モンテカルロ将棋における方策の学習

関栄二[†1]　三輪誠[†2]　近山隆[†1]

近年，特に UCT の登場以降，囲碁においてモンテカルロ法を用いた強いコンピュータプレイヤが作られている．こうした成功を受け，将棋においてもモンテカルロ法の適用が模索されている．本稿では，モンテカルロ将棋における方策学習への，Simulation Balancing の適用を提案する．1800 局面程度で学習し予備的評価を行ったが，利用した特徴数が多く学習前よりも弱くなるという結果となった．

Learning Policy in Monte-Carlo Shogi

EIJI SEKI,[†1] MAKOTO MIWA[†2] and TAKASHI CHIKAYAMA[†1]

Since the advent of UCT, strong computer players using Monte-Carlo Methods have been build for the game of Go. Following these attainments, schemes to apply the method to the game of Shogi have been explored. In this paper, we propose to apply *Simulation Balancing* to the studying policy of Monte-Carlo Shogi players. We learn by this method in about 1800 positions and did a preliminary evaluation. However, the number of used features was too large, and the player became weaker than before learning.

1. はじめに

モンテカルロ木探索は，特に UCT (Upper Confidence bound applied to Tree)[5] の登場以降，囲碁において大きな成功を収めている．この成功を受けて，将棋においても，並列化の容易さや従来のアルファベータ木探索が苦手な領域の補完への期待から，その適用が模索されている[7][8][9][10]．

モンテカルロ木探索では，より意味のあるプレイアウトを行うため，ゲーム固有の知識を用い，方策 (policy) の改善を行うことが必要である．この改善においては,「より現実にありそうなシミュレーションを行う」ことと「多様なシミュレーションを行う」ことの間にある，トレードオフの関係を調整することが重要である．前者の極端な例は，Minimax 戦略に基づいた決定的な指し手の選択であるし，後者の極端な例は，全ての合法手からランダムで等確率に指し手を選ぶ，確率的なものである．この間で適切に調整を行えれば，現実的な範囲で多様なシミュレーションができ，より少ないシミュレーション，より浅い木探索でも，比較的正確な平均報酬[⋆1]が得られることになる．

こうした調整を行う上で，方策の「バランス」という概念を導入した手法[6] が有用であることが，囲碁において分かっている[6][4]．この手法の基本的なアイディアは，プレイアウトを繰り返すことで得られる平均報酬を，Minimax 値に近づけるように，方策を学習するというものである．

本稿では，この手法による方策の学習を，モンテカルロ将棋に適用することを提案する．実際に，1800 局面程度を用いて学習を行った．そして，ランダムに等確率で指し手を選ぶ方策をとったモンテカルロ法との対戦という，予備的評価を行った．しかし，利用した特徴数が多すぎたため，学習前よりも弱くなるという結果であった．

2. 関連研究

2.1 将棋へのモンテカルロ法適用

将棋においては，ランダムなシミュレーションでは，多くとも 200 手程度といった，現実的な手数で終局に至らせることは難しい．こうした点を考慮しつつ，一定の強さを得るために，プレイアウトの方策について言及した研究や，終局に至らずにプレイアウトを打ち切った場合の評価方法について言及した研究がある．

方策に言及した研究としては佐藤らのものがある[7]．

†1 東京大学工学系研究科
Graduate school of Engeneering, The University of Tokyo
{seki,chikayama}@logos.ic.i.u-tokyo.ac.jp
†2 マンチェスター大学コンピュータ科学科
School of Computer Science, University of Manchester
makoto.miwa@manchester.ac.uk

⋆1 ここでいう報酬とは，プレイアウト末端における局面の評価のことを指す．例えば，勝敗や静的評価値などである

この研究では，Elo レーティングを用いて，プレイアウトにおける指し手の確率的選択を行っている．次の一手問題では従来のアルファベータ法並みの正答率をあげている．一方で，実際の対局ではまだ従来のものに比べて弱い．また，欠点として，レーティングのために多くの特徴を見るため，シミュレーションに時間がかかるということがある．完全にランダムなシミュレーション[⋆1]に比べ，4 分の 1 程度の速度である．

こうした速度の問題を解決するため，より少ない特徴で終局率を上げた研究として，宇賀神らのものがある[9]．方策としては，遷移確率を用いたものと，それに best-of-n アルゴリズムを組み合わせたものとが提案されている．後者の方法では，256 手以内の終局率が最大で 9 割以上となっており，また速度はランダムな場合の 6 割程度に抑えられている．

