Multi-Agent Alternate Q-Learning

Kefan Su
School of Computer Science,
Peking University
China
sukefan@pku.edu.cn

Siyuan Zhou
HKUST
China
elegyhunter@gmail.com

Jiechuan Jiang
School of Computer Science,
Peking University
China
jiechuan.jiang@pku.edu.cn

Chuang Gan
MIT-IBM Watson AI Lab
US
ganchuang@csail.mit.edu

Xiangjun Wang
inspir.ai
China
xj@inspirai.com

Zongqing Lu[†]
Peking University
China
zongqing.lu@pku.edu.cn

ABSTRACT

Decentralized learning has shown great promise for cooperative multi-agent reinforcement learning (MARL). However, non-stationarity remains a significant challenge in fully decentralized learning. In the paper, we tackle the non-stationarity problem in the simplest and fundamental way and propose *multi-agent alternate Q-learning* (MA2QL), where agents take turns updating their Q-functions by Q-learning. MA2QL is a *minimalist* approach to fully decentralized cooperative MARL but is theoretically grounded. We prove that when each agent guarantees ε-convergence at each turn, their joint policy converges to a Nash equilibrium. In practice, MA2QL only requires minimal changes to independent Q-learning (IQL). We empirically evaluate MA2QL on a variety of cooperative multi-agent tasks. Results show MA2QL consistently outperforms IQL, which verifies the effectiveness of MA2QL, despite such minimal changes.

KEYWORDS

Reinforcement learning; Fully decentralized learning; Q-learning

ACM Reference Format:
Kefan Su, Siyuan Zhou, Jiechuan Jiang, Chuang Gan, Xiangjun Wang, and Zongqing Lu. 2024. Multi-Agent Alternate Q-Learning. In *Proc. of the 23rd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2024), Auckland, New Zealand, May 6 – 10, 2024*, IFAAMAS, 9 pages.

1 INTRODUCTION

Cooperative multi-agent reinforcement learning (MARL) is a well-abstracted model for a broad range of real applications, including logistics [10], traffic signal control [33], power dispatch [30], and inventory management [4]. In cooperative MARL, centralized training with decentralized execution (CTDE) is a popular learning paradigm, where the information of all agents can be gathered and used in training. Many CTDE methods [5, 11, 13, 20, 23, 24, 29, 31, 38] have been proposed and shown great potential to solve cooperative multi-agent tasks.

Another paradigm is decentralized learning, where each agent learns its policy based on only local information. Decentralized learning is less investigated but desirable in many scenarios where the information of other agents is not available, and for better robustness, scalability, and security [36]. However, *fully decentralized learning* of agent policies (*i.e.*, without communication) is still an open challenge in cooperative MARL.

The most straightforward way for fully decentralized learning is directly applying independent learning at each agent [27], which however induces the well-known non-stationarity problem for all agents [36] and may lead to learning instability and a non-convergent joint policy, though the performance varies as shown in empirical studies [3, 18, 20, 34].

In the paper, we directly tackle the non-stationarity problem in the simplest and fundamental way, *i.e.*, keeping the policies of other agents fixed while one agent is learning. Following this principle, we propose ***multi-agent alternate Q-learning (MA2QL)***, a *minimalist* approach to fully decentralized cooperative multi-agent reinforcement learning, where agents take turns to update their policies by Q-learning. MA2QL is theoretically grounded and we prove that when each agent guarantees ε-convergence at each turn, their joint policy converges to a Nash equilibrium. In practice, MA2QL only requires minimal changes to independent Q-learning (IQL) [26, 27] and also independent DDPG [12] for continuous action, *i.e.*, simply swapping the order of two lines of codes as follows.

IQL

repeat
  all agents interact in the environment
  for $i \leftarrow 1, n$ **do**
    agent i updates by Q-learning
  end for
until terminate

MA2QL

repeat
  for $i \leftarrow 1, n$ **do**
    all agents interact in the environment
    agent i updates by Q-learning
  end for
until terminate

[†]Corresponding author

Proc. of the 23rd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2024), N. Alechina, V. Dignum, M. Dastani, J.S. Sichman (eds.), May 6 – 10, 2024, Auckland, New Zealand. © 2024 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org).

We evaluate MA2QL on a didactic game to empirically verify its convergence, and multi-agent particle environments [13], multi-agent MuJoCo [19], and StarCraft multi-agent challenge [21] to verify its performance with discrete and continuous action spaces, and fully and partially observable environments. We find that MA2QL consistently outperforms IQL, despite such minimal changes. This empirically verifies the effectiveness of alternate learning. The superiority of MA2QL over IQL suggests that simpler approaches may have been left underexplored for fully decentralized cooperative multi-agent reinforcement learning. We envision this work could provide some insights to further studies of fully decentralized learning.

2 BACKGROUND

2.1 Preliminaries

Dec-POMDP. Decentralized partially observable Markov decision process (Dec-POMDP) is a general model for cooperative MARL. A Dec-POMDP is a tuple $M = \{S, A, P, Y, O, I, n, r, \gamma\}$. S is the state space, n is the number of agents, $\gamma \in [0, 1)$ is the discount factor, and $I = \{1, 2 \cdots n\}$ is the set of all agents. $A = A_1 \times A_2 \times \cdots \times A_n$ represents the joint action space where A_i is the individual action space for agent i. $P(s'|s, \boldsymbol{a}) : S \times A \times S \to [0, 1]$ is the transition function, and $r(s, \boldsymbol{a}) : S \times A \to \mathbb{R}$ is the reward function of state s and joint action $\boldsymbol{a}$. Y is the observation space, and $O(s, i) : S \times I \to Y$ is a mapping from state to observation for each agent. The objective of Dec-POMDP is to maximize $J(\boldsymbol{\pi}) = \mathbb{E}_{\boldsymbol{\pi}}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, \boldsymbol{a}_t)\right]$, and thus we need to find the optimal joint policy $\boldsymbol{\pi}^* = \arg\max_{\boldsymbol{\pi}} J(\boldsymbol{\pi})$. To settle the partial observable problem, history $\tau_i \in \mathcal{T}_i : (Y \times A_i)^*$ is often used to replace observation $o_i \in Y$. Each agent i has an individual policy $\pi_i(a_i|\tau_i)$ and the joint policy $\boldsymbol{\pi}$ is the product of each π_i. Though the individual policy is learned as $\pi_i(a_i|\tau_i)$ in practice, as Dec-POMDP is undecidable [14] and the analysis in partially observable environments is much harder, we will use $\pi_i(a_i|s)$ in analysis and proofs for simplicity.

Dec-MARL. Although decentralized cooperative multi-agent reinforcement learning (Dec-MARL) has been previously investigated [3, 37], the setting varies across these studies. In this paper, we consider Dec-MARL as a *fully* decentralized solution to Dec-POMDP, where each agent learns its policy/Q-function from its own action individually ***without communication or parameter-sharing***. Therefore, in Dec-MARL, each agent i actually learns in the environment with transition function $P_i(s'|s, a_i) = \mathbb{E}_{a_{-i} \sim \pi_{-i}}[P(s'|s, a_i, a_{-i})]$ and reward function $r_i(s, a_i) = \mathbb{E}_{a_{-i} \sim \pi_{-i}}[r(s, a_i, a_{-i})]$, where π_{-i} and a_{-i} respectively denote the joint policy and joint action of all agents except for i. As other agents are also learning (*i.e.*, π_{-i} is changing), from the perspective of each individual agent, the environment is non-stationary. This is the non-stationarity problem, the main challenge in Dec-MARL.

