Cooperative Reinforcement Learning Aided Dynamic Routing in UAV Swarm Networks

Zunliang Wang*, Haipeng Yao*, Tianle Mai*, Zehui Xiong† and F. Richard Yu‡
*State Key Lab. of Net. and Switching Tech., Beijing Univ. of Posts and Telecom., Beijing, P.R. China
†Singapore Univ. of Tech. and Design, Singapore
‡Depart. of Sys. and Comp. Eng., Carleton Univ., Ottawa, ON, Canada

ICC 2022 - IEEE International Conference on Communications | 978-1-5386-8347-7/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICC45855.2022.9838808

***Abstract*—The Unmanned Aerial Vehicle (UAV) swarm has attracted widespread attention from both academia and industry. It has been widely adopted in disaster recovery, military communication, agricultural production, and industrial automation. In critical situations or places where communication infrastructure is lacking, deploying a UAV swarm network is a cost-effective solution. However, considering the high speed of UAV devices, designing an effective routing mechanism has been a challenging problem. In this paper, enlightened by the recent success of multi-agent reinforcement learning, we propose a multi-agent policy gradients-based UAV routing algorithm. We adopt a centralized training and decentralized executing framework, where a centralized training platform is implemented to guide the policy updating of each UAV node. Moreover, we introduce a counterfactual baseline scheme in our algorithm to improve the convergence speed. Extensive simulation results validate the effectiveness of the proposed algorithms compared to the state-of-the-art schemes.**

***Index Terms*—UAV, dynamic routing, multi-agent system, reinforcement learning**

I. Introduction

The past few years have witnessed the compelling applications of the Unmanned Aerial Vehicle (UAV) swarm ranging from military to commercial domains. With the explosion of various relevant applications [1], [2], the expectations for reliable and efficient UAV swarm networks are more significant than ever [3]. However, compared to the fixed terrestrial networks, the high-speed mobility of UAV nodes has resulted in a new clan of networks known as flying ad-hoc networks (FANETs). This new network presents many challenges in routing scheme design, such as dynamic topology, unstable link [4].

Recently, a rapidly growing number of studies have investigated designing the effective UAV swarm routing algorithm. In [5], Khaledi *et al*. proposed a distance-greedy routing algorithm, where each UAV node makes routing decisions based on the local observation. In [6], Wen *et al*. proposed a distributed routing method, in which the nodes transmit packets with the help of other cooperative nodes. These distributed schemes have severe non-convergence problems, especially when seeking a global optimum. In [7], Coelho *et al*. proposed a centralized routing solution that can select high-capacity paths among UAVs and avoid communications disruptions. In [8], Detti *et al*. proposed the Software-Defined Networking (SDN) based routing management method to increase the overall throughput of the network. However, the centralized solutions require collecting massive data from the controller, thus exhibiting delay to respond to network dynamics.

In this paper, inspired by the recent success of multi-agent reinforcement learning, we propose a hybrid routing control algorithm. We adopt a centralized training and decentralized executing framework, where a centralized training platform is implemented to guide the policy updating of each UAV node. Note that compared to the centralized solution, the centralized platform in our approach only acts as a policy critic rather than the controller. Based on this, we design a multi-agent policy gradients-based UAV routing algorithm. By adopting the centralized training and distributed execution framework, the UAV nodes can learn the globally optimal policy cooperatively. Besides, we introduce a counterfactual baseline scheme to explicitly calculate each agent's contribution to the global reward to enhance the convergence speed. Extensive simulation results are provided to demonstrate the effectiveness and feasibility of our proposed algorithms.

Our main contributions are summarized as follows.

- We introduce a multi-agent reinforcement learning algorithm into the highly dynamic network environment. This method can form an efficient multi-hop routing mechanism by using the centralized training decentralized execution framework and a counterfactual baseline scheme.
- Our proposed dynamic routing method can jointly optimize network quality in terms of survival time, packet delivery rate, and throughput.
- We have conducted extensive experiments in different dynamic network environments to verify the effectiveness of our deployed methods.

The rest of this paper is organized as follows. In Section II, we present the system model and problem formulation. In Section III, we propose a multi-agent policy gradients-based UAV routing algorithm in our system. In Section IV, we present the simulation results, and the conclusion is in Section V.

II. System Model and Problem Formulation

In this section, we present our system model, routing model, and problem formulation.

