Distributed Topology Control based on Swarm Intelligence In Unmanned Aerial Vehicles Networks

Qianyi Zhang, Gang Feng, *Senior Member, IEEE*, Shuang Qin, *Member, IEEE*, Yao Sun
National Key Laboratory of Science and Technology on Communications,
University of Electronic Science and Technology of China
E-mail:fenggang@uestc.edu.cn

***Abstract*—Unmanned aerial vehicles (UAVs) have shown enormous potential in both public and civil domains. Although multi-UAV systems can collaboratively accomplish missions efficiently, UAV network(UAVNET) design faces many challenging issues, such as high mobility, dynamic topology, power constraints, and varying quality of communication links. Topology control plays a key role for providing high network connectivity while conserving power in UAVNETs. In this paper, we propose a distributed topology control algorithm based on discrete particle swarm optimization with articulation points(AP-DPSO). To reduce signaling overhead and facilitate distributed control, we first identify a set of articulation points (APs) to partition the network into multiple segments. The local topology control problem for individual segments is formulated as a degree-constrained minimum spanning tree problem. Each node collects local topology information and adjusts its transmit power to minimize power consumption. We conduct simulation experiments to evaluate the performance of the proposed AP-DPSO algorithm. Numerical results show that AP-DPSO outperforms some known algorithms including LMST and LSP, in terms of network connectivity, average link length and network robustness for a dynamic UAVNET.**

I. Introduction

Unmanned aerial vehicle network (UAVNET) is a self-organizing, decentralized, peer-to-peer multi-hop communication wireless network without the support of a preexisting infrastructure, such as routers in wired networks or access points in managed wireless networks. The UAV nodes in the network have routing and packet forwarding functions and they can join or exit the network flexibly, and thus setting up an ad hoc network between high-speed mobile UAVs is challenging. Specifically, UAVNETs are different from the traditional mobile ad hoc networks (MANETs) and vehicle ad hoc networks (VANETs) in terms of node mobility, network connectivity, message routing, service quality and application fields. Moreover, due to limited energy, UAV nodes are prone to failures, which will result in network damage [1]. Thus it is very challenging to ensure network connectivity and network throughput while saving energy, reducing interference, and prolonging the network lifetime for UAVNETs. It is imperative to develop efficient topology control algorithms to improve UAVNET performance.

In UAVNETs, nodes can adjust the transmit power dynamically according to the network condition, thus to ensure the data transmission and network robustness in case of partial node failure and movement. Obviously, topology control in UAVENTS is more difficult than that in conventional MANETs due to the high dynamics. In addition, due to the lack of central controller, the complete topology information is unavailable for any node, which makes topology control more challenging. In general, the studies on topology control mainly includes two directions: power control and sleep scheduling. Power control aims to minimize transmit power for each node. Sleep scheduling reduces power consumption by switching off some nodes in the network. Since sleep scheduling is effective only for networks with high density or event-driven networks, in this study, we address the problem of topology control for UAVNETs by controlling the transmit power of nodes.

In traditional distributed topology control algorithms [2][3], each node in the network needs to execute the algorithm at least once, leading to a large amount of signaling exchange and overload. Meanwhile, the probability of packet collisions is greatly increased. In this study, we first find articulation points which can separate the network topology into some components, then execute the topology optimization algorithm only once in each connected component, which could reduce network load and greatly improve communication efficiency.

In this paper, we investigate wireless topology control mechanism for UAVNET based on articulation points, aiming to save power consumption and reduce communication overhead. We thus resort to discrete particle swarm optimization to solve the optimization problem. Considering distributed topology control, collaboration and interference between UAVs, we use swarm intelligence to solve the problem. Swarm Intelligence (SI) is a modern artificial intelligence that is concerned with collective behaviors in multi-agent systems. Instead of a sophisticated controller that governs the global behavior of the system, the swarm intelligence principle is based on individual agents that cooperate in order to exhibit a desired behavior[4]. SI techniques are typically inspired by natural phenomena, and they have exhibited remarkable capabilities in solving complex distributed problems [5]. Particle swarm optimization(PSO) algorithm is a kind of swarm intelligence—each particle use its own experience and group experience to decide its next iteration position. We choose the cross-variation operation of genetic algorithm to ensure the completeness of the feasible solution. In order to reduce the running time of the algorithm and improve the convergence speed, the algorithm provided in this paper only retains the feasible solution when updating all the particles

