G-109

オープンループサーチを用いた対戦格闘ゲーム AI の提案

Proposal of Fighting Game AI Using the Open-loop Search

吉田 修武† 朱 晉賢‡ 石原 誠‡
Shubu Yoshida Chu-Chun Yin Makoto Ishihara
中川 裕登‡ 原田 智広† ターウォンマット ラック†
Yuto Nakagawa Tomohiro Harada Ruck Thawonmas

1．はじめに

CIG で対戦格闘ゲーム AI の国際大会が 2013 年から開催されている．この大会は我々の研究室が開発した対戦格闘ゲームのプラットフォーム FightingICE（図 1）[1]を用いて出場した AI 同士を総当りで対戦させ，最も相手に打ち勝った AI を決めるものである．近年，優勝または上位の AI は学習を必要とせず，予め決められたルールで行動するルールベースの AI が多い．ルールベースは予め決められた行動しか選択肢がないため，人間のプレイヤーやその弱点を突く他の AI によって容易に対処されてしまう問題がある．

この問題に対して，ランダムシミュレーションに基づいて行動を決定するモンテカルロ木探索（MCTS）を格闘ゲーム AI に適用させることが考えられる．MCTS は，近年囲碁といったボードゲームにおいて顕著な成果を挙げている．また，ボードゲームのようなターンベースゲームだけでなく，Ms.Pac-Man のようなリアルタイムゲームでも成果を挙げており，2016 年には FightingICE の公式ホームページで MCTS を用いた AI（MCTSAI）[2]が公開された．

MCTS はリアルタイムゲームで優れた性能を引き出すために，短時間でより多く探索して様々な状況をシミュレートする必要がある．そこで本稿では，General Video Game Playing で優れた性能を示したオープンループサーチ（OLS）[3]を用いた格闘ゲーム AI を提案する．OLS はノードに統計情報のみを格納するため，シミュレートを毎回ルートノードから末端ノードまで行うので，多様なふるまいをする AI と対戦する際の様々な状況に対応することができる．

本稿では，MCTS に OLS を適用させる．そして，サンプルとして公開されている MCTSAI と対戦させ，その性能を評価する．

2．格闘ゲームにおけるゲーム木探索

ゲーム木探索とは，ゲーム内での AI の意思決定として最も多く用いられている技術の一つであり，もともとチェスや囲碁のような 2 プレイヤー対戦のボードゲームに用いられたものである．ミニマックスのような古くからある技術や MCTS のような最先端の技術などがある．

†立命館大学情報理工学部
College of Information Science & Engineering, Ritsumeikan University

‡立命館大学大学院情報理工学研究科

Graduate School of Information Science & Engineering, Ritsumeikan University
a) ruck@is.ritsumei.ac.jp

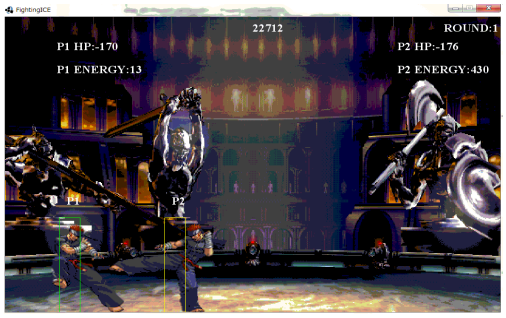

図 1．FightingICE

ゲーム木は，一般にノードをゲームの状態，ルートノードを現在のゲームの状態，エッジを行動として構築される．また，木の終端ノードはゲームの終了状態であり，そのゲームの勝敗が評価として得られる．しかし，格闘ゲームでは，1 フレームごとに行動を選択する必要があるため，探索に使用できる時間がターンベースゲームに比べて圧倒的に少なく，終端状態へたどり着けないことが多い．この場合，木の最大の深さDを定義することで現在の状態から移動できる深さを制限し，勝敗の代わりに非終端状態を評価するヒューリスティック関数を定義する．これによって，終端状態までシミュレートせずに状態を評価することができる．

3．提案手法

3.1　格闘ゲームにおける木探索の問題点

探索に使用できる時間が非常に短く，探索の試行回数が限られる問題点を解決するために，探索で得られた情報を再利用することで試行回数を増やす方法が考えられる．しかし，非決定性を持つ格闘ゲームでは同じ行動を選択したとしても次の状態は相手の行動によっても変化するため一意に定まらない．そのため，従来のゲーム木探索において，構築された木は行動決定が終わった後に破棄され，再利用できない．この問題に対して本研究では木の再利用が可能な OLS に着目する

3.2　オープンループサーチ

OLS は，ノードに状態を格納せず，状態から得られた統計情報のみを格納することによって木の再利用を可能にする方法である．ここで，統計情報とは，報酬，訪問回数のような探索で得られた情報のことである．そして，選択された行動のノードをルートノードへと切り替え，選択されなかった行動のノードを破棄することで木の再利用を実現する．（図 2）更に，木を再利用できるという利点から，プレイアウト時にノードを生成，保持することで効率的な探索が可能となる．

一般的に，木探索ではノードにそれまでの行動を行った

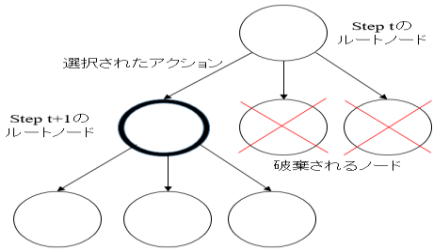

図 2. ルートノードの切り替え

シミュレーションを持たせるが，OLS ではこれを持たない．そのため，親ノードから当該ノードまでのアクションを行わなければ，その状態を取ることができず，ノードの途中からシミュレーションをすることはできない．しかし，試行毎に異なる状態を持つことができるという利点があるため，格闘ゲームのような非決定性を持つゲームに適している．