A. System Model

As shown in Fig. 1, we consider a set of UAV nodes $K = [k_0, k_1, ..., k_n]$ distributed in a set of zones $Z =$

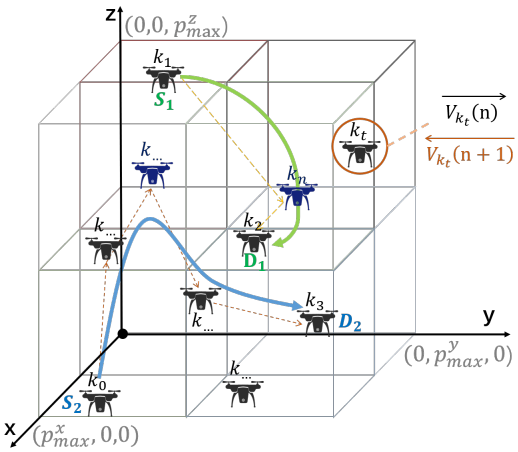


Fig. 1. System Model

$[zone_0, zone_1, ..., zone_m]$. We denote the system boundary as $(0,0,0),(p^x_{max},0,0),(0,p^y_{max},0),(0,0,p^z_{max})$, where p^x_{max}, p^y_{max} and p^z_{max} are the maximum boundary values of the X, Y, and Z dimensions, respectively. At discrete-continuous time n, the position of node $k_i \in K$ can be described as: $\overrightarrow{P_{k_i}(n)} = (p^x_{k_i}(n), p^y_{k_i}(n), p^z_{k_i}(n))$, and its speed can be described as: $\overrightarrow{V_{k_i}(n)} = (v^x_{k_i}(n), v^y_{k_i}(n), v^z_{k_i}(n))$. The movement in each dimension can be formulated as follows:

$$p^{\varphi}_{k_i}(n+1) = \begin{cases} p^{\varphi}_{k_i}(n) + v^{\varphi}_{k_i}(n), \text{if } p^{\varphi}_{k_i}(n) + v^{\varphi}_{u_i}(n) \in [0, p^{\varphi}_{max}) \\ p^{\varphi}_{u_i}(n) - v^{\varphi}_{u_i}(n), \text{else (need speed change)} \end{cases} \tag{1}$$

where $\varphi \in (x,y,z)$ represents different dimensions in 3-D space. Moreover, we use the 3-D Mixed Mobility Model [9] to formulate the mobility of UAV nodes. This model is based on the traditional MANET (Mobile Ad Hoc Network) Mobility Models [10] and therefore can factually simulate the movement of UAV nodes. Firstly, the movement process for each UAV node on Z-axis is modeled as a Random Way Point Mobility Process, which includes the following steps:

1) The UAV node randomly selects height h_t from $[p^l_{zone_i} \sim p^h_{zone_i}]$ as initial height, where $p^l_{zone_i}$ and $p^h_{zone_i}$ represent the lower bound and upper bound of $zone_i$.

2) The UAV node at h_t randomly selects height h_{t+1} from $[p^l_{zone_i} \sim p^h_{zone_i}]$ and moves towards it with a speed choosing uniformly random from $[V_{min} \sim V_{max}]$.

3) Once the node reaches h_{t+1}, it has a certain probability p_s of keep staying for a period of time T_s. The T_s is selected randomly from $[T_{min} \sim T_{max}]$, which corresponds to the minimum staying time and maximum staying time, respectively.

4) The UAV nodes repeat the movement process from step 2) to step 3). In the above steps, the staying probability p_s can be calculated by the following equation:

$$p_s = \frac{\mathbb{E}[T_s]}{\mathbb{E}[T_s] + \mathbb{E}[T_m]}, \tag{2}$$

where

$$\mathbb{E}[T_m] = \frac{ln(V_{max}/V_{min})}{V_{max} - V_{min}} \times \frac{p^z_{max}}{3} \tag{3}$$

and $\mathbb{E}[T_s] = (T_{min} + T_{max})/2$ is the mean stay time.

Secondly, the movement process on the XY plane belongs to a Uniform Mobility Model. When the UAV node is hovering on the Z-axis in period T_s, it will choose the velocity component of the X-axis and Y-axis randomly from $[V_{min} \sim V_{max}]$, together forming a new velocity vector. The above process can be described as: $\overrightarrow{V_{k_i}(n+1)} = (v^x_{k_i}(n+1), v^y_{k_i}(n+1), v^z_{k_i}(n))$. Besides, in this paper, we assume that the routing process is deployed in a limited 3-D space. Thus, the nodes will move in the opposite direction once they reach the region boundary.

