# ターン制ストラテジーのための状態評価型深さ限定モンテカルロ法

藤木翼　村山公志朗　池田心
北陸先端科学技術大学院大学
{s1310062,kosiro_murayama,kokolo}@jaist.ac.jp

将棋などと異なり，ターン制ストラテジーでは全ての駒を任意の順序で動かすことができ，合法手数が膨大になることが探索の障害となる．本稿では探索を合法手の一部のみに限定し，かつ状態評価関数を用いて深さを限定したモンテカルロ法が有効であることを示す．

# Depth-limited Monte-Carlo with a State Evaluation Function for Turn-based Strategy Games

Tsubasa FUJIKI, Koshiro MURAYAMA, Kokolo IKEDA
Japan Advanced Institute of Science and Technology

In turn-based strategy games, all the pieces can be moved in any order in one turn. Then it is efficient to limit the search to a subset of the resulting high number of legal moves and to use a depth-limited Monte Carlo search combined with a state evaluation function.

## 1　はじめに

これまで，チェスや将棋などの古典的なゲームのコンピュータプレイヤに関する研究は幅広く行われてきた．しかし，「一回の手番に複数の駒を動かせる」タイプのターン制ストラテジーと呼ばれるゲームは広く遊ばれているにも関わらず，新しいゲームであることや，探索空間の大きさ，ルールの煩雑さなどもあって，学術的研究は少数が行われているのみである．

ターン制ストラテジーの既存ゲームとしては Battle of Wesnoth，Final Fantasy Tactics，ファミコンウォーズなどが挙げられるが，コンピュータプレイヤは人間の中級者とハンデなしで戦える強さにはほど遠い．ゲーム内ではハンデでその弱さを補っているが，対等に戦いたいという要求には応えられていない．

そこで本稿では，自然さや楽しさといったものの前にまず“強い”コンピュータプレイヤ（AI と略す）を作成することを目的とする．

対象とするゲームは研究用ターン制ストラテジー「TUBSTAP」とし，着手を限定した原始モンテカルロ法と，状態評価関数を用いて深さまで限定するモンテカルロ法を紹介する．これらを対戦実験により性能評価する．

## 2　関連研究

### 2.1　TUBSTAP のルール

既存ターン制ストラテジーの要素は内政要素やキャラクタの成長要素など数多い[1]. TUBSTAP はその中でも重要な要素である複数着手性に注目し，不完全情報性や占領・生産などの要素を排除した二人零和有限確定完全情報ゲームである．将棋などと比べると複雑ではあるが，既存のターン制ストラテジーに比べると非常にシンプルなゲームとなっている．以下に重要なルールを抜粋する．尚，紹介するこれらの要素は既存のターン制ストラテジーであれば殆どが持っている共通の要素である．

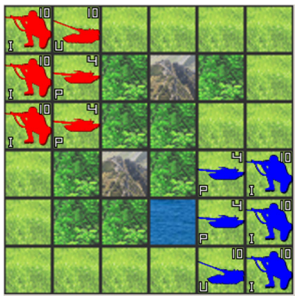

図 1. TUBSTAP のプレイ画面

#### R1. 駒

戦闘機(F), 攻撃機(A), 戦車(T), 対空戦車(R), 歩兵(I)，自走砲(U)の 6 種類の駒を用いる．これらの駒間には相性が存在する．相性は後述する HP に与えるダメージ量に影響する．

#### R2. マップ

将棋等では常に同じ盤面サイズ・駒の初期配置を用いる．しかし，ターン制ストラテジーでは通常，プレイ毎に異なる設定の駒配置や盤面を用い対戦を行う．マスには複数の種類（地形）があり，被るダメージが増減する．

R3. 着手順

各手番では，プレイヤは全ての自駒を 1 回ずつ自由な順番で行動させることが出来る．すべての駒を行動させると相手の手番となる．両者それぞれの手番をターンと呼ぶ．

R4. 勝利条件

いずれかのチームの駒が全滅した場合，駒が盤面に存在しているチームが勝利となる．ただし，ターン数に上限を設け，その上限以内に全滅条件を満たさない場合の判定条件はチーム毎の総 HP 量の差で決定される．

