Counterfactual Regret Minimization, Variants and its Application to Poker

Franz Srambical
Department of Computer Science
Technical University of Munich
85748 Garching, Bavaria

Abstract

All major bots of the poker variant *Texas Hold'em* of the last couple of years are fundamentally based on *counterfactual regret minimization* (CFR). An overview of the core concept of CFR, its mathematical model as well as its most famous variants is given. We explain the main techniques used in *Pluribus*, a poker bot that is able to extend CFR to a poker variant of unprecedented complexity achieving superhuman performance.

Introduction

Historically, tabletop games have often been used as benchmarks by artificial intelligence (AI) and game theory researchers in order to measure the performance of approaches and algorithms in a controlled environment. In recent years, there have been great strides in the application of self-play to board games such as Chess and Go - both notably perfect information games - leading to superhuman performances (Silver et al. 2017). Another such famous challenge problem is the game of Texas Hold'em - a variant of poker - owing to its popularity, simple rules and most importantly its property of being a game of imperfect information. After years of work solving two-player (so-called *heads-up*) limit poker (Bowling et al. 2015), the next milestone was the creation of a poker bot able to beat top professionals at the game of six-player no-limit Texas Hold'em. This seemingly small step turns out to be feat of engineering due to the substantially increased game complexity.

This paper will solely focus on Texas Hold'em. As such, any further mention of *poker* refers to Texas Hold'em.

Poker and its variants

With regards to game-theoretic specification poker is a zero-sum, extensive-form game of imperfect information.

There are two main properties to distinguish between Texas Hold'em variants: The level of granularity of bet sizes as well as the number of players. The former yields the distinction between *limit* and *no-limit* poker. As the name suggests, in limit poker the bet sizes and number of raises are limited, while there are no such restrictions in no-limit poker.

Poker can technically be played with an arbitrary number of players from two to 23, with the two most famous variants being heads-up poker as well as six-player poker.

Evaluation of poker bot strength

The most common units of measurement for poker bot strength are milli bets per hand (mb/hand) (Zinkevich et al. 2007) as well as milli big blinds per game (mbb/game) (Moravčík et al. 2017). Both of them are commonly used as units of *exploitability*: a poker bots performance against its worst-case opponent. A poker bot tries to minimize exploitability and maximize its winning margin.

Superhuman performance in heads-up poker

Bowling et al. have been able to *essentially weakly solve* heads-up limit poker - a game of approximately 10^{18} game states (Billings et al. 2003) -, where *essentially weakly solved* means that an ϵ-Nash-equilibrium with a sufficiently small ϵ is computed. In an ϵ-Nash-equilibrium no player can increase their utility by more than ϵ by deviating from their strategy. The poker bot of Bowling et al. reaches an exploitability of 0.986 mbb/game yielding an ϵ that is sufficiently small in so far as a human lifetime of play is not enough to differentiate its equilibrium from a Nash equilibrium.

Three years later Brown and Sandholm created a poker bot named Libratus which beat top professionals in heads-up no-limit poker exhibiting superhuman performance. This was another milestone in poker bot research, as the no-limit poker variant comes with an increase in complexity to 10^{75} game states (Johanson 2013). We emphasize that exhibiting superhuman performance is a far weaker benchmark than weakly solving a game. This is exemplified by the fact that the exploitability of Libratus remains above 70 mbb/game, even in scaled-down variants of heads-up no-limit poker (Brown and Sandholm 2018).

Definitions

Game tree Extensive-form games can be modelled using game trees, where each node is a decision point for a particular player or a chance node. An example of a chance node is the *flop* in poker.

Information set Due to poker being a game of imperfect information, many game states become indistinguishable to the player. Each such set of game states forms one information set. The strategy of an agent must solely depend on

its information set, as an agent cannot act based on hidden information like its opponents cards (Johanson 2013).

Strategy A strategy of player i σ_i is a function that assigns a probability distribution over all legal actions at every information set.

A strategy profile σ is formed by all player strategies.

History A history is a sequence of actions which lead to a specific game state.

Utility The utility of a terminal history in the game tree is positive if chips have been gained at the terminal history or negative if chips have been lost. In its simplest form, the utility of a terminal history corresponds to the change of a players own chip count.

The utility of a strategy is $u_i(\sigma) = \sum_{h \in Z} u_i(h)\pi^\sigma(h)$ where Z is the set of terminal game histories and $\pi^\sigma(h)$ is the probability that the given terminal history is reached with the given strategy.

Regret Regret refers to the loss of utility which occurs due to a player staying on one strategy instead of switching to a better one. More formally, it is the difference between the utilities of the played strategy and the best possible strategy at a particular time t.

Counterfactual regret minimization

Counterfactual regret minimization has been known as the most promising approach for creating approximate Nash equilibrium solutions for two-player extensive-form games with imperfect information. Both of the aforementioned poker bots use a variant of CFR.

Regret matching is an adaptive procedure that leads to a correlated equilibrium if used by every player of a game (Hart and Mas-Colell 2000, Main Theorem). At each time step, an agent can either remain on its strategy or switch to another strategy. The probability of switching to another strategy is proportional to the regret of not having always played the alternative strategy. This leads to the agents regrets converging to zero (Hart and Mas-Colell 2000, Theorem A).

