Analyzing simplified Geister using DREAM

Lucien TROILLET
Graduate School of Engineering
Kochi University of Technology
Kami, Japan
246005s@gs.kochi-tech.ac.jp

Kiminori MATSUZAKI
School of Information
Kochi University of Technology
Kami, Japan
matsuzaki.kiminori@kochi-tech.ac.jp

***Abstract*—Geister is a board imperfect information game created in Germany and presenting an interesting challenge for the field of artificial intelligence. In this study we apply DREAM (Deep Regret Minimization with Advantage Baselines and Model-free Learning), a neural-network variation of Counterfactual Regret Minimization developed by Steinberger et al., to multiple variants of Geister. This paper shows a methodological approach of evaluating game strategies on different variants of Geister and illustrates the possible generalizability of the DREAM algorithm on other board games.**

***Index Terms*—Geister, Imperfect Information Games, DREAM, CFR**

I. Introduction

Geister, also called Ghosts, is a Board Imperfect Information Game (IIG) from Germany [1]. Research on IIGs so far has been focused on card and video games [2] [3] [4] [5] [6] [7] but not much on board games despite them presenting different challenges, specifically, they have bigger input and action spaces, they require spatial awareness and, in many cases, they contain possible reoccurring states which can lead to infinite games lengths. Additionally, Geister is a non random game, its gameplay is very similar to that of some solved Perfect Information Games like Chess and it has multiple win conditions that a player must consider to be efficient.

While several studies on IIGs use multiple games as training and testing environments to improve and create better algorithms, they mostly do not analyze the games themselves. As far as the authors know, only a few studies did an attempt of analyzing games as in [8] where multiple players were used to gather information about game complexity on different game variants.

In this paper we propose a different approach to analyzing games through advanced players trained on game variants. We demonstrate our approach by using reduced variants of Geister similar to the one used in [9]. Geister has the advantage of being easily modified by increasing or decreasing its board size without needing to change any core rules of the game. These modifications do however change the strategies a player must employ to be effective. We take inspiration from [10] and use a CFR variation called DREAM to train efficient players on each Geister variant. We analyze how the trained players change their behaviors to better suit the different environment in which they evolve.

In section II we present the game of Geister and its simplified variants. In section III we briefly introduce the DREAM algorithm. We present our methodology in section IV then show the results in section V and discuss them in section VI. We conclude that the smallest variants of Geister are mostly random but that playing intelligently is rewarded once each player has at least three pieces and a board size big enough to accommodate them and that some game variants reward baiting and avoiding, some reward aggressive captures and some reward escaping.

II. Geister

A. Full Geister

Geister is played on a square board with 6 rows and 6 columns. Both players control eight pawns/ghosts, four of them are red/evil and four are blue/good. Each player sets their ghosts as they wish in a given space at the start of the game. They can't see the layout of the opponent's ghosts. Fig. 1 shows an example starting board. Sente, the first player, starts by moving one of their ghosts by one square in any straight direction. If a ghost moves to an opponent's ghost's square, the opponent's ghost is captured and its color revealed. Then Gote, the second player, can do the same. They continue

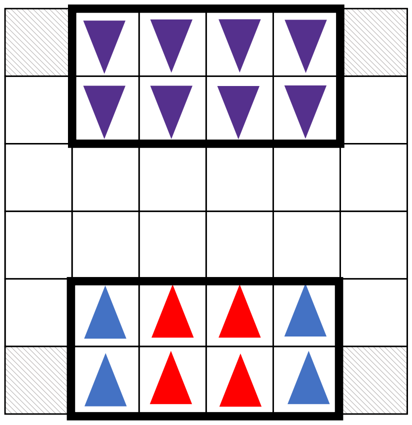
Fig. 1. Example starting board from Sente's point of view. Purple ghosts' true blue/red colors are hidden.

This work is partially supported by JSPS KAKENHI Grant Number JP20K12124.

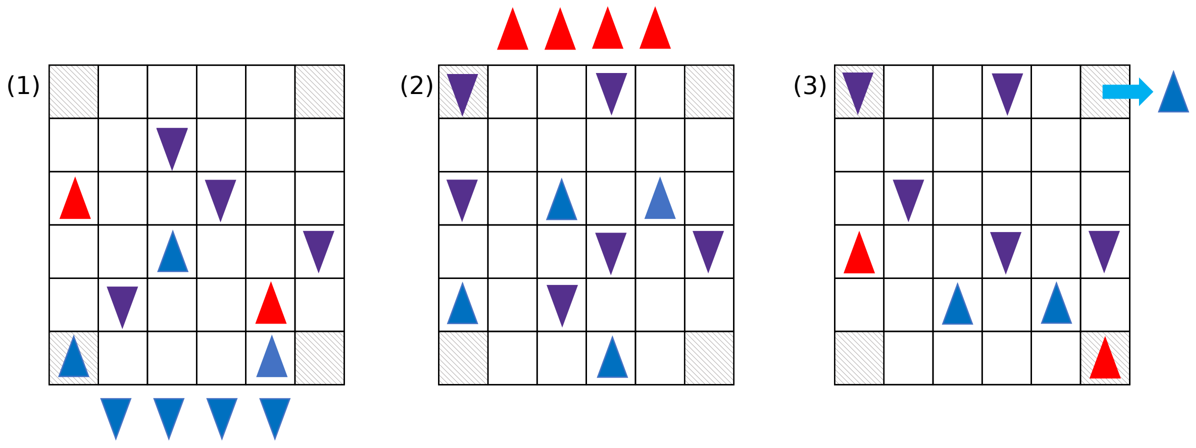

Fig. 2. Geister win conditions. Purple ghosts' true blue/red colors are hidden.

alternating until the end of the game without passes. The game ends when a player fulfills one of the following win conditions (Fig.2).

