Using Counterfactual Regret Minimization to Create Competitive Multiplayer Poker Agents

Nick Abou Risk
University of Alberta
Department of Computing Science
Edmonton, AB
780-492-5468
abourisk@cs.ualberta.ca

Duane Szafron
University of Alberta
Department of Computing Science
Edmonton, AB
780-492-5468
duane@cs.ualberta.ca

ABSTRACT
Games are used to evaluate and advance Multiagent and Artificial Intelligence techniques. Most of these games are deterministic with perfect information (e.g. Chess and Checkers). A deterministic game has no chance element and in a perfect information game, all information is visible to all players. However, many real-world scenarios with competing agents are stochastic (non-deterministic) with imperfect information. For two-player zero-sum perfect recall games, a recent technique called Counterfactual Regret Minimization (CFR) computes strategies that are provably convergent to an ε-Nash equilibrium. A Nash equilibrium strategy is useful in two-player games since it maximizes its utility against a worst-case opponent. However, for multiplayer (three or more player) games, we lose all theoretical guarantees for CFR. However, we believe that CFR-generated agents may perform well in multiplayer games. To test this hypothesis, we used this technique to create several 3-player limit Texas Hold'em poker agents and two of them placed first and second in the 3-player event of the 2009 AAAI/IJCAI Computer Poker Competition. We also demonstrate that good strategies can be obtained by grafting sets of two-player subgame strategies to a 3-player base strategy after one of the players is eliminated.

Categories and Subject Descriptors
I.2.8 [Artificial Intelligence] Problem Solving, Control Methods, and Search.

General Terms
Algorithms, Economics, Performance.

Keywords
Abstraction, Game Theory, Equilibrium, Computer Poker, Counterfactual Regret Minimization.

1. INTRODUCTION
An *extensive game* [13] models interactions between multiple autonomous agents by describing sequential decision-making scenarios, even in the context of imperfect information and non-determinism. A game tree represents the decisions. A non-terminal choice node exists for each player's possible decisions. The directed edges leaving a choice node represent the legal actions for that player. Each terminal node contains the utilities of all players for one potential game result. Stochastic events (Backgammon dice rolls or Poker cards) are represented as special nodes, for a player called the chance player. The directed edges leaving a chance node represent the possible chance outcomes.

Since extensive games can model a wide range of strategic decision-making scenarios, they have been used to study poker. Recent advances in computer poker have led to substantial advancements in solving general two player zero-sum perfect recall extensive games. Koller and Pfeffer [10] used linear programming to solve games with 10^8 game states. Competitors in the AAAI/IJCAI Computer Poker Competitions (denoted CP Competitions) [7] developed alternate techniques (gradient-based algorithms [4] [5] [6] and counterfactual regret minimization (CFR) [18]) to find near equilibrium strategy profiles for two-player, zero-sum, perfect recall games with 10^{12} game states.

Although these techniques compute near equilibrium strategy profiles for large extensive games, two-player limit Texas Hold'em (henceforth Hold'em) has 10^{18} game states [1] and two-player no-limit Hold'em has 10^{71} game states [6]. Abstraction [1] is used to reduce game size. Card abstraction combines similar hands into buckets. Betting abstraction eliminates some betting options. If a near equilibrium strategy is used to play in the un-abstracted game, then translation [6] [14] is needed to select actions in the un-abstracted game based on the abstract strategy.

Our goal is to compute winning strategies for large extensive *multiplayer games* (more than two players). Specifically we would like to derive winning strategies for 3-player limit Hold'em, where we have computed the number of game states to be 10^{24}. Some research on multiplayer extensive games has been done for a simple poker variant [4]. Ganzfried and Sandholm used techniques, involving extensions of fictitious play to compute ε-Nash equilibriums for a 3-player Poker game in [2] and [3]. They found ε-Nash equilibria for 3-player no-limit jam/fold Texas Hold'em, a one-round Poker game where each player has two options: fold or bet all remaining chips. These techniques require the ability to efficiently compute best responses. Due to the reduced betting space, the 3-player jam/fold game is fairly small and best responses can be computed efficiently. Unfortunately, even an abstracted version of the 3-player limit Hold'em game is too large to use this approach. We wish to use CFR to derive winning strategies in three-player limit Hold'em. There are two challenges. First, CFR is only guaranteed to converge to an ε-Nash equilibrium strategy profile for two-player zero-sum perfect recall games [18]. Second, even if CFR generates an ε-Nash

Cite as: Using Counterfactual Regret Minimization to Create Competitive Multiplayer Poker Agents, Nick Abou Risk and Duane Szafron, *Proc. of 9th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2010)*, van der Hoek, Kaminka, Lespérance, Luck and Sen (eds.), May, 10–14, 2010, Toronto, Canada, pp. 159-166

equilibrium strategy, there is no guarantee that this strategy will perform well in a multiplayer game. We show that CFR can compute strategies that are strong enough to win the 3-player limit Hold'em events of the 2009 CP Competition and introduce a strategy grafting approach that shows promise for helping to address the scalability issue in computing multiplayer strategies.

2. BACKGROUND

2.1 Poker

Poker is a class of multiplayer card games with many variants. Several similarities exist among the variants: they use a standard playing card deck (4 suits of 13 ranks), there are betting rounds, and the standard five-card hand ranking system is used to determine winner(s). Hold'em is the most popular poker variant.

In *Hold'em*, the *pot* is the collection of all chips wagered during a game. After each game finishes, a dealer button is moved clockwise by one position around the table. The player sitting in that position is designated the dealer or the *button*. The player to the left of the dealer is called the *small blind*. The player to the left of the small blind is called the *big blind*. There are four betting rounds. The *preflop* is the first betting round. The small blind and big blind place mandatory bets into the pot to give players an incentive to play hands. Each player then receives two private cards, face-down, called *hole cards*. The betting begins with the player to the left of the big blind.

