Received March 31, 2019, accepted April 22, 2019, date of publication April 29, 2019, date of current version November 8, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2913764

Centralized Patrolling With Weakly-Coupled Agents Using Monte Carlo Tree Search

XIN ZHOU[1], WEIPING WANG[1], YIFAN ZHU[1], TAO WANG[1], AND BO ZHANG[2]

[1]College of Systems Engineering, National University of Defense Technology, Changsha 410073, China
[2]School of Automation, Guangdong University of Technology, Guangzhou 510006, China

Corresponding author: Tao Wang (wangtao1976@nudt.edu.cn)

This work was supported in part by the Natural Science Foundation of China under Grant 61273198 and Grant 71373282.

ABSTRACT In this paper, a new optimal multi-agent continuous patrol algorithm is proposed to solve the information gathering problem in dynamic environments. First, the environment is modeled as a layout graph with information attached to vertices. Each agent patrols within a specified area and only interacts with its adjacent agents. The problem is then cast as the factored multi-agent partially observable Markov decision process (MPOMDP). Furthermore, a scalable centralized online planning algorithm, called the factored belief-based variable eliminated Monte Carlo planning algorithm, is proposed based on the Monte Carlo tree search (MCTS) method. The proposed algorithm constructs an independent local look-ahead tree for each agent, where actions are coordinated at specific locations of each tree based on the variable elimination algorithm. Finally, we mimic typical patrol problems to empirically evaluate the proposed algorithm by benchmarking it against some state-of-the-art solvers. The results demonstrate that the performance of the proposed algorithm is remarkable for multi-agent systems with the weakly-coupled structure in partially observable scenarios.

INDEX TERMS Multi-agent system, weakly-coupled structure, MPOMDP, MCTS, variable elimination.

I. INTRODUCTION

The Unmanned Aerial Vehicle (UAV) equipped with sensors is an important mean of achieving situational awareness, understanding and predicting what happens in highly dynamic environments [1], [2], such as disaster areas after the earthquake. In this paper, we study that a team of UAVs with weakly-coupled structure continuously monitor the environment in a collaborative manner.

A. INFORMATION GATHERING PROCESS IN DISASTER RESPONSE

The disaster emergency response system, such as the Human-Agent Collectives for Emergency Response [3], is able to respond to the occurrences of disaster, and provides decision supports [4]. UAVs are part of the system.

First, the disaster response system collects and preprocesses prior information about the disaster area, such as weather forecasts, satellite imageries, and crowdsourcing reports. Second, the disaster response system makes emergency response decisions based on prior information. A hierarchical organization of the Observe-Orientate-Decide-Act loop [5] is usually used in disaster response systems to divide decisions into strategic level, tactical level, and operational level. Different levels are interrelated, including commands flowing down the hierarchy, and status feedback and sensory information flowing up. In each level, the decision maker's perspective is different. In the strategic level, decision makers focus on the main objectives of emergency response effort. According to the main objectives, tactical decision makers plan the patrol area and route for each UAV in a high level. In the operational level, operators plan routes for UAVs in a low level. Third, the team of UAVs is dispatched to the target areas to collect reliable and high-quality data.

Unlike previous work on the control of UAVs in the operation level [6], [7], in this paper we concentrate on the research from the perspective of the tactical level. Specifically, the problem is featured as how to patrol the environment using a limited number of UAVs with limited detection capabilities. The environment is modeled as a layout graph with information attached to the vertices. The UAV is modeled as an agent, and each agent patrols within a specific patrol area.

The associate editor coordinating the review of this manuscript and approving it for publication was Xian Sun.

B. RELATED WORK

In this section, researches on the multi-agent continuous patrolling problem are first introduced. Then, formulations and solvers to the problem are presented respectively.

A number of coordination algorithms on the multi-agent continuous patrolling problem have been developed [8]. For the dynamic environment, previous researches [9], [10] study the fully observable scenarios, where agents can directly perceive the underlying states. In this paper, we focus on the partially observable scenarios where agents cannot make sense of the underlying states, but only the states of the current locations currently. In [11], an anytime planning algorithm is put forward for decentralized multi-agent information gathering problem, where each agent periodically transmits the compressed form of its look-ahead tree to other agents. However, it converges to the optimal policy under certain assumptions. In [12], a multi-agent online planning algorithm is proposed, where each agent builds a look-ahead tree and coordinates actions by the max-sum algorithm [13]. However, the max-sum algorithm cannot provide optimal guarantee when the cycle exists. In addition, some algorithms can achieve near-optimal performance [14] [15], but they require a submodular [16] objective function, which is not suitable for all scenarios. Thus, there are further work still required to improve the performance of multi-agent patrolling in dynamic environments.

The multi-agent continuous patrolling can be modeled as the sequential decision making problem. MPOMDPs are the extension of POMDPs [17], [18], which are able to capture the partially observable feature of multi-agent patrolling due to sensor capability constraints. In MPOMDPs, the uncertainty of the state is represented by the belief, and it allows agents to act in a centralized manner. The centralized controller takes joint observation and performs joint actions. In many real-world scenarios, global states can be factored into different state features. Our work is motivated by the transition-decoupled POMDP [19], which provides a natural interactive representation framework for agents with weakly-coupled structures. In the transition-decoupled POMDP, global states are represented in the local states of each agent, and each agent may share its local states with other agents. The local state is composed of mutually-modeled features, including unaffectable features, locally controllable features, and nonlocally-controlled features. The factorization of global states contributes to analyzing conditionally independent relationships between existing variables.

Partially observable Monte Carlo planning (POMCP) [20] is one of the leading methods for solving general MPOMDPs based on Monte Carlo tree search (MCTS) [21]. POMCP builds a search tree of the history and uses Monte Carlo simulation to evaluate the value of each node. However, the joint action space and the joint observation space increase exponentially with the number of agents, resulting in a high branching factor for the search tree [22]. Therefore, the decomposition of the global look-ahead tree into multiple local look-ahead trees can effectively avoid the problem of undersampling. Some decentralized online planning algorithms [11], [12] build a local search tree for each agent, and construct a decision-making coordination mechanism in the decentralized setting where the communication may be disturbed or interrupted. In this paper, we focus on the centralized setting where the communication is free of noise and latency. Thus, we can design the algorithm by taking advantage of the centralized structure.

