Improve Counterfactual Regret Minimization for Card Game Cheat

Cheng Yi[1,a)] Tomoyuki Kaneko[2,b)]

Abstract: Counterfactual Regret Minimization (CFR) is one of the most popular and effective iterative algorithms to solve large zero-sum imperfect information games, especially in the area of pokers. *Cheat* is one of the large card games due to the possible repetitions in the game histories. When solving such games, the challenges lie in lowering the cost of the computation time and storage space. In this paper, we implement the technique of External-Sampling Monte Carlo CFR and Best-Response Pruning on the game Cheat. Both of them have provably shown improvements in time and space compared to the vanilla CFR.

Keywords: Counterfactual Regret Minimization (CFR), External-Sampling Monte Carlo CFR, Imperfect Information games, Best-Response Pruning, Card Game Cheat

1. Introduction

In the Artificial Intelligence research area, games often act as our challenge problems and research benchmarks. We are always curious about learning how well players with different goals can effectively adjust their strategies in the interaction situations with other players. Games can be separated into two categories: one is perfect information games, such as Chess and Go and the other one is imperfect information games such as Mahjong and most of the poker games. The "information" here refers to the public information in a game that is available to all the players. In imperfect information games, the players do not know everything about their opponents.

When dealing with imperfect information games, we wish to find a Nash Equilibrium. A Nash Equilibrium is a strategy profile of a game where no one has the incentive to deviate from, because they cannot benefit from this. For smaller games with 10^8 or fewer nodes as mentioned in study [6], we can find the exact Nash Equilibrium, but beyond that we can only approach the Nash equilibrium by iterative algorithms. Counterfactual Regret Minimization (CFR) is a famous iterative algorithm converging to a Nash Equilibrium based on the regrets calculation for the players. Vanilla CFR requires a traversal of the whole game tree in each iteration and tries to minimize the regret at each node. But it becomes infeasible for us to store all the information and the computation is beyond the calculation power of normal computers when we are dealing with large games. The study [7] introduces Monte Carlo CFR (MCCFR), a domain-independent CFR sample-based algorithm. MCCFR restricts the number of terminal nodes we deal with each iteration and the sampling scheme we used in this paper is called External-Sampling. It samples the actions of the opponents and chance player that are "external" to the player. The researchers prove this approach helps to avoid traversing the entire game tree but is still capable of keeping the expectation of counterfactual regrets unchanged.

In this paper we also use the technique called *Best-Response Pruning* (BRP), a pruning for iterative algorithms, to improve our CFR performance. In the study [3], experiments show that BRP provably speeds up the convergence and reduces the space requirement by a factor of 7 and the larger the game is, the bigger the reduction factor becomes. The main idea of BRP is that, when our opponent's strategy does not change sufficiently (that is no faster than $\frac{1}{t}$, where t is the number of iterations so far), we can temporarily prune the actions in an infoset which have shown poorly behaviours compared to other actions against the opponent's average strategy. We can also bound the upper bound of improvements of such actions over specific number of iterations hence it is still safe to ignore them at the same time. If we focus on the competitive actions in every infoset, we can improve the overall convergence to the Nash Equilibrium.

2. Background

2.1 Notations and terminology

A finite extensive-form of an imperfect-information game is composed of the following parts: first, there is a finite-size group of *players*, $\mathcal{P}$. There is also a chance player, representing the outcome which are not controlled

[1] Graduate School of Interdisciplinary Information Studies, The University of Tokyo
[2] Interfaculty Initiative in Information Studies, the University of Tokyo
[a)] yi-cheng199@g.ecc.u-tokyo.ac.jp
[b)] kaneko@acm.org