IQL. Independent Q-learning (IQL) is a straightforward method for Dec-MARL, where each agent i learns a Q-function $Q(s, a_i)$ by Q-learning. However, as all agents learn simultaneously, there is no theoretical guarantee on convergence due to non-stationarity, to the best of our knowledge. In practice, IQL is often taken as a simple baseline in favor of more elaborate MARL approaches, such as value-based CTDE methods [20, 22]. However, much less attention has been paid to IQL itself for Dec-MARL.

2.2 Multi-Agent Alternate Policy Iteration

To address the non-stationarity problem in Dec-MARL, a fundamental way is simply to make the environment stationary during the learning of each agent. Following this principle, we let agents learn by turns; in each turn, one agent performs policy iteration while fixing the policies of other agents. This procedure is referred to as *multi-agent alternate policy iteration*. As illustrated in Figure 1, multi-agent alternate policy iteration differs from policy iteration in single-agent RL. In single-agent RL, policy iteration is performed on the same MDP. However, here, for each agent, policy iteration at a different round is performed on a different MDP. As π_{-i} is fixed at each turn, $P_i(s'|s, a_i)$ and $r_i(s, a_i)$ are stationary and we can easily have the following lemma.

Lemma 1 (multi-agent alternate policy iteration). *If all agents take turns to perform policy iteration, their joint policy sequence $\{\boldsymbol{\pi}\}$ monotonically improves and converges to a Nash equilibrium.*

Proof. In each turn, as the policies of other agents are fixed, the agent i has the following update rule for policy evaluation,

$$Q_{\pi_i}(s, a_i) \leftarrow r_i(s, a_i) + \gamma \mathbb{E}_{s' \sim P_i, a_i' \sim \pi_i}[Q_{\pi_i}(s', a_i')]. \tag{1}$$

We can have the convergence of policy evaluation in each turn by the standard results [25]. Moreover, as π_{-i} is fixed, it is straightforward to have

$$Q_{\pi_i}(s, a_i) = \mathbb{E}_{a_{-i} \sim \pi_{-i}}[Q_{\boldsymbol{\pi}}(s, a, a_i)]. \tag{2}$$

Then, the agent i performs policy improvement by

$$\pi_i^{\text{new}}(s) = \arg\max_{a_i} \mathbb{E}_{\pi_{-i}^{\text{old}}}\left[Q_{\boldsymbol{\pi}^{\text{old}}}(s, a_i, a_{-i})\right]. \tag{3}$$

As the policies of other agents are fixed (i.e., $\pi_{-i}^{\text{new}} = \pi_{-i}^{\text{old}}$), we have

$$\begin{aligned} V_{\boldsymbol{\pi}^{\text{old}}}(s) &= \mathbb{E}_{\boldsymbol{\pi}^{\text{old}}}[Q_{\boldsymbol{\pi}^{\text{old}}}(s, a_i, a_{-i})] \\ &= \mathbb{E}_{\pi_i^{\text{old}}} \mathbb{E}_{\pi_{-i}^{\text{old}}}[Q_{\boldsymbol{\pi}^{\text{old}}}(s, a_i, a_{-i})] \\ &\le \mathbb{E}_{\pi_i^{\text{new}}} \mathbb{E}_{\pi_{-i}^{\text{old}}}[Q_{\boldsymbol{\pi}^{\text{old}}}(s, a_i, a_{-i})] \\ &= \mathbb{E}_{\pi_i^{\text{new}}} \mathbb{E}_{\pi_{-i}^{\text{new}}}[Q_{\boldsymbol{\pi}^{\text{old}}}(s, a_i, a_{-i})] \\ &= \mathbb{E}_{\boldsymbol{\pi}^{\text{new}}}[Q_{\boldsymbol{\pi}^{\text{old}}}(s, a_i, a_{-i})] \\ &= \mathbb{E}_{\boldsymbol{\pi}^{\text{new}}}[r(s, a_i, a_{-i}) + \gamma V_{\boldsymbol{\pi}^{\text{old}}}(s')] \\ &\le \cdots \le V_{\boldsymbol{\pi}^{\text{new}}}(s), \end{aligned} \tag{4}$$

where the first inequality is from (3). This proves that the policy improvement of agent i in each turn also improves the joint policy. Thus, as agents perform policy iteration by turn, the joint policy sequence $\{\boldsymbol{\pi}\}$ improves monotonically, and $\{\boldsymbol{\pi}\}$ will converge to a Nash equilibrium since no agents can improve the joint policy unilaterally at convergence. □

Lemma 1 is simple but useful, which is also reached implicitly by Bertsekas [2]. Moreover, Lemma 1 immediately indicates an approach with the convergence guarantee for Dec-MARL and also tells us that if we find the optimal policy for agent i in each round k given the other agents' policies π_{-i}^k, then the joint policy will obtain the largest improvement. This result can be formulated as

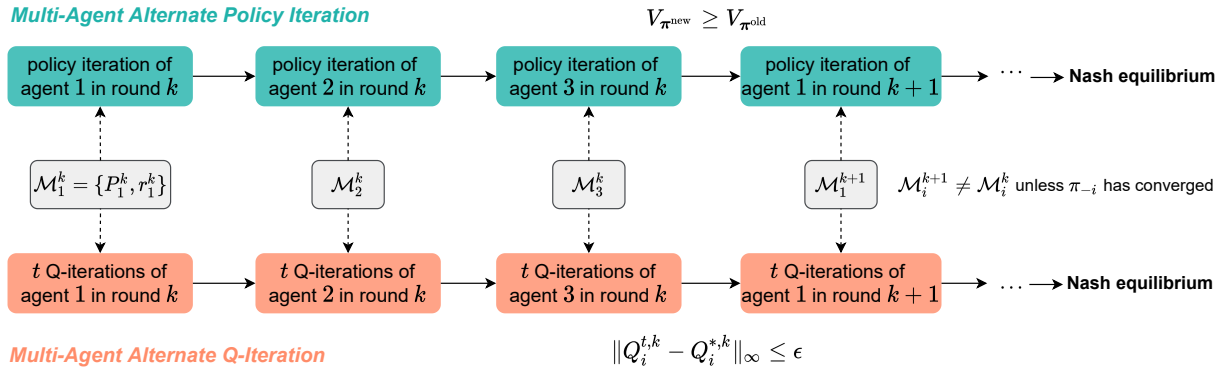


Figure 1: Illustration of *multi-agent alternate policy iteration* (upper panel) and *multi-agent alternate Q-iteration* (lower panel) of three agents. As essentially the MDP differs at different turns of each agent, policy iteration/Q-iteration of each agent iterates over different MDPs.

follows,

$$\begin{aligned} \pi_i^{*,k} &= \arg\max_{\pi_i} \mathbb{E}_{\pi_{-i}^k}\left[Q_{\pi_i,\pi_{-i}^k}(s, a_i, a_{-i})\right] \\ V_{\pi_i,\pi_{-i}^k}(s) &\leq V_{\pi_i^{*,k},\pi_{-i}^k}(s) \quad \forall \pi_i, \forall s. \end{aligned} \tag{5}$$

We could obtain this $\pi_i^{*,k}$ by policy iteration with many *on-policy* iterations. However, such a method will face the issue of sample inefficiency which may be amplified in MARL settings. We will use Q-iteration to settle this problem as in Bertsekas [2]. However, unlike existing work, we propose to truncate Q-iteration for fast learning but with the same theoretical guarantee and additionally focus on the *empirical performance* of such an approach.

