MFC-EQ: Mean-Field Control with Envelope Q-learning for Moving Decentralized Agents in Formation

Qiushi Lin and Hang Ma

Abstract— **We study a decentralized version of Moving Agents in Formation (MAiF), a Multi-Agent Path-Finding variant aiming to plan collision-free paths for multiple agents so that they can reach their goals quickly while staying close to a desired formation. The agents must balance these two objectives under partial observation and limited communication. The maintenance of the formation is determined by the joint state of all agents, whose dimension increases exponentially w.r.t. the number of agents, making the learning process intractable. Furthermore, learning a single policy that can handle different linear preferences for these two objectives adds to the challenge. In this paper, we propose mean-field Control with Envelop Q-learning (MFC-EQ), which provides a scalable and adaptable learning framework for this bi-objective multi-agent learning problem. We approximate the dynamic of all agents using the mean-field theory while learning a universal preference-agnostic policy with envelop Q-learning. We empirically evaluate MFC-EQ over numerous instances and demonstrate that it can outperform state-of-the-art centralized MAiF baselines. Furthermore, MFC-EQ tackles more complex scenarios where the desired formation changes dynamically—a challenge that existing MAiF planners cannot handle.**

I. Introduction

Multi-Agent Path Finding (MAPF) [1] is a widely used technique in various multi-agent systems to find collision-free paths for agents in a shared environment. Applications include warehouse management [2], airport surface operations [3], video games [4], and other multi-agent systems [5]. Additionally, many of these applications require agents to adhere closely to a designated formation to accomplish collaborative tasks or maintain an efficient communication network. For example, in warehouse logistics, multiple robots/vehicles are required to collaborate in transporting large objects. Maintaining a specific formation is critical to optimizing transport efficiency or ensuring reliable communication. Moreover, in video gaming or military strategy simulations, game characters or army personnel must move in formations to safeguard vulnerable members.

To tackle this challenge, [6] has formalized the bi-objective problem of Moving Agents in Formation (MAiF) that combines these two tasks and proposed a centralized MAiF planner based on the leader-follower scheme and a search-based MAPF algorithm. However, existing MAiF planners work only in a centralized setting and do not apply to practical scenarios where agents do not fully observe the environment. Furthermore, centralized MAiF planners suffer from a huge computational burden as the number of agents increases and are thus not suitable for planning in real time. Additionally, the only scalable MAiF planner, SWARM-MAPF [6], does not have the flexibility to adjust to the particular preferences between two objectives since it balances the two objectives only by setting the makespan allowance between two sets of heuristically determined waypoints, thus not guaranteed to optimize targeted preference. We propose a novel approach to learning a general MAiF solver for decentralized settings that can directly adapt to various preferences of the two objectives.

In the MAPF literature, reinforcement learning and imitation learning [7] have been introduced to solve MAPF in decentralized settings [8], [9], [10]. However, most learning-based MAPF solvers learn one homogeneous policy for any set of agents that treats nearby agents as part of the environment. This learning scheme does not translate seamlessly to decentralized MAiF. Unlike MAPF where the joint action cost can be directly decomposed to action costs of individual agents, the formation in MAiF is determined by the joint state of all agents at any given time. Each agent is thus required to not only avoid colliding with other agents but coordinate with them to maintain proximity to the desired formation. The dimension of the joint state space grows exponentially with the number of agents, incapacitating the scalability. Besides, trading off two objectives merely under partial observation and limited communication make this task even more difficult.

In this paper, we formalize the decentralized MAiF as a bi-objective multi-agent reinforcement learning task. The major contributions of our paper are as follows. We design a practical learning formalization for MAiF, including specifications for observations, actions, rewards, and inter-agent communication. To address the aforementioned challenges of MAiF, we propose a novel approach called **M**EAN **F**IELD **C**ONTROL WITH **E**NVELOP ***Q***-LEARNING (MFC-EQ), a multi-agent reinforcement learning technique that optimizes towards any linear combination of two objectives for any number of agents, ensuring a stable and efficient learning process. MFC-EQ leverages mean-field control to approximate the collective dynamics of the agents, treating the interaction of each agent within the formation as influenced by the collective effect of others. This design choice facilitates seamless scalability to large-scale instances. Furthermore, MFC-EQ extends envelope Q-learning to a multi-agent setting, enabling the learning of a universal preference-agnostic model adaptable to any linear combinations of the two objectives. To evaluate our method empirically, we extensively test MFC-EQ across various MAiF instances. Our results

The authors are with the School of Computing Science, Simon Fraser University, Burnaby, BC V5A 1S6, Canada
qiushi_lin@sfu.ca, hangma@sfu.ca

substantiate that MFC-EQ consistently produces solutions that dominate those generated by several centralized MAiF planners and scales up well to large numbers of agents without long planning time. Additionally, the learned policy of MFC-EQ can directly adapt to more challenging tasks, including dynamically changing desired formations, which proves to be difficult for centralized MAiF planners.

II. Problem Definition

In this section, we first describe the standard MAiF formulation in a convenient terminology to better present our learning approach. We then discuss how MAiF can be generalized to a partially observable environment, which is a more practical problem setting. Finally, we define relevant concepts and discuss the bi-objective optimization problem of MAiF.

A. Moving Agents in Formation

In the standard formulation, an MAiF instance is defined on an undirected graph $G = (V, E)$ in a d-dimensional Cartesian system. Each location in V can be recognized by its coordinates $\boldsymbol{v} = (v_1, \dots, v_d) \in \mathbb{R}^d$. In this paper, the subscripts represent agents' index numbers and the boldface font denotes multi-dimensional vectors. We also define $[M] = \{1, \dots, M\}$. We have a set of M agents $I = \{a^i | i \in [M]\}$. Each agent has a unique start location $\boldsymbol{s}^i \in V$ and goal location $\boldsymbol{g}^i \in V$. The time is discretized, and between two consecutive time steps, each agent can choose to wait at the current location or move from $\boldsymbol{v}$ to $\boldsymbol{v}'$ provided that $(\boldsymbol{v}, \boldsymbol{v}') \in E$. We also consider two types of collision between agents: a vertex collision $\langle a^i, a^j, \boldsymbol{v}, t\rangle$ means agent a^i and a^j occupy the same location at the same time step t, and an edge collision $\langle a^i, a^j, \boldsymbol{u}, \boldsymbol{v}, t\rangle$ happens when a^i travels from $\boldsymbol{u}$ to $\boldsymbol{v}$ while a^j travels backward.

The MAiF problem aims to find a set of M collision-free paths $\Pi = \{\Pi^i | i \in [M]\}$ as a solution, where $\Pi^i = (p^i_0, \dots, p^i_{T^i})$ represents agent i's trajectory. Every solution will be evaluated by two objectives, makespan and formation deviation. The makespan can be defined as $T = \max_{1 \le i \le M} T^i$, that is, the longest length among all paths. The *formation* at time t can be represented as an M-tuple, $\ell(t) = \langle \boldsymbol{p}^1(t), \dots, \boldsymbol{p}^M(t)\rangle$. The desired formation is the combination of all agents' goal locations, $\ell_{\boldsymbol{g}} = \langle \boldsymbol{g}^1, \dots, \boldsymbol{g}^M\rangle$. Following the definition in [6], the *formation distance* between any two formation $\ell = \langle \boldsymbol{u}^1, \dots, \boldsymbol{u}^M\rangle$ and $\ell' = \langle \boldsymbol{v}^1, \dots, \boldsymbol{v}^M\rangle$ indicates the least effort required to transform from ℓ to ℓ', defined as:

$$\begin{aligned}\mathscr{F}(\ell, \ell') &:= \min_{\boldsymbol{\Delta}} \sum_{i=1}^{M} \|\boldsymbol{u}^i - (\boldsymbol{v}^i + \boldsymbol{\Delta})\|_1 \\ &= \underbrace{\sum_{i=1}^{M} \sum_{j=1}^{d} |(\boldsymbol{u}^i_j - \boldsymbol{v}^i_j) - \boldsymbol{\Delta}_j|}_{\overset{\text{def}}{=} \mathscr{F}^i(\ell, \ell')}, \qquad (1)\end{aligned}$$

where j indexes the dimension for each vector and $\Delta_j = \textit{median}(\{\boldsymbol{u}^i_j - \boldsymbol{v}^i_j\}_{i \in [M]})$ is the median for the j-th dimension. The $\mathscr{F}^i(\ell, \ell')$ denotes the subpart only dedicated to the agent a^i. We consider the average formation deviation per agent across all time steps, unlike the total formation deviation in [6], and it can be defined as $\mathscr{F}_{avg} = \frac{1}{M} \sum_{t=0}^{T} \mathscr{F}(t)$, where $\mathscr{F}(t) = \mathscr{F}(\ell(t), \ell_{\boldsymbol{g}})$. We also consider a mix of these two objectives:

$$\texttt{MIX}(\lambda) = \lambda T + (1 - \lambda)\mathscr{F}_{avg},$$

which is the linear combination of these two objectives. There could certainly be non-linear combinations of different rewards, but, in this work, we only consider the linear cases, since it has been widely adopted in multi-objective or multi-task reinforcement learning (e.g., [11]).

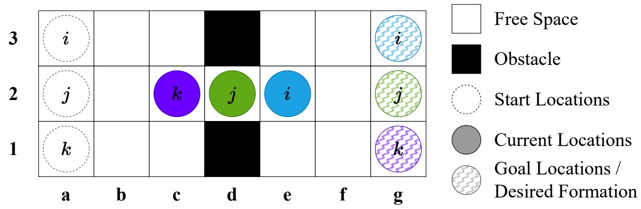

Fig. 1: Example of moving agents in formation.

Example A simple MAiF example is demonstrated in Fig. 1. The start formation is $\langle a3, a2, a1\rangle$ and the goal/desired formation is $\langle g3, g2, g1\rangle$. The group of agents cannot go through the d column while keeping the formation intact, so they have to change the formation. At the current time step t, the median position is $d2$ and the formation deviation is $\mathscr{F}(t) = \mathscr{F}^i(t) + \mathscr{F}^j(t) + \mathscr{F}^k(t) = 2 + 0 + 2 = 4$.

