情報処理学会論文誌 Vol.53 No.11 2533–2543 (Nov. 2012)

シミュレーション・バランシングを用いたモンテカルロ将棋の方策学習

関 栄二[1,a)] 三輪 誠[2,b)] 鶴岡 慶雅[1,c)] 近山 隆[1,d)]

受付日 2012年2月20日, 採録日 2012年9月10日

概要：モンテカルロ木探索（MCTS）の登場により，コンピュータ囲碁プレイヤの棋力は大幅に向上した．こうした成功を受け，静的評価関数とアルファベータ木探索による従来の手法が成功を収めてきた将棋においても，モンテカルロ法の適用が模索されている．MCTS の改良は，モンテカルロ木の扱いに関するものと，プレイアウトの方策に関するものに大別される．本稿では後者に着目し，モンテカルロ将棋の方策の学習に，囲碁で成功を収めているシミュレーション・バランシングを適用することを提案する．実際に，3,000 局面でのバランシング方策の学習を行った．対戦実験による評価の結果，提案手法の特に序中盤での有用性を示すことができた．また，バランシングを適用するうえで，詰み付近の局面における将棋特有の課題を解析し，プレイアウトに詰み探索を導入することで一定の改善が行えることを示した．

キーワード：コンピュータ将棋プレイヤ，モンテカルロ法，方策の学習

Policy Learning in Monte-Carlo Shogi Using Simulation Balancing

Eiji Seki[1,a)] Makoto Miwa[2,b)] Yoshimasa Tsuruoka[1,c)] Takashi Chikayama[1,d)]

Received: February 20, 2012, Accepted: September 10, 2012

Abstract: Since the advent of Monte-Carlo tree search (MCTS), strong computer players using Monte-Carlo methods have been built for the game of *go*. Following these successes, application of the methods has been explored to the game of shogi for which conventional methods have also been successful. Improvement efforts of MCTS can be roughly classified into two: the way to deal with Monte-Carlo tree and the simulation policy. In this paper, we propose to apply *simulation balancing* that has succeeded in *go* to learn the policy of Monte-Carlo shogi players. We use this learning method with 3,000 positions and evaluate the performance. The proposed method is found to be effective in opening and middle game. Then, we analyze a problem unique to shogi endgames and alleviate it by performing checkmate search in playout.

Keywords: computer shogi player, Monte-Carlo method, policy learning

1. はじめに

従来，強いコンピュータゲームプレイヤを作る研究においては，アルファベータ木探索と静的評価関数を組み合わせた方法がとられてきた．ここでの「木」とは，局面をノードとし，指し手を枝としたものである．また，静的評価関数とは，局面の有利・不利の程度を出力する関数である．こうした方法による研究は，チェスや将棋の分野で大きな成果をあげている．たとえば，チェスでは 1996 年に Deep Blue が人間の世界チャンピオンに勝利した．また，将棋ではアマトップレベルを凌ぐ強さが得られている．

しかし，囲碁の世界においては，従来の方法ではあまり成果が出てこなかった．適切な静的評価関数の設計が非常に難しいことが大きな要因としてあげられる．こうし

[1] 東京大学大学院工学系研究科
Graduate school of Engeneering, The University of Tokyo, Bunkyo, Tokyo 113–8656, Japan
[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, Manchester M13 9PL, UK
a) seki@logos.ic.i.u-tokyo.ac.jp
b) makoto.miwa@manchester.ac.uk
c) tsuruoka@logos.ic.i.u-tokyo.ac.jp
d) chikayama@logos.ic.i.u-tokyo.ac.jp

た中，モンテカルロ木探索（MCTS）[4] というまったく異なるアプローチによる囲碁のコンピュータプレイヤが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 [11], [13], [14], [15].

ゲームにおけるモンテカルロ法では，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）[*1]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プレイアウトを行う」ことと「多様なプレイアウトを行う」ことの 2 つの点を考慮しなければならない．前者は評価の良い手を高い確率で指す「強い」プレイヤを作ることであり，後者は極端には全候補手を一様な確率で指す「探索的な」プレイヤを作ることである．前者を考慮する代表的な手法としては，棋譜を教師とし，そこで指された手の尤度を最大化するような確率分布を方策として学習するというものがある．方策自体が棋譜と同じ手を指しやすい「強い」ものになれば，プレイアウトの質が良くなるはずだという考えに基づいている．しかし，この手法ではプレイアウトの多様性を考慮していないため，得られる手の評価には偏りが生じてしまう [9]．そのため，こうした偏りが生じにくい適切な多様性を獲得することが必要になる．

こうした調整を行ううえで，方策の「バランス」という概念を導入したシミュレーション・バランシング [9] が有用であることが，囲碁において分かっている [7], [9]．この手法の基本的なアイデアは，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．

一方，将棋におけるモンテカルロ法では，方策に言及した研究はある [11], [14] ものの，いずれもプロの棋譜を教師として，現実にありそうなプレイアウトを行う「強い」方策を作ることに重点を置いており，プレイアウトによる報酬の偏りをなくすための適切な多様性の考慮がなされていない．

本稿では，バランシング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．従来のモンテカルロ将棋で行われてきた，「強い」方策を作る学習手法との比較を通して，将棋におけるバランシングの課題を明らかにする．実際に 3,000 局面を用いて学習を行った．そして，ルートの次局面でのみ勝率を保持する単純なモンテカルロ法や，MCTS の一種である UCT（Upper Confidence bounds applied to Trees）[8] におけるプレイアウトにバランシングで学習した方策を適用して対戦実験を行った結果，特に序中盤での有用性を示すことができた．また，バランシングを適用するうえで，詰み付近の局面における将棋特有の課題を解析し，プレイアウトに詰み探索を導入することで一定の改善が行えることを示した．

[*1] 局面と指し手を変数とした確率分布．

2. 関連研究

2.1 ゲームにおけるモンテカルロ法

以下では簡単のため，終端で勝敗の決する 2 人有限確定ゼロ和ゲームを仮定する．ゲームにおけるモンテカルロ法では，次のような手順で指し手を選択する．

(1) それぞれの子ノードから，終端までランダムに指し手を選ぶシミュレーション（プレイアウト）を繰り返し，平均勝率を計算する．
(2) 最も勝率の高い手を指し手として選択する．

