Title: Combining CFR-based algorithms with deep reinforcement learning techniques (RLCCFR)

First Author: Behbod Keshavarzi

- Affiliation: [Shahed university, Mathematics]
- Email: Behbod.Keshavarzi@yahoo.com

Second Author: Hamidreza Navidi

- Affiliation: [Shahed university, Mathematics]
- Email: navidi@shahed.ac.ir

Three Author: Mojtaba Tefagh

- Affiliation: [Sharif university of technology, Computer science]
- Email: mtefagh@sharif.edu

Preprint not peer reviewed

This preprint research paper has not been peer reviewed. Electronic copy available at: https://ssrn.com/abstract=4806706

Combining CFR-based algorithms with deep reinforcement learning techniques (RLCCFR)

Behbod Keshavarzi[a,1], Hamidreza Navidi[a,*], Mojtaba Tefagh[b]

[a]*Iran, Tehran, Shahed University*
[b]*Iran, Tehran, Sharif University of Technology*

Abstract

Finding a Nash equilibrium in a two-player game is a common approach to solving two-agent decision-making issues in general. In a two-player, zero-sum game with imperfect information, counterfactual regret minimization (CFR) is a popular method for determining a Nash equilibrium strategy. Over the past few years, several CFR variations have emerged, such as DCFR, LCFR, DeepCFR, ECFR, RCFR, among others. The general classification of algorithms falls into three categories: 1. strategy updating, regret calculation, and table CFR-based algorithms; 2. sampling CFR-based algorithms; and 3. abstraction-based CFR-based algorithms. In this paper, the methods in each category can be put together with methods from other categories to make a novel algorithm. We aim to improve convergence and speed by considering two criteria: exploitability and speed-to-iteration ratio in this new algorithm. This process utilizes the DQN algorithm. This method is a combination of CFR and reinforcement learning, called reinforcement learning combined CFR (RLCCFR).

Keywords: Counterfactual regret minimization, Zero-sum games, Reinforcement learning, imperfect information

*Hamidreza Navidi
Email addresses: Behbod.keshavarzi@yahoo.com (Behbod Keshavarzi), navidi@shahed.ac.ir (Hamidreza Navidi), mtefagh@sharif.edu (Mojtaba Tefagh)

Preprint not peer reviewed

This preprint research paper has not been peer reviewed. Electronic copy available at: https://ssrn.com/abstract=4806706

1. Introduction

There are two types of games: perfect information games (PIGs) and imperfect information games (IIGs), depending on whether the player has full knowledge of the game state of all the other players. A game with imperfect information can be used to model strategic interactions involving multiple agents with incomplete information. Every player can see every piece on the board at all times in chess, which is a game with perfect information. Checkers, Go, Reversi, and tic-tac-toe are other games with perfect information. Games with imperfect information, such as poker and bridge, are examples.

Counterfactual regret minimization, often known as CFR, is a common approach to calculating the Nash equilibrium strategy [1]. A Nash equilibrium is eventually reached by applying CFR to two-player zero-sum games. CFRs have been established in a wide variety over the past few years [1, 2, 3, 4, 5]. Each repetition of vanilla CFR adds equally to the total regret in the table. But it is possible that certain iterations most notably the earlier ones contribute to far less accurate regret estimates [1]. A modification of regret matching known as regret matching+ is used by CFR^+. In this version of regret matching, regrets are required to be positive [5].

The development of Monte Carlo CFR (MCCFR) allowed sampling techniques to be introduced that greatly reduced the size of the game tree's traveled region, making CFR approaches much more broadly adaptable to various tree types and sizes [2]. The authors of this work provided three novel CFR versions that sampled less frequently than the usual method. The researchers demonstrated that public chance sampling converges faster than chance sampling on big games, resulting in a more efficient method for approximating equilibrium [3].

Discounted CFR works by assigning a weight to each iteration, and the impact of that iteration on the cumulative regret updates is proportional to the weight [4]. The linear CFR is one of the most famous and successful types of reduced CFR. Linear CFR increases the weight each iteration gets as train-

Preprint not peer reviewed

This preprint research paper has not been peer reviewed. Electronic copy available at: https://ssrn.com/abstract=4806706

ing progresses. In other words, at iteration t, cumulative regret changes are weighted by the multiplicative factor t [6]. In deep counterfactual regret minimization, deep neural networks are used instead of abstractions to approximate CFR behavior in full games [6]. Instead of using domain-specific abstractions, it uses two connected deep neural networks to learn strategy profiles directly from the data. They have also developed novel sampling techniques for IIGs that reduce variance and memory consumption. Their double neural counterfactual regret minimization (DNCFR) strategy was validated by experimental results [7]. Monte Carlo counterfactual regret minimization can now be performed with reduced variance. State-action baselines appear to be more accurate than state baselines in imperfect information games, indicating that this method is similar to existing RL methods of state and state-action baselines [8].

The exponential CFR (ECFR) speeds up the CFR by focusing on the most beneficial acts. The regret value of an action indicates how much better it is than the predicted action in the current plan [9]. RLCFR is a system that brings together the RL and the CFR. A Markov decision process is used to describe the dynamic process of iterative interactive strategy update in the framework [10]. In autoCFR, users learn how to create new versions of CFR. By using a formal language, CFR-type algorithms can be represented as computational graphs. A regularized evolution method is then used to search through the space of computational graphs quickly [11]. Games of partial knowledge are prevalent in a variety of leisure activities and real-world games. The authors discuss significant achievements in the area and show that abstraction has become a key facilitator for solving large incomplete-information games. Review the current justifications for abstracting games, highlighting the problem of pathology in abstraction as well as the use-able methods for action and information abstraction [12].

Using CFR-BR, a game solving algorithm that converges to one of these least exploitable abstract strategies without causing a high memory cost that previously rendered such a solution intractable, the effectiveness of this strategy was also demonstrated in the context of two-player limit Texas hold'em. With

Preprint not peer reviewed

This preprint research paper has not been peer reviewed. Electronic copy available at: https://ssrn.com/abstract=4806706

CFR-BR, it is possible to produce much closer approximations to the unknown, optimal Nash equilibrium strategy within an abstraction than is possible with previous state-of-the-art techniques [13]. Previous works on abstraction in big games can be seen as a principled generalization of this method. In [14], abstraction and equilibrium are both learned through self-playalization of this method. Demonstrate that the method produces better tactics than the best modeling techniques, given the same amount of resources [14]. In this paper, we describe the different types of CFR algorithms in order to develop an algorithm that is suitable for each environment. The next step is to discuss the structure of the reinforcement learning algorithm. As a final step, a comparison of the proposed algorithm with other algorithms is discussed and the results are tested.

