グループ化を用いたモンテカルロ木探索の性能の分析

坪倉　弘治[1,a)]　西野　順二[1,b)]

概要：モンテカルロ木探索は探索空間が大きい場合でもおおむね効果的ではあるが、1 ターンに全ての自分の駒を動かすことができるターン制戦略ゲームにおいては、着手の組み合わせ爆発によって分岐因子が膨大になるため、一手目の段階で全く考慮されないままの手が大多数となる。Grouping Nodes はこの問題を解決する方法の一つである。しかし、探索性能が向上した理由については議論されてこなかったため、本研究では大規模でありながら素性が明らかな実験ゲーム木の構成を提案し、この木を利用してモンテカルロ木探索の探索性能を検討した。その結果、グループ化により分岐因子を減少させることでモンテカルロ木探索の性能が上がり、グループ化を複数回行うことでより性能が上昇したが、その効果はグループ化の工夫と利得のまとまり方に依存することを明らかにした。

Analysis of performance of Monte Carlo tree search with grouping

Tsubokura Koji[1,a)]　Nishino Junji[1,b)]

Abstract: Although the Monte Carlo tree search is generally effective even when search space is large, in turn-based strategy game in which all the pieces can be moved in one turn, most of the moves are not considered at all in the first step because the branching factors become enormous due to the combination explosion of the moves. Grouping Nodes is one solution to this problem. However, the reason why the search performance has been improved has not been discussed. In this study, we propose composition of large scale experimental game trees with clear features and investigate the search performance using this tree. As a result, it was found that the performance of the Monte Carlo tree search was improved by reducing the branching factor by grouping, and by performing grouping more than once, but the effect depended on the grouping contrivance and how to arrange the gain.

1. はじめに

モンテカルロ木探索は、囲碁などの評価関数の作成が困難だが、簡単にゲームの勝敗判定ができるというゲームにおいて有効なアルゴリズムであり、探索空間が大きい場合でもおおむね効果的であることから、ターン制戦略ゲームにおいても用いられている。しかし、探索木の分岐因子が数億あるなど膨大なゲームでは、一手目の段階で全く考慮されないままの手が大多数となる。Grouping Nodes(グループ化) はこの問題を解決する方法の一つである。グループ化は TUBSTAP[1] における M-UCT[2] で効果があった。それに対して、M-UCT を発展させた S-UCT[3] はグループ化の効果があまり現れなかった。このようにグループ化の効果が現れる場合と現れない場合が存在するが、その違いや理由について議論はされてこなかった。そこで、本研究では大規模でありながら素性が明らかな実験ゲーム木の構成を提案し、この木を利用してモンテカルロ木探索の探索性能を検討する。

ターン制戦略ゲームとは、プレイヤがターンごとに行動するゲームで、1 ターンに全ての自分の駒（以下「ユニット」と呼ぶ）を動かすことができる（複数着手性）[1]。複数着手性による着手の組み合わせ爆発によって、ターン制戦略ゲームは分岐因子が膨大になる。

2. モンテカルロ木探索

モンテカルロ木探索 (Monte Carlo Tree Search; MCTS)

[1] 電気通信大学大学院
The University of Electro-Communications
[a)] tsubokura.koji@mail.uec.jp
[b)] nishinojunji@uec.ac.jp

は、モンテカルロ法に木探索の考えを取り入れた探索法であり、2006 年に R. Coulom によって提唱され、[4][5] 同氏が作成した囲碁プログラム「Crazy Stone[4]」や、「MoGo[6]」などで使用されている。モンテカルロ木探索は、囲碁などの評価関数の作成が困難だが、簡単にゲームの勝敗判定ができるというゲームにおいて有効なアルゴリズムであり、探索空間が大きい場合でもおおむね効果的であることから、ターン制戦略ゲームにおいても用いられている。

