Learning Strategies for Imperfect Information Board Games Using Depth-Limited Counterfactual Regret Minimization and Belief State

Chen Chen
Graduate School of Arts and Sciences
The University of Tokyo
Tokyo, Japan
chen@game.c.u-tokyo.ac.jp
chenchen-319@g.ecc.u-tokyo.ac.jp

Tomoyuki Kaneko
Graduate School of Arts and Sciences
The University of Tokyo
Tokyo, Japan
kaneko@graco.c.u-tokyo.ac.jp

***Abstract*—Counterfactual Regret Minimization (CFR) variants have mastered many Poker games by effectively handling a large number of opportunities in private information within relatively short playing histories of the game. However, for imperfect information board games with infrequent chance events but long histories or even loops, the effectiveness of CFR is often limited in practice as the computational complexity grows exponentially with the game length. In this paper, we propose Belief States with Approximation by Dirichlet Distributions and Depth-limited External Sampling for Board Games that enables an effective abstraction even with existence of loops. Experiments show that our proposed methods have the ability to learn reasonable strategies.**

***Index Terms*—CFR, Belief, Depth-limited, Regret Minimization, Imperfect Information, Board Games**

I. Introduction

Counterfactual regret minimization (CFR) is an effective method for obtaining near optimal strategies in imperfect information games [1]. In real-world tasks such as negotiations, auctions, and cybersecurity interactions, an agent needs to make a decision with limited information. Conducting research on imperfect information games is supposed to give us inspiration for resolving these decision-making problems, therefore it is meaningful for us to do these researches. While CFR worked dramatically well in Texas Hold'em, there is a severe limitation with game length in practice. It is because the computational cost depends on the number of information sets, each of which is a situation distinguishable for agents, and it usually grows exponentially with the game length. To have a near optimal strategy for each information set, CFR repeatedly visits every information set and stores and updates data in tables. The problem still exists for recent studies that incorporated neural networks to extend tabular representation [2]–[4] because the training time increases in general as the number of information sets does.

Geister [5] is a board game with imperfect information. We propose Belief States with Approximation by Dirichlet Distributions and Depth-limited External Sampling for Board Games to abstract the game and deal with loops in board games. We train agents with these methods and evaluate the agents by playing against a random and a heuristic player in a small variant of Geister. The results show that our proposed methods have the ability to learn reasonable strategies.

II. Background

A. Extensive Games

We follow the standard notation of extensive games. See the study [1] for the details. In a finite extensive game with imperfect information, N is a finite set of players, and c stands for a chance player. For player i, $-i$ stands for all other players. In two-player games including Geister, $N = \{1, 2\}$ and player -1 (-2) represents player 2 (1). H is a finite set of possible *histories* h including the sequence of the players' actions, and $Z \subseteq H$ is a finite set of all terminal histories. A prefix h' of history h means that h begins with h'. Because a history typically contains private information, a player cannot distinguish one another among a set of histories. $\mathcal{I}_i$ is a finite set of *information sets* I for player i where each information set is the set of histories that player i cannot distinguish one from another. The player to act and the legal actions at non-terminal history h or information set I are denoted by $P(h)$ and $A(h)$ or $P(I)$ and $A(I)$. Intuitively, an information set corresponds to a position or board state in many perfect information games in the sense that an optimal move to play depends on it. σ is a strategy profile consisting of a *strategy* σ_i for each player i. Σ_i is all strategies for player i. $\sigma_{I \to a}$ represents a strategy profile identical to σ except that action a is always chosen at information set I. In reinforcement learning, a strategy is called *policy*. $u_i(z)$ is player i's *utility* on terminal history $z \in Z$ (i.e., win or loss). $\pi^\sigma(h)$ stands for the probability of reaching history h if players act according to σ, and $\pi_i^\sigma(h)$ is player i's contribution.

B. The Game of Geister

1) Full Geister: Geister is a two-player board game on a 6×6 game board. Initially, each player has four blue ghosts and four red ghosts on the board. The position of each ghost

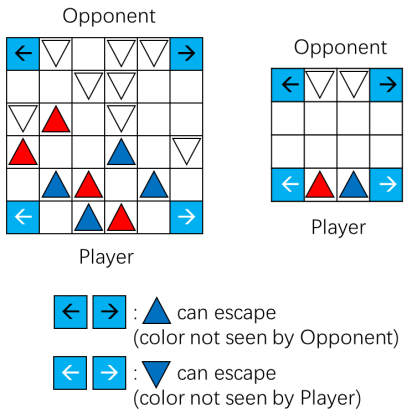


Fig. 1: The Sample Board Observation of Geister (left) and Mini Geister (right). The player can see the positions but not the color of the opponent ghosts.

is public, but its color is not visible to the opponent. Players can arrange their ghosts before the first turn. In each turn, a player moves one of ally ghosts one step vertically or horizontally inside the board without overlapping with ally ghosts. A ghost is captured, and its color is revealed to the public if the opponent's ghost enters its square. A blue ghost can *escape* from the designated corners in addition to ordinary moves. A player wins by one of the three conditions [5]: (1) capturing all the opponent's blue ghosts, (2) making all of ally red ghosts captured by the opponent, or (3) making one of ally blue ghosts escape. The left board in Fig. 1 shows a sample position. A player sees the placement of all ghosts and the colors of ally ghosts but does not those of the opponent's (shown in white). The designated corners to escape are at the opposite side of a player's initial position.

