2048へのMC Softmax探索の適用

渡邊翔太[1,a)]　松崎公紀[2,b)]

概要：MC Softmax 探索は，評価関数を併用するモンテカルロ木探索の一種である．本研究では，MC Softmax 探索と強化学習を用いて，確率的一人ゲーム 2048 のコンピュータプレイヤを作成する．評価関数には，既存の畳み込みニューラルネットワークを用いる．まず，ランダム性を固定した複数の木に対して MC Softmax 探索を適用するアルゴリズムを設計し，木の数を変更して実験を行った．実験の結果，最も優れたプレイヤは平均スコア 48 008 を達成した．次に，chance ノードを含む木に対して MC Softmax 探索を適用するアルゴリズムを設計し，いくつかの学習方法について実験を行った．これらの実験から，探索結果の木の値を直接学習すると，それにより得られる評価関数の返り値が無限大に発散してしまう問題が生じた．

Adaption of MC Softmax Search for 2048

Shota Watanabe[1,a)]　Kiminori Matsuzaki[2,b)]

Abstract: MC Softmax search is a Monte-Carlo Tree Search algorithm that uses an evaluation function. In this study, we develop a computer player for a stochastic single-player game 2048 based on reinforcement learning methods and MC Softmax search. We use an existing convolutional neural network for the evaluation function. Firstly, we developed an algorithm that applied MC Softmax search to multiple trees without randomness and conducted experiments by changing the number of trees. The best player achieved an average score of 48 008. Secondly, we developed an algorithm that applied MC Softmax search to a single tree with chance nodes and conducted experiments with several learning methods. The experiment results, however, showed an issue that the direct learning of values from search outcomes would result in evaluation functions returning infinitely large values.

1. はじめに

本研究で対象とするゲームは，G.Cirulli が 2014 年に公開した確率的一人ゲーム 2048 である．2048 は，ルールは簡単であるものの，高得点を得るのはそれほど容易ではない．これまでに作成された優れたコンピュータプレイヤの多くは，評価関数（N タプルネットワーク [2], [4] やニューラルネットワーク [3]）を TD 学習または類似の手法により学習させたものである．最先端のプレイヤ [2] は，N タプルネットワークと拡張した TD 学習に基づくもので，平均得点 625 377 を達成している．TD 学習やその派生学習アルゴリズム以外の学習アルゴリズムを利用したプレイヤとしては，方策勾配法を用いて平均得点 26 861 のプレイヤを達成したことが報告されている [8].

近年，AlphaZero [5] の登場により，モンテカルロ木探索（MCTS）とニューラルネットワーク（評価関数）を併用した強化学習が注目されている．AlphaZero の手法は様々なゲームに適用可能な反面，膨大な計算資源が必要であることは大きな問題点である．Antonoglou ら [1] は，AlphaZero を拡張した stochastic MuZero 手法を 2048 に適用し，平均得点約 550 000 を達成している．しかし，その手法は 10^{10} 局面のスケールでニューラルネットワークを学習させる必要がある．1 局面あたりの学習時間が著者らの先行研究 [3] と同程度であるとすると，10^{10} 局面の学

[1] 高知工科大学大学院工学研究科
Graduate School of Engineering, Kochi University of Technology
[2] 高知工科大学情報学群
School of Information, Kochi University of Technology
[a)] 265112q@gs.kochi-tech.ac.jp
[b)] matsuzaki.kiminori@kochi-tech.ac.jp

習には 150 日が必要であり，現実的とは言えない．このことから，より軽量な強化学習手法について検討することが必要だと著者らは考える．

バリュー（値）関数のみの学習であれば，TD 学習の派生アルゴリズムにより強いプレイヤができることが先行研究 [3] で示されている．MC Softmax 探索に基づく強化学習手法 [7] は，AlphaZero と同様に木探索と評価関数を併用する強化学習手法であるが，評価関数にはバリュー関数のみが必要でありポリシー（方策）関数を必要としない．そこで本研究では，MCSoftmax 探索に基づく強化学習手法を 2048 に適用し，優れたプレイヤを実現することを目指す．