1) Blue capture: A player captures all opponent's blue ghosts.
2) Red capture: A player gets all their own red ghosts captured by the opponent.
3) Escape: A player manages to bring one of their blue pawns to one of the corners on the opponent's side and makes it move out of the board.

B. Simplified Geister

We focus on simplified variants of Geister. The original size of Geister is modified by shrinking the board width and height, and by reducing the number of ghosts. The different board sizes and number of ghosts we use in our experiments are shown in table I. Our first design forced an even number of ghosts but we decided to include two cases, x3y3g3Rb0 and x3y3g3Bb0, where the number of ghosts is uneven to examine how an excess of red or blue ghosts might influence the game.

TABLE I
GEISTER VARIANTS

Board Identifier	Board width	Board height	Ghosts	Buffer columns
x6y6g8b1[a]	6	6	8	1
x2y3g2b0	2	3	2	0
x2y4g2b0	2	4	2	0
x4y2g2b0	4	2	2	0
x3y3g2b0	3	3	2[b]	0
x3y3g3Rb0	3	3	3 (2 Red)	0
x3y3g3Bb0	3	3	3 (2 Blue)	0
x4y3g2b1	4	3	2	1
x4y3g4b0	4	3	4	0
x4y4g2b1	4	4	2	1
x4y4g4b0	4	4	4	0
x4y4g4b1	4	4	4	1

[a]Default Geister
[b]Middle column left empty

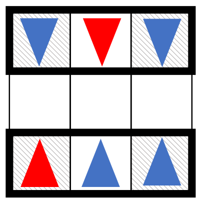

Fig. 3. Example Geister x3y3g3Bb0 starting board

The last setting we adapt is the number of columns left free of ghosts on each side of the board at the start of the game. We refer to them as buffer columns or simply buffer. The default Geister board has one buffer column but most of the variants we analyze do not have any.

Winning conditions stay the same for every variant.

III. DREAM

DREAM(Deep Regret Minimization with Advantage Baselines and Model-free Learning) is a regret-based deep reinforcement learning algorithm developed for Poker [11]. It is the latest variation of the basic CFR algorithm.

CFR and its multiple improvements serve as a basis for DREAM.

The original *CFR* algorithm [12] [13] iteratively calculates the regrets of not taking an action compared to usual gameplay and averages them across all iterations to create a policy.

Improving upon this, *Linear CFR* [14] weighs each iteration's regret update by the iteration number to accelerate the training process.

Monte Carlo CFR [15] additionally allows the regret updates to work without exploring the whole field of possibilities of the game. It uses sampling for a single agent at a time and only updates regrets along the explored agent's trajectories. It samples according to a combination of the latest calculated agent policies and some element of randomness to maintain exploration of new trajectories.

Variance-Reduced Outcome Sampling with Baselines' [16] [17] variant of CFR adds a user given baseline function to reduce the variance of sampled values by estimating them automatically on a given state.

Deep CFR and Single Deep CFR [18] add the use of a neural network to predict values of unseen states and then uses the network to sample further in a repeating loop.

Finally, *DREAM* uses and builds upon all these previous algorithms by also training a network to serve as a baseline function instead of having a user defined one. Thus, it trains two networks per player even though only one is used post learning process. It has already been applied to the game of Geister [10] with good results.

IV. Experiment

A. Adaptations

As explained in Sections I and II, Geister can go on for an infinite number of turns and requires strategy before even starting the game to place ghosts in an optimal manner. Others dealt with these issues by using multiple different techniques [10]. While some of them are also adapted to the simplified Geister variant we use here, some are not or are not used for the same reason. In the following sub-section we briefly go over game adaptations and how they are applied in this study.

Abstraction: CFR algorithms normally have access to the full infoset of a game. However, using only the latest turn's game state has been shown to work and results in players deemed acceptable [19].

Length limitation: Avoiding endless games is an important consideration to prevent training from being slowed down by playouts with excessive length. In our experiments we set a turn limit at 100. Any game reaching that point is considered a draw during training and experiments.

Random initialization: Given the very low number of possible starting boards for the simplified Geister variants, generating starting boards randomly will rapidly cover all possible scenarios and allow the network to adapt to each of them.

Finishing move assistance: The DREAM agents should naturally learn to perform escape actions when possible and random agents are used as an illustration of a non-strategic player. Thus we do not use such an assistance.

B. Encoding

A simple encoding of any board with three layers of matrices allows to describe the board state perceived by each player. One layer accounts for the position of a player's blue ghosts, another accounts for the same player's red ghosts, and the last one is used to track all opponent ghost positions. This is illustrated in Table II.