2. Methodology

The first step of this section is to analyze and classify the types of CFR algorithms and describe their convergence theorems, followed by the introduction of reinforcement learning algorithms as a way of combining the algorithms.

Table 1: Variable definitions

Variable	**Definition**
$\mathcal{I}_i$	The information set of player i.
T	Iteration time.
Z	Terminal states.
Δ	The range of utilities available to the player i. $\Delta_{u,i} = maxu_i(Z) - minu_i(Z)$
σ	The strategy where σ_i is the strategy of player i, σ_{-i} is the strategy of the other player.
$\pi^{\sigma}(h)$	If all players choose actions based on σ, history h may occur.
u	In this utility function, the utility of player i is represented by u_i.
$A(I)$	Sets of actions.
$P(h)$	The player who takes an action after the history h.
M_i	$\sqrt{\lvert\mathcal{I}_i\rvert} \leq M_i \leq \mathcal{I}_i$.

Preprint not peer reviewed

This preprint research paper has not been peer reviewed. Electronic copy available at: https://ssrn.com/abstract=4806706

2.1. Strategy updating, regret calculation, and table CFR-based algorithms

There have been many variations of CFR variants that have been developed since vanilla CFR was first made, for a variety of reasons [15]. There has been a significant increase in the original convergence rate of CFR as a result of this change [1]. CFR involves regret computation and matching. First, the regret values of each action are proposed. A new global strategy will be updated as the final step. CFR^+ is like CFR, but it has two small but significant changes that make it work better [1]. It also converges orders of magnitude faster than CFR. $R_i^T(I,a) = max(R_i^T(I,a), 0)$, then the cumulative regrets to obtain the new strategy:

$$\sigma_i^{T+1}(I)(a) = \begin{cases} \frac{R_i^{T,+}(I,a)}{\sum_{a\in A(I)} R_i^{T,+}(I,a)} & \sum_{a\in A(I)} R_i^{T,+}(I,a) > 0 \\ \frac{1}{A(I)} & \text{otherwise} \end{cases} \tag{1}$$

Theorem 1. *[1] In a zero-sum game at time T, if both players' average overall regret is less than ϵ, then σ^T is a 2ϵ equilibrium.*

Theorem 2. *[1] If player i selects actions according to Equation 1 then $R_{i,imm}^T(I) \leq \Delta_{u,i}\sqrt{|A|}/\sqrt{T}$ and consequently $R_i^T(I) \leq \Delta_{u,i}|\mathcal{I}_i|\sqrt{|A|}/\sqrt{T}$ where $|A_i| = max_{h:P(h)=i}|A(h)|$.*

First, instead of waiting for the cumulative regret to turn positive before using an action again, CFR^+ resets any action with a negative cumulative regret to zero at each iteration. This makes it possible to use an action again when it looks like it works well. Second, CFR^+ doesn't use a uniform weighted averaging strategy like CFR. Instead, it uses a weighted averaging strategy where T iterations are weighted. Linear CFR (LCFR) is similar to CFR, it gives the changes to the average and regret strategy weight on iteration t In other words, the iterates are linearly weighted [6]. On each repetition, one might essentially increase the total regret by $\frac{t}{t+1}$. $DCFR_{\alpha,\beta,\gamma}$ calculated by dividing the total number of positive regrets by $\frac{t^\alpha}{t^\alpha+1}$ and negative regrets by $\frac{t^\beta}{t^\beta+1}$, and $(\frac{t}{t+1})^\gamma$ on each iteration t

Preprint not peer reviewed

This preprint research paper has not been peer reviewed. Electronic copy available at: https://ssrn.com/abstract=4806706

contributions to the average strategy. Also LCFR is equivalent to $DCFR_{1,1,1}$ [4].

Theorem 3. *[4] Assume that T iterations of DCFR are conducted in a two-player zero-sum game. Then the weighted average strategy profile is a $6\Delta|\mathcal{I}|(\sqrt{|A|}+\frac{1}{\sqrt{T}})/\sqrt{T}$-Nash equilibrium.*

exponential counterfactual regret minimization (ECFR), During the iteration process, a reweighting of the instantaneous regret value is accomplished through the utilization of an exponential weighting technique. ECFR determines the weight of action a by $w(I,a) = exp(r_i(I,a) - \frac{1}{|A(I)|}\sum_{a\in A(I)} r_i(I,a)$ [9].

Theorem 4. *[9] Assume that the number of iterations is T and that ECFR is conducted as a two-player zero-sum game. Then the weighted average strategy profile is a $\Delta|\mathcal{I}|(\sqrt{|A|}e^{2T-T^2})/\sqrt{T}$ Nash equilibrium.*

Deep counterfactual regret minimization is a type of CFR that eliminates the requirement for abstraction by making use of deep neural networks to approximate the behavior of CFR while it is being applied to the full game [7]. This form of CFR is also known as deep counterfactual regret minimization. With this version of CFR, large games can be played successfully without using tables. In deep CFR, function approximations are made using deep neural networks to approximate the behavior of CFR without computing and collecting regrets at each infoset. The external sampling MCCFR is used to determine the route of each partial traversal of the game tree performed by Deep CFR on each iteration t [6].

2.2. Sampling CFR-based Algorithms

The challenge of developing more general CFR strategies for imperfect information games is finding procedures that converge to equilibrium for games with unboundedly vast trees. A variety of methods can be used for sampling. Therefore, sampling methods are used to solve extensive games with many dimensions. Chance-sampled CFR considers only one possible outcome per traversal,

Preprint not peer reviewed

This preprint research paper has not been peer reviewed. Electronic copy available at: https://ssrn.com/abstract=4806706

resulting in more rapid but less accurate iterations. CFR with chance sampling requires more iterations, but it converges faster in the end [3]. Monte Carlo CFR (MCCFR) introduced sampling methods that significantly reduced traversed areas in game trees [2].

The outcome sampling technique produces a single terminal history by only sampling one action per information set encountered during each iteration of the process. For each sampled information set or action pair, the tabular cumulative regrets will be updated [2, 15].

