FM3Q: Factorized Multi-Agent MiniMax Q-Learning for Two-Team Zero-Sum Markov Game

Guangzheng Hu , *Graduate Student Member, IEEE*, Yuanheng Zhu , *Senior Member, IEEE*, Haoran Li , *Member, IEEE*, and Dongbin Zhao , *Fellow, IEEE*

Abstract—**Many real-world applications involve some agents that fall into two teams, with payoffs that are equal within the same team but of opposite sign across the opponent team. The so-called two-team zero-sum Markov games (2t0sMGs) can be resolved with reinforcement learning in recent years. However, existing methods are thus inefficient in light of insufficient consideration of intra-team credit assignment, data utilization, and computational intractability. In this paper, we propose the individual-global-minimax (IGMM) principle to ensure the coherence between two-team minimax behaviors and the individual greedy behaviors through Q functions in 2t0sMGs. Based on it, we present a novel multi-agent reinforcement learning framework, Factorized Multi-Agent MiniMax Q-Learning (FM3Q), which can factorize the joint minimax Q function into individual ones and iteratively solve for the IGMM-satisfied minimax Q functions for 2t0sMGs. Moreover, an online learning algorithm with neural networks is proposed to implement FM3Q and obtain the deterministic and decentralized minimax policies for two-team players. A theoretical analysis is provided to prove the convergence of FM3Q. Empirically, we use three environments to evaluate the learning efficiency and final performance of FM3Q and show its superiority on 2t0sMGs.**

Index Terms—**Multi-agent reinforcement learning, minimax-Q learning, two-team zero-sum Markov games.**

I. INTRODUCTION

MARKOV games (MGs), also known as stochastic games, have been widely used to model the strategic interactions of multiple agents in dynamic environments with multiple states [1]. In recent years, multi-agent reinforcement learning (MARL) has achieved impressive success in its application to multi-agent systems [2], [3], [4], particularly in two-player

zero-sum games (2p0sMGs) [5], [6], [7], [8]. Some research on 2p0sMGs includes approaches to estimating the value at Nash Equilibrium (NE) based on Bellman-like operators [9]. In each optimization process, most of them have to solve the NE of the stage game in each specific state [10], [11]. Whilst it is theoretically possible to solve for NE in 2p0sMGs via linear programming in polynomial time, the optimization over joint-action space suffers from combinatorial explosion as the number of players increases [12]. One effective approach, population-based multi-agent reinforcement learning (PB-MARL) [13], continually generates advanced intelligence by leveraging auto-curricula and has achieved impressive successes in tackling multi-agent tasks, such as the variants of self-play (SP) that are able to achieve expert-level performance on OpenAI Five [14], and methods based on policy space response oracles (PSRO) that have recently achieved impressive performance on AlphaStar [7] and Stratego [15]. In addition, the variants of PSRO utilize different parallelized mechanisms and opponent selection to improve the computation and exploration efficiency at learning high-quality best responses [15], [16], [17], [18]. However, PB-MARL are relatively data-thirsty because of the intrinsic dynamics arising from multiple agents and populations. In other words, in order to be effective, these algorithms must generate a large amount of new data for training in each curriculum.

In addition to just two players, many real-world scenarios involve adversarial situations between two teams, which are typically modeled as 2t0sMGs. Intra-team cooperation and inter-team competition both exist, and it has attracted much interest from reinforcement learning, for example when playing complex games like Dota 2 [14], Starcraft [19], and Honor of kings arena [20]. Of particular interest is the distinction between two lines of research on 2t0sMGs, that only considers inter-team competition and another that also considers intra-team cooperation. The former line regards a team as a single individual and make the 2t0sMGs problem equivalent to 2p0sMGs. A typical example is the success of OpenAIfive [14], in which the presence of full information sharing between teammates makes it equivalent to 2p0sMGs. When addressing inter-team competition, it employs the SP; however, it does not consider the issue of intra-team cooperation and instead utilized an independent learning. The same applies to Honor of kings arena [20]. Another line of research puts the emphasis on intra-team cooperation to coordinate their strategies [21], or on the efficiency of computing NE policies [22]. These research efforts simplify 2t0sMGs to adversarial team Markov games, in which a team of identically

interested players is competing against an adversarial player. However, these methods have yet to address issues such as credit assignment among agents.

In recent years, the development of cooperative MARL methods has been advancing rapidly [23], [24], [25], [26]. Several value-decomposition-based algorithms have emerged in the area of addressing credit assignment. These algorithms are based on the Indivadual-Global-Max (IGM) principle, which not only factorizes the joint Q function into the independent ones, but also potentially solves the problem of credit assignment. As far as we know, this method has not yet been applied to 2t0sMGs.

This naturally leads to the following open question we are interested in: Can we extend the IGM condition and design an efficient MARL framework for 2t0sMGs, in consideration of intra-team credit assignment, data utilization, and computational intractability? To shed light on this open problem, we aim to enable agents to perform two-team minimax behaviors in a decentralized way, and learn such behaviors in an online mode. In this paper, we combine game theory, fitted Q-iteration (FQI), and factorized multi-agent Q-learning to reach the goal. We summarize our contributions as follows:

- We define a novel Individual-Global-MiniMax (IGMM) principle for 2t0sMGs, which specifies the coherence between two-team minimax behaviors and the individual greedy behaviors through Q functions.
- With the IGMM principle, Factorized Multi-Agent MiniMax Q-Learning (FM3Q), which can factorize the joint minimax Q function into individual ones and synchronously optimize the policies of all agents in two teams, is proposed. Moreover, an online learning algorithm with neural networks is proposed to implement FM3Q and learn the two-team minimax policies.
- On the basis of FQI, we prove the global convergence of FM3Q. Empirically, we evaluate the online FM3Q algorithm and baselines on Pong, MPE, and RoboMaster, and demonstrate the outstanding performance of the FM3Q.

II. RELATED WORK

A. MARL Research on 2t0sMGs

A lot of studies treat a team as a single individual, and 2t0sMGs as 2p0sMGs. 2p0sMGs has been widely recognized as the benchmark setting for MARL. The Minimax-Q algorithm focuses on the zero-sum setting with asymptotic convergent guarantees [10]. In the same vein as Minimax-Q learning, asymptotic convergence has also been established for other Q-learning variants beyond the zero-sum setting with coordination among agents, such as Nash-Q [9], Friend-or-Foe Q-Learning [11], and MiniMax Q Network [27]. Each agent in the above algorithms solves a linear program or quadratic program to solve a matrix game at each iteration. As the number of players increases, the optimization over joint-action space suffers from combinatorial explosion.

On the other hand, PB-MARL-type algorithms integrate reinforcement learning with dynamical population selection methods to produce auto-curricula. In PSRO, each player finds an approximate best response to its opponents' meta-strategies,

and the new policies are added into policy sets for the next iteration [16]. PSRO$_{rN}$ uses rectified Nash mixtures, that is, each learner only plays against other learners that it already beats, to encourage policy diversity [17]. Alpha-PSRO introduces the alpha-rank multi-agent evaluation metric and preference-based best response in PSRO and shows promising performance in computing equilibria [28]. There is an increasing emphasis on reducing the intensive computation of PSRO, and many new methods have been proposed. Deep Cognitive Hierarchies (DCH) is one such approach, which parallelizes PSRO to enable scaling to larger games and improve learning speed [16]. The scalability of PSRO can be enhanced through the implementation of a hierarchical pipeline of reinforcement learning agents, where each agent at a higher level is trained against the policies generated by agents at lower levels in the hierarchy [15]. Neural Extensive-Form Double Oracle (NXDO) can be viewed as a version of PSRO where the restricted game allows mixing population policies not only at the root of the game, but at every infostate [18]. Nevertheless, PB-MARL approaches still have certain limitations, including (1) the inefficient utilization of data from previous iterations, as it is discarded and not used to learn new policies, and (2) the need for additional computational resources for meta-solving and empirical payoff evaluation.

In addition, other researchers are dedicated to studying intra-team cooperation and other problems. Soft Team Actor-Critic (STAC) [21] is proposed to make intra-team members associate shared meanings to signals that are initially uninformative. Independent Policy GradientMax (IPGMAX) [29] is able to compute stationary ϵ-approximate NE in adversarial team Markov games with computational complexity that is polynomial in all the natural parameters of the game. They simplify 2t0sMGs to adversarial team Markov games, in which a team of players is competing against an adversarial player, not a team. Adversarial collaborative learning (ACL) exploits friend-or-foe Q-learning and mean-field theory, trains the friends and oppenents via adversarial max and min steps, and suffers from the curse of dimensionality [30]. However, ACL treats 2t0sMGs as 2p0sMGs and ignores credit assignment among agents. Additionally, empirical policy optimization (EPO) propose a novel multi-player reinforcement learning method, in which the parameters are trained based on the whole history of experience [31], but EPO is only applicable to the case that any party in the game is a single agent.

B. Value Factorization and Fitted Q-Iteration

Value decomposition has been increasingly popular in centralized traning and decentralized execution (CTDE)-based MARL for cooperative tasks. The IGM principle [32] of equivalence between joint greedy action and individual greedy action is critical. Existing value-decomposition algorithms satisfy the IGM consistency by expressing the joint Q function with global state as a function of individual Q functions with local observation. Additivity and monotonicity are respectively considered in VDN [33] and QMIX [32], with some variants proposed to relax the monotonicity constraint of QMIX [34], [35]. These methods

have achieved impressive performance on cooperative MARL but there are still some issues to be addressed in applying them to competitive tasks.

Deep Q-learning, as one of the core components of DRL, has shown great success in solving complicated problems. Fitted Q-Iteration (FQI) is based on iterative Bellman error minimization and utilizes a specific Q-function class to iteratively optimize the empirical Bellman error on a dataset [36], [37]. FQI has unique advantages from a theoretical perspective, and there is a growing trend to adopt FQI for theoretical and empirical analysis in DRL [38]. A closely related recent paper, Factorzed Multi-Agent Fitted Q-Iteration (FMA-FQI) [39] models the iterative training procedure of multi-agent Q-learning using empirical Bellman error minimization and formally analyzes cooperative MARL with value factorization. Therefore, if these methods can be applied to the competitive MARL, it would also be conducive to theoretical analysis.

III. PRELIMINARIES

To facilitate theoretical analysis, we divide two teams into the Protagonists (Pro) and the Antagonists (Ant), and use the Decentralized Partial-Observation Markov Decision Process (Dec-POMDP) framework to model the decision-making problem in 2t0sMGs.

