ターン制戦略ゲームに対する多段化探索木とその工夫の効果

提橋凛[1,a)]　坪倉弘治[2]　西野順二[3]

概要：複数着手性のあるターン制戦略ゲームにおける多段化探索木の効果を研究用プラットフォームである TUBSTAP を対象に検討した. ターン制戦略ゲームは行動の組合せ爆発によって 1 ターンの分岐因子が膨大になってしまうことから, 1 ターンの行動の組合せを 1 つのノードとする木で探索を行うことは難しい. 本研究では,1 つの行動を 1 つのノードに対応させた木で UCT 探索を行う手法が成果を挙げていることに着目し, 行動とノードの対応関係を変化させることによる性能の変化を検証する. 行動を「ユニットの選択」と「ユニットの行動の決定」に分割しそれぞれを 1 つのノードに対応させた木で UCT 探索を行う手法を提案し, 対戦実験を行った. 結果として, 提案手法は既存手法に対して約 50%の勝率であった.

Effect of Multistage Search Tree in Turn-Based Strategy Games

RIN SAGEHASHI[1,a)]　KOJI TSUBOKURA[2]　JUNJI NISHINO[3]

Abstract: Turn-based strategy games (TSG) are popular in the world. However, AI players for TSG is not enough strong to satisfy human players without handicap. M-UCT is a strong AI player using the Upper Confidence Tree Search (UCT) with the multistage search tree. This study aims to confirm the relationship between the number of the UCT stages and performance. We proposed the UCT with the search tree with twice the stage of M-UCT. This player competed against M-UCT with about 50 % winning percentage.

1. はじめに

近年, ゲームにおける「強いコンピュータプレイヤの作成」の研究の進歩は目覚ましく, チェスや将棋, 囲碁などの古典的ターン制戦略ゲームにおいてはコンピュータプレイヤが人間のトッププレイヤに勝利するといった成果が挙げられている. これらのゲームのコンピュータプレイヤは人間プレイヤが競争や上達を目的に使用することに耐えうる水準に達している.

一方で, より複雑なルールを持つゲームにはコンピュータプレイヤが十分な強さを持たないものも多く, 1 ターンに複数回の行動を行える性質（複数着手性）を持つターン制戦略ゲームもその一つである. 行動の組合せ爆発により 1 ターンの分岐因子が膨大になってしまうことや初期局面の多様さにより局面評価が難しいことなどが課題となっている. これらの課題に対するアプローチとして, 図 1 のように 1 ターンに複数回行える行動をそれぞれ 1 ノードに対応させ, 1 ターンを多段化して木で UCT 探索を行う手法 M-UCT [1] が提案されており, 成果をあげている [2]. しかし,M-UCT が有効な理由については未だ明らかになっていない.

本研究は既存手法 M-UCT の多段木に着目し, 行動とノードの対応関係を変化させることによる性能の影響について検証する.

本研究では対象ゲームとして, ターン制戦略ゲーム学術用基盤プロジェクトで開発・配布されている TUBSTAP を用いる [2], [3]. 使用するバージョンは 0108 である. TUBSTAP はユニットの生産などの内政要素の無いターン制戦略ゲームであり, 2 人零和有限確定完全情報ゲームである.

[1] 電気通信大学 情報理工学研究科
Graduate School of Informatics and Engineering, The University of Electro-Communications
〒 182–8585 東京都調布市調布ヶ丘 1 丁目 5–1
[2] 電気通信大学 情報理工学部
Faculty of Informatics and Engineering, The University of Electro-Communications
[3] 電気通信大学
The University of Electro-Communications
[a)] s1731075@edu.cc.uec.ac.jp

© 2018 Information Processing Society of Japan

TUBSTAP の盤面は長方形のマスで構成され, それぞれのマスに 6 種類の地形の内 1 つが割り当てられる. ユニットの種類と地形の種類の組み合わせは, ユニット同士の戦闘やユニットの移動に影響する.

