Vol.2014-GI-32 No.3
2014/7/5

コンピュータ囲碁におけるPTSを用いたモンテカルロ木探索

田中 一樹[1,a)]　藤田　玄[1]

概要：近年，モンテカルロ木探索，主にUCTと局面評価関数を組み合わせた手法が広く知られている．しかし，いまだコンピュータ囲碁の局面評価関数の多くは，中途の局面の評価を行うので一回の局面評価に時間がかかる傾向にある．本研究では，UCTの中で着手決定に用いられるプレイアウトの終局盤面に注目し，終局状態の局面の統計情報を用い，少ない演算コストで盤面評価を行う手法を提案する．また，既存手法であるFuegoに評価関数を組み合わせることで性能評価を行う． その結果，UCT単独のFuegoに対して6割強の勝率を示した．

Monte Carlo Tree Search Using the Playout Territory Statistics in Computer Go

Kazuki Tanaka[1,a)]　Gen Fujita[1]

Abstract: In recent years, methods of Monte Carlo tree search, combined evaluation function and UCT have been widely known. However, many evaluation functions of the computer go still tends to be heavy computational cost. In this study, we propose a evaluation function which focus on play out based on UCT, In the proposed evaluation function, a statistical information of resulting hoard with applied little computational cost. Performance evaluation shows the proposed method has over 60% winning percentage against Fuego.

1. はじめに

近年の研究でモンテカルロ木探索と評価関数を組み合わせた探索が有効であることが示されている [1][2].

2000年代からのコンピュータ囲碁では，評価関数に頼らない，モンテカルロ木探索 [3]，主に探索に期待値UCB（Upper Confidence Bound）[4] を用いたUCT(UCB applied to Trees)[5] が主流になっている．これは囲碁の石が将棋やチェスの駒と違い価値の明確な違いがないことや，短期的な良い手が長期的な良い手になるとは限らない，探索する範囲が広大である，感覚的な部分が多いこと，などが途中局面の有効な局面評価を行う際に実行時間がかかることが原因と考えられる [6]．また評価に必要な時間計算量が大きいことが木探索に用いる際のデメリットになる．そのためコンピュータ囲碁におけるモンテカルロ木探索と局面評価関数を組み合わせた研究は少ない．また，コンピュータ囲碁では，持ち時間の探索制約があるため，すべての合法手を探索することが難しい．Fuego[7] では，探索する候補をパターンマッチングで評価することで選出し探索の効率化を図っている．しかしながら少ない演算量で，精度の高い局面評価関数を用いることで，さらなる探索の効率化と棋力の向上が期待できる．

そこで本研究では，コンピュータ囲碁におけるモンテカルロ木探索にプレイアウト情報から生成した局面評価関数を組み合わせる手法を提案する．組み合わせる評価関数には，プレイアウトの情報を集計することで評価に必要な時間を削減した，局面評価関数 PTS(Playout Territory Statistics) を用いる．合法手から探索する候補手選出にPTSを用いることで，探索の効率化を図った木探索を提案

[1] 大阪電気通信大学

[a)] mi14a005 @ oecu.jp

する．PTS の評価値に基づき探索候補手の効率化の手法として，1)PTS の値をもとに合法手列をソートする手法，2)PTS の値に閾値を設け候補手列を生成する手法，3) 閾値以下の列を評価関数の評価値の絶対値でソートする手法を提案する．それらを従来の Fuego との対局による評価実験を行う．

2. UCT

UCT は，2006 年 9 月に Kocsis と Szepesvari によって発表された，モンテカルロ木探索の代表的なアルゴリズムである．UCT では，UCB 値を利用し，以下の 1 から 5 を時間内に繰り返すことで図 1 に示すような木を作成する [6]．