Counterfactual value $v_i(\sigma, h)$ describes the utility an agent would have received, had he played strategy σ at history h: $v_i(\sigma, h) = \sum_{z \in Z, h \sqsubset z} \pi^\sigma_{-i}(h)\pi^\sigma(h, z)u_i(z)$, where Z is the set of terminal histories, $\pi^\sigma_{-i}(h)$ is the probability that the other players strategies (and chance) lead to history h, and $\pi^\sigma(h, z)$ is the probability of going from history h to history z. $h \sqsubset z$ means that h is a proper prefix of z, which means that only those terminal game states are considered which pass through the game state at history h (Johanson et al. 2012).

Counterfactual regret of not taking action a at history h is defined as: $r(h, a) = v_i(\sigma_{I \to a}, h) - v_i(\sigma, h)$, where $\sigma_{I \to a}$ denotes the strategy profile σ with the modification that action a is played at information set I.

The counterfactual regret of not taking action a at information set I is defined as: $r(I, a) = \sum_{h \in I} r(h, a)$.

The cumulative counterfactual regret is then: $R_i^T(I, a) = \sum_{t=1}^T r_i^t(I, a)$, where $r_i^t(I, a)$ denotes the regret of players not choosing action a at information set I of player i at the time t (Zinkevich et al. 2007). $R_i^{T,+}$ is used as a notation for non-negative cumulative counterfactual regret.

Strategy updates To obtain an updated strategy for the time $T + 1$, we now use the regret matching algorithm of Hart and Mas-Colell with non-negative cumulative counterfactual regret. This yields the formula:

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

The entire game tree is traversed and for each information set I, the probability of each legal action in this information set is adapted to be proportional to the non-negative cumulative counterfactual regret of not having chosen that action. The player is able to precisely calculate the regret of not having chosen an action, because the player is playing against itself (or older copies of itself) so that a game with alternative decisions can just be simulated.

Minimization To minimize counterfactual regret, two or more players repeatedly play the game using the aforementioned strategy updating rule. After T iterations, the returned approximated equilibrium is $(\overline{\sigma}_1^T, \overline{\sigma}_2^T, \ldots, \overline{\sigma}_n^T)$, where $\overline{\sigma}_i^T$ denotes the average strategy of player i from time 1 to T. This average strategy is the computed approximate Nash equilibrium. Note that this requires storing the regrets of all actions and information sets as well as updating them after each iteration.

Variants of CFR

Vanilla CFR This is the an unoptimized version of CFR, where the entire game tree is traversed at every iteration and no sampling is used.

CFR$^+$ replaces regret matching with so-called *regret matching$^+$*, where cumulative counterfactual regret is replaced by *cumulative counterfactual regret$^+$*:

$$R_i^{+,T}(I, a) = max\{R_i^T(I, a), 0\}$$

where

$$R_i^T(I, a) = \begin{cases} R_i^{+,T-1}(I, a) + v_i(\sigma_{I \to a}^T, I) - v_i(\sigma^T, I), & T > 1 \\ v_i(\sigma_{I \to a}^T, I) - v_i(\sigma^T, I), & T = 1 \end{cases}$$

In CFR$^+$ it is the current strategy that *almost* converges or converges to a Nash equilibrium, which means that the averaging step could be skipped, however CFR$^+$ uses *weighted averaging* so that the strategies converge faster. The weight sequence that CFR$^+$ uses is $w^T = max\{T - d, 0\}$, where d is the averaging delay in number of iterations (Tammelin 2014).

Monte Carlo CFR In Monte Carlo CFR (MCCFR), instead of traversing the entire game tree on every iteration, we partition the set of terminal histories Z into blocks $\{Q_1, \ldots, Q_r\}$. On each iteration, one of these blocks is sampled and only the terminal histories of the sampled block are considered (Lanctot et al. 2009).

Chance-sampled CFR (CSCFR) is MCCFR where each block contains all terminal histories with the same sequence of chance outcomes. This works in games where the probability of a chance outcome is independent of players' decisions, as is the case in poker.

Outcome-sampling MCCFR In this variant of MCCFR each block contains a single terminal history. As such, on each iteration a single terminal history is sampled and only information sets along that history are updated.

External-sampling MCCFR In external-sampling we have a block for every strategy of the opponents and chance. This means that a block exists for every opponent-strategy-chance-combination.

Superhuman performance in six-player poker

A player playing a Nash equilibrium strategy in two-player zero-sum games is guaranteed not to loose in expectation. This is not necessarily the case in multiplayer games (Brown and Sandholm 2019). *Pluribus*, a poker bot created by Brown and Sandholm was able to consistently beat top professionals in six-player no-limit poker by using algorithms that are not even guaranteed to converge to a Nash equilibrium outside of two-player zero-sum games. Pluribus first computes a blueprint strategy offline, which is later improved upon via real-time search in live play.