2) Mini Geister: As the full game of Geister has up to 10^{18} possible ghost positions, here we introduce a simplified version with a tractable size, yet it is still strategically challenging. Mini Geister is a game identical to the full game, except that it takes place on a 4×4 board and that each player has one blue ghost and one red ghost. Note that in this simplified game, a single capture by either player will decide the winner, so distinguishing the opponent's ghost color becomes extremely important. The right board of Fig. 1 shows a sample position of Mini Geister.

III. Related Research

A. Counterfactual Regret Minimization

Counterfactual Regret Minimization (CFR) was developed by Zinkevich et al. to compute approximate Nash equilibrium in imperfect information extensive games [1]. CFR is an iterative method that repeatedly updates estimates in a game tree. The algorithm keeps track of the cumulative counterfactual regret for each action a at each information set I and calculates the average strategy over all iterations. Counterfactual value $v_i(\sigma, I)$ presents a weighted average utility reachable from information set I:

$$v_i(\sigma, I) = \sum_{z \in Z_I} u_i(z) \pi^{\sigma}_{-i}(z[I]) \pi^{\sigma}(z[I], z), \tag{1}$$

where Z_I is the set of terminal histories reachable from I and $z[I]$ is such a history that is a prefix of z and contained in I. The counterfactual regret r_i^t (cumulative counterfactual regret R_i^T) presents a relative preference of action a on iteration t (up to iteration T), given strategy profile σ^t of iteration t as:

$$r_i^t(I, a) = v_i(\sigma^t_{I \to a}, I) - v_i(\sigma^t, I), \tag{2}$$

$$R_i^T = \sum_{t=1}^{T} r_i^t(I, a). \tag{3}$$

By the regret matching algorithm, the strategy of the next iteration is defined so that the probability of playing a at I is proportional to the positive cumulative regret:

$$\sigma_i^{T+1}(I, a) = \begin{cases} R_i^{T,+}(I, a)/Z & \text{if } Z > 0 \\ 1/|A(I)| & \text{otherwise,} \end{cases} \tag{4}$$

where $R_i^{T,+}(I, a) = \max(R_i^T(I, a), 0)$ and Z is normalizing term $\sum_{a \in A(I)} R_i^{T,+}(I, a)$. Then, the average strategy $\bar{\sigma}$ weighted by the reach probability π_i approaches ϵ-Nash Equilibrium as the number of iterations T increases:

$$\bar{\sigma}_i^T(I, a) = \frac{\sum_{t=1}^{T} \pi_i^{\sigma^t}(I) \sigma^t(I, a)}{\sum_{t=1}^{T} \pi_i^{\sigma^t}(I)}. \tag{5}$$

B. Monte Carlo CFR

Monte Carlo Counterfactual Regret Minimization (MCCFR) improves the efficiency of each iteration and empirically converges faster than CFR [6]. While the original CFR visits all information sets (the entire game tree) on each iteration, only a part of the tree is sampled in MCCFR. In outcome-sampling (OSCFR), a single terminal history is sampled, and the estimates in information sets along this history are appropriately updated so that counterfactual values have the same value as those in the original CFR in expectation. Exploration strategy and importance sampling are introduced for a *traverser*, a player to learn, which typically alters on each iteration. In external-sampling (ESCFR), a traverser samples all actions of its own, but a single action is sampled for other players. We incorporate ESCFR with our method to balance computational cost and sample efficiency.

C. Deep CFR

Deep Counterfactual Regret Minimization (Deep CFR) incorporated deep neural networks to replace tabular representation in the original CFR. The *policy* and *value* networks approximate the strategy and advantage (proportional to regret), respectively, for a given information set. To improve training efficiency, Deep CFR also incorporates ESCFR and linear weighting of the advantage and policy on iteration t by t [7]. Without relying on advanced domain knowledge, Deep CFR shows strong performance in large poker games

relative to domain-specific abstraction techniques [2]. Our method also incorporates policy and value networks but the learning scheme is entirely different for handling games with long histories.

D. DREAM

Deep Regret minimization with Advantage baselines and Model-free learning (DREAM) is designed to learn from the self-play as in model-free reinforcement learning. To that end, OSCFR is introduced with *baseline* networks to reduce the variance in sampling. DREAM samples only one action at each decision point, yet achieved the state-of-the-art performance among model-free methods in popular benchmark games and is even competitive with non-model-free algorithms [3]. Our method also incorporated baseline networks, however, the purpose is not only to reduce the variance but to truncate a self-play with a fixed number of steps.

E. ReBeL

ReBeL is a generalization of reinforcement learning to imperfect information games, where the strategy of an agent is improved by experiences in self-play [4]. Specifically, ReBeL repeatedly runs a depth-limited version of CFR in a sampled sub-tree experienced in self-play and trains its neural networks. The soundness of ReBeL is explained by the fact that an imperfect information game has an equivalent perfect information game with public belief states. We also incorporated belief states but in a different way.

IV. Proposed Methods

To learn a decent strategy in imperfect information games with long histories or loops, such as Geister, we propose a new method on top of CFR and its state-of-the-art extensions. Our method is composed for alleviating problems caused by histories with loops by using **Belief States with Approximation by Dirichlet Distributions** and **Depth-limited External Sampling for Board Games**, where the former is inspired by public belief states in ReBeL [4], and the latter is inspired by both the baseline networks in DREAM [3] and the depth-limited search in ReBel [4].