The possible actions in every betting round are:

- *Bet* by placing a wager into the pot or *raise* by matching the outstanding bet and placing an extra bet into the pot. The minimum size of the extra bet is the difference between the two previous bet or raise sizes.
- *Call* by matching the amount of all outstanding bets or *check* by passing when there is no outstanding bet.
- *Fold* by passing when there is an outstanding bet and forfeiting the current game. A player who has folded can take no further actions in the current game.

After the pre-flop, the betting begins with the nearest active (non-folded) player to the dealer's left. The *flop* (second round) begins with three public community cards and ends with betting. The *turn* (third round) and *river* (final round) have one community card and betting. After the river betting, all active players enter a showdown where the hands – the best five card combination of each agent's two hole cards and the five community cards – are revealed and the agent with the highest ranking hand wins the pot or splits the pot if several agents have equivalent hands.

There are several variants of Hold'em in two orthogonal dimensions, number of players and betting structure. The *heads-up* variant has two players. The dealer is the small blind and acts first preflop and last postflop. The *multiplayer* variant has more than two players (usually three to ten). The betting structure in *limit* Hold'em has a fixed amount for each bet and raise, equal to the big blind amount for the preflop and flop rounds and equal to twice this amount for the turn and river. In limit, there is a cap of four bets per player per round. *No-limit* is a variant in which bets and raises can be any amount between the size of the big blind and all of a player's remaining chips. Betting all remaining chips is called going *all-in*. We focus on multiplayer limit Hold'em.

2.2 Extensive Form Games

An informal description of an *extensive game* was given in the introduction. A formal definition appears on page 200 of [13], and an excerpt appears in [14], along with formal definitions for *information set*, *strategy profile*, *Nash equilibrium*, *ε-Nash equilibrium*, and the *best response strategy* to a strategy profile. We provide more informal descriptions of these terms.

In an imperfect information game, there are game states that each agent cannot differentiate due to hidden information (opponent hole cards in poker). The set of all indistinguishable game states for a given agent is called an *information set* for that agent. Each choice node, in an imperfect information extensive game, corresponds to an information set instead of a full game state. In poker, there are many more game states than information sets.

In an extensive game, a *strategy profile* is a set of strategies, one for each player. In poker, we represent a strategy as a probability triple (f, c, r) at each information set, where f is the probability of folding, c is the probability of checking or calling, and r is the probability of betting or raising, with $f+c+r = 1$. A *best response* is a strategy that obtains the highest possible utility against the set of all other strategies in a strategy profile. A *Nash equilibrium* is a strategy profile, in which no agent can increase its utility by unilaterally changing its strategy. Each strategy from a Nash equilibrium strategy profile is a best response to the other strategies in that profile. A multiplayer Nash equilibrium does not guarantee maximum utility, regardless of what strategies are employed by other agents. If a single agent plays a strategy from an equilibrium profile and the other agents deviate, the single agent could obtain reduced or increased utility by deviating from the strategy included in the equilibrium strategy profile.

Equilibrium strategy profiles are important since for two-player zero-sum games they have additional properties. A *zero-sum game* is one in which the utility for all players sums to zero. In a two-player zero-sum game, every strategy for a given player in a Nash equilibrium strategy profile has the same utility and is a best response to all the strategies in equilibrium profiles for the other player. This means that if a player plays a strategy from one equilibrium strategy profile, the other player cannot gain utility by playing a strategy from a different equilibrium strategy profile (or any other strategy). For this reason we often refer to a strategy from a Nash equilibrium strategy profile simply as a Nash equilibrium strategy. This result is not true with more than two players [12] or if the game is not zero-sum. Computing an exact Nash equilibrium for a large extensive game such as poker is infeasible, even with extensive abstraction. Thus, we rely on finding approximations to Nash equilibria. An *ε-Nash equilibrium* is a strategy profile, in which no player can increase its utility by more than ε by unilaterally changing its strategy.

Card abstraction is the most popular way to reduce the size of the game tree. The simplest way to perform card abstraction is to apply a metric to poker hands such as hand strength (E[HS]) and to group hands that have similar metric values into the same bucket [1]. *Percentile bucketing* places an approximately equal number of hands into each bucket. Alternately, *uniform bucketing* uniformly partitions the metric interval [0, 1] into buckets. With N buckets, all hands with metric value in the [0, 1/N) range are placed in the same bucket, all hands with value in the [1/N, 2/N) range are together, and so on. In poker, there are many hands that are not very strong on a given round but have the potential of making a very strong hand on a future round (e.g. a straight or

flush). It turns out that a very effective alternative metric to E[HS] which intrinsically incorporates potential is called expected hand strength squared, $E[HS^2]$[8]. This metric, which is also in [0, 1], computes the expected value of the square of the hand strength.

A card game is transformed into a bucket game and all hands in the same bucket are played the same way. The information sets represent buckets instead of cards. If the abstraction is constructed so that all players can recall all previous actions at each information set, the abstraction is called *perfect recall*. With perfect recall, and N buckets per round, we have N bucket sequences preflop, N^2 bucket sequences on the flop, N^3 bucket sequences on the turn, and N^4 bucket sequences on the river. With a 2-bucket abstraction, this results in $2^4 = 16$ bucket sequences on the river. To reduce the size of the game tree, *imperfect recall* abstraction can be applied so that information sets from different buckets lead to common information sets in subsequent rounds. The game tree becomes a directed acyclic graph, but is reduced in size. In the extreme, the buckets of all previous rounds are forgotten and only the betting sequence and current betting round's bucket is remembered. For example, instead of a 2-bucket perfect recall abstraction, a game tree of about the same size could be created using a 16-bucket imperfect recall abstraction.

