Investigating the Feasibility of a Perfect Win Rate in Schnapsen Using Unconstrained Agents

Yamato Werner Kato and Misha Milashevskiy

Vrije Universiteit Amsterdam, De Boelelaan 1105, 1081 HV Amsterdam, Netherlands
y.w.kato@student.vu.nl, m.milashevskiy@student.vu.nl

Abstract. This paper investigates the upper bound of achievable performance in the Schnapsen card game by progressively removing the epistemic and aleatoric constraints that normally govern fair play. Without altering the game engine, we construct a series of similar bots with near-identical search architecture but different privileges granted, from imperfect-information play to oracle access and active game state manipulation. The agents are evaluated against a set of three standard Schnapsen opponents with different strategies. With the strongest version reaching 100% win rate in all games played, we conclude that a perfect win rate is achievable in Schnapsen once all uncertainty is removed.

Keywords: Game Theory · Imperfect Information · Search Algorithms · Schnapsen · State Manipulation

Table of Contents

1 Introduction

Game Theory has long been a benchmark in testing the capabilities of Artificial Intelligence. Specifically, strategic, adversarial, sequential and stochastic decision-making games serve as a testbed to observe the strengths and weaknesses of AI and to compare performance with human capabilities [8]. AlphaGo and DeepBlue serve as prime examples of such AI, as they have shown the advantages of complex, deep calculations over human intuition and simpler heuristics. Non-deterministic games, however, pose a much greater problem to solve for agents, as calculation power alone fails to solve most games within most practical time-frames. With the addition of an adversarial component, agents must also account for the decisions of an opponent who is actively attempting to minimize their success, further increasing the complexity of the problem and requiring strategies that go beyond brute-force computation.

An example of such a problem is the two-player adversarial card game Schnapsen. Consisting of two distinct phases, the first being non-deterministic, stochastic, and played with imperfect-information, and the second being a perfect-information, deterministic phase, Schnapsen offers a challenging environment well-suited in testing a wide range of problem-solving abilities for both humans and agents. Although consisting of only twenty cards, the game requires not only different playing styles and strategies, but also strong adaptability, assuming that the participating parties play within the legal bounds of the game.

In this paper, our aim is to analyze the upper bound of performance for an agent built around the capacity to disregard these legal bounds through the progressive removal of constraints.

1.1 Hypotheses

The hypotheses in this paper are based on the theoretical distinctions of epistemic and aleatoric uncertainty, both prominent in the pursuit of achieving a perfect win rate in Schnapsen.

Hypothesis 1 Passive constraint relief (Oracle Access) will yield a statistically significant improvement over the baseline but will fail to achieve a perfect win rate.

Hypothesis 2 Active constraint relief (State Manipulation) will significantly outperform Oracle Access. We predict that the ability to optimize card distribution is a stronger predictor of victory than the ability to see hidden cards.

Hypothesis 3 The Final Optimized Agent will achieve a win rate of 100% against all opponent types, effectively demonstrating that the game of Schnapsen is solved once stochasticity is removed.

2 Schnapsen

The rules described in this section follow the standard presented by Wisser [10]. As mentioned above, Schnapsen is a two-player adversarial card game. The game is played with a deck of twenty cards consisting of the Ace, Ten, King, Queen, and Jack of each suit. The game begins with each player being dealt 5 cards, the rest of the deck forming the talon. After each trick, players will draw one card from the talon to keep their hand size at five cards during the first phase of the game. One card from the talon is randomly assigned as the trump card, asserting the suit to be the trump suit, and all cards of that suit gain the highest priority in trick scoring.

The game is won by being the first player to reach at least 66 points. Points are awarded based on the ranks of the cards won in tricks, with additional points to be gained through special combinations such as marriages. Marriages consist of the King and Queen of a suit and yield 20 points, with royal marriages (King and Queen of the trump suit) being worth 40 points, and can only be played by the leading player through declaration. A trump exchange (exchanging the trump card with the Jack of the trump suit) may also be declared by the leading player. These special moves introduce trade-offs between immediate point gains and long-term positional advantages.

