Regret Minimization via Novel Vectorized Sampling Policies and Exploration

Hui Li[§*] **, Kailiang Hu**[§]**, Shaohua Zhang**[§]**, Lin Wang**[§]**, Jun Zhou**[§]**, Yuan Qi**[§]**, Le Song**[§♮]

[§]Ant Financial Services Group
[♮]Georgia Institute of Technology

Abstract

Imperfect information game is an important research topic in artificial intelligence. Solving large imperfect information game from sampling experiences is a challenging task. Monte Carlo counterfactual regret minimization (MCCFR) and its variants are the fundamental and effective techniques for solving imperfect information games. However, such sampling methods typically need a large number of iterations to approach an approximate Nash equilibrium. In this paper, we introduced several efficient vector-form sampling policies based on the public tree and proposed an efficient framework with novel exploration technique. The experiment results showed that the proposed methods empirically brought about $100\times \sim 1000\times$ speedup in many settings. Our methods potentially open up new avenues for research in these directions.

Introduction

Many remarkable advances have been made in artificial intelligence (AI) in recent years. AI researchers often use games as challenging problems and benchmarks for progress. Poker has served for decades as challenging benchmarks and milestones of solving imperfect information games. In such games, a player has only partial knowledge about her opponents before making a decision. The typical target of solving imperfect information games is to find a Nash equilibrium so that no player can unilaterally improve the reward.

When solving extremely large imperfect information games, it's difficult to fit a linear programming model with a manageable size. Many iterative techniques (Gilpin et al., 2007; Gordon, 2007; Kroer et al., 2015; Heinrich, Lanctot, and Silver, 2015) have been proposed as an alternative solution to these linear programming methods. Counterfactual regret minimization (CFR) has been one of the most efficient iterative algorithms in the progress of Poker AI, leading to solving heads-up limit Texas hold'em (Tammelin, 2014; Tammelin et al., 2015) and no-limit Texas hold'em (Moravcik et al., 2017; Brown and Sandholm, 2017, 2019b). Lanctot et al. (2009) propose a Monte Carlo CFR minimization (MCCFR) framework to solve imperfect information games from sampling experiences. Different from CFR, in each iteration, MCCFR samples a subset of nodes from the game tree according to some Monte Carlo methods. The superhuman AIs: Libratus (Brown and Sandholm, 2017) and Pluribus (Brown and Sandholm, 2019b) used MCCFR variants and finally beat top professionals in two-player and six-player no-limit Texas hold'em respectively. These works are considered as new milestones in artificial intelligence.

However, MCCFR variants typically have poor long-term performance due to the sampling policy and high variance. Some methods (Gibson et al., 2012; Johanson et al., 2012; Jackson, 2017; Li et al., 2020; Brown and Sandholm, 2019b) have been proposed to accelerate its convergence. Recently, Schmid et al. (2019); Davis, Schmid, and Bowling (2019) borrowed ideas of variance reduction from the community of reinforcement learning and estimated the baseline-correct expected value for MCCFR. Their experiment results showed that combining techniques of reinforcement learning and MCCFR is an important research direction of solving imperfect information games (Kovařík et al., 2019). Also, they show that performing MCCFR with vector-form implementation based on the simple uniform random Monte Carlo sampling can improve its convergence [1].

Following this direction, we have two questions:

- Is there any other technique from reinforcement learning community can be used to for the CFR community?
- Can one design novel Monte Carlo sampling policies to make vector-form MCCFR more efficient?

In this paper, we try to answer these two questions. In sum, we introduced a general framework of Monte Carlo counterfactual regret minimization by borrowing the exploration techniques from reinforcement learning communities. Based on this framework, we proposed several reasonable vectorized Monte Carlo sampling policies to make MCCFR more efficient. Empirically, our MCCFR variants perform significantly better than the state-of-the-art method. Some combinations of our ideas bring $2 \sim 3$ orders of magnitude speedup.

*Correspondence to Hui Li: lihuiknight@google.com

[1]Note that, vector-form MCCFR with uniform random sampling policy can leverage the advantages of matrix computing so that it's more efficient empirically, although it has the same theoretically floating-point operations with the value-form MCCFR. We will introduce value-form MCCFR and vector-form MCCFR in the next sections

Background and Notation

Notations for Imperfect Information Game

There are n players (except for chance) in extensive-form imperfect information games. $N = \{1, ..., n\}$ is a finite set and each member refers to a player. In two-player game, $N = \{1, 2\}$. These two players are denoted by $p1$ and $p2$. The *hidden information(variable)* of player i is unobserved by the opponents, which is denoted by h_i^v. Each member $h \in H$ refers to a possible *history* (or *state*). For player i, h_{-i}^v refers to all the players' hidden information except for i. The empty sequence $\emptyset$ is a member of H. $h_j \sqsubseteq h$ denotes h_j is a prefix of h. Z denotes the set of terminal histories and any member $z \in Z$ is not a prefix of any other sequences. A *player function* P assigns a member of $N \cup \{c\}$ to each non-terminal history, where c refers to the chance player. $P(h)$ is the player who takes actions at h. $A(h) = \{a : ha \in H\}$ is the set of available actions after $h \in H \setminus Z$. $\mathcal{I}_i$ of a history $\{h \in H : P(h) = i\}$ is an *information partition* of player i. A set $I_i \in \mathcal{I}_i$ is an *information set (infoset)* of player i and $I_i(h)$ refers to infoset I_i at state h, $i.e.$, each history h corresponds to an infoset $I_i(h)$. For $I_i \in \mathcal{I}_i$, we have $A(I_i) = A(h)$ and $P(I_i) = P(h)$. If all players in one game can recall their previous actions and infosets, we call it a *perfect-recall* game. The utility function $u_i(z)$ is the payoff of player i at terminal state z.

Game Tree and Public Tree

Given all players' histories, we can build a *prefix tree (trie)* recursively. Such prefix tree is called *game tree* in game theory (Johanson et al., 2011). Each node in the game tree refers to a history h. The *infoset tree* for each player is built on infosets rather than histories. *Public tree* is a prefix tree built on public sequences as illustrated in Figure 1. Each node $\vec{I}_i$ (*public node*) in public tree refers a vector of infosets $\vec{I}_i = [I_{i1}, I_{i2}, I_{i3}, ...]$. For $\forall I_{ij}, I_{ik} \in \vec{I}_i$, they can indicate exactly the same public sequence. $|\vec{I}_i|$ refers to length of the vector. Further details about game tree, infoset tree and public tree can be found in Johanson et al. (2011). Kovařík et al. (2019) gives a comprehensive definition of game tree and public tree. They call the public sequence in their paper as the public state.

