修　　士　　論　　文

将棋におけるモンテカルロ木探索の特性の解明にもとづいた方策の学習手法の提案

A New Learning Method for Policies Based on Analysis of the Monte-Carlo Tree Search in Shogi

指導教員　　**近山 隆 教授**

東京大学工学系研究科
電気系工学専攻

氏 名　　**37-116454 関 栄二**

提 出 日　　**平成 25 年 2 月 6 日**

概 要

モンテカルロ木探索は 2006 年の登場以降，囲碁を中心として大きな成功を収め，ゲーム・非ゲームを問わず様々な応用が模索されている．一方で，モンテカルロ木探索には未解決の課題も数多く，適用範囲の拡大の中で問題になっていくと考えられる．本研究ではその中でも，木探索を行う上でどのようなシミュレーションが有効であるかが不明確な点と，モンテカルロ木探索自体が従来のミニマックス探索と比べ何を得意とし不得意とするのかが不明確な点に着目する．このため，複数のシミュレーション方策の比較や，チェスや将棋で成果を挙げているミニマックス探索との比較を通じた，モンテカルロ木探索の特性の解明を目的とする．さらに，その結果をもとに新たなシミュレーション方策の学習手法の提案を行う．解析においては，異なる性質を持った二種類の方策の得失を将棋において明らかにし，ミニマックス探索との比較ではモンテカルロ木探索が最善手の「明確な」局面を苦手とすることを明らかにした．学習手法の提案において，将棋では従来手法と同程度の性能にとどまったものの，両方策の利点を共に有するような方策を学習することができた．

目 次

図 目 次

表 目 次

アルゴリズム，擬似コード

第1章　序論

1.1　背景と目的

人工知能の分野は現実の問題における人間の判断を代替する，もしくは支援することを一つの目的として発展してきた．そうした中でコンピュータゲームプレイヤの研究は，チェスや将棋など明示的なルールを持つゲームを対象とし，人工知能分野における試金石としての役割を果たしてきた．従来，優れたコンピュータゲームプレイヤを作成するための中心的な手法は，局面の良し悪しを示す静的評価値とそれにもとづいたミニマックスを始めとする探索（以降ミニマックス探索と呼ぶ）によるものであり，チェスや将棋など多くのゲームで成果を収めてきた．この手法では精度の良い静的評価値の推定が重要だが，それにはゲームに依存した多くの知識が必要とされる．

しかし，例えば囲碁においては局面から十分な知識を抽出することができず，静的評価値を用いた手法は十分な成果を納めてこなかった．そのため，ゲームに依存した事前の知識の代わりに，確率的なシミュレーションによって局面を評価するモンテカルロ法の適用が模索されていた．その中で，確率的なシミュレーションだけでなく，その結果にもとづいた決定的な木探索を行うモンテカルロ木探索 [8] が 2006 年に登場し，コンピュータ囲碁プレイヤは飛躍的に棋力を増すこととなった．囲碁における成功に加え，特定のゲームなど文脈に依存した知識を必要としない応用範囲の広さから大きな注目を集め，ゲーム・非ゲームを問わず様々な応用が模索されている [4].

モンテカルロ木探索の改善は，シミュレーションをいかに行うかというシミュレーション方策の改善と，そこで得られた知識をどのように扱うかという木探索の改善とに大別される．前者については様々なヒューリスティック，学習手法が提案されており，後者についても様々な数学的解析や改善方法の提案がなされている．しかし，どのような方策が木探索部，もしくはモンテカルロ木探索全体にどのような影響を与えるのかは十分に示されていない.「賢い」方策によるシミュレーションを行うことで，より強いプレイヤを作成できることが経験的に知られるにとどまっている．

そこで本研究では，どのような「賢さ」がモンテカルロ木探索にどのような影響を与えるのかの解明を目的とする．また，そもそもモンテカルロ木探索がミニマックス探索と比較したときに，具体的にどういった局面を得意・不得意とするのかは十分に明らかにされていない．そのため，その比較を通じたモンテカルロ木探索自体の特性の解明も目的とする．この際に，モンテカルロ木探索において複数の方策を用いることで，方策間の違いもより明瞭になるものと期待できる．最後に，以上の解析結果にもとづいて方策の学習手法の提案を行う．

具体的なゲームとしては将棋を用いる．将棋は，チェスと並んでモンテカルロ木探索が良い性能を出せていないゲームの代表例であり，ミニマックス探索に大きく劣る棋力しか得られていない．このため，従来モンテカルロ木探索研究の中心であった囲碁には現れにくい不得手な点が現れると期待できる．さらに，ミニマックス探索による手法が非常に発展しているゲームであり，ミニマックス探索との比較を通した解析を，広くかつ容易に行うことができると考えられる．

1.2 本研究の成果

シミュレーション方策の比較においては，遷移方策とバランシング方策という大きく異なる性質を持った方策の比較を行なった．前者はシミュレーション中の一手一手の精度の向上を図ったものであり，後者はシミュレーションの繰り返しで得られる局面の評価値を真の値に近づけるように学習したものである．対戦実験により，遷移方策がモンテカルロ木探索において高い性能を示すことを明らかにし，囲碁とは異なる結果が得られた．また，バランシング方策が相対的に序中盤を得意とする一方で，終盤を苦手とすることを明らかにした．

モンテカルロ木探索とミニマックス探索との比較では，モンテカルロ木探索が終盤を苦手としていることを明らかにした．モンテカルロ木探索は必敗手を避けることが苦手なことがチェスにおいて指摘されてきた [16]．本研究ではそうした詰み局面に限らないより一般的な解析として，局面の「明確さ」に着目した解析を行なった．その結果，モンテカルロ木探索が，終盤の最善手が「明確な」局面で間違えやすいことが明らかになった．これは将棋におけるモンテカルロ木探索の弱さにつながっていると考えられる．

遷移方策とバランシング方策それぞれに得失があるという解析の結果から，前者の目指す一手一手の精度と後者の目指す評価値の偏りの少なさを，ともに考慮した学習手法の提案を行った．モンテカルロ木探索における実用的な性能は遷移方策と同程度にとどまったものの，バランシング方策との比較においては，バランシング方策の利点を損なわずに性能を向上させることに成功している．このため，モンテカルロ木探索に導入するヒューリスティックとの相性や，対象とするゲーム次第では十分に有用な学習手法となり得ると言える．

1.3 本論文の構成

2 章ではゲームにおける探索手法一般や，モンテカルロ木探索の詳細，モンテカルロ木探索についての解析など関連研究について述べる．3 章では，実際に将棋においてモンテカルロ木探索の解析を行う．ここでの解析は，複数のシミュレーション方策の特徴についての解析と，ミニマックス探索との比較を通した解析とに分かれる．4 章では，解析結果にもとづいて方策の学習手法の提案と評価を行う．5 章では，本研究の結論と今後の課題について述べる．

第2章　関連研究

2.1　コンピュータゲームプレイヤにおける探索手法

以下では簡単のため，終端で勝敗の決する二人有限確定ゼロ和ゲームを仮定する．

2.1.1　ミニマックス探索

コンピュータゲームプレイヤの研究においては，状態空間が大きく，状態をすべて展開して解析するだけの計算資源の確保が事実上不可能なゲームを対象にするのが一般的である．こうしたゲームにおいては，各指し手を不完全な形で評価して手を選択する必要がある．この目的で静的評価関数と木探索を組み合わせたミニマックス探索が広く用いられてきた．静的評価関数とは局面の良し悪しを数値化して出力する関数であり，出力された値を静的評価値という．一般に，十分に精度の高い静的評価関数，すなわち単に次局面の静的評価値を高くする手を選ぶだけで十分強いプレイヤとなるほどの関数，を設計するのは困難である．そこで，n 手先の評価値を高くするような探索を組み合わせることで，その精度を大きく改善することができる．深さ 2 の探索の例を示したのが図 2.1 である．末端で静的評価関数により先手（青いノード）にとっての評価値を求め，後手（赤いノード）は評価値が低いノードを，先手は探索の評価値が高いノードを選ぶよう再帰的に探索を行う．最終的にプレイヤはこの探索による評価値が高い左の手を選ぶ．

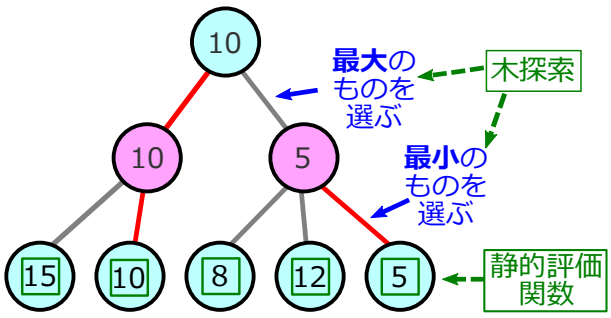


図 2.1: ミニマックス探索

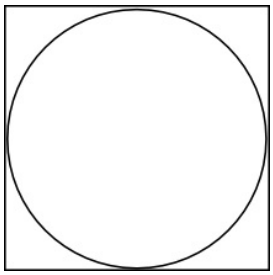


図 2.2: モンテカルロ法による円周率の近似的な計算

このミニマックス探索をもとに様々な改良を行った手法（以降総称してミニマックス探索と呼ぶ）は広く成功を納めてきた．よく知られたゲームとしては，チェスではすでに人間のトップをしのぐプレイヤが作られており，また将棋においてもアマのトップをしのぎ，プロに迫るプレイヤが作られている．

一方で，こうしたミニマックス探索が成果を上げてこなかったゲームもあり，囲碁はその代表的な例である．囲碁において性能が出ない主な理由として，精度の高い静的評価関数の設計が非常に難しいことがあげられる．これは，駒自体の評価値を利用することができない，盤面が広く位置による特徴が少ない，数十手先まで良し悪しの判断が難しい指し手が多々ある，などといったことに起因する．加えて，チェスや将棋と比べて大きな状態空間の影響もあり，木探索を組み合わせても十分な精度を得られない．結果として，ミニマックス探索による囲碁プレイヤは弱いアマチュアプレイヤ程度の棋力しか得られず，現在でも 5 級程度の強さにとどまっている [11]．

2.1.2 モンテカルロ法

ミニマックス探索とは大きく異なる考え方で局面（指し手）の評価を行う手法に，モンテカルロ法を利用したものがある．一般にモンテカルロ法とは，ランダムなシミュレーションの繰り返しにより，近似値を得る手法のことを指す．こうした一般的なモンテカルロ法の有名な例としては，円周率を近似するものがある．まず，図 2.2 のように，一つの円とそれに外接する正方形とを考える．次に，正方形内部にランダムに点を与え，円の内部にあるかを判定する．この操作を N 回繰り返す．すると，円の内部にあった点の数を n 個として，円周率 π の近似値は次のように計算できる．

$$\pi \approx \frac{4n}{N} \tag{2.1}$$

なお，大数の法則により，試行回数 N が増えるほど，近似の精度が高まる点は注目に値する．

さて，ゲームにおけるモンテカルロ法で近似するものは，局面もしくは指し手の評価値である．そして，最も評価の高くなる行動を選択するのである．具体的には，次のような手順で指し手を選択す

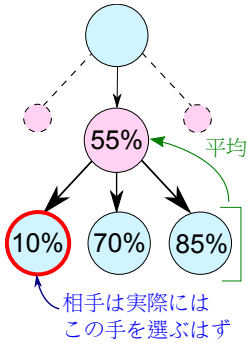


図 2.3: 原始的なモンテカルロ法において相手の悪手に期待してしまう例

る [1].

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．例えば，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）が一つだけあり，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 2.3 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二つの方向がある．

- 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)[8]）
- 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 2.4 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，例えば図 2.3 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．以下，子ノードの選択と木の展開における具体的な手法について述べる．

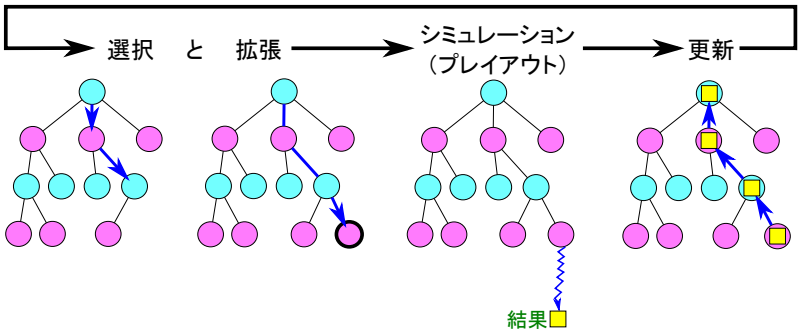


図 2.4: モンテカルロ木探索（Monte-Carlo Tree Search）

子ノードの選択に関しては，UCT（Upper Confidence bound applied to Trees）[14] が大きな成功を収めている．この手法では，式 (2.5) で表される，UCB（Upper Confidence Bound）値 [2] の高い子ノードを選択する．

$$UCB(i) = \bar{X}_i + C\sqrt{\frac{\log N}{n_i}} \tag{2.2}$$

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，プレイアウトの繰り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，C は係数である．UCB 値は勝率とその不確かさの和だということができる．この不確かさは，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，まだ不確かさが大きく，勝率の上がる余地の大き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一度に展開するのではなく，事前の知識から推定した有望さが高い順にノードを展開する Progressive Widening[6] がよく知られている．他にも，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 [27] 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[1]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，一回一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．例えば，図 2.3 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．

[1]例えば，囲碁であれば局所的なパターン [12]

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．これらの改良により，囲碁においてはすでに 4 段程度の棋力が得られており，9×9 の小サイズの囲碁ではトッププロをしのぐまでに至っている [11].

2.2 UCT

2.1.2 節で触れた UCT について詳しく述べる．

UCT はモンテカルロ木探索における子ノードの選択に関する手法であるが，一般にこの選択においては，搾取（exploitation）と探索（exploration）のジレンマを考慮する必要がある．これは，それまでの知識から有望と考えられるノードを選択すること（搾取）と，まだ十分な知識がなく今後有望になる可能性のあるノードを選択すること（探索）の間にあるジレンマである．モンテカルロ木探索においては，このジレンマを考慮した UCT が大きな成功を収めている．UCT では各子ノードを，搾取と探索のジレンマの典型例である多腕バンディット問題における各腕とみなす．

多腕バンディットとは K 台のスロットマシンのことであり，それぞれ独立で固有な一定の確率分布に従って当たり（簡単のため当たりは 1，外れは 0 とする）を出す．そして，多腕バンディット問題は K 台のマシン（腕）から 1 台を選んで報酬を得る行動を繰り返したときに，式 2.3 で表されるリグレット（regret，期待損失）が最小化される方策（行動指針）を求める問題である．なお，事前には各マシンの確率分布についての知識はなく，行動を繰り返す中で得られた知識をもとに次の行動を選択する．

$$\mu^* N - \mu_j \sum_i^K \mathbb{E}[T_i(N)] \quad where \quad \mu^* \stackrel{\text{def}}{=} \max_{1 \leq i \leq K} \mu_i \tag{2.3}$$

N は行動の回数，μ_i はマシン i に割り振られた当たりの確率（期待報酬），$T_i(N)$ はマシン i を選んだ回数を示す．したがってリグレットは，方策が常に最善のマシンを選べないことから生じる損失の期待値だといえる．このリグレットの増え方は少なくとも $O(\ln n)$ のオーダーになることが知られている [15]．リグレットの増え方が $O(\ln n)$ に従い，かつ実装が簡易で計算コストが小さい方策として，2.4 で表される UCB（Upper Confidence Bound）値が最大となるマシン i を選ぶ方策がある [2].

$$UCB(i) = \bar{X}_i + \sqrt{\frac{2 \log N}{n_i}} \tag{2.4}$$

$\bar{X}_i$ はマシン i から得られた平均の報酬，n_i はマシン i を選択した回数を表している．結局，UCB 値は平均報酬とその不確かさの和だということができる．この不確かさは，選択した回数が増えるほど減り，逆に選ばれなかった場合には増える．したがって，UCB 値を用いることで，基本的には

現時点で平均報酬が高い有望なマシンを多く選びつつ，まだ不確かさが大きく，平均報酬の精度が悪いマシンも選択することができる．

UCT では，前述のように各子ノードを多腕バンディット問題における一つのマシンとみなし，式 2.4 の探索項である第 2 項に係数 C を乗じた式 2.5 が最大となる子ノードを選択する．

$$UCB(i) = \bar{X}_i + C\sqrt{\frac{\log N}{n_i}} \tag{2.5}$$

多腕バンディット問題と異なり，UCT では木の展開にともなって各子ノードの期待報酬に変化が生じる．このため，係数 C による探索項の調整が必要になる．この係数はゲームや併用する手法・ヒューリスティックに依存して最適な値が異なり，適宜調整する必要がある．UCT には主に以下の特徴がある．

