7P-02

General Game Playing におけるモンテカルロ木探索のシミュレーション戦略改善に向けた検討

上宮　佳晃†　　横山　大作†

明治大学 理工学部情報科学科‡

1．はじめに

General Game Playing(GGP)とは、初見のゲームを与えられるゲームルールだけで、高い精度でプレイするプログラムを実現することを目標とした問題カテゴリである。様々な種類の未知のゲームをプレイするため、正確な評価関数を必要としないモンテカルロ木探索(MCTS)を用いたプログラムが主流となっている。MCTS では、ランダムにゲームをシミュレートする手法が一般的だが、プレイアウトの過程でのシミュレーション戦略を工夫することで、探索の精度と効率を向上したいと考えた。

本論文では GGP で扱う未知の様々な形式のゲームに対して、シミュレーションを工夫した MCTS の有効性を検証し、手法とその性能について議論する。プレイヤの構築のためにスタンフォード大学のロジックグループが標準化した GGP のプラットフォームを用いた。GGP ライブラリを用いて MCTS によるプレイヤを作成し、ライブラリにある原始モンテカルロとの対戦実験を行った。プレイアウト回数が同じなら強いという結果になり正しく構築できた。問題としては、シミュレーション戦略を用いなければ十分に木を作ることができないということがわかった。

2．MCTS による GGP プレイヤの構築

本研究では GGP ライブラリで GDL を読み込み、それを用いて MCTS を行う GGP プレイヤを作成した。

2.1　Game Description Language（GDL）

GGP で扱うゲームのルールは論理プログラミング言語の一種である Game Description Language（GDL）[1]で表現される。ゲームのルールには、各プレイヤの役割や合法手によってゲームの状態がどのように変化するのかといったことが記述されている。この合法手や各プレイヤの役割は GGP ライブラリではそれぞれ Move クラス、Role クラスで扱われる。ゲームの盤面情報は真の事実の集合で表現されていて MachineState クラスで扱われる。盤面ごとに各プレイヤは論理的なルールで記述された合法手をゲームマスターに送信する。このとき、手番でないプレイヤは noop（no operation）というゲームの状態に影響しない信号を送る。

2.2　MCTS

MCTS は正確な評価関数が不要であるため、様々な未知の種類のゲームを扱う GGP に適している[2]。ノードの選択には UCT を用いた。メモリ上に構成されるツリーのノードはゲームの盤面状態に対応する。

2.3　StateMachine クラス

GGP ライブラリには GDL の解析を行うツールがいくつかある。MCTS を構築する上で主に次のライブラリを使用した。

・`MachineState getNextState (MachineState state, List<Move> moves)`

現在の状況 state に対して合法手 moves を行った際の次の状態を返す。

・`int getGoal(MachineState state, Role role)`

引数の現在の状態 state とプレイヤ role に応じて０～１００の得点（Goalvalues）を返す。

・`List<Move> getRandomJointMove (MachineState state)`

引数で受け取る現在の状態から、それぞれのプレイヤが行えるランダムな合法手をプレイヤ順で記録された List<Move>型で返す。

・`MachineState getNextState (MachineState state, List<Move> moves)`

・現在の状況 state に対して合法手 moves を行った際の次の状態を返す。

2.4 MCTS の動作手順

各ターンにおいて MCTS による GGP プレイヤは次のように動作する。

step1.ターンの始めに現在の盤面状態を表す GDL を読み込む。この盤面情報が、構築しているモンテカルロ木にあれば、そのノードからリーフノードまで UCT を用いて探索を行う。

step2.リーフノードから終局状態に到達するまで getRandomJointMove メソッドでその盤面状態の合法手をランダムに取得し、getNextState メソッドを用いて次の盤面状態へと移行する。

step3.getGoal メソッドで各プレイヤの報酬を取得しバックプロパゲーションを行う。また、プレイアウトでリーフノードの次に出現したノードの拡張も行う。

3．実験

本稿では、MCTS,UCT を元に構築した MCTS プレイヤと GGP ライブラリにデフォルトで搭載されている

Towards an improvement of simulation strategy for Monte Carlo tree search in General Game Playing
†Yoshiaki Uemiya, Daisaku Yokoyama
‡Meiji University

原始モンテカルロプレイヤを対戦させ MCTS プレイヤの性能を検証していく。

3.1 実験設定

手番の持ち時間であるプレイクロックの時間を 15 秒として先攻と後攻を交互に変えて実験を行う。初めに両プレイヤプレイアウト回数を同じにして実験を行う。MCTS プレイヤは該当する盤面情報が構築されている木に含まれているかを確認する手順が発生するため、一回当たりのプレイアウト時間が原始モンテカルロプレイヤに比べて長くなってしまう。よって、原始モンテカルロプレイヤのプレイアウト回数を一手前の MCTS プレイヤのプレイアウト回数と同じになるようにする。

