Lecture Notes in Electrical Engineering 1206

Jianglong Yu
Yumeng Liu
Qingdong Li *Editors*

Proceedings of 2023 7th Chinese Conference on Swarm Intelligence and Cooperative Control

Swarm Perception and Navigation Technologies

Springer

Lecture Notes in Electrical Engineering 1206

The book series *Lecture Notes in Electrical Engineering* (LNEE) publishes the latest developments in Electrical Engineering—quickly, informally and in high quality. While original research reported in proceedings and monographs has traditionally formed the core of LNEE, we also encourage authors to submit books devoted to supporting student education and professional training in the various fields and applications areas of electrical engineering. The series cover classical and emerging topics concerning:

- Communication Engineering, Information Theory and Networks
- Electronics Engineering and Microelectronics
- Signal, Image and Speech Processing
- Wireless and Mobile Communication
- Circuits and Systems
- Energy Systems, Power Electronics and Electrical Machines
- Electro-optical Engineering
- Instrumentation Engineering
- Avionics Engineering
- Control Systems
- Internet-of-Things and Cybersecurity
- Biomedical Devices, MEMS and NEMS

For general information about this book series, comments or suggestions, please contact leontina.dicecco@springer.com.

To submit a proposal or request further information, please contact the Publishing Editor in your country:

China

Jasmine Dou, Editor (jasmine.dou@springer.com)

India, Japan, Rest of Asia

Swati Meherishi, Editorial Director (Swati.Meherishi@springer.com)

Southeast Asia, Australia, New Zealand

Ramesh Nath Premnath, Editor (ramesh.premnath@springernature.com)

USA, Canada

Michael Luby, Senior Editor (michael.luby@springer.com)

All other Countries

Leontina Di Cecco, Senior Editor (leontina.dicecco@springer.com)

**** This series is indexed by EI Compendex and Scopus databases. ****

Jianglong Yu · Yumeng Liu · Qingdong Li
Editors

Proceedings of 2023 7th Chinese Conference on Swarm Intelligence and Cooperative Control

Swarm Perception and Navigation Technologies

Springer

Editors
Jianglong Yu
School of Automation Science and Electrical Engineering
Beihang University
Beijing, China

Yumeng Liu
Institute of Software, Chinese Academy of Sciences
Beijing, China

Qingdong Li
School of Automation Science and Electrical Engineering
Beihang University
Beijing, China

ISSN 1876-1100 ISSN 1876-1119 (electronic)
Lecture Notes in Electrical Engineering
ISBN 978-981-97-3331-6 ISBN 978-981-97-3332-3 (eBook)
https://doi.org/10.1007/978-981-97-3332-3

This Springer imprint is published by the registered company Springer Nature Singapore Pte Ltd.
The registered company address is: 152 Beach Road, #21-01/04 Gateway East, Singapore 189721, Singapore

Contents

Attention Scheduler Based on Reinforcement Learning for Multi-robot System

Kun Jiang[1], Lingyue Kong[1], and Lu Dong[2](✉)

[1] School of Automation, Southeast University, Nanjing, China
[2] School of Cyber Science and Engineering, Southeast University, Nanjing, China
ldong90@seu.edu.cn

Abstract. In a manufacturing job shop, many machines demand the assistance of auxiliary robots, such as supplement raw materials. In order to balance energy saving and effective scheduling, auxiliary robots have to understand the urgency of tasks and plan a safe and stable working path in the job shop. However, previous works based on exact methods and approximation methods suffer from many realistic constraints, such as complex factory environments and non-deterministic polynomial (NP) characteristics. To address the shortcomings of those works, we propose an attention scheduling (Att-Sched) module for Multi-Agent Deep Deterministic Policy Gradient (MADDPG) framework. Instead of hand-crafted function-based algorithms, we leverage MADDPG to tackle nonlinear and NP characteristics between robots and machines with the job shops. To capture the spatial relationships between robots and learn prioritization dispatching rules respectively, we employ the attention mechanism for distinguishing the urgency of tasks. Through experiments on several simulation environments of job shops, we demonstrate our approach can achieve socially acceptable scheduling and fulfill the demands of machines.

Keywords: Multi-Robot system · reinforcement learning (RL) · attention · task assignment

1 Introduction

With the rapid development of artificial intelligence and information technologies, manufacturing-related technologies have also led to rapid social progresses in recent years. More and more automated guided vehicles (AGVs) and other mobile robots are adopted in job shops to ensure high efficiency and quality requirement in the enterprise. Job shop scheduling problem (JSSP) is a challenging task as it should fulfill the efficient arrangement of the multi-robot system

This work was supported by National Key R&D Program of China (2021ZD0112700), the National Natural Science Foundation of China (62173251), the "Zhishan" Scholars Programs of Southeast University, the Fundamental Research Funds for the Central Universities.

J. Yu et al. (Eds.): CCSICC 2023, LNEE 1206, pp. 1–12, 2024.
https://doi.org/10.1007/978-981-97-3332-3_1

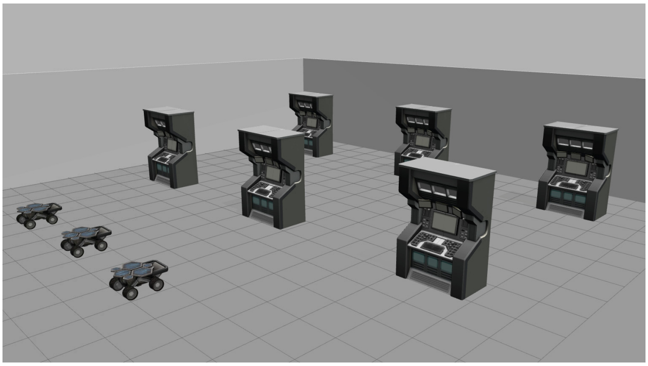


Fig. 1. Schematic diagram of factory scene.

under many realistic constraints of a complex factory environment and non-deterministic polynomial (NP) characteristics [17].

The importance and socio-economic benefits of JSSP attract many researchers to this field. As illustrated in Fig. 1, there are three auxiliary robots and six working machines in the experimental environment. The tasks of the robots are to provide auxiliary work, such as supplementing raw materials and solving machine stoppage, for machines, and every machine can send signals to the control unit to request assistance with emergencies. Mathematics-based methods are divided into exact methods and approximation methods for solving the NP features [29]. However, exact methods, like a branch and bound method [18], are only applicable to small-scale JSSP for its complex mathematical restriction. Approximation methods, including prioritization dispatching rules, can be applied to large-scale JSSP, but it deviates from practical situations occasionally. Swarm intelligence algorithms, including ant colony algorithm and genetic algorithm, have a great performance in tackling NP characteristics and have been adopted to JSSP. Nevertheless, the strong randomicity of swarm intelligence algorithms makes it inadequate for being stably applied in large-scale job shops.

