評価関数の違いがモンテカルロ木探索プレイヤの強さに与える影響

北村 直輝[1,a)]　松崎 公紀[1,b)]

概要：2006 年に R. Coulom によって提案されたモンテカルロ木探索は，囲碁などを中心にコンピュータプレイヤの強さを大きく向上した．モンテカルロ木探索をより効果的にする手法として，木探索部分における盤面の評価において，または仮想的に終局までプレイするプレイアウトにおいて評価関数を用いる手法がある．コンピュータ囲碁プレイヤ AlphaGo ではニューラルネットワークによる評価関数を併用して，プロ棋士との対戦で勝利するという結果を出している． より正しく盤面を評価するような評価関数を用いることでプレイヤの強さが向上すると期待されるが，評価関数の違いがモンテカルロ木探索プレイヤの強さに与える影響は，著者が知る限り，これまで明らかにされていない． 本研究では，優れた評価関数が既に得られているオセロを対象にこの問題に取り組む．評価関数にはオープンソースのオセロプログラム「Zebra」の評価関数及びそれを改変した複数の評価関数を用いる．これらの評価関数をモンテカルロ木探索に適用して作成したプレイヤの勝率を観察しより正しい評価関数を用いることの優位性を検証する．検証の結果，Zebra の評価関数を改悪して作ったモンテカルロ木探索プレイヤは，より低い勝率となった．また，評価関数の使い方によって勝率への影響が変化することがわかった．

キーワード：オセロ，モンテカルロ木探索，評価関数

1. はじめに

ゲームプログラミング分野において，2006 年に R. Coulom によって提案されたモンテカルロ木探索は，特に囲碁を中心としてコンピュータゲームプレイヤの強さを大きく向上させた．本研究は，そのモンテカルロ木探索の改良法について，その性質を調査するものである．

二人零和完全確定情報ゲームに属するゲームでは，minimax 法と呼ばれる手法によりゲームを解析することができる．オセロやチェスにおいては，探索ノードを枝刈りすることで高速に探索を行うアルファベータ法と，ゲーム局面の有利不利を数値化する評価関数とを組み合わせることで，人間のトッププレイヤに匹敵するほどの強さになる．しかし囲碁はそれらのような従来手法ではさほど強くならなかった．その理由のひとつは，探索空間が広く現在のコンピュータでは勝敗が決定する局面まで探索するのに，非現実的な時間を要することである．また，局面を評価するための要素が少なく，評価関数を作りにくいという点も理由に挙げられる [1].

2006 年にモンテカルロ木探索を用いる囲碁プレイヤ「Crazy Stone」が Computer Olympiad で優

[1] 高知工科大学
[a)] 180319a@ugs.kochi-tech.ac.jp
[b)] matsuzaki.kiminori@kochi-tech.ac.jp

勝し，さらに L. Kocsis ら [2] によって UCT アルゴリズムへと改良されたことで，モンテカルロ木探索はコンピュータ囲碁を中心に，ゲームプログラミング分野で多く用いられるようになった．特に，David Silver らによるコンピュータ囲碁プレイヤ「AlphaGo」は，主にニューラルネットワークを用いて実現された評価関数とモンテカルロ木探索によって実現されており，囲碁の世界王者に勝利する結果を出している [3].

モンテカルロ木探索では，ある局面から仮想的に終局までプレイするプレイアウトを繰り返し行う．理論的には，無限回のプレイアウトを行うことにより，最適解に収束することが示されている．しかし現実には，ある程度の時間内に良い解を返すことが必要である．そのような目的のため，プレイアウトや局面の評価の際に，評価関数を併用する手法がいくつか提案されている [1], [4], [5]．例えば，プレイアウト中の手の選択に評価関数を用いると，より現実に起きそうな手順に沿ってプレイアウトを実施することになり，より少ないプレイアウト回数でより良い結果を得ることができると期待される [*1]．直観的には，用いる評価関数が盤面を正しく評価するほどモンテカルロ木探索の精度も向上し，プレイヤが強くなると期待される．しかしながら，評価関数の違いがモンテカルロ木探索プレイヤの強さに与える影響については，著者の知る限りこれまで明らかにされていない．

本研究では，既に優れた評価関数が存在するオセロを対象にこの問題に取り組む．具体的には，異なる評価関数をモンテカルロ木探索に適用したプレイヤを作成し対戦させることで，より正しい評価関数を用いることの優位性を検証する．ここで検証の対象とする評価関数には，かなりの強さを誇るオープンソースのオセロプログラム Zebra [6] の評価関数を改変したものを用いる．

本稿の構成は以下のとおりである．第 2 章では本稿で用いるアルゴリズムや手法など，技術的な要素について解説を行う．第 3 章では評価関数の違いがモンテカルロ木探索の精度に与える影響を調べるための実験について，実験方法と実験結果を示す．そして第 4 章で実験結果をもとに考察を行う．第 5 章では，モンテカルロ木探索と評価関数について，関連研究を紹介する．第 6 章で本稿の内容をまとめる．

[*1] ただし，この場合には最適解への収束は証明されていない．

2. 要素技術

2.1 評価関数

局面の有利不利を評価し，数値化する関数を評価関数と呼ぶ．評価関数は，局面情報を受け取り，自分にとって有利な局面であれば正の数を，不利な局面であれば負の数を返す．その値を評価値と呼ぶ．後述する minimax 法やアルファベータ法では，この評価値をもとに探索を行う [7].