Dec-POMDP in 2t0sMGs is described as a tuple $\mathcal{MG} := \langle \mathcal{S}, \mathcal{A}, \mathcal{B}, \mathcal{X}, \mathcal{Y}, \Lambda, \Omega, \mathcal{N}, \mathcal{M}, P, R, \gamma \rangle$, where $\mathcal{S}$ denotes the state space; $\mathcal{N} \equiv \{1, \ldots, n\}$ and $\mathcal{M} \equiv \{1, \ldots, m\}$ are the finite sets of agents, and $\mathcal{X} = \{\mathcal{X}_i\}_{i=1,2,\ldots,n}$ and $\mathcal{Y} = \{\mathcal{Y}_i\}_{i=1,2,\ldots,m}$ represents the sets of observations of each agent in Pro and Ant, respectively; $\Lambda_i(s) : \mathcal{S} \to \mathcal{X}_i$ and $\Omega_i(s) : \mathcal{S} \to \mathcal{Y}_i$ are the observation function that determines the private observation, and the agent i receives a private observation by $x_i = \Lambda_i(s)$ or $y_i = \Omega_i(s)$. $\mathcal{A} = \{\mathcal{A}_i\}_{i=1,2,\ldots,n}$ and $\mathcal{B} = \{\mathcal{B}_i\}_{i=1,2,\ldots,m}$ denote the sets of actions of agents in Pro and Ant. $P(s'|s, \boldsymbol{a}, \boldsymbol{b}) : \mathcal{S} \times \mathcal{A} \times \mathcal{B} \times \mathcal{S} \to [0, 1]$ represents the state transition function, where $\boldsymbol{a} = (a_1, \ldots, a_n)$ and $\boldsymbol{b} = (b_1, \ldots, b_m)$ are the joint action. $R : \mathcal{S} \times \mathcal{A} \times \mathcal{B} \to \mathbb{R}$ indicate the set of reward function of Pro. $\gamma \in [0, 1]$ is the discount factor.

Data sampling is a significantly challenging endeavor in many real-world 2t0sMGs tasks, In transitive games [40], an accurate Nash equilibrium policy is supported by a group of nondominated policies with mixing probabilities. However, seeking such a precise Nash equilibrium in complex scenarios is computationally overwhelming. Therefore, works like AlphaGo [5] and Honor of kings arena [20] resort to the mechanism of self-play to tackle complex tasks. In other words, the primary concern shifts towards obtaining high-performance models using less data rather than acquiring accurate Nash equilibrium solutions at an unconstrained sampling cost. Hence, in this paper, our emphasis is on addressing issues related to data utilization and computational intractability in 2t0sMGs to attain superior performance while working within the confines of deterministic policy settings. Each agent in Pro has an observation-action history $\tau_i \in \mathcal{T}_i \equiv (\mathcal{X}_i \times \mathcal{A}_i)^*$, on which it conditions a deterministic policy $\pi_i(a_i \mid \tau_i) : \mathcal{T}_i \times \mathcal{A}_i \to \{0, 1\}$, and aims to maximize the expected discounted return $\mathbb{E}[\sum_{t=0}^{\infty} \gamma^t r_t]$, where

$r_t \sim R(s_t, \boldsymbol{a}_t, \boldsymbol{b}_t)$. Each agent in Ant has an observation-action history $v_i \in \mathcal{V}_i \equiv (\mathcal{Y}_i \times \mathcal{B}_i)^*$, on which it conditions a stochastic policy $\mu_i(b_i \mid v_i) : \mathcal{V}_i \times \mathcal{B}_i \to \{0, 1\}$, and aims to minimize the above-mentioned discounted return. $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_n)$ and $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_m)$ are the joint policy, and $\boldsymbol{\tau} = (\tau_1, \ldots, \tau_n)$ and $\boldsymbol{v} = (v_1, \ldots, v_m)$ represent the joint observation-action history. We overload $\tilde{s} = \langle \boldsymbol{\tau}, \boldsymbol{v}, s \rangle$ for simplicity. From a centralized perspective, $(\boldsymbol{\pi}, \boldsymbol{\mu})$ has a joint minimax Q function $Q_{\text{tot}}^{\boldsymbol{\pi}, \boldsymbol{\mu}}(\tilde{s}, \boldsymbol{a}, \boldsymbol{b}) = \mathbb{E}_{[\tilde{s}, \boldsymbol{a}, \boldsymbol{b}]_{t+1:\infty}}[\sum_{k=0}^{\infty} \gamma^k r_{t+k}]$, and a joint minimax V value function $V_{Q_{\text{tot}}}^{\boldsymbol{\pi}, \boldsymbol{\mu}}(\tilde{s}) = \mathbb{E}_{[\tilde{s}]_{t:\infty}}[\sum_{k=0}^{\infty} \gamma^k r_{t+k}]$. Our goal is to find the decentralized and deterministic policies $\boldsymbol{\pi}^*$ and $\boldsymbol{\mu}^*$, which satisfy $V_{Q_{\text{tot}}}^{\boldsymbol{\pi}^*, \boldsymbol{\mu}^*}(\tilde{s}) = \min_{\boldsymbol{\mu}} \max_{\boldsymbol{\pi}} V_{Q_{\text{tot}}}^{\boldsymbol{\pi}, \boldsymbol{\mu}}(\tilde{s})$. Further, we show that $\min_{\boldsymbol{\mu}} \max_{\boldsymbol{\pi}} V_{Q_{\text{tot}}}^{\boldsymbol{\pi}, \boldsymbol{\mu}}(\tilde{s}) = \min_{\boldsymbol{b}} \max_{\boldsymbol{a}} Q_{\text{tot}}^{\boldsymbol{\pi}^*, \boldsymbol{\mu}^*}(\tilde{s}, \boldsymbol{a}, \boldsymbol{b})$ under deterministic policies. Thus, we transform the goal to obtain $Q_{\text{tot}}^{\boldsymbol{\pi}^*, \boldsymbol{\mu}^*}(\tilde{s}, \boldsymbol{a}, \boldsymbol{b})$ and perform a global arg min max on it. In what follows, we drop the superscript $\boldsymbol{\pi}, \boldsymbol{\mu}$ for simplicity, and we define the superb Q function Q_{tot}^* and let $Q_{\text{tot}}^*(\tilde{s}, \boldsymbol{a}, \boldsymbol{b}) = Q_{\text{tot}}^{\boldsymbol{\pi}^*, \boldsymbol{\mu}^*}(\tilde{s}, \boldsymbol{a}, \boldsymbol{b})$.

IV. FACTORIZED MULTI-AGENT MINIMAX FITTED Q-ITERATION

In this section, we first introduce the IGMM principle to ensure the coherence between two-team minimax behaviors and individual greedy behaviors through Q functions. Then, based on the IGMM principle, we propose the FM3Q framework to factorize the joint minimax Q function into individual ones. FM3Q draws inspiration from the FQI [36] and iteratively optimizes the policies of all agents synchronously. Finally, we prove the convergence of FM3Q.

A. Individual-Global-Minimax

One idea for directly solving the 2t0sMGs is to divide all agents into two groups, with one group maximizing the rewards and the other group minimizing the rewards, using the joint minimax Q function and constructing linear programming to solve for the NE. However, the optimization over joint-action space suffers from combinatorial explosion as the number of agents increases. Under certain conditions, can we factorize the joint minimax Q function into the individual ones in order to reduce the computational complexity? We need to establish coherence between two-team minimax behaviors and individual greedy behaviors through Q functions. Inspired by the IGM condition used in cooperative MARL, we propose the Indivadual-Global-MiniMax (IGMM) principle to enforce the consistency of action selection between the global joint minimax Q function, denoted as Q_{tot}, and the individual Q functions, denoted as $[Q_i^+]_{i=1}^n$ and $[Q_i^-]_{i=1}^m$ for Pro and Ant, respectively. The IGMM principle can be represented as follows:

$$\arg\min_{\boldsymbol{b}} \max_{\boldsymbol{a}} Q_{\text{tot}}(\tilde{s}, \boldsymbol{a}, \boldsymbol{b}) = \arg\max_{\boldsymbol{a}} \min_{\boldsymbol{b}} Q_{\text{tot}}(\tilde{s}, \boldsymbol{a}, \boldsymbol{b})$$

$$= \left(\arg\max_{a_1} Q_1^+, \ldots, \arg\max_{b_m} Q_m^- \right) \quad (1)$$

Coherence can be achieved when the greedy decentralized policies are determined by an arg max over the $[Q_i^+]_{i=1}^n$ and $[Q_i^-]_{i=1}^m$,

and a global arg min max performed on Q_{tot} yields the same result as a set of individual arg max operations performed on $[Q_i^+]_{i=1}^n$ and $[Q_i^-]_{i=1}^m$. Based on the IGMM, multi-agent reinforcement learning algorithms satisfying the CTDE paradigm can be designed for 2t0sMGs, which not only meet the requirements of distributed execution in many real-world tasks but also significantly reduce computational complexity and improve the efficiency of algorithm training.

To ensure the rationality of the IGMM, it is necessary to impose constraints on it. We define $f([Q_i^+(\tau_i, a_i)]_{i=1}^n, [Q_j^-(v_j, b_j)]_{j=1}^m, s) = Q_{\text{tot}}(\tilde{s}, \boldsymbol{a}, \boldsymbol{b})$, and the constraints on the relationship between Q_{tot} and each Q_i^+ and Q_j^- as two monotonicities: $\frac{\partial Q_{tot}}{\partial Q_i^+} \geq 0, \forall i \in \mathcal{N}$ and $\frac{\partial Q_{tot}}{\partial Q_j^-} \leq 0, \forall j \in \mathcal{M}$, which are adequate to ensure the coherence referred to above. This is demonstrated by the following theorem.

Theorem 1: If $\frac{\partial Q_{\text{tot}}}{\partial Q_i^+} \geq 0, \forall i \in \mathcal{N}$ and $\frac{\partial Q_{\text{tot}}}{\partial Q_j^-} \leq 0, \forall j \in \mathcal{M}$, then the IGMM principle can be satisfied.

Proof: Since $\frac{\partial Q_{\text{tot}}}{\partial Q_i^+} \geq 0, \forall i \in \mathcal{N}$, the following holds for any $\boldsymbol{a}$ and Q_{tot}:

$$f\left([Q_i^+(\tau_i, a_i)]_{i=1}^n, [Q_j^-(v_j, b_j)]_{j=1}^m, s\right)$$

$$\leq f\left(\max_{a_1} Q_1^+(\tau_1, a_1), \ldots, Q_n^+(\tau_n, a_n), [Q_j^-(v_j, b_j)]_{j=1}^m, s\right)$$

$$\leq f\left(\left[\max_{a_i} Q_i^+(\tau_i, a_i)\right]_{i=1}^n, [Q_j^-(v_j, b_j)]_{j=1}^m, s\right)$$

This indicates that the joint Q-value increases with the individual Q-values of Pro. Similarly, since $\frac{\partial Q_{\text{tot}}}{\partial Q_j^-} \leq 0, \forall j \in \mathcal{M}$, the following holds for any $\boldsymbol{b}$ and Q_{tot}:

$$f\left([Q_i^+(\tau_i, a_i)]_{i=1}^n, [Q_j^-(v_j, b_j)]_{j=1}^m, s\right)$$

$$\geq f\left([Q_i^+(\tau_i, a_i)]_{i=1}^n, \max_{b_1} Q_1^-(v_1, b_1), \ldots, Q_m^-(v_m, b_m), s\right)$$

$$\geq f\left([Q_i^+(\tau_i, a_i)]_{i=1}^n, \left[\max_{b_j} Q_j^-(v_j, b_j)\right]_{j=1}^m, s\right)$$