ただし，こうした原始的なモンテカルロ法においては，相手の悪手に期待しがちである，という点が大きな問題となる．たとえば，相手番において，明らかな好手（こちらの勝率を大きく下げる手）が 1 つだけあり，それ以外が悪手（こちらの勝率を上げる手）である状況を仮定する．このとき，単純なプレイアウトにおいては，すべての手が均等に選ばれるため，平均的な勝率は高く出てしまう．つまり，相手の明らかな好手を見逃し，悪手に期待していることになる．この一例を表したのが**図 1** である．青いノードは自身の手番を，赤いノードは相手の手番を示し，ノード内の数字は自身の勝率を示している．

こうした問題を解決しない限り，プレイアウト数を増やしても，平均勝率，すなわち「各指し手の良さ」をうまく近似することができない．この解決法には以下の 2 つの方向がある．

- 木を展開し，勝率を保持するノードを増やす（モンテカルロ木探索（MCTS）[4]）．
- プレイアウトにおける方策を改善する．

まず，モンテカルロ木探索について述べる．これは**図 2** のように，プレイアウトを繰り返す中で良さそうな子ノードを展開し，勝率を保持するノードを増やしていく手法である．ルートノードから再帰的に子ノードを選択していき，リーフノードに至ったらそこからプレイアウトを行う．そして，その報酬を親ノードへ伝搬させていく．こうすることで，たとえば図 1 において，相手の好手（こちらの勝率を 10%とする手）を発見することができる．

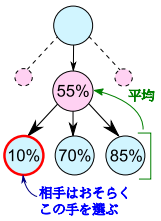


図 1 原始的なモンテカルロ法における，相手の悪手への期待の例
Fig. 1 An example of expecting the opponent to make a bad move in primitive Monte-Carlo methods.

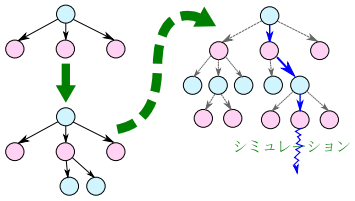


図 2 モンテカルロ木探索における木の拡張
Fig. 2 Tree expansion in Monte-Carlo Tree Search.

具体的な子ノードの選択方法，木の展開方法には以下のような方法がある．

子ノードの選択に関しては，UCT [8] が大きな成功を収めている．この手法では，式 (1) で表される，UCB 値 [2] の高い子ノードを選択する．

$$UCB(i) = \bar{X}_i + \alpha\sqrt{\frac{2\log N}{n_i}} \qquad (1)$$

ただし $\bar{X}_i$ は指し手 i を指した局面（子ノード）における，プレイアウトの繰返しで得られた勝率，n_i は手 i に対して割り当てられたプレイアウトの数，N は全子ノードの n_i の和を表している．また，α は係数である．結局，UCB 値は勝率とその誤差の和だということができる．この誤差は，割り当てたプレイアウト回数が増えるほど減っていく．したがって，UCT においては UCB 値を用いることで，基本的には現時点で勝率が高い有望な子ノードに多くのプレイアウトを割り当てつつ，まだ誤差が大きく，勝率の精度が悪い子ノードにもプレイアウトを割り当てることができる．

また，木の展開方法としては，すべての子ノードを 1 度に展開するのではなく，有望なノードを優先的に展開する Progressive Widening [3] がよく知られている．ほかにも，限られたプレイアウトを効率的に割り振るための枝刈り手法 [16] など，モンテカルロ木探索には様々な改良方法が提案されている．

次に，プレイアウトにおける方策について述べる．方策とは，局面と指し手を変数とした確率分布を表すものである．この方策へゲーム固有の知識[*2]を導入することで，モンテカルロ法によるゲームプレイヤの性能を大きく改善できることが知られている．これは，1 回 1 回のプレイアウトの精度が高まり，少ないプレイアウト数でもより正確な平均報酬を得ることができるためである．たとえば，図 1 においては，相手の好手が優先的に選ばれるようになれば，より正確な勝率が得られると期待できる．ただし，方策の改善にあたっては，得られる平均報酬に不適切な偏りが生じないよう，プレイアウトの多様性という点にも注意を払わなければならない．

なお，以上のモンテカルロ木探索の導入と方策の改善は，通常並行して行われるものである．すなわち，モンテカルロ木探索において保持している木の末端から行うプレイアウトにおいて，改善した方策を用いるということになる．

[*2] たとえば，囲碁であれば局所的なパターン [6].

2.2 将棋へのモンテカルロ法適用

将棋においては，一様な確率での手選択によるプレイアウトでは，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．これは，終盤に至っても手数が減らないがために，探索空間が非常に広い一方で，通常の対局においてありうる局面の空間は限られているためである．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 [11], [14] 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 [13] 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 [11]．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．これは，指し手の選択されやすさを指し手の特徴[*3]をもとに棋譜から学習し，得られた各特徴のレーティングを組み合わせて指し手を確率的に選ぶ手法である．この研究では，次の一手問題ではアマチュア初段程度の正答率をあげている．実際の対局ではまだ従来のものに比べて弱いが，アルファベータ木探索で時間内に探索できる以上の非常に深い読みを必要とする局面や，静的評価関数による評価の難しい局面など，一部の局面では従来の方法よりも良い結果を得ることに成功している．欠点として，レーティングの計算に多くの特徴を見るため，プレイアウトに時間がかかるということがある．完全にランダムなプレイアウト[*4]に比べ，4 分の 1 程度の速度である．モンテカルロ法の精度は回数にも依存するため，問題になりうる．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 [14]．方策としては，特徴量を 1 つだけ用いて計算される単純な遷移確率を利用したものが提案されている．この遷移確率とは，実戦棋譜における手の指されやすさを示すものであり，推定のためには，「王手をかける」や「直前に動いた駒の近く」などのある特徴を持った手が，教師データとなる棋譜において指される確率を用いる．さらに，遷移確率に best-of-n アルゴリズムを組み合わせたものも提案されている．この方法では，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 [13] 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，プレイアウトを打ち切った局面での評価値とを比較し，閾値以上高ければ勝ち，閾値以上低ければ負けとしている．なお，この静的評価関数を得る際には静止探索を用いている．静止探索とは，駒の取り合い局面で評価値が安定しないという問題を解決す

