Reflections on the First Man Versus Machine No-Limit Texas Hold 'em Competition

Sam Ganzfried

■ *The first human versus computer no-limit Texas hold 'em competition took place from April 24–May 8, 2015, at Rivers Casino in Pittsburgh, PA. In this article I present my thoughts on the competition design, agent architecture, and lessons learned. Several problematic hands from the competition are highlighted that reveal the most glaring weaknesses of the agent. The research underlying the agent is placed within a broader context in the AI research community, and several avenues for future study are mapped out.*

The first ever human versus computer no-limit Texas hold 'em competition took place from April 24–May 8, 2015, at Rivers Casino in Pittsburgh, PA, organized by Carnegie Mellon University Professor Tuomas Sandholm. Twenty thousand hands of two-player no-limit Texas hold 'em were played between the computer program Claudico and four of the top human specialists in this variation of poker, Dong Kim, Jason Les, Bjorn Li, and Doug Polk (so 80,000 hands were played in total).

While this was the first human versus machine competition for the no-limit variant of Texas hold 'em, there had been two prior competitions for the limit variant. Two-player limit Texas hold 'em is the smallest poker variant played competitively by humans. A breakthrough was achieved last year as the game was "essentially weakly solved" (an ε-Nash equilibrium was computed for such small ε to be statistically indistinguishable from zero in a human lifetime of play) by researchers at the University of Alberta (Bowling et al. 2015). In the limit variant all bets are of a fixed size, while in no-limit bets can be of any number of chips up to the amount

remaining in a player's stack (the stacks are reset to a fixed amount of 200 big blinds at the start of each hand). Thus, the game tree for no-limit has a much larger branching factor and is significantly larger; there are 10^{165} nodes in the game tree for no-limit, while there are around 10^{17} nodes for limit (Johanson 2013). In 2007 a program called Polaris that was created by researchers at the University of Alberta played four duplicate 500-hand matches against human professionals. The program won one match, tied one, and lost two, thus losing the match overall. In 2008 an improved version of Polaris competed against six human professionals in a second match, this time coming out victorious (three wins, two losses, and one tie). There have also been highly-publicized human versus machine competitions for other games; for example, chess program Deep Blue lost to human expert Garry Kasparov in 1996 and beat him in 1997, and *Jeopardy!* agent Watson defeated human champions in 2011.

Claudico is Latin for "I limp." Limping is the name of a specific play in poker. After the initial antes have been paid, the first player to act is the small blind and he has three available actions; fold (forfeit the pot), call (match the big blind by putting in 50 chips more), or raise by putting in additional chips beyond those needed to call (a raise can be any integral amount from 200 chips up to 20,000 chips in this situation). The second option of just calling is called *limping* and has traditionally been viewed as a very weak play only made by bad players. In one popular book on strategy, Phil Gordon writes,

> Limping is for Losers. This is the most important fundamental in poker — for every game, for every tournament, every stake: If you are the first player to voluntarily commit chips to the pot, open for a raise. Limping is inevitably a losing play. If you see a person at the table limping, you can be fairly sure he is a bad player. Bottom line: If your hand is worth playing, it is worth raising (Gordon 2011).

Claudico actually limps close to 10 percent of its hands, and based on discussion with the human players who did analysis it seems to have profited overall from the hands it limped. Claudico also makes several other plays that challenge conventional human poker strategy; for example it sometimes makes very small bets of 10 percent of the pot, and sometimes very large all-in bets for many times the pot (for example, betting 20,000 into a pot of 500). By contrast, human players typically utilize a small number of bet sizes, usually between half pot and pot.

Competition Design

To evaluate the performance, judges used "duplicate" scoring, in which the same hands were played twice with the cards reversed to reduce the role of luck (and thereby the variance). For example, suppose human A has pocket aces and the computer has pocket kings, and A wins $5,000. This would indicate that the human outplayed the computer. However, suppose human B has the pocket kings against the computer's pocket aces in the identical situation and the computer wins $10,000. Then, taking both of these results into account, an improved estimator of performance would indicate that the computer outplayed the human, after the role of luck in the result was significantly reduced. Each human was given a partner, who played the identical hands against Claudico with the cards reversed. Polk was paired with Les, and Kim was paired with Li. The players played in two different rooms of the casino simultaneously, with one player from each of the pairings in each room (so that both players in each room had the same cards, while both players in the other room had the cards that Claudico had in the first room).

In total, the humans ended up winning the match by 732,713 chips, which corresponds to a win rate of 9.16 big blinds per 100 hands (BB/100), a common metric used to evaluate performance in poker. (The small blind (SB) and big blind (BB) correspond to initial investments, or "antes" of the players. In the match, the SB was 50 chips and the BB was 100 chips.) This was a relatively decisive win for the humans and was statistically significant at the 90 percent confidence level, though it was not statistically significant at the 95 percent level. To put these results into some perspective, Dong Kim won a challenge match against a strong professional player Nick Frame by 13.87 BB/100 (he won by $103,992 over 15,000 hands with blinds SB=$25, BB=$50), and Doug Polk defeated Ben Sulsky in another high-profile challenge match by 24.67 BB/100 (he won by $740,000 over 15,000 hands with blinds SB = $100, BB = $200).