3 METHOD

To address the problem of multi-agent alternate policy iteration, we propose *multi-agent alternate Q-iteration*, which is sufficiently truncated for fast learning but still has the same theoretical guarantee. Further, based on multi-agent alternate Q-iteration, we derive *multi-agent alternate Q-learning*, which makes the minimal change to IQL to form a simple yet effective value-based decentralized learning method for cooperative MARL.

3.1 Multi-Agent Alternate Q-Iteration

Instead of policy iteration, we let agents perform Q-iteration by turns as depicted in Figure 1. Let $\mathcal{M}_i^k = \{P_i^k, r_i^k\}$ denote the MDP of agent i in round k, where we have $\mathcal{M}_i^k \neq \mathcal{M}_i^{k-1}$ unless π_{-i} has converged, and $Q_i^{t,k}(s, a_i)$ denote the Q-function of agent i with t updates in the round k. We define the Q-iteration as follows,

$$Q_i^{t+1,k}(s, a_i) \leftarrow r_i^k(s, a_i) + \gamma \mathbb{E}_{s' \sim P_i^k}\left[\max_{a_i'} Q_i^{t,k}(s', a_i')\right]. \tag{6}$$

Then, the sequence $\{Q_i^{t,k}\}$ converges to $Q_i^{*,k}$ with respect to the MDP $\mathcal{M}_i^k = \{P_i^k, r_i^k\}$, and we have the following lemma and corollary.

Lemma 2 (ε-convergent Q-iteration). *By iteratively applying Q-iteration* (6) *at each agent i for each turn, for any $\varepsilon > 0$, we have*

$$\left\|Q_i^{t,k} - Q_i^{*,k}\right\|_\infty \leq \varepsilon, \text{ when } t \geq \frac{\log\left((1-\gamma)\varepsilon\right) - \log(2R + 2\varepsilon)}{\log \gamma}, \tag{7}$$

where $R = \frac{r_{\max}}{1-\gamma}$ and $r_{\max} = \max_{s,\boldsymbol{a}} r(s, \boldsymbol{a})$.

Proof. From the definition of $Q_i^{t,k}$ (6), we have

$$\begin{aligned} &\left\|Q_i^{t+1,k} - Q_i^{t,k}\right\|_\infty \\ &= \left\|\gamma \mathbb{E}_{s' \sim P_i^k}\left[\max_{a_i'} Q_i^{t,k}(s', a_i') - \max_{a_i'} Q_i^{t-1,k}(s', a_i')\right]\right\|_\infty \\ &\leq \gamma \left\|Q_i^{t,k} - Q_i^{t-1,k}\right\|_\infty \leq \gamma^t \left\|Q_i^{1,k} - Q_i^{0,k}\right\|_\infty. \end{aligned} \tag{8}$$

Then for any integer $m \geq 1$, we have

$$\begin{aligned} &\left\|Q_i^{t+m,k} - Q_i^{t,k}\right\|_\infty \\ &\leq \left\|Q_i^{t+m,k} - Q_i^{t+m-1,k}\right\|_\infty + \cdots + \left\|Q_i^{t+1,k} - Q_i^{t,k}\right\|_\infty \\ &\leq \gamma^t \frac{1-\gamma^m}{1-\gamma} \left\|Q_i^{1,k} - Q_i^{0,k}\right\|_\infty. \end{aligned} \tag{9}$$

Let $m \to \infty$, and we have

$$\begin{aligned} &\left\|Q_i^{*,k} - Q_i^{t,k}\right\|_\infty \\ &\leq \frac{\gamma^t}{1-\gamma} \left\|Q_i^{1,k} - Q_i^{0,k}\right\|_\infty \\ &\leq \frac{\gamma^t}{1-\gamma} \max_{s,a_i} \left|r_i^k(s, a_i) + \gamma \mathbb{E}_{s' \sim P_i^k}\left[\max_{a_i'} Q_i^{0,k}(s', a_i')\right] - Q_i^{0,k}(s, a_i)\right|. \end{aligned} \tag{10}$$

As all agents update by turns, we have $Q_i^{0,k} = Q_i^{t_i^{k-1},k-1}$, where t_i^{k-1} is the number of Q-iteration for agent i in the $k-1$ round. Therefore, we have

$$\left\|Q_i^{0,k} - Q_i^{*,k-1}\right\|_\infty = \left\|Q_{i-1}^{t_i^{k-1},k-1} - Q_i^{*,k-1}\right\|_\infty \leq \varepsilon. \tag{11}$$

With this property, we have

$$\begin{aligned} &\left|r_i^k(s, a_i) + \gamma \mathbb{E}_{s' \sim P_i^k}\left[\max_{a_i'} Q_i^{0,k}(s', a_i')\right] - Q_i^{0,k}(s, a_i)\right| \\ &= \left|r_i^k(s, a_i) + \gamma \mathbb{E}_{s' \sim P_i^k}\left[\max_{a_i'} Q_i^{0,k}(s', a_i')\right] - Q_i^{*,k-1}(s, a_i) \right. \\ &\quad \left. + Q_i^{*,k-1}(s, a_i) - Q_i^{0,k}(s, a_i)\right| \\ &\leq \left|r_i^k - r_i^{k-1}\right| + \gamma \left|\mathbb{E}_{s' \sim P_i^k}\left[\max_{a_i'} Q_i^{0,k}(s', a_i')\right]\right. \\ &\quad \left. - \mathbb{E}_{s' \sim P_i^{k-1}}\left[\max_{a_i'} Q_i^{*,k-1}(s', a_i')\right]\right| + \left|Q_i^{*,k-1}(s, a_i) - Q_i^{0,k}(s, a_i)\right| \\ &\leq 2r_{\max} + \left(\frac{2\gamma r_{\max}}{1-\gamma} + \varepsilon\right) + \varepsilon = 2R + 2\varepsilon, \end{aligned} \tag{12}$$

where the second term in the last inequality is from

$$\|Q_i^{*,k-1}\|_\infty \le \frac{r_{\max}}{1-\gamma}, \|Q_i^{0,k}\|_\infty \le \|Q_i^{*,k-1}\|_\infty + \varepsilon,$$

and (11). Finally, by combining (10) and (12), we have

$$\left\|Q_i^{*,k} - Q_i^{t,k}\right\|_\infty \le \frac{\gamma^t}{1-\gamma}(2R + 2\varepsilon). \tag{13}$$

We need $\|Q_i^{*,k} - Q_i^{t,k}\|_\infty \le \varepsilon$, which can be guaranteed by

$$t \ge \frac{\log((1-\gamma)\varepsilon) - \log(2R + 2\varepsilon)}{\log\gamma}.$$

□

With Lemma 2, we can immediately obtain the following corollary.

Corollary 1. *For any $\varepsilon > 0$, if we take sufficient Q-iteration t_i^k, i.e., $Q_i^k = Q_i^{t_i^k,k}$, then we have*

$$\left\|Q_i^k - Q_i^{*,k}\right\|_\infty \le \varepsilon \quad \forall k, i.$$

With Lemma 1, Lemma 2, and Corollary 1, we have the following theorem.