本研究では大きく以下の 2 点に取り組む．

- ランダム性を固定した複数のゲーム木に対して MC Softmax 探索を適用するアルゴリズムを設計し，その結果を用いて強化学習を行う．この実験の結果得られた最も優れたプレイヤは，平均スコア 48 008，最高スコア 79 904 を達成した．（第 4 節）
- ランダム性を chance ノードとして含む木に対して MC Softmax 探索を適用するアルゴリズムを設計し，その結果を用いた強化学習手法をいくつか検討する．この実験では，探索結果として得られる木の値を直接学習すると，評価関数の返り値が無限大に発散してしまう問題が生じた．（第 5 節）

本研究では，残念ながら，既存の TD 学習ベースの強化学習を越える結果は得られなかった．学習がうまく進まなかった理由について，いくつかの考察を与える．

本論文の構成は以下のとおりである．第 2 節では，ゲーム 2048 のルールを説明した後，2048 の持つ他のゲームと異なる特徴について示す．第 3 節では，MC Softmax 探索とそれに基づく強化学習手法を概観する．第 4 節では，MC Softmax 探索の 2048 への適用方法の 1 つ目として，ランダム性を固定した複数のゲーム木に MC Softmax 探索を適用するアルゴリズムを設計し，学習を行った結果を示す．第 5 節では，MC Softmax 探索の 2048 への適用方法の 2 つ目として，ランダム性を chance ノードとして含む木に対して MC Softmax 探索を適用するアルゴリズムを設計し，学習を行った結果を示す．第 6 節で本論文をまとめる．

2. ゲーム 2048

2.1 ルール

ゲーム 2048 は，スライド& マージ型ゲームに分類される確率的一人ゲームである．4×4 の盤面でプレイされる．初期状態では，2 つのタイルがランダムに置かれ，それらタイルの値は 2（確率 0.9）か 4（確率 0.1）である．各ターンにおいて，プレイヤは上下左右のいずれかの方向を選択し，全てのタイルを選択した方向に移動する．同じ値をもつ 2 つのタイルが移動方向に衝突すると，それらのタイルは結合して和の値を持つタイルとなり，その和の値が得点に加算される．ここで，結合するタイルは奥側が優先され，一度結合したタイルは同じターンではそれ以上結合しない．したがって，ある行のタイルが 222␣，␣422，2222 であるときに右向きに移動した結果は，それぞれ ␣␣24，␣␣44，␣␣44 となる．どのタイルも移動・結合しない方向を選ぶことはできない．移動・結合の後，新しいタイルがいずれかの空きマスにランダムに 1 つ出現する．新しく出現するタイルの値は初期状態と同様に 2（確率 0.9）か 4（確率 0.1）である．プレイヤがどの方向にも移動・結合できなくなったら終局である．

もともとのゲーム 2048 における目標は，上記のルールのもとで移動と結合を繰り返して 2048 の値をもつタイルをつくることである．本研究では，2048 のタイルができた後もゲームを続け，できるだけ高得点をとることを目指す．

2.2 特徴

ゲーム 2048 には，これまでに多くの研究のある将棋や囲碁と違って以下のような特徴がある．

ランダム性 各ターンで，タイルの移動・結合の後で，ランダムな空きマスにタイルが追加される．追加されるタイルは，出現位置と値（2 が 90%，4 が 10%）がランダムに決まる．

ゲーム木の形 プレイヤが選択できる手は上下左右の 4 通りの方向である．ランダムなタイルの出現については，可能性は最大 30 通り（平均的にはその半分）である．優れたプレイヤでは，ゲームが 2 万手以上続く．したがって，将棋や囲碁と比べて，分岐数が小さく木の高さが大きい．

これらの特徴により，先行研究 [3], [6] では TD 学習に基づく評価関数の学習が行われており，優れた成果が得られている．とくに，評価関数の入力とする局面については，Szubert と Jaśkowski の研究により，学習においてランダム性の影響が小さくなるタイルの出現前の局面（Afterstate と呼ぶ，図 1）とすることが良いことが知られている．

本研究でも，学習する評価関数の入力は Afterstate とする．

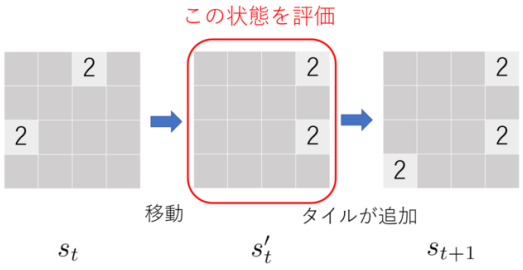

図 1 Afterstate

3. MC Softmax 探索とそれに基づく強化学習

MC Softmax 探索は，将棋ソフト「芝浦将棋 Softmax」で用いられている探索手法である [7].

