Tartanian5: A Heads-Up No-Limit Texas Hold'em Poker-Playing Program*

Sam Ganzfried and Tuomas Sandholm
Computer Science Department
Carnegie Mellon University
{sganzfri, sandholm}@cs.cmu.edu

Abstract

We present an overview of Tartanian5, a no-limit Texas Hold'em agent which we submitted to the 2012 Annual Computer Poker Competition. The agent plays a game-theoretic approximate Nash equilibrium strategy. First, it applies a potential-aware, perfect-recall, automated abstraction algorithm to group similar game states together and construct a smaller game that is strategically similar to the full game. In order to maintain a tractable number of possible betting sequences, it employs a discretized betting model, where only a small number of bet sizes are allowed at each game state. The strategies for both players are then computed using an improved version of Nesterov's excessive gap technique specialized for poker. To mitigate the effect of overfitting, we employ an ex-post purification procedure to remove actions that are played with small probability. One final feature of our agent is a novel algorithm for interpreting bet sizes of the opponent that fall outside our model. We describe our new approach in detail, and present theoretical and empirical advantages over prior approaches. Finally, we briefly describe ongoing research in our group involving real-time computation and opponent exploitation, which will hopefully be incorporated into our agents in future years.

Introduction

Developing effective strategies for agents in multiagent systems is an important and challenging problem. It has received significant attention in recent years from several different communities—in part due to the competitions held at top conferences (e.g, the computer poker, robo-soccer, and trading agent competitions). As many domains are so large that solving them directly (i.e., computing a Nash equilibrium or a solution according to some other relevant solution concept) is computationally infeasible, some amount of approximation is necessary to produce agents.

Specifically, significant work has been done on computing approximate game-theory-based strategies in large imperfect-information games (Sandholm 2010). This work typically follows a three-step approach, which is depicted in Figure 1. First, a smaller game G' is constructed which is strategically similar to the original game G. Initially this was done by a hand-crafted abstraction procedure (Billings et al. 2003), and more recently has been done by *automated abstraction algorithms* (Gilpin and Sandholm 2006; 2007; Shi and Littman 2002). Second, an *equilibrium-finding algorithm* is run on G' to compute an ϵ-equilibrium σ' (Hoda et al. 2010; Zinkevich et al. 2007). Third, a *reverse mapping* is applied to σ' to compute an approximate equilibrium σ in the full game G (Gilpin, Sandholm, and Sørensen 2008; Schnizlein, Bowling, and Szafron 2009).

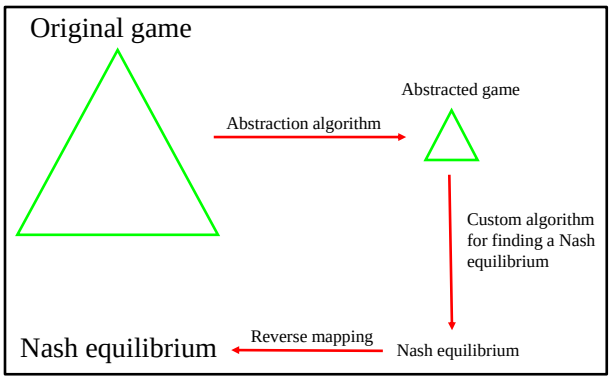

Figure 1: General game-theoretic approach for solving large games.

No-limit Texas Hold'em poker is an especially challenging domain, as its game tree contains 10^{71} states (Gilpin, Sandholm, and Sørensen 2008)[1]. State of the art equilibrium-finding algorithms can only scale to games with approximately 10^{12} game tree states (Hoda et al. 2010; Zinkevich et al. 2007); hence, significant abstraction and approximation is necessary to apply a game-theoretic approach. Interestingly, only two agents from the 2011 competition were based on approximate equilibrium strategies,

*This material is based upon work supported by the National Science Foundation under grants IIS-0964579, IIS-0905390, and CCF-1101668. We also acknowledge Intel Corporation and IBM for their machine gifts.

