ_1, λ_2, λ_3, λ_4, and μ in (12) are set to all 50. The number of particles N_p is set to 30, and the parameters w, c_1, and c_2 in (13) are set to 0.729, 1.4962, and 1.4962, respectively. The threshold for the velocity clamping V_j^{max} is set to 10 percent of the range of the search space, i.e., 150, 150, and 10 for the x-, y-, and z-axes, respectively. The termination criterion of our FANET topology construction algorithm is set to 300 iterations.

We compare the performance of our PSO-based FANET topology construction algorithm with that of the random-based one varying the number of RUs from 1 to 30. The random-based FANET topology construction algorithm creates a FANET topology by randomly deploying the given number of RUs in the deployable space $\mathcal{S}$, and evaluate the performance of the constructed FANET topology for 300 iterations, which is set to be equal to the termination criterion of our PSO-based algorithm for the sake of fair comparison, and then the best FANET topology is selected. The comparison results are shown in Fig. 2, which are obtained by a Monte-Carlo simulation with 30 independent runs. In Fig. 2a, we compare the achieved performance metric values of the algorithms, which are measures of the performance of the constructed FANET topology. This figure shows that our PSO-based algorithm always yields a lower value than the random-based one, which implies that our PSO-based algorithm makes a superior FANET topology in all cases. In Fig. 2b, we compare the longest link distances over all the active routing paths. We can see that the longest link distance in the random-based algorithm becomes lower than d_{cm} when at least 13 RUs are deployed to construct a FANET topology. On the other hand, our PSO-based algorithm can construct a FANET topology that gives the longest link distance lower than d_{cm} if more than or equal to 6 RUs are deployed. Hence, our proposed algorithm can support the end-to-end communication between each MU and the GCS with a much lower number of RUs. In addition, the longest link distances in our PSO-based algorithm are smaller than those in the random-based one in all cases, which implies that our PSO-based algorithm supports more reliable end-to-end communications than the random-based one. In Fig. 2c, we compare the achieved shortest inter-UAV distances. We can see that our PSO-based algorithm keeps a larger safety distance than the random-based one in all cases. This implies that our PSO-based algorithm constructs a more secure FANET topology from the collisions between UAVs than the random-based one. As a result, the above simulation results demonstrate that our PSO-based FANET topology construction algorithm is superior to the random-based one in all aspects such as the network performance, the end-to-end communication, and the safety from the collision between UAVs.

In the above simulation scenario, the distance between each MU and the GCS is about 644 meters. Thus, in order to support the end-to-end communications between each MU and the GCS, at least $\lceil 644/d_{cm} \rceil - 1 = 2$ RUs are required for each routing path. Let us consider the case in which 5 RUs are available. To minimize the longest link distance in all the routing paths, one RU should be located at (750, 750, 100), and the others at the middle of each MU and that RU. However, the longest link distance, in this case, is 318 meters, which is still greater than d_{cm}, and thus, the end-to-end communications cannot be supported. On the other hand, with 6 RUs, we can create a FANET topology as shown in Fig. 3 in which the longest link distance is smaller than d_{cm}. That



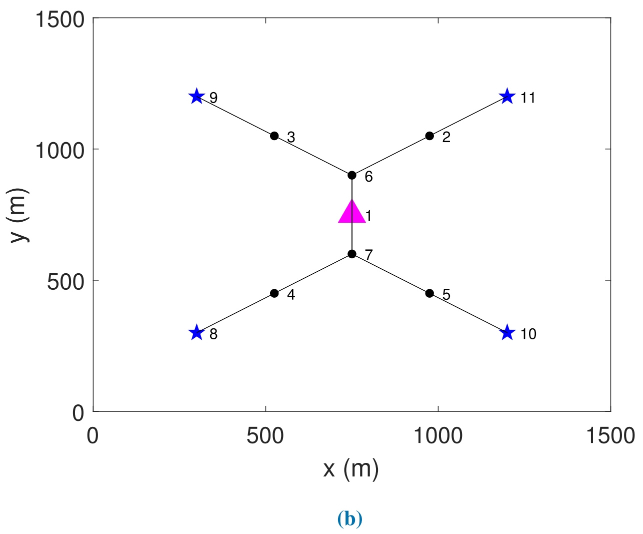


FIGURE 3. Example of the FANET topology created by Algorithm 2 when 6 RUs are available. (The triangle, pentagrams, and the dots represent the GCS, the MUs, and the RUs, respectively. The solid lines represent the wireless links belonging to the active routing paths.). (a) 3-dimensional view. (b) 2-dimensional view.

is, the end-to-end communications are supported. Hence, in the above simulation scenario, it is impossible to make a FANET topology that offers the end-to-end communication between all MUs and the GCS with less than 6 RUs, and thus, in the above results, we can see that our PSO-based algorithm constructs a FANET topology with a minimum number of RUs that supports the end-to-end communications.

B. PERFORMANCE OF THE FANET TOPOLOGY MANAGEMENT

In this subsection, we evaluate the performance of Algorithm 4 that manages a FANET topology considering a dynamic environment in which MUs keep moving according to their missions. We consider a scenario in which the GCS is located at the fixed point (750, 750, 0), four MUs keep moving for 10, 000 seconds according to a mobility model with a speed of 5 meters per time-step interval, starting from the points (300, 300, 100), (300, 1200, 100), (1200, 300, 100), and (1200, 1200, 100), respectively, and 10 RUs are used to manage the FANET topology.

