2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

POAQL: A Partially Observable Altruistic Q-Learning Method for Cooperative Multi-Agent Reinforcement Learning

Lesong Tao, Miao Kang, Jinpeng Dong, Songyi Zhang, Ke Ye, Shitao Chen and Nanning Zheng†

Abstract— **Multi-Agent Path Finding (MAPF) is an important issue in multi-agent cooperation. Many studies apply Multi-Agent Reinforcement Learning (MARL) to solve MAPF in partially observable settings. The objective of cooperative MARL is to maximize the cumulative team reward. Nevertheless, in partially observable settings, the team reward is misleading due to unpredictable factors from the behavior and state of unobserved agents. To address this issue, we propose a Partially Observable Altruistic Q-learning (POAQL) method. POAQL considers the cumulative reward of the observed subteam instead of the whole team, where Altruistic Q-learning plays an important role in learning the subteam action value. In addition, we design a new conflict resolution without additional guidance to emphasize the cooperative nature of MARL frameworks. Experimental results show that POAQL outperforms existing reinforcement learning methods in terms of efficiency and performance.**

I. INTRODUCTION

Cooperative Multi-Agent Reinforcement Learning (MARL) deals with a system of agents interacting within a common environment. Currently, MARL researchers focus on cooperative multi-agent problems with a single reward signal – the team reward. The shared objective of the agents is to maximize the cumulative team reward. In addition, there is another kind of problem in which each agent has its own reward signal – the individual reward, where the team reward is the sum of the individual rewards. The multi-agent pathfinding (MAPF) problem is one of the specific problems with widespread real-world applications, such as video games [1], automated warehouses [2] [3], robotics [4] etc. MAPF requires agents to reach their goal locations without colliding with obstacles or other agents. The objective of MAPF is to plan paths with the minimum sum of time steps for each agent to reach its goal [5]. Following this objective, the cumulative individual reward of an agent is related to the steps it takes. Furthermore, MARL researchers use partial observation to reduce the dimension of the observation space. In this manner, agents only perceive surrounding information [6].

In MARL, team reward motivates agents to cooperate. However, in partially observable settings, team reward is misleading for two main reasons. Firstly, the lazy agent effect arises when agents with poor policies hamper the team's overall performance [7]. Secondly, inconsistencies between the reward and observation occur as the team reward includes individual rewards from unobserved agents, leading to unpredictable team rewards. The unpredictable factors obstruct agents from finding optimal policies, particularly when unobserved agents predominate. To address these issues, researchers propose credit assignments to assist agents in focusing on their contributions [8], in which learning joint action value is necessary. Although credit assignment alleviates these problems, learning joint action value is intractable for numerous agents. Other approaches [9] [10] [11] present extra rewards and mechanisms. However, they rely on expert experience and may not be optimal.

To address these issues, we present a cooperative MARL method called POAQL. This method maximizes the cumulative reward of a subteam consisting of an agent and its neighbors. Since the agent observes all members of its subteam, rewards within the subteam can be predicted. Additionally, considering only neighbors is sufficient for large-scale cooperation based on the unique characteristics of multi-agent pathfinding (MAPF) in partially observable settings: The reward and state transition of unobserved agents are isolated from the subteam. However, updating the subteam action value is challenging since it becomes inconsistent when a member leaves the subteam. To address this, we propose the Altruistic Q-learning method, which converts the subteam action value into the sum of the egoistic and the altruistic action values and updates them respectively. Although the altruistic action value is also invalid for missing members, POAQL updates it with the corresponding egoistic action value. In addition, additional rewards and mechanisms such as "blocking penalties" [9] are redundant since cooperative frameworks implement team rewards to induce agents to cooperate. Hence, we propose a novel mechanism for resolving conflicts in cooperative MARL frameworks.

The main contributions of this paper are 1) A novel MARL framework based on Altruistic Q-learning, which considers the subteam action value to reduce unpredictable factors. 2) A new conflict resolution mechanism for cooperative MARL frameworks in MAPF. 3) A MAPF environment based on warehouse applications for reinforcement learning without relying on expert guidance.

II. RELATED WORK

A. Cooperative Multi-Agent Reinforcement Learning

MARL deals with systems consisting of more than one agent. The action space of multi-agent problems grows exponentially with the number of agents [6]. Existing studies

*This work was supported by the National Natural Science Foundation of China (Grant No. 62088102).

Lesong Tao, Miao Kang, Jinpeng Dong, Songyi Zhang, Ke Ye, Shitao Chen, and Nanning Zheng are with the National Key Laboratory of Human-Machine Hybrid Augmented Intelligence, National Engineering Research Center for Visual Information and Applications, and Institute of Artificial Intelligence and Robotics, Xi'an Jiaotong University

†Corresponding author

reduce the action space by applying distributed decision-making. Based on this idea, one of the classical methods is independent learning including independent Q-Learning (IQL) and independent actor-critic (IAC) [12]. In this method, agents maximize team rewards according to their own observations. However, independent learners find it difficult to distinguish their own contributions, leading to lazy agents that prevent the team from learning optimal policies [7]. Therefore, MARL researchers are searching for methods of credit assignments in which joint action value plays an important role. One of the most popular ideas is based on value decomposition [6]. VDN [7] decomposes the joint action value into the sum of the individual action values. QMIX [13] improves on VDN by using a network to represent the relationship between individual and joint values. Other value decomposition methods include QTRAN [14], ResQ [15], and so on. In addition to the value decomposition, COMA [16] introduces the advantage value calculated by the counterfactual baseline to assign credits.