C. CONTRIBUTION OF THIS PAPER

In the paper, the main challenges we need to address involve two aspects: First, the number of UAVs is limited and sensors of UAVs cannot cover the entire target area at any time. Second, the actions of UAVs should be coordinated to reduce conflicts and improve effects. To solve these challenges, the problem is cast as the factorized MPOMDP framework and a centralized online algorithm is proposed. The objective of the team of agents is to gather as much information as possible. In other words, the objective is to compute the optimal continuous patrol route for each agent to maximize the global cumulative discounted reward over time. The main contributions presented in this paper are as follows.

- A factorized MPOMDP framework for the centralized multi-agent patrolling problem is put forward. In the factorized MPOMDP, we relax the condition of the environment. In particular, the factored global states, global observations, global reward functions and beliefs lead to a natural decomposition of the joint decision model into local decision models.
- A multi-agent centralized online planning algorithm, called factored belief based variable eliminated Monte Carlo planning algorithm is proposed, by extending the MCTS method and the variable elimination algorithm. MCTS is a best-first search algorithm that can effectively solve the long-horizon planning problem of single agent. The variable elimination algorithm [23] is an optimal algorithm for solving one-shot case of multiple agents. The innovation of our proposed algorithm is to construct a local look-ahead tree for each agent, and to synchronize the actions at specific locations of each local search tree, so as to obtain the global optimal policy.

II. PROBLEM STATEMENT

In this section, a general formalization of multi-agent patrolling problem is introduced, which is inspired by [12].

A. PHYSICAL ENVIRONMENT

Definition 1 (Layout graph): The layout graph is defined as an undirected graph $G = (V, E)$, where V represents a set of spatial vertices embedded in Euclidean space, and E denotes a set of edges. Let the number of vertices in G be $|V|$.

The layout graph defines the layout of environment and the motion form of agents. In disaster response scenarios, a vertex represents a target area of interest to rescuers, such as communities, schools, and factories. The feasible traversable area between a pair of adjacent target areas is captured as an edge, that is, the path that agents move.

Definition 2 (Time): Time is encoded as a discrete set of temporal coordinates, denoted as $t \in \{0, 1, 2, \ldots\}$.

In each time step, the environmental information state changes once, and each agent completes an Observe-Orientate-Decide-Act loop. The actual time that a time step corresponds to is determined by the real scenario. If the UAV can reach the target area within 10 minutes and complete a loop, then a time step can be set as 10 minutes.

Definition 3 (Information State): The information state qualitatively represents the amount of new information at the vertex.

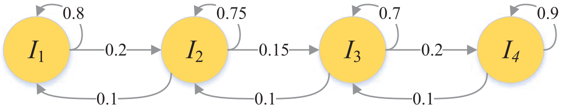


FIGURE 1. An example of the information state model represented by the Markov chain.

The environmental information is characterized as the discrete information state, which is shown as Fig. 1. The information state set has several information levels, denoted as $I = \{I_1, I_2, \ldots, I_N\}$, where I_n indicates the n-th information level and N is the size of the set. As the information level increases, the vertex has more unknown information. The information value is a quantitative representation of the information state, and the information value set is denoted as $F = \{F_1, F_2, \ldots, F_N\}$. The information value is computed by the information value function.

Definition 4 (Information Value Function): The information value function f is defined as a set function $f : I \to \mathbb{R}^+$, that assigns the information value to the information state.

In general, as the information level increases, the vertex has more unknown information. So we assume that the information value function is monotonically non-decreasing, i.e. $F_1 \leq F_2 \leq \ldots \leq F_N$. The information value function encodes the known priori information about the temporal and spatial characteristics of the environment, such as the type of phenomenon being monitored, and the speed at which the phenomenon changes. This definition ensures the generality of our model as it can vary notably relying on the characteristics of the environment [9].

For the environmental dynamics, we relax the condition that the information state of all vertices is independently subject to the discrete-time multi-state Markov chain. Specifically, the information state transition matrix is as follows.

$$P = \begin{pmatrix} p_{11} & p_{12} & \cdots & p_{1N} \\ p_{21} & p_{21} & \cdots & p_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ p_{N1} & p_{N2} & \cdots & p_{NN} \end{pmatrix} = \begin{pmatrix} P_1 \\ P_2 \\ \vdots \\ P_N \end{pmatrix} \tag{1}$$

where p_{ij} represents the transition probability from I_i to I_j. Before dispatching UAVs, prior information about the target area is collected from different sources. Based on the priori information, a statistical model of the information state transition matrix is computed through the machine learning technique [24]. In this paper, we assume that (1) is known in advance, and our work is based on these prior knowledge. In fact, our online planning algorithm can re-adjust the schedule for agents based on the new statistical model.

B. INFORMATION GATHERING AGENT

Definition 5 (Information Gathering Agent): The information gathering agent (agent for short) is a mobile autonomous entity patrolling in graph G, that directs its activities to gather information.

At any time step t, each agent is at a certain vertex in G. Agents can visit the same vertex at the same time.

Definition 6 (Patrol Area): Each agent $m_i \in M$ patrols in a subgraph $G_i = (V_i, E_i) \subseteq G$.

The patrol areas may overlap each other. Agent m_i moves on the vertices and edges in its patrol area G_i, and its movement occurs between two consecutive time steps. The example of patrol areas for agents is shown in Fig. 2.

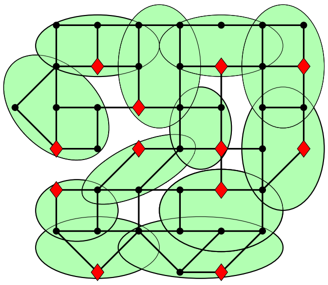


FIGURE 2. An example of the twelve-agent patrolling problem, where red diamonds represent agents, and green ellipses represent patrol areas of agents.

Definition 7 (Neighbor): The neighbor of agent m_i is a set of agents whose patrol areas overlap with the patrol area of agent m_i, denoted as $Ne_i \subseteq M$.

Due to the overlapping of patrol areas, agent m_i may be affected by its neighbors. In general, each agent has little impact on the other agents, that is, the team of agents has weakly-coupled structure. The weakly-coupling is a qualitative concept in this paper which means that the relative coupling degree tends to the weak end of the coupling spectrum. We regard that agent m_i belongs to its neighbors, i.e. $m_i \in Ne_i$. When an agent moves to a vertex, it can automatically gather the information of the vertex. At the same time, the information state of the vertex is reset to I_1, which means that there is no new information at the position currently. In addition, the perception of the agent is limited, which can only observe information of its current vertex at the moment.

III. FACTORED MPOMDP FORMULATION

The multi-agent patrolling problem is cast as the factored MPOMDP formulation, which is defined as a tuple $\langle M, S, A, O, \delta, Z, R, D, \gamma, B\rangle$. Without loss of generality, we take agent m_i as an example, where formulations of other agents are the same as that of agent m_i.

