D2CFR: Minimize Counterfactual Regret With Deep Dueling Neural Network

Hua Lei, Xuan Wang, Senior Member, IEEE, Zengyue Guo, Jiajia Zhang, and Shuhan Qi

Abstract—Counterfactual regret minimization (CFR) is a popular method for finding approximate Nash equilibrium in two-player zero-sum games with imperfect information. Solving large-scale games with CFR needs a combination of abstraction techniques and certain expert knowledge, which constrains its scalability. Recent neural-based CFR methods mitigate the need for abstraction and expert knowledge by training an efficient network to directly obtain counterfactual regret without abstraction. However, these methods only consider estimating regret values for individual actions, neglecting the evaluation of state values, which are significant for decision-making. In this article, we introduce deep dueling CFR (D2CFR), which emphasizes the state value estimation by employing a novel value network with a dueling structure. Moreover, a rectification module based on a time-shifted Monte Carlo simulation is designed to rectify the inaccurate state value estimation. Extensive experimental results are conducted to show that D2CFR converges faster and outperforms comparison methods on test games.

Index Terms—Counterfactual regret minimization (CFR), imperfect information games (IIGs), Nash equilibrium, neural network.

Manuscript received 21 October 2021; revised 23 April 2022, 4 October 2022, 14 March 2023, and 12 August 2023; accepted 5 September 2023. Date of publication 6 October 2023; date of current version 3 December 2024.

This work was supported in part by the National Natural Science Foundation of China under Grant 62376073 and Grant 62372139, in part by the Ministry of Science and Technology of China under Grant 2020AAA0104200, in part by the Guangdong Provincial Key Laboratory of Novel Security Intelligence Technologies under Grant 2022B1212010005, in part by the Key Fields Research of Guangdong Province under Grant 2020B0101380001, in part by the Shenzhen Foundational Research Funding under Grant JCYJ20200109113427092 and Grant JCYJ20220818102414030, in part by the Basic Research Programs of Taicang (2022) under Grant TC2022JC14, and in part by the PINGAN-HITsz Intelligence Finance Research Center. (Corresponding author: Xuan Wang.)

Hua Lei is with the School of Software, Northwestern Polytechnical University, Xi’an 710072, China, also with the Yangtze River Delta Research Institute, Northwestern Polytechnical University, Taicang 215400, China, also with the School of Computer Science and Technology, Harbin Institute of Technology, Harbin 150001, China, and also with the Guangdong Provincial Key Laboratory of Novel Security Intelligence Technologies, Shenzhen 518000, China (e-mail: huailei@nwpu.edu.cn).

Xuan Wang is with the School of Computer Science and Technology, Harbin Institute of Technology, Shenzhen 518055, China, also with the Peng Cheng Laboratory, Shenzhen 518000, China, and also with the Guangdong Provincial Key Laboratory of Novel Security Intelligence Technologies, Shenzhen 518000, China (e-mail: xuanxuan@cs.hitsz.edu.cn).

Zengyue Guo and Jiajia Zhang are with the School of Computer Science and Technology, Harbin Institute of Technology, Shenzhen 518055, China (e-mail: zengyue@hit.edu.cn).

Shuhan Qi is with the School of Computer Science and Technology, Harbin Institute of Technology, Shenzhen 518055, China, also with the Peng Cheng Laboratory, Shenzhen 518000, China (e-mail: shuhanqi@cs.hitsz.edu.cn).

This article has supplementary downloadable material available at https://doi.org/10.1109/TNNLS.2023.3314638 provided by the authors.

I. INTRODUCTION

In recent years, the research on imperfect information games (IIGs) has attracted more and more attention. Due to the presence of private information unobservable to other players, IIGs are usually considered to be more complex than perfect information games (PIGs) [1], [2]. A typical goal in IIGs is to approximate an equilibrium strategy in which all players’ strategies are optimal [3], [4], [5]. Generally, the solution of two-player IIGs is to find its Nash equilibrium [6], [7]. Counterfactual regret minimization (CFR) is a classical method to compute the Nash equilibrium in two-player IIGs [8]. Initially, CFR and its variants were primarily employed for solving poker games and have achieved great success in the field of IIGs [9], [10], [11], [12], [13]. Notable examples include DeepStack [10], Libratus [11], and Pluribus [13], which defeated top human professionals in their respective poker games. Recently, CFR-based methods have been gradually applied to other fields, such as cyber resource allocation [14], task planning problems [15], distributed intrusion detection [16], and anti-jamming of radar [17].

Although many agents based on deep reinforcement learning (DRL) have achieved remarkable success in recent years [18], [19], [20], their network training requires extensive computational support, and their models lack interpretability. Furthermore, reinforcement learning, particularly DRL, requires complex superparameter tuning during training, such as learning rate, discount factor, and cache size. Moreover, its convergence lacks theoretical guarantees, and network training demands substantial computing power. For instance, AlphaStar [18] requires 16 TPUs for data sampling and network training. Both CFR and DRL methods have their own merits and drawbacks. Given that this article initially investigates typical IIG problems such as poker games and previous CFR-related methods have achieved considerable success in this area, such as DeepStack [10] and Libratus [11], we focus on CFR-based methods. Moreover, compared to DRL methods, training CFR-related methods is relatively simple, with only one overall iteration number as a hyperparameter. Furthermore, strategies derived from DRL-based methods cannot be theoretically proven to be optimal or near optimal, unlike CFR-based methods. Therefore, this article studies solving IIGs with the CFR-based method.

Nevertheless, an inevitable challenge remains: the scale of problems solvable by CFR is limited, with at most $10^{18}$ states, while heads-up no-limit Texas hold’em (HNLH) encompasses nearly $10^{161}$ states. It is crucial to note that recent successful applications of CFR all employ abstraction techniques. Specifically, the original game must be abstracted first, and then, the abstracted game is solved using CFR-based methods.

2162-237X © 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
Consequently, this approach always necessitates certain domain knowledge in designing the abstraction method, which adds complexity to solving large-scale games. Moreover, such abstraction may result in information loss, further impacting the final results.

Recently, some CFR-based methods have integrated neural networks to speed up solutions, avoiding the need for expert knowledge and information loss caused by abstraction. DeepCFR [21] obviates the need for abstraction by using deep neural networks to approximate CFR behavior in the full game. In DeepCFR, the neural network takes an information set (observed cards and bet history) as input and outputs advantage values or probability values for each possible action. DeepCFR demonstrates that its convergence results in an $\epsilon$-Nash equilibrium in two-player zero-sum IIGs. However, it still requires improvement in terms of training efficiency and game performance. Concurrent work has also investigated a similar combination of deep learning with CFR in double neural CFR (DNCFR) [22], which employs two neural networks to fit cumulative regret and average strategy. DNCFR shows good performance in one-card poker and a large Leduc Hold’em, but it may not be theoretically sound and only considers small games [21]. Single DeepCFR (SD-CFR) [23] is a variant of DeepCFR that applies only one neural network instead of training an additional network to approximate the weighted average strategy. It stores all value networks from each CFR iteration to disk and mimics the average policy exactly during playing games. SD-CFR achieves a lower overall approximation error by avoiding the training of an average strategy network. However, it also shares similar problems with DeepCFR, as training efficiency and game performance still need improvement. Deep regret minimization with advantage baselines and model-free learning (DREAM) [24] is a neural form of CFR that samples only a single action at each decision point. In contrast to other regret-based deep learning algorithms, it does not require access to a perfect game simulator. DREAM minimizes regret and converges to an $\epsilon$-Nash equilibrium in two-player zero-sum IIGs with $\epsilon$ proportional to the modeling error. Neural fictitious self-play (NFSP) [25] is the first DRL algorithm to learn a Nash Equilibrium in two-player IIGs, which combines neural network [26] and fictitious self-play [27], [28], [29], [30] to fit an average response strategy approaching Nash equilibrium. The NFSP agent consists of two neural networks: the best response strategy network and the average strategy network. The best response strategy network learns an approximate best response to the historical behavior of other agents, which is trained by reinforcement learning from the memorized experience of playing against fellow agents. The average strategy network learns a model that averages over the agent’s own historical strategies, which is trained by supervised learning from memorized experience of the agent’s own behavior. The NFSP agent behaves according to a mixture of its average strategy and best response strategy when playing games. However, both DREAM and NFSP are very difficult to train in large-scale games.

Although many methods have combined CFR with neural networks, these approaches primarily focus on directly fitting the regret value of actions while neglecting the evaluation of the importance of different states. The regret value comprises two essential elements: state value and state-action value. Typically, the state value represents the expected return of a strategy in the current state, while the state-action value represents the expected return of executing a specific action based on the strategy in the current state. The state value is the expectation of state-action values for all actions, reflecting the advantages and disadvantages of a strategy in the current state [31]. In some DRL studies, researchers have shown that in scenarios where actions do not directly impact the environment, decisions can be made by accurately evaluating the state value without assessing each state-action value [32]. A similar phenomenon has been observed in many IIG scenarios. For example, in the preflop round of Leduc game, the actions call and raise have almost the same effect on the probability of winning when both private and public cards are Ace. It can be found that at this time, a reasonable decision can be made as long as the state value is known, and obtaining the exact value of each action is unnecessary. This article proposes the idea that in some specific states, knowing which action to take is important, while in many other states, only the accurate evaluation of the state value is required, and the choice of action has no repercussions on what happens. However, existing CFR methods for directly evaluating regret values are often unstable, as accurately evaluating the state value in many scenarios is difficult. Therefore, this article decouples the state value and state-action value from the regret value and applies the Monte Carlo (MC) simulation to rectify the inaccurate estimation of the state value. Specifically, an improved variant of DeepCFR, deep dueling CFR (D2CFR), is introduced to solve large-scale IIGs.