Once the cards in the talon are exhausted, the talon is considered closed and will transition the game into its second phase, where no further cards are drawn and all remaining cards are known to both players. The talon may also be closed voluntarily by the leading player at any time during the first phase, immediately forcing the game to transition into the second phase. In the second phase, players are required to follow suit and attempt to win the trick if possible, making the remainder of the game fully deterministic, introducing a strategic commitment mechanism, as closing the talon trades future card draws for full information. The dichotomy between phase 1 and phase 2 poses the challenge of adaptability to any player, which is why our agents will use two different approaches depending on the current phase of the game.

3 Related Work

Although research on AI in Schnapsen is limited, methods to manage the imperfect information prominent in the game include using Sims tables [9], Counterfactual Regret Minimization (CFR) [11] or Information Set Monte Carlo Tree Search (IS-MCTS) [4]. Some adaptations of IS-MCTS attempt to improve performance by modeling and predicting human decision-making [1]. However, these methods tend to be constrained by computational limits, specifically depth-limits in searches, which has been shown to hinder an agent's ability to adapt to opponent strategies beyond the search horizon [5]. In contrast, recent agent architectures have touched upon oracle access, the freedom to observe the full or partial hidden game state to aid in training and/or search, and have been found to increase performance in imperfect information domains [2]. This advancement in combining search with learning to handle imperfect information games

was significantly advanced by the ReBeL algorithm, which demonstrated that search over a belief state is feasible and yields notable results [3] [6]. Another study utilizing the "Student of Games" algorithm combines guided search and self-play learning that improves the navigation of large game trees and achieves strong performance in both perfect- and imperfect-information games [7]. Our work makes use of not only the oracle methodology to evaluate the theoretical performance ceiling, but also active state manipulation.

4 Theoretical Framework

In order to investigate the differences in performance between "constrained" and "unconstrained" agents, we will first define what "constraint" means in this context and how it will theoretically affect the agent's performance. Furthermore, it is important to note that any constraint relief will in this context be related to manipulating the agent's logic but is limited to the agent file only, meaning the main game file, or any other file for that matter, may not be altered.

We differentiate between two main ways of constraint relief, passive and active.

4.1 Passive Constraint Relief: Oracle Access

Passive constraint relief involves the removal of the information barrier, effectively granting the agent access to the full game state, including the opponent's hand and the cards in the talon. In the AI literature, an agent operating under such unconstrained conditions is formally referred to as an **Oracle** [2]. This is akin to a human player peeking the opponent's hand and the cards in the talon.

Unlike a standard player who must reason over a belief state (a probability distribution of possible hands), the Oracle operates on a singular, deterministic reality. This transformation allows us to isolate the value of information and investigate the theoretical performance gain achievable solely by resolving uncertainty without altering the physical mechanics of the game. For the remainder of this paper, we will refer to this mode of operation as **Oracle Access**.

4.2 Active Constraint Relief: State Manipulation

Active constraint relief is a much more drastic technique, as it involves the modification of the game state itself, rather than just the observation of it. While Oracle Access resolves the epistemic uncertainty (what is hidden?), the agent remains bound by the aleatoric uncertainty (the luck of the deal). A player with perfect information can still be dealt a mathematically unwinnable hand against a perfect opponent.

To overcome this, we introduce State Manipulation. In this mode, the agent is permitted to alter the distribution of cards, initially and retroactively, by exchanging weak cards from its own hand with stronger cards from the talon or

the opponent's hand. This relaxation allows for investigating the specific performance advantage gained by optimizing hand strength, independent of strategic planning. For the remainder of this paper, we will refer to this mode of operation as **State Manipulation**.

It is important to note that most changes across the agent versions focus on phase 1, but the final HonestBot also applies state manipulation at phase-2 decision points before selecting its move. The second phase can be solved with minimal difficulty using an alpha-beta algorithm, whereas the first phase poses greater challenges due to its stochastic nature.

5 Agent Architecture

The agent in this paper is constructed from the ground up to be compatible with progressive constraint relief. In the construction of the code, minimal changes were made throughout each version to keep the integrity of the research intact. We will start with a baseline agent as the basis upon which to build. Each version of the agent will progressively receive more privileges in terms of constraint relief, and we will observe the differences in win rate in addition to observing the changes in difficulty the agents might show against each opponent they face. This approach aims to clearly show the impact certain constraints impose on players both agentic and human.