However, Geister has other attributes that do not reside on the board. These meta-attributes are the number of blue ghosts a player has captured, the number of red ghosts they captured, the turn number and a boolean value indicating if it is that player's turn to move. We encode this information in a separate, four dimensional, vector detailed in Table III.

TABLE II
Board encoding

Encoding layer	Cell information
1	Player's blue ghosts
2	Player's red ghosts
3	Opponent's ghosts

TABLE III
Meta encoding

Encoding dimension	Meta Information
1	Captured blue pieces
2	Captured red pieces
3	Current turn
4	Is next player

C. Networks

As a board game, Geister lends itself well to the use of convolutional networks. However the meta-attributes are not well suited to convolution without some adaption. In [19], this issue is worked around by inserting the meta-information into the board encoding allowing for a pure convolutional network. However, we choose to combine the use of convolutional layers for the board encoding and fully connected layers for the meta information. We then flatten the convolutional layer outputs, concatenate them with the fully connected layer ones and feed them to a new set of fully connected layers, which we call *combined layers* for ease of distinction with the first set of fully connected layers. All layers in our networks are activated by a ReLU function.

Both the Baseline and Advantage Network follow this structure. The difference between the two is that the layer structure before concatenation is duplicated in the Baseline network to receive both Sente and Gote's infosets. Then, both flattened convolutional layer outputs and both fully connected layer outputs are concatenated and forwarded to the combined layers.

The network structure we use is made to scale seamlessly with the size of the board. This is done to simplify the creation of different experiments with different board sizes. To do so, we define the number of channels for each convolution layer output and the number of neurons in each fully connected layer and combined layer. The convolution process is always performed using 3x3 kernels and a padding value of one. These parameters allow any board to be used as input for the network and have a constant size through all convolution layers. It has the advantage of having the network adapt its size to different board types.

The fully connected layers and combined layers of the network mostly do not scale with board size. The output layer is the only exception. It contains four times the number of squares on the board. Each output neuron corresponds to one movement direction associated to one square. Border squares are also given four corresponding neurons even though some of the corresponding directions are not possible to move into. These impossible directions are filtered out from the output afterwards.

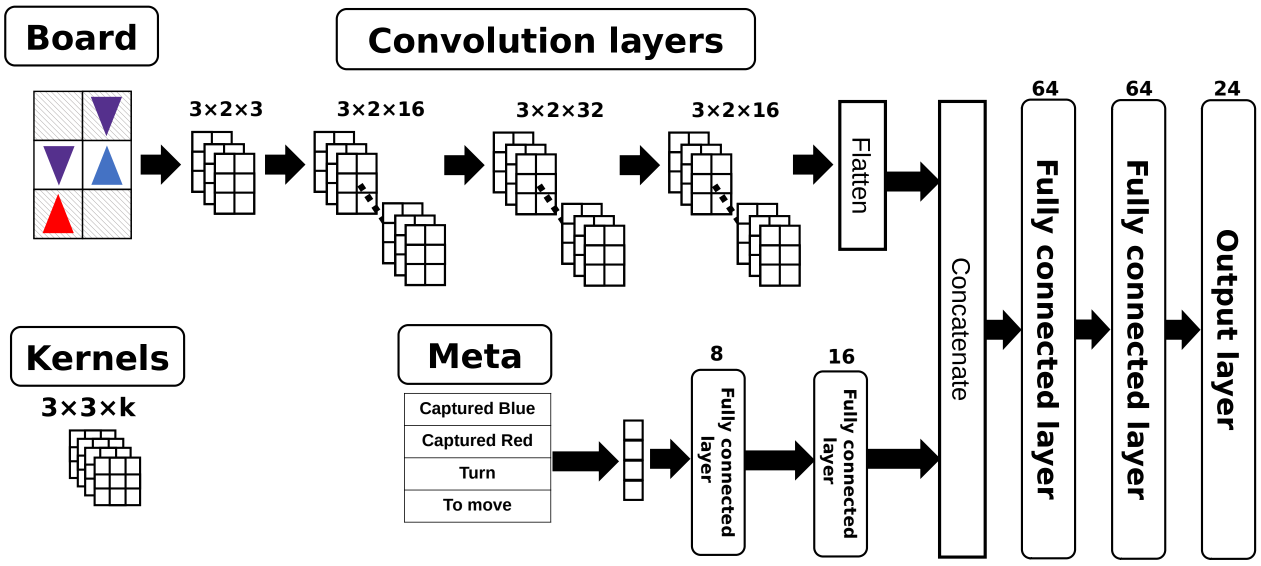

Fig. 4. Advantage Network with an x2y3g2b0 variant board

The three convolutional layers use 16, 32 and 16 channels, the fully connected layers have 8 and 16 neurons, and the combined layers have 64 and 64 neurons. Fig. 4 illustrates the network structure using a board example from the x2y3g2b0 game variant. During the learning process, a total of four networks are trained. One baseline and one advantage network for each player.

D. Experimental Parameters

Our code implementation was run on different machines that do not all support the same hardware. In the interest of brevity, we only present the results obtained on one type of machines equipped with an AMD Ryzen 7 3800X CPU and a NVIDIA RTX 2060 SUPER GPU using python 3.7.10, pytorch 1.8.0 and cudatoolkit 11.1.1. Performing the same experiments on different hardware did not change results significantly.