In D2CFR, a novel value network with dueling architecture is adopted, whose key insight is to emphasize on the accurate evaluation of state value. In the dueling architecture, the counterfactual value (i.e., state value) and counterfactual action value (i.e., state-action value) are decoupled from the instant regret and modeled separately, which enables the network to better handle the states that are less associated with actions. D2CFR not only improves the training efficiency of the model but also enhances the stability of the learning process. On the one hand, the dueling architecture allows the model to more efficiently capture the relationship between states and actions, as it can learn a single value for each state and a value for every state-action pair, rather than only a single value for each action. On the other hand, the separation of state value and state-action value streams in a dueling architecture can also improve the stability of the learning process. The state value function provides a baseline prediction for the expected state-action value of being in a given state, which can help to reduce the impact of noisy or outlier state-action values on the learning process. Furthermore, a rectification module based on time-shifted MC simulation is designed to rectify the state value estimation in the early stage of training, which accelerates the convergence of value network. The major contributions of this article can be summarized as follows.

1) This article introduces an improved variant of DeepCFR, termed D2CFR. D2CFR focuses on finding approximate Nash equilibrium in two-player large-scale IIGs without any abstraction, which obviates the need for expert knowledge.

2) This article proposes a novel value network with dueling architecture, which aims to decouple the state value estimation and state-action value estimation. This approach allows for accurate evaluation of state values. The novel
value network enables the network to better handle the states that are less associated with actions while also improving the training efficiency of the model and the stability of the learning process.

3) This article designs a rectification module composed of the value network and MC simulation, which further rectifies the inaccurate estimation of state value in the early stage of training.

4) Extensive experimental results demonstrate that D2CFR not only converges faster but also achieves superior performance compared to DeepCFR on test games.

The remainder of this article is organized as follows. Section II introduces the model of extensive-form game, Nash equilibrium, CFR, and Monte Carlo CFR (MCCFR). Section III describes the details of D2CFR. Section IV presents the theoretical analysis of D2CFR. Section V depicts the details of extended experiments. Finally, this article concludes with a summary of D2CFR.

II. BACKGROUND

A. Extensive-Form Game

In the field of IIGs, the extensive-form game is usually used to model sequential decision-making games [6]. Generally, a finite extensive-form IIG contains six components, represented as \( (N, H, P, f, I, u) \) [6]: player \( i \) represents a finite set \( N \) of game players, \( N = \{1, 2, \ldots, n\} \). A node (i.e., history) \( h \) is defined by all information of the current situation, including private knowledge known to only one player. There are a finite set \( H \) of sequences, the possible histories of actions, such that the empty sequence is in \( H \) and every prefix of a sequence in \( H \) is also in \( H \). The possible histories of actions \( a \in A \), \( A(h) = \{a(h, a) \in H\} \), are actions available after a nonterminal history \( h \in H \). \( I \subseteq H \) are terminal histories, for which no actions are available and which award a value to each player. \( P \) is the player function. \( P(h) \) is the player taking action \( a \) after history \( h \). \( P(h) = c \) represents that the chance determines the action after history \( h \). A function \( f_i \) that associates with every history \( h \) for which \( P(h) = c \) a probability measure \( f_i(\cdot|h) \) on \( A(h) \). The set \( I_i \in I \) is an information set of player \( i \). A partition \( I_i \) of \( h \in H: P(h) = i \) with the property that \( A(h) = A(h') \) whenever \( h \) and \( h' \) are in the same member of the partition. For any information set \( I_i \), all nodes \( h, h' \in I_i \), are indistinguishable to player \( i \). The payoff function \( u_i \) defines the payoff of terminal state \( z \) for each player \( i \). For a two-player zero-sum game, there is \( u_1 + u_2 = 0 \).

An extensive-form game is usually represented with a game tree. Fig. 1 shows a game tree for the game Coin Toss. In Fig. 1, each node represents a game state in the game tree. The leaf node, which is known as the terminal node, indicates that the game has ended. Meanwhile, the corresponding payoff is returned after the game ends. In addition, the edge between two nodes represents the action or the decision taken by the game player. In Fig. 1, the player \( P_1 \) can choose between actions left and right, with the action left leading to obtain the payoff directly. If the action right is selected by the player \( P_1 \), then \( P_2 \) has the opportunity to guess how the coin landed. If \( P_2 \) guesses correctly, \( P_1 \) will receive a reward of \(-1\) and \( P_2 \) will receive a reward of \(1\) [33].

B. Nash Equilibrium

Approximating Nash equilibrium has been proven to be an effective way in solving two-player IIGs. The Nash equilibrium is a strategy profile in which no player can improve their utility by deviating from this strategy. The definition of strategy and best response will be given first before introducing Nash equilibrium [7].

In an extensive-form game, a strategy \( \sigma_i(I) \) of player \( i \) is a probability vector over actions on the information set \( I \). A set of strategies for players, \( \sigma_1, \sigma_2, \ldots, \sigma_n \), makes up a strategy profile \( \sigma \) and \( \sigma_{-i} \) represents the set of strategies in \( \sigma \) except the strategy \( \sigma_i \) of player \( i \). In addition, \( \pi^T(h) \) is the probability with \( h \) occurring if all players make decision according to the strategy \( \sigma \) and \( \pi^T(i) = \sum_{h \in I} \pi^T(h) \). \( \pi^T_i(h) \) is the contribution of player \( i \) to this probability. Formally, \( \pi^T_i(h) = \prod_{i \neq j} \pi^T_{ij}(h) \). Accordingly, \( \pi^T(h) \) of history \( h \) is the contribution of all players (including chance player except player \( i \)). A best response to \( \sigma_{-i} \) is a strategy \( \text{BR}(\sigma_{-i}) \), \( \text{BR}(\sigma_{-i}) = \max_{\sigma_i} u_i(\sigma_i, \sigma_{-i}) \), and \( \sigma_i \) represents all possible strategy profiles for player \( i \).

A Nash equilibrium \( \sigma^\ast \) is a strategy profile that each player plays a best response: \( \forall i, u_i(\sigma_i^\ast, \sigma_{-i}^\ast) = \max_{\sigma_i} u_i(\sigma_i, \sigma_{-i}) \). The Nash equilibrium has been proven to exist in all finite games and many infinite games. Since it is difficult to compute Nash equilibrium in most cases, it is more common to compute approximate Nash equilibrium. An \( \epsilon \)-Nash equilibrium, \( u_i(\sigma_i^\ast, \sigma_{-i}^\ast) + \epsilon \geq \max_{\sigma_i} u_i(\sigma_i', \sigma_{-i}) \), is a strategy profile, in which no player can increase their utility by more than \( \epsilon \) by changing their strategy.

C. Counterfactual Regret Minimization

Counterfactual regret minimizing is a classical method to find Nash equilibrium in the two-player zero-sum IIGs [8]. It is an iterative solution method, which mainly includes two steps.

\textbf{Step 1:} Calculate the total regret of action \( a \) at the information \( I \) on the iteration \( T \). The total regret \( R(I, a) \) can be depicted as

\[
R_T(I, a) = \sum_{t=1}^{T} r_t(I, a) \tag{1}
\]

where \( r_t(I, a) \) is instant regret on the iteration \( t \), which is the difference between player \( i \)'s counterfactual values from playing action \( a \) versus playing strategy \( \sigma \) at information set \( I \). \( r_t(I, a) = v^T_t(I, a) - v^T_t(I) \). \( v^T_t(I) \) is counterfactual value
and also represents the state value in this article, which is the expected utility of player $i$ when the information set $I$ is reached, and $v^o_i(I) = \sum_{h \in I, h' \in Z} \pi^o_i(h')u_i(h')$. $v^o_i(I, a)$ is the counterfactual value of action $a$ and also represents the state-action value in this article, which is the same as the player $i$ selects action $a$ all the time at information set $I$ when other players act according to $\sigma_{-i}$, $v^o_i(I, a) = \sum_{h \in I, h' \in Z} \pi^o_i(h')u_i(h')a$. In addition, the positive regret is only considered in most cases, $R^{T \rightarrow \infty}_i(I, a) = \max(R^{T}_i(I, a), 0)$.