A. Belief States with Approximation by Dirichlet Distributions

Belief States with Approximation by Dirichlet Distributions is designed to abstract the information sets in a way that keeps an essential part of the history information. In many perfect information board games, computational costs are dramatically reduced by caching values of a board in transposition tables through ignoring past histories. In imperfect information games, it is not safe to ignore the entire history as the values are highly dependent on histories, but we can still incorporate abstraction. In Geister, the arrangement of the red and blue ghosts of the opponent is hidden from a player, while the locations of ghosts are public. Let denote *true state* be the board position including hidden information (i.e., ghosts with color), and a *board observation* be a player's view of it (i.e., without color of the opponent's ghosts). We define a *belief* as

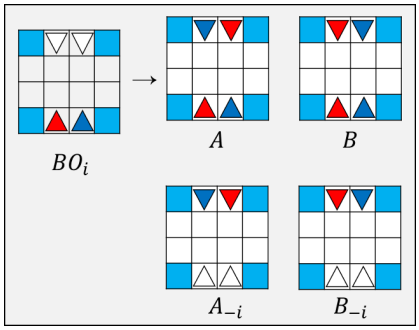


Fig. 2: Board observation of player i at the beginning (BO_i), its compatible *true states* (A and B), and their corresponding board observations from the opponent (A_{-i} and B_{-i})

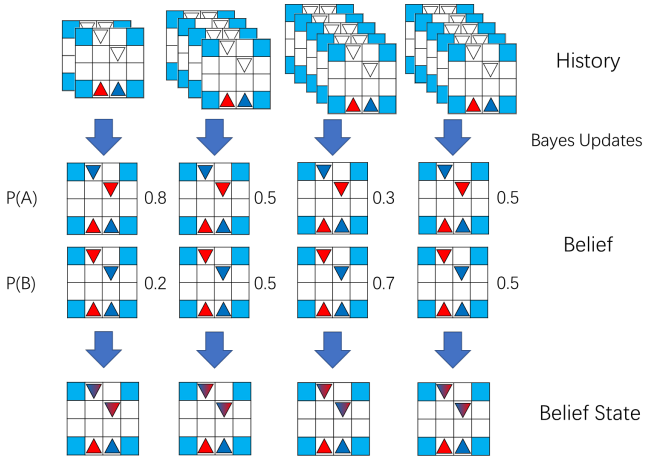


Fig. 3: Overview of belief states. The mixtures of blue and red illustrate the likelihood of the ghost being blue or red.

the probability distribution over true states that are compatible with the board observation of an agent. We define a *belief state* of a player as a pair of the board observation and the belief. Fig. 2 illustrates a board observation and true states on Mini Geister. Suppose the left (right) ghost of player i is red (blue). Because the color of the opponent's ghost is not visible, there are two possible true states; the blue ghost on the left (case A) or the red ghost is (case B). Note that the color of the right ghost is immediately determined by that of the left ghost in Mini Geister. A belief state of player i is such a set of histories that shares board observation and belief over the true states. Fig. 3 illustrates belief states in Mini Geister. While there are many histories compatible with a board observation (top), some share the same probability in belief (center). So, we group histories as group states by their belief (bottom). Our important assumption is that the strategies are equivalent or at least similar among the information sets included in each belief state near Nash equilibrium.

Belief is refined along with the game progress, as a player observes more moves by the opponent. The belief after a move is updated as the posterior distribution following Bayes' rule, given a strategy profile. To effectively present the

prior and posterior distribution of a belief, we approximate a belief by Dirichlet (or Beta if the number of true states is two) distributions keeping the expected value unchanged. Suppose an agent is player 1 for simplicity (the other case is straightforward by symmetry). Let α_1 and β_1 be the hyper parameters of the Beta distribution to approximate the belief of player 1. At the beginning of a game, they are initialized appropriately. Then, the belief is presented as the expected probabilities of the Beta distribution to represent the corresponding belief state $BS_1(BO_1, \alpha_1, \beta_1)$, such that $\Pr(A|BS_1) = \alpha_1/(\alpha_1 + \beta_1), \Pr(B|BS_1) = 1 - \Pr(A|BS_1)$. After the opponent makes a move m_2, we update the belief according to Bayes rule as Eq. (6). For this update, we need the opponent's policies $\Pr(m_2|BS_2(\cdots))$. We assume that the opponent uses the same policy and the same way to update beliefs as our agents. This assumption does not hold in general but is still reasonable when both players are near Nash equilibrium. Therefore, we maintain two belief states BS_1, BS_2, symmetrically. The latter consists of $\mathrm{Beta}(\alpha_2, \beta_2)$ from the viewpoint of the opponent that is updated when our agent makes a move. Let $\Pr(A|\alpha_1, \beta_1)$ be the prior probability governed by the beta distribution $\mathrm{Beta}(\alpha_1, \beta_1)$, and $\Pr(B|\alpha_1, \beta_1) = 1 - \Pr(A|\alpha_1, \beta_1)$. Denote A_2 and B_2 as the board observation of the opponent whose private information is the same as that in A and B, respectively, as shown in Fig. 2. Let $\Pr(m_2|BS_2(A_2, \alpha_2, \beta_2))$ and $\Pr(m_2|BS_2(B_2, \alpha_2, \beta_2))$ represent the probability of playing move m_2 if our agent were the opponent, i.e., the probability of playing m_2 if our agent is in the belief state BS_2 where the board observation is A_2 or B_2 with the belief $\mathrm{Beta}(\alpha_2, \beta_2)$. $\Pr(m_2|BS_1, \alpha_2, \beta_2)$ means the likelihood of playing the move under the current belief state with belief $\mathrm{Beta}(\alpha_2, \beta_2)$. $\Pr(A|m_2, \alpha_1, \beta_1)$ and $\Pr(B|m_2, \alpha_1, \beta_1)$ are the posterior probabilities after observing the move, whose distribution is defined by integration.