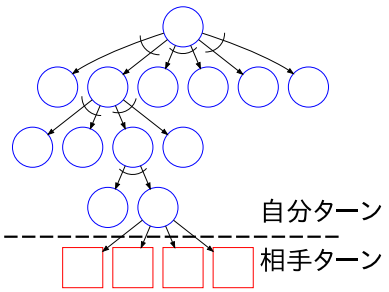

図 1　M-UCT の探索木

2. 関連研究

ターン制戦略ゲームに関連するゲームとして Arimaa [4] が挙げられる. Arimaa はチェスの盤と駒を用いる初期局面の多様さや複数着手性も持つゲームであり, ターン制戦略ゲームと同様に 1 ターンの分岐因子の大きさが AI 設計時の課題となりやすいが, 行える着手の回数が常に 1 回から 4 回までであることや駒の除去にトラップという特殊のマスを用いることなど決定的な違いを持つ.

TUBSTAP においては, 2018 年 3 月に開催された 3rd Game AI Tournament 2018 内で開催された GAT2018 大会で, M-UCT に対してサブゴールを加えた Subgoal M-UCT や M-UCT にヒューリスティックな枝刈りを加えた A-UCT が出場しており, A-UCT は各部門で 2 位といった顕著な成績を挙げている [2]. 表 1 に公開されている GAT2018 出場 AI をそれぞれ 100 回対戦させた結果を示す. ただし,GAT2018 大会での優勝 AI である ALPHABETA は公開されていないため表 1 には含まれていない.

表 1　AI1 の AI2 に対する勝数 (GAT2018 大会出場 AI)

AI1 \ AI2	AU	SM	SR	MH
A-UCT	-	64	74	92
Subgoal M-UCT	33	-	65	76
SatRandMM	19	28	-	61
M-UCT_HIMA	5	16	21	-

3. 提案手法

本研究では,M-UCT を以下のように変更する. TUBSTAP の着手は「行動させるユニットの決定」と「ユニットの行動の決定」の 2 段階の意思決定に展開できることを利用し, 意思決定のそれぞれをノードに対応させた木の上で UCT 探索を行う手法 S-UCT を提案する. 図 2 に提案手法 S-UCT の探索木を示す. 自分ターンにおける小さなノードは「ユニットの決定」に対応したノード, 大きなノードは「行動の決定」に対応したノードである. 既存手法 M-UCT の探索木である図 1 の深さ 1 のノードが図 2 の深さが偶数のノードはその 2 分の 1 の深さの図 1 のノードに対応している. そのため, 提案手法は既存手法 M-UCT と比較すると木の高さが 2 倍となる.

本研究では既存手法 M-UCT を実装した AI の木部分を改変することで提案手法 S-UCT の実装を行った. M-UCT では 1 ターンの思考に制限時間を設けており, 1 つの着手の探索に用いる時間は (制限時間)/(行う着手の数) である. S-UCT では着手の探索に用いる時間を「ユニットの決定」は R_{UNIT},「行動の決定」は $1-R_{\mathrm{UNIT}}$ の比率で分割した. ただし,$0 < R_{\mathrm{UNIT}} < 1$ である. 探索は M-UCT と同様の UCT 探索を行い, あるノードの UCB 値は係数 c を用いて

$$(\text{ノードの勝率}) + c\sqrt{\frac{\ln(\text{全ノードの訪問回数})}{(\text{ノードの訪問回数})}}$$

である. 係数 c プレイアウトは攻撃行動寄りのランダムシミュレーションを行い, 終端局面で勝利の場合は 1, 敗北の場合は 0, 引分の場合は 0.5 を報酬として返す. あるノードの勝率はそのノード自身と子孫ノードで行なわれたプレイアウトの報酬の平均値である. 子ノードの展開は 1 段ずつ行い, S-UCT においては「ユニットの決定」ノードの子ノードは「行動の決定」ノード,「行動の決定」ノードの子ノードは「ユニットの決定」ノードである. 子ノード展開のしきい値はノードがどちらに対応しているかに関わらず一定である.

提案手法 S-UCT のアルゴリズムをアルゴリズム 1 に示す.