The chips were just a placeholder to keep track of the score and did not represent real money; the humans were paid at the end from a prize pool of $100,000 which had been donated from Rivers Casino and Microsoft Research. The human with the smallest profit over the match received $10,000, while the other humans received $10,000 plus additional payoff in proportion to the profit above the lowest profit.

Agent Architecture

Claudico was an improved version of an earlier agent called *Tartanian7* that came in first place in the 2014 AAAI computer poker competition, beating each opposing agent with statistical significance. The architecture of that agent has been described in detail in a recent paper (Brown, Ganzfried, and Sandholm 2015). At a very high level, the design of the agent follows the three-step procedure depicted in figure 1, which is the leading paradigm used by many of the strongest agents for large games (that is, games that

are too large to be solved for equilibrium directly by existing techniques).

In the first step, the original game is approximated by a smaller abstract game that hopefully retains much of the strategic structure of the initial game. The first abstractions for two-player Texas hold 'em were manually generated (Shi and Littman 2002; Billings et al. 2003), while current abstractions are computed algorithmically (Gilpin and Sandholm 2006; 2007a; Gilpin, Sandholm, and Sørensen 2008; Waugh et al. 2009; Johanson et al. 2013). For smaller games, such as Rhode Island hold 'em, abstraction can be performed losslessly, and the abstract game is actually isomorphic to the full game (Gilpin and Sandholm 2007b). However, for larger games, such as Texas hold 'em, players must be willing to incur some loss in the quality of the modeling approximation due to abstraction.

The second step is to compute an approximate Nash equilibrium in the smaller abstracted game, using a custom iterative equilibrium-finding algorithm such as counterfactual regret minimization (CFR) (Zinkevich et al. 2007) or a generalization of Nesterov's excessive gap technique (EGT) (Hoda et al. 2010).

The final step is to construct a strategy profile in the original game from the approximate equilibrium of the abstracted game by means of a reverse mapping procedure. When the action spaces of the original and abstracted games are identical, this step is often straightforward, since the equilibrium of the abstracted game can be played directly in the full game. However, even in this simplified setting often significant performance improvements can be obtained by applying a nontrivial reverse mapping. Several procedures that modify the action probabilities of the abstract equilibrium strategies by placing more weight on certain actions have been shown to significantly improve performance (Ganzfried, Sandholm, and Waugh 2012; Brown, Ganzfried, and Sandholm 2015). These postprocessing procedures are able to achieve robustness against limitations of the abstraction and
equilibrium-finding phases of the paradigm.

When the action spaces of the original and abstracted games differ, an additional procedure is needed to interpret actions taken by the opponent that are not allowed in the abstract game model. Such a procedure is called an action translation mapping. The typical approach for performing action translation is to map the opponent's action to a nearby action that is in the abstraction (perhaps probabilistically), and then respond as if the opponent had taken this action.

An additional crucial component of Claudico, which was not present in Tartanian7 due to a last-minute technical difficulty (though a version of it was present in prior agent Tartanian6), is an approach for real-time computation of solutions in

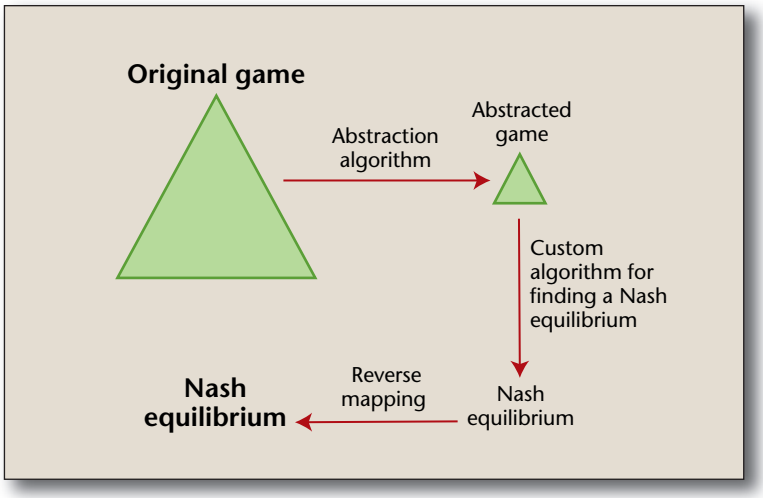

Figure 1. Leading Paradigm for Solving Large Games.