Notations for Strategy and Nash Equilibrium

For play $i \in N$, the *strategy* $\sigma_i(I_i)$ in an extensive-form game assigns an action distribution over $A(I_i)$ to infoset I_i. A *strategy profile* $\sigma = \{\sigma_i | \sigma_i \in \Sigma_i, i \in N\}$, where Σ_i is the set of all possible strategy for player i. σ_{-i} refers to all strategies in σ except for σ_i. $\sigma_i(I_i)$ is the strategy of infoset I_i. $\sigma_i(a|h)$ refers to the probability of action a taken by player i at state h. $\forall h_1, h_2 \in I_i$, we have $I_i = I_i(h_1) = I_i(h_2)$, $\sigma_i(I_i) = \sigma_i(h_1) = \sigma_i(h_2)$, $\sigma_i(a|I_i) = \sigma_i(a|h_1) = \sigma_i(a|h_2)$. For iterative learning method such as CFR, σ^t refers to the strategy profile at t-th iteration.

$\pi^\sigma(h)$ refers to the *state reach probability (also called state range)*,which is the product of all players'(include chance) strategies along the history h. For an empty sequence, $\pi^\sigma(\emptyset) = 1$. The reach probability can be decomposed into $\pi^\sigma(h) = \prod_{i \in N \cup \{c\}} \pi_i^\sigma(h) = \pi_i^\sigma(h)\pi_{-i}^\sigma(h)$, where $\pi_i^\sigma(h)$ is the product of player $i's$ strategy σ_i and $\pi_{-i}^\sigma(h)$ is the product of all players' strategy σ_{-i} except i. $\forall h \in I_i$, $\pi_i^\sigma(h) = \pi_i^\sigma(I_i)$. For two histories h_1 and h_2, $h1 \sqsubseteq h2$, $\pi^\sigma(h_1, h_2)$ refers to the product of all player's strategy from history $h1$ to $h2$. We define $\pi_i^\sigma(h_1, h_2)$ and $\pi_{-i}^\sigma(h_1, h_2)$ in the similar way. The *infoset reach probability (infoset range)* of I_i is defined by $\pi^\sigma(I_i) = \sum_{h \in I_i} \pi^\sigma(h)$. Similarly, $\pi_{-i}^\sigma(I_i) = \sum_{h \in I_i} \pi_{-i}^\sigma(h)$. For player i, the *expected game utility* is defined by $u_i^\sigma = \sum_{z \in Z} \pi^\sigma(z) u_i(z)$.

Given a fixed strategy profile σ_{-i}, $br(\sigma_{-i}) = \text{argmax}_{\sigma_i' \in \Sigma_i} u_i^{(\sigma_i', \sigma_{-i})}$ is a *best response*. An *ϵ-Nash equilibrium* is an approximated Nash equilibrium, whose strategy profile $\sigma^* = (br(\sigma_{-i}), br(\sigma_i))$ satisfies: $\forall i \in N$, $u_i^{(br(\sigma_{-i}), \sigma_{-i})} + \epsilon \geq \max_{\sigma_i' \in \Sigma_i} u_i^{(\sigma_i', \sigma_{-i})}$. Exploitability of a strategy σ_i is defined by $\epsilon_i(\sigma_i) = u_i^{\sigma^*} - u_i^{(\sigma_i, br(\sigma_i))}$. If the players alternate their positions in two-player zero-sum imperfect information game, the value of a pair of games is zero, $i.e.$, $u_1^{\sigma^*} + u_2^{\sigma^*} = 0$. Therefore, *exploitability* of a strategy profile σ is defined by $\epsilon(\sigma) = \frac{u_2^{(\sigma_1, br(\sigma_1))} + u_1^{(br(\sigma_2), \sigma_2)}}{2}$.

Counterfactual Regret Minimization

CFR is an iterative method for finding a Nash equilibrium on zero-sum perfect-recall imperfect information games (Zinkevich et al., 2007) [2]. Given σ^t, the *counterfactual value* $v_i^{\sigma^t}(I_i)$ is defined by

$$v_i^{\sigma^t}(I_i) = \sum_{h \in I_i} \pi_{-i}^{\sigma^t}(h) \sum_{h \sqsubseteq z, z \in Z} \pi^{\sigma^t}(h, z) u_i(z). \tag{1}$$

$v_i^{\sigma^t}(a|I_i)$ refers to the counterfactual value of action a and its regret is defined by $r_i^{\sigma^t}(a|I_i) = v_i^{\sigma^t}(a|I_i) - v_i^{\sigma^t}(I_i)$. The *cumulative regret* of action a after t iterations is $R_i^t(a|I_i) = R_i^{t-1}(a|I_i) + r_i^{\sigma^t}(a|I_i)$, where $R_i^0(a|I_i) = 0$. Define $R_i^{t,+}(a|I_i) = \max(R_i^t(a|I_i), 0)$, the *current strategy* at $t+1$ iteration is updated by $\frac{R_i^{t,+}(a|I_i)}{\sum_{a \in A(I_i)} R_i^{t,+}(a|I_i)}$ (if $\sum_{a \in A(I_i)} R_i^{t,+}(a|I_i) = 0$, it is updated by $\frac{1}{|A(I_i)|}$). The *average strategy* $\bar{\sigma_i}^T$ after T iterations is defined by

$$\bar{\sigma_i}^T(a|I_i) = \frac{\sum_{t=1}^T \pi_i^{\sigma^t}(I_i) \sigma_i^t(a|I_i)}{\sum_{t=1}^T \sum_{a \in A(I_i)} \pi_i^{\sigma^t}(I_i) \sigma_i^t(a|I_i)}. \tag{2}$$

CFR+ (Tammelin, 2014) is very similar to CFR. It replaces regret matching by regret matching+ and uses a weighted average strategy. CFR and CFR+ are proven to approach Nash equilibria after enough iterations. The best known theoretical bound for CFR and CFR+ to converge to ϵ-equilibrium is $\mathcal{O}(\frac{1}{\epsilon^2})$ (Zinkevich et al., 2007; Burch, 2017; Burch, Moravcik, and Schmid, 2018). This bound is slower than first-order methods that converge at rate $\mathcal{O}(\frac{1}{\epsilon})$ (Hoda et al., 2010; Kroer et al., 2015). However, CFR+ empirically converges much faster than $\mathcal{O}(\frac{1}{\epsilon})$ in many games.