1. ミニマックス値への収束性：ノードの真の評価値であるミニマックス値と，得られた平均報酬との期待誤差が $O(\log(N)/N)$ のオーダーで減少し，最適でないノードの選択確率が 0 に収束することが示されている [14]

2. 最良優先探索による非対称な木の成長：ある時点での平均報酬の情報をもとに有望そうなノードへ多くのプレイアウトを割り当てるため，そのノード以下が優先的に成長する非対称な木の生成が行われる

   - ミニマックス探索では一般に特定の深さまでのノードをすべて展開する．事前の知識や探索の途中結果にしたがって非対称な深さの成長を行う手法もしばしば用いられるが，ミニマックス探索の本来的な性質とは言いがたい．

3. Anytime なアルゴリズムである：モンテカルロ木探索では図 2.4 に示す一サイクルごとに情報が更新される．一サイクルにかかる時間は与えられる思考時間に対して一般に非常に短い．このため，事実上どの時点で思考を打ち切ったとしても，得た情報すべてを用いて手の選択を行うことができる．すなわち，思考時間すべてを有用に使うことができる，もしくは思考時間を細かく制御できるということである．

   - ミニマックス探索では一般に一定深さまでのノードをすべて探索するまでは手の判断を行えないか，著しく情報が欠けた状態で手の判断を行わなければならない．より細かな時間制御を目指す手法としては，思考時間内で浅い探索から始めて徐々に深さを増した探索を行う反復深化法が存在する．この手法でも，時間内に完了しなかった深さの探索の情報を用いることはできず，モンテカルロ木探索ほど細かい時間の制御は行えない．

2.3 シミュレーション方策の学習

2.3.1 方策の改善

最も単純な方策，すなわち一様な確率でランダムに手を選ぶ方策（以下一様方策と呼ぶ）ではプレイアウトの精度が悪い．UCT に代表されるモンテカルロ木探索では最善手への収束性が示されているものの，こうした精度の悪いプレイアウトでは収束が遅くなると考えられる．実際，時間あたりのプレイアウト回数が少なくなったとしても，より賢い方策を用いることで同一思考時間での棋力が向上することが囲碁 [10] や将棋 [21] などで知られている．具体的な方策の改善には様々な方法が提案されている．

まず，学習によらないルールベースでの方策の改善がある．例えば将棋で「『直前にこちらの駒を取った駒を取り返す手』は重要である」といった作成者のゲーム知識に従って手の優先度を決める手法である．代表的な例としては，囲碁に初めて UCT を適用し成功した MoGo[12] におけるプレイアウトの改良があげられる．MoGo では石が打たれる周囲 3×3 マスのローカルなパターンを主に利用している．これが囲碁において重要な特定の形（キリやハネなど）になる手があれば，それらの中から手を選ぶ，といったように用いる．ローカルなパターン以外にもいくつかのヒューリスティックを用いることで，一様方策に比べモンテカルロ木探索の棋力を大きく改善している．

また，モンテカルロ木探索における方策改善に機械学習を用いた初期の成功例として，囲碁ソフト CrazyStone がある [9]．CrazyStone では，各特徴の強さを Elo レーティングによって評価し，指し手をそれら特徴の組み合わせと考えて評価することで，各手の選択確率，すなわち方策とする手法を用いている．Elo レーティングが基礎としている Bradley-Trerry モデルについて述べる．これは過去の戦績からあるプレイヤの勝率を求めるためのモデルである．このモデルでは，n 人のプレイヤがおり，プレイヤ j がレーティング γ_j (> 0) [2]を持つとき，プレイヤ i の勝率を次のように推測する．

$$P(i \text{ の勝利}) = \frac{\gamma_i}{\sum_{j=1}^{n} \gamma_j} \tag{2.6}$$

さらに Bradley-Trerry モデルは，プレイヤが組み合わされたチームの勝率に対する予測も与えている．このとき，チームの強さはメンバーのレーティングの積で表される．例えば，1-2-3，4，1-5-6-7 という 3 つのチームが戦う場合，チーム 1-2-3 の勝率は，

$$P(1-2-3 \text{ の勝利}) = \frac{\gamma_1\gamma_2\gamma_3}{\gamma_1\gamma_2\gamma_3 + \gamma_4 + \gamma_1\gamma_5\gamma_6\gamma_7} \tag{2.7}$$

と表される．この例のように，あるプレイヤは同時に複数のチームに現れ得るが，同じチームに複数回現れることはない．次にレーティングベクトル γ の推定方法について述べる．これは，過去の試合結果の系列 R を利用して $P(\gamma|R)$ を最大化するような $\gamma = \gamma^*$ を見つけることで推定される．

[2]一般には $r_i = 400 \log_{10} \gamma_i$ が Elo レーティングと呼ばれる

CrazyStone では，γ^* を見つけるために少数化-最大化（minorization-maximization）アルゴリズムを用いており，各 γ_i の更新式は次のようになる．

$$\gamma_i \leftarrow \frac{W_i}{\sum_{j=1} N \frac{C_{ij}}{E_j}} \tag{2.8}$$

なお，W_i はプレイヤ i の勝ち数，C_{ij} は試合 j での i の属するチームの強さ，E_j は試合 j の参加者全員の強さである．CrazyStone はノビやアタリ，ローカルなパターンなどの各特徴を一つのプレイヤと考え，各手を特徴の組み合わせと見ることで Bradley-Trerry モデルを適用し，方策を学習している．こうして得られたレーティングをプレイアウトにおける方策や，Progressive Widening における手のオーダリングに用いることで大幅な棋力の向上がなされている．

複数の方策を比較した研究としては,「『強い』方策」と「バランスのとれた方策」という概念を導入したものがある [18].「強い」方策とはプレイアウト中の一手一手の精度が高い方策のことを指し，バランスのとれた方策とはプレイアウトを繰り返して得られる平均報酬の偏りが少ない方策を指す．それぞれ厳密には以下のように定義される．はじめに「強さ」の定義について述べる．まず，深さ t の局面 s_t におけるミニマックス値を $V^*(s_t)$ とする．このミニマックス値は局面 s_t の真の評価値であるので，以降プレイヤが互いに最適な行動をとり続けたと仮定すると，

$$V^*(s_t) = V^*(s_{t+1}) = V^*(s_{t+2}) = \cdots = V^*(s_T) \quad (s_T\text{は終端局面}) \tag{2.9}$$

となり増減することはない．したがって,「一手の誤差」は $\delta_t = V^*(s_{t+1}) - V^*(s_t)$ と定義することができる．これを用いて，方策 π_θ （θ はパラメータ）の局面 s_t における一手の誤差の期待値は $\mathbf{E}_{\pi_\theta}[\delta_t^2]$ と表される．そして，テスト局面のセット ρ における次の $J(\theta)$ の小ささが方策の「強さ」の定義である．

$$J(\theta) = \mathbf{E}_\rho[\mathbf{E}_{\pi_\theta}[\delta_t^2 | s_t = s]] \quad (s \in \rho) \tag{2.10}$$

多くの局面で誤差の少ない手を高い確率で指せる方策，すなわち一手一手の精度が高いものが「強い」方策であることが分かる．したがって，方策の確率に従ってランダムに手を選ぶダイレクトなプレイヤ同士では，一般には「強い」方策によるプレイヤの性能が高いと考えられる．

一方で，バランスのとれた方策ではこうした一手の精度・誤差ではなく，プレイアウトを繰り返した際の平均的な誤差に着目する．まず，プレイアウト一回の誤差，すなわちミニマックス値の変動は次のように計算できる．

$$\begin{aligned} \sum_j^{T-t} \delta_{t+j} &= V^*(s_T) - V^*(s_t) \\ &= z - V^*(s_t) \end{aligned} \tag{2.11}$$

z はプレイアウトの報酬であり，終端局面におけるミニマックス値 $V^*(s_T)$ と同値である．これより，ある局面 s_t において方策 π_θ による誤差の期待値は以下のように書ける．

$$\mathbf{E}_{\pi_\theta}[z - V^*(s_t)] = \mathbf{E}_{\pi_\theta}[z] - V^*(s_t) \tag{2.12}$$

このとき，$\mathbf{E}_{\pi_\theta}[z]$ はプレイアウトを繰り返すことで得られる期待報酬である．方策の「バランス」は局面セット ρ における以下のインバランス $B(\theta)$ の小ささと定義される．

$$B(\theta) = \mathbf{E}_\rho \left[(\mathbf{E}_{\pi_\theta}[z|s_t = s] - V^*(s))^2 \right] \tag{2.13}$$

以上，特に式 2.12 から分かるとおり，バランスの定義において一手の精度や一回のプレイアウトの精度を直接的には考慮していない．仮にそれらの精度が悪くとも，結果として平均報酬が真の値であるミニマックス値に対して偏りが少なければ「バランスがとれている」と考える．

一手の精度が高いことはモンテカルロプレイヤの改善につながると考えられてきたが，[10, 18] では「強い」方策が必ずしもモンテカルロプレイヤを改善しないことが示されている．これは,「強い」方策の作り方によっては，前述のバランスが保たれないためではないかとされている [18]．2.1.2 節で述べたように，モンテカルロ法では平均報酬をもとに手を選ぶため,「強く」ともバランスがとられず平均報酬に偏りが大きい方策では，モンテカルロプレイヤの強さにはつながらないということである．実際，盤面サイズが 5×5，6×6 の囲碁 [18] や 9×9 の囲碁 [13] において，バランスのとれた方策を用いた方がより強いモンテカルロプレイヤとなることが示されている．

2.3.2 シミュレーション・バランシング

2.3.1 で述べた方策のバランスをとるための学習方法として，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，囲碁においてその有用性が示されている [18, 13]．

バランシングにおける学習の目的は，式 2.13 で表されるインバランス $B(\theta)$ を最小化する方策のパラメータ θ を求めることである．すなわち，目的となる方策のパラメータ θ^* は式 2.14 のように表すことができる．

$$\theta^* = \arg\min_\theta \mathbf{E}_\rho \left[(V^*(s) - \mathbf{E}_{\pi_\theta}[z|s])^2 \right] \tag{2.14}$$

なお，真のミニマックス値 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 [14] 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 (2.15) 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．

$$\pi_\theta(s,a) = \frac{e^{\phi(s,a)^T\theta}}{\sum_b e^{\phi(s,b)^T\theta}} \tag{2.15}$$

この式 2.15 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，報酬をミニマックス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．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，かつ平均報酬がミニマックス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アルゴリズム 2.1 に示す．この手順は学習中の方策 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 とミニマックスの近似値 $\hat{V}^*$ 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 とミニマックスの近似値 $\hat{V}^*$ との比較から，平均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 (2.16) 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，学習中の方策 π_θ で手を指したときに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．

$$\begin{aligned} \psi(s,a) &= \nabla_\theta \log \pi_\theta(s,a) \\ &= \phi(s,a) - \sum_b \pi_\theta(s,b)\phi(s,b) \end{aligned} \tag{2.16}$$

Algorithm 2.1 シミュレーション・バランシングの更新アルゴリズム [18]

$\theta \leftarrow 0$
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

2.4 モンテカルロ木探索におけるパラメータの自動調整

モンテカルロ木探索では，選択や展開，プレイアウトにおいて様々なパラメータの調整を行う必要がある．例えば UCT を用いる場合，選択に用いる UCB 値の探索項の係数 C（式 2.5）や，子ノードの展開に関するパラメータがある．さらに，様々な改良によりパラメータが増加することが多い．しかし，これらパラメータの適切な値を直感的に求めるのは難しい．理由として，一般に各パラメータ間の依存性が強い[3]ことや，そもそもランダムなシミュレーションを基礎としているために，パラメータと木の成長やプレイヤの強さとの関係が直感的に明らかでないことなどが考えられる．そこで，自己対戦などにより適切なパラメータを推定する必要がある．ただし，一つ一つ順にパラメータを調整していくのは現実的ではない．前述のように，パラメータの数が多く，しかもそれぞれの間の依存性が考えられるためである．

こうした問題に対処するため，Chaslot らはクロスエントロピー法（Cross-Entropy Method; 以下 CEM）により，全パラメータをまとめて自動調整する手法を提案している [7]．パラメータベクトルを $\mathbf{x}$，パラメータベクトルを評価する関数を f とすると，最適なパラメータベクトル $\mathbf{x}^*$ は以下のように定義される．

$$\mathbf{x}^* = \arg\max_{\mathbf{x}} f(\mathbf{x}) \tag{2.17}$$

CEM では，以下の様な手順でこの $\mathbf{x}^*$ を推定する．

1. 各パラメータ x_j に適当なパラメトリックな確率分布 g_j を与える
2. 分布に従って N 個のサンプルパラメータベクトル $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(N)}$ を生成する
3. 各サンプルを f によって評価し，評価が閾値以上，もしくは上位数個のサンプルをエリートサンプルとする
4. エリートサンプルが出やすくなるように各パラメータの分布 g_j を更新する
5. 手順 2 から繰り返す

具体的に，分布 g_j がガウス分布 $N(\mu_j, \sigma_j^2)$ である場合を考える．このとき，エリートサンプルの集合を E，エリートサンプルの数を M とすると，手順 4 における新たな分布 $g\prime = (\mu\prime_j, \sigma\prime_j^2)$ を以下のように求める．

$$\mu\prime_j = \frac{1}{M} \sum_{\mathbf{x^{(i)}} \in E} x_j^{(i)} \tag{2.18}$$

$$\sigma\prime_j^2 = \frac{1}{M} \sum_{\mathbf{x^{(i)}} \in E} (x_j^{(i)} - \mu\prime_j)^2 \tag{2.19}$$

[3]例えば，前述の係数 C や展開のパラメータはいずれも木の成長に深く関わっている

すなわち，$\mu\prime_j$ はエリートサンプルのパラメータの平均値であり，$\sigma\prime_j^2$ は同パラメータの分散となる．なお，適当なステップサイズ定数 $\alpha(0 < \alpha \leq 1)$ を用いて，新たな分布を $\mu_j + \alpha(\mu\prime_j - \mu_j)$ のように求めるほうが望ましいとしている．

2.5 将棋におけるモンテカルロ法

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ず，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 [21, 25] 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 [26, 24] 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 [21]．この研究では，2.3.1 節で述べた Elo レーティングによる指し手の確率的選択手法を用いている．初期局面から 256 手以内のプレイアウトの終局率は，学習前が 19.3% であったのに対し，学習後は 90.5 % まで向上している．方策の改良に加え，ミニマックス探索による将棋プログラムで用いられてきたヒューリスティックを組み合わせることで，棋力の向上を図っている．特に重要なものとしてはキラームーブを挙げている．これは，兄弟ノードにおける最善手を示す用語であり，ミニマックス探索による将棋やチェスにおいては広く用いられる概念である．佐藤らは UCT の木の内部においてキラームーブを優先的に選ぶことで改良を行った．以上の結果として，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．しかし実際の対局ではまだ初段にも達しておらず，単純にミニマックス探索を上回ることは難しいとしている．一方で，一部の局面ではミニマックス探索よりも良い結果を得ることに成功している．その例として，ミニマックス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が挙げられている．佐藤らは，こうしたゲーム全体でモンテカルロ木探索を用いる手法に加え，序盤の定跡選択にモンテカルロ木探索を用いる手法も提案している．多くの将棋プログラムは，序盤においてミニマックス探索を用いずに定跡データベースを利用した指し手の選択を行なっている．従来は，指し手 m の選択確率を「(m が指された回数) / (現局面がデータベースに存在する数)」とした，単純な確率的選択手法が用いられてきたが，過度にデータベースを信頼してしまう点が問題だとしている．そこで，データベース中で現局面と一致する局面を持つ棋譜をプレイアウトとみなし，子ノードの展開においてはデータベースに含まれる手だけを利用するような UCT を用いて手を選択する手法を提案している．定跡選択においてこの手法を用いることで，ミニマックス探索によるプレイヤの棋力が向上しており，従来の単純な確率的選択手法を用いたプレイヤに対して 61 % の勝率を得ている．

宇賀神ら [25] は Elo レーティングを用いた手法の欠点として，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点を挙げている．実際，完全にランダムなプレイアウト[4]に

[4] 256 手かかった場合はプレイアウトを打ち切る

比べ，4 分の 1 程度の回数しかプレイアウトを行えない．モンテカルロ法の精度は回数にも依存するため，問題になり得る．こうした速度の問題を解決するため，方策においてより少ない特徴を利用して終局率の向上を図っている．具体的には，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．ただし，棋力に関する言及はなされていない．

