多様な特徴量を用いた Arimaa 評価関数の比較学習

川上　裕生[1,a)]　鶴岡　慶雅[1]

概要：Arimaa はチェス・将棋・囲碁などのゲームと比較し、ゲーム木のサイズが非常に大きく、コンピュータにとって難しいゲームである。チェスや将棋などのゲームでは、評価関数の学習手法として比較学習がよく用いられているが、 Arimaa では実用的な評価関数を比較学習で学習できていない。そこで、本研究では Arimaa の実用的な評価関数を棋譜から学習することを目指した。学習時に探索を行わない場合と学習時に探索を行った場合で実験を行ったが、どちらも人間が手でつけた重みベクトルよりも優れた結果を示すことはなかった。この原因としては、学習時に探索を行わない場合は教師データの質が低いこと、学習時に探索を行った場合では探索に時間がかかるために学習に利用できる局面の数が限られてしまうことが考えられる。

Comparison Training of Arimaa Evaluation Functions with Diverse Features

Yusei Kawakami[1,a)]　Yoshimasa Tsuruoka[1]

Abstract: Arimaa is a game more difficult for computers than chess, shogi and Go. One of the reasons is that the size of the Arimaa game tree is much bigger. Comparison training is often used for learning an evaluation function for a shogi program, but there is no research on learning a practical evaluation function for Arimaa by comparison training. In this work, we attempted to learn a practical evaluation function for Arimaa from game records. We built evaluation functions with and without search, but neither of them was better than the original handmade evaluation function. One possible reason is that the quality of the training data without search was too low. Another reason should be that the amount of the training data for learning with search was too small—we could not use large training data because even a shallow search is computationally demanding in Arimaa.

1. はじめに

機械学習を用いてコンピュータゲームの性能を向上させる研究はゲーム AI の研究の初期の頃から行われてきている [1]。機械学習を用いることで、そのゲームに対する深い知識を持たない人でもゲーム AI の作成が可能になり、手作業でヒューリスティックを組み込む必要もなくなるため手間も削減できる。

本稿では Arimaa を対象とする。 Arimaa はチェス・将棋・囲碁などと同じ、二人零和有限確定完全情報ゲームに分類されるゲームである。 Arimaa は 2002 年に Omar Syed 氏が、AI にとっては難しいものの、ルール自体は子供でも理解できるというコンセプトで作成したゲームである。このゲームが考案された契機として、1997 年にチェスプログラム、ディープブルーが世界チャンピオン相手に勝利したことがあげられる。

2004 年以降毎年 1 回、 Arimaa チャレンジとして、コンピュータ対人間の対戦が行われている。人間に対して勝ち越した最初のプログラムには 10,000 ドルの賞金が用意されているが、2014 年現在、まだそれを達成したプログラムは存在していない。

Arimaa が AI にとって難しい理由を説明する。まず一つ目に、一ターンごとの合法手の数が非常に多いことがあげられる。 Arimaa では一ターンに 4 回の移動を行うことができ、そのため一ターンに平均して 18,000 通りの合法手が存在する [2]。類似した他のゲームにおける一ターンの合法手の数の平均は、チェスでは 35 通り、将棋では 80 通り [3]、囲碁では 250 通り程度であり、 Arimaa の合法手の数は桁違いに多いということが分かる。合法

[1] 東京大学大学院工学系研究科
Graduate School of Engineering, The University of Tokyo
[a)] kawayu@logos.t.u-tokyo.ac.jp

手の数が多いことによりゲーム木のサイズが非常に大きくなるため、単純な力任せの探索方法では深い探索を行うことができない。二つ目に、駒の初期配置をプレイヤーが自由に決められるということがあげられる。ゲーム開始時に、先手番のプレイヤーが手前二列に好きなように駒を配置したあとに、後手番のプレイヤーも駒を配置する。この駒の初期配置の方法だけでも 4.2×10^{15} 通り存在し、このゲームを複雑にする一因となっている。また、初期配置が一つに定まっていないため、チェスや将棋などのゲーム AI で行われている定跡の利用が難しくなっている。

これらの要因により、 Arimaa では人間よりも強力なプログラムを作成するに至っていない。Arimaa の AI の研究を行うことにより、 AI が人間の頭脳に追いつき、追い越すための鍵となることが期待される。