Theorem 5. *[2, 15] For any $p \in (0, 1]$, when using outcome-sampling MCCFR where $\forall z \in Z$ either $\pi^{\sigma}_{-i}(z) = 0$ or $q(z) \geq \delta > 0$ at every timestep, with probability at least $1 - p$, average overall regret is bounded by, $R_i^T \leq (1 + \frac{\sqrt{2}}{\sqrt{p}})(\frac{1}{\delta})\Delta_{u,i}M_i\sqrt{|A_i|}/\sqrt{T}$.*

External sampling is the most popular MCCFR variant today, combining outcome sampling and vanilla CFR. In all game states where the regret-updating player is active, all actions are traversed in External sampling. An action is sampled, however, according to its outcome at all other nodes (the opponent and chance nodes) [2, 15].

Theorem 6. *[2] For any $p \in (0, 1]$, when using external-sampling MCCFR, with probability at least $1 - p$, average overall regret is bounded by, $R_i^T \leq (1 + \frac{\sqrt{2}}{\sqrt{p}})\Delta_{u,i}M_i\sqrt{|A_i|}/\sqrt{T}$.*

Sampling with probing: using this method, any generic estimator of the desired values can be used to generalize MCCFR. It is possible to minimize regret probabilistically by selecting an estimator that is unbiased and bound. Calculate regret from the estimate's variance and the algorithm's convergence rate [16]. Estimated counterfactual regret on iteration t for action a at I to be: $\hat{r}_i^t = \hat{v}_i(\sigma^t_{(I\to a)}, I) - \hat{v}_i(\sigma^t, I)$.

Theorem 7. *[16] Let $p \in (0, 1]$ and suppose that there exists a bound $\hat{\Delta}_i$ on the difference between any two estimates, $\hat{v}_i(\sigma^t_{(I\to a)}, I) - \hat{v}_i(\sigma^t, I) \leq \hat{\Delta}_i$. If strategies are selected according to regret matching on the estimated counterfactual*

Preprint not peer reviewed

This preprint research paper has not been peer reviewed. Electronic copy available at: https://ssrn.com/abstract=4806706

regrets, then with probability at least $1-p$*, the average regret is bounded by* $\frac{R_i^T}{T} \leq |\mathcal{I}|\sqrt{|A_i|}(\frac{\hat{\Delta}_i}{\sqrt{T}} + \sqrt{\frac{var}{pT} + \frac{cov}{p} + \frac{E^2}{p}})$ *where* $Var = max_{t\in 1,\ldots,t, I\in\mathcal{I}, a\in A(a)} = Var[r_i^t(I,a) - \hat{r}_i^t(I,a)]$*, with Cov and E similarly defined.*

Average Strategy Sampling (AS), This technique chooses a player's actions based on three predetermined parameters and the cumulative profile. At each information set I, a subset of player i's actions are sampled instead of just one (OS) or every action (ES), which is comparable to OS and ES. The actions of player i, $a \in A(I)$, are each randomly selected with probability $\rho(a,I) = max\{\epsilon, \frac{\beta+\tau s_i^T(I,a)}{\beta+\sum_{b\in A(I)} s_i^T(I,b)}\}$, $s_i^T(I,.)$ is the cumulative profile on iteration T, $\epsilon \in (0,1]$ is exploration parameter, $\tau \in [1,\infty)$ is threshold parameter and $\beta \in [0,\infty)$ is bonus parameter [17].

Theorem 8. *[17] Let X be one of CS, ES, or OS, let* $p \in (0,1]$*, and let* $\delta = min_{z\in Z} q_i(z) > 0$ *over all* $1 \leq t \leq T$*. When using X, with probability* $1-p$*, average regret is bounded by* $\frac{R_i^T}{T} \leq (M_i(\sigma_i^*) + \frac{\sqrt{2|\mathcal{I}_i||\mathcal{B}_i|}}{\sqrt{p}})(\frac{1}{\delta})\Delta_i\sqrt{|A_i|}/\sqrt{T}$*.*

Variance Reduction in Monte Carlo Counterfactual Regret Minimization (VR-MCCFR), Every iteration, estimated values and updates are reformulated as functions of sampled values and state-action baselines, similar to their use in policy gradient reinforcement learning. As a result of this formulation, estimates can be bootstrapped based on other estimates within the same episode, propagating the advantages of baselines along the sampled trajectory. In addition to being a precise baseline, the variance of the estimated values can be reduced to zero. baseline-enhanced estimate is $\hat{v}_i^b(\sigma,I,a) = \hat{v}_i(\sigma,I,a) - \hat{b}_i(\sigma,I,a) + b_i(\sigma,I,a)$, estimated counterfactual value is $\hat{v}_i$, control variate is $\hat{b}_i$ and the action-dependent baseline is b_i [8, 18].

2.3. Abstraction-based CFR-based Algorithms

Game abstraction entails restricting players' strategy spaces as a mode of abstraction. The most popular form of game abstraction is information abstraction, in which information states are grouped together. Another type of game abstraction is action abstraction, in which some actions are presumed to be

Preprint not peer reviewed

This preprint research paper has not been peer reviewed. Electronic copy available at: https://ssrn.com/abstract=4806706

useless. In such situations, lossy state-space abstraction can be used to create a smaller, more tractable abstract game while hoping that the equilibrium of the resultant abstract game is similar to the equilibrium strategy of the unabstracted game. As a result, CFR-BR avoids storing the opponent's entire strategy explicitly, which would have required enormous memory [13].