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，橋本らの研究 [26] と竹内らの研究 [24] がある．両研究ではともにプレイアウトを打ち切った局面での静的評価関数の利用を試みている．橋本らは 10 手程度でプレイアウトを打ち切り，駒割りのみによる単純な静的評価関数をその報酬としている．しかし，次の一手問題による性能評価では正答率が 3% 程度であり，非常に低い性能にとどまっている．

一方で竹内らは，報酬として単純に静的評価値を用いるのではなく，ルート局面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，それ以外の場合を引き分けとしている．勝敗の決定に評価値の絶対値ではなく相対値を利用しているのは，モンテカルロ木探索一般における，極端に優勢（劣勢）な局面が苦手という欠点に対処するためだとしている．この欠点は，例えば優勢な局面では多少悪い手を選んでもプレイアウトの結果が勝ちになりやすいため，良くない手を選択する傾向にあるというものである．相対値を用いることで，優勢な局面であってもより良くなる手を選ぶと期待できる．こうした問題意識は囲碁においても見られ，極端に優勢（劣勢）な局面ではプレイアウトの勝敗判定に用いるコミを動的に変更する手法が提案されている [3]．なお，竹内らの手法では静的評価値を計算するときに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．以上の結果として，ミニマックス探索には依然として及ばないものの，プレイアウトにおいて勝敗を報酬とする一般的なモンテカルロ木探索と比べて有意に強いプレイヤが作成された．

2.6 モンテカルロ木探索の解析

モンテカルロ木探索の解析を行った既存研究について述べる．

モンテカルロ木探索を含めた，ゲームにおける探索手法の評価方法を研究したものとしては，竹内らのもの [19, 20] がある．この研究では時間のかかる対戦実験によらない，探索手法の評価方法の提示を目標としている．その一つとして評価曲線（Evaluation Curve）を導入した．これは，横軸に

モンテカルロ木探索やミニマックス探索によって得られた局面の「評価値」，縦軸に熟練者の棋譜から得られた「勝率」をプロットしたグラフである．評価値の低い局面では実際の勝率が低く，逆に評価値の高い局面では実際の勝率が高い，といった単調性を満たす探索手法が良いプレイヤにつながると推測される．特に，プレイアウトの報酬を勝ちで 1，負けで 0 とするようなモンテカルプレイヤにおいては,「評価値」と「勝率」がより一致する手法が良いと考えられる．囲碁における研究では，実際に対戦実験において強いプレイヤーほど両者がよく一致していることを示している．また，シチョウと呼ばれる局面に限った場合の評価曲線も調べている．シチョウは，連続で限られた正しい手を選ぶことが要求され，経験的にモンテカルロプレイヤが苦手と言われる局面である．ここでは，囲碁に特有の強化をあまり施さない単純なモンテカルロプレイヤや UCT プレイヤが前述の単調性を保てていない一方で，様々な強化を行った UCT プレイヤである Fuego[5]は，こうした局面でも「評価値」と「勝率」が良く一致していることが示されている．他にも，局面の探索手法を，局面を勝ちと負けに分ける二値分類問題と見て，教師あり学習の分野で広く使われている指標 [5] によって評価している．

Ramanujan らはチェスにおいて，モンテカルロ木探索の苦手な点についての解析を行なっている [16]．チェスにおいてモンテカルロ木探索が良い性能を示せない原因として，浅いトラップ（shallow trap）の存在を挙げている．この浅いトラップとは自身が詰まされている局面であり，レベル n のトラップと言った場合には，n 手詰めの局面のことを指す．Ramanujan らは，チェスにはこうした浅いトラップへ導いてしまうトラップ手が数多く存在していることを示し，囲碁との大きな違いだとしている．全幅探索を行う一般的なミニマックス探索であれば深さ $n+1$ の探索を行うことでこうしたトラップ手を確実に避けることができる一方で，深さ優先探索であるモンテカルロ木探索では同じノード数を展開した場合でもトラップ手を必ずしも避けられない．実際にチェスにおけるレベル $1, 3, 5, 7$ のトラップについての解析から，モンテカルロ木探索がトラップ手を十分に避ける事ができないことを示している．具体的には，特にレベル $5, 7$ といった（相対的に）深いトラップが存在するときに，モンテカルロ木探索が最善とした手の評価値とトラップ手の評価値の間にほとんど差がなく，トラップ手を十分に避けられていないことが示された．また，トラップ手は必敗であるため，モンテカルロ木探索による評価値が-1[6]に近づくほど正しく評価できているといえる．しかし，レベル 5, 7 のような深いトラップになると，評価値が-1 にほとんど近づかないか非常に収束が遅いことを示している．

また，Ramanujan らはマンカラ（Mancala）というゲームにおいてもモンテカルロ木探索の解析を行なっている [17]．マンカラを対象としたのは，ゲーム固有のヒューリスティックな知識を入れない純粋なモンテカルロ木探索と，従来のミニマックス探索によるプレイヤが拮抗しており，解析に適しているためだとしている．マンカラにおけるモンテカルロ木探索とミニマックス探索との比較から，前者が浅い部分に多くの終局を含む局面を苦手とし，そうでない部分を得意とすることを明ら

[5] http://www.perfectsky.net/fuego/
[6] プレイアウトの報酬は，勝ち，負け，引き分けをそれぞれ 1, -1, 0 としている

かにしている.

第3章　将棋におけるモンテカルロ木探索の解析

本章では，将棋においてモンテカルロ木探索の解析を行う．本研究では，モンテカルロ木探索における子ノードの選択部分に 2.2 節で述べた UCT を用いた．以降単に「UCT」といった場合には，UCT を適用したモンテカルロ木探索のことを指す．また，今回はモンテカルロ木探索や学習を将棋ソフト「激指」上で実装している．したがって，学習に用いる特徴は「激指」が遷移方策に用いている特徴[1]を用いた．

3.1 節では解析に先立って，用いる方策，UCT に関する設定について述べる．また，シミュレーション・バランシングの将棋への適用を行ったため，その学習設定や学習の様子に関する記述も行う．

3.2 節では方策間での比較を行う．まず，モンテカルロ将棋において注目されることの多い終局率についての評価を行う．次に，各方策の特徴を明らかにするため，手の選択確率の偏り方について評価を行う．そして，対戦実験により実際の棋力の比較を行う．ここでは，手選択確率の偏りの評価結果による予想を確かめるため，中盤での強さ，終盤での強さといった点も調査する．最後に，この中盤，終盤での棋力差の原因について考察を行う．

3.3 節では，モンテカルロ木探索の得意・不得意とする局面を明らかにするために，ミニマックス探索との比較を通した解析を行う．まず，大まかに中盤・終盤の得手・不得手を調べた．将棋には駒組みや攻め，受けなど，局面によって必要とされる要素が大幅に変わる特性があり，各要素の出現頻度には局面の進行具合が大きく関わるためである．これより，アルゴリズムの違いが大きいモンテカルロ木探索とミニマックス探索では得手・不得手に差が出る可能性が考えられる．なお，序盤では定跡データベースに従って指すことが一般的であるため，序盤を除いた解析を行う．次に，より細かく特定の局面における解析を行った．一つ目には，2.6 節で述べた浅いトラップを含む局面について調べた．将棋はチェスと似た性質を持つゲームであり，同様に浅いトラップが課題となりうるためである．さらに，こうした詰み局面に限らないより一般的な解析として，局面の「明確さ」に対してどの程度良い手を選ぶことができるかを比較した．モンテカルロ木探索はランダムなプレイアウトの結果を用いるため，ミニマックス探索にとっては「明らか」な良手を見逃し，それがミニマックス探索に対する弱さにつながっている可能性が考えられるからである．なお，ミニマックス探索との比較においても，引き続き方策間の違いに着目する．

最後に 3.4 節で以上の結果について簡単にまとめる．

[1]詳細は文献 [22]「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

3.1 設定

3.1.1 利用する方策

モンテカルロ木探索の解析においては，以下の 3 種類の方策を用いる．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[2]に基づく方策，2.3.2 節で述べたシミュレーション・バランシングを適用して学習した方策の 3 種類である．以下，それぞれ「一様方策」,「遷移方策」,「バランシング方策」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には，遷移確率のごく低い手を除いた，遷移確率の高い上位 15 個の手のみを用いた．また，プレイアウト中に一手詰めの局面に至った場合はプレイアウトを打ち切り，その結果を利用する．

遷移方策とバランシング方策を用いるのは，両者が大きく異なる目的を持った学習手法により生成されるものであり，異なる特性を示す可能性が考えられるからである．それぞれの学習の目的は，前者では，棋譜を教師としてそこで指された手の尤度を最大化することであり，後者では，プレイアウトを繰り返すことで得られる平均報酬をミニマックス値に近づけることである．すなわち，遷移方策の学習は一手一手の精度が高い「強い」方策[3]を目指しており，一方でバランシング方策の学習は平均報酬に偏りが少ないバランスのとれた方策を目指しているといえる．ここで，囲碁においてはシミュレーション・バランシングによる「バランスのとれた」方策が有効であることが示されている [18, 13]．しかし，将棋は囲碁に比べて一手の間違えが重大な損失となる局面が多いことが経験的に知られている[4]．このため，直接的には一手一手の精度に着目しないシミュレーション・バランシングでは，こうした間違えを避けるのが難しく，結果として十分バランスのとれた方策を学習することが難しい可能性がある．よって，将棋においては両方策を比較することには意義があると考えられる．なお，学習によって得られた両方策の特徴を明らかにするため，一様方策についても解析を行う．

3.1.2 UCT のパラメータ設定

まず利用するパラメータについて述べる．

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，プレイアウトを最大 d_{cut} 手で打ち切り，報酬としては 2.5 節で述べた静的評価値の差を利用する竹内らの方法 [24] を採用する．これは，ルート局面との静的評価値に比べ，プレイアウト末端での静的評価値が閾値 Th 以上高くなっていれば勝ち，逆に Th 以上低くなっていれば負け，それ以外を引き分けとして評価を行う手法である．このとき，それぞれの報酬を 1，-1，0 とした．

[2]「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
[3]棋譜における手を「正解」，すなわちミニマックス値が増減しない手とみなせば，式 2.10 における「強さ」の定義とも一致する
[4]後述のように，チェスと同じく将棋にも多く含まれる浅いトラップはその一例だといえる

表 3.1: UCT に関するパラメータの初期値 $\boldsymbol{\mu}_0$ と CEM による調整結果 $\boldsymbol{\mu}_7$

分類	種類		$\boldsymbol{\mu}_0$	$\boldsymbol{\mu}_7$
UCB	探索項の係数：	C	1	0.985
プレイアウトの打ち切り	打ち切り深さ：	d_{cut}	5	3
	勝ち負け判定の閾値：	Th	500	478
手数の制限	木の内部：	l_{tree}	16	7
	プレアウト中：	$l_{playout}$	16	15
Progressive Widening	最初の展開：	Ex_{first}	3	1
	以降の展開間隔：	Ex_{next}	4	1
ノードの初期値	初期訪問回数：	$n_{default}$	5	3.56

表 3.2: CEM 調整における分散の初期値 $\boldsymbol{\sigma}_0^2$ と最終的な値 $\boldsymbol{\sigma}_7^2$

分類	種類		$\boldsymbol{\sigma}_0^2$	$\boldsymbol{\sigma}_7^2$
UCB	探索項の係数：	C	1	$3.55e^{-3}$
プレイアウトの打ち切り	打ち切り深さ：	d_{cut}	20	1.25
	勝ち負け判定の閾値：	Th	20000	3133.24
手数の制限	木の内部：	l_{tree}	16	0.24
	プレアウト中：	$l_{playout}$	16	11.36
Progressive Widening	最初の展開：	Ex_{first}	7	0.09
	以降の展開間隔：	Ex_{next}	12	0.09
ノードの初期値	初期訪問回数：	$n_{default}$	20	1.08

将棋には，ほとんど探索なしに悪いことが分かる明らかな悪手が数多く存在する．そのため，何らかの基準により探索すべき手を絞り込むのが適当と考えられる．本稿では基準として遷移確率を用いることで探索する手の絞り込みを行った．具体的には，モンテカルロ木の内部においては上位 l_{tree} 個の手を用い，プレイアウト中においては $l_{playout}$ 個の手を用いた．

ノードの展開においては，Progressive Widening を用い，遷移確率の高い手から順に展開した．展開条件は訪問回数のみに依存するものとし，最初の 1 つの子ノードを Ex_{first} 回目の訪問で展開し，以降は Ex_{next} 回ごとの訪問で展開する．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，棋力が向上することが知られている [10]．本評価では初期報酬は 0 とし，初期訪問回数を $n_{default}$ 回に設定する．なお，訪問「回数」であるため整数値とするのが一般的と考えられるが，UCB 値を求める式 2.5 で用いることが目的であるため実数値とした．

これらに 2.5 における探索項の係数 C を加え，2.4 節で述べた CEM により値を調整した．まず，各パラメータの初期平均値を遷移方策 UCT 同士の自己対戦により簡単に求め，この値をもとに初期分散を適宜定めた．これらを 0 世代目とする．具体的な値 $\boldsymbol{\mu}_0, \boldsymbol{\sigma}_0$ はそれぞれ表 3.1，3.2 のようにした．次に，各サンプルの良し悪しを判断するための関数 f として，初期パラメータ $\boldsymbol{\mu}_0$ によるプレイヤを相手とした遷移方策 UCT 同士の自己対戦を用いた．この際，両プレイヤのプレイアウト回数を 1,000 回，各サンプルの対戦回数を 100 回とした．また，1 世代ごとに生成するサンプルを 50 個とし，エリートとして抽出するサンプルを勝率の高い上位 10 個とした．

これらの設定で調整を 7 世代分繰り返した結果，平均値 $\boldsymbol{\mu}_7$ と分散 $\boldsymbol{\sigma}_7$ はそれぞれ表 3.1，3.2 のようになった．分散の表 3.2 に着目すると，平均値や初期分散に対して減少の度合いの大きいパラメータは，棋力への影響が大きいといえる．こうした観点からいえば，勝敗の閾値 Th やプレイアウト中での手数 $l_{playout}$ は，棋力への影響が小さいことが分かる．特に後者の $l_{playout}$ は，木の中での手数 l_{tree} が分散の小さい重要なパラメータであることとは対照的である．また平均値そのものに着目すると，プレイアウトの打ち切り深さ d の値が 3 となっている点が興味深い．一般的なモンテカルロ法においてはプレイアウトを終局まで行うことを考えると，非常に小さな値だといえる．将棋には，ランダムなプレイアウトでは意味のある結果を得ることが難しいという特性があることを示していると考えられる．なお，プレイアウト中の手数制限 $l_{playout} = 15$ と合わせて考えると，同一局面からのプレイアウトの打ち切り局面は最大で $15^3 = 3375$ 種類と少ない．しかし，後述の対戦実験などの解析結果から分かるように，方策の差異を調べる上では十分な深さである．

以下実際の棋力の変化について述べる．初期パラメータのプレイヤに対するエリートサンプルの平均勝率は，図 3.1 のように推移した．少なくとも今回の実験設定において，初期パラメータに対して有意に強いパラメータが得られたことが分かる．より客観的な棋力の変化を調べるため，パラメータとして $\boldsymbol{\mu}_0, \boldsymbol{\mu}_7$ それぞれを用いた遷移方策 UCT とオリジナルの「激指」との対戦実験を行った.「激指」は深さ 6 固定とし，様々なプレイアウト数で UCT の棋力を調べる．ただし,「激指」の序盤では定跡を用いた指し手の選択を行なっており単純な比較が難しいため，序盤はランダムに選んだプロの棋譜を読み込み，その時点からの対局を行った．詳細な対戦方法は 3.2.3.2 節で後述する「特定の進行度間での対局」における進行度 32 ～ 128 での対局と同様である[5]．この対戦実験の結果を図 3.2 に示す．調整後のパラメータはプレイアウト数が少ない場合には調整前より大きく棋力が向上している．しかしプレイアウト数がおよそ 6,400 回以上になると勝率が逆転し，むしろ棋力を落としていることが分かる．これは，パラメータ調整時のプレイアウト数である 1,000 回に過度に適合したためだと考えられる．例えば，UCT 木内部の手数制限のパラメータ l_{tree} について考えると，プレイアウト数が少ない場合は値を小さくして良さそうな部分にしぼって探索せざるを得ないが，プレイアウト数が多い場合は値を大きくし，探索をすることで初めて分かる良い手を拾いあげる余地が増えるはずである．実際に，調整後のパラメータ $\boldsymbol{\mu}_7$ において l_{tree} のみ 16 という調整前の値を

