棋譜データにおける勝率を利用したモンテカルロ木探索の性能評価手法

竹内聖悟[†] 金子知適[†] 山口和紀[†]

近年，ゲームプログラミングの分野においてモンテカルロ木探索が改良され，特に囲碁において強いコンピュータプログラムが生み出されている．本稿ではモンテカルロ木探索の正確さを測る手法を提案する．まず，棋譜中の局面の勝率をベンチマークとして利用し，プレイアウトによって得られる勝率との比較を行った．この手法を囲碁においてモンテカルロ木探索へと利用し，コウなどの局面の特徴，探索手法やパラメータの違いによって性能が変わることを確認した．続いて，探索手法の性能を評価するために数値指標として評価値の期待値を導入した．囲碁における実験では，探索手法やパラメータによる性能の違いに関する経験的な理解と実験結果が一致することを確認した．

Evaluation of Monte Carlo Tree Search, Based on Winning Probability of Game Records

SHOGO TAKEUCHI,[†] TOMOYUKI KANEKO[†] and KAZUNORI YAMAGUCHI [†]

Recent improvements on Monte Carlo tree search have produced strong computer Go programs. This paper presents a method of measuring the accuracy of Monte Carlo tree searches in game programming. We use win percentage of positions in a large database of game records as a benchmark and compare the win probability obtained by simulations with the benchmark. By applying to Monte Carlo tree search in Go, we found out the difference between the search methods and their parameters, and the effect of property of positions such as Ko. In this paper, we also introduce numerical metrics to evaluate the performance of search methods. Our experiments in Go showed that the metrics are quite close to our empirical understanding of the performance of various search methods and their parameters.

1. はじめに

コンピュータ囲碁の分野において，モンテカルロ木探索が多くのプログラムに用いられ，大きな成果を挙げている．例えば，2008 年 3 月には 9 路盤においてプロ棋士を相手に 1 勝 2 敗の成績をおさめている[19]．

モンテカルロ (MC) 法を用いた局面の評価は 1993 年に Bruegmann によって発案された手法[4] で，ランダムな着手で終局までゲームを行い，その結果から局面の評価を行う手法である．この MC 法を評価手法としてゲーム木探索と組み合わせた手法がモンテカルロ木探索である．例として UCT[10] がある．

MC 法の特徴としてゲームに関する知識はルール以外不要である点が挙げられるが，近年では性能改善のためにゲームの知識を入れる手法[8][9] が主流となっている．例えば，初めて訪れる局面に対し，あらかじめ作成しておいた評価関数の評価値を一定回数分のシミュレーションの結果として使う手法が用いられている．また，パターンの利用などシミュレーションの質を上げる手法や RAVE のようにシミュレーションを高速化する手法などが用いられている．これらの改善手法には何かしらのパラメータがあり，またモンテカルロ木探索自身もパラメータを持っており，パラメータの値をどうやって決めるかが問題となる．従来は，パラメータを変えたプログラムでの対戦結果から適切なパラメータの決定が行われてきたが，評価が間接的であることや有意な結果を得るには非常に時間がかかること，対戦相手によるバイアスが生じるというなどの問題点がある．

本稿では，棋譜データにおける勝率を利用したモンテカルロ木探索の評価手法について提案し，その有効性を示す．これまでに，棋譜と評価関数を用いた Evaluation Curve という手法により評価関数の問題点の図示化を行ってきた[16]．Evaluation Curve による評価を行うには，棋譜を評価させる必要があるがそのコストは対戦よりも低く，適切に棋譜を選ぶことでバイアスがかかることは回避可能である．Evaluation Curve は棋譜中の様々な局面を評価関数で評価した結果を使って，評価関数の評価をする手法であるが，本稿ではこの手法を改良し，モンテカルロ木探索の評価に応用す

† 東京大学大学院総合文化研究科
Department of General Systems Studies, Graduate School of Arts and Sciences, The University of Tokyo
{takeuchi,kaneko,yamaguch}@graco.c.u-tokyo.ac.jp

る．また，Evaluation Curve では，得られた結果を数値的に比較しにくいという問題点があったため，これを補完するための数値的指標を提案する．

本稿の構成は以下の通りである．まず，2 章で関連研究について述べる．3 章で提案手法について詳しく説明し，4 章で提案手法の評価実験の結果を示す．5 章では結論を述べる．

2. 関 連 研 究

2.1 Monte Carlo 囲碁

ブリッジ[11] やスクラブル[14]，ポーカー[2] などの不完全情報ゲームにおいて，サンプリングベースの手法が使われてきた．Abramson[1] はランダムサンプリングによる評価を利用する手法を提案している．最初にモンテカルロ法を完全情報ゲームである囲碁へと応用したのは Brügmann[4] で，その後 Bouzy[3] によって研究が進められた．