Theorem 9. *[13] After T iterations of CFR-BR, $\bar{\sigma}_1^T$ is player 1s part of an $\epsilon - Nash$ equilibrium, with $\epsilon = \frac{\Delta|\mathcal{I}_1|\sqrt{|A_1|}}{\sqrt{T}}$.*

Regression CFR (RCFR): CFR is calculated for each information set using estimations of counterfactual regret derived from a single common regressor shared by all information sets. A common regressor employs characteristics determined by the information-set/action combination, $\varphi(I, a)$. This enables the regression to generalize across comparable acts and contexts when calculating regret estimates [14]. The action space in such situations has previously been reduced using unsupervised clustering techniques applied to actions in feature space. In this algorithm, use this feature vector to retain structure during learning. Each action can be reduced to the standard framework by using an indicator feature. Each action $a \in A$ can be described by a feature vector, $\varphi(a) \in \chi$. The training approach, in particular, is predicted to yield f where $f(\varphi(a)) \approx R^t(a)/t$ or, $\tilde{R}^t(a) = tf(\varphi(a))$. The regressor's error in relation to the l_2 distance between the approximation and true cumulative regret vectors: $||R^t - \tilde{R}^t||_2$. This kind of abstraction provides a function f for translating whole game information sets into abstract game information sets. For this abstraction to be generated, some form of clustering, such as k-means, and some concept of similarity and compatibility between information sets are required. A function $\varphi : I \times A \longrightarrow \chi$ domain-specific features are mapped to information-set/action pairs χ. Consider a single iteration of the counterfactual regret update in both the original game and the abstract game where the players' tactics are fixed in order to compare the approach to RCFR [14]. $\tilde{r}_i(a|I^{abstract}) = \sum_{I^{full} \in f^{-1}(I^{abstract})} r_i^t(a|I^{full})$, $I^{abstract}$ is information set in the abstract game, the regret at information set in the abstract game is $\tilde{r}_i(.|I^{abstract})$.

Preprint not peer reviewed

This preprint research paper has not been peer reviewed. Electronic copy available at: https://ssrn.com/abstract=4806706

Theorem 10. *[14] If for each* $t \in [T]$, $||R^t - \tilde{R}^t||_2 < \epsilon$, *then regression regret-matching has regret bounded by* $\sqrt{TNL + 2T\sqrt{L\epsilon}}$.

Theorem 11. *[14] If for each* $t \in [T]$ *at every information set* $I \in \mathcal{I}_i$, $||R^t(.|I) - \tilde{R}^t(.|I)||_2 < \epsilon$, *then RCFR has external regret bounded by* $|\mathcal{I}_i|\sqrt{TNL + 2T\sqrt{L\epsilon}}$.

2.4. proof of convergence

Since the CFR algorithm is proved with different combinations in theorems 1 to 11, it is evident that this algorithm will converge in all cases. The CFR^+ algorithm is also very similar to the CFR algorithm, with a very small difference, so the combination of these algorithms will also converge. In the scenario of sampling and abstraction, the LCFR, DCFR, ECFR, and other forms of deep CFR have not been demonstrated to converge at the same rate. Figures 2, 3, and 4 show that the convergence is slightly faster in practice.

2.5. Deep RL Algorithm

In single-objective reinforcement learning, an agent interacts with the environment by recognizing its state $s_t \in S$ and playing an action $a_t \in A$ for each step t. Agents choose actions based on policies π. The agent receives a reward r for performing an action [19]. In the next state, the agent observes s_{t+1} and the process repeats. The agent's goal is to maximize the expected reward. $R_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k}$, $\gamma^k \in [0, 1]$ is the discount factor. Actions are selected in Q-learning based on $Q(s, a)$, which represents the expected discounted reward for performing action a in state s. For the given state s, $a = argmax_a Q(s, a)$ is the optimal action. Deep Q-learning uses deep neural networks to approximate $Q(s, a)$ values, as a result, many real-world applications can be met by this method. Deep Q Network (DQN) is one of the most important deep reinforcement learning (DRL) methods. The DQN method was initially suggested in 2013 by DeepMind researchers in the article Playing Atari with Deep Reinforcement Learning. For the regression task, the mean squared error (MSE) is often used as the loss function. $MSE = \frac{1}{K}\sum_{i=1}^{K}(y_i - \hat{y}_i)^2$, y is the goal value,

Preprint not peer reviewed

This preprint research paper has not been peer reviewed. Electronic copy available at: https://ssrn.com/abstract=4806706

$\hat{y}$ is the predicted value, and the number of training samples is K [20]. That the Bellman optimality equation can be used to get the optimal Q value:

$$Q^*(s,a) = E_{s' \sim p}[r + \gamma max_{a'} Q^*(s^{'}, a^{'})] \tag{2}$$

Due to the loss function, describe the loss function L as the difference between the goal value and the predicted value on DQN [20].

$$\mathcal{L}(\theta) = Q^*(s,a) - Q_\theta(s,a) \tag{3}$$

Due to the loss function can be expressed as [20]:

$$\mathcal{L}(\theta) = \frac{1}{K}\sum_{i=1}^{K}(y_i - Q_\theta(s_i, a_i))^2 \tag{4}$$

where $y_i = r_i + \gamma max_{a'} Q_\theta(s^{'}, a^{'})$

The reward will be the target value, as shown here:

$$y_i = \begin{cases} r_i & \text{if } s^{'} \text{ is terminal} \\ r_i + \gamma max_{a'} Q_\theta(s^{'}, a^{'}) & \text{if } s^{'} \text{ is not terminal} \end{cases} \tag{5}$$

Initialization:

- Initialize replay memory D with capacity N
- Initialize action-value function Q with random weights θ
- Initialize target action-value function Q' with weights $\theta' = \theta$
- Set exploration rate ϵ and its parameters (e.g., start and end values, decay schedule)

Training Loop:

- **For** episode = 1 **to** M **do**:
  - Observe initial state s_1
  - **For** $t = 1$ **to** T **do**:

Preprint not peer reviewed

This preprint research paper has not been peer reviewed. Electronic copy available at: https://ssrn.com/abstract=4806706

* With probability ϵ, select a random action a_t
* Otherwise, select $a_t = \text{argmax}_a Q(s_t, a; \theta)$
* Execute action a_t in the environment, observe reward r_t and next state s_{t+1}
* Store transition (s_t, a_t, r_t, s_{t+1}) in D
* Sample a random minibatch of transitions (s_j, a_j, r_j, s_{j+1}) from D
* Set target for Q-learning:

$$y_j = \begin{cases} r_j & \text{for terminal } s_{j+1} \\ r_j + \gamma \max_{a'} Q'(s_{j+1}, a'; \theta') & \text{for non-terminal } s_{j+1} \end{cases}$$

* Update Q by minimizing the loss:

$$\mathcal{L}(\theta) = \frac{1}{|\text{minibatch}|} \sum_j (Q(s_j, a_j; \theta) - y_j)^2$$

* Every C steps, update the target network Q':

$$\theta' \leftarrow \theta$$

* $s_t \leftarrow s_{t+1}$

2.6. Exploitability and Reward

A strategy σ_i exploitability can be defined as follows: $e_i(\sigma_i) = u_i(\sigma_i^*, \sigma_{-i}^*) - u_i(\sigma_i, BR(\sigma_{-i}))$. ϵ-Nash equilibrium means that no player has exploitability higher than ϵ. $e(\sigma) = \sum_{i \in N} = e_i(\sigma_i)/N$. Another parameter is added here to optimize the algorithm, namely its convergence speed, which is represented by S. As a result, the reward based on exploitability and speed is now defined as follows: $R = 1 - (\alpha e + \beta s)$, so that $\alpha, \beta \in [0, 1]$ and respectively, the weights of exploitability and, speed cannot both be zero at the same time.