**Step 2:** Update the strategy $\sigma^{T+1}_i(I, a)$ of next iteration $T + 1$. A regret matching (RM) algorithm [34] is used to update the strategy on each iteration. Formally, the strategy on the iteration $T + 1$ can be calculated as follows:

$$\sigma^{T+1}_i(I, a) = \frac{\sum_{a' \in A(I)} R^{T+1}_i(I, a') \cdot \sum_{a \in A(I)} R^{T+1}_i(I, a) > 0}{|A(I)|},$$

(2)

If a player plays according to CFR on each iteration, then $R^T_i \leq \sum_{I \in I^T} R^T(I)$. Thus, as $T \rightarrow \infty, (R^T_i / T) \rightarrow 0$. Moreover, the average strategy $\langle \bar{\sigma}_T^i, \bar{\sigma}_T^j \rangle$ forms a 2ε-Nash equilibrium, if the average total regret of both players satisfies $(R^T / T) \leq \varepsilon$ in two-player zero-sum games. Also, the average strategy for information set $I$ on iteration $T$ is $\bar{\sigma}_T^i(I) = (\sum_{t=1}^T \pi^o_i(I)\sigma^i(I))/T$ (5).

**D. Monte Carlo CFR**

Vanilla CFR needs to traverse the full game tree of games, which limits its application in large games. Toward this problem, MCCFR [35] expands the solution scale of vanilla CFR, which only needs to traverse a portion of the game tree. In this article, MCCFR is used as the main scheme. Instead of using counterfactual value in vanilla CFR, MCCFR utilized sampled counterfactual value, which is defined as follows:

$$\tilde{v}_i(\sigma, I|j) = \sum_{z \in Q \cap \sigma_i} \frac{1}{q(Z)}u_i(z)\pi^o_i(z[I]\sigma(z[I], z))$$

(3)

where $Q = \{Q_1, \ldots, Q_t\}$ is a set of subsets of $Z$, and one of these subsets can be called a block. $q_j$ is the probability when block $Q_j$ is considered for current iteration, and $\sum_{j=1}^t q_j = 1$, $q_j > 0$. $q(z) = \sum_{z \in Q \cap \sigma_i} q_j$. Also, the sampled instant regret $\tilde{r}_i(I, a)$ of action $a$ is

$$\tilde{r}_i(I, a) = \tilde{v}_i(\sigma^{\rightarrow}_{-a}, I) - \tilde{v}_i(\sigma^i, I).$$

(4)

Similar to CFR, the strategy of next iteration can be calculated with the RM. Moreover, Lacot et al. [35] proved that the counterfactual value of MCCFR is the same as the normal CFR on the expectation value.

MCCFR includes two kinds of sampling methods: outcome sampling (OS) and external sampling (ES). In ES-MCCFR, the action is sampled when it comes from the opponent and chance player. The sampled counterfactual value of every visited information set can be calculated as follows:

$$\tilde{r}_i(I, a) = (1 - \sigma(a|z[I])) \sum_{z \in Q \cap \sigma_i} u_i(z)\pi^o_i(z[I]|a, z).$$

(5)

Lacot et al. [35] gave a proof that the average strategy solved by ES-MCCFR converges the Nash equilibrium as the iteration increases. Also, its average overall regret is bounded by $R^T_i \leq (1 + (\sqrt{2}/\sqrt{T}))\Delta_{2T}M_i(|A_i|)^{1/2}/\sqrt{T}$.

**E. Conventional Solving Framework With CFR**

The conventional framework of CFR for solving IIGs is reviewed. Restricted by the solution scale of CFR, the game needs to be reduced to the scale that CFR can be solved. The conventional framework of CFR is shown in Fig. 2. Generally, it includes three steps to obtain the final strategy. First, abstract the original game [36], [37], [38]. Then, apply CFR to solve the strategy of the abstracted game. Finally, map the solved strategy back to the original game [39]. However, it can be clearly found that whether abstracting the original game or mapping strategy will bring certain errors to final results.

**III. OUR METHOD**

**A. Overview of the Framework**

Now, an overview of the proposed D2CFR is given. As shown in Fig. 3, each iteration of D2CFR can be divided
DeepCFR, the value network in D2CFR adopts a novel dueling information set is used as input, the instant regret value of game states to the game strategy. To be specific, when a given D2CFR and conventional CFR. The end-to-end means from to obtain the strategy, instead of traversing the whole game tree estimated by the value network. An end-to-end way is adopted combined with MC simulation is designed. Finally, the end-end policy output is realized through the policy network. D2CFR is to solve the strategy estimate the state value more accurately, a rectification module introduced, mainly including the value network, rectification module, and policy network. D2CFR will also be trained continuously, that is, the value network is used to fit the instant regret value, while the policy network is an approximation of the average strategy. It is worth noting that, compared with DeepCFR, D2CFR decouples the state value and state-action value from the regret value calculation and tries to learn these two values jointly. Moreover, D2CFR introduces a time-shifted MC simulation to rectify the state value estimation.

B. Method Architecture of D2CFR

In this section, the architecture of D2CFR will be detailedly introduced, mainly including the value network, rectification module, and policy network. D2CFR is to solve the strategy through CFR and the neural network. To this end, first, a value network is specially designed that can decouple the value function and the state-action value function. Then, in order to estimate the state value more accurately, a rectification module combined with MC simulation is designed. Finally, the end-to-end policy output is realized through the policy network. These three parts will be introduced in turn in the following.

1) Value Network: As described in (1) and (2), one of the preconditions of updating the strategy by the RM is to obtain the instant regret value of each action at information sets. In the proposed D2CFR, the instant regret value is estimated by the value network. An end-to-end way is adopted to obtain the strategy, instead of traversing the whole game tree completely in CFR. This is the essential difference between D2CFR and conventional CFR. The end-to-end means from game states to the game strategy. To be specific, when a given information set is used as input, the instant regret value of each action is directly output through the value network. Each action refers to all legal actions in the current information set.

Compared with the fully connected network structure in DeepCFR, the value network in D2CFR adopts a novel dueling network structure (called DNet), which aims to decouple the counterfactual value (i.e., state value) and the counterfactual action value (i.e., state-action value) from the instant regret estimation. As shown in Fig. 4, the dueling structure includes two sublayers: the shorter one denotes the sublayer used to explicitly estimate the mean counterfactual value of the information set, while the longer one represents the counterfactual value of an action for the information set. The two sublayers share a common feature learning module.

In the proposed DNet, the dueling network is not composed by simply dividing the full connection layer into two sublayers. In order to make the counterfactual value estimation more accurate, a multiply loss function has been utilized to learn the counterfactual value and instant regret value simultaneously. The key insight here is that estimating the counterfactual value accurately is more important than the counterfactual action value. Since for many states, estimating the counterfactual value of all actions is unnecessary. For example, in the game of driving a vehicle, when there is no car in front of the agent, the vehicle’s own actions are not very different. At this time, the agent pays more attention to the value of the state, while when there is a car in front of the agent, the agent starts to pay attention to the difference in the advantage values of different actions. Thus, in many cases, obtaining the exact value of each action is unnecessary, and a reasonable decision can be made as long as the state value is known. In this way, evaluating the counterfactual value, which reflects the value of state, is very important for the calculation of instant regret value.

2) Rectification Module: As mentioned above, the counterfactual value is valuable for the calculation of instant regret values. However, its estimation is a difficult task, especially in the early stage of iteration. This is because CFR is an iterative method, and making the strategy converge is a slow progress. In other words, in the early stage of iteration, the ground truth for training the network is totally inaccurate. This problem greatly limits the speed of strategy learning.

In this article, a rectification module is designed to improve the accuracy of the state estimation by introducing MC [40]. The rectification module consists of two parts: value network and MC simulation. In the rectification module, to estimate the counterfactual value of current state, a time-shifted weighted combination of the MC simulation and the value network is adopted. To be specific, the counterfactual value and the counterfactual action value are estimated in the penultimate network of the DNet, as \( r^\sigma(I, a) = v^\sigma(I, a) - v^\sigma(I) \) described. In the rectification module, the counterfactual value \( v^\sigma(I) \) in the equation comes from two parts: one is still from the...
Algorithm 1 D2CFR

**Input:** the game $G$, ES-MCCFR iteration number $T$, traversal number $K$, constants $c$, $\gamma$, parameters $\alpha$ and $\beta$, regret value network parameters $\theta$ for each player, regret value memories $M_{V,1}$, $M_{V,2}$, strategy memory $M_{T}$, Monte Carlo times $N$.

