CIS 6200 Final Research Project:
Understanding Poker Abstractions Using Counterfactual Regret Minimization

Abhinav Basvoju and Andrew Yeh

1. Introduction

In the recent decades, researchers have studied imperfect-information games to better understand how strategic interactions between two players can unfold. We see the application of this research in fields such as cybersecurity planning, negotiating, voting, auctioning, where information asymmetry is the premise and agents have only partial information to make their decisions. In these games, instead of machine learning or deep learning that can solve static problems, players turn to game theory and use algorithms to search for Nash equilibrium where neither player can improve their position by deviating from a particular strategy.

We investigate a specific class of algorithms for extensive form, imperfect-information games called Counterfactual Regret Minimization (CFR), which converges to Nash equilibrium in two-player zero-sum games by minimizing regret across all information sets. Different forms of CFR have been applied to games in the domain of Texas Hold'em Poker and the state of the art algorithms have essentially solved poker. However, the primary issue when looking to apply CFR to a game such as poker is that the game itself is extremely large and requires an abstraction, a simplification of the original game, that will still help us understand the effectiveness of the algorithm.

There have typically been two types of attempts at developing a CFR algorithm to find a Nash equilibrium in the game of Texas Hold'em Poker. Firstly, there have been efforts to run the algorithm on large-scale abstractions of poker or even on the game itself. While we see results that inform us how effective the algorithm is at minimizing counterfactual regret (and thus overall regret) [12] on an incredibly large number of information sets, there is ultimately a black-box issue with this research where we struggle to see how and why the algorithm decides to converge to the equilibrium that it does. In addition to this obscurity, the time, space, and computational power required to run a CFR algorithm on data of this size is extremely large and difficult to process. Code and algorithms in this field are sparse, hindering future research efforts, and the black box nature of the algorithms and the sheer size of the problem makes it difficult for even the researchers themselves to interpret. We are interested in studying how strategy is learned with respect to CFR's process by changing various parameters in a simpler setting.

Secondly, on the other hand, there have been efforts to run algorithms on overly simplified versions of poker. For instance, a common case of this is using a variant of poker known as Kuhn Poker where two players are each dealt one of three cards and then required to bet based on their knowledge of the opponent's past behavior, their own card, and their guess of their opponent's card. While the game itself is transparent and clear, the number of information sets is far too small and does not necessarily enable us to observe how strategy will be changed when different or more parameters are involved (e.g., more than three cards in the total deck, more than one card in the hand of the player, etc.). Another major issue is that it is not clear at all that findings in sweeping abstractions like Kuhn Poker or Leduc Hold'em will generalize to real poker. We address this issue by making an abstraction with tunable parameters that govern the abstraction's

complexity. This allows us to study the relationship between the abstraction complexity and performance, giving us more confidence that findings in abstractions generalize to the larger problem.

Our work will look to find a middle ground between these two approaches to understanding poker and CFR. We seek to bridge the gap between the extensive form of Texas Hold'em Poker and Kuhn poker while providing empirical evidence that variations in parameters such as granularity and strategy depth can affect poker strategy. Over the last two years, theoretical breakthroughs have finally been made that tie abstraction choices to solution quality in the original game [10]. This motivated us to empirically study the effect of an abstraction we have developed ourselves on solution quality and poker strategy.

We plan to develop our own abstraction that will simultaneously help us understand poker strategy using knowledge gained from simple versions of poker while simplifying unnecessary complexities within the original game. We develop a two-player version of poker called Abstract Poker that fulfills these particular requirements. At a high level, Abstract Poker removes some nuances from the original game of poker such as the exact cards that players hold and replaces them with Hand Strength Values (HSVs). Players are both assigned a single HSV between 1 and $\binom{52}{2}$, or 1326 (representing the total number of possible combinations of cards that a player dealt two cards at the beginning of a game of Texas Hold'em Poker could be assigned from a standard playing card deck). Effectively, each combination of cards is mapped to a HSV; we are not concerned with necessarily how or why exactly the mappings occur, but since certain hands are probabilistically better than others, this abstraction is easy to understand and does not take away from the inherent strategy involved in poker.

As we will explain later, we evaluate the game based on these HSVs rather than the potential for particular cards to be used in different hands, keeping the fundamental goal of holding the cards that would offer the highest chance of success intact, where the closer the HSV is to 1326, the better and the closer the HSV is to 1, the worse. In Abstract Poker, we also introduce a "shock factor" that is our abstraction of the event when a dealer reveals cards in a deck to all players. This shock factor will either add or subtract from each player's HSVs and influence each player's decision to bet going forward. However, in our implementation, we do not implement the shock factor event since it would only serve as noise–shock terms must be expectation 0 since HSVs encode all possible shocks into an expected value. Ultimately, the goal of Abstract Poker is to isolate the strategy of poker and abstract away from the institutional rules of poker. This will then make it relatively easy to understand why and how a poker bot might make the decisions that it would.

We will be testing a poker bot that runs CFR on Abstract Poker while manipulating different parameters to understand how the algorithm works and adapts. There are two main variables we are interested in studying: granularity and depth.