The second phase is handled through alpha-beta search for all bots, but the final HonestBot additionally applies state manipulation in phase 2 before selecting its move.

5.1 The Baseline (HB0)

The first agent HB0 (HonestBot 0) is built to receive constraint relaxation privileges, though it is not yet using any illegal advantages. The agent performs an alpha-beta search under a single assumed completion of the hidden game state with a depth limited to 8. By sampling a single possible world consistent with the legally available information, it is quite similar to the RDeep Schnapsen bot, with the key distinction that HB0 commits to this fixed determination and applies a deterministic alpha-beta search rather than averaging over multiple sampled worlds. This results in strong tactical play within this assumption, though it remains vulnerable to any incorrectly assumed elements.

5.2 Oracle Access (HB1)

The HB1 agent is largely identical to HB0, although it differs in one crucial aspect. Instead of sampling a single hypothetical completion of the hidden state, it accesses the full game state through the `_peek_full_state(perspective)` function and performs the exact same alpha-beta search on this game state. Otherwise, no changes are made; the agent is algorithmically identical to HB0 with Oracle Access, therefore showcasing the differences in performance when breaching the information constraint.

5.3 State Manipulation (HB2)

In contrast to HB1 and HB0, HB2 utilizes the ability to actively manipulate the game state before starting the decision-making process. This state manipulation is only performed when the agent is leading the trick, as modifying card ownership after the leader has already selected a move would invalidate the engine's internal assumptions. Therefore, state manipulation in HB2 is strictly limited to decision points where the agent controls the leading action of a trick. While HB2 is following, the agent passively observes the true game state and computes its response using the same alpha-beta search. If the agent is the leader at the beginning of the game, it forces a strong opening hand of five cards comprised of any ace and all trump cards (excluding the trump card of the talon) by exchanging them from the opponent's hand or the talon with its own cards. The trump of the talon is not modified as this card is treated as immutable by the engine. From this point on, HB2 will also rig its hand to the best five cards after every trick as long as it is leading the trick. The best five cards are evaluated based on a predetermined ranking. In HB0 and HB1, an auxiliary component referred to as `OneFixedMoveBot` is used during the alpha-beta search to deterministically replay a single predetermined move when simulating future tricks. This allows the alpha-beta procedure to resolve complete hypothetical tricks by temporarily instantiating fixed-move leader and follower bots, without altering the behavior of the main agent. In these earlier agents, the component directly stored and replayed concrete `Move` objects, which was sufficient as long as the underlying game state remained unmodified. In HB2, this component is rewritten in accordance with the issues arising from state copying during search. Unlike HB0 or HB1, HB2 stores a descriptor of the intended move (including card rank, suit, and move type) and dynamically selects the corresponding legal move from the current perspective. Beyond this constraint relaxation, HB2 preserves the same search structure, depth limits, and heuristic evaluation as HB0 and HB1. This ensures that any differences in performance can be attributed solely to the state-manipulation.

5.4 Optimized Agent (HonestBot)

As shown in Fig. 1, this version is a refined, more efficient and capable version of HB2, retaining all previously introduced constraint relaxations in addition to addressing implementation inefficiencies, stabilizing search processes, and having more move options. HonestBot also expands the state manipulation privileges with the capability to rig its hand even when it's the follower. This is achieved by exchanging cards with everything except for the card the leader has played when HonestBot is not leading, avoiding the risk of crashing the engine through stealing a card the opponent already selected to play. This function is also able to handle rigging as follower even when the leader plays a marriage by adhering to many specific marriage cases that were identified as problematic. In the current implementation, this rigging procedure is applied in both phase 1 and phase 2: HonestBot manipulates the state before selecting moves even after the talon