**Output:** $\theta_{T}$.

1: $E_{v} = G$.
2: Initialize each player’s value network $r_{v,NN}(I, a|\theta_{1})$ with parameters $\theta_{1}$.
3: Initialize reservoir-sampled regret value memories $M_{V,1}$, $M_{V,2}$ and strategy memory $M_{T}$.
4: for ES-MCCFR iteration $t = 1$ to $T$ do
5:   for each player $i$ do
6:     for traversal $k = 1$ to $K$ do
7:       Traverse$(\theta_{t}, i, \theta_{1}, \theta_{2}, M_{V,i}, M_{T,i}, t, N, \alpha, \beta)$ \r\l& Collect data from the game $G$ traversal with ES-MCCFR
8:     end for
9:   end for
10:   Train $\theta_{t}$ from scratch with loss $L(\theta_{t}) = \mathbb{E}_{(I,t,a') \sim M_{v}}[t' \sum_{a}(r_{v}^{t}(a) - r_{v,NN}(I, a|\theta_{t}))^{2}]$
11: end for
12: Train $\theta_{T}$ with loss $L(\theta_{T}) = \mathbb{E}_{(I,t,a') \sim M_{v}}[t' \sum_{a}(\sigma^{t}(a) - \Pi(I, a|\theta_{T}))^{2} - \gamma KL[\Pi_{\theta_{T}}^{t}(\cdot|I), \frac{1}{T} \sum_{t'=t}^{t'} \Pi_{\theta_{T}}^{t'}(\cdot|I)]]$
13: return $\theta_{T}$

DNet, represented with $v_{NN}^{\phi}(I)$; the other one is from the MC, represented with $v_{MC}^{\phi}(I)$. Here, $v_{MC}^{\phi}(I)$ is the value from MC simulation. Finally, $v_{NN}^{\phi}(I)$ in original DNet is replaced with the combination of $v_{NN}^{\phi}(I)$ and $v_{MC}^{\phi}(I)$. It can be formally described as

$$v_{i}^{\phi}(I) = \alpha v_{NN}^{\phi}(I) + \beta v_{MC}^{\phi}(I)$$

where $\alpha + \beta = 1$, $\alpha \geq 0$ and $\beta \geq 0$. It is worth noting that $\alpha$ and $\beta$ are varying with iterations, $\alpha = 0.01 + t/(t + 1)$ and $\beta = 0.99 - t/(t + 1)$. This means that in the early iterative training, the model relies on the MC simulation to update the model. With the progress of training, the model increasingly believes in the estimation of the value network itself.

3) **Policy Network:** Similar to DeepCFR, a policy network is applied to learn the average strategy for final decisions, which is shown in Fig. 4. Actually, it is a simple but effective method for strategy learning. For CFR-based methods, the average strategy obtained by iterative learning will approach its Nash equilibrium strategy. As described in DeepCFR, using a neural network to approximate the average strategy will eventually lead to a good policy network. Since the average strategy is not used in training, there is no need to consider the large approximation error in the early stage. Thus, it is reasonable to approximate the average strategy by the full connection network.

**C. Algorithm of D2CFR**

In this section, the overall algorithms of D2CFR are shown in Algorithms 1 and 2. The relationship between Algorithms 1 and 2 is whole and part. Algorithm 1 is an overall solution algorithm flow of D2CFR. Algorithm 2 is a detailed expansion of a specific function “traverse” in Algorithm 1, which is only a subset of Algorithm 1.

It can be found that D2CFR traverses the game tree several times by using ES-MCCFR for each player on each iteration. In the procedure of traversing the game tree, the KR calculates the strategy of next iteration through regret values, which are fit by a rectification module. Besides, the value network and the policy network are constantly trained and optimized with samples that are collected by traversing the game tree. Finally, the average strategy is approximated by the policy network, which is able to approach the approximate Nash equilibrium.

More specifically, D2CFR first needs to collect a large number of samples for network training. To collect training samples, it needs to traverse and solve the game tree. At this time, D2CFR implements this process through Algorithm 2. Algorithm 2 starts from the current node through ES-MCCFR and expands the game tree according to the current strategy until it reaches the terminal node. Then, it calculates the regret value of the action and finally collects the corresponding training samples. After training samples are obtained, the network model of D2CFR is trained through the process of steps 9–12 in Algorithm 1, and the D2CFR network model is finally obtained.

**D. Theoretical Analysis of D2CFR**

In order to solve the game, CFR needs to traverse the full game tree. It calculates the instant regret and total regret of legal actions on each information set according to the current strategy. Also, the strategy of next iteration is obtained with the RM algorithm. Consistent with DeepCFR, in D2CFR, it also adopts an improved variant of CFR, MCCFR [35]. Therefore, in this section, the theoretical analysis will be conducted from two aspects that are completely consistent with the proof steps of DeepCFR. First, the convergence guarantee of MCCFR will be introduced. Second, the convergence guarantee of D2CFR will be analyzed.

Unlike CFR traversing the complete game tree, MCCFR traverses only a part of the game tree in each traversal. On each iteration, MCCFR needs to traverse the part of the game tree many times. In addition, although MCCFR adopts the MC sampling technique, it still has a good theoretical guarantee. A sampled counterfactual value was designed to match the counterfactual value on expectation.

$$E_{\tilde{\sigma}_{1:|I|}}[v_{i}(\sigma, I)] = v_{i}(\sigma, I).$$

Compared with the regret bound $\Delta_{\sigma_{1:|I|}}$, the regret bound $R_{T}^{\phi}$ of MCCFR is $(1 + \sqrt{2}/(\rho K))\Delta_{\sigma_{1:|I|}}^{1/2}/\sqrt{T}$, where $\rho \in (0, 1]$. Thus, as $T \to \infty$, then $(R_{T}^{\phi}/T) \to 0$, that is to say, the average strategy obtained by MCCFR can reach approximate Nash equilibrium [35].

In DeepCFR [21], it has proved that the total regret at iteration $T$ is bounded by $R_{T}^{\phi} \leq (1 + \sqrt{2}/(\rho K))\Delta_{\sigma_{1:|I|}}^{1/2}/\sqrt{T} + 4T I_{p}(|A|\Delta_{\phi})^{1/2}$ with
Algorithm 2 Traverse the Game With ES-MCCFR

function Traverse(h, i, θ1, θ2, MV,i, MP, t, N, α, β)

Input: history h, traversal player i, regret value network parameters θ for each player, regret value memory MV for each player i, strategy memory MP, ES-MCCFR iteration t, Monte Carlo times N, parameters α and β.

1: if h is a terminal node then
2: return the payoff of the player i
3: else if h is a chance node then
4: a ∼ σ(h)
5: return Traverse(h · a, i, θ1, θ2, MV,i, MP, t, N, α, β)
6: else
7: if it’s the traverser’s turn to act then
8: Compute strategy σ′(I) from predicted regret values r_i,NN(I(h), a|θi) of rectification module by using the RM.
9: The predicted counterfactual value of rectification module V_i,I(h) = αV_i,NN(I(h)|θi) + βV_i,MC(I(h)).
10: V_i,NN(I(h)) is from the predicted value of value network, V_i,MC(I(h)) is from the value of N simulations in the current information set with MC.
11: for a ∈ A(h) do
12: vi(a) ← Traverse(h · a, i, θ1, θ2, MV,i, MP, t, N, α, β) → Traverse each action
13: r_i(I(a)) ← V_i(a) − Σa′∈A(h) σ(I, a′) · vi(a′) → Compute regret values
14: end for
15: Insert the information set and its action regret values (I, t, r_i(I)) into the regret value memory MV
16: else
17: Compute strategy σ′(I) from predicted regret values r−i,NN(I(h), a|θ−i) of rectification module by using the RM.
18: The predicted counterfactual value of rectification module V−i,I(h) = αV−i,NN(I(h)|θ−i) + βV−i,MC(I(h)).
19: Insert the information set and its action probability (I, t, σ′(I)) into the strategy memory MP.
20: Sample an action a from the probability distribution σ′(I).
21: end if
22: return Traverse(h · a, i, θ1, θ2, MV,i, MP, t, N, α, β)
23: end if

IV. EXPERIMENTS

In this section, the experimental setup and experimental results are introduced. The testbed, implementation, and evaluation metric will be detailedly described in the experimental setup. Comparative experiments and ablation studies are conducted in the experimental results.

A. EXPERIMENTAL SETUP

1) Experimental Testbed: Poker is a family of games that includes hidden information, deception, and bluffing, which has been used as a domain for testing game-theoretic techniques in the field of IIGs [9], [10], [11], [12], [13]. Many successful CFR-based methods and applications take poker games as the testbed to verify their effectiveness, such as DeepStack [10], Libratus [11], and Pluribus [13]. In this article, Leduc hold’em [42] and HNLH are used to test the effectiveness of D2CFR. These two games are both two-player games.

a) Game rules of Leduc [43]: Leduc hold’em is a popular benchmark for IIGs because of its size and strategic complexity. In Leduc hold’em, there are six cards: two each of jack, queen, and king. There are two rounds: preflop and flop. In the round of preflop, each player is dealt one card as a private card and an ante of 1 is placed in the pot. Player 1 goes first and the maximum betting number is 2 in the preflop round. Then, one public card is dealt before the flop round begins. Player 1 goes first again and the maximum betting number is 2 in the flop round. If one of the players has a pair with the public card, that player wins. Otherwise, the player with the higher card wins.

b) Game rules of HNLH [44]: HNLH is a two-player IIG. HNLH totally contains 52 cards and consists of four rounds. The four betting rounds are preflop, flop, turn, and river. Three kinds of actions, fold, call, and raise, can be chosen by each player on a round of betting. If the acting player chooses the action fold, it means that this player is out of the current game and cannot obtain any chip in this game.
If the acting player chooses the action call, it means that this player bets chips into the pot. The number of betting chips should be equal to the most chips that other players have contributed to the pot. If the acting player chooses the action raise, it means that this player can add more chips to the pot. Also, the number of raising chips should be more than any other player raised so far. In addition, there is no limit to the number of times a player can raise. The player can choose how much to raise and the subsequent raise on each round should be at least as large as the previous raising chips.