Granularity pertains to how the bot classifies and organizes the HSVs when deciding whether to bet or not. For instance, the bot may decide to separate the entire range of HSVs into terciles, where the first third of the HSVs might instead be assigned a "transformed HSV" (THSV) of 1, the second tercile with an NHSV of 2, and the third tercile an THSV of 3. This effectively

reduces Abstract Poker to a game of Kuhn Poker. The poker bot would then make decisions based on the THSVs rather than the original HSVs. However, should the range be reduced to quintiles, deciles, or left unmodified, we can then observe the amount of regret involved with respect to the Nash equilibria for these games.

Depth is relevant to our desire to study how the CFR algorithm develops over time, considering whether or not the algorithm takes into account previous actions taken by players. Abstract Poker can be played over multiple rounds of betting (although the game may not necessarily last the maximum number of rounds depending on each player's actions), so we are interested in studying how regret will be affected when (1) the bot does keeps track of prior round(s) of betting (a "dynamic playstyle"); or (2) the bot plays each round independently, with no memory of any previous rounds of betting (a "static playstyle").

Granularity and depth will help us study how regret is affected by determining to what extent artificially limiting action spaces through grouping similar HSVs either help or hinder the algorithm's learning process and how might recalling past action spaces contribute or detract from the algorithm's learning abilities with respect to the size and complexity of previous information sets.

Our goal is to combine Abstract Poker, our unique and strategically equivalent abstraction of Texas Hold'em Poker, and CFR to contribute to the field of research being conducted to learn how poker strategy is adapted and learned by CFR algorithms in imperfect-information games. Additionally, we contribute to research tying abstraction quality to solution quality and enable poker enthusiasts to "peek inside the black-box" of CFR-trained poker algorithms.

2. Related Work

CFR has become the most reliable algorithm for solving large imperfect-information games, converging to equilibrium by traversing game trees iteratively. The algorithm has seen several use cases, especially in voting systems, negotiations, cybersecurity infrastructure, and auction theory. As a result, there has been a plethora of research conducted using Texas Hold'em poker as the primary case study to identify the efficiency and effectiveness of CFR algorithms.

In the process of conducting these studies, there have been various incremental steps and tangents that have led to our current study. Firstly, with respect to identifying how large these poker games can be, research has shown that in the case of large no-limit poker games, the game is incredibly far more complex, making the simple measurement of the size of the game difficult with respect to accounting for all possible game states (around 10^{75}), information sets , and action spaces [6]. As a result, researchers in the domain of poker sought to create abstractions to apply algorithms in a more efficient and meaningful manner, looking to specifically create an information abstraction and action abstraction that is strategically similar to the original game [10].

While abstractions continued to be developed, researchers applied regret minimization algorithms to abstractions of Texas Hold'em Poker and introduced counterfactual regret as a

measure of the degree of incomplete information in an extensive game, proving that minimizing counterfactual regret minimizes overall regret, and can thus, via self-play, can compute a Nash equilibrium [12]. Other algorithms were also developed for similar problems, specifically in the case of online structured learning, where both statistically and computationally efficient methods such as variants of Follow the Perturbed Leader (FTPL) [5] proved to be very useful as a benchmark for CFR algorithms.

The rise of various new algorithms for imperfect-information games paved the way for the use of deep learning, reinforcement learning, and artificial intelligence to be incorporated into these CFR methodologies. Specifically, the tests for these new algorithms hinged on seeing their success in Texas Hold'em Poker. For instance, in 2020, the DREAM algorithm (Deep Regret minimization with Advantage baselines and Model-free learning) showed that in two-player zero-sum games, the deep reinforcement learning algorithm could find optimal strategies and converge to a Nash equilibrium [11]. The algorithm empirically reached state-of-the-art performance and even competed with algorithms that used a perfect simulator. Other algorithms were developed, such as Libratus, hailed as a "Superhuman AI for No-Limit Poker" for defeating top human professionals in the game [4]. Libratus developed a new nested subgame-solving algorithm that constructed a more detailed strategy as play progressed. Where recreational games like Texas Hold'em Poker have been considered a benchmark to evaluate the progress of various algorithms, these CFR variants have effectively pushed AI even further.

Researchers continued to work to generalize the field past the initial goal of simply finding a Nash equilibrium in limited versions of poker into beating other bots and learning more efficiently. Another variant of CFR, known as Deep Counterfactual Regret Minimization no longer even uses abstractions by instead utilizing deep neural networks to learn an encoding (abstraction) of poker for CFR in the full game of Texas Hold'em Poker [1]. Other research has been developed to leverage AI to build new algorithms, e.g. Pluribus, to outperform human professionals in not just two-player games but also multiagent imperfect-information games such as no-limit Texas Hold'em Poker [2]. In addition to obviating the need for abstractions and increasing player count, researchers have also begun to focus on not just solving large-scale problems or accelerating convergence speed, but also the "generalization ability" of the CFR method, using Reinforcement Learning CFR that utilizes iterative strategy updating to conduct regret updates [8].

Over the past decade, there have been various improvements to CFR, furthering the frontier for optimization efficiencies in Nash-equilibrium-solving algorithms via abstractions, machine learning, and other methodologies.