[2] Empirically, CFR can also use to solve perfect information game because it can be considered as a subset of imperfect information game.

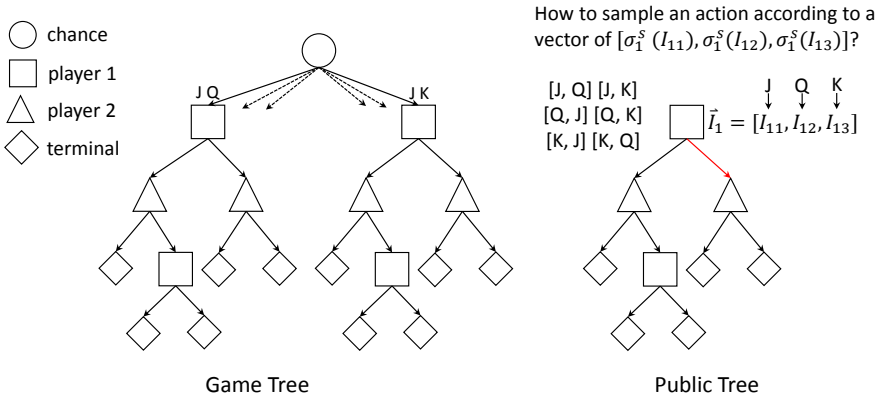


Figure 1: Game tree, public tree and the motivation on Kuhn.

Monte Carlo CFR

Lanctot et al. (2009) proposed a Monte Carlo CFR (MCCFR) to compute the unbiased estimate of counterfactual value by sampling subsets of infosets in each iteration. Define $\mathcal{Q} = \{Q_1, Q_2, ..., Q_m\}$, where $Q_j \in Z$ is a set (block) of sampled terminal histories generated by MCCFR, such that $\mathcal{Q}_j$ spans the set Z. Define q_{Q_j} as the probability of considering block Q_j, where $\sum_{j=1}^{m} q_{Q_j} = 1$. Define $q(z) = \sum_{j:z\in Q_j} q_{Q_j}$ as the probability of considering a particular terminal history z. The estimate of *sampled counterfactual value* of I_i is defined by

$$\tilde{v}_i^{\sigma}(I_i|Q_j) = \sum_{h\in I_i, z\in Q_j, h\sqsubseteq z} \frac{1}{q(z)} \pi_{-i}^{\sigma}(z)\pi_i^{\sigma}(h,z)u_i(z). \tag{3}$$

Define σ^s as sampled strategy profile, where σ_i^s is the sampled strategy of player i and σ_{-i}^s are those for other players except for i. The regret of the sampled action $a \in A(I_i)$ is defined by $\tilde{r}_i^{\sigma}(I_i, a|Q_j) = \tilde{v}_i^{\sigma}(I_i, a|Q_j) - \tilde{v}_i^{\sigma}(I_i|Q_j)$, where $\tilde{v}_i^{\sigma}(I_i, a|Q_j) = \sum_{z\in Q_j, ha\sqsubseteq z, h\in I_i} \pi_i^{\sigma}(ha, z)u_i^s(z)$, $u_i^s(z) = \frac{u_i(z)}{\pi_i^{\sigma^s}(z)}$ is the utility weighted by $\frac{1}{\pi_i^{\sigma^s}(z)}$. The estimate cumulative regret of action a after t iterations is $\tilde{R}_i^t(I_i, a|Q_j) = \tilde{R}_i^{t-1}(I_i, a|Q_j) + \tilde{r}_i^{\sigma^t}(I_i, a|Q_j)$, where $\tilde{R}_i^0(I_i, a|Q_j) = 0$.

MCCFR provably maintains an unbiased estimate of counterfactual value and converges to Nash equilibrium (Lanctot et al., 2009). Outcome sampling and external sampling are two popular sampling methods, which have served as benchmarks in many articles. The original outcome sampling (Lanctot et al., 2009) chooses one history according to two players' current strategy policy (or ϵ-greedy). The external sampling is very similar to outcome sampling except for one player taking all actions at her decision node. In each iteration, the classical MCCFR designates one player as the traverser, whose cumulative regret and strategy will be updated on this iteration. After that, another player will be designated as the traverser. Li et al. (2020) introduced robust sampling, which is a general version of outcome sample and external sampling. In robust sampling, the traverser samples k actions and the opponent samples one action. Recently, some related methods (Gibson et al., 2012; Johanson et al., 2012; Jackson, 2017; Brown and Sandholm, 2017; Schmid et al., 2019; Brown and Sandholm, 2019a; Davis, Schmid, and Bowling, 2019; Brown and Sandholm, 2019b) have been proposed to accelerate MCCFR's convergence.

Vectorized MCCFR and Motivation

We classify the common MCCFR variants into three categories: value-form, semi-vector-form, and vector-form MCCFR. To make a clear explanation, we introduce different MCCFR variants on Kuhn poker [3]. Assume there are two suits: ♣ and ♠ and three cards: J, Q, K. To simplify the explanation, we use the general robust sampling as the default sampling method and specify player $p1$ as the traverser. At each decision node, $p1$ samples one action according to uniform random policy and $p2$ samples one action according to her current strategy.

• **Value-form MCCFR:** At the start of each iteration, $p1$ and $p2$ are dealt with one private card respectively, such as ♣J for $p1$ and ♣Q for $p2$. Then they play against each other until the end. In perfect-recall two-player imperfect information game, given public sequence and $p2$'s private card, it's easy to locate a particular infoset $I_2 \in \mathcal{I}_2$. $p2$ samples one action according to $\sigma_2(I_2)$. In this scenario, the value-form MCCFR generates one history h on each iteration. The value of the terminal node is the game payoff.