by any player (e.g. the outcome of dealing the cards). A *history* $h \in \mathcal{H}$ is a node on the game tree, made up of all the information at that exact game state, including public information to all the players and private knowledge available to only one specific player. We use A to denote the set of all the *legal actions* in the game and $A(h)$ is the action space for players at the history h. If history h' is reached after we choose action $a \in A(h)$ at history h, which means h' is one of the child node of h and h is a prefix of h', we write $h \cdot a = h'$ or $h \sqsubseteq h'$ to represent this. A *terminal history* $z \in \mathcal{Z} \subseteq \mathcal{H}$ is where there is no more available actions and each player will get a payoff value for what they have done following the game tree respectively. For each player $i \in \mathcal{P}$, there is a payoff function $u_i : \mathcal{Z} \to \mathcal{R}$ and especially in two-player zero-sum games, $u_1 = -u_2$. Define $\Delta_i = \max_{z \in \mathcal{Z}} u_i(z) - \min_{z \in \mathcal{Z}} u_i(z)$ and $\Delta = \max_i \Delta_i$.

An *Information set* (infoset) is a set of histories that for a particular player, they cannot distinguish which history they are in between one another. Formally, $\forall h, h' \in I, A(h) = A(h') = A(I)$. The term $P(I)$ refers to the player who is supposed to take actions in the infoset I. $H(\sigma)$ is the set of information sets that can be reached if players follow the strategy σ. Note that any node/history $h \in H$ which are not terminal must and only belong to one of the information sets. $D(I, a)$ is the set of infosets which are reachable by the player who takes action a in the infoset I. Define $U(I)$ and $L(I)$ to be the upper and lower bounds of the payoff reachable after reaching the infoset I, i.e. $U(I) = \max_{z \in \mathcal{Z}, h \in I : h \sqsubseteq z} u_{P(I)}(z)$ and $L(I) = \min_{z \in \mathcal{Z}, h \in I : h \sqsubseteq z} u_{P(I)}(z)$ and $\Delta(I) = U(I) - L(U)$ to be the range of the corresponding payoffs. Similarly $U(I, a)$, $L(I, a)$ and $\Delta(I, a)$ are upper, lower and range of payoffs reachable by taking action a in the infoset I.

A *strategy* (or policy), $\sigma_i^t(I, a)$ for player i maps the information set I and the action $a \in A(I)$ to the probability that player i will exactly choose action a at the information set I on iteration t and σ_i is a probability vector for player i over all available strategies in the game. For all the histories in one information set, we should also have the identical strategy $\sigma(I)$. A strategy profile σ is a tuple of all the players' strategies. A strategy σ_i^* such that $u_i(\sigma_i^*, \sigma_{-i}) = \max_{\sigma_i' \in \Sigma} u_i(\sigma_i', \sigma_{-i})$ is a best response to σ_{-i}. So Nash equilibrium is formally defined as a strategy profile σ^*, in which every player is playing the best response. An *ϵ-equilibrium* is a strategy profile σ^* such that $\forall i, u(\sigma_i^*, \sigma_{-i}^*) + \epsilon \geq \max_{\sigma_i' \in \Sigma_i} u_i(\sigma_i', \sigma_{-i}^*)$.

Let $\pi^\sigma(h)$ denote the *reach probability* of reaching the game history h while all the players follow the strategy profile σ. The contribution of player i to this probability is $\pi_i^\sigma(h)$ and π_{-i}^σ denotes the contribution of the chance player and all the players other than i.

2.2 The game Cheat

Cheat (also known as *Doubt* or *Bullshit*) is a card game of lying and bluffing while also detecting opponents' deception. There are different versions played all over the world. In a common situation, one pack of 52 cards are well shuffled and dealt to the players as equally as possible. At the beginning of the game, a randomly chosen player is set to be the discard player and the "Rank" which all the players share is set to be Ace.