3. Methodology

3.1 Abstract Poker

We have developed an abstraction of the extensive form of Texas Hold'em Poker which we have called Abstract Poker. The goal of Abstract Poker is to bridge the gap between the original game and an overly simplified variant, such as Kuhn Poker or Leduc Hold'em. From a big picture

standpoint, a poker bot must estimate (1) its own hand strength, (2) the opponent's hand strength, (3) given both of these estimations, how strong each hand could get as new cards are revealed by the dealer, and then (4) make a decision on whether to bet based on those factors. From this standpoint, only (4) and (2) are interesting and relevant to poker strategy. However, training time usually includes spending effort on (1) and (3) as well, which are static problems that are better left to other algorithms, such as Bayesian optimization.

We propose Abstract Poker to allow bots to focus on learning (2) and (4). We will begin our methodology by explaining the rules of Abstract Poker.

Abstract Poker is a multi-round, two-player zero-sum game where the premise is that each player determines the strength of his potential hand based on the two cards he privately holds out of a standard playing card deck of 52 cards, similar to the original game. However, in the original game, hands are encoded as a combination of "aces", "face cards," "numbered cards," and it is up to the player (or bot) to understand the rules of poker and be able to map the raw input of his hand to a hand strength value. This strength is based on the probability that the final hand the player possesses is a higher ranking hand than their opponent's. We abstract away from this process of calculating the potential strength of a hand in Abstract Poker and instead directly assign a hand strength ranking. Since there are a total of $\binom{52}{2}$, or 1326 unique hands that a player could have, we assign a Hand Strength Value (HSV) from 1 to 1326 which is the hand that the player holds. Higher HSVs are considered better hands. This abstraction simplifies the original game of poker significantly by abstracting away from the rules of poker without removing any of the strategy. For example, pocket Aces is the strongest hand in the game - instead of having the poker bot learn that through trial and error, pocket Aces before any other cards are revealed will immediately map to an HSV of 1326 in Abstract Poker.

We are not concerned with determining why a particular hand might be mapped to a particular HSV as this is not relevant to the strategy we would like to observe from the poker bot. Then, instead of dealing a hand at the beginning of the game, both players are assigned an HSV that only they know at the beginning of Abstract Poker to proxy for the expected strength of their hand after all three dealer cards are revealed.

Play in Abstract Poker is similar to the original game, where after HSVs are dealt, the first player (P1) can either bet or check based on their assessment of their HSV and their opponent's HSV. Should a player decide to bet, the other player can either fold, ending the game and surrendering the betting pool ("pot"); or he can call, increasing the total size of the pot and beginning the next round. On the other hand, if P1 starts the round by checking, P2 can either check back, which would begin the second round, or bet. Checking is similar to "passing" in that it passes the turn but does not increase the size of the pot. Players cannot check in response to a bet.

Play in subsequent rounds is identical to the first round except that if it is the final turn, the game instead ends and both players reveal their hands (HSVs) to each other. The player with the higher HSV will win the pot. Betting is also simplified in Abstract Poker. Players bet a fixed amount every time they choose to bet or call. We determine this fixed amount to be one unit. The ante, or buy-in, for both players is also a fixed one unit though we experiment with different ante values.

When the game ends, the winning player will win both players' antes and any bets and calls added to the pot.

We have designed the game in such a way to create an analogy between the original Texas Hold'em Poker to observe poker strategy adaptation by the bot.

3.2 Shock Factor

While we do not implement the "shock factor" in this experiment since it does not necessarily change the expected values of the final outcomes, it was a point of research that we would like to further study later. However, the implementation of the shock factor is as follows:

If the next round begins, there is a shock factor introduced to each player's HSV. This shock factor corresponds to the dealer in Texas Hold'em Poker revealing a new card(s) in the deck, i.e. the flop, the turn, or the river. The card reveal may be beneficial or detrimental to the strength of each player's hand but since the reveal is completely random and HSV encodes the expected hand strength given all possible card reveals, the expected shock value is 0 by construction. The magnitude of the shock factor can be pre-determined and fixed or parametrized and encoded in a "volatility" factor. Every time the second round begins, both players are randomly assigned either a beneficial or detrimental shock factor. If it is a beneficial shock factor, that player will receive an addition to their HSV equivalent to the magnitude of the shock factor. If it is a detrimental shock factor, that player will receive a subtraction from their HSV equivalent to the magnitude of the shock factor. For instance, if P1 began with an HSV of 650 and received a detrimental shock factor of magnitude 50, the updated HSV would be 650 - 50 = 600. This updated HSV would then be used for making decisions going forward in the game tree.

3.3 Counterfactual Regret Minimization

The core idea behind CFR is to iteratively refine the strategies employed by a player by minimizing the regrets associated with past decisions. The algorithm begins by assigning initial strategies to the player at each decision point in the game. These strategies determine the probabilities with which the player will choose each action. In each iteration, CFR traverses the game tree, revisiting decision points encountered in previous plays. For each decision point, the algorithm simulates alternative actions and computes the regret, which is the difference between the actual outcome and the expected outcome if a different action had been chosen.

CFR maintains a regret sum for each action at every decision point. This regret sum represents the cumulative regrets for not having played that action more in the past. The algorithm then updates the player's strategy at each decision point based on the computed regrets. Actions associated with higher regrets are assigned lower probabilities, while actions with lower regrets receive higher probabilities. This process ensures that the player adjusts its strategy to favor actions that historically led to better outcomes.