[1]This is for the two-player version with starting stacks of 500 big blinds where stacks are reset after each hand. Initially the Annual Computer Poker Competition used 500 big blind starting stacks, but for the last few years 200 big blind stacks were used; thus, the number of game states is actually smaller than 10^{71}.

while 4 of the 7 agents that reported a description used handcrafted expert systems. The final agent used a case-based approach designed to imitate the strategies of top-performing agents in previous years' competitions (Rubin and Watson 2012). In 2010 two of the five agents submitted to the two-player no-limit competition employed approximate equilibrium strategies, including the previous variant of our program, Tartanian4, which finished in first place in the total bankroll competition and in third place in the bankroll instant run-off competition.

In this paper we describe our new no-limit approximate-equilibrium agent, Tartanian5. The primary components of our agent are a potential-aware automated abstraction algorithm, discretized betting model, custom equilibrium-finding algorithm, ex-post purification procedure, and a novel reverse mapping algorithm. We describe our new reverse mapping algorithm in detail, as it has changed significantly from the algorithm used in our prior agents. We present theoretical and empirical advantages of our new approach over prior approaches in simplified settings, which we expect to translate directly to improved performance in Texas Hold'em.

An overview of Tartanian5

In this section we present the different technical components of our agent Tartanian5, and discuss the key differences from prior versions.

Automated card abstraction

In order to reduce to size of the game tree to make equilibrium computation feasible, we applied a *potential-aware* automated abstraction algorithm (Gilpin, Sandholm, and Sørensen 2007). The algorithm uses clustering to group hands together that have similar histograms over future outcomes, and integer programing to determine how many children to allocate to each node.

Many abstraction algorithms group hands together that have similar expected hand strength against a uniform rollout of the opponent's hands. Our approach takes into account that the strength of hands can vary significantly over the course of a hand, and some hands with similar expected hand strength should be played quite differently. For example, QJ suited and 55 both win about 60% of the time against a random hand, but should be played very differently. Our algorithm uses *perfect recall*, in which each player is forced to remember his hole cards and the exact sequence of community cards throughout the hand. Future work will involve considering *imperfect recall abstractions*, in which agents potentially forget information that they once knew (Waugh et al. 2009).

The previous version of our program, Tartanian4, used a potential-aware card abstraction with branching factors of 12, 25, 6, and 6 for the four betting rounds. Perhaps counter-intuitively, this year we decided to use a smaller abstraction; Tartanian5 uses an abstraction with branching factors of 8, 12, 4, and 4. The reason for this change was that we realized the 12-25-6-6 abstraction was so large that our equilibrium-finding algorithm was not able to converge sufficiently fast.

Define the *exploitability* of strategy σ_i to be the difference between the value of the game to player v_i and the performance of σ_i against its nemesis, formally:

$$\text{expl}(\sigma_i) = v_i - \min_{\sigma_{-i}} u_i(\sigma_i, \sigma_{-i}).$$

Define the total exploitability of an agent which plays strategy σ_1 as player 1 and strategy σ_2 as player 2 to be

$$\frac{\text{expl}(\sigma_1) + \text{expl}(\sigma_2)}{2}.$$

Even within its abstracted game, Tartanian4 had exploitability of approximately 0.8 big blinds per hand. This is noteworthy for several reasons. First, notice that the strategy of always folding would have an exploitability of 0.75 big blinds per hand (since it would lose 0.5 big blinds when it is in the small blind and 1 big blind when it is in the big blind). So our program which came in first place in the 2010 no-limit Texas Hold'em division of the computer poker competition was more exploitable than always folding! Furthermore, the exploitability of 0.8 BB/hand assumed that the opponent was restricted to the same card and betting abstraction; the full game exploitability of our strategies was probably significantly higher. This indicates that there is still a long way to go in terms of developing strong game-theoretic agents for no-limit Texas Hold'em. It also suggests that perhaps worst-case exploitability is not the most useful metric for evaluating the quality of strategies.