Abstractions are used to reduce the complexity of the game. For one, insignificantly different possible actions are bucketed together and treated as identical. As an example, while any whole-dollar bet between \$100 and \$10 000 is allowed in Texas Hold'em, there is little difference between betting \$200 and \$201. This is called action abstraction. Additionally, similar information is bucketed together as well, which is called information abstraction. There is little to be gained differentiating between a 10-high straight and a 9-high straight.

Blueprint strategy In the first phase a *blueprint strategy* is computed using self-play with a form of MCCFR. Brown and Sandholm use several techniques to improve the computational efficiency:

As the difference between the counterfactual values gets added to the counterfactual regret in Pluribus' form of MCCFR, early strategy iterations, which are close to random, influence iterations far into the future. To combat this, Linear CFR discounts regrets based on the number of iterations that have passed. However, Linear CFR is only used in early iterations by Pluribus as Brown and Sandholm have empirically shown that the cost outweighs the benefit after a certain point.

Another one of the used optimization techniques is *pruning*, where actions with very high negative regret are not explored in 95% of iterations.

Live play During live play, Pluribus starts with its blueprint strategy. In later rounds a form of real-time search is used by looking ahead a couple of moves until the depth limit of the algorithms look-ahead is reached. During real-time search abstractions are removed to combat any abstraction-caused weaknesses. The value of the board configuration at the leaf node of the look-ahead is calculated by simulating the game with all players choosing between $k = 4$ strategies: the blueprint strategy and three variations of it biased towards folding, calling and raising. This results in a strategy that cannot be as easily exploited by opponents that switch strategies. The game simulation is done via Linear MCCFR if the subgame is relatively large, or Linear CSCFR otherwise.

To remain unpredictable to its opponents Pluribus reviews what it would have done in the current situation, had it had any other possible hand, and adjusts its strategy accordingly.

Evaluation Pluribus played 10 000 hands of six-player no-limit poker against top professionals, all of whom have won more than \$1 million playing poker professionally, and won an average of 48 mbb/game.

Conclusions

Six-player no-limit poker is surely not solved yet. Nonetheless, the emergence of a superhuman bot for six-player poker is a remarkable achievement due to the complexity increase from two-player poker. Game-theoretic techniques that are only proven leading to an *unbeatable* bot in two-player poker have still been able to consistently beat top professionals in six-player poker. This begs the question of whether game-theoretical performance bounds are an appropriate measure of algorithmic performance in scientific challenge problems.

References

Billings, D.; Burch, N.; Davidson, A.; Holte, R.; Schaeffer, J.; Schauenberg, T.; and Szafron, D. 2003. Approximating Game-Theoretic Optimal Strategies for Full-scale Poker. 661–668.

Bowling, M.; Burch, N.; Johanson, M.; and Tammelin, O. 2015. Heads-up limit hold'em poker is solved. *Science* 347(6218):145–149. Publisher: American Association for the Advancement of Science.

Brown, N., and Sandholm, T. 2018. Superhuman AI for heads-up no-limit poker: Libratus beats top professionals. *Science* 359(6374):418–424. Publisher: American Association for the Advancement of Science.

Brown, N., and Sandholm, T. 2019. Superhuman AI for multiplayer poker. *Science* 365(6456):885–890. Publisher: American Association for the Advancement of Science.

Hart, S., and Mas-Colell, A. 2000. A Simple Adaptive Procedure Leading to Correlated Equilibrium. *Econometrica* 68(5):1127–1150. Publisher: [Wiley, Econometric Society].

Johanson, M.; Bard, N.; Lanctot, M.; Gibson, R.; and Bowling, M. 2012. Efficient Nash equilibrium approximation through Monte Carlo counterfactual regret minimization. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems - Volume 2*, AAMAS '12, 837–846. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems.

Johanson, M. 2013. Measuring the Size of Large No-Limit Poker Games. Technical Report arXiv:1302.7008, arXiv. arXiv:1302.7008 [cs] type: article.

Lanctot, M.; Waugh, K.; Zinkevich, M.; and Bowling, M. 2009. Monte Carlo Sampling for Regret Minimization in Extensive Games. In *Advances in Neural Information Processing Systems*, volume 22. Curran Associates, Inc.

Moravčík, M.; Schmid, M.; Burch, N.; Lisý, V.; 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* 356(6337):508–513. Publisher: American Association for the Advancement of Science.

Silver, D.; Hubert, T.; Schrittwieser, J.; Antonoglou, I.; Lai, M.; Guez, A.; Lanctot, M.; Sifre, L.; Kumaran, D.; Graepel, T.; Lillicrap, T.; Simonyan, K.; and Hassabis, D. 2017. Mastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm. Technical Report arXiv:1712.01815, arXiv. arXiv:1712.01815 [cs] type: article.

Tammelin, O. 2014. Solving Large Imperfect Information Games Using CFR+. Technical Report arXiv:1407.5042, arXiv. arXiv:1407.5042 [cs] type: article.

Zinkevich, M.; Johanson, M.; Bowling, M.; and Piccione, C. 2007. Regret Minimization in Games with Incomplete Information. In *Advances in Neural Information Processing Systems*, volume 20. Curran Associates, Inc.