2.3 Counterfactual Regret Minimization

Regret (opportunity cost) is the difference between the highest utility attainable (over all actions) and the utility of the action that was taken. *Counterfactual regret minimization* (CFR) minimizes the positive immediate counterfactual regret at each information set. Zinkevich et al. [18] show that minimizing positive immediate counterfactual regret, minimizes the average overall regret and that in a two-player zero-sum perfect recall game, minimizing both players' average overall regret leads to an ε -Nash equilibrium strategy profile. Johanson [8] provides details on the implementation of CFR for general two-player zero-sum perfect recall games and for heads-up poker. CFR was a breakthrough for computing ε -Nash equilibrium strategy profiles for imperfect information games since the memory scales with the number of information sets instead of the number of game states, which is true for sequence-form solvers [9]. Due to these savings, Zinkevich et al. [18] were able to solve abstract poker games two orders of magnitude larger than previously solved. Another useful property of CFR is that it can compute an abstract game best response against a static opponent [8].

While the theoretical guarantees of CFR are limited to two-player zero-sum perfect recall games, CFR is quite robust in generating strong strategy profiles in two-player games when two of these restrictions (perfect recall and zero-sum) are relaxed. Some of the strongest heads-up limit Hold'em agents were created with imperfect recall. For example, Hyperborean08Limit, an imperfect recall agent, won the limit elimination (equilibrium) event in the 2008 CP Competition and placed second in the bankroll event. The imperfect recall agent, Hyperborean08NoLimit, won both the no-limit elimination and bankroll events in 2008. Strong heads-up limit Hold'em agents were also created for nonzero-sum games, with the agent obtaining bonuses for winning hands. These agents have little exploitability and actually perform better against human players (due to the agents' tendency to be aggressive and human players' tendency to fold too often). The agents that played in both of the Man-Machine competitions were computed with CFR using nonzero-sum abstract games [15].

CFR also has no theoretical guarantees for multiplayer zero-sum games and, before this research, no results had been reported on applying CFR to large multiplayer games. We show in this paper that CFR is also capable of generating winning computer poker agents for multiplayer games. CFR generated a three-player imperfect recall agent and a three-player perfect recall agent that placed first and second respectively in the 2009 CP Competition.

2.4 Equilibrium in a Multiplayer Game

Two player zero-sum games have convenient theoretical guarantees. In a two player zero-sum game, if one agent plays a strategy from a Nash equilibrium strategy profile, there is no strategy that the other agent can select that reduces the first agent's utility. However, in a multiplayer game, if one agent plays a strategy from a Nash equilibrium strategy profile, the other set of agents may play strategies that reduce the first agent's utility. Consider the following simple game:

- Three players ante 1 unit each.
- Each player has a coin and privately chooses heads or tails.
- After all three players have acted they reveal their coins.
- If all three players chose the same side, the antes are returned.
- If exactly two players have chosen the same side, they win and take back their initial ante and split the loser's ante.

All three agents choosing heads or tails with probability ½ is a Nash equilibrium strategy profile. Player i's strategy is $\sigma_i = (P(H), P(T))$ where $P(H)$ is the probability of choosing heads and $P(T)$ is the probability of choosing tails. Fix $\sigma_1 = \sigma_2 = (½, ½)$ and let $\sigma_3 = (p, 1-p)$. The utility of player 3 is $u = p \times EV(H) + (1-p) \times EV(T) = 0$. Regardless of the strategy chosen by player 3, the same utility is obtained so deviating from $\sigma_3 = (½, ½)$ does not unilaterally improve player 3's utility. The set of strategies $\sigma_1 = \sigma_2 = \sigma_3 = (½, ½)$ is a Nash equilibrium profile. Let player 3 select the strategy $\sigma_3 = (½, ½)$ and the other players change to the static strategies of always-heads, $\sigma_1 = \sigma_2 = (1, 0)$. There are two possible outcomes, HHH and HHT, with equal probabilities of ½. When player 3 selects heads, all three players take back their antes. When player 3 selects tails the ante is lost. Player 3's utility is $u = ½ \times (0) + ½ \times (-1) = -½$. However, if player 3 changes its strategy to always select heads, the utility increases to 0. Playing the strategy $\sigma_3 = (½, ½)$, which is part of a Nash equilibrium strategy profile is disastrous when the other two players both deviate from the equilibrium strategy profile. Playing an equilibrium strategy in a multiplayer game does not maximize the agent's worst case utility, as it does in the two-player zero-sum game.

3. CFR FOR MULTIPLAYER GAMES

There is no guarantee that CFR will generate a multiplayer ε-Nash equilibrium strategy profile. As illustrated by the matching-coins game, even if CFR could compute a multiplayer equilibrium strategy profile, there is no guarantee that one of the strategies would be robust against arbitrary opponents. Since CFR takes days or weeks on abstracted Hold'em games, we use small games to quickly evaluate new ideas. However, we require these small games to exhibit the properties of large poker games, such as stochasticity, hidden information, and action-based utility. Kuhn and Leduc Hold'em are small heads-up poker games that we generalized to 3-player games to help determine whether CFR is a viable option for generating robust multiplayer strategies.

3.1 3-player Kuhn Poker

Kuhn invented a simple one-round one-card poker game in 1950 so that the hand-computed optimal solution could be studied [11]. Kuhn poker includes bluffing, inducing bluffs, and value betting. In this paper, we have extended the definition of Kuhn to three players and the depth 6 betting tree has 25 game nodes:

- Each player antes 1 chip each before the cards are dealt.
- Each player is dealt one private card.
- The deck has 4 cards with ranking K > Q > J > T (no ties).
- There is one betting round with a 1-bet cap.
- If there is no outstanding bet, a player can check or bet 1 chip.
- If there is an outstanding bet, a player can fold or call.

3.2 3-Player Leduc Hold'em

Kuhn poker ignores many important aspects of real poker variants such as multi-round play, community cards, rounds with different bet sizes, split pots, and raising. Therefore, a small heads-up game, called Leduc Hold'em, which includes all of these important properties, was created [16]. Here, we have extended the game of Leduc Hold'em to three players.

