Shapley Counterfactual Credits for Multi-Agent Reinforcement Learning

Jiahui Li[1], Kun Kuang[1*], Baoxiang Wang[2,3], Furui Liu[4*], Long Chen[1†], Fei Wu[1], Jun Xiao[1]

[1]DCD Lab, College of Computer Science, Zhejiang University
[2]The Chinese University of Hong Kong, Shenzhen
[3]Shenzhen Institute of Artificial Intelligence and Robotics for Society
[4]Huawei Noah's Ark Lab
{jiahuil,kunkuang}@zju.edu.cn,bxiangwang@gmail.com,liufurui2@huawei.com
zjuchenlong@gmail.com,{wufei,junx}@cs.zju.edu.cn

ABSTRACT

Centralized Training with Decentralized Execution (CTDE) has been a popular paradigm in cooperative Multi-Agent Reinforcement Learning (MARL) settings and is widely used in many real applications. One of the major challenges in the training process is credit assignment, which aims to deduce the contributions of each agent according to the global rewards. Existing credit assignment methods focus on either decomposing the joint value function into individual value functions or measuring the impact of local observations and actions on the global value function. These approaches lack a thorough consideration of the complicated interactions among multiple agents, leading to an unsuitable assignment of credit and subsequently mediocre results on MARL. We propose Shapley Counterfactual Credit Assignment, a novel method for explicit credit assignment which accounts for the coalition of agents. Specifically, Shapley Value and its desired properties are leveraged in deep MARL to credit any combinations of agents, which grants us the capability to estimate the individual credit for each agent. Despite this capability, the main technical difficulty lies in the computational complexity of Shapley Value who grows factorially as the number of agents. We instead utilize an approximation method via Monte Carlo sampling, which reduces the sample complexity while maintaining its effectiveness. We evaluate our method on StarCraft II benchmarks across different scenarios. Our method outperforms existing cooperative MARL algorithms significantly and achieves the state-of-the-art, with especially large margins on tasks with more severe difficulties.

CCS CONCEPTS

• **Theory of computation → Online learning algorithms**; **Multi-agent learning**; **Multi-agent reinforcement learning**.

---

[*] Kun Kuang and Furui Liu are the corresponding authors.
[†] This work was done when Long Chen was a Ph.D. student at Zhejiang University.

KEYWORDS

Shapley Value; Counterfactual Thinking; Multi-Agent Systems; Reinforcement Learning; Credit Assignment

1 INTRODUCTION

Multi-Agent Systems (MAS) have attracted substantial attention in many sequential decision problems in recent years, such as autonomous vehicle teams [6, 18], robotics [21, 28], scene graph generation [8], and network routing [44], etc. Among the approaches, Multi-Agent Reinforcement Learning (MARL) has grown its popularity with its ability to learn without knowing the world model. A classical way in MARL to solve cooperative games is regarding the entire MAS as a single agent and optimize a joint policy according to the joint observations and trajectories [36]. With the joint action space of agents growing exponentially as the number of agents and the constraints of partial observability, the classical method faces insurmountable obstacles. This promotes the Centralized Training with Decentralized Execution (CTDE) [19, 26] paradigm, where a central critic is set up to estimate the joint value function, and the agents are trained with global information but executed only based on its local observes and histories.

The main challenge that restricts the effective CTDE in MARL is credit assignment, which attributes the global reward signals according to the contributions of each agent. Recent studies that attempt to solve this challenge can be roughly divided into two branches. 1) *Implicit methods* [29, 33, 35, 42]: it treats the central critic and the local agents as an entirety during the training procedure. A decomposition function (usually a neural network) is first set up to map the joint value function to local value functions. The central critic is then learned simultaneously with the decomposition function and the policy. Implicit methods suffer from inadequate decomposition limited by the design of the decomposition function. They also lack the interpretability for the distributed credits [16]. 2) *Explicit methods* [10, 38, 41]: it trains the central critic and the local actors separately. In each iteration, the critic is first updated, after which some strategies are leveraged to compute the reward or the

value function of each agent explicitly. Such reward signals or value functions are used to guide the training of local agents. Despite that the explicit methods overcome many shortcomings of the implicit counterpart, one has to algorithmically characterize the individual agent's contribution from the overall success, which can be very hard in the context of subtle coalitions under common goals. We address this challenge by using a counterfactual method with Shapley Value. Shapley Value [31] originates from cooperative game theory and is a golden standard to distribute benefits reasonably and fairly by accounting for the contribution of participating players. By treating the agents in MARL as the players in cooperative games, ideal credit assignment can be obtained up to computing the marginal contribution of Shapley Value. Inspired from this, Wang *et al.* [38] proposed SQDDPG, which utilized Shapley Value in deterministic policy gradient [22, 32] to guide the learning of local agents. However, the performance of SQDDPG relies highly on the designed framework for estimating the marginal contribution, and this framework is limited by an assumption that the actions of agents are taken sequentially, which is often unrealistic. These restrictions make SQDDPG perform unsatisfactory in many tasks. To this end, we extend the explicit methods and propose a novel method that leverages Shapley Value to allocate the credits for agents. We achieve it by leveraging a counterfactual method to estimate what would have happened without the participation of a set of agents. The quantification of the contribution of a set of agents is then computed as the change of the central critic value by setting their actions to a baseline. Then the changes of the contributions caused by an agent in different set unions are treated as marginal contributions, and Shapley Value can thus be obtained. Finally, these unified values play the role of credits in local policies and guide its training procedure.

Nevertheless, the computational complexity of the original Shapley Value grows factorially as the number of players increases. In many contexts of interest, such as network games, distributed control, and computing economics, this number can be quite large, which makes Shapley Value intractable. To alleviate the computational burden, we approximate Shapley Value through Monte Carlo sampling, which maintains the majority of the desired properties of Shapley Value. In our approach, the Shapley Value is computed by subsets of collaborators for each agent and is re-sampled at each time step. Our approach manages to reduce the computational complexity to polynomial in the number of players without much loss of effectiveness of Shapley Value.

Our main contributions can be summarized as follows:

(1) We leverage a counterfactual method with Shapley Value to address the problem of credit assignment in Multi-Agent Reinforcement Learning. The proposed Shapley Counterfactual Credits reasonably and fairly characterize the contributions of each local agent by fully considering their interactions.