評価関数には性能と速度が求められる．正確な評価ができない評価関数の値をもとに探索を行えば，最善手を選び損ねて悪手を選んでしまう可能性がある．また，評価に長時間を要すると探索全体の時間が長くなってしまう．

2.1.1 Zebra の評価関数

本研究では，オープンソースのオセロプログラム Zebra [6] の評価関数を使用する [*2]．Zebra の評価関数は盤面からのパターン抽出による局面評価を行う．

図 1 は，Zebra の評価関数で抽出されているパターンであり，パターンは計 11 種ある．Zebra の評価関数は，これらのパターンに沿って各マスの状態を抽出し，その状態に対応するスコアを事前に計算・分析して求めた表から引く．評価値は各パターンのスコアの和として求められる [8].

図 1 の色つきの四角は，そのパターンでみるマスである．afile2x の場合，外周の 8 マスと 1 つ内側の 2 マスの計 10 マスの状態を見ている．1 マスがとる状態は配置駒が黒，配置駒が白，配置駒なしの 3 通りなので，10 マスを見るパターン afile2x では 3^{10} 通りの状態がある．この状態 1 つひとつにスコアが設定されている．

2.1.2 塩田による評価関数

実験において，評価関数の類似度による影響が考えられるため，異なる設計による評価関数も用

[*2] もともと C で書かれていたものを Java に移植した．

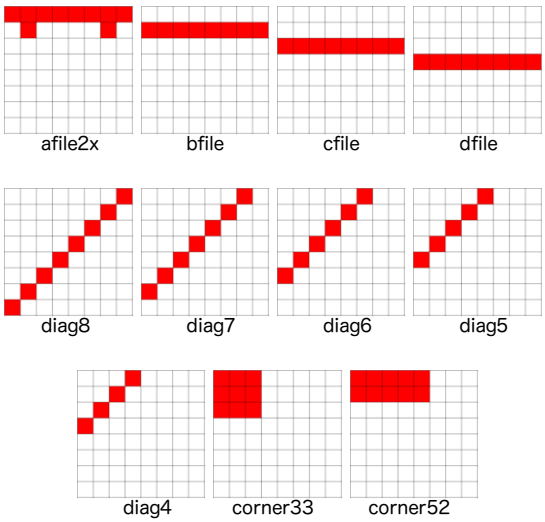

図 1　Zebra 評価関数のパターン

意する．そのようなものとして，塩田 [9] が提案したものを用いる．

塩田 [9] の評価関数は，以下の 3 つから構成される．塩田の評価関数では，これらの値に適当な大きさの乱数を加えた上で，重み和を計算し評価値としている．

盤位置　盤面の各位置に価値を持たせ，そこに自石がある場合には加算，相手石がある場合には減算することにより計算した値である．盤面の各位置の価値については，Sato により提案されたもの [10] を用いている．

確定石　どのようにプレイが進んだとしても，絶対に相手に取られることがない石を確定石と呼ぶ．塩田の評価関数では，外周の 4 辺における確定石の個数を計算対象としている．

候補数　ある局面において次に自分または相手が着手可能なマスの数を候補数と呼ぶ．

2.2　minimax 法とアルファベータ法

自分は最善の手を選び，相手は自分にとって最も不利な手を選ぶと仮定して探索を行う手法を，minimax 法と呼ぶ．

相手も最善手を選ぶと仮定した上で，最善の手を探索していく．手の評価には評価関数が用いられ [7]，評価値を見てその局面が最善手がどうか判定する．

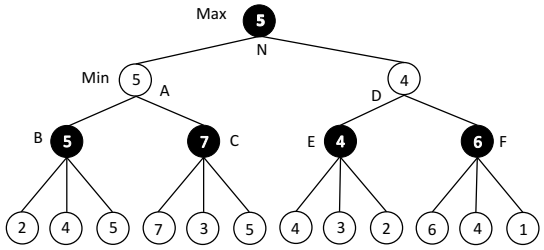

図 2　minimax 法の探索

図 2 で，minimax 法の探索手順を解説する．

まず，現在の局面に対応する根節点 N から探索を開始する．葉ではその局面の評価値を計算する．黒丸の節点では最大値を，白丸の節点では最小値を最善手とみなし選択する．

例えば，節点 B では最大値を選ぶので，5 となる．同様に C が選ぶ手も最大値の 7 となる．次に，A で選ばれる手は最小値なので 5 の B と 7 の C では B が選ばれ，A は 5 となる．D とその下の節点も同様に探索し，E は 4，F は 6 を選び，D は 4 を選ぶ．3 手先まで探索をしたので，N は 5 の A と 4 の D の中から最大値の手を選ぶ．よって，N は A を最善手として選ぶ．

minimax 法に対し，結果が変わらない節点の探索を省略すること (枝刈り) により高速化した手法のひとつが，アルファベータ法である [7]．

図 2 をアルファベータ法で探索した場合の探索手順を解説する．B で 5 を選んだ時点で，A は最小値を選ぶので 5 以下となる．したがって，最大値を選ぶ C は 7 を探索した段階で 7 以上となり，A には選ばれない．また，A が 5 で確定した時点で N は 5 以上となる．よって，C では 3 と 5 の探索を省略する．D，E，F についても，E が 4 に決まった時点で D は 4 以下，N は 5 以上なので D 以下をこれ以上探索する必要はなく，探索を終了して N は A を最善手として選ぶ．