We use the Lévy flight model [44] as the mobility model of the MUs, since the Lévy flight model is known to well describe the movements of creatures performing missions such as foraging [45]. The Lévy flight model is a probabilistic version of random walk models in which each travel distance is determined according to a heavy-tailed probability distribution. The trajectory of each MU is generated as follows [46]:

1) We calculate the number s obeying the Lévy flight distribution by

$$s = \frac{\eta_1}{|\eta_2|^{1/\beta}}, \tag{24}$$

where η_1 and η_2 are the random numbers obeying normal distributions with zero mean and variances σ_1^2 and σ_2^2, respectively. The standard deviations, i.e., the square roots of the variances, are given as

$$\sigma_1 = \left(\frac{\Gamma(1+\beta)\cdot\sin(\pi\beta/2)}{\Gamma((1+\beta)/2)\cdot\beta\cdot 2^{(\beta-1)/2}}\right)^{\frac{1}{\beta}}, \quad \sigma_2 = 1, \tag{25}$$

where the scale parameter β is set to 1.5 and Γ denotes the gamma function.

2) Repeating step 1), we get three numbers s_1, s_2, and s_3.
3) We move the MU by $(300\times s_1, 300\times s_2, 20\times s_3)$ from its current location, where the constant weights are equal to 20 percent of the range of the search space $\mathcal{S}$.
4) We repeat the previous steps.

An instance of trajectories for 10, 000 time steps of the four MUs, obtained by the above process, is shown in Fig. 4.

The parameters used in Algorithm 2 are set to be equal to those in Subsection IV-A, and the parameters used in Algorithm 3 are set as follows. The step size γ is set to 0.05, and the maximum travel distances per time-step interval of RUs are set to all 20 meters, i.e., $\gamma_r = 20$ for all $r \in \mathcal{R}$. Next, the parameters used in Algorithm 4 are set as: $w_1 = 30$, $w_2 = 30$, $w_3 = 0.5$, $w_4 = 1000$, $w_5 = 1000$, $\epsilon_1 = 700$, and $\epsilon_2 = 1000$.

We compare our integrated FANET topology management algorithm (IFTM) with three other algorithms. The first is the algorithm in which the initial topology is constructed by Algorithm 2, and then in each time step, the topology is adjusted by Algorithm 3 without updating routing paths and reconstructing the topology. Hence, this algorithm is IFTM without routing paths update and topology reconstruction, and we call this algorithm IFTM-RT. The second is the algorithm that is the same as IFTM except that only the routing paths update is performed, if the condition in Algorithm 4 is satisfied, without performing topology