This year, we decided to use a smaller abstraction that had a smaller exploitability in its abstraction. At the time of our decision, the 8-12-4-4 abstraction had an exploitability of 0.55 BB/hand, while the 8-4-3-3 abstraction had an exploitability of 0.13 BB/hand (and the 12-25-6-6 abstraction had an exploitability of 0.8 BB/hand). We played these three agents against each other, and the 8-12-4-4 agent (with intermediate abstraction size and exploitability) beat the other two. So we chose that for our abstraction size. The current exploitability of our strategy is approximately 0.3 BB/hand within the abstraction.

Discretized betting model

In limit Texas Hold'em, there is a fixed betting size; thus at each game state, a player must choose between 3 options: folding, calling/checking, or betting. By contrast in no-limit Texas Hold'em, a player can bet any amount (up to the number of chips in his stack) at any game state. This causes an enormous blowup in the size of the game tree. While two-player limit Texas Hold'em has about 10^{18} game states in its tree, two-player no-limit Texas Hold'em has about 10^{71} states. Thus, in order to develop a no-limit agent, some form of betting abstraction or discretization is needed.

Our approach is to discretize the betting space into a small number of allowable bet sizes at each game state (Gilpin, Sandholm, and Sørensen 2008). Folding, calling/checking, and going all-in are available at every game state. In addition, one or two intermediate bet sizes are also available at each game state (which ones are available depend on the betting history so far). These include bet sizes of $\frac{1}{2}$-pot, $\frac{2}{3}$-pot, and pot. This year we made some changes to our discretization, and allowed bets of 1.5-pot and 2-pot in certain situations.

Equilibrium finding

Once we fixed our card abstraction and betting model, we ran a perl script which automatically generates C++ code for running an equilibrium-finding algorithm on the abstract game (Gilpin, Sandholm, and Sørensen 2008). The algorithm we used is a gradient-based algorithm which uses an improved version of Nesterov's excessive gap technique specialized for poker (Hoda et al. 2010). The computation was run using 64 cores at the Pittsburgh Supercomputing center on the world's largest shared-memory supercomputer (called blacklight).

Purification

Rather than play the exact strategies output by our equilibrium-finding algorithm, we decided to modify the action probabilities by preferring the higher-probability actions. The intuition for doing this is that we should ignore actions that are played with small probability in the abstract equilibrium, as they are likely due to abstraction coarseness, overfitting, or failure of the equilibrium-finding algorithm to fully converge (Ganzfried, Sandholm, and Waugh 2012).

In 2010, we submitted two versions of our Tartanian4 agent; one that employed full *purification*, and one that employed *thresholding* using a threshold of 0.15. The purification agent played the highest-probability action with probability 1 at each information set, and played the other actions with probability 0 (randomizing for ties). The thresholded agent rounded all action probabilities below 0.15 to 0, then renormalized. The purification agent beat the thresholded agent head-to-head, and outperformed it against all competition opponents except one. Thus, we decided to use full purification in Tartanian5.

Reverse mapping

While the strategies we computed assume the opponent is following our discretized betting model, our actual opponent may in fact make bet sizes that fall outside of our model. Thus a *reverse mapping* algorithm is necessary to map his bet size to one of the bet sizes in our model. Tartanian5 uses a new reverse mapping algorithm that differs from our algorithm of previous years, as well as the algorithms used by prior agents.

Discussion of our new reverse mapping algorithm and comparison to prior approaches

In this section, we present our new reverse mapping and discuss advantages over prior approaches. We first present several basic properties that a reverse mapping should satisfy, and show that each of the prior mappings either fails to satisfy some of the properties, or produces strategies that are highly exploitable. Furthermore, the prior mappings are based on heuristics and do not have any theoretical justification. By contrast, our mapping satisfies all of the properties, has low exploitability, and is theoretically motivated as the solution to simplified poker games.

