# Possible Strategic Advancements in AGI Model NARS Through the Lens of Poker AI

Jun Zhou

No Institute Given

**Abstract.** In recent years, AI has achieved significant milestones in the gaming sector, often serving as a challenging benchmark and milestone in its development process. Notable achievements have been made in domains such as chess, Go, and video games, where players and AIs are presented with complete information to make relatively better decisions. However, poker, the quintessential game of imperfect information, poses greater challenges and offers better opportunities to refine model's decision-making ability. In this article, we review and analyze several AI models that have achieved professional player status to explore and theorize the computational and decision-making advancements possible for AGI, especially the NARS model, in the realm of poker. This study aims to further the capabilities of AGI, which aspires to try to solve all representable problems.

## 1 Introduction

Over the years, many games have been successfully utilized in the field of artificial intelligence research. Studies in chess, checkers, and Go have produced powerful deep learning models and achieved significant success. NARS, an AGI model, has also demonstrated excellent performance and potential in the Pong game.

The games mentioned above share beneficial attributes such as well-defined rules for possible actions, and clear goals and objectives. Poker also possesses these traits; however, its characteristic of imperfect information—where one cannot observe opponents' cards and randomness in the distribution of private and public cards—impacts the rules defining possible actions. This uncertainty, which can lead to changes in the rules of possible actions, is clearly beneficial for AGI research.

Past AI research on poker can broadly be classified into two main categories. The first category addresses the issue as a whole, creating autonomous systems that can play the game and make decisions. The second category aims to analyze subsets of isolated problems. Due to the exceptional results of the first category, mature models in this category typically possess algorithms capable of calculating comprehensive strategy blueprints, enriching the tactical details of sub-games achieved during play, and tailoring the overarching strategy blueprint in response to opponents. This article focuses on analyzing the feasibility and effectiveness of these algorithms when applied to AGI.

In this article, we first review and analyze several high-level deep learning models, identifying their common features and unique innovations. We then attempt to apply these insights to the AGI model NARS, exploring potential modifications and prospects for NARS's eighth layer of decision-making while adhering to NARS's principles of limited knowledge and resources. Finally, we summarize the results obtained.

The remainder of this section will briefly introduce the rules of the poker variant: Texas Hold'em.

### 1.1 Texas Hold'em

**Texas Hold'em** consists four stages: *pre-flop*, *flop*, *turn*, and *river*. Before the game begins, two players must make forced bets, known as the small and big blinds. The big blind is typically twice the amount of the small blind. The player to the left of the blinds initiates the action, which includes folding, calling, or raising.

The different actions in poker include:

- **Fold**: The player no longer contributes to the pot and forfeits the right to win the current pot.
- **Check/Call**: If no additional chips are needed to continue, the player can check; if more chips are required to stay in the game, the player must call.
- **Bet/Raise**: A player may choose to bet if no bets have been made previously; if there has been a bet, the player can also choose to raise, increasing the amount of chips in the pot.

In each stage, players combine their two private cards with the community cards to form the best possible hand. Each stage ends with a betting round. The game continues until the river round is completed, at which point any remaining players reveal their cards. The player with the best hand wins all the chips. If more than one player shares the highest hand, the chips are split equally among them.

## 2 Related Works

In this section, we will focus on the study of three deep learning models: Pluribus, Libratus, and DeepStack. These models engage in games of incomplete information, which require more complex reasoning than similarly sized games with perfect information. Each of these models, as well as most artificial intelligence systems designed to tackle poker, employs Counterfactual Regret Minimization (CFR) to reason through the game. This approach involves generating a complete strategic blueprint before the start of each game, as mentioned in the discussion above.

### 2.1 Introdution of CFR

Counterfactual Regret Minimization (CFR) is an iterative algorithm used to solve games with large information sets and complex strategy spaces under conditions of incomplete information. CFR dynamically adjusts players' strategies to minimize "regret." Here, "regret" refers to what players could have achieved had they chosen a different action at a certain decision point in the past, in terms of better average payoff.

**How CFR Works:**

1. **Initialization:** At the start of the game, all strategies are random, and all cumulative regrets are set to zero.
2. **Game Iteration:** For each iteration, the CFR algorithm simulates a game. At each decision point, an action is chosen based on the current strategy.
3. **Regret Update:** At the end of the iteration, the algorithm revisits each decision point and updates the regret values. Regret is quantified as the difference in average payoff that could have been achieved, had a different action been taken relative to the action actually taken.
4. **Strategy Update:** The strategy is adjusted based on the regret values. If an action has a high regret value, it is more likely to be chosen in future iterations.
5. **Repeat Process:** This process is repeated multiple times until the strategy converges.