B. Reinforcement Learning Based Multi-Agent Pathfinding

MAPF problem is one of the most common cooperative multi-agent problems. Some researchers adapt MARL to the MAPF problem. The methods based on independent learning include PRIMAL [9] [10] and DHC [11], in which agents are only rewarded for their own planning results. To make the agents cooperative, these methods introduce the "blocking penalty" [9], a large negative reward for conflict. When agents conflict, all of the conflicting agents are forced to stop and are punished. However, cooperation is highly dependent on the penalty. In addition, there are methods based on credit assignments, such as CM3 [17] inspired by COMA and methods based on QMIX [13]. CM3 adopts two-stage course learning [18] [19] and a specially designed counterfactual baseline. Although credit assignment with joint action value is better in cooperation, the complexity of learning joint action value is higher than independent one.

C. Search-Based Multi-Agent Pathfinding

In addition to MARL-based methods, the MAPF problem can also be solved by search-based methods [20]. These methods usually obtain the single-agent planning by heuristic search first and then obtain the multi-agent planning by screening the single-agent results through a specific mechanism. The representative optimal methods include ICTS [21] and CBS [22] [23]. As the complexity increases with the problem scale, finding a feasible suboptimal solution is efficient. ECBS [24] and EECBS [25] are the variants of CBS with a bounded-suboptimal searching. In addition, prioritized MAPF methods solve the problem efficiently where agents avoid conflicting with prior ones [26] [27]. Although the search-based methods are more stable, they are not as adaptable as reinforcement learning because re-planning is necessary when the environment changes.

III. BACKGROUND

A. Multi-Agent Reinforcement Learning

A multi-agent setting is denoted by a tuple $< N, \mathbb{S}, \mathbb{A}, \mathcal{R}, \mathcal{P}, \mathcal{O}, \gamma >$ [6] [28], in which N is the number of agents, $\mathbb{S} \subseteq \mathbb{S}_1 \times \mathbb{S}_2 \times \cdots \times \mathbb{S}_N$ is joint state space, $\mathbb{A}$ is joint action space which satisfies $\forall \boldsymbol{S} = (S_1, S_2, \cdots, S_n) \in \mathbb{S}, \mathbb{A}(\boldsymbol{S}) \subseteq \mathbb{A}_1(S_1) \times \mathbb{A}_2(S_2) \times \cdots \times \mathbb{A}_N(S_N)$, $\mathcal{R} : \mathbb{S} \times \mathbb{A} \to \mathbb{R}$ is joint reward function where $\mathbb{R}$ is the joint reward space, $\mathcal{O} : \mathbb{S} \to \mathbb{O}$ is the joint observation function where $\mathbb{O}$ is the joint observation space. $\mathcal{P}$ is transition probability between the joint states, which means $\boldsymbol{S}^{t+1} \sim \mathcal{P}(\cdot|\boldsymbol{S}^t, \boldsymbol{A}^t)$. If we do not care about the exact time t, $\boldsymbol{S}^{'}$ is the successor of $\boldsymbol{S}$ and $\boldsymbol{S}^n$ is the nth successor. A sequence of transitions is named a trajectory or an experience denoted as $(\boldsymbol{S}, \boldsymbol{O}, \boldsymbol{A}, \boldsymbol{R}, \boldsymbol{S}^{'}, \boldsymbol{O}^{'}, \boldsymbol{A}^{'}, \boldsymbol{R}^{'}, \cdots)$. The joint policy is denoted as $\boldsymbol{\pi} = (\pi_1, \pi_2, \cdots, \pi_N)$. For each independent learner i, it chooses an action according to its own policy π_i which means $A_i \sim \pi_i(\cdot|\boldsymbol{O}_i)$, and pursues the maximum gain $G_i = \sum_t \gamma^t R_i^t$ where γ is the discount. In cooperative settings, the goal of agents is to maximize the team gain G. The joint action value is defined as $Q^{\boldsymbol{\pi}}(\boldsymbol{O}, \boldsymbol{A}) = E_{\boldsymbol{\pi}}[G|\boldsymbol{O}, \boldsymbol{A}]$, and for an independent learner i, the action value is defined as $Q^{\boldsymbol{\pi}}(\boldsymbol{O}_i, A_i) = E_{\boldsymbol{\pi}}[G_i|\boldsymbol{O}_i, A_i]$.

B. Value-Based Deep Reinforcement Learning

The action value is essential for action selection and policy evaluation. Here we focus on an independent learner i. For the ϵ-greedy action selector, the agent selects the action with the maximum action value with $1 - \epsilon$ probability and a random action with ϵ probability. To find the optimal strategy, the agent updates the action value according to experiences gained from interacting with the environment. The experiences are stored in a replay buffer, and a batch of them is taken out for updates. In deep reinforcement learning, the action value is approximated by neural networks parameterized by θ. For the T-step update, the target value of each experience is calculated by $\hat{Q}(\boldsymbol{O}_i, A_i) = \sum_{t=0}^{T-1} \gamma^t R_i^t + \gamma * Q^{\boldsymbol{\pi}}(\boldsymbol{O}_i^T, A_i^T; \theta)$. The network parameter is updated by minimizing the MSE loss $\mathcal{L}(\theta) = \sum_{(\boldsymbol{O}_i, A_i) \in batch} [\hat{Q}(\boldsymbol{O}_i, A_i) - Q^{\pi}(\boldsymbol{O}_i, A_i; \theta)]^2$, where $(\boldsymbol{O}_i, A_i) \in batch$ means experiences in the batch [29].

