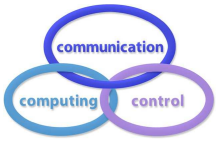
INTERNATIONAL JOURNAL OF COMPUTERS COMMUNICATIONS & CONTROL
Online ISSN 1841-9844, ISSN-L 1841-9836, Volume: 18, Issue: 6, Month: December, Year: 2023
Article Number: 5505, https://doi.org/10.15837/ijccc.2023.6.5505



CCC Publications

A Graph-Based PPO Approach in Multi-UAV Navigation for Communication Coverage

Zhiling Jiang, Yining Chen, Ke Wang, Bowei Yang, Guanghua Song*

Zhiling Jiang
School of Aeronautics and Astronautics
Zhejiang University, China
Hangzhou 310027, China
zhilingjiang@zju.edu.cn

Yining Chen
School of Aeronautics and Astronautics
Zhejiang University, China
Hangzhou 310027, China
ch19930611@zju.edu.cn

Ke Wang
School of Aeronautics and Astronautics
Zhejiang University, China
Hangzhou 310027, China
kwang_0228@zju.edu.cn

Bowei Yang
School of Aeronautics and Astronautics
Zhejiang University, China
Hangzhou 310027, China
boweiy@zju.edu.cn

Guanghua Song*
School of Aeronautics and Astronautics
Zhejiang University, China
Hangzhou 310027, China
ghsong@zju.edu.cn

Abstract

Multi-Agent Reinforcement Learning (MARL) is widely used to solve various problems in real life. In the multi-agent reinforcement learning tasks, there are multiple agents in the environment, the existing Proximal Policy Optimization (PPO) algorithm can be applied to multi-agent reinforcement learning. However, it cannot deal with the communication problem between agents. In order to resolve this issue, we propose a Graph-based PPO algorithm, this approach can solve the communication problem between agents and it can enhance the exploration efficiency of agents in the environment and speed up the learning process. We apply our algorithms to the task of multi-UAV navigation for communication coverage to verify the functionality and performance of our proposed algorithms.

Keywords: UAV Swarm Intelligence, Communication Coverage, Graph Learning, Multi-Agent Reinforcement Learning.

1 Introduction

With the development of modern technology, various productive activities require the assistance of the network. However, in certain unexpected situations, the infrastructure network systems may arise

some problems, for instance, network congestion issues may occur in areas with excessive population density during large event activities. Additionally, infrastructure failures caused by natural disasters could result in regional communications paralysis. In these situations, providing a temporary and rapid communication coverage service is a very important and real need. Unmanned aerial vehicles (UAVs), a vehicle with strong maneuverability and agile deployment capabilities, have become an important means to satisfy such needs [13] by carrying aerial devices to provide temporary communication capabilities for ground targets. In the UAV communication coverage tasks, each UAV is loaded with communication equipment [4]. An area of communication coverage is accomplished by a group of UAVs going together. The UAVs need to cooperate with each other to fully utilize their coverage capabilities and cover as many ground targets as possible [17]. If each aerial communication UAV node is considered as an intelligent agent node, it becomes a problem of multi-agent cooperative control [2][3], as shown in Figure 1.

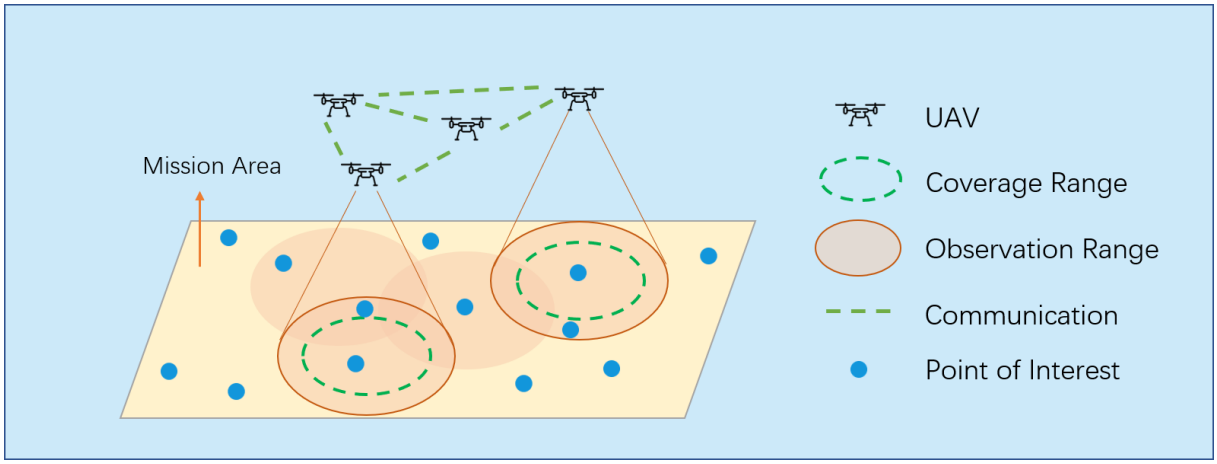


Figure 1: The Task of Multi-UAV Navigation for Communication Coverage

This paper aims to use reinforcement learning methods [12] to address such problems, the multi-agent algorithm can learn a good control scheduling strategy by interacting with the environment. However, the traditional multi-agent reinforcement learning cannot deal with the challenges of drone-to-drone communication, each agent can only perform its tasks lonely without the help of the companions. In our work, we aim to combine the Graph Aggregator with the multi-agent reinforcement learning algorithm PPO and enable the agents to communicate with each other. In this way, the intelligent agent can not only consider its own information but also the situation of neighboring agents. A better cooperative relationship can be formed between agents and their neighbors by using communication to exchange content in the observation with their neighbors. We combine the GCN graph network proposed by Thomas Kipf [10] and the GAT graph network proposed by Shaked Brody [1] with the PPO algorithm [19] proposed by John Schulman that is commonly used in modern industry in a smart way to solve the multi-agent cooperative tasks. We compare our algorithm with the regular algorithm to demonstrate the effectiveness and performance of our algorithm in solving the problem of multi-UAV navigation for communication coverage.