局面の評価手法として，従来は人間が評価関数を作っていたが，モンテカルロ木探索ではランダムサンプリングの結果を局面の評価として利用する．

モンテカルロ木探索の基本的なモデルでは，深さ 1 の探索を行い，各ノードのスコアの期待値を計算し，値が最大の手を選ぶ．局面のスコアの期待値は，その局面から始めた全ランダムゲームの終局局面におけるスコアの平均値によって定義される．以下，これらのランダムゲームのそれぞれをプレイアウト と呼ぶ．ランダムゲームにおいて，各プレイヤーは有効な手がなくなるまで，自分の” 眼” を埋める手を除いた合法手からランダムに手を選ぶ☆．しかし，この基本的なモデルにおいては，プレイアウト数を増やしても強さが頭打ちになることが確認されている[18]．

その後，多くの改良が行われてきた．例えば，終局時のスコアとして目差を利用するよりも勝敗を利用した方が適切であることが多くのプログラムで確認されている．最も大きな改良は，それまでの 1 手探索モデルを，再帰的にノードを展開し，効果的な手に多くのプレイアウトを割り振る手法へと改良したことである[7),13]．

2.1.1 UCT

UCT[13] はモンテカルロ木探索の代表的な手法である．多腕バンディット問題の理論に基づいた手法で，最も有効なノードを最良優先で展開していく．UCT の葉ノードにおいてモンテカルロシミュレーションが行われ，結果はそのノードとその祖先へと伝播され，勝率が決まる．ノードの有効性はノードの勝率とその分散から計算される．UCT において各ノードの有効性を計算する式にはいくつかバリエーションがある[6),10]．

ノード a において i 番目の手に対して s_i 回のプレイアウトが行われたとして，p_i をその勝率とする．ここで，ノード a とその子孫ノードの中で n 回のプレイアウトが行われたものとする．

UCB1 は次の式を最大化するノード a を選ぶ．

$$p_i + \sqrt{\frac{2\log n}{s_i}}$$

UCB1-Tuned は UCB1 の改良版で，次の式を最大化するノード a を選ぶ．

$$p_i + \sqrt{\frac{\log n}{s_i}\min\left(1/4, p_i - p_i^2 + \sqrt{\frac{2\log n}{s_i}}\right)}$$

さらに，最先端のプログラムでは，より信頼できる結果を得るためにパターンや Progressive Widening, RAVE など多くのヒューリスティックを利用している．パターンは，棋譜の静的な解析[8] や対局中に動的な解析を行うこと[15] で得られる．

2.2 ゲーム木探索の正確性

一般にゲーム木探索の正確性は 2 つのプログラムを対戦させて比較するなど間接的にしか評価できない．このため，統計的に有意な結果を得るためには膨大な時間がかかるという問題がある．

より多くの情報が入手可能であれば，直接評価することが可能な場合がある．探索やデータベースによって理論的に正しい評価が入手可能ならば，評価の誤差が直接計算できる．解析が行われた例としてオセロ[5] や Awari[17] の終盤データベースがあるが，解析が可能な分野は限られている．人間のプレイヤによる評価が入手可能であれば，人間のプレイヤによる評価との誤差を見ることができる[12] が，この手法も応用可能な分野が限られる．

棋譜の勝敗による勝率の近似と，Evaluation Curve による評価値と勝率の関係の可視化を我々の論文[16] で提案した．以前の論文では，様々な局面での評価関数の一貫性の評価が主であったが，本稿ではモンテカルロ木探索の評価を目的とする．ここで提案する手法において，局面に対して必要となるものは先手の勝敗だけである．

3. 勝率と Evaluation Curve

3.1 棋譜における勝率

MCTS のプログラムは最善手を選ぶため，ランダムゲームにより各局面の勝率を近似する．近似された勝

☆ 囲碁において自分の眼を埋めるのはあまりにも悪い手であるので除いている

率はそのプログラムが勝つ” 真の” 確率に近いほど良い．本稿では，モンテカルロシミュレーションによって得られた勝率を” 真の” 勝率と比較することにより正確性をテストする手法を提案する．しかし，“真の” 勝率を得るのは困難であるので，代わりに棋譜から得られる勝率を使う．2 つの勝率を区別するため，モンテカルロシミュレーションによって得られる勝率を評価値，棋譜から得られる勝率を勝率と呼ぶ．

多数の棋譜を集めた R があるとして，R を利用する勝率を評価値 v と R の関数として次のように定義する．

$$\textit{Win probability(v, R)} = \frac{|B_v(R)|}{|B_v(R)| + |W_v(R)|}, \quad (1)$$

where

$$P_v(R) = \{p \in R | v - \frac{\delta}{2} \leq \textit{eval}(p) < v + \frac{\delta}{2}\}, \quad (2)$$
$$B_v(R) = \{p \in P_v(R) | \textit{winner}(p) \textit{ is Black}\},$$
$$W_v(R) = \{p \in P_v(R) | \textit{winner}(p) \textit{ is White}\}.$$

p は R 中の局面で，δ は正の定数で区間の幅を示す．勝率を計算するため，まず棋譜中の各局面に対して評価値を求め，各局面の勝者としてその棋譜の勝者を利用する．棋譜の勝者を各局面の勝者とする手法は乱暴なようだが，我々の前の実験[16] では有効であった．最後に，区間 $[v - \frac{\delta}{2}, v + \frac{\delta}{2})$ に対して勝数 $|B_v|$, 敗数 $|W_v|$ を数え，式 (1) により勝率を計算する．評価関数の評価についての我々の以前の論文[16] では，評価関数によって得られた値を式 (2) の $\textit{eval}(p)$ として利用したが，本稿のモンテカルロ木探索の評価手法では，モンテカルロシミュレーションの結果を $\textit{eval}(p)$ として利用する．