[5]進行度とは盤面上の駒の情報などから局面の進み具合を数値化したものである．今回利用した「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．

用いたところ，プレイアウト数 12,800 回での深さ 6 の「激指」に対する勝率は $78.2 \pm 2.9\%$ と $\boldsymbol{\mu}_7$ に比べ大きく向上した．

最後に計算コストについて述べる．本研究では計算環境として以下を用いた．

- 1 ノードにつき，Intel Xeon CPU X5620 2.40GHz（4 コア × 2 スレッド）を 2 ソケット
- 上記を合計 14 ノード
  - 同時に走るスレッド数は最大で $224 = 4 \times 2 \times 2 \times 14$

この環境下で，7 世代分の計算を行うためにおよそ 20 時間を要した．より一般的な非クラスタ環境を想定すると，時間的に非常に高い計算コストだといえる．非クラスタ環境での利用を考えるのであれば，モンテカルロ木探索自体の高速化に加え，以下の様な場合について性能を調べる必要があるだろう．

- 対照プレイヤ（今回はプレイアウト数 1,000 回の遷移方策 UCT）としてより軽いものを利用した場合
  - プレイアウト回数を減らす，浅いミニマックス探索プレイヤを用いるなど
- CEM に関する各種パラメータの値としてより小さなものを用いた場合
  - サンプル数，対戦回数，UCT のプレイアウト回数

なお，竹内らは自己対戦に代わるより軽い指標を模索している [23]．この研究では，自己対戦による性能には及ばないものの，教師プレイヤとの一致数や二乗誤差を指標とした場合でも一定の性能が得られるようである．ここで教師とは，手調整したパラメータを用い，プレイアウト数 100,000 回という深い探索を行う UCT のことである．評価では棋譜からランダムに局面を抽出して用いる．これらの局面において，一致数の評価では生成されたサンプルプレイヤと教師の指し手の一致・不一致を調べ，二乗誤差の評価では各プレイヤから得られる平均報酬の差を調べている．これらの手法では教師値の計算に時間を要するものの，評価に使う局面は全世代で共通であり，事前に計算した値を繰り返し利用できるため対戦実験と比べて非常に高速だとしている．

3.1.3 シミュレーション・バランシングの学習について

3.1.3.1 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，$M = 1000, N = 1000$ のように設定した．学習係数 α は，t イテレーション目

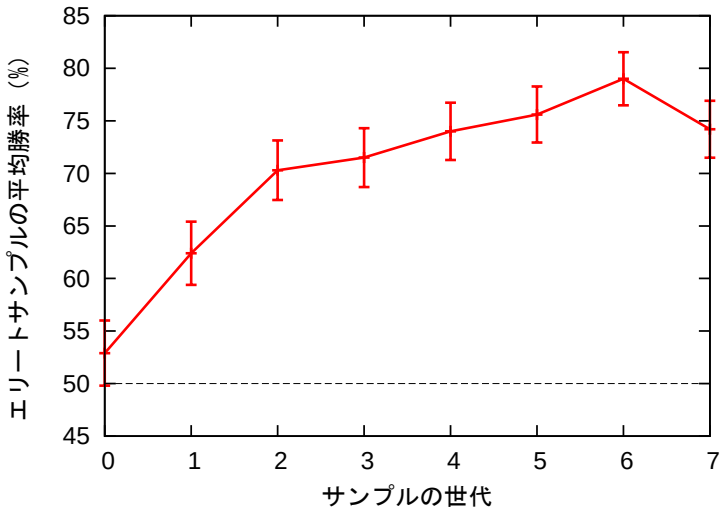


図 3.1: 初期パラメータプレイヤに対するエリートサンプルの勝率の推移

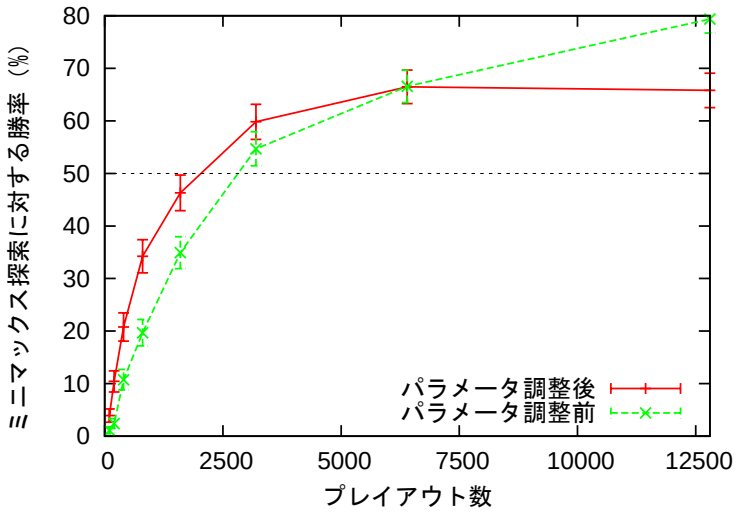
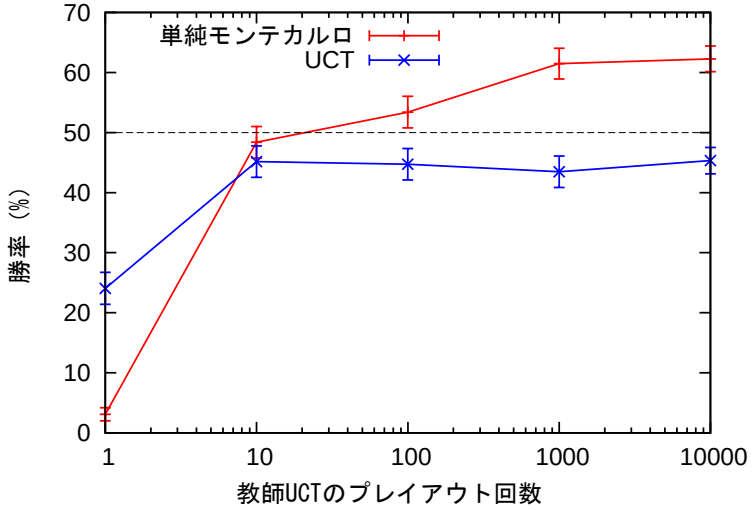
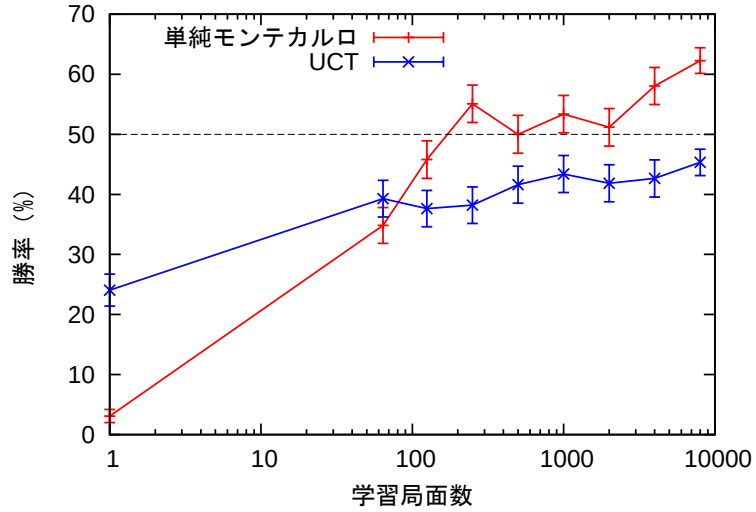


図 3.2: 深さ 6 のミニマックス探索に対する調整前後の棋力の変化



(a) 教師 UCT のプレイアウト回数（学習局面数：8,000）



(b) 学習局面数（教師のプレイアウト回数：10,000）

図 3.3: 様々なパラメータでの学習結果

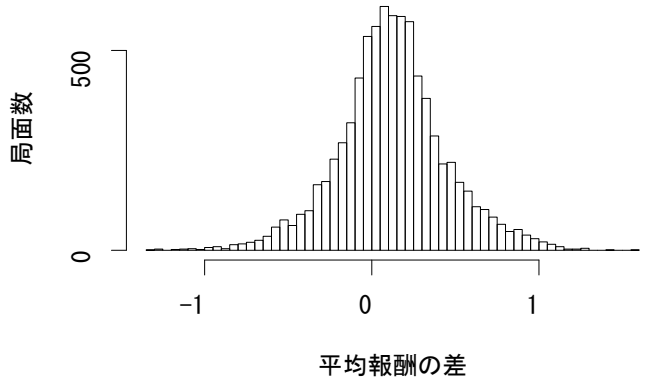


図 3.4: 遷移方策 UCT による，プレイアウト回数 10,000 回の場合の平均報酬 $\hat{V}_*$ と，10 回の場合の平均報酬 $\hat{V}'_*$ との差 $\hat{V}_* - \hat{V}'_*$ の 8,000 局面における分布

において $\alpha = 1/t$ とした．また，教師として用いるミニマックス値の近似値 $\hat{V}^*$ として，遷移方策 UCT による 10,000 回のプレイアウトの結果を用いた．なお，学習中の平均報酬やミニマックス値の近似値を求めるにあたり，プレイアウト中の一手詰み探索は利用していない．利用した場合に，生成されるバランシング方策の棋力が落ちることが確認されたためである．

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 8,000 局面を用意し，これを繰り返し用いて学習した．また，ミニマックス値と方策 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，一方を様々に変えて対戦実験を行った結果から決定した．このとき棋力の比較は遷移方策と行い，ルートの次のみで勝率を保持する単純モンテカルロ法プレイヤ同士，UCT 同士での 2 種類の対戦を行った．詳細な対戦条件は 3.2.3.1 で後述するものと同様である．以上の条件で，遷移方策に対する勝率は図 3.3 のように変化した．

ここで，図 3.3(a) に着目すると，バランシング方策 UCT の棋力は教師のプレイアウト回数が 10 回の場合でも 10,000 回の場合との有意差が見られない．遷移方策 UCT において，プレイアウト回数が 10 回のプレイヤと 10,000 回のプレイヤとでは大きく棋力が異なる．実際に 200 回対戦を行なったところ前者が後者に一度も勝てないほどに棋力の差は大きい．それにもかかわらず，それぞれを教師としたバランシング方策 UCT の棋力に差が見られない点について考察する．シミュレーショ

ン・バランシングにおいては教師 UCT による平均報酬（ミニマックス値の近似値）に着目しているため，遷移方策 UCT による，プレイアウト回数 10,000 回の場合の平均報酬 $\hat{V}_*$ と，10 回の場合の平均報酬 $\hat{V}'_*$ との差 $\hat{V}_* - \hat{V}'_*$ を調べた．学習に用いた 8,000 局面におけるこの差の分布を図 3.4 に示す．0.1 付近を中心にほぼ左右に均等に分布していることが分かる．このため，教師のプレイアウト回数が 10 回の場合でも，各局面の評価精度の悪さがある程度相殺されたと考えられる．こうしたことは，教師としているのが各プレイヤの指し手ではなく平均報酬であることから生じる現象だといえる．なお，中心値の差 0.1 は UCT では問題になっていないが，図 3.3(a) より単純モンテカルロ法においては棋力に影響が出ていることが分かる．

3.1.3.2　学習の様子

学習に伴う平均二乗誤差の推移は，図 3.5 のようになった．横軸は，学習のために用意した 8,000 局面を何回繰り返して学習したかを示している．学習の進行にともなって MSE が減少していることが分かる．

各特徴量の重みのヒストグラムは，図 3.6 のようになった．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 [13]．なお図中では上位 2.5% 個 下位 2.5% 個の特徴は表示していないが，最も高い重みと低い重みはそれぞれ $1.10, -0.98$ となった．

学習時間について述べる．CPU には，Intel Xeon CPU X5620 2.40GHz を用いた．メモリ容量は約 24.7GB である．高速化のため，ミニマックス値の近似値 $\hat{V}^*$ の値は 8,000 局面分を事前に計算して繰り返し用いている．また，激指では指し手の特徴の抽出に時間を要するため，ルート局面から深さ 3 までの局面では候補手とその特徴をキャッシュし，再度同じ局面を訪れた時にはそれを利用する．今回は UCT におけるパラメータ調整の結果（3.1.2 節）からシミュレーションの打ち切り深さを 3 としたため，すべてキャッシュ上にのることになる．なお，並列化は行なっていない．以上の条件で，1 イテレーションの学習時間は約 33 分であった．

3.2　シミュレーション方策間の比較

3.2.1　終局率の評価

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・対局両面においてプレイアウトの深さを増やしていく上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，n 手以内に終局に至る確率を調べた．この結果を図 3.7 に示す．一様方策に比べ，遷移方

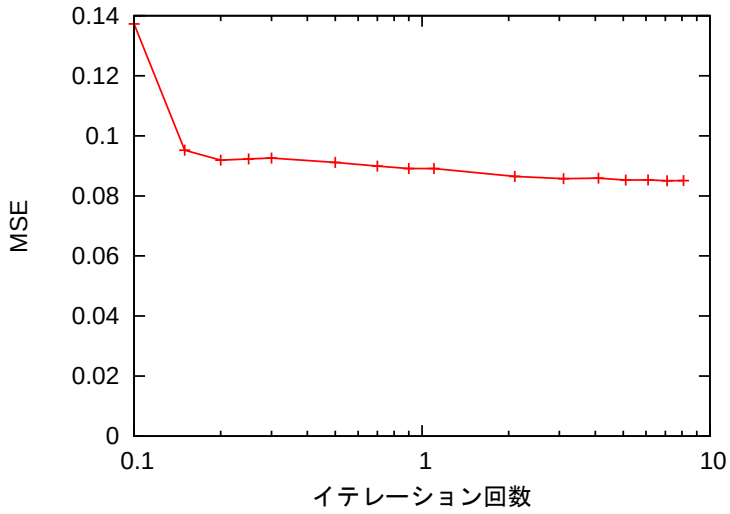


図 3.5: 学習の進展による平均二乗誤差（MSE）の推移

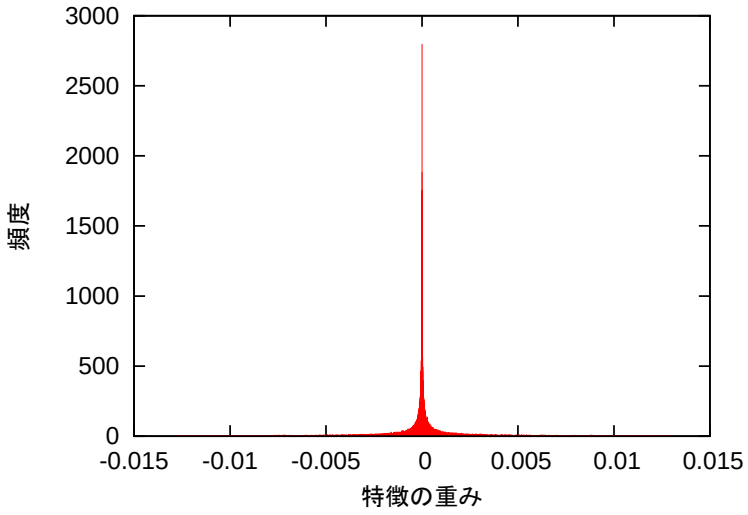


図 3.6: 特徴量に対する重みのヒストグラム

策とバランシング方策は終局率が向上していることが分かる．したがって，プレイアウトに知識を導入することで，終局率の向上が可能だと言える．また，知識を入れた二方策間で比較すると，遷移方策がバランシング方策よりもわずかに終局率が高いものの，大きな差は見られない．

前述のように，プレイアウトにおいては候補手を遷移確率の高い手に限定しており，さらに一手詰み探索を利用している．これらの影響を調べるため，一様方策において両方を用いた場合，一方のみを用いた場合，いずれも用いない場合について終局率の比較を行う．この結果が図 3.8 である．手数の制限と一手詰み探索の利用は，いずれも終局率の向上に大きく寄与していることが分かる．個別に見ると，相対的に浅い部分での終局率向上には一手詰み探索がより大きな役割を果たしており，逆に深い部分では手数の制限が大きな役割を果たしている．この原因として，浅い部分に多くの終局を含む局面（以下「最終盤」と呼ぶ）の存在が考えられる．すなわち,「最終盤」においてはどう指しても一手詰み局面にたどりつきやすいため一手詰み探索の影響が大きいが，そうでない局面においてはまず「最終盤」に至るために（少なくとも一方のプレイヤには）ある程度賢い指し方が求められるため，遷移確率による手数の制限が大きな影響を持つだろうということである．