C. Multi-Agent Pathfinding

The MAPF problem is a specific multi-agent cooperation problem denoted by a tuple $< N, \mathbb{G}, \mathbb{T} >$ where N is the number of agents, $\mathbb{G}(\mathbb{V}, \mathbb{E})$ is an undirected graph, $\mathbb{V}$ is the vertex set and $\mathbb{E}$ is the edge set, and $\mathbb{T} = \mathbb{V} \times \mathbb{V}$ is the task set where the elements are pairs of start and target vertexes. The goal of MAPF is to find collision-free planning that minimizes the sum of time steps for each agent to complete its tasks [5].

IV. METHOD

In this section, we illustrate how POAQL works. In addition, we present the design of the network structures and observations for the egoistic and altruistic action values along with the action space and reward in the MAPF of

warehouses. Finally, since the "blocking penalty" [9] is superfluous for cooperative frameworks, we introduce a new conflict resolution without it.

A. Partially Observable Altruistic Q-Learning

Each agent maximizes the gain of a subteam consisting of the observed agents. To figure out how an agent makes decisions and learns, we need to answer the following questions: First, how does an agent evaluate the gain of its subteam? Second, how does an agent learn or update the action value of its subteam? Third, under what conditions is global cooperation achieved by considering only the subteam? For the first question, using the subteam action value is intractable due to changing members. An available way is to transform the subteam action value into the sum of egoistic and altruistic action values. Before defining egoistic and altruistic action values, we introduce the components of the local observation. The local observation of the agent i is represented as $\boldsymbol{O}_i = (O_{i,k_1}, O_{i,k_2}, \cdots, O_{i,k_M})$, where $k_m \in \mathbb{K}$, and $|\mathbb{K}| = M \leq N$ is the set of observed agents' indexes. We name $O_{i,i}$ the observation of oneself as egoistic observation, and $O_{i,j}$ the observation of others as altruistic observation (see Fig. 1). For agent i and its neighbor j, the egoistic action value is defined as

$$Q_e^{\boldsymbol{\pi}}(O_{i,i}, A_i) = E_{\boldsymbol{\pi}}[G_i|O_{i,i}, A_i] \tag{1}$$

The definition of the egoistic action value in POAQL is similar to that in single-agent RL. The altruistic action value is defined as

$$Q_a^{\boldsymbol{\pi}}(O_{i,j}, A_i) = E_{\boldsymbol{\pi}}[G_j|O_{i,j}, A_i] \tag{2}$$

Comparing these definitions, we conclude that the actions in both are taken by the agent i, while the gains respectively belong to the agent i and its neighbor j. Noticing that the subteam's gain is the sum of the observed individuals' gains in MAPF, we get the action value of the subteam $Q^{\boldsymbol{\pi}}(\boldsymbol{O}_i, A_i)$

$$Q^{\boldsymbol{\pi}}(\boldsymbol{O}_i, A_i) = Q_e^{\boldsymbol{\pi}}(O_{i,i}, A_i) + \sum_{\substack{j=0 \\ O_{i,j} \in \boldsymbol{O}_i}}^{N} Q_a^{\boldsymbol{\pi}}(O_{i,j}, A_i) \tag{3}$$

where $O_{i,j} \in \boldsymbol{O}_i$ means agent j is observed by agent i and $Q_a^{\boldsymbol{\pi}}(O_{i,i}, A_i)$ is defined as zero. The decomposition in (3) serves two purposes: First, it is easier to handle the varying dimension of the observation. Second, it is easier to update, as we will describe below.

The T-step update of the subteam action value aims to estimate the expectation of gains within the original subteam. However, it is impossible to obtain the gains by accumulating the sum of individual rewards within the subteam, as subteam members may change within T steps. A feasible approach is to update the egoistic and altruistic action values in (3) with their corresponding individual rewards respectively. We mainly discuss the T-step update of the altruistic action value, since the T-step update of the egoistic action value is normal, as described in Section IIIB. The target value of the altruistic action value $\hat{Q}_a(O_{i,j}, A_i)$ depends on whether the subteam member j is observed after T steps. If the member j is observed ($O_{i,j}^T \in \boldsymbol{O}_i^T$) after T steps, POAQL uses the normal target value, which is

$$\hat{Q}_a(O_{i,j}, A_i) = \sum_{t=0}^{T-1} \gamma^t R_j^t + \gamma^T Q_a^{\boldsymbol{\pi}}(O_{i,j}^T, A_i^T) \tag{4}$$

If the member j is unobserved ($O_{i,j}^T \notin \boldsymbol{O}_i^T$) after T steps, we cannot get $Q_a^{\boldsymbol{\pi}}(O_{i,j}^T, A_i^T)$ as $O_{i,j}^T$ is missing. In this case, POAQL uses the egoistic action value j for updates[1], which is

$$\hat{Q}_a(O_{i,j}, A_i) = \sum_{t=0}^{T-1} \gamma^t R_j^t + \gamma^T Q_e^{\boldsymbol{\pi}}(O_{j,j}^T, A_j^T) \tag{5}$$

We replace (4) with (5) for those who have left the subteam since (4) and (5) are equivalent in expectation (both equal to $E_{\boldsymbol{\pi}}[G_j|O_{i,j}, A_i]$). In addition, we store egoistic and altruistic experiences in separate replay buffers for better generalization and efficiency.