The main contributions of this paper are as follows:

1) We propose a novel PPO algorithm which combines graph aggregator and PPO, named Graph-based PPO, that can handle graph topology information, and we apply the GCN and GAT graph aggregation modules to Graph-based PPO.

2) We use our method(Graph-based PPO) to solve the problem of multi-UAV navigation for communication coverage.

3) We do a series of experiments to compare the performance of the Graph-based PPO algorithm with the regular PPO algorithm in solving the problem of multi-UAV navigation for communication coverage.

The remaining parts of this paper will be organized in the following order: Section 2 presents an introduction to multi-agent reinforcement learning and provides an overview of policy-based re-

inforcement learning algorithms related to experiments. Section 3 describes the task of multi-UAV navigation for communication coverage, details the state space, action space of UAVs, and evaluation indicators of mission performance. After that, we detail the description of Graph-based PPO algorithm for multi-agent reinforcement learning. We validate the effectiveness of our proposed algorithm through experiments in Section 4. Finally, we summarize our work and provide some future directions in Section 5.

2 Background

Reinforcement learning has been widely used in solving intelligent transportation [7][21], UAV task scheduling [16], power grid distribution and voltage regulation [22], and other fields. In reinforcement learning, we hope to learn a strategy to solve our tasks. We can classify reinforcement learning algorithms into two training methods, on-policy and off-policy [14]. The on-policy method means the learning agent has the same network parameters as the agent interacting with the environment. The off-policy method means the learning agent does not need to keep consistent with the interacting agent. Some traditional algorithms, Q-learning [23] and DQN [11], are off-policy methods. However, the on-policy method is more popular in the industry, because it allows for online learning without loss of performance. Online learning allows the model to continuously interact with the environment to adapt to changes in the environment, which is important in applications. PPO is an online learning algorithm based on policy gradient, which has good convergence performance and high stability to deal with high-dimensional state space and action space. PPO is a very powerful algorithm that has a prominent and important place in the field of reinforcement learning. Our work is to use PPO as the base model, enhancing and improving the performance of PPO.

2.1 MAPPO

As a policy gradient algorithm, PPO follows the general formula of policy gradient to update the parameters of the network.

$$\nabla \bar{R}_\theta = E_{\tau \sim p_\theta(\tau)}[R(\tau) \nabla \log p_\theta(\tau)] \tag{1}$$

The formula calculates the expectation of the sampled trajectory τ under the current policy π_θ and performs the parameter update of the network. After the update, the trajectory τ and the current policy π_θ are no longer aligned, and the formula is no longer applicable. Therefore, the policy needs to be re-sampled to prepare for the next update. The subsequent sampled trajectory data can only update the parameters once. This results in very low sampling efficiency. To improve efficiency, PPO utilizes an important sampling technique to enhance the sample efficiency of policy gradient updates. Important sampling technique utilizes the trajectories τ sampled from the old policy $\pi_{\theta'}$ to update the current new policy π_θ. By using a single trajectory data, the policy π_θ is updated multiple times, performing multiple gradient ascent steps. However, this approach typically introduces bias. To address this issue, importance sampling introduces the concept of importance weights to adjust for bias. PPO algorithm incorporates the important weight as a crucial weighting term in the network parameter updates to correct the difference between the distributions of $\pi_{\theta'}$ and π_θ. Therefore, the updated equation for policy gradient becomes

$$\nabla \bar{R}_\theta = E_{\tau \sim p_{\theta'}(\tau)} \left[\frac{p_\theta(\tau)}{p_{\theta'}(\tau)} R(\tau) \nabla \log p_\theta(\tau) \right] \tag{2}$$

In practice, we often can not collect the full trajectory τ. We calculate the relative advantage of the current state by subtracting the baseline from the cumulative reward. Therefore, the advantage function $A(s_t, a_t)$ is used to replace the reward $R(\tau)$, as shown in Equation 3.

$$\nabla \bar{R}_\theta = E_{(s_t, a_t) \sim \pi_{\theta'}} \left[\frac{p_\theta(s_t, a_t)}{p_{\theta'}(s_t, a_t)} A^{\theta'}(s_t, a_t) \nabla \log p_\theta(a_t^n \mid s_t^n) \right] \tag{3}$$

Then, using the gradient backpropagation formula $\nabla f(x) = f(x)\nabla \log f(x)$, the optimization function for the parameters θ in the policy network is obtained, which is shown as Equation 4.

$$J^{\theta'}(\theta) = E_{(st,at)\sim\pi_{\theta'}}\left[\frac{p_\theta\left(a_t \mid s_t\right)}{p_{\theta'}\left(a_t \mid s_t\right)} A^{\theta'}\left(s_t, a_t\right)\right] \tag{4}$$

The training progress is following Equation 4. To prevent excessive variation between the new policy and the old policy during the update, PPO performs a clipping operation on the importance weight term during parameter optimization. MAPPO is a reinforcement learning algorithm [26] based on PPO and designed for multi-agent task systems. It also employs the Actor-Critic architecture [5], where the Actor receives local states and the Critic network evaluates the quality of the Actor's output actions. The main difference compared to PPO is in the Critic component. Instead of using local states as inputs, MAPPO utilizes a global state composed of the aggregated local state information from each agent. The global state is used as input to predict the value of the current state. The Actor component still uses local states as inputs. In scenarios with homogeneous agents, each agent shares the same set of Actor network parameters.

2.2 Graph Aggregator

Graph Aggregator is a method to process the data of topology graph information. Graph aggregator considers each agent in the environment as the information source node in the topology graph. And then uses the topology graph connectivity feature between nodes to extract more valuable information to help the policy network make decisions. The common graph networks include GCN [10], GAT [1][20], GraphSAGE [6] and so on.