Different from mathematics-based methods, reinforcement learning (RL) can collect data interactively. In RL, agents take actions by the reward and punishment obtained from the environment that is sparse, noisy, and delayed sometimes, without indicating how to accomplish tasks, which makes it can handle non-deterministic polynomial characteristics easily. In this paper, we leverage the characteristic of RL and propose our frame based on Multi-Agent Deep Deterministic Policy Gradient (MADDPG).

To learn the spatial relationships between robots and arrange various prioritization for the tasks, we introduce an attention mechanism to our MADDPG and propose our attention scheduling (Att-Sched) module. The attention mechanism can learn spatial relationships and select important frames for word prediction and has made progress in computer vision [19] and natural language processing [20].

We propose our algorithm under the framework of MADDPG to schedule the multi-robot tasks. To capture spatial interaction between robots with machines and arrange various importance to the tasks, we introduce an attention mechanism to our algorithm and propose an attention scheduling module (Att-Sched). Through experiments in several simulation environments of the job shop scenes, our algorithm is evaluated to balance the economically-acceptable schedule and makespan. There are two fundamental contributions of this paper as follows: *(i)* we propose to modal JSSP by MADDPG to learn long-term rewards and schedule energy-savingly for many nonlinear cases without hand-crafted functions. *(ii)* attention mechanism is introduced to arrange different importance to working machines and distinguish the prioritization of tasks, which is socially acceptable for job shops.

2 Preliminary Work

The multi-robot system has many excellent features, including efficiency and flexibility, and has the capability of splitting a complicated task into multiple smaller tasks for every robot, making it an effective solution to many complex nonlinear tasks. The research on job shop control algorithms of multi-robot scheduling has a long history [9]. Many swarm intelligence and artificial intelligence algorithms have been adopted to multi-robot system [10]. Davis et al. [11] proposed genetic-based algorithms (GA) for treating the non-deterministic polynomial properties in job shop scheduling. Due to the limitation of premature and local convergence features, GA has difficulty in dealing with complex JSSP. To address the shortcomings of GA, Liu et al. [21] proposed a PSO-GA hybrid algorithm, which was found to outperform regular standard particle swarm optimization (PSO) and GA. In their works, crossover and mutation operators in genetic algorithms are introduced to PSO to balance performance between the convergence rate and the convergence precision. Furthermore, Chong et al. [12] proposed a multi-agent genetic algorithm based on tabu search (MAGATS) for improving resource utilization and production efficiency of enterprises.

Inspired by the development of deep learning and neural networks, many deep learning-based algorithms, like DQN [13] and DDPG [14], are proposed to promote the further evolution of reinforcement learning. Simultaneously, multi-agent reinforcement learning (MARL) has also been further developed. The MARL algorithm can be divided into the following four fields, Analysis of emergent behaviors, Learning communication, and Learning cooperation. Under partially observable Markov decision process settings, DIAL [16] was introduced to pass messages between agents. In DIAL, centralized learning and Q-networks are combined, making it possible that gradients to flow from one agent to another. MADDPG [6] extended the DDPG algorithm to the multi-agent environment. MADDPG assumes a critic network and an actor network for every agent and assumes that each agent has its own independent reward function. It can simultaneously solve the collaborative problem, the competitive problem, and the mixed problem in a Multi-agent system (MAS). The COMA model [15] tackles the challenge of multi-agent credit assignment in the Dec-POMDP problem. COMA follows the centralized training distributed execution (CTDE) framework based on actor-critic. Different from MADDPG, COMA uses GRU on actor-network for better performance with local observation problems.

Inspired by the progress of MARL, many deep reinforcement learning-based algorithms are proposed [22] to cope with the dynamic environments in the JSSP. To obtain the capability of solving a wide variety of combinatorial optimization in the Vehicle Routing Problem (VRP), Nazari et al. [23] presented an RL-based end-to-end framework. The algorithm utilizes a self-driven learning procedure to handle the robustness of various problems, which is proved to perform better than the OR-Tools VRP engine [24]. Inspired by AlphaGo [26], Liu et al. [25] proposed a parallel training method based on Actor-Critic structure, and asynchronously update as well as DDPG. The model is evaluated on several JSSP benchmarks from the OR library. The results indicate that the proposed model is robust to both dynamic environments and static JSSP benchmark problems, and is of great balance between makespan and execution time.

Attention mechanism has been widely applied in natural language processing [7] and computer vision [8], especially for sequential models with Long Short Term Memory (LSTM). With the prior knowledge that attention mechanisms can handle spatial interaction between agents, Veličković et al. [26] applied masked self-attention to graph convolution network [28] (GCN) to propose Graph Attention Network (GAT). The GAT assigns different importance to various nodes without costly matrix operations. The evaluated results show that GAT can address the shortcomings of GCN, and it is efficient for spatial relationships since it is parallelizable and can specify arbitrary weights to the neighbors. Furthermore, Xu et al. [8] proposed their model based on soft and hard attention mechanisms, and employ soft attention as the gate of LSTM. In their work, attention can strengthen and weaken the extracted feature, making it flexible for the surrounding information like human beings.

Naive use of MADDPG for JSSP can not specify the prioritization of tasks. We introduce an attention mechanism to our algorithm as a scheduling module. To the best of our knowledge, this is the first work that combines MADDPG and attention to cope with JSSP problems.

Reinforcement Learning (RL). In our work, the task of AGVs scheduling is considered as Markov decision processes(MDPs). An MDP is described by the tuple $\langle S, A, r, P, \gamma \rangle$. The agent takes an action $a_t \in A$ according to the policy $\pi : S \rightarrow A$ and state $s_t \in S$ and gets the reward $r_t \in R$, and s_t transitions to next state s_{t+1} according to the transition probability $P : S \times A \rightarrow S$. The reinforcement learning solves the MDP problem by optimizing the policy π to maximize discounted reward $R_t = \mathbb{E}[\sum_{k=0}^{\infty} \gamma^k r_{t+k} | \pi]$, where $\gamma \in [0, 1]$ denotes the discount factor.