こうした予想を確かめるため，序中盤（進行度 $0 \sim 96$）と終盤（進行度 $96 \sim 127$）のそれぞれの局面からプレイアウトを行い，終局に至るまでの手数を調べた．まず終盤の結果を示す図 3.10 に着目すると，前述の図 3.8 と比べて一手詰み探索の影響が大きくなっていることが分かる．終盤には「最終盤」が多く含まれるためだと考えられる．一方で，序中盤の結果を示す図 3.9 の結果を見ると，一手詰み探索の影響が低下していることが分かる．終盤に比べて「最終盤」の割合が少ないためだと考えられる．

3.2.2 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 $D(s)$ 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 (3.1) 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．

$$D(s) = \sqrt{\sum_{i}^{m} \left(\pi(s, a_i) - \frac{1}{m} \right)^2} \tag{3.1}$$

これを，50,000 局面で求めた結果，各方策における $D(s)$ の分布は図 3.11 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，有効な手が多く，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，有効な手が一つしかなく，それ以外の手が全て悪手であ

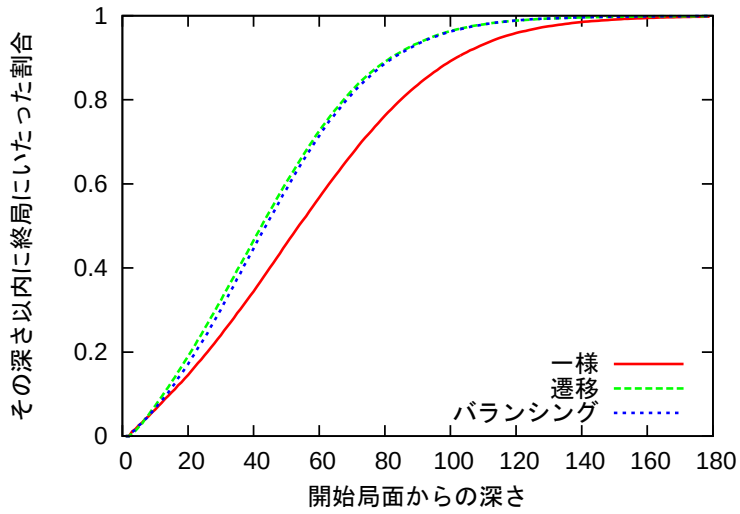


図 3.7: 終局までの手数：方策間での比較

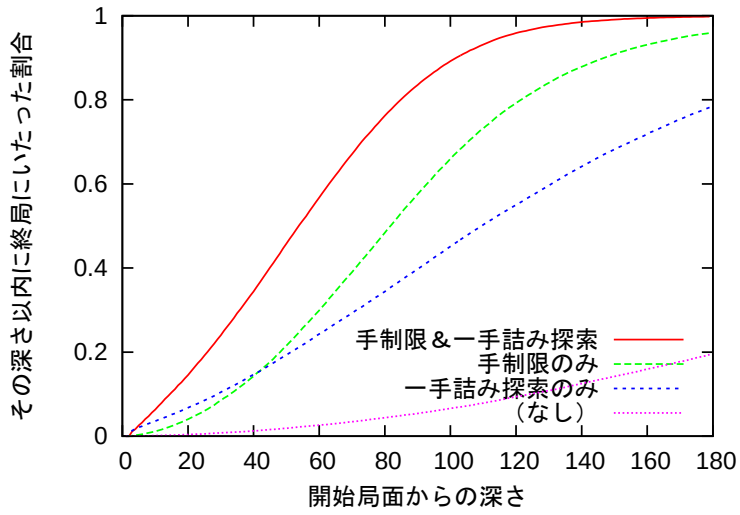


図 3.8: 終局までの手数：遷移確率による手の絞り込みや一手詰み探索の影響（一様方策）

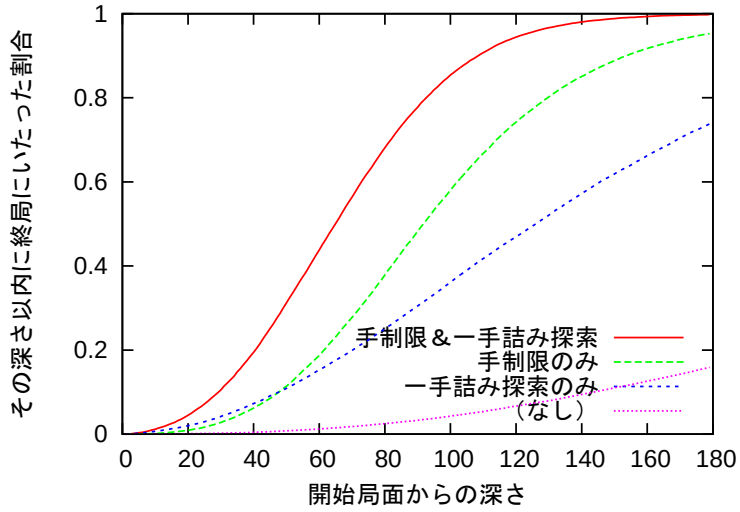


図 3.9: 終局までの手数：遷移確率による手の絞り込みや一手詰み探索の影響（中盤：進行度 0 ～ 96）

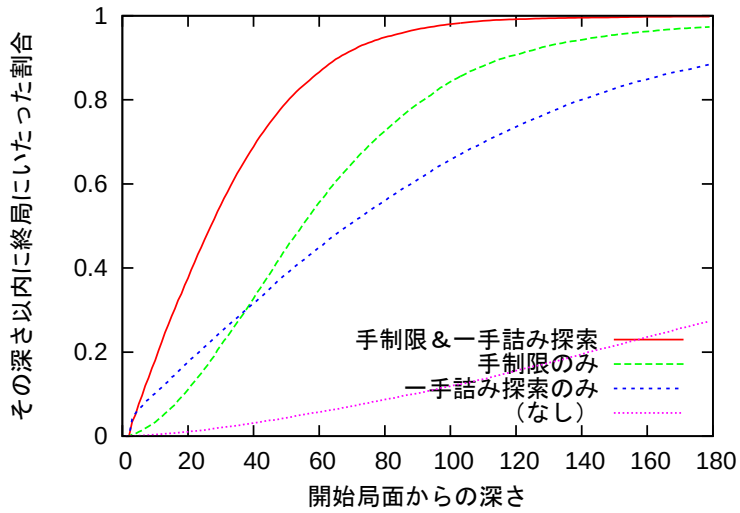
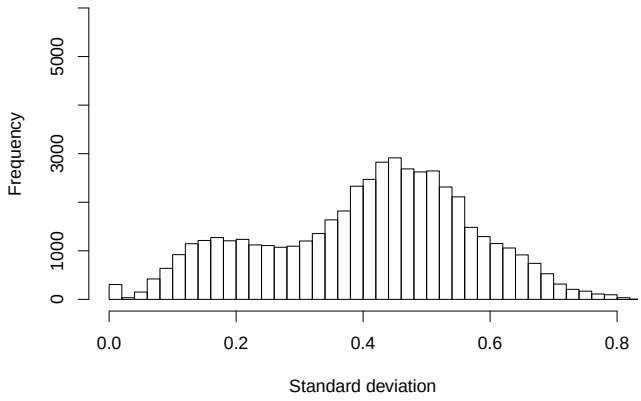
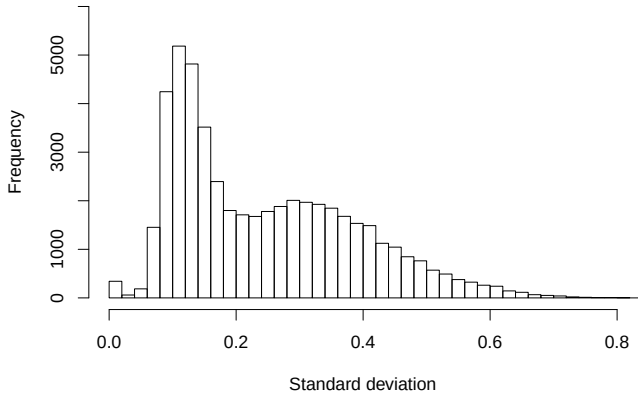


図 3.10: 終局までの手数：遷移確率による手の絞り込みや一手詰み探索の影響（終盤：進行度 96 ～ 127）



(a) 遷移方策



(b) バランシング方策

図 3.11: 手選択確率の偏りの分布

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．

3.2.3 対戦実験

3.2.3.1 モンテカルロプレイヤー同士の対戦

各方策によるプレイヤの強さを評価するため，方策間での対戦実験を行った．ここでは以下の 3 つの場合について強さを比較する．まず，プレイアウトを行わずに方策の確率に従ってダイレクトに手を選ぶプレイヤ（以下ダイレクトプレイヤ）同士での対戦を行った．これは，2.3.1 節で述べた方策の「強さ」の間接的な評価だといえる．次に，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同士での対戦を行った．これは，同じく方策のバランスの間接的な評価だといえる．最後に，UCT 同士の対戦を行った．ゲームにおけるモンテカルロ法では UCT を含むモンテカルロ木探索を用いるのが一般的であるため，方策の実用的な強さの評価だといえる．

単純モンテカルロ法同士，UCT 同士の対戦実験ではプレイアウト数はを 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同士での対局であれば，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．以上の条件のもと対戦実験を行った．

ダイレクトプレイヤ同士の対戦結果を表 3.3 に示す．これを方策の「強さ」の評価と考えると，遷移方策が最も「強い」方策であることが分かる．バランシング方策についても，一様方策との比較から，学習によって一定の「強さ」を得たことが分かる．次に，単純モンテカルロプレイヤ同士の対戦結果を表 3.4 に示す．これを方策のバランスの評価と考えると，バランシング方策が最もバランスのとれた方策であることが分かる．また，遷移方策と一様方策との結果から，将棋においてはバランスを考慮せずに「強さ」を目指すことでも一定のバランスを得られることが分かる．最後に UCT プレイヤ同士の対戦結果を表 3.5 に示す．これより，遷移方策が最も実用的な性能の高い方策であることが分かる．なお，ダイレクトプレイヤや単純モンテカルロプレイヤ同士の対戦実験と比べると，一様方策の負け方は小さくなっている．これは，UCT により木を構成することでの全方策共通の棋力向上が，方策の違いによる棋力の差を上回っているためだと考えられる．実際，一様方策 UCT と，単純モンテカルロ法同士では最も強いバランシング方策による単純モンテカルロプレイヤの対戦を行うと，一様方策 UCT の勝率は $95.2 \pm 1.3\%$ となる．

3.2.3.2 特定の進行度間での対戦実験

3.2.2 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，ある 2 種類のプレイヤー A, B に対して，以下のように特定の進行度 m 以上 n 未満（$m \sim n$ と記す）の間での対戦実験を行った．

表 3.3: 各種方策のダイレクトプレイヤ同士の対戦結果（左列の方策の勝率（%））

	（一様）	（遷移）	（バランシング）
一様	—	0.9 ± 0.5	2.8 ± 0.8
遷移	99.1 ± 0.5	—	83.2 ± 1.9
バランシング	97.2 ± 0.8	16.8 ± 1.9	—

表 3.4: 各種方策の単純モンテカルロプレイヤ同士の対戦結果（左列の方策の勝率（%））

	（一様）	（遷移）	（バランシング）
一様	—	5.0 ± 1.1	3.7 ± 1.0
遷移	95.0 ± 1.1	—	38.4 ± 2.5
バランシング	96.3 ± 1.0	61.6 ± 2.5	—

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. プレイヤー A 対 プレイヤー B で局面を進める
3. 進行度が n 以上になったら，両プレイヤを深さ 6 のオリジナルの「激指」に変えて詰みまで対局を行い，結果を見る ($n = 128$ のときは，詰みまでプレイヤー A 対プレイヤー B)
4. (2) でプレイヤー A, B の先後を入れ替えて同様の対局を行う

以上の条件における遷移方策に対するバランシング方策の勝率を表 3.6 に示す．$0 \sim 96$ 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．また，中盤の $32 \sim 96$ においては棋力差は見られない．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 3.5: 各種方策の UCT 同士の対戦結果（左列の方策の勝率（%））

	（一様）	（遷移）	（バランシング）
一様	—	24.5 ± 2.2	27.4 ± 2.3
遷移	75.5 ± 2.2	—	55.0 ± 1.3
バランシング	72.6 ± 2.3	45.0 ± 1.3	—

表 3.6: UCT 同士での特定進行度間での対戦結果

	m	n	バランシング方策の勝率 (%)
序中盤	0	96	52.5 ± 2.1
中盤	32	96	50.1 ± 1.8
終盤	96	128	43.2 ± 3.1

3.2.4 進行度別の棋力差についての考察

3.2.3.1 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 同士で負け越す原因は，3.2.3.2 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．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．

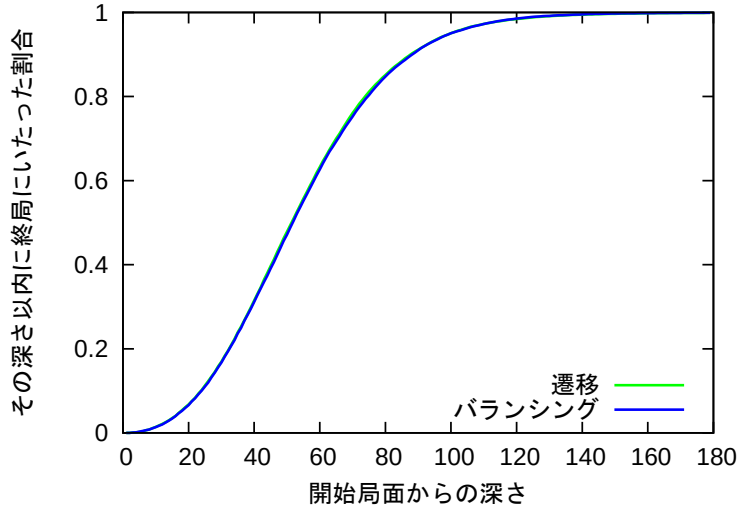
プレイアウト中の一手詰み探索のない対局条件では，遷移方策 UCT に対する勝率が 43.3 ± 1.3 に下がる．逆に言えば，一手詰み探索の導入による棋力への影響は バランシング方策の方が大きい．したがって，両方策には特に詰み付近に大きな違いがあるのではないかと考え，その点について調査を行うこととした．

まず，プレイアウトの一手詰み局面への至り易さについて調べた．具体的には，3.2.1 節と同様の方法で，終局率について終盤とそれ以外に分けて評価を行った．結果を図 3.12 に示す．終盤の結果である図 3.12(b) より，遷移方策の方がより短い手数で一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「終盤付近で一手詰み局面へ至るのが遅い」と言える．一方で，図 3.12(a) より序中盤においては両方策の終局率にほとんど差は見られない．以上が，バランシング方策の終盤付近での弱さにつながっている可能性がある．

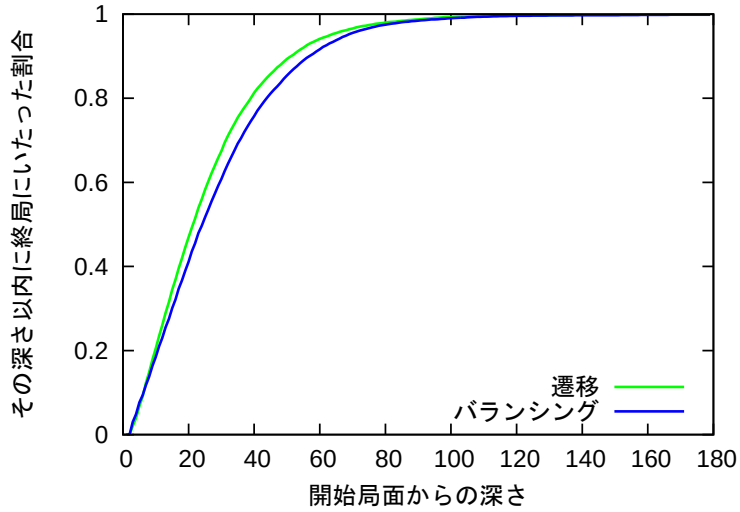
次に，実際に詰み手順に入ったときに，どの程度正しく指せるかについて調べた．具体的には，プレイアウト中に 3 手詰みの局面に至った場合に，どの程度の割合で詰み手順の最初の一手を正しく指せるかを調べた．これを，各方策ごとに約 18,000 局面[6]からのプレイアウトで実行した結果，バランシング方策は $10.97 \pm 0.45\%$ の割合で，遷移方策は $20.15 \pm 0.57\%$ の割合で正しく指せることが分かった．こうした詰み手順での不正確さは，終盤付近の弱さに直結する要素だと考えられる．実際に，プレイアウトに 3 手詰み探索を導入したところ，初手から詰みまでの対局条件では，遷移方策 UCT に対する勝率は 46.19 ± 1.4 となり，勝率はわずかに上昇した．なお，プレイアウトの打ち切り深さがより深い場合には，これらの詰み探索の効果がより増すものと考えられる．