The iterative nature of CFR allows the algorithm to learn and adapt over time. As the player repeatedly engages in the game and accumulates regret information, the strategies converge towards an equilibrium where regrets are minimized. The resulting strategies represent a

near-optimal policy for decision-making in the given game. CFR's effectiveness lies in its ability to strike a balance between exploration and exploitation. By continuously exploring alternative actions and updating strategies based on regret information, the algorithm converges to a solution that minimizes the player's overall regret and maximizes its expected utility.

Essentially, we utilize the algorithm similar to as described in [12], where CFR breaks down overall regret into a set of independently minimizable and additive regret terms (see [12] for a formal treatment). As the paper proves, overall regret is bounded by the sum of counterfactual regrets and counterfactual regret can be minimized at each information set independently. Given that we have developed our abstraction in the name of Abstract Poker, we can then use CFR to compute an approximate equilibrium for the game. The specific implementation will be further explained in the following section.

3.4 Applying CFR to Abstract Poker

We apply the counterfactual regret minimization algorithm to Abstract Poker as described above. Bots are trained through self-play over 100,000 hands. We do not implement HSV shocks which could be an area of future research. We implement two parameters of the abstraction: granularity and depth.

We implement granularity as follows. Each possible HSV is mapped into a partition of hand strength with the number of partitions corresponding to hand strength. Kuhn poker is equivalent to a granularity of 3 (which map to the Jack, Queen, and King respectively). Duplicates are allowed and result in ties in the training process. For a granularity-3 bot to play Abstract Poker, the raw HSV (between 1 and 1326) is mapped into a transformed HSV partition with 1-442 corresponding to a 1, 443-884 corresponding to a 2, and 885-1326 corresponding to a 3. Then, given their transformed HSV and any available context, the bot plays according to the strategy it learned through self-play.

We implement depth as follows. Bots are trained using a fixed number of turns in the Abstract Poker game. We call bots trained on a single turn “static” and bots trained on multiple turns “dynamic.” The advantage of dynamic bots is that their information set includes actions in prior turns as context for their final decision. Static bots make decisions based only on their HSV and any actions in the same turn.

3.5 Evaluation of the Poker Bot

Trained bots play against each other over 100,000 hands - they do not additionally learn or update from the played hands. The results of those 100,000 hands are averaged and reported as a scalar. A win for Player 1 is reported as a positive number while a win for Player 2 is reported as a negative number. Because Abstract Poker is a zero-sum game, Player 1’s winnings are Player 2’s losses and vice-versa. An overall positive number means that Player 1 is favored in the game. An overall negative number means that Player 2 is favored in the game. In Abstract Poker, Player 2 is always favored because he is not forced to reveal information by acting first. This is analogous to how, in real poker, being allowed to act last when playing “on the button” is seen as a major advantage, especially in 2-player poker.

4. Results

Figure 1 shows the competition results between two separately trained Abstract Poker bots with different granularities. Panels A and B separately report results for a static game with one turn and a dynamic game with two turns. The average result across 100,000 simulated hands played is presented as the values in the table where a negative result indicates that Player 2 wins and a positive result indicates that Player 1 wins. For example, a result of "-1" indicates that Player 2 wins one unit and Player 1 loses one unit.

Figure 1 shows that increased granularity leads to an advantage in both static and dynamic games whether the player goes first or second. As granularity increases for Player 1, the average result grows more and more positive which is visually represented by a lighter color going from top to bottom. As granularity increases for Player 2, the average result grows more and more negative which is visually represented by a darker color going from left to right.

Figure 2 shows the same conclusions but for depth of play. Bots trained on lower depth games who apply their low-depth strategies to high-depth games lose to bots trained on the higher-depth games to begin with. This may be surprising because hand strength never changes across turns in our implementation–one might initially expect there to be no difference then whether the game takes place over one turn or multiple turns. However, to the extent that the bots with greater depth can infer the opponent's hand strength from their actions, they can accumulate an advantage.

One of the main advantages of Abstract Poker and our experiments is that, unlike deep neural networks, it is relatively easy to visualize and peek into the strategy learned through CFR. For example, Figure 3 decomposes and displays one aspect of the learned Nash equilibrium strategy for the poker bots and shows how that strategy varies over different levels of granularity and depth. Specifically, it shows "aggression," which encompasses the probability of betting immediately as the first acting player ("initial bet") or the probability of responding to a check with a bet ("bet after check"). In both cases, you risk your own money and add money to the overall betting pool. Aggression pays off when the opponent folds or if the opponent has a lower HSV and calls.

In Panel A, note how the "initial bet" probabilities (blue bars) and "bet after check" probabilities (orange bars) are consistently the highest for both the weakest and strongest HSVs while being low for middle HSVs for all granularity levels. The aggressive play for the weakest HSVs and strongest HSVs complement each other: suppose, for example, that the strategy did not involve betting with weak hands. Then, when you place a bet, your opponent can infer that you have a strong hand and quickly fold. On the other hand, suppose your strategy did not involve betting with strong hands. Then, you would only win the ante after the last turn and the hands are revealed. Additionally, the bluffs with the weak HSVs would no longer work. For middle hand values, the CFR strategy prioritizes gathering information from the opponent's actions. Therefore, it is rare to aggressively bet. Note also that information is of less use if you have the weakest hand (you know you will only either lose or tie) or if you have the strongest hand (you

know you will only either win or tie). Aggression tends to increase as granularity increases and the bot has a stronger sense for whether his own hand is strong or weak.