- Each player antes 1 chip each before the cards are dealt.
- Each player is dealt one private card.
- The deck consists of 4 ranks and 2 suits (Td, Tc, Jd, Jc, Qd, Qc, Kd, Kc) with ranking K > Q > J > T.
- There are two betting rounds, each with a 2-bet cap and each betting round (the preflop and the flop) has a different bet size: preflop bets are 2 chips and flop bets are 4 chips.
- One community card is dealt before the flop betting.
- After the flop betting, active players showdown. The pot is split if two players tie with the best hand.
- A paired hand beats an unpaired hand, if there are no pairs, high card wins and flushes and straights do not exist.

3.3 Evaluation of 3-Player CFR Strategies

Let $\sigma = (\sigma_1, \sigma_2, \sigma_3)$ be a CFR generated 3-player strategy profile and σ_{-i} be all strategies in σ except σ_i. If σ is an ε-Nash equilibrium strategy profile, then no player, i, can gain more than ε by deviating from σ_i, while σ_{-i} remains fixed. By fixing σ_{-i}, CFR can generate a best response, σ_i^{BR}, to σ_{-i}. Doing so for every player results in a best response, $\sigma^{BR} = (\sigma_1^{BR}, \sigma_2^{BR}, \sigma_3^{BR})$, to the CFR strategy profile, σ. Here is a method to determine whether a CFR generated strategy profile is an ε-Nash equilibrium:

1. Generate a strategy profile $\sigma = (\sigma_1, \sigma_2, \sigma_3)$ using CFR.
2. Compute a best response strategy, σ_i^{BR} to σ_{-i} using CFR for each $i = 1, 2, 3$.
3. Combine the three computed best responses into a best response strategy profile, $\sigma^{BR} = (\sigma_1^{BR}, \sigma_2^{BR}, \sigma_3^{BR})$.
4. Compute the utilities for each position of the strategy profile σ by playing three σ's against each other.
5. Compute the utilities of the best response in each position by playing one σ^{BR} against two σ's.
6. Compare σ^{BR}'s utilities in each position to σ's utilities to determine how much extra BR wins in each position.
7. If the best response, σ^{BR}, does not improve by more than ε averaged over all positions then σ is an ε-Nash equilibrium.

We used 10^8 iterations of CFR (no theoretical guarantee of convergence). We found that CFR did derive an ε -Nash equilibrium strategy for 3-player Kuhn, but it did not derive an ε-Nash equilibrium strategy for the more complex 3-player Leduc. Table 1 and Table 2 show the utilities obtained by following the 7-step method for both games. CFR was used to create both the CFR strategy profile, σ and each *BR* strategy, σ_i^{BR} shown in the Tables. The utilities were then computed using a tool that walks the whole game tree applying the probability triples in each strategy at each node in the tree. The *CFR* row plays three σ strategies against each other. The *BR* row plays the best response, σ_i^{BR} against the σ_{-i} strategies. The final row is the difference between the utility of the *BR* row and the *CFR* row.

Table 1. Utilities for 3-player Kuhn (antes/hand)

	Player 1	Player 2	Player 3
CFR	-0.0544877	-0.0418710	+0.0963587
BR	-0.0543373	-0.0414071	+0.1008150
BR - CFR	+0.0001504	+0.0004639	+0.0044563

For Kuhn, the best response does not gain much utility by deviating from the CFR solution, so it is an ε-Nash equilibrium with ε = +0.0016902 (average of the final row of Table 1). This is an interesting result since there is currently no theory suggesting that CFR should produce an equilibrium for multiplayer games.

Table 2. Utilities for 3-player Leduc (antes/hand)

	Player 1	Player 2	Player 3
CFR	-0.243451	-0.1960510	+0.439502
BR	-0.124749	-0.0656892	+0.550788
BR - CFR	+0.118702	+0.1303618	+0.111286

However, CFR does not produce an ε-Nash equilibrium for Leduc (at least in 10^8 iterations). The hope of using CFR to generate ε-Nash equilibrium for more complex games has failed. It is possible that the CFR Leduc strategy is strong against two arbitrary opponents, but to verify this conjecture, we would need to play it against a suite of benchmark agents. If we are going to do that, we might as well play 3-player limit Hold'em directly since our goal is to produce strong agents for that game.

4. CFR FOR MULTIPLAYER HOLD'EM

A best response provides a metric for how much an agent could improve by unilaterally deviating from its strategy. CFR can compute an abstract game best response for heads-up limit Hold'em. However, computing an abstract game best response for a multiplayer poker game is infeasible due to its enormous size. Generating a CFR best response strategy takes N CFR runs for an

N-player game. Performing one CFR run (for a relatively low number of iterations) on a 3-player game takes months. Even if the best response computation was feasible, it would provide no indication of the strength of an agent against two arbitrary opponents. Thus, we evaluate the strength of the multiplayer agents by playing millions of poker hands against benchmark agents. This method of using tournament play (or cross-play) to rank agents is also used in the annual CP Competition.

4.1 Benchmark Agents

To evaluate an agent against other agents, millions of hands must be played between them. Relatively little research has been dedicated to multiplayer poker games so there are very few benchmark agents. *Poki* is a poker agent that is capable of playing heads-up and multiplayer games [1]. In fact, Poki won the multiplayer limit event of the 2008 CP Competition. Poki uses a collection of technologies including an expert system preflop, a formula for mapping hand strength and potential to actions postflop, and Monte Carlo simulations designed to estimate the most profitable action to take against an opponent model.