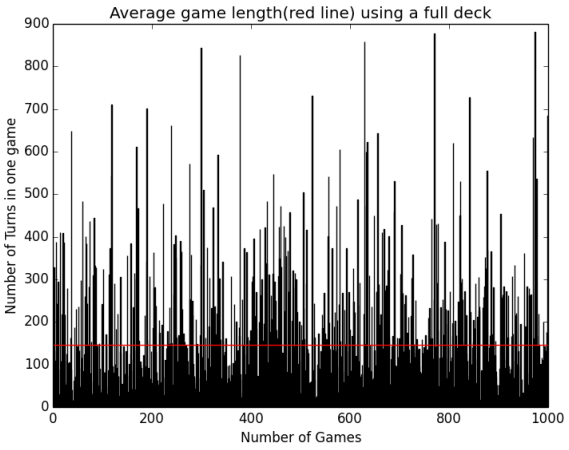

Fig. 1 Game length with two-player, one-deck

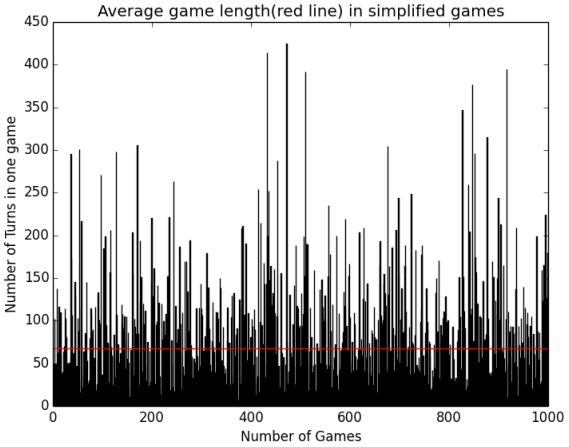

Fig. 2 Game length with two-player, half-deck

One turn in the game includes two phases, one is "Discard" and the other one is "Challenge". In Discard phase, the discard player discards cards, puts them facing down at the table and makes a claim. A claim includes the number of cards he discarded and the current rank, for example, "two aces". But the players can tell lies - that means the player can bluff it out when he is not holding any aces or discard different cards even if he has the cards. Then we move to the Challenge phase, all the other players, starting from the player sitting next to the discard player, are asked if they think the discard player is lying. If so, the player can challenge the discard player by saying "Cheat!". When there is a challenge, the cards last discarded will be revealed to see whether they are consistent with the claim. If the accused player did lie then they must take all the cards on the table back to his hands, otherwise the challenger takes the pile. The Challenge phase ends either after someone challenged the discard player or no one chose

to challenge after all the player other than the discard one has been asked. For the next turn, we increase the rank index (K is followed by Ace) and the discard player index by one. The one who first discards all the cards and is not successfully challenged at the last turn wins the game.

This game is often played among three or more players using two or more decks. Fig. 1 and Fig. 2 show the game length of 1,000 games and their averages between two Naive agents playing with the whole deck and half deck respectively. While the game is not perfect recall, since the game involves discarding and taking back cards, it is possible for players to run into similar or even the same states during the game, especially when the number of players is small. The repetitions in game histories result in deep game trees and in such cases, also an exponential increase in computation time and storage space when traversing the whole game tree. In this paper, we focus on a simplified but still strategically complex version of Cheat: a two-player 6-card setups, in order to refrain from violating the proved convergence of CFR in two-player zero-sum games.

2.3 Counterfactual Regret Minimization

Counterfactual Regret Minimization (CFR) is an algorithm built on the regrets calculation of each action no matter whether the player did or did not choose in the real game history. It was first propounded in 2008 by Zinkevich et al. in the paper [9] where the idea that claims minimizing overall regret can be used for approximating a Nash equilibrium in extensive games with incomplete information was demonstrated and proved. CFR is an iterative algorithm and the basic steps of one iteration of vanilla CFR are the following: first, it keeps a record of the regret values for all actions (all zeros at the beginning); second, the values are used to generate strategies and this step is called Regret-Matching; third, the regret values are updated based on the new strategies and then again from the first step. After all iterations, the average strategy obtained by normalizing overall actions belonging to the action space of this information set, is proved to converge to the best strategy and thus we approach Nash equilibrium as time tends to infinity. With the help of all the expert researchers, different variants of CFR have been developed one by one, such as AI agents called DeepStack[8] and Libratus[4] who defeated professional human players in two-player Head-up No-limit (HUNL) Texas Hold'em poker and Deep CFR[1] which combines the merits of Deep learning and CFR. And the latest breakthrough also made by Libratus' creators [5] - they have solved the multiplayer version of the same poker game: they have trained a stronger agent who can beat top human players in both 5-human-vs-1-AI and 5-AI-vs-1-human HUNL poker.