B. Partially Observable Environments

In this paper, we consider a more practical problem setting where, instead of assuming the full knowledge of the environment, each agent can only have a partial observation of its surroundings. We formulate decentralized MAiF as a decentralized partially observable Markov Decision Process (Dec-POMDP) [12]. A Dec-POMDP can be represented as a 7-tuple $\langle \mathcal{S}, \boldsymbol{A}, P_S, \boldsymbol{O}, P_O, R, \gamma\rangle$, where $\mathcal{S}$ is the global state space. $\boldsymbol{A} = \prod_{i=1}^{M} A^i$ and $\boldsymbol{O} = \prod_{i=1}^{M} S^i$, where A^i and S^i are agent i's action and observation space. $P_S : \boldsymbol{A} \times \mathcal{S} \to \mathcal{S}$ describes the state-transition function, and $P_O : \boldsymbol{A} \times \mathcal{S} \to \boldsymbol{O}$ is the observation-transition function. R is the reward function with the discount factor γ.

In decentralized MAiF, we assume the observation and the state-transition function are deterministic, in which each agent has full control of its next position and observation by taking a move or the wait action. Following the settings in existing learning-based MAPF methods, we formalize this problem on a 2D 4-neighbor grid, even though our method can also be easily generalized to other settings. The partial observability limits each agent's perception to a $\mathcal{L} \times \mathcal{L}$ square area with it sitting on the center, defined as its FOV. Each agent can take actions merely based on their local observation and limited communication with other agents.

C. Bi-Objective Optimization

We then formulate the goal of this bi-objective optimization problem. Each MAiF solution is evaluated as (v, w),

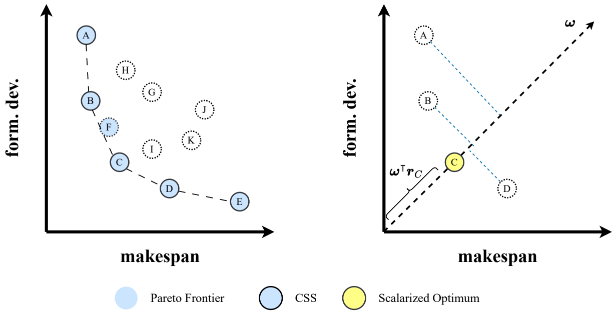

Fig. 2: Illustration of bi-objective optimums.

where v denotes its makespan and w denotes its average formation deviation per agent. We first define dominance. We say $\boldsymbol{r} = (v, w)$ dominates $\boldsymbol{r}' = (v', w')$, denoted as $\boldsymbol{r} \preceq \boldsymbol{r}'$, iff $v \leq v'$ and $w \leq w'$. A solution is Pareto-optimal if and only if there does not exist any solution that can dominate it. The Pareto-optimal frontier is a set of all Pareto-optimal solutions. In the MAiF setting, we are also interested in evaluating each solution r by a scalar function $f_{\boldsymbol{\omega}}(\boldsymbol{r}) = \boldsymbol{\omega}^\top \boldsymbol{r}$, where $\boldsymbol{\omega} \in \Omega$ is the linear preference and Ω is the set of all possible preferences. we let $\boldsymbol{\omega} = (\lambda, 1-\lambda)^\top$ where $0 \leq \lambda < 1$. Our goal is to find the convex convergence set (CSS). The CSS is a subset of the Pareto-optimal frontier, where for each solution in CSS, there exists a preference $\boldsymbol{\omega}$ such that it minimizes $f_{\boldsymbol{\omega}}$ among all possible solutions. Intuitively, as shown in Fig. 2, we can regard the scalar function as a projection to the preference $\boldsymbol{\omega}$. For example, the solution C belongs to CSS since it has the smallest projection into $\boldsymbol{\omega}$ compared to others.

III. Related Work

We now discuss related work on mean-field reinforcement learning and multi-objective reinforcement learning.

A. Mean-Field Reinforcement Learning

Inspired by the mean-field theory [13] from the physics world, the mean-field reinforcement learning has been proposed in [14] which estimates the dynamic within the entire group of agents as the interaction between each agent and the mean effect of all other agents as a whole. As the dimension of the mean effect is independent of the number of agents, this method does not suffer from the curse of dimensionality, providing a general framework for large-scale multi-agent tasks. This method has been extended to the partially observable stochastic settings [15], which utilizes certain distributions to sample agents' actions without the necessity of observing them. The sampling process only serves stochastic games, which does not apply to our task. This mean-field framework has also been used to solve multi-type multi-agent tasks [16], where agents are categorized into different types, and a set of mean effects is considered to reflect various types of agents.

B. Multi-Objective Reinforcement Learning

There exist three major categories of multi-objective reinforcement learning methods. Single-policy methods [17], [18] convert the multi-objective problem into a single-objective optimization by using linear or non-linear functions, but these methods cannot manage unknown preferences. Multi-policy methods [19], [20], [21] update on a set of policies to approximate the real Pareto-optimal frontiers, which requires immense computational resources. These methods are only applicable to problems with limited state and action space. The policy-adaptation methods either train a meta-policy that adapts to different preferences on the fly [22] or learn a policy that conditions on different preference weights [23], [24], [25]. The Envelop Q-learning [25] has been proposed to increase sample efficiency by introducing a novel envelop operator for updating the multi-objective Q-function, which has become a standard way to tackle multi-objective problems with linear preferences.

IV. MFC-EQ

In this section, we show how we design the learning framework for decentralized MAiF. We first design the learning environment with agents' observation, communication, action, and reward functions. Then, we elaborate on the bi-objective multi-agent learning process based on the mean-field theory and the envelope Q-learning.

A. Environment and Model Design

1) Observation: As most research in the MAPF community [1], we study our problem in the 2-dimensional 4-neighbor grids. To mimic many real-world robotics applications where robots have limited visibility and sense range, each agent, in our grid world, can only observe its field of view (FOV), represented by its surrounding $\mathcal{L} \times \mathcal{L}$ area. Each agent's observation is represented by 3-channel feature maps $\mathcal{F} \in \mathbb{R}^{\mathcal{L} \times \mathcal{L} \times 3}$. The first two channels indicate obstacles and other neighboring agents' positions. Inspired by some decentralized MAPF solvers [9], [10], the third channel encompasses the heuristic information where each grid in the FOV is assigned a value proportional to the short-path distance from that to the agent's goal.

2) Action: In 4-neighbor grids, agents can only travel to their cardinally adjacent grids for each step. The action taken by agent i at time t, denoted by $a_t^i \in \mathbb{R}^5$, is a 5-dimensional one-hot vector with each dimension representing one action from $\{up, down, left.right, wait\}$. The first four actions take agents to another location and their observation will shift accordingly. The last action is to have the agent wait at its current location, and it is especially crucial for formation control as one may have the choice for other agents to catch up for lower formation deviation.

3) Multi-Agent Communication: To keep the desired formation, agents not only need to communicate with nearby agents inside FOVs but also have to reach agents outside them. We specifically design the communication message so that it can pass along critical information under low communication bandwidth.

As in many real-world robot applications, each agent can only access the pairwise relative positions between other agents and itself. Assume that the current formation at time step t is $\ell_{\boldsymbol{p}} = \langle \boldsymbol{p}^1, \ldots, \boldsymbol{p}^M \rangle$ and the desired formation is $\ell_{\boldsymbol{g}} = \langle \boldsymbol{g}^1, \ldots, \boldsymbol{g}^M \rangle$. We define the relative position between agent i and j as $\boldsymbol{p}^{i,j} = \boldsymbol{p}^j - \boldsymbol{p}^i$ (resp., $\boldsymbol{g}^{i,j}$). Agent i receives $\{\boldsymbol{p}^{i,j}\}_{j \in [M]}$ in real time, and it holds the information of the relative positions in the goal formation, $\{\boldsymbol{g}^{i,j}\}_{j \in [M]}$, which can be calculated before execution. We show that, only with this information, even without knowing the agent's whereabouts, it can still calculate the formation deviation. As defined in Eq. (1), $\mathscr{F}(\ell_{\boldsymbol{p}}, \ell_{\boldsymbol{g}}) = \min_{\boldsymbol{\Delta}} \sum_{m=1}^{M} \|\boldsymbol{p}^m - (\boldsymbol{g}^m + \boldsymbol{\Delta})\|_1 = \sum_{m=1}^{M} \sum_{n=1}^{d} |(\boldsymbol{p}_n^m - \boldsymbol{g}_n^m) - \boldsymbol{\Delta}_n|$ where $\boldsymbol{\Delta}_n$ is the median of $\{\boldsymbol{p}_n^m - \boldsymbol{g}_n^m\}_{m \in [M]}$. Recall that d is the dimension of agents' coordinates. It is easy to verify that $\mathscr{F}(\ell_{\boldsymbol{p}}, \ell_{\boldsymbol{g}})$ is also equal to $\sum_{m=1}^{M} \sum_{n=1}^{d} |(\boldsymbol{p}_n^m - \boldsymbol{g}_n^m - \boldsymbol{C}_n) - \boldsymbol{\Delta}'_n|$ where $\boldsymbol{C}$ is any constant d-dimensional vector and $\boldsymbol{\Delta}'_n$ is the median of $\{\boldsymbol{p}_n^m - \boldsymbol{g}_n^m - \boldsymbol{C}_n\}_{m \in [M]}$, as all the values and the median are shifted by the same margin. Let $\boldsymbol{C} = \boldsymbol{p}^i - \boldsymbol{g}^i$. We can rewrite the definition of the formation deviation only using the relative position. $\sum_{m=1}^{M} \sum_{n=1}^{d} |(\boldsymbol{p}_n^m - \boldsymbol{g}_n^m) - \boldsymbol{\Delta}_n| = \sum_{m=1}^{M} \sum_{n=1}^{d} |[(\boldsymbol{p}_n^m - \boldsymbol{p}_n^i) - (\boldsymbol{g}_n^m - \boldsymbol{g}_n^i)] - \boldsymbol{\Delta}_n^*| = \sum_{m=1}^{M} \sum_{n=1}^{d} |(\boldsymbol{p}_n^{i,m} - \boldsymbol{g}_n^{i,m} - \boldsymbol{\Delta}_n^*|$ where i is the index of the observing agent and $\boldsymbol{\Delta}_n^*$ is the median of $\{\boldsymbol{p}_n^{i,m} - \boldsymbol{g}_n^{i,m}\}_{m \in [M]}$. Therefore, agent i can calculate the formation deviation merely based on the relative positions, which happens at each decentralized agent during execution with the time complexity of only $\mathcal{O}(d \cdot M)$.