Arimaa の評価関数を機械学習を用いて行った研究を 2 つ紹介する。まず一つ目に強化学習による Arimaa の評価関数の学習である [4]。この研究では、 TD, TD-Leaf, Rootstrap, Treestrap の 4 つの学習手法を用いて評価関数を学習している。対戦実験において学習前の評価関数を利用したものと比較して有意に勝率が向上しており、強化学習を利用することで評価関数を改善できることが分かる。二つ目に、将棋プログラム Bonanza [5] で用いられている比較学習 (comparison training) [6] という学習手法を Arimaa に適応したもの研究がある [7]。しかしこの論文では、駒の位置に対する重みを学習したにとどまっており、Arimaa の評価関数に有効であると考えられる他の特徴量を使用していない。そのため、プログラムの強さも、既存のプログラムより劣るものとなっている。

本研究では、 Arimaa の評価関数に有用な特徴量を比較学習を用いて学習することを目指す。チェスや将棋などのゲームでは、棋譜を利用した学習手法である比較学習を利用することで、有用な評価関数を学習することに成功している。そのため、Arimaa においても比較学習をうまく利用することで、有用な評価関数の学習が可能なのではないかと考えられる。

本稿の構成を以下に述べる。第 2 章で Arimaa のルールについて簡単に説明する。第 3 章で評価関数の学習に関する関連研究の紹介、第 4 章で本研究の提案手法の説明を行う。第 5 章で評価の結果を示し、第 6 章で考察、第 7 章でまとめと今後の課題について述べる。

2. Arimaa のルール

この章では、 Arimaa のルールについて述べる。

Arimaa では 8 × 8 のマス目を持った盤を用いる。駒はプレイヤーごとに象とラクダが各 1 個、馬と犬と猫が各 2 個、ウサギが 8 個である。盤のサイズがチェスと同じであり、また駒の種類と数もチェスと同じであるため、チェスセットを流用できるようになっている。駒の強さは強い方から順に象、ラクダ、馬、犬、猫、ウサギの順になっており、これは後に述べる PUSH や PULL、FREEZE に影響する。8 個のウサギのうちいずかを 1 個を反対側の一番奥の列まで先に到達させたプレイヤーの勝利となる。

駒の初期配置

Arimaa ではチェスや将棋などとは異なり、駒の初期配置は決まっておらず、プレイヤーが最初に決める必要がある。まず最初に先手のプレイヤーが手前二列に自分の駒を好きなように配置する。先手が全ての駒を配置したら、後手も同様に自分の駒を配置する。その後は交互に手番を消費して、駒を動かしていく。駒の初期配置の一例を図 1 に示す。

駒の動き

プレイヤーは一度の手番において最大 4 ステップまで動かすことができる。自分の駒を一度動かす行動は 1 ステップ分である。各駒は隣接する上下左右の空きマスに移動することができる。しかしウサギは後ろに移動することはできない。一度の手番で一つの駒を何度も動かすこともできるし、複数の駒を動かすこともできる。しかし、自分よりも強い相手の駒が隣接しており、かつ味方の駒が隣接していない場合、その駒は FREEZE し動かすことができない。

Arimaa には PUSH や PULL という特別な動きが存在し、これにより相手の駒も動かすことができる。 PUSH は、自分の駒と隣接している相手の駒を任意の方向の空きマスに移動させ、相手の駒が元々いた位置に自分の駒を移動させる。 PULL は相手の駒と隣接している自分の駒を移動させた後に、隣接していた相手の駒を自分が元いた位置に移動させる。 PUSH や PULL は自分の駒と相手の駒の移動で計 2 ステップ消費したことになる。PUSH や PULL は自分より弱い駒に対してのみ可能で

図 1 A game board of Arimaa[8]

あり、自分と同じ強さの駒や自分より強い駒に対しては行うことができない。

Trap

c3, c6, f3, f6 の 4 つのマスには trap が存在する。 Trap 上にいる駒は、味方の駒が隣接していない場合ゲームから除外される。PUSH や PULL を用いて相手の駒を trap にかけることができ、そのようにして相手の駒を減らしていくことがこのゲームの基本的な戦術となる。

3. 関連研究

Arimaa に類似したゲームの評価関数学習手法として、将棋の評価関数の学習の研究について述べる。将棋では、 Bonanza [5] が評価関数をプロ棋士の棋譜から学習することに成功して以来、教師あり学習がよく用いられている。