モンテカルロ木探索では、プレイアウトを繰り返し行い、節点の価値を近似的に求める。プレイアウトには次の 4 段階がある。

(1) 木探索:何らかの方法で根節点 v_0 から展開していない節点 (先端節点)v_l まで探索木を下る。
(2) 展開:v_l のプレイアウト回数が既定値を超えている場合に v_l を展開し、v_l の各子節点についてランダムシミュレーションとバックアップを行う。
(3) ランダムシミュレーション:v_l の状態 s_l からゲームの終端までランダムに着手し、v_0 のターンプレイヤ p から見た利得 Δ を得る。
(4) バックアップ:ランダムシミュレーションで得られた結果を元に v_l とその先祖節点の値を更新する。

ここで v_l のプレイアウト回数とは、木探索における v_l の訪問回数のことである。

ゲームの利得については、ゲームの勝ちを 1、引き分けを $\frac{1}{2}$、負けを 0 とすることが多い。

2.1 UCT 探索

多腕バンディット (Multi-Armed Bandit) 問題の解法の一つである UCB1 アルゴリズムを、モンテカルロ木探索の木探索で下る節点の選択方法に使用したのが UCT(UCB applied to Trees) 探索である。この探索法は Kocsis, Levente and Szepesvári, Csaba らによって 2006 年に提唱された。[7]

このアルゴリズムでは UCB 値が最も高い子節点が選択される。節点 j の子節点からバックアップされた利得の平均 (平均利得) を $\overline{x_j}$、プレイアウト回数を n_j、親節点のプレイアウト回数を n として、UCB 値は以下のように定義される。

$$ucb = \overline{x_j} + C\sqrt{\frac{2\log n}{n_j}} \qquad (1)$$

定数 C は問題に合わせて調整される。UCT 探索は十分な探索時間が与えられたとき、探索木はネガマックス木に収束する。[8]

今回グループ化を導入するモンテカルロ木探索は UCT 探索とした。

2.2 ターン制戦略ゲームにおける MCTS の問題点

モンテカルロ木探索は探索空間が大きい場合でもおおむね効果的ではあるが、ターン制戦略ゲームにおいては前節で述べた理由で分岐因子が膨大になるため、探索木の根節点の子節点の探索すら十分に行うことができなくなり、探索の結果得られた節点の価値の誤差が非常に大きくなる。したがって最適な手の決定において最適手を選択できない確率が高くなる。

3. Grouping Nodes

Grouping Nodes(以下節点のグループ化と呼ぶ) での探索木の節点は行動節点とグループ節点からなる。節点のグループ化とは、これまでの探索木の節点を行動節点とし、探索木の任意の節点の子節点についていくつかのグループに分け、各グループ毎にグループ節点を作成して親節点と子節点の間に挿入することである [9]。これにより探索木は、子孫節点をたどる際に行動節点とグループ節点が交互に現れるようになり、分岐因子が減少する。

例として、ターン制戦略ゲーム TUBSTAP の AI、M-UCT[2] や S-UCT[3] が存在する。M-UCT は、複数着手性に着目し探索木の枝を各ユニットの行動に対応させることで分岐因子を減少させた手法である。S-UCT は、M-UCT を発展させ、ユニット行動の意思決定をユニットの決定と行動の決定に分割させた手法である。グループ化は M-UCT で効果があった。それに対して、M-UCT を発展させた S-UCT はグループ化の効果があまり現れなかった。

本研究では、グループ化を複数回行えるようにするため、既存研究のグループ化を次のように改変した。

- グループ節点と行動節点の区別をつけない。
- 節点の展開では、グループ節点と行動節点を同時に展開していたのを、展開は子節点のみとする。
- グループ化を k 回行っている時、MCTS を $k+1$ 回繰り返し手を決定する。各 MCTS の探索回数 (時間) は $\frac{1}{k+1}$ する。

3.1 ゲーム木の作成方法

グループ化によるモンテカルロ木探索の性能の変化を調査するため、分岐因子が膨大な複数の異なるゲーム木を用意したい。まず、探索を行うゲームは二人零和有限確定完全情報ゲームとする。ゲーム木をランダムに作成する方法として P-game[10] が存在し、UCT 探索用に変更を加えたものが UCT 探索の性能分析のために用いられている [7][11]。しかし、P-game はプレイヤがどれだけ勝ちに近づいたかを表す数値を各枝にランダムに割り当てるため、すべての枝情報を保持する必要があり、巨大なゲーム木を扱うのは不可能である。その代わり、P-game ではミニマックス値を計算できるため Regret 等の指標を用いることができる。