Although POAQL ignores the gains of unobserved agents in (3), it approaches the global cooperation based on an important assumption: For most uniformly sampled cases, if agent j is not observed by agent i ($O_{i,j} \notin \boldsymbol{O}_i$), all actions of agent i have nearly equal impact on the gain of agent j, which means $\forall j, O_{i,j} \notin \boldsymbol{O}_i$,

$$E_{\boldsymbol{\pi}}[G_j|\boldsymbol{O}_i, A_i] \approx E_{\boldsymbol{\pi}}[G_j|\boldsymbol{O}_i] \tag{6}$$

Generally (6) holds under two conditions. The first is individual reward and transition are only related to observed agents. The second is long travel for or a low probability of interacting with unobserved agents in the future. MAPF satisfies these conditions: individual reward is only related to actions taken by oneself, and transition is only influenced by nearby agents. Besides, the unobserved agents are usually distant and rarely met. Since the altruistic action value for unobserved agents is constant according to (6), ignoring them in (3) does not affect global cooperation. However, POAQL benefits from it compared with IQL with team reward in which the action value is defined as

$$Q^{\boldsymbol{\pi}}(\boldsymbol{O}_i, A_i) = E_{\boldsymbol{\pi}}[\sum_{j=0}^{N} G_j|\boldsymbol{O}_i, A_i] \tag{7}$$

For unobserved agent j, since $\boldsymbol{O}_i$ contains no information about agent j, agent i cannot infer its trajectory through it. Thus, $E_{\boldsymbol{\pi}}[G_j|\boldsymbol{O}_i, A_i]$ is unpredictable as G_j is closely related to the trajectory of agent j. These unpredictable factors serve as noise that causes agents to make wrong decisions. Therefore, removing them improves the performance of POAQL.

B. Environment and Network Structures

In this part, we present the details of the MAPF environment, including observations, actions, rewards, and network structure of action value.

[1]The egoistic action value j can not be replaced with a fixed value such as 0 which ignores the further impact on agent j

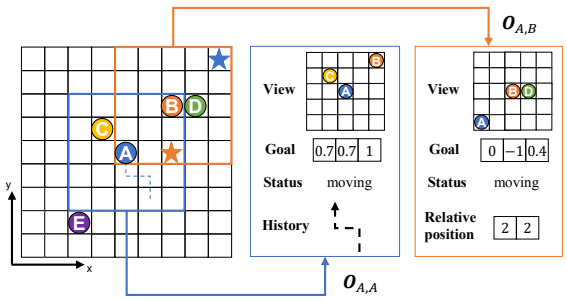

Fig. 1. The structures of observations. The subteam of agent A contains itself and agents B and C. Agents D and E are not in the subteam. Agent A receives the altruistic observations $O_{A,B}$ (presented) and $O_{A,C}$ (not presented).

a) Observation: The observation of an agent includes the egoistic part and the altruistic part. To ensure the consistency between (4) and (5), a principle for the altruistic design is that the trajectory of a subteam member is predictable by corresponding altruistic observations. Therefore, the altruistic observation should contain the principal component of the member's egoistic observation. Here is an instance of the design in MAPF. The egoistic observation consists of a local view, a relative position of the goal, a status, and a history that includes view differences and action history. The altruistic observation consists of a local view, a relative position of the goal, a status, and the relative position between the two agents. The first three components, obtained from the egoistic observation of the nearby agent, are essential for predicting the trajectories. The relative position indicates the specific subteam member. The structures of egoistic and altruistic observations are presented in Fig. 1.

The local view is egocentric and denoted by a square matrix of order W, where $0.0, 0.3, 0.7, 1.0, 2.0$ represent empty spaces, idle agents, loading agents, obstacles, and moving agents, respectively. The relative position of the goal consists of a unit direction vector $\boldsymbol{e}$ and a distance ρ truncated by W. The view difference stores the difference of the local view in a queue of fixed size and ignores the existence of agents. The action history stores previous actions, denoted by a one-hot vector, in a fixed-size queue. The view difference and the action history help agents bypass long obstacles.

b) Action and Reward: In the MAPF environment of PRIMAL and DHC, all agents must stay on their goals to succeed. However, agents can complete their tasks without waiting for others in the warehouse. Therefore, we design an environment suitable for warehouse applications [30]. The action space contains six discrete actions: "up", "down", "left", "right", "stop" and "load" (or "unload"). Invalid actions include those that cause a collision with obstacles and the "load" action when agents are not on their goals. Agents are not allowed to select invalid actions. We introduce the status to record the progress of its task: "moving", "loading" (or "unloading"), and "idle". We set the status to "moving" at the beginning of the task and to "loading" when the agent chooses the "load" action. Agents without tasks are in the "idle" status. During the "loading" status, the agent cannot break from the "load" action, so it stays until the process is finished.