This indicates that the joint Q-value decreases with the individual Q-values of Ant. We show that $\max_{\boldsymbol{a}} f(\cdot) = f([\max_{a_i} Q_i^+]_{i=1}^n, [Q_j^-]_{j=1}^m, s)$ and $\min_{\boldsymbol{b}} f(\cdot) = f([Q_i^+]_{i=1}^n, [\max_{b_j} Q_j^-]_{j=1}^m, s)$. Thus,

$$\max_{\boldsymbol{a}} \min_{\boldsymbol{b}} Q_{\text{tot}}(\tilde{s}, \boldsymbol{a}, \boldsymbol{b})$$

$$:= \max_{\boldsymbol{a}} \min_{\boldsymbol{b}} f\left([Q_i^+]_{i=1}^n, [Q_j^-]_{j=1}^m, s\right)$$

$$= f\left(\max_{a_1} Q_1^+, \ldots, \max_{b_m} Q_m^-, s\right).$$

$\min_{\boldsymbol{b}} \max_{\boldsymbol{a}} Q_{\text{tot}}(\tilde{s}, \boldsymbol{a}, \boldsymbol{b}) = f(\max_{a_1} Q_1^+, \ldots, \max_{b_m} Q_m^-, s)$ represents a similar process. Letting $(a_1^*, \ldots, a_n^*, b_1^*, \ldots, b_m^*) = (\arg\max_{a_1} Q_1^+, \ldots, \arg\max_{b_m} Q_m^-)$, we have that

$$f\left([Q_i^+(\tau_i, a_i^*)]_{i=1}^n, [Q_j^-(v_j, b_j^*)]_{j=1}^m, s\right)$$

$$= f\left(\max_{a_1} Q_1^+(\tau_1, a_1), \ldots, \max_{b_m} Q_m^-(v_m, b_m), s\right)$$

$$= \max_{\boldsymbol{a}} \min_{\boldsymbol{b}} Q_{\text{tot}}(\tilde{s}, \boldsymbol{a}, \boldsymbol{b}) = \min_{\boldsymbol{b}} \max_{\boldsymbol{a}} Q_{\text{tot}}(\tilde{s}, \boldsymbol{a}, \boldsymbol{b})$$

Hence,

$$\arg\min_{\boldsymbol{b}} \max_{\boldsymbol{a}} Q_{\text{tot}}(\tilde{s}, \boldsymbol{a}, \boldsymbol{b}) = \arg\max_{\boldsymbol{a}} \min_{\boldsymbol{b}} Q_{\text{tot}}(\tilde{s}, \boldsymbol{a}, \boldsymbol{b})$$

$$= \left(\arg\max_{a_1} Q_1^+, \ldots, \arg\max_{b_m} Q_m^-\right)$$

$\square$

B. Factorized Multi-Agent MiniMax Q-Learning

Recall that our goal is to obtain the superb Q function Q_{tot}^* and perform a global arg min max on it. Directly solving it faces the problems of high computational complexity and may not satisfy the requirement of distributed execution in some environments. The IGMM condition ensures consistency of action selection between the global joint minimax Q function and the individual Q functions. Therefore, based on it, we propose a new multi-agent method framework called Factorized Multi-Agent MiniMax Q-Learning (FM3Q) for solving 2t0sMGs. FM3Q factorizes the joint minimax-Q function subject to the IGMM principle and optimizes the policies through the use of factorized individual Q-functions, which satisfies the CTDE paradigm. To facilitate theoretical analysis and clarify our approach, we introduce the framework of Fitted Q-Iteration (FQI) [36], to optimize the policies of all agents by iteratively minimizing the empirical minimax Bellman error. We overload $\tilde{Q} = \langle Q_{\text{tot}}, [Q_i^+]_{i=1}^n, [Q_j^-]_{j=1}^m \rangle$ to indicate their association to facilitate further discussions. Additionally, we greedily turn $Q_{\text{tot}}(\tilde{s}, \boldsymbol{a}, \boldsymbol{b}) \in \mathbb{R}^{\mathcal{T} \times \mathcal{V} \times \mathcal{S} \times \mathcal{A} \times \mathcal{B}}$ into $V_{Q_{\text{tot}}}(\tilde{s}) := \min_{\boldsymbol{b}} \max_{\boldsymbol{a}} Q_{\text{tot}}(\tilde{s}, \boldsymbol{a}, \boldsymbol{b}) \in \mathbb{R}^{\mathcal{T} \times \mathcal{V} \times \mathcal{S}}$ to facilitate below discussions.

FM3Q is an iterative optimization framework like FQI that is based on a provided dataset D. Let T denote the total number of iterations. We first randomly initialize $\tilde{Q}^{(0)}$ at time 0 from $\mathcal{Q}^{\text{IGMM}}$, as defined in Definition 1. In the subsequent training, $\tilde{Q}$ is iteratively updated at iteration $t \in [0, T-1]$ as:

$$\tilde{Q}^{(t+1)} \leftarrow \mathcal{T}_D^{\text{IGMM}} \tilde{Q}^{(t)}$$

$$\equiv \arg\min_{\tilde{Q} \in \mathcal{Q}^{\text{IGMM}}} \mathbb{E}_{(\tilde{s}, \boldsymbol{a}, \boldsymbol{b}, r, \tilde{s}') \sim D} \left[e^{(t)}(\tilde{s}, \boldsymbol{a}, \boldsymbol{b}, r, \tilde{s}') - Q_{\text{tot}}(\tilde{s}, \boldsymbol{a}, \boldsymbol{b})\right]^2 \tag{2}$$

where

$$e^{(t)}(\tilde{s}, \boldsymbol{a}, \boldsymbol{b}, r, \tilde{s}') = r + \gamma \mathbb{E}_{\tilde{s}'} \left[\min_{\boldsymbol{b}'} \max_{\boldsymbol{a}'} Q_{\text{tot}}^{(t)}(\tilde{s}', \boldsymbol{a}', \boldsymbol{b}')\right] \tag{3}$$

denotes the one-step temporal difference target at time t. Based on $\tilde{Q}^{(t+1)}$, we can construct decentralized policies by individual value functions as:

$$\forall i \in \mathcal{N}, \pi_i^{(t+1)}(\tau_i) = \arg\max_{a_i \in \mathcal{A}_i} Q_i^{+(t+1)}(\tau_i, a_i);$$

$$\forall j \in \mathcal{M}, \mu_j^{(t+1)}(v_j) = \arg\max_{b_j \in \mathcal{B}_i} Q_j^{-(t+1)}(v_j, b_j).$$

Assumption 1. (Exploratory Data Collection)*:* The dataset D is collected by two joint exploratory policies $\boldsymbol{\pi}^D$ and $\boldsymbol{\mu}^D$ satisfying $\forall(\boldsymbol{\tau} \times \boldsymbol{v} \times \boldsymbol{a} \times \boldsymbol{b}) \in \mathcal{T} \times \mathcal{V} \times \mathcal{A} \times \mathcal{B}, \boldsymbol{\pi}^D(\boldsymbol{a} \mid \boldsymbol{\tau}) > 0$ and $\boldsymbol{\mu}^D(\boldsymbol{b} \mid \boldsymbol{v}) > 0$.

Definition 1: FM3Q specifies the $\tilde{Q}$ function class with a complete IGMM principle realization

$$\mathcal{Q}^{\text{IGMM}} = \Big\{ \tilde{Q} \mid$$

$$\arg\min_{\boldsymbol{b}} \max_{\boldsymbol{a}} Q_{\text{tot}}(\tilde{s}, \boldsymbol{a}, \boldsymbol{b}) = \begin{pmatrix} \arg\max_{a_1} Q_1^+(\tau_1, a_1) \\ \vdots \\ \arg\max_{b_m} Q_m^-(v_m, b_m) \end{pmatrix},$$

$$Q_{\text{tot}}(\tilde{s}, \boldsymbol{a}, \boldsymbol{b}) \in \mathbb{R}^{|\mathcal{T} \times \mathcal{V} \times \mathcal{A} \times \mathcal{B} \times \mathcal{S}|}, [Q_i^+]_{i=1}^n \in \mathbb{R}^{|\mathcal{T} \times \mathcal{A}|^n},$$

$$[Q_j^-]_{j=1}^m \in \mathbb{R}^{|\mathcal{V} \times \mathcal{B}|^m} \Big\}$$

The FM3Q algorithm minimizes the empirical minimax Bellman error by utilizing global reward signals obtained from the Pro. After iterative updates using FM3Q, the final two-team minimax Q function and the individual Q functions can be obtained. With the individual Q functions, the greedy action selection can be searched in the individual action spaces $\mathcal{A}_i$ or $\mathcal{B}_j$, rather than the joint action space $\mathcal{A} \times \mathcal{B}$, which significantly reduces the computation costs when distributedly executing.

C. Convergence of FM3Q

Lemma 1: The empirical minimax Bellman operator $\mathcal{T}_D^{\text{IGMM}}$ in (2) is a γ-contraction.

Proof: First, for a given $\tilde{Q} = \langle Q_{\text{tot}}, [Q_i^+]_{i=1}^n, [Q_j^-]_{j=1}^m \rangle$, the analytical solution of $\mathcal{T}_D^{\text{IGMM}}\tilde{Q}$ is presented as $\tilde{q} = \langle q_{\text{tot}}, [q_i^+]_{i=1}^n, [q_j^-]_{j=1}^m \rangle$. For a sample $(\tilde{s}, \boldsymbol{a}, \boldsymbol{b}, r, \tilde{s}')$ in the dataset D, let

$$\boldsymbol{a}^*, \boldsymbol{b}^* = [a_i^*]_{i=1}^n, [b_j^*]_{b=1}^m = \arg\min_{\boldsymbol{b} \in \mathcal{B}} \max_{\boldsymbol{a} \in \mathcal{A}} e(\tilde{s}, \boldsymbol{a}, \boldsymbol{b}, r, \tilde{s}').$$

We show that

$$q_{\text{tot}}(\tilde{s}, \boldsymbol{a}, \boldsymbol{b}) = e(\tilde{s}, \boldsymbol{a}, \boldsymbol{b}, r, \tilde{s}') \qquad (4)$$

and its corresponding decentralized local Q functions $[q_i^+]_{i=1}^n$, $[q_j^-]_{j=1}^m$:

$$q_i^+(\tau_i, a_i) = \begin{cases} 1, & \text{when } a_i = a_i^* \\ 0, & \text{when } a_i \neq a_i^* \end{cases} \qquad (5)$$

and

$$q_j^-(v_j, b_j) = \begin{cases} 1, & \text{when } b_j = b_j^* \\ 0, & \text{when } b_j \neq b_j^* \end{cases}. \qquad (6)$$

It is easy to see that $\langle q_{\text{tot}}, [q_i^+]_{i=1}^n, [q_j^-]_{j=1}^m \rangle$ satisfies the IGMM principle, and the empirical minimax Bellman error reach the minimum zero: $\sum p_D(\tilde{s}, \boldsymbol{a}, \boldsymbol{b}, r, \tilde{s}')(e(\tilde{s}, \boldsymbol{a}, \boldsymbol{b}, r, \tilde{s}') - q_{\text{tot}}(\tilde{s}, \boldsymbol{a}, \boldsymbol{b}))^2 = 0$, where $p_D(\tilde{s}, \boldsymbol{a}, \boldsymbol{b}, r, \tilde{s}')$ represents the probability of a sample in the dataset D. This proves that $\tilde{q}$ is the analytical solution of $\mathcal{T}_D^{\text{IGMM}}\tilde{Q}$.