• **Semi-vector-form MCCFR:** Suppose $p2$ is dealt with private card ♣Q and $p1$ is dealt with a vector of private cards [♣J, ♣K]. Similar to value-form MCCFR, these two players play against each other until the end. $p1$'s decision node maintains a vector of infosets $\vec{I}_1 = [I_{11}, I_{12}]$ and $p2$'s node maintains one infoset I_2. Also, $\vec{I}_1$ indicates a vector of policies $\vec{\sigma}_1 = [\sigma_{11}, \sigma_{12}]$. In this scenario, $p2$ samples her action according to $\sigma_2(I_2)$. When using robust sampling, $p1$ samples her actions according to uniform random policy rather than the vector of policies $\vec{\sigma}_1$, so that it's unnecessary to specify a particular current strategy as the sampling policy. Semi-vector-form MCCFR updates a vector of the traverser's regrets and strategies on her public state while only updates a particular regret and strategy for her opponent's infoset. It's expected that semi-vector-form MCCFR can benefit from efficient matrix manipulation and empirically converge faster than value-form MCCFR.

[3] https://en.wikipedia.org/wiki/Kuhn_poker

• **Vector-form MCCFR:** For this method, we don't need to specify private cards for $p1$ and $p2$. Both two players traverse along the public tree according to the specified sampling policy. We explain this method as follows. As shown in Figure 1, the decision node of player $i \in [1, 2]$ should maintain a vector of infosets $\vec{I}_i = [I_{i1}, I_{i2}, I_{i3}]$. In each iteration, the vector-form MCCFR generates a vector of sequences along the public tree. Because each decision node $\vec{I}_i$ indicates a vector of current strategies $\vec{\sigma}_i = [\sigma_{i1}, \sigma_{i2}, \sigma_{i3}]$, it's unclear how to design the sampling policy. Schmid et al. (2019); Davis, Schmid, and Bowling (2019) use a uniform sampling policy in their experiments so that each infoset in $\vec{I}_i$ shares the same uniform policy. Intuitively, we should pay more attention to the relatively important action. Therefore, **there remains an interesting question:**

> Can we design better sampling policies, which not only pay more attention to the relatively important action but also achieve better long-term performance?

Remark: Vector-form implementation is well-known for a long time, it's widely used in both CFR variants (without Monte Carlo sampling) and vector-form MCCFR (with the simple uniform random sampling policy). Obviously, there is no aforementioned problem in the CFR variants because the full-width methods don't need to sample action. Also, there is no such issue in the vector-form MCCFR with simply uniform random sampling policy because it doesn't use current strategy or average strategy as a part of its sampling policy. In the next section, we will address this question and provide several efficient sampling policies for vector-form MCCFR.

Vectorized Monte Carlo Sampling

In this section, we define a novel vectorized Monte Carlo CFR framework with an exploration technique and explain why these techniques works well.

Sampling Policies for Vector-form MCCFR

Suppose player i is the traverser and the decision-making node is $\vec{I}_i$ in the public tree. Recall $\vec{I}_i$ contains a vector of infosets, *i.e.*, $\vec{I}_i = [I_{i1}, I_{i2}, I_{i3}]$. Also, each infoset $I_{ij} \in \vec{I}_i$ indicates a current strategy $\sigma_i^t(\cdot|I_{ij})$ and an average strategy $\bar{\sigma}_i^t(\cdot|I_{ij})$. Now, we design several effective sampling policies for vector-form MCCFR.

• **Random Current Strategy (RCS):** When using RCS, player i randomly selects one infoset I_i from $\vec{I}_i$ and samples one action according to $\sigma_i^t(I_i)$. This method is very simple and naive.

• **Mean Current Strategy (MCS):** This sampling policy is the mean of the current strategy over all the infosets in $\vec{I}_i$, which is defined by

$$\sigma_i^{mcs}(a|\vec{I}_i) = \frac{\sum_{I_i \in \vec{I}} \sigma_i^t(a|I_i)}{\sum_{I_i \in \vec{I}_i} \sum_{a \in A(I_i)} \sigma_i^t(a|I_i)} = \frac{\sum_{I_i \in \vec{I}} \sigma_i^t(a|I_i)}{|\vec{I}_i|}. \tag{4}$$

The MCS gives different infoset $I_i \in \vec{I}_i$ the same weight.

• **Range-Weighted Current Strategy (WCS):** In the field of game theory, a player typically has a very low probability of taking disadvantageous action. Typically, the player makes different decisions in different situations. For example, the player may need to take a more aggressive strategy after beneficial public cards are revealed in a poker game. Following the idea of defining average strategy in Eq. (2), we weight different infoset $I_i \in \vec{I}_i$ by player i's range. The WCS sampling policy is defined by

$$\sigma_i^{wcs}(a|\vec{I}_i) = \frac{\sum_{I \in \vec{I}} \pi_i^{\sigma^t}(I) \sigma_i^t(a|I_i)}{\sum_{I \in \vec{I}_i} \sum_{a \in A(I_i)} \pi_i^{\sigma^t}(I) \sigma_i^t(a|I)}. \tag{5}$$

In addition, one can also replace the player i's own range $\pi_i^{\sigma^t}(I_i)$ in Eq. (5) by the opponent's range $\pi_{-i}^{\sigma^t}(I_i)$ and all players' range $\pi^{\sigma^t}(I_i)$. In many settings, all of those three methods approach Nash equilibrium efficiently.

• **Mean Average Strategy (MAS):** In CFR variants, the average strategy after enough iterations approaches a Nash equilibrium. Note that, the current strategy has not been proven to approach Nash equilibrium although it empirically has a strong performance in CFR+ variants on many settings (Burch, 2017). It's natural to ask the question: can we use the iterative average strategy to design our vectorized sampling policies and help the vector-form MCCFR approach Nash equilibrium efficiently ? In practice, we can use the average strategy within t iterations as a good approximation of Nash equilibrium. Specifically, we replace the current strategy σ^t in Eq. (4) by the average strategy $\bar{\sigma}^t$, and the weighted average strategy is defined by

$$\sigma_i^{mas}(a|\vec{I}_i) = \frac{\sum_{I \in \vec{I}} \bar{\sigma}_i^t(a|I_i)}{\sum_{I \in \vec{I}_i} \sum_{a \in A(I_i)} \bar{\sigma}_i^t(a|I_i)} = \frac{\sum_{I \in \vec{I}} \bar{\sigma}_i^t(a|I_i)}{|\vec{I}_i|}. \tag{6}$$

• **Range-Weighted Average Strategy (WAS):** Similarly, we can use range to weight the average strategy at iteration t like Eq. (5). The range-weighted average strategy is defined by

$$\sigma_i^{was}(a|\vec{I}_i) = \frac{\sum_{I \in \vec{I}} \pi_i^{\sigma^t}(I) \bar{\sigma}_i^t(a|I_i)}{\sum_{I \in \vec{I}_i} \sum_{a \in A(I_i)} \pi_i^{\sigma^t}(I) \bar{\sigma}_i^t(a|I)}. \tag{7}$$

We prefer to using $\pi_i^{\sigma^t}(I)$ rather than $\pi_i^{\bar{\sigma}^{t-1}}(I)$ as the weight of each infoset in Eq. (7), because Eq. (2) and Eq. (7) share the same weight.

