Improving Geister AI Through Strategy Mixing and Anti-Loop Mechanisms

Po-Yen Lai[†1], Shi-Jim Yen[†1], Jr-Chang Chen[†2], Jen-Yeu Chen[†3]

Abstract: *Geister is an imperfect-information board game in which players cannot directly know the true identities of the opponent's pieces. This uncertainty makes decision-making depend not only on board evaluation but also on risk control and hidden-information inference. In this work, we improve a PurpleMax-based Geister agent by introducing two mechanisms: a probabilistic sacrifice strategy and an anti-loop mechanism. The probabilistic sacrifice strategy combines conservative search behavior with occasional red-piece sacrifice to increase strategic diversity and reduce stagnation. The anti-loop mechanism detects short repetitive move cycles and replaces them with alternative actions chosen by rule-based evaluation. Based on these improvements, we implement the agent S50A2 and compare it with the baseline Naotti2020. Experimental results show that S50A2 substantially reduces draw-heavy behavior and achieves a markedly higher overall win rate. The proposed method demonstrates that effective improvements can be achieved without increasing search complexity or computational cost.*

Keywords: Geister, imperfect information game, PurpleMax, Minimax search, strategy mixing, anti-loop mechanism.

1. Introduction

Geister is a typical imperfect-information board game. Because each player cannot directly observe the true color of the opponent's pieces, the game requires both strategic board evaluation and inference under uncertainty. This makes Geister a useful benchmark for studying decision-making in hidden-information environments.

Among existing approaches, search-based methods remain attractive because they are efficient and interpretable. However, a major weakness of conservative search methods such as PurpleMax [1] is that they often become overly cautious when dealing with hidden information. In practice, this can lead to repetitive movement patterns, low initiative, and a high draw rate. When both players adopt similar conservative behaviors, the game may stall for a long time without producing a decisive result.

To address this issue, we propose two improvements to a PurpleMax-based baseline. First, we introduce a probabilistic sacrifice strategy that mixes standard conservative search with the deliberate sacrifice of Red pieces in some situations. This increases action diversity and helps the agent break through stagnant positions. Second, we introduce an anti-loop mechanism that can detect short, repetitive move sequences and select alternative moves when the loop may continue.

Fig. 1 shows the overall decision flow of the proposed agent. The goal of this study is to improve the practical playing strength of Geister AI while keeping the search framework simple and computationally efficient.

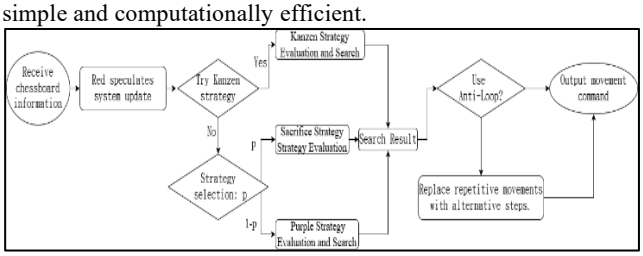


Fig. 1. Decision flow of the proposed Geister agent with strategy mixing and anti-loop handling.

2. Geister

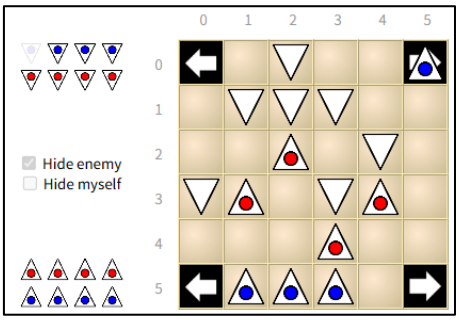


Fig. 2. Example of a ChessEscape Ending

Geister is an imperfect-information board game where each player has 8 pieces on a 6x6 board: 4 blue and 4 red. The core characteristic of the game is information asymmetry: while

†1 Department of Computer Science and Information Engineering, National Dong Hwa University, Hualien, Taiwan
†2 Department of Computer Science and Information Engineering, National Taipei University, Taipei, Taiwan
†3 Department of Electrical Engineering, National Dong Hwa University, Hualien, Taiwan

© 2026 Information Processing Society of Japan

players can observe the positions of all pieces on the board, they cannot determine the true color of their opponent's pieces until a capture event occurs. Each turn, a player can move one of their pieces one space to the adjacent square (up, down, left, or right).