B. Routing Model

In this part, we present the routing model in our system. We assume that the communication range of each UAV node is restricted to its own zone and neighbor zones. Besides, we assumed that the routing model is based on the hop by hop routing paradigm [11]. Two routing processes with a set of source-destination (SD) pairs are shown in Fig. 1: $\langle S_1, D_1 \rangle$ from k_1 to k_2 and $\langle S_2, D_2 \rangle$ from k_0 to k_3. The network environment has a set of packets $PKT(n)$, and the number of packets will increase as the new packet is injected into the network continuously.

Meanwhile, we introduce a packet dropout mechanism caused by transmission timeout. We designed the maximum transmission hop count as N^{max}_{hop}. For any data packet $pkt_i \in PKT(n)$, we consider the transmit process from $pkt^{S_k}_i \rightarrow pkt^{D_k}_i$: the packet will be discarded once $pkt^{N_{hop}(n)}_i \geq N^{max}_{hop}$, and set $pkt^{State}_i = -1$. Otherwise, if pkt_i is successfully transmitted, then set $pkt^{State}_i = 1$ and set $pkt^{N_{hop}(n)}_i$ to the constant value of the final hop count.

Moreover, we apply a queueing model for each node to smooth traffic fluctuation. For node $k_i \in K$, it has a buffer queue $Q_{k_i}(n) \in [0, Q^{max}_{k_i}]$, where $Q^{max}_{k_i}$ is the maximal capacity. We describe the input gate and the output gate as $g^{in}_{k_i}$ and $g^{out}_{k_i}$. Also, the increase of queue length can be formed as follows:

$$\Delta Q = v_{g^{in}_{k_i}}(n) - v_{g^{out}_{k_i}}(n), \tag{4}$$

$$\Delta Q_{k_i}(n+1) = \begin{cases} \Delta Q, & \text{if } 0 \leq Q_{k_i}(n) \leq Q^{max}_{k_i} \\ min(\Delta Q, 0), & \text{else} \end{cases}, \tag{5}$$

$$Q_{k_i}(n+1) = Q_{k_i}(n) + \Delta Q_{k_i}(n+1). \tag{6}$$

In the above equations, $v_{g^{in}_{k_i}}(n)$ and $v_{g^{out}_{k_i}}(n)$ represents the packet arrival rate from upstream and the packet service rate to downstream, respectively. We pre-cache ω packets in each UAV node $k_i \in K$ to maintain a high traffic load pressure in the network environment:

$$Q_{k_i}(0) = \omega. \tag{7}$$

Once the buffer queue is filled, the packet drop happens, and the network is considered as occurring the network congestion.

C. Problem Formulation

The dynamic routing process has two critical problems: How to increase the delivery ratio of the packet, and how to alleviate the traffic congestion. Therefore, the objective optimization function can be formulated as:

$$\lim_{n\to\infty} \textbf{min}\{\sqrt{\frac{1}{|K|} * \sum_{k_i \in K} (Q_{k_i}(n) - \overline{\sum_{k_j \in K} Q_{k_j}(n)})^2} + \sum_{pkt_i \in PKT(n)} pkt_i^{N_{hop}^{(n)}}\}. \quad (8)$$

Based on the above equation, the optimization objective includes two parts. The first part is to reduce the mean square error (MSE) of all buffer queue lengths, and hence it can relieve the traffic pressure among all the UAV nodes. The second part is to lower the mean hop count $pkt_i^{N_{hop}(n)}$ for all packets $pkt_i \in PKT(n)$ so that fewer packets will be dropped.

Moreover, the local observation space for each agent k_i at moment n can be described as follows:

$$[zone(D_k) \times 10 + pkt_j^{D_k}, zone(k_i) \times 10 + i, Q_{k_i}(n)], \quad (9)$$

which consists of three parts: the mixed id (zone id with node id) of packet pkt_j's destination node, the mixed id of node k_i, and the occupancy of its queue buffer. Also, the action space can be described as $[zone_{pkt_j}^{next}(n)]$, which denotes the next transmit zone of the current packet pkt_j in UAV node k_i. We designed the reward function as follow:

$$\begin{aligned} r_{k_i}(n) = \rho \times [&Q_{k_i}(n) - \overline{\sum_{k_t \in zone_{pkt_j}^{next}(n)} Q_{k_t}(n)}] \\ &+ \mu \times [dis(zone_{pkt_j}^{next}(n), zone(pkt_j^{D_k})) \\ &- dis(zone(k_i), zone(pkt_j^{D_k}))], \end{aligned} \quad (10)$$

where $\overline{\sum_{k_t \in zone_{pkt_j}^{next}(n)} Q_{k_t}(n)}$ is the mean queue buffer length among the UAV nodes in pkt'_j next-hop zone area, function $dis(A, B)$ calculates the distance between A and B, and ρ, μ are the scale factors.