At the beginning of the preflop round, two cards are dealt to each player from a standard 52-deck. Also, it should be noted that these two cards are private cards for each player, which are unobservable to each other. Three public cards are dealt in the flop round and a public card is dealt in the last two rounds. Here, the public card represents that this card is observable to each player. The player will be the winner and obtain all pot chips when this player is the only remaining player in the game. Otherwise, the player with five best cards that consists of two private cards of the player and three public cards from five public cards wins the pot. In the case of a tie, the pot will be split equally to winning players.

2) Implementation Detail: The experiments are conducted on the platform OpenSpiel [45], which is a collection of environments and algorithms for research in the field of IIGs. Where not otherwise noted, the comparison algorithms DeepCFR and NFSP both are trained completely according to the algorithm provided by the OpenSpiel. For SD-CFR, it is reproduced according to the original paper [23]. All parameters are set completely according to the original paper.

For D2CFR, hyperparameters are set as follows. For the DNet, it includes seven layers, and the information set is taken as input and outputs the regret value of each action. The policy network has seven fully connected layers and outputs the probability of each legal action. The batch size is 200. The parameters are updated by the Adam optimizer [46] with a learning rate of 0.001. The memory capacity is 100 000. The total number of iterations \( T \) is 1000; which is enough for all methods. For the times of MC simulation in the rectification module, the times \( N \) is 800 and \( \alpha = 0.01 + 1/(t+1) \) and \( \beta = 0.99 - t/(t+1) \). The parameters \( c = 10 \) and \( \gamma = 0.05 \). In addition, all experiments are conducted on four Xeon\(^1\) CPUs of E5-2640 with ten cores @2.40 GHz and one Tesla P100 GPU with 16-GB memory.

3) Evaluation Metric: In this article, the effectiveness of D2CFR will be evaluated with two popular metrics in the field of IIGs: exploitability and head-to-head performance, just like the experimental setup in previous works [8], [10], [11], [45].

Exploitability is a standard metric, which is used to measure the strategy in two-player IIGs. The exploitability \( e(\sigma_i) \) of strategy \( \sigma_i \) indicates how close strategy \( \sigma_i \) is to a Nash equilibrium strategy in a two-player zero-sum game. Also, the lower the exploitability, the better the strategy. The exploitability \( e(\sigma_i) \) is formally defined as

\[
e(\sigma_i) = u_i(\sigma_i^*, BR(\sigma_i^*)) - u_i(\sigma_i, BR(\sigma_i))
\]

where \( BR(\sigma_i) \) is the best response to the strategy \( \sigma_i \), which has been introduced in Section II-B.

Considering that the exploitability only can be calculated in small-scale games, it is only used in the evaluation of Leduc hold’em. For the HNLH, the head-to-head performance is measured to further verify the effectiveness of D2CFR. The head-to-head performance reflects the actual gaming ability of the method in the game. A head-to-head contest or competition is one in which two players or groups compete directly against each other. Under the rules of the HNLH, two agents implemented by two comparison methods are directly allowed to play against each other. These two agents continuously play 100 000 games, and finally, the game results are counted.

B. Comparative Experiment Results

In this section, the comparison experiment is conducted to verify the effectiveness of D2CFR. Four state-of-the-art methods in recent years, NFSP [25], DeepCFR [21], SD-CFR [23], and DREAM [24], are used as comparative methods. The comparison experiments are conducted on the HNLH, a popular and classical testbed for CFR-based methods in the field of IIGs.

In order to show the performance of D2CFR, the exploitability is first tested on the Leduc compared with the other three CFR-based methods DeepCFR, SD-CFR, and DREAM, as shown in Fig. 5. It should be noted that NFSP is not used in this experiment, because NFSP is not a CFR-based method and CFR iteration is not involved in its training process. Second, the head-to-head performance is tested on both Leduc and HNLH, as shown in Table I.

Fig. 5 shows that D2CFR reaches a lower level of exploitability compared with other methods. It can be found that as the number of iterations increases, the exploitability of these five methods presents a decreasing trend and gradually converges. The exploitability represents the error with the Nash equilibrium strategy, so it can be concluded that with the increase of the number of iterations, the finally solved strategy is an approximate Nash equilibrium strategy. Specifically, compared with the DeepCFR, the exploitability of D2CFR has been significantly lower than that of DeepCFR except for 110th–120th iterations. This result shows that D2CFR is very effective in improving DeepCFR. In addition, it can be found that except in the early 170th iterations, SD-CFR outperforms DeepCFR in exploitability. This result further shows that the replication of SD-CFR is successful and the experimental result is credible.

Table I gives the detailed head-to-head performance on Leduc and the HNLH, which is measured in milli–big blinds per game (mbb/g), the average number of big blinds won per
1000 games. The results are recorded with average winning in mbb/g followed by the 95% confidence interval (for 95% confidence interval, the result can be represented with $\bar{x} \pm Z(s/\sqrt{n})$, where $\bar{x}$ is the average winning, $Z = 1.960$, $s$ is the standard deviation, and $n$ is the number of games, $n = 10\,000$). This representation of game results used here is a commonly used representation method in the field of IIGs, especially in large-scale poker games. For example, in DeepStack [10], Libratus [11], and DeepCFR [21]. D2CFR defeated DeepCFR, SD-CFR, NFSP, and DREAM by 276.1 $\pm$ 47.2, 40.85 $\pm$ 9.2, 219 $\pm$ 28.33, and 37.2 $\pm$ 40.08 mbb/g on Leduc, respectively. Also, on the HNLH, D2CFR defeated DeepCFR, SD-CFR, NFSP, and DREAM by 75.52 $\pm$ 16.0, 64.08 $\pm$ 15.7, 119.44 $\pm$ 27.48, and 7.8 $\pm$ 8.9 mbb/g, respectively. These results show that D2CFR is obviously better than the comparison methods.

To sum up, first, D2CFR shows much better performance than DeepCFR in terms of exploitability and head-to-head performance. It shows that our improvement on DeepCFR is very effective. Second, the better results of head-to-head performance with SD-CFR, NFSP, and DREAM further verify the excellent performance of the proposed method D2CFR.

C. Ablation Study

In this section, the ablation study is conducted, which analyzes the effect of each proposed component (DNet and rectification module). The ablation study includes three aspects. First, the effectiveness of the DNet is evaluated. Second, the experiment is conducted to test the performance of the rectification module. Third, the different setting values of $N$ in the MC simulation are conducted.

1) Effectiveness of DNet: D2CFR takes DNet as the value network compared with the fully connected network in vanilla DeepCFR. “D2CFR w/o DNet” can be regarded as DeepCFR in this article. The winning and policy loss are used to measure this improvement from the performance and the convergence of the policy network. Here, the winning is obtained by head-to-head gaming of two models at the same number of iterations. The results are shown in Fig. 6.

It can be found that the DNet obviously improves the performance of D2CFR in Fig. 6(a). A positive winning means that our method is better than comparison methods. After the 100th iterations, the “D2CFR with DNet” is always better compared with the “D2CFR w/o DNet” from the winning. Fig. 6(b) shows that the policy loss of the “D2CFR with DNet” is far below than that of “D2CFR w/o DNet” all the time. Also, the gap between the two networks is widening after the 120th iterations, which shows that the improvement is also very helpful to improve the convergence of the policy network.

2) Effectiveness of Rectification Module: The rectification module is the core of D2CFR when training the whole neural network, which is used to correct the inaccuracy of state estimation. Here, the winning and the value loss are used to test the effectiveness of this component. In addition, “ReM” is used to represent the rectification module in the following. “D2CFR with ReM” and “D2CFR w/o ReM” mean D2CFR with and without the rectification module, respectively.

It can be found that the rectification module obviously reduces the approximation error from Fig. 7(b). The loss of the rectification module is always lower than that of “D2CFR w/o ReM” since the 12th iterations. Also, there is a clear gap from the 20th to 500th iteration, which reflects that the rectification module is effective in reducing loss, especially in the early stage of iterations. Fig. 7(a) shows that the
winning is basically the same as that of “D2CFR w/o ReM” after the 200th iterations. Specifically, the winning has begun to increase from the 100th iterations. This shows that the performance of “D2CFR with ReM” has not still decreased when the approximation error with the rectification module is reduced.

3) Different Setting Values of N in MC Simulation: Different setting values will have different effects on the MC simulation. Eight different settings of N are tested, N = 50, 100, 150, 200, 300, 500, 800, and 1000. The policy loss and exploitability are used to evaluate its performance for selecting an optimal value of N. The experiment results are shown in Fig. 8.