- $M = \{m_1, m_2, \ldots, m_K\}$ is a set of agents, where K is the number of agents.
- S is a set of states, factored into the information states of vertices and the position states of agents, denoted as $S = [S^V, S^I]$. The local state of agent m_i is a subset of the global state, denoted as $s_i = [s^V_{Ne_i}, s^I_{G_i}]$, where $s^V_{Ne_i}$ indicates the position states of its neighbors and $s^I_{G_i}$ indicates the information states of all the vertices in its patrol area G_i.
- $A = \{A_1, A_2, \ldots, A_K\}$ is the set of joint actions. The action of agent m_i is denoted as $a_i \in A_i$, and the joint action of its neighbors is denoted as $a_{Ne_i} \in A_{Ne_i}$. Specifically, the action of agent m_i is the movement from its current position v to one of the adjacent vertices $adj_{G_i}(v)$ in G_i, where $v \in adj_{G_i}(v)$.
- $O = \{O_1, O_2, \ldots, O_K\}$ is the set of joint observations. The observation of agent m_i is denoted as $o_i \in O_i$. The position state of agents is completely observable, and agent m_i can only observe the information state of its current position at the moment, i.e. $o_i = [s^V_{Ne_i}, s^I_i]$.
- δ is the set of joint state transition probabilities, denoted as $\delta(s(t+1)|s(t), a(t)) = \prod_{i=1}^K \delta^I_i(s^I_{G_i}(t+1)|s^I_{G_i}(t)) \cdot \delta^V_i(s^V_i(t+1)|s^V_i(t), a_{Ne_i}(t))$, where δ^I_i is the local information state transition probability of agent m_i, which follows the multi-state Markov chain. The local position state transition probability of agent m_i is as follows.

$$\delta^V_i(s^V_i(t+1)|s^V_i(t), a_{Ne_i}(t)) = \begin{cases} 1, & s^V_i(t+1) = s^V_{goal} \\ 0, & s^V_i(t+1) \neq s^V_{goal} \end{cases} \tag{2}$$

where s^V_{goal} is the target position of agent m_i at $t+1$.
- Z is the set of joint observation transition probabilities, denoted as $Z(o|s,a) = \prod_{i=1}^K Z_i(o_i|s_i, a_{Ne_i})$. Z_i is the local observation transition probability of agent m_i, defined as follows.

$$Z_i(o_i|s_i, a_{Ne_i}) = \begin{cases} 1, & o_i = [s^V_{Ne_i}, s^I_i] \\ 0, & o_i \neq [s^V_{Ne_i}, s^I_i] \end{cases} \tag{3}$$

- R is a decomposable global immediate reward function, which is generated by summing local immediate reward functions of all agents, $R(s,a) = \sum_{i=1}^K R_i(s_i, a_{Ne_i})$. The local reward function R_i of agent m_i is defined as follows.

$$R_i(s_i, a_{Ne_i}) = \frac{f(I_i)}{n_i} \tag{4}$$

where n_i refers to the number of agents visiting the same vertex at the same time as the agent m_i; and I_i is the information state of the vertex that agent m_i visits.
- D is the planning horizon of each agent.
- γ is the discount factor.
- B is a probability distribution over the state, called the belief. It includes the information belief and position belief, denoted as $B = [B^V, B^I]$. The local belief of agent m_i is denoted as B_i. As mentioned above, the observation of position state is completely observable, and we focus on the information belief here. The information state of each vertex changes independently based on (1). Therefore, B^I can be represented as a factored belief [25], which is as follows.

$$B^I = [b^I_1, b^I_2, \ldots, b^I_{|V|}] \tag{5}$$

where $b^I_i = [p^{I_1}_i, p^{I_2}_i, \ldots, p^{I_N}_i]$ indicates the information belief of vertex v_i; $p^{I_n}_i$ denotes the conditional probability when the information state at v_i is I_n, and $\sum_{n=1}^N p^{I_n}_i = 1$. The factored belief greatly reduces the computational and storage complexity. In addition, the agent maintains its local belief during execution by belief updates, denoted as $B(t+1) = T(B(t))$. Without loss of generality, the information belief of the vertex v_i is updated as follows.

$$b^I_i(t+1) = \begin{cases} ll\Lambda \cdot P, & v_i = v_{cur} \\ b^I_i(t) \cdot P, & v_i \neq v_{cur} \end{cases} \tag{6}$$

where $\Lambda = [1, 0, 0, \ldots]$ is the unit vector with N elements, and v_{cur} denotes the vertex visited by any agent at the t.

IV. CENTRALIZED ONLINE PLANNING

In this section, the factored belief based variable eliminated Monte Carlo planning (FB-VEMCP) algorithm is first introduced to solve the factorized MPOMDP formulation. Then, the variable elimination based decision coordination (VE-DC) algorithm is proposed. Third, the performance of the FB-VEMCP algorithm is analyzed.

Our objective is to generate optimal policy $\pi = [\pi_1, \pi_2, \ldots, \pi_K]$ to maximize the global value function.

Definition 8 (Global value function): The global value function V^π is the expectation of the discounted summation of global rewards given that agents adopts the joint policy π:

$$V^\pi = E_\pi[\sum_{d=0}^{D-1} \gamma^d R(s(d), a(d))] \tag{7}$$

Based on (4), the joint value function $V^\pi = \Sigma_i V^{\pi_i}_i$ is the summation of local value functions of all agents [26]. The local value function $V^{\pi_i}_i(h_i)$ of agent m_i is as follows.

$$V^{\pi_i}_i(h_i) = E_{\pi_i}[\Sigma_{d=0}^{D-1} \gamma^d R_i] \tag{8}$$

where h_i is the local history of agent m_i, which consists of a sequence of actions and observations, i.e., $h_i(t) = \{a_{Ne_i}(0), o_i(0), \ldots, a_{Ne_i}(t), o_i(t)\}$.

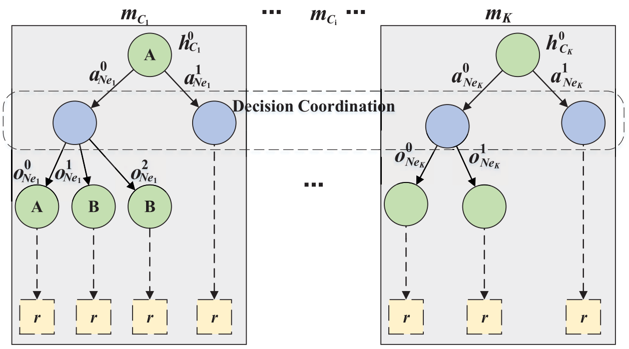


FIGURE 3. The factored look-ahead trees for all agents. Green cycles are *h* nodes and blue cycles are *ha* nodes.