2.3　モンテカルロ法

乱数によるシミュレーションを行い，その結果から手を決定する手法をモンテカルロ法と呼ぶ [1]．ゲームプログラミング分野において，ある局面から終局まで仮想的に乱数でプレイすることをプレ

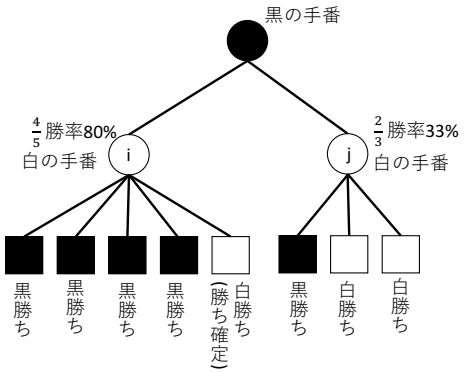

図 3 不利な手を選んでしまうモンテカルロ法の例

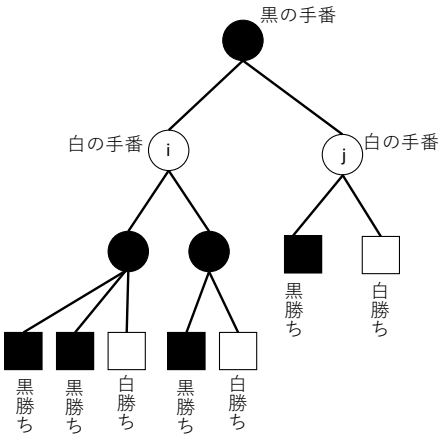

図 4 モンテカルロ木探索の例

イアウトと呼び，モンテカルロ法ではこのプレイアウトを複数回行ってその結果から手を選ぶ．

プレイアウト回数が多くなるほど，シミュレーションの精度も上がる．しかし，モンテカルロ法は minimax 法やアルファベータ法などと違い，相手が最善手を選ぶ可能性を考慮していない．つまり，相手も自分と同じくランダムにプレイするという前提で手を選んでいる．これは，時に相手のミスを期待した手を選ぶ可能性がある [1]．

図 3 は，モンテカルロ法で不利な手を選んでしまう例である．黒側がプレイアウトを 5 回行い，4 回勝利した局面 i があったとすると，プレイアウト結果からの勝率は 80%になる．しかし局面 i で負けた 1 回が相手の勝ちが確定する手であったとする．その場合，相手が常に最善手を選ぶと仮定すると，プレイアウト結果からの勝率が 80%でも局面 i に黒の負けが確定の手があるので負けの手ということになる．

2.4 モンテカルロ木探索

モンテカルロ法に木探索を組み合わせた手法をモンテカルロ木探索と呼ぶ [1]．モンテカルロ法と同様に，モンテカルロ木探索もプレイアウトを行いその結果から手を選ぶ．しかし，有望な手にプレイアウトを多く割り当てる点と，ある局面で行われたプレイアウト回数が閾値を超えると 1 手進んだ先からプレイアウトを始める点が異なる．

モンテカルロ木探索の基本形を，図 4 を用いて示す．

(1) 根節点から，有望な手を辿っていき，末端の節点まで到達する (末端の節点とは，プレイアウト回数が閾値に達していない手を指す)
(2) 末端の節点 i で行われたプレイアウト数が閾値を超えていればその節点を展開し，有望な節点を選んでひとつ降りる (図 4 は節点 i が展開された後を示している)
(3) 末端の節点でプレイアウトを行う
(4) 末端の節点でのプレイアウト結果を，辿ってきた経路上の節点に反映する
(5) プレイアウト回数や時間の制限に到達していればそこで終了し，そうでなければ (1) に戻る

有望な手の判断基準として，UCB1 値というものを基準にする方法がある．また，UCB1 値で有望な手を選ぶモンテカルロ木探索を UCT アルゴリズムと呼ぶ [2]．

UCB1 値は，式 1 のように定義される．ただし，$sum(X(s))$ は局面 s で行なったプレイアウトの勝数，n_s は局面 s のプレイアウト数，n はプレイアウト数の合計，C は適当な定数とする．

$$\frac{sum(X(s))}{n_s} + C\sqrt{\frac{\log n}{n_s}} \tag{1}$$

3. 実験

3.1 Zebra 評価関数のパターン除去

本研究で使用する Zebra の評価関数は，パターンによる局面評価を行なっている．そこで，抽出するパターンをいくつか除去すれば，同じ局面で

表 1 各評価関数で除去したパターン

評価関数	除去したパターン
W0	除去パターンなし
W1	afile2x, diag8, corner33, corner52
W2	afile2x, corner52
W3	afile2x, corner33
W4	bfile, cfile
W5	diag5, diag4
W6	afile2x, diag8
W7	afile2x, bfile, cfile, dfile
W8	diag7, diag6, diag5, diag4
W9	bfile, diag8, corner33, corner52