Here is some mathematical details of vanilla CFR. In a finite extensive form game that is played repeatedly, define the *counterfactual value* in the infoset I is:

$$v_i^{\sigma}(I) = \sum_{z \in Z} \pi_{-i}^{\sigma}(I)\pi^{\sigma}(I, z)u_i(z) \tag{1}$$

The *counterfactual value* of an action a is:

$$v_i^{\sigma}(I, a) = \sum_{z \in Z} \pi_{-i}^{\sigma}(I)\pi^{\sigma}(I \cdot a, z)u_i(z) \tag{2}$$

A *Counterfactual Best Response* (CBR) is a strategy that maximizes counterfactual value at infosets that it does not have to actually reach. A Counterfactual Best Response to σ_{-i} is defined as $CBR(\sigma_{-i})$ such that $CBR(\sigma_{-i})(I, a) > 0$ then $v^{\langle CBR(\sigma_{-i}),\sigma_{-i}\rangle}(I, a) = \max_{a'} v^{\langle CBR(\sigma_{-i}),\sigma_{-i}\rangle}(I, a')$. The *instantaneous regret* for player i not choosing action a when reaching the infoset I on iteration t is:

$$r_i^t(I, a) = \pi_{-i}^{\sigma^t}(v_i^{\sigma^t}(I, a) - v_i^{\sigma^t}(I)), \tag{3}$$

where $v(I, a)$ is the value of choosing action a in the infoset I. Then the *cumulative regret* on iteration T is:

$$R_i^T(I, a) = \sum_{t=1}^{T} r_i^t(I, a). \tag{4}$$

Additionally, $R_{i,+}^T(I, a) = \max\{R_i^T(I, a), 0\}$ and $R_i^T(I) = \max_a R_{i,+}^T(I, a)$.

Player i will select action $a \in A(I)$ according to a probability distribution over actions in an infoset with probability of each action proportional to the positive regret on it (regret-matching):

$$\sigma_i^{t+1}(I, a) = \begin{cases} \frac{R_{i,+}^T(I,a)}{\sum_{a' \in A(I)} R_{i,+}^T(I,a')}, & \text{if } \sum_{a' \in A(I)} R_{i,+}^T(I, a') > 0 \\ \frac{1}{|A(I)|}, & \text{otherwise} \end{cases} \tag{5}$$

The *average overall regret* for player i on iteration T is:

$$R_i^T = \frac{1}{T} \max_{\sigma_i^* \in \sum_i} \sum_{t=1}^{T} (u_i(\sigma_i^*, \sigma_{-i}^t) - u_i(\sigma^t)) \tag{6}$$

And the *average strategy* for player i on iteration T in the information set I is:

$$\bar{\sigma}_i^T(I) = \frac{\sum_{t=1}^{T} \pi_i^{\sigma^t}(I)\sigma_i^t(I)}{\sum_{t=1}^{T} \pi_i^{\sigma^t}(I)} \tag{7}$$

Marc Lanctot et al.(2009)[7] presented Outcome-Sampling and External-Sampling as two refined sampling schemes of Monte Carlo Counterfactual Regret Minimization (MCCFR). They first prove that in spite of different regret updating algorithms, MCCFR remains unchanged as CFR in the view of expectation of regret value then show the overall regret of both sampling schemes are bounded. Hence, the improved algorithm can bring a faster convergence in various games.

MCCFR restricts the number of terminal histories we need to consider on every iteration. Generally, there is a set of subsets of terminal histories, $\mathcal{Q} = \{Q_1, \ldots, Q_r\}$ that spans $\mathcal{Z}$. On each iteration, the algorithm will sample one of the subsets (also referred as blocks), and focus on the terminal histories in that subset.