### 2.2 Simplified calculations and strategy optimization

The computation required to generate a complete strategy blueprint before each game is immense. Therefore, simplification of decision points, calculations, and the overall blueprint is essential. The blueprints of all three models translate the original game into an abstract game to some extent. For example, in the original game, a pair of aces and a pair of kings have certain differences, but these differences do not significantly impact decision-making, meaning that within certain limits, differences in hands and stakes do not affect decision choices. This allows the original poker scenarios, which could amount to $10^{160}$ situations, to be compressed into approximately $10^{14}$ abstract scenarios. Even so, this number is still vast.

To address this, Libratus and DeepStack simplify calculations and enhance efficiency by breaking the complex poker game into smaller subgames. This method enables them to focus on the most critical decision points at the moment, reducing the number of strategy combinations they need to process and thus lowering the demand for computational resources. By dynamically adjusting their strategies to adapt to their opponents' behavior and the progression of the game, these systems can make precise decisions in real-time. The subgame structure also allows these models to learn from each independent decision, giving them the ability to adjust their strategies to cope with high-level opponents.

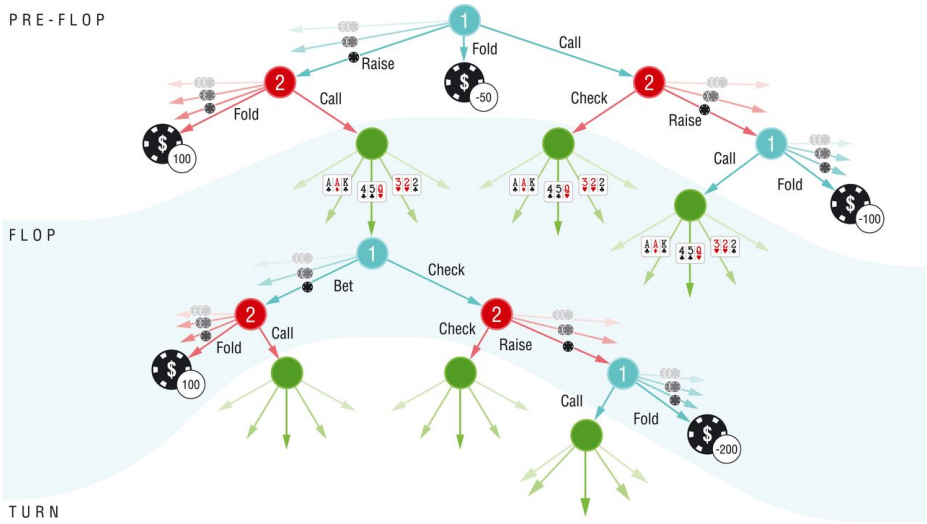


Fig. 1: Example of a decision tree from the model "DeepStack". Nodes represent public states, whereas edges represent actions.

## 3 Method

Clearly, we cannot achieve the same level of precision and completeness in NARS as with the aforementioned deep learning models. However, we can still integrate NARS's eighth layer with poker rules. The goal can be succinctly described as obtaining as much revenue as possible, even if this revenue might be negative. The possible operations include fold, check, call, bet, and raise. The number of operations should remain constant. Most statements in this case are events because the sequence of time, which can also be referred to as the order of events, is especially important in poker. For example, "Player A has a pair of aces" is only relevant in the current round of the game and does not affect the next or previous rounds. Similarly, "Player A makes a $5 raise" has different implications depending on whether it occurs before or after Player B's check. All external updates related to the game should carry timestamps and desire values for this system.

A statement should be an interpretation based on the knowledge NARS has learned so far, combined with currently "valid" events, such as "Based on Player A's hand, they can win," accompanied by a truth value.

Ideally, in each round of the game, whenever a new event occurs, it should conform to the relationship of

$$(\textit{condition, operation}) \mapsto \textit{consequence}$$

That is, whenever a new event occurs, the system makes the most appropriate operation based on current knowledge and different expected values of events.

The resulting consequence updates the value of currently valid events and generates new statements.