Panel B shows aggression of the CFR strategy across different depths. Note that, as depth increases, aggression decreases. This is due to the bots with a longer horizon prioritizing information gathering in the first turn instead of building up the total pot size.

Figure 4 analyzes one source of the bot's successes when it has higher granularity or depth. The figure plots the advantage of a more granular bot or a bot with higher depth against the size of the ante. If the success of more granular (higher depth) bots stems from their ability to increase the size of the pot and win larger bots, then this plot should be relatively flat. Instead, there is a direct and highly significant correlation between the size of the ante and the size of the advantage: a 10x increase in the ante from 0.2 to 2.0 increases the advantage by around 4x for more granular bots and 5-6x for higher depth bots.

Altogether, the experimental results suggest that higher granularity and higher depth poker bots perform better, providing support for the idea that a CFR algorithm's performance on the original game can be tied to the features of the abstraction.

5. Conclusion

5.1 Connecting Results to Our Conclusion

The experimental results demonstrate a direct link between the realism of the abstraction and the quality of the bot trained under a CFR algorithm. To the extent that these results continue to generalize to more and more fine-grained abstractions, we would naturally expect world-class performance from poker bots like [2] and [9].

Additionally, we were able to investigate the optimal strategies learned under our abstraction and how these strategies change with increased granularity and depth. Finally, we decomposed the source of the outperformance of more granular, higher depth bots in terms of the size of the ante.

5.2 Primary Contributions

We have presented a few contributions to the field of poker strategy and game-theoretic algorithm development. Specifically, we sought to bridge the gap between extremely massive and extensive forms of the original game of Texas Hold'em Poker and overly simplified forms of poker such as Kuhn Poker. As a result, our research hopes to contribute to the field of poker strategy by observing how different parameters such as granularity and depth will affect algorithmic performance in imperfect-information games. This would help studies looking at how a CFR bot might behave with different strategic inputs and playstyles in an abstraction strategically equivalent to the original game. Thus, our work might contribute to generalizations to other games with incomplete information as opposed to games delineated with rules.

In addition, our other contributions to the field include providing empirical evidence on how more or less granular abstractions affect performance. This also extends to providing empirical evidence on whether implementing a dynamic or static playstyle affects performance. Moreover, our research helps provide actual working code in an applied, no-regret space while other poker bots are not tested with released code. From our research, we did not find any papers that released code other than some pseudo code.

5.3 Other Notes and Further Research

We thought it would be interesting to mention that before approaching this study, we reached out to various relevant professors studying CFR or poker bots at the University of Pennsylvania, researchers at OpenAI, and poker bot scientists across the country requesting guidance on how we might approach our research question. We had few responses and upon requesting assistance on developing a poker bot, no researchers were able to provide code, publicly available data, or substantial help in building our algorithm. It appears that scientists in the field keep their cards close to themselves, making it difficult for us to begin approaching this study. We hope that this changes in the near future as AI and no-regret learning algorithms continue to evolve and develop our approaches to imperfect-information games.[1]

With respect to future research we would be interested to explore, we would firstly look to implement our shock factor event to see how risk and variance might affect the poker bot's performance in a more extensive form of Abstract Poker. Other points of interest include expanding our work to multiagent settings with more than two players. These expansions would further expand the generalization of our research to different applications such as rank-choice voting, blind auctions, or medical treatment planning.