GCN generates an adjacency matrix and a degree matrix using the topology graph. In an environment with N agents, the adjacency matrix is a directed graph matrix of shape N*N, the elements 0 and 1 are used in the matrix to describe whether there is a connection relationship between the agents, 1 represents a connected relationship and 0 represents no relationship. The degree matrix is a graph matrix of shape N*N, each element on the diagonal represents the degree of the current node. $\vec{h}$ represent the feature vector of a node, the aggregate process of GCN can be shown in this equation

$$\vec{h}^{l+1} = \sigma\left(D^{-1/2}AD^{-1/2}\vec{h}^{l}W^{l}\right) \tag{5}$$

h is the feature vector of a node, and W represents the weights of the network. In GCN, the distribution of node weights is determined by the adjacency matrix and degree matrix.

GAT uses a different calculation method, GAT assigns weight to the edge weights between two nodes by using the feature information between two nodes. The weight calculation can be expressed by this equation below:

$$\alpha_{ij} = \frac{\exp\left(LeakyReLU\left(\overrightarrow{a}^{T}\left[W\vec{h}_i \| W\vec{h}_j\right]\right)\right)}{\sum_{k\in N_i}\exp\left(LeakyReLU\left(\overrightarrow{a}^{T}\left[W\vec{h}_i \| W\vec{h}_k\right]\right)\right)} \tag{6}$$

where $\overrightarrow{a}$ and W are the parameters of the neural networks. After calculating the edge weights, the following step is similar to the GCN.

Using the below equation to calculate the feature vector after aggregation.

$$\vec{h}^{l+1} = \sigma\left(\alpha_{ij}\vec{h}^{l}W^{l}\right) \tag{7}$$

Graph aggregation is widely used in game tasks [15], recommendation systems, social networks, knowledge graphs [24], biomedical [25] and other fields. Applying graph aggregation to off-policy learning has been studied by many previous researchers [8][18]. In this paper, we apply the graph aggregation to on-policy learning to solve the UAVs scheduling problem. Graph aggregator is used to aggregate the information in multi-UAV communication network and help PPO to handle the UAV swarm topology relationships. In the following section, we detail the description of our algorithm Graph-based PPO.

3 Proposed Method

We proposed a PPO algorithm with a graph aggregation (Graph-based PPO), it can be used to solve the task of multi-UAV navigation for communication coverage. In this section, we first discuss the scenario of multi-UAV navigation and the task for communication coverage, and then we introduce the design of the Graph-based PPO algorithm. The design of the Graph-based PPO algorithm is divided into three parts. The first part presents the overall structure of the algorithm for multi-agent control. The second part describes the graph aggregation employed in the algorithm. The third part explains the design and significance of the Actor and Critic components.

3.1 Problem Statement

3.1.1 Multi-UAV Navigation Scenario

We use a scenario of multi-UAV navigation for communication coverage, as shown in Figure 1. A group of UAVs need to cover targets on the ground as extensively as possible in order to provide the communication services. We consider each target as a Point of Interest(PoI). The world is a 2D map, we randomly generate 120 PoI in the map and our UAVs are required to cover these PoI as much as possible. Each UAV has a coverage range, and only the PoI within this range can receive communication services from the UAVs. At the same time, each UAV has an observation range that allows them to detect the PoI on the ground. Additionally, UAVs have limited communication capabilities among themselves, which allows them to communicate and exchange observations with neighbor UAVs for better cooperation.

3.1.2 Observation Space and Action Space

Each UAV has its own observation space and action space. The contents in the observation space are used as inputs to the policy network for making UAV flight action decisions. The observation space consists of several components, including the UAV's identifier, binary encoding of its x-axis position and y-axis position, velocity in the positive x-axis direction and y-axis direction. The UAV perceives the content in its field of view, which includes the positions of PoI within the observation range and neighboring UAVs in the neighborhood. These elements are concatenated to generate the observation space used for action decision-making. At each time step, in addition to outputting an observation status for each UAV, the scenario also outputs an adjacency matrix. The adjacency matrix is an N*N matrix that records the communication connectivity between the n UAVs. Each UAV has a communication range, allowing UAVs within this range to establish communication and exchange information. The corresponding positions in the adjacency matrix are set to 1 to indicate the connectivity between UAVs. Each UAV has 17 action options, corresponding to acceleration in different directions. This includes acceleration options for east, west, south, north, southeast, northeast, southwest, and northwest directions. Each direction has two acceleration options, and there is also an option for not accelerating.

3.1.3 Evaluation Metrics

The quality of UAVs' performance is decided by the coverage of Points of Interest (PoI) on the ground and the energy consumption of the UAVs. The coverage performance of UAVs is determined by both individual rewards and group rewards. The individual coverage reward is determined by the number of PoIs that the UAV covers independently. When a UAV does not cover any PoI, $individual\_reward = -1$, and in other cases $individual\_reward = n$, n is the number of PoI. The group reward is determined by the average number of PoIs covered by the remaining UAVs in the group, excluding the number of PoIs independently covered by the current UAV, $group\_reward = 0.1 * (group\_covered - individual\_covered)$. The individual reward and group reward are summed together to evaluate the final performance, $reward = individual\_reward + group\_reward$. The reward is utilized to optimize the policy network of the UAVs. To better evaluate the performance of UAVs, we also take into account the energy consumption of the UAVs' batteries, The reward is

inversely proportional to the rate of energy consumption, the faster the battery is depleted, the worse the performance of the strategy, $reward = reward/energy\_consumption$.

3.2 Graph-based PPO

3.2.1 Structure

We are dealing with a cooperative control problem involving multiple UAVs, where the actions of each UAV can influence the decisions of other UAVs. The environment is uncertain and the randomness makes the training more challenging. Since each UAV has the same objective which is to cover PoI as extensively as possible, we use a centralized training distributed execution (CTDE) reinforcement learning [8].