It can be found that the exploitability of N = 800 and N = 1000 is lower than that of other settings from Fig. 8(b). Also, the exploitability of N = 800 is better than that of N = 1000 except for 350th–680th iterations. Fig. 8(a) shows that N = 300 and N = 800 are better than that of others in terms of the policy loss. Also, the policy loss of N = 800 is lower than that of N = 300 except 150th–550th iterations. Therefore, considering these three aspects, N = 800 is set as the final setting in the experiment.

V. CONCLUSION

In this article, we present an improved version of D2CFR based on DeepCFR, which can approach approximate Nash equilibrium in two-player IIGs. D2CFR constructs a DNet as the value network that decouples the state value estimation and the state-action value estimation, which can provide accurate value network that decouples the state value estimation and the state-action value estimation, which can provide accurate value estimation. Also, this value network enables the network to better handle the states that are less associated with actions. Moreover, a rectification module based on MC simulation is designed, which can further rectify the error estimation of states in the early stage. Extensive experimental results show that the improvement of D2CFR is effective, and D2CFR outperforms other state-of-the-art methods on test games.

In the future, there are still several parts of work worth further studying based on this article. First, better fine-tuned network architecture will be helpful to improve the performance of D2CFR. Second, it would be interesting to expand D2CFR to larger and more complex games than poker games (i.e., StarCraft II with more players). Third, it is also very challenging to explore the application of D2CFR in other fields of IIGs (i.e., resource allocation of security games).

APPENDIX A

CALCULATION PROCESS DESCRIPTION OF CFR

In order to more clearly show the total regret process of CFR calculation, the “rock–paper–scissors” game will be briefly described. Consider an initial random strategy. The probability calculation, the “rock–paper–scissors” game will be briefly described. Consider an initial random strategy. The probability calculation, the “rock–paper–scissors” game will be briefly described. Consider an initial random strategy. The probability calculation, the “rock–paper–scissors” game will be briefly described. Consider an initial random strategy. The probability calculation, the “rock–paper–scissors” game will be briefly described. Consider an initial random strategy. The probability calculation, the “rock–paper–scissors” game will be briefly described. Consider an initial random strategy. The probability calculation, the “rock–paper–scissors” game will be briefly described. Consider an initial random strategy. The probability calculation, the “rock–paper–scissors” game will be briefly described. Consider an initial random strategy. The probability

\begin{align*}
\text{v}_i(I, a_{\text{rock}}) &= 0.3 \times 1 \times 0 = 0. \quad (8) \\
\text{v}_i(I, a_{\text{paper}}) &= 0.3 \times 1 \times 1 = 0.3. \quad (9) \\
\text{v}_i(I, a_{\text{scissor}}) &= 0.3 \times 1 \times (-1) = -0.3. \quad (10)
\end{align*}

Then, for the acting player i, the counterfactual value \( v_i(I) \) can be calculated according to

\[ v_i(I) = \sum_{h \in \mathcal{H}} \pi^c_{\pi^h}(h)r_i(h, a, h')u_i(h'). \]

The total regret process of CFR

\begin{align*}
\text{r}_i(I, a_{\text{rock}}) &= 0 - (-0.09) = 0.09 \quad (12)
\end{align*}

where \( r_i(I, a_{\text{rock}}) \) represents the instant regret of the action “rock” for the player i on the first iteration, and the superscript “1” of r indicates the current number of iterations. \( r_i(I, a) \) for the action “paper” is given as follows:

\begin{align*}
\text{r}_i(I, a_{\text{paper}}) &= 0.3 - (-0.09) = 0.39. \quad (13)
\end{align*}
The specific calculation method for each variable of player $i$ in the CFR method has been given, and the same calculation for the action “rock,” for the action “paper,” and for the action “scissor” can also be calculated.

Finally, based on the calculation of previous variables, the total regret of the action “rock” for the player $i$ on the first iteration and the superscript “1” of $R_i$ indicates the current number of iterations. The number of iterations in the calculation of the total regret is “1” because one iteration is conducted at this time. For example, if four iterations are conducted, the total regret will be $R^4_i(a_{\text{scissor}}) = \sum_{t=1}^{4} r^t_i(a_{\text{scissor}})$. $R^1_i(a_{\text{paper}})$ for the action “paper” is given as follows:

$$R^1_i(I, a_{\text{paper}}) = \sum_{t=1}^{1} r^t_i(I, a_{\text{paper}}) = 0.39. \quad (16)$$

$R^1_i(I, a_{\text{scissor}})$ for the action “scissor” is given as follows:

$$R^1_i(I, a_{\text{scissor}}) = \sum_{t=1}^{1} r^t_i(I, a_{\text{scissor}}) = -0.21. \quad (17)$$

where the positive regret is only considered in most cases, $R^{t+1}_i(I, a) = \max(R^t_i(I, a), 0)$. Thus, $R^1_i(I, a_{\text{scissor}})$ for the action “scissor” is $R^{1+1}_i(I, a_{\text{scissor}}) = \max(-0.21, 0) = 0$.

Through the above calculation process, the calculation of each variable [counterfactual action value $v(I, a)$, counterfactual value $v(I)$, instant regret $r(I, a)$, and total regret $R(I, a)$] in the CFR method can be obtained. After obtaining the above values, the RM formula (2) in Section II-C is used to carry out the strategy for the next iteration.

The specific calculation method for each variable of player $i$ in the CFR method has been given, and the same calculation method is also applicable to the opponent player $-i$. For the opponent player $-i$, the counterfactual action value $v_{-i}(I, a)$ for the action “rock,” paper, and scissor can also be calculated. $v_{-i}(I, a_{\text{rock}})$ for the action “rock” is given as follows:

$$v_{-i}(I, a_{\text{rock}}) = 0.5 \times 1 \times 1 = 0.5. \quad (18)$$

$v_{-i}(I, a_{\text{paper}})$ for the action “paper” is given as follows:

$$v_{-i}(I, a_{\text{paper}}) = 0.5 \times 1 \times (-1) = -0.5. \quad (19)$$

$v_{-i}(I, a_{\text{scissor}})$ for the action “scissor” is given as follows:

$$v_{-i}(I, a_{\text{scissor}}) = 0.5 \times 1 \times 0 = 0. \quad (20)$$

Then, for the opponent player $-i$, the counterfactual value $v_{-i}(I)$ can be calculated as follows:

$$v_{-i}(I) = 0.3 \times 0.5 + 0.2 \times (-0.5) + 0.5 \times 0 = 0.5. \quad (21)$$

Then, for the opponent player $-i$, $r^{1}_{-i}(I, a_{\text{rock}})$ for the action “rock” is given as follows:

$$r^{1}_{-i}(I, a_{\text{rock}}) = 0.5 - 0.05 = 0.45. \quad (22)$$

$r^{1}_{-i}(I, a_{\text{paper}})$ for the action “paper” is given as follows:

$$r^{1}_{-i}(I, a_{\text{paper}}) = -0.5 - 0.05 = -0.55. \quad (23)$$

$r^{1}_{-i}(I, a_{\text{scissor}})$ for the action “scissor” is given as follows:

$$r^{1}_{-i}(I, a_{\text{scissor}}) = 0 - 0.05 = -0.05. \quad (24)$$

Finally, for the opponent player $-i$, the total regret $R^{1}_{-i}(I, a_{\text{rock}})$ for the action “rock” is given as follows:

$$R^{1}_{-i}(I, a_{\text{rock}}) = \sum_{i=1}^{1} r^{1}_{-i}(I, a_{\text{rock}}) = 0.45. \quad (25)$$

$r^{1}_{-i}(I, a_{\text{paper}})$ for the action “paper” is given as follows:

$$R^{1}_{-i}(I, a_{\text{paper}}) = \sum_{i=1}^{1} r^{1}_{-i}(I, a_{\text{paper}}) = -0.55. \quad (26)$$

where $R^{1+1}_{-i}(I, a_{\text{paper}}) = \max(-0.55, 0) = 0$ and $R^{1+1}_{-i}(I, a_{\text{scissor}})$ for the action “scissor” is given as follows:

$$R^{1+1}_{-i}(I, a_{\text{scissor}}) = \max(-0.05, 0) = 0. \quad (27)$$

In this way, after the first game, the values [counterfactual action value $v(I, a)$, counterfactual value $v(I)$, instant regret $r(I, a)$, and total regret $R(I, a)$ in the CFR method] of both sides of game players are calculated in detail.

**APPENDIX B**

**PROOF SUPPLEMENT IN SECTION III-D**

The detailed proof analysis of D2CFR will be given in this section.

A. Theoretical Analysis of D2CFR

In order to solve the game, CFR needs to traverse the full game tree. It calculates the instant regret and total regret of legal actions on each information set according to the current strategy. Also, the strategy of next iteration is obtained with the RM. Consistent with DeepCFR [21], in D2CFR, it also adopts an improved variant of CFR, ES-MCCFR [35]. The convergence proof of ES-MCCFR is introduced in Section C-A in the Supplementary Material. Moreover, the detailed proof of DeepCFR is described in Section C-B in the Supplementary Material.