Algorithm 1 FB-VEMCP Algorithm

```
1  procedure Search($h_i$)
2  begin
3      $s_i^V \sim B_i^V(h_i)$;
4      while termination is not meet do
5          $s_{G_i}^I \sim B_i^I(h_i)$;
6          $s_i \leftarrow [s_i^V, s_{G_i}^I]$;
7          $\overline{\psi}_i, \overline{R}_i \leftarrow$ ;
8          $\overline{\psi}_i(0) \leftarrow \psi_i(h_i)$;
9          $Simulation(s_i, h_i, 0, \overline{\psi}_i, \overline{R}_i)$;
10     $a^* \leftarrow DecisionCoordination1(\mathcal{T}(h_i))$;
11     $o_i^* \xleftarrow{a^*_{Ne_i}} s_i^V$;
12     $h_i \leftarrow h_i a_i^* o_i^*$;
13     return $a_i^*$
14 procedure Simulation($s_i, h_i, d, \overline{\psi}_i, \overline{R}_i$)
15 begin
16     if $d \geq D$ then
17         return 0
18     if $h_i \notin \mathcal{T}_i$ then
19         for $a_{Ne_i} \in A_{Ne_i}$ do
20             $(B_i(h_i a_{Ne_i}), \overline{\psi}_i(d+1)) \xleftarrow{a_{Ne_i}} T(B_i(h_i), \overline{\psi}_i(d))$;
21             $\mathcal{T}_i(h_i a_{Ne_i}) \leftarrow$
               $\langle N_{init}, V_{init}, B_i(h_i a_{Ne_i}), \overline{\psi}_i(d+1)\rangle$;
22         return $Rollout(s_i, h_i, depth)$;
23     if $d = flag$ then
24         $a^* \leftarrow DecisionCoordination2(\mathcal{T}_i(h_i), d)$;
25         else $a^* \leftarrow \pi_{temp}(d)$;
26     $(s_i', o_i^*, R_i) \sim G(s_i, a^*_{Ne_i})$;
27     $\overline{R}_i(d) \leftarrow R_i$;
28     $R_i' \leftarrow \gamma \cdot Simulation(s_i', h_i a^*_{Ne_i} o_i^*, d+1, \overline{\psi}_i, \overline{R}_i)$;
29     $Count \leftarrow 1$;
30     for $k = 0 \rightarrow D-1$ do
31         if $\varphi_i(d) = \varphi_i(k)$ then
32             $R_i \leftarrow R_i + \overline{R}_i(k)$;
33             $Count \leftarrow Count + 1$;
34     $R_i' \leftarrow R_i' + \frac{R_i}{Count}$;
35     $N(h_i) \leftarrow N(h_i) + 1$;
36     $N(h_i a^*_{Ne_i}) \leftarrow N(h_i a^*_{Ne_i}) + 1$;
37     $V(h_i a^*_{Ne_i}) \leftarrow V(h_i a^*_{Ne_i}) + \frac{R_i' - V(h_i a^*_{Ne_i})}{N(h_i a^*_{Ne_i})}$;
38     return $R_i'$;
39 procedure Rollout($s_i, h_i, d$)
40 begin
41     if $d \geq D$ then
42         return 0
43     $a_{Ne_i} \sim \pi_{rollout}(h_i, \cdot)$;
44     $(s_i', o_i, R_i) \sim G(s_i, a_{Ne_i})$;
45     return $R_i + \gamma \cdot Rollout(s_i', h_i a_{Ne_i} o_i, d+1)$
```

A. FB-EAMCP ALGORITHM

Fig. 3 shows the factored look-ahead trees for all agents. In FB-EAMCP, the global look-ahead tree is factored into multiple local look-ahead trees, and each agent constructs a local look-head tree in parallel, where actions of all agents are synchronized when selecting the optimal *ha* node from its parent *h* node for each depth in each look-ahead tree, and *ha* represents the history of agents, i.e. $h(t)a(t+1) = \{a(0), o(0), \ldots, a(t), o(t), a(t+1)\}$. For the weakly-coupled multi-agent system, the decomposition greatly reduces the number of branches in each look-ahead tree. In the continuous patrolling problem, there are two relationships between two *h* nodes in a trajectory: node *A* in different depths represents that information beliefs of the current positions of two nodes are the same; node *A* and node *B* represent that information beliefs of the current positions of two nodes are different.

Each node in the local look-ahead tree is a tuple with $\langle N_i, V_i, B_i, \psi_i \rangle$, where N_i is the number of visiting, V_i is the value function, B_i is the belief state, and ψ_i is the number of transitions of the information belief which helps to simplify the calculation. Based on (1) and (6), the information belief is related to the number of transitions. The update formulation for ψ_i is as follows:

$$\psi_i(t+1) = \begin{cases} ll0, & v_i = v_{cur} \\ \psi_i(t) + 1, & v_i \neq v_{cur} \end{cases} \tag{9}$$

Algorithm 1 denotes the FB-EAMCP algorithm. Without loss of generality, we take agent m_i as an example, where planning algorithms of other agents are the same as Algorithm 1. In the procedure *Search* (lines 1-13), agent m_i samples hidden states $s_i = [s^V_{Ne_i}, s^I_{G_i}]$ based on $B_i(h_i)$ at the root node (line 3 and line 5), where '$\sim$ ' represents a sampling of the belief state. After completing sampling (lines 5-6) and initializing vectors $\overline{\psi}_i, \overline{R}_i$ (lines 7-8), it cooperates with neighbors to compute the optimal action through the procedure *DecisionCoordination*1 (line 10). As mentioned above, the observation is the current state according to (3). In particular, the observation of the information state is directly reflected in the immediate reward R_i (line 26). The position state transition is deterministic according to (2) (line 11). The observation of the position state is used to construct the next node (line 12). This greatly reduces the number of child nodes of the *ha* node.