3.2 Evaluation Curve

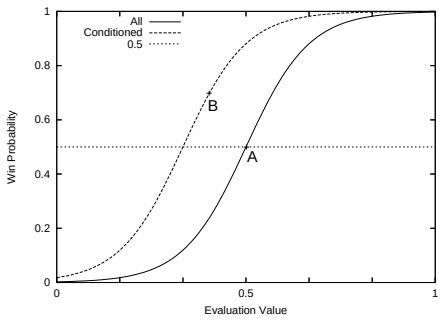

図 1 Example of a poor evaluation function

勝率と評価値との関係は X 軸に評価値を，Y 軸を勝率としてプロットすることで図 1 のように図示できる．

これを *Evaluation Curve* と呼ぶ．良いシミュレーションの Evaluation Curve は単調増加になるが，これだけでは評価には不十分である．2 節で述べた基本的なモデルのモンテカルロプログラムがあるとする．全局面を対象とした Evaluation Curve を実線で，相手の群が危険である，などの条件を満たした局面だけを対象としたものを点線で描いたとする．これが仮に図 1 のように分離したとすると，勝率の低い局面が選ばれうる．例えば，探索中で図 1 の点 A，B のどちらかを選択する状況では，A の評価値が B よりも高いため A を選択してしまう．しかし，本来は勝率が高い B を選択するべきである．従って，この評価関数は条件に関して局面を適切に評価できておらず，問題のある評価関数だと判断できる．このようにして，Evaluation Curve を様々な条件の局面に対して描くことでシミュレーションの問題点を見つけることが可能である．

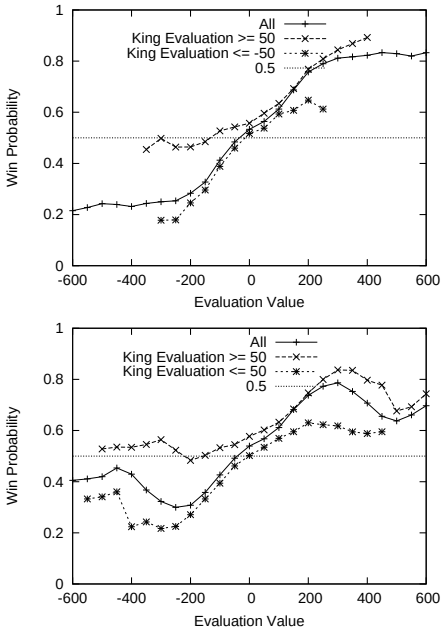

図 2 Evaluation curves in Chess (King Safety) (upper: with quiesce, lower: w/o quiesce)

この評価手法がどのように働くかの例として，チェスでの Evaluation Curve を取り上げる．図 2 はあるチェスプログラムにおける Evaluation Curve であり，上図と下図はそれぞれ静止探索の有無に対応する．また，条件は King が危険な時で，この時グラフはわかれている．図を見ての通り，実験のしたプログラムの Evaluation Curve は必ずしも単調に増加になっていない．

この手法は，モンテカルロ木探索の性能を直接的に評価でき，対戦に比べて時間がかからないという利点

がある．

3.3 期待値による数値的評価

3.2 節で述べた Evaluation Curve の” 良さ” を簡単に比較できるように数値的指標を導入する．

ある局面 i での評価値を v_i とし，$[0,1]$ の範囲にあるものとする．また，局面の勝敗を r_i とし，先手が勝ちなら 1, 負けなら 0 の値をとるものとする．