We reward agents with -1 in "moving" status and with 0 otherwise. The team reward is the sum of the individual rewards. -1 for moving follows the objective of MAPF, and 0 for idle highlights the altruism of the agents. In contrast, the reward design of existing methods such as PRIMAL and DHC, which reward agents with 0 for stepping on their goals and -1 (unitized) for other actions, omits the time cost of temporarily stepping on goals. This omission encourages agents to step on goals instead of staying on goals[2]. As a result, it is difficult for agents to complete the task without expert guidance.

c) Network Structure: The egoistic and the altruistic action value networks approximate the action values. The network structures are shown in Fig. 2. Both networks consist of two modules: observation preprocessing and action value prediction. In the observation preprocessing module, the local view is processed by a residual block with two convolutional layers. The history is input only for the egoistic action value network: The view difference is processed by a GRU [32], and the elements in the action history accumulate by discount. The status is used only for the altruistic action value net, as the egoistic action value is zero except for the "moving" status. All the preprocessed features are mapped into equal-length vectors by the linear layers and then concatenated. In the action value prediction module, we use a dueling network [31] to predict the action value. Since the goal distance is truncated, we add a heuristic value to the output for compensation. We only use the Manhattan distance as the heuristic value in our experiments.

C. Conflict Resolution

To teach egoistic agents cooperation, existing methods such as PRIMAL and DHC force conflicting agents to stop with a large negative reward. However, this penalty is not necessary for frameworks using team rewards. As the extra penalty affects the results, we design a new mechanism for cooperative frameworks. This mechanism coordinates a conflicting pair by forcing one to bypass the other. To implement this, we divide the decision-making into four stages: pre-decision, coordination, adjustment, and execution. The flow is shown in Fig. 3. In the pre-decision stage, agents select actions according to their own observations and policies. In the coordination stage, we randomly choose a pair of conflicting agents. Then, we randomly send bans to one of them who has alternative choices. Bans prohibit actions that conflict with the other agent's chosen actions, and they are accumulated and not released until all conflicts are resolved. In the adjustment stage, the agent receiving bans must choose another action. If without choices, the agent is forced to choose the "stop" action. After that, the process returns to the coordination stage until the preselected actions are safe to execute.

[2]Consider two avoidance trajectories where the agent steps on the goal at the beginning: Trajectory A: The agent stays for a long time, but eventually steps out. Trajectory B: The agent steps out for a long time, but eventually returns. The trajectory A is better than the trajectory B in terms of gains. However, the task fails in trajectory A

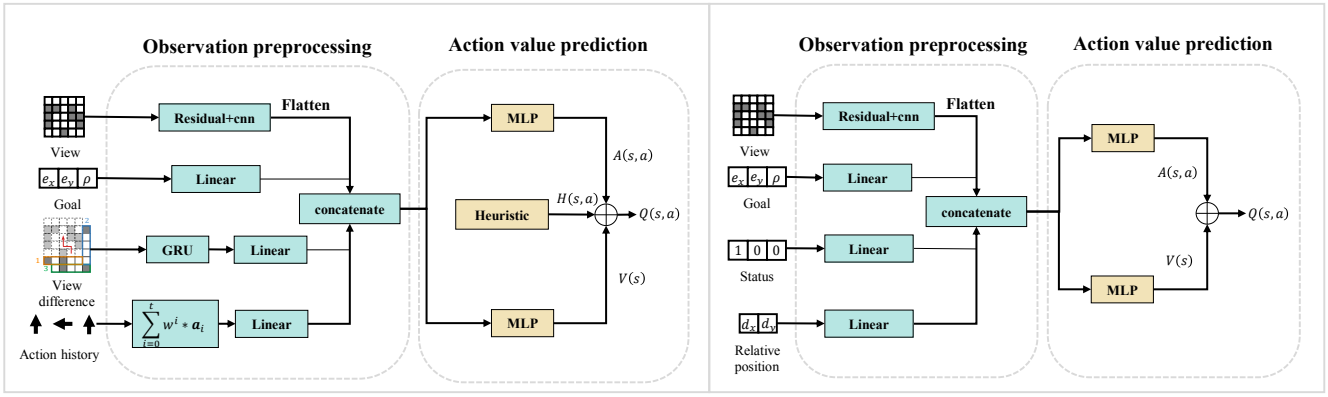

Fig. 2. The structures of the egoistic network (left) and the altruistic network (right). Dueling networks [31] output action values $Q(s,a)$, where s is the state or observation and a is the action. $A(s,a)$ is the advantage value (1×6 sized). $V(s)$ is the state value (1×1 sized). $H(s,a)$ is the heuristic value (1×6 sized).

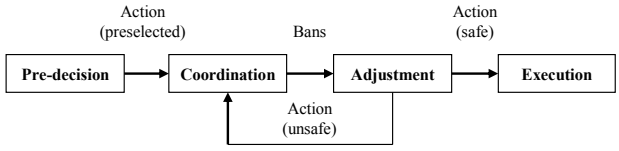

Fig. 3. The flow of conflict resolution.