MC Softmax 探索は，他のモンテカルロ木探索と同様に選択→展開→評価→バックアップの 4 フェーズからなるシミュレーションを繰り返す．古典的なモンテカルロ木探索との違いは，(1) 評価フェーズにおいてプレイアウトではなく評価関数による値を返すこと，(2) 選択フェーズとバックアップフェーズにおいて評価値によるボルツマン分布を用いること，(3) 木探索の結果をサンプリングして評価関数の学習データを得ることである．

3.1 ボルツマン分布

MC Softmax 探索では，選択フェーズとバックアップフェーズにおいて，ボルツマン分布を用いる．p を現在のノード，$x_i \in children(p)$ を p の子ノード，$v(x_i)$ を x_i の評価値，T を温度パラメータとする．このとき，ノード x_i の選択確率はボルツマン分布により次のように与えられる．

$$\pi(x_i) = \frac{\exp(v(x_i)/T)}{\sum_{x_j \in children(p)} \exp(v(x_j)/T)}$$

ボルツマン分布における温度パラメータ T を調整することで，探索における「探索と活用」をコントロールできる．

MC Softmax 探索では，このように評価値によるボルツマン分布により方策が与えられる．AlphaZero [5] とは異なり，ポリシー（方策）関数は不要であり，バリュー（値）関数のみが必要となる．

3.2 選択フェーズ

選択フェーズでは，子ノードの評価値によるボルツマン分布に従う確率で子を選択する．選択フェーズで用いる温度パラメータを，選択温度と呼ぶこととする．

図 2 は，選択ステップの一例を示したものである．親ノードは値が 0.5，0.45，0.1 の 3 つの子ノードを持っている．選択温度を $T = 0.2$ とすると，3 つの子を選ぶ確率はそれぞれ 0.522, 0.406, 0.070 である．この確率で子ノードを選び，例では 2 番目に良いノードが選ばれている．

3.3 バックアップフェーズ

バックアップフェーズでは，子ノードの評価値によるボルツマン分布を重みとして，子ノードの値の重み付き平均を親ノードの値とする．バックアップフェーズで用いる温度パラメータを，バックアップ温度と呼ぶこととする．

図 3 はバックアップフェーズの例である．まず，2 番目の子の値が 0.33 に更新されたとする．このとき確率を計算しなおすと，0.405，0.335，0.260 となる．この重みを用いて重み付き平均を計算すると 0.34 となり，この値が親ノードの値となる．

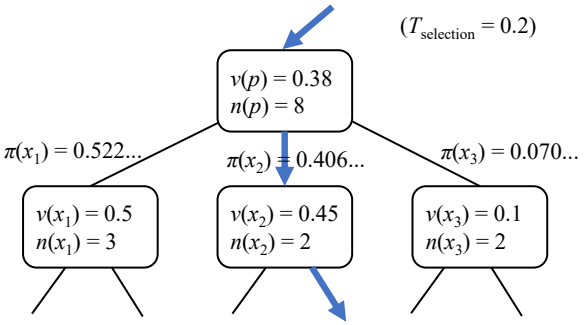

図 2　MC Softmax の選択フェーズの例

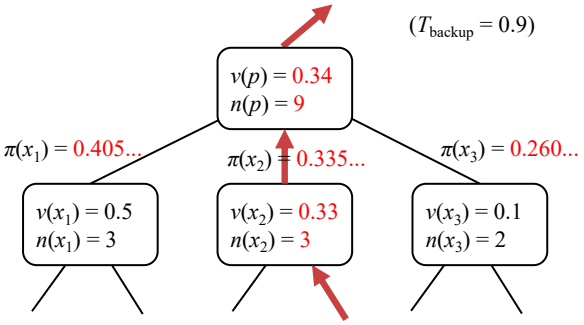

図 3　MC Softmax のバックアップフェーズの例

3.4 MC Softmax に基づく強化学習

MC Softmax 探索を行うと，より深いノードの評価値により根に近いノードの値が再計算される．したがって，探索後の内部ノードの値は，評価関数の値に比べてより正しい値を持つと期待される．この差を利用することで，評価関数を強化学習により調整することが可能である．