Theorem 1 (multi-agent alternate Q-iteration). *Suppose that $Q_i^*(s,\cdot)$ has the unique maximum for all states and all agents. If all agents in turn take Q-iteration to $\|Q_i^k - Q_i^{*,k}\|_\infty \le \varepsilon$, then their joint policy sequence $\{\boldsymbol{\pi}^k\}$ converges to a Nash equilibrium, where $\pi_i^k(s) = \arg\max_{a_i} Q_i^k(s,a_i)$.*

Proof. First, from Lemma 1, we know $Q_i^{*,k}$ also induces a joint policy improvement, thus $Q_i^{*,k}$ converges to Q_i^*. Let

$$\pi_i^*(s) = \arg\max_{a_i} Q_i^*(s,a_i),$$

then $\boldsymbol{\pi}^*$ is the joint policy of a Nash equilibrium.

Then, we define Δ as

$$\Delta = \min_{s,i} \max_{a_i \ne \pi_i^*(s)} |Q_i^*(s,\pi_i^*(s)) - Q_i^*(s,a_i)|. \tag{14}$$

From the assumption we know that $\Delta > 0$. We take $\varepsilon = \frac{\Delta}{6}$, and from Lemma 2, we know there exists k_0 such that

$$\left\|Q_i^* - Q_i^{*,k}\right\|_\infty \le \varepsilon \quad \forall k \ge k_0. \tag{15}$$

For $k \ge k_0$ and any action $a_i \ne \pi_i^*(s)$, we have

$$\begin{aligned}
&Q_i^k(s,\pi_i^*(s)) - Q_i^k(s,a_i) \\
&= Q_i^k(s,\pi_i^*(s)) - Q_i^{*,k}(s,\pi_i^*(s)) + Q_i^{*,k}(s,\pi_i^*(s)) - Q_i^*(s,\pi_i^*(s)) \\
&\quad + Q_i^*(s,\pi_i^*(s)) - Q_i^*(s,a_i) + Q_i^*(s,a_i) - Q_i^{*,k}(s,a_i) \\
&\quad + Q_i^{*,k}(s,a_i) - Q_i^k(s,a_i) \\
&\ge Q_i^*(s,\pi_i^*(s)) - Q_i^*(s,a_i) - |Q_i^k(s,a_i) - Q_i^{*,k}(s,a_i)| \\
&\quad - |Q_i^{*,k}(s,a_i) - Q_i^*(s,a_i)| - |Q_i^*(s,\pi_i^*(s)) - Q_i^{*,k}(s,\pi_i^*(s))| \\
&\quad - |Q_i^{*,k}(s,\pi_i^*(s)) - Q_i^k(s,\pi_i^*(s))| \\
&= \Delta - 4\varepsilon = \Delta/3 > 0,
\end{aligned} \tag{16}$$

which means

$$\pi_i^k(s) = \arg\max_{a_i} Q_i^k(s,a_i) = \arg\max_{a_i} Q_i^*(s,a_i) = \pi_i^*(s).$$

Thus, Q_i^k of each agent i induces π_i^* and all together induce $\boldsymbol{\pi}^*$, which the joint policy of a Nash equilibrium. □

Theorem 1 assumes that for each agent, Q_i^* has the unique maximum over actions for all states. Although this may not hold in general, in practice we can easily settle this by introducing a positive random noise to the reward function. Suppose the random noise is bounded by δ, then we can easily derive that the performance drop of optimizing environmental reward plus noise is bounded by $\delta/(1-\gamma)$. As we can make δ arbitrarily small, the bound is tight. Moreover, as there might be many Nash equilibria, Theorem 1 does not guarantee the converged joint policy is optimal.

3.2 Multi-Agent Alternate Q-Learning

From Theorem 1, we know that if each agent i guarantees ε-convergence to $Q_i^{*,k}$ in each round k, multi-agent alternate Q-iteration also guarantees a Nash equilibrium of the joint policy. This immediately suggests a simple, practical fully decentralized learning method, namely multi-agent alternate Q-learning (MA2QL).

For learning Q-table or Q-network, MA2QL makes minimal changes to IQL.

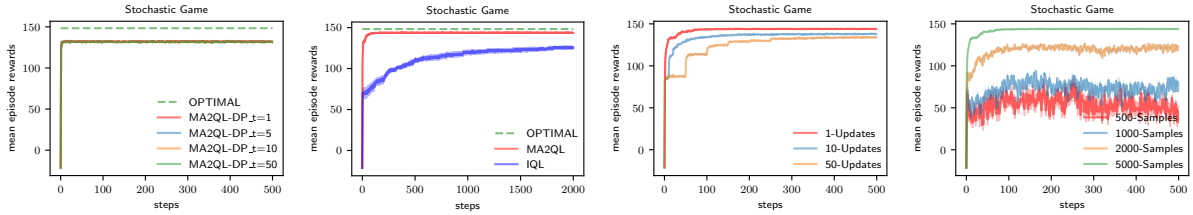
- For learning Q-tables, all agents in turn update their Q-tables. At a round k of an agent i, all agents interact in the environment, and the agent i updates its Q-table a few times using the collected transitions $\langle s, a_i, r, s'\rangle$.
- For learning Q-networks, all agents in turn update their Q-networks. At a round of an agent i, all agents interact in the environment, and each agent j stores the collected transitions $\langle s, a_j, r, s'\rangle$ into its replay buffer, and the agent i updates its Q-network using sampled mini-batches from its replay buffer.

There is a slight difference between learning Q-table and Q-network. Strictly following multi-agent alternate Q-iteration, Q-table is updated by transitions sampled from the current MDP. On the other hand, Q-network is updated by mini-batches sampled from the replay buffer. If the replay buffer only contains the experiences sampled from the current MDP, learning Q-network also strictly follows multi-agent alternate Q-iteration. However, in practice, we slightly deviate from that and allow the replay buffer to contain transitions of past MDPs, following IQL [18, 20, 24] for sample efficiency, the convergence may not be theoretically guaranteed though.

MA2QL and IQL can be simply summarized and highlighted as ***MA2QL agents take turns to update Q-functions by Q-learning, whereas IQL agents simultaneously update Q-functions by Q-learning.***

4 RELATED WORK

CTDE. The most popular learning paradigm in cooperative MARL is centralized training with decentralized execution (CTDE), including value decomposition and multi-agent actor-critic. For value decomposition [20, 22, 24, 29], a joint Q-function is learned in a centralized manner and factorized into local Q-functions to enable



(a) Q-iteration (b) MA2QL (c) Q-table updates (d) samples

Figure 2: Empirical studies of MA2QL on the didactic game: (a) different numbers of Q-iterations performed by dynamic programming at each turn; (b) learning curve of MA2QL compared with IQL and the global optimum; (c) different numbers of Q-table updates at each turn; (d) different numbers of sampled transitions at each turn, where x-axis is learning steps.

decentralized execution. For multi-agent actor-critic, a centralized critic, Q-function or V-function, is learned to provide gradients for local policies [5, 13, 34]. Moreover, some studies [19, 23, 32] combine value decomposition and multi-agent actor-critic to take advantage of both, while others rely on maximum-entropy RL to naturally bridge the joint Q-function and local policies [6, 31, 38].