This work has been supported by the Natural Science Foundation of China (Grant No. 61871099), the Fundamental Research Funds for the Central Universities (Grant No. ZYGX2019J122).

in the storage. We conduct simulation experiments to validate the effectiveness of our proposed topology control scheme. Numerical results demonstrate that our proposed AP-DPSO algorithm outperforms conventional local minimum spanning tree (LMST) and local shortest path (LSP) algorithm in terms of network connectivity and robustness when UAVs are constantly moving.

The rest of the paper is organized as follows. In Section II, we describe the system model. Section III formulates the topology control problem. We present the solution to the UAVNET topology control problem in Section IV. Section V provides numerical results for the performance comparison. Finally, Section VI concludes the paper.

II. System Model

In this paper, the UAVNET is modeled as a graph *G=(V,E)*, where *V* and *E* represent the sets of nodes and links respectively, and all links in graph *G* are bi-directional. Each node can adjust its transmit power from 0 to p_{max} continuously, where p_{max} is the maximum transmission power. Let p_u and r_u represent the transmit power and transmit radius of node *u* respectively. When all nodes are operating at p_{max}, the graph is denoted by G_{max}=*(V,*E_{max}*)*. If the distance between nodes *u* and *v* is less than the communication radius of two nodes, there is a communication link between them, denoted as $e(u, v)$. Let G_u=*(*V_u*,*E_u*)* represent the local topology of node *u* when *u* works at p_{max}. V_u represents the one-hop neighborhood set of node *u*, and E_u represents edges between nodes.

$$V_u = \{v|v \in V, (u, v) \in E_{max}\} + \{u\}, \tag{1}$$

$$E_u = \{(x, y)|x, y \in V_u, (x, y) \in E_{max}\}, \tag{2}$$

We assume each node is equipped with an omnidirectional antenna, and has the same energy consumption characteristics. Each UAV node has a unique ID and location information, and all nodes can exchange information within a certain range. When nodes communicate and transmit data with each other, certain interference will be caused to the surrounding nodes within the transmission range. Let $d(u, |u, v|)$ represent the circular, where node *u* is located at the center with radius $|u, v|$. $I(u, v)$ is equal to the number of nodes that are in the range of $d(u, |u, v|)$ and $d(v, |v, u|)$, i.e.

$$I(u, v) = \{w \in V|d(u, w) < d(u, v) \cup d(v, w) < d(v, u)\}, \tag{3}$$

In this paper, considering the robustness, interference and Euclidean distance between nodes, the weight function of the communication link is designed as follows:

$$w_{i,j} = a_1 \times d(i, j)^{\alpha} + a_2 \times I(i, j) + a_3 \times \frac{1}{R(i, j)^{\beta}}, \tag{4}$$

The first term measures the power; the second one denotes link interference; and the last one indicates the robustness of the communication link. a_1,a_2,a_3 are the weighting factors, and *R(i,j)* indicates robustness of the communication link between *i* and *j*:

$$R(i, j) = \frac{e_i e_j}{(e_i^2 + e_j^2)^{\frac{1}{2}}}, \tag{5}$$

where e_i and e_j represent the remaining energy of nodes *i*and *j* respectively. In the FreeSpace propagation model, the relation between transmit power P_t and receiving power P_r can be characterized as:

$$P_r = \frac{P_t G_t G_r \lambda^2}{(4\pi d)^{\alpha} L}, \tag{6}$$

where G_t is the antenna gain of transmitter, G_r is the antenna gain of the receiver, λ is the wave length,d is the distance between the antenna of the transmitter and receiver, and L is the system loss. Since G_t,G_r,λ,L are constant in FreeSpace propagation model, the relation between P_t and P_r can be expressed as $P_t = CP_r d^{\alpha}$. Therefore, if the nodes *u* and *v*communicate directly, they should satisfy:

$$p_u(d_{uv})^{-\alpha} \geq \tau, \tag{7}$$

where α is the path loss factor, and we design α=2, τ is the minimum average SNR required for the receiver node.