プレイアウトを打ち切った場合の評価に言及したものとしては，竹内らの研究[8] がある．静的評価関数の利用を試みており，現局面での評価値と，シミュレーション末端での評価値とを比較し，閾値以上高くなっていれば勝ち，閾値以上低くなっていれば負けとしている．また，方策においても静的評価関数を利用し，評価値の高い 5 つの手の中からランダムに選択するという方法を提案している．

2.2 Monte-Carlo Simulation Balancing

プレイアウトにおける方策の学習方法として，Simulation Balancing という手法が提唱され[6]，囲碁においてその有用性が示されている．

Balancing における学習の目的は，局面 s において，方策 π_θ における期待報酬 $\mathbf{E}_{\pi_\theta}[z|s]$ と Minimax 値 $V^*(s)$ の二乗誤差を最小にする $\theta = \theta^*$ を求めることであり，式 1 のように表される．

$$\theta^* = \arg\min_\theta \mathbf{E}_\rho \left[(V^*(s) - \mathbf{E}_{\pi_\theta}[z|s])^2 \right] \qquad (1)$$

なお，真の Minimax 値 V^* を求めることは，現実的には不可能であるため，深いモンテカルロ木探索による[5] 近似値 $\hat{V}^*(s)$ を用い，$V^*(s) \approx \hat{V}^*(s)$ とする．

具体的な方策は，式 2 のようなソフトマックス方策となる．$\phi(s,a)$ は，局面 s における指し手 a についての特徴ベクトルを示し，θ は各特徴に対する重みのベクトルを示している．

$$\pi_\theta(s,a) = \frac{e^{\phi(s,a)^T\theta}}{\sum_b e^{\phi(s,b)^T\theta}} \qquad (2)$$

この式 2 に注目すると，指し手の選択は，重みの絶対値が大きくなるほど決定的になり，小さいほど確率的になることが分かる．よって，報酬を Minimax 値に近づける上で寄与の大きい特徴の重みの絶対値は大きくなり，寄与の小さい特徴のそれは小さくなる．こうすることで，前述の「現実的なシミュレーション」と「多様なシミュレーション」の間のトレードオフ関係のバランスを取ることができる．

このような調整を行うための，パラメータの具体的な更新手順をアルゴリズム 1 に示す．この手順は，現在の方策 π_θ による，M 回のプレイアウトによって得られる平均報酬 V を求める部分，N 回のプレイアウトによって得られる勾配 g を求める部分，そしてこの V,g と Minimax の近似値 $\hat{V}^*$ から，方策のパラメータ θ を更新する部分からなる．また，g の更新に用いられている $\psi(s,a)$ は，ソフトマックス方策の log の勾配であり，式 3 のように表される．なお，α は定数であり，T はプレイアウトの開始点から，終端までに指された手の数である．

$$\begin{aligned}\psi(s,a) &= \nabla_\theta \log \pi_\theta(s,a) \\ &= \phi(s,a) - \sum_b \pi_\theta(s,b)\phi(s,b)\end{aligned} \qquad (3)$$

Algorithm 1 Balancing におけるパラメータの更新手順

$\theta \leftarrow 0$
for all $s_i \in$ training set **do**
  $V \leftarrow 0$
  for $i = 1$ to M **do**
    simulate $(s_1, a_1, \ldots, s_T, a_T; z)$ using π_θ
    $V \leftarrow V + \frac{z}{M}$
  end for
  $g \leftarrow 0$
  for $j = 1$ to N **do**
    simulate $(s_1, a_1, \ldots, s_T, a_T; z)$ using π_θ
    $g \leftarrow g + \frac{z}{NT} \sum_{t=1}^{T} \psi(s_t, a_t)$
  end for
  $\theta \leftarrow \theta + \alpha(\hat{V}^*(s_1) - V)g$
end for

図 1 Balancing におけるパラメータの更新手順

3. 提案手法

本稿では，2.2 で述べた Balancing を，モンテカルロ将棋における方策の学習に用いることを提案する．この手法は，現局面から一手先の局面の勝率を比較するという，単純なモンテカルロ法で，深いモンテカルロ木探索を行った場合と同等の評価精度を得るというものである．これが達成できれば，対局においてモン

