The 32nd Annual Conference of the Japanese Society for Artificial Intelligence, 2018

格闘ゲームにおけるモンテカルロ木探索の探索時間と即応性のトレードオフ

Trade-off between search time and responsiveness of MCTS in a fighting video game

亀垣 航 Wataru Kamegaki　森山 甲一 Koichi Moriyama　武藤 敦子 Atsuko Mutoh　犬塚 信博 Nobuhiro Inuzuka

名古屋工業大学
Nagoya Institute of Technology

In real-time strategy games, it is difficult for a computer to defeat a human player under the condition that the computer has same recognition ability to the player. It is because many factors in the environment are changing in a very short time during which the computer has to choose an action. Although the Monte Carlo Tree Search (MCTS) algorithm obtains decent results in such games by searching a good action from simulation, it will be better if it has more time. On the other hand, it may be worse due to delay of response to the environment. In this work, we investigated the trade-off property in a fighting video game.

1. はじめに

近年、人工知能 (AI) にコンピュータゲームをプレイさせる試みが盛んに行われている。その中でも囲碁や将棋などいったプレイヤーの行動順が明確に決まっているターン制戦略ゲームの分野では研究が大きく発展している。例を挙げると、Google DeepMind の「AlphaGo」[1] という囲碁の対局プログラムが人間のプロプレイヤーに勝利するといった報告がされている。

一方、格闘ゲームやシューティングゲームといった行動順に規定の無いリアルタイム戦略ゲームの分野では、人間の能力に合わせてコンピュータの認識能力に制限をかけた条件下では、人間のプレイヤーに勝利することは困難である [2]。人間とコンピュータが平等な条件下においてコンピュータが人間に勝利することが困難なのは、常に多くのパラメータがごくわずかな時間で変動し続けるため、限られた時間内で膨大な量の情報を処理して行動を選択することが求められるからである。

本研究では、リアルタイム戦略ゲームの中でも格闘ゲームを対象とし、人間を超えない程度の認識能力になるよう制限をかけた条件下において、コンピュータが対戦を行う際の行動選択能力の向上を目的とする。

格闘ゲーム AI の競技会では、モンテカルロ木探索を用いた AI が優秀な成績を収めている。しかし、モンテカルロ木探索でも計算時間が不足し、行動選択のための探索が十分に行われていない [8]。そのため、本研究では応答時間を延長させることによって、モンテカルロ木探索の探索時間増加による行動選択能力の向上と、ゲーム環境の変化への対応能力である即応性の兼ね合いを観察する。題材として FightingICE [3] と呼ばれる研究用の格闘ゲームを使用した。

本論文は全 6 章で構成されており、第 2 章では格闘ゲームと今回使用する FightingICE、第 3 章ではモンテカルロ木探索と既存研究について紹介する。第 4 章では提案手法、第 5 章で実験の内容と結果について記述し、最後の第 6 章でまとめと今後の課題について述べる。

連絡先: 亀垣 航，名古屋工業大学工学部情報工学科
名古屋市昭和区御器所町，
w.kamegaki.653@nitech.jp

2. 格闘ゲーム

2.1 格闘ゲーム

格闘ゲームとは、プレイヤーがキャラクターを操作して殴る蹴るといった格闘技や武器などを駆使して、対戦相手を倒すことを目的としたリアルタイム戦略ゲームの一種である。基本的には人間同士又は人間対コンピュータで対戦を行う。

格闘ゲームでは先に対戦相手の操作するキャラクターの体力を 0 にすることで対戦相手を倒したプレイヤーの勝利となり、制限時間内に終わらなかった場合は制限時間が 0 になった時点でのキャラクターが受けたダメージ量で判定を行い、受けたダメージの少ないプレイヤーの勝利となる。

コンピュータが格闘ゲームにおいて対戦を行う場合、多くの場合、予め作成された行動規則に従って動くので行動パターンが単調になりやすく、熟練した人間相手では何戦か対戦することで行動を簡単に予測されてしまうため、勝利するのは極めて困難である。

2.2 FightingICE

2.2.1 概要

FightingICE [3] とは、研究目的で開発された 2 次元格闘ゲームである。ゲーム内のキャラクターを操作するプログラムを Java 言語で作成可能である。なお本稿では以後このプログラムのことをエージェントと称する。