Second, let us show that for all $\tilde{s}$, $|V_{Q_{\text{tot}}}(\tilde{s}) - V_{Q'_{\text{tot}}}(\tilde{s})| \leq \min_{\boldsymbol{b}} \max_{\boldsymbol{a}} |Q_{\text{tot}}(\tilde{s}, \boldsymbol{a}, \boldsymbol{b}) - Q'_{\text{tot}}(\tilde{s}, \boldsymbol{a}, \boldsymbol{b})|$. Then, we can derive the following:

$$\min_{\boldsymbol{b}} \max_{\boldsymbol{a}} Q_{\text{tot}} - \min_{\boldsymbol{b}} \max_{\boldsymbol{a}} Q'_{\text{tot}}$$

$$= -\left[\min_{\boldsymbol{b}} \max_{\boldsymbol{a}} Q'_{\text{tot}} - \min_{\boldsymbol{b}} \max_{\boldsymbol{a}} Q_{\text{tot}} \right]$$

$$\leq -\min_{\boldsymbol{b}} \left[\max_{\boldsymbol{a}} Q'_{\text{tot}} - \max_{\boldsymbol{a}} Q_{\text{tot}} \right]$$

$$= \max_{\boldsymbol{b}} \left[\max_{\boldsymbol{a}} Q_{\text{tot}} - \max_{\boldsymbol{a}} Q'_{\text{tot}} \right]$$

$$\leq \max_{\boldsymbol{b}} \max_{\boldsymbol{a}} [Q_{\text{tot}} - Q'_{\text{tot}}] \leq \max_{\boldsymbol{b}} \max_{\boldsymbol{a}} |Q_{\text{tot}} - Q'_{\text{tot}}|$$

$$\min_{\boldsymbol{b}} \max_{\boldsymbol{a}} Q_{\text{tot}} - \min_{\boldsymbol{b}} \max_{\boldsymbol{a}} Q'_{\text{tot}} \geq \min_{\boldsymbol{b}} \left[\max_{\boldsymbol{a}} Q_{\text{tot}} - \max_{\boldsymbol{a}} Q'_{\text{tot}} \right]$$

$$\geq \min_{\boldsymbol{b}} \left[-\max_{\boldsymbol{a}} (Q'_{\text{tot}} - Q_{\text{tot}}) \right] \geq -\max_{\boldsymbol{b}} \max_{\boldsymbol{a}} [Q'_{\text{tot}} - Q_{\text{tot}}]$$

$$\geq -\max_{\boldsymbol{b}} \max_{\boldsymbol{a}} |Q'_{\text{tot}} - Q_{\text{tot}}|$$

Therefore, we can conclude: $|V_{Q_{\text{tot}}}(\tilde{s}) - V_{Q'_{\text{tot}}}(\tilde{s})| \leq \max_{\boldsymbol{b}} \max_{\boldsymbol{a}} |Q_{\text{tot}}(\tilde{s}, \boldsymbol{a}, \boldsymbol{b}) - Q'_{\text{tot}}(\tilde{s}, \boldsymbol{a}, \boldsymbol{b})|$. Using this,

$$\left\| (\mathcal{T}_D^{\text{IGMM}} Q)_{\text{tot}} - (\mathcal{T}_D^{\text{IGMM}} Q')_{\text{tot}} \right\|_\infty = \|q_{\text{tot}} - q'_{\text{tot}}\|_\infty$$

$$= \gamma \left\| \mathbb{E}[V_{Q_{\text{tot}}}] - \mathbb{E}[V_{Q'_{\text{tot}}}] \right\|_\infty \leq \gamma \left\| V_{Q_{\text{tot}}} - V_{Q'_{\text{tot}}} \right\|_\infty$$

$$= \gamma \max_{\boldsymbol{\tau}} \max_{\boldsymbol{v}} \max_{\tilde{s}} |V_{Q_{\text{tot}}}(\tilde{s}) - V_{Q'_{\text{tot}}}(\tilde{s})|$$

$$\leq \gamma \max_{\boldsymbol{\tau}} \max_{\boldsymbol{v}} \max_{\tilde{s}} \max_{\boldsymbol{a}} \max_{\boldsymbol{b}} |Q_{\text{tot}}(\tilde{s}, \boldsymbol{a}, \boldsymbol{b}) - Q'_{\text{tot}}(\tilde{s}, \boldsymbol{a}, \boldsymbol{b})|$$

$$= \gamma \|Q_{\text{tot}} - Q'_{\text{tot}}\|_\infty$$

Now, we can prove that $\mathcal{T}_D^{\text{IGMM}}$ is a γ-contraction. □

Theorem 2: FM3Q globally converges to the superb Q function under deterministic setting in the Dec-POMDPs if IGMM condition is satisfied.

Proof: Recall that $Q_{\text{tot}}^*(\tilde{s}, \boldsymbol{a}, \boldsymbol{b}) = \max_{\boldsymbol{\pi} \in \Pi} \min_{\boldsymbol{\mu} \in \Phi} Q_{\text{tot}}^{\boldsymbol{\pi}, \boldsymbol{\mu}}(\boldsymbol{\tau}, \boldsymbol{v}, s, \boldsymbol{a}, \boldsymbol{b})$, where Q_{tot}^* denotes the superb Q function, and Π and Φ are the space of all deterministic policies of both sides if IGMM condition is satisfied.

First, we want to prove that $Q_{\text{tot}}^* = (\mathcal{T}_D^{\text{IGMM}} Q^*)_{\text{tot}}$.

$$Q_{\text{tot}}^*(\tilde{s}, \boldsymbol{a}, \boldsymbol{b}) = \max_{\boldsymbol{\pi}} \min_{\boldsymbol{\mu}} Q_{\text{tot}}^{\boldsymbol{\pi}, \boldsymbol{\mu}}(\tilde{s}, \boldsymbol{a}, \boldsymbol{b})$$

$$= \max_{\boldsymbol{\pi}} \min_{\boldsymbol{\mu}} \left\{ r + \gamma \mathbb{E}\left[V_{Q_{\text{tot}}}^{\boldsymbol{\pi}, \boldsymbol{\mu}}(\tilde{s}') \right] \right\}$$

$$= r + \gamma \mathbb{E}\left[\max_{\boldsymbol{\pi}} \min_{\boldsymbol{\mu}} V_{Q_{\text{tot}}}^{\boldsymbol{\pi}, \boldsymbol{\mu}}(\tilde{s}') \right]$$

$$= r + \gamma \mathbb{E}\left[\max_{\boldsymbol{\pi}} \min_{\boldsymbol{\mu}} Q_{\text{tot}}^{\boldsymbol{\pi}, \boldsymbol{\mu}}(\tilde{s}', \boldsymbol{\pi}(\boldsymbol{\tau}'), \boldsymbol{\mu}(\boldsymbol{v}')) \right]$$

$$= r + \gamma \mathbb{E}\left[\max_{\boldsymbol{\pi}, \boldsymbol{a}'} \min_{\boldsymbol{\mu}, \boldsymbol{b}'} Q_{\text{tot}}^{\boldsymbol{\pi}, \boldsymbol{\mu}}(\tilde{s}', \boldsymbol{a}', \boldsymbol{b}')) \right]$$

$$= r + \gamma \mathbb{E}\left[\max_{\boldsymbol{a}'} \min_{\boldsymbol{b}'} Q_{\text{tot}}^*(\tilde{s}', \boldsymbol{a}', \boldsymbol{b}') \right]$$

$$= r + \gamma \mathbb{E}\left[V_{Q_{\text{tot}}^*}(\tilde{s}') \right] = (\mathcal{T}_D^{\text{IGMM}} Q^*)_{\text{tot}}$$

Second, $\forall Q'_{\text{tot}} \in \mathcal{Q}^{\text{IGMM}}$,

$$\left\| Q_{\text{tot}}^* - (\mathcal{T}_D^{\text{IGMM}} Q')_{\text{tot}} \right\|_\infty$$

$$= \left\| (\mathcal{T}_D^{\text{IGMM}} Q^*)_{\text{tot}} - (\mathcal{T}_D^{\text{IGMM}} Q')_{\text{tot}} \right\|_\infty \leq \gamma \|Q_{\text{tot}}^* - Q'_{\text{tot}}\|_\infty.$$

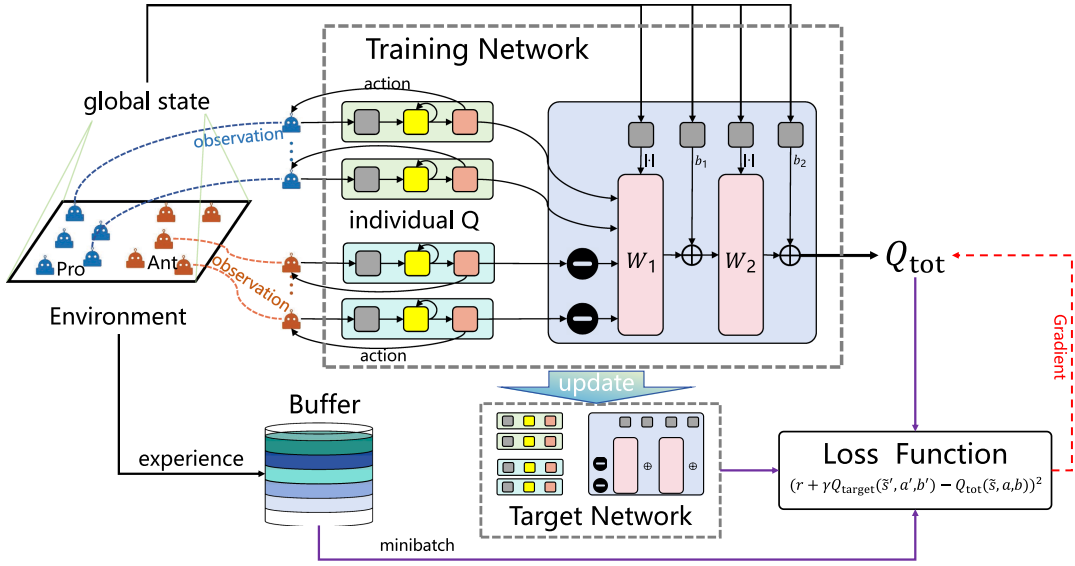Thus, FM3Q will globally converge to a superb Q function. □

Fig. 1. Overall architecture of FM3Q, where two-team agents acquire observations from the environment and derive their actions and corresponding individual Q-values through their utility modules. The individual Q-values of the Ant undergo the negation process and, combined with the individual Q-values of the Pro, are processed through a mixing network to yield their minimax joint Q-values. The black arrows indicate that FM3Q interacts with the environment and stores experience when performing execution. The purple arrows indicate the foward propagation process when training model. The red arrow show the gradient flow.

V. ONLINE LEARNING OF FM3Q

In this section, we give a formal introduction to the online learning of FM3Q, which includes the processes of forward propagation, action selection, and training. Moreover, we provide a novel coordinator when training that significantly differ from previous DRL training methods. It is of importance for training FM3Q, and failure to adhere to it may lead to suboptimal outcomes. The architecture and the detailed process of the sampling and training of FM3Q.