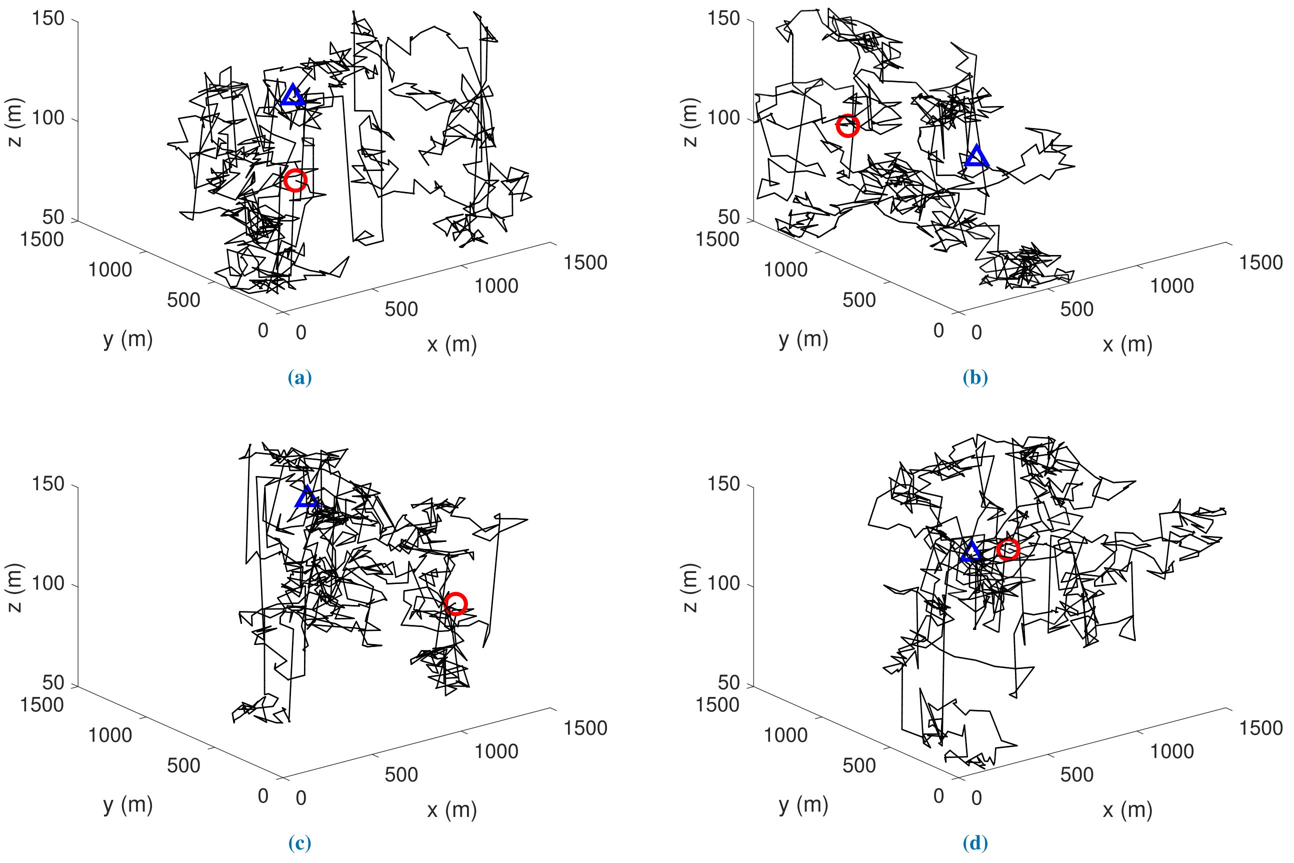


FIGURE 4. **Instance of trajectories for 10, 000 time steps of the four MUs, generated according to the Lévy flight model. (The circle and triangle represent the starting and terminal points, respectively. The solid lines represent the trajectory.). (a) MU m_1. (b) MU m_2. (c) MU m_3. (d) MU m_4.**

reconstruction. We call this algorithm IFTM-T. The comparison of our IFTM with IFTM-RT and IFTM-T can show how much the routing paths update and the FANET topology reconstruction have impact on the performance of FANET. The last one is the algorithm in which our PSO-based topology construction algorithm, Algorithm 2, is used for both topology construction and adjustment. Hence, in this algorithm, based on the new locations of MUs, our PSO-based topology construction algorithm is performed in each time step. We call it PSO-Only. PSO-Only can be used as a benchmark to evaluate the (sub-)optimality of our IFTM.

We first compare the performance of our IFTM with those of IFTM-RT, IFTM-T, and PSO-Only when considering an instance of the trajectories of the four MUs, given as shown in Fig. 4. Although the one instance of the trajectories is considered, due to the stochastic characteristics of the PSO algorithm, the simulation results are obtained by a Monte-Carlo simulation with 30 independent runs.

Fig. 5 shows the simulation results of achieved performance metric values, longest link distances over the active routing paths, and shortest inter-UAV distances of the algorithms. In addition to their exact trajectories of results over time, i.e., Figs. 5a, 5c, and 5e, we also provide their cumulative moving average results, i.e., Figs. 5b, 5d, and 5f, which are obtained by averaging all of the simulation results up until the current datum point. Thanks to the characteristics of the cumulative moving average method that smooth out the fluctuations and outline a long-term trend, it is useful to observe time series data with short-term fluctuations. In Fig. 5a, IFTM and PSO-Only look better than the others, and IFTM-RT looks the worst, which is shown more clearly in Fig. 5b. IFTM-T gives better performance than IFTM-RT because the former can update the current routing paths to the optimal ones, which become different as the FANET topology changes. However, IFTM-T is worse than IFTM and PSO-Only because it primarily responds to the topological changes only by updating the routing paths, but it cannot handle situations in which the cumulative topological changes become large. In Figs. 5c and 5e, we can see that IFTM and PSO-Only give the longest link distance smaller than d_{cm} and the shortest inter-UAV distance greater than d_{sf} over all periods of time. In fact, IFTM-RT and IFTM-T encounters situations where the end-to-end communication constraint (2) and the safety constraint (3) are not satisfied, whereas IFTM and PSO-Only satisfy the constraints all the time. From a long-term perspective in Figs. 5d and 5f, we can clearly