III. Problem formulation

In the network topology control, since each node is interfered by its neighbors within its transmission range, it is necessary to constrain the connections of node to reduce communication interference between nodes. Thus the problem is transformed into a degree-constrained minimum spanning tree problem, which is proved to be a typical NP-hard problem [6]. We assume that $E = \{e_{12}, e_{13}, \cdots e_{ij}, \cdots e_{n-1,n}\}$ is the edge set of G_u, which is defined as follows:

$$e_{i,j} = \begin{cases} 1 & if\ v_i, v_j\ exits\ a\ wireless\ link \\ 0 & otherwise \end{cases}, \tag{8}$$

where $i = 1, 2, \cdots, n-1, j = i+1, \cdots, n$. Let *X* represent a spanning tree of G_u,$X = (x_{12}, x_{13} \cdots x_{ij} \cdots x_{n-1,n})$.

$$x_{i,j} = \begin{cases} 1 & if\ e_{ij} = 1\ and\ e_{ij}\ is\ selected \\ 0 & otherwise \end{cases}, \tag{9}$$

Therefore, the objective function of the model can be formulated mathematically as follows:

$$minf(x) = \sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}x_{ij} \tag{10}$$

$$\text{s.t.}\quad 1 \leq \sum_{\forall i \neq j \& i \in V_i} x_{ij} \leq d_i(j) \quad \forall\ i \in V_i \tag{10.1}$$

$$\sum_{i=1}^{n}\sum_{j=1}^{n} x_{ij} = n-1 \tag{10.2}$$

$$\sum_{i,j \in S} x_{ij} \leq |S| - 1\ \forall S \subset V and S \neq \emptyset \tag{10.3}$$

$$x_{ij} = \{0, 1\} \tag{10.4}$$

The first constraint represents the degree constraint of each code, where $d_i(j)$ is the degree upper bound of node *j* in local spanning tree X_i. The second constraint is to ensure that the spanning tree has *n-1* edges. The last is to avoid generating loops when constructing the spanning tree.

IV. TOPOLOGY CONTROL ALGORITHM

Our algorithm consists of four phases. First each node sends the HELLO data packet carrying its own ID number and node location information with the maximum transmit power, collects neighbor information. Then we use connection adjacency matrix (CAM) algorithm to find articulation points, so the entire network can be divided into some connected components by APs. Next we execute discrete particle swarm optimization algorithm (DPSO) in each connected component. At last, each node adjusts its transmit power according to the results.

A. Algorithms for Identifying APs and components

In this paper, we use the connection adjacency matrix (CAM) matrix method [10] to find articulation points. At the beginning of the detection, each node first sends a message to the neighboring nodes, including its own IP address, transmission time, and lifetime TTL(Time to live) threshold. Each node in the system has a link list: <IP address of the candidate node, time, link 1, link 2$\cdots$>, the node processes the received information through its stored link list.

When a node receives at least two same pieces of information from different links, it will return a confirmation message to the candidate node. The specific process is shown in Figure 1. Assume that the initial value of the TTL of the lifetime is 3, and the TTL value is decremented by 1 after each node forwards the probe information. The communication links between the candidate node and the neighboring nodes are labeled as 1, 2, 3, and 4, respectively. In Figure 1-(a), the candidate node sends the probe information to the neighboring nodes B, D, E, and G, and the TTL value becomes 2. In Fig 1-(c), nodes B, D, E, and G forward the received information to their neighbors. At this time, nodes C and F receive two pieces of probe information from different communication links, so node C returns the confirmation message of the received link 1, 2 to the candidate node. Node F returns the confirmation message of the received link 3, 4. Each candidate node maintains its own arrival list and

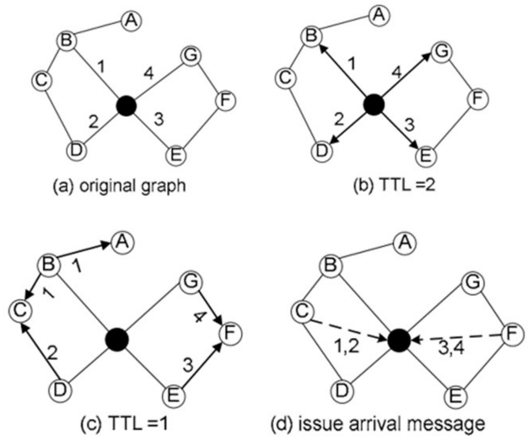


Fig. 1. Probe process.