[*3] 駒を取る手，王手など．
[*4] 256 手かかった場合はプレイアウトを打ち切る．

るため，駒の取り合いが終わるまで，駒を取る手のみで局面を進めてから評価値を得るものである．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．

2.3 シミュレーション・バランシング

プレイアウトにおける方策の学習方法として，シミュレーション・バランシング（Simulation balancing）という手法が提唱され [9]，囲碁においてその有用性が示されている．

この手法は，方策のパラメータ θ を学習によって調整することで，プレイアウトを繰り返すことで得られる平均勝率を，minimax 値に近づけるというものである．2.1 節で説明したように，モンテカルロ法においては，得られた平均勝率の最も高い指し手を選択する．したがって，この平均勝率を真の勝率である[*5]minimax 値に近づけることができれば，つねに最善もしくはそれに近い指し手を選択できるようになる．これが，バランシングの基本的なアイデアとなっている．

バランシングにおける学習の目的について詳しく見る．これは，式 (2) のように表され，局面 s において，方策 π_θ によるプレイアウトの繰返しで得られる期待報酬 $\mathbf{E}_{\pi_\theta}[z|s]$ と minimax 値 $V^*(s)$ の二乗誤差を最小にする $\theta = \theta^*$ を求めることと説明できる．

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

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 [8] 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 (3) 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．

$$\pi_\theta(s,a) = \frac{e^{\phi(s,a)^T\theta}}{\sum_b e^{\phi(s,b)^T\theta}} \tag{3}$$

この式 (3) 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 値に近づけるうえ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．以上のようにして，特定の特徴を重視する現実味のあるプレイアウトを行い，かつ平均報酬が minimax 値に近づくような適切な多様性を獲得している．

このような調整を行うための，パラメータ θ の具体的な更新手順を**図 3** 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平均勾配 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．すなわち，平均報酬 V と minimax の近似値 $\hat{V}^*$ との比較から，報酬をどの程度増やすべきか，もしくは減らすべきかを決定し，そのためにどの重みをどの程度調整すればよいかを平均勾配 g によって決め，それらの情報をもとに，適当な学習率 α をかけて重みベクトル θ の更新を行うのである．なお，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 (4) のように表される．これは，プレイアウトのある局面における，実際に指された手の特徴ベクトルと，期待される特徴ベクトルとの差で計算される．なお，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．

$$\begin{aligned}\psi(s,a) &= \nabla_\theta \log \pi_\theta(s,a) \\ &= \phi(s,a) - \sum_b \pi_\theta(s,b)\phi(s,b)\end{aligned} \tag{4}$$

```
θ ← 0
for all s_i ∈ training set do
  V ← 0
  for i = 1 to M do
    simulate (s_1, a_1, ..., s_T, a_T; z) using π_θ
    V ← V + z/M
  end for
  g ← 0
  for j = 1 to N do
    simulate (s_1, a_1, ..., s_T, a_T; z) using π_θ
    g ← g + (z/NT) Σ_{t=1}^{T} ψ(s_t, a_t)
  end for
  θ ← θ + α(V̂*(s_1) − V)g
end for
```

図 3 シミュレーション・バランシングの更新アルゴリズム [9]
Fig. 3 Update algorithm in simulation balancing [9].

3. 提案手法

本稿では，2.3 節で述べたバランシングを，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．

囲碁においては，「それ自体が強い方策」（以下強い方策）と「モンテカルロプレイヤを強くする方策」は必ずしも一致しないことが指摘されている [5], [9]．これは強い方策が，たとえばプロの指し手を教師とするなどして，現実にありそうなプレイアウトを行うことを目指して作られている一方で，プレイアウトの多様性を十分に考慮していないために，得られる平均報酬に偏りが生じるためだと考えられる．このため，プレイアウトにおける 1 手 1 手の強さをいくらか犠牲にしてでも，平均報酬を minimax 値に近づけるような適切な多様性を学習したバランシングが，モンテカルロプレイヤを強くする方策としては優れていたといえる．

一方で従来のモンテカルロ将棋では，佐藤らの手法 [11]

[*5] 終端で勝敗の決する 2 人有限確定ゼロ和ゲームという仮定のもとでは，このようにいえる．

表 1 利用した代表的な特徴
Table 1 Representative features.

駒の種類
取り合い評価の結果
移動先の升目に味方と敵の駒がきいているかどうか
駒を取る手かどうか（取る駒の種類）
成る手かどうか
逃げる手かどうか
駒の利きに関する特徴
王手をかける手かどうか
あたりをかける手かどうか
ひもをつける手かどうか
相手の大駒をはじく手かどうか
自他玉との相対位置

や遷移方策を用いた手法 [14] など，プロの指し手を教師とした強い方策を作ることが目標となっていた．これらの手法では，前述のように平均報酬に不適切な偏りが生じる可能性がある．そのため，バランシングの適用によりモンテカルロ将棋プレイヤの棋力を改善できるのではないかと考えた．

なお，関連研究で述べたように，将棋におけるプレイアウトには，一様な確率での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，2.2 節で述べた，静的評価値の差を利用する竹内らの方法 [13] を採用する．このとき，勝ちを 1，負けを −1，引き分けを 0 として報酬を与える．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も，学習中の重みを用いた方策 π_θ に従うものとする．

将棋において，指し手の特徴に着目する代表的な手法としては遷移確率がある．そこで，バランシングの学習や指し手の選択においては，「駒の種類」や「王手をかける手か」などの遷移確率で利用しているものと同じ特徴を利用する．囲碁における既存研究と比較した場合，この特徴数が非常に多い，という点が大きな相違点である．既存研究においては，利用している特徴数は，107 個 [9] や 2,051 個 [7] である一方で，たとえば本稿で用いた「激指」[*6] であれば，18 万強個であり，代表的な特徴は**表 1** のようになっている[*7]．

4. 評価

本評価では，まず将棋におけるバランシングの学習の様子を示し，次に終局率と手選択確率の偏りの評価と対戦実験により，他の方策との比較を行う．

[*6] http://www.logos.t.u-tokyo.ac.jp/~gekisashi/
[*7] 詳細は文献 [12]「第 4 章『激指』の最近の改良について—コンピュータ将棋と機械学習—」を参照．

表 2 各方策における平均二乗誤差
Table 2 Mean Squared Error (MSE) of each policy.