Remark: Gibson et al. (2012); Gibson (2014) used average strategy as the sampling policy, however, they didn't address the weighted strategy like Eq. (7). Also, they didn't discuss vectorized Monte Carlo sampling.

Monte Carlo Sampling with Exploration

Decision-making problems in the partially observable environment are typically modeled as partially observable stochastic games within reinforcement learning community or extensive-form imperfect information games within the game theory community (Kovařík et al., 2019). Both these two communities use Monte Carlo sampling to solve the large-scale imperfect information games. However, these

two communities are mostly distinct with little sharing of the advanced ideas.

Exploitation and exploration are the key trade-offs in reinforcement learning (Deisenroth and Rasmussen, 2011). Empirically, Monte Carlo sampling with suitable exploration technique will help reinforcement learning method achieve better performance with fewer samples (iterations) (Schulman et al., 2015). Recently, Jin et al. (2018) introduced that Q-Learning with upper confidence bound (UCB) is provably efficient. Although many advanced exploration techniques on reinforcement learning have been proposed (Qian et al., 2018; Taïga et al., 2019), few of these techniques have been used to solve imperfect information games in game theory community (especially in counterfactual regret minimization community). A well-known application is the hybrid method of ϵ-greedy and outcome sampling (Lanctot et al., 2009). However, the effectiveness of hybrid methods with advanced exploration techniques is unclear.

To fill that gap, in this paper, we introduce a novel Monte Carlo sampling framework with an exploration technique for counterfactual regret minimization. In general, the mixture sampling policy in this framework is defined by

$$\sigma_i^{se}(a|\vec{I}_i) = (1-\alpha) * \sigma_i^{s}(a|\vec{I}_i) + \alpha * \sigma_i^{e}(a|\vec{I}_i), \quad (8)$$

where $\sigma_i^{s}(a|\vec{I}_i)$ refers to selected sampling policy, $\sigma_i^{e}(a|\vec{I}_i)$ refers to the exploration policy. $\alpha \in [0,1]$ is a decayed mixture factor, which is used to control the weight of exploration and typically holds $\lim_{t\to\infty}\alpha = 0$. One can specify any vector-form sampling policy σ_i^s, *e.g.*, σ_i^{rcs}, σ_i^{mcs}, and σ_i^{was}. Both σ_i^s and σ_i^e holds $\sum_{a\in A(\vec{I}_i)}\sigma_i^s(a|\vec{I}_i) = 1$, $\sum_{a\in A(\vec{I}_i)}\sigma_i^e(a|\vec{I}_i) = 1$. Therefore, σ_i^{se} holds $\sum_{a\in A(\vec{I}_i)}\sigma_i^{se}(a|\vec{I}_i) = 1$.

Remark: In this paper, we design vector-form Monte Carlo sampling policy based on public tree. For the value-form case, Eq.8 can be defined by $\sigma_i^{se}(a|I_i) = (1-\alpha) * \sigma_i^{s}(a|I_i) + \alpha * \sigma_i^{e}(a|I_i)$. Strictly speaking, the value-form method is a special case of the vector-form method, whose vector size is 1.

Vectorized Monte Carlo Sampling with Count-Based Exploration

We borrow the idea from reinforcement learning and apply a count-based exploration into vectorized Monte Carlo counterfactual regret minimization.

Define $c^t(a|\vec{I}_i)$ as the sampling times for action a at public node $\vec{I}_i$ in iteration t. If $\vec{I}_i$ or action a is not sampled in this iteration, the $c^t(a|\vec{I}_i)$ is 0. The cumulative sampling times are defined by $C^t(a|\vec{I}_i) = \sum_{j=1}^{t} c^j(a|\vec{I}_i)$. Note that, when using mini-batch MCCFR (Li et al., 2020), $c^t(a|\vec{I}_i)$ could be larger than 1 because a mini-batch of public sequences are sampled in one iteration. The exploration policy in our method is defined by

$$\sigma_i^{e,t}(a|\vec{I}_i) = \frac{\left(1+\beta/\sqrt{C^t(a|\vec{I}_i)}\right)}{\sum_{a\in A(\vec{I}_i)}\left(1+\beta/\sqrt{C^t(a|\vec{I}_i)}\right)}, \quad (9)$$

where $\sigma_i^{e,t}$ refers to the exploration policy in iteration t, β is a nonnegative real number.

Intuitive explanation. If $\beta = 0$, then $\sigma_i^{e,t}(\cdot|I_i)$ is a uniform random exploration. If $\beta > 0$ and action a at I_i is sampled over and over again, according to Eq. (9), $\sigma_i^{e,t}(a|I_i)$ tends to become small so that there is a potentially lower probability to sample this action again than the one without exploration. We find exploration is empirically helpful in MCCFR. Intuitively, we think the potential reason likes that: If the cumulative regret of one action is negative, its current strategy is zero. In this situation, this action will not be sampled in the next few iterations. However, this action could have a larger overall regret than other actions after long-running iterations. Therefore, it will need to perform a lot of iterations to make its value change from negative cumulative regret to positive value. However, after using exploration, MCCFR has a certain probability to sample this action even when its cumulative regret is negative (the corresponding sampling policy $\sigma_i^{se}(a|\vec{I}_i)$ in Eq. (8) is zero at t iteration).