CAM matrix. The format of the arrival list is: <IP address of the sending node, time, link 1, link 2$\cdots$>. The row and column of the CAM matrix respectively represents the number of the communication link between the node and the neighbor node, x represents the row number of the CAM matrix, and y represents the column number of the CAM matrix. When the link x and the link y are in the same arrival list, the corresponding position (x, y) and (y, x) have a matrix value of 1, otherwise set to 0. After reaching a certain time, the candidate node remaps the CAM matrix to a new undirected graph, if and only if (x, y) = 1, there is an edge between nodes x and y. When the number of connected components of the CAM graph of the node exceeds two, the algorithm determines that the candidate node is an articulation point.

According to the detected articulation points, the entire network topology can be divided into some connected areas. It is known that a connected graph can be decomposed into some connected components, and they are connected to each other through articulation points.

Traditional methods for calculating connected components include the Hopcroft and Tarjan algorithms based on depth-first search [11]. Although these algorithms can accurately detect articulation points and segment out connected components, they usually require long message delivery and collect complete network topology information. In the distributed network topology control, we divide the connected components by the detected articulation points. It is assumed that each node has its own ID number and the corresponding number of connected components. First, each articulation point starts to send its own ID information to neighboring nodes. When the neighbor node receives the information, it first determines whether it has been visited before. If not, it sets the value of received node to its own CN value of the component, otherwise, it compares the previous stored CN value with the received node ID value, then selects a larger value assigned to the CN. Then the neighbor node continues to forward the message to its neighboring nodes until the receiver is an articulation point.

B. Particle Swarm Algorithm Design

1) Population Initialization: In discrete particle swarm optimization, each particle represents an initial solution to an optimization problem, and the size of the population is the number of particles, which usually is a fixed value or can be changed according to some strategies with iteration. The size of the population affects both the complexity and the optimization performance of the algorithm. Reasonable selection of the particle can effectively reduce the number of iterations for the final result at a faster speed. In order to reduce the running time of the algorithm and improve the convergence speed, we consider two constraints when building the initial space: a) Satisfy the degree constraint. First, we design a set N with indefinite length, and N contains all nodes. The total number of occurrences of each node in the set is equal to d-1 ,where d is maximum degree. For example, assume node set is 1,2,3,4,5, and the degree constraint of every node is 4, then set N is 1,1,1,2,2,2,3,3,3,4,4,4,5,5,5. We use prüfer sequence to encode spanning trees. Each spanning tree in the graph can be represented by a prüfer sequence. Thus we can select n-2 numbers directly from set N . b) The initial solution must match with a spanning tree of the neighborhood graph. For complete graphs, every prüfer sequence can match with a spanning tree in the graph. But it is not true for incomplete connected graphs. Therefore, we also need to remove the leaf nodes and ensure the generated edges are a subset of the original edges after selecting *n-2* numbers from set *N*.

2) Particle Coding Mechanism: According to the features of spanning trees, in our algorithm we adopt the prüfer sequence to encode every spanning tree. Encoding process is stated as follows:

a)Mark the node number with the smallest label in the leaf node of the tree T as *j*;

b)If k is neighbor node of *j*, add k to the end of the encoded string;

c)Delete *j* and the edges connecting *j* and *k* in tree *T*. At this time, the tree only contains *n-1* vertices;

d)Repeat the above steps until there is only one edge left in the tree *T*.

