修士論文

完全情報ゲームの評価値を用いた二人零和不完全情報ゲーム『ガイスター』における混合戦略 AI の研究

1910071 川上 直人

主指導教員 池田 心

北陸先端科学技術大学院大学
先端科学技術研究科
（情報科学）

令和 3 年 2 月

Abstract

In recent years, artificial intelligence (AI) players of perfect information games such as Go and Shogi have exceeded top human players' strength. In 2015, AlphaGo was proposed, which combines Monte-Carlo tree search and deep learning techniques. AlphaGo defeated a professional Go player without handicaps for the first time and attracted public attention.

Not only great achievements have been made in perfect information games, AI for imperfect information games such as poker and mahjong has also attracted attention and made great strides. Players in imperfect information games cannot fully observe the game states, making it not trivial to search the game trees. One group of approaches is counterfactual regret minimization (CFR), which was proposed in 2008 and has achieved great results in some games. A famous example is heads-up limit hold'em poker, whose Nash equilibrium has been approximated. Another group of approaches is deep reinforcement learning, where a mahjong AI Suphx has achieved top human players' levels in 2019. Meanwhile, for other imperfect information games such as Geister, AI players' strength is still limited.

Geister is a two-player, zero-sum, deterministic, and imperfect information game. Players have blue and red pieces and can capture the opponents' pieces, similar to chess. However, a big difference from chess is that the colors are hidden to the opponents until pieces are captured, which is an interesting point of the game. It is important to guess the colors of the opponents' pieces from the past movements, or conversely to move pieces in ways that make the opponents difficult to guess.

Research on Geister AI is still under development. For example, a purple-piece-AI method won an AI competition. However, there is some regularity in the movement, and the pieces' colors are easily predicted. Specifically, most of the pieces adjacent to the opponents' pieces are red ones, and if this is known, the purple-piece-AI is easy to defeat. Therefore, it is necessary to move the pieces stochastically. Such stochastic behavior has been analyzed in 3×2 Geister with a limited move number, but the method is challenging to handle larger board sizes since the whole search tree is expanded.

This research aims to handle Geister on a larger board size (4×4) and produce stochastic behaviors. For this purpose, we do not expand the whole search tree to calculate the expected win rates. Instead, we employ a heuristic function to give evaluation values. There are many variations in how to give evaluation values. In this research, we first propose a method that evaluates leaf nodes with non-terminal states as draws. We then propose another method that generates the win/loss database of a perfect information version of Geister and evaluates leaf nodes by the corresponding perfect information states.

To evaluate the performance of the proposed methods, we conducted battle ex-

periments employing four types of benchmark AI players based on purple-piece-AI. First, in the games between benchmark players, it was confirmed that each player had some opponent(s) that it was difficult to win, where the win rates were under 15%. Next, the proposed methods battled against the four benchmark players. Even for the opponents most difficult to win, the win rates were 19% and 24%. Although it is hard to say that the proposed methods are overall stronger than the purple-piece-AI, we believe that stochastic moves have succeeded in reducing the risks of being exploited.

概要

近年，囲碁や将棋など完全情報ゲームの人工知能（AI) は，トッププレイヤの実力を上回るなど大きな成果を挙げている．2015 年には，モンテカルロ木探索と深層学習の技術を組み合わせた囲碁 AI『AlphaGo』が，プロ囲碁棋士にハンディキャップなしで初めて勝利し，世間から注目を集めた．

完全情報ゲームが大きな成果を挙げている一方，ポーカーや麻雀といった不完全情報ゲームの AI も注目され，大きな進歩を遂げている．不完全情報ゲームは，ゲームの状態を完全には観測できないため，ゲーム木探索をおこなうのが困難なことが課題である．アプローチの 1 つは，2008 年に提案され，一部のゲームで大きな成果を挙げた Counterfactual Regret minimization (CFR) である．有名な例として，2015 年にはヘッズアップリミットホールデムポーカーにおいてナッシュ均衡に近い戦略が計算された．もう 1 つのアプローチとして，深層強化学習がある．2019 年には麻雀 AI『Suphx』が人間トップレベルの性能を達成した．一方，『ガイスター』などまだ十分強いコンピュータプレイヤが報告されていないゲームもある．

ガイスターは青駒，赤駒をチェスのように取り合う，二人零和確定不完全情報ゲームである．このゲームの面白い点は，まだ取っていない相手駒の色を観測できない点であり，相手駒の色をそれまでの動きから推測したり，逆に推測されにくい動きをすることが重要である．

ガイスター AI の研究は発展途上である．例えば紫駒 AI と呼ばれる手法は，2020 年の AI 大会で優勝しているが，駒の動きに法則性があり，行動から状態を推定されやすい課題がある．具体的には，相手駒に隣接させる駒はほとんど赤駒であり，これを知られてしまうと簡単に負けてしまう．したがって，駒の動きを確率的におこなう必要がある．短い手数制限を加えた 3×2 盤のガイスターでは，確率的行動が分析されているが，ガイスターを表現するゲーム木を終局まで展開しているため，それ以上大きなサイズの問題を扱うことが難しい．

そこで本研究では，より大きなサイズのガイスター（4×4 盤）でも確率的行動をおこなうために，ガイスターを表現するゲーム木を何手か先まで展開したら期待勝率の代わりに何らかの評価値を与える方法を提案する．評価値の与え方には様々なバリエーションが考えられる．本研究では，まず，未決着なリーフノードを引き分けと評価する方法を提案する。次に，完全情報なガイスターの勝敗データベースを生成し，リーフノードが持つ完全情報状態を評価する方法を提案する．

本研究では，提案手法の性能を評価するため，紫駒 AI とそれに付け込む AI を含む 4 種類のベンチマーク AI との対戦実験をおこなった．まず，ベンチマーク AI 同士の対戦では，どの AI についても勝率 15%を下回る苦手な相手が存在することを確認した．続いて，提案手法 2 種類とベンチマーク AI との対戦をおこなうと，最も苦手な相手に対しても 19%，24%の勝率となることが確認できた．残念ながら紫駒 AI に比べて総合的に強くなったとは言い難いが，確率的な着手によって，付け込まれる余地を減らすことには成功したと考えている．

目 次

図 目 次

表 目 次

第1章　はじめに

近年，人工知能（AI）の進歩により，自動車の運転や翻訳といった今まで人間がおこなってきたタスクの自動化が進み，AI は生活に欠かせない存在となっている．AI の研究対象として注目されている分野の 1 つが「ゲーム」である．ゲームはルールが明確に定められており，勝敗などにより AI の性能評価をおこないやすい．特に，囲碁や将棋といったゲームでは，熟練した人間プレイヤがいるため，トッププレイヤの実力を上回る AI を実現するなどの目標を設定できる．

ゲームにおいて，状態を完全に観測できるものを完全情報ゲームという．完全情報ゲームでは，MinMax 法，$\alpha\beta$ 法 [1]，モンテカルロ木探索 [2][3] など，ゲーム木探索による手法が提案されている．また近年では，熟練者の試合や自己対戦の棋譜を用いた深層学習などにより，複雑な局面評価関数を自動生成できるようになり，将棋や囲碁において，人間トッププレイヤの実力を大きく上回る AI が出現した．特に，Google DeepMind によって開発された囲碁 AI『Alpha Go[5]』は，2015 年 10 月に人間のプロ囲碁棋士をハンディキャップなしで破る快挙を成し遂げた．Alpha Go では，畳み込みニューラルネットワーク（CNN) を用いて状態評価値を出力するバリューネットワーク，行動評価値を出力するポリシーネットワークを表現し，モンテカルロ木探索を組み合わせた深層学習をおこなうことで，人智を超える戦略を獲得した．さらに 2017 年 10 月には，過去の試合データを使わず，自己対局のみで成長する囲碁 AI『AlphaGo Zero[6]』が発表され，2016 年 3 月に囲碁トッププレイヤのイ・セドルに勝利したプログラム『AlphaGo Lee』に対し，わずか 3 日で勝利（100 勝 0 敗）した．

次の課題として不完全情報ゲームが挙げられる．不完全情報ゲームは隠された情報のあるゲームである．例として，ババ抜き，ポーカー，麻雀などが挙げられる．不完全情報ゲームの研究は，未知の情報がある中で最良な意思決定をおこなうタスクへの応用が期待できる．例えば，株取り引きにおいて，株価・他の参加者の行動を推測し，投資をおこなう方法を AI が提案できるかもしれない．あるいは，ロボットサッカーなどの AI スポーツにおいて、観測外で相手が取る行動を推測しつつ、味方チームを勝利へ導くための適切な行動を選択できるかもしれない．不完全情報ゲームでは，ある行動を取ったときにゲームがどのように進行するかが確定しないため，完全情報ゲームのようにゲーム木探索をおこなうことは難しい．また、敵対するプレイヤがいる場合，洗練されたプレイヤ同士ではブラフなどの確率的な意思決定を求められることがある．

不完全情報ゲームにおいて，ポーカー，麻雀では人間トッププレイヤに匹敵す

る AI が研究されている．2015 年には 2 人対戦の Heads-up limit hold'em porker において，Counterfactual Regret minimization (CFR)[7][8] という手法により，ナッシュ均衡に近い戦略が計算された [9]．2019 年には 6 人対戦のポーカーにおいて，Facebook とカーネギーメロン大学（CMU）が開発した AI がトッププロに勝利した [10]．また，同年には Microsoft 社が開発した麻雀 AI『Suphx [11]』が天鳳十段を達成し，人間上位プレイヤの実力に達した．このように不完全情報ゲームにおいても大きな成果を挙げているものはある一方，まだ人間に勝てていない不完全情報ゲームもある．例えば,『ガイスター[1]』や軍人将棋といった，チェスのように駒を交互に動かす不完全情報ゲームでは大きな成果が挙がっていない．