Deep Deterministic Policy Gradient (DDPG). Different from many algorithms for stochastic strategies like stochastic policy gradient (SPG) and Deep Q Network (DQN) [3], DDPG is an off-policy algorithm based on deterministic policy gradient [2]. Policy gradient [4] methods aim at maximizing the objective $J(\theta) = \mathbb{E}_{s \sim p^{\pi}, a \sim \pi_{\theta}}[R]$ according to the policy gradient $\nabla_{\theta} J(\theta)$. Deterministic policy gradient is an extension of policy gradient, and DPG optimizes the deterministic policy $\mu_\theta : S \rightarrow A$ along the gradient direction of deterministic policy $\nabla_\theta J(\theta) = \mathbb{E}_{s\sim D}[\nabla_\theta \mu_\theta(a|s) \nabla_a Q^\mu(s,a)|_{a=\mu_\theta(s)}]$. Based on DPG, DDPG [14] is expanded to actor-critic algorithm framework [5]. There are two deep neural networks parameterized by θ^μ and θ^Q respectively, and named by actor network and critic network. The function of the actor network is to select an action according to deterministic policy $a = \mu(s|\theta^\mu)$, while the critic network computes the value function of state-action and the gradient.

Multi-Agent Deep Deterministic Policy Gradient (MADDPG). Multi-agent reinforcement learning has been widely used to solve multi-agent scheduling. Based on DDPG, MADDPG [6] utilizes an actor-critic model in the multi-agent environment. A centralized critic takes the observations and actions of all agents as input, and the decentralized actor receives information from their corresponding agents. During the learning, the centralized action-value Q_i^μ is updated as $L(\theta_i) = \mathbb{E}_{x,a,r,x'}[(Q_i^\mu(x, a_1, ..., a_N) - y)^2], y = r_i + \gamma Q_i^{\mu'}(x', a_1', ..., a_N')|_{a_j'=\mu_j'(o_j)}$, where μ' is the target policies with delayed parameters θ_i', and o_j is the obvercation of agent j. The state information x could consist of the observations of all agents. And the gradient of $k-th$ sub-policy $\mu_i^{(k)}$ is updated with respect to $\theta_i^{(k)}$ as:

$$\begin{aligned}\nabla_{\theta_i^{(k)}} J_e(\mu_i) = \frac{1}{K} \mathbb{E}_{x,a\sim D_i^{(k)}}[\nabla_{\theta_i^{(k)}} \mu_i^{(k)}(o_i) \nabla_{a_i} \\ Q^{\mu_i}(x, a_1, ..., a_N)|_{a_i=\mu_i^{(k)}(o_i)}].\end{aligned} \tag{1}$$

where $D_i^{(k)}$ is the experience replay buffer for agent i.

Attention. Self-attention is usually adopted to learn the relative weights and importance between agents and targets. For each agent i, the query matrix $q^i = f_q(h^i)$ and key matrix $k^i = f_k(h^i)$ are learned separately, where h^i is the agent state and f is the fully connected network with $relu$ block. The attention for each agent i is represented as $Attention(q^i, k^i) = softmax(\frac{q^i k^{iT}}{\sqrt{d_k}})$, where d_k is the dimensionality of the matrix to ensure numerical stability.

3 Method

3.1 Architecture

In this section, we will propose a multi-robot task scheduling algorithm under the framework of MADDPG and the idea of an attention mechanism. We call it the attention scheduling algorithm (Att-Sched). The whole architecture is shown in Fig. 2. Att-Sched consists of the following three components: *(i)* actor network, *(ii)* attention scheduler, and *(iii)* critic network. The actor network outputs action based on its own observation o_i and task T_i^{sch} distributed from the Attention Scheduler. The critic network is to evaluate the centralized action-value function, its input includes actions $\boldsymbol{a}^t = (a_1^t, \cdots, a_N^t)$ and observations

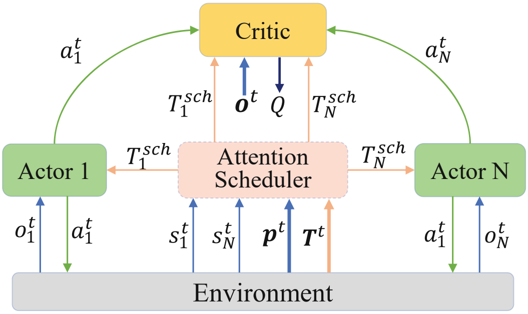


Fig. 2. The entire architecture of Att-Sched.

$\boldsymbol{o}^t = (o_1^t, \cdots, o_N^t)$ of all agents, as well as scheduled tasks from Attention Scheduler. Therefore, the actor networks in Att-Sched are updated by the following gradient from critics:

$$\nabla_{\theta_i^{(k)}} J_e(\mu_i) = \frac{1}{K} \mathbb{E}_{x,a \sim D_i^{(k)}} [\nabla_{\theta_i^{(k)}} \mu_i^{(k)}(o_i, T_i^{sch}) \nabla_{a_i} \\ Q^{\mu_i}(x, a_1, ..., a_N, \boldsymbol{T}^{sch})|_{a_i=\mu_i^{(k)}(o_i, T_i^{sch})}]. \tag{2}$$

where $\boldsymbol{T}^{sch} = (T_1^{sch}, \cdots, T_N^{sch})$. The Attention Scheduler is the most important part of the Att-Sched algorithm, its input includes the dynamic states $\boldsymbol{s}^t = (s_1^t, \cdots, s_N^t)$ of all agents, as well as the locations $\boldsymbol{T}^t = (t_1^t, \cdots, t_M^t)$ and prioritization $\boldsymbol{p}^t = (p_1^t, \cdots, p_M^t)$ of all tasks. The sum of all elements in $\boldsymbol{p}^t$ equals 0.