In the procedure *Simulation* (lines 14-38), if h_i is a new node, then the node is added to the tree $\mathcal{T}_i$ by initialing parameters of the $h_i a_{Ne_i}$ node, i.e. the initial visiting count N_{init}, the initial local value function V_{init}, the local belief B_i, and the count vector ψ_i (lines 18-22). Specifically, the overall initial visiting count of the h_i node is $N_{init}(h_i) = \sum_{a_{Ne_i}} N_{init}(h_i a_{Ne_i})$. The information belief B_i^I is updated according to (6), and the count vector ψ_i is updated according to (9) (line 20). In addition, if h_i is not a new node, then the simulator G draws a sample (lines 26) after selecting the optimal action a^* by the procedure *DecisionCoordination*2 or π_{temp} (lines 23-25). The global variable π_{temp} computed by Algorithm 2 is a set of policies, that records temporary actions of all the agents from $depth = 0$ to $depth = D-1$. Specifically, different local look-ahead trees may locate in different depths when executing the procedure *DecisionCoordination*2, so a global variable *flag* is used to record the coordinated depth. If *depth* is equal to *flag*, then a^* is selected by the procedure *DecisionCoordination*2; otherwise $\pi_{temp}(depth)$ is assigned to a^*. The variable π_{temp} and *flag* are initialized to $\emptyset$ and 0 respectively. After searching and expending the tree, it updates the corresponding variables (lines 29-37). In order to improve the accuracy of the evaluation, it detects whether or not there are vertices with the same information belief in a trajectory (lines 30-34). In the procedure *Rollout* (lines 39-45), the potential long-term reward is estimated by using random simulations (line 43).

B. VE-DC ALGORITHM

All the agents run FB-VEMCP algorithm in parallel. Two situations are required to coordinate actions to maximize $Q(s, a) = \sum_i Q_i(s_i, a_{Ne_i})$, where $Q_i(s_i, a_{Ne_i})$ is a local utility function of agent m_i. Here, $Q(s, a)$ is denoted as $Q(a)$ for short.

The first situation is that the optimal action is selected to perform after searching and expanding the look-ahead tree through the procedure *DecisionCoordination*1. The variable $Q_i(a_{Ne_i})$ is as follows.

$$Q_i(a_{Ne_i}) = V_i(h_i a_{Ne_i}) \tag{10}$$

The second situation is that the optimal action is selected to search the tree through the procedure *DecisionCoordination*2. The variable $Q_i(a_{Ne_i})$ is based on upper confidence bounds [27], which is as follows.

$$Q_i(a_{Ne_i}) = V_i(h_i a_{Ne_i}) + c\sqrt{\frac{\log(N(h_i)+1)}{N(h_i a_{Ne_i})+1}} \tag{11}$$

It seems intractable to compute the optimal actions that maximize $Q(a)$, as it requires enumerating the joint action space of all agents. Fortunately, by exploiting the cooperation graph implicit in $Q(a)$, we can very efficiently compute the optimal action a^*. Fig. 4 shows the cooperation graph of twelve-agent patrolling problem.

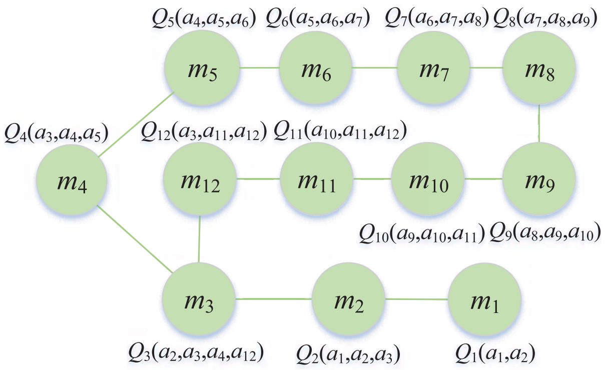


FIGURE 4. **The coordination graph for twelve-agent patrolling problem. The connection between two agents indicates that their patrol areas are overlapped and their actions can affect each other.**

Algorithm 2 denotes the VE-DC algorithm. In the procedure *DecisionCoordination*2 (lines 9-20), it is first in a wait state until all agents enter the procedure (line 11). Let $\hat{Q}$ be the set of local utility functions (line 12). It then computes utility functions of all agents based on (11) (lines 13-15). After that, the optimal actions of all agents at current state are selected through the procedure *Coordination* (line 16). Third, the variable π_{temp} and *flag* are updated (lines 16-19). In addition, the procedure *DecisionCoordination*1 (lines 1-8) is similar to the *DecisionCoordination*2. However, it computes utility functions of all agents based on (10) (lines 5-7).

In the procedure *Coordination* (lines 21-35), the core idea is that, instead of maximizing $Q(a)$ directly, one agent's action is maximized at a time. When maximizing over a_i, only summands of utility functions involving a_i participate in the maximization. We give some definitions here. Let $\hat{C}$ denote the set of temporary utility functions; let C_i denote the temporary utility function with index i; let Ce_i denote the index set of temporary utility functions in $\hat{C}$ involving a_i; let a_{Ce_i} denote the joint action that affects C_i; and let $a_{Ce_i \setminus i}$ denote the joint action, whose elements are equal to the elements in a_{Ce_i} except a_i.

First, it chooses an action a_i that has not been eliminated (line 24) and builds a new joint action a_{Ce_i}. The elements in a_{Ce_i} include all the elements in $a_{Ce_j}, j \in Ce_i$ and $a_{Ne_k}, k \in Ne_i$. After numerating $a_{Ce_i} \in A_{Ce_i}$ and assigning a_{Ce_i} to the corresponding elements in a_{Ce_j} and a_{Ne_k}, the utility function $C_j(a_{Ce_j})$ and $Q_k(a_{Ne_k})$ are computed (line 25). Second, it maximizes $C_i(a_{Ce_i})$ to compute the optimal actions a_i' under the constraint of $a_{Ce_i \setminus i}$, denoted as $a_i'(a_{Ce_i \setminus i})$ (line 26). Third, after removing $C_j, j \in Ce_i$ and $Q_k, k \in Ne_i$ from $\hat{C}$ and $\hat{Q}$ separately, it adds $C_i(a_{Ce_i})$ to $\hat{C}$ (lines 27-31). Fourth, it computes the optimal joint action a^* of all agents in the reverse direction (lines 32-34).