$$\begin{aligned}\Pr(m_2|BS_i, \alpha_2, \beta_2) = &\Pr(A|\alpha_1, \beta_1)\Pr(m_2|BS_2(A_2, \alpha_2, \beta_2)) \\ &+ \Pr(B|\alpha_i, \beta_i)\Pr(m_2|BS_2(B_2, \alpha_2, \beta_2)) \\ \Pr(A|m_2, \alpha_1, \beta_1) \propto &\Pr(A|\alpha_1, \beta_1)\Pr(m_2|BS_1, \alpha_2, \beta_2) \\ \Pr(B|m_2, \alpha_1, \beta_1) \propto &\Pr(B|\alpha_1, \beta_1)\Pr(m_2|BS_1, \alpha_2, \beta_2)\end{aligned} \tag{6}$$

Although the resulting posterior distribution no longer follows a Beta distribution, we approximate it by a Beta distribution and define the hyper parameters using Eq. (7) to keep the expected value unchanged from that of the true posterior distribution. This approximation enables simple and unified updates in each move, while the naive procedure becomes much more complicated as the number of moves increases.

$$\begin{aligned}\alpha_1' = &\alpha_1 + \frac{A'}{A' + B'}, \quad \text{and} \quad \beta_1' = \beta_1 + \frac{B'}{A' + B'} \\ \text{where } A' = &\alpha_1 \cdot \Pr(m_2|BS_2(A_2, \alpha_2, \beta_2)) \\ B' = &\beta_1 \cdot \Pr(m_2|BS_2(B_2, \alpha_2, \beta_2))\end{aligned} \tag{7}$$

The initial values can be chosen by the model of the opponent if available or initialized equally as $\alpha_1 = \beta_1$ otherwise. As the most natural choice we believe, we initialize $\alpha = \beta = 0.5$ and keep the sum of them to the move number (starting from one), by increasing the sum by 1.0 after each move. The magnitude of initial values affects the stability of updated values. Note that the hyper parameters for a Dirichlet distribution are updated similarly for beliefs where the number of the true states per each observation is greater than two. We used Beta distribution in this example for simplicity where the true states for a given observation is always two. The main target of our methods is Geister, but we suppose that it is also effective in many other imperfect information board games with long or infinite histories but with a limited number of hidden arrangements.

B. Depth-limited External Sampling for Board Games

While the memory consumption is reduced with the proposed belief states, it is still needed to handle computational costs caused by long histories. A straightforward approach would be to track a single path with outcome sampling methods and to forcibly terminate the game after reaching a threshold of the game length. Unfortunately, it was known that the learned strategies by such an approach are not sufficient in that agents were weaker than heuristic players though better than random players [8]–[10]. We also show our comparative experiments in Sect. V-B4.

Here we propose Depth-limited External Sampling for Board Games, to effectively integrate our belief states and the state-of-the-art techniques. To handle games with long histories, our method expands a partial game tree with a limited depth where the root of trees is sampled from self-play as in reinforcement learning. To improve the learning efficiency, each node in our partial game tree represents a belief state instead of an information set. We perform external sampling on our partial game trees consisting of belief states. To approximate the expected utilities in leaf states that are not the terminal state of a game, we introduce a function approximator called *baseline networks*, typically implemented with neural networks. Similarly, we introduced value and policy networks to approximate regrets and average strategies for scalability. In short, we sample a given belief state and conduct external sampling on a partial game tree with a preset *depth limitation* to collect training data for our networks. Fig. 4 sketches an overview of our method, and the pseudo-code of the algorithm is shown in Algorithms 1, 2, and 3. In Algorithm 1, we first initialize the buffers for data storage and the networks for training and predicting. Each iteration t contains a several number of traversals $N_{\text{traversals}}$ to sample extra data for training. Iteration number t is used in linear weighting as in study [7]. A typical iteration (Algorithm 2) consists of self-play (left in Fig. 4) and learning (right), similarly to AlphaZero. Belief states are sampled among game records yielded by self-play with the current accumulative strategies (approximated by regret matching with policy networks). In the learning, for each sampled belief state, a subtree rooted at the state is examined by a depth limited variant of ESCFR (Algorithm 3), which is conducted to improve the baseline, value, and policy networks: At each time a leaf is visited (L. 4), the utility is approximated by the baseline networks

Algorithm 1 Depth-limited External Sampling for Board Games

Function TRAIN($N_{\text{iter}}, N_{\text{traversals}}$)
  Initialize value buffer $B_v = \emptyset$ and policy buffer $B_p = \emptyset$.
  Initialize baseline buffer $B_b[i] = \emptyset, i = 1, 2$.
  Initialize value network V_v, policy network V_p and baseline networks $V_b[i], i = 1, 2$ randomly.
  for $t = 1, 2, \ldots, N_{\text{iter}}$ **do**
    for $k = 1, 2, \ldots, N_{\text{traversals}}$ **do**
      ITERATION(t)
  return V_p

Algorithm 2 Iteration

Function ITERATION(t)
  Self-play using strategies provided by V_p with ϵ on-policy to collect N belief states
  for $R = BS_1, BS_2, \ldots, BS_N$ **do**
    for $p = 1, 2$ **do**
      TRAVERSE($R, p, 0, t$)
  Train all the networks using the corresponding buffers.