具体的には，MC Softmax 探索の計算が終わった後で，選択フェーズと同様の手法により学習に用いる局面（ノード）をサンプリングする．ここで用いる温度（サンプリング温度）は選択フェーズのそれと同じである必要はない．

4. 実験 1：ランダム性を固定した複数の木に対する MC Softmax 探索

2048 では，タイルの出現位置とその値がランダムに決まるため，それらを完全に表現しようとするとゲーム木が膨大になってしまう．そこで，まず，それらのランダム要素を固定した小さな木に対して MC Softmax 探索を適用する方法を考えた．その代わり，ランダム性を固定化する悪影響を緩和するために，複数のゲーム木を生成してそれぞれに探索を行う．

4.1 評価関数

評価関数には Matsuzaki [3] が設計した CNN22 を使用した．CNN22 は，2 層の畳み込み層と 3 層の全結合層からなる畳み込みニューラルネットワークである（図 4）．学習の最初に，CNN22 に含まれるパラメータを正規分布に従う乱数で初期化する．学習では，MC Softmax 探索で得ら

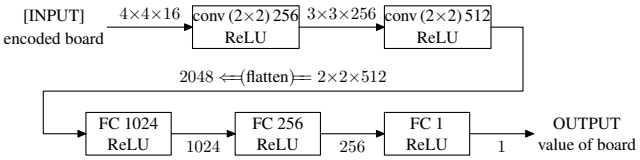

図 4　CNN22 の構成 [3]

表 1　探索木の数，シミュレーション数，サンプリング数

プレイヤ	探索木	シミュレーション	サンプリング
greedy	1	10	1
1-tree	1	100	5
5-tree	5	20	2
10-tree	10	10	2

れた探索木中の Afterstate とその値を訓練データとして与える．ただし，同一局面に対する評価値との学習する値との差が最大 100 となるように制限（クリッピング）した．

このようにして学習した評価関数は，与えられた局面からゲーム終了までに得られるスコアを近似することを期待している．著者らの経験から，2048 におけるスコア（および評価関数の値）は，勝率とは異なり値が大きくかつゲーム状況で値が大きく変わる．この問題に対応するため，ボルツマン分布への計算の直前で，子の評価値のうち最大のものを 1 とするようにスケーリングすることとした．

4.2　ハイパーパラメータ

ここでの探索・学習アルゴリズムには以下に示すハイパーパラメータがある．

- 探索木の数
- 各探索木に対するシミュレーション回数
- 各探索木にサンプリング数（パス数）
- 選択温度 T_s
- バックアップ温度 T_b
- サンプリング温度 T_p

探索木の数，シミュレーション数，サンプリング数については，表 1 に示す 4 つの組合せを試した．プレイヤ 1-tree，5-tree，10-tree は，いずれも探索木の数とシミュレーション数の積が 100 となっている．一方，プレイヤ greedy は，探索木が 1 つでシミュレーション数が少なく，貪欲な探索による学習を模倣することを期待したものである．

選択温度 T_s とバックアップ温度 T_b は，ゲーム・学習の進行度に依らず，全体の n 中の i 番目のシミュレーションにおいて $T = 1 - (i-1)/n$ と設定した．一方，学習データのサンプリングにおいては，サンプリング回数が少ないため，サンプリング温度は $[0.005, 1)$ の範囲の乱数で決定した．

4.3　実験環境と実験結果

実験には，CPU：Intel Core i7-9800X (3.8 GHz, 8 コア/16 スレッド)，メモリ：128 GB，GPU：NVIDIA GeForce RTX 2080Ti (GPU メモリ 11 GB) × 2 を持つ計算機を用いた．ただし，各プログラムは GPU を 1 つだけ使うように設定し，同時に 2 つのプログラムを動かした．また，乱数による影響を小さくするため，各プログラムを 4 回実行してその平均値を評価に用いた．