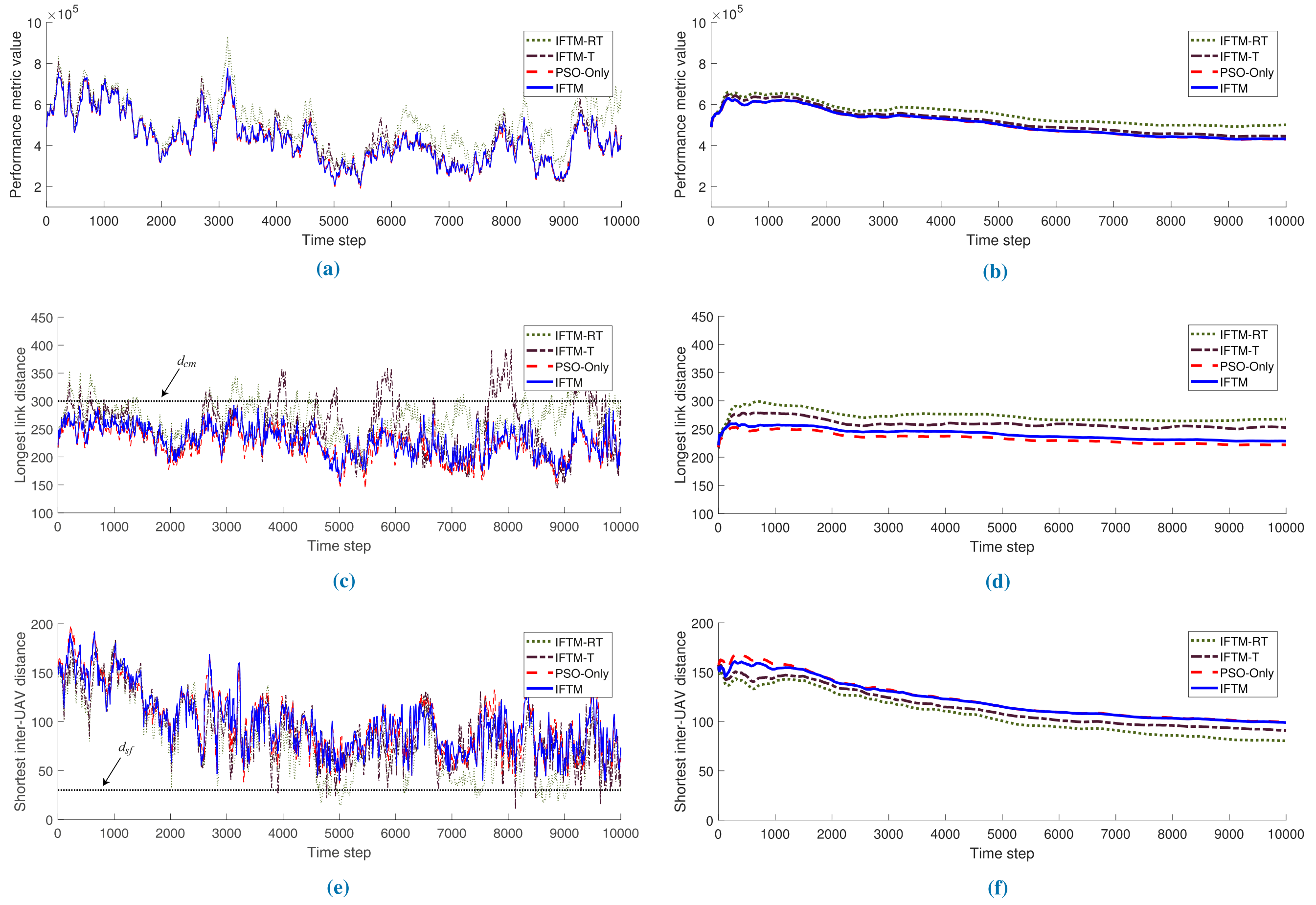


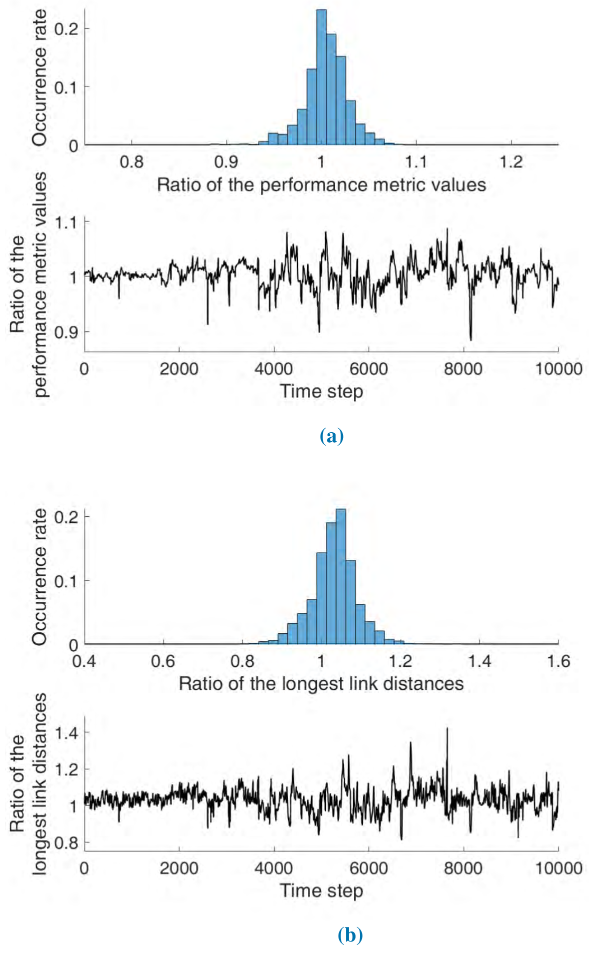
FIGURE 5. Comparison of our IFTM with IFTM-RT, IFTM-T, and PSO-Only when considering the instance given as in Fig. 4. (a) Performance metric value. (b) Cumulative moving average of Fig. 5a. (c) Longest link distance over active routing paths. (d) Cumulative moving average of Fig. 5c. (e) Shortest inter-UAV distance. (f) Cumulative moving average of Fig. 5e.

recognize that IFTM and PSO-Only outperform IFTM-RT and IFTM-T in terms of both the longest link distance and the shortest inter-UAV distance.

Now, to compare our IFTM and PSO-Only in more detail, we provide the ratios of achieved performance metric, longest link distance, and shortest inter-UAV distance of IFTM in each time step to those of PSO-Only, and also the histograms of their occurrence rates over 10, 000 time steps in Fig. 6. The ratio of one implies that IFTM and PSO-Only provide the same performance for each case. For the ratio of the achieved performance metric values in Fig. 6a, its average is 1.0044 and its standard deviation is 0.0236. For the ratio of the longest link distances in Fig. 6b, its average is 1.0325 and its standard deviation is 0.0595. For the ratio of the shortest inter-UAV distances in Fig. 6c, its average is 1.0072 and its standard deviation is 0.1178. In addition, the figure clearly shows that in most of time steps, the ratios of all three cases are very close to one, which implies that our IFTM provides almost the same performance to PSO-Only not only on average, but also in most of time steps.