Algorithm 3 Traverse

Function TRAVERSE(BS, p, d, t)
  if $BS \in Z$ **then** ▷ is terminal
    return $u_p(BS)$ ▷ player p's utility of terminal BS
  if $d \geq d_{\text{Limit}}$ **then**
    return $V_b[p](BS)$ ▷ predicted baseline value of BS
  $\sigma(BS) \leftarrow$ REGRETMATCHING($V_v(BS)$)
  if $P(BS) = p$ **then**
    for $a \in A(BS)$ **do** ▷ for each legal action in BS
      $v(a) \leftarrow 0$ ▷ utility for each action
      $R(a) \leftarrow 0$ ▷ regret for each action
    $\bar{v} \leftarrow 0$
    for $a \in A(BS)$ **do**
      $v(a) \leftarrow$ TRAVERSE($BS \cdot a, p, d + 1, t$)
      $\bar{v} \leftarrow \bar{v} + \sigma(BS, a) \cdot v(a)$
    for $a \in A(BS)$ **do**
      $R(a) \leftarrow v(a) - \bar{v}$
    Add (BS, R, t) to value buffer B_v.
    Add $(BS, \bar{v})$ to baseline buffer $B_b[p]$.
    return $\bar{v}$
  else
    Sample $a \sim \sigma(BS)$
    Add $(BS, \sigma(BS), t)$ to policy buffer B_p.
    return TRAVERSE($BS \cdot a, p, d + 1, t$)

$V_b[p](BS)$. Leaf values are used to calculate regret values for their parent or ancestor states through recursion (L. 12). The expected values of non-leaf states, regret values, and strategies are stored in B_b, B_v, and B_p to train the baseline, value, and policy networks, respectively. We chose three as the depth limit in our partial game trees for efficiency, which is also justified by the fact that a loop in board games typically needs the length of four.

V. EXPERIMENTS

To clearly evaluate the performance, we implemented our proposed methods on Mini Geister using tabular representation. In our Mini Geister, the initial arrangements of the ghosts are generated randomly without the players' actions. Note that our methods are also capable if the players can arrange their ghosts by themselves. For tabular representation, all the networks are replaced by key-value tables. The belief states contain real number probabilities which are hard to handle by tables, so they are quantized to one-fourths and then converted to strings as the keys for tables. The regret and policy tables store accumulated regret and policy values for each quantized belief state, respectively.

A. Evaluation

In our evaluation environment, the initial arrangements of the ghosts are also randomly generated. All the players, including the random one, adopted an enhancement that they always play an escape move to win the game if they can. The game will be terminated with a draw if the game length reaches the preset limitation. In Mini Geister experiments, the limitation is 30 moves if not further mentioned. Our trained agents are evaluated by playing against the random and a heuristic player. We count the win rate and the winning moves to analyze our agents' performance. The winning moves can be classified into the following categories according to the game rule: loss by taken blue, loss by take red, loss by escape, draw, win by taken red, win by take blue, and win by escape. The ratio of each category shows us the preference of the agents' strategies.

The heuristic player acts based on simple evaluation functions without search, whose features are; (1) Attack: encouraging the agent to move ally blue ghosts towards opponent corners, (2) Avoid: discouraging the agent from taking unknown opponent ghosts and encouraging the agent to reduce the number of neighboring opponent ghosts of ally blue ghosts, and (3) Defense: encouraging the agent to move ally ghosts towards ally corners. Empirically, the heuristic player is good at escaping and can win the random player at a win rate of about 70% in Mini Geister (30 moves limit) and over 80% in the full game of Geister (300 moves limit). Fig. 5 shows the details.

We observe that the sampled belief states are often poorly distributed even with ϵ on-policy. To increase sampling efficiency, we also introduce a new hyperparameter ϵ_{belief} to regenerate the parameters of the belief with probability ϵ_{belief}. When the belief is regenerated, the true state of the belief state is sampled according to the new belief; otherwise, the true state in the belief state is retained. In our experiments, $\epsilon = 0.25$ and $\epsilon_{\text{belief}} = 0.125$. For experiments without the belief state, ϵ_{belief} is not used. The utilities for terminal histories are $+1, 0$, and -1 for win, draw, and loss, respectively. Under other situations, the utilities are considered zero if needed. In our preliminary experiments, the learning of baseline values for non-traversal players was not stable. Therefore, we only learned baseline values for a traverser and kept those baseline values all zeros for a non traverser. The baseline tables update the values by exponential moving average ($\alpha = 0.5$), which does the update as $b^k = (1 - \alpha)b^{k-1} + \alpha b^*$, where the k-