(1) UCB1 値による局面選択
根節点から，UCB 値の高い子接点を選択しながら末端節点まで枝をたどる．

(2) 局面の展開
末端の節点のプレイアウト回数が閾値を越えていれば，末端節点の局面から合法手を選出し，末端の子節点として節点を展開する．

(3) プレイアウト
2 で展開した節点の局面からプレイアウトを行う．

(4) UCB1 値の更新
プレイアウトの結果によって得られた評価値を根節点までたどって節点の値を更新する．

(5) 探索続行の判断制限時間や総プレイアウト数などの制限に達していなければ終了する．そうでなければ 1 に戻る．

囲碁に置き換えた場合，最後的に根節点の子節点から評価値が一番高い手を選択し着手を決定する．

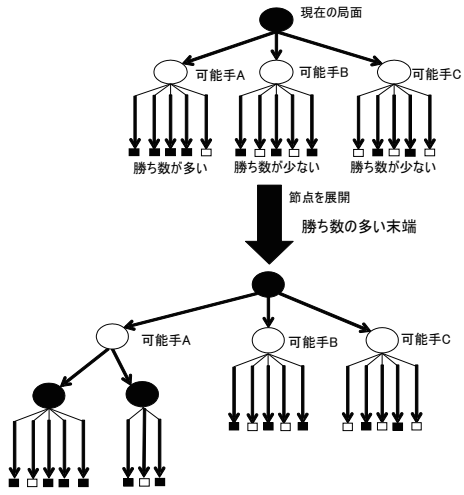

図 1 UCT のノードの展開

2.1 Fuego の探索候補手選出

コンピュータ囲碁は，持ち時間の制約によりすべての着手可能手を探索評価することができない，そのため Fuego では，手順 (1) の段階で盤面上の着手可能手に対しパターンマッチングを用いることで，探索する候補を選出し限られた時間でプレイアウトをよりよい候補に集中させている．このとき候補手列の先頭にある合法手が優先的に探索される．しかし Fuego の合法手列挙は盤面の端から順番に列挙されていくので，有効な手が探索されないことがある．

3. 提案手法

本研究では，Fuego の探索候補手選出の問題点に着目し，評価関数を用いた候補手選出を行うことで探索の効率化を図る．使用する評価関数はプレイアウトの統計情報をもとに評価する PTS を提案する．

これにより，相手が有利な交点，自分が有利な交点，どちらにも属していない交点がわかる．パターンマッチだけでなく地も考慮した候補手選出が行える．

探索の具体的な効率化を図る手法としては，PTS の値をもとに合法手列をソートする手法，PTS の値に閾値を設け合法手列を分け閾値以下の列を先頭につなげる手法，閾値以下の列を PTS の評価値の絶対値でソートする手法，3 つのパターンを提案する．

3.1 PTS(Playout Territory Statistics)

PTS はプレイアウトの最終局面の石の配置の統計を取ることで盤面上の石の存在する確率を求める局面評価関数である．

交点の石の存在する確率を求めることで，現局面まだ着手されていない未確定な交点が，どちらの色に属している確率を計ることができる．

またプレイアウトの情報を再利用することでを PTS を求めることによる計算量の増加はわずかである．

評価値が取る範囲は+1 から-1 の範囲であり，評価値が-1 に近いほど白に属した地である．+1 に近いほど黒に属した地である．0 に近いほどどちらにも属していない．n がプレイアウトの総数，n_i は i 番目のプレイアウト，$state(p)$ は点 p の状態を意味する，$P(n_i, p)$ が i 番目のプレイアウト時の点 p の状態 (式 1)，としたとき点 p の PTS の値を求める式は以下のようにあらわされる (式 2)．

$$P(n_i,p) = \begin{cases} 1(state(p) = \text{黒のとき}) \\ 0(state(p) = \text{空点のとき}) \\ -1(state(p) = \text{白のとき}) \end{cases} \quad (1)$$

$$PTS(p) = \Sigma_{i=1}^{n} \frac{P(n_i,p)}{n} \quad (2)$$

3.2 PTS を使った探索候補手選出の効率化

プレイアウトの統計情報に基づいた評価関数で，盤面上の合法手を評価し結果を候補手選出に用いることで探索の効率化を図る．評価結果を元に効率化する手法として，

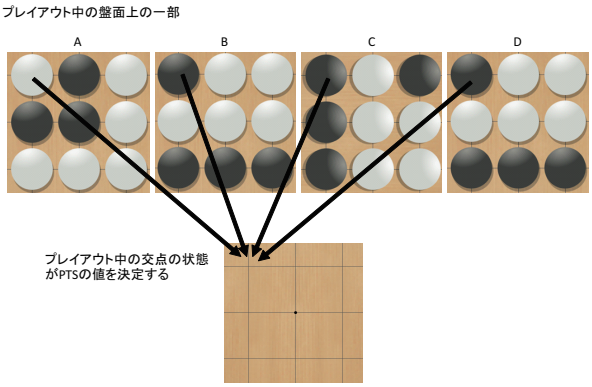

図 2 PTS の集計イメージ

PTS の値をもとに合法手列をソートする手法，PTS の値に閾値を設け合法手列を分け閾値以下の列を先頭にする手法，閾値以下の列を PTS の評価値の絶対値でソートする手法の 3 つがある．以下に詳細を説明する．