the part of the game tree that has reached a greater degree of accuracy than in the offline computation, called *endgame solving,* which is depicted in figure 2 (Ganzfried and Sandholm 2015). At a high level, endgame solving works by assuming both agents follow the precomputed approximate equilibrium strategies for the trunk portion of the game prior to the end game; then the end game induced by these trunk strategies is solved, using Bayes' rule to compute the input distributions of players' private information leading into the end game. In general, such a procedure could produce a nonequilibrium strategy profile (even if the full game has a unique equilibrium and a single end game); for example, in a sequential version of rock-paper-scissors where player 1 acts and then player 2 acts without observing the action taken by player 1, if we fix player 1 to follow his equilibrium strategy of randomizing equally among all three actions, then any strategy for player 2 is an equilibrium in the resulting end game, because each one yields her expected payoff 0. In particular, the equilibrium solver could output the pure strategy Rock for her, which is clearly not an equilibrium of the full game. However, endgame solving is successful in other games; for example in a game where player 1 first selects an action a_i and then an imperfect-information game G_i is played, we could simply solve the G_i corresponding to the action a_i that is actually taken, provided that the G_i are independent and no information sets extend between several G_i. Furthermore, endgame solving has been previously demonstrated to improve performance empirically against strong computer programs in no-limit Texas hold 'em (Ganzfried and Sandholm 2015).

We used the end-game solver to compute the strategies in real time for the final betting round of each hand, called the *river* (the rules of no-limit Texas hold 'em will be discussed in a subsequent section).

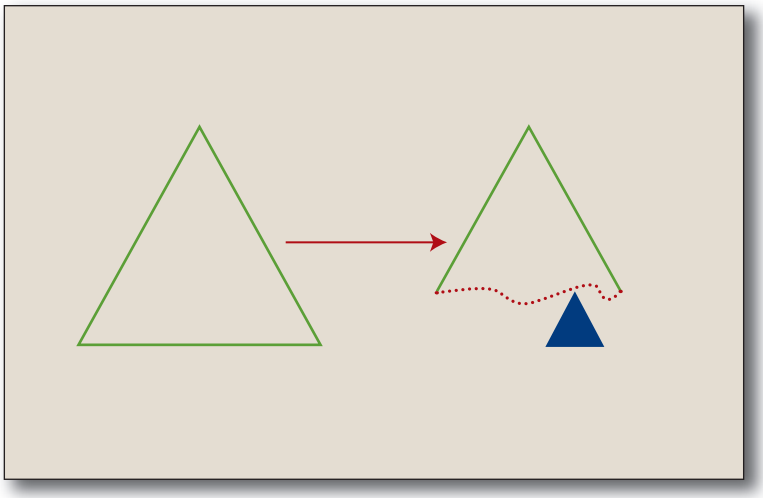

Figure 2. Endgame Solving.

Endgame solving (re)-solves the relevant endgame that has actually been reached in real time to a greater degree of accuracy than in the offline computation.

Despite the theoretical limitation of the approach, Doug Polk related to me in personal communication after the competition ended that he thought the river strategy of Claudico using the end-game solver was the strongest part of the agent.

Offline Abstraction and Equilibrium Computation

Claudico's action abstraction was manually generated and consisted of sizes ranging from 0.1 pot in certain situations to all-in (wagering all of one's remaining chips). The information abstraction was computed using a hierarchical algorithm that first clustered the three-card public flop boards into public buckets, then clustered the private information states for each postflop round (that is, flop, turn, river) separately for each public bucket (no information abstraction was performed for the preflop round) (Brown, Ganzfried, and Sandholm 2015). The equilibrium-finding algorithm used by today's strongest Texas hold 'em agents is a Monte Carlo version of the counterfactual regret minimization algorithm (MCCFR) (Lanctot et al. 2009). That algorithm involves repeatedly sampling chance outcomes and actions down the tree, and updating regret and average strategy values that are stored at each information set. A major reason that CFR has become more popular than EGT is that the best abstraction algorithms use imperfect recall (Waugh et al. 2009; Johanson et al. 2012; Brown, Ganzfried, and Sandholm 2015), and CFR can be run straightforwardly on imperfect-recall game abstractions while no implementations of EGT have been developed for doing so. Our hierarchical abstraction algorithm allowed us to apply a new scalable distributed version of MCCFR (Brown, Ganzfried, and Sandholm 2015). We ran the equilibrium-finding algorithm for several months on Pittsburgh's Blacklight supercomputer using 961 cores (60 blades of 16 cores each, plus one core for the head blade, with each blade having 128 GB RAM).

Action Translation

For the action translation mapping, we used the pseudo-harmonic mapping, which maps a bet x of the opponent to one of the nearest sizes in the abstraction A, B according to the following formula, where $f(x)$ the probability that x is mapped to A (Ganzfried and Sandholm 2013):

$$f(x) = \frac{(B-x)(1+A)}{(B-A)(1+x)}$$

This mapping was derived from analytical solutions of simplified poker games and has been demonstrated to outperform prior approaches in terms of exploitability in simplified games, as well as the best prior approach in empirical performance against no-limit Texas hold 'em agents. The mapping also satisfies several axioms and theoretical properties that the best prior mappings do not satisfy, for example it is Lipschitz continuous in A and B, and therefore robust to small changes in the actions used in the action abstraction.