Decentralized learning. Another learning paradigm in cooperative MARL is decentralized learning, where the simplest way is for each agent to learn independently, *e.g.*, independent Q-learning (IQL) or independent actor-critic (IAC). These methods are usually taken as simple baselines for CTDE methods. For example, IQL is taken as a baseline in value decomposition methods [20, 24], while IAC is taken as a baseline in multi-agent actor-critic [5, 34]. Some studies consider decentralized learning with *communication* [9, 37], but they are not fully decentralized methods. More recently, IAC (*i.e.*, independent PPO) has been empirically investigated and found remarkably effective in several cooperative MARL tasks [3, 34], including multi-agent particle environments (MPE) [13] and StarCraft multi-agent challenge (SMAC). However, as actor-critic methods follow the principle different from Q-learning, we will not focus on IAC for comparison in the experiment. On the other hand, IQL has also been thoroughly benchmarked and its performance is close to CTDE methods in a few tasks [18]. This sheds some light on the potential of value-based decentralized methods. Although there are some Q-learning variants, *i.e.*, hysteretic Q-learning [15] and lenient Q-learning [17], for fully decentralized learning, they are heuristic and their empirical performance is even worse than IQL [8, 35]. MA2QL may be built on top of these Q-learning variants, *e.g.*, the concurrent work [7, 8], which however requires a thorough study and is beyond the scope of this paper.

5 EXPERIMENTS

In this section, we empirically study MA2QL on a set of cooperative multi-agent tasks, including a didactic game, multi-agent particle environments (MPE) [13], multi-agent MuJoCo [19], and StarCraft multi-agent challenge (SMAC) [21], to investigate the following questions.

1. *Does MA2QL converge and what does it converge to empirically, compared with the optimal solution and IQL? How do the number of Q-function updates, environmental stochasticity, and update order affect the convergence?*
2. *As MA2QL only makes the minimal changes to IQL, is MA2QL indeed better than IQL in both discrete and continuous action spaces, and in more complex tasks?*

In all the experiments, the training of MA2QL and IQL is based on the same number of environmental steps (*i.e.*, the same number of samples). Moreover, as the essential difference between MA2QL and IQL is that MA2QL agents take turns to update Q-function while IQL agents update Q-function simultaneously, for a fair comparison, the total number of Q-function updates for each agent in MA2QL is set to be the same with that in IQL. For example, in a setting of n agents, if IQL agents update Q-function m steps (*e.g.*, gradient steps) every environmental step, then each MA2QL agent updates its Q-function $n \times m$ steps each environmental step during its turn. For a training process of T environmental steps, the number of updates for each IQL agent is $T \times m$, while for each MA2QL agent it is also $\frac{T}{n} \times n \times m$. For learning Q-networks, the size of the replay buffer is also the same for IQL and MA2QL. Moreover, we *do not use parameter-sharing*, which should not be allowed in decentralized settings [28] as a centralized entity is required to collect all the parameters. Detailed experimental settings, hyperparameters, and additional results are available in Supplementary (available at https://arxiv.org/abs/2209.08244). All results are presented using the mean and standard deviation of five random seeds.

5.1 A Didactic Game

The didactic game is a cooperative stochastic game, which is randomly generated for the reward function and transition probabilities with 30 states, 3 agents, and 5 actions for each agent. Each episode in the game contains 30 timesteps. For comparison, we use dynamic programming to find the global optimal solution, denoted as OPTIMAL. For MA2QL and IQL, each agent independently learns a 30×5 Q-table.

First, we investigate how the number of Q-iterations empirically affects the convergence of multi-agent alternate Q-iteration, where Q-iteration is performed by dynamic programming (full sweep over the state set) and denoted as MA2QL-DP. As shown in Figure 2(a), we can see that different numbers of Q-iterations (*i.e.*, $t = 1, 5, 10, 50$) that each agent takes at each turn do not affect the convergence in the didactic game, even when $t = 1$. This indicates ε-convergence of Q-iteration can be easily satisfied with as few as one iteration. Next, we compare the performance of MA2QL and IQL. As illustrated in Figure 2(b), IQL converges slowly (about 2000 steps), while MA2QL converges much faster (less than 100 steps) to a better return and

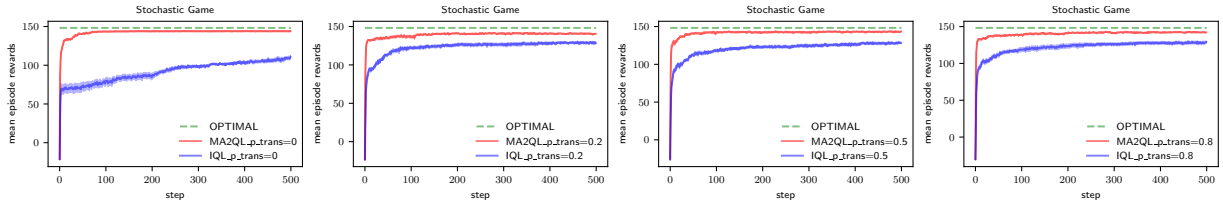


Figure 3: Learning curves of MA2QL compared with IQL in the didactic game under different stochasticity levels, where p_trans is the probability that a state transitions to the next state uniformly at random and hence larger p_trans means a higher level of stochasticity. MA2QL outperforms IQL in all stochasticity levels.

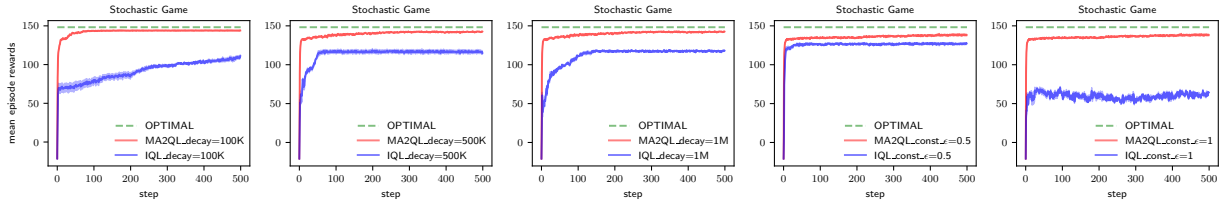


Figure 4: Learning curves of MA2QL compared with IQL in the didactic game under different exploration schemes, where decay_1M means ϵ decays from 1 to 0.02 over 10^6 environment steps, similarly for others. MA2QL outperforms IQL under all exploration schemes.

also approximates OPTIMAL. Once again, MA2QL and IQL use the same number of samples and Q-table updates for each agent. One may notice that the performance of MA2QL is better than MA2QL-DP. This may be attributed to sampling and exploration of MA2QL, which induces a better Nash equilibrium. Then, we investigate MA2QL in terms of the number of Q-table updates at each turn, which resembles the number of Q-iterations by learning on samples. Specifically, denoting K as the number of Q-table updates, to update an agent, we repeat K times of the process of sampling experiences and updating Q-table. This means with a larger K, agents take turns less frequently. As shown in Figure 2(c), with larger K, the learning curve is more stair-like, which means in this game a small number K is enough for convergence at each turn. Thus, with larger K, the learning curve converges more slowly. Last, we investigate how the number of collected transitions at each turn impacts the performance of MA2QL. As depicted in Figure 2(d), the performance of MA2QL is better with more samples. This is because the update of Q-learning using more samples is more like to induce a full iteration of Q-table.

Exploration. We further investigate the performance of MA2QL under different exploration schemes, including different decaying rates and constant ϵ. We use decay_1M to denote the scheme ϵ decays from 1 to 0.02 over 10^6 environment steps and const_ϵ = 0.5 to denote the scheme ϵ maintains 0.5 in training, similarly for others. As illustrated in Figure 4, MA2QL again outperforms IQL in all these exploration schemes. It can be found that IQL's performances are more unstable under different exploration schemes. These results show the robustness of MA2QL under various exploration schemes, even when $\epsilon = 1$. The reason can be attributed to the convergence guarantee of MA2QL.