3.2.1 手法 1　PTS の値で合法手列をソート

盤面上の着手可能な合法手を列挙しできた候補手列をＰＴＳで求めた評価値に基づきソートする．また現在プレイしている側の色に合わせてＰＴＳ値を変換する (本来黒が正の数になるところを白を正の数，黒を負の数へ)．

変換した値に基づきソートした列は昇順にすることにより相手側が，降順にすることで自分側が有利な交点が優先的に候補手に選ばれる．

3.2.2 手法 2　PTS の値に閾値を与えた候補手列

手法 1 では，すべての合法手をソートしていたため，候補手選出に偏りが出たが，閾値を用いることで候補手選出にある程度のばらつきを持たせることができ，PTS の値で合法手列をソートすることで実際は良い手が列の後方に集中するといった状態を避ける．盤面上の着手可能な合法手を列挙する際に，その交点のＰＴＳ値の絶対値を調べ，あらかじめ決めた閾値より小さければα列に追加し，大きければβ列に追加する．盤面上すべての合法手の列挙が終了後，α列の後ろにβ列を結合し候補手列とする．それにより閾値以下の交点が優先的に候補手に選ばれる．

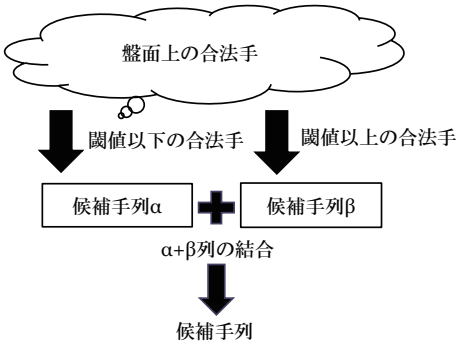

図 3 手法 2 の候補手列の生成手順

3.3 手法 3　手法 2+ソート

手法 3 は，手法 2 で候補手選出にある程度のばらつきを持たせた状態で，候補手列αを昇順ソートすることで攻めの手を優先させることができる，候補手列αを降順ソートをすることで守りの手を優先させることができる．

手法 2 で，できたα列を結合前にＰＴＳ値に基づきソートする．手法 1 と違いソートの際は絶対値によるソートを行う．

今回の実験では PTS の評価値の絶対値の昇順と降順をそれぞれ実験した．昇順はどちらにも属していない交点が優先的に，降順はどちらかに属している交点が優先的に探索される．

4. 性能評価

提案手法の 3 つの手法それぞれを既存の Fuego との対局による評価実験を行い本手法の評価を行う．

表 1 に今回の評価実験を行った条件と環境を示す．

表 1 評価条件

使用プログラム	Fuego Version1.0
プロセッサ	Intel(R)i7-2600 CPU3.4GHz
メモリ	16GB
OS	Windows7 64bit
一手あたりの思考時間	10 秒

4.1 評価結果

実験の結果を以下に示す．今回の実験で，手法 1 での 9 路盤の結果を表 2 に，19 路盤での結果を表 3 に示す．手法 2 の 9 路盤での結果を表 4 に，19 路盤での結果を表 5 に示す．手法 2 の閾値を複数パターン評価するのは，有効な閾値を探すためである．手法 3 の結果を表 6 に示す．それぞれ 9 路盤では 50 試合，19 路盤では 100 試合を行った (手法 2 では 9 路盤でも 100 試合行っている)．

表 2 手法 1 の 9 路盤での 50 試合の結果

ソート	勝率	勝ち数 (黒勝ち，白勝ち)	負け数
昇順	32%	16 (0，16)	34
降順	44%	22 (5，17)	28

表 3 手法 1 の 19 路盤での 100 試合の結果

ソート	勝率	勝ち数 (黒勝ち，白勝ち)	負け数
昇順	15%	15(1，14)	85
降順	17%	17(4，13)	83

表 4 手法 2 の 9 路盤での 100 試合の結果

閾値	勝率	勝ち数 (黒勝ち , 白勝ち)	負け数
0.10	68%	68(31 , 37)	32
0.15	58%	58(32 , 26)	42
0.20	60%	60(30 , 30)	40
0.22	67%	67(29 , 38)	33
0.25	63%	63(31 , 32)	37
0.27	66%	66(37 , 29)	34
0.30	62%	62(32 , 30)	38
0.35	64%	64(34 , 30)	36