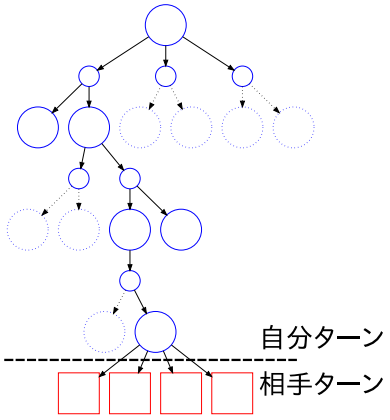

図 2　S-UCT の探索木

4. 実験

提案手法を評価するために S-UCT とベースプレイヤとして用いた M-UCT 及び TUBSTAP サンプルプレイヤの Sample_MaxActionEvalFunc (SMAEF) [3] の対戦実験を

© 2018 Information Processing Society of Japan

アルゴリズム 1 S-UCT アルゴリズム

R_{UNIT}: ユニット決定に掛ける時間のパーセンテージ
T_{ELAPSED}: ターン開始からの経過時間
T_{TURN}: 1 ターンの制限時間
function S-UCT(現局面)
  $T_{\mathrm{REMAINING}} \leftarrow T_{\mathrm{TURN}} - T_{\mathrm{ELAPSED}}$
  $N_{\mathrm{UNIT}} \leftarrow$ 自分チームの行動可能ユニットの数
  $T_{\mathrm{LIMIT}} \leftarrow T_{\mathrm{ELAPSED}} + \frac{T_{\mathrm{REMAINING}}}{N_{\mathrm{UNIT}}} \times \frac{R_{\mathrm{UNIT}}}{100}$
  root node $\leftarrow$ 現局面から生成したノード
  while $T_{\mathrm{ELAPSED}} < T_{\mathrm{LIMIT}}$ **do**
    SEARCH(*root node*)
  unit node $\leftarrow$ 勝率最高の *root node* の子ノード
  $T_{\mathrm{LIMIT}} \leftarrow T_{\mathrm{ELAPSED}} + \frac{T_{\mathrm{REMAINING}}}{N_{\mathrm{UNIT}}} \times \frac{100-R_{\mathrm{UNIT}}}{100}$
  while $T_{\mathrm{ELAPSED}} < T_{\mathrm{LIMIT}}$ **do**
    SEARCH(*unit node*)
  action node $\leftarrow$ 勝率最高の *unit node* の子ノード
  action $\leftarrow$ *action node* の行動
  return *action*
function SEARCH(*node*)
  $n \leftarrow$ *UCB* が最大の *node* の子ノード
  if n の訪問回数 $>$ しきい値 **then**
    if n が葉ノード **then**
      if n がユニットノード **then**
        n の子ノードにアクションノードを展開
      else if n がアクションノード **then**
        n の子ノードにユニットノードを展開
    SEARCH(n)
  else
    プレイアウトを実行
    先祖ノードにプレイアウトを反映

行った. SMAEF は攻撃行動を優先するルールベースプレイヤである. 実験に使用したマップを図 3 に示す. このマップは TUBSTAP の配布パッケージに含まれている map01 と同一である. 実験に用いた計算機のプロセッサは Intel(R) Core(TM) i7-6700 CPU 3.40GHz で, RAM は 32.0GB, OS は Windows 10 Pro の 64 ビットである.

4.1 探索時間の割合に関する対戦実験

提案手法 S-UCT が「ユニットの決定」と「行動の決定」のために行う探索の時間を偏らせることによって性能に変化が生じるかを調べる. S-UCT は「ユニットの決定」の探索に割く時間の割合 R_{UNIT} が 0.1,0.2,0.3,0.4,0.5 の 5 つのプレイヤを用意し, M-UCT と SMAEF に対してそれぞれ 200 戦行った. S-UCT と M-UCT は 1 ターンの制限時間は 5000 ミリ秒, それぞれの UCB 値計算に用いる係数 c は 0.15, 探索木の子ノード展開のしきい値は 5 とした. また, 参考として M-UCT と SMAEF の対戦実験を 100 戦行った.

4.1.1 結果

表 2 に S-UCT の「ユニットの決定」の探索に割く時間の割合 R_{UNIT} 毎の M-UCT との対戦実験結果を示す. 表 3 に S-UCT の SMAEF との対戦結果を示す. 表 4 に M-UCT と SMAEF の対戦結果を示す.