理想の評価関数を考えると，先手勝ちなら $v_i = 1$, 負けなら $v_i = 0$ となるのが望ましい．多数の棋譜を評価させていき，その評価値の分布を考えると，先手勝ちなら $v_i = 1$ に集まり，先手負けなら $v_i = 0$ に集まる分布が望ましい．これは，評価値の分布の期待値で，先手勝ちなら $E(v)|_{r=1}$ が大きく, 後手勝ちなら $E(v)|_{r=0}$ が小さいことが望ましいと表すことができる．それぞれの期待値は

$$E(v)|_{r=1} = \frac{\sum_{\{i|r_i=1\}} v_i}{\sum r_i}$$

$$E(v)|_{r=0} = \frac{\sum_{\{i|r_i=0\}} v_i}{\sum (1-r_i)}$$

となる．ここで，r をわけずに期待値を統合することを考える．$r = 0$ のケースは小さいほど良く，v は $[0,1]$ の範囲にあるため対称性から，v の代わりに $1-v$ を利用することにする．すると，

$$E(v) = \frac{\sum v_i^{r_i} * (1-v_i)^{(1-r_i)}}{\sum 1} \quad (3)$$

となり，これは評価値 v の分布のもとで勝敗 r を観測する尤度となる．式 (3) の $E(v)$ を評価関数の評価指標として提案する．$E(v) = 0.5$ となるのは，全局面を 0.5 と評価した場合でこれがベースラインとなる．全局面を正しく評価すれば $E(v) = 1$ となる．Evaluation Curve では評価値の分布による影響を無視しているため，本手法により Evaluation Curve を補完することが可能であると考えられる．

なお，今は v の範囲を $[0,1]$ としたが，$r = 0$ の場合は v の代わりに $-v$ を使った次式により，$(-\infty, \infty)$ に適用可能とすることができる．

$$E(v) = \frac{\sum v_i^{r_i} * (-v_i)^{(1-r_i)}}{\sum 1}$$

この指標では Evaluation Curve で使用している評価値や勝敗の利用ができるため，数値化するコストがかからないという利点がある．

4. 実 験 結 果

4.1 プログラムと棋譜

まず，利用したプログラムと棋譜について説明する．

- Fuego: 様々な拡張がされた UCT を利用するプログラムとして Fuego[☆1] (version 0.1.1) を利用した．9×9 路盤の CGOS (Internet Go server) [☆2] でのレーティングは 2,300 であった．
- UCT: UCT を使用するプログラムとして libego [☆3] (version 0.116) を利用した．CGOS (9×9) でのレーティングは 1,800 であった．
- MC: モンテカルロ法のプログラムとして，libego のコマンドを利用した．ルートにおける勝率が求まるが，他のプログラムではプレイアウト数を全合法手に振り分けている．公正な比較を行うため，各局面の合法手でプレイアウト数を割ることで調整を行った．
- MC-Score: 評価値を得るさいに，勝敗の代わりに各シミュレーションの最終局面での目数差を利用するプログラム．このプログラムを利用する目的は，2 節で説明した基本的なモデルでのシミュレーションの質を測ることである．
- GnuGo: 評価関数を利用した従来型のプログラムとして GnuGo [☆4] (version 3.7.12) を利用した．

評価の際，各プログラムでは中国ルールを利用した，これは日本ルールに非対応なプログラムがあったためである．

実験で用いた棋譜は 2001 年から 2005 年に KGS でプレイされた 9 路の棋譜である．コミが 0.5 目でプレイヤのレーティングが 3 級以上である棋譜を 2,000 局選び，そこから 111,946 局面を得た．

4.2 Evaluation Curve と評価値の期待値

まず，Evaluation Curve の結果を示す．X 軸は評価値を，Y 軸は勝率を表し，100 局面以上の点のみをプロットした．図 3 は Fuego, UCT, MC の Evaluation Curve を，図 4 は MC-Score と GnuGo の Evaluation Curve を表す．なお，値は $[-81, +81]$ の範囲にある．Fuego ではプレイアウトの数を増やしても結果はほとんど変わらなった．

UCT, MC, MC-Score ではプレイアウト数が 500 のものとそれ以外とでは大きく異なるという結果になった．つまり，プレイアウトの数によってモンテカルロ法が返す勝率と実際の勝率が大きく異なるということである．UCT では，勝率とプレイアウト数によって決まる指標の和を用いているが，この Evaluation Curve を利用して勝率を補正した方が良いと考えられる．MC, MC-Score では，プレイアウト数が 500 の時に単調増

[☆1] http://fuego.sourceforge.net/
[☆2] http://cgos.boardspace.net/9x9/
[☆3] http://www.mimuw.edu.pl/%7elew/hg/libego/
[☆4] http://www.gnu.org/software/gnugo/gnugo.html

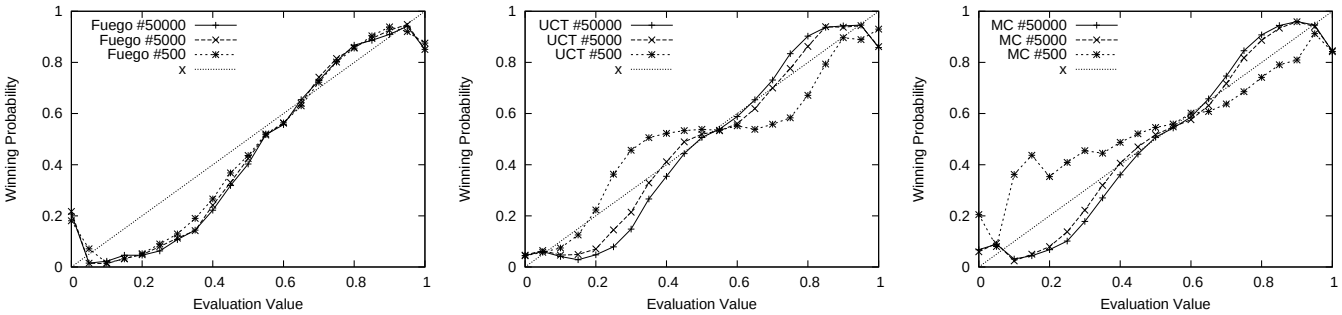

図 3 Evaluation Curves for Fuego (left), UCT (center), MC (right)