(2) We adopt a Monte Carlo sampling-based method to approximate Shapley Value and decrease its computational complexity growth from factorial to polynomial, which makes our algorithm viable for large-scale, complicated tasks.

(3) Extensive experiments show that our proposed method outperforms existing cooperative MARL algorithms significantly and achieves state-of-the-art performance on StarCraft II

benchmarks. The margin is especially large for more difficult tasks.

The rest of this paper is organized as follows. In *Section 2*, we first briefly reviews all related works. And we introduce the preliminaries, including Dec-POMDPs, Shapley Value, and explicit framework for MARL in *Section 3*. The details of our proposed algorithm for credit assignment are introduced in *Section 4*. Experimental results and analyses are reported in *Section 5*. Finally, we conclude our paper and discuss on future directions in *Section 6* .

2 RELATED WORK
2.1 Implicit Credit Assignment
Most of the implicit methods follow the condition of Individual-Global-Max (IGM), which means the optimal joint actions among the agents are equivalent to the optimal actions of each local agent. VDN [35] makes a hypothesis of the additivity to decompose the joint Q-function into the sum of individual Q-functions. QMIX [29] gets rid of this assumption but adds a restriction of the monotonicity. LICA [45] promotes QMIX to actor-critic as well as proposes an adaptive entropy regularization. Weighted QMIX adapts a twins network and encourages the underestimated actions to alleviate the risk of suboptimal results. QTRAN [33] avoids the limitations of VDN and QMIX by introducing two regularization terms but has been proved to behave poorly in many situations. Qatten [42] employs a multi-head attention mechanism to compute the weights for the local action value functions and mix them to approximate the global Q-value. All of these methods aim to learn a value decomposition from the total reward signals to the individual value functions, which suffer from several problems: (i) The performance of the model highly relies on the decomposition function. (ii) The lacking of interpretability for the distributed credits. (iii) The high risk of the joint policy tends to fall into sub-optimal results [1, 24, 39].

2.2 Explicit Credit Assignment
Explicit methods attribute the contributions for each agent that are at least provably locally optimal. The most representative method is COMA [10], which utilizes a counterfactual advantage baseline to guide the learning of local policies. However, it treats each agent as an independent unit and overlooks the complex correlations among agents. Thus, it becomes inefficient when encounters complex situations. SQDDPG [38] proposes a network to estimate the marginal contribution, which is further used to approximate Shapley Value. Then, Shapley Value is used to guide the learning of local agents. However, such estimation for marginal contribution doesn't make sense in many situations because the network over-relies on the assumption that the agents take actions sequentially. QPD [41] designs a multi-channel mixer critic and leverage integrated gradients to distribute credits along paths, which achieves state-of-the-art results in many tasks. Intuitively, mining the relations between the agents is essential for the policy gradient in cooperative games. But the correlations are too complicated and are often underestimated by the models. To this end, we propose a Shapley Counterfactual Critic for credit assignment in MARL. Thanks to Shapley Value, the relations between the agents are considered sufficiently without prior knowledge, which further promotes the learning of local agents. Different from SQDDPG [38], we compute the marginal

contributions according to a counterfactual method rather than building a network, which is more stable and efficient in complicated situations.

2.3 Shapley Value and Approximate SV

Shapley Value [4, 23, 31, 34] originates from cooperative game theory in the 1950s, which assigns a unique distribution of total benefits generated by the coalition of all players. Shapley Value satisfies the properties of *efficiency, symmetry, nullity, linearity* and *coherency*. It is a unique and fairly way to quantify the importance of each player in the overall cooperation and widely used in economics. However, the computational complexity of Shapley Value grows factorially with respect to the number of participating players [20]. Thus, in order to decrease the computation, several recent studies start to approximate the exact Shaply Value [7, 9, 12, 37] by sacrificing some properties. For example, Frye *et al.* [11] and Tom *et al.* [15] utilize casual knowledge to simplify its calculation, which breaks the axiom of *symmetry*. L-Shapley and C-Shapley only consider the interactions among the local and connected player, which slightly break the properties of *efficiency*. DASP [2] and Neuron Shapley [13] adapt sample methods to approximate Shapley Value, which also slightly breaks the properties of *efficiency* and *symmetry*.

3 PRELIMINARIES

3.1 Dec-POMDPs

A fully cooperative multi-agent sequential decision-making task with n agents $A = \{1, 2, ..., n\}$ can be modeled as a decentralised partially observable Markov decision process (Dec-POMDP) [3, 5, 14, 25, 27]. Dec-POMDP is canonically formulated by the tuple:

$$G = (S, U, P, r, Z, O, n, \gamma).$$

In the process, $s \in S$ represents the true state of the environment. At each time step, each agent $a \in A$ chooses an action $u_a \in U$ simultaneously to formulate a joint action $u \in U^n$. The action produces a state transition on the environment which is described by the Markov transition probability function $P(s'|s, u): S \times U^n \rightarrow S$. All of the agents share a same global reward function $r(s, u): S \times U^n \rightarrow \mathbb{R}$.

In the setting of partial observability, the observations of each agent $z \in Z$ are generated by a observation function $O(s, a): S \times A \rightarrow Z$. Each agent owns an action-observation history $\tau_a \in T$, where $T = (Z \times U)^*$ denotes the set of sequences of state-action pairs with arbitrary length. On this history, each agent conditions a stochastic policy $\pi^a(u_a|\tau_a): T \times U \rightarrow [0, 1]$. The common goal of all agents is to maximize the expected discounted return $R_t = \sum_{i=0}^{\infty} \gamma^i r_{t+i}$.

3.2 Shapley Value

Assume a coalition consists of N players and they cooperate with each other to achieve a common goal. For a particular player i, let S be a random set that contains player i and $S\backslash\{i\}$ represents the set with the absence of i, then the marginal contribution of i in S is defined as:

$$\Delta v(i, S) = v(S) - v(S\backslash\{i\}), \tag{1}$$

where $v(\cdot)$ refers to the value function for estimating the cooperated contribution of a set of players.