3.2 使用するゲーム

今回の実験では終局状態までの最大ステップ数や合法手の数が異なる以下の三つの有限完全情報ゲームを行う。

(1)Tic-Tac-Toe：三目並べ。3×3 の盤面に縦・横・斜めいずれかに 3 つ直線状に自分のマークを並べたプレイヤの勝ちとなる。

(2)ConnectFour：重力付き四目並べ。6×8 の盤面にいずれかの列を選択すると自分のディスクが落下し積み重なる。縦・横・斜めいずれかに 4 つ直線状に自分のディスクを並べたプレイヤの勝ちとなる。

(3)ConnectFive：五目並べ。8×8 の盤面に縦・横・斜めいずれかに 5 つ直線状に自分のマークを並べたプレイヤの勝ちとなる。

3.3 結果と考察

表 1 原始モンテカルロプレイヤに対しての MCTS プレイヤの成績(同じプレイアウト回数)

	勝ち	負け	引き分け	平均ターン数
Tic-Tac-Toe	21	0	29	7.35
ConnectFour	43	7	0	23.5
ConnectFive	33	17	0	35.6

表 1 から、Tic-Tac-Toe ではゲームの性質上 MCTS プレイヤが先行の場合は勝ちとなり、後攻のときは引き分けとなることが多かった。また ConnectFour では原始モンテカルロプレイヤに対して優れた性能を示すことができた。だが、ConnectFive では約 3 割の負けが発生した。前者二つは自分の手番の合法手が多くても 8,9 個であるため MCTS の木を上手く構築することができたが、後者では序盤は合法手の数が約 60 個あり、終局状態までの最大ステップ数も大きいためプレイアウト回数が少なくなってしまい十分に木を構築できなかったと考えられる。

4. プレイアウトの効率化手法の検討

4.1 N-Gram selection Technique (NST)

少ないプレイアウト回数でも MCTS を十分に機能させるためにはシミュレーション戦略を工夫する必要がある。効率的なプレイアウトを行う方法として、プレイアウトで選択した手およびその平均報酬を記録し、それを用いた手法が考えられてきた。N-Gram selection Technique (NST) [2]では、平均報酬が記録された長さ 1,2,3 の連続した手のシーケンスを用いてプレイアウトの推測を行う。これらのシーケンスは MCTS で終局状態に到達した後、シミュレーションで出現した長さ 1, 2, 3 のすべての手の組合せを抽出することで形成される。

プレイアウトの過程では、どの手を選択するかを決定するために N-Gram が使用される。ノード N から次の手 n を選択する際に、長さ L=1,2,3 のシーケンスを決定する。このとき L＝１のシーケンスは、n 単体、L＝２のシーケンスは、n と１個前のノードである N、L＝３のシーケンスは、n と N と n の２個前のノードとなる。それぞれのシーケンスには平均報酬が存在し、三つのシーケンスの平均報酬の非加重平均を用いて n のスコアを計算する。全ての合法手に対してこのスコアを計算し、これらのスコアを ϵ-greedy 法に適用してどの手を選択するか決定する。この手法は本研究に適用できるはずなので適用を試みていく。

5. おわりに

本研究では GGP ライブラリを使用して GDL で表現されたゲームの定義を読み込み、読み込んだ情報を用いて MCTS で動作するプレイヤを作成した。プレイアウトを同じ回数に固定して GGP ライブラリの原始モンテカルロプレイヤと対戦させた結果、構築した MCTS プレイヤの方が強いという結果になった。このことから作成した MCTS を用いた GGP プレイヤが正しく構築できたことがわかった。また、ゲームによっては十分に木を作れなかったが、これはシミュレーション戦略がないからである。

NST を用いたシミュレーション戦略では、平均報酬をシーケンスとした N-gram を用いてプレイアウト時の手の選択を行っていた。そこで GDL で表現された盤面情報を要素として、平均報酬を含んだ二つの連続した盤面情報の組合せをシーケンスとする bi-gram をシミュレーション戦略の手法として検討していきたい。

参考文献

[1] N. Love, T. Hinrichs, D. Haley, E. Schkufza, and M. Genesereth.“General game playing: Game description language specification”, 4, March 2008.

[2] Mandy J.W. Tak, Mark H.M. Winands, Member, IEEE, and Yngvi Bjornsson “Decaying Simulation Strategies”IEEE Transactions on Computational Intelligence and AI in Games, vol. 6, pp. 395-406, 11March 2014.