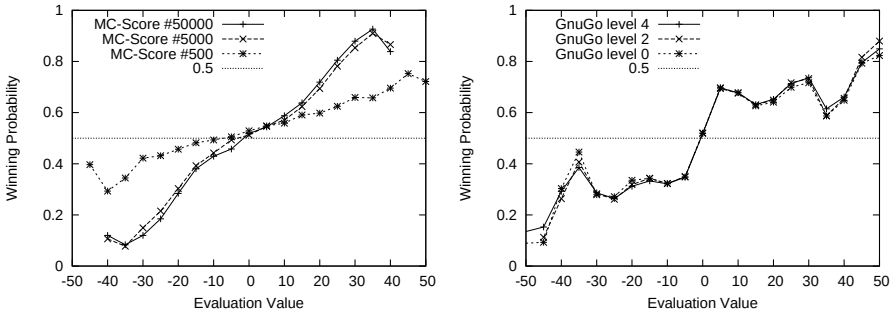

図 4 Evaluation Curves for MC-Score (left), GnuGo (right)

加でなくなっていて，プレイアウト数を 5,000 から 50,000 に増やしても大きな変化がない．後者は，吉本らによって報告された収穫逓減の効果ではないかと考えられる[18]．

GnuGo はレベルを変えても変化が見られず，Evaluation Curve が単調増加ではない．これは，人間の棋譜から計算される勝率と GnuGo の評価が異なることを示している．

評価値の期待値は表 1 のようになった．なお，MC-Score, GnuGo は評価値が $[-81, 81]$ の範囲にあるため Fuego ではプレイアウトの数が顕著に影響を及ぼしていることがわかる．また，GnuGo においてレベルが高くなるにつれ，指標の値も高くなった．GnuGo の Evaluation Curve を見ると，レベルによる違いは評価値の絶対値が大きなところに見られる．数は少ないが評価値が大きいため，大きな影響をもたらしたと考えられる．

これらの結果は Evaluation Curve では発見できておらず，提案指標による Evaluation Curve の補完が可能であると言える．

性能に関しては $Fuego > UCT > MC > MC-Score$ で，プレイアウト数は多いほど強いと言われており，Evaluation Curve と指標の結果はこれと一致し，有効性が期待できる．

表 1 Performance of Evaluation Methods: Go

methods	#playouts	$E(v)$
Fuego	50,000	0.635
	5,000	0.629
	500	0.623
UCT	50,000	0.560
	5,000	0.548
	500	0.553
MC	50,000	0.548
	5,000	0.547
	500	0.540
MC-Score	50,000	2.493
	5,000	2.451
	500	2.492
GnuGo	level 4	3.797
	2	3.660
	0	3.556

4.3 複雑な局面による Evaluation Curve の違い

モンテカルロ法は複雑な局面では，他の局面と比べると間違いを犯しやすい傾向があると言われる．tactical な局面では良い手を得るために探索が必要であり，深さ 1 の探索しか行わないモンテカルロ法では正しく評価できないことが原因だと考えられる．実験では，