Through prior trials, we tested a variety of different hyperparameters on the simplest Geister variant used in our experiments, x2y3g2b0. The smallest variant is inherently the most random as there is almost no space on the board for a player to get past its opponent's ghosts and manage to win by escape and because any capture of a ghost ends the game immediately. Having too small hyperparameters for this variant of the game results in very unstable losses through the training process. We used this to scale hyper parameters up until losses were more stable while still having a reasonable experiment duration.

The batch size was selected to match the GPU cuda core count which is 2176 for the RTX2060S. The buffer capacity was 1280000 and was originally chosen to be 500 times the batch size of the RTX2070 SUPER on which we experimented in early tests. This capacity was then kept the same for all other experiments to have the same number of samples the networks could train with in all runs. The training lasted 1000 iterations. For each iteration the game was traversed 4096 times using 4 samplers. Alternating iterations were used to train alternating players. Half were updating the network for Sente and half for Gote.

Both Baseline and Advantage networks were tuned using the Adam optimizer with a learning rate of 0.001 and a mean square error Loss function. To have the sampled losses of later iterations be more relevant, they were weighted by the iteration they were sampled at.

Once the training is finished, the resulting DREAM players are compared to random players by having them play 100 games. For each game, the starting board is generated randomly and the agents play the game twice, once as Sente and once as Gote.

This testing process is repeated every ten iterations of the learning process using the generated networks to dictate which move the DREAM player chooses.

V. Results

A. Loss

There are significant differences among loss behaviors of different variants of the game. The advantage loss scales with the number of iterations so it is not possible to see a stabilization by observing it. However, the baseline network loss has a more stable behavior so we can use it to have an insight in the learning progression.

Fig. 5 graphs the baseline network loss for Sente and Gote with varying board sizes. A decrease of loss can be seen with increased board sizes. The 4x4 board size delivering the lowest losses for Sente and Gote among the graphed boards. It also has the most flat loss curves. The x2y3 board has the most

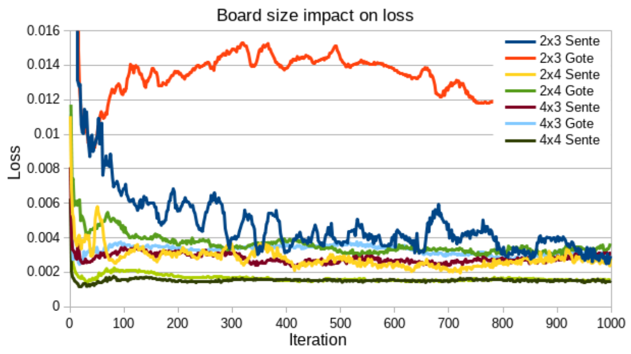

Fig. 5. Baseline network loss progress for varying board sizes (2 ghosts)

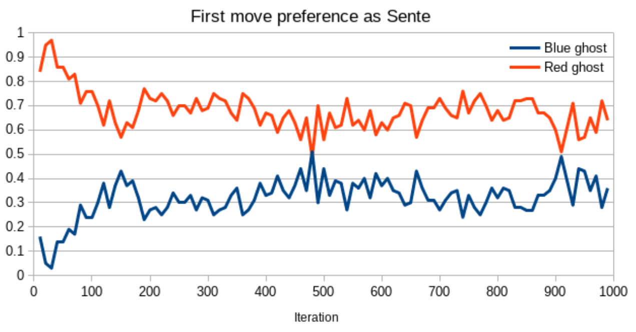

Fig. 7. Color of first piece moved as Sente in x2y3g2b0 variant

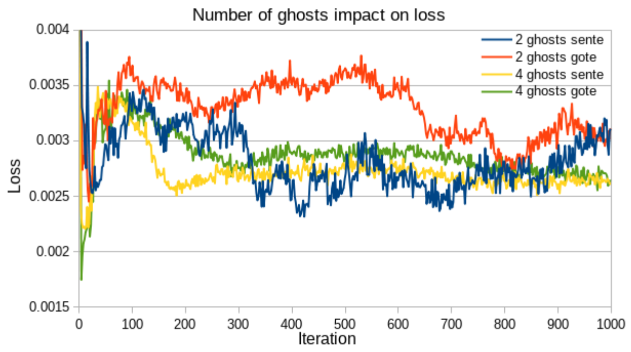

Fig. 6. Baseline network loss with x4y3 board and varying number of ghosts

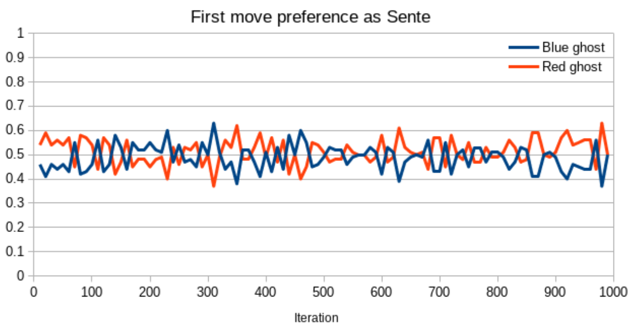

Fig. 8. Color of first piece moved as Sente in x4y4g4b0 variant

high and unstable losses. Both x2y4 and x4y3 boards achieve similar loss progressions.