また，棋譜の指し手との一致率を進行度別に調べたところ，図 3.13 のような結果になった．まず図 3.13(a) よりダイレクトプレイヤによる一致率を見ると，いずれの進行度においても遷移方策が良

[6]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 18,000 局面であった．99%以上の局面は 128 手以内に一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一手詰み局面に至るような自殺手を指したことになる

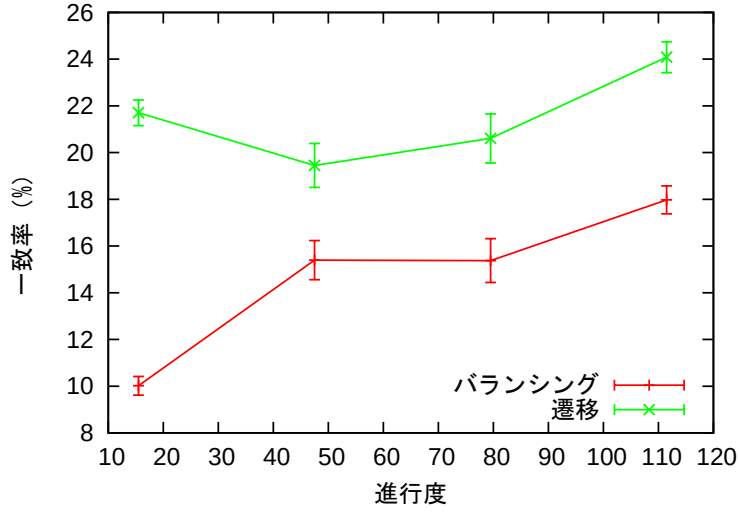


(a) 序中盤（進行度：0 ～ 96）

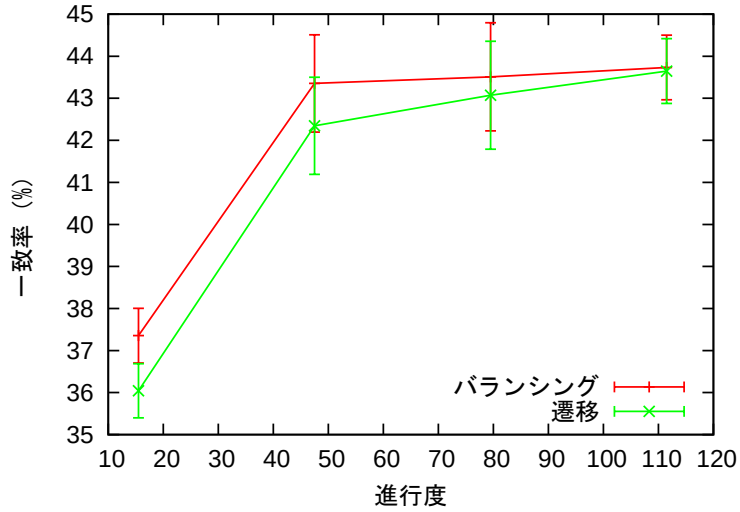


(b) 終盤（進行度：96 ～ 127）

図 3.12: 序中盤・終盤における遷移方策とバランシング方策の終局率の比較



(a) ダイレクトプレイヤ



(b) UCT プレイヤ

図 3.13: 進行度別の一致率

い性能を示している．一方で図 3.13(b) より UCT プレイヤによる一致率を見ると，序盤では若干バランシング方策の性能が良いものの，終盤を含め全体としては両方策で有意差は見られない．したがって，バランシング方策は終盤においても平均的には遷移方策と同程度に良い手を選べるものの，勝率への寄与が大きい局面で悪手を選んでいると推測できる．そして，ダイレクトプレイヤでの結果から，そうした局面では一手の精度が重要な場合が多いと推測できる．前述の詰み付近はそのような局面の一種だといえる．

3.3 ミニマックス探索との比較

3.3.1 中盤や終盤それぞれでの棋力比較

3.2.3.2 節で述べた特定の進行度間での対局により，各方策についてプレイアウト数を変えながら深さ 6 のオリジナル「激指」に対する勝率を調べた．結果を図 3.14 に示す．図 3.14(a) を見ると，中盤から終盤にかけての全体的な棋力では遷移方策がバランシング方策よりも若干良い性能を示している．中盤付近の結果である図 3.14(b) からは，遷移方策とバランシング方策ではほぼ同等の強さであることが分かる．ただし，終盤の結果に比べると方策間の差異は明確ではない．逆に，図 3.14(c) より，終盤付近では遷移方策のほうがバランシング方策に比べて相対的に強いことが分かる．これらの結果は，方策間の直接対戦を行った 3.2.3.2 節と一致する．

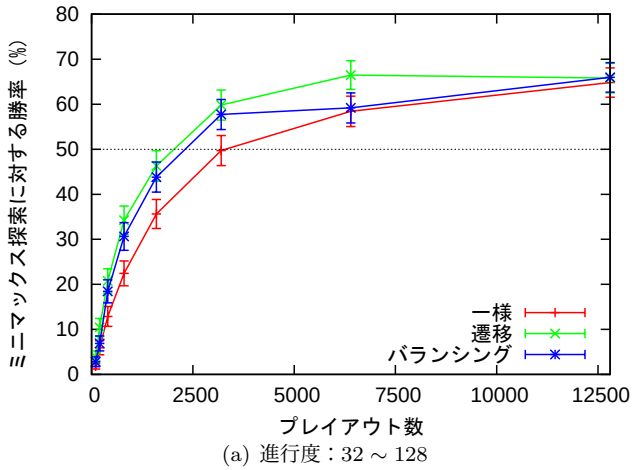
モンテカルロ木探索とミニマックス探索との比較という観点から，両者の棋力が拮抗するプレイアウト数に着目する．すると，中盤付近ではおよそプレイアウト数 2,000 回程度で勝率が 50% となる一方で，終盤付近では約 3,000 回のプレイアウトを要している．これより，モンテカルロ木探索は相対的には中盤を得意とし，終盤を苦手とすることが分かる．なお，遷移方策とバランシング方策に関しては，図 3.14(a) より全体的な棋力が深さ 6 のミニマックス探索と一致するプレイアウト数は 2,500 回程度だと分かる．このため，以降の比較においてはモンテカルロ木探索のプレイアウト回数を 2,500 回とした．

3.3.2 浅いトラップ

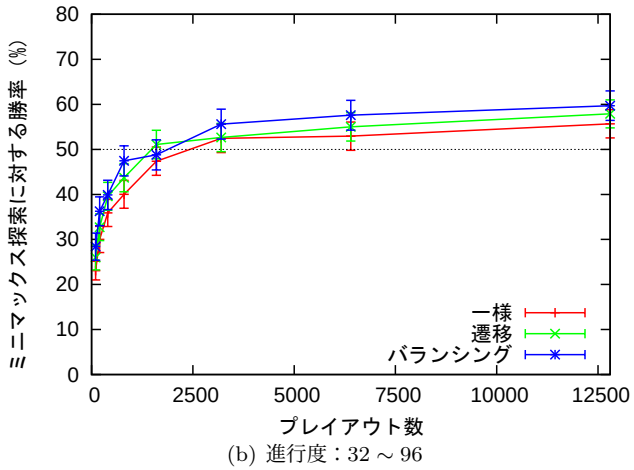
2.6 節で述べたように，多数の浅いトラップの存在がモンテカルロ木探索の棋力を落としている原因の一つであるとの指摘が，チェスにおいてなされている [16]．将棋はチェスと同じ起源を持つゲームであり，同様に多くの浅いトラップが存在する可能性が考えられる．そのため，将棋における浅いトラップの存在と，トラップへのかかりやすさを調べた．

プロの棋譜からランダムに抽出した 50,000 局面を用い，進行度別にレベル $1, 3, 5, 7$ の浅いトラップの出現割合を調べたものが図 3.15 である．これより，チェス [16] における結果と比較して[7]，将

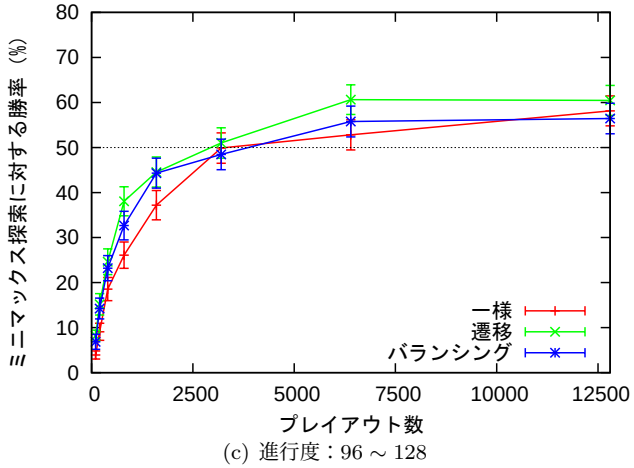
[7]チェス [16] では，進行度ではなく初期局面からの手数で分類を行なっているため正確な比較はできない．参考までに，



(a) 進行度：32 ～ 128

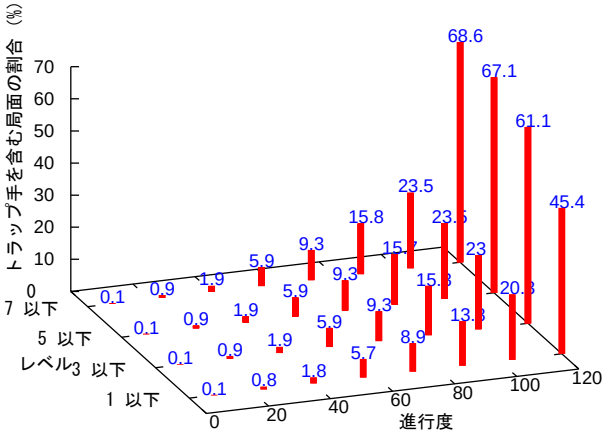


(b) 進行度：32 ～ 96

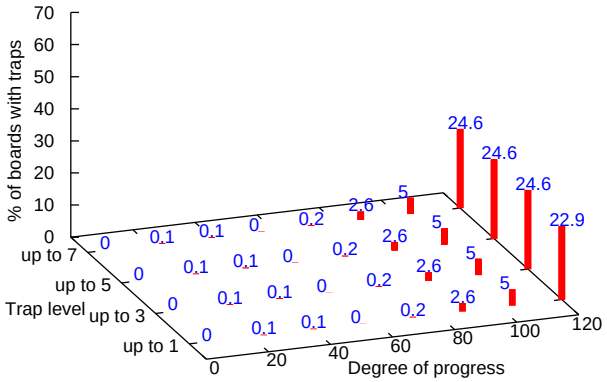


(c) 進行度：96 ～ 128

図 3.14: 深さ 6 のミニマックス探索に対する，各方策の UCT の勝率の推移（進行度別）



(a) 全ての合法手を利用



(b) 手を遷移確率の高い 7 手に限定

図 3.15: 将棋における浅いトラップの出現割合

棋は非常に多くの浅いトラップを持つゲームであることが分かる．なお，3.1.2 節での CEM によるパラメータ調整の結果から，本稿ではモンテカルロ木探索の候補手を遷移確率の高い 7 手に絞っている．このため，全ての合法手を用いる場合に比べ，トラップ手の出現割合は下がると考えられる．実際に，こうした 7 手に候補手を制限した場合についても調べた結果，図 3.15(b) のようになった．確かに，全ての候補手を用いる場合に比べて出現割合は下がっているものの，依然として多くのトラップ手を含むことが分かる．ただし，レベル 1 や 3 の相対的に浅いトラップの存在が支配的であり，レベル 5 や 7 の相対的に深いトラップの存在は少ないことも見てとれる．

実際にどの程度トラップ手を避けられるのかを評価した．まず，UCT が最善と判断した手（以下「最善手」と呼ぶ）に与えた評価値と，トラップ手に与えた評価値とを比較し，平均的にどの程度トラップ手を区別できているのかを調べた．具体的には，トラップ手を含む 3,000 局面それぞれから 2,500 回のプレイアウトを行い，各手の評価値を比較した．結果を図 3.16(a) に示す．深いトラップほど「最善手」とトラップ手との評価値が近づいているものの，平均的にはトラップを区別できていることが分かる．次に，実際にトラップの存在する局面において，トラップ手を選んだ割合を図 3.16(b) に示す．これより，相対的に深いトラップほど選んでしまいやすいことが分かる．相対的に浅いレベル 1 や 3 のトラップはそれらよりは正しく避けているものの，それでも 1% 程度の場合は選んでしまうことが分かる．

3.3.3 局面の「明確さ」に対する得手・不得手

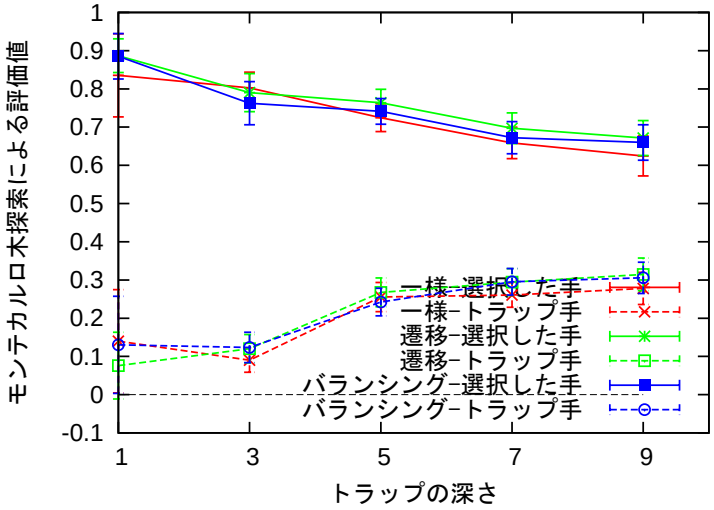
局面の「明確さ」に対して，どの程度良い手を選ぶことができるかを比較した．前節で述べたトラップへのかかりやすさは大きな問題ではあるものの，詰み探索の導入により多くの場合は解決可能だと考えられる．そこで本節では，局面の「明確さ」に着目し，より一般的に「明らかに」良い手を見逃して悪い手を選んでしまう可能性について調べる．

この「明確さ」の指標として，深いミニマックス探索で得られた最善手と次善手の評価値の差を用いる．差が大きい局面は最善手が「明確な」局面であり，小さい局面は最善手が「曖昧な」局面であると考える．この局面の「明確さ」は深さ 14 のミニマックス探索によって評価する．

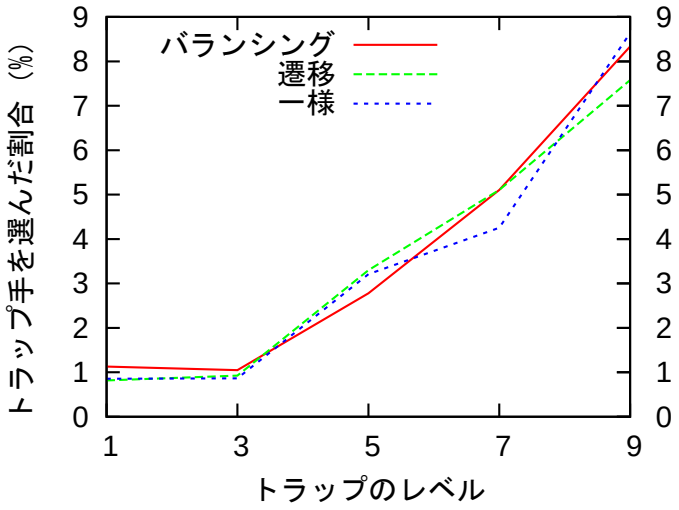
プロの棋譜から抽出したある「明確さ」の局面において，どの程度正しい手を指せたかを示したものが図 3.17，3.18 である．縦軸は，プロによる指し手との一致率を表す．なお，UCT のプレイアウト回数は 2,500 回とした．これらの図より，UCT かミニマックス探索かを問わず,「明確な」局面ほど一致率が高くなることが分かる．局面の明確さを測る上で，最善手と次善手の評価値差を用いることは妥当であるといえる．ただし，特に「明確な」局面において，中盤と終盤の一致率の間には大きな相違があり，指標としてこれだけでは不十分な場合もあると考えられる．