C. PERFORMANCE ANALYSIS

The most desirable quality bound is to express performance relative to POMCP. The optimality and convergence of

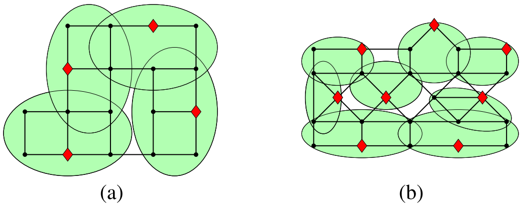
Algorithm 2 VE-DC Algorithm

```
procedure DecisionCoordination1($\mathcal{T}_i$)
begin
    $WaitforAllTrees$;
    $\hat{Q} \leftarrow \emptyset$;
    for $m_i \in M$ do
        $Q_i(a_{Ne_i}) \leftarrow V_i(h_i a_{Ne_i})$;
        $\hat{Q} \leftarrow \hat{Q} \cup Q_i(a_{Ne_i})$;
    return $Coordination(\hat{Q})$
procedure DecisionCoordination2($\mathcal{T}_i, depth$)
begin
    $WaitforAllTrees$;
    $\hat{Q} \leftarrow \emptyset$;
    for $m_i \in M$ do
        $Q_i(a_{Ne_i}) = V_i(h_i a_{Ne_i}) + c\sqrt{\frac{\log(N(h_i)+1)}{N(h_i a_{Ne_i})+1}}$;
        $\hat{Q} \leftarrow \hat{Q} \cup Q_i(a_{Ne_i})$;
    $\pi_{temp}(depth) \leftarrow Coordination(\hat{Q})$;
    if $depth < D - 1$ then
        $flag \leftarrow depth + 1$;
        else $flag \leftarrow 0$;
    return $\pi_{temp}(depth)$
procedure Coordination($\hat{Q}$)
begin
    $\hat{C}, a^* \leftarrow \emptyset$;
    for $i = 1 \rightarrow K$ do
        $C_i \leftarrow \sum_{j \in Ce_i} C_j(a_{Ce_j}|a_{Ce_i}) + \sum_{k \in Ne_i} Q_k(a_{Ne_k}|a_{Ce_i})$;
        $a'_i(a_{Ce_i \backslash i}) \leftarrow \max_{a_i} C_i(a_{Ce_i})$;
        for $j \in Ce_i$ do
            $\hat{C} \leftarrow \hat{C} \setminus C_j$;
        for $k \in Ne_i$ do
            $\hat{Q} \leftarrow \hat{Q} \setminus Q_k$;
        $\hat{C} \leftarrow \hat{C} \cup C_i(a_{Ce_i})$;
    for $i = K \rightarrow 1$ do
        $a^*_i \leftarrow a'_i(a_{Ce_i \backslash i}|a^*)$;
        $a^* \leftarrow a^* \cup a^*_i$
    return $a^*$
```

FB-EAMCP depend on two aspects: variable elimination algorithm and MCTS. For the one-shot case, the variable elimination algorithm can compute the global optimal action in the current situation [28]. In other words, in the process of searching and expanding the look-ahead tree, the effect of action selections in FB-EAMCP is equivalent to that in POMCP. On this basis, the optimality and convergence of MCTS for online planning in partially observable scenarios has been established in [20], which can be extended to FB-EAMCP directly. That is, as long as sufficient samples are drawn from the true belief, the global value function computed by FB-EAMCP will converge in probability to the optimal global value function. Therefore, FB-EAMCP has the same optimality and convergence as POMCP. In addition, FB-EAMCP is suitable for multi-agent systems with weakly-coupled structures. As the cost of the variable elimination algorithm is exponential in the induced width of the coordination graph [23].

V. EMPIRICAL EVALUATION

A response scenario after the earthquake disaster is taken into consideration (see [29] for details). In this paper the research is conducted in a high level and the response scenario is modeled based on our proposed problem formulation. After collecting and preprocessing prior information, a team of UAVs is dispatched to the target area. Then the team of UAVs continues to patrol their specified areas to gather information to assist in subsequent rescue missions, such as distributing food, excavating victims from the rubble, extinguishing fire and providing medical support. To test the performance of FB-VEMCP, the graphs shown in Fig. 2 and Fig. 5 are used to model typical patrol problems.



(a)

(b)

FIGURE 5. **Patrol areas for four agents and eight agents.**

Given this, FB-VEMCP is compared with POMCP, TD-FMOP, Dec-MCTS, and VE-POMCP. The POMCP algorithm is the most advanced general online planning algorithm. Given the true belief, actions computed by POMCP can converge to the optimal actions for any limited horizon POMDP problem. The transition-decoupled factored belief based Monte Carlo online planning (TD-FMOP) algorithm is a decentralized online planning algorithm that combines MCTS with the max-sum algorithm. Although TD-FMOP can guarantee the optimality for the acyclic factor graph, it cannot provide optimal guarantee when the loop exists. Dec-MCTS is a decentralized multi-robot online planning algorithm, where robots periodically communicate a compressed form of trees, which are used to update the joint distribution using a distributed optimization approach. VE-POMCP is an extension of FB-VEMCP, the difference between them is the way of updating beliefs. FB-VEMCP updates beliefs based on (6), while VE-POMCP maintains beliefs through the particle filtering, which can be applied to problems that are difficult to express beliefs with explicit probability distribution.

Each algorithm runs 50 time steps in a round, and runs 30 rounds for each scenario. For all scenarios, let the discount factor γ be 0.9, and let the coefficient c in (11) be 2. The information value set is $F = \{0, 1, 2, 3\}$, corresponding to the information state set $I = \{I_1, I_2, I_3, I_4\}$. The information state transition is shown in Fig. 1. The average total reward of each round and the average time of each decision making are used to evaluate the performance of each algorithm. These experiments run on a machine with 2.5 GHz dual-core CPU and 8 GB RAM.

A. EVALUATION OF SCALABILITY

We benchmark FB-VEMCP against POMCP, TD-FMOP, Dec-MCTS, and VE-POMCP to empirically assess the scalability of algorithms, and each algorithm runs 500 simulations in each scenario. Three scenarios are constructed, the three environments are used to evaluate the scalability of the proposed approach. which are as follows.

- *Scenario A*1: As shown in Fig. 5(a), the graph consists of 18 vertices and 25 edges. Four agents are allocated to the designated areas. Each agent with horizon 6 has about 3 neighbors.
- *Scenario A*2: As shown in Fig. 5(b), the graph consists of 25 vertices and 44 edges. Eight agents are allocated to the designated areas, and each agent with horizon 6 has about 3 neighbors.
- *Scenario A*3: As shown in Fig. 2, the graph consists of 44 vertices and 66 edges. Twelve agents are allocated to the designated areas, and each agent with horizon 6 has about 3 neighbors.

TABLE 1. Results (Scenario A1).

Algorithm	TD-FMOP	FB-VEMCP	VE-POMCP
Value	127.67	140.40	132.67
Time(s)	31.81	35.82	37.02
Algorithm	Dec-MCTS	POMCP	Random
Value	132.93	128.8	109.63
Time(s)	51.50	0.10	–

Negligible time is indicated by −.