1 試合は 3 ラウンド、1 ラウンドの制限時間は 60 秒で構成されている。1 秒を 60 分割したものをフレームと呼び、フレームごとにその時の各キャラクターの情報が記録され、エージェントはその情報を取得することができる。

キャラクターの情報としては以下の表 1 の「HP」,「Energy」,「X,Y」,「Speed」,「State」,「Motion」がある。

そして、対戦結果の評価指標として対戦スコア Score がある。対戦スコアはラウンドごとに相手から受けたダメージ量である HP によって (1) 式によって算出される。

$$Score = \frac{oppHP}{oppHP + myHP} \times 1000 \tag{1}$$

Score はラウンド内でどちらが多くのダメージを受けたか判断するものであり、*Score* が 500 の場合引き分けとなる。

表 1: FightingICE におけるキャラクターデータ

情報	データ型	詳細
HP	int	対戦相手からどれだけダメージを与えられたかを表す。
Energy	int	キャラクターが所持するエネルギーの合計量を表す。エネルギーはキャラクターの一部の技を使用する際に必要なポイントであり、技ごとに必要なエネルギーは異なり、使用するとその分だけエネルギーは減少する。
X,Y	int	キャラクターの位置座標を表す。ゲーム画面の左上を $(X, Y) = (0, 0)$ として水平方向を X 方向、垂直方向を Y 方向とする。
Speed	int	キャラクターの移動量を表す。水平方向を X 方向、垂直方向を Y 方向とする。この値を現在の座標に加算することで次のフレームでの座標が算出される。
State	String	キャラクターの現在の状態を表す。状態は 4 種類あり、STAND(直立), CROUCH(しゃがみ), AIR(空中), DOWN(転倒) となっている。
Motion	String	キャラクターが行っている行動を表す。行動は大まかに Base(基本状態), Move(移動行動), Guard(防御行動), Recovery(復帰行動), Action(攻撃行動) の 5 つに分類でき、全 55 種類の行動がある。

2.2.2 認識能力の制限

FightingICE では、エージェントがキャラクターの位置や行動などのゲーム情報を取得する際は、15 フレーム (約 0.25 秒) 以前の情報しか取得することができない。これは人間が視覚から情報を取得して反応するまでにかかる時間が約 0.2 秒 [5] であるので、コンピュータの認識能力を人間の認識能力の限界にあわせるためである。このルールが無い場合、エージェントは 1 フレーム (約 0.017 秒) 前の状況を見て判断することができてしまうため、人間側が不利になってしまう。

2.2.3 行動入力

FightingICE では、ゲーム環境が 1 フレームごとに更新されていくため、エージェントは行動を 1 フレーム以内に入力しなければならない。行動が入力されなかった場合、エージェントが最後に入力した行動が再度入力されたものとして扱う。

3. モンテカルロ木探索

3.1 概要

モンテカルロ木探索 [6] とは、評価関数の代わりに確率的シミュレーションを使用する探索手法の 1 つであり、評価関数の作成が困難な問題に対して有効な探索アルゴリズムである。

探索は図 1 のような木構造を作成していくことで行っていく。ゲームの状況を根ノードに渡し、実行可能な行動を 1 つずつ子ノードに分けて木を作成する。そして有望なノードに子ノードを展開して木を成長させることで深く探索していく。ゲーム終了までランダムに行動を行うシミュレーションの勝率または評価値を求めた結果をプレイアウトと称する。

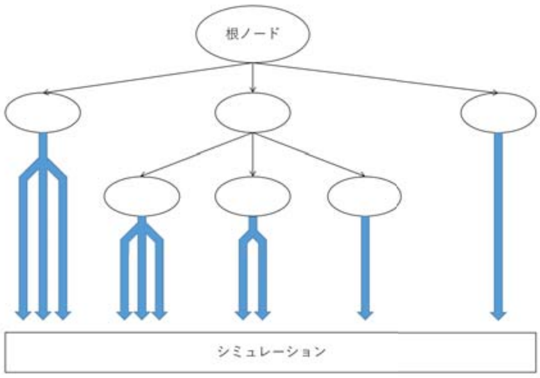

図 1: モンテカルロ木探索の例

有望なノードの指標として $UCB1$ があり、これは以下の (2) 式によってノードごとに算出される。$\overline{X}$ はそのノードのプレイアウトの平均、C は定数、N^p はそのノードの親ノードが選択された回数、N はそのノードが選択された回数である。