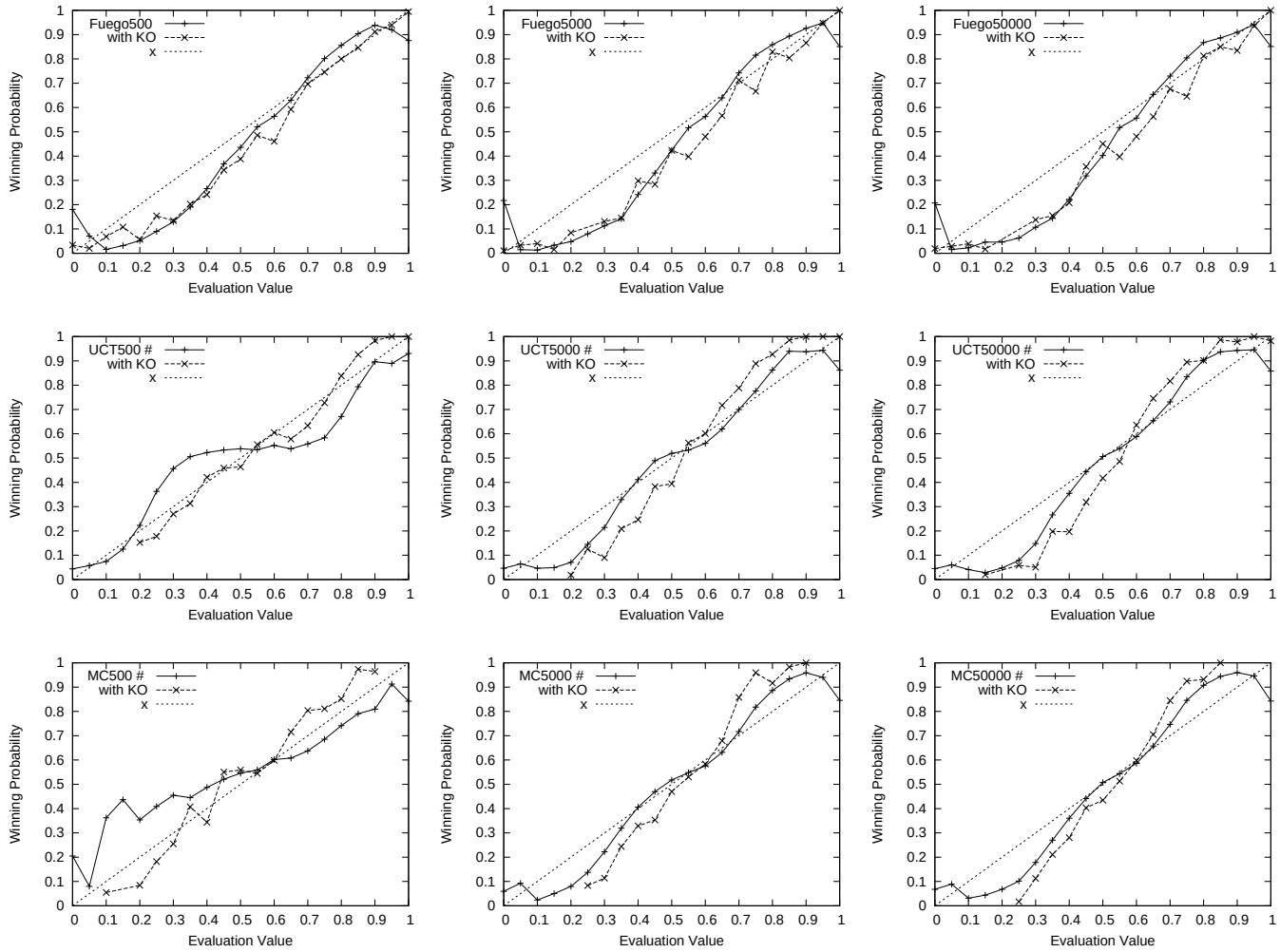

図 5 Fuego, UCT and MC with Ko position. (up: Fuego, center: UCT, low: MC, left: 500 playouts, center 5,000 playouts, right: 50,000 playouts)

コウが存在する 2,218 の局面を選んで実験を行った．

図 5 は Fuego, UCT, MC の，図 6 は MC-Score, GnuGo のコウが存在する時の Evaluation Curve である．Fuego は評価値が 0.5 以上の範囲ではカーブがわかれているが，他のプログラムは評価値の範囲に関わらず，コウに関しては正しく評価できてないことがわかる．なかでも MC-Score がもっとも正しく評価できていない．傾向としてはサンプル数が増えると差が少なくなっているようである．

コウが現れる局面はある程度手数が進んだ局面であるため，棋譜の全局面での指標と比較するのは公正ではない．公正な比較を行うならば，コウが現れる前後の局面でコウがない局面を対象とした評価と比較する必要があるだろう．

4.4 将棋での実験

将棋でもモンテカルロ木探索を実装し，同様の実験を行った．実験では，単純なモンテカルロ法と UCT の 2 つのプログラムを利用した．プログラムには，GPS 将棋☆ を利用し，棋譜は将棋倶楽部 24[20] の棋譜 2,000 局を利用した．モンテカルロ法の手の選び方はランダムとし，UCT は選択手法として UCB1-tuned を利用し，モンテカルロシミュレーションはランダムに手を選ぶようにした．また，探索中には 1 手詰め判定関数を利用し，詰みがあればそこで終局とした．

Evaluation Curve は図 7 のようになった．プレイアウ

☆ http://gps.tanaka.ecc.u-tokyo.ac.jp/gpsshogi/ (GPS 将棋 rev.1363 , OSL rev.3203 を使用)