2.7. Proposed Algorithm

The goal is to combine the algorithms and present a novel CFR algorithm based on the types of games. The DQN algorithm is used to select the best

Preprint not peer reviewed

This preprint research paper has not been peer reviewed. Electronic copy available at: https://ssrn.com/abstract=4806706

algorithm. According to Figure 1, there are three states: State A is selecting the best algorithm based on the reward received by DQN_A algorithm during iterations and selecting the best strategy update; state B requires choosing the best sampling algorithm for solving games of large dimensions in real world examples, so the DQN_B algorithm selects the optimal algorithm based on the rewards it receives during each iteration. In state C, the sampling algorithm fails to produce acceptable results when the game has very large dimensions. Using the reward that the DQN_C algorithm receives in each iteration, this state selects the best abstraction algorithm. There are three general categories of CFR-based algorithms, as shown in Table 1 and † indicates algorithms presented in this paper.

Table 2: Table CFR-based algorithms: The selection of algorithms is based on their specific properties. Deep-based algorithms are chosen because they differ from current algorithms that are based on a table. However, this restriction may be resolved by using machine learning techniques.

CFR-based algorithms		
Strategy updating, regret calculation, table and without table	Sampling	Abstraction
CFR [1]	Chance sampling [2]	CFR-BR [13]
CFR^+ [1]	MC Outcome Sampling [2]	RCFR [14]
LCFR [6],$LCFR^+$	MC external Sampling [2]	No abstraction
DCFR [4],$DCFR^+$	Sampling with probing [16]	
ECFR [9],$ECFR^+$ †	MC Average Sampling [17]	
DeepCFR [6],$DeepCFR^+$ †	Variance Reduction MC [8]	
DeepLCFR †,$DeepLCFR^+$ †	No sampling	
DeepDCFR †,$DeepDCFR^+$ †		
DeepECFR †,$DeepECFR^+$ †		

It would be better if we used multiple DQNs. The actions that we take can be divided into three categories: $[a_{11}, a_{12}, ..., a_{116}]$, $[a_{21}, a_{22}, ..., a_{27}]$ and

†In this paper

Preprint not peer reviewed

This preprint research paper has not been peer reviewed. Electronic copy available at: https://ssrn.com/abstract=4806706

Table 3: Detail of algorithms

Name Algorithm	Sterategy updating	Regret calculate
CFR	$\sigma_i^{T+1}(I,a) = \frac{R_i^T(I,a)}{\sum_{a\in A(I)} R_i^T(I,a)}$	$R_i^T(I,a) = \sum_{a\in A(I)} r_i^T(I,a)$
CFR^+	$\sigma_i^{T+1}(I,a) = \frac{R_i^{T,+}(I,a)}{\sum_{a\in A(I)} R_i^{T,+}(I,a)}$	$R_i^{T,+}(I,a) = \sum_{a\in A(I)} r_i^{T,+}(I,a)$
LCFR	$\sigma_i^{T+1}(I,a) = \frac{tR_i^T(I,a)}{\sum_{a\in A(I)} tR_i^T(I,a)}$	$R_i^T(I,a) = \sum_{a\in A(I)} tr_i^T(I,a)$
DCFR	$\sigma_i^{T+1}(I,a) = \frac{(\frac{t}{t+1})^\gamma R_i^T(I,a)}{\sum_{a\in A(I)}(\frac{t}{t+1})^\gamma R_i^T(I,a)}$	$R_i^T(I,a) = \sum_{a\in A(I)}(\frac{t}{t+1}) r_i^T(I,a)$
ECFR	$\sigma_i^{T+1}(I,a) = \frac{e^\alpha R_i^T(I,a)}{\sum_{a\in A(I)} R_i^T(I,a)}$	$R_i^T(I,a) = \sum_{a\in A(I)} e^\alpha r_i^T(I,a)$

$[a_{31}, a_{32}, a_{33}]$. Generally, $3\times7\times16 = 336$ algorithms can be evaluated on a game environment and the best algorithm can be selected based on the evaluation.

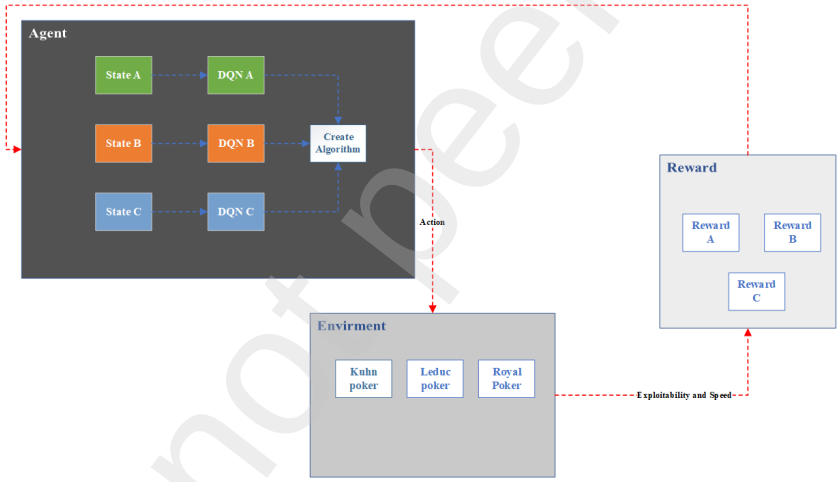

Figure 1: The combined CFR-based algorithms using reinforcement learning are illustrated in the above figure.