During the training process, the reinforcement learning architecture receives global observation information from N agents in the environment. The critic network takes input from the observations of these N agents and outputs a value estimation of the current state, which helps to optimize the network parameters of the Actor. During the training process, the critic and actor networks work together. The actor network only receives the local state of the agent for decision-making, and the critic network receives global states from N agents to provide a more accurate assessment of the current state and offer better guidance to the Actor network. This training process is a centralized training and it receives information from all agents.

After training is completed, the critic network is no longer used during execution. In the execution phase, only the actor network is utilized, and it allows the agent to make decisions based on the UAV's own observation. This is a distributed execution process where the decision-making process does not interfere with each other. Since all UAVs share the same task and have identical observation and action spaces, multiple agents can utilize a single actor network and share the parameters of the network in the training process. During execution, the actor network is distributed to each UAV. This approach provides the structure with great flexibility and enables it to adapt to scenarios with varying numbers of UAVs. This structure depicts in Figure 2

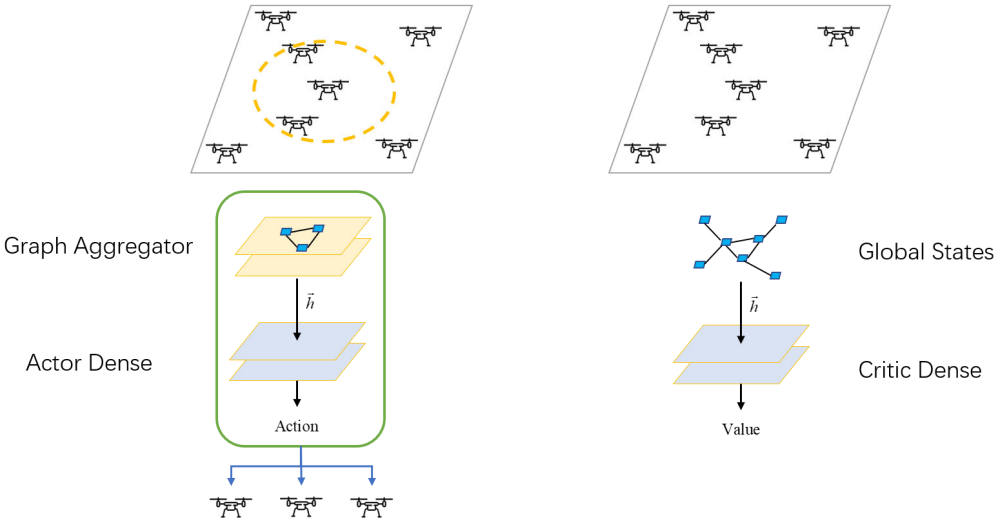


Figure 2: The Framework of Our Graph-based PPO

3.2.2 Graph Module

In our scenario, the UAV has the ability to communicate with its companions, and the graph module can make full use of the information of the neighbor companions to help the action decision-making. The graph module can address the neighbor topology by extracting features from the UAVs' observation information and utilizing graph topology to aggregate neighboring information for decision-making.

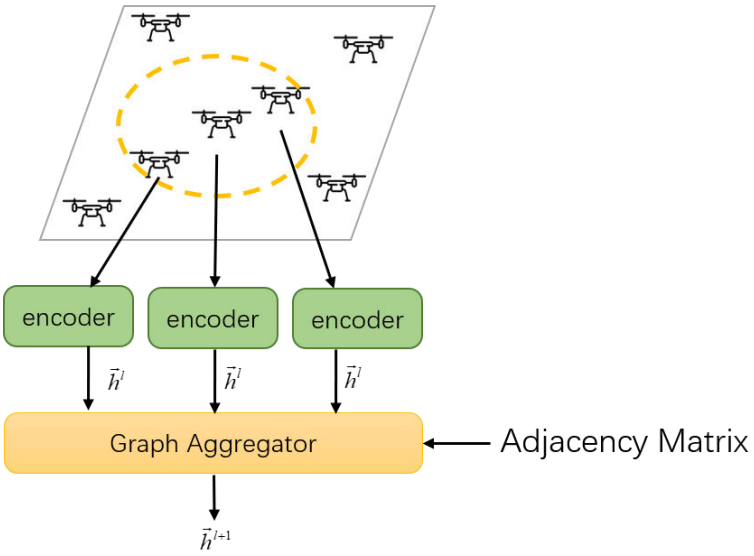


Figure 3: The Process of Neighbor Information Aggregate

The graph module receives two inputs, as shown in Figure 3: the observation information of the UAVs and the topology graph representing the neighbor relationships between the UAVs. The module first encodes the observation information of the UAVs to reduce the dimensionality of the data and extract features denoted as $\vec{h}$. Then, the module selectively aggregates information from neighboring UAVs based on the topology graph representing the relationships between the UAVs in the group. During the aggregation process, the module can also consider the similarity between the feature vectors of the own node and its neighboring nodes. After the aggregation is complete, a new feature vector $\vec{h}'$ is outputted, which includes both the own node's content and the information from the neighbors. This feature vector serves as the new node representation for decision-making.

3.2.3 Graph Fusion

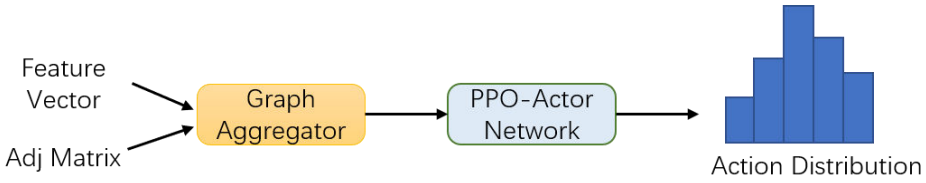


Figure 4: Combined Actor Branch with Graph Aggregator

We integrate the graph into the PPO algorithm, which consists of both Actor and Critic neural networks working together to accomplish the task. The critic network directly receives the global observation information of the UAVs, so there is no need for aggregation. The actor network in our Graph-based PPO fully utilizes the observation data of neighbor agents and the connectivity in the topology graph between UAVs. The actor network takes the graph G representing the relationship of UAVs as input, and then it outputs a probability distribution over a set of 17 actions. The sampler randomly selects an action from these 17 actions based on their respective probabilities for the UAV to execute, as shown in Figure 4.