The difference between the two sampling schemes is the number of terminal histories in one such block, which we have to update the regret and strategy of. For Outcome-Sampling, this number is one ($\forall Q \in \mathcal{Q}, |Q| = 1$) so we only choose a single terminal history and only update information along that history. For each visited information set, the sampled counterfactual regret is:

$$\tilde{r}(I,a) = \begin{cases} \omega_I \cdot (1 - \sigma(a \mid z[I])), & \text{if } (z[I] \cdot a) \sqsubseteq z \\ -\omega_I \cdot \sigma(a \mid z[I]), & \text{otherwise} \end{cases} \tag{8}$$

where

$$\omega_I = \frac{u_i(z)\pi^{\sigma}_{-i}(z) \cdot \pi^{\sigma}_i(z[I] \cdot a, z)}{\pi^{\sigma'}(z)}.$$

While the External one only samples the chance nodes and nodes where the opponents choose actions yet considers all the possible actions of the player himself. In this case, the counterfactual regret of each of the information sets which have been visited is:

$$\sum_{z \in Q \bigcap Z_I} u_i(z)(\pi^{\sigma}_i(z[I] \cdot a, z) - \pi^{\sigma}_i(z[I], z)) \tag{9}$$

2.4 Best-Response Pruning

Pruning allows us to skip some parts of the game tree in the CFR conduction. A basic pruning technique called *Partial Pruning* allows the history h with $\pi^{\sigma}_{-i}(I) = 0$ to be ignored. From equation (1) and (2) the strategy at such history contributes nothing to the corresponding regret of I and the infosets beyond I. Hence there is no need to traverse the paths that our opponent reaches with zero probability. An improved pruning called *Regret-Based Pruning* allow the paths where the *traverser* reaches with zero probability to be temporarily pruned. Formally, Regret-Based Pruning allows one to skip $D(I,a)$ as long as $\sigma^t(I,a) = 0$.

Best-Response Pruning (BRP) will prune an action when even we follow the Counterfactual Best Response strategy (CBR), we still do worse than what has been achieved at the current stage. The pruning will continue until our opponent's average strategy change so sufficiently that the condition we just claimed no longer holds. When the pruning ends, a CBR is computed against the average strategy of our opponent so far and the regrets of pruned poorly-performing actions are set to be exactly same as if we played the CBR every iteration from the beginning, even some of them are before the pruning begins.

Following the definitions in study [3], define a strategy $\beta(\sigma_{-i}, T)$ as a *T-near counterfactual best response* (T-near CBR) to σ_{-i} if for all I belonging to player i:

$$\sum_{a \in A(I)} (v^{\langle \beta(\sigma_{-i}),T),\sigma_{-i}\rangle}(I,a) - v^{\langle \beta(\sigma_{-i}),T),\sigma_{-i}\rangle}(I))^2_+ \leq \frac{x_I^T}{T^2} \tag{10}$$

where x_I^T lies in the range $[0, (\Delta(I))^2 \|A(I)\| T]$. If $x_I^T = 0$, then a T-near CBR is always a CBR. We also define *T-near counterfactual best response value* as $\psi^{\sigma_{-i},T}(I) = \min_{\sigma'_i \in \Sigma^{\beta}(\sigma_{-i},T)} v^{\langle \sigma'_i, \sigma_{-i}\rangle}(I)$ and $\psi^{\sigma_{-i},T}(I,a) = \min_{\sigma'_i} v^{\langle \sigma'_i, \sigma_{-i}\rangle}(I,a)$, where $\Sigma^{\beta}(\sigma_{-i},T)$ is the set of strategies that are T-near CBRs to σ_{-i}.