また、グループ化の効果を検証するため、グループ化し

たゲーム木で根節点から終端節点までに下った枝に対応する行動列と利得が得られたときに、もともとのゲーム木において同じ終端節点とその利得を適切に対応付け、節点の値を追跡できる必要もある。

そこで本研究では、ターン制戦略ゲームのように巨大なゲーム木でグループ化の効果を検証するため、元のゲーム木とグループ化したゲーム木どちらでも木の全体を保持することなく、節点の値を追跡できる手法を提案する。提案手法では次のような手法で探索を行うゲーム木を作成する。

- ゲーム木は、葉節点以外の節点が持つ子節点の数が N で、葉節点の深さが揃っている完全 N 分木とする。ゲーム木の深さを d とする。
- ゲーム木の枝に対応する行動 a は $a \in \{0, 1, \ldots, N-1\}$ と非負整数で表されるとする。
- 終端節点の利得は関数

$$f : \mathbb{R} \to [0, 1] \tag{2}$$

  で与える。(利得関数)
- 根節点から終端節点までの行動列 $a_0, \cdots, a_{d-1}$ が与えられたとき、行動列から f の定義域への変換は、行動列を N 進数の数とみなして整数値に変換する。

実験上、任意の利得の分布を持つ木を扱いたい。これを実現するためには、木の構造をランダムに組み替えるのではなく、全体が順序付けられた終端節点上にランダムな利得関数を設定すればよい。このため、以降では木の構造ではなく、終端節点への利得関数を様々に設定して実験を行うこととした。

4. グループ化の効果比較実験

グループ化により分岐因子を減らすことでモンテカルロ木探索の性能が向上するかを確認する実験を行った。実験は提案手法で作成した二人零和有限確定完全情報ゲームのゲーム木での、グループ化を行った AI(GroupAI) と行っていない AI(NativeAI) の対戦実験とし、ゲームが終局したときの終端節点の利得の平均値 $\bar{v}$ を求めた。

- AI の探索手法は UCT 探索とした。
- 対戦は先攻後攻 500 戦ずつ計 1000 戦行う。
- UCT 探索の探索回数 (プレイアウト回数) の制限を変化させ、$\bar{v}$ の変化を調べた。
- ゲーム木のサイズは分岐因子 256、深さ 4 とした。終端節点の数は $256^4 \approx 5 \times 10^9$

4.1 ゲーム木の設定

実験では、利得関数として次の 3 種類を用いた。

(1) $x \times \sin$

$$f(x) = \frac{|x \sin(x)|}{k}, (0 \le x \le k) \tag{3}$$

(2) 正弦波の重ね合わせ

$$f(x) = \left| \sum_{i=1}^{10} a_i \sin\left(\frac{x}{2i-1}\right) \right| \tag{4}$$