[21] Provides a guide for applying multi-objective methods to challenging problems for researchers who have already used single-objective reinforcement learning and planning methods and who wish to adopt a multi-objective perspective on their work. It is intended for researchers who are already familiar with single-objective reinforcement learning and planning methods but wish to adopt a multi-objective perspective on their research. Within [22], the multi-objective node selection issue is expressed using the Markov decision process

Preprint not peer reviewed

This preprint research paper has not been peer reviewed. Electronic copy available at: https://ssrn.com/abstract=4806706

(MDP) with the dual goals of reducing training time and increasing model accuracy. In order to find the best collection of participants for each iteration, deep Q-networks (DQN) are suggested. In the proposed structure of the reinforcement learning algorithm, it consists of three parts: agent, environment, and reward.

1. Agent: There are three categories of DQN algorithms in this part of the algorithm, DQN_A algorithm aims to choose the best algorithm in $[a_{11}, a_{12}, ..., a_{116}]$, DQN_B algorithm aims to choose the best algorithm in $[a_{21}, a_{22}, ..., a_{27}]$ and DQN_C algorithm aims to choose the best algorithm in $[a_{31}, a_{32}, a_{33}]$. The output of this section is action (creat algorithm).
2. Environment: There are all kinds of extensive form games in this section, some of the most famous examples of which can be seen in Figure 1. Also, the output of this section is two values exploitability and speed.
3. Reward: according to the function $R = 1 - (\alpha e + \beta s)$, $\alpha, \beta \in [0, 1]$, DQN inputs can have different values of alpha and beta. The output of this section is R_A, R_B and R_C. ($R_A = 1-(\alpha_A e+\beta_A s)$, so that $\alpha_A, \beta_A \in [0, 1]$. $R_B = 1 - (\alpha_B e + \beta_B s)$, so that $\alpha_B, \beta_B \in [0, 1]$. $R_C = 1 - (\alpha_C e + \beta_C s)$, so that $\alpha_C, \beta_C \in [0, 1]$).

The proposed technique involves considering three games by default. Each game is then abstracted in two different ways, as seen in Table 2. This results in a total of nine games, out of which six are abstracted versions and three are in their original manner. In the initial stage of the algorithm, it is necessary to specify the type of game. All nine games have already been implemented and are provided as input to the algorithm. In the subsequent stage, the type of sampling must be determined based on Table 2. Finally, the type of algorithm

Preprint not peer reviewed

This preprint research paper has not been peer reviewed. Electronic copy available at: https://ssrn.com/abstract=4806706

should be taken into account to update the strategy and regret calculate.

Initialize:

Initialize the game (Kuhn, Leduc, or Royal Poker). Choose between the abstraction game (CFR-BR or RCFR) and the normal game (without abstraction);

Initialize strategies σ^t for all players;

Initialize cumulative regrets R^t for all players;

Initialize cumulative strategy profiles $\bar{\sigma}^t$ for all players;

Set external sampling probability ρ $(0 < \rho \leq 1)$;

for $t = 1$ **to** T **do**

  Simulate the game from the initial state;

  while *current state is not terminal* **do**

    if *current state is a chance node* **then**

      Resolve chance event;

    else

      Select current player;

      if *current player is a real player* **then**

        Sample action according to strategy σ^t;

        if *random_uniform(0, 1)* $< \rho$ **then**

          // External sampling Sample regret from an external state;

        end

        else

          // Internal sampling Sample regret from current state;

        end

        Update cumulative regrets and strategy profiles according to Table 3;

      end

    end

    Move to the next player or state;

  end

  Update strategies based on cumulative regrets;

end

Algorithm 1: External Sampling MCCFR (ES-MCCFR) Combine with every algorithm in Table 2, column 1, 3.

Preprint not peer reviewed

This preprint research paper has not been peer reviewed. Electronic copy available at: https://ssrn.com/abstract=4806706

Algorithm 1 demonstrates an example combination of techniques from columns 1 and 3 in Table 2. Each combination results in a new algorithm that utilizes the DQN algorithm to identify the optimal combination for each game. The implementation of the library includes all algorithm combinations, which are categorized and chosen according to the DQN algorithm.

3. Experiments

Our evaluation is based on two metrics: speed and exploitability. First, we present the experimental setup, which includes games, configuration games, and training details.

3.1. Experimental Setup

We compare our proposed algorithm with existing CFR-based approaches using three poker games: Kuhn, Leduc, Royal, and Goofspiel. The Kuhn Poker deck contains only three cards and gives the player one chance to bet for each player, there are no public cards, and each player has one hand. There are two rounds in Leduc Poker, which is a larger game with a 6-card deck. In the first round, each player has a private hand, and in the second round, each player has a public hand. Three rounds are played in royal poker, and there are eight cards. A private card is issued to each player in the first round, followed by a public card in the second and third rounds. In Goofspiel (x), each player has x cards, and in x rounds, they make secret bids to gain more points. Anaconda-3 Python 3.9 has been used to implement all the implemented codes on the Linux Debian 9.5 operating system and the Acer Aspire 7 laptop (CPU AMD R7, GPU 1060, RAM 16 GB), as well as the open spiel library [23] for comparing algorithms.

3.2. Experimental Result

Eight state-of-the-art methods were used in the first group experiment. Which are CFR , $LCFR$, $ECFR$, $DCFR$, $DeepCFR$, $MCCFR$ and $RLCCFR$ respectively. We selected 1000 and 10,000 iterations for testing, and after 10,000

Preprint not peer reviewed

This preprint research paper has not been peer reviewed. Electronic copy available at: https://ssrn.com/abstract=4806706

iterations, for all methods, there was a very small reduction in exploitability. In order to demonstrate the effectiveness of each method, we limited the testing to 10,000 iterations. Six different experimental approaches are shown in figures 2, 3, 4, and 5. These methods were compared to our RLCCFR method. An advantage of CFR is that regret-minimizing algorithms' averaged strategy profiles tend to be Nash equilibrium-like. Iteration regrets are reweighted differently in LCFR and DCFR, both CFR-based approaches. We select $DCFR_{1.5,0,2}$. As iteration continues, the immediate regret value is reweighted using ECFR, which is an exponential weighting method. Deep CFR introduces a value net, a neural network that approximates $R_t(I, a)$ and the cumulative regret value. A Monte Carlo CFR (MCCFR) was developed to reduce the size of the game tree traversed by introducing sampling methods; we chose external sampling.

According to Fig. 2, the Kuhn poker game evaluates seven algorithms, and the proposed algorithm RLCCFR with selected parameters gives better results. Figures 3, 4, and 5 show that some algorithms have better performance than the proposed algorithm RLCCFR in terms of exploitability, because the speed parameter has gained more weight, but they have worse conditions in terms of speed. According to the selected parameters, the proposed algorithm RLCCFR performs better than other algorithms in terms of convergence speed, as shown in Table 2.