The above equation indicates that our routing algorithm has the following functions. The first part is to transmit packets among each UAV node to balance the occupancy of their queue buffer. The second part is to calculate whether the next-hop zone of pkt_j is closer to its destination zone, therefore decreasing the total hop count $pkt_j^{N_{hop}(n)}$ (i.e., shortest path).

III. Multi-agent Reinforcement Learning

Inspired by the recent success of multi-agent reinforcement learning, we design a multi-agent reinforcement learning-based routing algorithm.

A. Decentralized Markov Decision Processes

Multiple UAV nodes form a decentralized Markov Decision Process (MDP) model. In the decentralized MDP, depending on the action they executed, each agent gained immediate reward. Meanwhile, the environment will change to a new state [12]. In our paper, the current system state can be formulated as S. Each agent has their local observations, which can be represented by S: $[o_1, ..., o_n]$. Then, the agents choose action $[u_1, ..., u_n]$ by their local policies $[\pi_{\theta_1}, ..., \pi_{\theta_n}]$. In the system, the observations of all agents form a joint observation $\mathbf{O} = [o_1, ..., o_n]$, and the actions form a joint action $\mathbf{U} = [u_1, ..., u_n]$. The change of network state can be described as $S \times \mathbf{U} \rightarrow S'$. Each agent gets the same reward $r(S, \mathbf{U})$, also received a new local observation from the changed network state: $S' \rightarrow [o'_1, ..., o'_n]$. Besides, we denoted the τ_t^i as the action-observation history.

The goal of each agent is to maximize the cumulative discount reward:

$$R_t = \sum_{k=0}^{\infty} \gamma^k r^{t+k}, \quad (11)$$

where γ is the discount factor and $\gamma \in [0, 1]$. Besides, combining the joint policy $\pi = [\pi_{\theta_1}, ..., \pi_{\theta_n}]$, system state S_t, joint action $\mathbf{U}$ and the cumulative discount reward, a joint value function can be designed as: $Q^{\pi}(S_t, \mathbf{U}_t) = \mathbb{E}_{S_{t+1:\infty}; \mathbf{U}_{t+1:\infty}}[R_t | S_t, \mathbf{U}_t]$.

B. Multi-agent Policy Gradients Based Routing Algorithm

The easiest way to solve our dynamic routing problem is to maintain multiple single-agent algorithms in parallel. In this section, we firstly propose a PG-based routing algorithm. PG is a traditional policy-gradient based single-agent reinforcement learning algorithm [13], the gradient can be described as:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta}[\nabla_\theta log \pi_\theta(S, a) Q^{\pi}(S, a)], \quad (12)$$

where a is the action chosen by the single agent.

In order to enhance the cooperative learning ability of distributed UAV nodes, we further propose a multi-agent policy gradient-based routing algorithm. As shown in Fig. 2, it maintains a centralized training with a decentralized execution framework. In the training phase, we implement a training center to generate the joint value $Q(S, \mathbf{U})$ by inputting the environment state S and the joint action $\mathbf{U} = [u_1, ..., u_n]$ into the neural network, while the policy of each decentralized actor is trained by following a gradient depends on $Q(S, \mathbf{U})$. Then, in the execution phase, each agent chooses the action u_i based on their local observation o_i. Notice that we use the $\epsilon - greedy$ scheme during the action selection process to balance exploring and exploiting.