Stochasticity. We investigate MA2QL under different stochasticity levels of the environment. We control the stochasticity level by introducing p_trans. For any transition, a state has the probability of p_trans to transition to next state uniformly at random, otherwise follows the original transition probability. Thus, larger p_trans means higher level of stochasticity. As illustrated in Figure 3, MA2QL outperforms IQL in all stochasticity levels.

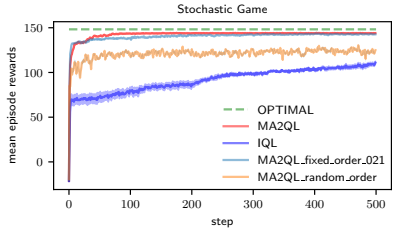


Figure 5: Learning curves of MA2QL under different update orders, including two pre-defined orders (there are three agents in the didactic game, so there are essentially two different pre-defined orders: [0, 1, 2] (MA2QL) and [0, 2, 1]) and random order at each turn. MA2QL outperforms IQL under all update orders.

Update Order. Our theoretical result holds on any update order of agents, which means the order does not affect the convergence theoretically. Here we investigate the empirical performance of MA2QL under different pre-defined orders and the random order at each round. As there are three agents in this environment, there are essentially *two* different pre-defined orders. Suppose that three agents are indexed from 0 to 2. The alternating update orders [0, 1, 2], [1, 2, 0], and [2, 0, 1] are the same, while [0, 2, 1], [1, 0, 2], and [2, 1, 0] are the same. As illustrated in Figure 5, the performance of MA2QL is almost the same under the two orders (MA2QL is the order of [0, 1, 2]), which shows the robustness of MA2QL under different pre-defined orders. As for the random order at each round, the performance drops but is still better than IQL. One possible reason is that agents are not evenly updated due to the random order at each round, which may consequently induce a worse Nash equilibrium.

In the didactic game, we show that MA2QL consistently outperforms IQL under various settings. In the following experiments, for

a fair comparison, the hyperparameters of IQL are well-tuned and we directly build MA2QL on top of IQL. As for the update order of MA2QL agents, we use a randomly determined order throughout the training process.

5.2 MPE

MPE is a popular environment in cooperative MARL. We consider three partially observable tasks: 5-agent simple spread, 5-agent line control, and 7-agent circle control [1], where the action space is set to *discrete*. Moreover, we use the sparse reward setting for these tasks, thus they are more difficult than the original ones. More details are available in Supplementary. For both IQL and MA2QL, Q-network is learned by DQN [16].

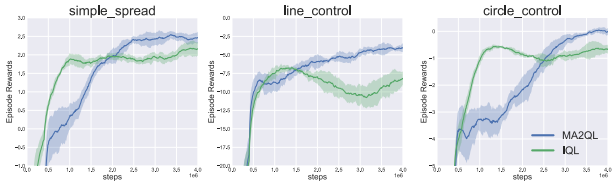


Figure 6: Learning curves of MA2QL compared with IQL in 5-agent simple spread, 5-agent line control, and 7-agent circle control in MPE, where x-axis is environment steps. All tasks are partially observable.

Figure 6 shows the learning curve of MA2QL compared with IQL in these three MPE tasks. In simple spread and circle control, at the early training stage, IQL learns faster and better than MA2QL, but eventually MA2QL converges to a better joint policy than IQL. The converged performance of IQL is always worse than MA2QL, similar to that in the didactic game. Moreover, unlike the didactic game, simultaneous learning of IQL may also make the learning unstable even at the late training stage as in line control and circle control, where the episode rewards may decrease. On the other hand, learning by turns gradually improves the performance and converges to a better joint policy than IQL.

As MA2QL and IQL both use replay buffer that contains old experiences, ***why does MA2QL outperform IQL?*** The reason is their experiences are generated in different manners. In Q-learning, for each agent i, the ideal target is

$$y_i = r_i^{\boldsymbol{\pi}}(s, a_i) + \gamma \mathbb{E}_{s' \sim P_i^{\boldsymbol{\pi}}(\cdot|s,a_i)}[\max_{a_i'} Q_i(s', a_i')]$$

and the practical target is

$$\tilde{y}_i = r_i^{\boldsymbol{\pi}_{\mathrm{D}}}(s, a_i) + \gamma \mathbb{E}_{s' \sim P_i^{\boldsymbol{\pi}_{\mathrm{D}}}(\cdot|s,a_i)}[\max_{a_i'} Q_i(s', a_i')],$$

where $\boldsymbol{\pi}_{\mathrm{D}}$ is the average joint policy for the experiences in the replay buffer. We can easily obtain a bound for the target that

$$|y_i - \tilde{y}_i| \leq \frac{2-\gamma}{1-\gamma} r_{\max} D_{\mathrm{TV}}\left(\pi^{-i}(\cdot|s) \| \pi_{\mathrm{D}}^{-i}(\cdot|s)\right)$$
$$\text{where } r_{\max} = \max_{s,\boldsymbol{a}} r(s, \boldsymbol{a}).$$

We can then give an explanation from the aspect of the divergence between $\boldsymbol{\pi}$ and $\boldsymbol{\pi}_{\mathrm{D}}$. MA2QL obtains experiences with only one agent learning, so the variation for the joint policy is smaller than that of IQL. Thus, in general, the divergence between $\boldsymbol{\pi}$ and $\boldsymbol{\pi}_{\mathrm{D}}$ is smaller for MA2QL, which is beneficial to the learning.

5.3 Multi-Agent MuJoCo

Multi-agent MuJoCo has become a popular environment in cooperative MARL for *continuous* action space. We choose three robotic control tasks: 2×3 HalfCheetah, 3×1 Hopper, and 3×2 Walker2d. To investigate continuous action space in both partially and fully observable environments, we configure 2×3 HalfCheetah and 3×1 Hopper as fully observable, and 3×2 Walker2d as partially observable. More details and results on multi-agent MuJoCo are available in Supplementary. For both IQL and MA2QL, we use DDPG [12] as the alternative to DQN to learn a Q-network and a deterministic policy for each agent to handle continuous action space. Here we abuse the notation a little and still denote them as IQL and MA2QL respectively.

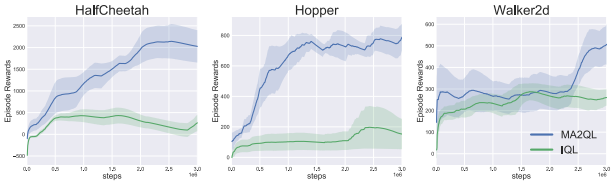


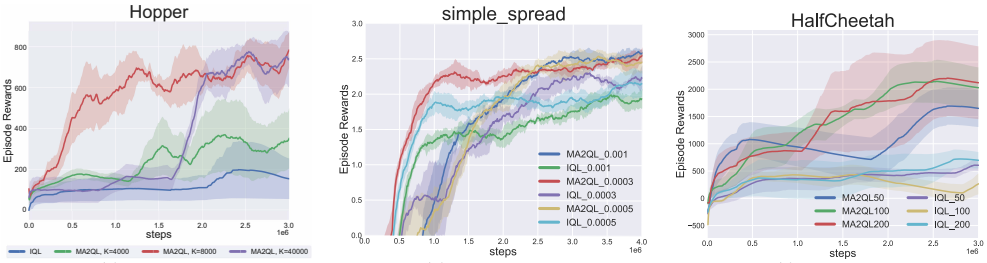
Figure 7: Learning curves of MA2QL compared with IQL in 2×3 HalfCheetah (fully observable), 3×1 Hopper (fully observable), and 3×2 (partially observable) Walker2d in multi-agent MuJoCo, where x-axis is environment steps.