We also mentioned that we can infer the mean action based on the relative positions. Given $\boldsymbol{p}^{i,j}(t)$ and $\boldsymbol{p}^{i,j}(t{+}1)$, we can get that $\boldsymbol{p}^{i,j}(t{+}1) - \boldsymbol{p}^{i,j}(t) = (\boldsymbol{p}^j(t{+}1) - \boldsymbol{p}^j(t)) - (\boldsymbol{p}^i(t{+}1) - \boldsymbol{p}^i(t)) = \boldsymbol{a}^j(t) - \boldsymbol{a}^i(t)$. Hence, agent i can calculate agent j's action by $\boldsymbol{a}^j(t) = \boldsymbol{p}^{i,j}(t{+}1) - \boldsymbol{p}^{i,j}(t) + \boldsymbol{a}^i(t)$. Therefore, we pass this to a linear layer to get the relative position encoding. Besides, each agent can infer other agents' actions by simply comparing relative positions in two consecutive time steps, which will later be used to compute the mean action.

4) Reward: The reward function for agent i after taking action a at time step t, $\boldsymbol{r}_t^i(s_t^i, a_t^i) \in \mathbb{R}^2$, is represented by a 2-tuple. The first element is designated for the makespan. We modify the individual cost function for makespan from DHC [10] which, instead, intends to minimize the sum of all path lengths (a.k.a., flowtime). The moving cost of agent i at time step t with action a^i is:

$$c_t^i(s^i, a^i) = \begin{cases} -0.075 & \text{collision-free } a^i \\ -0.5 & \text{collision (with obstacles or agents)} \\ 3 & \text{reach goal (first time)} \end{cases}.$$

Each collision-free action, including move (up, down, left, or right) and wait (on goal or away goal), is slightly penalized so that agents are incentivized to approach their goals as quickly as they can. The second element is for the formation deviation. As defined in Eq. (1), we add the individual portion of the collective formation deviation that is dedicated to agent j, namely $\mathscr{F}_t^j(\ell_t, \ell^g)$. We negate the formation deviation so that it will be minimized through maximizing rewards. Hence, the reward function can be represented as:

$$\boldsymbol{r}_t^j(s_t^j, a_t^j) = (c_t^j, -\mathscr{F}_t^j(\ell_t, \ell^g))^\top. \tag{2}$$

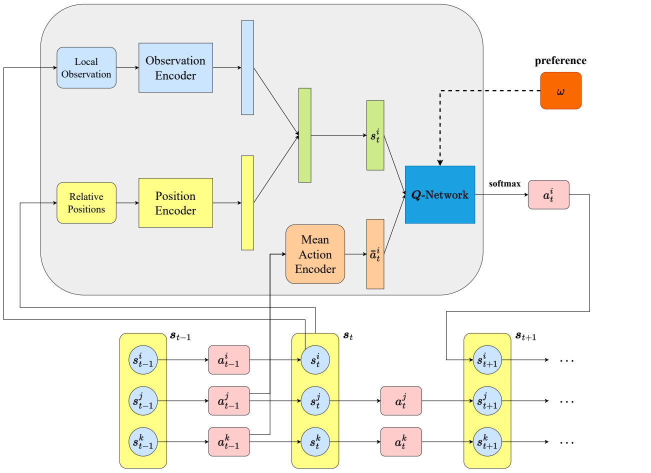

Fig. 3: Illustration of the model architecture of MFC-EQ. The bottom demonstrates the state/observation transition in the partially observable environment. The agent's Q-network gathers information from the environment through partial observation and limited communication and chooses the next action accordingly.

5) Model Architecture: Given the partially observable multi-agent environment, we further design the Q-network, whose learning algorithm will be introduced later. As in Fig. 3, we aim to project each agent's observations and communication messages into a corresponding action. Firstly, we feed the local observation and relative positions into two separate encoders. The observation encoder consists of several stacked convolution layers followed by linear layers. The relative position encoder includes two simple linear layers. Then, we concatenate these two encodings and forward them to another linear layer to obtain the final state representation, s_t^i, for agent i's perception at time t. We then collect other agents' actions from the previous time step to calculate the mean action. Lastly, we use stacked linear layers to project them to the Q-values which condition on the state, the action, the mean action, and the given preference. The agent will decide its next action that maximizes the Q-function produced by the Q-network.

B. Mean-Field and Envelop Optimality

In the rest of this section, we will discuss the details of the learning algorithm. Learning multiple policy networks, $\pi = [\pi^1, \ldots, \pi^M]$, for this bi-objective multi-agent task can be extremely challenging. Therefore we simplify it by making some common assumptions.

1) Mean-Field Approximation: The goal of MAiF is to minimize the makespan and the formation deviation. With the specifically designed reward function, the return, the discounted sum of all rewards from the initial joint state to the goal joint state, $\sum_t \sum_j \gamma^t \boldsymbol{r}_t^j(\boldsymbol{s}_t, \boldsymbol{a}_t)$, can reflect the actual values for the two objectives. Therefore, the goal of the learning is to find a set of policies to maximize the general sum of Q-values $\arg\max_{\pi^1, \ldots, \pi^M} \sum_{j=1}^{M} \boldsymbol{\omega}^\top \boldsymbol{Q}^{\pi^j}(s^j, \boldsymbol{a})$ with the given linear preference $\boldsymbol{\omega}$. However, the dimension of

$\boldsymbol{s}$ and $\boldsymbol{a}$ grows exponentially w.r.t. the number of agents, rendering it infeasible to learn efficiently.

To tackle this problem, we introduce mean-field reinforcement learning. We first lay out two common assumptions of homogeneity and locality that are made in [14] and many other multi-agent reinforcement learning works. The homogeneity assumes each agent shares the same policy, meaning that $\pi^i = \pi^j$ for all $i \neq j$. The locality assumption comes from partial observability, which suggests that agents' actions can only depend on their visible surroundings.

Then, assuming the actions are represented by one-hot vectors, we define the mean action:

$$\bar{a}_t^j = \frac{1}{|\mathcal{N}^i|} \sum_{j \in \mathcal{N}^i} a_t^j, \; a_t^j \sim \pi^j(\cdot | s^j, \bar{a}_{t-1}^j), \tag{3}$$

where $\mathcal{N}^i$ denotes agent j's neighboring agents and π^j represents its policy. With the assumptions of homogeneity and locality, under certain preference $\boldsymbol{\omega}$, the local pairwise interactions can be approximated by the interplay of each agent with the mean effect from its neighbors:

$$\boldsymbol{\omega}^\top \boldsymbol{Q}(s_t^j, \boldsymbol{a}_t) = \frac{1}{|\mathcal{N}^j|} \sum_{k \neq j} \boldsymbol{\omega}^\top \boldsymbol{Q}(s_t^j, a_t^j, a_t^k) \tag{4}$$

$$= \boldsymbol{\omega}^\top \boldsymbol{Q}(s_t^j, a_t^j, \bar{a}_t^j), \tag{5}$$

where $\boldsymbol{a}_t$ is the joint action, a_t is the single-agent action, and $\bar{a}_t$ is the mean action. Given this approximated Q-function, we can derive the agent's policy function with the softmax parameterization:

$$\pi^j(a_t^j | s_t^j, \bar{a}_{t-1}^j) = \frac{\exp(\beta \boldsymbol{\omega}^\top \boldsymbol{Q}(s_t^j, a_t^j, \bar{a}_{t-1}^j))}{\sum_{a \in A^j} \exp(\beta \boldsymbol{\omega}^\top \boldsymbol{Q}(s_t^j, a, \bar{a}_{t-1}^j))}, \tag{6}$$

where β is the Boltzmann parameter.

2) Bellman Optimality Operator: To extend this framework to multi-objective reinforcement learning, we modify the envelop Q-learning [25] by combining the mean-field operator with the envelop optimality operator. We first condition all the Q-values on the linear preference $\boldsymbol{\omega}$, as in $\boldsymbol{Q}(\boldsymbol{s}, \boldsymbol{a}, \boldsymbol{\omega})$. As the standard Q-learning [26], we define the bi-objective multi-agent Bellman optimality operator $\mathcal{T}$ as:

$$(\mathcal{T}\boldsymbol{Q})(\boldsymbol{s}_t, \boldsymbol{a}_t, \boldsymbol{\omega}) := \sum_{j=1}^{M} \boldsymbol{r}^j(s_t^j, a_t^j) + \gamma \mathbb{E}_{s_{t+1}} \arg_Q \big\{$$

$$\max_{\boldsymbol{\omega}' \in \Omega} \sum_{j=1}^{M} \max_{a^j \in A^j} \boldsymbol{\omega}^\top \boldsymbol{Q}(s_{t+1}^j, a^j, \bar{a}_{t+1}^j, \boldsymbol{\omega}') \big\}, \tag{7}$$

where $\arg_Q$ takes the maximized bi-objective Q-values before the linear scalarization. This operator resembles the Bellman optimality operator in the standard Q-learning for single-agent RL and provides the temporal difference (TD) target. The difference is that it also optimizes over the parameter of preference $\boldsymbol{\omega}$. By maximizing $\boldsymbol{\omega}'$ over the next state and its onward trajectory, this approach provides an optimistic perspective for its future rewards. Iteratively applying this operator to the Q-function, we will be able to reach the convergence of the near-optimal Q-function, which has been proven in [25].