ガイスターは 6×6 の盤と 2 種の駒を用いて遊ぶ，二人零和確定不完全情報ゲームである．各プレイヤは赤駒，青駒をそれぞれ 4 個ずつ持ち，盤の指定領域に自由に初期配置をおこない，交互着手で駒を移動し，取り合い，勝利を目指す．ガイスターの大きな特徴として，まだ取っていない対戦相手の駒の種類を観測できないことが挙げられ，相手の駒の種類をそれまでの動きなどから推測する必要がある．実際のゲームでは，駒色は駒の背面に小さな印として書かれており，これを互いに自分だけ見えるようにして戦う．この設定により人間同士の対決では，ブラフなどの心理戦を楽しむことができる．

ガイスターの先行研究として，モンテカルロ法 [13]，Q 学習と必勝手探索 [14]，ルールベースによる駒推定 [15]，MinMax 法 [16]，駒推定と深層学習 [17] などヒューリスティックに基づくもの，CFR[18]，Deep CFR[19] を用いたナッシュ均衡の理論に基づくもの，キーパー戦略 [20] と呼ばれる限定的なルールベースが報告されているが，人間に勝てるほどの AI は報告されていない．特に，MinMax 法 [16] による手法を用いた AI は，GAT2020 杯ガイスター AI 大会にプログラム名「Naotti2020-3」で参加し，8 チーム中 1 位を獲得しているが[2]，初期配置以外の確率的行動が無く，対策されやすい．具体的には，相手駒に赤を隣接させやすく，青を隣接させにくいといった特徴があり，隣接させてきた駒を赤と推定されるなどすると弱いと考えられる．また，文献 [18] では，3×2 盤のミニガイスターにおいて，CFR を用いて混合戦略を計算しているが，計算量が大きく，短い手数制限を加えている．混合戦略とは，確率的に意思決定をおこなう戦略のことである．6×6 盤のガイスターにおける混合戦略については，CFR を深層学習に適用した Deep CFR を用いた研究が報告されているが [19]，現実的な時間では学習が十分な性能まで収束せず，既存のプログラムよりはるかに弱い性能となった．

本研究の目的は，不完全情報ゲーム『ガイスター』において，ブラフなどの確率的行動を自動で獲得する仕組みを作ることである．そのためには，ゲームの手数が 50，100 と長くなっても，現実的な時間で計算できる必要がある．本研究で

[1] メビウスゲームズ：ガイスター，ボードゲーム・カードゲームの店 メビウス ゲームズ，入手先〈http://www.mobius-games.co.jp/Gester.htm〉（参照 2021-01-15）.

[2] ガイスター AI 大会
入手先〈http://www2.matsue-ct.ac.jp/home/hashimoto/geister/GAT/2020/〉（参照 2021-01-15）

は，特定の相手にのみ勝てるAIではなく，どのような相手にも搾取されにくいものを目指す．

2章では不完全情報ゲームの基礎理論について述べ，基本的な概念や理論を紹介する．3章ではガイスターのルールと基本的な戦術を解説し，4章ではガイスターの先行研究について紹介する．5章では提案手法として完全情報ゲームの評価値を用いた混合戦略ＡＩについて述べる．6章では対戦実験をおこない性能評価をおこなう．7章では本研究の総括をおこない今後の展望について述べる．

第2章　不完全情報ゲームの基礎理論

2.1　展開型ゲーム

展開型ゲーム [7] は，複数プレイヤがおこなうターン性のゲームを記述するモデルである．オセロ・チェス・将棋・囲碁などの完全情報ゲームだけでなく，ポーカーや麻雀のような不完全情報ゲームも展開型ゲームに含まれる．不完全情報な展開型ゲームについても，オセロなどの完全情報ゲームと同様に，ゲーム木によって表すことができる．終端でないゲーム状態では，手番プレイヤが行動を選択し，終端の状態ではそれぞれのプレイヤに利得（勝敗などのスコア）が与えられる．完全情報ゲームと大きく異なる点は，情報集合と呼ばれる，手番プレイヤにとって区別できない状態の集合が定義されることである．展開型ゲームでは，同じ情報集合に属する状態について，同じ行動戦略を立てなければならない．

有限の展開型ゲームは以下の要素から構成される [7].

- プレイヤの有限集合 $N = \{1 \cdots n\}$.
- 行動の履歴 h の有限集合 H．履歴 h は過去の行動の列によって表現され，ゲーム木の頂点に対応する．特に H は空の履歴を持つ．また，任意の履歴 $h \in H$ について，h の接頭辞は全て H に含まれる．例えば，$h = (action_1, action_2)$ が H に含まれていれば，$h = (action_1)$ も必ず H に含まれる．
- 終端履歴（それ以上行動を取ることができない履歴）の集合 $Z \subseteq H$.
- 履歴 $h \in H \backslash Z$ で取ることができる行動の集合 $A(h) = \{a \mid ha \in H\}$．行動 $a \in A(h)$ はゲーム木の辺に対応する．
- 履歴 $h \in H \backslash Z$ で行動するプレイヤ $P(h) \in N \cup \{c\}$.
  c をチャンスプレイヤと呼び，ルーレットなどの不確定要素はチャンスプレイヤ c による行動とみなす.
- チャンスプレイヤ c が $P(h) = c$ なる履歴 h について行動 a を選択する確率 $\sigma_c(h, a)$.
- 手番プレイヤにとって区別できない履歴の集合 I．情報集合と呼ぶ．$h, h' \in I$ のとき，手番プレイヤからは h と h' を区別できない．このとき，$A(h) = A(h')$,

$P(h) = P(h')$ となるため，I_i で取れる行動の集合を $A(I)$，I で行動するプレイヤを $P(I)$ と表現する．

- プレイヤ i の情報集合の集合 $\mathcal{I}_i$．$\mathcal{I}_i$ はプレイヤ i の情報分割と呼ばれる．
- プレイヤ $i \in N$ が終端履歴 $z \in Z$ で獲得する利得 $u_i(z)$．$u_i(z)$ は実数．また，$\Delta_{u,i} = max_z u_i(z) - min_z u_i(z)$ をプレイヤ i に対する効用の範囲と定義する．

注意点として，上記で説明したような情報分割は，プレイヤが自分の過去を忘れることを（システム的な都合で）強制されるような，奇妙な状況をもたらす可能性がある [7]．もし，全てのプレイヤが過去の情報と対応する情報集合を思い出すことができるならば，そのゲームは完全記憶ゲームと呼ばれる [7]．多くのターン制の不完全情報ゲームは，完全記憶な展開型ゲームとして表現することができる．さらに，完全情報ゲームは，任意の情報集合のサイズが 1 になる展開型ゲームとして記述できる．また，任意の終端履歴 $z \in Z$ について，$\sum_{i \in N} u_i(z) = 0$ を満たすとき，零和ゲーム（ゼロサムゲーム）と呼ばれる．

2.2 戦略，到達確率，期待利得

展開型ゲームにおけるプレイヤ i の行動戦略 σ_i（以降，戦略と呼ぶ）は，各情報集合 $I_i \in \mathcal{I}_i$ から行動集合 $A(I_i)$ への確率分布として表現され，プレイヤ i が取ることのできる戦略の集合を Σ_i で表す．更に，全プレイヤの戦略組 $\sigma = \sigma_1 \cdots \sigma_n$ を戦略プロファイルと呼ぶ．戦略プロファイル σ が決まると，各プレイヤの期待利得が定まる．全てのプレイヤが戦略プロファイル σ に従って行動するとき，初期状態から履歴 $h \in H$ に到達する確率を $\pi^\sigma(h)$ と定義すると，プレイヤ i の期待利得は $u_i(\sigma) = \sum_{z \in Z} u_i(z)\pi^\sigma(z)$ で表すことができる．

また，ナッシュ均衡の概念やそれを求める手法を説明するため，戦略や到達確率について，以下のように定義する [7]．

- σ_{-i} : 戦略プロファイル $\sigma = \sigma_1 \cdots \sigma_n$ から，プレイヤ i の戦略 σ_i を除いたもの．
- $\pi_i^\sigma(h)$ $(h \in H)$: プレイヤ i は戦略 σ に従って行動し，それ以外のプレイヤは必ず履歴 h へ向かうよう行動するとき，履歴 h に到達する確率．
- $\pi_{-i}^\sigma(h)$ $(h \in H)$: プレイヤ i 以外は戦略 σ に従って行動し，プレイヤ i は必ず履歴 h へ向かうよう行動するとき，履歴 h に到達する確率．
- $\pi^\sigma(I)$ $(I \subseteq H)$: 戦略プロファイル σ に従って行動するとき，情報集合 I に属するいずれかの履歴 $h \in I$ に到達する確率．すなわち，$\sum_{h \in I} \pi^\sigma(h)$．同様に，$\pi_i^\sigma(I) = \sum_{h \in I} \pi_i^\sigma(h)$，$\pi_{-i}^\sigma(I) = \sum_{h \in I} \pi_{-i}^\sigma(h)$ と定義する．

- $\pi^{\sigma}(h, z)$ $(h \in H, z \in Z)$：全てのプレイヤが戦略プロファイル σ に従って行動するとき，履歴 h から開始した場合に終端履歴 z に到達する確率.