3.2 Attention Scheduler

We consider a multi-robot job shop scheduling system, the robot should supply materials according to the prioritization of the machine's demand for materials. Therefore, the Attention Scheduler needs to allocate tasks appropriately based on the adaptability of each robot to all tasks. Firstly, we encode the dynamic state of each agent and each task to be performed:

$$x_i^t = \boldsymbol{\mu}_{\theta_i^x}(s_i^t), \quad y_j^t = \boldsymbol{\mu}_{\theta_j^y}(T_j^t). \tag{3}$$

where $\boldsymbol{\mu}_{\theta_i^x}$ and $\boldsymbol{\mu}_{\theta_j^y}$ are multi-layer perceptron (MLP) parameterized by θ_i^x and θ_j^y, respectively. Further, we calculate the query of x_i^t and the key of y_j^t according to (4), where $\boldsymbol{\mu}_{\theta_i^q}$ and $\boldsymbol{\mu}_{\theta_j^k}$ are multi-layer perceptron (MLP) parameterized by θ_i^q and θ_j^k, respectively.

$$q_i^t = \boldsymbol{\mu}_{\theta_i^q}(x_i^t), \quad k_j^t = \boldsymbol{\mu}_{\theta_j^k}(y_j^t). \tag{4}$$

We compute the interactions between each query q_i^t and all the keys k_i^t by performing Hadamard product $\odot$. We then apply a linear transformation $\boldsymbol{W}_{iq}^{[d_q \times d_1]}$ to every interaction:

$$qh_{ij}^t = q_i^t \odot k_j^t, \quad qu_{ij}^t = \boldsymbol{W}_{iq}^{[d_q \times d_1]} qh_{ij}^t. \tag{5}$$

where the dimensions of q_i^t, k_j^t, and qh_{ij}^t are both $\mathbb{R}^{d_q}$, and the dimension of qu_{ij}^t is $\mathbb{R}^1$. The purpose of the linear transformation $\boldsymbol{W}_{iq}^{[d_q \times d_1]}$ is to produce the adaptive weight of a robot for a task.

We then perform a *softmax* operation on qu_{ij}^t to obtain the attention weight δ_i^t of each robot for all tasks, which measures the adaptability of a robot to all tasks.

$$\delta_i^t = \mathrm{softmax}[\frac{\mathrm{qu_{i1}^t}}{\sqrt{\mathrm{d_q}}} \cdots \frac{\mathrm{qu_{ij}^t}}{\sqrt{\mathrm{d_q}}} \cdots \frac{\mathrm{qu_{iM}^t}}{\sqrt{\mathrm{d_q}}}] \tag{6}$$

According to the attention weight δ_i^t, the Attention Scheduler assigns the task T_i^{sch} corresponding to the largest weight in δ_i^t to an agent. That means the agent is best suited to perform the selected task. After that, each agent takes action according to its own observation and assigned tasks.

3.3 Training of Attention Scheduler

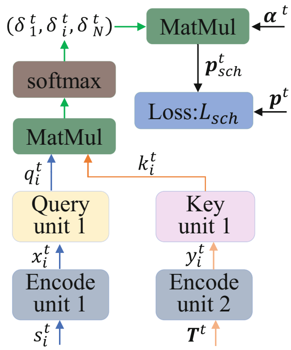


Fig. 3. Schematic process diagram of calculation and training of attention scheduler.

Through all the operations in the Attention Scheduler, we can get the team attention matrix $\boldsymbol{\delta}^t = [\delta_i^t, \cdots, \delta_N^t]^T$. The dimension of matrix $\boldsymbol{\delta}^t$ is $\mathbb{R}^{d_N \times d_M}$ and all elements in matrix $\boldsymbol{\delta}^t$ are nonzero. We define a matrix $\boldsymbol{\alpha}^t$ with dimension $\mathbb{R}^{d_1 \times d_N}$, where all elements are 1. The matrix $\boldsymbol{\alpha}^t$ is used to replace the key unit in the self-attention [7]. Then, we wrote the final attention task prioritization as:

$$\boldsymbol{p}_{sch}^{t} = \frac{1}{N}\boldsymbol{\alpha}^{t}\boldsymbol{\delta}^{t} \tag{7}$$

The expression in $\boldsymbol{p}_{sch}^{t}$ is the prioritization reassigned to all tasks in combination with the dynamic characteristics of the robot. Besides, we use the technique of replay buffer, so the Attention Scheduler is trained to minimize the following loss:

$$\mathcal{L}_{sch} = \sum_{i=1}^{b}(\boldsymbol{p}_{sch}^{t} - \boldsymbol{p}^{t})^{2}. \tag{8}$$

where b is the batch size of sampled data from the replay buffer, the above process of calculating attention weight and loss can be summarized in Fig. 3.

4 Experiment

In this section, We will demonstrate the performance of the Att-Sched algorithm in simulation environments. We refer to the continuous environment in [6] and make some adjustments. The simulation environment is shown in Fig. 4, n robots need to work and collaborate to provide materials to m machines that lack materials. The robots receive the collective rewards based on the sum of distance $-d$ between every robot and its nearest machine(Y). In addition, if a robot reaches a machine(N) that does not need to provide materials, the robot team will give a penalty of -2. Since the working condition of the machine is relatively stable, the machine that needs to be provided materials will not change too quickly, We randomly specify m machines that need to be supplied with materials and the corresponding prioritization value at the beginning of each episode. Although the simulation environment we show is relatively simple, our algorithm can be easily extended to larger and more complex scheduling systems. After some testing, the selected hyperparameters are shown in Table 1.

Firstly, We trained 6 sets of training processes and compared the performance differences between Att-Sched and MADDPG algorithms, as shown in Fig. 5. It can be seen that the reward gradually increases with the number of training episodes until it converges, indicating that Att-Sched has carried out an effective learning process. In addition, the confidence band of the reward curve is very small, indicating that the performance of our proposed algorithm is stable. Finally, we can see that the Att-Sched algorithm with the attention mechanism performs better than MADDPG algorithm. We then show the trajectory of the robot in several task scenarios in Fig. 6. We only show a few typical task scenarios, but in other task scenarios, the attention scheduler can still assign tasks reasonably. We can see that the robot team can reach the machine that needs to be provided with materials under the task assigned by the Attention Scheduler. For Fig. 6(c), robot 1 is assigned to provide materials to the machine with

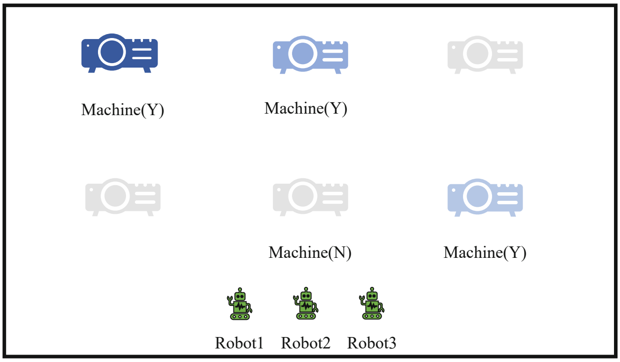


Fig. 4. Simulation environment of the robot scheduling system. The blue machine needs to be provided with materials by the robot, and its color from dark to light means that the prioritization is gradually reduced. The gray machine does not need to be provided with materials by robots.