The average rewards and standard deviations for the four-agent patrolling problem are shown in Table 1. The average reward of FB-VEMCP is 9.97% higher than that of TD-FMOP, and the average reward of VE-POMCP is 3.92% higher than that of TD-FMOP. Although the average runtime of each decision making of POMCP is much lower than that of other algorithms (except the random algorithm), it gets the lowest average reward in these algorithms.

Similar results are seen in the eight-agent patrolling problem, which are shown in Tabble 2. FB-VEMCP gets the highest average reward in these algorithms, which is 4.87% larger than that of TD-FMOP, while FB-VEMCP and TD-FMOP have similar average runtime. Moreover, FB-VEMCP produces a higher value than that of VE-POMCP, Dec-MCTS, and POMCP. Although POMCP achieves a very low runtime, the average reward is much lower than other algorithms.

Table 3 shows the results of the twelve-agent patrolling problem. It is out of memory when conducting POMCP. Specifically, FB-VEMCP exceeds TD-FMOP, VE-POMCP, Dec-MCTS and the random algorithm by 8.21%, 4.89%, 7.27%, and 22.78% separately. In addition, FB-VEMCP achieves the lowest runtime in these algorithms (except the random algorithm).

TABLE 2. Results (Scenario A2).

Algorithm	TD-FMOP	FB-VEMCP	VE-POMCP
Value	202.76	212.63	197.10
Time(s)	62.33	63.28	63.42
Algorithm	Dec-MCTS	POMCP	Random
Value	189.56	147.67	182.23
Time(s)	66.16	4.97	–

Negligible time is indicated by −.

TABLE 3. Results (Scenario A3).

Algorithm	TD-FMOP	FB-VEMCP	VE-POMCP
Value	332.23	359.50	335.13
Time(s)	101.34	100.79	103.28
Algorithm	Dec-MCTS	POMCP	Random
Value	342.73	–	292.80
Time(s)	242.21	–	–

Memory overflow or negligible time is indicated by −.

These results clearly illustrate that FB-VEMCP can achieve a high reward in a reasonable time. It contributes to the correct choice when computing the optimal action after searching and expanding the look-ahead tree, and the usage of continuous patrol characteristics, allowing each node to have a more accurate assessment with the same number of samples. In weakly-coupled agents, each look-ahead tree in FB-VEMCP has lower branching factor than that in POMCP. Thus, FB-VEMCP performs better than POMCP with a small number of simulations. In addition, the max-sum algorithm in TD-FMOP cannot guarantee the optimality for the acyclic factor graph, and different look-ahead trees may be at different depths in the coordination process in TD-FMOP. Dec-MCTS is a decentralized algorithm, where its reward and runtime are affected by the number of interactions. Therefore, it leads to FB-VEMCP slightly outperforming TD-FMOP and Dec-MCTS in these scenarios.

B. EVALUATION OF HORIZON

We empirically evaluate the influence of the horizon D in this section. As shown in Fig. 5(a), the graph consists of 18 vertices and 25 edges, and four agents patrol in designed areas. Four scenarios are constructed, and each algorithm runs 1000 simulations in each scenario.

- *Scenario B*1: The horizon is 1 time step.
- *Scenario B*2: The horizon is 3 time steps.
- *Scenario B*3: The horizon is 6 time steps.
- *Scenario B*4: The horizon is 10 time steps.

The average total rewards and standard deviations are tabulated in Fig. 6. Specifically, FB-VEMCP and VE-POMCP have similar rewards in these scenarios. For all scenarios, FB-VEMCP outperforms POMCP slightly, and outperforms

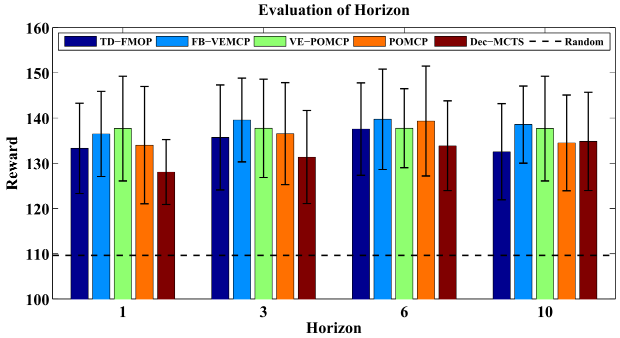


FIGURE 6. **Rewards of the four-agent patrolling problem with different horizons.**

TD-FMOP by 2.40% in *Scenarios B*1, by 2.85% in *Scenarios B*2, by 1.57% in *Scenarios B*3, and by 4.55% in *Scenarios B*4. Additionally, FB-VEMCP exceeds Dec-MCTS ranging from 2.77% to 6.59%, and is at least 24.51% better than the random algorithm in these scenarios.

In these scenarios, as the horizon increases, the rewards does not increase monotonically, and each algorithm achieves the highest rewards in *Scenarios B*3. Because the number of branches in the tree will increase exponentially with the horizon. However, the number of samples is fixed in these scenarios, and it may lead to undersampling of the node in the long horizon planning. Insufficient sampling of the node results in inaccurate evaluation of the value function of the node.

VI. CONCLUSION

In this paper, a novel approach is put forward to solve multi-agent centralized patrolling under dynamic environments. It involves first formalizing the multi-agent patrolling problem as the factored MPOMDP framework, and second proposing an online planning algorithm by extending the MCTS method. In particular, the proposed formulation is a very general model, which is suitable for the centralized patrolling settings and may provide a new idea for the decentralized patrolling settings. The proposed algorithm is empirically compared with some state-of-the-art solvers in typical patrol scenarios. The results showed that the performance of the proposed algorithm is remarkable with a small number of simulations, which is suitable for agents with weakly-coupled structure. Moreover, in practical disaster response applications, the real-time and up-to-date situation awareness provided by teams of UAVs can assist in the decision making of commands. The commands only need to specify tasks for UAVs instead of knowing the details of our algorithm. In general, we provide a foundational step for multi-robot automatic planning before dispatching rescues to risky environments.

VII. CONFLICTS OF INTEREST

The authors declare no conflict of interest.

REFERENCES

[1] H. Zhang, C. Cao, L. Xu, and T. A. Gulliver, "A UAV detection algorithm based on an artificial neural network," *IEEE Access*, vol. 6, pp. 24720–24728, 2018.

[2] A. Kolling, P. Walker, N. Chakraborty, K. Sycara, and M. Lewis, "Human interaction with robot swarms: A survey," *IEEE Trans. Human-Mach. Syst.*, vol. 46, no. 1, pp. 9–26, Feb. 2016.