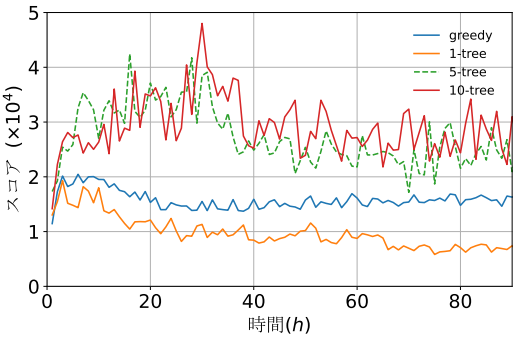

図 5　実験 1：ランダム性を固定した複数の木に対する MC Softmax 探索における学習の結果

それぞれのプレイヤについて，1 時間毎のプレイ結果の平均点をプロットしたものを図 5 に示す．本研究の実験で最も良い結果は，最も探索木の多い 10-tree によるもので，平均得点の最高値は 48 008（30 時間後）となった．また最高得点として 79 904 を達成した．複数の探索木を用いる 10-tree と 5-tree では近い結果となっている一方，探索木の数を 1 とした 1-tree では学習時間を大きくすると平均点が悪くなるという結果となった．実際，探索をほとんど行わない greedy と比べても悪い結果となっており，探索においてランダム性を適切に考慮することが重要であることが示唆される．

先行研究 [3] では，24 時間の学習後に探索なしで平均得点約 8 万点を達成している．したがって，学習データ生成の際に探索の有無が異なる方法を直接比較するのは適切ではないかもしれないが，MC Softmax 探索による本研究での学習は探索なしでの学習を超えていない．また，今回の実験の結果では，約 30 時間の学習までは性能向上が見られるものの，それ以降は平均 25000〜30000 点に留まってしまっている．プレイにおける評価値を確認したところ，値が安定していないことが確認できた．

5.　実験 2：Chance ノードを含む木に対する MC Softmax 探索

実験 1 の結果より，ランダム性を適切に木に含むことが MC Softmax 探索においても重要であることが示唆された．そこで，ランダム性を固定せず，それを chance ノードとして含む木に対して MC Softmax 探索を適用するアルゴリズムについて設計する．

付録に示す予備実験より，適切な評価値のマッピングや温度パラメータを設定することで，chance ノードを含む木に対しても MC Softmax 探索が有用であることが分かっ

ている．以降では，選択・バックアップを次のように行う．

chance ノード　選択フェーズでは，タイルの出現確率に基づいて子ノードを選択する．バックアップフェーズでは，タイルの出現確率に基づいて重み付き平均を計算する．

max ノード　選択フェーズでは，子ノードの最大値 $v_{\max}$ により $v' = v/v_{\max}$ とマッピングした値を評価値とする．選択温度は，対象ノードの訪問回数を n として，$T = 2.0/2^{\log_{10} n}$ とする．バックアップフェーズでは，バックアップ温度を $T = 0$ とし，最大の子供の値を更新に用いる．

本節の実験には，CPU:Intel Core i3-8100 (3.6GHz, 4 コア/4 スレッド)，メモリ: 16GB, GPU: NVIDIA GeForce 1080Ti (GPU メモリ 11GB) を持つ計算機を使用した．

5.1　探索結果の木に含まれる局面を直接学習

予備実験として，探索結果の木に含まれる局面（評価値）を直接学習する以下の方法を試したが，いずれも評価値が安定しなかった．

- 探索結果として得られる木の内部ノード（局面）をすべて学習データとする．
- 探索結果として得られる木のノード（局面）のうち，訪問回数が閾値（10）を超えるノードの値を学習データとする．
- 評価関数の値と学習データの差分が一定範囲（± 100）に入るようクリッピングする．

5.2　探索により選択された手のみ学習

予備実験では学習を安定的に行うことができなかった．そこで，探索により選ばれた手（選択手）をもとに探索開始局面の評価値のみを更新して学習を行う方法を試した．具体的には，学習の目標値により以下の 2 つの手法を実装した．

NodeV+Reward　探索により計算された選択手の評価値に，その手で得られる得点を足した値を目標値とする．

EvalV+Reward　選択手の Afterstate に対する評価関数の値に，その手で得られる得点を足した値を目標値とする．

ただし，選択手を実行した結果ゲームオーバーとなった際には，目標値は 0 とした．

この実験では，1 手あたりの MC Softmax 探索のシミュレーション回数は $N = 100$ とした．サンプリングは，1 手あたり選択手から得られる 1 局面のみである．学習を開始するにあたり，既存手法 [3] の学習で得られた CNN22 の重みを初期値とした[*1]．