### 3.1 Implementation of CFR

In order to introduce the concept of an event's desire value and Counterfactual Regret Minimization (CFR) in NARS, I will present a specific example. Suppose at the start of this round of the game, player A has a pair of aces. Thus, the event $S$ is "player A has a pair of aces", and the desired state $D$ in this case would be how to maximize expected benefits from this advantageous hand. The desired state would similarly apply if there were a very poor hand. In such $S \rightarrow D$ scenarios, the desire value quantifies the degree of contribution to achieving the maximum return.
Since CFR focuses on decision-making to minimize regret for not taking alternative strategies, for "player A has a pair of aces": CFR would involve calculating the regret associated with all possible actions (such as folding, calling, raising) given the hand of a pair of aces, and then choosing the action that leads to the least regret based on past outcomes. It also evaluates the impact of each potential action on the desired state. The desire value indicates which actions are more aligned with achieving the desired outcome, thus guiding the CFR process to prioritize strategies that not only minimize regret but also maximize alignment with the desired value outcomes. The desire value can influence the CFR strategy by adjusting the importance or weight of different actions based on their anticipated contribution to the desired state. For example, if raising significantly increases the likelihood of maximizing returns (high desire value), CFR would adapt to prioritize raising, unless historical regret data suggest a better strategic choice.
Of course, how to implement CFR is an issue I have not been able to solve due to my limitations, which is why there are no practical experiments to prove these theories.

### 3.2 Monte Carlo simulation

Regarding how NARS can deduce that "Player A has a pair of aces" is advantageous for the player, it can be evaluated by simulating thousands of game hands using Monte Carlo simulations to assess the performance of this hand under various game conditions. During the simulation process, the potential hands of opponents and the progress of the game are randomly varied to evaluate the win rate of a pair of aces in different scenarios. The obtained win rate can be quantified as the strength of the hand pair of aces in actual games. Combined with the Counterfactual Regret Minimization (CFR) strategy optimization method, the results of the Monte Carlo simulations can be used to fine-tune the decision tree in NARS, ensuring that the system can make the optimal decisions when facing different opponents and game situations. Through Monte Carlo simulations, NARS can not only quantitatively analyze the potential value of a pair of aces but also adjust its decision-making framework based on the dynamic changes in

actual games, making its decisions more scientific. The information provided to the system's beliefs through simulation is relatively accurate and effective, and they can also serve as reliable experiences within the system.

However, since the accuracy of Monte Carlo simulations changes with the number of simulations, and given that NARS is based on a system with limited knowledge and resources, I do not believe it is the best possible application for NARS.

## 4 Conclusion

The best choices in incomplete information games can be applied not only in poker, but also have implications in real-world scenarios involving asymmetric information. For example, medical diagnostic systems and poker systems both involve statistical issues and the need to make relatively stable decisions. If NARS can demonstrate certain prospects and performance in poker games, it can also show potential and a certain degree of reliability in making stable judgments in medical diagnostic systems.

## References

1. Brown, N., Lerer, A., Gross, S., & Sandholm, T. (2019). Deep counterfactual regret minimization. In *International Conference on Machine Learning* (pp. 793–802).
2. Rubin, J., & Watson, I. (2011). Computer poker: A review. *Artificial Intelligence*, 175(5-6), 958-987. https://doi.org/10.1016/j.artint.2010.12.005
3. Brown, N., & Sandholm, T. (2019). Superhuman AI for multiplayer poker. *Science*, 365(6456), 885-890. https://www.science.org/doi/abs/10.1126/science.aay2400
4. Brown, N., & Sandholm, T. (2018). Superhuman AI for heads-up no-limit poker: Libratus beats top professionals. *Science*, 359(6374), 418-424. https://www.science.org/doi/abs/10.1126/science.aao1733
5. Moravčík, M., Schmid, M., Burch, N., Lisý, V., Morrill, D., Bard, N., ... & Bowling, M. (2017). DeepStack: Expert-level artificial intelligence in heads-up no-limit poker. *Science*, 356(6337), 508-513. https://www.science.org/doi/abs/10.1126/science.aam6960
6. Jagielska, I., Matthews, C., & Whitfort, T. (1999). An investigation into the application of neural networks, fuzzy logic, genetic algorithms, and rough sets to automated knowledge acquisition for classification problems. *Neurocomputing*, 24(1-3), 37-54.