↓UCT に用いる方策	一様	遷移	バランシング
バランシング	0.0603	0.0632	0.0227
各平均報酬を求めるために利用したものと同一の方策	0.0523	0.0525	0.0227

4.1 比較対象となる方策

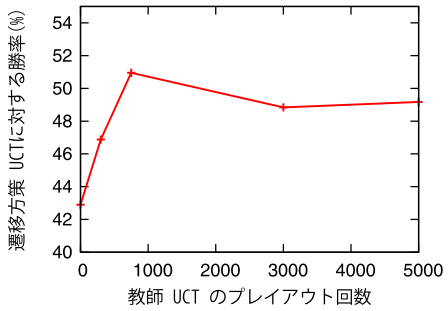
以下，候補手の中から等確率にランダムで手を選ぶ方策，遷移確率[*8]に基づく方策，シミュレーション・バランシングを適用した方策という 3 種類の方策を利用しているが，それぞれ「一様（Uniform）方策」，「遷移（Transition）方策」，「バランシング（Balancing）方策」と呼ぶこととする．いずれの方策についても，プレイアウト中の指し手と最終的な指し手ともに，遷移確率のごく低い手を除いた，遷移確率の高い上位 16 手のみを用いた．

ここでは，特に遷移方策について，比較対象としての妥当性を方策自体の強さと，平均報酬の偏りという 2 つの観点から検証する．

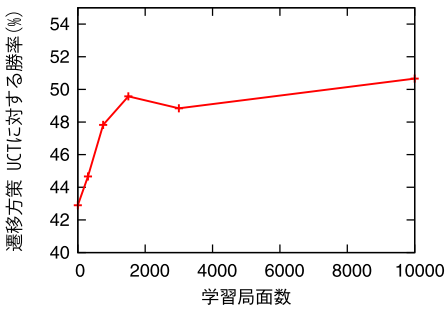
各方策に従った確率で候補手から指し手を選択するプレイヤを作成し，バランシング方策の強さを調査したところ，遷移方策に対しては 18.9%の勝率となった．これより，遷移方策がより現実味のある指し手選択を行う「強い」方策であることが分かる．さらに，バランシング方策と遷移方策は指し手について同じ特徴を見ていることからも，比較対象として十分妥当であるといえる．なお，一様方策に対しては 95.7%の勝率であり，一様方策に比べればはるかに現実味のある指し手選択を行えていることが分かる．

次に，平均報酬の偏りについて調べるために，UCT によって求めた minimax 値の近似値と，各方策によるプレイアウトの繰返しで得られる平均報酬との間の平均二乗誤差（MSE）を比較する．これには，プロの棋譜から，序盤局面を除いてランダムに抽出した 5,000 局面を用い，minimax 値の近似値を求めるためには 3,000 回の，平均報酬を求めるためには 1,000 回のプレイアウトを行う．結果は**表 2** のようになった．なお，UCT における方策としては，バランシング方策を用いた場合と，各平均報酬を求めるために利用したものと同一の方策を用いた場合の 2 つの場合を示している．いずれにおいても，バランシング方策は MSE が他の方策を大きく下回っており，平均報酬の偏りが少ないと考えられる．一方で，遷移方策における MSE は一様方策のものと比べてもほとんど変わらないか，上回っていることが分かる．したがって，平均報酬の偏りという観点ではバランシング方策が遷移方策に比べて優れていると考えられる．この点からも，遷移方策が比較対象として妥当

[*8] 「激指」では，16,000 棋譜，局面数にして 1,769,674 局面を利用して学習している．



(a) 教師 UCT のプレイアウト回数（The number of playouts of UCT in learning）



(b) 学習局面数（The number of positions used for learning）

図 4 様々なパラメータでの学習結果

Fig. 4 The result of learning with various values of parameters.

だといえる．

4.2 学習の設定

本研究では，UCT やバランシングによる学習を「激指」上で実装した．したがって，利用する静的評価関数や，遷移確率および利用する特徴は，「激指」で用いているものと同様である．学習におけるパラメータは，$M = 650$，$N = 500$，$\alpha = 1$ のように設定した．また，教師として用いる UCT には，方策 π_θ における選択確率の高い手を優先的に展開する Progressive Widening [3] を実装し，プレイアウト回数は 3,000 回とした．また，プレイアウトの打ち切り深さは 5 とした．これは，打ち切り深さを変えた遷移方策 UCT どうしでの対戦実験を行い，その中で良い結果を示した値を利用している．

こうした条件のもと，全特徴の重みの初期値を 0 として学習を行った．学習用の棋譜として，プロの棋譜から，序盤局面を除いてランダムに抽出した 3,000 局面を用意し，これを繰り返し用いて学習した．また，minimax 値と方策 π_θ による平均報酬との間の平均二乗誤差（MSE），すなわち $\mathbf{E}_\rho\left[(V^*(s) - \mathbf{E}_{\pi_\theta}[z|s])^2\right]$ の算出のためのテストセット ρ は，同様の手順で抽出した 5,000 局面とした．

なお，教師 UCT のプレイアウト回数や学習局面数については，プレイアウト回数を 5,000 回とした場合や，学習局面数を 10,000 局面とした場合でも，4.6.1 項で後述する対戦実験と同様の条件での対戦では，棋力の向上は見られなかった．実際に，これら 2 つのパラメータの一方を様々に変えて対戦実験を行った結果，比較対象の方策に対する勝率は**図 4** のように変化した．

また，CPU には，Intel Xeon CPU X5560 2.80 GHz を用いた．メモリ容量は約 24.7 GB である．

4.3 学習の様子

学習にともなう平均二乗誤差の推移は，**図 5** のようになった．横軸は，学習のために用意した 3,000 局面を何回繰り返して学習したかを示している．このグラフを見ると，学習はまだ収束していないように見えるが，後述の対戦実験では 4 回目の繰返しで得られた結果が最も良い性能を示していたため，学習プロセスを止めている．

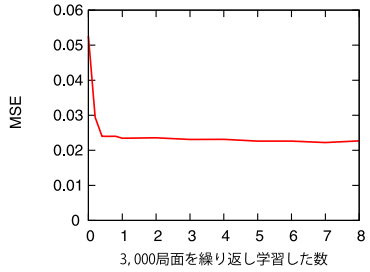


図 5 学習の進展による二乗誤差（MSE）の推移

Fig. 5 Mean Squared Error (MSE) with progress of learning.