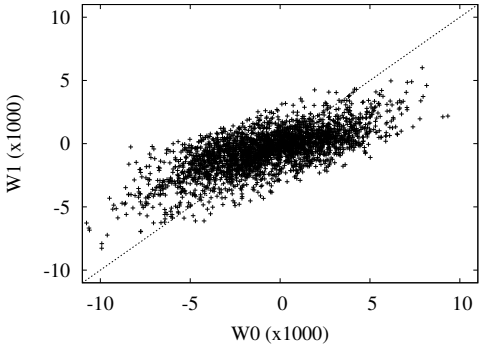

図 5 W0 と W1 の評価値の散布図

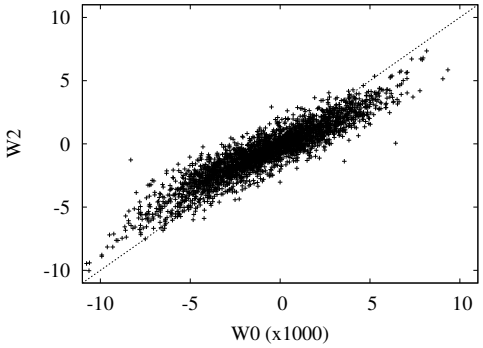

図 6 W0 と W2 の評価値の散布図

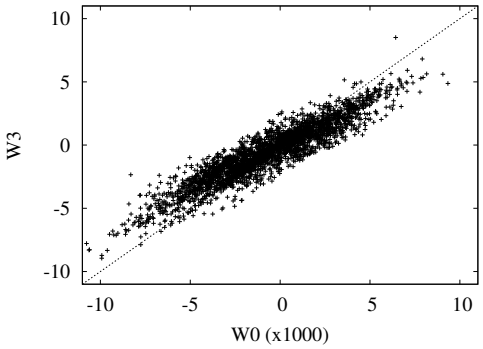

図 7 W0 と W3 の評価値の散布図

も評価にばらつきが生じ勝率に影響が出ると予想される．パターンを 2 つまたは 4 つを除去したものを盤面評価の正しさが異なる評価関数とし，これを 9 種用意する．パターンを除去していないものを加え計 10 種類の評価関数を作り，各評価関数はここでは W0 から W9 と呼ぶものとする．W0 をパターンを除去していない評価関数，W1 から W9 を盤面から抽出するパターンを 2 つまたは 4 つ除去した評価関数とする．

表 1 は，各評価関数で除去したパターンである．

パターンの除去は，局面評価上の影響を基準とした．図 1 の各パターンについて，角や確定石の多い外周とその周囲を見るパターンは局面評価に大きな影響があると思われ，逆に外周をほとんどカバーしていないパターンは除去しても影響が少ないと思われる．例えば，図 1 の afile2x は，外周 8 マスと 1 つ内側の 2 マスの計 10 マスを見ており局面評価上の影響が大きいと思われる．逆に，bfile や cfile，dfile は角や外周が中心のパターンではなく，局面評価上の影響は小さいと考えられる．このような局面評価上の影響を基準に，影響が大きいと思われるパターンのみ除去した評価関数，あまり影響のないと思われる範囲で除去した評価関数，影響が大きいものと小さいもの両方のパターンを除去した評価関数を作成した．

3.1.1 評価関数同士の評価値の比較

オリジナルの Zebra の評価関数である W0 と，パターンを除去した W1 から W9 の評価関数について，盤面 3000 件の評価値を散布図にした．これを図 5 から図 13 に示す．横軸が W0 による評価値，縦軸が W1 から W9 による評価値である．評価する盤面が同じでも，W0 とそれ以外とで評価値が大きく違う場合やそうでない場合が見られる．このことから，パターンを除去することで局面評価に多少の違いが生じ，評価関数の性能が変化していることが分かる．W4 はバラつきが最も少な