表 5 手法 2 の 19 路盤での 100 試合の結果

閾値	勝率	勝ち数 (黒勝ち , 白勝ち)	負け数
0.10	51%	51(22 , 29)	49
0.15	58%	58(24 , 34)	42
0.20	63%	63(29 , 34)	37
0.22	54%	54(25 , 29)	46
0.25	64%	64(35 , 29)	36
0.27	68%	68(34 , 34)	32
0.30	64%	64(33 , 31)	36
0.35	58%	58(30 , 28)	42

表 6 手法 3(閾値 0.25) での 19 路盤 100 試合の結果

ソート	勝率	勝ち数 (黒勝ち , 白勝ち)	負け数
昇順	62%	62(28 , 34)	38
降順	54%	54(22 , 32)	46

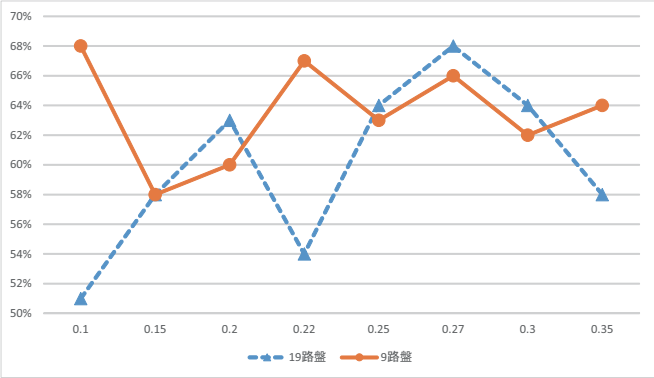

図 4 手法 2 の閾値による勝率の変化

表 7 手法 2 の 19 路盤での一秒間のプレイアウト回数

囲碁プログラム	平均プレイアウト数/秒
Fuego1.0	10648
閾値 0.10	10366
閾値 0.15	9895
閾値 0.20	9876
閾値 0.22	9876
閾値 0.25	9954
閾値 0.27	9962
閾値 0.30	10089
閾値 0.35	9956

4.2 考察

結果として，手法 1 は，地になる確率をもとに探索候補手を優先的に決める手法だが，表 2，表 3 から，あまり有効でないことが分かる．これは，相手の着手に大きく影響を受け安く悪手に近いような着手が多く選ばれるようになってしまっていることが，原因であると考えられる．

表 4，表 5 から手法 2 の閾値による候補手列が効果があることがわかる．また閾値が効果がある範囲が 9 路盤と 19 路盤では違いがあることも見て取れる (図 4)．これは序盤での着手可能な交点の数によると考えられる．

表 6 から手法 3 は，あまり有効でないことがわかる．これは，ソートにかかるオーバーヘッドがプレイアウト数の低下につながり勝率が下がっていると考えられる．手法 1，手法 3 は攻めの手が比較的有効であることも分かる．

次に，手法 2 における一秒間あたりの平均プレイアウト数を求めた結果を表 7 に載せる．表 7 から手法 2 を採用した際のオーバーヘッドによる計算量の低下は少ないと考えられる．また図 4 から閾値による，計算量の変化と，勝率との関連性は小さいと考えられる．

5. おわりに

本研究では，プレイアウトの統計情報を使った木探索の探索候補手選出の効率化を提案した．プレイアウトの統計情報を使った木探索の探索候補手選出の効率化の有効性を示すことができた．閾値を用いた候補手列生成では，UCT を上回る勝率を出すことができた．今後の課題としては，閾値を用いた候補手列生成を行ったときの処理時間削減やさらに細かな閾値を調べることでより有効な閾値を導く必要がある．

参考文献

[1] 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).
[2] 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, The 18th Game Programming Workshop 2013
[3] Coulom, R.: Efficient selectivity and backup operators in montecarlo tree search, Proceedings of the 5th International Conference on Computers and Games, Turin, Italy (2006).
[4] Auer, P., Cesa-Bianchi, N. and Fischer, P.: Finite time Analysis of the Multi-armed BanditProblem, Machine Learning, Vol. 47, pp. 235-256 (2002).
[5] Kocsis, L. and Szepesvari, C.: Bandied Based Monte-Carlo Planning, Proceedings of the 15th European Conference on Machine Learning, pp.282-293 (2006).
[6] 美添 一樹 , 山下 宏 , コンピュータ囲碁 ―モンテカルロ法の理論と実践―, 共同出版社,(2012).
[7] Fuego, ”http://fuego.sourceforge.net/”