3) Update particle: In order to maintain the diversity of the population, the mutation and crossover operator of the genetic algorithm are incorporated to avoid falling into the local optimal solution. We use the mutation operator *M* in the genetic algorithm. X_i^{t-1} represent the solution of the *i-th* particle in the *t-1-th* generation.

$$A_i^t = F_1(X_i^{t-1}, w) = \begin{cases} M(X_i^{t-1}) & r_1 < w \\ X_i^{t-1} & otherwise \end{cases}. \quad (11)$$

The mutation operator *M* is as follows: first randomly select a position point *k* in the particle *X*, then randomly select a value from neighbor node set $1, 2, 3, \cdots, n$ to replace the original value.

$$B_i^t = F_2(A_i^t, c_1) = \begin{cases} C_p(A_i^t) & r_2 < c_1 \\ A_i^t & otherwise \end{cases}, \quad (12)$$

$$X_i^t = F_3(B_i^t, c_2) = \begin{cases} C_g(B_i^t) & r_3 < r_2 \\ B_i^t & otherwise \end{cases}. \quad (13)$$

r_1, r_2, r_3 are random values between 0~1, $C_p(A_i^t)$represent A_i^t will intersect with the *i-th* particle's own optimal solution p_i. $C_g(B_i^t)$represent A_i^t will intersect with the global optimal solution p_g.C_p, C_g are two-point crossover operators. As the picture shows: A_i is the current particle. Its own optimal solution is p_i ,then we randomly select two locations k_1 and k_2 for crossover.

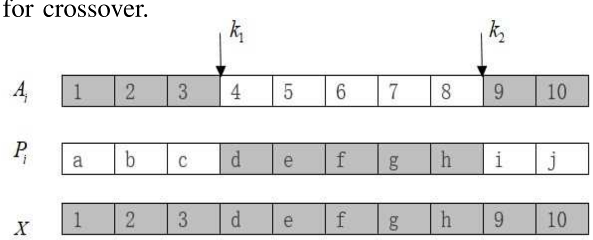


Fig. 2. Two-point hybridization of particles.

Therefore, the position update of particle *X* can be expressed as:

$$X_i^t = F_3(F_2(F_1((X_i^{t-1}, w), c_1), c_2). \quad (14)$$

4) Parameter Settings: The inertial weight *w* reflects how much the previous particle's speed influence the current particle's speed. Acceleration factor c_1, c_2 adjust the particle's own experience and group experience for influencing the particle's trajectory. If c_1 is 0, the particle is only influenced by group experience. At this time, the convergence speed is faster, but it may lead to local convergence; if c_2 is 0, only its own experience plays a role in the motion of the particles. In order to improve the search efficiency of the algorithm, the algorithm adopts the following parameter adjustment methods:

TABLE I
SIMULATION PARAMETERS

Parameter	Value
Area	1000*1000
Number of UAV	30
Maximum emission radius	400
Propagation model	Free space model
Mobility model	Leader-follower
Path loss factor	$\alpha = 2$

TABLE II
DPSO ALGORITHM PARAMETERS

Parameter	Value
Maximum number of iterations T	800
Inertia weight	[0.4,0.8]
r_1, r_2, r_3	[0,1]
Initial population size	200

$$w = (w_{max} - w_{min}) \times (gen - mgen)/gen + w_{min}, \quad (15)$$

$$c_1 = 1.0 - mgen/gen, \quad (16)$$

$$c_2 = 1.0 - c_1. \quad (17)$$

5) Fitness value function: When calculating the fitness value of a particle, we need to first decode the sequence into a spanning tree. The specific decoding process is:

a) The code string is P,and the set of those numbers whose vertex numbers do not appear in P is $\overline{P}$;

b) The smallest number in $\overline{P}$ is *j*, and the leftmost number in P is *k*. Then add the edge connecting *j* and *k* in the tree, and delete *j* and *k* from and $\overline{P}$ respectively.If *k* is no longer appear in P, add *k* in $\overline{P}$;

c) Repeat the above steps until P is empty;

d) When P is an empty string, there are only two vertex numbers left in $\overline{P}$, and add the edge connecting the two vertices to the tree. In this paper, we use the objective optimization function, as the fitness value function directly.

$$f(x) = \sum^n \sum^n w_{ij} x_{ij}. \quad (18)$$

C. TOPOLOGY CONTROL ALGORITHM

Based on the above analysis, we propose a heuristic algorithm for topology control based on DPSO, as shown in Algorithm 1.