Then the Shapley Value of player i is computed as the weighted average of the marginal contributions in all of the subsets of N:

$$\phi_v(i) = \frac{1}{N} \sum_{k=1}^{N} \frac{1}{\binom{N-1}{k-1}} \sum_{S \in S_k(i)} \Delta v(i, S), \tag{2}$$

where $S_k(i)$ denotes a set with size k that contain the player i. Shapley Value satisfy the following properties:

- **Efficiency.** The credits generated by the big coalition $v(\{1, ..., N\}) - v(\phi)$ is equal to the sum of the Shapley Values of all of the participating players $\sum_{i=1}^{N} \phi_v(i)$.
- **Symmetry.** If $\Delta v(i, S) = \Delta v(j, S)$ for all subsets S then $\phi_v(i) = \phi_v(j)$.
- **Nullity.** If $\Delta v(i, S) = 0$ for all subsets S then $\phi_v(i) = 0$.
- **Linearity.** Let u and w represent the associated gain functions, then $\phi_{v+w}(i) = \phi_v(i) + \phi_w(i)$ for every $i \in N$.
- **Coherency.** When another value function $\Delta v'(i)$ is utilized to measure the marginal contribution of i, if $\Delta v(i, S) \geq \Delta v'(i, S)$ for all subsets S, then $\phi_v(i) \geq \phi'_v(i)$.

3.3 Explicit Framework for MARL

Explicit methods are interpretable for the allocated credits, which can reduce the suspicion of users to the rationality of the learned local agents. For this reason, we extend the explicit methods [10, 38, 41], which first train the central critic according to the joint states and actions and then distribute the global reward signals according to the contributions of local agents to the critic.

Following QPD [41], we model our critic network with three components as shown in Figure 1, that is, the feature extraction module, the feature fusion module, and the Q-function estimation module. The first module consists of 2 dense layers with ReLU non-linearity, which is used to extract the features of a particular agent's observations and actions. Then the features of all agents are concatenated thus merged into a global feature. Finally, the joint Q-value is computed according to the global feature. As Yang *et al.* [41] illustrated, different agents may own the same attributions, so can be categorized into different groups. For this reason, the agents within the same group are modeled using the same subnetwork. Meanwhile, in order to simplify the network architecture and accelerate the learning procedure, the agents of the same group share the same parameters. We represent the central critic as:

$$Q_{tot} = f(o_1, u_1, ..., o_n, u_n), \tag{3}$$

where o_i and u_i denote the observation and the action of the i-th agent, respectively.

In our implementation, each local agent is realized with a Recurrent Deep Q-Network, which is composed of an Long Short-Term Memory (LSTM) layer and a Multi-Layer Perceptron (MLP). We represent the local agent as:

$$Q_i = g(o_i; h), \tag{4}$$

where h is the hidden state of LSTM. For the exploration policy, ϵ-greedy is adopted and the exploration rate of episode *eps* is:

$$\epsilon(eps) = \max(\epsilon_{start} - eps \cdot \sigma, 0), \tag{5}$$

where ϵ_{start} is the initial exploration rate and σ represents the decreasing count of ϵ each episode.
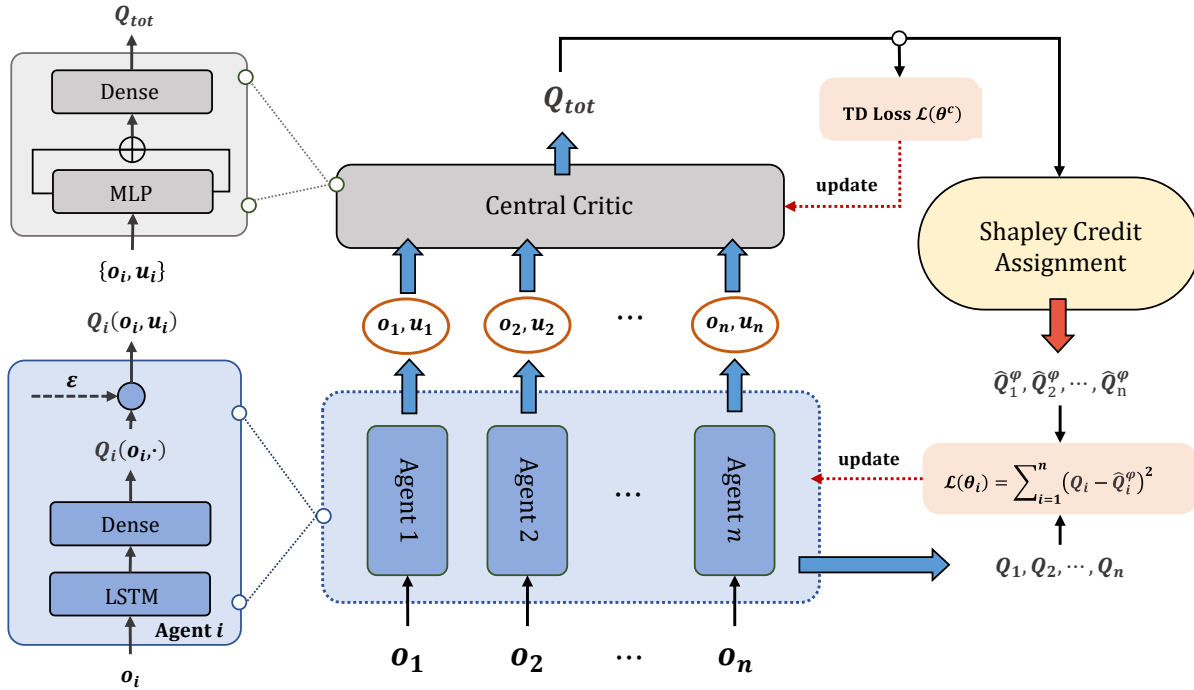
Figure 1: The framework of our method. We adopt a two-stage way that trains the central critic and the local policies separately. First, the central critic is updated with TD-loss. Then the credits of each agent are calculated by our proposed counterfactual method with approximate Shapley Value. Finally, the local policies are updated using the Shapley Counterfactual Credits.

4 SHAPLEY COUNTERFACTUAL CREDITS FOR MARL

The framework of our approach is illustrated in Figure 1. First, the central critic takes the actions and observations of each agent as input and approximates the total Q value. Then the contributions of the individual agents are distributed by the counterfactual method with Shapley Value. Finally, the local agents update their parameters according to the credits they earned.

In this section, we systematically describe our "Shapley Counterfactual Credits" for Multi-Agent Reinforcement Learning. First, we will introduce a counterfactual method with Shapley Value to address the problem of "credit assignment", which can fully mine the correlations among the local agents in *Section 4.1*. To downgrade the computational complexity. we replace the truly Shapley Value with its approximation, and this will be discussed in *Section 4.2*. The details of the proposed algorithm and the loss function will be introduced in *Section 4.3*.