the highest priority and closer distance. Robot 2 needs to bypass machine(N) to provide materials to the machine with the highest priority, so robot 2 finally is assigned to provide materials to another priority machine. This causes Robot 1 and Robot 2 to cross on the trajectory. In other scenarios, the attention scheduler makes a good task allocation to the machines that need to be provided with materials

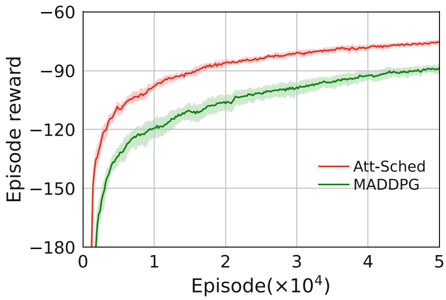
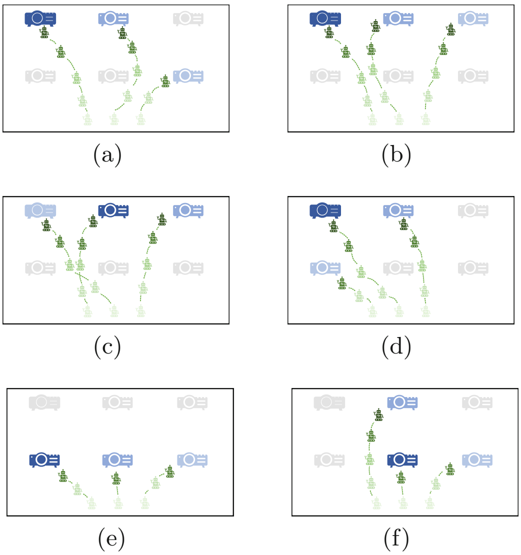


Fig. 5. The performance curve of Att-Sched and MADDPG algorithm.

Table 1. Hyper-parameters settings for the simulation environment.

Hyper-parameters	Value
Discount factor (γ)	0.99
Learning rate (α)	0.01
Initialized greedy rate	0.1
Batch size	1024
Replay buffer size	250000
Episode number (N_e)	50000
Trajectory length (T)	50
Robot number	3
Machine(Y) number	3
Encoder units	20
Query units	20
key units	20
Actor units	(32, 32)
Critic units	(80, 80)



(a) (b) (c) (d) (e) (f)

Fig. 6. Robot trajectories in some mission scenarios of Att-Sched algorithm

5 Conclusion

This paper presents a MADDPG-based framework with an attention-scheduling module for multi-robot task scheduling. Contrary to many works which used hand-crafted functions and rules for scheduling, we utilize MADDPG to tackle the nonlinear and non-deterministic polynomial characteristics in job shop scheduling. Our attention scheduling module assigns different importance to robots and learns the urgency of tasks to assign prioritization dispatching rules. After empirical evaluations of the simulation environments, our approach can make energy-saving scheduling, and fulfill the demands of machines with prioritization.

However, there are two drawbacks to our work: *(i)* Our simulation environment is based on a small-scale job shop. Since MADDPG can handle more complex situations, we will increase the number of robots and machines and evaluate our approach with more realistic constraints for future work, such as obstacles. *(ii)* Our attention scheduling module can not discriminate assigned machines at the first time (shown as Fig. 6 (d), robot 2 comes close to the first machine in the second row at the first time. After robot 1 comes to the machine, robot 2 changes its way to the highest priority machine.). The reason for the limitation is that all robots are independent, and there is no communication between robots. Future work will attempt to tackle this by using a control module to plan paths for all robots simultaneously.

References

1. Wang, Y., Liu, H., Zheng, W., et al.: Multi-objective workflow scheduling with deep-Q-network-based multi-agent reinforcement learning. IEEE Access **7**, 39974–39982 (2019)
2. Silver, D., Lever, G., Heess, N., et al.: Deterministic policy gradient algorithms. In: International Conference on Machine Learning, pp. 387–395. PMLR (2014)
3. Mnih, V., et al.: Human-level control through deep reinforcement learning. Nature **518**(7540), 529 (2015)
4. Eshkevari, S.S., Eshkevari, S.S., Sen, D., et al.: Active structural control framework using policy-gradient reinforcement learning. Eng. Struct. **274**, 115122 (2023)
5. Hong, M., Wai, H.T., Wang, Z., et al.: A two-timescale stochastic algorithm framework for bilevel optimization: complexity analysis and application to actor-critic. SIAM J. Optim. **33**(1), 147–180 (2023)
6. Lowe, R., Wu, Y.I., Tamar, A., et al.: Multi-agent actor-critic for mixed cooperative-competitive environments. Adv. Neural Inf. Process. Syst. **30** (2017)
7. Vaswani, A., Shazeer, N., Parmar, N., et al.: Attention is all you need. Adv. Neural Inf. Process. Syst. 5998–6008 (2017)
8. Parvaiz, A., Khalid, M.A., Zafar, R., et al.: Vision transformers in medical computer vision-a contemplative retrospection. Eng. Appl. Artif. Intell. **122**, 106126 (2023)
9. Baker, A.D.: A survey of factory control algorithms that can be implemented in a multi-agent heterarchy: dispatching, scheduling, and pull. J. Manuf. Syst. **17**(4), 297–320 (1998)
10. Dahiya, A., Aroyo, A.M., Dautenhahn, K., et al.: A survey of multi-agent Human-Robot Interaction systems. Robot. Auton. Syst. **161**, 104335 (2023)
11. Davis, L.: Job shop scheduling with genetic algorithms. In: Proceedings of an International Conference on Genetic Algorithms and Their Applications **140** (1985)