V. PERFORMANCE EVALUATION

A. Simulation Configuration and Metrics

Assume that 30 UAV nodes are uniformly deployed in the area, each node maintains maximum transmit power at the beginning. Propagation model is free space model, and we set path loss factor. Assume that the remaining energy of each node is a random value between 10 and 40.

In order to evaluate the proposed algorithm, the following performance metrics are used.

Algorithm 1 Discrete Particle Swarm Optimization

input: Original local network topology $G_u(V_u, E_u)$,maximum transmission radius of the node R_{max}.

output: Node *u*'s local degree-constrained minimum spanning tree.

Initialization:
i)Create a set *N* based on degree constraints, generate *M* initial solutions by directly selecting *n-2* number from *N*;Check whether it is a feasible solution, if no, delete it. Then get the final initial population *X* and population size *H*.
ii)Set maximum number of iterations T , inertia weight w_{min}, w_{max}, acceleration factor c_1, c_2, $t = 0$ (Number of iterations)
Calculate individual fitness $A_i^{h,0}$ in *X*.Let $p_{best}^i = A_i^{h,0}$.Find the minimum individual fitness, record the corresponding optimal solution g_{best}.
Repeat
if($r_1 < w$)
Perform mutation operation to all particles.
else if($r_2 < c_1$)
Each particle crossovers with the personal optimal solution p_{best}^i.
else if($r_3 < r_2$).
Each particle crossovers with the personal optimal solution g_{best}.
Check whether the particles are feasible solutions after the operation, if not, the particles remain the same.
Obtain population X_t .Calculate the fitness of each particle, update the optimal solution of the particle itself p_{best}^i and the global optimal solution g_{best}.
t=t+1.
Get the global optimal position g_{best} as output.

1)Average degree: It is the average degree of nodes in the network after the execution of the topology control algorithm. In general, the lower the average degree is, the lower the probability of message collisions is:

$$d(G) = \frac{1}{n}\sum_{u=1}^{n} d(u) \tag{19}$$

2)Average link length: The link length is the transmission length between two nodes, which is critical for point to point communication. The average link length in the network reflects the power consumption level. A small average link length indicates low power consumption.

$$APL(G) = \frac{\sum_{i \geq j} l_{i,j}}{\frac{1}{2}N(N-1)} \tag{20}$$

3)Average link robustness: It reflects the entire network robustness.

$$R_{i,j} = \frac{e_i \times e_j}{(e_i^2 + e_j^2)^{1/2}} \tag{21}$$

4)Connectivity ratio: It demonstrates the connectivity of network and is calculated as the percentage of connected node pairs out of the total number of node pairs in the network.

$$C = \frac{\sum_{u,v \in N} c_{uv}}{|N|(|N|-1)} \tag{22}$$

$$c_{uv} = \begin{cases} 1 & if\ u \neq v\ and\ (u,v)\ is\ connected \\ 0, & otherwise \end{cases} \tag{23}$$

B. Leader-follower Mobility Model

In the UAVNET, the network topology is changeable due to the movement of UAVs. Assume that UAVs have the same motion characteristics, and each UAV has its own direction.

$$\begin{bmatrix} x_i \\ y_i \\ \theta_i \end{bmatrix} = \begin{bmatrix} v_i^x \\ v_i^y \\ w_i \end{bmatrix} = \begin{bmatrix} cos\theta_i & 0 \\ sin\theta_i & 0 \\ 0 & 1 \end{bmatrix} \begin{pmatrix} v_i \\ w_i \end{pmatrix}, i = 1, 2, \ldots, n \tag{24}$$

where x_i, y_i is the location of node *i*, and θ_i is the azimuth of node *i*. Speed v_i and angular velocity w_i are the control inputs for the UAV.

Taking a formation system consisting of three UAVs and one leader as an example, three UAVs form an isosceles triangle, and the formation leader is at the midpoint of the bottom edge and moves along the planning path. During the formation flight, each UAV and leader must maintain a certain relative distance and azimuth with the direction of the leader's movement, that is, each UAV will change the relative position of the leader according to the direction of motion.