Algorithm 1: Mean-Field Control with Envelop Q-learning

```
1  Initialize the $Q$-network $\boldsymbol{Q}_\theta$ and the target $Q$-network $\boldsymbol{Q}_{\bar\theta}$
2  Initialize the replay buffer $\mathcal{D}$ and set $\zeta = 0$
3  for episode = $1, \dots, E$ do
4      Initialize $\bar{a}_0^j$ for all $j \in [M]$
5      for $t = 1, \dots, T_{max}$ do
6          Sample $a_t^j$ $\epsilon$-greedily from $\boldsymbol{Q}_\theta$ by Eq. (6) for all $j \in [M]$
7          Compute new mean actions $\bar{a}_t^j$ by Eq. (3) for all $j \in [M]$
8          Take the joint action $\mathbf{a}_t = [a_t^1, \dots, a_t^M]$ from the state $\mathbf{s}$ to
             the next state $\mathbf{s}_{t+1}$
9          Compute the reward $\mathbf{r}_t = [r^1, \dots, r^M]$ by Eq. (2)
10         Store the transition, $\langle \boldsymbol{s}_t, \boldsymbol{a}_t, \boldsymbol{r}_t, \boldsymbol{s}_{t+1}, \bar{\boldsymbol{a}} \rangle$, into $\mathcal{D}$, where
             $\bar{\boldsymbol{a}}_t = [\bar{a}_t^1, \dots, \bar{a}_t^M]$ is the collection of mean actions
11     if update then
12         Sample $N$ transitions from $\mathcal{D}$ and $N_{\boldsymbol{\omega}}$ preferences from $\Omega$
13         Compute the TD target using the operator in Eq. (7)
14         Update $Q_\theta$ by minimizing the loss from Eq. (8)
15     Update $\boldsymbol{Q}_{\bar\theta}$ with the learning rate $\alpha$: $\bar\theta \leftarrow \alpha\theta + (1-\alpha)\bar\theta$
16     Increase $\zeta$ along the predefined homotopy path
```

C. Double Q-learning

We design our learning algorithm based on the double Q-learning [27] with two different loss functions and the target network. Algorithm 1 presents the detailed learning framework. During the rollout phase (Line 5-10), we sample the transitions in the multi-agent environment with the homogeneous policy. After we obtain enough transitions in the replay buffer, we enter the learning phase (Line 11-14). Given a mini-batch of N transitions and $N_{\boldsymbol{\omega}}$ preferences, we can estimate the TD target $\boldsymbol{y} = (\mathcal{T}\boldsymbol{Q})(\boldsymbol{s}, \boldsymbol{a}, \boldsymbol{\omega})$ via Eq. (7). The first loss function can be computed as the L_2-norm of the multi-objective TD:

$$L_A(\theta) = \mathbb{E}_{\boldsymbol{s},\boldsymbol{a},\boldsymbol{\omega}} \Big[\| \boldsymbol{y} - \sum_{j=1}^{M} \boldsymbol{Q}_\theta(s^j, a^j, \bar{a}^j, \boldsymbol{\omega}) \|_2^2 \Big].$$

Although this loss function is close to the true expected return, the non-smooth surface makes the learning process difficult in the early steps. We combine this with another additional loss function with the projected temporal difference:

$$L_B(\theta) = \mathbb{E}_{\boldsymbol{s},\boldsymbol{a},\boldsymbol{\omega}} \Big[| \boldsymbol{\omega}^\top (\boldsymbol{y} - \sum_{j=1}^{M} \boldsymbol{Q}_\theta(s^j, a^j, \bar{a}^j, \boldsymbol{\omega})) | \Big].$$

$L_A(\theta)$ provides a closer estimation of the true Q-function, while $L_B(\theta)$ makes the landscape of optimization smooth. We train the Q-network via the homotopy optimization [28] based on the combination of these two loss functions:

$$L(\theta) = (1 - \zeta) L_A(\theta) + \zeta L_B(\theta), \tag{8}$$

where, in our case, we gradually increases ζ from 0 to 1 exponentially as the learning progresses.

V. Empirical Evaluation

In this section, we implement MFC-EQ and experimentally evaluate it with other methods on a server equipped with Intel 2.3GHz 16-Core CPUs and NVIDIA A40 GPUs.

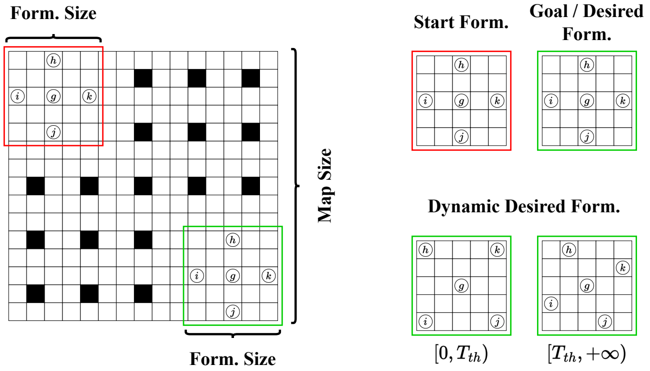

Fig. 4: Demonstration of experiment environments.

A. Experimental Setups

We use 4-neighbor grids with two obstacle-free corners in the top-left and the bottom-right. The default obstacle density for grids outside these two corners is set to be 10%. The agents start at the top-left corner and travel toward the bottom-right corner. The formation in the goal position represents the desired formation. We refer to the size of grids as map size and the size of corners as formation size. For each data point, we averaged over 100 samples by crossing 10 random maps and 10 random formations.

B. Centralized Baselines

We compare our method with several methods which have to plan all paths on centralized servers before execution.

1) Scalarized Prioritized Planning (SPP): Since it is NP-hard to solve this problem optimally, we come up with an efficient yet suboptimal baseline based on the prioritized planning algorithm [29]. We first give each agent a unique priority, and in that priority order, a low-level A* search will be invoked to plan the path from the start location to the goal location while respecting the already planned paths of all agents with higher priorities. The low-level A* search uses a scalarized f-value for each state, which is a mix of the makespan f-value, f_{MS}, and the formation deviation f-value, f_{FD}:

$$f(n) = \underbrace{\lambda\left[cost(\boldsymbol{v}^i) + dist(\boldsymbol{v}^i, g^i)\right]}_{f_{MS}(n)} + (1-\lambda)\underbrace{\sum_{t=1}^{cost(\boldsymbol{v}^i)} \mathscr{F}_P^i(t_n)}_{f_{FD}(n)},$$

where $\mathscr{F}_P^i(t_n)$ is the partial formation deviation among agent i and all other agents with higher priorities. This baseline is not complete but, in most cases, can find a possible solution much more quickly, albeit the solutions usually have poor quality, especially in congested environments with large numbers of agents. Moreover, this planner, unlike SWARM-MAPF, can target any given linear preference.

We use the scalarized f-value to combine these two objectives. The weights for makespan and formation deviation are set to λ and $1-\lambda$, respectively. In the experiments, we vary λ from $\{0.1, 0.3, 0.5, 0.7, 0.9\}$ to test its performance under different linear preferences.

TABLE I: Results of MFC-EQ with different preferences evaluated by different scalarized objectives.

$\boldsymbol{\omega}(\lambda)$	Make-span	Form. Dev.	$\texttt{MIX}(0.1)$	$\texttt{MIX}(0.3)$	$\texttt{MIX}(0.5)$	$\texttt{MIX}(0.7)$	$\texttt{MIX}(0.9)$
0.1	106.33	14.67	**23.84**	42.17	60.50	78.83	97.16
0.3	101.14	15.37	23.95	**41.10**	58.26	75.41	92.56
0.5	98.64	16.84	25.02	41.38	**57.74**	74.10	90.46
0.7	96.74	19.16	26.92	42.43	57.95	**73.47**	88.98
0.9	96.42	21.75	29.22	44.15	59.09	74.02	**88.95**

2) SWARM-MAPF (SWARM): The most effective centralized method, SWARM-MAPF, has been proposed in [6], which combines the swarm-based formation control with the conflict-based MAPF algorithms. The SWARM-MAPF is a two-phase algorithm. In Phase 1, it first calculates the lower bound of the makespan $B = \max_{1 \leq i \leq M} dist(s_i, g_i)$. Given a user-provided parameter $w \geq 1$, SWARM-MAPF selects a leader from the group of agents such that its path, whose length is bounded by wB, can be sufficiently far away from the obstacles and thus others agents can preserve their formation as much as they can. In Phase 2, it will invoke the modified conflict-based search [30] (CBS-M) to minimize the makespan and replanning some critical segments. This planner is complete and suboptimal, but it cannot specifically target any given preference, since we cannot control the trade-off between two objectives based on the parameter w.

3) Joint State A (JSA*):* The joint state A* [31] directly applies the ϵ-constraint search algorithm [32] in the joint state space. The joint state assigns all agents a set of different locations. The operator assigns each agent a set of non-colliding move or wait actions. The OPEN list sorts nodes based on makespan, while the FOCAL list breaks ties based on the formation deviation. Details of this algorithm can be found in [6]. Since the joint state space grows exponentially w.r.t. the number of agents, this method can only be applied to instances with relatively small agents (less than 5 agents in our setups). By varying the ϵ in the focal search, this method is guaranteed to find the Pareto-optimal frontier.

C. Main Results

1) Linear Preferences: We first evaluate the ability of our learned Q-Network to adapt to different preferences. We use the environment that has 16 agents with 48×48 map size and 9×9 formation size. We test different preferences $\boldsymbol{\omega} = (\lambda, 1-\lambda)^\top$ by varying λ from 0.1 to 0.9. We also evaluate each result under different $\texttt{MIX}(\lambda)$ objectives by varying λ from the same set of values. Table I provides 5 different solutions, and every $\texttt{MIX}$ column marks the solution that minimizes the projection onto that particular preference. As we can observe, all $\texttt{MIX}(\lambda)$ objectives are minimized by feeding the corresponding preference $\boldsymbol{\omega}(\lambda)$ into the Q-network. This suggests that the learned Q-network using MFC-EQ can adapt to different preferences and produce multiple solutions that fit the given preferences.

TABLE II: Results for MFC-EQ and centralized baselines with different numbers of agents in various sizes of grids.