is exhausted. During minimax recursion, most changes address cases where the legacy version accidentally maximized the opponent's outcome. The agent now correctly maximizes/minimizes without relying on fragile boolean flags and determines which players' utility to optimize based on stable bot identity. In terms of special moves, HonestBot now competently plays and handles marriage moves based on talon size, opponent trump exchange risk, payoff as well as robust handling for specific marriage cases that are not addressed otherwise. When the opponent leads a marriage in phase 1 and HonestBot is follower, it first checks whether it can win the trick with the cheapest available winning response (often the lowest trump); if not, it falls back to the standard search-based decision. HonestBot also prevents playing incorrect or mismatched marriage moves during search by storing a descriptor of the move, rather than a concrete Move object. A transposition table is introduced to cache canonical state features, leader move descriptor, player to move identity and depth, preventing redundant evaluation of identical states. A node budget is also added to regulate the search expansion, ensuring a valid best move is always available even under tight computational limits.

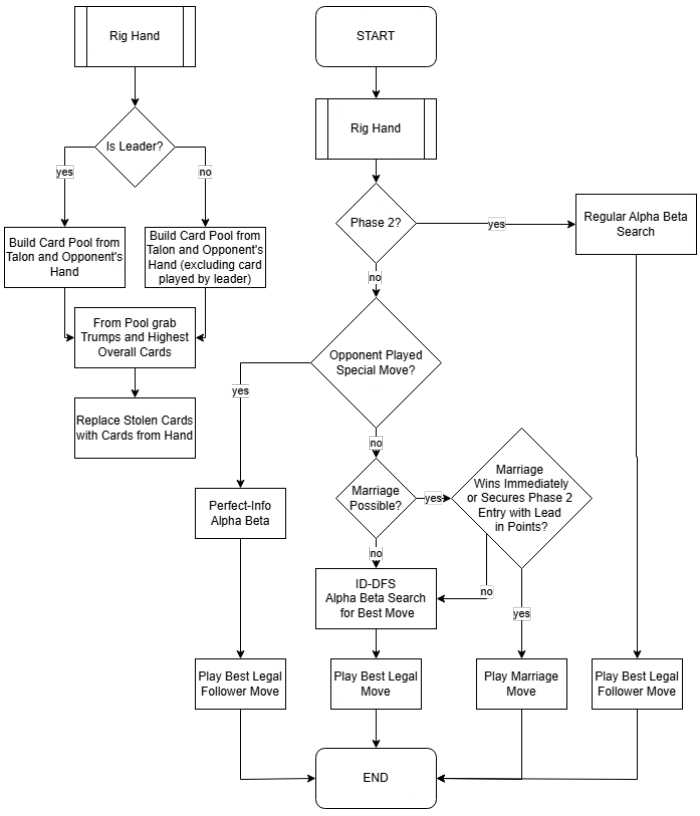

Fig. 1. Decision pipeline of the HonestBot

6 Experimental Setup

The aim of the experimental evaluation is to quantify the effect of progressive constraint relaxation on agent performance. As described in Section 5, the agents differ mostly, though not exclusively, in their active and passive constraints. Each agent is evaluated against a set of three opponent bots to represent a range of play styles and difficulty levels. The set includes both deterministic and stochastic agents, ensuring that performance differences are not limited to a single opponent profile. All experiments were performed on a 2023 M2 Pro Apple MacBook Pro with 16GB of memory.

6.1 Agents

The following agents are evaluated:

- **HB0**: The baseline agent using deterministic alpha-beta search under a single, randomly assumed completion of the hidden game state.
- **HB1**: An agent with oracle access, without any state manipulation.
- **HB2**: An agent with oracle access and state manipulation restricted to leader decision points.
- **HonestBot**: A fully unconstrained agent incorporating oracle access and optimized state manipulation across both phases, in addition to heuristic/search optimizations and proper marriage logic.

6.2 Opponents

- **RandBot**: A baseline agent simply selecting moves randomly from the set of all available legal moves at any decision point, not using search, reasoning, or inference.
- **BullyBot**: BullyBot is a deterministic, heuristic-based agent with an emphasis on aggressive play. At any point, if a trump card is available, BullyBot plays it. Otherwise, when following, the bot attempts to follow suit with a randomly selected matching card if available, otherwise selecting the highest-scoring legal regular move.
- **RDeep**: The toughest opponent, RDeep, performs many random roll outs of the game and evaluates moves to play by average outcome. RDeep is very similar to HB0 in reasoning over assumed completions of the hidden game state but evaluates moves using stochastic roll outs with random play instead of alpha-beta search, and selects the move with the highest mean value. Just like HB0, the roll outs are depth-limited.