4.1 Counterfactual Method with Shapley Value for Credit Assignment

The main challenge we need to address is how to measure the contributions of the agent. In other words, we need to quantify how the agents' actions influence the output of the central critic. COMA [10] proposed a special critic and utilize a counterfactual baseline, which estimated the advantages of action value over expected value as this influence but shows poor performance on many tasks. Wolpert *et al.* [40] computed the influence by using *difference rewards* which

compares the global reward to the reward received when the action of an agent is replaced with a default action. Inspired by these ideas, we also proposed a counterfactual method in our central critic to measure the effect of the actions taken by the agents.

We consider the contribution of an action taken by an agent is equal to "how the output will change when this action is absent?" We formulate the contributions of the action performed by the i-th agent to the central critic as:

$$v_i = f(o_1, u_1, ..., o_i, u_i, ..., o_n, u_n) - f(o_1, u_1, ..., o_i, \tilde{u}, ..., o_n, u_n), \quad (6)$$

where $\tilde{u}$ denotes a baseline that means the action is replaced by a default one.

However, such estimation for the contributions is insufficient since the agents are cooperating with each other and cannot be treated as independent units. We then desire to quantify the credits made by an agent precisely from the intricate relationship among agents, but the environment is complex, and there is no prior knowledge to indicate how they cooperated with each other. To this end, we propose to utilize Shapley Value for credit assignment, and this will be introduced in the next subsection.

As we mentioned before, Shapley Value distributes the credits fairly by considering the contributions of the participating players and satisfies many good properties such as *efficiency*, *additivity*, and *coherency*. Thus, we utilize this tool to extend the counterfactual method.

For convenience, we shorthand Equation (6) and change agent i to a set S of agents:

$$v_{S \ in \ A} = f(H_A) - f(H_{A \setminus S}), \tag{7}$$

where A denotes all of the agents, H_A represents the actions and observations of A, and $H_{A \setminus S}$ denotes that the actions of all agents in S are replaced with default actions.

To compute the Shapley Value of the i-th agent in the big coalition, we need to compute its marginal contributions when this agent play roles in all of the subset of the big coalition A. We define the marginal contribution of the i-th agent in the subset S of A as:

$$\Delta v(i, S) = v_{S \ in \ A} - v_{S \setminus i \ in \ A}, \tag{8}$$

where $S \setminus i$ denotes S with the removal of the i-th agent.

After getting the marginal contribution, we compute the Shapley Counterfactual Credits Q_i^φ as:

$$Q_i^\varphi = \frac{1}{N} \sum_{j=1}^{N} \frac{1}{\binom{N-1}{j-1}} \sum_{S \in S_j(i)} \Delta v(i, S), \tag{9}$$

where $S_j(i)$ denotes the set of agents with size j that contains the i-th agent.

4.2 Approximation of Shapley Value

However, the main drawback of Shapley Value is that the computational complexity grows factorially as the number of the agents increases [20]. So recent studies usually use an approximation of Shapley Value as a substitution [7, 12, 37]. Since the number of the agents may bring an unacceptable computational cost, for alleviating the computational burden, the approximation of Shapley Value is necessary. Thus, we adopt the Monte Carlo sampling method to get the approximated Shapley Value:

$$\hat{Q}_i^\varphi = \frac{1}{M} \sum_{j=1}^{M} \Delta v(i, S_{MC_j}(i)), \tag{10}$$

where M represents the times of Monte Carlo sampling, $S_{MC_j}(i)$ represents a subset of A sampled in j-th time that contains the i-th agent.

According to this approximation, we downgrade the computational complexity of the truly Shapley Value of an agent from $O(N!)$ to $O(M)$, where N is the number of agents that may be very large in some situations, and M is a hyperparameter which represents the times of Monte Carlo sampling and can be a small positive integer. To be noticed that, such an approximation of Shapley Value might slightly break some of its properties such as *efficiency* and *Symmetry*. Recent literature sacrificed its properties in varying degrees but got an acceptable computational costs [7, 12, 37]. We deem that such an approximation is necessary and will not bring too much impact to the model's performance.

4.3 Loss Function and Training Algorithm

We show the details of our algorithm in Algorithm 1. Our whole framework is updated in two stages. First, the local agents interact with the environment and take actions according to their observations and history. Then, these actions and observations act as the input of the central critic to estimate the joint Q-function. Afterward, in the first stage, we update the central critic by minimizing

the TD-loss $\mathcal{L}(\theta^c)$:

$$\begin{aligned} \mathcal{L}(\theta^c) &= (Q_{tot} - y)^2, \\ y &= r + \gamma(\tilde{Q}_{tot}), \end{aligned} \tag{11}$$

where θ^c is the parameters of the central critic, Q_{tot} is the output of the central critic, and $\tilde{Q}_{tot}$ represents the output of target network of the central critic.

In the second stage, we first get the Shapley Counterfactual Credits Q_i^φ of each agent according to Equation (10). Then each agent is trained by minimizing the loss:

$$\mathcal{L}(\theta^i) = (Q_i - \hat{Q}_i^\varphi)^2, \tag{12}$$

where θ^i denotes the parameters of the i-th local agent, and Q_i is the output of the i-th agent.

Algorithm 1 Shapley Counterfactual Credits Algorithm for MARL

Initialize: Central critic network θ^c, target central critic network $\tilde{\theta}^c$, local agents' networks $\theta^\pi = (\theta^1, ..., \theta^n)$

1: **for** each training episode *eps* **do**
2:     s_0 = initial state, $t = 0$, $h_0^i = 0$ for each agent i
3:     **while** $s \neq$ terminal **and** $t < T$ **do**
4:         $t = t + 1$
5:         **for** each agent i **do**
6:             $Q_i(o_t^i,), h_t^i = Agent_i(o_t^i; h_{t-1}^i)$
7:             Sample u_t^i from $\pi(Q_i(o_t^i,), \epsilon(eps))$
8:         Execute the joint action $(u_t^1, u_t^2, ..., u_t^n)$
9:         Get reward r_{t+1} and next state s_{t+1}
10:    Add episode to replay buffer
11:    Collate episodes in buffer into a single batch
12:    **for** b in batch **do**
13:        **for** $t = 1$ to T **do**
14:            Compute the targets y^t using central target network
15:    Update central critic network θ^c with (11)
16:    Every C episodes reset $\tilde{\theta}^c = \theta^c$
17:    **for** b in batch **do**
18:        **for** $t = 1$ to T **do**
19:            Compute credits for each agent via (10)
20:    Update the local agents θ^π with (12)

---

5 EXPERIMENTS

We focus on addressing the problem of credit assignment in MARL with cooperative settings explicitly. We compare our proposed method with several baselines, including VDN [35], QMIX [29], COMA [10], QTRAN [33], QPD [41], and SQDDPG [38]. The training configurations, experiment results, as well as the analysis will be described in detail in this section.