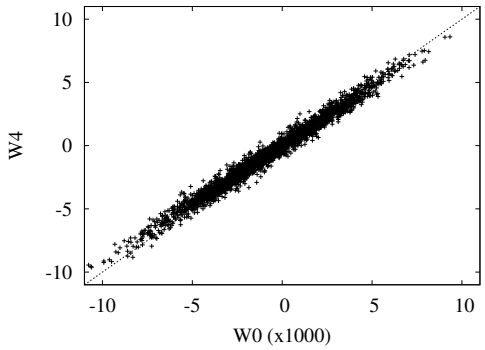

図 8　W0 と W4 の評価値の散布図

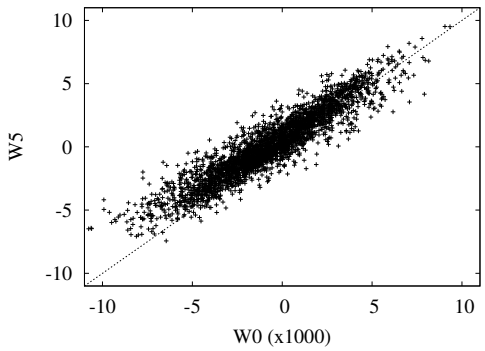

図 9　W0 と W5 の評価値の散布図

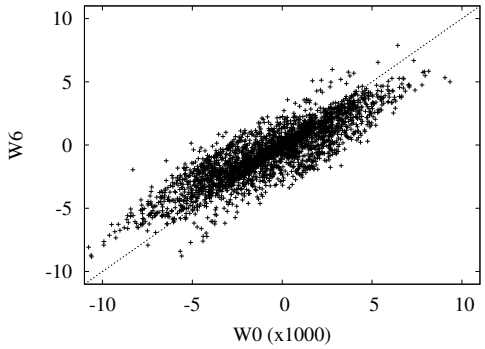

図 10　W0 と W6 の評価値の散布図

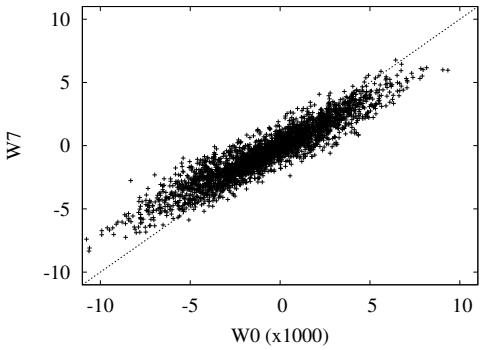

図 11　W0 と W7 の評価値の散布図

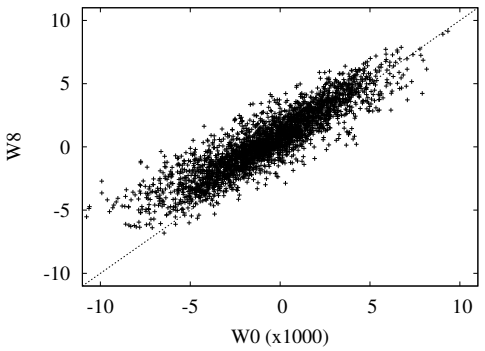

図 12　W0 と W8 の評価値の散布図

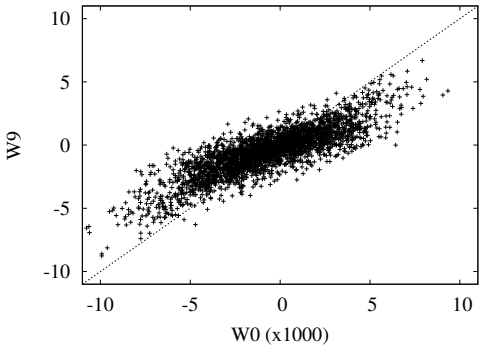

図 13　W0 と W9 の評価値の散布図

く W0 に非常に近い局面評価を行うと思われる．一方で，W1 や W6 は幅が広くなっており，W9 は傾きがずれている．よって，これらの評価関数は W0 と比較して，評価の正しさが大幅に変化していると思われる．

3.2　評価関数を併用したモンテカルロ木探索プレイヤ

評価関数は以下の場所で使用する．

- モンテカルロ木探索のプレイアウト中の着手
- モンテカルロ木探索の UCB1 値

モンテカルロ木探索プレイヤは，評価関数を併用する場所が異なる，2 つのプレイヤを用意する．

表 2 プレイヤ E とプレイヤ T で評価関数を適用する場所

評価関数を適用する場所	プレイヤ E	プレイヤ T
プレイアウト	○	× (ランダム)
UCB1 値	○	○

プレイアウトと UCB1 値の両方で評価関数を併用するモンテカルロ木探索プレイヤをプレイヤ E，UCB1 値のみに評価関数を併用し，プレイアウトはランダムに行うモンテカルロ木探索プレイヤをプレイヤ T とする．これにより，同じ評価関数でも併用場所によって影響に差が出るか確かめる．

プレイヤ E とプレイヤ T で評価関数を適用する場所を，表 2 にまとめる．○が評価関数を使用する，×が使用しない (従来のモンテカルロ木探索と同じ) とする．

また，プレイヤ E とプレイヤ T 共に，使用する評価関数には W0 から W9 を適用し，これを逐次切り替える．そのため，使用する評価関数が異なるモンテカルロ木探索プレイヤが，プレイヤ E とプレイヤ T とで共に 10 種類ずつ，計 20 種類あることになる．

3.2.1 プレイアウトへの評価関数の適用

プレイアウト中の着手に評価関数を併用する方法では，プレイアウト中の着手可能な手を評価関数で評価する．その評価値を参考にしつつランダム要素を残してプレイアウト中の手を決定する．評価値の高さ順に着手可能な手をソートし，評価値が高い手はひとつ下の手より 2 倍の確率でランダムに選ばれるようにする．

3.2.2 UCB1 値への評価関数の適用

UCB1 値に評価関数を併用する方法は，橋本ら [4] によって提案された手法を参考にしている．また，本研究での改良として UCB1 値を式 2 のように変更する．ただし，$E(s)$ は局面 s の評価値を勝率に変換した値，W は定数とする．W をプレイアウト数，$E(s)$ を盤面 s で W 回プレイアウトした結果得られた勝率と考えると，プレイアウト数が少なくても評価値から変換した勝率が UCB1 値による有望な手の判断で考慮される．今回は，W の値は 5 とした．

$$\frac{E(s) * W + sum(X(s))}{n_s + W} + C\sqrt{\frac{\log n + W}{n_s + W}} \quad (2)$$

評価値の勝率への変換は，まずランダムプレイヤ同士をオセロで対戦させて 5 手目以降の盤面 10000 件を用意した．その盤面の評価値を Zebra の評価関数 (W0) で求め，評価値の範囲を求めたところほとんどの評価値は −5000 から 5000 の範囲内だった．そこで，−5000 から 5000 の範囲で評価値を勝率 0.0 から 1.0 に線形にマッピングする．その計算式を式 3 に示す．ただし，$P(s)$ は局面 s の評価値とする．また，式 3 で得られた値が 0.0 未満だった場合は 0.0，1.0 より大きい場合は 1.0 とみなす．

$$E(s) = 0.5 + P(s)/5000/2 \quad (3)$$

3.3 実験方法

実験は，モンテカルロ木探索プレイヤとアルファベータ法プレイヤをオセロで対戦させて行う．対戦相手にアルファベータを選んだ理由としては，処理時間が速く強さも十分で，強さを測る相手として十分と判断したためである．