[3] S. D. Ramchurn *et al.*, "HAC-ER: A disaster response system based on human-agent collectives," in *Proc. AAMAS*, 2015, pp. 533–541.

[4] D. Mcmenemy, G. V. Avvari, D. Sidoti, A. Bienkowski, and K. R. Pattipati, "A decision support system for managing the water space," *IEEE Access*, vol. 7, pp. 2856–2869, 2019.

[5] X. Zhou, W. Wang, T. Wang, X. Li, and T. Jing, "Continuous patrolling in uncertain environment with the UAV swarm," *PLoS ONE*, vol. 13, no. 8, 2018, Art. no. e0202328.

[6] H. H. Kang, S. S. Lee, S. H. You, and C. K. Ahn, "Finite memory output feedback control for unmanned aerial vehicle," *IEEE Access*, vol. 6, pp. 47397–47407, 2018.

[7] M. Li and Y. Chen, "Robust tracking control of networked control systems with communication constraints and external disturbance," *IEEE Trans. Ind. Electron.*, vol. 64, no. 5, pp. 4037–4047, May 2017.

[8] S. Gao, R. Song, and Y. Li, "Cooperative control of multiple nonholonomic robots for escorting and patrolling mission based on vector field," *IEEE Access*, vol. 6, pp. 41883–41891, 2018.

[9] R. Stranders, E. M. Cote, A. Rogers, and N. R. Jennings, "Near-optimal continuous patrolling with teams of mobile information gathering agents," *Artif. Intell.*, vol. 195, pp. 63–105, Feb. 2013.

[10] A. Farinelli, A. Rogers, A. Petcu, and N. R. Jennings, "Decentralised coordination of low-power embedded devices using the max-sum algorithm," in *Proc. AAMAS*, 2008, pp. 639–646.

[11] G. Best, M. Forrai, R. R. Mettu, and R. Fitch, "Planning-aware communication for decentralised multi-robot coordination," in *Proc. ICRA*, vol. 21, 2018, pp. 1050–1057.

[12] S. Chen, F. Wu, L. Shen, J. Chen, and S. D. Ramchurn, "Decentralized patrolling under constraints in dynamic environments," *IEEE Trans. Cybern.*, vol. 46, no. 12, pp. 3364–3376, Dec. 2016.

[13] D. Ye, M. Zhang, and A. V. Vasilakos, "A survey of self-organization mechanisms in multiagent systems," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 47, no. 3, pp. 441–461, Mar. 2017.

[14] S. Garg and N. Ayanian, "Persistent monitoring of stochastic spatio-temporal phenomena with a small team of robots," in *Proc. RSS*, 2014, pp. 1–10.

[15] T. Patten, R. Fitch, and S. Sukkarieh, "Large-scale near-optimal decentralised information gathering with multiple mobile robots," in *Proc. ACRA*, 2013, pp. 1–10.

[16] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher, "An analysis of approximations for maximizing submodular set functions—I," *Math. Program.*, vol. 14, no. 1, pp. 265–294, 1978.

[17] F. A. Oliehoek, "Interactive learning and decision making: Foundations, insights & challenges," in *Proc. IJCAI*, 2018, pp. 5703–5708.

[18] T. P. Le, N. A. Vien, and T. Chung, "A deep hierarchical reinforcement learning algorithm in partially observable Markov decision processes," *IEEE Access*, vol. 6, pp. 49089–49102, 2018.

[19] S. J. Witwicki and E. H. Durfee, "Influence-based policy abstraction for weakly-coupled Dec-POMDPs," in *Proc. ICAPS*, 2010, pp. 185–192.

[20] D. Silver and J. Veness, "Monte-Carlo planning in large POMDPs," in *Proc. NIPS*, 2010, pp. 2164–2172.

[21] M. U. Chaudhry and J.-H. Lee, "Feature selection for high dimensional data using Monte Carlo tree search," *IEEE Access*, vol. 6, pp. 76036–76048, 2018.

[22] C. Amato *et al.*, "Scalable planning and learning for multiagent POMDPs," in *Proc. AAAI*, 2015, pp. 1995–2002.

[23] C. Guestrin, D. Koller, and R. Parr, "Multiagent planning with factored MDPs," in *Proc. NIPS*, 2002, pp. 1523–1530.

[24] Z. Li and D. Hoiem, "Learning without forgetting," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 12, pp. 2935–2947, Dec. 2017.

[25] S. Chen, F. Wu, L. Shen, J. Chen, and S. D. Ramchurn, "Multi-agent patrolling under uncertainty and threats," *PLoS ONE*, vol. 10, no. 6, 2015, Art. no. e0130154.

[26] S. J. Witwicki, "Abstracting influences for efficient multiagent coordination under uncertainty," Ph.D. dissertation, Dept. Comput. Sci. Eng., Univ. Michigan, Ann Arbor, MI, USA, 2011.

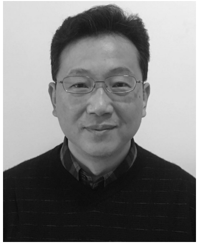
[27] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Mach. Learn.*, vol. 47, no. 2, pp. 235–256, 2002.

[28] C. Guestrin, M. Lagoudakis, and R. Parr, "Coordinated reinforcement learning," in *Proc. ICML*, vol. 2, 2002, pp. 227–234.

[29] S. D. Ramchurn *et al.*, "A disaster response system based on human-agent collectives," *J. Artif. Intell. Res.*, vol. 57, pp. 661–708, Dec. 2016.

XIN ZHOU received the master's degree in systems simulation from the National University of Defense Technology (NUDT), Changsha, China, where he is currently pursuing the Ph.D. degree in systems engineering. His research interests include systems engineering and simulation, multi-agent decision making under uncertainty, and reinforcement learning.

WEIPING WANG received the Ph.D. degree in systems engineering from the National University of Defense Technology (NUDT), Changsha, China, where he is currently a Professor. His research interest includes systems engineering and simulation.



YIFAN ZHU received the Ph.D. degree in systems engineering from the National University of Defense Technology (NUDT), Changsha, China, where he is currently a Professor. His research interest includes systems engineering and simulation.

TAO WANG received the Ph.D. degree in software engineering from the National University of Defense Technology (NUDT), Changsha, China, where he is currently an Associate Professor. His research interests include systems engineering and simulation, multi-agent decision making under uncertainty, and data mining.

BO ZHANG received the Ph.D. in systems engineering from the South China University of Technology, Guangzhou, China. He is currently an Associate Professor with the Guangdong University of Technology, Guangzhou. His research interests include the control of memristive systems, stability and control of stochastic systems, and synchronization of chaotic systems.