[1] Our code is available at https://github.com/mixerupper/abstract_poker_bot/

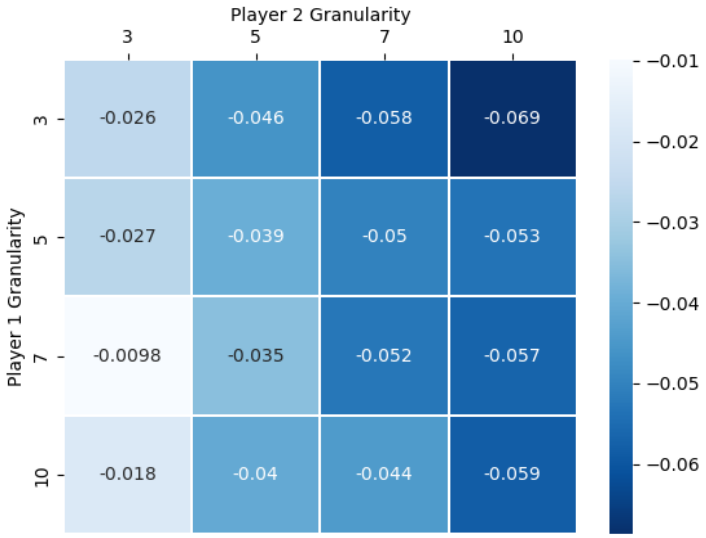
References

1. Brown, Noam, Adam Lerer, Sam Gross, and Tuomas Sandholm. “Deep Counterfactual Regret Minimization.” In *Proceedings of the 36th International Conference on Machine Learning*, 793–802. PMLR, 2019. https://proceedings.mlr.press/v97/brown19b.html.
2. Brown, Noam, and Tuomas Sandholm. “Superhuman AI for Heads-up No-Limit Poker: Libratus Beats Top Professionals.” *Science* 359, no. 6374 (January 26, 2018): 418–24. https://doi.org/10.1126/science.aao1733.
3. ———. “Superhuman AI for Multiplayer Poker.” *Science* 365, no. 6456 (August 30, 2019): 885–90. https://doi.org/10.1126/science.aay2400.
4. Brown, Noam, Tuomas Sandholm, and Strategic Machine. “Libratus: The Superhuman AI for No-Limit Poker.” In *IJCAI*, 5226–28, 2017. https://www.onlinecasinoground.nl/wp-content/uploads/2018/10/Libratus-super-human-no-limit-poker-Sandholm-Brown.pdf.
5. Cohen, Alon, and Tamir Hazan. “Following the Perturbed Leader for Online Structured Learning.” In *Proceedings of the 32nd International Conference on Machine Learning*, 1034–42. PMLR, 2015. https://proceedings.mlr.press/v37/cohena15.html.
6. Johanson, Michael. “Measuring the Size of Large No-Limit Poker Games.” arXiv, March 7, 2013. http://arxiv.org/abs/1302.7008.
7. Johanson, Michael, Kevin Waugh, Michael Bowling, and Martin Zinkevich. “Accelerating Best Response Calculation in Large Extensive Games,” n.d.
8. Li, Huale, Xuan Wang, Fengwei Jia, Yulin Wu, Jiajia Zhang, and Shuhan Qi. “RLCFR: Minimize Counterfactual Regret by Deep Reinforcement Learning.” *Expert Systems with Applications* 187 (January 1, 2022): 115953. https://doi.org/10.1016/j.eswa.2021.115953.
9. Moravčík, Matej, Martin Schmid, Neil Burch, Viliam Lisý, Dustin Morrill, Nolan Bard, Trevor Davis, Kevin Waugh, Michael Johanson, and Michael Bowling. “DeepStack: Expert-Level Artificial Intelligence in No-Limit Poker.” *Science* 356, no. 6337 (May 5, 2017): 508–13. https://doi.org/10.1126/science.aam6960.
10. Sandholm, Tuomas. “Abstraction for Solving Large Incomplete-Information Games.” *Proceedings of the AAAI Conference on Artificial Intelligence* 29, no. 1 (March 4, 2015). https://doi.org/10.1609/aaai.v29i1.9757.
11. Steinberger, Eric, Adam Lerer, and Noam Brown. “DREAM: Deep Regret Minimization with Advantage Baselines and Model-Free Learning.” arXiv, November 29, 2020. http://arxiv.org/abs/2006.10410.
12. Zinkevich, Martin, Michael Johanson, Michael Bowling, and Carmelo Piccione. “Regret Minimization in Games with Incomplete Information.” In *Advances in Neural Information Processing Systems*, Vol. 20. Curran Associates, Inc., 2007. https://proceedings.neurips.cc/paper/2007/hash/08d98638c6fcd194a4b1e6992063e944-Abstract.html.

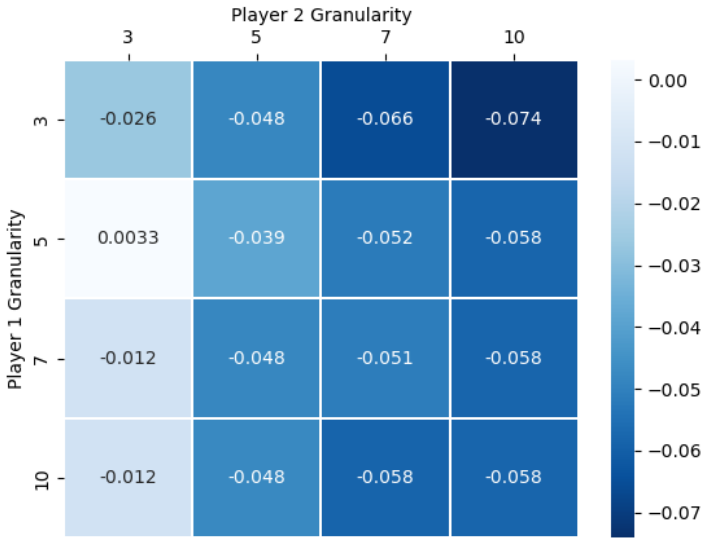
Appendix

Figure 1. Effects of Granularity on Abstract Poker Performance

Panel A. Varying granularity, static game (depth = 1)



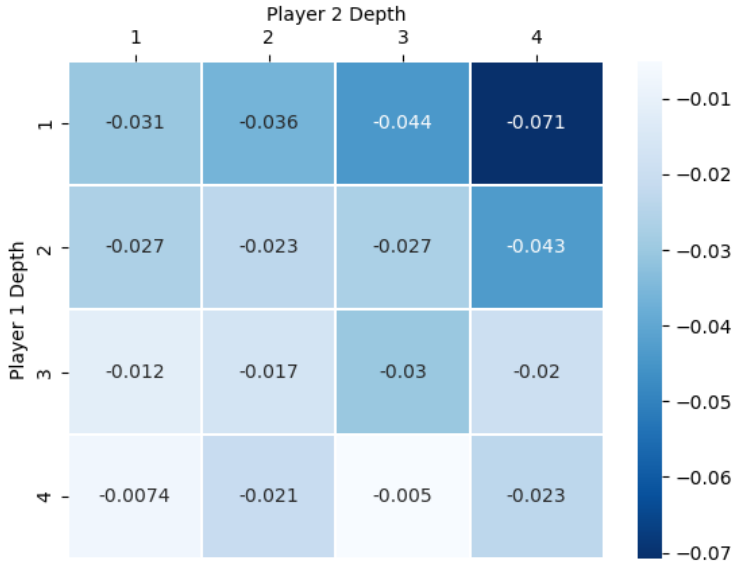
Panel B. Varying granularity, dynamic game (depth = 2)