2.3 ナッシュ均衡

展開型ゲームにおいて，各プレイヤはどのような戦略を取るのが最適だろうか．一つの考えは，自分は他プレイヤの戦略を全て知っており，それに対して自分の期待利得が最大になるような戦略を立てることである．しかし，各プレイヤが自分の期待利得を上げるために自分の戦略を変更する操作は，しばしば堂々巡りになってしまう．例えば 2 人のじゃんけんにおいて，相手が必ずグーを出すことを知っていれば，自分は必ずパーを出すのが最適になるため，自分は必ずパーを出す戦略に変更する．しばらくして，それを知った相手は必ずチョキを出すよう戦略を変更するだろう．さらに，それを知った自分は必ずグーを出すよう戦略を変更する．そのため，各プレイヤがある戦略に落ち着くためには，別の意味で「最適性」を考える必要がある．

先ほどの 2 人じゃんけんの例では，グー，チョキ，パーのいずれかを必ず出す戦略を考えていたが，今度は，お互いにグー，チョキー，パーを 1/3 ずつの確率で出す戦略を取っている場合を考える．勝ちなら 1 点，負けなら-1 点，引き分けなら 0 点と考えれば，自分も相手も期待利得は 0 になる．そして，相手の戦略が変わらないとすると，自分は他の戦略に変更しようとも 0 を超える期待利得を得られないので，戦略を変更する動機が発生しない．相手についても同様に，戦略を変更する動機が発生しない．このように,「どのプレイヤも自分だけが戦略を変えることによって，自身の期待利得を上げることができないような均衡状態」になると，各プレイヤの戦略は落ち着くことになる．

ナッシュ均衡は，上記のアイデアを定式化したものである．より形式的には，戦略プロファイル $\sigma = \sigma_1 \cdots \sigma_n$ が式 (2.1) を満たすとき，σ をナッシュ均衡という [7].

$$\forall i \in N, u_i(\sigma_i, \sigma_{-i}) \geq max_{\sigma_i' \in \Sigma_i} u_i(\sigma_i', \sigma_{-i}) \tag{2.1}$$

完全記憶な展開型ゲームについては，2.2 節のように定義された行動戦略についてナッシュ均衡の存在が保証されている．特に，展開型ゲームのうち二人零和不完全情報ゲームにおいては，式 (2.1) の条件を満たす戦略プロファイル $\sigma = \sigma_1, \sigma_2$ があるとき，プレイヤ 1 は σ_1 に従った戦略を取ることで，プレイヤ 2 がどのような戦略を取っても，プレイヤ 1 の期待利得を $u_1(\sigma)$ 以上にすることができる．逆に，プレイヤ 2 が σ_2 を取る場合，プレイヤ 1 の期待利得は $u_1(\sigma)$ 以下に抑えられる．よって，二人零和不完全情報ゲームの場合，ナッシュ均衡に従った戦略を取ることで，最も苦手な相手プレイヤに対する自分の期待利得を最大化することができる．さらに，じゃんけんのように，ルールが対称な二人零和不完全情報ゲームを考える場合，ナッシュ均衡に従った戦略は「十分な試行回数の元，負け越さ

ない戦略」となる．そのため，二人零和不完全情報ゲームでは，ナッシュ均衡を求めることを究極的な目標とすることがしばしばある．

また，ナッシュ均衡の近似として，$\epsilon-$ ナッシュ均衡が定義されている．戦略プロファイル $\sigma = \sigma_1 \cdots \sigma_n$ が式 (2.2) を満たすとき，σ は $\epsilon-$ ナッシュ均衡であるという [7].

$$\forall i \in N, u_i(\sigma_i, \sigma_{-i}) + \epsilon \geq max_{\sigma'_i \in \Sigma_i} u_i(\sigma'_i, \sigma_{-i}) \tag{2.2}$$

2.4 Counterfactual Regret minimization (CFR)

Counterfactual Regret minimization (CFR) [7] は，完全記憶な二人零和不完全情報ゲームにおいて，ナッシュ均衡への収束が保証された手法の 1 つである．CFR では，後悔 (regret) を最小化することで，ナッシュ均衡への収束を効率的におこなう．CFR について説明する前に，まずは regret の概念を定義する．今，展開型ゲームを繰り返しプレイすることを考える．t 回目のゲームでプレイヤ i が戦略 σ_i^t を使用した場合，プレイヤ i の時刻 T における average overall regret R_i^T は式 (2.3) で定義される [7].

$$R_i^T = \frac{1}{T} \max_{\sigma_i^* \in \Sigma_i} \sum_{t=1}^{T} \left(u_i \left(\sigma_i^*, \sigma_{-i}^t \right) - u_i \left(\sigma^t \right) \right) \tag{2.3}$$

この値は，他プレイヤの戦略 σ_{-i}^t を固定したとき，プレイヤ i がある戦略 σ_i^* を取り続けていれば，現状と比べて 1 ゲームあたり最大でどれだけ多くの期待利得を獲得できたかを表す量である．さらにプレイヤ i の時刻 1 から T における平均戦略を，情報集合 $I \in \mathcal{I}_i$，行動 $a \in A(I)$ について，式 (2.4) で定義する [7].

$$\bar{\sigma}_i^T(I)(a) = \frac{\sum_{t=1}^{T} \pi_i^{\sigma^t}(I) \sigma^t(I)(a)}{\sum_{t=1}^{T} \pi_i^{\sigma^t}(I)} \tag{2.4}$$

完全記憶な二人零和不完全情報ゲームにおいて，時刻 T におけるプレイヤ 1, 2 の average overall regret R_1^T，R_2^T がいずれも ϵ 以下ならば，その平均戦略のプロファイル $\bar{\sigma}^T$ は $2\epsilon-$ ナッシュ均衡であることが証明されている．したがって，regret を最小化することで，ナッシュ均衡を求めることができる．

CFR の基本的な考え方は，average overall regret を，独立して最小化できる regret の加法に分解することである．CFR では，各情報集合 I 上で定義される「反事実的後悔 (counterfactual regret)」と呼ばれる概念が用いられる．まず，反事実的利得（counterfactual utility）を式 (2.5) で定義する [7]．式 (2.5) は，全プレイヤが初期状態から戦略プロファイル σ に従って動くときの，情報集合 I におけるプレイヤ i の期待利得となる．

$$u_i(\sigma, I) = \frac{\sum_{h \in I, z \in Z} \pi^{\sigma}_{-i}(h) \pi^{\sigma}(h, z)\, u_i(z)}{\pi^{\sigma}_{-i}(I)} \tag{2.5}$$

CFR では，情報集合 I において手番プレイヤ $P(I)$ が必ず行動 $a \in A(I)$ を取るよう戦略を変更するとき，反事実的利得がどのように変化するかを表す量に着目する．そのように変更した戦略を $\sigma^t|_{I \to a}$ とおくとき，情報集合 I における immidirate counterfactual regret は式 (2.6) で定義される [7].

$$R^T_{i,\mathrm{imm}}(I) = \frac{1}{T} \max_{a \in A(I)} \sum_{t=1}^{T} \pi^{\sigma^t}_{-i}(I) \left(u_i\left(\sigma^t\big|_{I \to a}, I\right) - u_i\left(\sigma^t, I\right)\right) \tag{2.6}$$

$R^T_{i,\mathrm{imm}}(I)$ は負になることもある．ここで，$R^{T,+}_{i,\mathrm{imm}}(I) = \max\left(R^T_{i,\mathrm{imm}}(I), 0\right)$ とおくと，重要な性質として，average overall regret を式 (2.7) のように上から抑えることができる [7]．したがって，各情報集合について $R^T_{i,\mathrm{imm}}(I)$ を最小化することで，average overall regret を最小化できる．

$$R^T_i \le \sum_{I \in \mathcal{I}_i} R^{T,+}_{i,\mathrm{imm}}(I) \tag{2.7}$$

CFR は，反復的に戦略の更新を繰り返すことで，各情報集合 I における $R^T_{i,\mathrm{imm}}(I)$ を最小化する．まず，情報集合 $I \in \mathcal{I}_i$，行動 $a \in A(I)$ についての counterfactual regret を式 (2.8) で定義する [7].

$$R^T_i(I, a) = \frac{1}{T} \sum_{t=1}^{T} \pi^{\sigma^t}_{-i}(I) \left(u_i\left(\sigma^t\big|_{I \to a}, I\right) - u_i\left(\sigma^t, I\right)\right) \tag{2.8}$$

さらに，$R^{T,+}_i(I, a) = \max\left(R^T_i(I, a), 0\right)$ と定義し，$T+1$ 回目の反復でプレイヤ i が用いる戦略 σ^{T+1}_i を式 (2.9) のように更新する [7].

$$\sigma^{T+1}_i(I)(a) = \begin{cases} \frac{R^{T,+}_i(I,a)}{\sum_{a \in A(I)} R^{T,+}_i(I,a)} & \text{if} \sum_{a \in A(I)} R^{T,+}_i(I, a) > 0 \\ \frac{1}{|A(I)|} & \text{otherwise.} \end{cases} \tag{2.9}$$

反復を T ステップ繰り返し，最後に時刻 1 から T における平均戦略 $\bar{\sigma}^T$ を式 (2.4) によって求めると，求めた平均戦略について，

$$R^T_{i,\mathrm{imm}}(I) \le \Delta_{u,i} \sqrt{|A_i|} / \sqrt{T} \tag{2.10}$$

が成り立つことが証明されている [7]．ただし，

$$|A_i| = \max_{h : P(h) = i} |A(h)| \tag{2.11}$$

としている．よって，$\bar{\sigma}^T$ について，

$$R_i^T \leq \Delta_{u,i} \left|\mathcal{I}_i\right| \sqrt{|A_i|}/\sqrt{T} \tag{2.12}$$

が成り立つ [7]．したがって，十分多くの反復を繰り返すことで，平均戦略がナッシュ均衡へ収束する．

第3章　ガイスターについて

3.1　ルール

ガイスター（Geister）[12] は，6×6 の盤と 2 種の駒を用いて遊ぶ，二人零和確定不完全情報ゲームである．ドイツのゲームデザイナーであるアレックス・ランドルフ氏によって考案され，1982 年，ドイツ年間ゲーム大賞にノミネートされた．本ゲームの大きな特徴として，プレイヤは自駒の色を観測できるが，まだ取っていない相手駒の色を観測できないことが挙げられる．相手駒の色は，それまでの動きなどから推測する必要がある．