Problem formulation

Suppose our opponent makes a bet of size $x \in [A, B]$, where A denotes the largest betting size in our abstraction less than or equal to x, and B denotes the smallest betting size in our abstraction greater than or equal to x (we require that $0 \leq A < B$.) The *reverse mapping problem* is to determine whether we should map x to A or to B (perhaps probabilistically). Thus, our goal is to find a function $f_{A,B}(x)$, which denotes the probability that we map x to A ($1 - f_{A,B}(x)$ denotes the probability that we map x to B); this is our *reverse mapping function*. We assume the pot size is 1, and that all values have been normalized accordingly. For simplicity, we will sometimes write $f_{A,B}$ just as f.

Basic properties a reverse mapping algorithm should have

It seems clear that every reverse mapping function should satisfy the following basic properties:

1. $f_{A,B}(A) = 1$
2. $f_{A,B}(B) = 0$
3. $f_{A,B}$ is monotonically decreasing
4. $\forall k > 0, x \in [A, B], f_{kA,kB}(kx) = f_{A,B}(x)$

Boundary constraints If the opponent takes an action that is actually in our abstraction, then it is natural to map his action x to the corresponding action with probability 1. Hence we require that $f(A) = 1$ and $f(B) = 0$.

Monotonicity As the opponent's action x moves away from A towards B, it is natural to require that the probability of his action being mapped to A decreases. Thus we require that f be monotonically decreasing.

Scale invariance This condition requires that scaling A, B, and x by some multiplicative factor $k > 0$ does not affect our reverse mapping. In poker for example, it is common to scale all bet sizes by the size of the big blind or the size of the pot.

Survey of existing reverse mapping algorithms

In this section, we will present several reverse mapping algorithms have been proposed in the literature (Gilpin, Sandholm, and Sørensen 2008; Rubin and Watson 2012; Schnizlein, Bowling, and Szafron 2009). For each algorithm, we will check whether it satisfies the above properties, and discuss additional strengths and limitations.

Deterministic arithmetic The deterministic arithmetic algorithm is perhaps the simplest. If $x < \frac{A+B}{2}$, then x is mapped to A with probability 1; otherwise x is mapped to B with probability 1.

This mapping satisfies Properties 1, 2, and 4, but not Property 3. (It is constant from $x = A$ to $\frac{A+B}{2}$ and from $x = \frac{A+B}{2}$ to B, and is therefore not monotonically decreasing). Furthermore, it is potentially highly exploitable. For example, suppose A is a pot-size bet and B is an all-in. Then the opponent could significantly exploit us by betting slightly under $\frac{A+B}{2}$ with his strong hands.

Randomized arithmetic

$$f_{A,B}(x) = \frac{B-x}{B-A}$$

This mapping satisfies all four properties; but it is exploitable in the same way as the deterministic arithmetic mapping (though to a lesser extent).

Deterministic geometric The motivation behind this algorithm is to reduce the level of exploitability of the arithmetic approach by making our threshold x^* at the point where the ratios of x^* to A and B to x^* are the same (rather than the differences). Specifically, if $\frac{A}{x} > \frac{x}{B}$ then x is mapped to A with probability 1; otherwise x is mapped to B with probability 1. Thus, the threshold will be $x^* = \sqrt{AB}$ rather than $\frac{A+B}{2}$. This will diminish the effectiveness of the exploitation described above; namely to make a large value bet just below the threshold.

Like the deterministic arithmetic mapping, this mapping satisfies Properties 1, 2, and 4, but not Property 3. An additional undesirable property is that $f(x) = 0$ for all x at $A = 0$ (i.e., A corresponds to the check action). If $A = 0$, $B = 1$, and $x = 0.1$ for example, we will believe the opponent's small bet of 0.1 times the pot is actually a full pot-sized bet. This function also behaves undesirably as $A \to 0$.