図 3.17，3.18 それぞれについて見ていく．まず，UCT の各種方策間で比較を行った図 3.17 につい

チェスにおいては，調べられていた中で最も深い 63 手目の局面において，レベル 5 以内のトラップの出現割合が約 17.00%であった



(a) 「最善手」とトラップ手それぞれ評価値の平均



(b) トラップ手を選んだ割合

図 3.16: トラップへのかかりやすさの評価

て見る．中盤（図 3.17(a)）に関しては，どの方策でも傾向に差は見られない．終盤（図 3.17(b)）に関しては,「曖昧な」局面ではバランシング方策がわずかに良い値を示している一方で,「明確な」局面ではバランシング方策の一致率が相対的に低くなり，遷移方策が最も良い値となった．

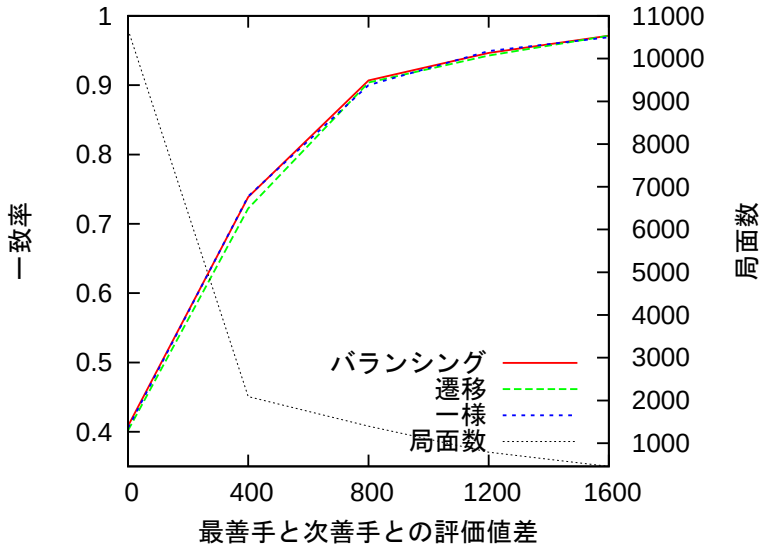
次に，UCT（バランシング方策）とミニマックス探索との比較を行った図 3.18 について見る．中盤（図 3.18(a)）に関しては,「曖昧な」局面では深さ 6 のミニマックス探索よりも良い一致率を示しているが,「明確な」局面ではそれほど差がなくなっている．ただし,「明確な」局面では深さ 6 と 8 のミニマックス探索との間の差も少なく，ミニマックス探索では中盤の「明確な」局面は浅い探索で十分に良い指し手の選択ができていることが分かる．終盤（図 3.18(b)）に関しては,「曖昧な」局面では深さ 6 のミニマックス探索よりも良い性能を示しているものの,「明確な」局面になると深さ 6 のミニマックス探索と比べても悪い結果となっている．図 3.17(b) の結果を見ても，方策にかかわらずモンテカルロ木探索にはミニマックス探索と比べて,「曖昧な」局面では比較的性能が良く,「明確な」局面では性能が悪いという傾向があることが分かる．

ここで「明確な」局面は，間違えた手を選ぶと大きく評価値を落とし，不利になる局面と見ることもできる．このため，モンテカルロ木探索が終盤付近における「明確な」局面で間違えやすいことが，ミニマックス探索に対する弱さにつながっていると考えられる．

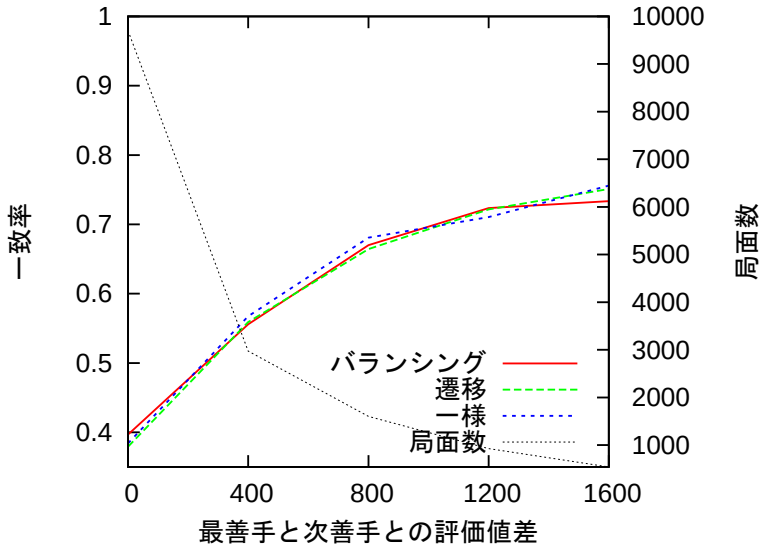
3.4 解析のまとめ

3.2 節では方策間での比較を行なった．対戦実験では,「強さ」を目指して学習した遷移方策が実際に「強い」方策であったこと，バランスを目指して学習したバランシング方策が実際にバランスのとれた方策であったこと，そして UCT における実用的な性能では遷移方策が優れていることを示した．また，遷移方策でも一定のバランスを得られていることや，バランシング方策でも一定の「強さ」を得られていることを示した．進行度別の対戦実験では，バランシング方策が序中盤を得意とする一方で，終盤を苦手とすることを明らかにした．

3.3 節ではモンテカルロ木探索とミニマックス探索との比較を行った．進行度別の対戦実験では，モンテカルロ木探索は終盤を苦手としていることを明らかにした．より具体的な局面として，浅いトラップについての解析では，チェスにおける結果 [16] と同様にモンテカルロ木探索はトラップが深くなるほど避けるのが困難になることを確認した．こうした詰み局面に限らないより一般的な解析として，局面の「明確さ」に着目した解析も行なった．ここでは，モンテカルロ木探索が，終盤において最善手が「明らか」な局面で間違えやすいことを示した．これは将棋におけるモンテカルロ木探索の弱さにつながっていると考えられる．

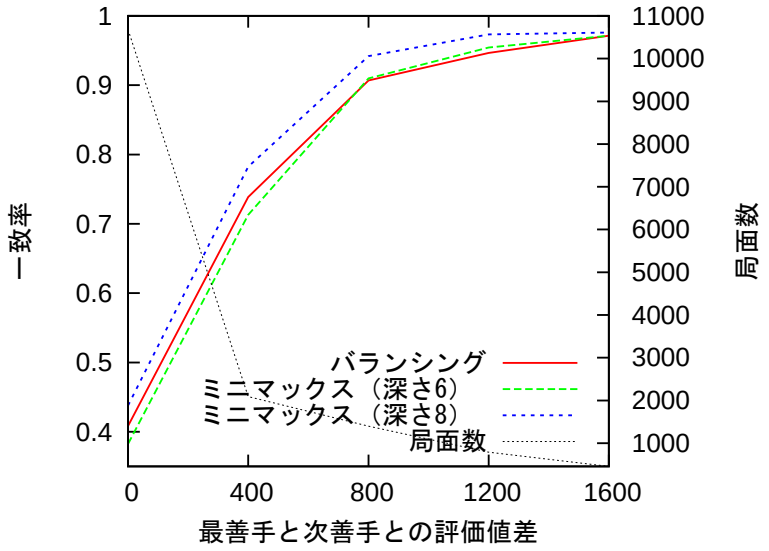


(a) 進行度：32 ～ 96

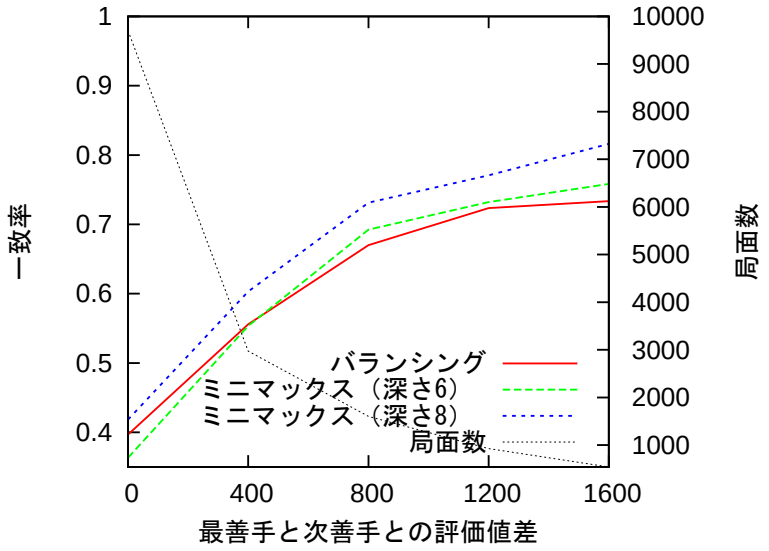


(b) 進行度：96 ～ 128

図 3.17: 局面の「明確さ」対する一致率の推移（UCT：各方策）



(a) 進行度：32 ～ 96



(b) 進行度：96 ～ 128

図 3.18: 局面の「明確さ」対する一致率の推移（バランシング方策 UCT，ミニマックス探索）

第4章　方策の学習手法の提案

4.1　学習局面を限定したシミュレーション・バランシング

学習局面の選び方の変更によるバランシング方策の改善について検証する．3.3.1 節や 3.3.3 節の結果から，バランシング方策は中盤では比較的良い指し手の選択ができていると言える．しかし，学習に用いた 8,000 局面[1]のうち，進行度が 32 ～ 96 となる局面は 3 割にも満たなかった[2]．そこで中盤付近の棋力に関しては，進行度が 32 ～ 96 となる局面のみを学習に用いることで改善できる可能性がある．一方，バランシング方策の苦手とする終盤（進行度：96 ～ 128）についても，学習局面は全体の 3 割に満たない．したがって中盤の場合と同様に，終盤の局面のみであらためて学習を行うことで，終盤の棋力を改善できる可能性がある．仮にそれぞれで棋力の向上が見られれば，進行度に従った各方策の統合などにより全体的な棋力の向上が期待できる．

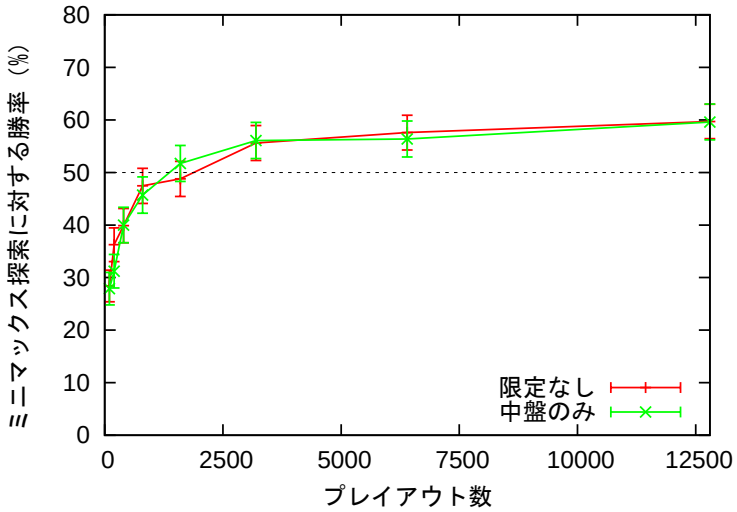
実際に，中盤・終盤それぞれに限って 8,000 局面を抽出し直してバランシング方策の再学習を行い，元のバランシング方策との比較を行った．比較は深さ 6 のオリジナル「激指」に対する勝率を見ることで行い，このとき対戦はそれぞれの学習で用いた進行度に限定して行う．まず，中盤 8,000 局面を用いて再学習を行った結果が図 4.1(a) である．これより，中盤での棋力は向上見られないことが分かる．終盤 8,000 局面の結果である図 4.1(b) からも，同様に終盤での棋力の向上は見られない．以上の結果から，中盤や終盤付近でより良く指すためには，単純に学習局面を進行度で限定するだけでは不十分だといえる．

4.2　方策の「強さ」とバランスをともに考慮した学習手法

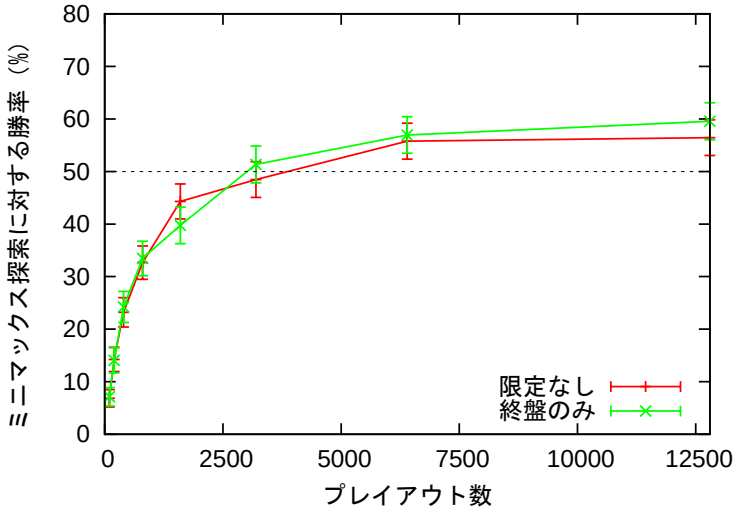
4.2.1　提案手法

ここまでの結果から，将棋においては「強い」方策が良い場面が多いことが分かる．しかし，囲碁における結果 [18, 13] から必ずしも汎用性のある方策であるとは言えず，また，学習において直接的にはバランスを考慮していないため，将棋においてもまだ改善の余地があると考えられる．実際，ここまでの解析でバランシング方策と比べ序中盤を苦手とすることを示した．よって，あらゆる場面でバランシング方策よりも優れているわけではないと言える．そこで本節では,「強い」方策をも

[1] プロの棋譜から，初手や最終手から 10 手程度を除いてランダムに抽出
[2] 「激指」は局面の進行度を 0 もしくは 127 と判断する場合が多いため



(a) 中盤のみで学習，中盤での対戦結果（進行度：32 ～ 96）



(b) 終盤のみで学習，終盤での対戦結果（進行度：96 ～ 128）

図 4.1: 学習局面を限定したことによる棋力への影響

とに，一定の「強さ」を保ちつつよりバランスのとれた方策を学習する手法を提案する．以下では「強い」方策を π_s とおく．

バランシングの目的関数である，式 2.13 で表されるインバランス $B(\theta)$ にクロスエントロピーの項 $H(\pi_\theta, \pi_s)$ を加え，学習の目的関数を式 4.1 のように定義する．

$$\begin{aligned} B'(\theta) &= \mathbf{E}_\rho \left[\left(V^*(s) - \mathbf{E}_{\pi_\theta}[z|s]\right)^2 + H(\pi_\theta, \pi_s) \right] \\ &= \mathbf{E}_\rho \left[\left(V^*(s) - \mathbf{E}_{\pi_\theta}[z|s]\right)^2 - \sum_b \pi_s(s,b) \log(\pi_\theta(s,b)) \right] \end{aligned} \tag{4.1}$$

こうすることで,「強い」方策 π_s の確率分布によって制約をかけながらバランスのとれた方策 π_θ を学習することができる．学習局面ごとの更新式は右辺を θ によって微分し，更新パラメータ α, β を加えることで以下のように書ける．

$$\theta \leftarrow \theta + \alpha \left((1-\beta)(V^* - V)g + \beta \sum_b \pi_s(s,b)\psi_\theta(s,b) \right) \tag{4.2}$$

このとき，$\beta = 0$ による強バランシング方策は通常のバランシング方策と同一であり，β の値が大きいほど遷移方策に近づく．

4.2.2 設定と評価

更新式 4.2 を用いた学習と方策の評価を行った．パラメータの設定について述べる．β については複数の値で学習した結果を示す．また,「強い」方策 π_s としては遷移方策を用いる．その他のパラメータは 3.1.3 節 で述べたバランシング方策の学習と同様に設定し，$\alpha = \frac{1}{t}$（t はイテレーション回数）とした．初期パラメータについても，同様に重みをすべて 0 としている．以下，このように学習して得られた方策を強バランシング方策と呼ぶ.