ガイスターでは，6×6 のマスからなる 36 マスの盤面を用いる．最初，各プレイヤは青駒，赤駒を 4 個ずつ，自陣 8 マスに自由な組み合わせで初期配置する．初期配置をおこなった盤面例を図 3.1 に示す．本論文では，マスの座標を a-f，1-6 の組み合わせで表現する，先手から見て，左上を a1，右上を f1，左下を a6，右下を f6 とする．駒色は先手赤，先手青，後手赤，後手青をそれぞれ R, B, r, b と表記し，先手色非公開駒，後手色非公開駒をそれぞれ U, u と表記する．先手自陣は b5, e6 を対角線とする長方形領域，後手自陣は b1, e2 を対角線とする長方形領域である．

ゲームは交互着手により進行する．手番プレイヤは自分の駒を 1 つ選び，上下左右 4 方向のいずれかに 1 マス動かす．自駒があるマスには移動できないが，移動先に相手駒があればそれを取り，色を観測できる．図 3.1 の矢印が書かれたマスはゴールを表しており，先手は a1, f1，後手は a6, f6 をゴールとする．ゴールに自分の青駒が置かれている場合，次のターンにその駒を盤外へ脱出させることができ，勝利となる．なお赤駒は盤外へ脱出できない．

ガイスターでは次の 3 条件のいずれかを満たしたプレイヤが勝利となる．

- 相手の青駒を全て取る
- 自分の赤駒を全て取らせる
- 自分の青駒を 1 つでも脱出させる（ゴールに移動してから，もう 1 手番必要）

逆に次の 3 条件のいずれかを満たしたプレイヤは敗北となる．

- 相手の赤駒を全て取ってしまう
- 自分の青駒を全て取られてしまう
- 相手の青駒が 1 つでも脱出してしまう

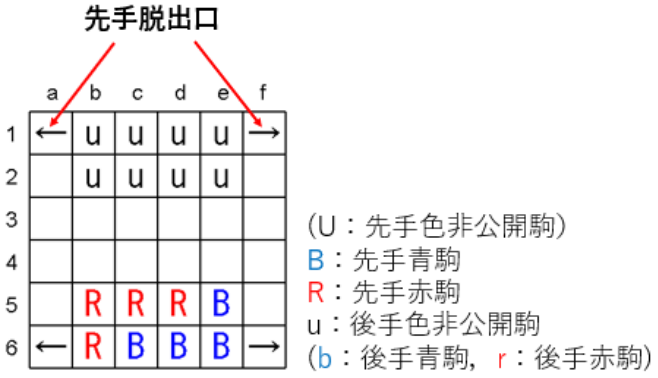

図 3.1: ガイスター初期配置例

3.2 アクション表現

ガイスターのアクションは，初期配置，駒移動の 2 種類からなる．相手の初期配置を観測することはできないが，自身の初期配置，自分および相手の駒移動は観測できる．本研究では，各プレイヤの初期配置を，上の段から順番に，同じ段なら左の列から順番に配置する駒の色（R, B, r, b）を書いた文字列で表す．例えば，図 3.1 の初期配置は，”RRRBRBBB” と表記する．駒移動は，移動元マス（a1-f6），移動方向（↑→↓←），および取った駒の色を表す文字によって表記する．例えば，図 3.2 において，b5 から b4 へ先手赤駒 R を移動する動きは b5 ↑と書く．e6 から f6 へ先手赤駒 R を移動し後手青駒 b を取る動きは e6 → (b) と，取った駒の色を () で付け加えて書く．

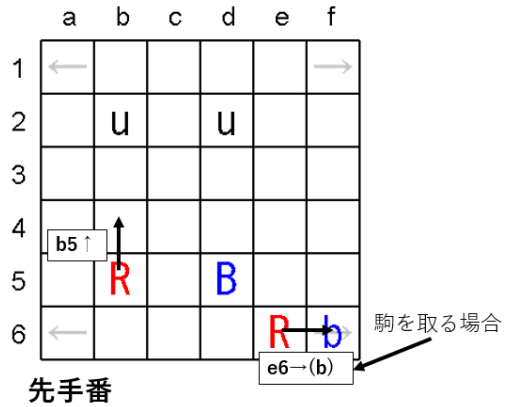

図 3.2: 移動表現

3.3 基本戦術

ガイスターは基本的に「いかに自分の赤を取らせ，相手の赤を見極めるか」,「いかに相手の青脱出を阻止し，自分の青を脱出させるか」を考えるゲームになる．対戦相手の駒色が見えないため，いかに自分の駒色を騙し，相手の駒色を見極めるかが攻略の鍵となる．また，特定の相手を攻略する場合，相手駒の色と動かし方に偏りがあれば，それを上手く見破ることが重要である．逆に，どのような相手にも搾取されにくい戦術を練る場合，自駒を動かす際に，色と動かし方に偏りが出すぎないように気を付けることが大切である．ガイスターの戦術についてはいくつか考察されている [21] が，本論文でも [21] を参考にするなどして，基本戦術を説明する．

まず，初心者でも思いつきやすい戦術の 1 つとして「赤特攻 [21]」が考えられる．お互いに対戦相手の赤はなるべく取りたくないと考えられるため，それを利用して赤を積極的に攻め駒にする戦術である．赤を取った相手は苦しみ，取らなければその駒で敵陣を攻めることができるため，低リスクに見える．しかし，赤ばかりで攻めていると，相手に赤を見破られてしまう．もし，自分が攻めた駒を相手に赤と決めつけられ，それが当たっていた場合，相手は決めつけた駒を無視し取らないことで，赤全取り負けのリスクを避けることができる．すると，自分にとって大きく不利になってしまうだろう．だからといって,「青特攻」ばかりおこなうと，今度は相手に青を取られて負けてしまう．

したがって，赤青を混ぜて攻めることが大事と考えられる．このとき，自分が赤も青も攻める可能性があれば，相手は行動から駒色を確実に判別することができず，相手が色を誤認した場合には自分有利になると考えられる．例えば，相手に青が攻めていると思わせ赤を取らせたり，逆にこれは赤が攻めているだろうと思わせ青脱出をおこなうことができる．なお，単純なランダム行動では,「青特攻」や赤を取らせようとする相手に弱いため，どのようにして赤青を混ぜるかはよく考える必要がある．

例として，図 3.3 の局面を挙げる．図 3.3 は，先手のゴールマス a1, f1 に自駒と相手駒が隣接しており，もし a1, f1 に自駒を進めれば，相手はそれを取るか取らないか考える必要がある．この局面において，先手が必ず赤をゴールへ近づけるとしたら，そのことを後手が知っている場合は，後手はゴールへ近づいてきた赤を無視すれば良い．また，先手が必ず青をゴールへ近づけるとしても，後手はゴールへ近づいてきた青を取れば良い．一方，先手が確率的に行動を選び，赤も青もゴールへ近づける可能性を発生させると，後手にはゴールへ攻めてきた駒が赤なのか青なのか区別できなくなる．すると後手に赤を取らせ先手有利になる可能性，後手に青を取らせず勝利する可能性を発生させることができる．よって，先手の立場では，赤も青も確率的にゴールへ近づけ，後手の駒色誤認を狙うことが重要である．一方，後手視点では，先手の駒がゴールへ攻めてきた場合に，駒を取るか取らないか考える必要がある．ここで，後手が「取る」を必ず選択するとしたら，先手がそれを知っている場合は，赤をゴールへ近づけ取らせるだろう．また，後手が「取らない」を必ず選択するとしたら，先手がそれを知っている場合は，青をゴールへ近づけ脱出を狙う．よって，後手の立場では，先手が駒を攻めてきた場合に，駒を取るか取らないかを確率的に選択するのが良いと考えられる．

これらの議論とは別に，ガイスターの必勝盤面に関する研究が報告されている [22][23][24]．文献 [22][23] では，ガイスター初心者教育用コンテンツとして『詰めガイスター問題』を提案し，問題生成をおこなった．また，文献 [24] では，後退解析によって，自他ともに 2 駒の場合において必勝局面の列挙をおこなった．詰めガイスター問題の例を図 3.4 に示す．図 3.4 は先手局面であり，後手非公開駒 u のうち 1 つは赤 r，もう 1 つは青 b である．図 3.4 において，先手が初手 e2 ↑と e1 に赤 R を移動すると，後手は「赤を全て取ったら負け」なので e1 の駒を取ることができない．よって，先手は f3 にある青 B を f1 から必ず脱出させることができ，先手必勝となる．

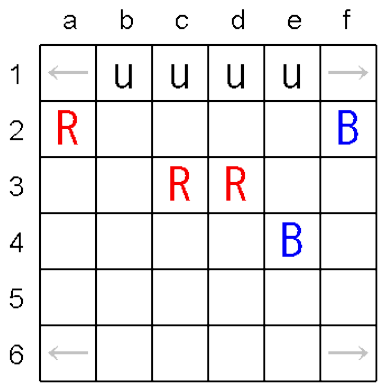

図 3.3: 局面例

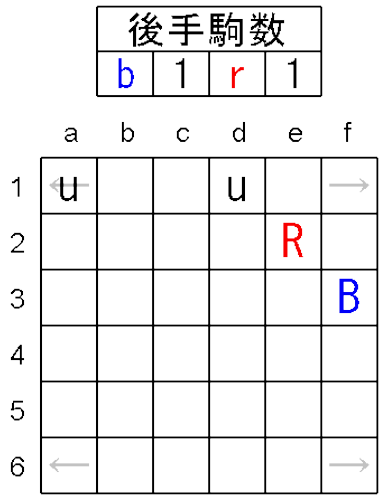

図 3.4: 先手必勝局面の例（B 脱出）

第4章　ガイスターの先行研究

4.1　ガイスターのAIプレイヤ

ガイスターのAIプレイヤについては様々な手法が試されているが，高水準な強さを持つAIプレイヤはまだ確認されていない．先行研究では性能評価のためのAIとして，着手可能な手からランダムに選択する『ランダムAI』，ひたすら脱出のために青をゴールへ向かわせる『猪突猛進AI』などが用いられている．さらに高レベルな比較対象として，モンテカルロ法を用いた『モンテカルロ法AI』も用いられている．