As an example to demonstrate the operation of the algorithm, suppose the opponent bets 100 into a pot of 500, and that the closest sizes in our abstraction are to "check" (that is, bet 0) or to bet 0.25 pot: so $A = 0$ and $B = 0.25$. Plugging these in gives $f(x) = 1/6 = 0.167$. This is the probability his bet is mapped down to 0 and interpreted as a check. A random number is then selected in [0,1], and if it is above 1/6 the bet is interpreted as 0.25 pot, otherwise as a check.

Postprocessing

We used additional postprocessing techniques to round the action probabilities that had been computed by the offline equilibrium-finding algorithm (Ganzfried, Sandholm, and Waugh 2012). We used a generalization of the prior approach that applied a different rounding threshold for each betting round (that is, action probabilities below the threshold were rounded to zero and then all probabilities were renormalized), with a more aggressive (that is, larger) threshold used for the later betting rounds, since the equilibrium-finding algorithm obtains worse convergence for those rounds due to having fewer samples. We did not apply any postprocessing for ourselves on the river when using the end-game solver, and assumed neither agent used any postprocessing in the generation of the trunk strategies used as inputs to the end-game solver.

It may seem somewhat strange that we applied postprocessing for our own play, but assumed that no postprocessing was applied for the trunk strategies entering the end game, and that this may be prob-

lematic due to the mismatch between our own strategy and the model of it entering the end game. We chose to do this because the endgame solving approach can be less robust if the input strategies have weight on only a small number of hands (as an extreme example, if all the weight was on one hand, then the end-game solver would assume that the other agent knew our exact hand, and the solution would require us to play extremely conservatively). The approach is much more robust if we include a small probability on many different hands before applying the postprocessing. We believed that the gain in robustness outweighed the limitation of the mismatch (in addition to the reasons given above, we already expect there to be a mismatch between the input trunk strategy for the opponent, which is based off our offline equilibrium computation, and his own actual strategy, and thus we would not be removing this mismatch completely even if we eliminated it for our own strategy).

Endgame Solving

The endgame solving algorithm consists of several steps (Ganzfried and Sandholm 2015). First, the joint hand-strength input distributions are computed by applying Bayes' rule to the precomputed trunk strategies, utilizing a recently developed technique that requires only a linear number of lookups in the large strategy table (while the naïve approach requires a quadratic number of lookups and is impractical). Then the equity is computed for each hand, given these distributions. The equity of a hand against a distribution for the opponent is the probability of winning plus one half the probability of tying. Then hands are bucketed separately for each player based on the computed equities for the given situation by applying an information abstraction algorithm. Finally an exact Nash equilibrium is computed in the game corresponding to this information abstraction and an action abstraction that had been precomputed for the specific pot and stack size of the current hand. All of this computation was done in real time during gameplay. To compute equilibria within the end games, we used Gurobi's parallel linear program solver[1] to solve the sequence-form optimization formulation (Koller, Megiddo, and von Stengel 1994).

Rules of No-Limit Texas Hold 'em

Two-player no-limit Texas hold 'em works as follows. Initially two players each have a stack of chips (worth $20,000 in the computer poker competition). One player, called the s*mall blind,* initially puts $50 worth of chips in the middle, while the other player, called the *big blind,* puts $100 worth of chips in the middle. The chips in the middle are known as the pot, and will go to the winner of the hand.

Next, there is an initial round of betting. The player to act can choose from three available options:

Fold: Give up on the hand, surrendering the pot to the opponent.

Call: Put in the minimum number of chips needed to match the number of chips put into the pot by the opponent. For example, if the opponent has put in $1000 and we have put in $400, a call would require putting in $600 more. A call of zero chips is also known as a check.

Bet: Put in additional chips beyond what is needed to call. A bet can be of any size from 1 chip up to the number of chips a player has left in his stack, provided it exceeds some minimum value and is a multiple of the smallest chip denomination (by contrast, in the limit variant, all bets must of a fixed size, which equals the big blind for the first two rounds and twice the big blind for the final two rounds). The minimum allowable bet size is the big blind for the first bet of a round and the size of the previous bet in the current round for subsequent bets. A bet of all of one's remaining chips is called an *all-in bet.* If the opponent has just bet, then our additional bet is also called a *raise.* In some variants, the number of raises in a given round is limited (for limit it is limited to three and for no-limit it is unlimited), and players are forced to either fold or call at that point.

The initial round of betting ends if a player has folded, if there has been a bet and a call, or if both players have checked. If the round ends without a player folding, then three public cards are revealed face-up on the table (called the *flop*) and a second round of betting takes place. Then one more public card is dealt (the turn) and a third round of betting, followed by a fifth public card (the *river*) and a final round of betting. If a player ever folds, the other player wins all the chips in the pot. If the final betting round is completed without a player folding, then both players reveal their private cards, and the player with the best five-card hand (out of his two private cards and the five public cards) wins the pot (it is divided equally for a tie).