以下強バランシング方策の評価を行う．まず，3.2.3.1 節と同様に遷移方策との棋力差をダイレクトプレイヤ，単純モンテカルロプレイヤ，UCT プレイヤ同士の対戦実験で比較した．複数の β による結果を図 4.2 に示す．これより，β が大きいほど「強い」方策となる（ダイレクトプレイヤの結果）一方で，バランスを失っていることが分かる（単純モンテカルロプレイヤの結果）．UCT における実用的な性能に着目すると，$\beta = 0.1 \sim 0.8$ で遷移方策と同程度の棋力となっていることが分かる．したがって，バランシング方策に比べれば実用的な性能の高い方策となったことが分かる．

なお，$\beta = 1$ の場合はバランスの項が反映されず,「強さ」の項だけが残る．しかし，棋力は元の遷移方策に比べて大きく劣っている.「激指」で 1,769,674 局面を利用して学習していることや重みを 0 から学習していることを考えると，これは学習局面数が少ないためだと考えられる．

次に，3.2.3.2 節で述べた特定の進行度間での対局を UCT プレイヤ同士で行い，序中盤（進行度 $0 \sim 96$）と終盤（進行度 $96 \sim 128$）それぞれで棋力の比較を行なった．結果を図 4.3 に示す．これより，$\beta = 0 \sim 0.2$ 程度の部分では，バランシングと比べて序中盤での棋力を損なわずに，終盤での棋力が増していることが分かる．遷移方策と比べると，序中盤に強く終盤に弱い方策であることが分かる．

以上の結果より，強バランシングの実用的な性能は遷移方策と並んではいるものの，超えるものとはなっていない．しかし，遷移方策とは異なる特性を持っている点（進行度別の対戦実験より）や，$\beta = 0.1 \sim 0.2$ 程度ではバランシングの利点，すなわちバランスがとれていることや序中盤に強いといった利点を損なわずに「強さ」と実用的な性能を向上させている点は注目に値する．囲碁において単純に「強い」方策が良い性能を発揮できないことを考え合わせると，適用するゲームによっては良い性能を発揮する可能性は十分に考えられる．

また，3.2.2 節で述べた手選択確率の偏り $D(s)$ の評価も行った．結果を図 4.4 に示す．遷移方策の結果（図 3.11(a)）とバランシング方策の結果（図 3.11(b)）の中間的な結果になっていることが分かる．

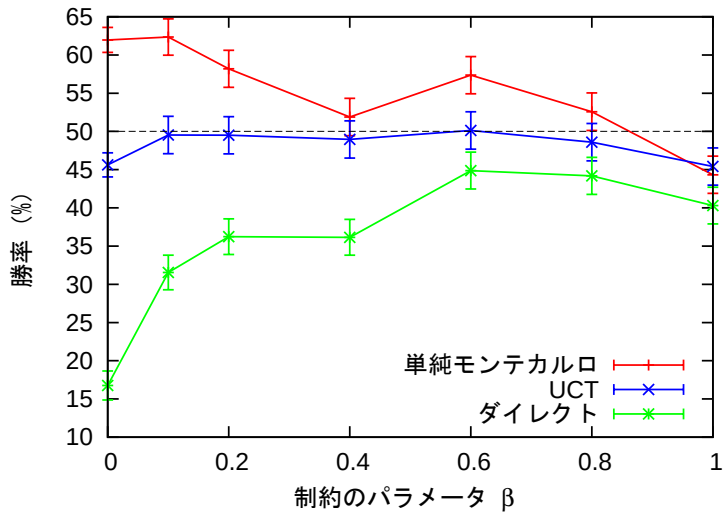


図 4.2: 複数の β による強バランシング方策の遷移方策に対する勝率

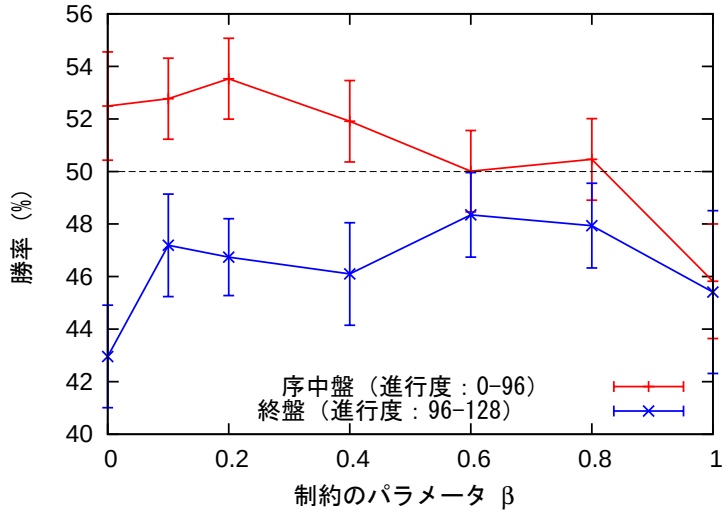


図 4.3: 複数の β による強バランシング方策の遷移方策に対する進行度別の勝率（UCT プレイヤ同士での比較）

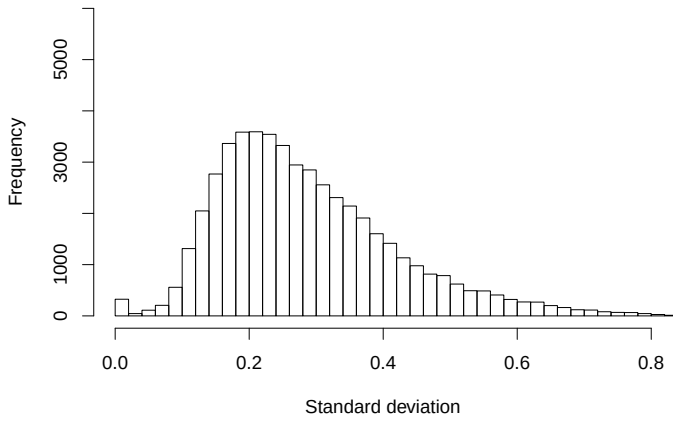


図 4.4: 手選択確率の偏りの分布（$\beta = 0.2$）

第5章　結論

5.1　まとめ

本研究では，将棋におけるモンテカルロ木探索の特性の解明を行った．このとき，シミュレーションの方策の違いやミニマックス探索と比較した得手・不得手を明らかにした．また，以上の解析にもとづいた学習手法の提案を行なった．

複数の方策間の比較においては遷移方策とバランシング方策という二種類の方策を用いた．前者は一手一手の精度の高い「強い」方策を目指したものであり，後者はプレイアウトの繰り返しで得られる平均報酬が真の値であるミニマックス値に近いバランスのとれた方策を目指したものである．これらの比較から以下の点を明らかにした．

- UCT における実用的な性能が良いのは「強い」方策を目指した遷移方策である
  - 将棋においては「強さ」を目指すことでも一定のバランスが獲得された．これは囲碁における結果とは必ずしも一致しない
- ただし，序中盤ではバランシング方策の性能が良く，終盤において遷移方策に大きく劣っていた
  - バランシング方策が終盤の詰み付近を苦手とすることを示した．こうした終盤における重要な局面を苦手とすることが，全体的な性能の悪さにつながっていると考えられる

ミニマックス探索との比較においては以下の点を明らかにした．

- モンテカルロ木探索は相対的に中盤を得意とし，終盤を苦手とする
- 最善手が「曖昧な」局面ではモンテカルロ木探索は比較的良い手選択を行えている一方で，特に終盤の最善手が「明確な」局面が苦手である
  - 後者の局面は勝敗への寄与が大きいと考えられる．それは,「明確な」局面は最善手以外を選んだ場合に大きく評価値が下がる局面となっているからである．よって，これが将棋におけるモンテカルロ木探索の弱さにつながっていると考えられる．

学習手法の提案においては，二種類の手法の提案と評価を行った．まず，シミュレーション・バランシングにおいて中盤（終盤）の局面のみを用いて学習を行い，中盤（終盤）の棋力の向上を図った．

これは，バランシング方策が中盤では比較的良く終盤で悪いという解析の結果がある一方で，学習局面に中盤・終盤局面が混ざっており，十分な性能が発揮されていないのではないかと考えたためである．しかし，このように単純に学習局面を限定する方法では棋力の向上は見込めなかった．次に，遷移方策とバランシング方策それぞれに得失があるという解析の結果から，前者の目指す一手一手の精度と後者の目指す評価値の偏りの少なさを，ともに考慮した学習手法の提案を行った．UCT における実用的な性能は遷移方策と同程度にとどまったものの，バランシング方策との比較においては，バランシング方策の利点を損なわずに性能を向上させることに成功した．このため，モンテカルロ木探索に導入するヒューリスティックとの相性や，対象とするゲーム次第では十分に有用な学習手法となり得るといえる．

5.2 課題

解析に関しては，本研究で行なってきた統計的な解析に加え，今後は具体的な局面における特性を調べることが必要だと考える．これは，方策間の比較についても，ミニマックス探索との比較についても言えることである．こうしたより詳細な方向での解析とは逆に，将棋以外の問題に対しても今回の解析と類似の解析を行うことができれば，より一般的な知見が得られるものと期待できる．また，本研究では方策の違いに着目したが，解析手法そのものはモンテカルロ木探索一般での改善手法の評価に用いることも可能だと考える．

学習手法に関しては，学習局面数など様々なパラメータを調整する余地が残る．その一環として，初期値を「強い」方策から始めた場合についても調べたい.「強い」初期値を作成するためには，強バランシングの更新式 4.2 において，$\beta = 1$ すなわち「強さ」の項のみで多数の局面における学習を行えば良い[1]．学習局面数については，通常のシミューレーション・バランシングでは，大幅に増やすことは難しい．平均報酬や平均勾配を求めるためのシミュレーションに時間がかかるためである．しかし,「強さ」の項のみで更新を行う場合にはそうした制約はないため，多くの局面を用いて学習を行うことができる．また，解析と同様に将棋以外のゲームへの適用により，その有用性を確かめたい．その中では，適切なパラメータ β をいかに求めるかという点が課題になる．例えば，シミュレーション・バランシングによるバランスのとれた方策と教師の間の二乗誤差と,「強い」方策と教師の間の一致率を事前に求め，これらの値をもとにある程度自動的に適切なパラメータを推定する方法が考えられる．

[1]学習における更新方法の違いから，遷移確率における重みをそのまま強バランシング方策の初期値とするのは適当ではない

参考文献

[1] B. Abramson. Expected-outcome: a general model of static evaluation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, Vol. 12, No. 2, pp. 182 –193, Feb. 1990.

[2] P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, Vol. 47, No. 2, pp. 235–256, 2002.

[3] P. Baudiš. Balancing mcts by dynamically adjusting komi value. *ICGA Journal*, Vol. 34, No. 3, 2011.

[4] C. Browne, E. Powley, D. Whitehouse, S. Lucas, P. Cowling, P. Rohlfshagen, S. Tavener, D. Perez, S. Samothrakis, and S. Colton. A survey of monte carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in Games*, Vol. 4, No. 1, pp. 1–43, 2012.

[5] R. Caruana and A. Niculescu-Mizil. An empirical comparison of supervised learning algorithms using different performance metrics. In Proc. 23 rd Intl. Conf. Machine learning (ICML '06, pp. 161–168, 2005.

[6] G.M.J.-B. Chaslot, M.H.M. Winands, H.J. van den Herik, J.W.H.M. Uiterwijk, and B. Bouzy. Progressive strategies for monte-carlo tree search. *New Mathematics and Natural Computation*, Vol. 4, No. 3, pp. 343–357, 2008.

[7] G.M.J.-B. Chaslot, M.H.M. Winands, I. Szita, and H.J. van den Herik. Cross-entropy for monte-carlo tree search. *ICGA Journal*, pp. 145–156, 2008.

[8] R. Coulom. Efficient selectivity and backup operators in monte-carlo tree search. In *In: Proceedings Computers and Games 2006*, pp. 72–83. Springer-Verlag, 2006.

[9] R. Coulom. Computing elo ratings of move patterns in the game of go. *ICGA journal*, Vol. 30, No. 4, pp. 198–208, 2007.

[10] S. Gelly and D. Silver. Combining online and offline knowledge in uct. In *Proceedings of the 24th international conference on Machine learning*, ICML '07, pp. 273–280, New York, NY, USA, 2007. ACM.

[11] S. Gelly and D. Silver. Monte-carlo tree search and rapid action value estimation in computer go. *Artificial Intelligence*, Vol. 175, No. 11, pp. 1856–1875, Jul. 2011.

[12] S. Gelly, Y. Wang, R. Munos, and O. Teytaud. Modification of UCT with patterns in monte-carlo go. Rapport de recherche RR-6062, INRIA, 2006.

[13] S. Huang, R. Coulom, and S. Lin. Monte-carlo simulation balancing in practice. In *International Conference on Computers and Games*, pp. 81–92, 2010.

[14] L. Kocsis and C. Szepesvári. Bandit based monte-carlo planning. *Machine Learning: ECML 2006*, pp. 282–293, 2006.

[15] T. Lai and H. Robbins. Asymptotically efficient adaptive allocation rules. *Advances in Applied Mathematics*, pp. 4–22, 1985.

[16] R. Ramanujan, A. Sabharwal, and B. Selman. On adversarial search spaces and sampling-based planning. *ICAPS-10: 20th International Conference on Automated Planning and Scheduling*, pp. 242–245, 2010.

[17] R. Ramanujan and B. Selman. Trade-offs in sampling-based adversarial planning. *In ICAPS-11: 30th International Conference on Automated Planning and Scheduling*, 2011.

[18] D. Silver and G. Tesauro. Monte-carlo simulation balancing. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pp. 945–952, New York, NY, USA, 2009. ACM.

[19] Seigo Takeuchi, Tomoyuki Kaneko, and Kazunori Yamaguchi. Evaluation of monte carlo tree search and the application in go. *IEEE Conference on Computational Intelligence and Games*, pp. 191–198, 2008.

[20] Seigo Takeuchi, Tomoyuki Kaneko, and Kazunori Yamaguchi. Evaluation of game tree search methods by game records. *IEEE Conference on Computational Intelligence and Games*, pp. 288–302, 2010.

[21] 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, Nov. 2008.

[22] 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.

[23] 竹内聖悟, 金子知適. 探索パラメータの調整に適した目的関数の調査 - モンテカルロ木探索将棋の探索パラメータの調整 -. ゲームプログラミングワークショップ 2012 論文集, No. 6, pp. 84–91, Nov. 2012.

[24] 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, Nov. 2010.

[25] 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.

[26] 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.

[27] 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.

発表文献

査読付会議論文

1. 関 栄二, 三輪 誠, 近山 隆, “モンテカルロ将棋における方策の学習”, 第 16 回ゲームプログラミングワークショップ, pp. 104-108, Nov. 2011.

2. 関 栄二, 三輪 誠, 鶴岡 慶雅, 近山 隆, “将棋におけるモンテカルロ木探索の特性の解明”, 第 17 回ゲームプログラミングワークショップ, pp. 68-75, Nov. 2012. ＊ゲームプログラミングワークショップ研究奨励賞受賞

論文誌

1. 関 栄二, 三輪 誠, 鶴岡 慶雅, 近山 隆, “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”, 情報処理学会論文誌, Vol.53, No.11, pp.2533-2543, Nov. 2012.

謝辞

本研究を進めるにあたり，多くの方々に世話になりました．

指導教員である近山隆教授には，発表や論文の提出の際など要所要所で含蓄のあるアドバイスを頂きました．研究の根本に関わるような助言・質問も多く，研究の表層に限らない様々なことを考えさせられました．

鶴岡慶雅准教授には，激指を含む将棋一般や学習についての多くの助言を頂きました．将棋や学習の研究において普通はどうするものなのか，といった知見は中々に得がたいもので，大変ありがたかったです．

研究室 OB の三輪誠さんには，論文や資料の作り方の他，日頃からの様々な助言など多数の面で大変お世話になりました．優先順位をつけ，立ち止まることも多い中で少しでも円滑に研究を進めていく上で，三輪さんの助言には非常に大きいものがありました．

博士課程の浦晃さんには，特に研究室に配属されたばかりの何から手をつけて良いか分からない時期に，大変お世話になりました．研究の方向性についてや，激指や計算環境の使い方など技術的な面での多くの助言も頂き，研究を進める上で大いに役立ちました．

同期の古居敬大君とは，普段から様々な雑談や研究についてのとりとめのない話などもし，楽しんで過ごしたり研究について色々と考えたりすることができました．また，研究室の皆様や金田研の同期で同じく将棋の研究をしている鈴木君など多くの方々のお世話になりました．

最後に，実家暮らしながら不規則な生活を続け様々に迷惑をかけたと思いますが，食事などの面で健康に研究生活を送れたことに家族には感謝しています．

ここに，お世話になった皆様への心からの感謝の意を表します．

平成 25 年 2 月 6 日