モンテカルロ木探索で併用する評価関数や，評価関数を使用する場所，アルファベータ法の先読み数などを変えて勝率を比較する．また，勝率からレートを計算し評価する．

3.3.1 モンテカルロ木探索プレイヤ

勝率に影響する要素の変更は評価関数の変更のみにとどめるため，プレイヤ E とプレイヤ T のプレイアウト数やノード展開の閾値，UCB1 値の定数 C は最適値ではなく適当な値に設定し，できるだけ変更しない．プレイアウト数は 1000 と 10000 で行い，閾値はプレイアウト 1000 の時に閾値 31，プレイアウト 10000 の時に閾値 100 に設定する．UCB1 値の定数 C は 0.1 に固定する．

3.3.2 アルファベータ法プレイヤ

アルファベータ法で用いる評価関数は，Zebra の評価関数と，塩田による評価関数との 2 つを切り替えて用いる．

アルファベータ法の先読みは，先読み 1 と先読み 2 で切り替えて行う．

3.3.3 レートの計算

レート差と対戦の勝率との間に，ロジスティク曲

表 3　アルファベータ法プレイヤのレート

評価関数	先読み 1 のレート	先読み 2 のレート
W0	42	611
W1	−377	−168
W2	−99	194
W3	−155	259
W4	−10	631
W5	−55	398
W6	−123	299
W7	−47	487
W8	−73	412
W9	−170	155

線で示される関係があると仮定する．具体的には，プレイヤ A のレートが R_A，プレイヤ B のレートが R_B のとき，プレイヤ A の期待勝率が

$$E_A = \frac{1}{1 + 10^{(R_B - R_A)/400}} \tag{4}$$

であると仮定する [*3]．この式を変形すると，勝率からレート差を計算する式

$$R_B - R_A = 400 \log_{10} \left(\frac{1}{E_A} - 1 \right) \tag{5}$$

を得る．ただし，この式は，勝率が 0 または 1 に近いとき，誤差が大きい．そこで，複数プレイヤを相手に対戦を行い，その勝率 E_A より計算したレートを重み [*4] $E_A(1 - E_A)$ で加重平均をとることでできるだけ誤差を小さくした．

3.4　実験結果

まず，塩田による評価関数を用いた 3 つのアルファベータ法プレイヤを基準として，Zebra の評価関数を改変して作成した評価関数を用いたアルファベータ法プレイヤのレートを計算した．その結果が表 3 である．

表 4 は，勝率比較の基準にするため，評価関数を全く使用しないモンテカルロ木探索プレイヤを黒駒，アルファベータ法プレイヤを白駒として対戦させた結果である．アルファベータ法プレイヤは AB で表し，その後に先読み数を示す．AB_1 の場合，先読み 1 のアルファベータ法プレイヤとなる．なお，表 4 のアルファベータ法プレイヤで使用した評価関数は W0 である．

[*3] これは Elo レーティングの数学的背景と一致する．

[*4] この重みは $E_A = 0$ または 1 のとき 0 となり，$E_A = 0.5$ のときに最大となる．

表 4　評価関数を使用しないモンテカルロ木探索プレイヤの勝敗数

プレイアウト数	相手	黒勝ち/白勝ち/引き分け
1000	AB_1	49/50/1
10000	AB_1	66/34/0
1000	AB_2	2/98/0
10000	AB_2	8/92/0

実験結果を図 14，図 15，図 16，図 17 に示す．

図 14 から図 17 は，横軸がモンテカルロ木探索プレイヤに適用した評価関数の種類，縦軸が 100 戦中のモンテカルロ木探索プレイヤの勝利回数である．対戦相手のアルファベータ法プレイヤで使用した評価関数は W0，先読みは 1 と 2 である．

各評価関数毎のグラフは，左 (青) がアルファベータ法の先読み 1 が相手の場合の勝利数，右 (オレンジ) がアルファベータ法の先読み 2 が相手の勝利数である．

プレイヤ E とプレイヤ T に共通する点として，アルファベータ法プレイヤの先読みが 1 のときはモンテカルロ木探索プレイヤがほぼ有利で強さの比較が可能である．しかし，先読みが 2 の場合はモンテカルロ木探索プレイヤがほぼ 9 割以上負けており，評価関数の違いによる強さの差を比較しづらい結果となっている．

図 14 及び図 15 はそれぞれプレイヤ E のプレイアウト 1000 と 10000 の場合の勝数である．使用した評価関数のほとんどが，評価関数を全く使用していないモンテカルロ木探索プレイヤより大幅に強い結果となった．また，評価関数毎の勝利数の差は最大で 4 割以上になり，評価関数の違いが勝率に大きく影響していることが分かる．

図 16 及び図 17 はそれぞれプレイヤ T のプレイアウト 1000 と 10000 の場合の勝数である．評価関数なしのモンテカルロ木探索プレイヤと比較しても，プレイヤ E ほどの勝率の大きな向上は認められない．評価関数 W1 はむしろ勝利数が基準より下がっているが，表 6 では，評価関数未使用のモンテカルロ木探索のレートを上回っている．よっ