$$UCB1 = \overline{X} + C\sqrt{\frac{2 \log N^p}{N}} \tag{2}$$

3.2 手順

モンテカルロ木探索では、以下の 1〜4 の手順を一定回数又は一定時間の間繰り返した後に根ノードの子ノードの中で最もプレイアウトの数が多いノードが表す行動を返す。

1. **選択** $UCB1$ が最大の子ノードを選択する。これを展開済みの末端ノードになるまで繰り返す。

2. **拡張** 選択された回数が閾値を超えたノードで子ノードを展開する。展開した子ノードの中から $UCB1$ が最大のノードを選択する。

3. **シミュレーション** 1、2 で選択したノードの行動、ランダムに選択した行動の順でシミュレーションを行って、プレイアウトを求める。

4. **逆伝播** 3 の結果から 1、2 で選択したノードの $UCB1$ を更新する。

3.3 FightingICE での実装

吉田ら [7] は、エージェントが操作するキャラクターの行動選択をモンテカルロ木探索によって行った。格闘ゲームにおいては、計算量が膨大になるためプレイアウトをラウンド終了まで求めるのは困難であるので、$\overline{X}$ を (3) 式によって算出する。

$$\overline{X} = \frac{1}{N}\sum_{i=1}^{N}(difHP_i^{my} - difHP_i^{opp}) \tag{3}$$

N はノードの総探索回数、$difHP_i^{my}$ で i 番目のシミュレーション前後での自身の HP の変化量、$difHP_i^{opp}$ で対戦相手の HP の変化量を表す。

モンテカルロ木探索において設定するパラメータは表 1 があり、吉田らは $C = 3$, $N_{max} = 10$, $D_{max} = 2$, $T_{sim} = 60$, $T_{max} = 16.5$, $A_{num} = 5$ と設定した。

表 2: モンテカルロ木探索におけるパラメータ

パラメータ	詳細
C	$UCB1$ を算出する際に使用する定数。数値が高いほど選択回数が少ないノードが選ばれやすくなる。
N_{max}	子ノードを展開する際の閾値。選択された回数がこの閾値を超えると子ノードを展開する。
D_{max}	木の深さの最大値。この値まで木を展開することができる。
T_{sim}	シミュレーション時間。シミュレーションをこの値フレーム先まで行う。単位はフレーム
T_{max}	探索時間。この時間の間モンテカルロ木探索で探索を続ける。単位は秒
A_{num}	行動数。シミュレーションする際にエージェントが行う行動の数

FightingICE ではゲームの状況が 1 フレーム (1/60 秒) 毎に変化し続けるため、1 フレーム以内に行動を選択しなければならない。吉田らのモンテカルロ木探索による行動選択では、その時間の制約から図 2 のように、ゲーム環境から自身と対戦相手のキャラクターデータなどのフレームデータを 1 フレーム毎に取得し、対戦相手の行動予測やエージェントが行動を選択するための計算を行っている。計算時間が 1 フレームしかないためプレイアウトの回数が不足し、行動予測や行動選択を行うための探索が足りていない [8] と思われる。

4. 提案手法

本研究では、プレイアウトの回数を増加させるために、応答時間を延長することでモンテカルロ木探索の探索時間を増加させる手法について述べる。加えて、延長したフレーム数が多いほど、フレームデータを取得した時と行動を入力した時とのゲーム環境の差が大きくなり、推定した状況と実際の状況との差が大きくなってしまうことが予測されるため、応答時間を延長するフレーム数と即応性のトレードオフを観察する。

提案手法では、図 3 のように応答時間を 1 フレームから N フレーム (N は自然数) に延長する。モンテカルロ木探索の探索時間に使用できる時間を増加させることによって、行動選択能力の向上を図った。ただし、応答時間を延長するほどフレームデータを取得した時と行動を入力した時とのゲーム環境の差が大きくなってしまうため、選択した行動が実行する状況に適していない場合が考えられる。そのため、応答時間を延長するフレーム数とゲーム環境の即応性のトレードオフを観察する。

なお、FightingICE の仕様上、エージェントは 1 フレーム毎に行動を入力しなければならないので、図 2 の斜めの矢印のように応答時間を延長して計算時間が 1 フレームを超える毎に、一番最後に入力した行動と同じ行動をゲーム環境に入力される。