12. Peng, C., Wu, G., Liao, T.W., et al.: Research on multi-agent genetic algorithm based on tabu search for the job shop scheduling problem. PLoS ONE **14**(9), e0223182 (2019)
13. Mnih, V., Kavukcuoglu, K., Silver, D., et al.: Human-level control through deep reinforcement learning. Nature, **518**(7540), 529–533 (2015)
14. Lillicrap, T.P., Hunt, J.J., Pritzel, A., et al.: Continuous control with deep reinforcement learnings. In International Conference on Learning Representations (2016)
15. Foerster, J., Farquhar, G., Afouras, T., et al.: Counterfactual multi-agent policy gradients. Proc. AAAI Conf. Artif. Intell. **32**(1) (2018)
16. Foerster, J., Assael, I.A., De Freitas, N., et al.: Learning to communicate with deep multi-agent reinforcement learning. Adv. Neural Inf. Process. Syst. **29** (2016)
17. Garey, M.R., Johnson, D.S., Sethi, R.: The complexity of flowshop and jobshop scheduling. Math. Oper. Res. **1**(2), 117–129 (1976)
18. Balas, E.: Machine sequencing via disjunctive graphs: an implicit enumeration algorithm. Oper. Res. **17**(6), 941–957 (1969)
19. Zhu, W., Wang, Z., Wang, X., et al.: A dual self-attention mechanism for vehicle re-identification. Pattern Recogn. **137**, 109258 (2023)
20. Wang, G., Zhao, Y., Tang, C., et al.: When shift operation meets vision transformer: an extremely simple alternative to attention mechanism. Proc. AAAI Conf. Artif. Intell. **36**(2), 2423–2430 (2022)
21. Liu, L.L., Hu, R.S., Hu, X.P., et al.: A hybrid PSO-GA algorithm for job shop scheduling in machine tool production. Int. J. Prod. Res. **53**(19), 5755–5781 (2015)
22. Cunha, B., Madureira, A.M., Fonseca, B., Coelho, D.: Deep reinforcement learning as a job shop scheduling solver: a literature review. In: Madureira, A.M., Abraham, A., Gandhi, N., Varela, M.L. (eds.) Hybrid Intelligent Systems: 18th International Conference on Hybrid Intelligent Systems (HIS 2018), pp. 350–359. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-14347-3_34
23. Nazari, M., Oroojlooy, A., Snyder, L., Takác, M.: Reinforcement learning for solving the vehicle routing problem. In: Proceedings of the Advanced Neural Information Processing System, pp. 9839–9849 (2018)
24. Google. Inc. Google's optimization tools (or-tools) (2018). https://github.com/google/or-tools
25. Liu, C.L., Chang, C.C., Tseng, C.J.: Actor-critic deep reinforcement learning for solving job shop scheduling problems. IEEE Access **8**, 71752–71762 (2020)
26. Silver, D., et al.: Mastering the game of go with deep neural networks and tree search. Nature **529**(7587), 484–489 (2016)
27. Veličković, P., Cucurull, G., Casanova, A., et al.: Graph attention networks. In: International Conference on Learning Representations (2018)
28. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: International Conference on Learning Representations (2017)
29. Chaudhry, I.A., Khan, A.A.: A research survey: review of flexible job shop scheduling techniques. Int. Trans. Oper. Res. **23**(3), 551–591 (2016)

Collaborative Search Method of Heterogeneous UAVs in Gray Area

Hongmin Wang, Yunong Bu, Jiajia Zhao, and Mingrui Hao(✉)

Science and Technology on Complex System Control and Intelligent Agent Cooperation Laboratory, Beijing 100074, China
hmrhit@163.com

Abstract. A collaborative search planning method based on an information map is proposed for the problem of multi-moving target detection in gray areas using a heterogeneous UAV swarm. This method takes into account the detection probability and false alarm probability of sensors, the heterogeneity and flight constraints of UAVs, and the random movement of targets. The mathematical planning model for the collaborative search of multiple UAVs is built by balancing short-term gain, long-term gain, and coordination gain. An information map for search is designed, incorporating target existence probability, environmental uncertainty, and revisiting pheromones. Different planning schemes are designed based on the heterogeneous characteristics of UAVs. Through numerical simulations in typical collaborative search scenarios, the effectiveness of the proposed method is validated. The simulation results show that the proposed method can make search trajectory decisions for each UAV within seconds. The organic combination of short-term, long-term, and coordination gain can guide the UAV swarm to capture more targets. Comparative simulation results demonstrate that the proposed method can capture more targets with fewer false alarms, effectively improving the task efficiency of heterogeneous multi-UAV collaborative search.

Keywords: Collaborative Search · Information Map · Heterogeneous UAVs · revisit mechanism · receding horizon

1 Introduction

The gray area multi-UAV cooperative search problem has high practical value in various applications such as battlefield reconnaissance, urban tracking, maritime rescue, security monitoring, and military operations, which has attracted wide attention and extensive research in the academic community [1]. From the perspective of method characteristics, it can be divided into two categories: rule-based search methods and mathematical optimization-based search methods.

Rule-based cooperative search methods are designed for specific task scenarios, primarily focusing on achieving full coverage through the design of search rules. References [2–4] propose parallel search strategies, using "Z-shaped" parallel flight paths to achieve complete coverage of the area. References [5, 6] introduce algorithms for motion target perpendicular search and motion target diagonal search. References [7, 8] aiming

J. Yu et al. (Eds.): CCSICC 2023, LNEE 1206, pp. 13–28, 2024.
https://doi.org/10.1007/978-981-97-3332-3_2

to address the low search efficiency of traditional perpendicular search algorithms for moving targets, propose side-by-side backtrack search algorithm and transverse equally divided perpendicular search algorithm to enhance the search capability for moving targets. The aforementioned cooperative search methods are designed based on predetermined search rules, which are less flexible and may have limited effectiveness in searching for moving targets.

Mathematical programming-based collaborative search methods typically involve constructing a mathematical programming model for collaborative search, designing a search information map that describes environmental information, and then using heuristic algorithms [9–11] or reinforcement learning [12, 13] methods to solve the model and obtain the optimal search trajectory. As the search progresses, the search information map is dynamically updated to effectively utilize real-time detection information, making it suitable for dynamic search processes.

Reference [14] proposes a UAV cooperative search method based on a coverage distribution information map. Reference [15] designs an independent layered decomposed environmental information map system to achieve coverage search in the area. Reference [16] considers constraints such as sensors, UAV motion, and communication, and presents a probability graph search model based on Bayesian updates. Reference [17] introduces a method for motion target prediction and designs a search information map that incorporates environmental uncertainty, target probability, and pheromones. Based on this, a cooperative search model is constructed, considering the cost of environmental search, target discovery, and inter-UAV coordination. Reference [18] establishes a probability search information map and proposes a distributed algorithm for information fusion and cooperative control based on it. Reference [19] considers revisiting high uncertainty regions and establishes a cooperative search model that includes environmental search benefit, target detection and grid revisit benefit, and inter-UAV coordination cost. A multi-UAV cooperative search method with pheromone revisiting is proposed. Reference [20] designs a search information map that incorporates target probability distribution, environmental uncertainty, and environmental search status. A cooperative search model is established, considering target search benefit, environmental search benefit, expected detection benefit, and cooperative benefit, and a revisiting mechanism is customized to guide UAVs in revisiting.