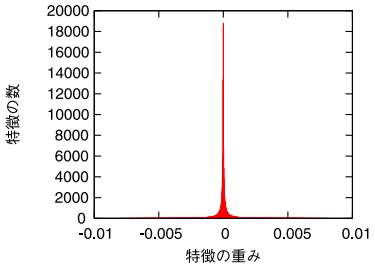


図 6 特徴量に対する重みのヒストグラム

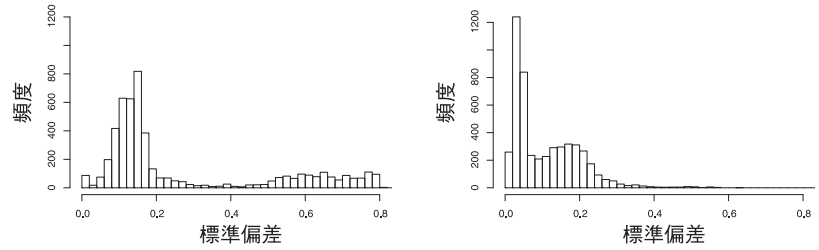
Fig. 6 Histogram of feature weights.

なお，3,000 局面を 1 回学習するために要した時間は約 3 時間 40 分であった．

各特徴量の重みのヒストグラムは，**図 6** 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 1.79，−0.30 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．こうした傾向は，囲碁におけるバランシングの学習においてもみられるものである [7]．

4.4 終局率の評価

今回の手法は，必ずしもプレイアウトにおける終局率の向上を目標とはしていないが，終局率は一般にモンテカルロ将棋において注目される点である．さらに，今後学習・



(a) 遷移方策（Transition policy） (b) バランシング方策（Balancing policy）

図 7 手選択確率の偏りの分布

Fig. 7 Standard deviation in move selection probability.

表 3 128 手以内の終局率

Table 3 Percentage of games finished in 128 moves.

方策	終局率
一様	97.7%
遷移	99.5%
バランシング	99.0%

対局両面においてプレイアウトの深さを増やしていくうえでも重要と考えられるため，この点についても評価を行った．

具体的には，学習セットの選択と同様の方法で抽出した50,000 局面のそれぞれからプレイアウトを開始し，1 手詰みを発見した場合にはその手を指すという 1 手詰み探索を利用しつつ，128 手以内に詰み局面に至った場合を「終局に至った」とした．結果を**表 3** に示す．方策ごとの違いがそれほど見られないが，これは，遷移確率により候補手を制限していることや，1 手詰み探索を行っていることが大きな要因となっている．たとえば，一様方策における終局率は，候補手の制限のみを利用した場合は 82.2%，1 手詰み探索のみを利用した場合は 56.1%，いずれも利用しない場合は 9.81%となる．

4.5 方策ごとの手選択確率の偏り

プレイアウト中における手選択の傾向を知るため，遷移方策とバランシング方策における手選択確率の偏り方を調べる．具体的には，局面 s における手選択確率の標準偏差 $B(s)$ を，候補手の数を m，方策 π における指し手 a_i の選択確率を $\pi(s, a_i)$ として，式 (5) のようにして求める．なお，$1/m$ は手選択確率の平均である．

$$B(s) = \sqrt{\sum_{i}^{m} \left(\pi(s, a_i) - \frac{1}{m} \right)^2} \tag{5}$$

これを，5,000 局面で求めた結果，各方策における $B(s)$ の分布は**図 7** のようになった．遷移方策に比べ，バランシング方策は手選択確率が極端に偏る局面が少ないことが分かる．したがって，バランシング方策は遷移方策に比べて，多様なプレイアウトを行う「探索的な」プレイヤであることが分かる．こうした性質は，有効手が多く，様々な手を探索する必要がある場面では有利に働くと考えられる．逆に，将棋に数多く存在する，有効な手が 1 つしかなく，それ以外の手がすべて悪手であるような局面では不利に働くと考えられる．特に，終盤には連続してそうした局面が続くことが多く，棋力を落とす原因になりうる．

4.6 対戦実験

4.6.1 モンテカルロプレイヤどうしの対戦

強さを評価するため，対戦実験を行った．大きく分けて 2 種類の対戦実験を行っており，1 つはルートの次の局面でのみ勝率を保持する単純なモンテカルロ法どうしの対戦，もう 1 つは UCT どうしの対戦である．それぞれ，一様方策，遷移方策に対する，バランシング方策の強さを評価した．

両実験に共通する設定について述べる．まず，プレイアウト数は，いずれのプレイヤにおいても 1,000 回固定とした．プレイアウト数によって棋力に大きな差が生じるものの，同じプレイアウト数どうしでの対局であれば，得られる結果に特別異なる傾向は見られないことが，プレイアウト回数を 3,000 回とした予備実験で分かったためである．また，プレイアウトの打ち切り深さは，学習時と同じく深さ 5 とした．さらに，プレイアウト中に 1 手詰みを見つけた場合は，手の選択確率にかかわらずその手を指すこととした．なお，1 手詰み探索の導入によりいずれの方策でも棋力の向上が見られたものの，バランシング方策の学習において導入した場合，棋力の劣化が見られたため対戦実験のみで用いた．以上の条件のもと，各 3,000 回の対戦実験を行った．

また，UCT に特有の設定について述べる．Progressive Widening では，遷移方策と一様方策においては遷移確率の高い手から順に展開し，バランシング方策においては，予備実験による調整の結果，バランシング方策において選択確率の高い手から順に展開することとした．展開条件は訪問回数のみに依存するものとし，遷移方策 UCT どうしでの予備実験の結果から，最初の 1 つの子ノードを 3 回目の訪問で展開し，以降は 4 回ごとの訪問で展開している．また，木の成長やプレイアウトに知識を導入している場合，ノードの初期報酬，訪問回数を適切に設定することで，棋

表 4 単純モンテカルロ法どうしの対局におけるバランス方策の勝率
Table 4 Winning rates of the balance policy against two simple Monte-Carlo players.