Fig. 6 illustrates how different number of ghosts also result in different loss behaviors. Going from 2 to 4 ghosts per player with the same board size of x4y3 generates a more stable loss progression and causes sente and gote to have closer losses.

B. First move color

Table IV is composed of basic DREAM player statistics we tracked for each session of 100 battles between it and the Random player over all iterations.

The DREAM player has a disproportionate tendency to move a red ghost first across all iterations in almost all boards. This is even true for boards with an uneven number of blue and red pieces: the x3y3g3 variants also have a disproportionate amount of red first moves relative to their proportion of red ghosts.

However this tendency becomes less visible with bigger boards. The x4y4g4b0 board variant even has a statistical tie with regards to the first move preference. Figures 7 and 8 show the progression of the tendency over all iterations for a smaller and bigger variant and reinforce that bigger boards and higher number of ghosts make the preference disappear. The shift to more neutral first moves with increasing number of ghosts is disputable looking at the data from Table IV, more experiment runs would be needed to ensure the statistical significance of this observed trend.

There is one case where the DREAM player has a preference to move a blue ghost first. It is the x4y2g2b1 variant where it starts playing with a blue ghost move almost two thirds of the time.

Looking at Fig. 7 it is also apparent that the red move first bias is stronger in early stages up to around the 100th iteration. As observable in Fig. 9, this coincides with a higher number of wins when playing as Sente at the same range of iterations.

C. Win rate

The win rates for Sente and Gote are presented in Table IV. The win rate as Gote for the smallest variant of the game is statistically random. The difference in number of wins when playing as Sente or Gote becomes much smaller as the game size increases.

Regarding the win rate, the highest performances of the DREAM player against the Random player is achieved in variants x4y3g4b0 and x4y4g4b0. The two following best performances both come from the x3y3g3b0 variants of the game.

D. Win type

Since the first move tendencies disappear in bigger variants of the game, we decided to run a new set of battle experiments using the same networks and added tracking of what win condition players fulfilled in each game.

Figures 10 and 11 illustrate this ratio of win types over all 100 played games in DREAM Sente vs. Random Gote and

TABLE IV
GEISTER VARIANTS STATISTICS[a]

Board Identifier	First move Blue	First move Red	Wins as Sente	Wins as Gote	Total Wins
x2y3g2b0	0.33 — 0.33 (0.08)	0.67 — 0.67 (0.08)	62 — 61.35 (6.33)	52 — 52.27 (4.56)	113 — 113.63 (7.45)
x3y3g2b0	0.4 — 0.39 (0.09)	0.6 — 0.61 (0.09)	61 — 62.03 (4.64)	58 — 57.8 (4.76)	119 — 119.83 (6.9)
x3y3g3Rb0	0.08 — 0.09 (0.05)	0.92 — 0.91 (0.05)	69 — 68.57 (4.71)	65 — 64.79 (4.95)	134 — 133.35 (7.78)
x3y3g3Bb0	0.6 — 0.6 (0.07)	0.4 — 0.4 (0.07)	66 — 65.9 (4.88)	64 — 64 (4.97)	130 — 129.9 (7.14)
x2y4g2b0	0.36 — 0.36 (0.07)	0.64 — 0.64 (0.07)	62 — 62.49 (5.3)	59 — 58.98 (5.05)	122 — 121.47 (7.18)
x4y2g2b1	0.62 — 0.63 (0.08)	0.38 — 0.37 (0.08)	63 — 63 (5.14)	53 — 53.41 (5.27)	116 — 116.41 (7.43)
x4y3g2b1	0.41 — 0.41 (0.08)	0.59 — 0.59 (0.08)	66 — 65.85 (5.53)	63 — 62.89 (4.99)	128 — 128.74 (8.1)
x4y3g4b0	0.34 — 0.34 (0.07)	0.66 — 0.66 (0.07)	77 — 76.69 (5)	76 — 75.76 (6.19)	153 — 152.44 (9.08)
x4y4g2b1	0.45 — 0.44 (0.08)	0.55 — 0.56 (0.08)	65 — 65.19 (5.48)	64 — 64.51 (5.35)	131 — 129.7 (8.01)
x4y4g4b0	0.49 — 0.49 (0.05)	0.51 — 0.51 (0.05)	73 — 72.67 (5.55)	71 — 70.34 (6.83)	145 — 143.01 (10.16)
x4y4g4b1	0.55 — 0.55 (0.06)	0.45 — 0.45 (0.06)	66 — 64.82 (6.97)	62 — 61.99 (6.27)	129 — 126.81 (10.25)

[a] Median — Average (Standard deviation)