The game has multiple victory conditions, including the following three:

1. ChessEscape: Any of your Blue pieces successfully moves to one of the two corner escape points on the opposite side of the board and is removed from the board in the next turn.

2. NoRedGhost: All of your Red pieces are captured by the opponent.

3. NoBlueGhost: All of the opponent's Blue pieces are captured.

As shown in Fig. 2, the game interface displays the remaining pieces for both sides. The opponent's pieces are shown at the top, while your pieces are shown at the bottom. Captured pieces are marked in gray. From a player's perspective, the colors of their own pieces are fully observable, whereas the opponent's pieces have observable positions but hidden colors.

3. Related research

3.1 Core search algorithm: PurpleMax

The AI architecture in this study is an extension and improvement of the PurpleMax algorithm [1]. PurpleMax is an efficient search mechanism designed for imperfect-information games such as Geister, aiming to address the state-space explosion problem caused by the unknown color of the opponent's pieces. Traditional methods require enumerating all combinations of unknown colors of the opponent's pieces, resulting in computational costs that grow with the number of possible configurations, making deep searching difficult. PurpleMax simplifies the problem through a clever abstraction: it treats all unknown colors of the opponent's pieces as "Purple Ghosts". During the Minimax search, the evaluation function adopts a pessimistic assumption:

- When we consider capturing a Purple Ghost, we assume it must be a Red piece. This means that capturing an unknown piece is evaluated as the worst-case scenario—we capture the opponent's Red pieces, bringing the opponent closer to the winning condition of "losing all Red pieces".
- When the opponent's Purple Ghost is considering escaping, we assume it must be a Blue piece. This means that any unknown piece near the exit is considered a direct threat to us.

By employing this worst-case assumption, PurpleMax avoids enumerating all possible color assignments while still making conservative decisions, ensuring that the AI doesn't easily make actions that lead to immediate defeat. This method significantly reduces search complexity, allowing it to reach deeper search layers within a limited time. However, it is precisely this extreme risk-averse characteristic that leads to a conservative playing style, making it prone to stalemates in balanced situations. This is the starting point for improvement in this research.

3.2 Initial configuration study

The initial configuration has a profound impact on the overall strategy of the game. Due to information asymmetry, the initial piece layout can influence the opponent's interpretation of the hidden information and subsequent decisions. In their research, Kouki Suetsugu and Yusuke Oda [2] proposed a "probabilistic non-random configuration", in which the highest probability configuration is three Red pieces connected in the front row and one Red piece in the back row placed below the Red pieces in the upper row of Blue pieces, in order to induce the opponent to make a wrong capture decision when there is insufficient information, thereby protecting the Blue pieces in the back. This study is greatly inspired by their opening layout and adopts their "three Red pieces in the front row and one Red piece in the back row" configuration idea, aiming to maximize the tactical decoy effect at the beginning of the game.

3.3 Hybrid Strategy and Nash Equilibrium

Hibiki Harada and Noriaki Yoshiura [3] used the R-NaD algorithm in their research and eventually learned a hybrid strategy that approximates Nash equilibrium. In this equilibrium strategy, the AI dynamically weighs two actions: one is to "let the real Blue piece escape," and the other is to "let the Red piece pretend to be the Blue piece and make a feint to escape."

The AI can learn to find an optimal hybrid ratio between these two seemingly contradictory actions, making it difficult for the opponent to discern the agent's true intentions. This confirms that in high-level games of Geister, a single pure strategy (such as always protecting the Blue piece or always sacrificing the Red piece) is not the optimal solution. A successful AI needs to have the ability to execute hybrid strategies. This study draws on this idea and implements a controllable hybrid strategy using predefined probabilistic rules to explore the tactical benefits of sacrificing the Red piece under the PurpleMax architecture.

4. Proposed Method

4.1 Initial configuration

Based on the experimental results of the initial configurations reported in [2] and [3], this study adopts an initial board configuration consisting of three Red pieces in the front row and one Red piece in the back row.

The three Red pieces in the front row can occupy four possible positional configurations, while the Red piece in the back row can be placed at either of two edge positions. Therefore, there are eight possible initial configurations, as shown in Fig. 3.