(3) 線形関数

$$f(x) = \frac{x}{\max(x)} \tag{5}$$

とした。k をランダムに変化させることで複数のゲーム木を表現する。

$k = 5$ と $k = 25$ の場合の $x \times \sin$ 関数のグラフをそれぞれ図 1・2 に示す。

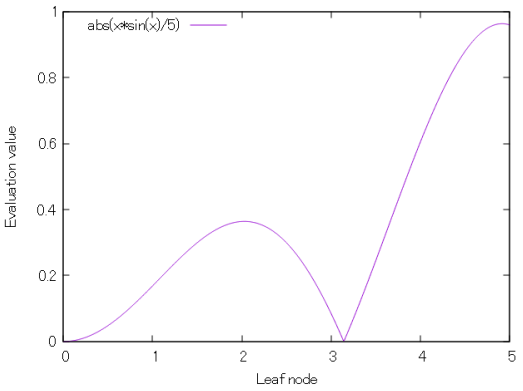

図 1 $k = 5$ の $x \times \sin$

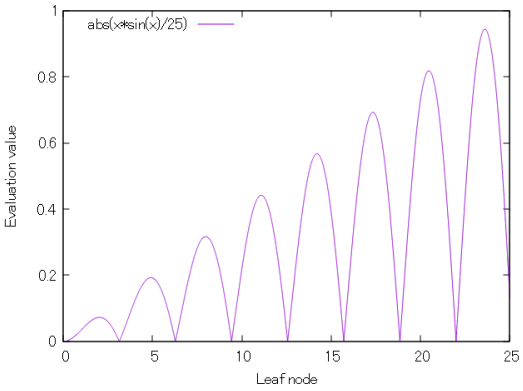

図 2 $k = 25$ の $x \times \sin$

k が大きいほど関数の山の数が多くなることがわかる。

この関数を実験のために使用する理由としては、山が複数あることで局所解が複数生まれうること、最適手が一方の端に偏ることが想定されることである。

次に正弦波の重ね合わせの関数の一例のグラフを図 3 に、線形関数のグラフを図 4 に示す。

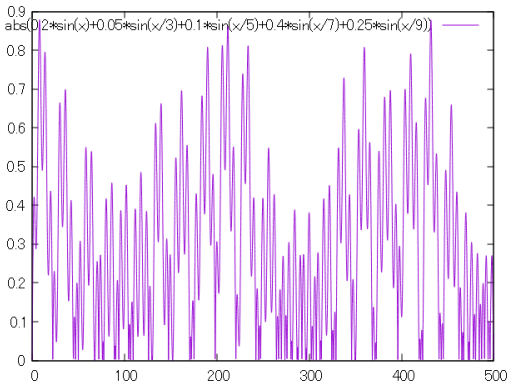

図 3 $\frac{|x\sin(x)|}{5}, (0 \leq x \leq 5)$

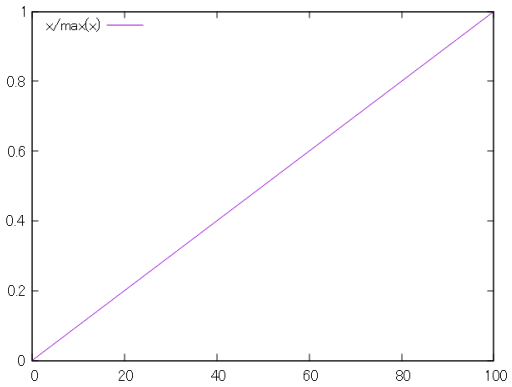

図 4 $\frac{|x\sin(x)|}{25}, (0 \leq x \leq 25)$

正弦波の重ね合わせはランダムな値を返す関数の代わりとして用い、そのような場合での性能の変化を調査するのが目的である。

4.2 実験の結果と考察

実験の結果を横軸に探索回数の制限であるプレイアウト回数、縦軸にゲームが終局したときの終端節点の利得の平均値 $\bar{v}$ をとったグラフにプロットした。その図を図 5・6・7・8 に示す。