Randomized geometric 1

$$g_{A,B}(x) = \frac{\frac{A}{x} - \frac{A}{B}}{1 - \frac{A}{B}}$$

$$h_{A,B}(x) = \frac{\frac{x}{B} - \frac{A}{B}}{1 - \frac{A}{B}}$$

$$f_{A,B}(x) = \frac{g_{A,B}(x)}{g_{A,B}(x) + h_{A,B}(x)}$$

Like the deterministic geometric mapping, this violates monotonicity at $A = 0$. This mapping has been used in previous competition agents (e.g., Hyperborean from Alberta (Schnizlein, Bowling, and Szafron 2009) and Sartre from Auckland (Rubin and Watson 2012)).

Randomized geometric 2

$$f_{A,B}(x) = \frac{AB(A+B)}{(B-A)(x^2+AB)} + \frac{A}{A-B}$$

This mapping behaves similarly to the first randomized geometric mapping, and also violates monotonicity at $A = 0$. This was the mapping we used in our Tartanian4 agent. Note that both randomized geometric mappings have $f_{A,B}(\sqrt{AB}) = \frac{1}{2}$.

Our new reverse mapping algorithm

The motivating problem for our new reverse mapping is a half-street no-limit clairvoyance game (Ankenman and Chen 2006):

- There are two players, P1 and P2.
- The initial size of the pot is \$1.
- Both players have stacks of size n.
- P1 is clairvoyant, and his hand is randomly drawn from a distribution that is half winning hands and half losing hands.
- P1 is allowed to bet a fixed amount s, where $s \in [0, n]$.
- P2 is allowed to call or fold.

For a given bet size s, equilibrium strategies for both players are as follows (Ankenman and Chen 2006):

- P1 bets s with probability 1 with a winning hand.
- P1 bets s with probability $\frac{s}{1+s}$ with a losing hand (and checks otherwise).
- P2 calls with probability $\frac{1}{1+s}$.

In fact, these betting and calling frequencies are shown to be optimal in many other no-limit poker games as well (Ankenman and Chen 2006).

Using this as motivation, our reverse mapping will be the solution to

$$f_{A,B}(x) \cdot \frac{1}{1+A} + (1 - f_{A,B}(x)) \cdot \frac{1}{1+B} = \frac{1}{1+x}.$$

Specifically, our mapping is:

$$f_{A,B}(x) = \frac{(B-x)(1+A)}{(B-A)(1+x)}.$$

This is the only mapping consistent with calling a bet of size x with probability $\frac{1}{1+x}$ for all $x \in [A, B]$.

This mapping satisfies all of the properties, is well-behaved as $A \to 0$, and is not as susceptible to the exploitations previously described. The threshold x^* for which $f_{A,B}(x^*) = \frac{1}{2}$ is

$$x^* = \frac{A+B+2AB}{A+B+2}.$$

Examples

In Figure 2 we plot all four randomized reverse mapping algorithms using $A = 0.01$ and $B = 1$. As the figure shows, both of the randomized geometric algorithms map a bet size of 0.1-pot to A and B with roughly equal probability, while the corresponding threshold of the arithmetic algorithm is around 0.5-pot and the new algorithm is around 0.35-pot. In this case, the algorithms differ quite significantly.

In Figure 3, we plot the algorithms using $A = 1$ and $B = 4$. In this case our mapping is relatively similar to the geometric mappings, while the arithmetic mapping differs significantly from the others.

Additional research relevant to poker and imperfect-information games

In addition to the research that went into Tartanian5, our group has worked on several other topics highly related to imperfect-information games, and to poker in particular.

- We have developed algorithms for computing an ϵ-equilibrium in the endgame of a 3-player no-limit Texas Hold'em tournament for provably small ϵ (Ganzfried and Sandholm 2008; 2009).