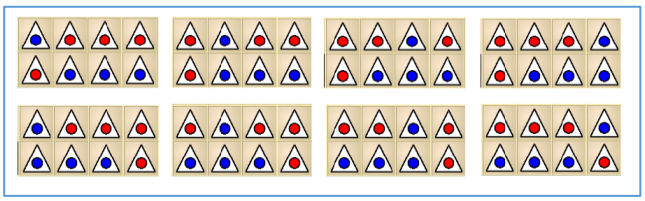


Fig. 3. Initial configuration of the experiment

4.2 Naotti2020 Analysis

Naotti2020 is an implementation of the method proposed in [1]. Naotti2020 is currently open source, making its implementation relatively accessible and reproducible. Therefore, Naotti2020 is used as the baseline for comparison in all experiments in this study.

Different experimental versions of the Agent were also based

© 2026 Information Processing Society of Japan

on the source code of Naotti2020, which was reproduced in the sample code of the competition platform and modified.

Agent	Opponent	WinRate	Win	Draw	Lose
Naotti2020	Naotti2020	19.60%	98	306	96

Table 1 Baseline self-battle results

By observing the matches between Naotti2020 and Naotti2020 shown in Table 1, it can be found that 306 of the 500 games ended in draws, corresponding to a draw rate of 61.2%, which is the main improvement target of this study.

Naotti2020 primarily uses two different evaluation functions, Purple and Kanzen, for the Minimax search. The Kanzen strategy performs a more complete search when clear information about the opponent's Red pieces is available. This study does not modify the Kanzen strategy, so the following discussion will focus on the operation of the Purple strategy and the changes in its process.

4.3 Purple strategy

When clear information about the opponent's pieces is unavailable during the game, the purple strategy is a relatively conservative evaluation method. Its main evaluation function is as follows:

$$\boldsymbol{purple} = \boldsymbol{self} - \boldsymbol{opp} \quad (1)$$

$$\boldsymbol{self} = \boldsymbol{W1} \times \boldsymbol{B} - \boldsymbol{W2} \times \boldsymbol{Dself} \quad (2)$$

$$\boldsymbol{opp} = \boldsymbol{W2} \times \boldsymbol{Dopp} \quad (3)$$

Formula (1) is the evaluation function "purple" for the Purple strategy. Formula (2) details the "self" scoring algorithm, where "B" represents the number of surviving Blue pieces, "W1" and "W2" are weights (W1 = 1000, W2 = 1), and "Dself" is the score for the distance from the player's escape points. "opp" is the opponent's evaluation, with "W2" also being a weight of 1, and "Dopp" referring to the score for the distance the opponent travels to the opponent's escape points.

4.4 Sacrifice strategy

Naotti2020 attempts a Kanzen search followed by a Purple search on each move, depending on the game state. This search method is highly efficient and avoids immediate defeat due to rule-based design, but it also makes it easier for opponents to identify movement patterns.

The idea behind the hybrid strategy is as follows. The Purple strategy often causes Red pieces to behave in a manner similar to "mimicking the Blue piece's escape," while the Sacrifice strategy explicitly aims for "Red pieces actively approaching the opponent in the early game, creating attrition and control." And "using the risk of the opponent capturing a Red piece in the late game, allowing the Blue piece to escape more aggressively." Both can lead to Red piece attrition, but their objectives differ: the former is more goal-oriented, while the latter is more tactical and spatially pressure-oriented, aiming to make it more difficult for the opponent to infer the colors of its pieces.

Early Game (more than one Red piece remains):

In "Purple", Red pieces tend to be conservative due to a lack of positive incentives in the evaluation. This study introduces a reward for "Red pieces approaching the opponent" in the Sacrifice mode, increasing the frequency of interactions during the early and mid-game. The goal of this evaluation is to encourage Red pieces to actively approach the opponent's group to create harassment, while simultaneously restricting ineffective captures.

$$\boldsymbol{Sacrifice}(\boldsymbol{early}) = \boldsymbol{Eself} - \boldsymbol{opp} \quad (4)$$

$$\boldsymbol{Eself} = \boldsymbol{self} + \boldsymbol{red} - \boldsymbol{cap} \quad (5)$$

$$\boldsymbol{red} = (\mathbf{110} - \mathbf{10} \times \boldsymbol{D}) + \mathbf{100} \times \boldsymbol{A} \quad (6)$$

$$\boldsymbol{cap} = \mathbf{1000} \times \boldsymbol{max}(\mathbf{0}, \boldsymbol{C}) \quad (7)$$

"Sacrifice" follows the original search method, but only retains the "self" evaluation score of formula (2) in the scoring formula (5) "Eself" for our side, and adds two new scores, "red" and "cap". "opp" also maintains the approach of formula (3).