4. Conclusion and Future Work

This work presents a framework for RLCCFR designed as combined CFR variants using DQN and new CFR designs. We developed three main categories of CFR-based algorithms in this work. Several algorithms listed in Table 2 are designed in this paper. A new CFR-based algorithm is constructed utilizing the new reinforcement learning algorithm based on the best category of each algorithm. As shown in Figures 2 through 5 and Table 3, the proposed algorithm competed with the most famous algorithms in this field in four games of convergence and convergence speed. With this method, new algorithms are cre-

Preprint not peer reviewed

This preprint research paper has not been peer reviewed. Electronic copy available at: https://ssrn.com/abstract=4806706

ated that are optimal in terms of speed and approximation of Nash equilibrium. According to the experiments conducted and the results obtained in different environments, our method shows that it is better in terms of convergence and speed of convergence. Future work will include classifying and combining other algorithms used to solve extended games that are not CFR-based, such as fictitious self-play(FSP).

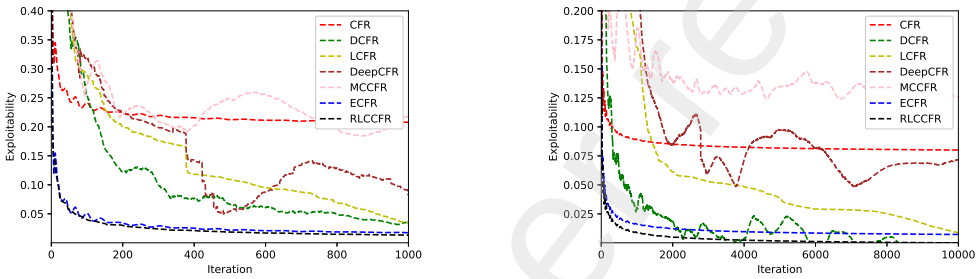

Figure 2: Tested on Kuhn poker and RLCCFR with parameters $\alpha_A = 1$ and $\beta_A = 0.4$, $\alpha_B = 0.8$ and $\beta_B = 0.6$, $\alpha_C = 0.9$ and $\beta_C = 0.5$. The X-axis represents the number of iterations, and the Y-axis represents the exploitability. RLCCFR: A smaller exploitability is better in general.

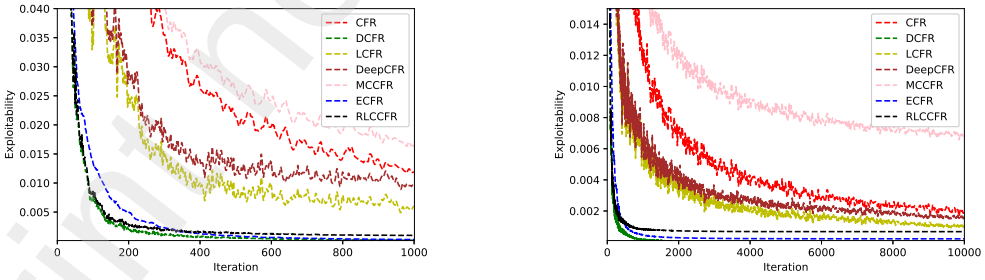

Figure 3: Tested on leduc poker and RLCCFR with parameters $\alpha_A = 0.6$ and $\beta_A = 1$, $\alpha_B = 0.4$ and $\beta_B = 0.9$, $\alpha_C = 0.2$ and $\beta_C = 0.8$. The X-axis represents the number of iterations, and the Y-axis represents the exploitability. DCFR is better than other algorithms.

Preprint not peer reviewed

This preprint research paper has not been peer reviewed. Electronic copy available at: https://ssrn.com/abstract=4806706

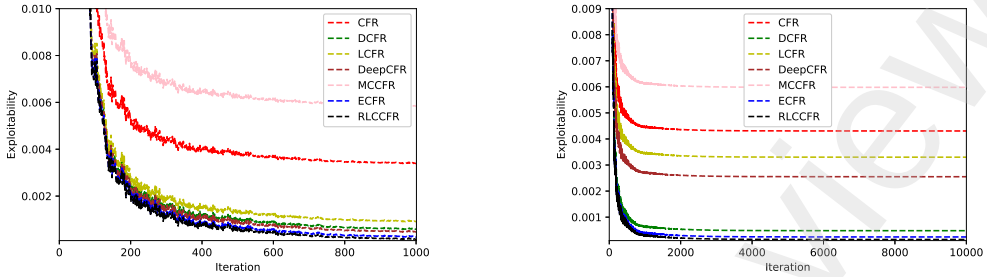

Figure 4: Tested on royal poker and RLCCFR with parameters $\alpha_A = 0.7$ and $\beta_A = 0.8$, $\alpha_B = 0.6$ and $\beta_B = 0.6$, $\alpha_C = 0.4$ and $\beta_C = 0.9$. The X-axis represents the number of iterations, and the Y-axis represents the exploitability. RLCCFR: A smaller exploitability is better in general.

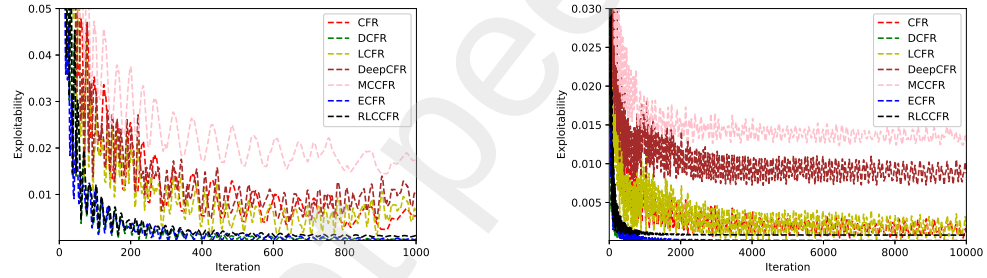

Figure 5: Tested on goofspiel v4 and RLCCFR with parameters $\alpha_A = 0.6$ and $\beta_A = 0.6$, $\alpha_B = 0.5$ and $\beta_B = 0.8$, $\alpha_C = 0.2$ and $\beta_C = 0.7$. The X-axis represents the number of iterations, and the Y-axis represents the exploitability. ECFR is better than other algorithms.