⋆1 256 手かかった場合はシミュレーションを打ち切る

テカルロ木を構成する必要がなくなり，並列化の面で大きな利点が得られる．

なお，前述のように，将棋におけるプレイアウトには，確率的な指し手選択では，現実的な手数で終局に至ることが難しいという問題がある．このため，プレイアウトの報酬としては，2.1 で述べた，静的評価値の差を利用する，竹内らの方法[8] を採用する．また，教師，すなわち Minimax 値の近似値 $\hat{V}^*$ を与えるプレイヤには，UCT を用いる．この教師におけるプレイアウトでは，遷移確率をもとに，ソフトマックス確率で指し手を選ぶものとする．すなわち，局面 s において，指し手 a が選ばれる確率は，式 4 となる．$r(s,a)$ は，局面 s における指し手 a の遷移確率を表し，A は定数である．

$$\pi_r(s,a) = \frac{e^{A*r(s,a)}}{\sum_b e^{A*r(s,b)}} \quad (4)$$

Balancing の学習や指し手の選択においては，実現確率で利用しているものと同じ特徴を利用する．

4. 評　　価

本研究では，UCT や上記の学習方法を激指[1] 上で実装した．したがって，利用する静的評価関数や，実現確率および利用する特徴は激指に準じる．学習におけるパラメータは，$M = 650, N = 500, \alpha = 50 \sim 300$ のように設定した．また，UCT には Progressive Widening[3] を実装し，プレイアウト回数は 1000 回とした．また，アルゴリズム 1 中のプレイアウト（simulation），教師である UCT 中でのプレイアウト共に，打ち切り深さは 20 とした．

こうした条件のもと，およそ 1800 局面を用いて学習を行った．1 局面につき，2 回続けて学習を行なっている．

強さを評価するため，ランダムに等確率で指し手を選ぶ方策をとったモンテカルロ法との対戦を行った．プレイアウト回数の一手あたりの総数は，Balancing を利用したプレイヤ，対照プレイヤ共に 10000 回[⋆1] とした．なお，静的評価値の絶対値が 4000 以上になった時点で打ち切り，勝敗とした．200 回対局した結果，学習を行ったものは 75 勝であり，負け越している．

5. 考　　察

学習の結果，学習を行なっていないプレイヤに勝ち越せなかった原因について考察する．

⋆1 一手あたりの評価に，100 ∼ 500 回程度のプレイアウトが割り当てられる

学習が十分に収束していないことが，原因として考えられる．学習の結果，$(\hat{V}^* - V)^2$ は図 2 のようになった[⋆2]．これを見ると，全体としてまだ学習が収束していないことが分かる．

また，収束が見られていない状態での結果ではあ

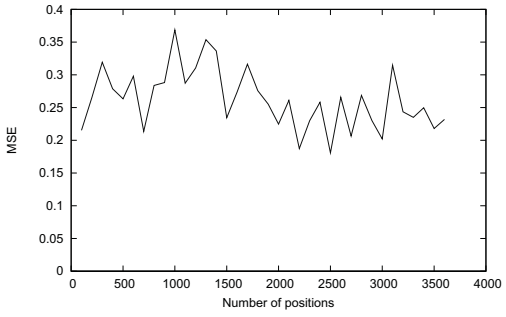

図 2　学習局面数に対する，$(\hat{V}^* - V)^2$ の変化

るが，各特徴の重みのヒストグラムは，図 3 のようになった．ただし，最も高い重みと低い重みは，それぞれ 11.8, −8.1 となっているが，図中では，これらを含む上位 1% 個 下位 1% 個の特徴は表示していない．特徴の重みが，0 付近に著しく偏っていることが分かる．ただし，こうした傾向自体は，囲碁における Balancing の学習においてもみられるものである[4]．

次に，学習により，$(\hat{V}^* - V)^2$ が収束していない原

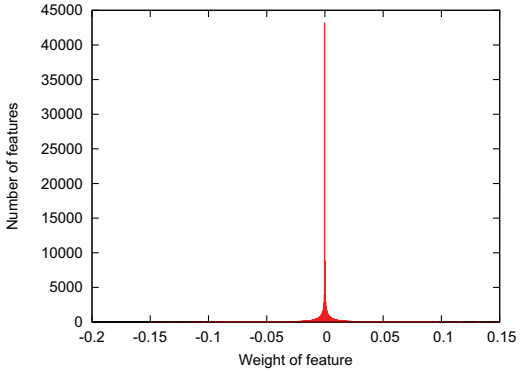

図 3　学習した重みのヒストグラム

因について考える．まず，学習局面が少なすぎるという点があげられる．本稿で用いた特徴数は，18 万強と多く，収束に必要な局面数が多いことが予想される．囲碁における Balancing では，利用している特徴は，107 個[6] や 2051 個[4] であり，本稿で利用した特徴数と