V. EXPERIMENT

A. Experiment Setup

a) Compared Methods: We compare POAQL with several MARL methods. IQL-team [12] rewards agents for the team; IQL-penalty only uses individual reward along with "block penalty" [9] [11] which punishes conflicting agents with -5 [3] and forces them to stop. VDN, a representative method of credit assignments [6] [4] , additively decomposes the joint action value [7]. The network structures for all compared methods are based on the egoistic one in Fig. 2 and added similar structures for "loading" and "idle" status, which improves their performance. POAQL masks the egoistic action value in (3) for the "idle" status, as idle agents are entirely altruistic.

b) Cases: We randomly generate cases according to a combination of parameters: map size, obstacle density, and number of agents. The map sizes are 10×10, 40×40, 100×100. The obstacle densities are 10%, 20%, 30%. The ratios of the number of agents to the width of the map are 0.5, 1.0, and 2.0. In all cases, agents are assigned only one task. Detailed parameters are listed in Table I.

c) Criteria: In generalization tests, we measure the sum of costs (the sum of time steps required for each task), the complete rate (the ratio of completed tasks to total tasks), and the success rate (success in completing all tasks) over 100 settings. In training, we additionally measure the average fps (frames per second).

[3]We choose this value according to the ratio of conflict reward to step reward in [9] which is between 4.0 and 6.7

[4]We will show from the results that the inefficiency in training is the main problem of decomposition. Therefore, we choose VDN instead of algorithms with more complex decomposition, such as QMIX.

TABLE I

PARAMETERS IN ENVIRONMENTS AND TRAINING

Name	Value	**Name**	Value
field of view	9×9	**length of history**	10
steps for "loading"	3	**episode length**	10×map width
T of T-step	3	**discount γ**	1.0
optimizer	Adam [33]	**learning rate**	1e-4
initial ϵ	0.8	**end ϵ**	0
ϵ decay	1e-5	**batch size**	64

TABLE II

TRAINING PERFORMANCE IN THE 10×10-SIZE MAP

Method	fps↑	sum of costs↓	complete rate↑	success rate↑
POAQL(ours)	**1.56**	**137.2**	**0.98**	**0.84**
VDN[7]	0.13	287.2	0.95	0.58
IQL-team[12]	1.24	174.4	0.95	0.58
IQL-penalty[9]	1.36	243.8	0.88	0.32
POAQL-penalty	1.59	190.5	0.96	0.71

d) Training: We train in the case with 10×10 map size, 20% obstacle density, and 10 agents, which suits for all methods. We randomly generate the map and tasks at the beginning of each epoch. Without any pre-training, we train all methods with decayed ϵ-greedy policy for more than $200k$ steps until performance is stable (variations in complete rate and success rate are less than 1%). Detailed parameters are listed in Table I.

B. Result and Analysis

a) Analysis of Training Results: The final performance is shown in Table II. From the results, POAQL is the most efficient and the best performing. In terms of performance, POAQL completes most tasks at minimal cost. The main reason is that the subteam's consideration is with more certainty and is enough for cooperation according to (6). In terms of efficiency, POAQL trains with the lowest fps. Deep analysis reveals that gradient calculation dominates the efficiency. POAQL is better than IQL in fps because it skips updates of egoistic action value for "idle" status. VDN is almost ten times worse than the IQL because of the gradient calculation of the joint action value, which is related to the number of agents.

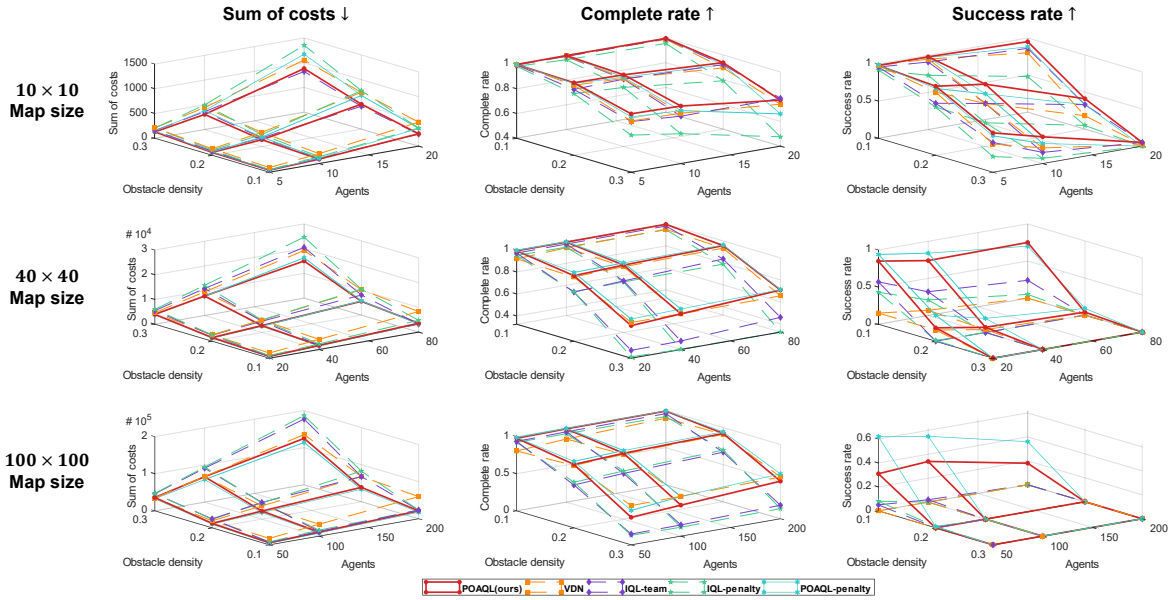

Fig. 4. Generalization result. This graph shows the performance of all methods in various cases. In most cases, the subplots in the "sum of costs" column show meshes of the POAQL sink bottom, and the subplots in the "complete rate" and "success rate" columns show meshes of the POAQL float top. Thus, POAQL completes most tasks at minimal cost.