It can be observed that current research on collaborative search mainly has two problems. Firstly, most studies only consider the short-term gain of UAV search (i.e., the benefits obtained from searching the maximum possible area at present) and coordination gain (i.e., the benefits obtained from avoiding overlapping flight paths), while giving less consideration to long-term gain (i.e., the benefits obtained from reaching subsequent promising search areas after executing the current optimal trajectory). Secondly, there is limited research on collaborative search involving heterogeneous aircraft. Generally, it is assumed that all UAVs are of the same type, while collaborative search with heterogeneous UAVs can better leverage the advantages of different types of UAVs, leading to improved search performance.

To address the aforementioned issues, this paper proposes an Information Map-based Collaborative Search Planning for Heterogeneous UAV Swarms (IM-CSPH) method.

Firstly, a multi-UAV collaborative search planning model is established, which comprehensively balances the short-term gain, long-term gain, and coordination gain of UAV search. Secondly, a search information map is designed to dynamically describe the environment, incorporating a revisiting pheromone factor to guide UAVs in revisiting. Based on this, different search planning methods are designed for different types of UAVs, taking into account the heterogeneity of the UAV swarm. The collaborative search planning model is solved using a receding horizon architecture. Finally, the effectiveness of the proposed method is validated via numerical simulations.

2 Problem Formulation of Heterogeneous UAVs Collaborative Search

A heterogeneous UAV swarm consisting of n UAVs collaboratively searches for k targets within the region S. The UAVs have a speed of v_d, and the targets within the region S have unknown initial positions and move randomly with a speed of v_t, as shown in Fig. 1. The onboard sensors of the UAVs have detection probability and false alarm probability. After confirming the presence of a target, the UAV captures it and then proceeds to search for other targets. The objective of designing a cooperative search method is to capture as many moving targets as possible within a specified time frame while minimizing false alarms on non-existent targets.

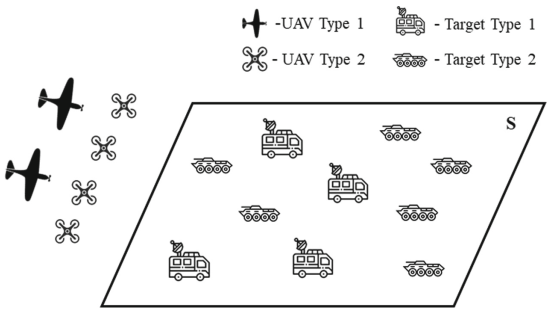


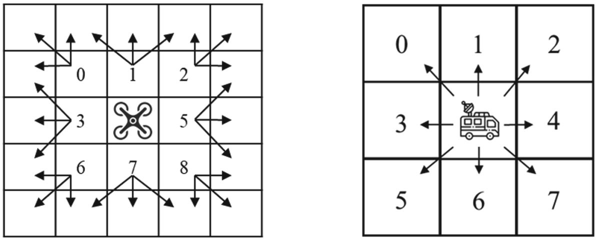
Fig. 1. Heterogeneous UAV Swarm Collaborative Search Task.

2.1 Task Area Mode

The task area S is modeled using a grid-based approach, dividing it into $L_x \times L_y$ grids. Each grid is identified by its coordinates x and y. For example, the grid in the bottom-left corner is denoted as $G_{1,1}$.

2.2 UAV Model

The motion range of the UAV is discretized, assuming that the UAV moves by one grid cell at a time. The UAV's movement step is greater than the minimum trajectory segment constraint, and θ represents the maximum turning angle of the UAV. By limiting the maximum turning angle and minimum trajectory segment length, the UAV's normal overload constraint is satisfied to ensure the feasibility of the planned trajectory. The UAV has eight possible motion directions at each moment, as shown in Fig. 2(a).



(a) UAV Behaviors (b) Motion Target Behaviors

Fig. 2. Model Behaviors.

Different types of UAVs have different detection ranges, detection probabilities, and false alarm probabilities. The detection probability is defined as:

$$P_D = P(Detection\ target\ present|Actual\ target\ present)$$

The false alarm probability is defined as:

$$P_F = P(Detection\ target\ present|Actual\ target\ not\ present)$$

Considering the types and detection capabilities of real UAVs, UAVs can be abstracted into two categories. The first category includes UAVs equipped with radar-based detection payloads, offering a wide coverage range but with a low detection probability and a high false alarm probability. In contrast, the second category comprises UAVs equipped with optical-based detection payloads, featuring a narrow coverage range, high detection probability, and low false alarm probability.

2.3 Motion Target Model

The moving target (e.g., radar vehicle) can move on the ground in a two-dimensional plane. The target has 8 possible behaviors at each moment, as shown in Fig. 2(b). The initial positions of the targets within the task area are randomly assigned, and their movements are random as well. However, they will not move outside of the task area.

2.4 Collaborative Search Planning Model

Taking into account the UAV's flight and collision avoidance constraints, with the core objective of balancing the short-term, long-term, and coordination gain of multi-UAV search, and aiming to maximize the overall search efficiency, the problem model for multi-UAV collaborative search is established as follows:

$$\text{Find } R = \{R(1), R(2), \cdots, R(t), \cdots, R(T)\}$$

$$max \sum\nolimits_{t=0}^{T} J(t, R(t)) = \sum\nolimits_{t=0}^{T} [w_V J_V(t, R(t)) + w_E J_E(t, R(t)) + w_C J_C(t, R(t))] \quad (1)$$

$$\text{s.t.} t \in [0, T]$$

$$R(t) = \{R_1(t), R_2(t), \cdots, R_n(t)\}$$

$$R_i(t) \cap R_j(t) = \emptyset, \forall i, j = 1, 2, \cdots, n, i \neq j \quad (21)$$

$$R_i(t) \in F, i = 1, 2, \cdots, n \quad (3)$$

where R is the decision variable representing the trajectory of the UAV swarm, $R(t)$ represents the trajectory of the UAV swarm at time t, T is the total duration of the search task, and n represents the number of UAVs.

Equation (1) represents the objective function, where $J(t, R(t))$ represents the search efficiency of the UAV swarm at time t, and $J_V(t, R(t)), J_E(t, R(t)), J_C(t, R(t))$ represent the value search gain, potential search gain, and coordination search gain of the UAV swarm at time t, respectively. w_V, w_E, w_C represent the weight coefficients for the gain.