不完全情報ゲームにおけるモンテカルロ法では，状態を仮定し，仮定した状態それぞれについてプレイアウトを割り振る方法がよく用いられる[13][14]．各プレイアウトは完全情報ゲームのシミュレーションとなる．このとき，各状態についてどのくらいプレイアウトを割り振るか，評価値をどのように集計するかといった問題が存在する．ベンチマークとしてよく用いられるモンテカルロ法AIでは，仮定される全ての状態について等しい回数のプレイアウトを割り振り，着手の評価値を各状態における勝率の算術平均とし，最大評価の手を指す．しかし，高確率な状態に多くのプレイアウトを割り振る，算術平均以外の集計方法を考えるなどして，性能改善できる余地もある．そのことから，モンテカルロ法AIを評価対象とするだけでなく，その性能改善を図る研究もおこなわれている[13]．

三塩・小谷は，モンテカルロ法AIにおけるプレイアウト回数の改善をおこなった．過去の着手などから，各状態にいる確率のようなものを計算し，より確率の高そうな状態に多くのプレイアウトを割り振るUPP（Using Past Playout）というアルゴリズムを提案した[13]．UPPをガイスターAIプレイヤに用いた結果，猪突猛進AIには圧勝し，さらにUPPを用いなかったモンテカルロ法AIに勝率55%を挙げ，ガイスターにおけるUPPの有効性を示した．

佐藤は，自己対戦による強化学習をおこない，3層ニューラルネットワークによって表現された行動価値関数を改善することで，強いAIの開発を試みた[14]．有限手数の完全情報ゲームにおける行動価値関数では，現在の盤面 s においてある行動 a を取った場合の行動価値 $Q(s,a)$ を考えれば理論的には十分だが，不完全情報ゲーム『ガイスター』では，状態の推測などを考慮するために，これまでの着手が重要となる可能性がある．そこで佐藤は，今までの駒の動きから計算される特徴量 f を追加した行動価値関数 $Q(f,s,a)$ を考案し，強化学習をおこなった．さらに，ゲームが引き分けになりにくいよう着手制限を与え，学習を進めやすく

した．さらに，相手の駒を「取ったら赤に変化し，脱出時は青に変化する駒（紫駒）」とみなすことで，完全情報ゲーム的な方法で必勝手探索をおこなう方法を提案し，df-pn[25] アルゴリズムにより必勝手の検出をおこなった．結果，ランダムＡＩに勝率約 70%を挙げ，UCT アルゴリズムを用いたモンテカルロ法ＡＩに勝ち越した．

末續・織田は，ルールベースにより駒推定，および行動をおこなう AI を開発した [15]．今までの駒の動きから相手駒の青らしさを評価し，青らしさが何番目以内であれば取るなどの行動をルールベースにより決定している．さらにパターンマッチによる必勝形の検索もおこなっており，終盤読み切り力の向上を図っている．結果，GAT2018 杯ガイスター AI 大会では 3 プログラム中 1 位を獲得した．木村・伊藤は，畳み込みニューラルネットワーク（CNN) を用いてバリューネットワーク，ポリシーネットワークを表現し，モンテカルロ木探索（MCTS) を組み合わせた深層学習によって，強いガイスター AI プレイヤの開発を試みた [17]．ここで，通常のモンテカルロ木探索では最も有力と判断した手を選ぶが，100%その手を選ぶと同じ盤面で毎回同じ手を打つことになってしまうため，ばらつきを与えるためにボルツマン分布を用いて他の手も選ばれるようにしている．予備実験として完全情報ガイスターのプレイヤを生成した結果，対ランダム AI には 97 勝 3 敗 0 分，対 MCTS には 53 勝 44 敗 3 分した．

川上・橋本は，佐藤の研究 [14] で提案された紫駒のアプローチをゲーム木探索に応用し，「紫駒 AI（PurpleAI)」と呼ばれる AI を開発した [16]．具体的には相手駒を「取ったら赤，脱出時は青になる駒（紫駒)」に変換した局面を考え，深さ 5 の MinMax 法をおこなっており，評価関数には『青駒の個数』『駒の位置』を用いた．[16] では，青駒が相手より多ければ経験的に有利，駒位置については自分の駒が敵陣に近いほど経験的に有利と考えており，これらを反映した評価関数を設計している．結果，ランダム AI，猪突猛進 AI に圧勝，GPW2017 杯ガイスター AI 大会優勝 AI およびその評価関数改良版にも勝ち越し，GPW2018 杯ガイスター AI 大会において 6 プログラム中 2 位を獲得した．また，GAT2020 杯ガイスター AI 大会では [16] の手法を用いたＡＩ「Naotti2020-3」が 8 プログラム中 1 位を獲得した．一方で，初期配置以外のランダム性がなく，相手に戦略を知られている場合は弱い可能性がある．例えば，赤駒を相手駒に隣接させやすく，青駒を相手駒に隣接させにくい特徴があり，隣接させてきた駒を赤駒と推定し，隣接状態を回避させた駒を青駒と推定するような相手には弱いと考えられる．

Chen Chen, Tomoyuki Kaneko は，Chance-sampled CFR[8] および Pure CFR[26] を 3×2 盤のガイスターに適用し，計算時間，戦略の分析をおこなった [18]．ただし，ボードゲームにおける無限ループの問題に対処するため，2〜20 手の手数制限を設け，その手数に達したら強制的に引き分けとした．また実装の都合により，初期配置をランダムとした．結果，3×2 ガイスターにおける先手初手の最適戦略「自分の赤駒を前進させる」を Chance-sampled CFR, Pure CFR 双方とも得ることができた．また，同じイテレーション数で計算時間を比較しており，Chance-sampled

CFR では手数に対して指数的に計算時間が増加する一方で，Pure CFR では手数の増加に対してほぼ線形な計算時間となり，手数の長いゲームにおいては Pure CFR がより速く収束するだろうと考察された．

さらに Chen Chen, Tomoyuki Kaneko は，深層学習の技術を CFR に取り入れた手法 Deep CFR により，6×6 盤ガイスターにおいて，履歴を考慮しない AI プレイヤ，直前の着手も考慮した AI プレイヤの開発を試みた [19]．履歴を考慮しない AI プレイヤについて，1000 イテレーションの学習をおこなったプレイヤを 6 回生成し，ランダムプレイヤと 2000 回対戦させた．結果，勝率 70%〜85%を挙げ，性能にばらつきこそあるものの，ガイスターにおいて学習が進むことを示した．しかし，川上・橋本の研究 [16] で開発された PurpleAI の深さ 1 バージョンと対戦させたところ，履歴を考慮しない AI プレイヤでは勝率 38%程度，履歴を考慮する AI プレイヤについては勝率 34%程度となり，まだ十分強いとは言えなかった．Deep CFR では，ゲーム木のうちサンプリングがおこなわれたノードしか訪問しないため，現実的な時間ではゲームの全体像を十分に得られず，局所解に陥った可能性がある．DeepCFR が十分な性能に収束するためには多くの試行時間がかかるため，現実的な時間では強い戦略を生成することが難しい．

以上のようにガイスターにおいて，十分強い AI はまだ報告されていない．特に，2020 年のガイスター AI 大会で優勝した「紫駒 AI（PurpleAI）」[16] は，駒の動きにランダム性がなく， 戦略を知っている相手には簡単に付け込まれてしまう．CFR[7] は確率的な戦略を取るのに適しており，3×2 盤のガイスターへの適用例があるが [18]，計算時間が大きな課題である．また，Deep CFR による AI[19] は，現実的な時間で十分な強さまで収束していない．

4.2 ガイスターにおける問題生成

ガイスター初心者の上達を促すためのコンテンツとして，詰め問題生成について研究が報告されている．

石井らは，後手駒の色がどうあろうと，後手がどのように動こうと，必ず先手側が勝利する局面に着目し，詰めガイスター問題を提案した [22]．文献 [22] では，後手側の駒を完全に非公開とした『一般問題』，後手の駒色を一部公開した『一部公開問題』を提案し，ランダム生成法による問題生成をおこなった．必勝判定では，紫駒 [14][16] の概念を用い，$\alpha\beta$ 法を適用している．また，単純すぎる問題を取り除くため，様々な条件で盤面の絞り込みをおこない，最後の先手赤駒を壁として利用する問題など，狙った問題の生成手法を提案した．

さらに石井らは，問題生成アルゴリズム，必勝判定の改善をおこない，より難しいガイスター問題の生成をおこなった [23]．具体的には，ランダム生成法ではなく，逆向き生成法を用いることで，長手数の問題を効率的に生成できるようになった．また，dfpn アルゴリズム [25] を用いることで，高速な必勝判定を可能した．その結果，文献 [22] では 11 手問題までしか生成できなかったのに対し，文献

[23] では 19 手問題の生成に成功している．また，生成された問題について面白さ，難しさを推定するモデルを生成し，被験者実験によりある程度の精度で推測できることを示した．

また川上らは，駒数が最小である自他共に赤青 1 つずつの局面に抑え，後退解析によって詰めガイスター問題の列挙をおこない，盤面数や最長手数などを調査した [24]．後退解析とは，勝敗が決定した局面から手を戻していき，戻した局面の勝敗を決定していくアルゴリズムである．結果,「一般問題」では先手勝ち 191,992 個，その他の盤面は 514,868 個となり，最長手数は 19 手であった．さらに、駒が完全に公開された「完全公開問題」では，先手必勝盤面は 783,232 個，その他の盤面は 630,488 個となり，最長手数は 37 手となった．