b) Analysis of Generalization Results: To measure the generalization of methods, we test them in cases different from training. From the result in Fig.4, POAQL performs best in most cases. VDN ranks second and slightly outperforms POAQL in cases with 100×100 map size and 30% obstacle density (the subplot in the 3rd row and 2nd column), which is explained by the cumulative error in the sum of altruistic action values in (3). Based on this explanation, we consider the credit assignment within the subteam as a remedy. Compared with IQL-team, POAQL gets better results in all cases. IQL-penalty ranks last because agents focus only on their own gains and cooperate passively.

c) Ablation: To analyze the unpredictable factors within the team reward, we additionally train POAQL and IQL-team with an increasing proportion of unobserved agents and present the results in Table III. Within a fixed-size field of view, more agents are unobserved as the size of the map and the number of agents increase. From the complete rate and the success rate in the result, we see that POAQL maintains its performance while IQL-team decreases sharply. It shows that POAQL reduces unpredictable factors, especially when unobserved agents are in the majority. To evaluate our conflict resolution, we train POAQL with the conflict resolution of IQL-Penalty (POAQL-penalty). The results in Fig. 4 show that POAQL wins on small maps (subplots above the anti-diagonal), while POAQL-penalty wins on large maps (subplots below the anti-diagonal). The main reason is that the penalty induces agents to keep a safe distance, which is beneficial in a large enough space, but useless in smaller ones. No matter what conflict resolution, POAQL and POAQL-penalty have advantages over IQL-penalty in all cases, indicating that altruistic consideration benefits cooperation.

TABLE III

TRAINING PERFORMANCE WITH INCREASING UNOBSERVED AGENTS

10×10 map size, 0.2 obstacle density, 10 agents			
Method	sum of costs↓	complete rate↑	success rate↑
POAQL(ours)	**137.2**	**0.98**	**0.84**
IQL-team[12]	174.4	0.95	0.58
40×40 map size, 0.2 obstacle density, 40 agents			
Method	sum of costs↓	complete rate↑	success rate↑
POAQL(ours)	**1511**	**0.99**	**0.64**
IQL-team[12]	4119	0.82	0
100×100 map size, 0.2 obstacle density, 100 agents			
Method	sum of costs↓	complete rate↑	success rate↑
POAQL(ours)	**8609**	**0.99**	**0.42**
IQL-team[12]	100000	0	0

VI. CONCLUSIONS

We propose a partially observable Altruistic Q-learning for specific multi-agent cooperative problems. To reduce unpredictable factors, we aim to design a predictable action value to estimate the team gain, and that is the subteam action value. The critical component of the subteam action value is the altruistic action value. We design a new T-step update of the altruistic action value to make it available for changing subteam members. In addition, we design a MAPF environment for warehouse applications, together with the observation and the network for altruistic Q-Learning. Furthermore, we propose a new conflict resolution in MAPF for cooperative MARL frameworks. In our experiments, we compare our method with existing frameworks in MAPF of warehouse applications and demonstrate its superiority in terms of the sum of costs, complete rate, and success rate. Future work may apply credit assignment within the subteam to reduce the cumulative error in the subteam action value.

REFERENCES

[1] H. Ma, J. Yang, L. Cohen, T. Kumar, and S. Koenig, "Feasibility study: Moving non-homogeneous teams in congested video game environments," in *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, vol. 13, no. 1, 2017, pp. 270–272.

[2] P. R. Wurman, R. D'Andrea, and M. Mountz, "Coordinating hundreds of cooperative, autonomous vehicles in warehouses," *AI magazine*, vol. 29, no. 1, pp. 9–9, 2008.

[3] J. Li, A. Tinka, S. Kiesel, J. W. Durham, T. S. Kumar, and S. Koenig, "Lifelong multi-agent path finding in large-scale warehouses," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 13, 2021, pp. 11 272–11 281.

[4] M. Veloso, J. Biswas, B. Coltin, and S. Rosenthal, "Cobots: Robust symbiotic autonomous mobile service robots," in *Twenty-fourth international joint conference on artificial intelligence*. Citeseer, 2015.

[5] R. Stern, N. Sturtevant, A. Felner, S. Koenig, H. Ma, T. Walker, J. Li, D. Atzmon, L. Cohen, T. Kumar, *et al.*, "Multi-agent pathfinding: Definitions, variants, and benchmarks," in *Proceedings of the International Symposium on Combinatorial Search*, vol. 10, no. 1, 2019, pp. 151–158.

[6] A. Oroojlooy and D. Hajinezhad, "A review of cooperative multi-agent deep reinforcement learning," *Applied Intelligence*, vol. 53, no. 11, pp. 13 677–13 722, 2023.

[7] P. Sunehag, G. Lever, A. Gruslys, W. M. Czarnecki, V. Zambaldi, M. Jaderberg, M. Lanctot, N. Sonnerat, J. Z. Leibo, K. Tuyls, *et al.*, "Value-decomposition networks for cooperative multi-agent learning," *arXiv preprint arXiv:1706.05296*, 2017.

[8] S. Gronauer and K. Diepold, "Multi-agent deep reinforcement learning: a survey," *Artificial Intelligence Review*, pp. 1–49, 2022.