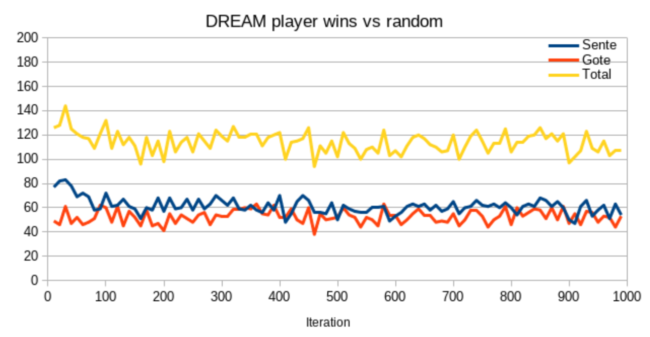

Fig. 9. Number of wins against random player over 100 games in x2y3g2b0 variant

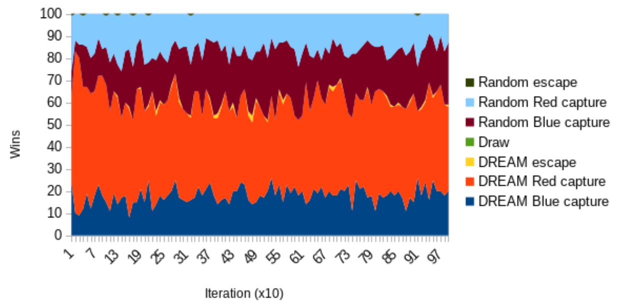

Fig. 10. Win type progress with DREAM as Sente in x2y3g2b0 variant

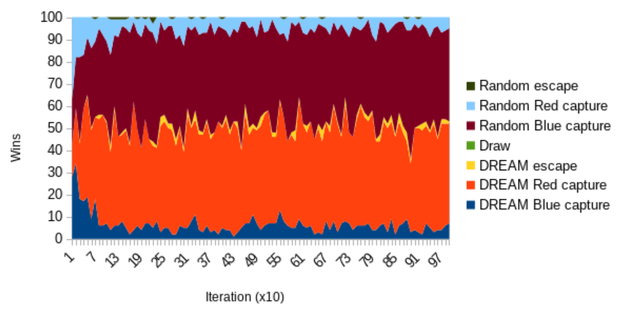

Fig. 11. Win type progress with DREAM as Gote in x2y3g2b0 variant

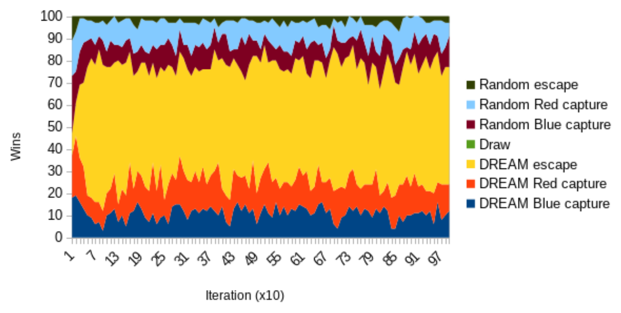

Fig. 12. Win type progress with DREAM as Sente in x4y3g4b0 variant

DREAM Gote vs. Random Sente configurations respectively. In both cases there is an extremely low number of wins by escaping the board. When the DREAM Player is in the Gote position, most wins come from having its opponent capture its red ghost and most of the random player's wins come from capturing the DREAM agent's blue ghost.

In stark contrast to the smallest variant of Geister, the x4y3g4b0 variant, already shown to be the one with the highest win rate for the DREAM agent is also the one with the most wins by escape, as shown in Fig. 12. In this case, the DREAM agent has a strong preference for wins by escape.

The two x3y3 variants of the game show contrasting behaviors. Fig. 13 has a very high fraction of wins achieved by capturing a blue ghost.

Fig. 14 shows the opposite behavior emerging. The majority of the DREAM agent's wins are the result of the Random agent capturing its red ghost.

VI. ANALYSIS

A. Red baiting

The observed tendency of the red ghost to be moved first, especially in smaller games, points to the DREAM agent attempting to bait the other player into capturing their red ghost by making it more easily accessible. The only case with a strong blue first move preference can also be seen to reinforce this baiting strategy hypothesis. In the x4y2g2b1

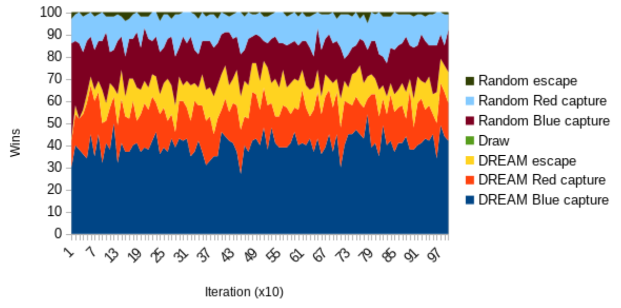

Fig. 13. Win type progress with DREAM as Sente in x3y3g3Rb0 variant

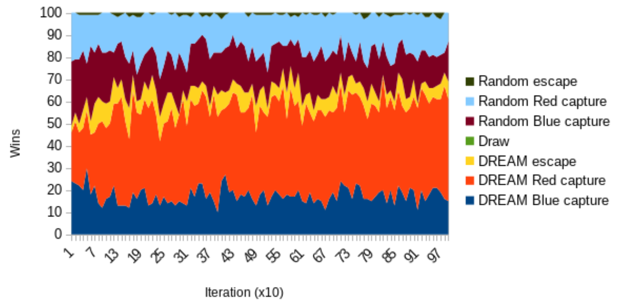

Fig. 14. Win type progress with DREAM as Sente in x3y3g3Bb0 variant

board, where the height is only 2 squares, Sente and Gote's ghosts start right in front of each other. It makes sense that the DREAM player would develop a tendency to move the blue ghost first as it allows to shift it sideways, out of immediate risk of being taken by Gote's first move, this leaves the red ghost in Gote's range and brings its blue ghost closer to escaping.