[*1] CNN22 を用いた 1 手先読み（貪欲プレイ）では平均スコア 206 802，予備実験の MC Softmax 探索のみ（学習なし）ではシミュレーション数 $N = 100$ で平均スコア 314 311 であった．

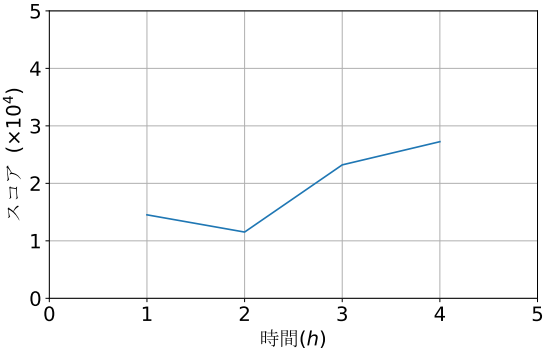

図 6　NodeV+Reward の結果

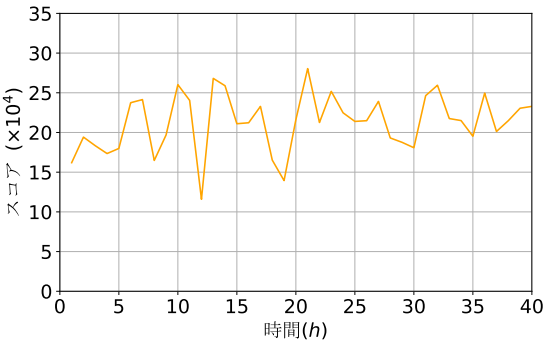

図 7　EvalV+Reward の結果

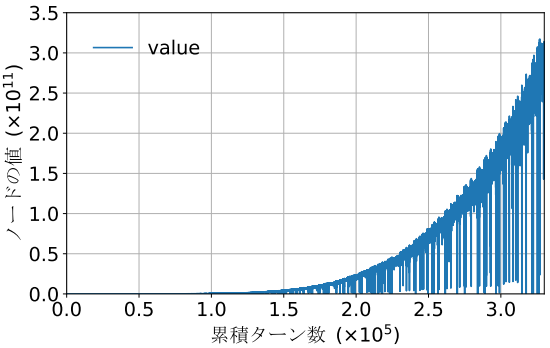

図 8　NodeV+Reward における値の変動

NodeV+Reward による学習結果を図 6 に，EvalV+Reward による学習結果を図 7 に示す．

NodeV+Reward は学習の途中で評価値の発散によりプログラムが停止してしまったため，4 時間分のデータしか得られなかった．また，貪欲プレイによるプレイヤと比較しても獲得スコアが大幅に下がった．図 8 にプレイヤが選択したノードの値を累積ターン毎に示す．指数関数的に評価値が増加しており，安定していないことがわかる．一方で，EvalV+Reward は評価値が安定してはいたが，学習を行わない MC Softmax 探索による平均点を下回る結果となり，40 時間の結果では学習によって性能が向上しているとはいえなかった．

以上の結果より，探索によって得られた評価値を直接学習させることが値の発散に繋がっていると著者らは予想する．この原因を解明するとともに，探索によって得られた

評価値を学習に用いる適切な方法を見出すことは重要な今後の課題である．

6. まとめ

本研究では，2048 に MC Softmax 探索を適用し，評価値の決定方法やハイパーパラメータについて検討した．1 つ目の実験では，ランダム性を固定した木を複数用いる方法を設計した．そのような木に対して MC Softmax 探索を適用し，探索結果の値を学習に用いるアルゴリズムを実験により評価した．その結果，最も優れたプレイヤは，平均スコア 48 008 を達成した．2 つ目の実験では，予備実験として行った学習を伴わない MC Softmax 探索で成功したアルゴリズムを基礎として，MC Softmax 探索とその学習とを行うアルゴリズムを複数試した．この実験では，探索結果の木の値を直接学習すると評価値が無限大に発散してしまった．今後の課題として，評価値が安定しない原因を追究し，学習方法を工夫する必要がある．

参考文献

[1] I. Antonoglou, J. Schrittwieser, S. Ozair, T. K. Hubert, and D. Silver: Planning in stochastic environments with a learned model, in *The Tenth International Conference on Learning Representations (ICLR 2022)*, 2022.