[9] G. Sartoretti, J. Kerr, Y. Shi, G. Wagner, T. S. Kumar, S. Koenig, and H. Choset, "Primal: Pathfinding via reinforcement and imitation multi-agent learning," *IEEE Robotics and Automation Letters*, vol. 4, no. 3, pp. 2378–2385, 2019.

[10] M. Damani, Z. Luo, E. Wenzel, and G. Sartoretti, "Primal _2: Pathfinding via reinforcement and imitation multi-agent learning-lifelong," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 2666–2673, 2021.

[11] Z. Ma, Y. Luo, and H. Ma, "Distributed heuristic multi-agent path finding with communication," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 8699–8705.

[12] M. Tan, "Multi-agent reinforcement learning: Independent vs. cooperative agents," in *Proceedings of the tenth international conference on machine learning*, 1993, pp. 330–337.

[13] T. Rashid, M. Samvelyan, C. S. De Witt, G. Farquhar, J. Foerster, and S. Whiteson, "Monotonic value function factorisation for deep multi-agent reinforcement learning," *The Journal of Machine Learning Research*, vol. 21, no. 1, pp. 7234–7284, 2020.

[14] K. Son, D. Kim, W. J. Kang, D. E. Hostallero, and Y. Yi, "Qtran: Learning to factorize with transformation for cooperative multi-agent reinforcement learning," in *International conference on machine learning*. PMLR, 2019, pp. 5887–5896.

[15] S. Shen, M. Qiu, J. Liu, W. Liu, Y. Fu, X. Liu, and C. Wang, "Resq: A residual q function-based approach for multi-agent reinforcement learning value factorization," *Advances in Neural Information Processing Systems*, vol. 35, pp. 5471–5483, 2022.

[16] J. Foerster, G. Farquhar, T. Afouras, N. Nardelli, and S. Whiteson, "Counterfactual multi-agent policy gradients," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 32, no. 1, 2018.

[17] J. Yang, A. Nakhaei, D. Isele, K. Fujimura, and H. Zha, "Cm3: Cooperative multi-goal multi-stage multi-agent reinforcement learning," in *International Conference on Learning Representations*, 2020.

[18] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning," in *Proceedings of the 26th annual international conference on machine learning*, 2009, pp. 41–48.

[19] S. Narvekar, B. Peng, M. Leonetti, J. Sinapov, M. E. Taylor, and P. Stone, "Curriculum learning for reinforcement learning domains: A framework and survey," *The Journal of Machine Learning Research*, vol. 21, no. 1, pp. 7382–7431, 2020.

[20] A. Felner, R. Stern, S. Shimony, E. Boyarski, M. Goldenberg, G. Sharon, N. Sturtevant, G. Wagner, and P. Surynek, "Search-based optimal solvers for the multi-agent pathfinding problem: Summary and challenges," in *Proceedings of the International Symposium on Combinatorial Search*, vol. 8, no. 1, 2017, pp. 29–37.

[21] G. Sharon, R. Stern, M. Goldenberg, and A. Felner, "The increasing cost tree search for optimal multi-agent pathfinding," *Artificial intelligence*, vol. 195, pp. 470–495, 2013.

[22] G. Sharon, R. Stern, A. Felner, and N. R. Sturtevant, "Conflict-based search for optimal multi-agent pathfinding," *Artificial Intelligence*, vol. 219, pp. 40–66, 2015.

[23] E. Boyarski, A. Felner, R. Stern, G. Sharon, O. Betzalel, D. Tolpin, and E. Shimony, "Icbs: The improved conflict-based search algorithm for multi-agent pathfinding," in *Proceedings of the International Symposium on Combinatorial Search*, vol. 6, no. 1, 2015, pp. 223–225.

[24] M. Barer, G. Sharon, R. Stern, and A. Felner, "Suboptimal variants of the conflict-based search algorithm for the multi-agent pathfinding problem," in *Proceedings of the International Symposium on Combinatorial Search*, vol. 5, no. 1, 2014, pp. 19–27.

[25] J. Li, W. Ruml, and S. Koenig, "Eecbs: A bounded-suboptimal search for multi-agent path finding," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 14, 2021, pp. 12 353–12 362.

[26] D. Silver, "Cooperative pathfinding," in *Proceedings of the aaai conference on artificial intelligence and interactive digital entertainment*, vol. 1, no. 1, 2005, pp. 117–122.

[27] H. Ma, D. Harabor, P. J. Stuckey, J. Li, and S. Koenig, "Searching with consistent prioritization for multi-agent path finding," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 33, no. 01, 2019, pp. 7643–7650.

[28] M. T. Spaan, "Partially observable markov decision processes," in *Reinforcement learning: State-of-the-art*. Springer, 2012, pp. 387–414.

[29] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, *et al.*, "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[30] W. Hönig, S. Kiesel, A. Tinka, J. W. Durham, and N. Ayanian, "Persistent and robust execution of mapf schedules in warehouses," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1125–1131, 2019.

[31] Z. Wang, T. Schaul, M. Hessel, H. Hasselt, M. Lanctot, and N. Freitas, "Dueling network architectures for deep reinforcement learning," in *International conference on machine learning*. PMLR, 2016, pp. 1995–2003.

[32] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," in *NIPS 2014 Workshop on Deep Learning, December 2014*, 2014.

[33] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.