6.3 Evaluation Protocol

Games are played using the standard Schnapsen rule set, with randomized initial shuffles for each match unless otherwise specified. For each pairing, 1000 games are played with alternating starting positions to control for first-move advantage. To reduce variance, all pairings of 1000 games are played on the same seeds. This ensures that observed differences in performance are attributable to agent behavior rather than favorable or unfavorable random deals.

6.4 Metrics

The primary performance metric observed is the win rate of each agent against each opponent.

6.5 Hypothesis Testing

The experimental results are analyzed with respect to the hypotheses stated in Section 1.1 and tested against their corresponding null hypotheses as follows:

- **H1** H_0: Oracle Access will not show a statistically significant improvement in the win rate over the baseline agent.
- **H2** H_0: State Manipulation will not show statistically significant improvement in win rate over Oracle Access.

(*Note that for H3, since a perfect win rate means zero variance, we instead test for statistically significant improvement over HB2.*)

7 Results

Table 1 summarizes the win rate of each version of HonestBot against RandBot, BullyBot and RDeepBot. Each pairing (i.e. HB1 vs RandBot etc.) shows the win rate out of 1000 games each, making a total of 12,000 simulated games for testing.

Table 1. Win Rates (in %) over 1000 Games per Opponent

Agent	**vs RandBot**	**vs BullyBot**	**vs RDeep**	**Overall**
HB0	72.00	79.60	42.20	64.60
HB1	87.90	94.20	69.70	83.93
HB2	97.80	97.20	93.20	96.07
HonestBot	100.00	100.00	100.00	100.00

Across all opponents, we can see that performance increases consistently as the constraints are progressively relaxed. The histogram in Figure 2 shows a clear depiction of this phenomenon and gradual increase in performance.

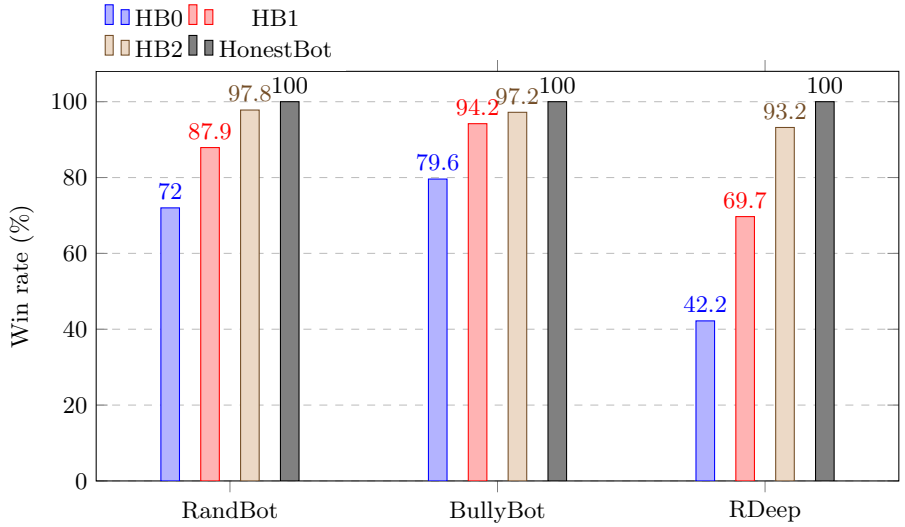

Fig. 2. Win rates by opponent (1000 games per pairing).

7.1 Statistical Significance

To validate the results, we conducted a one-sided Z-test to compare each version to its successor and set our significance level α at 0.05.

Table 2. Z-test results comparing each agent to its successor against all opponents.