対戦相手の方策	一様	遷移
勝率	100%	85.2%

表 5 UCT どうしの対局におけるバランス方策の勝率
Table 5 Winning rates of the balance policy against two UCT players.

対戦相手の方策	一様	遷移
勝率	57.7%	48.8%

力が向上することが知られている [5]．本評価では初期報酬は 0[*9]とし，初期訪問回数は遷移方策 UCT どうしの対戦実験から求めた 5 回という値を用いた．なお，プレイアウトにおける 1 手詰み探索と同様の理由で，初期報酬，訪問回数は学習時には用いていない．

まず，単純なモンテカルロ法どうしでの結果を**表 4** に示す．これより，一様方策に対しても，遷移方策に対しても，大きく勝ち越していることが分かる．すなわち，バランシング方策はプレイアウトの多様性を「適切に」考慮することができているために，「強い」方策に対して優位に勝ち越すことができたといえる．

次に，UCT どうしでの対局の結果を**表 5** に示す．一様方策に対しては大きく勝ち越しているものの，遷移確率方策に対しては負け越している．

4.6.2 $\alpha\beta$ プレイヤとの対戦

バランシング方策 UCT がどの程度の棋力が得られたかを明らかにするため，$\alpha\beta$ 探索と静的評価関数による従来法との対戦実験を行った．比較対象としてはオリジナルの「激指」を用い，深さ 6 の探索を行うこととする．

UCT のプレイアウト回数を 1,000〜30,000 とし，それぞれ 1,000 回の対戦実験を行った結果，オリジナルの「激指」に対する勝率は**図 8** のようになった．なお，オリジナルの「激指」は序盤では定跡を用いて指し手を選択している．そこで，より正確にアルゴリズムどうしの比較を行うため，序盤局面を除いた場合の対戦[*10]結果も併記した．

バランシング方策が 1 秒間に行えるプレイアウト数が平均して 500 回程度である一方で，オリジナルの「激指」が深さ 6 の探索に要する時間は平均して 5〜10 ミリ秒程度である．よって，現状では提案手法を導入した場合でも，同等の思考時間では従来法に比べて大きく棋力が劣ることが分かる．ただし，図 8 の結果より，高速化による棋力の向上が期待できる．高速化の方法としては，たとえば特徴数の削減による指し手の精度と速度のトレードオフの調整や，並列化などが考えられる．

*9 平均報酬の範囲は [−1, 1].
*10 ランダムに選んだプロの棋譜を進行度（後述）が 32 を超えるまで読み込み，その局面から対戦を開始した．4.6.3 項で後述する特定の進行度間での対戦実験における，$m = 32$，$n = 128$ の場合と同様である．

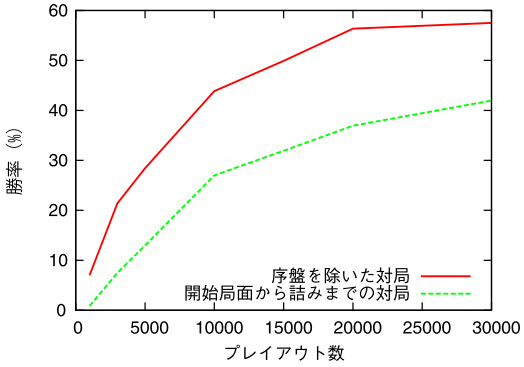


図 8 $\alpha\beta$ 探索によるコンピュータ将棋プレイヤとの対局
Fig. 8 Compare with the player using $\alpha\beta$ search.

表 6 特定の進行度間での対局結果
Table 6 Result during some degree of progress.

m	n	バランシング方策の勝率
0	96	55.7%
32	96	53.5%
96	128	45.7%

4.6.3 特定の進行度間での対戦実験

4.5 節における，序中盤と終盤での棋力差に関する予想を確かめるために，序中盤のみや終盤のみといった条件で対戦実験を行う．具体的には，以下のように特定の進行度 m〜n の間での対戦実験を行った．なお，進行度とは盤面上の駒の情報などから局面の進み具合いを数値化したものである．「激指」では 0〜127 の整数値で出力され，数値が大きいほど終盤に近いと判断している．

(1) ランダムに選んだプロの棋譜を，進行度が m 以上になるまで読み込む（$m = 0$ のときは初期局面からの対局）．
(2) 遷移方策 UCT 対バランシング方策 UCT で局面を進める．
(3) 進行度が n 以上になったら，両プレイヤを遷移方策 UCT に変えて詰みまで対局を行い，結果を見る（$n = 128$ のときは詰みまで遷移方策対バランシング方策）．
(4) (2) の先後を入れ替えて同様の対局を行う．

以上の条件で 3,000 回（先後の入れ替えがあるため，選んだ棋譜数は 1,500）の対局を行った結果，遷移方策に対するバランシング方策の勝率は**表 6** のようになった．$m = 0, 32$，$n = 96$ の結果より，序中盤では遷移方策に対して有意に良い指し方をしていることが分かる．一方で，$m = 96$，$n = 128$ の結果からは，終盤付近では遷移方策がより良く指していることが分かる．

表 7 手の絞り込みを行わない場合の遷移方策に対する勝率
Table 7 Winning rate against transition policy without refining moves.