A *chump* is an agent that disregards its cards and takes an action based on the same distribution at every decision point. Since Poki was the only benchmark agent we had available for multiplayer, we also used chumps to evaluate our newly created agents. An *Always-Fold* chump has a probability triple of (f, c, r) = (1, 0, 0) and always folds on its first action. An *Always-Call* chump has a probability triple of (0, 1, 0) and always checks if there is no outstanding bet and calls otherwise. An *Always-Raise* chump has a probability triple of (0, 0, 1) and always bets if there is no outstanding bet and raises otherwise. The final chump we used for evaluating our 3-player agents is called *Probe* and it has a probability triple of (0, 0.5, 0.5) so it chooses equally between check/call and bet/raise at every action. We generated a perfect recall and an imperfect recall CFR agent, using card abstractions based on percentile bucketing of the $E[HS^2]$ metric.

4.2 PR2, a Perfect Recall CFR Agent

Due to the enormous size of the 3-player limit Hold'em game tree, there are very few options available in terms of bucketing. In fact, it takes over 32 GB of RAM to store the game tree with no betting abstractions and only two buckets per round. That is, on every round, the agent can only tell if its hand strength squared compared to a random hand is above or below average!

We ran CFR on a 2-bucket perfect recall abstraction for 20 million iterations on a 32 GB RAM machine and called the agent PR2. However, we were required to reduce the betting on the river from a 4-bet cap to a 3-bet cap to fit the abstraction into memory. In the event that PR2 falls off-tree while playing the real game (i.e. the betting is capped on the river), PR2 simply calls. The 20 million iteration CFR computation took about three weeks to run. Since 100 million (10^8) iterations would require 15 weeks of computation and since we had the 32 GB RAM machine for a limited time, only 20 million iterations were completed.

4.3 IR16, an Imperfect Recall CFR Agent

Several weeks after starting the PR2 computation, we gained access to a 64 GB RAM machine for a limited time. We decided to create a 16-bucket imperfect recall abstraction for the 3-player game to compare the strength of this abstraction with the similar sized 2-bucket perfect recall abstraction. At each round all previous bucketing information is lost.

We used CFR on this abstraction for 20 million iterations (corresponding to the number of iterations used for PR2) and named the agent IR16S (S – for short). We also continued the CFR computation until the start of the 2009 CP Competition. One of the agents submitted to the 2009 CP Competition completed 43 million iterations and was named IR16L (L – for long). With access to the 64 GB RAM machine, we were able to include a 4-bet cap on the river as compared to PR2's 3-bet cap. The CFR computation took about one month to reach 43 million iterations.

4.4 Evaluation of CFR Hold'em Strategies

To reduce variance in a match between three agents A, B, and C, we want each agent to play in the same situation (i.e. same cards, position and relative position). We have 3! = 6 permutations of agents for every set of pre-generated cards since there are three positions (button, small blind, big blind) and two relative position orderings: ABC, BCA, CAB, ACB, CBA and BAC. We average each player's win rate over all of these permutations to get a win rate for that particular matchup.

Agent strength can be evaluated in several different ways. We use the technique used in the annual CP Competition. A 3-player tournament with N agents consists of ${}_NC_3$ distinct 3-player matchups where each agent plays against ${}_{N-1}C_2$ pairs of opponents. In a *bankroll event*, all agents are simply ranked based on their total win rate against all pairs of opponents. The aim of this event is to determine which agent exploits its opponents by the most. In an *elimination event*, the winner is determined by removing the worst-ranked agent from the bankroll event and determining a new bankroll ranking assuming the worst agent had not entered. This process continues recursively (always eliminating the worst remaining agent) until exactly three agents remain in which case they are ranked by win rate in that last matchup. The aim of this event is to determine which agent is the most robust or least exploitable, so it is sometimes referred to as an *equilibrium event*. The outcome of both of these events can be determined with exactly the same set of tournament hand data (i.e. it is not necessary to run two separate tournaments). It is also not necessary to run a new event once an agent is eliminated from the elimination event as we can simply compute new win rates after removing all matchups involving the eliminated agent.

4.5 Experimental Results

A tournament was run with seven 3-player limit Hold'em agents: Always-Call (AC), Probe, Always-Raise (AR), Poki, PR2, IR16S, IR16L. This tournament consisted of ${}_7C_3 = 35$ matchups, where each player played against ${}_6C_2 = 15$ pairs of opponents. Each matchup consisted of 1.2 million hands (6 positional permutations of 200,000 hands). The bankroll results are displayed in Table 3. Each entry is the win rate of the column agent versus the two row agents. The final row is the overall win rate of the column agent against all pairs of opponents; this is the metric used to evaluate and rank the agents. All win rates are in millibets (10^{-3} x the size of the big blind) per hand (mb/h). For comparison, the Always-Fold strategy loses 500 mb/h in a 3-player game against any two agents who do not fold at the start. We computed 95% confidence intervals for each matchup. The size of the confidence intervals varied from 14 to 78 mb/h. However, the largest intervals ±39 mb/h only occurred in the matches with Always-Raise and Probe present. Most of the confidence intervals were within ±10 mb/h.

Table 3. The cross-table for a 7-agent 3-player tournament. Units are mb/h and overall win rates in the last row are within ±6 mb/h with 95% confidence

	PR2	IR16L	IR16S	Poki	AR	Probe	AC
Poki,AC	398	497	507	-	-597	-467	-
Poki, AR	784	493	479	-	-	71	-601
Poki, Probe	639	575	575	-	80	-	-469
Poki, PR2	-	97	97	-	-1054	-839	-972
Poki, IR16S	-32	-	55	-	-816	-765	-1042
Poki, IR16L	-34	53	-	-	-797	-772	-1046
AC, AR	1086	428	441	1197	-	-3	-
AC, Probe	899	737	731	936	6	-	-
AC, PR2	-	510	516	574	-545	-448	-
AC, IR16S	401	-	503	545	-216	-369	-
AC, IR16L	393	489	-	540	-218	-364	-
AR, Probe	734	-636	-725	-150	-	-	-4
AR, PR2	-	380	355	270	-	-371	-541
AR, IR16S	759	-	393	323	-	324	-212
AR, IR16L	763	413	-	318	-	373	-223
Probe, PR2	-	506	506	200	-363	-	-450
Probe, IR16S	624	-	515	190	312	-	-368
Probe, IR16L	631	519	-	198	352	-	-367
PR2, IR16S	-	-	50	-65	-1139	-1130	-911
PR2, IR16L	-	48	-	-63	-1118	-1137	-908
IR16S, IR16L	-39	-	-	-108	-805	-1034	-992
Overall	530	341	333	327	-461	-462	-607