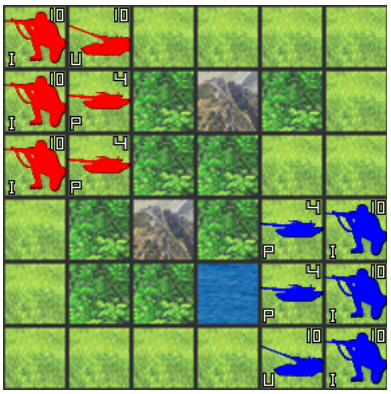

図 3 実験に使用したマップ

表 2 持ち時間 5000 ミリ秒での S-UCT と M-UCT の対戦結果

R_{UNIT}	勝ち	負け	引分
0.1	25	169	6
0.2	40	155	5
0.3	26	169	5
0.4	27	167	6
0.5	29	167	4

表 3 持ち時間 5000 ミリ秒での S-UCT と SMAEF の対戦結果

R_{UNIT}	勝ち	負け	引分
0.1	111	63	26
0.2	110	59	31
0.3	122	51	27
0.4	119	51	30
0.5	111	61	28

表 4 M-UCT と SMAEF の対戦結果

M-UCT の勝利	SMAEF の勝利	引分
91	5	4

4.2 UCB 値の係数 c 及び持ち時間に関する対戦実験

提案手法 S-UCT において UCB 値の計算の際に用いる係数 c を変化させることで, 性能の変化が見られるかを対戦実験を行うことで調べる. S-UCT は係数 c の値が異なる 6 つのプレイヤを使用し, 値はそれぞれ 0.15,0.30,0.50,0.60,0.70,0.80 である. M-UCT は係数 c の値が 0.15 のプレイヤを使用した. S-UCT と M-UCT の探索木の子ノード展開のしきい値は 5,1 ターンの持ち時間は 5000 ミリ秒とした. S-UCT の「ユニットの探索」に使用する時間の割合 R_{UNIT} は 0.5 である. 係数 c の値が異なる S-UCT のプレイヤそれぞれと M-UCT の対戦を 200 戦ずつ行った. また,S-UCT の持ち時間が 50000 ミリ秒,M-UCT の持ち時間が 5000 ミリ秒の場合についても, 係数 c の値が異なる S-UCT のプレイヤそれぞれと M-UCT の対戦を 200 戦ずつ行った.

4.2.1 結果

表 5 に 1 ターンの持ち時間が 5000 ミリ秒の S-UCT の 5000 ミリ秒の M-UCT との対戦実験の結果を示す. 表 6 に 1 ターンの持ち時間が 50000 ミリ秒の S-UCT の 5000 ミリ秒の M-UCT との対戦実験の結果を示す. 図 4 に 1 ターンの

© 2018 Information Processing Society of Japan

持ち時間が 5000 ミリ秒の S-UCT の 5000 ミリ秒の M-UCT との対戦実験における勝率を示す. 図 5 に 1 ターンの持ち時間が 5000 ミリ秒の S-UCT の 5000 ミリ秒の M-UCT との対戦実験における勝率を示す. ここで言う勝率とは引分を 0.5 勝 0.5 敗として計算したものである.

表 5 持ち時間 5000ms の S-UCT の勝率

c	勝ち	負け	引分
0.15	29	167	4
0.30	57	131	12
0.50	78	108	14
0.60	92	97	11
0.70	95	92	13
0.80	88	90	22

表 6 持ち時間 50000ms の S-UCT の勝率

c	勝ち	負け	引分
0.15	25	170	5
0.30	88	107	5
0.50	149	41	10
0.60	151	37	12
0.70	172	20	8
0.80	161	26	13