Formula (6) "red" represents the extra points awarded to the Red piece. "D" is the Manhattan distance from the Red piece to the nearest enemy piece (if the nearest enemy piece is at distance 1, the score is 110-10=100; at distance 2, the score is 110-20=90). A is the number of enemy pieces that are orthogonally adjacent to the Red piece in four directions (0-4).

Formula (7) "cap" is the penalty for the capturing behavior. It deducts 1000 points for each enemy piece captured, where C is the number of enemy pieces captured along the search path from the root node to the current node, thus preventing the Red piece from choosing to capture and move closer when this strategy is used.

In terms of scoring, the "red" score cap is 500 points, and the penalty for actively capturing is 1000 points, corresponding to the Blue piece retention weight W1 = 1000 in the Purple strategy. In the scoring design, a Red piece's move that approaches and becomes adjacent to an enemy piece rewards it with a score "red" roughly equivalent to half the retention value of a Blue piece (W1 / 2); conversely, the penalty "cap" for capturing is equivalent to the value of a Blue piece. This design aims to encourage Red pieces to actively approach the enemy early on to create tactical opportunities, rather than focusing primarily on capturing.

Late Game (One side has one Red piece remaining):

In endgames with fewer pieces, the strategy should focus explicitly on victory, rather than emphasizing "interaction." This phase employs hard positional weights to force convergence in endgame behavior, ensuring that every move advances towards a clear win-or-lose objective.

$$\boldsymbol{Sacrifice}(\boldsymbol{late}) = \boldsymbol{Lself} - \boldsymbol{opp} \quad (8)$$

$$\boldsymbol{Lself} = \boldsymbol{self} + \boldsymbol{Wp} \quad (9)$$

(9) Wp (Position Weight):

Escape point: The point where the player's Blue piece can escape and win (top left, top right).

Defense point: The point where the opponent's Blue piece can

© 2026 Information Processing Society of Japan

escape and win (bottom left, bottom right).

- Blue piece at the escape points: +8000 (Enhances escape victory)
- Red piece at the escape points: -8000 (Prevents Red from mistakenly taking the escape route)
- Red piece at the defense point: +4000 (Encourages defense)
- Blue piece at the defense point: +2000 (Preserves Blue's offensive/defensive flexibility)

In formula (8), "opp" is evaluated using the same method as in formula (3), and "self" in formula (9) is evaluated using the same method as in formula (2).

The additional score designed in this stage no longer adds the penalty for early approach and capture, but is purely based on position, so that the algorithm can quickly converge to the most favorable endgame configuration. 8000 is twice the typical value of "4 Blue pieces remaining" (4×1000 + distance×1), so that the Blue pieces that escape the exit are considered to be on the verge of victory in the search, and the behavior of Red pieces moving towards the exit point in the endgame is penalized with negative points; the 4000 points for Red pieces at the defense point (equivalent to the value of 4 Blue pieces) is used to encourage defense without being overly forced, and gives Blue pieces a certain incentive score for defending at the defense point, but still focuses on escaping. The score for Blue pieces that stay in defense is only 1/4 of that for going to the exit point.

4.5 Anti-loop mechanism

Under the evaluations of Purple and Sacrifice, the AI may still get caught in short-cycle back-and-forth movements in the mid-to-late game when the game state does not guarantee a win, because repeating moves allows it to avoid losing decisions (e.g., moving a piece from position A to position B in one move, then immediately moving it back from B to A in the next move). This behavior usually stems from the fact that the evaluation function may assign similar or favorable scores to repeated positions, while the finite search depth prevents the algorithm from recognizing the long-term consequences of such repetition. To suppress this loop, this study incorporates a rule-based substitution step mechanism to reduce the number of draws. However, when in a disadvantageous position, repeated moves may be the optimal solution at that moment. This study designs two versions of the loop substitution:

- AntiLoop1: As soon as a move that immediately reverses the "last move" is detected, a substitution step is forcibly used.
- AntiLoop2: The position is evaluated first, and a substitution step is only used when the position is "advantageous," avoiding disruption of the survival strategy when at a disadvantage.