Moreover, we propose an advanced counterfactual baseline method to effectively evaluate each agent's contribution to the global reward, so as to enhance the convergence speed:

$$A^i(S, \mathbf{U}) = Q(S, \mathbf{U}) - \sum_{u'^i} \pi^i(u'^i | \tau^i) Q(S, (\mathbf{U}^{-i}, u'^i)). \quad (13)$$

The equation subtracted the average action Q-value of agent i from the global Q-value (calculated from the centralized

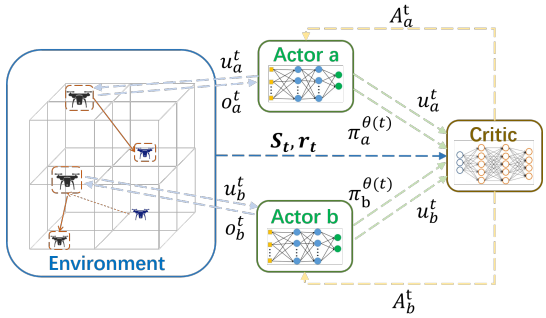


Fig. 2. The Multi-agent Actor Critic Reinforcement Learning Framework

training center). Therefore, each agent can obtain a precise, independent reward, and the algorithm can finally gain a better convergence ability. We use the following equation to calculate the gradient of agent i:

$$\nabla_{\theta_i} J(\theta_i) = \mathbb{E}_{\pi_{\theta_i}}[\nabla_{\theta_i} log \pi_{\theta_i}(u^i|\tau^i) A^i(S, \mathbf{U})]. \quad (14)$$

Besides, our routing algorithm trains the critic in an on-policy scheme, the loss function can be described as follows:

$$\mathcal{L}(\theta^c) = (\gamma^{(\lambda)} - f^c(\cdot_t, \theta^c)), \quad (15)$$

where $\gamma^{(\lambda)} = (1-\lambda)\sum_{n=1}^{\infty} \lambda^{b-1} G_t(n)$. $G_t(n) = \sum_{l=1}^{\infty} \lambda^{l-1} r_{t+l} + \gamma^n f^c(\cdot_{t+n}, \theta^c)$, which can be calculated by the TD(λ) method using a mixture of n-step returns [14]. The dynamic routing algorithm is shown in Algorithm 1.

Algorithm 1 The dynamic routing algorithm

Initialize the system environment and the neural network parameters
for $episode = 1$ to $MAX_{episode}$ **do**
  Initialize the network environment, redeployed each UAV node, and clear their queue buffer
  for each UAV node $k_i \in K$ **do**
    Generate ω packets with randomly $< S, D >$ pair into buffer Q_{k_i}
  end for
  for $t = 1$ to MAX_{step} **do**
    for each UAV node $k_i \in K$ **do**
      if $Q_{k_i}(t) = Q_{k_i}^{max}$ **then**
        End episode
      end if
      Get pkt_i from queue buffer, generate observation o_i^t and choose action u_i^t based on policy $\pi_{\theta_i}^t$
      Transmit pkt_i to other UAV nodes according to u_i^t
    end for
    Form joint action $\mathbf{U}^t$, change the network environment $S_t \rightarrow S_{t+1}$, each agent get reward r_t and new observation o_i^{t+1}
    Generate new packets and handle the moving process for each UAV node
  end for
  Update critic network through through Eq. (15)
  Update actor network for agents through Eq. (14)
end for

IV. Simulation Results and Discussions

In this section, we firstly discuss network environment settings and the simulation settings. Secondly, we present the simulation results to evaluate the performance of our proposed algorithm. Our experiments simulate Ubuntu 16.04 with 32g RAM, Nvidia RTX 2060, and intel i7-10875H.