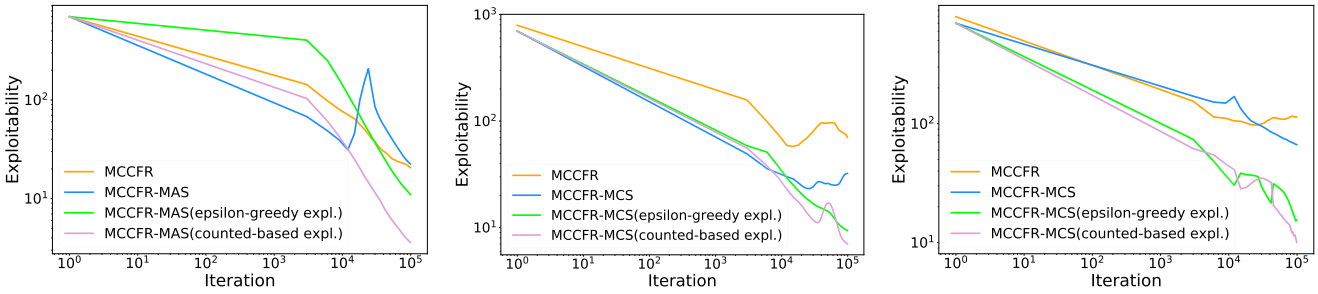
Remark. In this section, we introduce a simple count-based exploration and try to fill the gap between reinforcement learning community and CFR community in the field of applying exploration technique into vectorized MCCFR. We believe that our work potentially opens up new avenues for research in this direction. Some other latest exploration techniques in reinforcement learning (Jin et al., 2018; Qian et al., 2018; Taïga et al., 2019) are waiting for us to explore in the future.

Experiment

Poker has served for decades as a challenging benchmark and milestone of solving imperfect information games (Zinkevich et al., 2007; Lanctot et al., 2009; Moravcik et al., 2017; Brown and Sandholm, 2019a,b). In this paper, we evaluated our methods on heads-up no-limit preflop hold'em poker (NLPH). We also report the convergence of hybrid methods: combining the vectorized Monte Carlo MCCFR with the advanced skipping mechanism and discounting technique, the experimental results show that our method can benefit such advanced techniques and obtains the state-of-the-art convergence. Because of the limitation of space, we present detailed comparisons on vectorized Monte Carlo sampling policies with exploration on NLPH. Empirically, our method achieves 2 or 3 orders of magnitude improvement and makes MCCFR much more efficient in solving large-scale imperfect information games.

Metric and Parameters

Different from the standard metric: average episode reward in the reinforcement learning community, game theory community evaluates imperfect information solving algorithm by the standard metric: exploitability (Zinkevich et al., 2007; Michael Bowling, 2015). The unit of exploitability in our paper is *chips per game*. It denotes how many chips one player losses on average per hand of poker when she plays against a Nash equilibrium strategy. That is, exploitability refers to the worst loss on average. For the abstracted large games, the exploitability is computed on the abstracted game. The



(a) Average Sampling (b) Robust Sampling (c) Outcome Sampling

Figure 2: Log-log plots of the convergence for MCCFR variants with/without exploration on NLPH. We test exploration technique on three different sampling methods: (a) average sampling, (b) robust sampling and (c) outcome sampling. MCCFR refers to the semi-vector-form MCCFR. Lower exploitability indicates better results.

method with a lower exploitability is better. Nash equilibrium has zero exploitability.

Game Rules

HUNL is a primary benchmark for the imperfect information game solving methods. The HUNL we used in this paper is the standard version in the Annual Computer Poker Competition[4]. At the start of HUNL, the two players have 20000 chips. HUNL has at most four betting rounds if neither players fold in advance. The four betting rounds are named by preflop, flop, turn, and river respectively. At the start of each hand, both players are dealt with two private cards from a 52-card deck. One player at the position of the small blind should firstly put 50 chips into the pot and the other player at the big blind should put 100 chips into the pot. Their positions alternate after each hand. Each player can choose fold, call, or raise. If one player chooses fold, then she will lose the money in the pot and this hand is over. If one player chooses call, she should place a number of chips into the pot so that her total chips are equal to the opponent's chips. If one player chooses raise, she should add more chips into the pot than the opponent does. After the preflop round, three public cards are revealed and then the flop betting round occurs. After this round, another public card is dealt and the third betting round takes place. After that, the last public card is revealed, then the river round begins.

Although HUNL contains about 10^{161} infosets (Johanson, 2013) and is too large to traverse all the nodes, state-of-the-art agents such as Libratus (Brown and Sandholm, 2017) and DeepStack (Moravcik et al., 2017) solve the subgame of the full HUNL in real time based on action abstraction or card abstraction techniques. To reduce the computation, we also use similar abstraction technique and consider 1x the pot and all in the each betting round without any card abstraction.

NLPH has only one betting round and the value for the terminal node is represented by the expected game utility under the uniform random community cards, which is precomputed and saved on the disk. NLPH contains 7.8×10^4 infosets and 1.0×10^9 states.

Settings

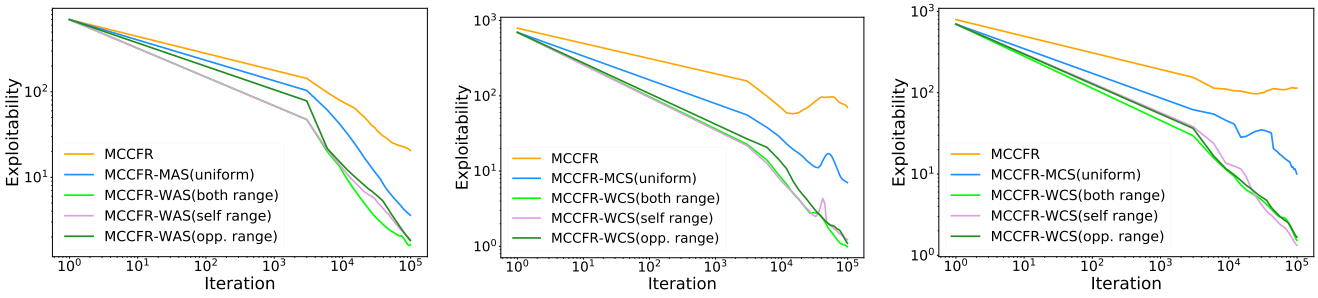
We conducted a set of comparison on four sampling methods: (1) *average sampling*: the traverser samples 1 action according to uniform random policy while the opponent samples 1 action according to the designed vectorized MAS Monte Carlo sampling policies. (2) *outcome sampling*: both the traverse and the opponent sample 1 action according to MCS policy. (3) *external sampling*: the traverser takes all action in her decision-making node while the opponent samples 1 action according to the designed vectorized MCS policy. (4) *robust sampling*: the traverser samples $k = 1$ action in her decision-making node according to uniform random policy while the opponent samples 1 action according to the designed vectorized MCS policy.