Equation (2) represents the collision avoidance constraint, where $R_i(t)$ represents the search trajectory of the $i-$ th UAV at time t.

Equation (3) represents the flight constraints of the UAVs, where F represents the set of flight constraints, including constraints on the UAV's flight direction and flight distance.

Value Search Gain J_V.
Value search gain represents the amount of value obtained by the UAV in searching the current most promising search area during the search process, reflecting the idea of short-term search gain. The value search gain of the UAV swarm at time t, denoted as $J_V(t, R(t))$, is given by the following equation:

$$J_V(t, R(t)) = \sum\nolimits_{i=1}^{n} \sum\nolimits_{(x,y) \in G_i(R_i(t))} V(x, y, t) \quad (4)$$

where $G_i(R_i(t))$ represents the detection range of UAV i at time t; $V(x, y, t)$ represents the value of grid $G_{x,y}$ at time t, calculated as follows:

$$V(x, y, t) = w_1 p(x, y, t) + w_2 \chi(x, y, t) + w_3 s(x, y, t) \quad (5)$$

where $p(x, y, t)$ represents the probability of the grid $G_{x,y}$ having a target at time t, $\chi(x, y, t)$ represents the uncertainty, $s(x, y, t)$ represents the revisit pheromone, and w_1, w_2, w_3 is the weight assigned to the aforementioned three factors.

Potential Detection Gain J_E.
In order to enhance the global search capability and enable UAVs to reach subsequent potential and promising search areas after executing the current search path, laying the foundation for subsequent planning, the concept of potential detection gain J_E has been designed to reflect the long-term search benefits. The potential detection gain $J_E(t, R(t))$ of the UAV swarm at time t are expressed as follows:

$$J_E(t, R(t)) = \sum_{i=1}^{n} \left(\frac{\sum_{(x,y)\in A_i^N(R_i(t))} V_p(x, y, t)}{n_{A_i^N(R_i(t))}} - \frac{\sum_{(x,y)\in A_i^F(R_i(t))} V_p(x, y, t)}{n_{A_i^F(R_i(t))}} \right) \tag{6}$$

$$V_p(x, y, t) = \chi(x, y, t) + s(x, y, t)$$

where $A_i^N(R_i(t))$ represents the proximity detection range of UAV i at time t, $A_i^F(R_i(t))$ represents the distancing detection range of UAV i at time t, $n_{A_i(R_i(t))}$ represents the number of grids in zone $A_i(R_i(t)$, $\chi(x, y, t)$ represents the uncertainty of grid $G_{x,y}$ at time t, and $s(x, y, t)$ represents the revisit pheromone of grid $G_{x,y}$ at time t.

The definitions of proximity and distancing detection ranges are shown in Fig. 3. Assuming the entire task area is divided into 9 sub-zones, after executing the trajectory, the UAV may exhibit the following tendencies, excluding the zone it currently occupies: getting closer to a zone, moving farther away from a zone, or maintaining the same distance from a zone. The grids in the zones where the distance gets closer are considered as the proximity detection range, represented by the solid shaded area in the figure. The grids in the zones where the distance increases are considered as the distancing detection range, represented by the diagonal shaded area in the figure.

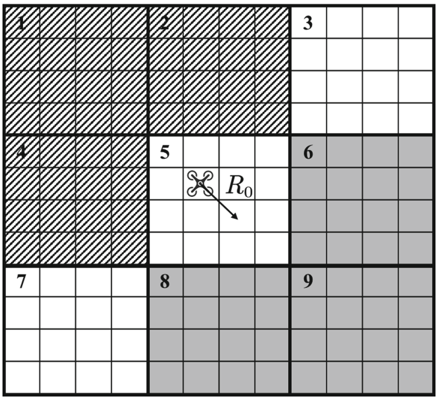


Fig. 3. Potential detection range.

Coordinated Search Gain J_C.
To enhance the efficiency of collaborative search among UAVs, minimize redundant search efforts between UAVs, avoid flight conflicts, and expand the search range of the UAV swarm as much as possible, a coordinated search gain J_C has been designed. It embodies the idea of mutual coordination among the UAV swarm and the avoidance of overlapping flight paths. The coordinated search gain $J_C(t, R(t))$ of the UAV swarm at time t is expressed as follows:

$$
\begin{aligned}
J_C(t, R(t)) = \sum\nolimits_{(x,y)\in G(R(t))} & V(x, y, t) + \\
\frac{1}{n_{A^N(R(t))}} \sum\nolimits_{(x,y)\in A^N(R(t))} & \left[\chi(x, y, t) + s(z, y, t)\right] - \\
\frac{1}{n_{A^F(R(t))}} \sum\nolimits_{(x,y)\in A^F(R(t))} & \left[\chi(x, y, t) + s(z, y, t)\right]
\end{aligned}
\tag{7}
$$

where $G(R(t))$ represents the detection range of the UAV swarm at time t; $A^N(R(t))$ represents the proximity detection range of the UAV swarm at time t; $A^F(R(t))$ represents the distancing detection range of the UAV swarm at time t; $n_{A(R(t))}$ represents the number of grids in area $A(R(t))$.

3 Construction and Updating of the Search Information Map

The search information map primarily consists of the target probability distribution map, environmental uncertainty map, and revisit pheromone map.

3.1 Target Probability Distribution Map

The target probability distribution map $p(x, y, t) \in [0, 1]$ represents the probability of grid $G_{x,y}$ having a target at time t.

During the execution of the search mission, UAVs dynamically update the target existence probabilities $p(x, y, t)$ in the task area based on the detection information $b(x, y, t)$ from their own sensors, considering the detection probability P_D and false alarm probability P_F of the sensors. The Bayesian criterion is used to update the target existence probabilities in the detected grids.

$$
p(x, y, t+1) = \begin{cases} \frac{P_D p(x,y,t)}{P_D p(x,y,t) + P_F(1-p(x,y,t))}, b(x, y, t) = 1 \\ \frac{(1-P_D)p(x,y,t)}{(1-P_D)p(x,y,t) + (1-P_F)(1-p(x,y,t))}, b(x, y, t) = 0 \end{cases}
\tag{8}
$$

where $b(x, y, t)$ represents the detection result of UAV for grid $G_{x,y}$ at time t. $b(x, y, t) = 1$ indicates that the grid is detected to have a target, and vice versa if there is no target.

Due to the random movement of targets, it is possible that targets may still exist in areas that have been previously detected as empty. Therefore, a compensation mechanism