Specifically, on iteration T of CFR, if:

$$T(\psi^{\bar{\sigma}_{-i},T}(I,a)) \leq \sum_{t=1}^{T} v^{\sigma^t}(I) \tag{11}$$

then $D(I,a)$ can be pruned for

$$T' = \frac{\sum_{t=1}^{T} v^{\sigma^t}(I) - \psi^{\bar{\sigma}_{-i},T}(I,a)}{U(I,a) - L(I)} \tag{12}$$

iterations. After those T' iterations, a $T+T'$-near CBR is calculated in $D(I,a)$ to the opponent's average strategy. We then suppose that $T+T'$-near CBR had been played on every iteration so far and then set the regret.

If $\pi^{\sigma^T_{-i}}(I)$ is very low, the (11) would continue to hold for even more iterations. Specifically, we can prune $D(I,a)$ from iteration t_0 to t_n as long as

$$t_0(\psi^{\bar{\sigma}_{-i},T}(I,a)) + \sum_{t_0+1}^{t_n} \pi^{\sigma^T_{-i}}(I)U(I,a) \leq \sum_{t=1}^{t_n} v^{\sigma^t}(I) \tag{13}$$

3. Experiments

3.1 Experimental Details

The game Cheat and CFR training and testing were implemented in Python language. We used a single thread of AMD Ryzen 5 1400 Quad-Core Processor Unit.

The basic setup of the game environment is two-player, half deck. By half deck, we mean from rank Ace to Rank 7, four cards of each rank so 28 cards in total. To avoid perfect-information cases and endless repetitions, we only randomly deal 6 cards to each player so only 12 cards are actually used in every game.

Our CFR agent does not need any domain-specific knowledge. The nodes in the game trees of Cheat in the view of CFR agent contain the following information: the number of cards holding by every player, the number of cards in the pile on the table and the currently holding cards of the CFR agent. We ignore the histories because of possible repetitions: for example, CFR agent is supposed to discard at the beginning of the game and it is then challenged by any other player and loses. The cards it just discarded went straightly back to its hand and the game state is the same as the very beginning. Similar situations can happen every so often and since the Rank index will be played in a loop we can make an slight abstraction by ignoring the past history in these cases. Fig. 3 is a simple representation of repetitions in the game. On the other hand, the repeated histories will bring exponential increases when storing them. We choose to use the External-Sampling scheme to run a MCCFR algorithm. In this case, we sample the chance node and the nodes where the opponent make decisions. The counterfactual regret of the visited infoset is calculated using function (9). On every iteration, we check whether the actions with very negative regrets meets the pruning starting conditions and

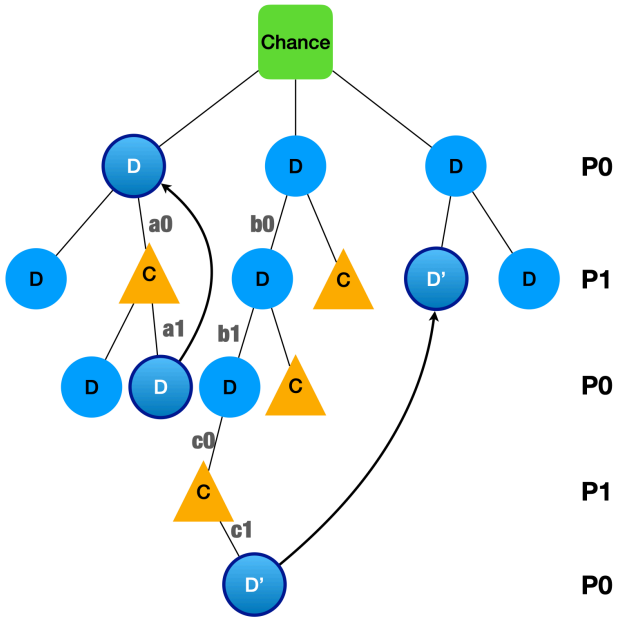

Fig. 3 An example of repetitions in the history

if so, we prune the brunches.