From Table 3, the ranking of agents in the bankroll event is PR2, IR16S, IR16L, Poki, Always-Raise, Probe and Always-Call. From Table 3, we determine the rankings of the top three agents in the elimination event, by eliminating the worst performer recursively. The elimination event winning order is: IR16L, IR16S, PR2, Poki, Probe, Always-Raise, and Always-Call. All three of our new CFR-generated agents are ranked in the top three for both events and out-perform the 2008 CPC multiplayer champion, Poki. PR2 is the best exploitative player by exploiting the chumps. PR2 remains the top player, when Always-Call is removed, but falls to third place when Always-Raise is removed, since there is only a single chump left (Probe) to exploit. The final round of the elimination event has IR16L first (50 mb/h), IR16S second (48 mb/h) and PR2 third (-98 mb/h).

The IR16 agents are the least exploited agents, only losing chips in one matchup, where they are between Always-Raise and Probe. Poki also loses the most against Always-Raise and Probe. On the other hand, PR2 is the only agent to win in this matchup and does so by a substantial amount. Our intuitive assumption that the 2-bucket abstraction would play very poorly was incorrect. Even after 20 million iterations, a fairly strong 3-player agent can be created, although 100 million (10^8) CFR iterations are typically used for a 5-bucket heads-up limit Hold'em abstraction [18] to reduce ε to 3 mb/h. After about twice as many iterations (43 million), there seems to be little gain. Perhaps the IR16L agent is trading off exploitative default play with more unexploitable play.

5. USING HEADS-UP EXPERTS

Heads-up situations arise frequently in 3-player limit Hold'em after one agent folds. Since the resulting heads-up subgame is two-player, zero-sum, and perfect recall, we can use CFR and are guaranteed to find an ε-Nash equilibrium in the subgame. Since the heads-up subtree is much smaller than the corresponding 3-player tree, strategies using much larger abstractions can be used. We call the agents resulting from the CFR solution to these heads-up subgames, heads-up experts (HUEs). We must assume some distribution of buckets is propagated to a HUE subtree by each of the agents. We then solve the corresponding heads-up subgame. Later, we substitute this strategy in place of the 3-player agent's strategy when the HUE subtree is reached during a match.

Columns 2 to 5 of Table 4 show empirical frequencies of bet sequences (column 1) that end in a preflop fold (columns 6 to 8 are used in Section 5.2). The agents used were: Poki, PR2, IR16S, and IR16L. The frequencies were computed from 1.2 million hands of self-play between three identical agents and rounded to two decimal places. Some entries not included in the table (such as *crcf*) were also non-zero, but rounded to 0.00% and one entry (*rrrf*) actually occurred 0.01% of the time for IR16S and IR16L.

Table 4. Empirical frequencies of the most common bet sequences ending in a fold for 1.2 million hands of self-play between three identical agents using four different strategies

sequence	Poki	PR2	IR16S	IR16L	btn	sb	bb
f	20.67	29.27	36.03	36.14	1-2	-	-
rf	9.55	25.60	26.40	26.37	4-5	1-4	-
cf	16.43	0.02	0.09	0.04	3	1-2	-
rrf	0.54	1.51	6.64	6.61	3-5	5	1-4
rcf	1.52	9.96	0.66	0.65	3-5	4	1-2
crf	3.35	0.01	0.04	0.02	3	5	1-2
Total	52.05	66.38	69.87	69.84			

About 50% to 70% of the hands resulted in heads-up play before the flop. The most frequent bet sequences with a preflop fold are: *f*, *rf*, *cf*, *rrf*, *rcf*, and *crf*. We conjectured that creating experts for these common subgame situations might provide improvements.

However, this strategy-knitting approach faces the same potential problems faced by PsOpti, where different strategies were knit together [1]. PsOpti0 used an always-call policy preflop and an appended 3-round postflop model that assumed uniform distributions postflop. PsOpti1 used a 1-round preflop model and an appended postflop model solved for a given pot-size – there are four pot-sizes that reach the flop corresponding to the number of raises – and assuming a uniform distribution of hands for both players. Finally, PsOpti2 used the distributions on the flop from a 3-round preflop model as input to seven 3-round postflop models (corresponding to the seven preflop betting sequences reaching the flop). As shown by experimental data, PsOpti1 out-performed PsOpti2, which out-performed PsOpti0. However, the authors point out that several bugs in PsOpti2 were discovered after the data was collected and claim that a bug-free PsOpti2 should result in the strongest player among the three presented.

We are not proposing to knit together pre-flop and post-flop strategies, but must still be aware of strategy connection problems. Instead, we are proposing to use a static strategy such as IR16L for the full game when a fold does not occur as one of the betting sequences shown in column 1 of Table 4. However, when a sequence from Table 4 does occur, we switch to a HUE strategy for all subsequent actions in that game. In the next sub-section we explain how we knit the strategies together.

5.1 Strategy Seeding

We need to determine a reasonable strategy that will be used to "seed" the HUE subtrees, since each subtree does not start at the beginning of a two-player game. Two issues are important. First, there may be some extra money in the pot at the start of each HUE strategy compared to a normal two-player Hold'em game and second, we cannot necessarily assume a uniform distribution of cards, since some actions have already been taken.