Problematic Hands

Several hands stood out during the course of the competition that highlighted weaknesses of the agent.

In one hand, Claudico had A4s (ace and four of the same suit) and folded preflop after it had put in over half of its stack (the human opponent had 99). This is regarded as a bad play, since it would only need to win around 25 percent of the time against the opponent's distribution for a call to be profitable at this point (Claudico wins about 33 percent of the time against the hand the human had). The problem was that the translation mapping mapped the opponent's raise down to a smaller size, which caused the agent to look up a strategy that had been computed thinking that the pot size was much smaller than it had thought it was (Claudico thought it had invested around 7,000 when it had actually invested close to 10,000 — recall that the starting stacks are 20,000).

These translation issues can get magnified further as the hand develops if the agent thinks it has bet a percentage (for example, 2/3) of the (correct) size of the pot, while the strategies that were precomputed assumed a different size of the pot.

In another hand Claudico had KT and folded to an all-in bet on the turn after putting in about 3/4 of its stack despite having top pair and a flush draw (there were three diamonds on the board and Claudico had the king of diamonds; the opponent actually had A2 with the ace of diamonds, for a better flush draw but worse hand due to Claudico having a pair). The issue for this hand was that the human made a raise on the flop which was slightly below the smallest size Claudico had in its abstraction in that situation, and it ended up mapping it down to just a call (it was just mapped down with around 3 percent probability in that situation, and so Claudico ended up getting pretty "unlucky" that the action was mapped in the "wrong" direction). This caused the agent to think it had committed far fewer chips to the pot at that point than it actually had.

The problem in these hands was not due simply to a flaw in the action translation mapping, or even to a flaw in the action abstraction (though of course improvements to those would be very beneficial as well); even if Claudico had used a different translation mapping and/or used different action sizes in the abstraction, it would still have potentially sizable gaps between certain sizes of the abstraction due to the fact that the agent can only select so many to keep the abstraction sufficiently small so that it can be solved within time and memory limits. That means that, given the current paradigm, the agent will necessarily have to map bets to sizes somewhat far away with some probability, which will cause its perception of the pot size to be incorrect, as these hands indicate. This is called the off-tree problem, which has received very little study thus far. Some agents, such as versions of the agent from the University of Alberta, attempt to mitigate this problem by specifically taking actions aimed to get back on the tree (for example, making a bet that would not ordinarily be made to correct for the pot size disparity). However, this is problematic too, as it requires the agent to take an undesirable action. The endgame solving approach provides a solution to this problem by inputting the correct pot size to the end-game solving algorithm, even if this differs from the agent's perception of it at that point due to the opponent having taken an action outside of the action abstraction. In general, real-time endgame solving could correct for many misperceptions in game-state information that have been accumulated along the course of game play; however, this would not apply to the preflop, flop, and turn rounds, where endgame solving is not used. Thus it is necessary to explore additional approaches to this problem; improved algorithms for real-time computation for the earlier rounds is a potentially promising direction, and perhaps new approaches can also be developed for addressing the off-tree problem independently of endgame solving.

I went over the log files for these two specific hands with Doug Polk in person after the competition had ended, and he agreed that Claudico's plays in both hands were reasonable had the pot size been what the computed strategies perceived it to be at that point. Of course, we both agreed that the hands were both major mistakes if you include the misperception of the pot size. Even though these were only low-probability mistakes due to the randomization outcome selected by the translation mapping, these types of mistakes can become a significant liability in aggregate, particularly when playing against humans who are aware of them and actively trying to exploit them. Doug alluded to this point as well in an interview after the competition. Based on Doug's interview and subsequent conversations it seems that he views this as Claudico's biggest weakness, and it will be interesting to see what improvements can be found, and whether those can be exploited in turn by good countermeasures.

In one other problematic hand, Claudico made a large all-in bet (of around 19,000) into a relatively small pot of around 1700. There were three of a suit (spades) on the board, and Claudico had a very weak hand without a fourth spade (so the bet was a "bluff," hoping the opponent would fold a stronger hand). The problem is not that Claudico made a large bet per se, or even that it did so with a very weak hand; extremely large bets are correct and part of equilibrium strategy in certain situations, and in such situations they must be made with some weak hands as bluffs to balance with the very strong "value" hands or else the strategy would be too predictable (if an agent never bluffed, then the opponent would just fold everything except his hands that beat half of the value hands, and then the bets with the bottom half of the value hands would be unprofitable). Thus, making large bets as bluffs is needed in certain situations. The problem is that certain hands are much better suited for them than others. For example, suppose the board was JsTs4sKcQh, and suppose the agent could have 3c2c (three and two of clubs) versus 3s2c (three of spades and two of clubs). Both hands are extremely weak (they produce the worst possible five-card hand); however, having the spade three actually has a subtle and very significant benefit: it significantly reduces the probability that the opponent holds an extremely strong hand (for example, an ace-high or king-high flush) because several of the hands that would constitute that strength would contain that card, for example, As3s and Ks3s. Thus, this would make a much better choice for a hand to make a large bet with, since the opponent is less likely to have a hand strong enough to call, making the bluff bet more effective. The endgame-solving algo-