Anti-Loop1(A1)

A lightweight intervention mechanism is employed. After the AI has acquired information on all the legal moves on the board, if specific conditions are detected or a situational adjustment is needed, this evaluation module will be activated to perform a four-component scoring process to ultimately determine the best move. When generating the set of legal moves, any immediate reverse move is assigned a large negative score (-10,000) rather than being removed from the set of legal moves. Therefore, if the immediate reverse move is the only available option, it can still be selected.

$$\boldsymbol{Score = loop + cap + def + dir} \quad (10)$$

In formula (10):

Loop: Reverse Step Suppression:

To prevent the AI from getting stuck in an ineffective, blind loop, the system tracks the "last move taken". If the current candidate move is immediately reversed from the previous move (e.g., left ➞ right then right ➞ left, up ➞ down then down ➞ up), a severe penalty (-10000 points) will be imposed.

cap: Legality and Capture Restrictions:

To prevent the AI from being defeated immediately due to mistakenly capturing the opponent's Red pieces, while also considering the possibility of a comeback in the endgame, strict filtering is applied to moves that "capture the opponent's pieces". After obtaining all legal moves, capture moves are filtered out before the scoring stage, except for captures involving pieces at the four corners. Because the goal is to break the cycle and maintain the conservative style of the original version.

def: Defensive point entry/exit rating:

The strategic objective of the defensive point is for " one Red piece and one Blue piece" to jointly block the escape points, forcing the opponent to decide whether to capture it. The score is based on 100 points and is dynamically evaluated based on whether the pieces "enter" or "leave" the defensive point and the current status of the defensive point. In order to strengthen the division of labor between the Red pieces' defense and the Blue pieces' escape, the Red pieces' rewards and penalties are given double the weight.

A. Entering (Moving into) a Defense Point: The score is based on the current state of your pieces at the defense point (target square):

Currently 0 pieces (no pieces at any defense point):

- Red enters: +200 (Base 100 × 2)
- Blue enters: +100

1 Red piece is already stationed:

- Red enters: -200 (Avoids overlapping waste)
- Blue enters: +200 (Achieves a perfect defense with one Red and one Blue piece; the Blue piece receives twice the base reward)

1 Blue piece is already stationed:

- Red enters: +200 (Achieves a perfect defense with one Red and one Blue piece)
- Blue enters: -100 (Avoids overlapping waste)

B. Leaving (Moving out of) a Defense Point: To prevent your

© 2026 Information Processing Society of Japan

blocking pieces from easily leaving and allowing the opponent to escape during endgame defense, a penalty is imposed for leaving:

- Red leaves a defense point: -200
- Blue leaves a defense point: -100

dir: Directional preference score:

Directional movement aims to guide the pieces' overall strategic thinking. Its scoring weight is approximately 10% of the defensive point score, ensuring the AI's strategy of "defense first, direction second."

Blue piece (responsible for escaping, with the highest priority being to move closer to the opponent's bottom line to escape the axis):

- Priority direction: Movement that reduces the Y coordinate (closer to the opponent's bottom line, Y = 0 escape axis): +20
- Secondary direction: Movement that reduces min(x, 5-x) (closer to the left or right sidelines, X = 0 or X = 5): +10

Red piece (responsible for blocking, with the priority being to control the sidelines and pressure the opponent):

- Priority direction: Sideline movement: +10
- Secondary direction: Bottom line movement: +5

The Blue piece's score is twice that of the Red piece's baseline score, intended to encourage the Blue piece to have more escape opportunities.

After comprehensively calculating the scores across the four levels, the AI will first scan the scores of all legal moves and establish a "set of candidate moves with the highest score." If there is only one move in this set, it will be used directly. If multiple candidate moves achieve the same highest score, the system will randomly select one to execute.

Target Effects:

1. The impact of defensive point occupation (±200) is far greater than that of directional preference (+5 to +20).
2. Draw-breaking randomization prevents the AI from following a deterministic move pattern, making its behavior less predictable.
3. Effectively suppresses immediate reversal loops. An immediate reverse move receives a -10,000-point penalty and is therefore unlikely to be selected unless it is the only legal move available.