第5章　完全情報ゲームの評価値を用いた混合戦略AI

5.1　課題背景

ガイスターの先行研究において紫駒 AI[16] は，2020 年のガイスター AI 大会で優勝しているが，駒の移動にランダム性がないため,「この行動からこの状態はあり得ない」といった絞り込みを相手にされる可能性がある．そのため，戦略を知っている相手には付け込まれやすいと考えられる．例えば，紫駒 AI には赤駒を相手駒に隣接させやすく，青駒をめったに相手駒に隣接させない特徴がある．よって，この特徴を知っている相手には簡単に負けてしまう．紫駒 AI に限らず，確定的な行動をおこなうプレイヤは，どのような局面でどのような行動を取るかを対戦相手に完全に熟知された場合，行動から簡単に状態を絞り込まれ，負けてしまうと考えられる．本研究では，確率的行動を取ることで，相手に付け込まれにくくすることを目指す．ガイスターは二人不完全情報ゲームであるため，最も苦手な相手に対する勝率を最大化する戦略は，ナッシュ均衡に従った戦略となる．そのため，究極的にはナッシュ均衡を求めることがゴールとなる．

ガイスターは，ゲーム木によって表現することができるため，ゲーム木を生成し，CFR などを適用することでナッシュ均衡を求めることができる．ガイスターを表現するゲーム木の概念を図 5.1 に表す．図 5.1 は，先手初期配置，後手初期配置，先手駒移動，後手駒移動…の方法を木構造で表したものであり，各ノードには状態が書かれている．特に，深さ 2 以降のノードは完全情報な局面を持つ．ガイスターにおいて，手番プレイヤにとって区別できないノードのグループ（情報集合）は，まだ取っていない相手駒の色組み合わせのみが異なるノード同士であり，同じ情報集合に属するノードでは同じ行動確率で着手をおこなわなければならないという制約がつく．そのため，MinMax 法などのゲーム木探索を直接適用することはできない．

先行研究では，短い手数制限を加えた 3×2 盤のガイスターについてナッシュ均衡を求めている [18]．しかし，終局までゲーム木を展開しているため計算量が大きく，それ以上大きなサイズのガイスターについては現実的な計算量で行動を求めることができない．

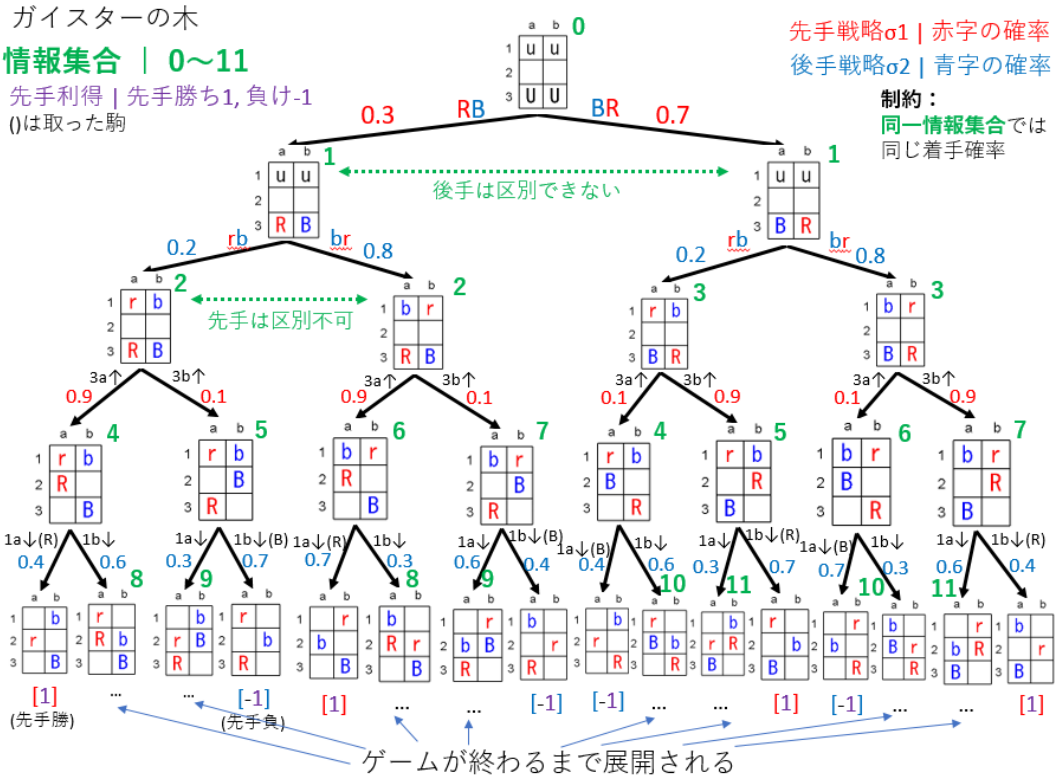

図 5.1: 素朴なゲーム木（終局まで展開）

5.2　アルゴリズムの概要

本研究では，5.1 節で解説したゲーム木展開における計算量の問題を解決するため，終局までノードを展開せずに，ある深さまで展開したら，リーフノードの良し悪し（期待利得）を予測するためのヒューリスティックな評価値を与える方法を提案する．このような限定的なゲーム木を生成し，CFR をおこなうことで，現実的な時間で，現局面に対する確率的な行動を得ることが期待できる．図 5.2 に，本研究で生成するゲーム木の概念を示す．図 5.1 と比べると，3 手先以上（深さ 5 以上）のノード展開をおこなわず，深さ 4 のノードに評価値を与えているところが異なる．図 5.2 は，完全情報なガイスターの勝敗を評価値とした例であり，どのリーフノードが持つ盤面も先手が b を取り勝利することから勝率 1.0 と推測している．

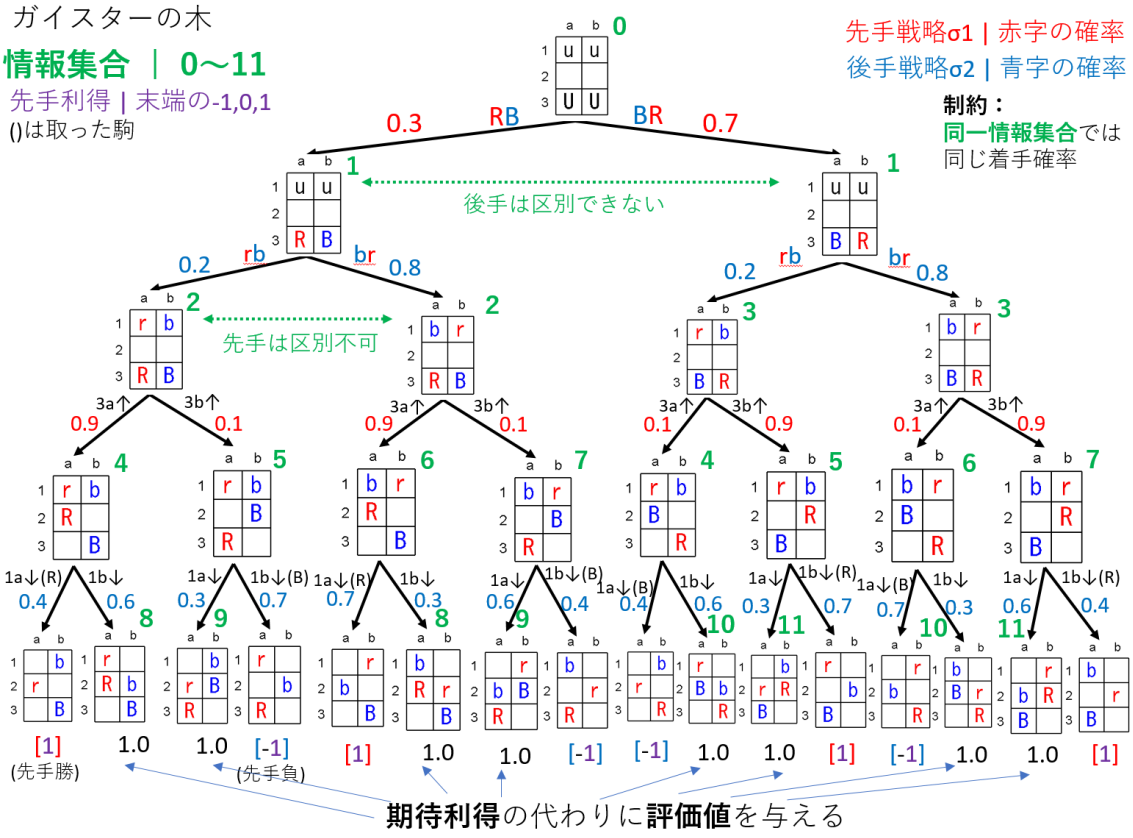

図 5.2: 評価値を与えたゲーム木（深さ 4 まで展開する例）

具体的には次の手順で実行する．ただし，駒の初期配置については単純なランダムでおこなう．リーフノードへの具体的な評価値の与え方は次節で述べる．

1. 現局面（過去の着手を含まない）について，先手の駒色，後手の駒色をそれぞれ隠した局面 S を生成する．

2. S をルートノードとし，ある深さまで展開したゲーム木を生成する．自分は深さ偶数手番とする．

3. リーフノードについて評価値を与え，これを利得の代わりとする．

4. 生成したゲーム木において，CFR をおこない，混合戦略（各ノードに対する確率的行動）を計算する．

5. 現局面と対応するノードを参照し，確率的なアクションを取る．現局面に対応するノードとしては，深さ 2 のノード（初期配置が終わった直後の局面を持つ）であって，現局面と自分の色配置が一致するものを任意に取る．