Bonanza で用いられている学習手法は、チェスで用いられていた比較学習 [6] をベースとした手法であり、プロ棋士の指した手と異なる手を指す回数が少なくなるように学習が行われる。数式で表すと以下の誤差関数 J を最小化する特徴ベクトル v の最適化問題である。

$$J(P_0, ..., \mathbf{v}) = \sum_{i=0}^{N-1} \sum_{m=1}^{M} T[\xi(p_m, \mathbf{v}) - \xi(p_{m=0}, \mathbf{v})] \quad (1)$$

N は学習データの局面数、P_i は局面、M はその局面における合法手の数、ξ は局面の評価関数、T は評価値の差を棋譜の指し手との一致度に変換する関数であり、シグモイド関数などが用いられる。p_m は局面 P の指し手 m の後の局面を探索して得られた最善応手手順の末端局面である。これに、過学習を防ぐための正則化項を加えたものを目的関数として、最急降下法により目的関数 J を最小化することで重みベクトルの学習を行う。

また、激指 [9] では比較学習と平均化マージンパーセプトロンを利用した手法が用いられている [10]。この手法では、ある局面 P に対する指し手 m のあとの最善応手手順を探索し、得られた末端局面を利用した学習を行う。その局面に対する評価値を返す関数を $\xi(p_m, \mathbf{v})$ とし、全ての合法手の集合を M とする。まず棋譜の指し手が他の指し手と比べて相対的に高く評価されているかどうかを確認し、棋譜の指し手より評価値が高い、もしくはある margin を持って評価値が近いものを発見する。具体的には局面 P における棋譜の指し手 $m = 0$ に対して

$$M' = \{m \in M | \xi(p_m, \mathbf{v}) + \text{margin} > \xi(p_{m=0}, \mathbf{v})\} \quad (2)$$

となる指し手の集合 M' を求める。M' に含まれている指し手は棋譜の指し手と比較して評価値が十分小さくはないということになる。

全ての $m \in M'$ の特徴ベクトルについて棋譜の指し手の特徴ベクトルとの差を取り、その平均を重みベクトル $\mathbf{v}$ に加える。式で表すと式 (3) のようになる。

$$\mathbf{v} \leftarrow \mathbf{v} + \frac{1}{|M'|} \sum_{m \in M'} (\phi(p_{m=0}) - \phi(p_m)) \quad (3)$$

$\phi(P)$ は、局面 P の特徴ベクトルである。

それを全ての局面に対して繰り返し、最後に学習途中に現れた重みベクトルの平均を取り、それを最終的な重みベクトルとしている。

4. 多様な特徴量を用いた Arimaa の評価関数の学習

本研究では、 Arimaa の実用に足る評価関数を棋譜から学習することを目指す。

Arimaa において将棋と同様の手法で評価関数の学習を行おうとした場合に問題点として、将棋では浅い探索を非常に短い時間で行うことができるが、 Arimaa では浅い探索を行うことにすら時間がかかってしまうことが挙げられる。例えば今回実験を行った “Faerie” [8] という Arimaa プログラムでは、次の自分の手番の指し手まで（3 手番先）を読むのにだいたい 1 秒から 2 秒程度の時間を要する。

学習時に探索を行わない場合と、探索を行う場合の 2 つの手法で学習を行った。探索を行わない場合は、棋譜の指し手を指した局面の特徴量と、ランダムな合法手を指した局面の特徴量の比較を行った。この場合は、探索を行っていないため多くの指し手について評価値の比較を行うことができる。探索を行う場合は、全ての指し手について探索を行うのではなく、現状の評価関数で最善であると考えられた手のみに絞った。棋譜の指し手を指さずに深さ d の探索を行って得られた最善応手手順の末端局面の特徴量と、棋譜の指し手を指した後の局面を深さ $d-1$ の探索を行って得られた最善応手手順の末端局面の特徴量を比較し、学習を行う。学習の手順を図 2、図 3 に示す。