rithm described in Endgame Solving section takes this "card removal" factor into account to an extent, since the equities are computed for each hand against the distribution the opponent could hold given that hand; however, this does not fully take into account the card removal effect. For example, the 3c2c and 3s2c hands would both have the lowest possible equity (it would be slightly above zero only because of possible ties), and would be necessarily grouped into the same bucket by the end-game information-abstraction algorithm (the worst bucket) despite the fact that they have very different card removal properties.

Doug Polk said that he thought the river strategy using the end-game solver overall was the strongest part of Claudico; however, he thought that utilizing the large betting sizes without properly accounting for card removal was actually a significant weakness, since Claudico would be bluffing with nonoptimal hands. The Claudico team came to this conclusion ourselves as well during the competition, and for this reason decided to take out the large bets for Claudico from the end-game solver partway through the competition, since this issue is most problematic for those bet sizes (for smaller bet sizes, card removal is still important, but significantly less important since we are not just trying to "block" the opponent from having a small number of extremely strong hands, since he will be calling with many more hands). Interestingly, Dong Kim told me after the competition that they had conducted analysis and Claudico was actually profiting on the large bet sizes during the time they were used, despite the theoretical issue described above. I think everyone agrees that massive "overbets" are part of full optimal strategies, and likely underutilized by even the best human players. But card removal is also particularly important for these sizes, and I think for an agent to use them successfully an improved algorithm for dealing with blockers/card removal would need to be developed, though I am still quite curious how well Claudico would have performed if it continued with those sizes included in the agent.

Conclusion

It is one thing to evaluate a poker agent against other computer agents, who largely also play static approximations of equilibrium strategies; it is another to compete against the strongest human specialists, who will adapt and attempt to capitalize on even the smallest perceived weaknesses. This was the first time a no-limit Texas hold 'em agent has competed against human players of this caliber, and our team really had no idea what to expect entering the competition, as previously all of our experiments had been against computer agents from the AAAI Annual Computer Poker Competition. Many valuable lessons were learned that will be pivotal in developing improved agents going forward. I have highlighted the two most important avenues for future research. The first is to develop an improved approach for the "off-tree" problem where a mistake is made due to a misperception of the actual size of the pot after translating an action for the opponent that is not in our action abstraction. I have outlined promising agendas for attacking this problem, including improved action abstraction and translation algorithms, novel approaches for real-time computation that address the portion of the game prior to the final round, and entirely new approaches specifically geared at solving the off-tree problem independently of the other problems. And the second is to develop an improved approach for information abstraction that better accounts for card removal/blockers (that is, that accounts for the fact that having certain cards in our hand modifies the probability of the opponent having certain hands). This issue is most problematic within the information abstraction algorithm for the end game, where the card removal effect is most significant due to the distributions for us and the opponent being the most well defined (that is, there is no more potential remaining in the hand due to uncertainty of public cards, and this relative certainty will likely cause the distributions to put positive weight on fewer hands), and it limits our ability to utilize large bet sizes, which have been demonstrated to be optimal in certain settings. Of course, it would be beneficial to develop an improved information abstraction algorithm that accomplishes this in the part of the game prior to the end game as well.

At first glance it may appear that these issues are purely pragmatic and specific to poker. While one of the main goals is certainly to produce a poker agent that can beat the strongest humans in two-player no-limit Texas hold 'em, there are deeper theoretical questions related to each component of the agent that has been described. Endgame solving has been proven to have theoretical guarantees in certain games while it can lead to strategies with high exploitability in others (even if the full game has a single Nash equilibrium and just a single end game is considered) (Ganzfried and Sandholm 2015). It would be interesting to prove theoretical bounds on its performance on interesting game classes, perhaps classes that include variants of poker. Empirically the approach appears to be very successful on poker despite its lack of theoretical guarantees. Recently an approach has been developed for game decomposition that has theoretical guarantees (Burch, Johanson, and Bowling 2014); however, from personal communication with the authors I have learned that the approach performs worse empirically than our approach that does not have a worst-case guarantee.