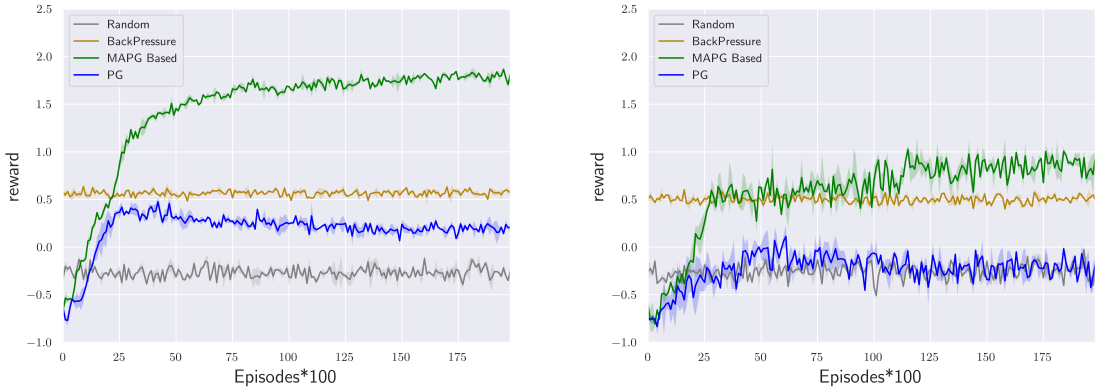
A. Environment Settings

In the experiment, the boundary value of our experiment is: $p_{max}^x = 10m$, $p_{max}^y = 10m$, $p_{max}^z = 10m$. We set the total number of UAV nodes and space zones as $N = 10$ and $M = 8$, and the initial deployment process is described as follows: UAV nodes $[k_0, k_1, ..., k_7]$ are placed in zones $[zone_0, zone_1, ..., zone_7]$, respectively, and UAV nodes $[k_8, k_9]$ are randomly deployed in the whole zone area. This initialization process will be executed in each training episode. Besides, we set the speed of UAV nodes as: $V_{max} = 2.6m/s$, $V_{min} = 1.2m/s$. Furthermore, we set the minimum staying time T_{min} and maximum staying time T_{max} as $2s$ and $6s$, respectively. The staying probability is $p_s \approx 0.5$, as calculated from Eq. (2). In each training step, our environment will generate a new packet in each UAV node, and the node can only transmit a single packet to the downstream in each training step (i.e., $v_{g_{k_i}^{out}}(n) = 1$). Other parameter settings include: $N_{hop}^{max} = 4$, $\omega = 30$, and $\forall k_i \in K, Q_{k_i}^{max} = 50$.

Finally, we configure the network environment at different dynamic levels to evaluate the performance of our proposed algorithm. For the UAV swarm, considering its high-speed mobility, it is necessary to deploy a backbone network architecture to ensure the basic communication quality [15]. In the highly-dynamic network scenario, UAV nodes $[k_0, k_2, k_4, k_6]$ are in the hover state to form the backbone nodes, while other nodes keep moving randomly. In a lowly-dynamic network scenario, all UAV nodes are in the hover state. However, the network still has an unstable topology structure between different training episodes due to the random deployment process of nodes k_8 and k_9. We also set three baseline algorithms to evaluate the relative performance of our proposed dynamic routing algorithm: the PG-based routing algorithm, the BackPressure routing algorithm (decides the routing policy based on congestion gradients) [16], and the Random routing algorithm.

B. Convergence Performance

In this section, we evaluate the convergence performance of our algorithm. Fig. 3 shows the average reward in different dynamic network environments with different algorithms. Both the PG-based algorithm and our proposed algorithm can achieve convergence, while the former performs the worse. This is most likely caused by the purely distributed framework consisting of each single PG agent. This structure will cause each agent to focus only on maximizing their local reward instead of the global reward, forming a selfish routing policy. However, our proposed algorithm performs better because the



(a) lowly dynamic (b) highly dynamic

Fig. 3. Average Reward

centralized training distributed execution framework enhanced the cooperative routing ability among nodes.

Besides, the average reward of the PG-based routing algorithm and our proposed algorithm increase sharply before around 5000 episodes while tending to be flat after that. This is caused by the ϵ factor. Once ϵ reaches the maximum, the algorithm will become greedy and reduce the exploration of the environment. Also, we can find that the fluctuating range of those two algorithms' learning curves becomes higher as the dynamic level of our network increases. However, the BackPressure routing and Random routing method seem to be more stable, exposing their poor network state awareness ability. Nonetheless, the PG-based routing algorithm and our proposed intelligent routing algorithm can adjust their policy quickly once the network environment changes. Finally, our method performs the best among other algorithms, which means it can form a robust routing method in such a dynamic network environment.

C. Performance Analysis

To evaluate our algorithm performance, we will test the network quality from various indicators (e.g., survival time, delivery rate, and throughput) in this section.

1) Average Survival Time Analysis: First of all, we analyze the average survival time with different networks dynamic. As shown in Fig. 4, the PG-based routing method cannot relieve the network congestion effectively. It even performs worse than the Random routing method in the highly-dynamic network environment. In contrast, our algorithm postpones the congestion occurrence time by 47% of Random algorithm, 13% of BackPressure algorithm, and 29% of PG algorithm. This demonstrates that our proposed algorithm can effectively prolong the network lifetime. However, we notice that the BackPressure method presents better performance than ours in the highly-dynamic network environment. This is probably because the BackPressure method only considers the congestion gradient as the routing decision basis, it built its survival time advantages at the expense of the transmission delay.