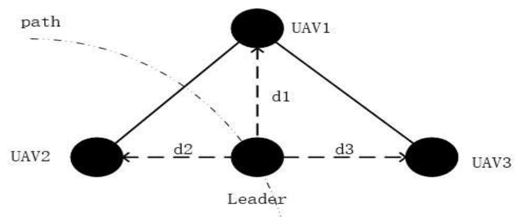


Fig. 3. Mobility model.

C. Numerical Results

In order to visualize the performance of the topology control algorithms, we first plot an instance where 30 UAVs are uniformly distributed in a $1000 \times 1000\ m^2$ area and assume that all nodes work at maximum transmission power at the beginning. Fig.4 exhibits the topologies derived from LMST, LSP, and the proposed DPSO. It can be seen that the network topology obtained by LMST is sparse. Although such a structure can reduce power consumption and interference, it also means failure of any one or two nodes may cause the entire network to be disconnected. The topology diagram obtained by the LSP algorithm has higher redundancy and it has not considered the robustness of topology structures. The DPSO algorithm has made a good trade-off between low power consumption, high robustness and low interference. We next compare the network connectivity under three network topology control algorithms at different moving speeds. As figure 5 shows, the DPSO algorithm can maintain better network connectivity than other algorithms while the nodes are constantly moving, so that the stability of the network is better and the communication will not be interrupted by the change of node position.

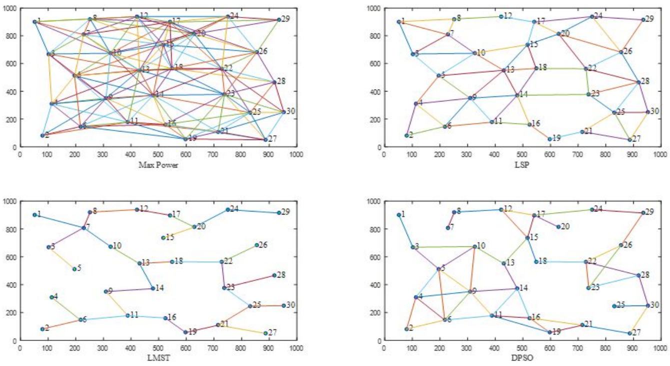


Fig. 4. Network topologies derived from different algorithms.

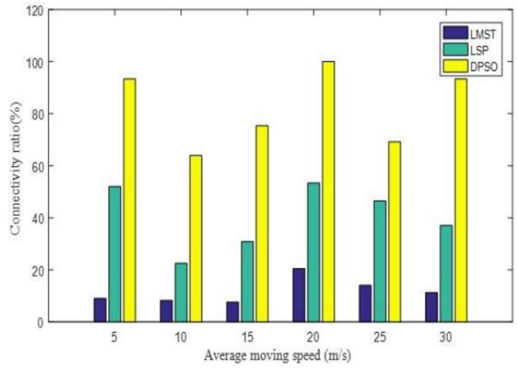
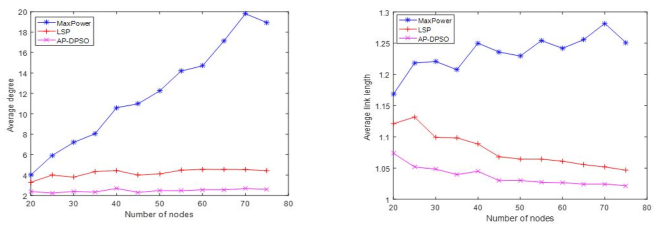


Fig. 5. Network connectivity.

Then we change the number of UAV nodes and study the algorithm performance under different node density conditions. Fig. 6(a) shows that topology control algorithms can reduce the average node degree. The average node degree obtained by the DPSO algorithm basically satisfies the degree constraint, and it does not change greatly with the increase of the number of nodes. In contrast, our algorithm has a lower average node degree than LSP. Fig. 6(b) shows that the average link length decreases obviously after topology control. The smaller the average link length, the smaller the required transmit power. So DPSO algorithm has better performance in saving power and prolonging network lifetime. We next investigate the average



(a) Average degree. (b) Average link length.