In Fig. 5, even though our IFTM and PSO-Only provide similar performances, PSO-Only provides slightly better performances than IFTM. However, since PSO-Only is required to perform the PSO algorithm, which has a large computational overhead, in each time step, its overall computational overhead will be much large. To clearly show it, we compare the execution times of IFTM and PSO-Only in Fig. 7. As shown in this figure, the execution time of PSO-Only is much larger than our IFTM. The average execution times of IFTM and PSO-Only are 118 seconds and 2, 370 seconds, respectively, and thus, the execution time of PSO-Only is about 20 times larger than that of IFTM. Hence, even though PSO-Only gives slightly better performances than our IFTM, it requires too much execution time, which is a serious flaw in FANET whose topology changes rapidly. On the other hand, our IFTM provides quite comparable performances to PSO-Only while requiring much smaller execution time.

We now compare the performances of our IFTM with those of PSO-Only with 30 different instances of the trajectories for 10, 000 time steps, where the instances are generated independently with the Lévy flight model. Hence, we have total 300, 000 samples. For each of performance metric value, longest link distance, and shortest inter-UAV



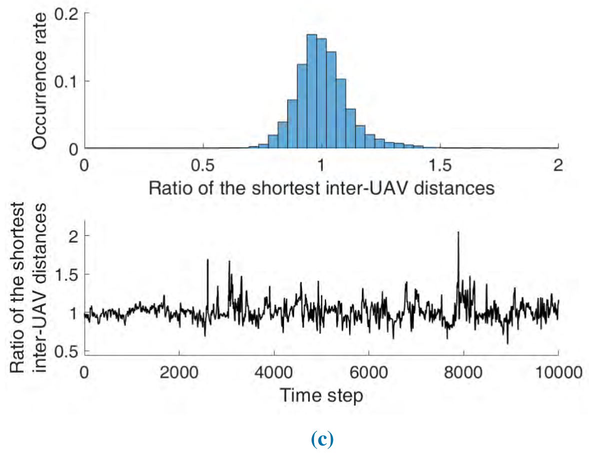


FIGURE 6. Performances of our IFTM normalized by those of PSO-Only when considering the instance given as in Fig. 4. (a) Performance metric value. (b) Longest link distance. (c) Shortest inter-UAV distance.

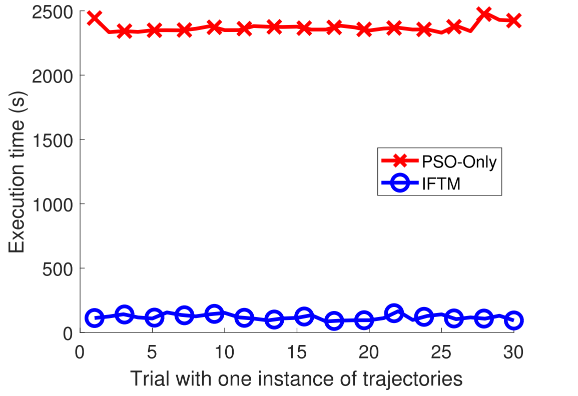


FIGURE 7. Comparison of the execution times of IFTM and PSO-Only when considering the instance given as in Fig. 4. (The execution time was measured using MATLAB software on a computer with Intel Core i7-7700 CPU (3.60 GHz) and 16.0 GB RAM.).

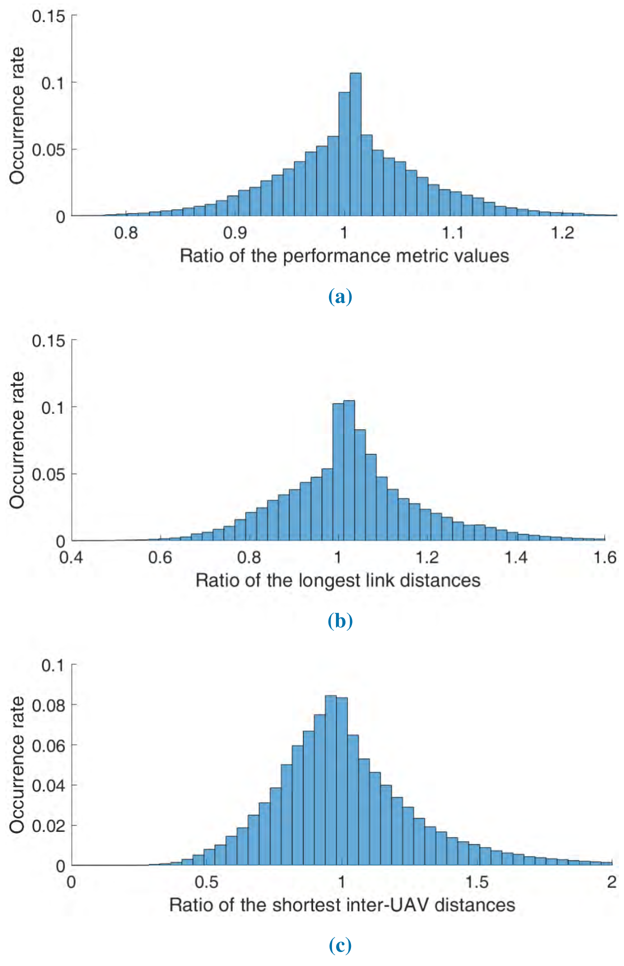


FIGURE 8. Performances of our IFTM normalized by those of PSO-Only when considering 30 different instances of the trajectories of the four MUs. (a) Performance metric value. (b) Longest link distance. (c) Shortest inter-UAV distance.