Anti-Loop2(A2)

Anti-Loop2 operates in the same way as Anti-Loop1, the difference being that it only substitutes when we have the advantage, and the advantage is determined as follows:

$$\boldsymbol{Advantage = rb + def + escape} \qquad (11)$$

In formula (11):

rb: (My_B – My_R) – (Opp_b – Opp_r):

Ideally, we have 4B1R and the opponent has 1b4r (Uppercase letters represent the agent side, and lowercase letters represent the opponent side), resulting in +6; conversely, it results in -6. This directly reflects the capture risk for both sides: the fewer Red pieces we have, the greater the capture risk for the opponent, and the higher our advantage, and vice versa.

def: Defensive scoring:

Defense scoring is simply based on whether there is a piece on our defensive point. One piece is worth +5 points, and the best situation is to occupy both points, which is worth +10 points. This item can force the opponent to decide whether to capture the corner piece and escape when we have only one Red piece left, putting the opponent at the risk of either escaping to win or accidentally capturing a Red piece and losing.

escape: Escape speed score:

This score is calculated based on the shortest escape distance for both sides. +5 is awarded at the escape points, and -1 is awarded for each additional distance, with a minimum of 0. The final score is calculated as the player's score minus the opponent's score.

The threshold for advantage score (Advantage) can be customized. In this experiment, the threshold is set to Advantage ≥ 0. Since the overall advantage score is the sum of these three components, a value greater than or equal to zero indicates that the AI is considered to have an advantage; in this case, the anti-loop substitution is applied to reduce the occurrence of cycles.

If it is less than 0, it means that the AI is behind in terms of the number of pieces or its escape progress is slower than the opponent's, or when no piece is available to occupy the defensive points. In this case, the original cycle movement may be the optimal solution, so maintaining the cycle movement is encouraged.

4.6 Changes to the Decision-Making Process

Naotti2020:

1. Read the board state and update "red" information.
2. Attempt a Kanzen search (if the redness reaches the threshold).
3. If Kanzen fails or no reliable information is available, perform a Purple search.
4. Return the search results.

"red" is the mechanism Naotti2020 uses to determine the current red chess information, and it is also the threshold for determining whether to enable the Kanzen strategy. This experiment will not modify this part.

In this process, if information is insufficient, the system relies entirely on Purple's evaluation, resulting in a single strategy and conservative behavior.

Proposed Agent (Fig. 1):

1. Read the board state and update "red" information.
2. Attempt a Kanzen search (if the redness reaches the threshold); if successful, use the search result directly.
3. If the Kanzen search does not produce a valid result, select between Sacrifice and Purple according to a predefined probability p, where p ∈ {0, 25, 50, 75, 100} %.
4. After generating candidate moves, check if the "immediate reversal" rule is triggered; if triggered, activate the anti-loop rule to replace the move.

© 2026 Information Processing Society of Japan

5. Return the final move.

Strategy mixing prevents the agent from relying exclusively on Purple when highly reliable information is unavailable. Sacrifice is randomly mixed in, and cyclical substitution steps are used to reduce the occurrence of draws, increase decision diversity, and change the density of mid-game interactions.

5. Experiments and results

5.1 Experimental Background

The experimental platform used a Geister Version 1.0.5 (Windows) local gaming environment. Game rules were set according to the competition: each player had a total thinking time limit of 600 seconds, a maximum of 200 moves for a tie, and alternating turns. Each group played 500 games to reduce the impact of small sample fluctuations on the results and improve the stability of win rate and tie rate estimates.

Win rate was calculated as the number of wins for the agent divided by the total number of games played.

Local gameplay experiments were conducted using GamePlatform (website: http://120.126.151.216/index.php), the official gaming platform provided by the TCGA competition. Results were statistically analyzed and tables were generated.

5.2 Evaluation of the Sacrifice Strategy

Agent	Opponent	WinRate	Win	Draw	Lose
S0	Naotti2020	19.60%	98	306	96
S25	Naotti2020	37.20%	186	195	119
S50	Naotti2020	35.00%	175	200	125
S75	Naotti2020	30.00%	150	227	123
S100	Naotti2020	25.20%	126	254	120

Table 2 Pure sacrifice strategy

The Agent column in Table 2 refers to the Strategy selection in Fig. 1: p. The number p, from 0 to 100, represents the move ratio of the Sacrifice strategy. For example, S0 is the original Naotti2020. The results show that S25, i.e., a mixture of 25% Sacrifice strategy and 75% Purple strategy, produces the best performance among the pure sacrifice settings.