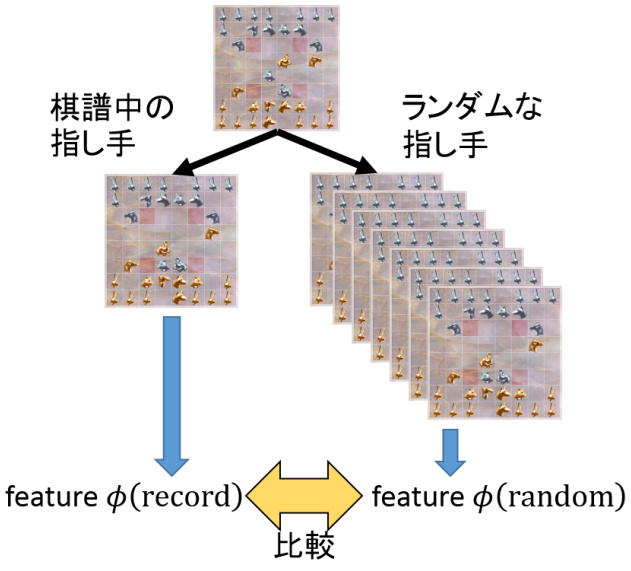

図 2 探索を行わない学習

5. 評価

5.1 評価設定

インターネット上で公開されている Arimaa プログラムである Faerie を用いて実験を行った。学習用の棋譜としては、 Arimaa のウェブサイト [8] からダウンロードできる棋譜約 30 万局 (2014 年 8 月時点) のうち、対戦者の Rating が 2,200 を超えている棋譜 2,321 局を利用した。

学習の手法は激指のものをベースとした。探索を行わない場合は、棋譜中の各局面について棋譜の指し手とは別に 1,000 手を比較対象として用いた。探索を行う場合は 2,321 局 (186,775 局面) 全てに対して探索を行うのには時間がかかってしまうため、局面をランダムに局面を選び出して探索を行わせた。重みベクトルの初期値として、 Faerie で利用されている重みを用いた。

学習した評価関数は棋譜との指し手の一致率と、対戦実験によって性能を測定した。対戦実験では、評価関数を学習していないオリジナルの Faerie を対戦相手とした。対戦者の Rating が 2,000 を超えている棋譜から、重複のないように初期局面を 100 局面選び出し、先手後手を入れ替えて計 200 対戦行った。

棋譜との指し手の一致率はテストデータ用の棋譜として Rating が 2,200 を超えている棋譜 126 局を利用した。学習した評価関数で探索を行い得られた最善手と、棋譜中の指し手との比較を行った。 Arimaa では一度の手番において最大 4 ステップ分の動きが可能であり、異なる指し手が 4 ステップ中 3 ステップは同じ動きをしていたり、異なる指し手であっても手番終了時の局面が同一であったりすることが考えられる。そのため、各手番において同じステップが出現する回数と、手番終了時の局面が一致する回数を調査した。

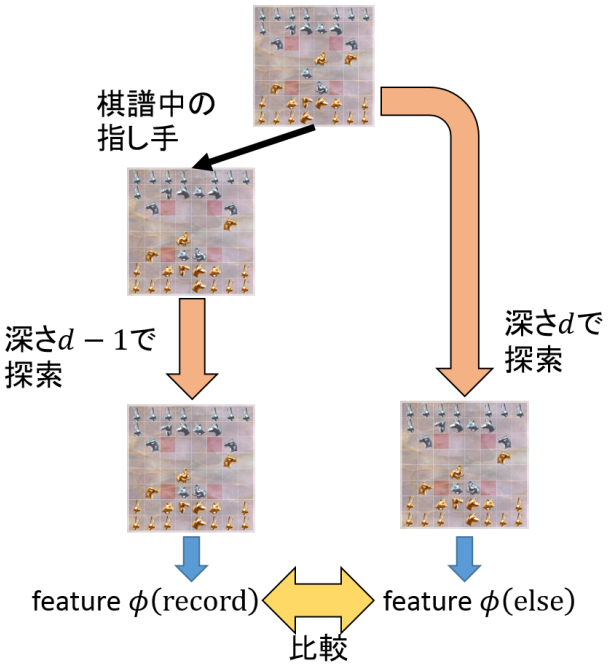

図 3 探索を行った学習

5.2 特徴量

評価関数の特徴量は、 Faerie で用いられていた特徴量をそのまま採用した。Faerie の特徴量は大きく分けて「 trap に関する特徴量」、「駒の価値に関する特徴量」、「ウサギに関する特徴量」の 3 つから成り立っている。