A. Forward Propagation and Action Selection

The overall process of forward propagation is carried out in the direction indicated by the black arrowheads in the left part of Fig. 1. The current observation of the agent (x_i or y_j) is processed through its respective Q networks, as shown in the pink boxes, and the action (a_i or b_j) and the Q function ($Q_i^+(\tau_i, a_i)$ or $Q_j^-(v_i, b_i)$) are determined through action selection. The Q function of the Pro agents are directly input into the mixing network, while the ones of the Ant agents first undergo a negative module (as shown in the blue box). Finally, the mixing network outputs the joint minimax Q value. The specific implementation details are represented below.

For each agent i in Pro, there is an individual Q-network $Q_i^+(\tau_i, a_i; \theta_i^+)$, parameterized by θ_i^+, similarly in Ant, Q-network $Q_i^-(v_i, b_i; \theta_i^-)$, parameterized by θ_i^-. Neural networks in these modules can switch flexibly with the application scenario. For example, the Gated Recurrent Unit (GRU), as a memory-based method, can be selected in the partially observable scenario for better performance in execution. Set ϵ value to guide whether it is exploration or exploitation. The goal of ϵ-greedy exploration is to make all possible actions under a certain state have a non-zero probability of being selected

for execution. We can obtain the policy for agent i in Pro by $\pi_i(\cdot|\tau_i) = \mathcal{G}(Q_i^+(\tau_i, \cdot|\theta^+), \epsilon)$, detailed as

$$\pi_i(a_i|\tau_i) = \begin{cases} \frac{\epsilon}{|\mathcal{A}|} + 1 - \epsilon & \text{if } a_i = \underset{a\in\mathcal{A}}{\operatorname{argmax}} Q_i^+(\tau_i, a|\theta^+) \\ \frac{\epsilon}{|\mathcal{A}|} & \text{otherwise} \end{cases}$$

(7)

The Pro agent selects the action by $a_i \sim \pi_i(\cdot|\tau_i)$. The same holds true for $\mu_j(b_j|v_j)$ and $Q_j^-(v_j, b|\theta^-)$ in the Ant agents.

To ensure that $\frac{\partial Q_{tot}}{\partial Q_i^+} \geq 0, \forall i \in \mathcal{N}$ and $\frac{\partial Q_{tot}}{\partial Q_j} \leq 0, \forall j \in \mathcal{M}$ as required by IGMM condition, we need to construct a monotonic mixing network to mix the corresponding individual Q functions. We formulate the monotonic mixing network as $Q_{tot} = \text{Mix}([Q_i^+]_{i=1}^n, -[Q_j^-]_{j=1}^m, s; \phi)$, parameterized by ϕ, and let it satisfy $\frac{\partial Q_{tot}}{\partial Q_i^+} \geq 0, \forall i \in \mathcal{N}$ and $\frac{\partial Q_{tot}}{\partial Q_j^-} \leq 0, \forall j \in \mathcal{M}$, respectively, by constraining the weights within the mixing network to a range greater than 0. The mixing network takes the Pro network outputs and the negative of the Ant network outputs as input and mixes them monotonically, producing the values of Q_{tot}. The weights of the mixing network ω, as shown in Fig. 1, are generated by individual hypernetworks, each of which receives the state variable s as input and produces the weights for a specific layer of the mixing network. The absolute activation function enforces the non-negativity of the mixing weights in each hypernetwork. Biases are also generated by hypernetworks, using the same approach, but without the non-negativity constraint. The introduction of the global state s is aimed at allocating different weights to different agents, potentially addressing the issue of credit assignment. Benefiting from the monotonicity constraint in (1), minimaximizing joint Q_{tot} is the equivalent of maximizing individual Q of all agents, resulting in and allowing for superb individual action to

---

Algorithm 1: Online Learning of FM3Q.

Initialize the network parameters, the replay buffer D, the current buffer size L, the batch size B, and the update times after a sampling U.

Input: the number of episodes $n\_episodes$

Output: $[\pi_i]_{i=1}^n$, $[\mu_j]_{j=1}^m$

1 **for** $episode = 1$ to $n\_episodes$ **do**
2   **while** *not terminal* **do**
3   Compute the individual Q function
4   Sample the action for each agent
5   Interact with the environment
6   **end**
7   Store the trajectory in the replay buffer D
8   Sample U batch of samples with size $B = L/U$
9   **for** $count = 1$ to U **do**
10   Update the traning networks by minimizing the loss Equation (10) based on the batch
11   **end**
12   Update the target networks
13 **end**

---

maintain consistency with minimax joint action. Thus, a centralized factorizable Q_{tot} parameterized by $\boldsymbol{\theta}^+, \boldsymbol{\theta}^-, \phi$ is used to estimate the joint minimax Q function as follows:

$$Q_{\text{tot}}(\tilde{s}, \boldsymbol{a}, \boldsymbol{b}; \boldsymbol{\theta}^+, \boldsymbol{\theta}^-, \phi)$$
$$= \text{Mix}\left(\left[Q_i^+\left(\tau_i, a_i; \theta_i^+\right)\right]_{i=1}^n, -\left[Q_j^-\left(v_j, b_j; \theta_j^-\right)\right]_{j=1}^m, s; \phi\right) \quad (8)$$

B. Loss and Training

As shown in the right part of Fig. 1, the agents interact with the environment, and the generated data is stored in the replay buffer, which differs from the ones used in prior methods as described in Remark 1. Samples from the buffer are then used to calculate the loss, with the coordinator, as described in Remark 2, controlling the update frequency of the target network to ensure optimization stability. Finally, the parameters of the network are updated via gradient backpropagation in the direction indicated by the red arrow in Fig. 1. The entire online learning process of FM3Q is illustrated in Algorithm 1. The details are indicated below.

Based on (3), the TD target is substituted with approximate target value, using parameters $\hat{\boldsymbol{\theta}}^+, \hat{\boldsymbol{\theta}}^-, \hat{\phi}$ from some previous iteration to let the target network provide a stable supervisory signal and avoid the training instability:

$$e = r + \gamma Q_{\text{tot}}\left(\tilde{s}', \boldsymbol{a}', \boldsymbol{b}'; \hat{\boldsymbol{\theta}}^+, \hat{\boldsymbol{\theta}}^-, \hat{\phi}\right) \quad (9)$$

The whole network trains the parameters $\boldsymbol{\theta}^+, \boldsymbol{\theta}^-, \phi$ by backpropagation of the loss $\mathcal{L}$. The loss calculation is shown as follows:

$$\mathcal{L} = \mathop{\mathbb{E}}_{(\tilde{s}, \boldsymbol{a}, \boldsymbol{b}, r, \tilde{s}') \sim D}\left(e - Q_{\text{tot}}(\tilde{s}, \boldsymbol{a}, \boldsymbol{b}; \boldsymbol{\theta}^+, \boldsymbol{\theta}^-, \phi)\right)^2 \quad (10)$$

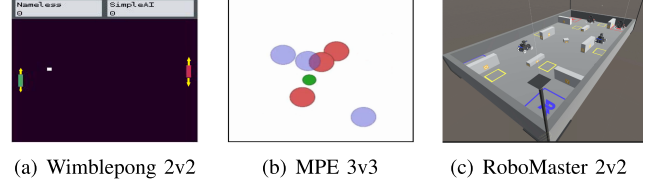(a) Wimblepong 2v2   (b) MPE 3v3   (c) RoboMaster 2v2

Fig. 2. Illustrations of Wimblepong, MPE, and RoboMaster. (a) A green and a red paddle represent two players, respectively, and a white circle represents a ball. (b) Three blue agents and three red agents move, collide, and obstruct each other to maneuver their teams as close as possible to a green target. (c) There are two blue robots, two red robots, and nine obstacles.

Remark 1: The replay buffer enables the use of historical data for training, which greatly enhances sample efficiency. Furthermore, randomly sampling from the replay buffer breaks the correlation between data samples, reducing biases introduced by consecutive sampling. This helps reduce inter-sample correlations and potential sample selection biases, preventing learning instability due to dependencies among the data. In competitive multi-agent tasks, the game space is typically both transitive and cyclic [17]. Specifically, when the replay buffer covers a small space, the minimax Q solution will be limited in the game space covered by the current data. Therefore, we recommend using the largest replay buffer possible when training FM3Q and, in some cases, incorporating all data generated by interaction. We present experimental evidence in the experiments (Section VI-D) that validates the efficiency of this approach through ablation studies.

Remark 2: Because Q-based methods regard predicting the Q value as a regression problem, which requires a supervisory signal at each update. DQN [41] introduces the target network, which is only updated with the training network parameters every U steps and is held fixed between individual updates. As above mentioned, in FM3Q, the distribution of the data in the replay buffer is widely divergent from that generated by the current policy. In order to let the target network provide a more stable supervisory signal, we suggest incorporating as much data as possible in the training process before updating the target network. Of course, this suggestion can be moderately relaxed according to different environments. The amount of data in the replay buffer keeps increasing with training. We draw inspiration from the EPO [31] and modified the update approach for the target network in FM3Q. We denote the current size of the replay buffer as L and the batch size as B. Recall that we hope as much data as possible participates in a round of training before updating the target network. It is recommended that ($B = L/U$), and following the update of the training network, the target network should be updated directly, which is presented in line 10 of Algorithm 1.

VI. Experiments

To verify the effectiveness of FM3Q and its superiority over other methods, we conduct experimental validation in three scenarios. Wimblepong is shown in Fig. 2(a), MPE is shown in Fig. 2(b), and RoboMaster, which simulates competition between two groups of robots in the real world, is illustrated in Fig. 2(c).

Wimblepong 2v2 [31]: It is based on a 2-player version of the Atari game Pong. We modify the environment to create a cooperative-competitive setting involving two teams of two agents each. In this setup, the outputs of the two agents within each team are combined and applied to a paddle as a collective force. In discrete action settings, the agent takes one of three actions: moving up or down, or staying in place. The game state consists of the positions of the two paddles and the position and velocity of the ball. If a player misses a ball, it receives a -10 reward, and the opponent receives a +10 reward.

MPE 3v3 [42]: A simple multi-agent particle world with a continuous observation and discrete action space, along with some basic simulated physics. Each of the two sides, red and blue, consists of three agents that collaborate with their allies and compete with the other group to seize the green target. The group with the closest average distance to the target receives a higher reward. Each agent can take one of five actions: moving up or down, moving left or right, or staying in place.

RoboMaster 2v2 [43]: We introduce a multi-robot environment that involves both intra-team cooperation and inter-team competition, named RoboMaster. Robots cooperate with teammates and fight opponents by shooting. The robots defeat the opponents as much as possible within a certain time limit to win the game. The state consists of information about itself, its ally, and two opponents, including position, angle, candidate points, HP, bullet count, and the time remaining. The robot determines which candidate point goes (4-discrete) and which opponent is hit (2-discrete). This environment adopts three reward functions: hitpoint damage dealt ($r_h = 0.02$), enemy units killed ($r_k = 3$), and a bonus for winning the battle ($r_w = 20$).