		Success Rate			Makespan			Form. Dev.			`MIX`(0.5)		
Map Size	M	SPP	SWA-RM	MFC-EQ	SPP	SWA-RM	MFC-EQ	SPP	SWA-RM	MFC-EQ	SPP	SWA-RM	MFC-EQ
32 × 32	10	1.00	1.00	1.00	48.30	59.32	60.24	29.79	6.07	4.35	39.05	32.70	**32.30**
	20	1.00	0.99	0.99	49.03	63.17	60.38	32.42	12.38	10.04	40.73	37.78	**35.21**
	30	0.79	0.96	0.90	51.54	59.09	54.59	42.25	20.64	20.32	46.90	39.87	**37.46**
48 × 48	10	1.00	0.99	0.99	80.44	98.10	88.07	53.91	8.18	11.05	67.18	53.14	**49.56**
	20	0.95	0.99	0.96	82.07	108.84	104.28	70.44	23.70	21.49	76.26	66.27	**62.89**
	30	0.74	0.94	0.88	84.92	101.52	107.42	96.04	36.30	37.18	90.48	**68.91**	72.30
64 × 64	10	1.00	0.99	0.99	113.38	144.54	137.14	97.92	15.00	16.43	105.65	79.77	**76.79**
	20	1.00	0.97	0.93	114.56	156.03	141.26	113.52	33.24	28.34	114.04	94.64	**84.80**
	30	0.22	0.98	0.90	115.59	142.65	145.51	107.64	57.31	61.43	111.62	**99.98**	103.47

TABLE III: Results for MFC-EQ and centralized baselines under different formation sizes in various sizes of grids.

		Success Rate			Makespan			Form. Dev.			`MIX`(0.5)		
Map Size	Form Size	SPP	SWA-RM	MFC-EQ	SPP	SWA-RM	MFC-EQ	SPP	SWA-RM	MFC-EQ	SPP	SWA-RM	MFC-EQ
32 × 32	7 × 7	0.94	1.00	0.97	53.81	66.40	68.30	44.66	12.37	9.06	49.24	39.39	**38.68**
	9 × 9	1.00	1.00	1.00	48.56	63.20	67.33	29.34	9.10	8.72	38.95	**36.15**	38.03
	11 × 11	1.00	1.00	1.00	44.18	57.75	55.12	20.80	7.03	8.32	32.49	32.39	**31.72**
48 × 48	7 × 7	0.98	1.00	0.87	86.26	110.94	107.37	82.85	19.84	22.40	84.56	65.39	**64.89**
	9 × 9	0.93	0.96	0.93	81.49	109.67	98.64	65.74	21.03	16.84	73.62	65.35	**57.74**
	11 × 11	1.00	1.00	1.00	77.06	105.06	97.26	60.65	15.12	14.08	68.86	60.09	**55.67**
64 × 64	7 × 7	0.87	0.99	0.96	118.64	155.36	138.84	115.38	31.07	55.42	117.01	**93.22**	97.13
	9 × 9	1.00	1.00	0.96	113.87	153.33	133.92	108.57	25.95	33.20	111.22	89.64	**83.56**
	11 × 11	1.00	0.95	1.00	109.43	149.61	131.08	97.68	23.02	27.75	103.56	86.32	**79.42**

2) Number of Agents: We evaluate our methods under different numbers of agents in different map sizes and compare the results with centralized baselines. The λ is set to 0.5 for SPP and MFC-EQ. The w is set to 1.0 for SWARM. The runtime limit is only 30 seconds for MFC-EQ and 5 minutes for SWARM and SPP. Due to the partially observable environment, our method naturally does not have perfect success rates, but they are relatively high and acceptable. As we can observe from Table II, SWARM has greater performance in almost all the test cases. We also can see when projected to the demanded preference, our method can outperform SWARM in most instances. This experiment shows that our method can scale up well to instances with large numbers of agents in different sizes of maps.

3) Formation Size: We repeat the experiment above with various formation sizes. We choose different sizes of obstacle-free corners in which the formation is randomly generated. The larger the corner is, the more spread out the formation will be. The number of agents is fixed at 16. As shown in Table III, we see that smaller formations are usually more difficult to solve, resulting in larger makespan and formation deviation. Compared to SWARM, SPP generally has a better makespan but much worse formation deviation and `MIX`. MFC-EQ solves most instances with greater solution quality in both objectives when compared to the baselines.

TABLE IV: Results for MFC-EQ and centralized baselines for tests of dynamic formations

	Success Rate			Makespan			Form. Dev.			`MIX`(0.5)		
M	SPP	SWA-RM	MFC-EQ	SPP	SWA-RM	MFC-EQ	SPP	SWA-RM	MFC-EQ	SPP	SWA-RM	MFC-EQ
10	1.00	0.98	0.96	48.19	59.00	56.33	127.90	172.40	104.61	88.05	115.70	**80.47**
15	1.00	1.00	1.00	48.29	63.85	57.56	132.71	210.82	114.33	90.50	137.34	**85.95**
20	0.97	1.00	1.00	49.64	63.60	59.07	141.18	208.28	118.42	95.41	135.94	**88.75**
25	0.72	1.00	1.00	50.56	62.58	61.29	149.61	204.33	123.20	100.09	133.46	**92.25**
30	0.90	0.98	0.93	50.00	59.31	62.50	146.82	187.08	129.74	98.41	123.20	**96.12**
35	0.48	0.94	0.87	51.75	57.87	64.71	163.40	189.64	133.07	107.58	123.76	**98.89**
40	0.25	0.81	0.74	52.68	54.12	65.29	168.64	165.05	137.33	110.66	109.59	**101.31**

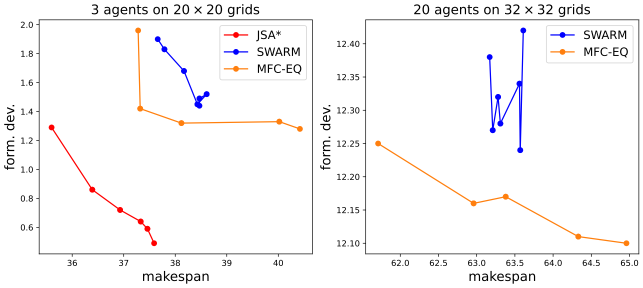

Fig. 5: Trade-off of makespan and formation deviation.

4) Dynamic Formation: We also put these methods into more challenging tests where the agents will be asked to adjust to different formations on the fly. We evaluate agents' formations with one desired formation before $T_{th} = 30$ and another different formation after that. The centralized scheme cannot handle such tasks as agents' paths will have to be planned before execution. In our method, we can simply notify each decentralized agent of the new formation which will result in different ways of calculating relative positions, and therefore the agents can adjust to the new formation seamlessly. The results are shown in Table IV, suggesting that our method has the flexibility to tackle changeable formations, while others result in much larger deviation.

5) Makespan and Formation Trade-off: We further compare our method with others under different preferences. Due to the limited scalability of JSA*, we first use the environment with 20×20 map size, 3×3 formation size, 15% obstacle density, and 3 agents. We vary ϵ of JSA* from 1.0 to 1.8 and w of SWARM from 1.0 to 1.6. The value of λ for $\boldsymbol{\omega}$ in MFC-EQ is varied from 0.1 to 0.9. JSA* can provide the Pareto-optimal frontier only for small-scale instances. We then repeat this experiment in larger instances with 32×32 map size, 9×9 formation size, and 20 agents. Fig. 5 shows the results. Although SPP is also tested, it only gives solutions that have near-optimal makespan but significantly larger formation deviation than the shown results. In large-scale cases, SWARM tends to fluctuate, meaning that, even given more makespan allowance, it may result in solutions with worse formation deviation. The envelope generated by our method is near-convex and can cover all solutions from SWARM, albeit still suboptimal. It also has a wider range of makespan with greater solution variety.

VI. Conclusion and Future Work

We proposed MFC-EQ, a general Q-learning framework for solving decentralized MAiF with partial observation and limited communication. MFC-EQ utilizes the mean-field approximation to simplify the complex multi-agent interaction and employs the envelop Q-learning to enable the adaptability to various preferences for this bi-objective task. Our theoretical proofs further show that the combination of these two operators can still converge to a fixed optimum. Empirical results demonstrate that MFC-EQ outperforms existing centralized baselines in most cases and is more versatile in handling dynamically changing desired formations. Moreover, MFC-EQ is not limited to solving MAiF and has great potential to be generalized to other multi-objective tasks in multi-agent systems.

Although we have great performance in the first attempt at this task, there is certainly room to improve as seen in the experimental results. One promising direction is to design a mixing network [33], [34] to estimate the joint action-value function for handling both objectives. We could also consider applying other multi-objective RL algorithms that can directly approach the Pareto frontier (e.g., [21]). In terms of agents' policy networks, we could integrate the current architecture with more complicated network designs with possible communication mechanisms. We leave these directions to the future.

References

[1] 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.

[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] R. Morris, C. S. Pasareanu, K. S. Luckow, W. Malik, H. Ma, T. S. Kumar, and S. Koenig, "Planning, scheduling and monitoring for airport surface operations." in *AAAI Workshop: Planning for Hybrid Systems*, 2016, pp. 608–614.

[4] 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.

[5] A. Gautam and S. Mohan, "A review of research in multi-robot systems," in *ICIIS*. IEEE, 2012, pp. 1–5.

[6] J. Li, K. Sun, H. Ma, A. Felner, T. Kumar, and S. Koenig, "Moving agents in formation in congested environments," in *Proceedings of the International Symposium on Combinatorial Search*, vol. 11, no. 1, 2020, pp. 131–132.

[7] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[8] 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.

[9] Z. Liu, B. Chen, H. Zhou, G. Koushik, M. Hebert, and D. Zhao, "Mapper: Multi-agent path planning with evolutionary reinforcement learning in mixed dynamic environments," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 11 748–11 754.

[10] 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.

[11] A. Barreto, W. Dabney, R. Munos, J. J. Hunt, T. Schaul, H. P. van Hasselt, and D. Silver, "Successor features for transfer in reinforcement learning," *Advances in neural information processing systems*, vol. 30, 2017.

[12] M. L. Littman, "Markov games as a framework for multi-agent reinforcement learning," in *Machine learning proceedings 1994*. Elsevier, 1994, pp. 157–163.

[13] H. E. Stanley, *Phase transitions and critical phenomena*. Clarendon Press, Oxford, 1971, vol. 7.

[14] Y. Yang, R. Luo, M. Li, M. Zhou, W. Zhang, and J. Wang, "Mean field multi-agent reinforcement learning," in *International conference on machine learning*. PMLR, 2018, pp. 5571–5580.