The loss for the Actor network is calculated as follows:

$$Loss_{actor}(\theta) = E_{\tau}\left[\min\left(\frac{p_{\theta}\left(a_t \mid s_t\right)}{p_{\theta'}\left(a_t \mid s_t\right)} A^{\theta'}\left(s_t, a_t\right),\quad clip\left(\frac{p_{\theta}\left(a_t \mid s_t\right)}{p_{\theta'}\left(a_t \mid s_t\right)}, 1-\varepsilon, 1+\varepsilon\right) A^{\theta'}\left(s_t, a_t\right)\right)\right] \tag{8}$$

Where

$$A\left(s_t, a_t\right) = -Value\left(s_t, a_t\right) + Return\left(s_t\right) \tag{9}$$

$$Return\,(s_t) = \sum_{s_i=s_t}^{\infty} reward(s_i) \tag{10}$$

$$Value(s_t, a_t) = Critic(s_t, a_t) \tag{11}$$

The loss for the Critic network is calculated as follows:

$$Loss_{critic}(\theta) = E_\tau \left[(-Critic\,(s_t, a_t) + Return\,(s_t))^2 \right] \tag{12}$$

Algorithm 1 describes the process of the Graph-based PPO :

Algorithm 1 Graph-Based PPO

Initialize weights for aggregator, actor and critic networks
for episode $epi = 1$ to T **do**
  For each agent i, set an empty trajectory $\tau = []$
  Reset the environment and initialize the observation obs for each agent i
  for time steps $t = 1$ to episode length **do**
    for agent $i = 1$ to N **do**
      $h_i \leftarrow AGGREGATOR(obs, adj)$
      Select action $action_i$ in hidden state h_i via $ACTOR$ $\pi_i(action_i \mid h_i)$
      Execute agent's action $action_i$ to the environment
    end for
    Collect the action, reward, observation, and relationships between agents
    Push the experience into trajectory $\tau += [action, reward, obs', adj]$
    $obs = obs'$
  end for
  Computes $advantage$ and $return$ using trajectory τ
  for update epoch $epoch = 1$ to update times **do**
    Update weights for actor network
    Update weights for aggregator network
    Update weights for critic network
  end for
end for

4 Experiment

In this section, we present the design details of the experiment. We will introduce the experimental environment platform for the multi-UAV navigation for communication coverage. Finally, we present the usability and performance of graph-based PPO in the UAV tasks, explain the training process of the experiment and analyze the results of the experiment.

4.1 Experimental Settings

We deploy our model on a Ubuntu 18.04.5 server with 1 NVIDIA GeForce GTX 1080 Ti GPU and 1 Intel i7-7700K CPU @ 4.20GHz CPU and conduct experiments, the agents implement in Python 3.7.10, Pytorch 1.12.1, CUDA 11.4.

In the experiment, we use the proposed Graph-based PPO as the decision-making agent for UAVs executing tasks. In the scenario of UAV navigation for communication coverage, the UAVs are required to cover PoIs on the ground as extensively as possible. UAVs can establish communication and exchange information with each other. We consider the communication relationships between UAVs as a graph. Graph-based PPO can utilize not only the feature vector of the agent but also the topological information of the graph representing the relationships between UAVs to help make decisions. We explore various graph aggregation techniques to process the graph information and compare their effects on the performance of PPO in our multi-UAV navigation missions.

4.2 Simulation Scenarios

In this section, we present the design details of the UAV scenario. There are 20 UAVs in the map and each UAV has a coverage range of 10 units, an observation range of 13 units and a communication range of 18 units between UAVs. The map is a rectangular area of 200*200 units. In each episode, the scenario randomly generates 20 UAVs and 120 PoIs at different positions. The goal of the 20 UAVs is to utilize their coverage range to cover as many PoIs as possible. Each episode consists of 100-time steps. We use 10 as the random seed to generate various random numbers in the training process. At the beginning of each episode, we randomly generate different scenario maps.

4.3 Experience Results and Discussion

We verify the performance of Graph-based PPO algorithm proposed above, test its effectiveness in controlling UAVs on the task of multi-UAV navigation for communication coverage. In the experiments, we use different graph aggregation methods and use the PPO algorithm without graph aggregation as the baseline. The performance of the algorithms is evaluated by examining their control over the UAV swarm. The evaluation indexes include the convergence of the reward curve, the coverage of PoIs, the energy consumption of the UAV swarm and a comprehensive metric that combines the PoI coverage and energy efficiency.

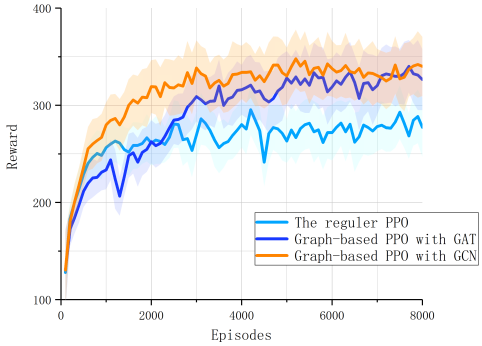


Figure 5: Episodic Reward Curves

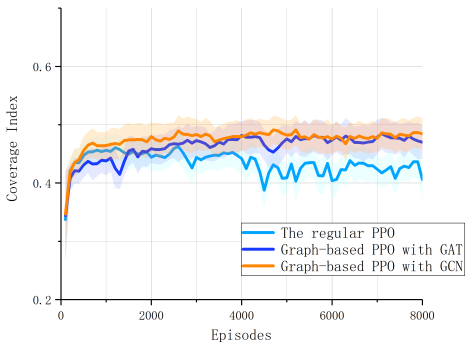


Figure 6: Coverage Index Curves

By analyzing the reward curve in Figure 5, we can find that the graph-based PPO has better convergence performance. After 1000 episodes of training, the final convergence results of the Graph-based PPO with GCN and GAT are generally consistent. But the regular PPO without graph aggregation does not achieve such good results. In terms of convergence speed, Graph-based PPO with GCN performs the best. The graph-based PPO with GAT has a lower convergence speed than regular PPO in the first 2000 episodes, but it has a better performance after 2000 episodes and gets a score that is equally good as GCN finally.