⋆2 あらゆる局面で，この値が 0 へ収束することが理想である

は大きく異なる．

他の要因としては，学習に用いた特徴の数が，収束に必要な程度と比べて，少ないということも考えられる．前述の 18 万強個の特徴は，実現確率の学習の結果，ある程度以上の重みが得られたものであり，指し手についてとれる特徴すべてではない．実現確率の学習において重みが低くなる特徴と，Balancing の学習において重みが低くなる特徴は，必ずしも一致しないはずである．そのため，より多くの特徴を使って学習をしなければ，学習局面数を増やしても，学習が収束しない可能性が考えられる．たとえば，アルゴリズム 1 において，$\alpha = 100$ とし，同一局面での学習を繰り返した場合，5 回程度で収束が見られた．一方で，図 2 のように，複数の局面で学習を行うと一向に収束が見られない．こうしたことからも，Balancing において重要な特徴が十分に利用できていないのではないか，ということが推測できる．

また，教師である UCT においては，遷移確率が低い，明らかに悪手と思われる手は，モンテカルロ木の構成や，プレイアウトの際に除かれている．一方，Balancing では，重みに従って，「すべての合法手の中から」指し手を選択している．将棋においては，明らかな悪手は評価値を大きく下げる方向に働くため，$(\hat{V}^* - V)^2$ の収束に影響を与えているのではないかと推測している．

6. おわりに

本稿では，Simulation Balancing をモンテカルロ将棋における方策の学習へ適用することを提案した．また，1800 局面程度での学習を行い，学習を行わないプレイヤとの対戦実験による評価を行ったものの，利用した特徴数が多すぎ，弱くなっているという結果であった．

5 で述べたように，今後，学習する局面数や利用する特徴の数を増やす必要がある．加えて，学習に関するパラメータを変化させ，収束性を調べたい．また，最大の重みを持った手に対し，一定以上重みの低い手は選択しない，とした場合についても学習を行う．「明らかな悪手」についても一定の学習が行うことができ，$(\hat{V}^* - V)^2$ の減少への寄与があると期待する．

収束が確認できれば，学習を行なっていないモンテカルロ将棋プレイヤや，教師，アルファベータを利用した通常の激指などとの対戦実験により，あらためて強さについての評価を行う．また，モンテカルロ将棋においては，プレイアウトの終局率が注目されることが多いため，この点についても評価を行う．

現状では教師として用いている UCT プレイヤ自体が，アルファベータ探索を用いたコンピュータプレイヤに比べて非常に弱いため，改良が必要である．そこで，現実的な時間内でより多くのプレイアウトを行うための，高速化・並列化が必須である．また，Killer Move[2] や枝刈り[11] など，木の扱いの工夫による，プレイアウトの効率的な割り当てといった改善方法も有効だと考えられる．

参考文献

1) : 将棋プログラム「激指」のページ，http://www.logos.ic.i.u-tokyo.ac.jp/gekisashi/.
2) Akl, S.G. and Newborn, M.M.: The principal continuation and the killer heuristic, *Proceedings of the 1977 annual conference*, ACM '77, New York, NY, USA, ACM, pp.466–473 (1977).
3) Chaslot, G., Winands, M., Herik, H., Uiterwijk, J. and Bouzy, B.: Progressive strategies for monte-carlo tree search, *New Mathematics and Natural Computation*, Vol.4, No.3, p.343 (2008).
4) Huang, S.-C., Coulom, R. and Lin, S.-S.: Monte-Carlo Simulation Balancing in Practice, *International Conference on Computers and Games* (2010).
5) Kocsis, L. and Szepesvári, C.: Bandit based monte-carlo planning, *Machine Learning: ECML 2006*, pp. 282–293 (2006).
6) Silver, D. and Tesauro, G.: Monte-Carlo simulation balancing, *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, New York, NY, USA, ACM, pp.945–952 (2009).
7) 佐藤佳州，高橋大介：モンテカルロ木探索によるコンピュータ将棋，ゲームプログラミングワークショップ 2008 論文集 (2008).
8) 竹内聖悟，金子知適，山口和紀：将棋における，評価関数を用いたモンテカルロ木探索，ゲームプログラミングワークショップ 2010 論文集 (2010).
9) 宇賀神拓也，小谷善行：モンテカルロ将棋における遷移確率を用いたプレイアウトの改良，ゲームプログラミングワークショップ 2009 論文集，pp. 107–110 (2009).
10) 橋本隼一，橋本　剛，長嶋　淳：コンピュータ将棋におけるモンテカルロ法の可能性，ゲームプログラミングワークショップ 2006 論文集，pp. 195–198 (2006).
11) 北川竜平，栗田哲平，近山　隆：投入計算量の有限性に基づく UCT 探索の枝刈り，ゲームプログラミングワークショップ 2008 論文集，pp.46–53 (2008).