[15] S. G. Subramanian, M. E. Taylor, M. Crowley, and P. Poupart, "Partially observable mean field reinforcement learning," *arXiv preprint arXiv:2012.15791*, 2020.

[16] S. G. Subramanian, P. Poupart, M. E. Taylor, and N. Hegde, "Multi type mean field reinforcement learning," *arXiv preprint arXiv:2002.02513*, 2020.

[17] Z. Gábor, Z. Kalmár, and C. Szepesvári, "Multi-criteria reinforcement learning." in *ICML*, vol. 98, 1998, pp. 197–205.

[18] S. Mannor and N. Shimkin, "The steering approach for multi-criteria reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 14, 2001.

[19] S. Natarajan and P. Tadepalli, "Dynamic preferences in multi-criteria reinforcement learning," in *Proceedings of the 22nd international conference on Machine learning*, 2005, pp. 601–608.

[20] S. Parisi, M. Pirotta, N. Smacchia, L. Bascetta, and M. Restelli, "Policy gradient approaches for multi-objective sequential decision making," in *2014 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2014, pp. 2323–2330.

[21] K. Van Moffaert and A. Nowé, "Multi-objective reinforcement learning using sets of pareto dominating policies," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 3483–3512, 2014.

[22] X. Chen, A. Ghadirzadeh, M. Björkman, and P. Jensfelt, "Meta-learning for multi-objective reinforcement learning," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 977–983.

[23] A. Castelletti, F. Pianosi, and M. Restelli, "Multi-objective fitted q-iteration: Pareto frontier approximation in one single run," in *2011 International Conference on Networking, Sensing and Control*. IEEE, 2011, pp. 260–265.

[24] A. Abels, D. Roijers, T. Lenaerts, A. Nowé, and D. Steckelmacher, "Dynamic weights in multi-objective deep reinforcement learning," in *International conference on machine learning*. PMLR, 2019, pp. 11–20.

[25] R. Yang, X. Sun, and K. Narasimhan, "A generalized algorithm for multi-objective reinforcement learning and policy adaptation," *Advances in neural information processing systems*, vol. 32, 2019.

[26] C. J. Watkins and P. Dayan, "Q-learning," *Machine learning*, vol. 8, pp. 279–292, 1992.

[27] H. Hasselt, "Double q-learning," *Advances in neural information processing systems*, vol. 23, 2010.

[28] L. T. Watson and R. T. Haftka, "Modern homotopy methods in optimization," *Computer Methods in Applied Mechanics and Engineering*, vol. 74, no. 3, pp. 289–305, 1989.

[29] 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.

[30] 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.

[31] J. Pearl and J. H. Kim, "Studies in semi-admissible heuristics," *IEEE transactions on pattern analysis and machine intelligence*, no. 4, pp. 392–399, 1982.

[32] Y. Haimes, "On a bicriterion formulation of the problems of integrated system identification and system optimization," *IEEE transactions on systems, man, and cybernetics*, no. 3, pp. 296–297, 1971.

[33] 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.

[34] T. Hu, B. Luo, C. Yang, and T. Huang, "Mo-mix: Multi-objective multi-agent cooperative decision-making with deep reinforcement learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023.

Appendix

In this appendix, we provide a theoretical analysis of the proposed method.

A. Mean-Field Approximation in Multi-Objective Reinforcement Learning

In Section IV-B, we have Eq. 5 that gives an approximation for all pairwise interactions among agents. Here, we provide a theorem to analyze the legitimacy of such approximation in the context of multi-objective reinforcement learning.

Theorem 1 (Mean-field Approximation): With the assumptions of homogeneity and locality, under certain preference $\boldsymbol{\omega}$, the local pairwise interactions can be approximated by the interplay of each agent with the mean effect from its neighbors:

$$\frac{1}{|\mathcal{N}^j|}\sum_{k\neq j}\boldsymbol{\omega}^\intercal\boldsymbol{Q}(s_t^j,a_t^j,a_t^k)\approx\boldsymbol{\omega}^\intercal\boldsymbol{Q}(s_t^j,a_t^j,\bar{a}_t^j).$$

Proof: First, we denote the difference between each neighboring agent's action and the mean action as $\delta_t^{j,k}=a_t^k-\bar{a}_t^j$ for agent j where $\bar{a}_t^j$ is the mean action. According to the definition of the mean action, $\sum_{k\neq j}\delta_t^{j,k}=0$. Assuming the given Q-function is twice-differentiable, we then approximate each pairwise Q-function at $\bar{a}_t^j$ w.r.t. a_t^k using Taylor's theorem:

$$\begin{aligned}
&\frac{1}{|\mathcal{N}^j|}\sum_{k\neq j}\boldsymbol{\omega}^\intercal\boldsymbol{Q}^j(s_t^j,a_t^j,a_t^k)\\
=&\frac{1}{|\mathcal{N}^j|}\sum_{k\neq j}\Bigg[\boldsymbol{\omega}^\intercal\boldsymbol{Q}^j(s_t^j,a_t^j,\bar{a}_t^j)+\nabla_{\bar{a}_t^j}\boldsymbol{\omega}^\intercal\boldsymbol{Q}^j(s_t^j,a_t^j,\bar{a}_t^j)\cdot\delta_t^{j,k}+\frac{1}{2}(\delta_t^{j,k})^\intercal\nabla^2_{\bar{a}_t^j}\boldsymbol{\omega}^\intercal\boldsymbol{Q}^j(s_t^j,a_t^j,\bar{a}_t^j)(\delta_t^{j,k})\Bigg]\\
=&\boldsymbol{\omega}^\intercal\boldsymbol{Q}^j(s_t^j,a_t^j,\bar{a}_t^j)+\frac{1}{|\mathcal{N}^j|}\nabla\boldsymbol{\omega}^\intercal\boldsymbol{Q}^j(s_t^j,a_t^j,\bar{a}_t^j)\sum_{k\neq j}\delta_t^{j,k}+\frac{1}{2|\mathcal{N}^j|}\sum_{k\neq j}(\delta_t^{j,k})^\intercal\nabla^2\boldsymbol{\omega}^\intercal\boldsymbol{Q}^j(s_t^j,a_t^j,\bar{a}_t^j)(\delta_t^{j,k})
\end{aligned}$$

We first drop the second term since $\sum_{k\neq j}\delta_t^{j,k}=0$. Given that the Q-functions are L-smooth, we know that for each dimension (objective) i of $\boldsymbol{Q}^j$, we have that $\nabla^2 Q_i^j\preceq LI_{|A|}$, meaning that $\sigma_{\text{max}}(\nabla^2 Q_i^j)\leq L$ and $\sigma_{\text{min}}(\nabla^2 Q_i^j)\geq -L$, where σ_{max} is the largest eigenvalue and σ_{min} is the smallest eigenvalue. Without the loss of generality, we then limit the preference ω such that $|\omega|=1$. As mentioned above, in our task, we define $\omega=(\lambda,1-\lambda)^\intercal$, which satisfies this condition. Then the largest eigenvalue of the Hessian matrix of $\boldsymbol{\omega}^\intercal\boldsymbol{Q}^j$ is:

$$\begin{aligned}
\sigma_{\text{max}}(\nabla^2\boldsymbol{\omega}^\intercal\boldsymbol{Q}^j(s_t^j,a_t^j,\bar{a}_t^j))&\leq\sigma_{\text{max}}(\nabla^2\Big[|\boldsymbol{\omega}|\cdot|\boldsymbol{Q}^j(s_t^j,a_t^j,\bar{a}_t^j))|\Big]) &&\text{(Cauchy–Schwarz inequality)}\\
&=|\boldsymbol{\omega}|\cdot\sigma_{\text{max}}(\nabla^2|\boldsymbol{Q}^j(s_t^j,a_t^j,\bar{a}_t^j))|)\leq L. &&(|\boldsymbol{\omega}|=1\text{ and }\sigma_{\text{max}}\leq L)
\end{aligned}$$

Similarly, we can also have $\sigma_{\text{min}}(\nabla^2\boldsymbol{\omega}^\intercal\boldsymbol{Q}^j(s_t^j,a_t^j,\bar{a}_t^j))\geq -L$. Since the Hessian matrix is symmetric and diagonalizable, we can apply the orthogonal decomposition: $\nabla^2\boldsymbol{\omega}^\intercal\boldsymbol{Q}^j(s_t^j,a_t^j,\bar{a}_t^j)=U^\intercal\Sigma U$, where U is an orthogonal matrix and $\Sigma:=\text{diag}[\sigma_1,\ldots,\sigma_{|A|}]$ is the eigenmatrix of $\nabla^2\boldsymbol{\omega}^\intercal\boldsymbol{Q}^j(s_t^j,a_t^j,\bar{a}_t^j)$. It can be shown that

$$\begin{aligned}
(\delta_t^{j,k})^\intercal\nabla^2\boldsymbol{\omega}^\intercal\boldsymbol{Q}^j(s_t^j,a_t^j,\bar{a}_t^j)(\delta_t^{j,k})=&(\delta_t^{j,k})^\intercal U^\intercal\Sigma U(\delta_t^{j,k})\\
\leq&\sigma_{\text{max}}(U^\intercal\delta_t^{j,k})^2\\
=&L\cdot(\delta_t^{j,k})^\intercal(\delta_t^{j,k}) &&(U^\intercal U=1\text{ and }\sigma_{\text{max}}\leq L)\\
=&L\cdot(a_t^k-\bar{a}_t^j)^\intercal(a_t^k-\bar{a}_t^j) &&(\text{definition of }\sigma_t^{j,k})\\
\leq&L\cdot[(a_t^k)^\intercal(a_t^k)+(\bar{a}_t^j)^\intercal(\bar{a}_t^j)]\\
\leq&2L. &&\text{(one-hot encoding)}
\end{aligned}$$

Similarly, we can show that $(\delta_t^{j,k})^\intercal\nabla^2\boldsymbol{\omega}^\intercal\boldsymbol{Q}^j(s_t^j,a_t^j,\bar{a}_t^j)(\delta_t^{j,k})\geq -2L$. Therefore, we prove that this term is bounded by $[-2L,2L]$. In this symmetrical range, with L being relatively small and the assumptions of homogeneity and locality, these terms across the neighborhood tend to cancel each other, resulting in the third term close to 0. Finally, we prove that

$$\frac{1}{|\mathcal{N}^j|}\sum_{k\neq j}\boldsymbol{\omega}^\intercal\boldsymbol{Q}^j(s_t^j,a_t^j,a_t^k)\approx\boldsymbol{\omega}^\intercal\boldsymbol{Q}^j(s_t^j,a_t^j,\bar{a}_t^j)$$

■

B. Convergence of MFC-EQ

Here, we provide a theoretical analysis of the convergence of MFC-EQ. The multi-object multi-agent Bellman operator can be regarded as a combination of the mean-field operator and the envelop optimality operator. However, it is not obvious that the combined operator can still guarantee the convergence of Q-function to any fixed optimum.