Except for the CFR agent, we also built two bots: Naive player and Heuristic player, to test and evaluate the performance of the algorithms. The Naive player chooses most of the actions randomly, but it can detect obvious lies for example when it is holding three fives and its opponent claims to discard two fives. It will also discard all the cards when they are all the correct rank of that round so the bot can win the challenge thus the game by doing so. The Heuristic player is more intelligent: it will memorize the cards which were once in its hands and now in the pile on the table or were revealed during the Challenge phase and now are holding by other players in the game. It will update its memory during the game play. Notice that even though the Heuristic player keeps a record of the cards that are discarded, it is still not completely perfect recall because there are turns where no one wants to challenge thus no one except the one who discarded the cards that turn can be sure about those cards. The Heuristic player is much stronger than the Naive ones, with a winning rate about 98% in both 1-vs-1 and 1-vs-multiple games but on the other hand, takes more time and memory to compute.

We compare the performance of CFR with BRP to no pruning at all over 1000 iterations of training and after all the iterations we compare the results of both CFR agents playing against Naive agent and Heuristic agent, in the view of computation time and winning rates.

3.2 Results and Conclusion

It is clear that BRP does help us accelerate the traversals of CFR on each iteration. Our External-Sampling MCCFR agents took over 280,000 seconds to run 100 iterations without any pruning, while with BRP we have shortened the time into 30,000 seconds.

In Fig. 4, two lines represent the average game values of

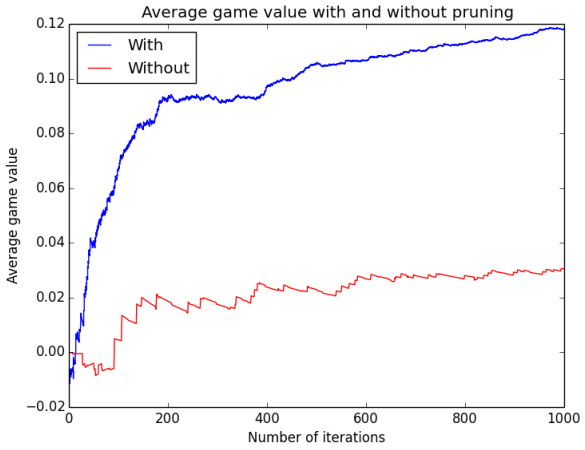

Fig. 4 Average game values for CFR training with and without pruning

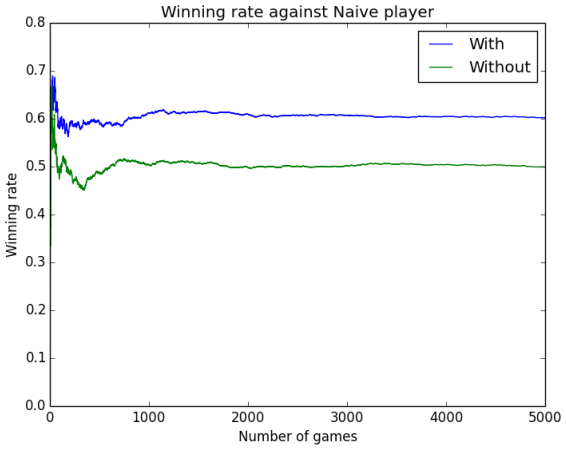

Fig. 5 Winning rates of two CFR agents playing against Naive Player

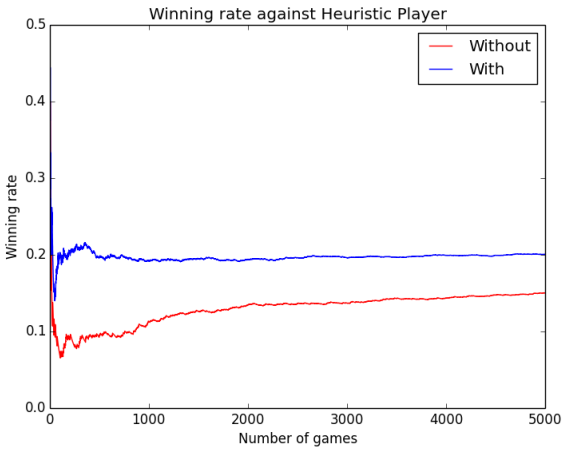

Fig. 6 Winning rates of two CFR agents playing against Heuristic Player

CFR agents with and without BRP. We can see that the blue line increases fast during the first 200 iterations of the game and the rate of increasing decreases afterward and the final average game value is about 0.12. While the red line performs rather poorly, reaching 0.02 after a steady

growth. BRP helps us speed up the convergence to the equilibrium.