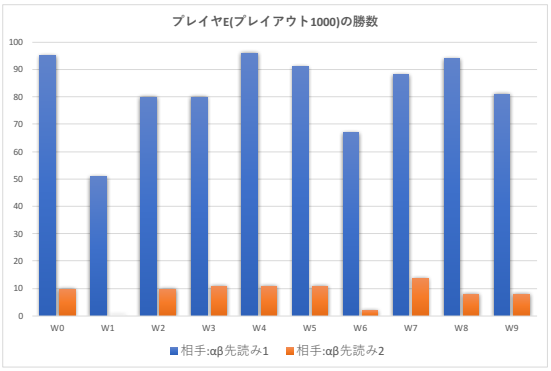

図 14 プレイヤ E(プレイアウト 1000) の勝数

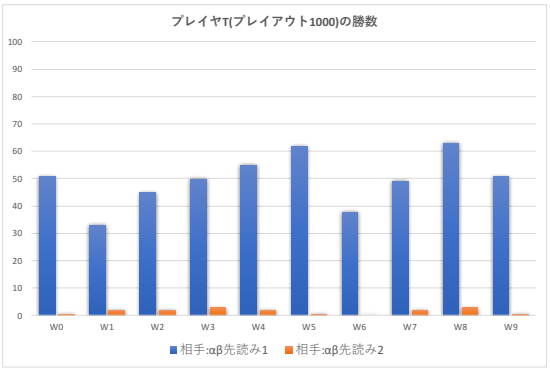

図 16 プレイヤ T(プレイアウト 1000) の勝数

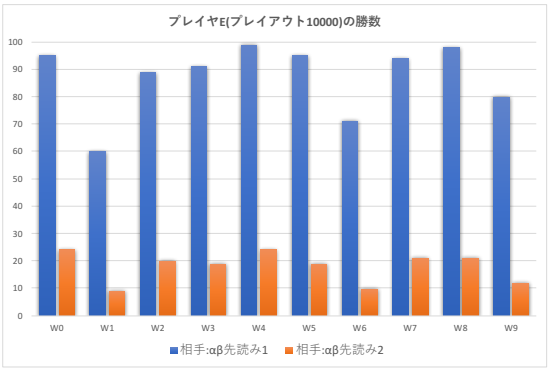

図 15 プレイヤ E(プレイアウト 10000) の勝数

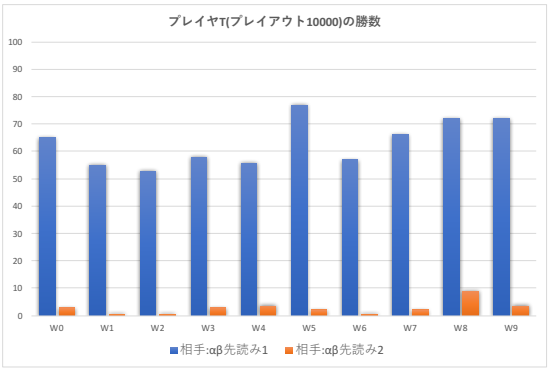

図 17 プレイヤ T(プレイアウト 10000) の勝数

て，評価関数を使うことで強さ自体は向上しているものの，プレイヤ E と比べて全体的に勝率の伸びが悪いと分かる．また，評価関数の違いによる影響も勝率にはっきりと現れていない．

表 5 及び表 6 は，評価関数を使わないモンテカルロ木探索プレイヤ，そしてプレイヤ E とプレイヤ T のレートを塩田による評価関数を用いた 3 つのアルファベータ法プレイヤを基準として計算したものである．レートの比較では，プレイヤ E とプレイヤ T 共に，評価関数を使わないモンテカルロ木探索より強いことが分かる．また，評価関数 W0 を用いた場合，プレイヤ E とプレイヤ T の両方で W1 から W9 を用いた場合より強い結果となっている．このことから評価関数 W0 は，少なくとも本研究で用いた 10 種類の評価関数の中では最も正しい評価関数と分かる [*5]．また，その性能差は勝

[*5] ただし，Zebra の評価関数そのものの正しさの証明にはならない

表 5 評価関数なしモンテカルロ木探索のレート

プレイアウト数	レート
1000	3
10000	110

率に最大で 4 割程度の差が生じる程である．

4. 考察

実験結果より，評価関数をモンテカルロ木探索に用いることの優位性を確認した．

評価関数 W0 は，プレイヤ E においては評価関数毎の強さでも最も強い評価関数の 1 つに入っており，レートも最高である．一方で，W1 はプレイヤ・プレイアウト数・レート問わずほぼ最弱であり，W0 と W1 の評価値の散布図は図 5 を見ると分かる通りかなりバラけている．このことから，W1 は W0 と比較して盤面評価の正しさが大きく劣ると思われる．よって，より正しい盤面評価を行う評価関数を用いた方が勝率の大きな向上が期

表 6　プレイヤ E とプレイヤ T のレート

プレイヤ	評価関数	プレイアウト数 1000 のレート	プレイアウト数 10000 のレート
E	W0	481	669
E	W1	196	320
E	W2	464	537
E	W3	401	546
E	W4	455	646
E	W5	395	583
E	W6	406	514
E	W7	452	617
E	W8	347	513
E	W9	262	326
T	W0	132	230
T	W1	51	135
T	W2	91	183
T	W3	132	180
T	W4	94	199
T	W5	114	164
T	W6	100	195
T	W7	104	188
T	W8	72	183
T	W9	79	146

待できると考えられる．

一方で，プレイヤ T は評価関数を用いてもプレイヤ E と比べて強さの伸びが悪く，評価関数毎の強さの差も小さい．これは勝敗数とレートの両方で確認でき，本研究で行なったような UCB1 値への評価関数の併用の仕方では，強さの大幅な向上そのものが期待できないと思われる．