distance, we provide the occurrence rate of the ratio of IFTM to PSO-Only in Fig 8. In each graph, most of the achieved ratios are concentrated around one, which implies that our IFTM provide similar performances to PSO-Only. For the ratio of the achieved performance metric values in Fig. 8a, its average is 1.0065 and its standard deviation is 0.0727. For the ratio of the longest link distances in Fig. 8b, its average is 1.0352 and its standard deviation is 0.1657. For the ratio

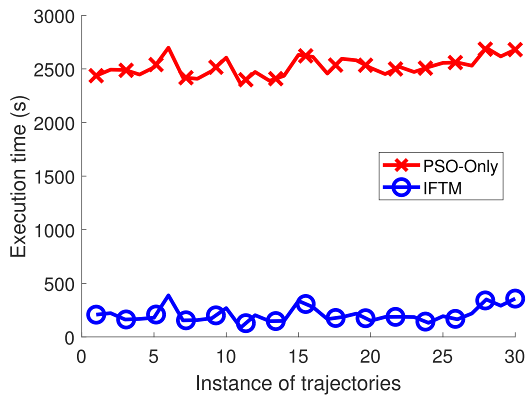


FIGURE 9. **Comparison of the execution times of IFTM and PSO-Only when considering 30 different instances of the trajectories of the four MUs. (The execution time was measured using MATLAB software on a computer with Intel Core i7-7700 CPU (3.60 GHz) and 16.0 GB RAM.).**

of the shortest inter-UAV distances in Fig. 8c, its average is 1.0252 and its standard deviation is 0.2965. As a result, the figure clearly shows that the ratios of all three cases are very close to one in most of time steps. In other words, our IFTM provides almost the same performances to PSO-Only not only on average but also in most of time steps regardless of the movement trajectories of MUs.

We also compare the execution times of our IFTM and PSO-Only in Fig. 9. As before, this figure shows that the execution time of IFTM is much smaller than that of PSO-Only, which implies that the computational overhead of IFTM is much lower than that of PSO-Only. Hence, even though PSO-Only is slightly better than our IFTM, its large execution time, i.e., the excessive computational overhead, should be a serious drawback especially in FANET whose topology changes rapidly.

V. CONCLUSION

In this paper, we have studied a topology management problem in FANET, which aims at maximizing the network performance via deploying and moving RUs adapting to the unpredictable mission-based movements of MUs. When formulating the FANET topology management problem, we have considered the dependency between the performance of the network topology and the routing protocol, the end-to-end communication constraint to support reliable ad hoc communications, and the safety constraint to maintain the safety distance between UAVs. We first developed the PSO-based FANET topology construction algorithm, which deploys RUs based on the locations of MUs and the routing protocol in use. We then developed a simple FANET topology adjustment algorithm with low computational overhead, which adjusts the locations of RUs adapting to the movements of MUs. Finally, proposing a new logical metric, topology edit distance, which measures the topological changes and how well the constraints are satisfied, we developed an IFTM algorithm that integrates the FANET topology construction and adjustment algorithms with additional consideration of routing paths update and topology reconstruction processes. From the simulation results, we have first clearly shown that our PSO-based FANET topology construction algorithm outperforms the random-based one in all aspects such as the network performance, the longest link distance, and the shortest inter-UAV distance. In addition, we have shown the effectiveness of consideration of the routing paths update and the topology reconstruction in our IFTM to the FANET topology management. We also have shown that our IFTM provides comparable performance to PSO-Only but needs much smaller execution time than PSO-Only, which is a very important point in FANET with rapid and frequent topological fluctuations.

REFERENCES

[1] D. Glade, "Unmanned aerial vehicles: Implications for military operations," Air Univ. Press Maxwell Air Force Base, Montgomery, AL, USA, Tech. Rep. ADA425476, Jul. 2000.

[2] T. Coffey and J. A. Montgomery, "The emergence of mini UAVs for military applications," *Defense Horizons*, no. 22, pp. 1–8, Dec. 2002.

[3] M. A. Ma'sum *et al.*, "Simulation of intelligent unmanned aerial vehicle (UAV) for military surveillance," in *Proc. ICACSIS*, Bali, Indonesia, Sep. 2013, pp. 161–166.

[4] S. Siebert and J. Teizer, "Mobile 3D mapping for surveying earthwork projects using an unmanned aerial vehicle (UAV) system," *Autom. Construct.*, vol. 41, pp. 1–14, May 2014.

[5] A. Rango *et al.*, "Unmanned aerial vehicle-based remote sensing for rangeland assessment, monitoring, and management," *J. Appl. Remote Sens.*, vol. 3, no. 1, pp. 033542-1–033542-15, Aug. 2009.