5.1 Experiment Settings

Environment. We perform extensive experiments on the Star-Craft II (a real-time strategy game) micromanagement challenge, in which each army is controlled by an agent and act based on its local observations and the opponent's army are controlled by the hand-coded built-in StarCraft II AI. Each unit in StarCraft contains
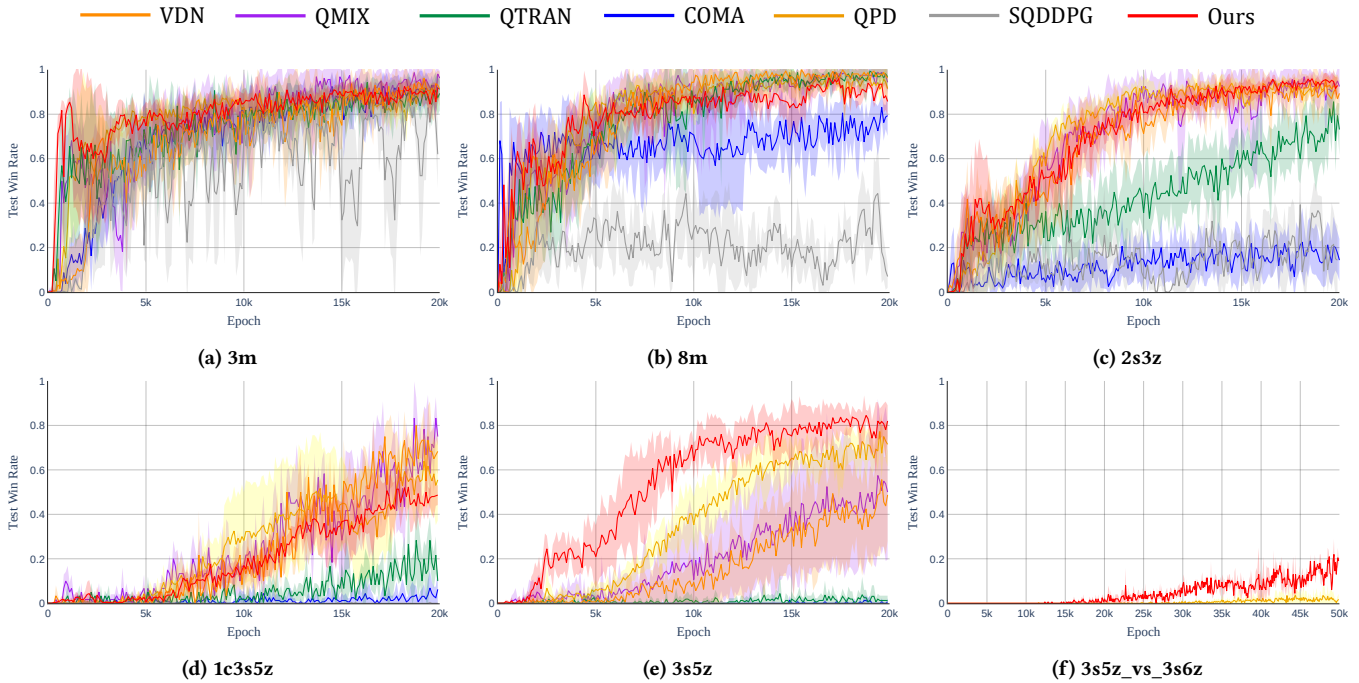
Figure 2: The mean win rates of our method compared with others in different map scenarios of StarCraft II. The shaded areas represent the standard deviation.

Table 1: Median and mean win rate of our method compared with other methods. $\tilde{m}$ represents the median of the test win rates and avg represents mean test win rates.

Map	Methods													
	VDN		QMIX		QTRAN		COMA		QPD		SQDDPG		OURS	
	$\tilde{m}$	avg	$\tilde{m}$	avg	$\tilde{m}$	avg	$\tilde{m}$	avg	$\tilde{m}$	avg	$\tilde{m}$	avg	$\tilde{m}$	avg
3m	**100**	**100**	**100**	**100**	**100**	**100**	95	96	99	99	64	65	99	99
8m	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	95	95	92	90	98	97
2s3z	**100**	**100**	**100**	**100**	92	91	45	45	99	98	60	55	**100**	**100**
1c3s5z	88	85	**95**	**90**	40	41	15	15	77	72	2	2	61	60
3s5z	80	69	80	67	12	13	5	3	79	80	1	1	**92**	**90**
3s5z_vs_3s6z	0	0	0	0	0	0	0	0	3	5	0	0	**20**	**20**

a rich set of complex micro-actions, which allow the learning of complex interactions between the agents that cooperate with each other. The overall goal is to maximize the accumulated rewards for each battle scenario. The environment produces rewards based on the hit-point damage dealt and enemy units killed. Besides, another bonus is given when the battle wins. At each time step, each agent can only receive the local observations within its field of view. Meanwhile, an agent can only observe the other agents alive and located in its sight range. Besides, all agents can only attack the enemies within their shooting range, which is set to 6. The global state consists of the joint observations without the restriction of the sight range, which will be used in the central critic during the training procedure. All features are normalized by their maximum

values before sent to the neural network. StarCraft Multi-Agent Challenge (SMAC) environment [30] is used as testbed, and we set the difficulty of the game AI as "very difficult" level.