Preprint not peer reviewed

This preprint research paper has not been peer reviewed. Electronic copy available at: https://ssrn.com/abstract=4806706

Table 4: Tested on Kuhn poker for evaluate speed (Iteration per Speed) with parameters $\alpha_A = 0.3$ and $\beta_A = 0.8$, $\alpha_B = 0.4$ and $\beta_B = 0.7$, $\alpha_C = 0.8$ and $\beta_C = 1$.

Name Algorithm	Speed Iteration 10^3 itr/s	Speed Iteration 10^4 itr/s
CFR	161.51	153.66
DCFR	110.12	114.94
LCFR	118.46	124.16
MCCFR	208.34	212.74
DeepCFR	90.31	95.64
ECFR	127.91	133.64
RLCCFR	212.32	215.54

References

[1] M. Zinkevich, M. Johanson, M. Bowling, C. Piccione, Regret minimization in games with incomplete information, Advances in neural information processing systems 20 (2007) 1729–1736.

[2] M. Lanctot, K. Waugh, M. Bowling, M. Zinkevich, Monte carlo sampling for regret minimization in extensive games, Advances in neural information processing systems 22 (2009) 1078–1086.

[3] N. B. M. L. R. G. M Johanson, M. Bowling, Efficient nash equilibrium approximation through monte carlo counterfactual regret minimization, Aamas 2 (2012) 837–846.

[4] N. Brown, T. Sandholm, Solving imperfect-information games via discounted regret minimization, Proceedings of the AAAI Conference on Artificial Intelligence 33 (2019) 1829–1836.

[5] M. Zinkevich, N. Burch, M. Johanson, O. Tammelin, Heads-up limit holdem poker is solved, Science 347 (2015) 145–149.

[6] N. Brown, A. Lerer, S. Gross, T. Sandholm, Deep counterfactual regret minimization, International Conference on Machine Learning (2019) 739–802.

Preprint not peer reviewed

This preprint research paper has not been peer reviewed. Electronic copy available at: https://ssrn.com/abstract=4806706

[7] H. Li, K. Hu, Z. Ge, T. Jiang, Y. Qi, L. Song, Double neural counterfactual regret minimization, arXiv preprint arXiv:1812.10607.

[8] M. Schmid, N. Burch, M. Lanctot, M. Moravcik, R. Kadlec, M. Bowling, Variance reduction in monte carlo counterfactual regret minimization (vr-mccfr) for extensive form games using baselines, Proceedings of the AAAI Conference on Artificial Intelligence 33 (2019) 2157–2164.

[9] H. Li, X. Wang, S. Qi, J. Zhang, Y. Liu, Y. Wu, F. Jia, Solving imperfect-information games via exponential counterfactual regret minimization, arXiv preprint arXiv:2008.02679.

[10] H. Li, X. Wang, S. Qi, J. Zhang, Y. Liu, Y. Wu, F. Jia, Rlcfr: Minimize counterfactual regret by deep reinforcement learning, Expert Systems with Applications 187 (2022) 115953. doi:https://doi.org/10.1016/j.eswa.2021.115953.

[11] H. Xu, K. Li, H. Fu, Q. Fu, J. Xing, Autocfr: Learning to design counterfactual regret minimization algorithms, Proceedings of the AAAI Conference on Artificial Intelligence 36 (2022) 5244–5251.

[12] T. Sandholm, Abstraction for solving large incomplete-information games, Proceedings of the AAAI Conference on Artificial Intelligence 29. doi:https://doi.org/10.1609/aaai.v29i1.9757.

[13] M. Johanson, N. Bard, N. Burch, M. Bowling, Finding optimal abstract strategies in extensive-form games, Proceedings of the AAAI Conference on Artificial Intelligence 26 (2012) 1371–1379. doi:https://doi.org/10.1609/aaai.v26i1.8269.

[14] K. Waugh, D. Morrill, J. Bagnell, M. Bowling, Solving games with functional regret estimation, Proceedings of the AAAI Conference on Artificial Intelligence 29. doi:10.1609/aaai.v29i1.9445.

[15] M. G. Meyer, Toward generality: Building better counterfactual regret minimization for imperfect information games, Ph.D. thesis, Harvard Uni-

Preprint not peer reviewed

This preprint research paper has not been peer reviewed. Electronic copy available at: https://ssrn.com/abstract=4806706

versity (2022). doi:https://nrs.harvard.edu/URN-3:HUL.INSTREPOS:37370951.

[16] R. Gibson, M. Lanctot, N. Burch, D. Szafron, M. Bowling, Generalized sampling and variance in counterfactual regret minimization, Proceedings of the AAAI Conference on Artificial Intelligence 26 (2012) 1355–1361. doi:https://doi.org/10.7939/R3M61BP9B.

[17] R. Gibson, N. Burch, M. Lanctot, D. Szafron, Efficient monte carlo counterfactual regret minimization in games with many player actions, Advances in neural information processing systems 25 (2012) 1880–1888. doi:/doi/10.5555/3326943.3327000.

[18] P. Kuchar, Reducing variance in monte carlo counterfactual regret minimization, Faculty of Electrical Engineering Department of computer science.

[19] S. Ravichandiran, Hands-On Reinforcement Learning with Python, Packt Publishing Ltd, 2018.

[20] Deep Reinforcement Learning with Python: Master classic RL, deep RL, distributional RL, inverse RL, and more with OpenAI Gym and TensorFlow, Packt Publishing Ltd, 2020.

[21] C. F. Hayes1, R. Radulescu, A practical guide to multi.objective reinforcement learning and planning, Autonomous Agents and Multi-Agent Systems 36 (2022) 26. doi:doi.org/10.1007/s10458-022-09552-y.

[22] T. Xu, Y. Liu, Z. Ma, Y. Huang, P. Liu, A dqn-based multi-objective participant selection for efficient federated learningdoi:10.20944/preprints202304.0734.v1.

[23] M. Lanctot, E. Lockhart, J. Lespiau, V. Zambaldi, S. Upadhyay, Openspiel: A framework for reinforcement learning in games, arXiv preprint arXiv:1908.09453.

Preprint not peer reviewed

This preprint research paper has not been peer reviewed. Electronic copy available at: https://ssrn.com/abstract=4806706