As stated in this article, D2CFR is entirely based on the improvement of DeepCFR [21]. The theoretical analysis of D2CFR will also be based on the theoretical foundation of DeepCFR. In DeepCFR [21], it has proved that the total regret at iteration $T$ is bounded by $R_T \leq (1 + \sqrt{2/(\rho K^{1/2})}) \Delta(I) \sqrt{T} + 4T ||A|| \Delta \epsilon^{1/2}$ with probability $1 - \rho$. Also, $L_T' \leq \epsilon_L$, where $L_T'$ is the average mse loss for $V_p(I, a|\theta_p)$ on a sample in $M_{V,p}$ at iteration $t$ and $L_T'$ is the minimum loss achievable for any function $V$. The corresponding proof is described in Section C-B in the Supplementary Material.

In D2CFR, $L_T'$ is different from that in DeepCFR. Samples in the value memory come from the rectification module, which is composed of the value network and the MC simulation. The function in D2CFR is $r_{p,NN}(I, (h), a|\theta_p) = V_p(I, a|\theta_p) - V_p(I(h), a|\theta_p)$ and $V_p(I(h), a|\theta_p) = \alpha V_{p,NN}(I(h), a|\theta_p) + \beta V_{p,MC}(I(h))$, where $\alpha + \beta = 1$ and $\alpha \geq 0$ and $\beta \geq 0$. It should be noted that both subscripts $p$ and $i$ represent acting player, and their meanings

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
are consistent. The unified use of $p$ here is to maintain the consistency of the symbols in the proof. Here, $V_{p, MC}$ obtained by the MC simulation is a finite constant.

It should be noted here that the predicted value $V_{p}(I, a|\theta)$ called the advantage value in DeepCFR [21] is essentially the instant regret, which can be verified from the description of this article and the introduction of the algorithm in [21] (e.g., in [21, Sec. 4]: “a new network is trained from scratch to determine parameters $\theta^s$ by minimizing mse between predicted advantage $V_{p}(I, a|\theta)$ and samples of instant regrets from prior iterations $t' \geq t$, $\hat{r}(I, a)$, drawn from the memory”). Therefore, although the value network output in DeepCFR is represented by $V_{p}(I, a|\theta)$, it corresponds to the instant regret $r(I, a|\theta)$ in this article. In this case, $\|\hat{r}(I) - V(I|\theta^s)\|_2^2$ in the definition of loss function $\mathcal{L}_V$ mentioned in Section V-C2 or in DeepCFR [21] and $\|\hat{r}(I) - r(I|\theta^s)\|_2^2$ in D2CFR are consistent.

Based on the above analysis, consider that D2CFR is consistent in terms of the network input to output form, both of which are inputting states and outputting instant regrets. It is obvious that the only difference of $\|\hat{r}(I) - V(I|\theta^s)\|_2^2$ is the instant regret value $r(I|\theta^s)$ of the two methods DeepCFR and D2CFR. Based on the definition of instant regret $r(I|\theta^s) = V(I, a) - V(I)$ and the value network in D2CFR, it can be found that the difference in instant regret between the two methods comes from the difference in the counterfactual value $V_{NN}(I)$. D2CFR decoupled $V_{NN}(I)$ and $V_{NN}(I, a)$ through a dueling structure and corrected $V_{NN}(I)$ through a rectification module. At this point, based on the theory analysis of DeepCFR, it only needs to analyze the impact of $V_{NN}(I)$ on $V_{NN}(I, a|\theta)$ in D2CFR ($r_{NN}(I, a|\theta)$) is corresponding to $V(I, a|\theta)$ in DeepCFR and $V_{NN}(I)$ represents this value that comes from the network).

Specifically, in D2CFR, for the purpose of differentiation, $\hat{V}(I)$ is used to represent the counterfactual value here, and $\hat{V}(I) = V_{NN}(I) + \beta V_{MC}(I)$. $V_{NN}(I)$ is from the DNet and $V_{MC}(I)$ is from the MC simulation in the rectification module. $\alpha + \beta = 1$, $\alpha \geq 0$, $\beta \geq 0$, $\alpha = 0.01 + t/(t + 1)$, $\beta = 0.99 - t/(t + 1)$, and the detailed descriptions can be seen in Section III-B2. According to the setting of $\alpha$ and $\beta$, $\beta = 0$ and $\alpha = 1$ when iteration $t \geq 99$. At this time, $\hat{V}(I) = \alpha V_{NN}(I) + \beta V_{MC}(I) = V_{NN}(I)$, $t \geq 99$. In this case, the instant regret of D2CFR is the same as the instant regret (called advantage value) in DeepCFR, both of which are completely estimated by their respective neural networks. It can be directly concluded that the regret boundary of D2CFR is consistent with that of DeepCFR (when $t \geq 99$).

From a practical perspective, the total iteration number $T$ during usage is in the hundreds and thousands, which is far greater than 99 ($T = 1000$ in D2CFR and $T$ is more than 500 in Libratus [11]). In this case, $R_t^2$ is definitely bounded. Moreover, in theoretical proof in (63) in the Supplementary Material, it can be verified that $R_t^2$ has an upper bound when the total iteration number approaches infinity.

Of course, it can be analyzed from a theoretical perspective when the total iteration $1 \leq T \leq 98$. For $\hat{V}(I)$

$$
\hat{V}(I) = \alpha V_{NN}(I) + \beta V_{MC}(I)
= (1 - \beta) V_{NN}(I) + \beta V_{MC}(I) \leftarrow \alpha + \beta = 1
\geq (1 - \beta) V_{NN}(I), \quad \beta \geq 0, \quad V_{MC}(I) \geq 0.
\text{(28)}
$$

Here, it is first demonstrated that $V_{MC}(I)$ itself is a finite value. In D2CFR, the MC simulation is added to the rectification module to correct the counterfactual value estimated by the neural network in the early iteration stage, so as to reduce the estimation deviation. Based on this, the average value of MC simulation after 800 simulations is introduced as the final $V_{MC}$ value (the MC simulation is from the current node to the end of the leaf node). $V_{MC}(I)$ for the player can be described as follows:

$$
V_{MC}(I) = \frac{1}{N} \sum_{n=1}^{N} V_{MC}^{n}(I) = \frac{1}{N} \sum_{n=1}^{N} \pi_{MC}^{n}(a, z) \times u(z)
\text{(29)}
$$

where $N$ is the total time of MC simulation and $N = 800$ in this article, $u(z)$ is the payoff of the leaf node $z$ and $\pi_{MC}^{n}(a, z)$ is the probability with $z$ that occurs if all players make a decision according to the MC simulation strategy, $\pi_{MC}^{n}(a, z) \leq 1$. In the simulation process, the main factor that affects the $V_{MC}$ value is the payoff $u(z)$ of the leaf node. For the game in this article, their payoff in the leaf node is a finite constant. To sum up, the $V_{MC}$ value is a finite constant $u(z)$ multiplied by a number $\pi_{MC}^{n}(a, z)$ less than or equal to 1, which is still a finite constant. Then,

$$
V(I) = \alpha V_{NN}(I) + \beta V_{MC}(I)
< V_{NN}(I) + V_{MC}(I), \quad \leftarrow \alpha + \beta = 1, \quad \beta \geq 0, \quad \alpha \geq 0
\text{(30)}
$$

because the value of $V_{NN}(I)$ and $V_{MC}(I)$ both is bounded on each iteration; for $g \in R$, there must be a number $g$, making the condition $V_{NN}(I) + V_{MC}(I) < g$ hold. Based on the above analysis, it can be concluded that $\hat{V}(I)$ has the upper and lower bounds, $(1 - \beta) V_{NN}(I) \leq \hat{V}(I) < g$. Furthermore, $\|\hat{r}(I) - r(I|\theta^s)\|_2^2$ is bounded, and thus, $\mathcal{L}_V$ is bounded. At present, the conclusion that $R_t^2$ is bounded still holds.

ACKNOWLEDGMENT

The computing resources of Pengcheng Cloud Brain are used in this research.

REFERENCES

[1] A. L. Samuel, “Some studies in machine learning using the game of checkers,” IBM J. Res. Develop., vol. 44, no. 1, pp. 206–226, Jan. 2000.
[2] R. B. Myerson, Game Theory: Analysis of Conflict. Cambridge, U.K.: Cambridge Univ. Press, 1997.
[3] Z. Deng and X. Nian, “Distributed generalized Nash equilibrium seeking algorithm design for aggregative games over weight-balanced digraphs,” IEEE Trans. Neural Netw. Learn. Syst., vol. 30, no. 3, pp. 695–706, Mar. 2019.
[4] P. Zhang, Y. Yuan, H. Yang, and H. Liu, “Near-Nash equilibrium control strategy for discrete-time nonlinear systems with round-robin protocol,” IEEE Trans. Neural Netw. Learn. Syst., vol. 30, no. 8, pp. 2478–2492, Aug. 2019.
[5] M. Li, J. Qin, Q. Ma, W. X. Zheng, and Y. Kang, “Hierarchical optimal synchronization for linear systems via reinforcement learning: A Stackelberg–Nash game perspective,” IEEE Trans. Neural Netw. Learn. Syst., vol. 32, no. 4, pp. 1600–1611, Apr. 2021.
[6] M. J. Osborne and A. Rubinstein, A Course in Game Theory. Cambridge, MA: MIT Press, 1994.
[7] J. Nash, “Non-cooperative games,” Ann. Math., vol. 10, pp. 286–295, Sep. 1951.
[8] M. Zinkevich, M. Johanson, M. Bowling, and C. Piccione, “Regret minimization in games with incomplete information,” in Proc. Adv. Neural Inf. Process. Syst., 2008, pp. 1729–1736.
[9] M. Bowling, N. Burch, M. Johanson, and O. Tammelin, “Heads-up limit hold’em poker is solved,” Science, vol. 347, no. 6218, pp. 145–149, 2015.