Trap に関する特徴量

- 各 trap に隣接している先手の駒の強さの合計
- 各 trap に隣接している後手の駒の強さの合計
- 各 trap に隣接している駒のうち最も強い駒を持つプレイヤー

駒の強さは象から順に 6, 5, 4, 3, 2, 1 とする。

駒の価値に関する特徴量

Fairie の駒の価値に関する評価関数は、駒の種類をベースとして、その駒に対して各フラグが立っているかどうかを利用して重みを計算しており、特徴量の線形和としては設計されていない。そのため、学習を行う際には “フラグ 1, 2, 4 が成立している象” の重みといったように特徴量を書き換えている。

各フラグを以下に示す

- その駒が FREEZE しているか
- 隣接する空きマスがあるか
- Trap に隣接している場合、他に同じ trap に隣接する味方の駒が他にいるか
- Trap から二マス離れた地点にいる場合、 trap の隣が空きマスか
- Trap 上にいる場合、複数の敵の駒に隣接されているか
- 犬や猫の駒の場合、その駒のいる列が 3 列目以降か

ウサギに関する特徴量

- 各ウサギのいる列
- 七列目にいるウサギの前と左右が空きマスか、味方の駒か、それともそれ以外か

5.3 実験結果

探索を行わない場合はイテレーションを 10 回、探索を行った場合ではイテレーションを 1 回回した。学習時間は探索を行わない場合で約 8 時間、探索を行った場合 10,000 局面で約 14 時間、30,000 局面で 41 時間かかった。

探索を行わない学習中にランダムな指し手が棋譜中の指し手よりも良いと判断された確率を図 4 に示す。教師データに対する一致率は学習が進むに連れて上昇しており、きちんと学習が行われていることが分かる。

実験の結果を表 1 に示す。探索を行わない場合は勝率・一致率ともに低い結果となった。教師データの数を多く用意できるものの、1 手先の評価のみしか学習されていないため、探索を行う際に適した評価関数が学習で

表 1 勝率と一致率

評価関数	勝率	一致率	
		ステップ	ターン
faerie	50%	21.8%	7.8%
探索なし	17% (34 - 166)	11.6%	1.0%
探索有り (10,000 局面)	45.5% (91 - 109)	21.5%	7.8%
探索有り (30,000 局面)	48.5% (197- 203)	21.7%	7.3%

きてないのだろう。探索を行った場合では、元の評価関数と精度が殆ど変化していない。これは、探索時間との兼合いで学習に用いることができる局面数が少ないため、重みベクトルが大きく更新されることがなかったからだろう。学習に用いる局面の数を更に増やした場合の変化も確認する必要がある。

6. 考察

6.1 特徴量

今回の実験においては学習の効果を示すために Faerie で用いられている特徴量と同じ特徴量を用いて実験を行ったが、精度の高い評価関数を作るためには、更に特徴量を増やす必要があると考えられる。

例えば二駒間の位置関係のような特徴量は、将棋やチェスの評価関数の特徴量としてよく用いられているが、要素数が非常に多く手動で調整するのは困難である。このように、手動で調整するのが困難な特徴量を学習することができれば、機械学習によって強いプログラムを作成することが可能になる。

6.2 静止探索

学習を改善するための手法として、静止探索を利用することが考えられる。静止探索とは、チェスや将棋などのゲームにおいて、末端の局面の評価値を安定させるために、探索の末端局面から駒の取り合いなどの評価値が大きく変動する指し手のみを考えて探索する手法のことである。

Arimaa における静止探索では、駒を取る手や駒を取られないように守る手を生成すればいいと考えられる。しかし、一度の手番で最大 4 回まで駒を動かせるため、駒を取る指し手に絞っても数多くの指し手が存在する。そのため、静止探索において探索を行う指し手中の動きを駒の取ることに関係するに限定し、一度駒を取ったらステップが残っていてもパスを選択するなど、更に条件をつける必要があると考えられる。駒を取ることに関係する指し手であるかどうかを判断するためには数ステップ先まで考える必要があるため、駒を取ることに関係する指し手という制限にしても、その判断に時間がかかることになりかねない。PUSH や PULL のような指し手のみに限定すると、駒を取るのに邪魔になっている自分の駒を移動させなければ駒を取れない場合を見落としてしまう。静止探索にかける時間と精度の間でうまく妥協点を見つけて静止探索を行う必要がある。