The coverage index in Figure 6 is used to evaluate the coverage ratio of PoIs on the ground by the UAV swarm. When all PoIs are covered, the index is 1. And it is 0 when nothing is covered. During the experiments, we find that the PPO is very weak at the beginning of the training. The coverage index is only 0.2. After 100 episodes, the coverage index achieves 0.3 and they achieve a score of 0.4 after 200 episodes. In the first 200 episodes, all algorithms showed significant improvements in performance. However, after 500 episodes, all algorithms reached a similar level of improvement, entering a phase of slow growth. The graph-based PPO with GCN consistently maintains a lead, but the difference with regular PPO is not significant, while the graph-based PPO with GAT lags behind relatively. After 2000 episodes, the graph-based PPO with GAT starts to surpass regular PPO, and by the time 6000 episodes are reached, the graph-based PPO with GAT and the graph-based PPO with GCN have maintained a comparable level of performance. However, the regular PPO performs poorly. Although it can achieve good performance relatively quickly, it exhibits poor stability in convergence after 2000 episodes.

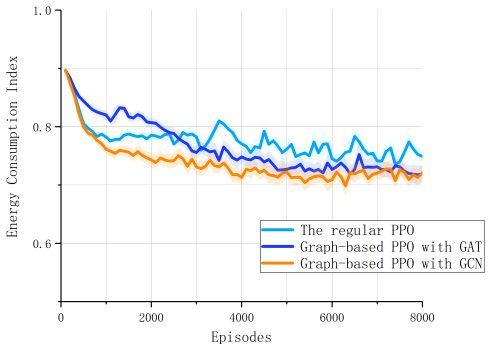


Figure 7: Energy Consumption Index Curves

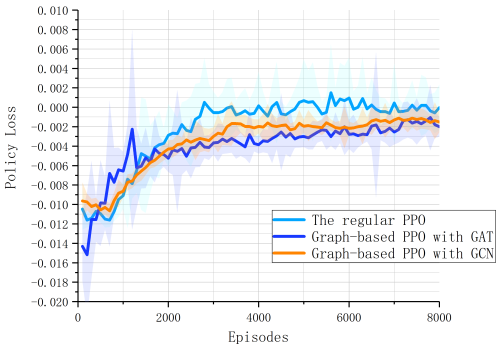


Figure 8: Policy Loss Curves

The energy consumption index shown in Figure 7 is a metric used to measure the energy consumption of UAV flight. The index is 1 when UAVs are flying at full speed and 0.5 when the UAVs are hovering. it calculates the average consumption in 100 steps. We can observe that as the training progresses, the algorithm continuously searches for a more energy-efficient flight mode and the index is continuously decreasing. In the first 500 episodes, the algorithm focuses on finding a cooperative strategy that improves PoI coverage while considering energy consumption during flight. After training for 500 episodes, the efficiency of PoI coverage reaches a stable value. The algorithm then primarily optimizes the energy consumption during UAV flight. As the training progresses, three algorithms have found more optimal flight strategies that save energy, Overall, the Graph-based PPO algorithms have better performance than the regular PPO.

The policy loss curve reflects the effort of optimization of a policy. As shown in Figure 8, we can observe that in the first 2000 episodes, there is significant optimization of the policy network. After 2000 episodes, each training iteration only performs slight adjustments to the network, resulting in the loss value approaching 0. The policy network reaches a stable state.

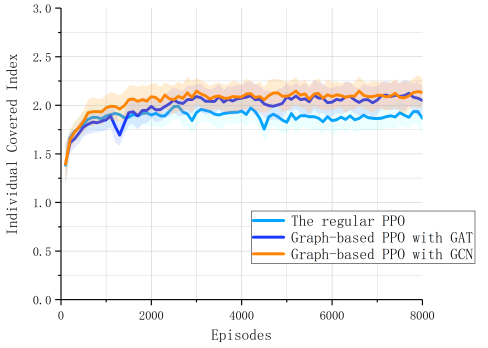


Figure 9: Indivudual Covered PoI per Episode

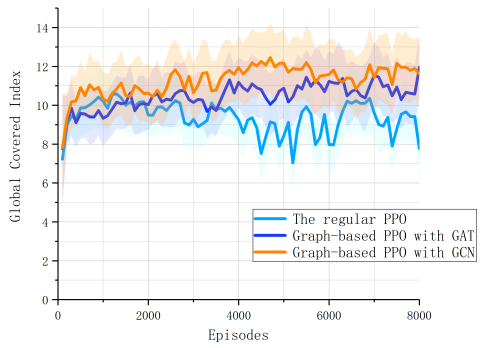


Figure 10: Group Covered PoI per Episode

Individual covered PoI refers to the average number of PoIs covered independently by each UAV in the map. As the training progresses, the UAVs can hardly cover the PoIs at the beginning. But eventually, each UAV is able to independently cover 2 PoIs. The training converges after 1000 episodes.

Group covered index refers to the average number of PoIs covered by the UAV group in the map. Several communication-linked UAVs form a UAV group. When the training reaches 1000 episodes, the group coverage reaches its peak, with an average of 10 PoIs covered per UAV. This index is heavily influenced by the map of the distribution of PoIs in the scenario, but it generally reaches a coverage level of 7 or more PoIs.