We generated 2-player zero-sum CFR strategies with two different strategy seedings for each of the 6 betting sequences listed in Table 4. The first seeding is *uniform seeding*, where we assume that the players take all of the actions in the forced action sequence 100% of the time. That is, the distribution over hands or buckets of all active players is completely uniform. The second is an *expert seeding*. Instead of uniformly seeding the strategies, we devise a sensible preflop strategy based on expert knowledge (advice from poker professionals and/or based on strong computer agents). When using a one-dimensional metric to bucket hands, it is conventional to place hands with higher metric values in higher numbered buckets. We adopt this convention here; namely, we have a bucket ranking 5 > 4 > 3 > 2 > 1 for our 5-bucket $E[HS^2]$ abstraction. We used pure prior distributions: the probability of taking an action in a bucket is either 0 or 1. The highest (i.e. strongest) buckets are used for raise actions, the next highest buckets are used for call actions, and the lowest buckets are folded. Columns 6 to 8 in Table 4, shows the expert knowledge seeding. Each entry indicates which of the 5 buckets will be used for the sequence in each position. For example, the *rrf* row indicates that the button (btn) will make the first raise with hands in buckets 3-5. The small blind (sb) will make the second raise only with a hand in bucket 5 and the big blind will make the fold action with a hand in buckets 1-4. For uniform seeding, bucket ranges in Table 4 were replaced by the range 1-5.

5.2 Experimental Results

We conducted a 7-agent tournament to evaluate the HUE agents and did not include any chumps (Always-Call, Always-Raise, and Probe). There were two reasons: removing chumps should create a playing field more comparable to those in the annual CP Competition and including just three more entrants would result in $_{10}C_3$ = 120 three-agent matchups instead of 35. The tournament used Poki, PR2, UnM2 (PR2 with uniform seeded HUEs), ExM2 (PR2 with expert seeded HUEs), IR16, Un16 (IR16L with uniform seeded HUEs), Ex16 (IR16L with expert seeded HUEs). Only IR16L (IR16 run for 43 million CFR iterations) was entered into this tournament since the results of IR16S and IR16L were quite similar in Section 4.5 and including both of the IR16 agents in the tournament would have created many more matchups. Therefore all three of the IR16 agents use IR16L as the base player.

The bankroll results are displayed in Table 5. The format is similar to Table 3. All individual matchup win rates are within ±9 mb/h with 95% confidence. The confidence of the overall win rates are ±6 mb/h with 95% confidence. Recall, the Always-Fold strategy loses 500 mb/h in a 3-player game. The rankings for the bankroll event are: IR16, Ex16, Un16, ExM2, UnM2, PR2, and Poki. The rankings for both the elimination and bankroll events were identical. In the tournament results from Section 4.5, PR2 won the bankroll event as it exploited chumps more than any other agent. However, there were only strong agents in this tournament. The top two bankroll players (IR16 and Exp16) win in every matchup they participate in, while the opposite is true for the worst two bankroll players (Poki and PR2). Although poker is a non-transitive game, Table 5 shows trends. As you move across the columns of a given row (corresponding to lower ranked bankroll players), the win rates decrease monotonically with only a few exceptions. Having all AI agents (instead of including chumps) causes the results to be much more predictable. Another trend is that every agent loses the most against the top two bankroll players and every agent wins the most against the worst two bankroll players. The regularity of these trends is the reason we obtain the same rankings for both events.

The expert seeded HUEs did better than the uniform seeded HUEs when applied to both PR2 and IR16 so the assumption that an opponent holds a completely random hand after several betting actions is a poor one. PR2's performance was improved by adding HUEs while IR16 performs worse. This demonstrates just how effective CFR can be for generating very abstract, imperfect recall, 3-player agents despite having no theoretical guarantees. Our intention of incorporating and testing the HUEs was to determine whether fairly abstract and naively created strategies had the potential to improve play. Better strategy seeding is possible. Due to timing constraints and machine availability, we generated the HUEs and base players simultaneously so we were unable to use the base players' strategies to seed their own HUEs. Seeding each HUE with the 3-player base strategy it will be used with should produce better results.

Table 5. The cross-table for a 7-agent 3-player tournament with four HUEs. Units are mb/h and overall win rates in the last row are within ±6 mb/h with 95% confidence

	IR16	Ex16	Un16	ExM2	UnM2	PR2	Poki
Poki, PR2	95	86	70	60	35	-	-
Poki, UnM2	84	78	54	50	-	-1	-
Poki, ExM2	63	63	43	-	14	-10	-
Poki, IR16	-	50	19	29	-9	-30	-
Poki, Un16	70	67	-	37	7	-20	-
Poki, Ex16	56	-	22	30	-6	-27	-
PR2, UnM2	81	71	57	28	-	-	-34
PR2, ExM2	65	58	49	-	4	-	-50
PR2, IR16	-	39	23	2	-25	-	-65
PR2, Un16	60	53	-	5	-14	-	-50
PR2, Ex16	51	-	28	4	-19	-	-59
UnM2, ExM2	55	45	39	-	-	-31	-64
UnM2, IR16	-	30	9	-13	-	-57	-75
UnM2, Un16	53	42	-	-6	-	-42	-60
UnM2, Un16	40	-	13	-10	-	-53	-72
ExM2, IR16	-	15	-2	-	-43	-67	-92
ExM2, Un16	34	25	-	-	-33	-53	-80
ExM2, Un16	22	-	0	-	-35	-62	-92
IR16, Un16	-	9	-	-32	-61	-83	-89
IR16, Ex16	-	-	-29	-36	-69	-90	-106
Un16, Ex16	20	-	-	-25	-55	-81	-90
Overall	57	49	26	8	-21	-47	-72

6. COMPETITION RESULTS

The results of the 2009 CP Competition were announced at the IJCAI conference on July 15, 2009 in Pasadena, California. There were 14 competitors (10 universities and 4 independents) from 7 different countries and 25 agents submitted (13 for heads-up limit, 5 for heads-up no-limit, and 7 for ring limit). The ring limit events

this year were 3-player limit Hold'em. Each 3-player event used 660,000 hands (6 position permutations by 110 matches of 1000 hands). There were 7 agents submitted from 5 teams denoted: IR16 (Hyperborean-Eqm U. of Alberta), Ex16 (Hyperborean-BR U. of Alberta), dpp (Technical U. Darmstadt), dcu (dcu3pl Dublin City U.), blu (Bluechip U. of Michigan), cmu (CMURingLimit Carnegie Mellon U.) and ak (akuma Technical U. Darmstadt). Matchups including two agents from the same team were not used in determining the winners of the events. This means there are no matches including both IR16 and Ex16 or both dpp and ak.