Opponent	**Comparison**	**Win Rate**	**Z-Score**	***p*-value**
RandBot	HB0 → HB1	72.0% → 87.9%	8.88	< 0.001
	HB1 → HB2	87.9% → 97.8%	8.59	< 0.001
	HB2 → HonestBot	97.8% → 100%	4.72	< 0.001
BullyBot	HB0 → HB1	79.6% → 94.2%	9.68	< 0.001
	HB1 → HB2	94.2% → 97.2%	3.31	< 0.001
	HB2 → HonestBot	97.2% → 100%	5.33	< 0.001
RDeep	HB0 → HB1	42.2% → 69.7%	12.39	< 0.001
	HB1 → HB2	69.7% → 93.2%	13.52	< 0.001
	HB2 → HonestBot	93.2% → 100%	8.39	< 0.001

8 Analysis

HB0 The baseline agent, HB0, performs exceedingly well against both RandBot (72.0%) and BullyBot (79.6%); however, it struggles heavily against RDeep where it won only 42.2% of games. These results match with HB0's overall design: during phase 1, it runs alpha-beta depth limited search on a single assumed

game state. HB0 assumes a new game state after every trick, whereas RDeep assumes 16 different game states each trick (this was the value chosen to use for testing). Since HB0 relied far more on luck in getting a decent assumption of the game state than RDeep did, it performed far worse than all other pairings in the data.

HB1 Oracle access in HB1 produces a sizable improvement against all three opponents, raising the overall win rate from 64.6% to 83.93%. The increase is most significant against RDeep, rising from 42.2% to 69.7%. The code between HB0 and HB1 is near-identical with the only difference being that the observed game state in HB1 is the true game state rather than a random assumption. Oracle access perfects the agent's foresight and its ability to manage high-value captures and marriage declarations without walking into lines that only look optimal under an incorrect assumption. As a result, the feature that was the sole weakness HB0 had when facing RDeep has been thoroughly addressed and essentially perfected, evidenced by the increase in win rate.

HB2 Allowing active state manipulation in HB2 is a much bigger step towards achieving that coveted perfect win rate. It achieved 97.8% against RandBot, 97.2% against BullyBot, and a massive improvement to 93.2% against RDeep, for an overall win rate of 96.07%. This provides strong evidence that removing the "luck of the deal" produces a larger advantage than simply removing epistemic uncertainty. Oracle access improves decision quality given a fixed distribution of resources, but it cannot change the underlying distribution itself. In contrast, state manipulation lets the agent systematically construct high-leverage advantages, such as a hand with consistently concentrated trump and ace/ten presence. This advantage has the convenient perk of making it immensely difficult for the opponent to contest tempo or even accumulate points at all.

HonestBot HonestBot achieves perfect performance against RandBot, BullyBot and against RDeep (100% in all three cases), yielding the coveted perfect 100% win rate overall. Compared to HB2, HonestBot goes further in two ways: it has more freedom to manipulate the state, and it is much more stable in practice. We fixed minimax bookkeeping issues, made special-move handling more reliable (especially marriages seeing as trump exchanges become rare once the hand-rigging is in place), and added search improvements like a transposition table and a node budget. Overall, these changes reduce brittle edge cases and make failures less likely to come from technical issues rather than actual game-play decisions.

8.1 Hypothesis Evaluation

The results support all three hypotheses.

Hypothesis 1 is supported. Oracle access alone produces a substantial improvement in win rate over the baseline, with a Z-score of 12.39 and $p < 0.001$ against the strongest opponent (RDeep), demonstrating that imperfect information in phase 1 is a major driver of suboptimal play. However, oracle access does not produce a perfect win rate, especially against stronger opponents. This indicates that even with full information, some losses still occur under certain deals and continuations, consistent with the idea that unfavorable card distributions and forced commitment constraints can still lead to defeat against strong play.

Hypothesis 2 is supported. Active constraint relief via state manipulation significantly outperforms oracle access alone across all opponents. The key comparison is against RDeep: HB2 improves from 69.7% in HB1 to 93.2% (Z=13.52, $p < 0.001$). This supports the claim that optimizing the card distribution is a stronger predictor of victory than merely observing hidden information, because it directly removes the most damaging factor for a strong agent: starting from a resource configuration that cannot be reliably converted into a win against optimal opposition.