[10] M. Moravcik et al., “DeepStack: Expert-level artificial intelligence in heads-up no-limit poker,” Science, vol. 356, no. 6357, pp. 508–513, May 2017.

[11] N. Brown and T. Sandholm, “Superhuman ai for heads-up no-limit poker: Libratus beats top professionals,” Science, vol. 359, no. 6374, p. 1733, 2017.

[12] M. Schmid, N. Burch, M. Lanctot, Moravcik, R. Kadlec, and M. Bowling, “Variance reduction in Monte Carlo counterfactual regret minimization (VR-MCCFR) for extensive form games using baselines,” in Proc. AAAI Conf. Artif. Intell., vol. 33, 2019, pp. 2157–2164.

[13] N. Brown and T. Sandholm, “Superhuman AI for multiplayer poker,” Nature, vol. 565, pp. 885–890, Aug. 2019.

[14] A. Krith and D. Ahner, “Counterfactual regret minimization for integrated cyber and air defense resource allocation,” Eur. J. Oper. Res., vol. 292, no. 1, pp. 95–107, Jul. 2021.

[15] J. Zhang, K. Li, B. Zhang, M. Xu, and C. Wang, “Parallel counterfactual regret minimization in crowdsourcing imperfect-information expanded game,” in Proc. IEEE Int. Conf. Parallel Distrib. Process. Appl., Big Data Cloud Comput., Sustain. Comput. Commun., Social Comput. Netw., Proc. 34th AAAI Conf. Artif. Intell., vol. 20, no. 1, pp. 111–115, 2020.

[16] M. Rehák, J. Stiborek, and M. Grill, “Intelligence, not integration: Deep reinforcement learning and its contribution to IDS control,” in Proc. 6th Int. Joint Conf. Auto. Agents Multiagent Syst., May 2007, pp. 1–8.

[17] S. Ganzfried and T. Sandholm, “Action translation in extensive-form games with large action spaces: Axioms, paradoxes, and the pseudo-harmonic mapping,” in Proc. Workshops 27th AAAI Conf. Artif. Intell., 2013, pp. 1–20.

[18] E. Zio, Monte Carlo Simulation: The Method, Cham, Switzerland: Springer, 2013.

[19] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in Proc. Int. Conf. Mach. Learn., 2015, pp. 159–167.

[20] D. Morrill, “Using regret estimation to solve games compactly,” in Proc. 21st Conf. Uncertainty Artif. Intell., 2005, pp. 106–117.

[21] J. Shi and M. L. Littman, “Abstraction methods for game theoretic models,” in Proc. 35th Conf. Uncertainty Artif. Intell., 2019, pp. 53–62.

[22] B. Noam and T. Sandholm, “Safe and nested subgame solving for imperfect-information games,” in Proc. AAAI Conf. Artif. Intell., 2017, pp. 295–303.

[23] B. Noam and T. Sandholm, “Safe and nested subgame solving for imperfect-information games,” in Proc. AAAI Conf. Artif. Intell., 2017, pp. 295–303.

[24] D. P. Foster and R. Vohra, “Regret in the on-line decision problem,” Games Econ. Behav., vol. 29, nos. 1–2, pp. 7–35, Oct. 1999.

[25] M. Lancot, K. Waugh, M. Zinkevich, and M. Bowling, “Monte Carlo sampling for regret minimization in extensive games,” in Proc. Adv. Neural Inf. Process. Syst., 2009, pp. 1078–1086.

[26] J. Shi and M. L. Littman, “Abstraction methods for game theoretic poker,” in Proc. Int. Conf. Comput. Games, Cham, Switzerland: Springer, 2000, pp. 333–345.

[27] A. Gilpin, T. Sandholm, and T. B. Sørensen, “Potential-aware automated abstraction of sequential games, and holistic equilibrium analysis of Texas hold’em poker,” in Proc. Nat. Conf. Artif. Intell., vol. 22, no. 1, Menlo Park, CA, USA: MIT Press, 2007, p. 50.

[28] A. Gilpin and T. Sandholm, “Better automated abstraction techniques for imperfect information games, with application to Texas Hold’em poker,” in Proc. 6th Int. Joint Conf. Auto. Agents Multiagent Syst., May 2007, pp. 1–8.

[29] S. Ganzfried and T. Sandholm, “Action translation in extensive-form games with large action spaces: Axioms, paradoxes, and the pseudo-harmonic mapping,” in Proc. Workshops 27th AAAI Conf. Artif. Intell., 2013, pp. 1–20.

[30] E. Zio, Monte Carlo Simulation: The Method, Cham, Switzerland: Springer, 2013.

[31] M. D. Schluchter, “Mean square error,” in Encyclopedia of Biostatistics, vol. 5. Hoboken, NJ, USA: Wiley, 2005.

[32] F. Soutey et al., “Bayes’ bluff: Opponent modelling in poker,” in Proc. 21st Conf. Uncertainty Artif. Intell., 2005, pp. 550–558.

[33] S. Ganzfried, “Computing strong game-theoretic strategies and exploiting suboptimal opponents in large games,” Ph.D. dissertation, IBM, Armonk, NY, USA, 2015.

[34] Wikipedia Contributors. (2023). Texas Hold’EM—Wikipedia, the Free Encyclopedia. Accessed: Aug. 8, 2023. [Online]. Available: https://en.wikipedia.org/wiki/index.php?title=Texas_hold%27em&oldid=1166652747

[35] M. Lancot et al., “OpenSpiel: A framework for reinforcement learning in games,” 2019, arXiv:1908.09453.

[36] D. P. Kingma and J. Ba. “Adam: A method for stochastic optimization,” 2014, arXiv:1412.6980.

[37] J. C. Harsanyi, “Games with incomplete information,” Amer. Econ. Rev., vol. 85, no. 3, pp. 291–303, 1995.

[38] W. Plesniak, “Markov’s inequality and the existence of an extension operator for C0 functions,” J. Approximation Theory, vol. 61, no. 1, pp. 106–117, Apr. 1990.

[39] D. Morrill, “Using regret estimation to solve games compactly,” M.S. thesis, Dept. Comput. Sci., Univ. Alberta, Edmonton, AB, Canada, 2016.

[40] J. J. Ruel and M. P. Ayres, “Jensen’s inequality predicts effects of environmental variation,” Trends Ecol. Evol., vol. 14, no. 9, pp. 361–366, Sep. 1999.

Huale Li received the master’s degree in circuits and systems from Lanzhou University, Lanzhou, China, in 2017, and the Ph.D. degree in computer science from the Harbin Institute of Technology, Shenzhen, China, in 2022. He is currently an Associate Professor with the School of Software, Northwestern Polytechnical University, Xi’an, China. His research interests include computer games, reinforcement learning, and machine learning.
Xuan Wang (Senior Member, IEEE) is currently a Professor and a Ph.D. Supervisor with the Harbin Institute of Technology, Shenzhen, China. He is also the Director of the Computer Application Research Center and the Dean of the Guangdong Provincial Key Laboratory of Novel Security Intelligence Technologies, Shenzhen. His research interests include artificial intelligence, computer games, and machine learning.

Zengyue Guo is currently pursuing the master’s degree in computer science with the Harbin Institute of Technology, Shenzhen, China. His research interests include computer games, reinforcement learning, and machine learning.

Jiajia Zhang received the M.S. and Ph.D. degrees in computer sciences from the Harbin Institute of Technology, Harbin, China, in 2009 and 2015, respectively.

He held a post-doctoral position at Peking University, Beijing, China, from 2015 to 2018. He is currently the Deputy Director of the Institute of Decision Intelligence, Harbin Institute of Technology, and the Vice Director of the HIT (Shenzhen)-Pingan Joint Research Center, Shenzhen. He is also an Associate Research Fellow with the Harbin Institute of Technology, Shenzhen, China. His main research interests include artificial intelligence, computer games, intelligence strategy decision, and cyberspace security.

Shuhan Qi received the M.S. and Ph.D. degrees in computer sciences from the Harbin Institute of Technology, Harbin, China, in 2011 and 2017, respectively.

He was a Visiting Scholar at the National University of Singapore, Singapore, from 2013 to 2016, advised by Prof. Chua Tat-Seng. He is currently an Assistant Professor in computer sciences with the Harbin Institute of Technology, Shenzhen, China. His research interests include multimedia pattern recognition and computer games.