R5. HP

各駒は 1 から 10 の HP を持つ．攻撃を受けることで HP は減少し，0 になるとその駒は盤上から取り除かれる．

### 2.2 TUBSTAP の特徴

1 ターン内における複数着手性により，ターン制ストラテジーではプレイヤがとれる行動の組み合わせ（合法手）が非常に多い．例えば 1 駒あたりの平均合法手が 10, 駒数が 6 とすると，1 ターン内に取れる行動の数は 7 億 2000 万通り($= 10^6 \times 6!$)にも及ぶ（同一局面に遷移するものも含む）．実際にはより駒数や 1 駒あたりの合法手数が多い場合も珍しくない．

これらの特徴から，ターン制ストラテジーでは Mini-Max 探索などの従来型の探索手法の適用が困難であり，過去に提案された手法では何らかの探索量削減が図られてきた．

### 2.3 着手順の固定

村山らは着手順に注目した手法を提案した[1]．これは全ての駒の行動順を考慮するのではなく，攻撃可能な駒から，かつ戦果の大きい順に動かそうとすることで，計算量を抑えようとする手法である．攻撃行動の組み合わせを考慮したほうが性能を向上させることが報告されている一方で，全ての攻撃行動の組合せを考えることは計算量の爆発を起こすとされている．

また，TUBSTAP の標準 AI では行動の組み合わせを考えず，単純に全ての攻撃行動の中で評価関数値が最も高いものを選択するというアルゴリズムが実装されている．

### 2.4 駒の行動限定

加藤らは着手順ではなく駒ごとの合法手，特に移動行動を限定する手法を提案している[2]．これは，この種のゲームでは駒の移動可能範囲が非常に広く，かつそれを全て考慮することが無駄な場合が多いためである．

実際には，敵駒を攻撃する行動と，次ターンに敵を攻撃できる最も遠いマスへの移動のみに着手を限定し，これらの限られた有望な戦略を見つけ出す事に特化して探索を行う．加藤らはこの手法を UCT で実装することで単純な UCT に対して勝ち越す結果を得ている．

## 3 着手限定原始モンテカルロ

### 3.1 モンテカルロ木の生成法

ターン制ストラテジーにおける木探索の実装には加藤らによると大きく分けて 2 通りが存在する[2]．1 つは，個々の駒の行動を枝として 1 ターン内の行動を数段に分けた探索木を作る方法，もう 1 つは 1 ターン内に可能な行動の全ての組み合わせを 1 つの枝として探索木を作成する方法である．本稿では後者を採用しており，加藤ら[2]，村山ら[1]は前者を採用している．

本稿で後者を採用した理由は，TUBSTAP では両軍が接触するタイミングでの陣形が重要となり，1 ターン内全ての駒の行動が終了した盤面の駒の“位置取り”が最も重要な評価対象であると考える為である．しかし，行動の組み合わせ全てを評価する場合，枝数が非常に多くなる．そこで枝数削減のためにランダムサンプリングを用いた前向き枝刈りによる着手限定を採用することにした．

### 3.2 手法説明

まず以下に，本稿で使用する記号を示す．

s：盤面の状態．
S：全ての盤面の状態 s の集合．
$a$：1 ターン内の全ての駒の行動の組み合わせ．
$\mathrm{A}(s)$：状態sにおける全ての行動$a$の集合．
$\mathrm{s}'(\mathrm{s},a)$：状態sで行動$a$を取った場合の次状態．

【手法 1: 着手限定モンテカルロ】

I. 行動$a$をランダムにm個サンプルする．
$$\mathrm{A}'(\mathrm{s}) = \{a_1, a_2, \ldots, a_m\} \subset \mathrm{A}(\mathrm{s})$$

II. $\mathrm{A}'(\mathrm{s})$に含まれる行動$a$による次局面$\mathrm{s}'(\mathrm{s},a)$からn回終局までシミュレーションし，勝率を測定する．ここで$f_i(a)$は i 回目のシミュレーション結果を表す．
$$f_i(a): S \rightarrow \{0, 0.5, 1\}$$
$$f_n(a) = \sum_{i=1}^{n} f_i(a)$$