Configurations. The central critic of our method is the same as QPD [41], which consists of the feature extraction layers, the feature fusion operation, and the Q-function estimation layers. First, the agents are grouped according to their attributions, and 2 dense layers are used to extract the features of their observations and actions. Each dense layer consists of 64 neurons for each channel. For accelerating the learning procedure, we adopt parameter sharing technique [17, 43] where the agents within the same group share the parameters of the feature extraction layers. Then, we concatenated the features of all agents to fuse them into a global feature.
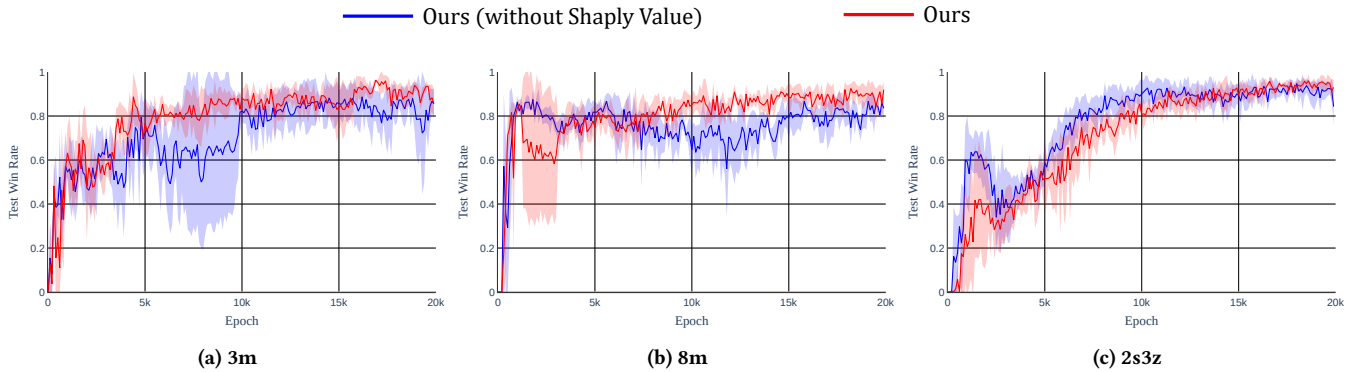
Figure 3: Ablation study of Counterfactual Shapley Credits.

Finally, for the final Q-function estimation, we adapt another dense layer with one output neuron. In the procedure of computing Shapley Value, we adapt the Monte Carlo sampling method to sample 5 subsets for each agent at each time step. We set the counterfactual baseline $\tilde{u}$ in the central critic as zero vector for convenience. We model the local agents with an LSTM layer and 2 fully connected layers. The dimensional of hidden state in LSTM is set as 64, the units of the two fully connected layers are set as 64 and $|U|$ separately, where $|U|$ is the size of action space. We set the discount rate γ for TD-loss as 0.99. The replay buffer stores the most recent 1000 trajectories. During training, we update the central critic with RMSprop and local agent networks with Adam. We copy the parameters of the central critic to its target network every 200 training episodes. The full hyperparameters of our Shapley Counterfactual Credits are shown in Table 2. The map *3s5z_vs_3s6z* is much harder than the other maps, and the allied forces have one unit less than the enemy. During training, the win rates remain 0 even when the returns are relatively high. For this reason, we set the number of the training episodes of map *3s5z_vs_3s6z* to 50000, while the others are set to 20000.

5.2 Results and Analysis

To demonstrate the efficiency of our proposed method, we perform experiments on 6 maps of StarCraft II (*3m, 8m, 2s3z, 1c3s5z, 3s5z, 3s5z_vs_3s6z*), including both homogeneous and heterogeneous scenarios. Figure 2 depicts the curve of mean win rates of our method compared to the baselines. The final results of our method are depicted in Table 1, where $\tilde{m}$ represents the median of the test win rates and *avg* represents mean test win rates.

All of the methods show high performance on three simple scenarios (*3m, 8m, 2s3z*), and our Shapley Counterfactual Credits algorithm is competitive with the state-of-the-art algorithm, and achieves nearly 100% mean win rates. Both sides have 3 *Marines* in map *3m*, and 8 *Marines* in map *8m*. As the arms of both sides are single and the numbers are equal, each agent only needs to focus on beating enemies and avoid taking redundant actions. Concretely, from the replay, in map *3m* and *8m*, units learned to stand in a line or semicircle in order to set fire to the incoming enemies. Such a pattern is easy for models to learn, and agents hardly need to

Table 2: Hyperparameters of Shapley Counterfactual Credit Algorithm

Settings	Value
Batch size	32
Replay buffer size	1000
Training episodes	1000
Exploration episodes	1000
Start exploration rate	1
End exploration rate	0
TD-loss discount	1000
Target central critic update interval	200 episodes
Evaluation interval	100 episodes
Evaluation battel number	100
Agent optimizer	RMSProp
Central Critic optimizer	Adam
Agent learning rate	0.005
Central critic learning rate	0.01
Dense units	64
LSTM hidden units	64
Baseline for Shapley Value	0 vector
Times for Monte Carlo Sampling	5

consider how to cooperate with its friendly forces. In map *2s3z*, both sizes have 2 *Stalkers* and 3 *Zealots*. Since that *Zealots* counter *Stalkers*, the *Stalkers* need to hide behind the own side *Zealots*. Such a small number of units does not bring too much challenge for the learning of the model.

Our algorithm falls behind the other methods in map *1c3s5z*, where both sizes have 3 *Stalkers*, 5 *Zealots* and an *Colossus*. Since the *Colossus* is more threatening, and becomes the priority target, which reduces the difficulty of the game. Here, we divide the learned ability of an agent into the personal ability and the cooperative ability. For example, "kite the enemy" as well as "attack high-threat targets" belongs to the former, and "move to protect the allies" belongs to the latter. In this map, all of the agents need to learn the pattern to attack the enemy's *Colossus* first, which makes other

actions less important. Since Shapley Value focuses more on mining the correlation between agents, our method does not perform very well in this scenario.

Our algorithm shows obvious advantages in two maps *3s5z* and *3s5z_vs_3s6z* which are much more difficult than others. In map *3s5z*, both sizes have 3 *Stalkers* and 5 *Zealots*, and we got the mean win rates of 90%. In this scenario, not only the agents of *Stalkers* need to stand behind the allied *Zealots*, but learn to attack the enemy *Stalkers* with high priority. Meanwhile, the allied *Zealots* need to protect allied *Stalkers* as well as attack the nearest enemy *Stalkers*. In this complex situation, cooperation among agents is more important than before. Our counterfactual method with Shapley Value fully considers the correlation and interactions between units and distributes a moderate credit for the actions taken by each agent, thus outperforms the baselines significantly. For instance, a "movement" of a *Zealots* may affect other friendly forces in varying degrees; we measure its contribution by considering how the results will change when different kinds of correlations are absent. Especially, in map *3s5z_vs_3s6z*, where ally has 3 *Stalkers* and 5 *Zealots* while the enemy has 6 *Zealots*, all of the current method except QPD got the mean win rates of zero. The reason for the poor performance of these methods is that cooperative behavior such as "block" rather than "kite" play more important roles in such settings. The *Zealots* need to attract firepower in order to protect the allied *Stalkers*, which is the only way to get the final victory. In this scenario, Shapley value fully demonstrates its superiority. Our method achieves the mean win rates of 20%, and reach the state-of-the-art.