5.3 Evaluation of the Anti-Loop Mechanism

Agent	Opponent	WinRate	Win	Draw	Lose
Naotti2020	Naotti2020	19.60%	98	306	96
A1	Naotti2020	58.60%	311	13	207
A2	Naotti2020	55.00%	275	16	209

Table 3 Pure Anti-Loop

Table 3 shows that using anti-loop mechanism alone substantially contributes to eliminating draws, reducing 306 draws to 13 and 16, respectively. The draw rate decreased from 61.2% (306/500) to 2.4% (13/531) for A1 and 3.2% (16/500) for A2, and increased the overall win rate, increasing it from 19.6% to 58.6% (A1) and 55% (A2). In terms of win rate performance across different modes, based on the original search, global substitution (A1) performs slightly better than advantageous substitution (A2). This is because the pure anti-loop mechanism essentially only eliminates cycles and does not change the core strategy; therefore, under the same search strategy, the difference between global substitution and advantageous substitution is relatively small.

5.4 Combination analysis

Agent	Opponent	WinRate	Win	Draw	Lose
Naotti2020	Naotti2020	19.60%	98	306	96
S0A1	Naotti2020	58.60%	311	13	207
S0A2	Naotti2020	55.00%	275	16	209
S25A1	Naotti2020	60.60%	303	16	181
S25A2	Naotti2020	63.00%	315	19	166
S50A1	Naotti2020	63.20%	316	12	172
S50A2	Naotti2020	66.20%	331	19	150
S75A1	Naotti2020	60.20%	301	20	179
S75A2	Naotti2020	61.20%	306	20	174
S100A1	Naotti2020	60.00%	260	17	156
S100A2	Naotti2020	55.60%	278	34	188

Table 4 Combination

The S0A1 (A1) and S100A1 experiments did not contain exactly 500 games (531 and 433 games, respectively) due to an experimental recording issue. Therefore, their win rates were calculated based on the actual number of games played rather than 500 games. Since the win rate is defined as the number of wins divided by the actual total number of games played, the win rate was calculated using the same definition as the other experimental settings.

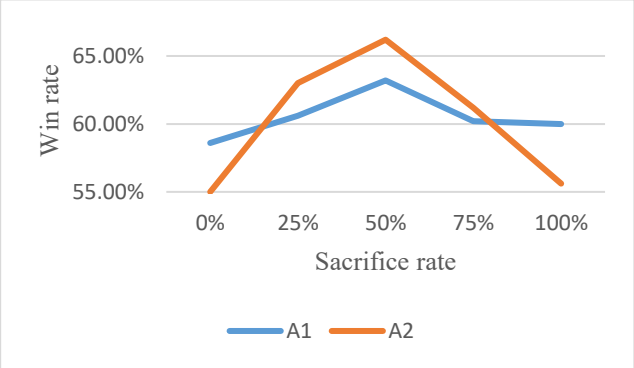


Fig. 4. Combination win rate trend

Of all the combinations obtained in Table 4, S50A2 has the most wins and the fewest losses, and has the highest win rate of 66.2% against Naotti2020.

Fig. 4 shows the win rate line graphs for different sacrifice rates combined with different anti-loop mechanism patterns. The X-axis represents the sacrifice rate, starting from 0% and conducting experiments every 25% up to 100%; the Y-axis represents the win rate. The blue line represents the win rate trend of A1, and the orange line represents the win rate trend of A2.

Based on the trends in Fig. 4, it can be seen that the combination of the sacrifice strategy and the anti-loop mechanism pattern complement each other. Regardless of the substitution pattern, a 50% sacrifice rate yields the best performance after combining with the sacrifice strategy. Moving the sacrifice rate towards extreme values (0, 100) actually

© 2026 Information Processing Society of Japan

decreases the win rate. The experimental results show that when the original strategy and the sacrifice strategy are evenly matched (sacrifice rate = 50%), the combination with anti-loop mechanism yields the best results.

Except for the single-strategy settings (S0 and S100), advantageous substitution (A2) is superior to global substitution (A1) in all other cases. This shows that after a cycle has occurred, substitution should only be made when in an advantageous position; while when in a disadvantageous position, continuing the cycle to maintain a draw is also effective.