In comparison to discrete action space, training multiple cooperative agents in continuous action space still remains challenging due to the difficulty of exploration and coordination in continuous action space. Thus, the evaluation on these multi-agent MuJoCo tasks can better demonstrate the effectiveness of decentralized cooperative MARL methods. As illustrated in Figure 7, in all the tasks, we find that MA2QL consistently and significantly outperforms IQL while IQL struggles. We believe the reason is that the robotic control tasks are much more dynamic than MPE and the non-stationarity induced by simultaneous learning of IQL may be amplified, which makes it hard for agents to learn effective and cooperative policies. On the other hand, alternate learning of MA2QL can deal with the non-stationarity and sufficiently stabilize the environment during the learning process, especially in HalfCheetah and Hopper, where MA2QL stably converges to much better performance than IQL. According to these experiments, we can verify the superiority of MA2QL over IQL in the continuous action space.

5.4 SMAC

SMAC is a popular partially observable environment for benchmarking cooperative MARL algorithms. SMAC has a much larger exploration space, where agents are much easy to get stuck in sub-optimal policies especially in the decentralized setting. We test our method on three representative maps for three difficulties: `3s_vs_4z` (easy), `5m_vs_6m` (hard), and `corridor` (super hard), where harder map has more agents. Results on more maps are available in Supplementary. It is worth noting that we do not use any global state in the decentralized training and each agent learns on its own trajectory.

The results are shown in Figure 9. On the map `3s_vs_4z`, IQL and MA2QL both converge to the winning rate of 100%. However,



(a) effect of K (b) learning rate (c) batch size

Figure 8: The effect of hyperparameters: (a) learning curves of MA2QL with different K in 3×1 Hopper, compared with IQL; (b) learning curves of MA2QL compared with IQL with different learning rates in simple spread in MPE; (c) learning curves of MA2QL compared with IQL with different batch sizes in 3×2 HalfCheetah in multi-agent MuJoCo. It is shown that the gain of MA2QL over IQL is robust to hyperparameters.

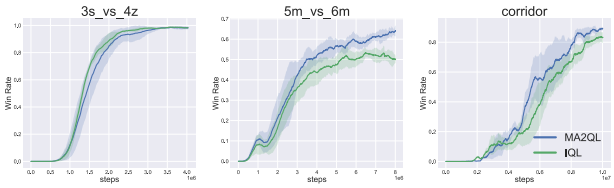


Figure 9: Learning curves of MA2QL compared with IQL on `3s_vs_4z` (easy), `5m_vs_6m` (hard) and `corridor` (super hard) in SMAC, where x-axis is environment steps. All tasks are partially observable.

on the hard and super hard map `5m_vs_6m` and `corridor`, MA2QL achieves stronger than IQL. It is worth noting that the recent study [18] shows that IQL performs well in SMAC, even close to CTDE methods like QMIX [20]. Here, we show that MA2QL can still outperform IQL in three maps with various difficulties, which indicates that MA2QL can also tackle the non-stationarity problem and bring performance gain in more complex tasks.

5.5 Hyperparameters and Scalability

(a) 15-agent simple spread (b) 17×1 Humanoid

Figure 10: Learning curves of MA2QL compared with IQL in (a) 15-agent simple spread in MPE and in (b) 17×1 Humanoid in multi-agent MuJoCo. MA2QL still outperforms IQL in these many-agent tasks, which indicates the good scalability of MA2QL.

We further investigate the influence of the hyperparameters on MA2QL and the scalability of MA2QL. First, we study the effect of K (the number of Q-network updates at each turn) in the robotic control task: 3-agent Hopper. We consider $K = [4000, 8000, 40000]$. As shown in Figure 8(a), when K is small, it outperforms IQL but still gets stuck in sub-optimal policies. On the contrary, if K is large, different K affects the efficiency of the learning, but not the final performance.

As discussed before, the hyperparameters are the same for IQL and MA2QL, which are well-tuned for IQL. To further study their effect on MA2QL, we conduct additional experiments in simple spread with different learning rates and in HalfCheetah with different batch sizes. As shown in Figure 8(b) and Figure 8(c), under these hyperparameters, the performance of IQL and MA2QL varies, but MA2QL consistently outperforms IQL, which can be evidence of the gain of MA2QL over IQL is robust to the hyperparameters of IQL. The default learning rate in MPE is 0.0005 and the default batch size in multi-agent MuJoCo is 100.

As for scalability, we additionally evaluate MA2QL in 15-agent simple spread in MPE and 17×1 Humanoid in multi-agent MuJoCo. As illustrated in Figure 10, MA2QL brings large performance gains over IQL in both tasks. More agents mean the environments become more complex and unstable for decentralized learning. IQL is easy to get stuck by the non-stationarity problem while MA2QL can handle it well. These results indicate the good scalability of MA2QL comparing with IQL.

6 CLOSING REMARKS

In the paper, we propose MA2QL, a simple yet effective value-based fully decentralized cooperative MARL algorithm. MA2QL is theoretically grounded and requires minimal changes to independent Q-learning. We prove the ε-convergence property of MA2QL. Empirically, we verify the effectiveness of MA2QL in a variety of cooperative multi-agent tasks, including a cooperative stochastic game, MPE, multi-agent MuJoco, and SMAC. The results show that, in spite of such minimal changes, MA2QL outperforms IQL consistently across these tasks. In practice, MA2QL can be easily realized by letting agents follow a pre-defined schedule for learning. However, such a schedule is convenient. One limitation of MA2QL is it only guarantees the convergence of Nash equilibrium in tabular cases. As there are usually multiple Nash equilibria, the converged performance of MA2QL may not be optimal as shown in the stochastic game. Nevertheless, learning the optimal joint policy in fully decentralized settings is still an open problem.

ACKNOWLEDGMENTS

This work was supported by NSFC under grant 62250068.

REFERENCES

[1] Akshat Agarwal, Sumit Kumar, Katia Sycara, and Michael Lewis. 2020. Learning Transferable Cooperative Behavior in Multi-Agent Teams. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*.

[2] Dimitri Bertsekas. 2020. Multiagent value iteration algorithms in dynamic programming and reinforcement learning. *Results in Control and Optimization* 1 (2020), 100003.

[3] Christian Schroeder de Witt, Tarun Gupta, Denys Makoviichuk, Viktor Makoviychuk, Philip HS Torr, Mingfei Sun, and Shimon Whiteson. 2020. Is independent learning all you need in the starcraft multi-agent challenge? *arXiv preprint arXiv:2011.09533* (2020).

[4] Mingxiao Feng, Guozi Liu, Li Zhao, Lei Song, Jiang Bian, Tao Qin, Wengang Zhou, Houqiang Li, and Tie-Yan Liu. 2022. Multi-Agent Reinforcement Learning with Shared Resource in Inventory Management.