We consider SP [14], PSRO [16], EPO [31], and NXDO [18] to be popular methods that can be employed to address 2t0sMGs. SP involves an agent improving its performance through repeated interaction with itself, while PSRO pertains to a game-solving framework that iteratively refines strategies by considering response dynamics in the space of policies. EPO introduces an innovative multiplayer reinforcement learning approach where parameters are trained using the whole historical experience and optimized through proximal policy gradient. Unlike PSRO, which mixes the best responses at the root node of the game, NXDO mixes the best responses at each infostate. We run all experiments with FM3Q and all baselines, and each experiment is repeated eight times to reduce the impact of randomness. For the sake of fairness, we utilize QMIX as an individual in the policy populations of SP, PSRO, and NXDO and set some identical hyperparameters, such as network and learning rate. All methods related to FM3Q adhere to the recommendations mentioned in Remark 2. Some important hyperparameters used for training are shown in Appendix. In each scenario, the five methods save the learned models at different phases of their training for subsequent performance testing and comparison. We evaluate the performance of the five methods from the following aspects: In Section VI-A, we compare the performance differences between the models obtained by three algorithms during the training process and the script-based bots. In Section VI-B, we compare the exploitability of each method against the others and their approximate NashConv during the training process. In Section VI-C, we investigate the optimization trends of FM3Q by comparing the performance of models at different phases during training. In Section VI-D, we validate the impact of buffer size on the training of FM3Q through ablation.

A. Performance Against Script-Based Bots

In this section, we utilize script-based bots to compare the performance differences between the three methods and the bots using the same amount of training data. In Pong and MPE, the script-based bots select actions to approach the ball, while in RoboMaster, the bot prioritizes attacking the unit with the least health. The model with superior performance should achieve a higher positive difference. For the Wimblepong, MPE, and RoboMaster, we use a total of 13 k, 13 k, and 80 k episodes to train FM3Q and baselines, and test the performance with the rule-based bots every 1 k, 1 k, and 8 k episodes, respectively. The results of the learned models of all methods during their training process playing against the bots are shown in Fig. 3.

We can draw evident conclusions from the three result graphs: FM3Q exhibits the fastest optimization, the highest performance, and the smallest variance when compared to other baselines. Firstly, in Wimblepong, MPE, and RoboMaster, FM3Q requires fewer than 2 k, 1 k, and 10 k episodes, respectively, to achieve performance of 0.5, 10, and 0.5. In contrast, the baselines necessitates a minimum of 4 k, 6 k, and 40 k episodes, respectively, to reach the same performance levels in the three environments. It demonstrates the advantage of FM3Q in achieving better performance with less data.

In addition, it is evident from the results that FM3Q outperforms other algorithms significantly. The performance of the models obtained by FM3Q in the three different environments against the bots is approximately 0.95, 30, and 0.7, respectively. In contrast, the best performance achieved by other algorithms in the three environments is as follows: PSRO achieves approximately 0.8 in Wimblepong, NXDO achieves approximately 20 in MPE, and NXDO achieves approximately 0.55 in RoboMaster.

We observe that the performance fluctuations is minimal for FM3Q, followed by NXDO, PSRO, and SP, with the largest fluctuations observed for EPO. The primary reason for this is that FM3Q retains and effectively leverages all historical experience, while NXDO and PSRO engage more opponents relative to SP, making them relatively more stable. The reason for the inferior performance and variance of EPO lies in its single-agent optimization method, which lacks consideration for credit assignment among agents.

B. Exploitability Evaluation

The above experiments only demonstrate that, when competing with the fixed bots, FM3Q outperforms the baselines during the training process. Next, we utilize the different metrics, round-robin (RR) tournament results to estimate the exploitability of each method against all other baselines [44], and approximate NashConv when facing the best response [45].

We evaluate RR tournament results where each method is matched against every other method. A round-robin tournament is a competition where each contestant meets every other
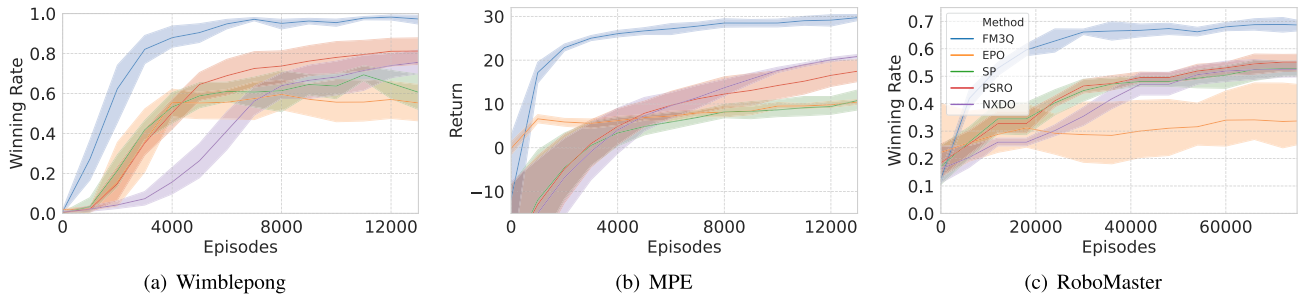
Fig. 3. Illustration of the performance of FM3Q during training by evaluating against the rule-based bots.
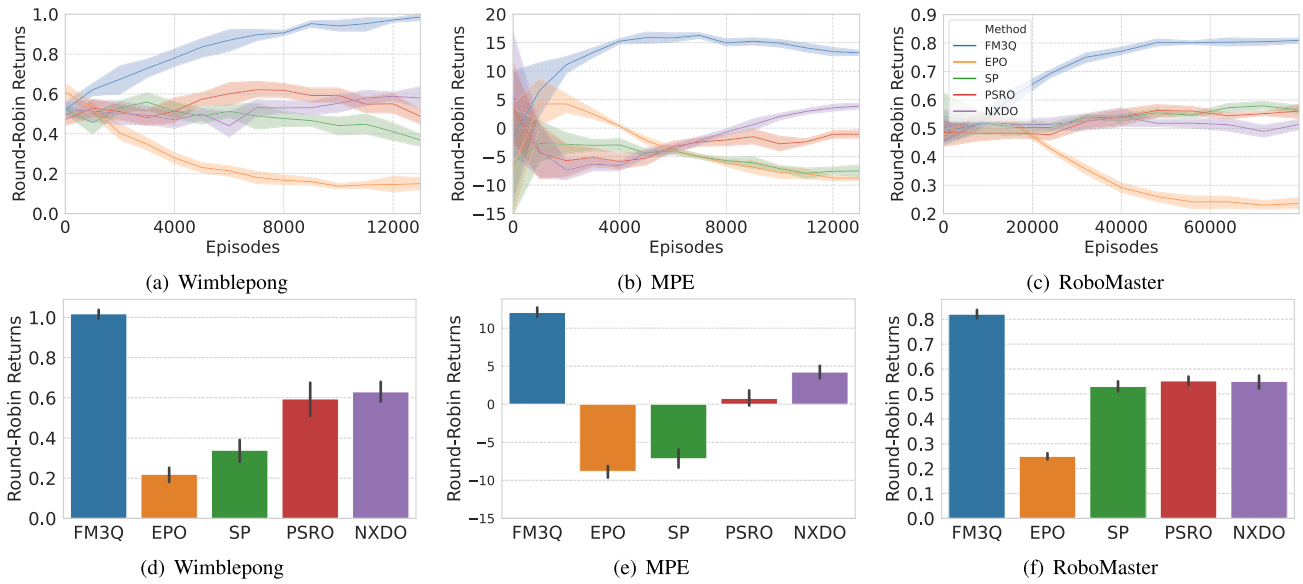

Fig. 4. Illustration of the Round-Robin results of FM3Q and baselines. (Top) normalised RR returns during training. (Bottom) normalised RR returns at the end of training.
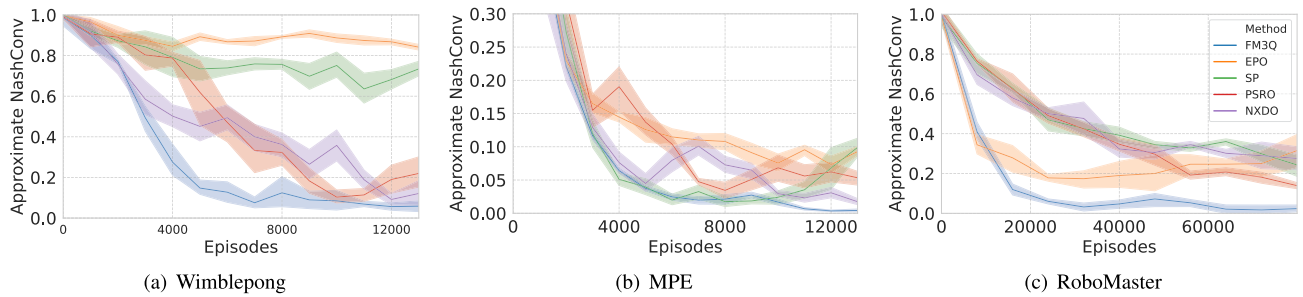

Fig. 5. Illustration of the approximate NashConv of FM3Q and baselines during training.

participant. In particular, during the training process, we preserve models of all methods across different training phases. Then we select models from each method at the same training phase and engage them in cross-play to obtain outcomes. Finally, the method accumulates its scores against all other methods. Less exploitable agents should attain a higher RR returns than all other agents. The three subplots at the top of Fig. 4 respectively depict the normalised RR returns of all methods during training

in three environments, and show that FM3Q is able to quickly improve its performance and outperforms baselines. The three subplots at the bottom of Fig. 4 display the normalised RR returns at the end of training, and show that FM3Q outperforms all baselines and highlight the superiority of FM3Q.

Next, we employ the NashConv metric to demonstrate the efficacy of the policy. Calculating NashConv necessitates finding the best response to the policy, yet precise best responses

(a) Small buffer size on Wimblepong.　(b) Large buffer size on Wimblepong.　(c) Full buffer size on Wimblepong.

(d) Small buffer size on MPE.　(e) Large buffer size on MPE.　(f) Full buffer size on MPE.