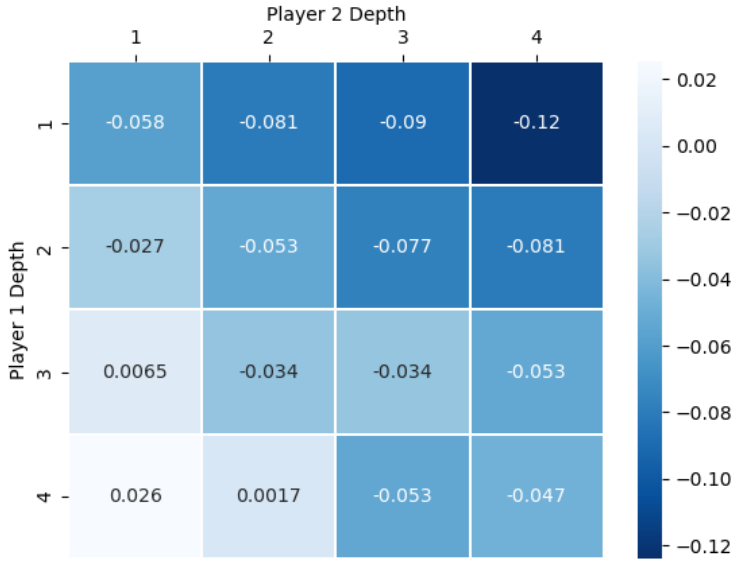
This exhibit shows the competition results between two separately trained Abstract Poker bots with different granularities for a static game with one turn (Panel A) and a dynamic game with two turns (Panel B). Bots are trained through self-play over 100,000 hands and play against each other over 100,000 hands. The result of those 100,000 hands is averaged and presented as the values in the table. A negative value represents Player 2's average winnings per hand and (as this is a zero-sum game) Player 1's average losses per hand. A positive value would represent the converse. Colors are presented by the average result value with darker colors representing a more negative average result and lighter colors representing a more positive average result.

Figure 2. Effects of Depth on Abstract Poker Performance

Panel A. Varying depth, fixed (low) granularity (granularity = 3)



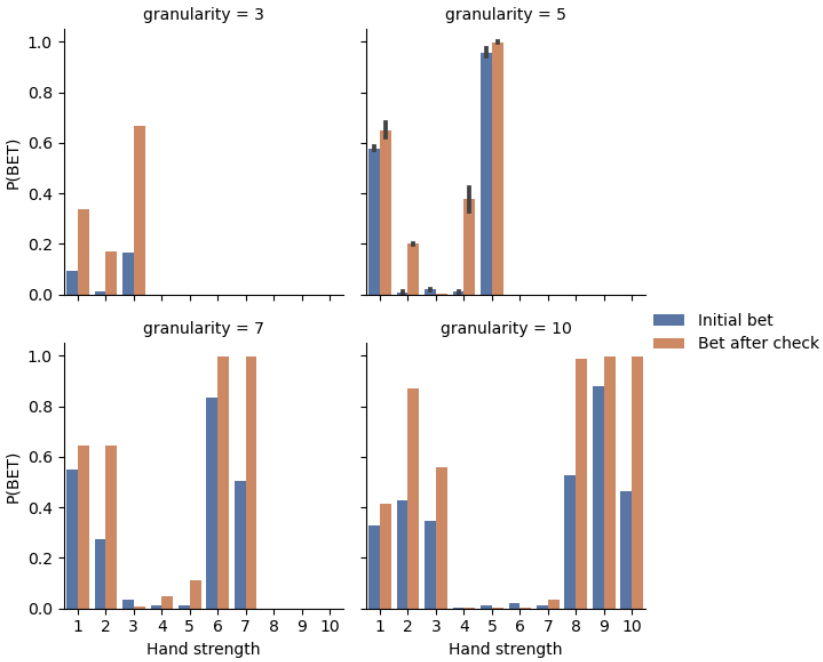
Panel B. Varying depth, fixed (high) granularity (granularity = 10)