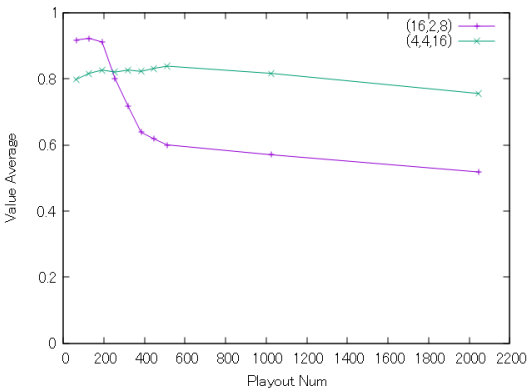

図 5 $x \times \sin \quad (k \in [5, 15])$ の結果

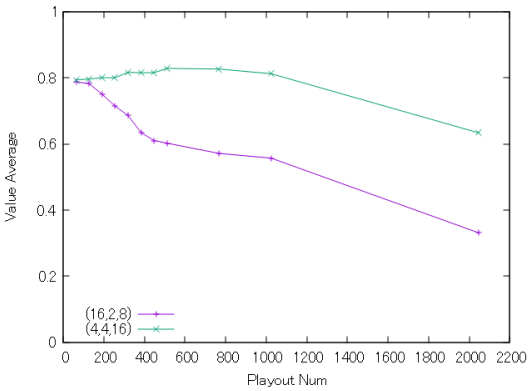

図 6 $x \times \sin \quad (k \in [5, 100005])$ の結果

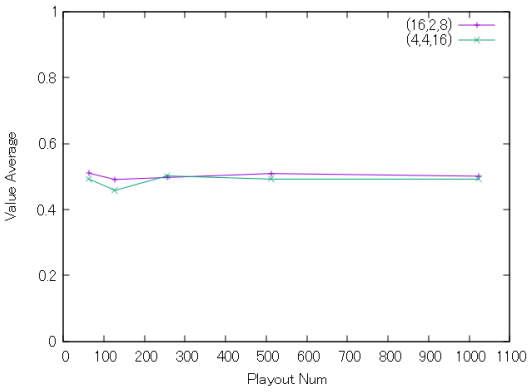

図 7 正弦波の重ね合わせの結果

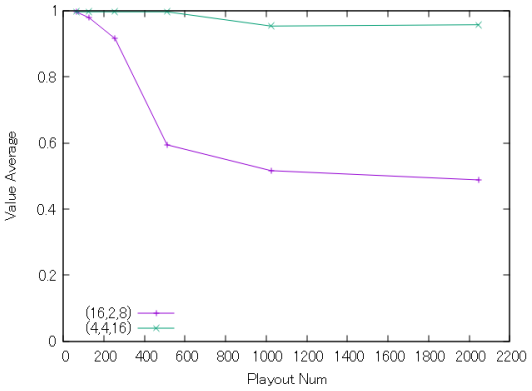

図 8 線形関数の結果の結果

図 5・6 より、プレイアウト回数が少ない場合に、グループ化を行った方がより優れた性能を示された。そしてプレイアウト回数が増えるに従い性能差が縮むが、複数回グループ化を行うほうがより優れた性能を示し、その差も比較的縮まらないことも示された。また、線形関数に関しては $x \times \sin$ と同様の結果が得られた。

次に正弦波の重ね合わせに関しては、プレイアウト回数に関わらず性能に差が現れなかった。

5. 終端節点の利得の偏りとグループ化の効果

グループ化による効果が現れるゲーム木と現れないゲーム木にこのような違いが現れた理由は、利得の値が近い終端節点を同じグループにまとめられたかそうではないかにあると考えられる。モンテカルロ木探索では、平均利得 $\overline{x_j}$ を節点 j の子節点からバックアップされた利得の平均で求めている。これは、子節点に突出して高い平均利得を持つ節点があったとしても、それ以外の節点で平均利得が低ければ、親節点の平均利得は平均化され低い数値になってしまうという問題が存在する。よって探索終了時に、本来であれば平均利得が高い子節点を持つ節点ではなく別の異なる節点が最適な節点として選択されうる。k 回のグループ化によって 1 ターン分の行動決定は $k+1$ 回、そしてモンテカルロ木探索も $k+1$ 回繰り返し行う必要があるため、もし最適手から外れた節点を一度でも選択してしまうと、それ以降の探索では決して最適手に到達できなくなってしまう。

UCT 探索においては UCB 値を利用して下る節点を決定するため、十分なプレイアウトを重ねれば、平均利得の高い子節点に多くのプレイアウトを振り分けることができ、この問題も解消できる。しかし、割り当てられるプレイアウト回数が少ない場合 (すなわち単純に探索時間が短いかプレイアウト回数に対して分岐因子が多すぎる場合) には、十分なプレイアウトを行うことができず同じ問題が発生する。

ここで、子節点に平均利得が近い節点が集まっている場合を考えれば、平均利得が子節点と親節点との間で大きく変化することがないので、そのような問題は発生しない。そして図 1・2・4 を見ると、ごく短い範囲において関数値が大きく変化するということがないため、利得の値が近い終端節点を同じグループにまとめられていると考えられる。次に図 3 を見ると、ごく短い範囲において関数値が大きく変化しているため、利得の値が近い終端節点を同じグループにまとめられていないと考えられる。したがって、グループ化による効果が現れるゲーム木と現れないゲーム木の違いは、利得の値が近い終端節点を同じグループにまとめられたかそうではないかにあると考えられる。