The red baiting strategy idea is reinforced by the correlation between red first move and win percentage in early iterations in the x2y3g2b0 board. It is likely a sign that, early in its training process, the DREAM player finds out that it can get a lot of wins by increasing the likelihood of its red ghost to be captured by placing it in front of enemy pieces. This is a good strategy against the random player which has a 50% chance to capture the ghost on such a small board. However the decline of this tendency versus the random player seemingly indicates the DREAM player learns not to overexploit this strategy once its opponent starts adapting to it.

The last strong evidence for the red baiting strategy in smaller boards is the high proportion of games ending with the random player capturing a ghost when the DREAM player is Gote, proving that the game almost always ends with a capture move from the random player. Thus, **the DREAM player actively avoids capturing the opponent's ghost** at least during its first turn and **attempts to end the game by baiting with its red ghost**. This is visible when the DREAM agent plays as Sente too but to a lesser degree.

B. Asymmetrical gameplay

The loss figures seem to show that bigger boards result in smaller losses. This is probably due to bigger boards allowing players to use more strategy in movements as opposed to playing purely reactively to the opponent's movements. This reactive gameplay seems especially significant for Gote who, in smaller boards, never achieves losses similar to Sente's.

The stronger red baiting presence and the high loss difference between Sente and Gote on smaller boards points to the gameplay for Gote being highly reactive. I.e. **smaller Geister variants are not games where each player is free to use a strategy**. Sente can behave as they wish and Gote must adapt hence creating a one-sided game.

Further reinforcing the idea of a one-sided game with smaller variants are the win rates for Sente and Gote presented in Table IV. The statistically random win rate as Gote for the smallest variant of the game shows how much that variant's gameplay is dictated by Sente's actions. The difference in number of wins when playing as Sente or Gote becomes much smaller as the game size increases. This can also be seen in the win type tables where, when the DREAM agent plays as Gote, despite it visibly avoiding capture moves to bait a red capture, it doesn't manage to get a significant advantage. However, the win rate of the DREAM player not being much higher when playing as Sente also indicates that, **despite the first player advantage, the smallest game variant remains very random and hard to strategize efficiently for**.

C. Number of Ghosts

Having an increase in number of ghosts seems to modify the loss behavior. It should be pointed out that going from 2 to 4 ghosts removes the "sudden death" component of a capture. Multiple ghosts of the same color allow to capture and then adapt. The stabler loss behavior is probably due to this change.

The highest win rates being achieved in the only variants of Geister to use 4 ghosts per player is a strong indication that the added number of pieces is an important element to have in order to allow an agent to benefit from a strategy. The second highest win rates being achieved on the boards with the second highest number of pieces despite a smaller board size further reinforces that **the number of ghosts has a bigger impact on how efficient a strategy can be than the size of the board**. This is probably also why loses with bigger game variants tend to be better and more stable.

Analyzing the most common types of wins in different variants of the game also reinforces how **having a higher number of pieces influences the behavior of a player**. This is especially visible in the x3y3g3Rb0 and x3y3g3Bb0 variants of the game. In the x3y3g3Rb0 variant, the DREAM player visibly becomes much more aggressive with regards to capturing other ghosts. The reason for this is probably that the DREAM player tries to capture a piece as soon as possible knowing it will always survive this first capture and can still adapt its strategy if this first capture does not result in a win. It probably also allows it to be less restricted in its movements by the enemy ghosts. This is reversed in the x3y3g3Bb0 variant:

the DREAM agent seems to be attempting to bait the other player into capturing its only red ghost.

Going back to the 4 ghosts variants of the game, they show the highest percentage of wins by escaping. In these variants, the DREAM agent seems to prefer the strategy least reliant on randomness as escaping is the only win type not reliant on hidden information.

VII. Conclusion

In this paper we presented the game of Geister, its rules and the modified variants we created. We explained the challenges of Board IIGs and the more specific challenges of Geister. We briefly explained the DREAM algorithm and used it to generate intelligent players for each Geister variant we studied. We then compared their performance to random players by having them play against each other throughout the training of the DREAM agents.

Over all tested variants, the DREAM player favors strategies that reduce the influence of imperfect information. Intelligent agents prefer avoidance strategies in smaller variants, escape strategies in bigger variants and capture strategies only when they are low risk high reward.

An intelligent player's performance in the smallest variants of Geister is limited by the inherent randomness of having too few ghosts and forced reactionary gameplay. Having at least three ghosts per player and a board size allowing them to move more freely allows intelligent strategies to give them a true advantage and mitigates the impact of the luck inherent to trying to capture an opposing ghost.

While we saw more complex behaviors emerging with bigger Geister variants, we have not yet properly tested our implementation up to the default Geister game. There might be more interesting things to observe when scaling up but bigger networks and hyperparameters will probably be needed to capture the increased complexity. Our future work will analyze the game's bigger variants. Applying similar experiments to other games might also yield interesting insights.