IR16 and Exp16, placed first and second respectively in both events. Table 6 shows the win rates for these agents against pairs of the rest of the agents in millibets per hand (mb/h). Recall, the Always-Fold strategy loses 500 mb/h in a 3-player game.

Table 6. Cross-table results for IR16 and Ex16 games in the 2009 IJCAI Computer Poker Competition. Units are mb/h and results are within ±6 to ±8 mb/h with 95% confidence

	dpp dcu	dpp blu	dpp cmu	ak dcu	ak blu	ak cmu	dcu blu	dcu cmu	blu cmu	total
IR16	257	384	271	217	374	231	442	255	438	319
Ex16	224	360	235	19	364	215	426	235	440	299

7. CONCLUSION

In this paper, we've shown that despite a lack of theoretical guarantees for CFR regarding multiplayer games, CFR can be used to generate very abstract perfect recall 3-player agents. One such agent, PR2, outperforms Poki, the 2008 CP Competition ring limit event champion. This work represents the first reported research results performed with CFR on multiplayer games.

CFR can be used to generate two-player ε -Nash equilibrium profiles, called heads-up experts (HUEs), for 3-player subgames where one of the players folds preflop. Since the resulting subgame is a two-player zero-sum game, we used CFR to compute an ε-Nash equilibrium for each HUE and knitted them to our CFR generated 3-player strategy, PR2. This knitted strategy showed promise by outperforming the base PR2 strategy.

CFR can be used to generate very abstract imperfect recall 3-player agents. One such agent, IR16S, outperformed Poki, PR2 and the HUE versions of PR2, even though its abstraction is approximately the same size as PR2. IR16S even defeated HUEs that use it as a base player, but we expect that HUEs with better strategy knitting will outperform it.

Two of the agents created using CFR, IR16 and Ex16, placed first and second in the two ring limit events, bankroll and elimination, of the 2009 CP Competition.

CFR was created to compute ε -Nash equilibria for two-player zero-sum, perfect recall games. We have shown that it has the potential to compute winning strategies in 3-player zero-sum games using both perfect recall and imperfect recall abstractions

8. ACKNOWLEDGMENTS

Thanks to the Computer Poker Research Group at the University of Alberta for their insights and discussions. This research was supported in part by research grants from the Natural Sciences and Engineering Research Council of Canada (NSERC) and the Alberta Informatics Circle of Research Excellence (iCORE).

9. REFERENCES

[1] 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. IJCAI.

[2] Ganzfried, S. and Sandholm, T. 2008. Computing an approximate jam/fold equilibrium for 3-agent no-limit texas hold'em tournaments. AAMAS.

[3] Ganzfried, S. and Sandholm, T. 2009. Computing Equilibria in Multiplayer Stochastic Games of Imperfect Information. IJCAI.

[4] Gilpin, A., Hoda, S., Peña, J. and Sandholm, T. 2007. Gradient-based algorithms for finding nash equilibria in extensive form games. Workshop on Internet and Network Economics, (WINE), LNCS 4858.

[5] Gilpin, A., Sandholm, T. and Sorensen, T.B. 2008. A heads-up no-limit texas hold'em poker player: discretized betting models and automatically generated equilibrium-finding programs. AAMAS.

[6] Gilpin, A., Sandholm, T. and Sorensen, T.B. 2007. Potential-aware automated abstraction of sequential games, and holistic equilibrium analysis of Texas Hold'em poker. AAAI.

[7] Hamilton, C. 2008. AAAI Third Annual Computer Poker Competition. AI Magazine - AAAI News, 29 (1).

[8] Johanson, M.B. 2007. Robust strategies and counter-strategies: Building a champion level computer poker player. MSc thesis, University of Alberta, Edmonton.

[9] Koller, D., Megiddo, N. and Von Stengel, B. 1996. Efficient computation of equilibria for extensive two-person games. Games and Economic Behavior, 14(2).

[10] Koller, D. and Pfeffer, A. 1997. Representations and solutions for game-theoretic problems. Artificial Intelligence, 94.

[11] Kuhn, H.W. 1950. A simplified two-person poker. Contributions to the Theory of Games, 1.

[12] Nash, J.F. 1950. Equilibrium points in n-person games. National Academy of Sciences of the USA.

[13] Osborne, M. and Rubenstein, A. 1994. A Course in Game Theory. The MIT Press.

[14] Schnizlein, D., Bowling, M. and Szafron, D. 2009. Probabilistic state translation in extensive games with large action sets. IJCAI.

[15] The Second Man-Machine Poker Competition. http://poker.cs.ualberta.ca/man-machine/.

[16] Southey, F., Bowling, M., Larson, B., Piccione, C., Burch, N., Billings, D. and Rayner, C. 2005. Bayesbluff: Opponent modelling in poker. UAI.

[17] Waugh, K., Schnizlein, D., Bowling, M. and Szafron, D. 2009. Abstraction pathology in extensive games. AMMAS.

[18] Zinkevich, M., Johanson, M., Bowling, M. and Piccione, C. 2008. Regret minimization in games with incomplete information. NIPS.