[5] Jakob Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. 2018. Counterfactual multi-agent policy gradients. In *AAAI Conference on Artificial Intelligence (AAAI)*.

[6] Shariq Iqbal and Fei Sha. 2019. Actor-attention-critic for multi-agent reinforcement learning. In *International Conference on Machine Learning (ICML)*.

[7] Jiechuan Jiang and Zongqing Lu. 2022. Best Possible Q-Learning.

[8] Jiechuan Jiang and Zongqing Lu. 2022. I2Q: A Fully Decentralized Q-Learning Algorithm. In *Advances in Neural Information Processing Systems (NeurIPS)*.

[9] Wenhao Li, Bo Jin, Xiangfeng Wang, Junchi Yan, and Hongyuan Zha. 2020. F2a2: Flexible fully-decentralized approximate actor-critic for cooperative multi-agent reinforcement learning. *arXiv preprint arXiv:2004.11145* (2020).

[10] Xihan Li, Jia Zhang, Jiang Bian, Yunhai Tong, and Tie-Yan Liu. 2019. A Cooperative Multi-Agent Reinforcement Learning Framework for Resource Balancing in Complex Logistics Network. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*.

[11] Yueheng Li, Guangming Xie, and Zongqing Lu. 2022. Difference Advantage Estimation for Multi-Agent Policy Gradients. In *International Conference on Machine Learning (ICML)*.

[12] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. 2016. Continuous control with deep reinforcement learning. In *International Conference on Learning Representations (ICLR)*.

[13] Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. 2017. Multi-agent actor-critic for mixed cooperative-competitive environments. In *Advances in Neural Information Processing Systems (NeurIPS)*.

[14] Omid Madani, Steve Hanks, and Anne Condon. 1999. On the undecidability of probabilistic planning and infinite-horizon partially observable Markov decision problems. In *AAAI/IAAI*.

[15] Laëtitia Matignon, Guillaume J Laurent, and Nadine Le Fort-Piat. 2007. Hysteretic q-learning: an algorithm for decentralized reinforcement learning in cooperative multi-agent teams. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

[16] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. 2013. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602* (2013).

[17] Gregory Palmer, Karl Tuyls, Daan Bloembergen, and Rahul Savani. 2018. Lenient multi-agent deep reinforcement learning. In *International Conference on Autonomous Agents and MultiAgent Systems*.

[18] Georgios Papoudakis, Filippos Christianos, Lukas Schäfer, and Stefano V Albrecht. 2021. Benchmarking multi-agent deep reinforcement learning algorithms in cooperative tasks. In *Advances in Neural Information Processing Systems (NeurIPS)*.

[19] Bei Peng, Tabish Rashid, Christian Schroeder de Witt, Pierre-Alexandre Kamienny, Philip Torr, Wendelin Böhmer, and Shimon Whiteson. 2021. Facmac: Factored multi-agent centralised policy gradients. In *Advances in Neural Information Processing Systems (NeurIPS)*.

[20] Tabish Rashid, Mikayel Samvelyan, Christian Schroeder, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. 2018. Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning. In *International Conference on Machine Learning (ICML)*.

[21] Mikayel Samvelyan, Tabish Rashid, Christian Schroeder de Witt, Gregory Farquhar, Nantas Nardelli, Tim GJ Rudner, Chia-Man Hung, Philip HS Torr, Jakob Foerster, and Shimon Whiteson. 2019. The StarCraft Multi-Agent Challenge. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*.

[22] Kyunghwan Son, Daewoo Kim, Wan Ju Kang, David Earl Hostallero, and Yung Yi. 2019. Qtran: Learning to factorize with transformation for cooperative multi-agent reinforcement learning. In *International Conference on Machine Learning (ICML)*.

[23] Kefan Su and Zongqing Lu. 2022. Divergence-Regularized Multi-Agent Actor-Critic. In *International Conference on Machine Learning (ICML)*.

[24] Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Vinícius Flores Zambaldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, Joel Z Leibo, Karl Tuyls, et al. 2018. Value-Decomposition Networks For Cooperative Multi-Agent Learning Based On Team Reward. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*.

[25] Richard S Sutton and Andrew G Barto. 2018. Reinforcement learning: An introduction.

[26] Ardi Tampuu, Tambet Matiisen, Dorian Kodelja, Ilya Kuzovkin, Kristjan Korjus, Juhan Aru, Jaan Aru, and Raul Vicente. 2015. Multiagent Cooperation and Competition with Deep Reinforcement Learning. *arXiv preprint arXiv:1511.08779* (2015).

[27] Ming Tan. 1993. Multi-agent reinforcement learning: independent versus cooperative agents. In *International Conference on Machine Learning (ICML)*.

[28] Justin K Terry, Nathaniel Grammel, Ananth Hari, Luis Santos, and Benjamin Black. 2020. Revisiting parameter sharing in multi-agent deep reinforcement learning. *arXiv preprint arXiv:2005.13625* (2020).

[29] Jianhao Wang, Zhizhou Ren, Terry Liu, Yang Yu, and Chongjie Zhang. 2021. QPLEX: Duplex dueling multi-agent q-learning. In *International Conference on Learning Representations (ICLR)*.

[30] Jianhong Wang, Wangkun Xu, Yunjie Gu, Wenbin Song, and Tim C Green. 2021. Multi-Agent Reinforcement Learning for Active Voltage Control on Power Distribution Networks. In *Advances in Neural Information Processing Systems (NeurIPS)*.

[31] Jiangxing Wang, Deheng Ye, and Zongqing Lu. 2022. More Centralized Training, Still Decentralized Execution: Multi-Agent Conditional Policy Factorization. *arXiv preprint arXiv:2209.12681* (2022).

[32] Yihan Wang, Beining Han, Tonghan Wang, Heng Dong, and Chongjie Zhang. 2020. Off-Policy Multi-Agent Decomposed Policy Gradients. In *International Conference on Learning Representations (ICLR)*.

[33] Bingyu Xu, Yaowei Wang, Zhaozhi Wang, Huizhu Jia, and Zongqing Lu. 2021. Hierarchically and Cooperatively Learning Traffic Signal Control. In *AAAI Conference on Artificial Intelligence (AAAI)*.

[34] Chao Yu, Akash Velu, Eugene Vinitsky, Yu Wang, Alexandre Bayen, and Yi Wu. 2021. The surprising effectiveness of mappo in cooperative, multi-agent games. *arXiv preprint arXiv:2103.01955* (2021).

[35] Haifeng Zhang, Weizhe Chen, Zeren Huang, Minne Li, Yaodong Yang, Weinan Zhang, and Jun Wang. 2020. Bi-level actor-critic for multi-agent coordination. In *AAAI Conference on Artificial Intelligence (AAAI)*.

[36] Kaiqing Zhang, Zhuoran Yang, and Tamer Basar. 2019. Multi-Agent Reinforcement Learning: A Selective Overview of Theories and Algorithms. *arXiv preprint arXiv:1911.10635* (2019).

[37] Kaiqing Zhang, Zhuoran Yang, Han Liu, Tong Zhang, and Tamer Baar. 2018. Fully Decentralized Multi-Agent Reinforcement Learning with Networked Agents. In *International Conference on Machine Learning (ICML)*.

[38] Tianhao Zhang, Yueheng Li, Chen Wang, Guangming Xie, and Zongqing Lu. 2021. Fop: Factorizing optimal joint policy of maximum-entropy multi-agent reinforcement learning. In *International Conference on Machine Learning (ICML)*.