また同様に、プレイアウト回数が増えるに従い性能差が縮むことと、複数回グループ化を行うほうがより優れた性能を示し、その差も比較的縮まらないことも説明できる。まずプレイアウト回数が増えるに従い性能差が縮むことは、UCT 探索は十分なプレイアウトを重ねれば平均利得の高い子節点に多くのプレイアウトを振り分けることができ、親節点の平均利得は平均化され低い数値になってしまうという問題が解決できるので、最適手を選択できる確率が上昇することから説明できる。そして複数回グループ化を行うほうがより優れた性能を示しその差も比較的縮まらないことは、プレイアウト回数に対して分岐因子を減少させることで各節点に割り当てるプレイアウト回数が相対的に増加することから説明できる。

6. まとめ

本論文の結果は以下のようにまとめられる。

- 元のゲーム木とグループ化したゲーム木どちらでも木の全体を保持することなく、節点の値を追跡できる手法を提案した。
- グループ化により分岐因子を減少させることでモンテカルロ木探索の性能が上がった。
- グループ化を複数回行うことでより性能が上昇した。
- グループ化によって性能が向上するゲーム木としないゲーム木が存在する。
- その違いは利得の値が近い終端節点を同じグループにまとめられたかそうでないかにあると考えられる。

今後の課題としては、より多くの異なる利得関数で表現できるゲーム木で実験を行うことや、グループ化をする際に異なるグループ間で子節点が重複するような場合への対応があげられる。

参考文献

[1] 村山公志朗, 藤木翼, 池田心ほか. 学術研究用プラットフォームとしての大戦略系ゲームのルール提案. ゲームプログラミングワークショップ 2013 論文集, pp. 146–153, 2013.

[2] 武藤孝輔, 西野順二. ターン制戦略ゲームにおける uct とファジィ評価の適用. 日本知能情報ファジィ学会 ファジィシステム シンポジウム 講演論文集第 31 回ファジィシステムシンポジウム, pp. 226–229. 日本知能情報ファジィ学会, 2015.

[3] 提橋凜, 西野順二. ターン制戦略ゲームにおけるユニット抽象化探索の性能. 日本知能情報ファジィ学会 ファジィシステム シンポジウム 講演論文集, Vol. 34, pp. 810–814, 2018.

[4] Rémi Coulom. Efficient selectivity and backup operators in monte-carlo tree search. In *International conference on computers and games*, pp. 72–83. Springer, 2006.

[5] Cameron B Browne, Edward Powley, Daniel Whitehouse, Simon M Lucas, Peter I Cowling, Philipp Rohlfshagen, Stephen Tavener, Diego Perez, Spyridon Samothrakis, and Simon 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.

[6] Sylvain Gelly, Yizao Wang, Olivier Teytaud, Modification Uct Patterns, and Projet Tao. Modification of uct with patterns in monte-carlo go. 2006.

[7] Levente Kocsis and Csaba Szepesvári. Bandit based monte-carlo planning. In *European conference on machine learning*, pp. 282–293. Springer, 2006.

[8] Levente Kocsis, Csaba Szepesvári, and Jan Willemson. Improved monte-carlo search. *Univ. Tartu, Estonia, Tech. Rep*, Vol. 1, , 2006.

[9] Jahn-Takeshi Saito, Mark HM Winands, Jos WHM Uiterwijk, and H Jaap Van Den Herik. Grouping nodes for monte-carlo tree search. In *Computer Games Work-*

shop, pp. 276–283. Citeseer, 2007.

[10] Stephen JJ Smith and Dana S Nau. An analysis of forward pruning. In *AAAI*, pp. 1386–1391, 1994.

[11] 孝久今川, 知適金子. 多腕バンディットアルゴリズムの mcts への応用と性能の分析. ゲームプログラミングワークショップ 2014 論文集, 第 2014 巻, pp. 145–150, oct 2014.