To analyze the convergence, we first define the metric of distance between any two considered Q-function.

Definition 1: The distance between any two Q-functions $\boldsymbol{Q}$ and $\boldsymbol{Q}'$ in $\mathcal{Q} \subseteq S^j \times A^j \times \bar{A}^j \times \Omega \to \mathbb{R}^2$ is defined as:

$$d(\boldsymbol{Q}, \boldsymbol{Q}') := \max_{\substack{\boldsymbol{s}\in\mathcal{S}, \boldsymbol{a}\in\boldsymbol{A} \\ \boldsymbol{\omega}\in\Omega}} \left| \sum_{j=1}^{M} \boldsymbol{\omega}^{\intercal} (\boldsymbol{Q}(s^j, a^j, \bar{a}^j, \boldsymbol{\omega}) - \boldsymbol{Q}'(s^j, a^j, \bar{a}^j, \boldsymbol{\omega})) \right|,$$

where $\bar{a}$ is the mean action defined by Eq. (3).

This metric forms a complete pseudo-metric space. We then generalize the theorems from [25] to multi-agent environments over the defined metric space $\langle \mathcal{Q}, d \rangle$. The proofs follow the framework of the well-known Banach's Fixed-Point Theorem. We first define the optimal Q-function and prove that it is a fixed point of the operator $\mathcal{T}$ in Eq. (7).

To prove that MFC-EQ is still guaranteed to converge in multi-agent systems, we follow the proof framework of the well-known Banach's Fixed-Point Theorem. We first define the multi-objective multi-agent Bellman optimality operator $\mathcal{T}$. Then we prove the following theorems. Theorem 2 states that there exists an optimal Q-function that is a fixed point under the multi-objective multi-agent operator $\mathcal{T}$. Theorem 3 states that $\mathcal{T}$ is a contraction map, meaning that iterative applying it to the Q-function will lead to a closer distance to the fixed point. Lastly, Theorem 4 concludes that in the pseudo-metric space $\langle \mathcal{Q}, d \rangle$, $\mathcal{T}$ results in convergence to the optimal point.

Theorem 2 (Fixed Point): Let $\boldsymbol{Q}^*$ be the optimal Q-function under the preference $\boldsymbol{\omega}$, defined as:

$$\boldsymbol{Q}^* = \arg_Q \max_{\pi} \mathbb{E}_{\tau\sim(\boldsymbol{s},\boldsymbol{a},\pi,P_S)} \left[\sum_{t=0}^{\infty} \gamma^t \boldsymbol{\omega}^{\intercal} \sum_{j=1}^{M} \boldsymbol{r}^j(s_t^j, a_t^j) \right],$$

where τ denotes the trajectory under π. Then, $\boldsymbol{Q}^*$ is a fixed point under $\mathcal{T}$, that is, $\boldsymbol{Q}^* = \mathcal{T}\boldsymbol{Q}^*$.

Proof: We first expand $\boldsymbol{\omega}^{\intercal}\mathcal{T}\boldsymbol{Q}^*$:

$$\begin{aligned}
\boldsymbol{\omega}^{\intercal}\mathcal{T}\boldsymbol{Q}^* &= \boldsymbol{\omega}^{\intercal}\mathcal{T} \arg_Q \max_{\pi} \mathbb{E} \left[\sum_{t=0}^{\infty} \gamma^t \boldsymbol{\omega}^{\intercal} \sum_{j=1}^{M} \boldsymbol{r}^j(s_t^j, a_t^j) \right] && \text{(definition of } \boldsymbol{Q}^*\text{)} \\
&= \boldsymbol{\omega}^{\intercal} \sum_{j=1}^{M} \boldsymbol{r}^j(s_t^j, a_t^j) + \gamma \mathbb{E}_{s_{t+1}} \sum_{j=1}^{M} \max_{a^j\in A^j} \max_{\boldsymbol{\omega}'\in\Omega} \boldsymbol{\omega}^{\intercal} \arg_{Q^j} \max_{\pi} \mathbb{E}_{\substack{\tau\sim\pi \\ s_0=s_{t+1}}} \left[\sum_{t=0}^{\infty} \gamma^t \boldsymbol{\omega}'^{\intercal} \sum_{j=1}^{M} \boldsymbol{r}^j(s_t^j, a_t^j) \right] && \text{(definition of } \mathcal{T}\text{)}
\end{aligned}$$

Assume that $\boldsymbol{\omega}'_*$ is the maximum and the corresponding policy is $\pi_{\boldsymbol{\omega}'_*}$. It can be shown that $\boldsymbol{\omega}^{\intercal} \arg_{Q^j} \max_{\pi} \boldsymbol{Q}(\boldsymbol{\omega}) \leq \max_{\boldsymbol{\omega}'} \boldsymbol{\omega}^{\intercal} \arg_Q \max_{\pi} \boldsymbol{Q}(\boldsymbol{\omega}')$ due to the definition of $\max_{\boldsymbol{\omega}'}$. We also know that $\max_{\boldsymbol{\omega}'} \boldsymbol{\omega}^{\intercal} \arg_Q \max_{\pi} \boldsymbol{Q}(\boldsymbol{\omega}') = \boldsymbol{\omega}^{\intercal} \boldsymbol{Q}^{\pi_{\boldsymbol{\omega}'_*}} \leq \boldsymbol{\omega}^{\intercal} \arg_Q \max_{\pi} \boldsymbol{Q}(\boldsymbol{\omega})$ due to the definition of $\arg_Q$ and $\max_{\pi}$. Hence, the two terms have to be equal to avoid contradictory, as in $\boldsymbol{\omega}^{\intercal} \arg_Q \max_{\pi} \boldsymbol{Q}(\boldsymbol{\omega}) = \max_{\boldsymbol{\omega}'} \boldsymbol{\omega}^{\intercal} \arg_Q \max_{\pi} \boldsymbol{Q}(\boldsymbol{\omega}')$. We plug that into the equation above.

$$\begin{aligned}
&= \boldsymbol{\omega}^{\intercal} \sum_{j=1}^{M} \boldsymbol{r}^j(s_t^j, a_t^j) + \gamma \mathbb{E}_{s_{t+1}} \sum_{j=1}^{M} \max_{a^j\in A^j} \boldsymbol{\omega}^{\intercal} \arg_{Q^j} \max_{\pi} \mathbb{E}_{\substack{\tau\sim\pi \\ s_0=s_{t+1}}} \left[\sum_{t=0}^{\infty} \gamma^t \boldsymbol{\omega}^{\intercal} \sum_{j=1}^{M} \boldsymbol{r}^j(s_t^j, a_t^j) \right] \\
&= \boldsymbol{\omega}^{\intercal} \sum_{j=1}^{M} \boldsymbol{r}^j(s_t^j, a_t^j) + \gamma \mathbb{E}_{s_{t+1}} \boldsymbol{\omega}^{\intercal} \sum_{j=1}^{M} \arg_{Q^j} \max_{\pi} \mathbb{E}_{\substack{\tau\sim\pi \\ s_0\sim P_S(\cdot|s_t^j,\pi(s_0))}} \left[\sum_{t=0}^{\infty} \gamma^t \sum_{j=1}^{M} \boldsymbol{r}^j(s_t^j, a_t^j) \right] && \text{(rearrange)} \\
&= \boldsymbol{\omega}^{\intercal} \sum_{j=1}^{M} \boldsymbol{r}^j(s_t^j, a_t^j) + \gamma \arg_Q \max_{\pi} \sum_{j=1}^{M} \boldsymbol{\omega}^{\intercal} \mathbb{E}_{\substack{\tau\sim\pi \\ s_0\sim P_S(\cdot|s_t^j,\pi(s_0))}} \left[\sum_{t=0}^{\infty} \gamma^t \sum_{j=1}^{M} \boldsymbol{r}^j(s_t^j, a_t^j) \right] && \text{(homogeneity)} \\
&= \boldsymbol{\omega}^{\intercal} \sum_{j=1}^{M} \boldsymbol{r}^j(s_t^j, a_t^j) + \gamma \arg_Q \max_{\pi} \boldsymbol{\omega}^{\intercal} \mathbb{E}_{\substack{\tau\sim\pi \\ s_0\sim P_S(\cdot|s_t^j,\pi(s_0))}} \left[\sum_{t=0}^{\infty} \gamma^t \sum_{j=1}^{M} \boldsymbol{r}^j(s_t^j, a_t^j) \right] && \text{(definition of } \arg_Q\text{)} \\
&= \boldsymbol{\omega}^{\intercal} \arg_Q \max_{\pi} \boldsymbol{\omega}^{\intercal} \mathbb{E}_{\substack{\tau\sim\pi \\ s_0=s_t^j}} \left[\sum_{t=0}^{\infty} \gamma^t \sum_{j=1}^{M} \boldsymbol{r}^j(s_t^j, a_t^j) \right] && \text{(rearrange)} \\
&= \boldsymbol{\omega}^{\intercal} \boldsymbol{Q}^*. && \text{(definition of } \boldsymbol{Q}^*\text{)}
\end{aligned}$$

For any arbitrary $\boldsymbol{\omega}$, we have $\boldsymbol{\omega}^{\intercal}\boldsymbol{Q}^* = \boldsymbol{\omega}^{\intercal}\mathcal{T}\boldsymbol{Q}^* \Rightarrow \boldsymbol{\omega}^{\intercal}(\boldsymbol{Q}^* - \mathcal{T}\boldsymbol{Q}^*) = 0$. As defined in Eq. (**??**), since all the observation spaces and action spaces for each agent are essentially the same, we can finally conclude that $\boldsymbol{Q}^* = \mathcal{T}\boldsymbol{Q}^*$. ■

We then state that the operator $\mathcal{T}$ forms a contraction mapping under the metric d in Q-function space $\mathcal{Q}$.