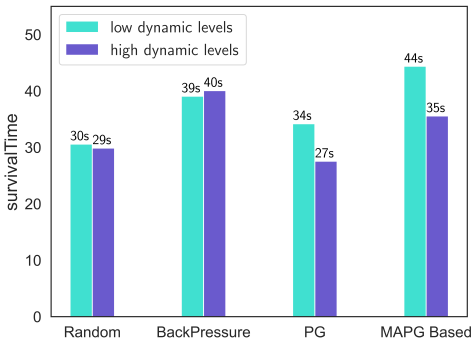


Fig. 4. Average Survival Time

2) Average Delivery Rate Analysis: Secondly, as shown in Fig. 5, we evaluate the average delivery rate. It shows that the average transmission success rate of our algorithm is greater than other baseline algorithms. The PG-based routing algorithm, on the other hand, has a similar delivery effect with the BackPressure method and Random method. Although the BackPressure algorithm has a better congestion control ability, it shows a lower packet delivery capability. It indicates that this method pays too much attention to the queue balancing problem among UAV nodes without considering how to transmit packets effectively. It is worth mentioning that the destination node cannot receive many packets because of a heavy network load environment. Hence, a packet delivery rate of about 20% maintained by our proposed algorithm is relatively acceptable under a highly dynamic network environment.

3) Average Throughput Analysis: Finally, Fig. 6 demonstrates the throughput performance of our proposed algorithm. Generally, the throughput is defined as the rate of successful message delivery over the whole network and a higher network throughput means a more robust packet processing ability. Similar to the packet delivery performance in Fig. 5, our

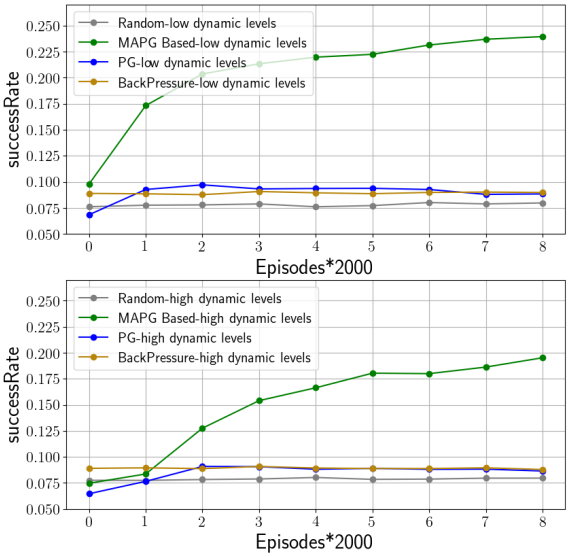


Fig. 5. Average Delivery Rate

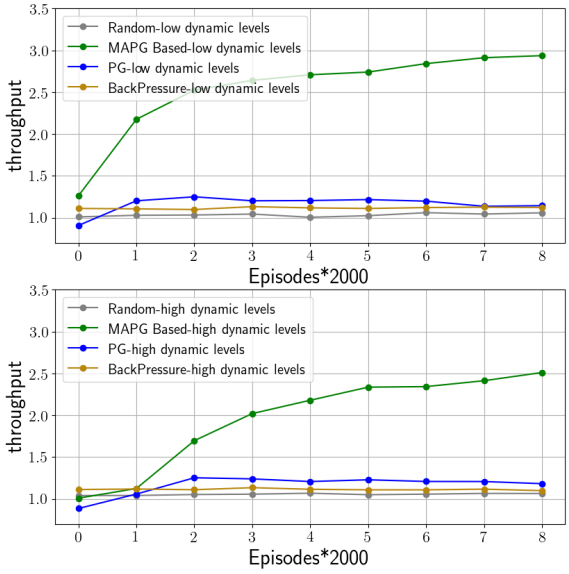


Fig. 6. Average Throughput

proposed routing algorithm performs the best among all the other algorithms. However, the throughput capacity of the PG-based routing algorithm and BackPressure method only shows a weak advantage compared with the Random method.

From the above analysis results, we conclude the main advantage of our proposed algorithm as follows. By maintaining the centralized training and decentralized execution framework together with a counterfactual baseline scheme, our algorithm greatly enhances the cooperative among the UAV nodes. Overall, we find that it can effectively balance the network congestion and packet delivery while maintaining a high throughput ability. Finally, our proposed algorithm emerges with a swarm-intelligent capability and can converge into a robust routing policy within a dynamic network environment.