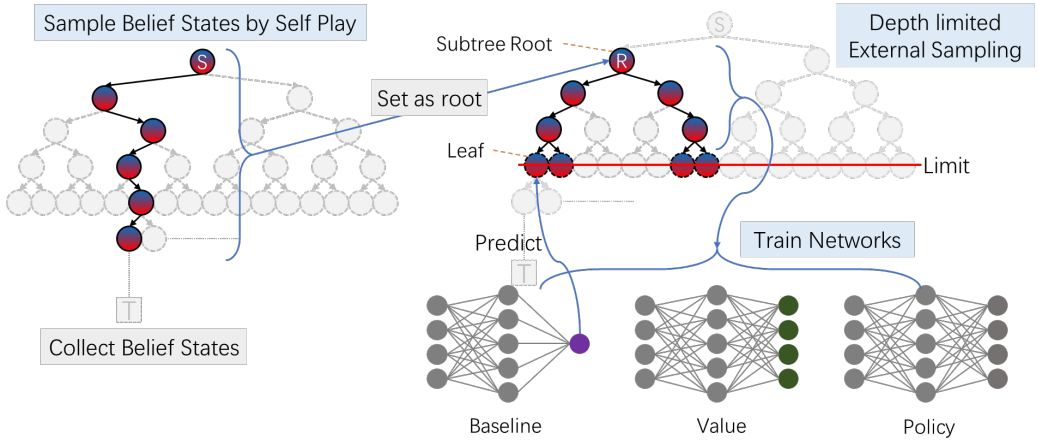


Fig. 4: Overview of Depth-limited External Sampling for Board Games. Circles colored in blue and red represent belief states.

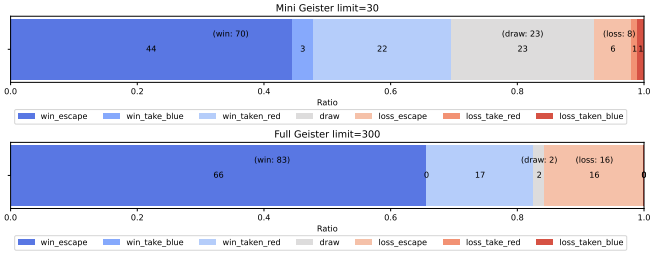
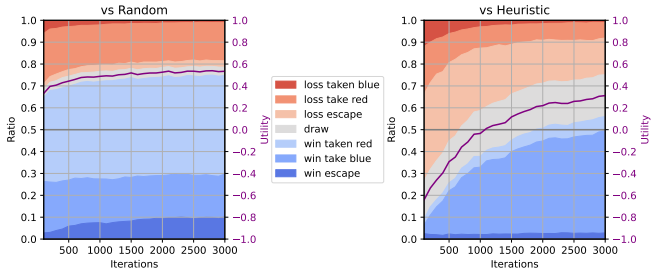


Fig. 5: Self-play results between Heuristic Player v.s. Random Player. Ratio is measured over 10 000 games and shown in percentage rounded to integer.



(a) against the Random player (b) against the Heuristic player

Fig. 6: Learning curves of the proposed agents (Tabular), evaluated by winning ratio every 100 iterations.

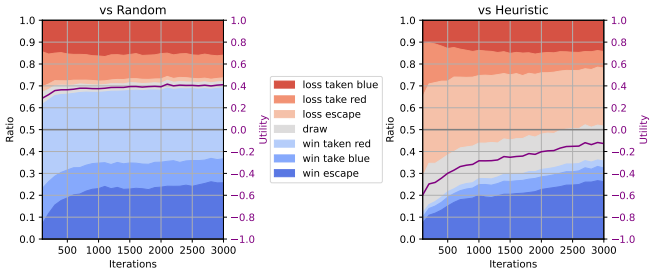
th update of value b is b^* and $b^0 = 0$. Our experiments are run on single-chip computers without clustering, each with an AMD® Ryzen 9 or Threadripper CPU. All the source code is written in Python 3, and self-play and Traverse run in parallel with the standard multiprocessing library.

B. Results and Analysis

1) Agents with Belief States: We execute our algorithm for 3 000 iterations t with two traversals k per iteration. The results of our trained agents playing against the random and the heuristic player are shown in Fig 6a and 6b. Data show the average of four independent training from scratch.

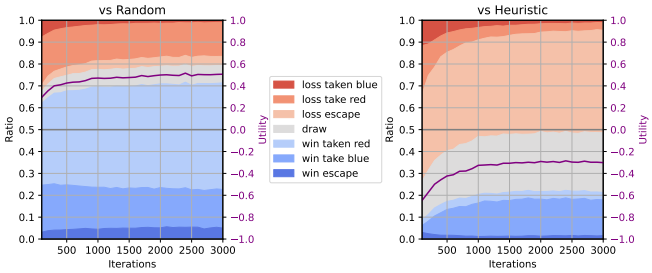
We can observe from Fig. 6a that our agents learned reasonable strategies for the game. The winning ratio steadily increases as the area of the blue region does along with iterations. Although our belief in the opponent's ghost colors does not make sense when playing against a random opponent, our agents show good performance without exerting the full strength of belief states. During the training, our agents gradually grasp a way to escape from the corners while protecting the ally blue ghost from being captured. Although the total win rate grows slower after 1 500 iterations, the increasing ratio of winning by escape and decreasing ratio of capturing enemy ghosts indicate that the learning is still progressing. We notice that our agents often capture enemy red ghosts by mistake due to the randomness of the opponent which confuses our belief. However, our agents gradually acquires the skill to win by escaping or letting ally red be captured to decrease uncertain capturing. Finally, our agents obtained a win rate of nearly 75% against the random player.