III. $A'(s)$の中で勝率最大の行動$a^*$を選択する．

$$a^* = \underset{a \in A'(s)}{\operatorname{argmax}} f_n(a)$$

ランダムサンプリングは合法手の枝刈りにおいて大きな役割を果たしており，良い手の見落としのリスクがある一方で，計算量の爆発を防ぐためには必要な措置である．本手法では完全にランダムなサンプリングという手法をとっているが，良さそうな行動のみにある程度限定した上でのサンプリングなども考えられる．

### 3.3 シミュレーションのヒューリスティック

TUBSTAP における駒の行動は大きく分けて単純に移動するだけの行動と攻撃を含む行動の 2 通りがある．1 つの駒が持つ行動のうち攻撃する行動は移動するだけの行動に比べると平均的には非常に少ない．そのため，完全にランダムなシミュレーションを行うと指定ターン内に勝敗が決せず，引き分けで終わることが多々発生する．これはチェス等でも指摘される点である．では攻撃を優先してシミュレーションを行えば良いかというと，今度は相手の安易な攻撃を期待しての待ちの行動が多くなってしまう．

そこで本稿では攻撃可能な駒が存在する状況ならば必ず攻撃を行う方策と完全にランダムな行動を行う方策の 2 つを混合して用いている（前者が 8 割，後者が 2 割）．

本稿で使用している攻撃優先のシミュレーションは駒間の相性を考えることなく，攻撃可能であれば攻撃を行うといった非常に単純なものを使用しているが，相性を考えるシミュレーションを用いることで性能の向上が見込める．

## 4 深さ限定モンテカルロ

原始モンテカルロ法では終局までのシミュレーションを行うが，シミュレーションを途中で打ち切り，その局面を状態評価関数で評価し，この数値をシミュレーション結果の代わりとすることも行われる（本稿では深さ限定モンテカルロと呼ぶ）．原始モンテカルロ法に比べると 1 回のシミュレーションにかかる計算コストが少なく，非常に高速な動作が可能となる．本手法はボードゲーム Amazon において優れた結果を示している[3][4].

### 4.1 手法説明

【手法 2: 深さ限定モンテカルロ】

I. 行動$a$をランダムにm個サンプルする．

$$A'(s) = \{a_1, a_2, \ldots, a_m\} \subset A(s)$$

II. $A'(s)$に含まれる$a$による次局面$s'(s,a)$を d ターン先までシミュレーションし，**状態**評価関数$g(s)$により評価を行う．これをn回繰り返し，評価値の合計を測定する．

$h_i(s): S \to S$　i 回目のシミュレーション．d ターン進める．

$g(s): S \to R$　評価する

$g_n(a) = \sum_1^n g(hi(s'(s,a)))$

III. $A'(s)$の中で評価最大の行動$a^*$を選択する．

$$a^* = \underset{a \in A'(s)}{\operatorname{argmax}} g_n(a)$$

図 2 に概念図を示す．II において次局面$s'(s,a)$から d ターン先と説明しているが，これはルートノードから(d+1)ターン先を意味している．

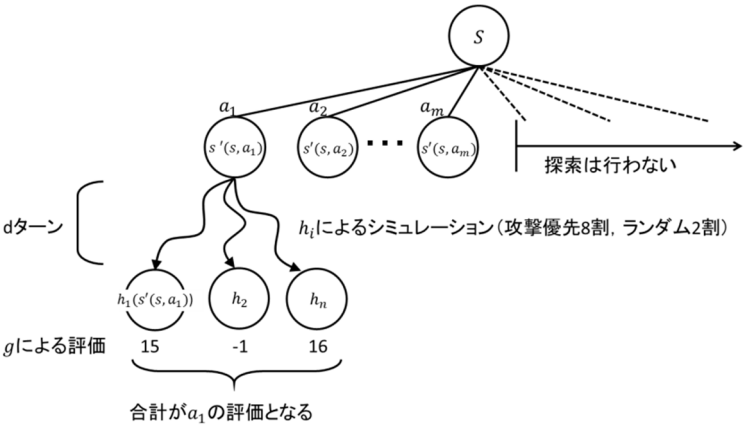


図 2. 手法 2 の概念図

### 4.2 状態評価関数

打ち切った後に使用する状態評価関数$g(s)$は重要な要素ではあるが，1 回のシミュレーション毎に使用するため，ある程度低計算量でなければいけない．そこで本稿ではシンプルな状態評価関数を用いている．

$$h(s) = (M - E) \times B$$

$$B = \begin{cases} 2, & \text{if } M = 0 \text{ or } E = 0 \\ 1, & \text{else} \end{cases}$$