単純モンテカルロ法どうし	UCT どうし
12.5%	38.5%

表 8 各種条件下での単純モンテカルロ法どうしでの対戦実験
Table 8 Winning rate of the simple Monte-Carlo player in diverse condition.

条件 1	20.9%
条件 2	44.3%
条件 3	58.2%

4.6.4 遷移確率による絞り込みを行わない対戦実験

4.1 節で述べたように，学習と対戦実験ともにすべての方策で遷移確率の高い上位 16 手のみを用いるように，手の絞り込みを行った．ここでは絞り込みの影響を調べるため，全方策について，そうした絞り込みを行わない場合の結果について示す．

まず，バランシング方策の重みとしてここまでと同じもの，すなわち手の絞り込みを行って学習を行ったもの，を用いた場合の遷移方策に対する結果を，**表 7** に示す．

この結果より，手の絞り込みを行った場合と比べ，勝率が下がっていることが分かる．特に，単純モンテカルロ法どうしの対局では勝率が顕著に下がっている．この原因としては，次のようなものが考えられる．

- 学習時には絞り込みを行っているため，絞り込みを行わないプレイアウトでは，平均報酬に偏りが出てしまう．
- 4.5 節に示したように，遷移方策は手選択確率が著しく偏る場面がある．実際，プレイアウトを繰り返してもせいぜい 2，3 手程度しか選ばれていない局面がしばしば出現する．すなわち，事前の絞り込みの有無にかかわらず，手選択の傾向に変化がない局面があることになる．特に，こうした局面は終盤に多く出現すると考えられ，バランシング方策は絞り込みの有無による影響を相対的に強く受けることになる．
- 候補手が増えたことで，より一様にプレイアウトを割り振るバランシング方策は，少ないプレイアウト数では不利になる．

以上の予想を確かめるため，以下の 3 つの条件で単純モンテカルロ法どうしの対戦実験を行う．

条件 1 手の絞り込みを行わずに学習した結果を用いて，対戦実験を行う．

条件 2 条件 1 に加え，4.6.3 項と同様に，終盤を除いた特定の進行度間 $m = 0$，$n = 96$ での対戦実験を行う．

条件 3 条件 2 に加え，両プレイヤのプレイアウト回数を 5,000 回に増やして対戦実験を行う．

各条件下における単純モンテカルロ法どうしでの対戦実験の結果，**表 8** のようになった．まず条件 1 について，MSE を調べたところ再学習後には 0.410 から 0.183 へと改善されており，平均報酬の偏りが減少したことが棋力の向上につながったと考えられる．次に条件 2 の結果より，バランシング方策は終盤付近で絞り込みの有無による影響を強く受けることが確かめられた．さらに条件 3 の結果より，候補手が増加すると，より一様にプレイアウトを割り振るバランシング方策で十分な棋力を得るためには，多くのプレイアウトが必要であることが確かめられた．

また，手の絞り込みによる棋力の向上はこれらの改善策を上回っており，手の絞り込みはバランシング方策における以上の弱点を改善するうえで有効な手段であったと考えることができる．

5. 考察

4.6.1 項で示した遷移方策とバランシング方策の対局において UCT どうしで負け越す原因は，4.6.3 項より，バランシング方策が終盤付近において弱いためだと考えられる．また，4.6.4 項からも，バランシング方策が終盤付近に課題を持つことが分かる．このため，バランシング方策の終盤付近における問題点を考察する．

プレイアウト中の 1 手詰み探索のない対局条件では，遷移方策 UCT に対する勝率が 43.0%に下がる．逆にいえば，1 手詰み探索の導入による棋力への影響はバランシング方策の方が大きい．したがって，両方策には特に詰み付近に大きな違いがあるのではないかと考え，その点について調査を行うこととした．

まず，プレイアウトの 1 手詰み局面への至りやすさについて調べた．具体的には，プロの棋譜からランダムに抽出した進行度 96 以降の約 15,000 局面からプレイアウト（128 手で打ち切り）を開始し，何手で 1 手詰み局面に至るかを調べた．この結果を**図 9** に示す．縦軸は n 手「以内」に 1 手詰み局面に至った局面数を表している．この図より，遷移方策の方がより短い手数で 1 手詰み局面に至ることが多いと分かる．すなわち，バランシング方策は相対的に「終盤付近で 1 手詰み局面へ至るのが遅い」といえる．これが終盤付近での弱さにつながっている可能性がある．

次に，実際に詰み手順に入ったときに，どの程度正しく指せるかについて調べた．具体的には，プレイアウト中に 3 手詰みの局面に至った場合に，どの程度の割合で詰み手順の最初の 1 手を正しく指せるかを調べた．これを，各方策ごとに約 20,000 局面[*11]からのプレイアウトで実行し

[*11] 実際には 50,000 局面からプレイアウトを行ったが，いずれの方策も 3 手詰みの局面に至ったのは約 20,000 局面であった．99%以上の局面は 128 手以内に 1 手詰み局面に至るため，およそ 6 割のプレイアウトは，先手後手いずれかが自ら 1 手詰み局面に至るような自殺手を指したことになる．

表 9 進行度別の一致率
Table 9 Agreement rates at ranges of degrees of progress.

	方策の選択確率に従って手を選ぶプレイヤ		単純モンテカルロ		UCT	
↓ 進行度	遷移	バランシング	遷移	バランシング	遷移	バランシング
0～31	7.58	6.48	18.0	23.0	23.4	23.7
32～63	9.03	10.1	26.2	32.4	34.0	35.5
64～95	9.47	11.1	28.0	34.1	37.6	38.6
96～127	15.0	15.1	30.4	36.1	38.2	38.1

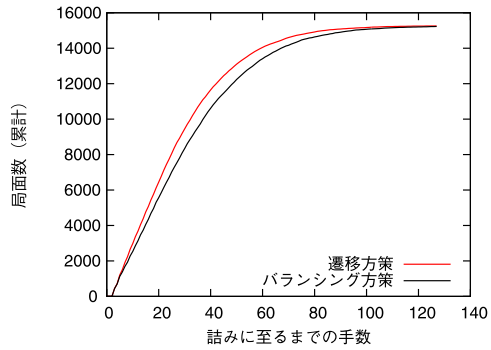


図 9 終局までの手数
Fig. 9 Number of moves to checkmate.