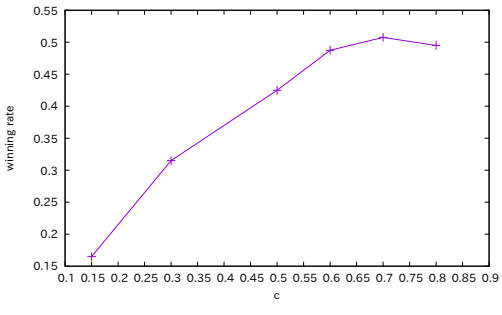

図 4 持ち時間 5000ms の S-UCT の勝率

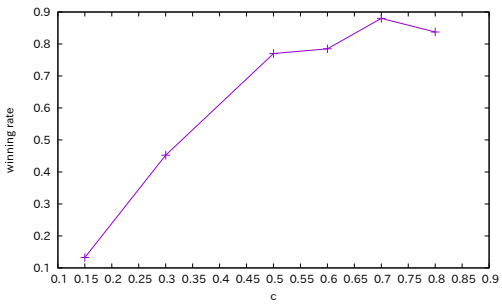

図 5 持ち時間 50000ms の S-UCT の勝率

5. 提案手法の評価

提案手法 S-UCT の UCB 値の計算に用いる係数 c が 0.15 のとき, 表 2 と表 3 より提案手法 S-UCT はルールベースプレイヤの SMAEF に対しては勝ち越すことができたが, ベースとした M-UCT に対しては大きく負け越した. また, 表 3 と表 4 より S-UCT と M-UCT の SMAEF に対する対戦実験の結果を比べると S-UCT は M-UCT に比べて勝率が低い. よって, 提案手法 S-UCT は既存手法 M-UCT に比べて勝率を向上させる効果は見られず, むしろ勝率は低下することがわかった. また, 表 2 と表 3 より R_{UNIT} 毎の結果について, 勝ちを 1 点, 負けを 0 点, 引分を 0.5 点として勝点群を作成し任意の 2 群に対して t 検定を行ったが有意差無し ($p > 0.05$) であった. よって, 提案手法において UCB 値計算に用いる係数 c が 0.15 のとき, $R_{\mathrm{UNIT}} \in \{0.1, 0.2, 0.3, 0.4, 0.5\}$ では「ユニットの決定」と「行動の決定」の時間分配を偏らせることによる性能の変化は見られなかった.

図 4 と図 5 より提案手法 S-UCT は UCB 値計算に用いる係数 c を変化させることで性能に変化が見られた. UCB 値の計算に用いる係数 c が 0.15 のときは,S-UCT の 1 ターンの持ち時間が 5000 ミリ秒, 50000 ミリ秒の双方の場合で M-UCT に大きく負け越した. 係数 c の値が 0.30 以上のとき, 持ち時間が 5000 ミリ秒のプレイヤに比べて持ち時間が 50000 ミリ秒のプレイヤの方が M-UCT に対する勝率は高い. 持ち時間が 5000 ミリ秒の場合, 係数 c の値が 0.60,0.70,0.80 のときは S-UCT と M-UCT の性能に有意な差は見られなかった. 持ち時間が 50000 ミリ秒の場合, 係数 c の値が 0.60,0.70,0.80 のときは S-UCT は M-UCT に比べて有意に勝率が高かった. よって, 提案手法 S-UCT は UCB 値の計算に用いる係数 c を変化させることで性能が変化することや 1 ターンの持ち時間を長くすることで性能が高くなることがわかった.

5.1 考察

提案手法 S-UCT は UCB 値の計算に用いる係数 c が 0.15 の際には既存手法 M-UCT に対して大きく負け越したが, 係数 c が 0.60,0.70,0.80 の際には同じ持ち時間で M-UCT と同程度の勝率を得られた. このことから, 提案手法では期待報酬が低いノードの子孫ノードに有望なノードがある場合が多く, c の値が低いと有望なノードを訪問することが出来ず, 性能が低くなっていると考えられる.

6. おわりに

ターン制戦略ゲームの研究基盤プロジェクトである TUBSTAP において, 着手を 2 段階の決定に分け, それぞれを 1 つのノードに対応させた木で UCT 探索を行う手法を提案した. 1 つの行動を 1 つのノードに対応させた木で UCT 探索を行う既存手法と提案手法の対戦実験を行った. UCB 値

© 2018 Information Processing Society of Japan

の計算の際に用いる係数 c を適切に調整することで S-UCT の性能は変化し,0.75 付近では今回の実験の回数では有意とは言えないものの M-UCT と同程度の勝率になった. この結果から, 提案手法は適切に調整することで既存手法と同程度の性能になることがわかった.

今後も多段化探索木での探索について検証し, 例えば子ノード展開のしきい値を変更した場合の性能への影響などを探求していきたい.

参考文献

[1] Fuzzy Evaluation of Macroscopic Situation for Turn Based Strategic Games, Kosuke Muto, Junji Nishino, 2016 Joint 8th International Conference on Soft Computing and Intelligent Systems (SCIS) and 17th International Symposium on Advanced Intelligent Systems (ISIS), 2016 .

[2] ターン制戦略ゲーム学術用基盤プロジェクト, http://www.jaist.ac.jp/is/labs/ikeda-lab/tbs/ (2018 年 7 月 31 日アクセス)

[3] 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, 村山公志朗, 藤木翼, 池田心, 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 .

[4] Arimaa-a new game designed to be difficult for computers, Omar Syed, Aamir Syed, ICGA Journal, pp.138–139, 2003 .

© 2018 Information Processing Society of Japan