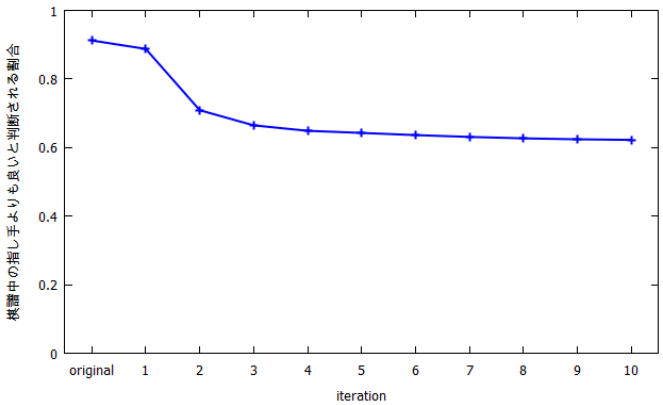

図 4 学習の進行状況

6.3 枝刈り

Arimaa において探索時間を減らすための枝刈り手法として、 Move Ranking が有効であることが知られている [4][11]。枝刈りを行うことで、探索時間が短縮でき、教師局面数を増やすことが可能になる。

今回の実験では探索時間がネックになって利用できていない教師データが多く存在するため、探索時間を短縮することで更に多くの局面からの学習が可能になり、学習の効果が高まることが期待される。

7. 終わりに

本稿では、 Arimaa の評価関数の精度を高めるために、Arimaa の評価関数を棋譜から学習することを目指した。Arimaa では浅い探索にすら時間がかかってしまうため、学習時に探索を行わない場合と、教師データの数を絞って探索を行って学習した場合の二つの手法について評価を行った。実験の結果、学習前の評価関数よりも良い評価関数を学習することはできなかった。学習時に探索を行わない場合は教師データの質が低すぎること、学習時に探索を行った場合は利用可能な教師データの数が少ないことが、理由として考えられる。

学習を改善する手法として、評価関数の特徴量を増やす、静止探索を行う、枝刈りを行って探索にかかる時間を減らすなどの手法が挙げられる。今後は、これらの手法を用いることで評価関数を比較学習を用いて行うことができるかどうか検討したい。

参考文献

[1] Samuel, A. L.: Some Studies in Machine Learning Using the Game of Checkers, *IBM J. Res. Dev.*, Vol. 3, No. 3, pp. 210–229 (1959).

[2] Brian, H.: A Look at the Arimaa Branching Factor, http://arimaa.janzert.com/bf_study/ (2006).

[3] 松原　仁，半田剣一：ゲームとしての将棋のいくつかの性質について，情報処理学会研究報告. 人工知能研究会報告，Vol. 94, No. 83, pp. 21–30 (1994).

[4] Wu, D. J.: Move Ranking and Evaluation in the Game of Arimaa, PhD Thesis, Harvard University (2011).

[5] 保木邦仁：局面評価の学習を目指した探索結果の最適制御，第 11 回ゲームプログラミングワークショップ 2006，pp. 78–83 (2006).

[6] Tesauro, G.: Comparison training of chess evaluation functions, *Machines that learn to play games* (Fürnkranz, J. and Kubat, M., eds.), Nova Science Publishers, Inc., Commack, NY, USA, pp. 117–130 (2001).

[7] Kanjanapa, T., Kanako, K. and Yoshiyuki, K.: Design and implementation of Bonanza Method for the Evaluation in the Game of Arimaa, Technical Report 4, Department of Computer and Information Sciences, Tokyo University of Agriculture and Technology (2012).

[8] Arimaa - Intuitively simple ... intellectually challenging, http://arimaa.com.

[9] 将棋プログラム「激指」のページ, http://www.logos.t.u-tokyo.ac.jp/~gekisashi/.

[10] 鶴岡慶雅：3 選手権優勝記：激指の技術的改良の解説 (<ミニ特集>コンピュータ将棋の不遜な挑戦)，情報処理，Vol. 51, No. 8, pp. 1001–1007 (2010).

[11] Jain, A., Ebrahim-Zadeh, N., Mohan, V. and Choksi, V.: Move Ranking in Arimaa (2013).