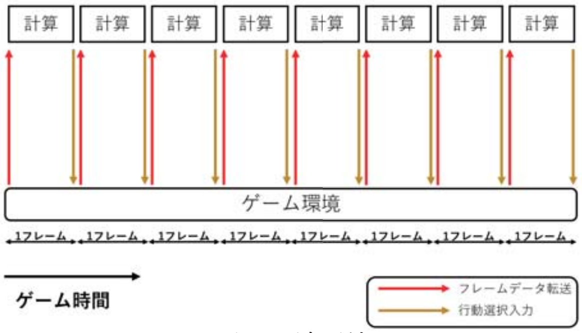

図 2: 既存手法

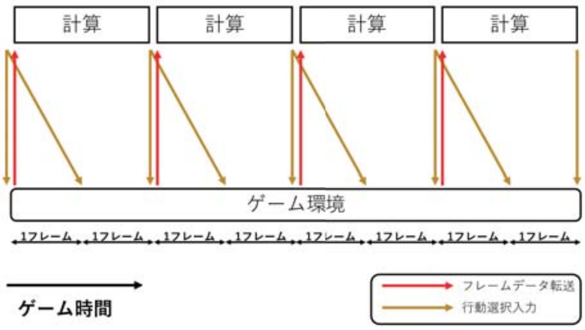

図 3: 提案手法 ($N = 2$ の場合)

5. 実験・考察

応答時間を延長したモンテカルロ木探索を用いて行動選択を行うエージェントの性能評価を行い、実験結果から考察を行った。今回、提案手法はモンテカルロ木探索によって行動選択を行う MctsAI [3] の応答時間を延長することで実装した。実験では吉田らや MctsAi と同様のパラメータである、$C = 3$, $N_{max} = 10$, $D_{max} = 2$, $T_{sim} = 60$, $T_{max} = 16.5$, $A_{num} = 5$ とした。

5.1 対戦スコア比較

対戦スコアを計測する際の対戦相手として MctsAI [3] を用いた。MctsAi パラメータは提案手法と同様にしたため、応答時間が 1 フレームの場合は MctsAi と同様となる。各応答時間ごとに 10 試合を行い、その対戦スコアを算出した。

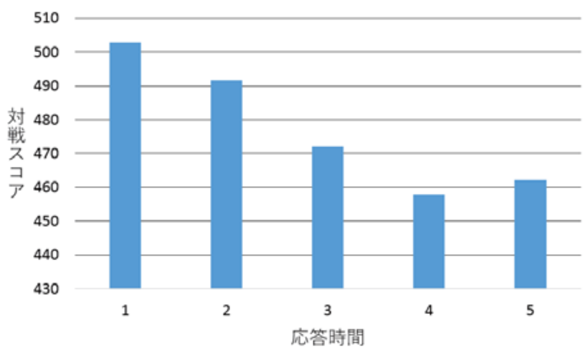

図 4: 対戦スコア

図 3 の対戦結果から応答時間が大きくなるほど対戦スコアが低下していく傾向が読み取れる。応答時間を延長することでモンテカルロ木探索の探索時間に使用できる時間を増え、プレ

イアウトの回数が増加し行動選択能力が上昇すると思われた。しかし、応答時間を延長することによってゲームの情報を取得してから実際に行動するまでの時間が大きくなり、推定した状況と実際の状況との差が大きくなってしまうことで状況に適した行動ではなくなってしまうことの影響の方が大きかったと推測される。

5.2 探索スコア比較

応答時間を延長することが行動選択にどのような影響を与えているのかを検証するために、モンテカルロ木探索によって行動を決定した時の探索スコア（(3) 式の $\overline{X}$）を比較する。キャラクターの位置や状況などが同様な状況下である方が好ましいため、対戦相手として単一ルールによって行動を選択する Jay Bot[3] を用いた。各応答時間毎に 5 試合行い、応答時間内で行った探索回数の平均と選択した行動の探索スコアの平均を求めた。