Fig. 5 and Fig. 6 show the winning rate of CFR agents after training 1000 iterations by self-playing and we can see that the one with BRP performs better in both cases. The performance of agents first experienced an unstable stage and then reached stable winning rates of 60% and 20% playing against Naive player and Heuristic player respectively. Although the overall winning rates do not seem competitive against these two players, it is better than without any pruning.

4. Further Expectation

Hereby, we have shown the effectiveness of BRP in improving the performance of CFR in our simplified version of game Cheat and the reduction factor of computation time. For the next step, we will focus on improving the winning rates against players built by hands. After reaching a reasonable winning rate, we can then expend the game environments by using the whole deck and increasing the number of cards actually dealt to each player in games. We can also partition the information sets based on the number of cards in one player's hand, and calculate the regrets for information sets with n cards by utilizing the result from information sets with $n-1$ cards. As mentioned in the paper [3] the reduction factor increases with game size, we hope a larger improvement can be achieved at that time.

Although the BRP pruning helps us save time and space, its performance remains unchanged compared to the vanilla CFR at the early stage. In order to skip the early iterations we can first solve an abstracted game and then use the result to warm start the CFR. In the study [2], the researchers also mentioned that the effectiveness of warm starting is magnified by pruning. So we also want to see how large the magnitudes of speed and space reductions we can reach by combining these two methods.

For the next step, we will try to track the details of the CFR agent's behaviour. For example, with different training opponents (against Heuristic player and Naive player or self-playing), how differently the CFR agent will perform by keeping a record of the lying rate in Discard phase, the challenging rate and the challenge-and-winning rate in the Challenge phase.

References

[1] Brown, N., Lerer, A., Gross, S. and Sandholm, T.: Deep counterfactual regret minimization, *International conference on machine learning*, pp. 793–802 (2019).

[2] Brown, N. and Sandholm, T.: Strategy-based warm starting for regret minimization in games, *Thirtieth AAAI Conference on Artificial Intelligence* (2016).

[3] Brown, N. and Sandholm, T.: Reduced space and faster convergence in imperfect-information games via pruning, *International conference on machine learning*, pp. 596–604 (2017).

[4] Brown, N. and Sandholm, T.: Superhuman AI for heads-up no-limit poker: Libratus beats top professionals, *Science*, Vol. 359, No. 6374, pp. 418–424 (2018).

[5] Brown, N. and Sandholm, T.: Superhuman AI for multiplayer poker, *Science*, p. eaay2400 (2019).

[6] Gilpin, A. and Sandholm, T.: Finding equilibria in large sequential games of imperfect information, *Proceedings of the 7th ACM conference on Electronic commerce*, pp. 160–169 (2006).

[7] Lanctot, M., Waugh, K., Zinkevich, M. and Bowling, M.: Monte Carlo sampling for regret minimization in extensive games, *Advances in neural information processing systems*, pp. 1078–1086 (2009).

[8] Moravčík, M., Schmid, M., Burch, N., Lisỳ, V., Morrill, D., Bard, N., Davis, T., Waugh, K., Johanson, M. and Bowling, M.: Deepstack: Expert-level artificial intelligence in heads-up no-limit poker, *Science*, Vol. 356, No. 6337, pp. 508–513 (2017).

[9] Zinkevich, M., Johanson, M., Bowling, M. and Piccione, C.: Regret minimization in games with incomplete information, *Advances in neural information processing systems*, pp. 1729–1736 (2008).