Fig. 6. Comparisons of LSP and DPSO.

link robustness under different algorithms. We calculate the robustness of the link by the remaining energy of the node. The higher the energy of the link end node, the less likely the link is disconnected, and the higher the stability of the whole network is. Fig.7 shows that the average link robustness of the network topology obtained by the DPSO algorithm is significantly higher than other algorithms, which shows our algorithm can greatly improve network stability and enhance the overall communication link.

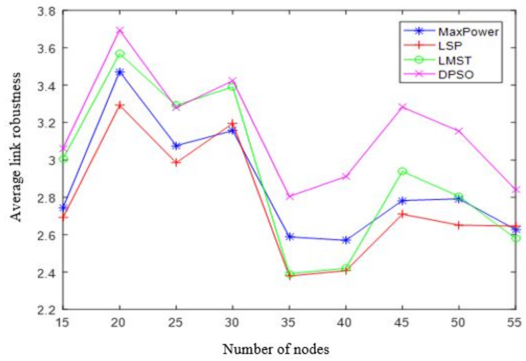


Fig. 7. Average link robustness.

VI. Conclusion

In this paper, we have investigated the topology control problem of the UAVNET aiming to improve network connectivity and reduce communication overhead. We first find articulation points in the network according to the CAM algorithm, then the network could be divided into some components. We consider the robustness, interference and power loss of the link when designing the link weight, so the topology optimization problem is modeled as finding local degree-constrained spanning tree. The problem is solved by the discrete particle swarm algorithm. The simulation results show that the discrete particle swarm optimization algorithm can maintain higher network connectivity than other algorithms while the nodes are constantly moving. Moreover, the performance in terms of network robustness, reducing interference are also significantly improved.

References

[1] Gupta, Lav, Raj Jain, and Gabor Vaszkun. "Survey of important issues in UAV communication networks." IEEE Communications Surveys & Tutorials 18.2 (2016): 1123-1152.

[2] Li N, Hou J C, Sha L. Design and analysis of an MST-based topology control algorithm[J]. IEEE Transactions on Wireless Communications, 2005, 4(3): 1195-1206.

[3] Li X Y, Weng Y, Wan P J, et al. Localized low-weight graph and its applications in wireless ad hoc networks[C]//INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies. IEEE, 2004, 1: 431-442.

[4] Li X, Clerc M. Swarm intelligence[M]//Handbook of Metaheuristics. Springer, Cham, 2019: 353-384.

[5] Slowik A, Kwasnicka H. Nature inspired methods and their industry applications—Swarm intelligence algorithms[J]. IEEE Transactions on Industrial Informatics, 2018, 14(3): 1004-1015.

[6] Narula S C, Ho C A. Degree-constrained minimum spanning tree[J]. Computers & Operations Research, 1980, 7(4): 239-249.

[7] Sabino S, Grilo A. Topology control of unmanned aerial vehicle (uav) mesh networks: A multi-objective evolutionary algorithm approach[C]//Proceedings of the 4th ACM Workshop on Micro Aerial Vehicle Networks, Systems, and Applications. ACM, 2018: 45-50.

[8] Jingnan L, Pengfei L, Kai L. Research on UAV communication network topology based on small world network model[C]//2017 IEEE International Conference on Unmanned Systems (ICUS). IEEE, 2017: 444-447.

[9] Sahingoz O K. Networking models in flying ad-hoc networks (FANETs): Concepts and challenges[J]. Journal of Intelligent & Robotic Systems, 2014, 74(1-2): 513-527.

[10] X. Liu, L. Xiao, A. Kreling, and Y. Liu, "Optimizing overlay topology by reducing cut vertices," in Proc. 2006 Int. Workshop Netw. Oper. Syst.

[11] Support Digit. Audio Video, 2006, pp. 17:1–17:6.J. Hopcroft and R. Tarjan, "Algorithm 447: Efficient algorithms for graph manipulation," Commun.ACM, vol.16,no.6,pp.372378 Jun.1973.