The learning curves in Fig. 6b clearly show the rapid learning progress of our agents. The increasing ratio of capturing enemy blue and decreasing ratio of capturing enemy red by mistake show that our agents quickly master how to identify enemy blue ghosts and capture them to win the game. The heuristic player is good at making their blue ghosts escape, whose movement is caught by our agents using our belief states. It can be inferred that our agents gain an understanding of the strategy that the heuristic player uses as our agents predict the heuristic player's moves with high accuracy. Our agents are also getting better at preventing enemy escapes, obtaining a decreasing ratio of losses by escape. Similar to the results of playing against random players, our agents also learn to protect ally blue ghosts from being captured and let ally red ghosts be captured. As the heuristic player is designed to be discouraged from uncertain captures, we think our agents also learn how to deceive the opponent by moving a red ghost towards enemy corners. We also observe that the training progress decelerates after about 2 000 iterations. We find that



(a) against the Random player (b) against the Heuristic player

Fig. 7: Ablation: Without belief states, evaluated by winning ratio every 100 iterations.



(a) against the Random player (b) against the Heuristic player

Fig. 8: Ablation without baseline, evaluated by winning ratio every 100 iterations.

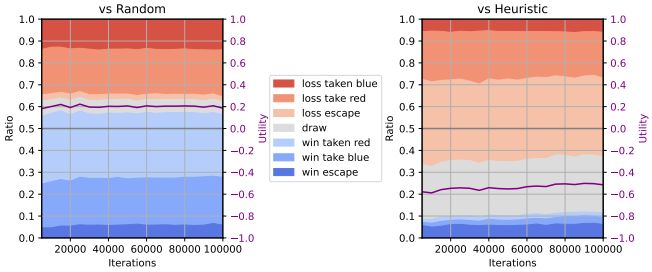
the ratio of draw is relatively high during the whole training, indicating both players avoid immediate loss and reach the limitation. If we lengthen the game length limitation during the evaluation, we expect our agents to win most of the draw games. Finally, our agents earned a win rate of over 55% against the heuristic player. After training for 3 000 iterations, although the growing speed drops significantly, there is still an increasing trend in win rate. We believe our agent will achieve even better performance if we continue training.

Note that our agent is not specialized for a single opponent but uses the same strategy against random or heuristic players. From the results of playing against these different players, we argue that our agents have acquired decent skills to play Mini Geister using tabular representation approaching Nash equilibrium.

2) Ablation study 1: Agents Without Belief States: To evaluate the effectiveness of our belief states, we also implemented our method without Belief States, where each information set consists of a current board observation ignoring any history. It means that the same strategy is used for each board observation without inspecting the arrangement of the opponent's ghosts. We trained the agents under the same conditions as those with belief states, and show the results in Fig 7a and 7b. The curves here and in the following experiments are the average of three independent training from scratch.

The ablated agent played similarly against the random player but poorly against the heuristic player. Fig. 7a shows that the agent reached a win rate of about 70% against the random player, but with a very different strategy. As the ablated agents have no information about the enemy ghost colors, they focus on how to escape while reducing capturing. However, the agents fail to protect ally the blue ghosts from being captured. Fig. 7b shows that although the training gradually proceeds, the ablated agents fail to compete with the heuristic player, with a win rate of only about 35%. It is natural that the belief state is more effective against the heuristic player than against the random one because moves by the heuristic player are more meaningful and tend to contain information about the arrangement of ghost colors.

3) Ablation study 2: Agents without Baseline: To evaluate the effectiveness of the baseline network, we also trained



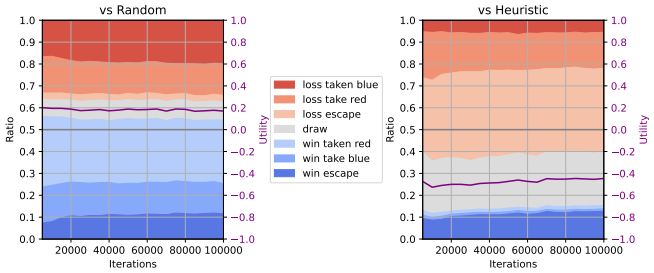
(a) against the Random player (b) against the Heuristic player

Fig. 9: Learning curves of DREAM agents, evaluated by winning ratio every 5 000 iterations.

agents without the baseline network by setting its output to zero. The results are shown in Fig 8a and 8b. We can observe from the figures that our agents are able to learn to identify enemy piece colors, but the resulting strategies are relatively weak, with a win rate of only about 20% against the heuristic player.

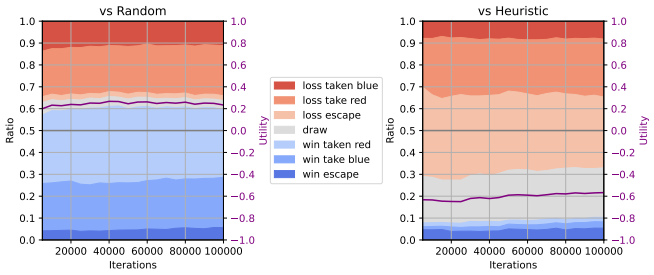
4) DREAM and OSCFR: To demonstrate the effectiveness of our proposed depth-limited methods, we implemented other CFR variants with and without belief states and compared the performance of the trained agents. We implemented DREAM and outcome-sampling MCCFR (OSCFR) in Geister following the previous studies [8]–[10], but in a tabular form. For DREAM, we used accumulative strategies instead of sampling from value networks as we are able to store the exact values using the tabular representation. The results are shown in Figs 9a, 9b, 10a, 10b, 11a, 11b, 12a and 12b. These figures clearly show that all of these agents learn extremely slowly and make little progress in learning. The learning efficiency is so limited that applying belief states does not show apparent improvement in performance. In contrast, the agents without belief states appear to have a superficial understanding of escaping. The resulting agents are barely superior to the random player, with a win rate of only about 10% against the heuristic player. Therefore, the depth-limited method is crucial in learning.