Acknowledgment

We would like to thank Chen and Kaneko for the help they gave us by providing much welcomed aid in understanding the DREAM algorithm and giving us access to their basic implementations of the algorithm for Kuhn Poker which we used as a basis to develop our own experiments.

References

[1] "Top | Geister Online." [Online]. Available: https://geister.tetsis.com/

[2] int8, "Counterfactual Regret Minimization - the core of Poker AI beating professional players," Sep. 2018.

[3] R. G. Gibson, "Regret minimization in games and the development of champion multiplayer computer poker-playing agents," Ph.D. dissertation, Department of Computing Science, University of Alberta, 2014.

[4] M. Bowling, N. Burch, M. Johanson, and O. Tammelin, "Heads-up limit hold'em poker is solved," *Science*, vol. 347, no. 6218, pp. 145–149, Jan. 2015.

[5] N. Brown and T. Sandholm, "Superhuman AI for multiplayer poker," *Science*, vol. 365, no. 6456, pp. 885–890, Aug. 2019.

[6] N. Bard, J. N. Foerster, S. Chandar, N. Burch, M. Lanctot, H. F. Song, E. Parisotto, V. Dumoulin, S. Moitra, E. Hughes, I. Dunning, S. Mourad, H. Larochelle, M. G. Bellemare, and M. Bowling, "The Hanabi challenge: A new frontier for AI research," *Artificial Intelligence*, vol. 280, p. 103216, Mar. 2020.

[7] O. Vinyals, I. Babuschkin, J. Chung, M. Mathieu, M. Jaderberg, W. M. Czarnecki, A. Dudzik, A. Huang, P. Georgiev, R. Powell, T. Ewalds, D. Horgan, M. Kroiss, I. Danihelka, J. Agapiou, J. Oh, V. Dalibard, D. Choi, L. Sifre, Y. Sulsky, S. Vezhnevets, J. Molloy, T. Cai, D. Budden, T. Paine, C. Gulcehre, Z. Wang, T. Pfaff, T. Pohlen, Y. Wu, D. Yogatama, J. Cohen, K. McKinney, O. Smith, T. Schaul, T. Lillicrap, C. Apps, K. Kavukcuoglu, D. Hassabis, and D. Silver, "AlphaStar: Mastering the real-time strategy game StarCraft II," https://deepmind.com/blog/alphastar-mastering-real-time-strategy-game-starcraft-ii/, 2019.

[8] M. Wakatsuki, Y. Kado, Y. Takeuchi, S. Okubo, and T. Nishino, "What are the Characteristics of the Card Game Daihinmin?" in *2019 8th International Congress on Advanced Applied Informatics (IIAI-AAI)*, Jul. 2019, pp. 587–592.

[9] C. Chen and T. Kaneko, "Counterfactual regret minimization for the board game Geister," in *Proceedings of the Game Programming Workshop (GPW 2018)*, nov 2018, pp. 137–144.

[10] ——, "Application of DREAM to the board game Geister," in *Proceedings of the Game Programming workshop (GPW 2020)*, nov 2020, pp. 111–117.

[11] E. Steinberger, A. Lerer, and N. Brown, "DREAM: Deep Regret minimization with Advantage baselines and Model-free learning," *arXiv*, vol. 2006.10410 [cs, stat], Nov. 2020.

[12] M. Bowling, M. Johanson, M. Zinkevich, and C. Piccione, "Regret minimization in games with incomplete information," in *NIPS'07: Proceedings of the 20th International Conference on Neural Information Processing Systems*, 2007, pp. 1729–1736.

[13] T. W. Neller and M. Lanctot, "An introduction to counterfactual regret minimization," in *Proceedings of Model AI Assignments, The Fourth Symposium on Educational Advances in Artificial Intelligence (EAAI-2013)*, 2013.

[14] N. Brown and T. Sandholm, "Solving imperfect-information games via discounted regret minimization," in *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019*, 2019, pp. 2157–2164.

[15] M. Lanctot, K. Waugh, M. Zinkevich, and M. Bowling, "Monte Carlo Sampling for Regret Minimization in Extensive Games," *Advances in Neural Information Processing Systems*, vol. 22, pp. 1078–1086, 2009.

[16] M. Schmid, N. Burch, M. Lanctot, M. Moravcik, R. Kadlec, and M. Bowling, "Variance Reduction in Monte Carlo Counterfactual Regret Minimization (VR-MCCFR) for Extensive Form Games using Baselines," *arXiv*, vol. 1809.03057 [cs], Sep. 2018.

[17] T. Davis, M. Schmid, and M. Bowling, "Low-variance and zero-variance baselines for extensive-form games," *arXiv*, vol. 1907.09633 [cs], Jul. 2019.

[18] N. Brown, A. Lerer, S. Gross, and T. Sandholm, "Deep counterfactual regret minimization," in *Proceedings of the 36th International Conference on Machine Learning, ICML 2019*, 2019, pp. 793–802.

[19] C. Chen and T. Kaneko, "Acquiring strategies for the board game Geister by regret minimization," in *2019 International Conference on Technologies and Applications of Artificial Intelligence (TAAI)*, Nov. 2019, pp. 1–6.