V. Conclusion

In this paper, we focus on the dynamic routing problem in a UAV swarm network environment. We have proposed a multi-agent policy gradients-based UAV routing algorithm, which uses a centralized critic to evaluate the contribution of each decentralized actor and adjust their policy updating. Moreover, we have introduced a counterfactual baseline scheme in our algorithm to encourage convergence speed. Finally, extensive simulations have been conducted to evaluate the performance of our proposed algorithm. In the future, we will evaluate the performance and the limitations of our proposed dynamic routing method in a real UAV swarm network.

References

[1] R. Ch, G. Srivastava, T. R. Gadekallu, P. K. R. Maddikunta, and S. Bhattacharya, "Security and privacy of uav data using blockchain technology," *Journal of Information Security and Applications*, vol. 55, p. 102670, 2020.

[2] J. Wang, C. Jin, Q. Tang, N. Xiong, and G. Srivastava, "Intelligent ubiquitous network accessibility for wireless-powered mec in uav-assisted b5g," *IEEE Transactions on Network Science and Engineering*, 2020.

[3] X. Pang, M. Liu, N. Zhao, Y. Chen, Y. Li, and F. R. Yu, "Secrecy analysis of uav-based mmwave relaying networks," *IEEE Transactions on Wireless Communications*, 2021.

[4] D. Zhai, H. Li, X. Tang, R. Zhang, Z. Ding, and F. R. Yu, "Height optimization and resource allocation for noma enhanced uav-aided relay networks," *IEEE Transactions on Communications*, 2020.

[5] M. Khaledi, A. Rovira-Sugranes, F. Afghah, and A. Razi, "On greedy routing in dynamic uav networks," in *2018 IEEE International Conference on Sensing, Communication and Networking (SECON Workshops)*, pp. 1–5, 2018.

[6] S. Wen, C. Huang, X. Chen, J. Ma, N. Xiong, and Z. Li, "Energy-efficient and delay-aware distributed routing with cooperative transmission for internet of things," *Journal of Parallel and Distributed Computing*, vol. 118, pp. 46–56, 2018.

[7] A. Coelho, E. N. Almeida, P. Silva, J. Ruela, R. Campos, and M. Ricardo, "Redefine: Centralized routing for high-capacity multi-hop flying networks," in *2018 14th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, pp. 75–82, 2018.

[8] A. Detti, C. Pisa, S. Salsano, and N. Blefari-Melazzi, "Wireless mesh software defined networks (wmsdn)," in *2013 IEEE 9th international conference on wireless and mobile computing, networking and communications (WiMob)*, pp. 89–95. IEEE, 2013.

[9] P. K. Sharma and D. I. Kim, "Coverage probability of 3-d mobile uav networks," *Wireless Communications Letters, IEEE*, vol. 8, no. 1, pp. 97–100, 2019.

[10] R. R. Roy, *Handbook of Mobile Ad Hoc Networks for Mobility Models*. Handbook of Mobile Ad Hoc Networks for Mobility Models, 2011.

[11] M. Li, F. R. Yu, P. Si, R. Yang, Z. Wang, and Y. Zhang, "Uav-assisted data transmission in blockchain-enabled m2m communications with mobile edge computing," *IEEE Network*, vol. 34, no. 6, pp. 242–249, 2020.

[12] M. T. Spaan, "Partially observable markov decision processes," in *Reinforcement Learning*, pp. 387–414. Springer, 2012.

[13] R. S. Sutton, D. A. McAllester, S. P. Singh, Y. Mansour *et al.*, "Policy gradient methods for reinforcement learning with function approximation." in *NIPs*, vol. 99, pp. 1057–1063. Citeseer, 1999.

[14] H. Yao, T. Mai, C. Jiang, L. Kuang, and S. Guo, "Ai routers & network mind: A hybrid machine learning paradigm for packet routing," *IEEE Computational Intelligence Magazine*, vol. 14, no. 4, pp. 21–30, 2019.

[15] M. Gerla and K. Xu, "Minuteman: Forward projection of unmanned agents using the airborne internet," in *Aerospace Conference Proceedings, 2002. IEEE*, 2002.

[16] M. J. Neely, "Stochastic network optimization with application to communication and queueing systems," *Synthesis Lectures on Communication Networks*, vol. 3, no. 1, pp. 1–211, 2010.