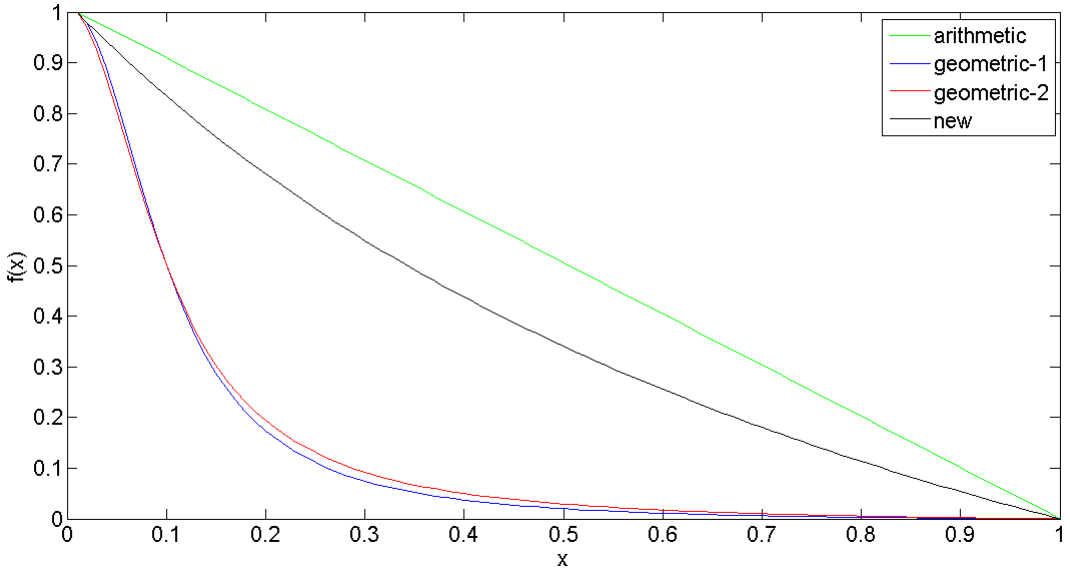

Figure 2: Plots of the four randomized reverse mapping algorithms using $A = 0.01$ and $B = 1$.

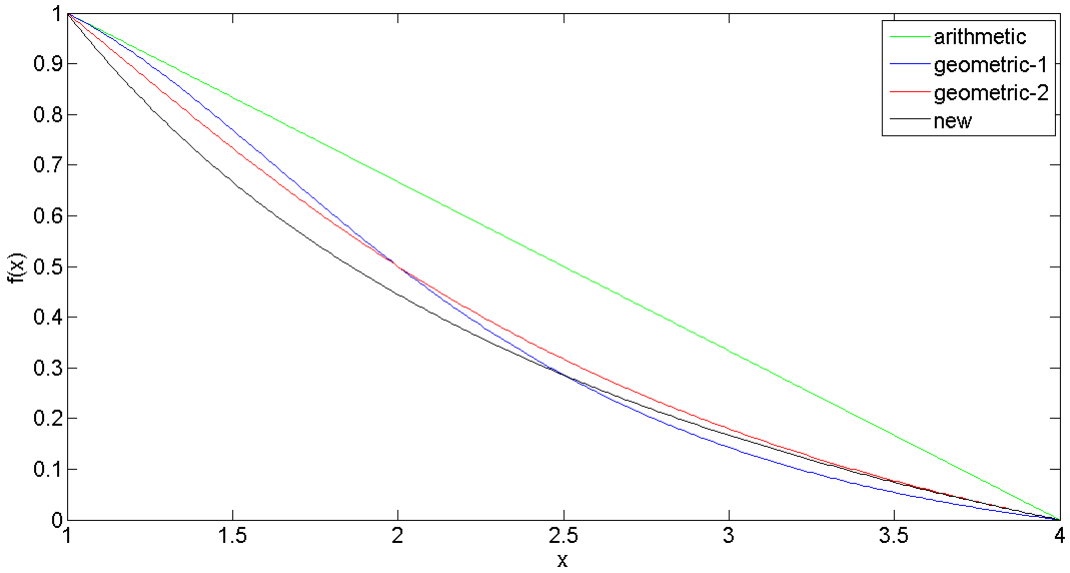

Figure 3: Plots of the four randomized reverse mapping algorithms using $A = 1$ and $B = 4$.