Note that, the concurrent work (Davis, Schmid, and Bowling, 2019) used both uniform sampling (both players used uniform random sampling policy) and robust sampling in their experiment. Their results showed robust sampling achieved better performance than uniform sampling. Without loss of generality, in this paper, we compared different MCCFR variants on NLPH poker based on the robust sampling. Typically, in MCCFR, the public tree or game tree is traversed separately for each player (Brown and Sandholm, 2017, 2019b). We follow this criterion in our experiment. We set default $\alpha = \frac{1}{\ln(t+10)}$, $\beta = \ln(t+10)$ and $\gamma = 0.5$.

Comparison

Improvement of exploration. Figure 2 demonstrates the convergence of different MCCFR variants before/after using exploration technique on average sampling method, robust sampling and outcome sampling. All the curves are log-log plots (log scale of both x-axis and y-axis). The label MCCFR in this figure refers to semi-vector Monte Carlo CFR method. Because few articles have addressed the exploration in MCCFR, We compared our exploration technique against the MCCFR variants without exploration and with ϵ-greedy (Lanctot et al., 2009). In Figure 2 (a) and (b), the MCCFR-MAS with both $epsilon$-greedy exploration and the proposed counted-based exploration converge slower than the one without exploration technique within the first 5000 iterations, however they achieve much lower exploitability (better performance) in long-term running. It seems that the method via average sampling is better than robust sampling and robust sampling is better than outcome sampling. The reason is that the average strategy learned by MCCFR approaches to Nash equilibrium while current strategy doesn't hold this property. Using the iterative average strategy is more stable than current strategy in vectorized Monte Carlo sampling. We believe the advanced exploration

[4]http://www.computerpokercompetition.org/



(a) Average Sampling (b) Robust Sampling (c) Outcome Sampling

Figure 3: Log-log plots of the convergence on NLPH. Comparison for vectorized Monte Carlo sampling polices.

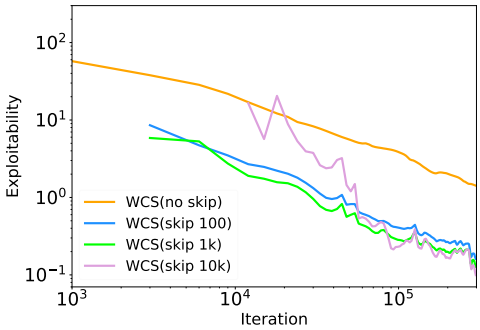
techniques in reinforcement learning may also work well but they are out of the scope of this paper. Our work opens up the avenues for research in this direction.

Improvement of sampling policies. Figure 3 presents the convergence of the designed vectorized Monte Carlo sampling policies. Note that, in Figure 3, we used the mixture of vectorized sampling policies and exploration as the default setting to demonstrate the incremental improvement of the proposed vectorized Monte Carlo sampling methods. As introduced in the aforementioned section, players' range can be used to weight the vectorized sampling policies. We demonstrate four different settings: (1) uniform weight, which refers to mean average strategy(MAS) in average sampling or mean current strategy(MCS) in robust sampling and outcome sampling; (2) weighted by both players' range ($\pi(I_i)$; (3) weighted by the traverser's range $\pi_i(I_i)$; (3) weighted by the opponent's range $\pi_{-i}(I_i)$. The experimental results show that the methods weighted by range are better than uniform weight. That's reasonable, because $\vec{I}_i$ contains a vector of infosets, but these infosets have different probabilities to reach the sharing public node. In imperfect information games, range denotes the reach probability. From figure 3, we can see that all the vectorized MCCFR achieves much better performance than the semi-vector MCCFR (about $2 \sim 3$ orders of magnitude improvement). Note that, typically semi-vector-form MCCFR converged faster than its value-form version so that we didn't present the convergence curve for the value-form MCCFR. In average sampling, the method weighted by both players' range is slightly better than the one weighted by traverser's or opponent's range. In robust sampling and outcome sampling, these three weighting methods have similar performance and all of them are significant than uniform weighting method.

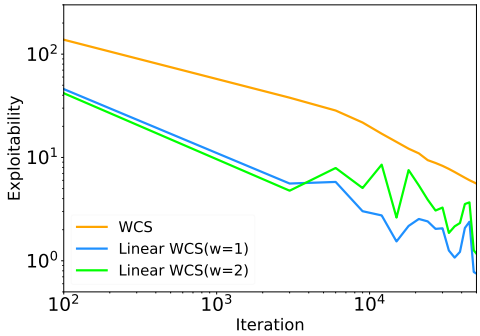
Hybrid Methods with Acceleration Techniques

We use some acceleration techniques to improve the convergence of the vectorized MCCFR.

Hybrid Methods with Skipping Mechanism. In CFR, the cumulative regret is initialized by zero and the current strategy starts from a uniform random strategy. Only the average strategy profile within all iterations is proved to converge to Nash equilibrium. Recently, some works (Moravcik et al., 2017; Li et al., 2019) demonstrated that skipping previous iterations of CFR can obtain faster convergence empirically although this practical technique is lack of theoretical convergence. We wonder whether this technique can also be used in vectorized Monte Carlo framework



(a) Skipping Mechanism



(b) Discounting Updates

Figure 4: Hybrid methods with two acceleration techniques on NLPH.

and makes our method more efficient. In figure 4(a), we presented the performance of the hybrid MCCFR variants by combining vectorized Monte Carlo sampling and skipping mechanism. We report the convergences of three vectorized MCCFR settings: skipping the first 100, 1k and 10k iterations. It's clear that the skipping mechanism dramatically improved the performance. The method by skipping previous 10k iterations approached 0.94-Nash equilibrium (the method without skipping mechanism approached 1.42.).

Hybrid Methods with Discounting Updates. Recently, Brown and Sandholm (2019a) proposed an efficient linear CFR, which empirically converges faster than the previous state-of-the-art CFR+. The linear CFR weights the regrets and average strategies with the iteration t. In our experiment, we combined this discounting mechanism with our vector-form MCCFR and specified the weight by t^w ($w = 1$ and $w = 2$). Figure 2(b) showed that the method weighted by $w = 1$ is better than $w = 2$ and both of them converge more efficient the one without this acceleration technique.