5.3 リーフノードの利得評価関数

未決着なリーフノードについて，どのくらいの利得を期待できるか評価するために，利得評価関数を設計する．最も素朴な評価は，未終局なら引き分け扱いとする方法だが，ゲーム序盤ではリーフノードの多くが未決着となるため，行動の良し悪しを評価することができない．実際には，リーフノードは完全情報な局面を持つので，その局面を評価することで，単に引き分けを返すよりも高い精度でリーフノードの良し悪しを評価できるのではないかと考えられる．そこで本研究では，各リーフノードの良し悪しを評価するため，そのリーフノードが持つ完全情報局面の評価をおこなう．評価には様々な方法が考えられるが，本研究では，リーフノードが持つ完全情報盤面から完全情報なガイスターをプレイした際の理論的な勝敗を求め，その勝敗を評価値として用いる．

本研究では，4×4 盤，各プレイヤの青赤が 2 個以下の完全情報なガイスターについて，後退解析を用いて，互いに勝ちを目指して最善を尽くした場合の勝敗分を記録したデータベースを生成する．後退解析は，勝敗が決定した局面から手を戻していき，戻した局面の勝敗を決定していく手法である．相手に負け局面を渡すことができる局面は勝ち，どのように手を進めても相手に勝ち局面を渡してしまう局面は負けとなり，最後まで勝敗が確定しなかった局面は引き分けとなる．これは，相手の勝ちを阻止できる一方で相手に負け局面を渡すこともできないためである．

第6章　実験

6.1　事前実験

本研究で用いるリーフノードの利得評価関数を設計するための準備として，完全情報なガイスターを考えた場合の勝敗を後退解析を用いて計算した．具体的には，4×4 盤，各プレイヤの青赤が 2 個以下の完全情報局面について，先手目線での勝敗を計算した．結果，71,001,840 局面のうち，勝ちは 49,821,729 局面，負けは 17,051,259 局面，引き分けは 4,128,852 局面となった（表 6.1）．また，勝ち局面数と手数の関係は表 6.2 のようになり，最長勝ちは 47 手と確認された．負け局面数と手数の関係は表 6.3 のようになり，最長負けは 48 手と確認された．

表 6.1: 勝敗と局面数

全局面数	71,001,840
勝ち局面数	49,821,729
負け局面数	17,051,259
引き分け局面数	4,128,852

表 6.2: 手数と勝ち局面数

1	21,405,186
3	13,300,150
5	4,188,579
7	2,643,147
9	1,993,674
11	1,661,412
13	1,312,694
15	1,020,795
17	762,849
19	548,515
21	386,897
23	258,268
25	153,386
27	88,879
29	50,575
31	26,115
33	12,319
35	5,096
37	2,085
39	736
41	264
43	80
45	24
47	4

表 6.3: 手数と負け局面数

2	7,514,602
4	3,239,989
6	1,690,200
8	957,902
10	846,053
12	691,214
14	581,617
16	465,892
18	351,644
20	255,988
22	177,502
24	114,142
26	71,675
28	44,969
30	25,196
32	12,968
34	5,893
36	2,458
38	920
40	286
42	108
44	28
46	11
48	2

6.2 対戦実験

提案手法の性能を評価するため，ベンチマークプレイヤを用意し対戦実験をおこなう．対戦実験では，最も苦手な相手に対する勝率（最小勝率）に焦点を当て，ベンチマークプレイヤと提案手法の性能比較をおこなう．また，提案手法における利得評価関数の違いによってどのような差が生まれるのかを調査する．さらに，勝敗の内訳などを調べ，どのような方法で勝利・敗北しているかを考察する．

6.2.1 実験方法

4×4 盤のガイスターについて対戦実験をおこない，勝敗などを調査する．まずベンチマークプレイヤ同士の総当たり戦をおこない，勝敗数などを分析する．その後，提案手法とベンチマークプレイヤを戦わせ，最も苦手な相手に対する勝率などの比較をおこなう．各組み合わせでは，先手後手がそれぞれ 50 回ずつ現れるようにし，100 試合おこなう．初期配置は手前の一段（先手は a4-d4，後手は a1-d1）におこない，どのプログラムもランダムに選択する．なお，無限にゲームが終わらないのを防ぐため，200 手で決着がつかない場合は引き分けとする．ベンチマークプレイヤとしては『ランダム』『猪突猛進』『紫駒 AI』『対紫駒』を用意する．それぞれ以下の内容である．

- ランダム
  - 合法手のなかからランダムに選択し，行動する
  - ただし，1 手で脱出可能なときは脱出をおこなう
  - 攻めも守りも貧弱なため，弱い
- 猪突猛進
  - 1 つの青駒をひたすらゴールへ近づけるプログラム．取られたら別の青駒で同じことをする．
  - 具体的には近い方のゴールまでのマンハッタン距離が最小の青駒を選ぶ．複数あれば，最も上段，それでも複数あれば，最も左列にある駒を選ぶ．
  - ゴールへなるべく近づけるように移動する．移動方向が複数考えられるときは，上方向にゴールがあるとして，左，右，上，下の順で優先し，移動方向を選ぶ．ただし本実験の設定では，下が選ばれる状況は発生しない．
  - もし，選んだ青駒がゴールにある場合は，その駒を脱出させる．

- 基本的には，近づけた駒を積極的に取る相手に弱く，近づけた駒を取らない相手に強い．

- 紫駒 AI
  - 相手駒を「取ったら赤，脱出時は青に変化する駒（紫駒）」とみなし，深さ 5 の MinMax 法をおこなう．
  - 評価関数は文献 [16] と同様,「青の個数」「駒がゴールからどれだけ離れているか」の 2 つから作る．
  - 詰みを検出できるなどの強みがあり，GPW2018 杯準優勝 AI，GAT2020 杯優勝 AI にもこの手法が用いられた．
  - 一方，赤ばかり相手駒に隣接させるなど，色を簡単に推定できるような動き方をする．

- 対紫駒
  - 紫駒 AI の弱点をあらわにするために用意したプレイヤ．
  - 紫駒 AI が最初に突っ込ませる駒はほとんど赤だろうという仮定をもとに開発する．
  - 最初に突っ込んできた駒を赤とみなし，ゲーム終了まで絶対取らないようにする．それ以外の駒は，取っても良い駒（青）とみなす．
  - 深さ 5 の MinMax 法により完全情報なガイスターを探索し，手を決定する．
  - 評価関数には,「青駒の個数差」「駒がゴールからどれだけ離れているか」の 2 つを用いる．相手の青駒数を考慮すること以外は紫駒 AI と同様である．自分の青駒を保持したい，相手の青駒（とみなした駒）を取りたい，ゴールへ駒を近づけたいという観点から設計されている．
  - 駒が突っ込むまでは紫駒 AI と同様の意思決定をおこなう．
  - 基本的には，赤ばかり近づける相手に強く，青ばかり近づける相手に弱い．

また，提案手法におけるプログラム名と利得評価関数の対応を次のようにする．

- CFR-A，未終局のリーフノードを引き分けと評価する AI.
- CFR-B，リーフノードが持つ完全情報局面 S を評価する AI．S から開始して完全情報ガイスターをおこなった場合の勝敗を評価値とする．

提案手法において，ゲーム木の展開深さは 5 とする．深さ 0，1 では色配置の列挙をおこなっているため，駒移動に関する木の展開は 3 手までおこなうことになる．CFR のイテレーション数は 1000 とする．

6.2.2 実験結果

ベンチマークプレイヤとの対戦結果を表 6.4 に示す．表 6.4 の最小勝率は，最も低い勝率となった相手に対する勝率を表している．

まず，4 つのベンチマークプレイヤの最小勝率に着目すると，いずれについても最小勝率 15%を下回る相手が存在することが分かる．特に表 6.4 の紫駒 AI は，ランダム相手に勝率 90%，猪突猛進には勝率 100%と大きく勝ち越した一方で，対紫駒には勝率 13.5%と大きく負け越した．原因として，紫駒 AI は確率的な行動を取らず，行動から状態を見破られやすいことが挙げられる．駒を取った場合を除外して駒隣接回数をカウントした結果を表 6.5 に示す．表 6.5 より，紫駒 AI は赤駒を相手駒に隣接させ，青駒はめったに相手駒に隣接させない特徴を持つことが分かる．よって，紫駒 AI が対紫駒に大敗したのは，多くの試合で赤を見破られ，見破った駒以外を積極的に取られたからだと推測できる．実際，紫駒 AI の勝敗内訳は表 6.6 のようになり，対紫駒に対しては 6 割の試合で青を取られ負けている．また，対紫駒は最初に接近してきた駒を取らないため，相手が最初に青を接近させると，その青の脱出を阻止できない．そのため，対紫駒は青を積極的に近づける相手に弱く，表 6.4 のように猪突猛進には大敗している．

表 6.4: ベンチマークプレイヤとの対戦結果（勝-敗-分)

自分縦軸，相手横軸	ランダム	猪突猛進	紫駒 AI	アンチ紫	最小勝率 [%]
ランダム		9-91-0	10-90-0	30-70-0	9
猪突猛進	91-9-0		0-100-0	92-8-0	0
紫駒 AI	90-10-0	100-0-0		12-85-3	13.5
アンチ紫	70-30-0	8-92-0	85-12-3		8
CFR-A	78-22-0	36-64-0	24-76-0	43-57-0	24
CFR-B	78-22-0	71-29-0	20-80-0	19-81-0	19

表 6.5: 1 試合あたりの駒隣接回数（紫駒 AI）

	ランダム	猪突猛進	対紫駒
赤	4.29	1.07	3.95
青	0.12	0	0.44
平均手数 (1 試合)	28.54	10.85	28.82

一方，表 6.4 において，CFR-A，CFR-B のベンチマークプレイヤに対する最小勝率に注目すると，CFR-A では最小勝率が 24%，CFR-B では最小勝率が 19%となり，どのベンチマークプレイヤよりも最小勝率が大きいことが分かる．また，ランダムにはどちらの手法も勝率 78%を挙げた．駒の隣接回数についても表 6.7，6.8 のように，相手駒に隣接させる駒がほとんど赤といった現象は起きていないこと