[6] Z. Xu *et al.*, "Development of an UAS for post-earthquake disaster surveying and its application in Ms7.0 Lushan earthquake, Sichuan, China," *Comput. Geosci.*, vol. 68, pp. 22–30, Jul. 2014.

[7] V. Sharma and R. Kumar, "A cooperative network framework for multi-UAV guided ground ad hoc networks," *J. Intell. Robotic Syst.*, vol. 77, nos. 3–4, pp. 629–652, 2015.

[8] Y. Zhou, N. Cheng, N. Lu, and X. S. Shen, "Multi-UAV-aided networks: Aerial-ground cooperative vehicular networking architecture," *IEEE Veh. Technol. Mag.*, vol. 10, no. 4, pp. 36–44, Dec. 2015.

[9] Y. Zeng, R. Zhang, and T. J. Lim, "Wireless communications with unmanned aerial vehicles: Opportunities and challenges," *IEEE Commun. Mag.*, vol. 54, no. 5, pp. 36–42, May 2016.

[10] J. Wang, C. Jiang, Z. Han, Y. Ren, R. G. Maunder, and L. Hanzo, "Taking drones to the next level: Cooperative distributed unmanned-aerial-vehicular networks for small and mini drones," *IEEE Veh. Technol. Mag.*, vol. 12, no. 3, pp. 73–82, Sep. 2017.

[11] J. Li, Y. Zhou, and L. Lamont, "Communication architectures and protocols for networking unmanned aerial vehicles," in *Proc. IEEE Globecom Workshops*, Dec. 2013, pp. 1415–1420.

[12] İ. Bekmezci, O. K. Sahingoz, and Ş. Temel, "Flying ad-hoc networks (FANETs): A survey," *Ad Hoc Netw.*, vol. 11, no. 3, pp. 1254–1270, 2013.

[13] L. Gupta, R. Jain, and G. Vaszkun, "Survey of important issues in UAV communication networks," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 2, pp. 1123–1152, 2nd Quart., 2016

[14] O. S. Oubbati, A. Lakas, F. Zhou, M. Güneş, and M. B. Yagoubi, "A survey on position-based routing protocols for flying ad hoc networks (FANETs)," *Veh. Commun.*, vol. 10, pp. 29–56, Oct. 2017.

[15] I. F. Senturk, K. Akkaya, and S. Yilmaz, "Relay placement for restoring connectivity in partitioned wireless sensor networks under limited information," *Ad Hoc Netw.*, vol. 13, pp. 487–503, Feb. 2014.

[16] M. Mozaffari, W. Saad, M. Bennis, and M. Debbah, "Efficient deployment of multiple unmanned aerial vehicles for optimal wireless coverage," *IEEE Commun. Lett.*, vol. 20, no. 8, pp. 1647–1650, Aug. 2016.

[17] M. Younis, I. F. Senturk, K. Akkaya, S. Lee, and F. Senel, "Topology management techniques for tolerating node failures in wireless sensor networks: A survey," *Comput. Netw.*, vol. 58, pp. 254–283, Jan. 2014.

[18] S. Hayat, E. Yanmaz, and R. Muzaffar, "Survey on unmanned aerial vehicle networks for civil applications: A communications viewpoint," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 4, pp. 2624–2661, 4th Quart., 2016.

[19] A. Bujari, C. E. Palazzi, and D. Ronzani, "FANET application scenarios and mobility models," in *Proc. 3rd Workshop Micro Aerial Veh. Netw. Syst. Appl.*, Niagara Falls, NY, USA, Jun. 2017, pp. 43–46.

[20] J. Hoydis, M. Petrova, and P. Mähönen, "Effects of topology on local throughput-capacity of ad hoc networks," in *Proc. IEEE PIMRC*, Cannes, France, Sep. 2008, pp. 1–5.

[21] G. Ausiello, P. Crescenzi, G. Gambosi, V. Kann, A. Marchetti-Spaccamela, and M. Protasi, *Complexity and Approximation: Combinatorial Optimization Problems and Their Approximability Properties*. Berlin, Germany: Springer, 2012.

[22] D.-Y. Kim and J.-W. Lee, "Topology construction for flying ad hoc networks (FANETs)," in *Proc. ICTC*, Jeju, South Korea, Oct. 2017, pp. 153–157.

[23] E. L. Lloyd and G. Xue, "Relay node placement in wireless sensor networks," *IEEE Trans. Comput.*, vol. 56, no. 1, pp. 134–138, Jan. 2007.

[24] H.-T. Roh and J.-W. Lee, "Joint relay node placement and node scheduling in wireless networks with a relay node with controllable mobility," *Wireless Commun. Mobile Comput.*, vol. 12, no. 8, pp. 699–712, Jun. 2012.

[25] S. Lee and M. Younis, "Optimized relay placement to federate segments in wireless sensor networks," *IEEE J. Sel. Areas Commun.*, vol. 28, no. 5, pp. 742–752, Jun. 2010.

[26] M. Rebai, M. Le Berre, H. Snoussi, F. Hnaien, and L. Khoukhi, "Sensor deployment optimization methods to achieve both coverage and connectivity in wireless sensor networks," *Comput. Oper. Res.*, vol. 59, pp. 11–21, Jul. 2015.