- We have developed an algorithm for quickly solving river subgames in real time in limit Texas Hold'em that exploits qualitative information on the structure of equilibrium strategies (Ganzfried and Sandholm 2010).
- We have a developed a opponent exploitation algorithm that leads to improved performance against a variety of agents in two-player limit Texas Hold'em (Ganzfried and Sandholm 2011). It works by constructing a model of the opponent based on deviations from a precomputed approximate equilibrium, and computing approximate best responses in real time.
- We have developed algorithms for exploiting suboptimal opponents that guarantee at least the value of the game in expectation (Ganzfried and Sandholm 2012). Our algorithms led to a significant performance improvement against several opponent classes in Kuhn poker.

Conclusion

We introduced our no-limit Texas Hold'em agent Tartanian5 and described its primary components: an automated abstraction algorithm, a discretized betting model, a custom equilibrium-finding algorithm, an ex-post purification procedure, and a new reverse mapping algorithm. We also briefly described ongoing research in our group involving real-time computation and opponent exploitation, which will hopefully be incorporated into our agents in future years. We described several theoretical and empirical advantages of our new reverse mapping algorithm over prior approaches. Specifically, all prior mappings either fail to satisfy some basic properties or are highly exploitable; furthermore, they are all ad hoc and lack any theoretical justification. By contrast, our mapping satisfies all of the properties, has low exploitability, and is theoretically motivated as the solution to simplified poker games. Future work will involve doing a thorough comparison of the various approaches against agents submitted to this year's competition.

References

Ankenman, J., and Chen, B. 2006. *The Mathematics of Poker*. ConJelCo LLC.

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)*.

Ganzfried, S., and Sandholm, T. 2008. Computing an approximate jam/fold equilibrium for 3-player no-limit Texas Hold'em tournaments. In *International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*.

Ganzfried, S., and Sandholm, T. 2009. Computing equilibria in multiplayer stochastic games of imperfect information. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI)*.

Ganzfried, S., and Sandholm, T. 2010. Computing equilibria by incorporating qualitative models. In *International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*.

Ganzfried, S., and Sandholm, T. 2011. Game theory-based opponent modeling in large imperfect-information games. In *International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*.

Ganzfried, S., and Sandholm, T. 2012. Safe opponent exploitation. In *Proceedings of the ACM Conference on Electronic Commerce (ACM-EC)*.

Ganzfried, S.; Sandholm, T.; and Waugh, K. 2012. Strategy purification and thresholding: Effective non-equilibrium approaches for playing large games. In *International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*.

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 National Conference on Artificial Intelligence (AAAI)*.

Gilpin, A., and Sandholm, T. 2007. Lossless abstraction of imperfect information games. *Journal of the ACM* 54(5).

Gilpin, A.; Sandholm, T.; and Sørensen, T. B. 2007. Potential-aware automated abstraction of sequential games, and holistic equilibrium analysis of Texas Hold'em poker. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*.

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 *International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*.

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.

Rubin, J., and Watson, I. 2012. Case-based strategies in computer poker. *AI Communications* 25(1):19–48.

Sandholm, T. 2010. The state of solving large incomplete-information games, and application to poker. *AI Magazine* 13–32. Special issue on Algorithmic Game Theory.

Schnizlein, D.; Bowling, M.; and Szafron, D. 2009. Probabilistic state translation in extensive games with large action sets. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI)*.

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*. 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)*.

Zinkevich, M.; Bowling, M.; Johanson, M.; and Piccione, C. 2007. Regret minimization in games with incomplete information. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NIPS)*.