本稿では， FightingICE のサンプルである MCTSAI に使用されている MCTS[2]に OLS の木構造を適用した AI（OLSAI）を提案する．

格闘ゲーム AI に OLS を適用する場合，格闘ゲームにおいて取れる行動が多いため，実際のゲームでの結果と異なるシミュレーションによって，統計情報がノードに格納されたまま引き継がれてしまう問題が発生する．これは訪問回数が十分確保できれば，回避できるが，格闘ゲームでは取れる行動が多く，シミュレーション時間内で十分な訪問回数を確保できない．

前述の問題点の対策として，ノードの情報をルートノードに切り替えた時に，後述する基準を満たさない場合は同情報をリセットする．これにより，十分な回数シミュレートされていない信頼性の低い統計情報を引き継ぐ可能性を低減することができ， OLS の木を再利用するという特徴を残しつつ，問題点を解決する．

4．OLSAI の評価実験

4.1 FightingICE

FightingICE は，1 試合が 1 ラウンド 60 秒の 3 ラウンドで構成されている，各キャラクターの HP の初期値は 0 に設定されており，攻撃を受けると際限なく減少していく．そして 60 秒が経過した時点で次のラウンドに移行し，HP が 0 にリセットされる．各ラウンド終了時のお互いの HP から，式（5）によってスコアがラウンドごとに算出される．

$$Score_{my} = \frac{HP_{opp}}{HP_{my} + HP_{opp}} * 1000 \qquad (4)$$

Hp_{my}，HP_{opp}はそれぞれ自分，相手 AI の HP である．同式より，スコアの上限は 1000 である.自分よりも相手の HP の方が小さい場合，自分の獲得スコアは 500 より大きくなり，逆に相手の HP の方が大きい場合,500 より小さくなる．本実験では，FightingICE version 1.23 を使用する．

4.2 実験手法

OLSAI と，FightingICE の公式ホームページで公開されている MCTSAI で対戦を行う．1P 側で 50 試合，2P 側で 50 試合の計 100 試合行う．そして，獲得スコアによって提案 AI の性能を評価する．AI に用いたパラメータは[2]のものを使用している．ノードの情報をリセットする基準として以下の 4 種類を用い，それぞれ p0， p1， p2， p3 とし，

表 1．100 試合における各 AI の MCTSAI に対して獲得したスコアの平均及び信頼区間（信頼度 95%）

AI NAME	Score	Confidence interval
p0	336	±6.5
p1	498	±5.2
p2	499	±5.1
p3	505	±6.0

OLSAI-p0 のように表記する．

- p0：ノードの統計情報をリセットしない．
- p1：ルートノード変更時にすべてのノードの統計情報をリセットする．
- p2：訪問回数がルートノード訪問回数より 10 分の 1 以下となるノードの統計情報をリセットする．
- p3：訪問回数がルートノード訪問回数より 5 分の 1 以下となるノードの統計情報をリセットする．

4.3 実験結果

表 1 に 100 試合における各提案 AI が MCTSAI に対して獲得したスコアの平均及び信頼区間（信頼度 95%）を示す．表より，OLSAI-p0 は MCTSAI に比べて明らかにスコアが低いことがわかる．これに対し，提案手法 OLSAI-p3 は MCTS に対して優位な性能を示している．このことから，提案手法を加えたことで OLSAI の性能が向上されていることがわかる．また，OLSAI-p1， OLSAI-p2 は優位な性能を示すことができなかった．よって，ある基準値以上の訪問回数を持つノードは統計情報をリセットせずに次のステップでも保持し続けることが有用であるとわかる．

以上より，訪問回数が基準値以下のノードの情報をリセットする OLS を導入することで格闘ゲーム AI の性能が向上されていることが明らかになった．

5．結論

本稿より，格闘ゲームにおいて OLS を単純に適用しただけでは性能があまり出ないことがわかった．しかし，訪問回数の低いノードの統計情報をリセットする機構を組み込むことで MCTS よりも良い性能を示した．今後の課題として，より OLS を強化するため，シミュレーションで相手の行動をランダムに選択せず，相手の行動を予測し，予測した行動を基にシミュレーションを行う．これにより，考えうる状態を減らすことができると考えられる．

参考文献

[1] Feiyu Lu, Kaito Yamamoto, Luis H. Nomura, Syunsuke Mizuno, YoungMin Lee, and Ruck Thawonmas, "Fighting Game Artificial Intelligence Competition Platform," Proc. of the 2nd IEEE Global Conference on Consumer Electronics (GCCE 2013), Tokyo, Japan, pp. 320-323, Oct., 2013.

[2] Shubu Yoshida, Makoto Ishihara, Taichi Miyazaki, Yuto Nakagawa, Tomohiro Harada, and Ruck Thawonmas, "Application of Monte-Carlo Tree Search in a Fighting Game AI," Accepted for presentation at the 2016 Global Conference on Consumer Electronics (GCCE 2016), Kyoto, Japan, Oct., 2016.

[3] Diego Perez, Jens Dieskau, Martin Hünermund, Sanaz Mostaghim, and Simon M. Lucas. "Open Loop Search for General Video Game Playing," Proc. of the Conference on Genetic and Evolutionary Computation (GECCO 2015), Madrid, Spain, pp. 337-344, Jul., 2015.