Theorem 3 (Contraction): $\mathcal{T}$ forms a contraction mapping on the pseudo-metric space $\langle \mathcal{Q}, d \rangle$, that is, for any $\boldsymbol{Q}, \boldsymbol{Q}' \in \mathcal{Q}$, $d(\mathcal{T}\boldsymbol{Q}, \mathcal{T}\boldsymbol{Q}') \leq \gamma d(\boldsymbol{Q}, \boldsymbol{Q}')$, where γ $(0 \leq \gamma < 1)$ is the discount factor.

Proof: We first expand the expression of $d(\mathcal{T}\boldsymbol{Q}, \mathcal{T}\boldsymbol{Q}')$:

$$\begin{aligned} d(\mathcal{T}\boldsymbol{Q}, \mathcal{T}\boldsymbol{Q}') &= \max_{\substack{\boldsymbol{s}\in\mathcal{S},\boldsymbol{a}\in\boldsymbol{A}\\ \boldsymbol{\omega}\in\Omega}} \left| \sum_{j=1}^{M} \boldsymbol{\omega}^{\mathsf{T}} \big(\big[\mathcal{T}\boldsymbol{Q}(s^j, a^j, \bar{a}^j, \boldsymbol{\omega})\big]_j - \big[\mathcal{T}\boldsymbol{Q}(s^j, a^j, \bar{a}^j, \boldsymbol{\omega})\big]_j \big) \right| \\ &= \max_{\substack{\boldsymbol{s}\in\mathcal{S},\boldsymbol{a}\in\boldsymbol{A}\\ \boldsymbol{\omega}\in\Omega}} \Bigg| \sum_{j=1}^{M} \gamma \cdot \boldsymbol{\omega}^{\mathsf{T}} \Bigg[\mathbb{E}_{s'} \arg_{Q^j} \Big\{ \max_{\boldsymbol{\omega}'_Q \in \Omega, a^j_Q \in A^j} \boldsymbol{\omega}^{\mathsf{T}} \boldsymbol{Q}((s')^j, a^j_Q, (\bar{a}')^j, \boldsymbol{\omega}'_Q) \Big\} \\ &\qquad\qquad - \mathbb{E}_{s'} \arg_{Q^j} \Big\{ \max_{\boldsymbol{\omega}'_{Q'} \in \Omega, a^j_{Q'} \in A^j} \boldsymbol{\omega}^{\mathsf{T}} \boldsymbol{Q}'((s')^j, a^j_{Q'}, (\bar{a}')^j, \boldsymbol{\omega}'_{Q'}) \Big\} \Bigg] \Bigg| \end{aligned}$$

We first apply the Cauchy–Schwarz inequality to pull the sum over all the agents out. We then loosen the expectation to the maximum over the next state. Next, we drop the ω^{T} inside $\arg_Q$ as we did in Theorem 2. We will have the following bound.

$$\leq \gamma \cdot \max_{\boldsymbol{s}'\in\mathcal{S},\boldsymbol{\omega}\in\Omega} \sum_{j=1}^{M} \left| \max_{\boldsymbol{\omega}'_Q \in \Omega, a^j_Q \in A^j} \boldsymbol{\omega}^{\mathsf{T}} \boldsymbol{Q}((s')^j, a^j_Q, (\bar{a}')^j, \boldsymbol{\omega}'_Q) - \max_{\boldsymbol{\omega}'_{Q'} \in \Omega, a^j_{Q'} \in A^j} \boldsymbol{\omega}^{\mathsf{T}} \boldsymbol{Q}'((s')^j, a^j_{Q'}, (\bar{a}')^j, \boldsymbol{\omega}'_{Q'}) \right|$$

Let $\boldsymbol{a}^*_Q$ and $\boldsymbol{\omega}^*_Q$ be the optimal vectors that maximizes $\boldsymbol{Q}$. Without the loss of generality, we assume that $\boldsymbol{\omega}^{\mathsf{T}} \boldsymbol{Q}(s^j, a^j, \bar{a}^j, \boldsymbol{\omega}') \geq \boldsymbol{\omega}^{\mathsf{T}} \boldsymbol{Q}'(s^j, a^j, \bar{a}^j, \boldsymbol{\omega}')$. Then we continue the expansion and rearrangement.

$$\begin{aligned} . &\leq \gamma \cdot \max_{\boldsymbol{s}'\in\mathcal{S},\boldsymbol{\omega}\in\Omega} \sum_{j=1}^{M} \left| \boldsymbol{\omega}^{\mathsf{T}} \boldsymbol{Q}((s')^j, (a^*)^j_Q, (\bar{a}')^j, \boldsymbol{\omega}^*_Q) - \max_{\boldsymbol{\omega}'_{Q'} \in \Omega, a^j_{Q'} \in A^j} \boldsymbol{\omega}^{\mathsf{T}} \boldsymbol{Q}'((s')^j, a^j_{Q'}, (\bar{a}')^j, \boldsymbol{\omega}'_{Q'}) \right| \\ &\leq \gamma \cdot \max_{\boldsymbol{s}'\in\mathcal{S},\boldsymbol{\omega}\in\Omega} \sum_{j=1}^{M} \left| \boldsymbol{\omega}^{\mathsf{T}} \boldsymbol{Q}((s')^j, (a^*)^j_Q, (\bar{a}')^j, \boldsymbol{\omega}^*_Q) - \boldsymbol{\omega}^{\mathsf{T}} \boldsymbol{Q}'((s')^j, (a^*)^j_Q, (\bar{a}')^j, \boldsymbol{\omega}^*_Q) \right| \\ &\leq \gamma \cdot \max_{\substack{\boldsymbol{s}\in\mathcal{S},\boldsymbol{a}\in\boldsymbol{A}\\ \boldsymbol{\omega}\in\Omega}} \sum_{j=1}^{M} \left| \boldsymbol{\omega}^{\mathsf{T}} \boldsymbol{Q}(s^j, a^j, \bar{a}^j, \boldsymbol{\omega}_Q) - \boldsymbol{\omega}^{\mathsf{T}} \boldsymbol{Q}'(s^j, a^j, \bar{a}^j, \boldsymbol{\omega}_Q) \right| \\ &= \gamma \cdot d(\boldsymbol{Q}, \boldsymbol{Q}') \end{aligned}$$

Therefore, we prove that $d(\mathcal{T}\boldsymbol{Q}, \mathcal{T}\boldsymbol{Q}') \leq \gamma \cdot d(\boldsymbol{Q}, \boldsymbol{Q}')$, and thus $\mathcal{T}$ is a contraction mapping over $\mathcal{Q}$. ■

Finally, we show that by iterating $\mathcal{T}$ to any Q-function, it will asymptotically converge to the fixed optimum under d.

Theorem 4 (Convergence): Given the defined $\boldsymbol{Q}^*$ in Theorem 2 and the contraction mapping $\mathcal{T}$ on the pseudo-metric space $\langle \mathcal{Q}, d \rangle$ in Theorem 3, iterating $\mathcal{T}$ leads to convergence to $\boldsymbol{Q}^*$, for any $\boldsymbol{Q} \in \mathcal{Q}$, that is, $\lim_{n\to\infty} d(\mathcal{T}^n \boldsymbol{Q}, \boldsymbol{Q}^*) = 0$.

Proof: We prove that $\{\mathcal{T}^n \boldsymbol{Q}\}$ is a Cauchy sequence under the metric d. We first look at

$$d(\mathcal{T}^{m+1}\boldsymbol{Q}, \mathcal{T}^m \boldsymbol{Q}) \leq \gamma d(\mathcal{T}^m \boldsymbol{Q}, \mathcal{T}^{m-1}\boldsymbol{Q}^*) \leq \cdots \leq \gamma^m d(\mathcal{T}\boldsymbol{Q}, \boldsymbol{Q}).$$

For any m and l (w.l.o.g., $m > l$), by the triangular inequality we have

$$\begin{aligned} d(\mathcal{T}^m \boldsymbol{Q}, \mathcal{T}^l \boldsymbol{Q}) &\leq d(\mathcal{T}^m \boldsymbol{Q}, \mathcal{T}^{m-1} \boldsymbol{Q}) + \cdots + d(\mathcal{T}^{l+1} \boldsymbol{Q}, \mathcal{T}^l \boldsymbol{Q}) \\ &\leq (\gamma^{m-1} + \cdots + \gamma^l) \cdot d(\mathcal{T}\boldsymbol{Q}, \boldsymbol{Q}) \\ &= \frac{\gamma^m - \gamma^l}{1-\gamma} \cdot d(\mathcal{T}\boldsymbol{Q}, \boldsymbol{Q}) \leq \frac{\gamma^m}{1-\gamma} \cdot d(\mathcal{T}\boldsymbol{Q}, \boldsymbol{Q}) \end{aligned}$$

Since $0 \leq \gamma < 1$, we know that $d(\mathcal{T}^m \boldsymbol{Q}, \mathcal{T}^l \boldsymbol{Q})$ will converge to 0 as $m, l \to 0$, and hence $\{\mathcal{T}^n \boldsymbol{Q}\}$ is a Cauchy sequence. We can further conclude that $\lim_{n\to\infty} d(\mathcal{T}^n \boldsymbol{Q}, \boldsymbol{Q}_\infty) = 0$ such that $d(\boldsymbol{Q}_\infty, \boldsymbol{Q}^*) = 0$ which yields that $\lim_{n\to\infty} d(\mathcal{T}^n \boldsymbol{Q}, \boldsymbol{Q}^*) = 0$. ■

These theorems guarantee that through applying our proposed operator repeatedly, any initialization of the Q-function can converge to a point that is inherently equivalent to the fixed optimum. Even with one homogeneous Q-function distributed to multiple agents, the Q-learning can be tractable and sample-efficient. This generalization is beneficial not only to our problem but to other homogeneous multi-agent systems.