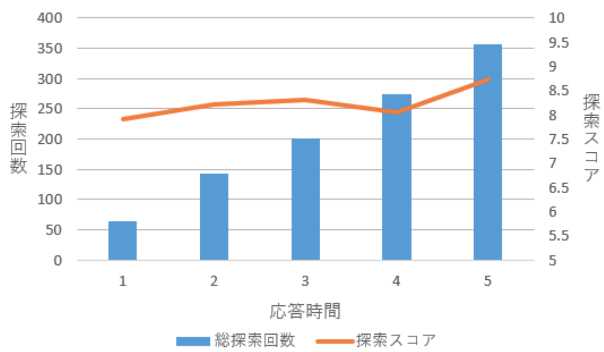

図 5: 探索スコア比較

図 4 の実験結果から、応答時間が増加するにつれて探索回数は増加したが探索スコアはそれほど上昇しない傾向が見られた。原因としては、探索回数を増加しても同じノードを探索して他のノードをあまり探索していない可能性が考えられる。そのため、この実験での総探索回数のうち最も探索回数が多いノードが占める割合を求めた。

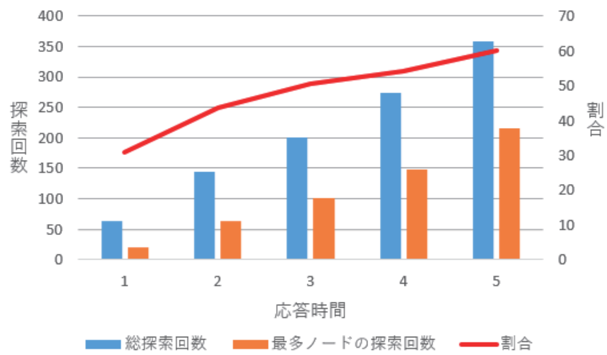

図 6: 探索回数割合比較

図 5 より応答時間を延長するのに伴って、総探索回数のうちで最も探索回数が多いノードが占める割合が上昇しているのが見て取れる。このことから探索回数の増加分が最多ノードの探索に費やされており、他のノードをあまり探索していない可能性があることが推定される。

5.3 考察

実験より、探索回数を増やしただけでは探索スコアの上昇よりも、応答時間の延長によるデメリットの方が大きく、対戦に悪影響を及ぼしてしまうことが判明した。探索スコアが上昇しにくかった原因として、今回利用したモンテカルロ木探索のパラメータが応答時間を延長した場合に適していなかったことが推測される。よって延長するフレーム数ごとに適したパラメータを設定する必要があると推察する。

6. おわりに

6.1 まとめ

本研究では、格闘ゲーム AI の性能向上を目的として応答時間を延長することで計算時間を確保し、モンテカルロ木探索の探索回数を増加させた場合における試合への影響を確認した。結果は、延長するフレーム数を増やすほど対戦スコアは減少してしまった。延長するフレーム数と即応性のトレードオフが見られなかった原因としては、モンテカルロ木探索で設定するパラメータが応答時間を延長した場合に適していなかったことが挙げられる。

6.2 今後の課題

今後の課題は、応答時間に対応した適切なパラメータの設定により探索効率を向上することでトレードオフを観察し、最適な計算時間を求めることが挙げられる。

参考文献

[1] David Silver et al. Mastering the game of Go with deep neural networks and tree search, *Nature*, Vol. 529, pp. 484–489, 2016.

[2] 遠藤雅伸, 「ディジタルゲームにおける”AI” の役割」, 情報処理学会, Vol. 53, No.2, pp.146-152 2012.

[3] http://www.ice.ci.ritsumei.ac.jp/~ftgaic/(2018 年 1 月 23 日参照)

[4] Man-Je Kim, Kyung-Joong Kim, ”Opponent Modeling based on Action Table for MCTS-based Fighting Game AI”, *IEEE Conference on Computational Intelligence and Games (CIG)*, 2017.

[5] 井上由紀子, 山根一人, 大河俊博, 中原啓晶, 高橋由起子, 落保子「単純反応時間の学習効果：特に健常人に対する検索」, 理学療法学, 17, p. 51, 1990.

[6] 美添一樹 「モンテカルロ木探索」, 「人工知能学大辞典」, pp. 1193–1195, 共立出版, 2017.

[7] Shubu Yoshida, Makoto Ishihara, Taichi Miyazaki, Yuto Nakagawa, Tomohiro Harada and Ruck Thawonmas, “Application of Monte-Carlo Tree Search in a fighting game AI,” *IEEE Global Conference on Consumer Electronics*, 2016.

[8] Man-Je Kim, Kyung-Joong Kim, ”Opponent Modeling based on Action Table for MCTS-based Fighting Game AI”, *IEEE Conference on Computational Intelligence and Games (CIG)*, 2017.