The main abstraction algorithms that have been successful in practice are heuristic and have no theoretical guarantees. It is extremely difficult to prove meaningful guarantees when performing such a large

degree of abstraction, for example, approximating a game with 10^{165} states by one with 10^{14} states. There has been some recent work done on abstraction algorithms with theoretical guarantees, though that work does not scale to games nearly as large as no-limit Texas hold 'em. One line of work performs lossless abstraction, which guarantees that the abstract game is exactly isomorphic to the original game (Gilpin and Sandholm 2007b). This work has been applied to compute equilibrium strategies in Rhode Island hold 'em, a medium-sized (3.1 billion nodes) variant of poker. Recent work has also presented the first lossy abstraction algorithms with bounds on the solution quality (Kroer and Sandholm 2014). However, the algorithms are based on integer programming formulations, and only scale to a tiny poker game with a five-card deck. It would be very interesting to bridge this gap between heuristics that work well in practice for large games with no theoretical guarantees, and the approaches with theoretical guarantees that have more modest scalability.

Scalable algorithms for computing Nash equilibria have diverse applications, including cybersecurity (for example, determining optimal thresholds to protect against phishing attacks), business (for example, auctions and negotiations), national security (for example, computing strategies for officers to protect airports), and medicine. For medicine, algorithms that were created in the course of research on poker (Johanson et al. 2012) have been applied to compute robust policies for diabetes management (Chen and Bowling 2012); recently it has been proposed that equilibrium-finding algorithms are applicable to the problem of treating diseases such as the HIV virus that can mutate adversarially (Sandholm 2015).

For the pseudoharmonic action translation mapping, in addition to showing that it outperforms the best prior approach in terms of exploitability in several games, we have also presented several axioms and theoretical properties that it satisfies; for example, it is Lipschitz continuous in A and B, and therefore robust to small changes in the actions used in the action abstraction (Ganzfried and Sandholm 2013). Another mapping that has very high exploitability in several games also satisfies these axioms, and further investigation can lead to deeper theoretical understanding of this problem and potentially new improved approaches.

Even the postprocessing approaches, which appear to be purely heuristic, have interesting theoretical open questions. For example, it has been shown that purification (that is, selecting the highest-probability action with probability 1) leads to an improved performance in uniform random 4 x 4 matrix games using random 3 x 3 abstractions when playing against the Nash equilibrium of the full 4 x 4 game for the opponent (Ganzfried, Sandholm, and Waugh 2012). These results were based off simulations that were statistically significant at the 95 percent confidence level, and it would be interesting to provide a formal proof. Furthermore, that paper provided a conjecture for the specific supports of the games for which the approach would improve or not change performance, which was also based on statistically-significant simulations. It would be interesting to prove this formally as well, and to generalize the results to games of arbitrary size. On a broader level, there is relatively little theoretical understanding for why the postprocessing approaches — which one would expect to make the strategies more predictable — have been shown to be consistently successful. Surprisingly, the improvements in empirical performance do not necessarily come at the expense of worst-case exploitability, and a degree of thresholding has been demonstrated to actually reduce exploitability for a limit Texas hold 'em agent (Ganzfried, Sandholm, and Waugh 2012).

Acknowledgements

The competition was organized by Professor Tuomas Sandholm, and the agent was created by Noam Brown, Sam Ganzfried, and Tuomas Sandholm. Some of the work described was performed while the author was a student at Carnegie Mellon University before the completion of his Ph.D.; the article reflects the views of the author alone and not necessarily those of Carnegie Mellon University. The work done at Carnegie Mellon University was supported by the National Science Foundation under grants IIS-1320620, IIS-0964579, and CCF-1101668, as well as XSEDE computing resources provided by the Pittsburgh Supercomputing Center.

Note

1. See the *Gurobi Optimizer Reference Manual Version 6.0* available from Gurobi Optimization, Inc. (www.gurobi.com).

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. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence* (IJCAI) San Francisco: Morgan Kaufmann Publishers.

Bowling, M.; Burch, N.; Johanson, M.; and Tammelin, O. 2015. Heads-Up Limit Hold'em Poker Is Solved. *Science* 347(6218): 145–149.

Brown, N.; Ganzfried, S.; and Sandholm, T. 2015. Hierarchical Abstraction, Distributed Equilibrium Computation, and Post-Processing, with Application to a Champion No-Limit Texas Hold'em Agent. In *Proceedings of the International Conference on Autonomous Agents and Multi-Agent Systems* (AAMAS). New York: Association for Computing Machinery.

Burch, N.; Johanson, M.; and Bowling, M. 2014. Solving Imperfect Information Games Using Decomposition. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence* (AAAI). Palo Alto, CA: AAAI Press.

Chen, K., and Bowling, M. 2012. Tractable Objectives for Robust Policy Optimization. In *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012* (NIPS). Red Hook, NY: Curran Associates.

Ganzfried, S., and Sandholm, T. 2013. Action Translation in Extensive-Form Games with Large Action Spaces: Axioms, Paradoxes, and the Pseudo-Harmonic Mapping. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence* (IJCAI). Palo Alto, CA: AAAI Press

Ganzfried, S., and Sandholm, T. 2015. Endgame Solving in Large Imperfect-Information Games. In *Proceedings of the International Conference on Autonomous Agents and Multi-Agent Systems* (AAMAS). Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems.