In conclusion, our proposed Shapley Counterfactual Credits algorithm shows its strength and beats all of the other methods in complicated scenarios where cooperation among agents plays an essential role. Our proposed algorithm also exhibits the competitive results with the state-of-the-art algorithm in the scenarios that need to pay more attention to personal ability.

5.3 Ablation Study

To demonstrate the advantage of Shapley Value [31] to the counterfactual method, we perform ablation study on three maps (*3m*, *8m*, *2s3z*). The difficulty of these three maps increases sequentially. The results are shown in Figure 3. The blue curves represent that the credits are allocated by the counterfactual method without Shapley Value. The red curves represent that the credits are distributed by Shapley Counterfactual Credits. For the balance between the performance and computational costs, we set the times of the Monte Carlo sampling for approximating Shapley Value as 5, and the analysis is shown in the next subsection.

In map *3m* and *8m*, the units need to learn the strategies that stand in a suitable position to fire the same enemy unit together. Thus, the ability to cooperate is relatively important, and the use of Shapley Value brings an improvement of the performance. While in *2s3z*, the *Stalkers* need to "kite the *Zealots*" and the number of the units is small, which means personal ability is more important. So our method loses advantage in this scenario. It is worth mentioning that the use of Shapley Value makes learning more stable and reduces the standard deviation (the shaded part in the figure) of the win rates significantly. That because Shapley Value considers
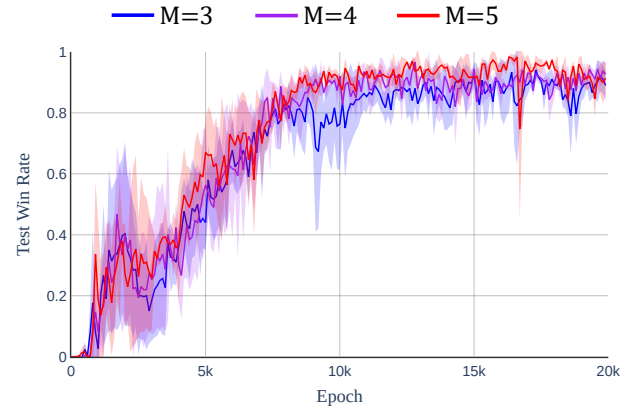

Figure 4: The mean win rates of the approximated Shapley Counterfactual Credits with different sample size in map *2s3z*.

a variety of combinations among agents and measure the contribution of an agent via the weighted average of the counterfactual results of these combinations.

5.4 The Choice of Sample Times for Shapley Approximation

We approximated the truly Shapley Value via Monte Carlo sampling. Concretely, at each time step, we sample M subsets randomly for each agent a, and average the marginal contributions of a in these subsets to represent its approximated Shapley Value. However, a large M will still bring pressure to the computation costs, and small M will lead to an inaccurate approximation. We performed extensive experiments to find a moderate hyperparameter, and the results are depicted in Figure 4. We conclude that 4 times sampling are sufficient to reach an ideal result. But to make the performance more stable, we set M to 5 for in our experiments.

6 CONCLUSION AND FUTURE WORK

In this paper, we investigate the problem of credit assignment in Multi-Agent Reinforcement Learning. We extend the methods of explicit credit assignment and leverage a counterfactual method to measure the contributions of local agents to the central critic. To fully describe the relationships among the cooperative agents, Shapley Value is utilized with a sample-based method, with a Monte-Carlo sampling variant to decrease its computational complexity from factorial to polynomial. Experiments on the StarCraft II micromanagement tasks show the superiority of our method as we reach the state-of-the-art on various scenarios.

For future work, it could be interesting to investigate the causal knowledge among the cooperative agents. With this inferred knowledge, Shapley Value can be approximated in a more accurate way and the credit assignment can be more precise. Our method can also be extended to the scenarios with competitive settings, where variants of Shapley Value are proved to be effective.

ACKNOWLEDGMENTS

REFERENCES

[1] Zafarali Ahmed, Nicolas Le Roux, Mohammad Norouzi, and Dale Schuurmans. 2019. Understanding the impact of entropy on policy optimization. In *International Conference on Machine Learning*. 151–160.

[2] Marco Ancona, Cengiz Oztireli, and Markus Gross. 2019. Explaining deep neural networks with a polynomial time algorithm for shapley value approximation. In *International Conference on Machine Learning*. 272–281.

[3] Daniel S Bernstein, Robert Givan, Neil Immerman, and Shlomo Zilberstein. 2002. The complexity of decentralized control of Markov decision processes. *Mathematics of operations research* 27, 4 (2002), 819–840.

[4] Jesus Mario Bilbao and Paul H. Edelman. 2000. The Shapley value on convex geometries. *Discrete Applied Mathematics* 103, 1-3 (2000), 33–40.

[5] Lucian Busoniu, Robert Babuska, and Bart De Schutter. 2008. A comprehensive survey of multiagent reinforcement learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 38, 2 (2008), 156–172.

[6] Yongcan Cao, Wenwu Yu, Wei Ren, and Guanrong Chen. 2012. An overview of recent progress in the study of distributed multi-agent coordination. *IEEE Transactions on Industrial informatics* 9, 1 (2012), 427–438.

[7] Jianbo Chen, Le Song, Martin J Wainwright, and Michael I Jordan. 2018. L-shapley and C-shapley: Efficient model interpretation for structured data. In *International Conference on Learning Representations*.

[8] Long Chen, Hanwang Zhang, Jun Xiao, Xiangnan He, Shiliang Pu, and Shih-Fu Chang. 2019. Counterfactual critic multi-agent training for scene graph generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 4613–4623.