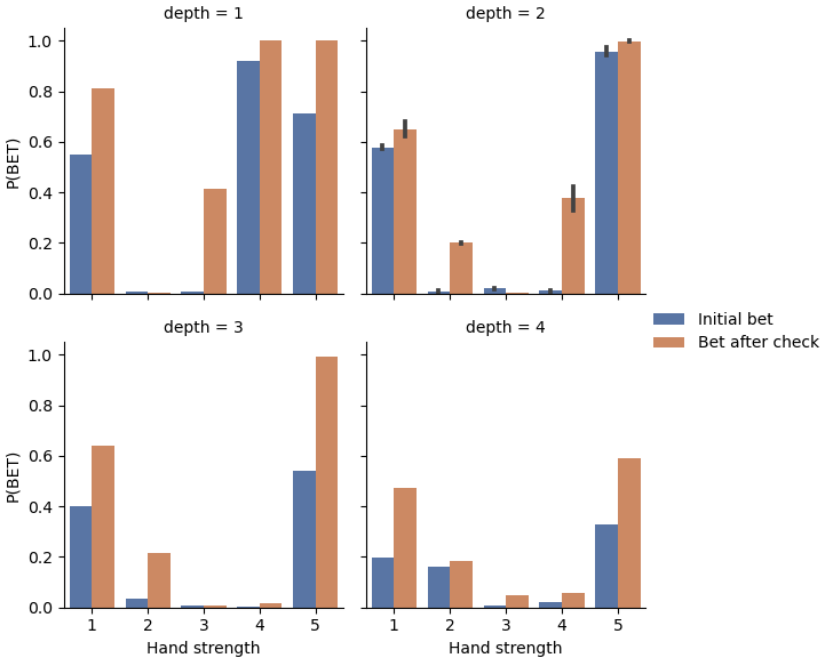
This exhibit shows the competition results between two separately trained Abstract Poker bots with different depths for bots trained with low granularity (Panel A) and high granularity (Panel B) to capture any interaction effects. Bots are trained through self-play over 100,000 hands and play against each other over 100,000 hands. The result of those 100,000 hands is averaged and presented as the values in the table. A negative value represents Player 2's average winnings per hand and (as this is a zero-sum game) Player 1's average losses per hand. A positive value would represent the converse. Colors are presented by the average result value with darker colors representing a more negative average result and lighter colors representing a more positive average result.

Figure 3. Aggression in Learned Strategies

Panel A. Varying granularity, fixed depth (depth = 2)

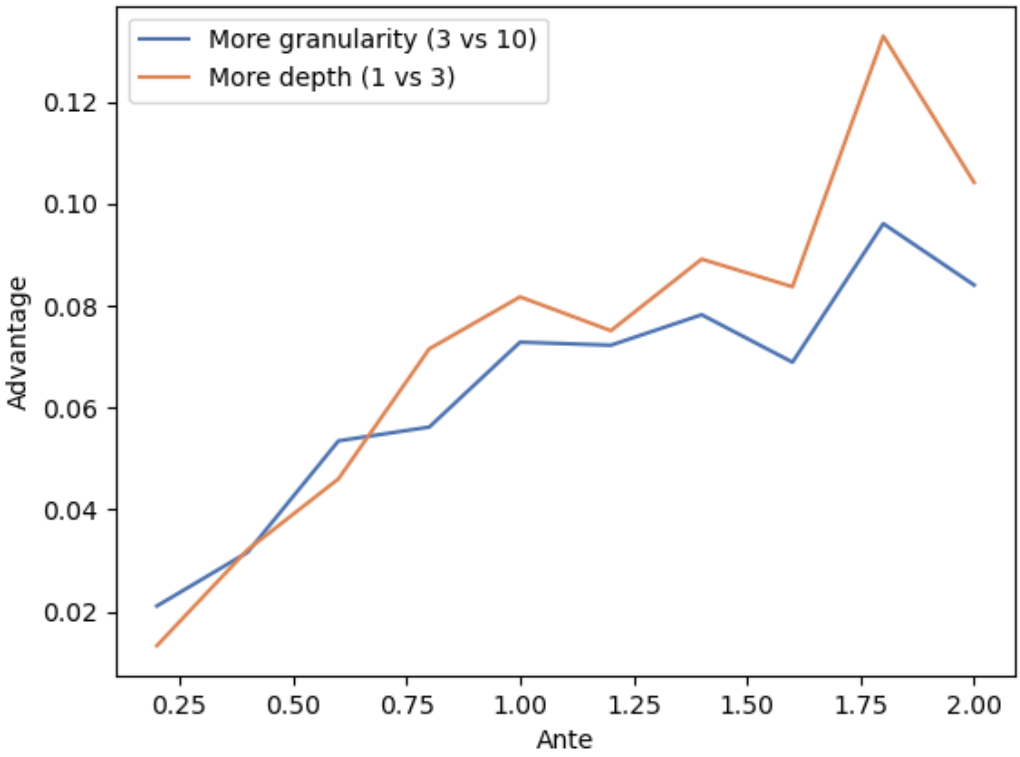


Panel B. Varying depth, fixed granularity (granularity = 5)



This exhibit shows the aggression of the learned betting strategies. Panel A reports aggression across different granularities and fixed depth of 2 turns. Panel B reports aggression across different levels of trained depth and fixed granularity of 5 partitions of HSV. The x-axis represents the hand strength of the decision-making player. The blue bars represent the likelihood of that player betting as P1 on the first turn after observing his hand strength. The orange bars represent the likelihood of betting as P2 after P1 has initiated with a check.

Figure 4. Advantages by Ante Size



This figure shows the advantage gained from higher granularity (3 vs 10) or higher depth (1 vs 3) and how those factors translate into a per-hand advantage in Abstract Poker with respect to the size of the ante. In both simulations, the bot with the higher granularity or depth is the second player. Advantage in this figure is the negative of the average result which gives the average winnings for Player 2. Ante is the initial bet required for all bots before play begins and varies from 0.1 to 2 across the x-axis. All other bets besides the ante are equal to 1. All other exhibits report results with an ante of 1 and bet size of 1.