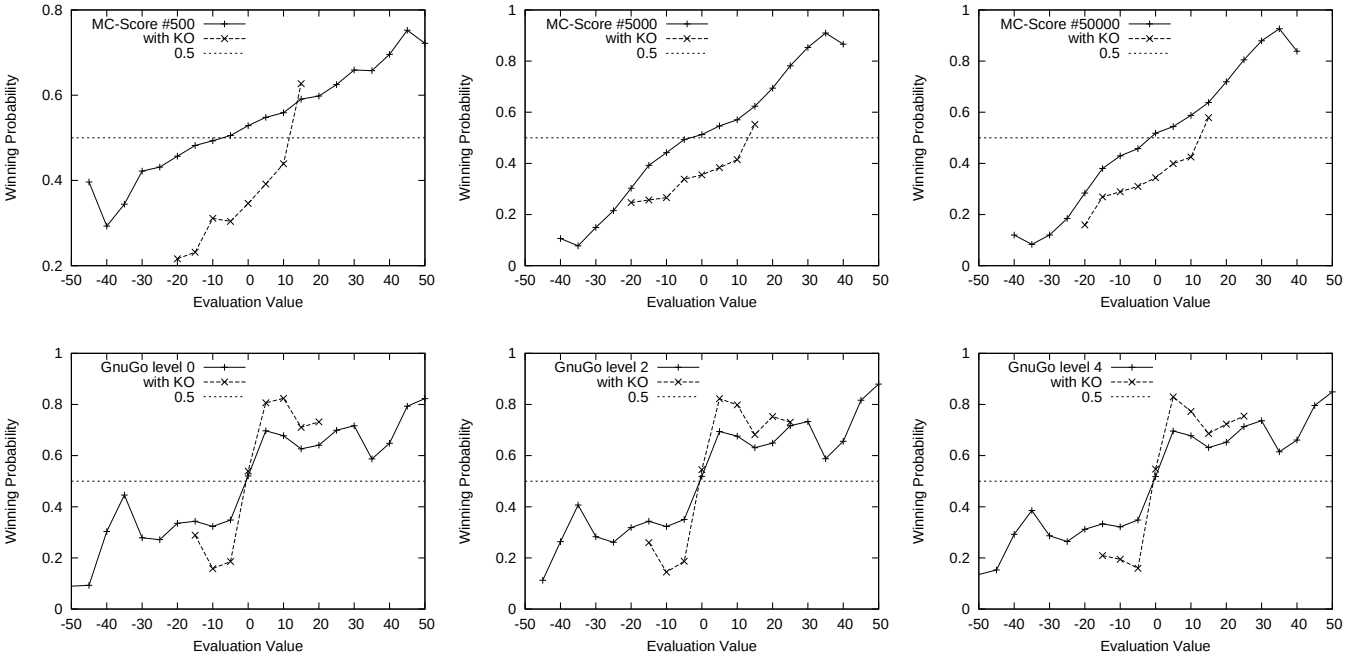

図 6 MC-Score and GnuGo with Ko position. (up: MC-Score, low: GnuGo, left: 500 playouts / level 0, center 5,000 playouts / level 2 , right: 50,000 playouts / level 4)

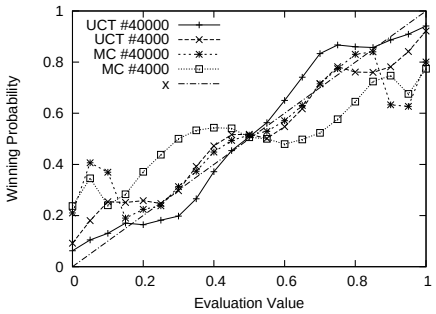

図 7 UCT and MC in Shogi

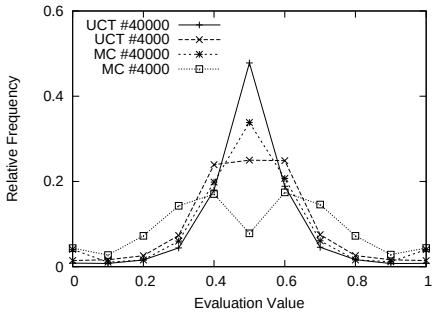

図 8 Distribution of Evaluation Value : UCT and MC in Shogi

表 2 Performance of Evaluation Methods: Shogi

	#playouts	$E(v)$
UCT	40,000	0.538
	4,000	0.541
MC	40,000	0.540
	4,000	0.542

ト数が 4,000 のモンテカルロ法では Evaluation Curve が単調増加していない．また，プレイアウト数 4,000 の UCT とプレイアウト数 40,000 のモンテカルロ法も，評価値 0,1 付近では単調ではなくなっている．プレイアウト数 40,000 の UCT が中では一番安定しており，最も正確であると考えられる．

評価値の期待値を計算した結果を表 2 にまとめた．モンテカルロ法が UCT がよりも良く，プレイアウト数が少ない方が良いなど，値の差は小さいながら，予想と全く逆の結果が得られた．

評価値の分布は図 8 のようになっており，プレイアウト数 40,000 の UCT の評価の多くが 0.5 付近に集まっていることがわかる．そのために期待値が 0.5 に近くなり，表 2 のように他の手法よりも低い結果となったと考えられる．

現在のモンテカルロシミュレーションはただランダムに選ぶだけなので，人間の強さと比べると非常に弱いプログラムになっている．そのため，実験で使用したプログラムでは人間の棋譜を正しく評価するのが難しかったということが考えられる．シミュレーション