In addition to comparison of win rates against random or heuristic players, we also conducted a direct match between



(a) against the Random player (b) against the Heuristic player

Fig. 10: Learning curves of DREAM agent without belief states, evaluated by winning ratio every 5 000 iterations.



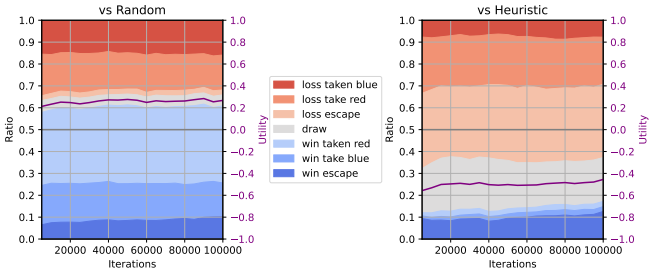
(a) against the Random player (b) against the Heuristic player

Fig. 11: Learning curves of OSCFR agents, evaluated by winning ratio every 5 000 iterations.

the agents with and without belief states. We choose the execution of the highest win rate against the heuristic player for agents without belief states and the lowest for the agent with belief states. The result of 10 000 games is shown in Fig 13. The result shows that our agents with belief states have the ability to recognize enemy's blue ghosts with high accuracy and protect ally blue ghosts from being captured. Although the skill to escape from the corners is inferior to that of the agent without belief states, they have a significantly better performance in identifying blue ghosts. As the two types of agents are trained in similar settings, we propose that the application of belief states will enable the agents to acquire more knowledge in playing the game.

VI. Conclusions and Future Works

In this study, we proposed Belief States with Approximation by Dirichlet Distributions and Depth-limited External Sampling for Board Games to abstract the game and deal with loops in board games. Our careful combination of the methods and ideas has enabled CFR variants to be applied to board games with long histories and/or loops. We applied our methods to the game Mini Geister to train agents and evaluated them by playing against the random player and the heuristic player. Although the tabular representation we use has a limited precision in data processing, the experimental results showed that our proposed method learned important strategies in this game, including escaping from the corners, identifying enemy blue ghosts, and avoiding ally blue ghosts



(a) against the Random player (b) against the Heuristic player

Fig. 12: Learning curves of OSCFR agents, without belief states, evaluated by winning ratio every 5 000 iterations.

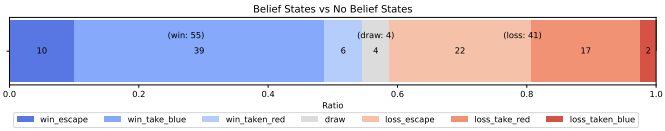


Fig. 13: Self-play results between proposed agent (Belief States, worst agent out of four executions) v.s. ablated agent (Without Belief States, best one out of three executions). Ratio is measured over 10 000 games and shown in percentage, rounded to integer.

being captured. Our proposed method showed a significantly better performance than the baseline and therefore showed the ability to learn reasonable strategies with tabular representation. We are working on the neural networks representation to extend our methods to show the scalability in larger games.

References

[1] M. Zinkevich, M. Johanson, M. Bowling, and C. Piccione, "Regret minimization in games with incomplete information," in *Advances in Neural Information Processing Systems*, 2008, pp. 1729–1736.

[2] N. Brown, A. Lerer, S. Gross, and T. Sandholm, "Deep counterfactual regret minimization," in *International Conference on Machine Learning*, 2019, pp. 793–802.

[3] E. Steinberger, A. Lerer, and N. Brown, "Dream: Deep regret minimization with advantage baselines and model-free learning," 2020.

[4] N. Brown, A. Bakhtin, A. Lerer, and Q. Gong, "Combining deep reinforcement learning and search for imperfect-information games," in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, Eds., vol. 33. Curran Associates, Inc., 2020, pp. 17 057–17 069.

[5] BoardGameGeek, "Phantoms vs phantoms," https://www.boardgamegeek.com/boardgame/2290/phantoms-vs-phantoms.

[6] M. Lanctot, K. Waugh, M. Zinkevich, and M. Bowling, "Monte carlo sampling for regret minimization in extensive games," in *Advances in Neural Information Processing Systems 22*, 2009, pp. 1078–1086.

[7] N. Brown and T. Sandholm, "Solving imperfect-information games via discounted regret minimization," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 1829–1836.

[8] 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)*, 2019, pp. 1–6.

[9] ——, "Utilizing history information in acquiring strategies for board game geister by deep counterfactual regret minimization," in *Game Programming Workshop 2019 Proceedings*, vol. 2019, nov 2019, pp. 20–27.

[10] ——, "Application of dream to the board game geister," in *Game Programming Workshop 2020 Proceedings*, vol. 2020, nov 2020, pp. 111–117.