M:自チームの駒の HP の総和（駒 I は 0.2 倍）
E:敵チームの駒の HP の総和（駒 I は 0.2 倍）
B:ボーナス係数

ここで B は勝敗が決した際に，対戦中の場合と区別し，評価値を倍にするための係数である．勝利であれば自チームの方が敵チームよりも HP の総和が大きいためより良い評価となり，敗北であれば敵チームの方が自チームよりも HP の総和が大きいためより悪い評価となる．

また，駒 I の HP を 0.2 倍して使用しているのは，I の攻撃力が非常に弱く，活躍があまり期待できないためである．その他の駒に重みはかけられていない．ここにも改善の余地は残されている．

## 5　対戦実験

対戦実験として TUBSTAP に標準で搭載されている AI と，本稿で紹介した着手限定モンテカルロ（手法 1），深さ限定モンテカルロ（手法 2）をそれぞれ 100 回ずつ戦わせた．なお，TUBSTAP に標準で搭載されている AI は 2.3 節で紹介した手法を用いているものである．

【対戦条件】

- 図 1 に示すマップを使用し，先手後手を各 50 回，計 100 回の対戦を行う．小さなマップではあるが初手による次局面は駒の同一性など重複を考慮しても概算で約 160 万通りに及ぶ．
- 対戦時に使用する上限ターン数は 30 とした．このターン数を越した場合はチーム間の駒の総 HP 量の差によって勝敗が決まり，10 以上であった場合，総 HP 量の多いチームの勝利とする．それ以外であれば引き分けとして扱う．
- 手法 1，2 のシミュレーション数 n，サンプル数 m はそれぞれ 100 とした．この場合手法 2 のほうが 10 倍程度高速である．

【対戦結果】

結果を表 1 に示す．尚，表内の数値は左から win-draw-loss となっており，括弧内の数値は引き分けを 0.5 とした場合の winrate を表している．

| AI 名 | vs 手法 1 | vs 手法 2 | vs 標準 AI |
|---|---|---|---|
| 手法 1 | － | 35-36-29 (49.5) | 55-31-14 (70.5) |
| 手法 2 | 36-35-29 (50.5) | － | 62-23-15 (73.5) |
| 標準 AI | 14-31-55 (29.5) | 15-23-62 (26.5) | － |

表 1. 対戦結果（勝数，敗数，引分数，勝率）

【考察】

手法 1,2 はどちらも標準 AI に勝ち越す結果となった．手法 1 と手法 2 の差は僅差であるが，手法 2 が 10 倍ほど高速であることを考えれば，シミュレーション数 n，サンプル数 m を多くすることで手法 1 よりも強くなると予想される．

## 6　まとめと今後の予定

本稿では TUBSTAP における原始モンテカルロの適用法と状態評価関数を用いた深さ限定モンテカルロによる手法を紹介したうえで，これらと TUBSTAP に搭載されている標準の AI の対戦実験を行った．これらの結果として深さ限定モンテカルロが他の 2 つの AI より優れていることを示した．

今後は，シミュレーション回数 n とサンプル数 m を変化させての対戦実験や，総シミュレーション回数を一定とした場合におけるパラメータ n, m の最適値などを調べたいと考えている．

参考文献


[1] 村山，藤木，池田，“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”，IPSJ-GPW 2013-11-01，pp.146-153

[2] 加藤，三輪，鶴岡，“ターン制ストラテジーゲームにおける戦術決定のための UCT 探索とその効率化”，IPSJ-GPW 2013-11-01，pp.138-145

[3] Kloetzer, J., Iida, H., Bouzy, B.，“The Monte-Carlo Approach in Amazons” Computer Games Workshop,2007

[4] Kloetzer, J.，“Monte-Carlo Techniques:Applications to the Game of the Amazons”, Japan Advanced Institute of Science and Technology, 2010，博情第 240 号 Includes bibliographical references pp. 87-92

[5] ターン制戦略ゲーム 学術用基盤プロジェクト TUBSTAP, http://www.jaist.ac.jp/is/labs/ikeda-lab/tbs