に知識を利用するなど，評価の精度を改善することでこの問題は解決する可能性がある．

5. おわりに

本稿では，囲碁におけるモンテカルロ木探索手法を対象として，棋譜データにおける勝率を利用したEvaluation Curveによる評価を提案し，実験によりその有効性を示した．さらに，Evaluation Curveでは評価値の分布による影響を無視していたが，それを補完する新しい評価手法を提案した．その実験ではEvaluation Curveでは見分けられなった，探索手法やパラメータの違いによる性能の違いを見つけるなど，提案手法の有効性を示すことができた．今後は，指標の妥当性やEvaluation Curveと指標の結果が異なる場合についてなど研究を深めていきたい．

また，この手法はモンテカルロ木探索手法だけでなく，評価関数とミニマックス探索との組み合わせである従来型の手法でも有効である．今後は様々なゲーム，プログラムを対象として実験を行いたい．

参考文献

1) B. Abramson. Expected-outcome: A general model of static evaluation. *IEEE Trans. Pattern Analysis and Mach. Intell.*, 12(2):182–193, 1990.
2) D. Billings, A. Davidson, J. Schaeffer, and D. Szafron. The challenge of poker. *Artificial Intelligence*, 134(1-2):201–240, 2002.
3) B. Bouzy and B. Helmstetter. Monte Carlo Go developments. In *Advances in Computer Games. Many Games, Many Challenges*, pp. 159–174. Kluwer Academic Publishers, 2003.
4) B. Brügmann. Monte Carlo Go. Technical report, Physics Department, Syracuse University, 1993.
5) M. Buro. Improving heuristic mini-max search by supervised learning. *Artificial Intelligence*, 134(1–2):85–99, Jan. 2002.
6) Chaslot, M. H. Winands, I. Szita, and J. H. van den Herik. Parameter tuning by the cross-entropy method. In *Proceedings of EWRL*. Springer, 2008.
7) R. Coulom. Efficient selectivity and backup operators in monte-carlo tree search. In H. J. van den Herik, P. Ciancarini, and H. H. L. M. Donkers eds., *Computers and Games*, Vol. 4630 of *Lecture Notes in Computer Science*, pp. 72–83. Springer, 2006.
8) R. Coulom. Computing elo ratings of move patterns in the game of go. In *Computer Games Workshop*, Amsterdam / The Netherlands, 2007.
9) S. Gelly and D. Silver. Combining online and offline knowledge in uct. In *ICML '07: Proceedings of the 24th international conference on Machine learning*, pp. 273–280, New York, NY, USA, 2007. ACM.
10) S. Gelly, Y. Wang, R. Munos, and O. Teytaud. Modification of uct with patterns in monte-carlo go. Technical Report RR-6062, INRIA, 2006.
11) M. L. Ginsberg. GIB: steps toward an expert-level bridge-playing program. In *Sixteenth International Joint Conference on Artificial Intelligence*, pp. 584–589, 1999.
12) D. Gomboc, M. Buro, and T. A. Marsland. Tuning evaluation functions by maximizing concordance. *Theor. Comput. Sci.*, 349(2):202–229, 2005.
13) L. Kocsis and C. Szepesvari. Bandit based monte-carlo planning. In *Machine Learning: ECML 2006*, Vol. 4212, pp. 282–293. Springer, 2006.
14) B. Sheppard. World-championship-caliber Scrabble. *Artificial Intelligence*, 134(1-2):241–275, 2002.
15) D. Silver, R. Sutton, and M. Müller. Sample-based learning and search with permanent and transient memories. In A. McCallum and S. Roweis eds., *Proceedings of the 25th Annual International Conference on Machine Learning (ICML 2008)*, pp. 968–975. Omnipress, 2008.
16) S. Takeuchi, T. Kaneko, K. Yamaguchi, and S. Kawai. Visualization and adjustment of evaluation functions based on evaluation values and win probability. In *Proceedings of the Twenty-Second National Conference on Artificial Intelligence (AAAI-2007)*, pp. 858–863, 2007.
17) J. van Rijswijck. Learning from perfection: A data mining approach to evaluation function learning in awari. In T. A. Marsland and I. Frank eds., *Computer and Games*, No. 2063 in LNCS, pp. 115–132, Hamamatsu, Japan, Oct. 2001. Springer-Verlag.
18) H. Yoshimoto, K. Yoshizoe, T. Kaneko, A. Kishimoto, and K. Taura. Monte carlo go has a way to go. In *Proceedings of the Twenty-First National Conference on Artificial Intelligence (AAAI-2006)*, pp. 1070–1075, 2006.
19) 美添. モンテカルロ木探索 – コンピュータ囲碁に革命を起こした新手法. 情報処理, 49(6):686–693, 2008.
20) 久米. 将棋倶楽部 24 万局集. ナイタイ出版, 2002.