表 6.6: 紫駒 AI の勝敗内訳

	ランダム	猪突猛進	対紫駒
青取り勝ち	11	67	0
赤取り負け	0	0	0
赤取らせ勝ち	58	33	2
青取られ負け	0	0	60
脱出勝ち	21	0	10
脱出負け	10	0	25

が確認できる．また，相手駒が自駒に隣接してきた直後その相手駒を取る確率は表 6.9 のようになった．相手赤駒が残り 1 個の際，紫駒 AI は駒を絶対に取らないが，CFR-A では 11%，CFR-B では 20%の確率で駒を取るため，相手赤駒が残り 1 個の状況では、簡単に脱出されにくくなったと考える．残念ながら全てのベンチマークプレイヤに勝ち越すことはできなかったものの，確率的な行動を取ることで，相手に付け込まれにくくなったと考える．

表 6.7: CFR-A の駒隣接回数

	ランダム	猪突猛進	紫駒 AI	対紫駒
赤	2.26	0.59	3.69	1.70
青	1.46	0.47	2.40	1.13
平均手数 (1 試合)	24.83	11.06	34.60	16.70

表 6.8: CFR-B の駒隣接回数

	ランダム	猪突猛進	紫駒 AI	対紫駒
赤	2.40	0.82	4.16	2.03
青	0.73	0.15	1.11	0.77
平均手数 (1 試合)	21.61	10.12	26.14	16.14

表 6.9: 相手駒が自駒に隣接してきた直後その相手駒を取る確率

	紫駒 AI	CFR-A	CFR-B
全体	0.20	0.21	0.33
相手赤 1 個	0	0.11	0.20

次に，CFR-A，CFR-B の比較をおこなう．まず表 6.4 より，CFR-B は CFR-A よりも猪突猛進に強い一方で，対紫駒には弱いことが分かる．CFR-B の勝敗内訳は表 6.11 のようになり，猪突猛進に対しては青駒を取り勝利することができている．一方で，CFR-A の勝敗内訳（表 6.10）とも比較すると，対紫駒については CFR-B の方が，赤を取らされ負ける試合数が多かった．よって，CFR-B では駒を積極的に取りすぎている可能性がある．また，CFR-A，CFR-B は共通して紫駒 AI に弱かった．表 6.10，6.11 の勝敗内訳について，紫駒 AI の列に着目すると，いずれも赤を取らされ負ける試合が半数以上あった．一般にガイスターでは相手の赤駒を取ってしまうと，それだけ不利に試合が展開されると考えられる．しかし，CFR-A ではリーフ局面を引き分けと判断し，CFR-B ではリーフ局面以降は互いの駒色が見える状況を扱っており，相手の赤が少なくなることのリスクを十分評価できていない可能性がある．

表 6.10: CFR-A の勝敗内訳

	ランダム	猪突猛進	紫駒 AI	対紫駒
青取り勝ち	12	21	4	0
赤取り負け	6	0	56	13
赤取らせ勝ち	31	10	0	36
青取られ負け	7	8	3	31
脱出勝ち	35	5	20	7
脱出負け	9	56	17	13

表 6.11: CFR-B の勝敗内訳

	ランダム	猪突猛進	紫駒 AI	対紫駒
青取り勝ち	9	58	2	0
赤取り負け	14	0	64	41
赤取らせ勝ち	49	13	0	8
青取られ負け	5	0	3	32
脱出勝ち	20	0	18	11
脱出負け	3	29	13	8

第7章　おわりに

本研究では，二人零和不完全情報ゲーム『ガイスター』において，対戦相手に付け込まれる余地を減らすため，確率的行動を取る AI を開発した．現実的な時間で確率的行動を計算するため，本研究では数手先までしかゲーム木を展開せず，リーフノードに評価値を与える手法を提案し，4 種類のベンチマーク AI との対戦実験をおこなった．結果，4×4 盤のガイスターについて，ベンチマーク AI 同士の対戦ではどの AI についても勝率 15%を下回る苦手な相手が存在した一方で，提案手法 2 種とベンチマーク AI の対戦では，最も苦手な相手に対しても 19%，24%の勝率となった．残念ながら全てのベンチマーク AI に勝ち越すことはできなかったが，確率的な着手によって付け込まれる余地を減らすことには成功したと考えられる．

今後の方針として，過去の着手を考慮することが挙げられる．本研究の手法においては，過去の着手を用いず，現局面のみを入力としていたが，過去の着手を用いることでさらなる性能の向上が期待できる．例えば，脱出しなかった駒をそれ以降のターン赤とみなすことができれば，試合を有利に運ぶことができる．今後の方針として，過去の着手を用いた手法を考えたい．

参考文献

[1] Donald E. Knuth and Ronald W. Moore. *An Analysis of Alpha-Beta Pruning*, Artificial Intelligence, Vol.6, No.4, pp.293-326 (1975).

[2] Remi Coulom. *Efficient Selectivity and Backup Operators in Mote-Carlo Tree Search*, Proc. 5th International Conference on Computers and Games, Turin, Italy (2006).

[3] 美添一樹：コンピュータ囲碁に革命を起こした新手法，情報処理，Vol.49, No.6, pp.686-693 (2008).

[4] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. *Playing Atari with Deep Reinforcement Learning*, NIPS Deep Learning Workshop, pp.1-9 (2013).

[5] David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, loannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, llya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. *Mastering the Game of Go with deep neural networks and tree search*, Nature 529, pp.484-489 (2016).

[6] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, Yutian Chen, Timothy Lillicrap, Fan Hui, Laurent Sifre, George van den Driessche, Thore Graepel, and Demis Hassabis. *Mastering the Game of Go without Human Knowledge*, Nature 550, pp.354-359 (2017).

[7] Zinkevich, M., Johanson, M., Bowling, M. and Piccione, C. *Regret Minimization in Games with Incomplete Information*, Advances in Neural Information Processing Systems 20, pp.1729-1736 (2008).

[8] Neller, T. W. and Lanctot, M. *An Introduction to Counterfactual Regret Minimization*, Proceedings of Model AI Assignments, The Fourth Symposium on Educational Advances in Artificial Intelligence (EAAI 2013) (2013).

[9] Michael Bowling, Neil Burch, Michael Johanson, Oskari Tammelin. *Heads-up limit hold'em porker is solved*, Science Vol. 347, Issue 6218, pp.145-149 (2015).

[10] Noam Brown and Tuomas Sandholm. *Superhuman AI for multiplayer poker*, Science Vol. 365, 885-890 (2019).

[11] Junjie Li, Sotetsu Koyamada, Qiwei Ye, Guoqing Liu, Chao Wang, Ruihan Yang, Li Zhao, Tao Qin, Tie-Yan Liu, and Hsiao-Wuen Hon. *Suphx: Mastering Mahjong with Deep Reinforcement Learning*, Artificial Intelligence, arXiv:2003.13590v2, pp.1-28 (2020).

[12] ガイスター．[http://www.mobius-games.co.jp/Gester.htm]．（アクセス：2021/02/03）．

[13] 三塩 武徳，小谷 善行：ゲームの不完全情報推定アルゴリズム UPP とそのガイスターへの応用，情報処理学会研究報告，Vol.2014-GI-31, No.4, pp.1-6 (2014).

[14] 佐藤 佑史：ガイスターにおける自己対戦による行動価値関数の学習，電気通信大学学術機関リポジトリ (2015).

[15] 末續 鴻輝，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，GAT2018, pp.13-16 (2018).

[16] 川上 直人, 橋本 剛：完全情報ゲームの探索を用いたガイスター AI の研究，第 23 回ゲームプログラミングワークショップ，pp.35-42 (2018).

[17] 木村 勇太, 伊藤 毅志：深層強化学習を用いたガイスター AI の構築，第 24 回ゲームプログラミングワークショップ，pp.130-135 (2019).

[18] Chen Chen and Tomoyuki Kaneko. *Counterfactual Regret Minimization for the Board Game Geister*, 第23回ゲームプログラミングワークショップ, pp.137-144 (2018).

[19] Chen Chen and Tomoyuki Kaneko. *Utilizing History Information in Acquiring Strategies for Board Game Geister by Deep Counterfactual Regret Minimization*, 第 24 回ゲームプログラミングワークショップ, pp.20-27 (2019).

[20] 伊藤 雅士, 大久保 壮浩, 木谷 裕紀, 小野 廣隆：ガイスター AI のキーパー戦略の有効性，情報処理学会研究報告，Vol.2019-GI-42, No.3, pp.1-7 (2019).

[21] ガイスター考察 - 静岡大学浜松キャンパスゲーム研究会 log，入手先〈https://su-gameken.hatenadiary.org/entry/20141119/1416394953〉（参照 2021-01-15）

[22] 石井 岳史，川上 直人，橋本 剛，池田 心 : 不完全情報ゲーム『ガイスター』における 2 種の詰め問題の提案と考察，Vol.2019-GI-41, No.19, pp.1-8 (2019).

[23] 石井 岳史，川上 直人，橋本 剛，池田 心 : 難しい詰めガイスター問題の生成法，第 24 回ゲームプログラミングワークショップ，pp.12-19 (2019).

[24] 川上 直人，池田 心，石井 岳史，橋本 剛 : 後退解析による詰めガイスター問題の列挙，Vol.2020-GI-43, No.13, pp.1-8 (2020).

[25] Nagai, A.: *Df-pn Algorithm for Searching AND/OR Trees and Its Applications*, Ph. D. thesis, Department of Information Science, University of Tokyo (2002).

[26] Gibson, R.: *Regret Minimization in Games and the Development of Champion Multiplayer Computer PokerPlaying Agents*, PhD Thesis, University of Alberta (2014).