Remark: There are also some other acceleration tech-

niques, such as mini-batch updates (Li et al., 2020), variance reduction method (Schmid et al., 2019; Davis, Schmid, and Bowling, 2019), targeted CFR (Jackson, 2017). We believe one can design more hybrid methods based on our vectorized Monte Carlo framework and the advanced acceleration technique, which could lead to more efficient MCCFR variants. However, investigating all these hybrid methods is out of the scope of this paper.

Conclusion and Future Work

We designed a novel vectorized Monte Carlo counterfactual regret minimization framework and applied the spirit of exploration from reinforcement learning into MCCFR. Based on this framework, we introduced several powerful vectorized Monte Carlo sampling policies. Empirically, our method obtained $2 \sim 3$ orders of magnitude improvement. We believe our works potentially open up new avenues for research in these directions.

Acknowledgments We would like to thank Noam Brown, Marc Lanctot, Trevor Davis, Martin Schmid and Neil Burch for insightful discussions and feedback. We also would like to thank the anonymous reviewers for pointing out some issues in the previous version of this work.

References

Brown, N., and Sandholm, T. 2017. Superhuman AI for heads-up no-limit poker: Libratus beats top professionals. *Science* eaao1733.

Brown, N., and Sandholm, T. 2019a. Solving Imperfect-Information Games via Discounted Regret Minimization. *AAAI*.

Brown, N., and Sandholm, T. 2019b. Superhuman ai for multiplayer poker. *Science* eaay2400.

Burch, N.; Moravcik, M.; and Schmid, M. 2018. Revisiting CFR+ and Alternating Updates. *arXiv preprint arXiv:1810.11542*.

Burch, N. 2017. Time and Space: Why Imperfect Information Games are Hard. PhD thesis.

Davis, T.; Schmid, M.; and Bowling, M. 2019. Low-variance and zero-variance baselines for extensive-form games. *arXiv preprint arXiv:1907.09633*.

Deisenroth, M., and Rasmussen, C. E. 2011. Pilco: A model-based and data-efficient approach to policy search. In *ICML*, 465–472.

Gibson, R.; Lanctot, M.; Burch, N.; Szafron, D.; and Bowling, M. 2012. Generalized sampling and variance in counterfactual regret minimization. In *AAAI*.

Gibson, R. G. 2014. Regret minimization in games and the development of champion multiplayer computer poker-playing agents.

Gilpin, A.; Hoda, S.; Pena, J.; and Sandholm, T. 2007. Gradient-based algorithms for finding Nash equilibria in extensive form games. In *International Workshop on Web and Internet Economics*, 57–69. Springer.

Gordon, G. J. 2007. No-regret algorithms for online convex programs. In *NIPS*, 489–496.

Heinrich, J.; Lanctot, M.; and Silver, D. 2015. Fictitious self-play in extensive-form games. 805–813. ICML.

Hoda, S.; Gilpin, A.; Pena, J.; and Sandholm, T. 2010. Smoothing techniques for computing nash equilibria of sequential games. *Mathematics of Operations Research* 35(2):494–512.

Jackson, E. G. 2017. Targeted cfr. In *Workshops on AAAI*.

Jin, C.; Allen-Zhu, Z.; Bubeck, S.; and Jordan, M. I. 2018. Is q-learning provably efficient? In *NIPS*, 4863–4873.

Johanson, M.; Waugh, K.; Bowling, M.; and Zinkevich, M. 2011. Accelerating best response calculation in large extensive games. In *IJCAI*.

Johanson, M.; Bard, N.; Lanctot, M.; Gibson, R.; and Bowling, M. 2012. Efficient nash equilibrium approximation through monte carlo counterfactual regret minimization. In *International Conference on Autonomous Agents and Multiagent Systems*.

Johanson, M. 2013. Measuring the size of large no-limit poker games. *arXiv preprint arXiv:1302.7008*.

Kovařík, V.; Schmid, M.; Burch, N.; Bowling, M.; and Lisỳ, V. 2019. Rethinking formal models of partially observable multiagent decision making. *arXiv preprint arXiv:1906.11110*.

Kroer, C.; Waugh, K.; Kilinç-Karzan, F.; and Sandholm, T. 2015. Faster first-order methods for extensive-form game solving. In *Proceedings of the Sixteenth ACM Conference on Economics and Computation*, 817–834. ACM.

Lanctot, M.; Kevin, W.; Martin, Z.; and Bowling, M. 2009. Monte Carlo sampling for regret minimization in extensive games. NIPS.

Li, H.; Hu, K.; Qi, Y.; and Song, L. 2019. Efficient cfr for imperfect information games with instant updates. *ICML on RWSDM*.

Li, H.; Hu, K.; Qi, Y.; and Song, L. 2020. Double Neural Counterfactual Regret Minimization. *ICLR*.

Michael Bowling, Neil Burch, M. J. O. T. 2015. Heads-Up Limit Texas Hold'em is solved. *Science* 347(6218):145–149.

Moravcik, M.; Martin, S.; Neil, B.; Viliam, L.; Morrill, D.; Bard, N.; Davis, T.; Waugh, K.; Johanson, M.; and Bowling, M. 2017. Deepstack: Expert-level artificial intelligence in heads-up no-limit poker. *Science* (6337):508–513.

Qian, J.; Fruit, R.; Pirotta, M.; and Lazaric, A. 2018. Exploration bonus for regret minimization in undiscounted discrete and continuous markov decision processes. *NIPS*.

Schmid, M.; Burch, N.; Lanctot, M.; Moravcik, M.; Kadlec, R.; and Bowling, M. 2019. Variance Reduction in Monte Carlo Counterfactual Regret Minimization (VR-MCCFR) for Extensive Form Games using Baselines. *AAAI*.

Schulman, J.; Levine, S.; Abbeel, P.; Jordan, M.; and Moritz, P. 2015. Trust region policy optimization. In *ICML*, 1889–1897.

Taïga, A. A.; Fedus, W.; Machado, M. C.; Courville, A.; and Bellemare, M. G. 2019. Benchmarking bonus-based exploration methods on the arcade learning environment. *arXiv preprint arXiv:1908.02388*.

Tammelin, O.; Burch, N.; Johanson, M.; and Bowling, M. 2015. Solving heads-up limit Texas Hold'em. In *IJCAI*.

Tammelin, O. 2014. Solving large imperfect information games using CFR+. arXiv preprint.

Zinkevich, M.; Michael, J.; Michael, B.; and Piccione, C. 2007. Regret minimization in games with incomplete information. NIPS.