[27] M. Minelli *et al.*, "Optimal relay placement in cellular networks," *IEEE Trans. Wireless Commun.*, vol. 13, no. 2, pp. 998–1009, Feb. 2014.

[28] R. Magán-Carrión, R. A. Rodríguez-Gómez, J. Camacho, and P. García-Teodoro, "Optimal relay placement in multi-hop wireless networks," *Ad Hoc Netw.*, vol. 46, pp. 23–36, Aug. 2016.

[29] H.-T. Roh and J.-W. Lee, "Communication-aware position control for mobile nodes in vehicular networks," *IEEE J. Sel. Areas Commun.*, vol. 29, no. 1, pp. 173–186, Jan. 2011.

[30] H.-T. Roh and J.-W. Lee, "Distributed node placement algorithm utilizing controllable mobility in mobile ad hoc networks," *Ad Hoc Netw.*, vol. 15, pp. 67–77, Apr. 2014.

[31] Z. Han, A. L. Swindlehurst, and K. J. R. Liu, "Optimization of MANET connectivity via smart deployment/movement of unmanned air vehicles," *IEEE Trans. Veh. Technol.*, vol. 58, no. 7, pp. 3533–3546, Sep. 2009.

[32] S. Kim, H. Oh, J. Suk, and A. Tsourdos, "Coordinated trajectory planning for efficient communication relay using multiple UAVs," *Control Eng. Pract.*, vol. 29, pp. 42–49, Aug. 2014.

[33] P. Ladosz, H. Oh, and W.-H. Chen, "Optimal positioning of communication relay unmanned aerial vehicles in urban environments," in *Proc. ICUAS*, Arlington, VA, USA, Jun. 2016, pp. 1140–1147.

[34] O. Dengiz, A. Konak, and A. E. Smith, "Connectivity management in mobile ad hoc networks using particle swarm optimization," *Ad Hoc Netw.*, vol. 9, no. 7, pp. 1312–1326, Sep. 2011.

[35] R. Magán-Carrión, J. Camacho, P. García-Teodoro, E. F. Flushing, and G. A. Di Caro, "A dynamical relay node placement solution for MANETs," *Comput. Commun.*, vol. 114, pp. 36–50, Dec. 2017.

[36] M. Lange, D. Zühlke, O. Holz, and T. Villmann, "Applications of l_p-norms and their smooth approximations for gradient based learning vector quantization," in *Proc. ESANN*, Bruges, Belgium, Apr. 2014, pp. 271–276.

[37] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

[38] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. ICNN*, Perth, WA, Australia, Nov./Dec. 1995, pp. 1942–1948.

[39] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983.

[40] E. K. P. Chong and S. H. Żak, *An Introduction to Optimization*, 4th ed. Hoboken, NJ, USA: Wiley, 2013.

[41] F. van den Bergh and A. P. Engelbrecht, "A study of particle swarm optimization particle trajectories," *Inf. Sci.*, vol. 176, no. 8, pp. 937–971, 2006.

[42] I.-L. Lin, "Particle swarm optimization for solving constraint satisfaction problems," M.S. thesis, Simon Fraser Univ., Burnaby, BC, Canada, 2005.

[43] A. Sanfeliu and K.-S. Fu, "A distance measure between attributed relational graphs for pattern recognition," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-13, no. 3, pp. 353–362, May/Jun. 1983.

[44] R. Metzler, A. V. Chechkin, V. Y. Gonchar, and J. Klafter, "Some fundamental aspects of Lévy flights," *Chaos, Solitons Fractals*, vol. 34, no. 1, pp. 129–142, Oct. 2007.

[45] G. M. Viswanathan, V. Afanasyev, S. V. Buldyrev, E. J. Murphy, P. A. Prince, and H. E. Stanley, "Lévy flight search patterns of wandering albatrosses," *Nature*, vol. 381, pp. 413–415, May 1996.

[46] R. N. Mantegna, "Fast, accurate algorithm for numerical simulation of Lévy stable stochastic processes," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 49, no. 5, pp. 4677–4683, May 1994.

DO-YUP KIM received the B.S. degree *(summa cum laude)* in electronics and communications engineering from Kwangwoon University, Seoul, South Korea, in 2016. Since 2016, he has been pursuing the Ph.D. degree in electrical and electronic engineering with Yonsei University, Seoul. His research interests include communication networks, mobile ad hoc networks, optimization, and simultaneous wireless information and power transfer.

JANG-WON LEE (M'04–SM'12) received the B.S. degree in electronic engineering from Yonsei University, Seoul, South Korea, in 1994, the M.S. degree in electrical engineering from the Korea Advanced Institute of Science and Technology, Daejeon, South Korea, in 1996, and the Ph.D. degree in electrical and computer engineering from Purdue University, West Lafayette, IN, USA, in 2004. From 1997 to 1998, he was with the Dacom Research and Development Center, Daejeon. From 2004 to 2005, he was a Post-Doctoral Research Associate with the Department of Electrical Engineering, Princeton University, Princeton, NJ, USA. Since 2005, he has been with the School of Electrical and Electronic Engineering, Yonsei University, where he is currently a Professor. His research interests include resource allocation, QoS and pricing issues, optimization, performance analysis in communication networks, and smart grid.