図 5 から図 13 の評価値の散布図で，最もバラつきが小さいのは W4 である．しかし，W4 は全ての条件で勝敗数やレートにおいて上位にあるわけではない．プレイヤ E の勝敗数では，図 14 及び図 15 から分かる通り W0 と並んで上位であり，レートでも，表 6 より，プレイアウト 1000 のプレイヤ T を除き上位である．しかし，図 17 では W5 に勝数で劣っている．これは W0 も同様である．だが，表 6 では，W0 との評価値のバラつきがほとんどないにも関わらず，プレイヤ T の W0 に 38 の差で劣っている．そして，W4 より評価値のバラつきが大きい評価関数の内いくつかが，レート上では強い形になっている．

以上の点から，評価関数の違いによる強さの差には一貫性があるとは言い難い．併用の仕方によって，勝率が向上しやすい評価関数，向上しにくい評価関数などがあると考えられ，場合によっては強さの逆転が起こる可能性があると思われる．

よって，評価関数の違いは，評価関数の使い方によって逆転したり，大きな差が出ない場合があると考えられる．

5. 関連研究

モンテカルロ木探索の精度を向上させる方法の一つとして，プレイアウトに評価関数を併用する手法がある．プレイアウト中の手の選択はランダムに行うのが基本だが，これに評価関数による評価を組み込むことで，ある程度の強さのプレイヤが選びそうな手順のプレイアウトになり，短時間で高精度の結果が得られると期待される．

モンテカルロ木探索と評価関数を組み合わせた強いコンピュータプレイヤとして，囲碁コンピュータプレイヤの AlphaGo が挙げられる．AlphaGo には評価関数が 2 つ用いられており，ニューラルネットワークで学習させている．これらの評価関数による手の評価と，プレイアウト結果をもとに手を選ぶ．その強さは 2016 年に，囲碁の世界王者相手に 4 勝 1 敗で勝利する程である [3]．

UCT アルゴリズムに評価関数を使用する手法について，UCB1 値に評価関数を組み込む手法が提案されている [4]．この手法は，UCB1 値に局面評価関数の評価値を組み込むというもので，UCB1 値によって選ばれる手は局面評価上よいとされるものになる [4]．これにより，評価関数の性能と勝率との相関が高い場合は短時間で精度の高い結果が得られるとされている．

また，文献 [4] の手法を改良し，さらに新たな評価関数の使用法について提案がされている [5]．UCB1 値に組み込む評価関数の他に，プレイアウト中の先手後手それぞれの評価値を加算し，そこから求めた適当な評価値を UCB1 値に加えている．また，プレイアウト中の着手可能な手を評価関数で評価し，評価値が高いものをプレイアウトで選んでいる．さらに，プレイアウトを途中で打ち切

り評価関数の値で勝ち負けを判定する手法も行われている．

文献 [4] および文献 [5] ともにオセロによる検証が行われており，従来の UCT アルゴリズムよりも強いという結果が示されている．

このように，モンテカルロ木探索に評価関数を用いる手法は，強いコンピュータプレイヤを作る上で有効な手法であることが既存研究より分かる．しかし，評価関数の性能がモンテカルロ木探索プレイヤの強さに与える影響については，著者の知る限り明らかになっていない．

6. まとめ

評価関数の違いが，モンテカルロ木探索プレイヤの強さに対し，規模は様々だが影響を与えることを確認した．そして，より良い評価関数を使用することの優位性を示した．また，評価関数をどのように用いるかによって，評価関数の違いが強さに与える影響が変化する可能性があることを明らかにした．

今後の課題としては，Zebra の評価関数自体の盤面評価がどのくらい正しいのか，確かめる必要がある．また，なぜ評価関数の使い方を変えると，評価関数の違いによる強さへの影響が変化するのか，その原因を突き止めたい．

謝辞 本稿に掲載した研究成果は，その多くを高知工科大学の IACP クラスタ計算機を使用して得られたものである．

参考文献

[1] 松原仁: コンピュータ囲碁 モンテカルロ法の理論と実践, 共立出版, 2012.
[2] L.Kocsis, Cosaba Szepesvári: Bandit Based Monte-Carlo Planning, ECML, pp. 282–293, 2006.
[3] Kazuto Seki: 囲碁の最強人工知能 AlphaGo (アルファ碁) の仕組みとは?, https://tech-camp.in/note/technology/32855/.
[4] 橋本剛，前原彰太，川島哲哉，小林康幸: 局面評価関数を使う新たな UCT 探索法の提案とオセロによる評価, 情報処理学会論文誌, Vol. 54, pp. 1930–1936, 2013.
[5] 松本渉，小林康幸: UCT 探索における局面評価関数の使用方法と性能評価, ゲームプログラミングワークショップ 2013 論文集, pp. 170–174, 2013.
[6] Gunnar Andersson: Zebra, http://radagast.se/othello/index.html.
[7] 瀧澤武信 他: 人間に勝つコンピュータ将棋の作り方, 技術評論社, 2012.
[8] Gunnar Andersson, 訳 奥原俊彦: 強いオセロプログラムの内部動作, http://www.amy.hi-ho.ne.jp/okuhara/howtoj.htm.
[9] 塩田 好: リバーシの評価関数について, 近畿大学理工学部情報学科 卒業研究報告書, 2016.
[10] Kozo Sato: 評価関数を考える, http://www.geocities.co.jp/SiliconValley-Bay/4543/Osero/Value/Value.html.