(g) Small buffer size on RoboMaster.　(h) Large buffer size on RoboMaster.　(i) Full buffer size on RoboMaster.
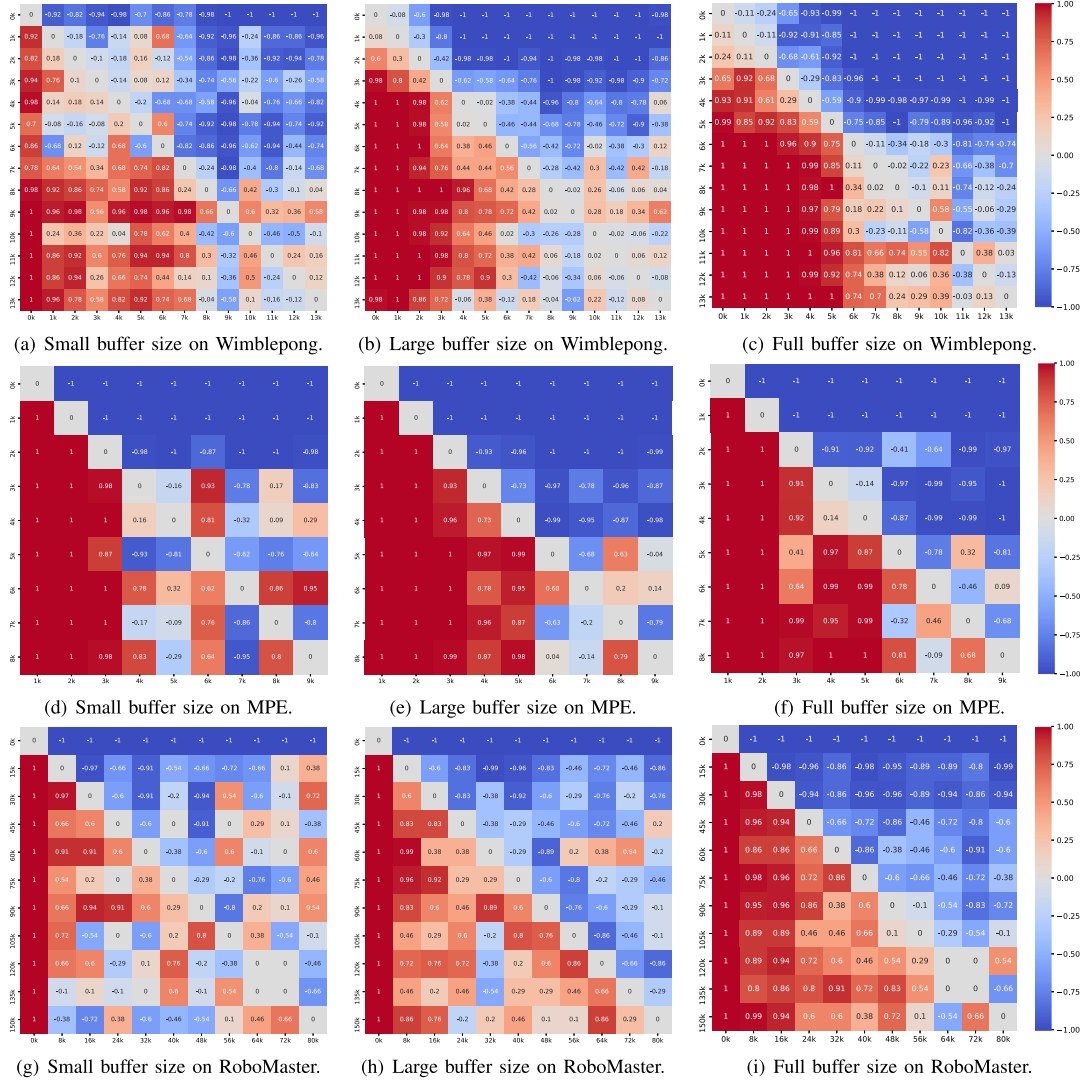
Fig. 6. Illustration of the payoff values of FM3Q populations in the three scenarios under different replay buffer sizes. In each scenario, from left to right, the red squares on the lower triangle portion of each table become darker as the buffer size gradually increases. In other words, the proportion of new policies that outperform the old policies increases, and the degree of outperformance also becomes higher.

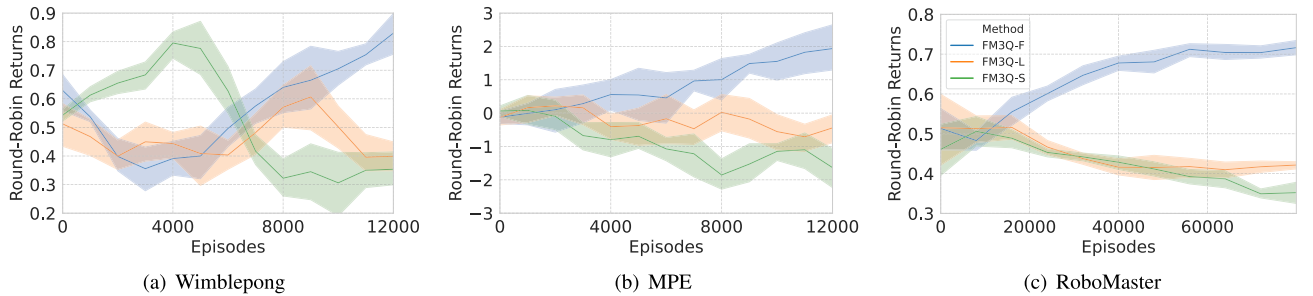(a) Wimblepong　(b) MPE　(c) RoboMaster

Fig. 7. Illustration of the Round-Robin returns of FM3Q-S, FM3Q-L, and FM3Q-F during training.

are unattainable in the current experimental setting. Therefore, we resort to using an approximate NashConv. We train the RL-based agents independently, allowing them to compete against periodically stored models for evaluating the performance improvements achievable by these RL-based agents. Less

exploitable agents should attain a lower approximate NashConv than all other agents. The approximate Nashconv curves of models trained in three different environments are depicted in Fig. 5. It can be observed that, across these three environments, the approximate Nashconv of FM3Q consistently exhibits a

decreasing trend throughout the training process, eventually converging to very small values. Furthermore, it outperforms the contemporaneous models from other algorithms, indicating that FM3Q possesses superior exploitability. NXDO and PSRO also demonstrate decreasing trends, with their values trailing behind FM3Q but surpassing those of SP and EPO during the same time period. Notably, the Nashconv curve of SP exhibits significant fluctuations in the Pong and MPE environments, while EPO attains the highest level of exploitability.

C. Optimization Trend of FM3Q

We then investigate the performance of FM3Q across different training phases and display the results using payoff values calculated using the average winning rates, or returns by cross-play. Fig. 6(c), (f) and (i) show the payoff tables for Wimblepong, MPE, and RoboMaster, respectively. Taking Fig. 6(c) as an example, we save a total of 14 models at different phases. These models are evaluated using the round-robin tournament results, and the average winning rate between each pair of models is presented in the form of a payoff value. The red squares indicate victories, with darker shades representing higher winning rates. Conversely, the blue squares indicate defeats, with darker shades representing lower winning rates.

It can be seen that the squares in the lower triangle of the tables are almost all red, and that the squares in the same row always have darker shading on the left than on the right. So, the first thing we can say is that the models that are trained later can almost always do better than the ones that are trained earlier.

Of course, in the lower triangle of each of the three payoff tables, there are a few individual cells that are colored blue. However, the proportion of these cells is very small, meaning that the probability of the models trained later being defeated by the earlier ones is extremely low. In the lower triangle of Fig. 6(c), (f), and (i), these cells only account for $5/196$, $2/64$, and $1/121$, respectively. It can be concluded from this phenomenon that, the later the model is learned, the better it will perform.

D. Ablation Study

We argue that using a larger buffer as much as possible is conducive to training FM3Q; we even need to use all the data generated by interaction in some tasks. We investigate the effect of historical experience in FM3Q and demonstrate the above argument by running experiments with different sizes of replay buffer. For the purpose of distinguishing among different replay buffer sizes, we refer to the FM3Q trained on small, large, and full buffer sizes as FM3Q-S, FM3Q-L, and FM3Q-F, respectively.

First, we investigate the effect of historical experience in the optimization tread of FM3Q. We compare their performance across different training phases and display the results using payoff values calculated using the average winning rates, or returns. Fig. 6 shows the learning performance of FM3Q-S, FM3Q-L, and FM3Q-F in different replay buffer size sets in the three scenarios. The replay buffer size from left to right in each scenario (each row) decreases in turn. We focus on the lower triangle portion of each table and find that, as replay buffer size gradually increases, the red part becomes more and more dark, while the blue part becomes less and lighter. It is concluded that, as the buffer size increases, the proportion of new policies that outperform the old policies increases, and the degree of outperformance also becomes higher. A small replay buffer causes instability in FM3Q's learning. It is worth noting that a complete or sufficient replay of historical experience causes FM3Q to update policies in a transitive or monotone mode. Therefore, the results are consistent with the argument.

Second, we also utilize the RR returns as comparative metrics to estimate the exploitability of FM3Q with each buffer size against all other sizes. We evaluate all pairs of methods in cross-play during their training processes. Less exploitable agents should attain a higher RR returns than all other agents. Fig. 7 displays the performance of FM3Q-S, FM3Q-L, and FM3Q-F on the three scenarios. We can see that FM3Q-F finally outperforms FM3Q-S and FM3Q-L in RR returns. Therefore, we can conclude that an increase in replay buffer size contributes to the improvement of the training effectiveness and performance of the FM3Q.

VII. Conclusion

To design an efficient MARL framework for 2t0sMGs, we propose the IGMM to extend factorizable tasks to 2t0sMGs. Leveraging the IGMM principle, we introduce FM3Q and its online learning algorithm based on CTDE to factorize the joint minimax Q function into individual ones while synchronously optimizing the networks of all agents. Additionally, we prove the convergence of FM3Q and empirically demonstrate the superiority of FM3Q over existing methods in terms of learning efficiency and final performance. However, there is scope for improvement in the FM3Q algorithm. First of all, because FM3Q's policy is deterministic, it can be challenging or impossible to identify the exact NE in some competitive tasks that only use mixed (stochastic) policies. Secondly, FM3Q retains historical experience, requiring all experience to be trained before updating the target network. This results in increased hardware costs for storage and computation, particularly in large games. We suggest incorporating stochastic policy methods into FM3Q to ensure convergence to the NE. In addition, the optimization efficiency can be improved by performing quality filtering on the data in the buffer to remove low-quality samples.

Appendix

For the sake of fairness, we utilize QMIX as an individual in the SP, PSRO and NXDO policy populations. FM3Q, SP, PSRO, and NXDO are realized in the framework of QMIX implementation. The important hyperparameters of all methods in experiments are listed in Table I. n_ep, mix_hidden_dim, n_gene, and ep_per_gene represent the number of training episodes, the dimension of the hidden layers in the mixing networks, the number of generations, and the number of training episodes per generation, respectively. Details about our project, including the codes and implementation, can be accessed at https://github.com/bithgz/fm3q/tree/master.

TABLE I
IMPORTANT HYPERPARAMETERS OF ALL METHODS IN EXPERIMENTS

Alg.	Hyperparameters	Wimblepong	MPE	RoboMaster
Common	n_episodes	1.3e4	1.3e4	8e4
	n_seeds	8	8	8
	gamma	0.99	0.98	0.99
	hidden_layers	[64,64]	[64,64]	[128,128]
	mix_hidden_dim	32	32	32
	learning_rate	5e-4	5e-4	5e-4
FM3Q	full buffer_size	4e6	4e5	4e6
	large buffer_size	1e6	8e4	1e6
	small buffer_size	2e5	2e4	4e5
SP/ PSRO/ NXDO	n_genes	13	13	10
	ep_per_gene	1e3	1e3	8e4
	batch_size	1e3	1e3	2e3
	buffer_size	2e5	2e4	2e5
EPO	pi_learning_rate	3e-4	3e-4	3e-4
	vf_learning_rate	1e-3	1e-3	1e-3
	train_iters	80	80	80
	target_kl	0.02	0.02	0.02

REFERENCES

[1] M. L. Littman, "Markov games as a framework for multi-agent reinforcement learning," in *Machine Learning Proceedings*, 1994. Amsterdam, The Netherlands: Elsevier, 1994, pp. 157–163.