After analyzing the performance of the algorithm, let's now examine the generalization capability of the trained models. We will investigate how the performance of the models is affected when the

number of UAVs varies in the scenario, as shown in Figures 11, 12, 13 and 14. We will vary the size of the UAV swarm, setting different scenarios with 5, 10, 15, 20, 25, and 30 UAVs. We conducted 100 random experiments for each scenario and calculated the average values for analysis.

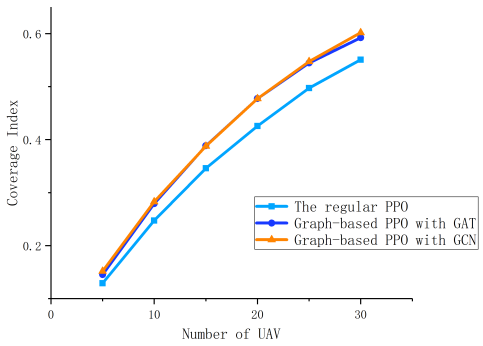


Figure 11: Coverage Index in the environment with different numbers of UAV

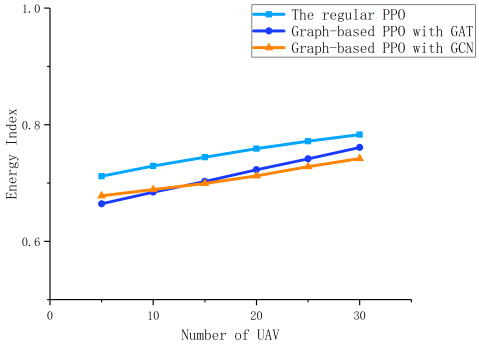


Figure 12: Energy Index in the environment with different numbers of UAV

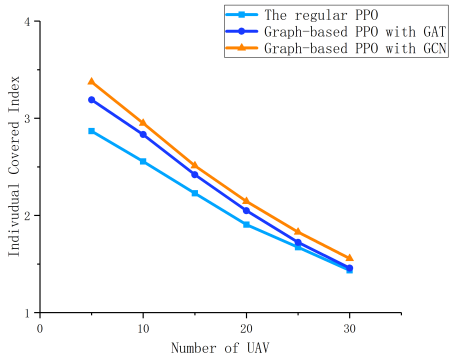


Figure 13: Individual covered index in the environment with different numbers of UAV

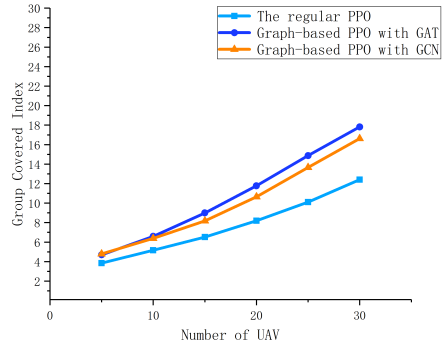


Figure 14: Group Covered index in the environment with different numbers of UAV

Through our experiments, we have discovered that the trained policies can be scaled to different sizes of UAV scenarios in our framework, demonstrating the effectiveness of our algorithm in different UAV cooperative tasks. We find that the performance of Graph-based PPO is better than regular PPO. In terms of the covered index, Graph-based PPO performs better than regular PPO, with a more significant performance improvement as the number of UAVs increases. Regarding the energy index, we observed that Graph-based PPO has lower energy consumption. In terms of the individual covered index, the Graph-based PPO outperforms the regular PPO. This index decreases with an increasing number of UAVs because they are more likely to collaborate as the number of UAVs increases. This trend is also reflected in Figure 14, where a higher number of UAVs leads to a greater number of PoI covered by group collaboration.

5 Conclusions and Future Work

In this paper, we designed a novel PPO algorithm, Graph-based PPO. Graph-based PPO can address the problem of multi-node information communication and decision-making. With the help of the graph topological matrix, we can enhance the learning of reinforcement learning decision-making policies. Additionally, Graph-based PPO can address the communication issues among multiple agents that regular PPO struggles to solve. We conducted experiments on the performance of the algorithm in the scenario of multi-UAV navigation for communication coverage and compare the performance

difference between Graph-based PPO and regular PPO. We experimented with different modes of graph aggregation in the Graph-based PPO, GAT and GCN. The experimental results show that although both Graph-based PPO and regular PPO can solve the problem. But with the limited training episodes, Graph-based PPO can achieve better performance. Moreover, the convergence stability and convergence speed have been improved to a certain extent. In the future, we would develop heterogeneous [9] Graph-based PPO to control different types of UAVs.

Funding

This work was supported by the National Natural Science Foundation of China under Grant No. 62236007.

Author contributions

The authors contributed equally to this work.

Conflict of interest

The authors declare no conflict of interest.

References

[1] Brody, Shaked.; Alon, Uri.; Yahav, Eran. (2021). How attentive are graph attention networks?, *arXiv preprint arXiv:2105.14491*, 2021.

[2] Canese, Lorenzo.; Cardarilli, Gian Carlo.; Di Nunzio, Luca.; Fazzolari, Rocco.; Giardino, Daniele.; Re, Marco.; Spanò, Sergio. (2021). Multi-agent reinforcement learning: A review of challenges and applications, *Applied Sciences*, 11(11), 4948, 2021.

[3] Gronauer, Sven.; Diepold, Klaus. (2022). Multi-agent deep reinforcement learning: a survey, *Artificial Intelligence Review*, 1–49, 2022.

[4] Gupta, Lav.; Jain, Raj.; Vaszkun, Gabor. (2015). Survey of important issues in UAV communication networks, *IEEE communications surveys & tutorials*, 18(2), 2015.

[5] Haarnoja, Tuomas.; Zhou, Aurick.; Abbeel, Pieter.; Levine, Sergey. (2018). Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor, *International conference on machine learning*, 1861–1870, 2018.

[6] Hamilton, Will.; Ying, Zhitao.; Leskovec, Jure. (2017). Inductive representation learning on large graphs, *Advances in neural information processing systems*, 30, 2017.