[9] Shaheen S Fatima, Michael Wooldridge, and Nicholas R Jennings. 2008. A linear approximation method for the Shapley value. *Artificial Intelligence* 172, 14 (2008), 1673–1699.

[10] Jakob Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. 2018. Counterfactual multi-agent policy gradients. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 32.

[11] Christopher Frye, Ilya Feige, and Colin Rowat. 2020. Asymmetric Shapley values: incorporating causal knowledge into model-agnostic explainability. *Conference and Workshop on Neural Information Processing Systems*.

[12] Amirata Ghorbani and James Zou. 2019. Data shapley: Equitable valuation of data for machine learning. In *International Conference on Machine Learning*. 2242–2251.

[13] Amirata Ghorbani and James Zou. 2020. Neuron shapley: Discovering the responsible neurons. *arXiv preprint arXiv:2002.09815* (2020).

[14] Jayesh K Gupta, Maxim Egorov, and Mykel Kochenderfer. 2017. Cooperative multi-agent control using deep reinforcement learning. In *International Conference on Autonomous Agents and Multiagent Systems*. 66–83.

[15] Tom Heskes, Evi Sijben, Ioan Gabriel Bucur, and Tom Claassen. 2020. Causal Shapley Values: Exploiting Causal Knowledge to Explain Individual Predictions of Complex Models. *Conference and Workshop on Neural Information Processing Systems*.

[16] Alexandre Heuillet, Fabien Couthouis, and Natalia Díaz-Rodríguez. 2021. Explainability in deep reinforcement learning. *Knowledge-Based Systems* 214 (2021), 106685.

[17] Shariq Iqbal and Fei Sha. 2019. Actor-attention-critic for multi-agent reinforcement learning. In *International Conference on Machine Learning*. 2961–2970.

[18] Tamás Keviczky, Francesco Borrelli, Kingsley Fregene, Datta Godbole, and Gary J Balas. 2007. Decentralized receding horizon control and coordination of autonomous vehicle formations. *IEEE Transactions on control systems technology* 16, 1 (2007), 19–33.

[19] Landon Kraemer and Bikramjit Banerjee. 2016. Multi-agent reinforcement learning as a rehearsal for decentralized planning. *Neurocomputing* 190 (2016), 82–94.

[20] I Elizabeth Kumar, Suresh Venkatasubramanian, Carlos Scheidegger, and Sorelle Friedler. 2020. Problems with Shapley-value-based explanations as feature importance measures. In *International Conference on Machine Learning*. 5491–5500.

[21] 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. *International Conference on Learning Representations* (2016).

[22] Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. 2017. Multi-agent actor-critic for mixed cooperative-competitive environments. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*. 6382–6393.

[23] Fan-Yong Meng. 2012. The Core and Shapley Function for Games on Augmenting Systems with a Coalition Structure. *International Journal of Mathematical and Computational Sciences* 6, 8 (2012), 813–818.

[24] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. 2016. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*. 1928–1937.

[25] Frans A Oliehoek and Christopher Amato. 2016. *A concise introduction to decentralized POMDPs*. Springer.

[26] Frans A Oliehoek, Matthijs TJ Spaan, and Nikos Vlassis. 2008. Optimal and approximate Q-value functions for decentralized POMDPs. *Journal of Artificial Intelligence Research* 32 (2008), 289–353.

[27] Gregory Palmer, Karl Tuyls, Daan Bloembergen, and Rahul Savani. 2018. Lenient Multi-Agent Deep Reinforcement Learning. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*. 443–451.

[28] Sarvapali D Ramchurn, Alessandro Farinelli, Kathryn S Macarthur, and Nicholas R Jennings. 2010. Decentralized coordination in robocup rescue. *Comput. J.* 53, 9 (2010), 1447–1461.

[29] 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*. 4295–4304.

[30] 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. (2019).

[31] Lloyd S Shapley. 1953. A value for n-person games. *Contributions to the Theory of Games* (1953).

[32] David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. 2014. Deterministic policy gradient algorithms. In *International conference on machine learning*. 387–395.

[33] 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*. 5887–5896.

[34] Mukund Sundararajan and Amir Najmi. 2020. The many Shapley values for model explanation. In *International Conference on Machine Learning*. 9269–9278.

[35] Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Vinicius 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 *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*. 2085–2087.

[36] Ming Tan. 1993. Multi-agent reinforcement learning: Independent vs. cooperative agents. In *Proceedings of the tenth international conference on machine learning*. 330–337.

[37] Jiaxuan Wang, Jenna Wiens, and Scott Lundberg. 2021. Shapley Flow: A Graph-based Approach to Interpreting Model Predictions. (2021).

[38] Jianhong Wang, Yuan Zhang, Tae-Kyun Kim, and Yunjie Gu. 2020. Shapley Q-value: A Local Reward Approach to Solve Global Reward Games. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 7285–7292.

[39] Ronald J Williams and Jing Peng. 1991. Function optimization using connectionist reinforcement learning algorithms. *Connection Science* 3, 3 (1991), 241–268.

[40] David H Wolpert and Kagan Tumer. 2002. Optimal payoff functions for members of collectives. In *Modeling complexity in economic and social systems*. World Scientific, 355–369.

[41] Yaodong Yang, Jianye Hao, Guangyong Chen, Hongyao Tang, Yingfeng Chen, Yujing Hu, Changjie Fan, and Zhongyu Wei. 2020. Q-value path decomposition for deep multiagent reinforcement learning. In *International Conference on Machine Learning*. 10706–10715.

[42] Yaodong Yang, Jianye Hao, Ben Liao, Kun Shao, Guangyong Chen, Wulong Liu, and Hongyao Tang. 2020. Qatten: A General Framework for Cooperative Multiagent Reinforcement Learning. *arXiv e-prints* (2020).

[43] Yaodong Yang, Rui Luo, Minne Li, Ming Zhou, Weinan Zhang, and Jun Wang. 2018. Mean field multi-agent reinforcement learning. In *International Conference on Machine Learning*. 5571–5580.

[44] Dayong Ye, Minjie Zhang, and Yun Yang. 2015. A multi-agent framework for packet routing in wireless sensor networks. *sensors* 15, 5 (2015), 10026–10047.

[45] Meng Zhou, Ziyu Liu, Pengwei Sui, Yixuan Li, and Yuk Ying Chung. 2020. Learning Implicit Credit Assignment for Multi-Agent Actor-Critic. *Conference and Workshop on Neural Information Processing Systems*.