[2] Y. Chen and H. Wang, "IntelligentCrowd: Mobile crowdsensing via multi-agent reinforcement learning," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 5, no. 5, pp. 840–845, Oct. 2021.

[3] N. Anastassacos, S. Hailes, and M. Musolesi, "Partner selection for the emergence of cooperation in multi-agent systems using reinforcement learning," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 7047–7054.

[4] C. Zheng, S. Yang, J. M. Parra-Ullauri, A. Garcia-Dominguez, and N. Bencomo, "Reward-reinforced generative adversarial networks for multi-agent systems," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 6, no. 3, pp. 479–488, Jun. 2021.

[5] D. Silver et al., "Mastering the game of go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[6] D. Silver et al., "A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play," *Science*, vol. 362, no. 6419, pp. 1140–1144, 2018.

[7] O. Vinyals et al., "Grandmaster level in StarCraft II using multi-agent reinforcement learning," *Nature*, vol. 575, no. 7782, pp. 350–354, 2019.

[8] Z. Tang, Y. Zhu, D. Zhao, and S. M. Lucas, "Enhanced rolling horizon evolution algorithm with opponent model learning," *IEEE Trans. Games*, vol. 15, no. 1, pp. 5–15, Mar. 2023.

[9] J. Hu and M. P. Wellman, "Nash Q-learning for general-sum stochastic games," *J. Mach. Learn. Res.*, vol. 4, pp. 1039–1069, 2003.

[10] M. L. Littman, "Markov Games as a Framework for Multi-Agent Reinforcement Learning," in *Machine Learning Proceedings*. Amsterdam, The Netherlands: Elsevier, 1994, pp. 157–163.

[11] M. L. Littman et al., "Friend-or-foe Q-learning in general-sum games," in *Proc. Int. Conf. Mach. Learn.*, 2001, pp. 322–328.

[12] X. Feng et al., "Neural auto-curricula in two-player zero-sum games," in *Proc. Adv. Neural Inf. Process. Syst.*, 2021, pp. 3504–3517.

[13] M. Zhou et al., "MALib: A parallel framework for population-based multi-agent reinforcement learning," *J. Mach. Learn. Res.*, vol. 24, pp. 1–12, 2023.

[14] C. Berner et al., "Dota 2 with large scale deep reinforcement learning," 2019, *arXiv:1912.06680*.

[15] S. McAleer, J. B. Lanier, R. Fox, and P. Baldi, "Pipeline PSRO: A scalable approach for finding approximate Nash equilibria in large games," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 20238–20248.

[16] M. Lanctot et al., "A unified game-theoretic approach to multiagent reinforcement learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 4193–4206.

[17] D. Balduzzi et al., "Open-ended learning in symmetric zero-sum games," in *Proc Int. Conf. Mach. Learn.*, 2019, pp. 434–443.

[18] S. McAleer, K. A. Wang, P. Baldi, and R. Fox, "XDO: A double oracle algorithm for extensive-form games," in *Proc. Adv. Neural Inf. Process. Syst.*, 2021, pp. 23128–23139.

[19] K. Shao, Y. Zhu, and D. Zhao, "StarCraft micromanagement with reinforcement learning and curriculum transfer learning," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 3, no. 1, pp. 73–84, Feb. 2019.

[20] D. Ye et al., "Mastering complex control in moba games with deep reinforcement learning," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 6672–6679.

[21] A. Celli, M. Ciccone, R. Bongo, and N. Gatti, "Coordination in adversarial sequential team games via multi-agent deep reinforcement learning," 2019, *arXiv:1912.07712*.

[22] F. Kalogiannis, I. Anagnostides, I. Panageas, E.-V. Vlatakis-Gkaragkounis, V. Chatziafratis, and S. A. Stavroulakis, "Efficiently computing nash equilibria in adversarial team Markov games," in *Proc. Int. Conf. Learn. Representations*, 2023, pp. 1–15.

[23] C. Yu, A. Velu, E. Vinitsky, Y. Wang, A. Bayen, and Y. Wu, "The surprising effectiveness of PPO in cooperative, multi-agent games," in *Proc. Adv. Neural Inf. Process. Syst.*, 2022, pp. 24611–24624.

[24] T. Zhang, Z. Liu, Z. Pu, and J. Yi, "Automatic curriculum learning for large-scale cooperative multiagent systems," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 7, no. 3, pp. 912–930, Jun. 2023.

[25] G. Hu, Y. Zhu, D. Zhao, M. Zhao, and J. Hao, "Event-triggered communication network with limited-bandwidth constraint for multi-agent reinforcement learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 8, pp. 3966–3978, Aug. 2023.

[26] B.-C. Lai, W.-Y. Chiu, and Y.-P. Tsai, "Multiagent reinforcement learning for community energy management to mitigate peak rebounds under renewable energy uncertainty," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 6, no. 3, pp. 568–579, Jun. 2022.

[27] Y. Zhu and D. Zhao, "Online minimax Q network learning for two-player zero-sum Markov games," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 3, pp. 1228–1241, Mar. 2022.

[28] P. Muller et al., "A generalized training approach for multiagent learning," in *Proc. Int. Conf. Learn. Representations*, 2020, pp. 1–35.

[29] I. Anagnostides, F. Kalogiannis, I. Panageas, E.-V. Vlatakis-Gkaragkounis, and S. McAleer, "Algorithms and complexity for computing Nash equilibria in adversarial team games," in *Proc. ACM Conf. Eco. Comput.*, 2023, p. 89.

[30] G. Luo, H. Zhang, H. He, J. Li, and F.-Y. Wang, "Multiagent adversarial collaborative learning via mean-field theory," *IEEE Trans. Cybern.*, vol. 51, no. 10, pp. 4994–5007, Oct. 2021.

[31] Y. Zhu, W. Li, M. Zhao, J. Hao, and D. Zhao, "Empirical policy optimization for n-player Markov games," *IEEE Trans. Cybern.*, vol. 53, no. 10, pp. 6443–6455, Oct. 2023.

[32] T. Rashid, M. Samvelyan, C. Schroeder, G. Farquhar, J. Foerster, and S. Whiteson, "QMIX: Monotonic value function factorisation for deep multi-agent reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 4295–4304.

[33] P. Sunehag et al., "Value-decomposition networks for cooperative multi-agent learning based on team reward," in *Proc. Int. Conf. Auton. Agents Multiagent Syst.*, 2018, pp. 2085–2087.

[34] T. Rashid, G. Farquhar, B. Peng, and S. Whiteson, "Weighted QMIX: Expanding monotonic value function factorisation for deep multi-agent reinforcement learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 10199–10210.

[35] J. Chai et al., "UNMAS: Multiagent reinforcement learning for unshaped cooperative scenarios," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 4, pp. 2093–2104, Apr. 2023.

[36] M. Riedmiller, "Neural fitted Q iteration–first experiences with a data efficient neural reinforcement learning method," in *Proc. Eur. Conf. Mach. Learn.*, 2005, pp. 317–328.

[37] A. Antos, C. Szepesvári, and R. Munos, "Fitted Q-iteration in continuous action-space MDPs," in *Proc. Adv. Neural Inf. Process. Syst.*, 2007, pp. 9–16.

[38] A.-M. Farahmand, C. Szepesvári, and R. Munos, "Error propagation for approximate policy and value iteration," in *Proc. Adv. Neural Inf. Process. Syst.*, 2010, pp. 568–576.

[39] J. Wang, Z. Ren, B. Han, J. Ye, and C. Zhang, "Towards understanding cooperative multi-agent Q-learning with value factorization," in *Proc. Adv. Neural Inf. Process. Syst.*, 2021, pp. 29142–29155.

[40] X. Liu et al., "Towards unifying behavioral and response diversity for open-ended learning in zero-sum games," in *Proc. Adv. Neural Inf. Process. Syst.*, 2021, pp. 941–952.

[41] V. Mnih et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[42] J. Terry et al., "PettingZoo: Gym for multi-agent reinforcement learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2021, pp. 15032–15043.

[43] G. Hu, H. Li, S. Liu, M. Ma, and D. Zhao, "NeuronsMAE: A novel multi-agent reinforcement learning environment for cooperative and competitive multi-robot tasks," in *Proc. Int. Joint Conf. Neural Netw.*, 2023, pp. 1–8.

[44] M. Samvelyan et al., "MAESTRO: Open-ended environment design for multi-agent reinforcement learning," in *Proc. Int. Conf. Learn. Representations*, 2023, pp. 1–29.

[45] F. Timbers et al., "Approximate exploitability: Learning a best response," in *Proc. Int. Joint Conf. Artif. Intell.*, 2022, pp. 3487–3493.
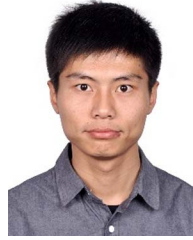
Haoran Li (Member, IEEE) received the B.S. degree in detection, guidance, and control technology from Central South University, Changsha, China, in 2015, and the Ph.D. degree in control theory and control engineering with the Institute of Automation, Chinese Academy of Sciences, Beijing, China, in 2020. He is currently an Associate Professor with the Institute of Automation, Chinese Academy of Sciences, Beijing, and College of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing. His research interests include deep learning, deep reinforcement learning, and robotic learning.

Guangzheng Hu (Graduate Student Member, IEEE) received the B.S. and M.E. degrees in automation from the Beijing Institute of Technology, Beijing, China, in 2016 and 2019, respectively. He is currently working toward the Ph.D. degree in computer applications with The State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing, and the School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing. His research interests include multi-agent reinforcement learning and multi-robot games.

Yuanheng Zhu (Senior Member, IEEE) received the B.S. degree in automation from Nanjing University, Nanjing, China, in 2010, and the Ph.D. degree in control theory and control engineering from the Institute of Automation, Chinese Academy of Sciences, Beijing, China, in 2015. From 2015 to 2017, he was an Assistant Professor with the Institute of Automation, Chinese Academy of Sciences, where he is currently an Associate Professor. From 2017 to 2018, he was a Visiting Scholar with the Department of Electrical, Computer, and Biomedical Engineering, University of Rhode Island, Kingston, RI, USA. His research interests include deep reinforcement learning, game theory, game intelligence, and multiagent learning. He is an Associate Editor for IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS and IEEE TRANSACTIONS ON GAMES.

Dongbin Zhao (Fellow, IEEE) received the B.S., M.S., and Ph.D. degrees from the Harbin Institute of Technology, Harbin, China, in 1994, 1996, and 2000, respectively. He is currently a Professor with the Institute of Automation, Chinese Academy of Sciences, Beijing, and School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing. He has authored or coauthored seven books, and more than 100 international journal papers. His research interests include deep reinforcement learning, autonomous driving, game artificial intelligence, and robotics. He was the recipient of the 2022 Best Paper Reward of IEEE Transactions on Automation Science and Engineering, 2022 Outstanding Paper Reward of IEEE Transactions on Emerging Topics in Computational Intelligence, and 2020 Outstanding Paper Reward of IEEE Transactions on Cognitive and Developmental Systems. He is an Associate Editor for IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS: SYSTEMS and IEEE TRANSACTIONS ON ARTIFICIAL INTELLIGENCE. He is the Guest Editor of renowned international journals. He is involved in organizing many international conferences.