5.5 Actual match analysis

Name	Rank	Point
Anon Taiwan	1	1108
S50A2	2	966
RedSearch	3	947
Benny	4	862
JoJo	5	583

Table 5 TCGA2026 Geister match result (Point)

Agent	Opponent	WinRate	Win	Draw	Lose
S50A2	Anon Taiwan	40.00%	40	0	60
S50A2	Benny	61.00%	61	2	37
S50A2	JoJo	73.00%	73	1	26
S50A2	RedSearch	47.00%	47	0	53

Table 6 TCGA2026 Geister match result (Win rate)

Table 5 shows the scores during the TCGA2026 Geister competition. The base score is 300 points, then +3 points for a win and +1 point for a draw. The S50A2 proposed in this study came in second place.

In terms of win rate performance in Table 6, the lowest win rate is 40% and the highest is 73%. Even against the champion, the win rate remained at 40%, and the second-lowest win rate (47%, against RedSearch) was close to 50%. Since the rankings are based on points and S50A2 did not suffer any heavy defeats, it ultimately secured second place.

6. Conclusions and Future Prospects

This study proposes an additional evaluation mechanism for Red pieces, implemented as the Sacrifice strategy, combined with a probabilistic strategy-mixing mechanism and a cyclical substitution mechanism, implemented as the Anti-Loop mechanism. These methods aim to address the limitations of traditional search strategies in high-draw situations in Geister. Experimental results demonstrate that the hybrid strategy improves the agent's ability to overcome stagnant positions against homogeneous conservative opponents and substantially reduces the draw rate. Furthermore, when facing opponents with unknown strategies, strategy diversification makes it more difficult for opponents to predict the agent's exact move patterns, while strategic actions targeting corner positions can provide favorable outcomes. However, against opponents with stronger positional evaluation and escape strategies, the probabilistic hybrid strategy proposed in this study may become a weakness, as the introduced randomness may prevent the agent from selecting optimal actions at critical moments.

Future improvements can be made in the following directions:

1. Introduce opponent type identification (conservative/aggressive/random) and adaptively adjust the mixed probabilities.

2. Establish a dynamic weighting mechanism to automatically adjust the Sacrifice evaluation items according to the stage of the game.

3. Extend the substitution step from a single-step rule to a short-window historical evaluation to improve the anti-loop decision-making quality.

4. Design a compensation mechanism for situations where the lure fails (i.e., the Red piece reaches the escape points but is not captured), aiming to guide the game to a draw to reduce losses.

5. Conduct more sample and cross-version matches to verify the generalization ability of the strategy under different game distributions.

Overall, this study demonstrates that, without relying on deep learning models, the decision-making quality and tactical performance of AI agents in imperfect-information environments can be improved by optimizing local evaluation functions, employing probabilistic strategy mixing, and introducing a cyclical substitution mechanism. The experimental results show that the proposed S50A2 agent reduced the number of draws from 306 in the baseline experiment to 19 in the 500-game evaluation, while achieving a win rate of 66.2%. In addition, S50A2 maintained a win rate of at least 40% against all four tested opponents in the TCGA 2026 competition. Although the proposed strategy still has room for improvement against opponents with stronger positional evaluation capabilities, this study demonstrates that combining a traditional search architecture with explicit strategy design can achieve competitive performance without relying on deep learning models.

It is hoped that the method and experimental results presented in this study can serve as a reference for future AI research in Geister and other imperfect-information board games, and provide valuable development directions for the design of related decision-making systems.

Acknowledgement

The authors would like to thank anonymous referees for their valuable comments in improving the overall quality of this paper. This work was supported in part by the National Science and Technology Council under contracts NSTC 113-2221-E-259-018-MY3 and NSTC 115-2622-E-259-001.

REFERENCES

[1] 川上直人 and 橋本剛, "『紫駒』を用いた MinMax 探索によるガイスターAI の研究," IPSJ Journal, vol. 62, no. 10, pp. 1716–1723, Oct. 2021.

[2] 末續鴻輝 and 織田祐輔, "機械学習を用いないガイスターの行動アルゴリズム開発," in Proc. Game AI Tournament (GAT2018), Information Processing Society of Japan, 2018.

[3] 原田響己 and 吉浦紀晃, "R-NaD を用いたガイスターAI の研究," 情報科学技術フォーラム (FIT), vol. 23, no. 2, pp. 121–124, 2024.

© 2026 Information Processing Society of Japan