Hypothesis 3 is supported. HonestBot not only achieved a 100% win rate against RandBot, BullyBot and RDeep across 3000 games, but also showed a statistically significant improvement over HB2 (Z=8.39, $p < 0.001$). This strongly suggests that once both informational uncertainty and the randomness of the deal are removed through oracle access and state manipulation, Schnapsen becomes trivial for the agent. That being said, these are still empirical results rather than a formal proof that Schnapsen is "solved".

9 Conclusion

We have constructed an agent built around the capability to rig the game in various different ways in order to observe the changes in performance this would yield. Starting from an honest, non-cheating bot, we progressively removed constraints and optimized the play around these relaxations. The experiments performed indicate that with an agent having oracle access and the ability to actively manipulate the game state, a perfect win rate is achievable. Overall, having perfect information in an imperfect information game helps a lot, but being able to manipulate who has which cards trivializes Schnapsen and pushes the win rate up to 100%. Future work could focus on gradually restricting the level of manipulation the bot can utilize before the perfect win rate recedes, such as limiting the amount of swaps or restricting manipulation to the talon only. As well as, perhaps, observing interactions between two of these omnipotent Schnapsen bots playing against each other.

References

1. Bitan, M., Kraus, S.: Combining prediction of human decisions with ismcts in imperfect information games. In: Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS 2018) (2018). https://doi.org/10.65109/CIFD3480
2. Boney, R., Ilin, A., Kannala, J., Seppanen, J.: Learning to play imperfect-information games by imitating an oracle planner. IEEE Transactions on Games (2021). https://doi.org/10.1109/TG.2021.3067723
3. Brown, N., Bakhtin, A., Lerer, A., Gong, Q.: Combining deep reinforcement learning and search for imperfect-information games. In: Proceedings of the 34th International Conference on Neural Information Processing Systems. Curran Associates Inc. (2020), https://proceedings.neurips.cc/paper/2020/file/c61f571dbd2fb949d3fe5ae1608dd48b-Paper.pdf
4. Cowling, P.I., Powley, E.J., Whitehouse, D.: Information set monte carlo tree search. IEEE Transactions on Computational Intelligence and Ai in Games (2012). https://doi.org/10.1109/TCIAIG.2012.2200894
5. Milec, D., Kovařík, V., Lisý, V.: Adapting beyond the depth limit: Counter strategies in large imperfect information games. In: Proceedings of the 24th International Conference on Autonomous Agents and Multiagent Systems. International Foundation for Autonomous Agents and Multiagent Systems (2025), https://www.ifaamas.org/Proceedings/aamas2025/pdfs/p2675.pdf
6. Schmid, M.: Search in imperfect information games (2021), https://arxiv.org/abs/2111.05884
7. Schmid, M., Moravčík, M., Burch, N., Kadlec, R., Davidson, J., Waugh, K., Bard, N., Timbers, F., Lanctot, M., Holland, G.Z., Davoodi, E., Christianson, A., Bowling, M.: Student of games: A unified learning algorithm for both perfect and imperfect information games. Science Advances (2023), https://www.science.org/doi/abs/10.1126/sciadv.adg3256
8. Sun, M.: Research of artificial intelligence in imperfect information card games. Applied and Computational Engineering (2024). https://doi.org/10.54254/2755-2721/33/20230256
9. Wisser, F.: Creating possible worlds using sims tables for the imperfect information card game schnapsen. In: 2010 22nd IEEE International Conference on Tools with Artificial Intelligence. IEEE (2010). https://doi.org/10.1109/ICTAI.2010.76
10. Wisser, F.: Resources on the card game schnapsen (nd), https://www.dbai.tuwien.ac.at/user/wisser/schnapsen/rules/, accessed 23 January 2026
11. Zinkevich, M., Johanson, M., Bowling, M., Piccione, C.: Regret minimization in games with incomplete information. In: Advances in Neural Information Processing Systems. Curran Associates, Inc. (2007), https://proceedings.neurips.cc/paper_files/paper/2007/file/08d98638c6fcd194a4b1e6992063e944-Paper.pdf