Ganzfried, S.; Sandholm, T.; and Waugh, K. 2012. Strategy Purification and Thresholding: Effective Non-Equilibrium Approaches for Playing Large Games. In *Proceedings of the International Conference on Autonomous Agents and Multi-Agent Systems* (AAMAS). Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems.

Gilpin, A., and Sandholm, T. 2006. A Competitive Texas Hold'em Poker Player via Automated Abstraction and Real-Time Equilibrium Computation. In *Proceedings of the Twenty-First National Conference on Artificial Intelligence*. Palo Alto, CA: AAAI Press.

Gilpin, A., and Sandholm, T. 2007a. Better Automated Abstraction Techniques for Imperfect Information Games, with Application to Texas Hold'em Poker. In *Proceedings of the International Conference on Autonomous Agents and Multi-Agent Systems* (AAMAS). Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems.

Gilpin, A., and Sandholm, T. 2007b. Lossless Abstraction of Imperfect Information Games. *Journal of the ACM* 54(5).

Gilpin, A.; Sandholm, T.; and Sørensen, T. B. 2008. A Heads-Up No-Limit Texas Hold'em Poker Player: Discretized Betting Models and Automatically Generated Equilibrium-Finding Programs. In *Proceedings of the International Conference on Autonomous Agents and Multi-Agent Systems* (AAMAS). Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems.

Gordon, P. 2011. *Phil Gordon's Little Gold Book: Advanced Lessons for Mastering Poker 2.0*. New York: Simon & Schuster, Inc. / Gallery Books.

Hoda, S.; Gilpin, A.; Peña, J.; and Sandholm, T. 2010. Smoothing Techniques for Computing Nash Equilibria of Sequential Games. *Mathematics of Operations Research* 35(2): 494–512.

Johanson, M.; Bard, N.; Burch, N.; and Bowling, M. 2012. Finding Optimal Abstract Strategies in Extensive-Form Games. In *Proceedings of the 26th AAAI Conference on Artificial Intelligence* (AAAI). Palo Alto, CA: AAAI Press.

Johanson, M.; Burch, N.; Valenzano, R.; and Bowling, M. 2013. Evaluating State-Space Abstractions in Extensive-Form Games. In *Proceedings of the International Conference on Autonomous Agents and Multi-Agent Systems* (AAMAS). Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems.

Johanson, M. 2013. Measuring the Size of Large No-Limit Poker Games. Technical report, University of Alberta.

Koller, D.; Megiddo, N.; and von Stengel, B. 1994. Fast Algorithms for Finding Randomized Strategies in Game Trees. In *Proceedings of the 26th ACM Symposium on Theory of Computing* (STOC), 750–760. New York: Association for Computing Machinery.

Kroer, C., and Sandholm, T. 2014. Extensive-Form Game Abstraction with Bounds. In *Proceedings of the 15th ACM Conference on Economics and Computation* (EC). New York: Association for Computing Machinery.

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 22: 23rd Annual Conference on Neural Information Processing Systems* (NIPS). Red Hook, NY: Curran Associates.

Sandholm, T. 2015. Steering Evolution Strategically: Computational Game Theory and Opponent Exploitation for Treatment Planning, Drug Design, and Synthetic Biology. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence* (AAAI). Senior Member Blue Skies Subtrack. Palo Alto, CA: AAAI Press

Shi, J., and Littman, M. 2002. Abstraction Methods for Game Theoretic Poker. In *CG '00: Revised Papers from the Second International Conference on Computers and Games*. London, UK: Springer-Verlag.

Waugh, K.; Zinkevich, M.; Johanson, M.; Kan, M.; Schnizlein, D.; and Bowling, M. 2009. A Practical Use of Imperfect Recall. In *Proceedings of the Eighth Symposium on Abstraction, Reformulation and Approximation* (SARA). Palo Alto, CA: AAAI Press.

Zinkevich, M.; Bowling, M.; Johanson, M.; and Piccione, C. 2007. Regret Minimization in Games with Incomplete Information. In *Advances in Neural Information Processing Systems 20, Proceedings of the Twenty-First Annual Conference on Neural Information Processing Systems* (NIPS). Red Hook, NY: Curran Associates.

Sam Ganzfried recently completed his Ph.D. in computer science at Carnegie Mellon. He has an A.B. in math from Harvard. His interests are in artificial intelligence, game theory, multiagent systems, multiagent learning, large-scale optimization, large-scale data analysis and analytics, and knowledge representation. Ganzfried's dissertation, "Computing Strong Game-Theoretic Strategies and Exploiting Suboptimal Opponents in Large Games," was on scalable approaches for computing game-theoretic solution concepts and learning in imperfect-information games. In addition to Claudico, he created the agent Tartanian7, which came in first in the 2014 Annual Computer Poker Competition. He organized the AAAI Workshop on Computer Poker and Imperfect Information in 2014 and 2015 and organized the first tutorial on Computer Poker at the Conference on Economics and Computation in summer 2016.