[7] Hart, Patrick.; Knoll, Alois. (2020). Graph neural networks and reinforcement learning for behavior generation in semantic environments, *2020 IEEE Intelligent Vehicles Symposium (IV)*, 1589–1594, 2020.

[8] Jiang, Jiechuan.; Dun, Chen.; Huang, Tiejun.; Lu, Zongqing. (2018). Graph convolutional reinforcement learning, *arXiv preprint arXiv:1810.09202*, 2018.

[9] Jiang, Zhiling.; Chen, Yining.; Song, Guanghua.; Yang, Bowei.; Jiang, Xiaohong. (2023). Cooperative planning of multi-UAV logistics delivery by multi-graph reinforcement learning, *International Conference on Computer Application and Information Security (ICCAIS 2022)*, 12609, 129–137, 2023.

[10] Kipf, Thomas N.; Welling, Max. (2016). Semi-supervised classification with graph convolutional networks, *arXiv preprint arXiv:1609.02907*, 2016.

[11] Mnih, Volodymyr.; Kavukcuoglu, Koray.; Silver, David.; Graves, Alex.; Antonoglou, Ioannis.; Wierstra, Daan.; Riedmiller, Martin. (2013). Playing atari with deep reinforcement learning, *arXiv preprint arXiv:1312.5602*, 2013.

[12] Mnih, Volodymyr.; Kavukcuoglu, Koray.; Silver, David.; Rusu, Andrei A.; Veness, Joel.; Bellemare, Marc G.; Graves, Alex.; Riedmiller, Martin.; Fidjeland, Andreas K.; Ostrovski, Georg. (2015). Human-level control through deep reinforcement learning, *nature*, 518(7540), 529–533, 2015.

[13] Moradi, Mehrdad.; Sundaresan, Karthikeyan.; Chai, Eugene.; Rangarajan, Sampath.; Mao, Z Morley. (2018). SkyCore: Moving core to the edge for untethered and reliable UAV-based LTE networks, *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking*, 35–49, 2018.

[14] Oroojlooy, Afshin.; Hajinezhad, Davood. (2022). A review of cooperative multi-agent deep reinforcement learning, *Applied Intelligence*, 1–46, 2022.

[15] Pan, Wei.; Liu, Cheng. (2023). A Graph-Based Soft Actor Critic Approach in Multi-Agent Reinforcement Learning, *INTERNATIONAL JOURNAL OF COMPUTERS COMMUNICATIONS & CONTROL*, 18(1), 2023.

[16] Ren, Yixiang.; Ye, Zhenhui.; Song, Guanghua.; Jiang, Xiaohong. (2022). Space-Air-Ground Integrated Mobile Crowdsensing for Partially Observable Data Collection by Multi-Scale Convolutional Graph Reinforcement Learning, *Entropy*, 24(5), 638, 2022.

[17] Ruan, Lang.; Wang, Jinlong.; Chen, Jin.; Xu, Yitao.; Yang, Yang.; Jiang, Han.; Zhang, Yuli.; Xu, Yuhua. (2018). Energy-efficient multi-UAV coverage deployment in UAV networks: A game-theoretic framework, *China Communications*, 15(10), 194–209, 2018.

[18] Ryu, Heechang.; Shin, Hayong.; Park, Jinkyoo. (2020). Multi-agent actor-critic with hierarchical graph attention network, *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05), 7236–7243, 2020.

[19] Schulman, John.; Wolski, Filip.; Dhariwal, Prafulla.; Radford, Alec.; Klimov, Oleg. (2017). Proximal policy optimization algorithms, *arXiv preprint arXiv:1707.06347*, 2017.

[20] Veličković, Petar.; Cucurull, Guillem.; Casanova, Arantxa.; Romero, Adriana.; Lio, Pietro.; Bengio, Yoshua. (2017). Graph attention networks, *arXiv preprint arXiv:1710.10903*, 2017.

[21] Wang, Enshu.; Liu, Bingyi.; Lin, Songrong.; Shen, Feng.; Bao, Tianyu.; Zhang, Jun.; Wang, Jianping.; Sadek, Adel W.; Qiao, Chunming. (2023). Double graph attention actor-critic framework for urban bus-pooling system, *IEEE Transactions on Intelligent Transportation Systems*, 2023.

[22] Wang, Yi.; Qiu, Dawei.; Wang, Yu.; Sun, Mingyang.; Strbac, Goran. (2023). Graph Learning-Based Voltage Regulation in Distribution Networks with Multi-Microgrids, *IEEE Transactions on Power Systems*, 2023.

[23] Watkins, Christopher JCH.; Dayan, Peter. (1992). Q-learning, *Machine learning*, 8, 279–292, 1992.

[24] Xu, Xiaohan.; Zhang, Peng.; He, Yongquan.; Chao, Chengpeng.; Yan, Chaoyang. (2022). Subgraph neighboring relations infomax for inductive link prediction on knowledge graphs, *arXiv preprint arXiv:2208.00850*, 2022.

[25] You, Jiaxuan.; Liu, Bowen.; Ying, Zhitao.; Pande, Vijay.; Leskovec, Jure. (2018). Graph convolutional policy network for goal-directed molecular graph generation, *Advances in neural information processing systems*, 31, 2018.

[26] Yu, Chao.; Velu, Akash.; Vinitsky, Eugene.; Gao, Jiaxuan.; Wang, Yu.; Bayen, Alexandre.; Wu, Yi. (2022). The surprising effectiveness of ppo in cooperative multi-agent games, *Advances in Neural Information Processing Systems*, 35, 24611–24624, 2022.

Cite this paper as:

Jiang, Z.; Chen, Y.; Wang, K.; Yang, B.; Song, G. (2023). A Graph-Based PPO Approach in Multi-UAV Navigation for Communication Coverage, *International Journal of Computers Communications & Control*, 18(6), 5505, 2023.
https://doi.org/10.15837/ijccc.2023.6.5505