た結果，バランシング方策は 7.65%の割合で，遷移方策は 15.1%の割合で正しく指せることが分かった．こうした詰み手順での不正確さは，終盤付近の弱さに直結する要素だと考えられる．実際に，プレイアウトに 3 手詰み探索を導入したところ，初手から詰みまでの対局条件では，遷移方策 UCT に対する勝率は 51.9%となり，勝ち越すことができた．ただし，進行度 96～128 における対局の勝率は 48.2%であり，依然として負け越している．

なお，棋譜の指し手との一致率を進行度別に調べたところ，**表 9** のような結果になったが，遷移方策と比較したとき，バランシング方策が終盤付近で序中盤よりも一致率が悪化する，という傾向は特に見られない．したがって終盤においても，平均的には遷移方策と同程度かそれ以上に良い手を選んでいるものの，前述の詰み付近やその他のより勝率への寄与が大きい局面で，悪手を選んでいるのではないかと推測できる．

6. おわりに

本稿では，シミュレーション・バランシングを，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．実際に 3,000 局面を用いての学習を行った．そして，対戦実験による遷移方策との比較を行い，方策自体が弱くても単純なモンテカルロ法の棋力を大きく上げる，バランスのとれた方策を作成できることを示した．また，より強力なプレイヤである UCT のプレイアウトへもバランシング方策を導入し，特に序中盤における有用性を示した．そして，将棋においては，バランシングを単純に適用するだけでは終盤，特に詰み付近の局面に問題があることを明らかにし，プレイアウトにより深い詰み探索を導入することで，一定の改善が行えることを示した．

しかし，そもそも詰み探索の成功する局面に適切に近づく可能性が低いという問題があり，詰み探索を深くするだけでは改善は頭打ちになると考えられる．したがって，プレイアウトにおいて詰みを考慮する方法を今後は考えたい．なお，モンテカルロ木中において詰みを扱う手法としては，MCTS-Solver [10] が Lines of Action において提案されている．

また，現状では教師として用いている UCT プレイヤ自体が，従来のコンピュータ将棋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・並列化が必要である．また，Killer Move [1] や枝刈り [16] など，木の扱いの工夫による，プレイアウトの効率的な割当てといった改善方法も有効だと考えられる．

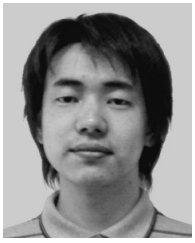
参考文献

[1] Akl, S.G. and Newborn, M.M.: The principal continuation and the killer heuristic, *Proc. 1977 Annual Conference, ACM '77*, pp.466–473, ACM, New York, NY, USA (1977).
[2] Auer, P., Cesa-Bianchi, N. and Fischer, P.: Finite-time analysis of the multiarmed bandit problem, *Machine Learning*, Vol.47, No.2, pp.235–256 (2002).
[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, pp.343–357 (2008).
[4] Coulom, R.: Efficient selectivity and backup operators in Monte-Carlo tree search, *Proc. Computers and Games 2006*, pp.72–83, Springer-Verlag (2006).
[5] Gelly, S. and Silver, D.: Combining online and offline knowledge in UCT, *Proc. 24th International Conference on Machine Learning, ICML '07*, pp.273–280, ACM, New York, NY, USA (online), DOI: http://doi.acm.org/10.1145/1273496.1273531 (2007).
[6] Gelly, S., Wang, Y., Munos, R. and Teytaud, O.: Modification of UCT with Patterns in Monte-Carlo Go, Rapport de recherche RR-6062, INRIA (2006).
[7] Huang, S.-C., Coulom, R. and Lin, S.-S.: Monte-Carlo Simulation Balancing in Practice, *International Conference on Computers and Games*, pp.81–92 (2010).
[8] Kocsis, L. and Szepesvári, C.: Bandit based monte-carlo

planning, *Machine Learning: ECML 2006*, pp.282–293 (2006).
[9] Silver, D. and Tesauro, G.: Monte-Carlo simulation balancing, *Proc. 26th Annual International Conference on Machine Learning, ICML '09*, pp.945–952, ACM, New York, NY, USA (2009).
[10] Winands, M.H., Björnsson, Y. and Saito, J.-T.: Monte-Carlo Tree Search Solver, *Proc. 6th International Conference on Computers and Games, CG '08*, Berlin, Heidelberg, pp.25–36, Springer-Verlag (2008).
[11] 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，No.11, pp.1–8 (2008).
[12] 松原　仁：コンピュータ将棋の進歩 6—プロ棋士に並ぶ，共立出版 (2012).
[13] 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，No.12, pp.86–89 (2010).
[14] 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).
[15] 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).
[16] 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).

関 栄二　（学生会員）

東京大学大学院工学系研究科電気系工学専攻修士 2 年．



三輪 誠　（正会員）

2008 年 3 月東京大学大学院博士課程修了．博士（科学）．同年 4 月より東京大学大学院情報理工学系研究科特任研究員を経て，2011 年 4 月より英国マンチェスター大学コンピュータ科学科リサーチアソシエイト．ACL，言語処理学会，人工知能学会各会員．自然言語処理，機械学習，コンピュータゲームプレイヤに興味を持つ．

鶴岡 慶雅　（正会員）

2002 年東京大学大学院博士課程修了．博士（工学）．2002 年科学技術振興事業団研究員．2006 年英国マンチェスター大学研究員．2009 年北陸先端科学技術大学院大学准教授．2011 年より東京大学大学院工学系研究科准教授．機械学習を用いた自然言語処理，テキストマイニング，ゲーム AI 等に関する研究に従事．

近山 隆　（正会員）

1977 年東京大学工学部卒業，1982 年同大学大学院工学系研究科博士課程修了．工学博士．同年より（財）新世代コンピュータ技術開発機構において第五世代コンピュータプロジェクトの研究開発に従事．1995 年東京大学工学系研究科助教授，1996 年同教授．以降，学内の異動を経て，2008 年 4 月より工学系研究科電気系工学専攻教授．