[2] H. Guei, L.-P. Chen, and I.-C. Wu: Optimistic temporal difference learning for 2048, *IEEE Transactions on Games*, 2021.

[3] K. Matsuzaki: Developing Value Networks for Game 2048 with Reinforcement Learning, *Journal of Information Processing*, Vol. 29, pp. 336–346, 2021.

[4] W. Jaśkowski: Mastering 2048 with delayed temporal coherence learning, multi-stage weight promotion, redundant encoding and carousel shaping, *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 10, no. 1, pp. 3–14, 2018.

[5] D. Silver, *et al.*: Mastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm, *arXiv*, Vol. 1712.01815 [cs.AI], 2017.

[6] M. Szubert and W. Jaskowski: Temporal Difference Learning of N Tuple Networks for the Game 2048, *2014 IEEE Conference on Computational Intelligence and Games*, pp. 1–8, 2014.

[7] 五十嵐治一, 森岡祐一, 山本一将: MC Softmax 探索における局面評価関数の学習, ゲームプログラミングワークショップ 2018 論文集, pp. 212–219, 2018.

[8] 山下修平, 金子知適: 2048 への方策勾配法の適用, ゲームプログラミングワークショップ 2021 論文集, pp. 179–185, 2021.

付　　録

A.1 学習を行わない MC Softmax 探索の評価

本研究の主題は MC Softmax 探索を用いた学習の有効性の評価である．ここでは，モンテカルロ木探索の一手法として見たときの MC Softmax 探索の性能について行った予備実験について，その結果を簡単に示す．なお，本付録に示す成果は，TAAI2022 に投稿中の論文からの抜粋である．

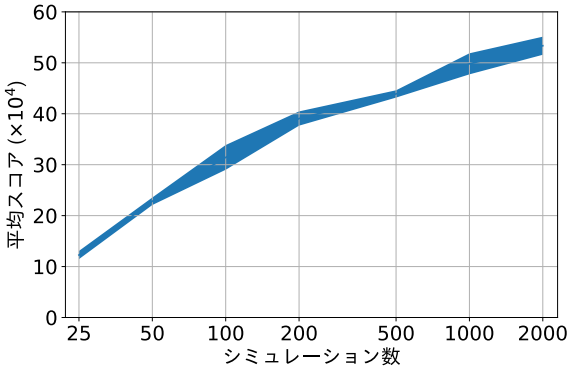

図 A·1　1 手あたりのシミュレーション回数に対する平均スコア

選択方策とバックアップ方策には，以下の 6 通りの方策を定義した．

L-log　$v' = \ln v$ と変換した値を評価値とし，対象ノードの訪問回数を n として，温度パラメータを $T = 0.4/2^{\log_{10} n}$ とする．

G-log　$v' = \ln v$ と変換した値を評価値とし，全体のシミュレーション数を n に対し，温度パラメータを $T = 0.4/2^{\log_{10} n}$ とする．

L-max　子ノードの最大値 $v_{\max}$ を用いて $v' = v/v_{\max}$ と変換した値を評価値とし，対象ノードの訪問回数 n に対し温度パラメータを $T = 2.0/2^{\log_{10} n}$ とする．

G-max　子ノードの最大値 $v_{\max}$ を用いて $v' = v/v_{\max}$ と変換した値を評価値とし，全体のシミュレーション数 n に対し温度パラメータを $T = 2.0/2^{\log_{10} n}$ とする．

ave　温度を $T = \infty$ と固定する．選択の際には等確率で選び，バックアップの際には平均をとる．

best　温度を $T = 0$ と固定する．選択・バックアップの際には最大値を選ぶ．

評価関数には，Matsuzaki による，TD 学習を 240 時間行った CNN22 ネットワークを用いた．

実験の結果，選択方策には L-max をバックアップ方策には best をそれぞれ用いるものが，最も高い平均点を得ることができた．図 A·1 に，最も優れた方策の組合せに対して，シミュレーション回数を変えて実行した場合の平均スコアを示す．用いた CNN22 ネットワークは，1 手先読みで貪欲にプレイした場合の平均スコアが 206 802 であり，MC Softmax では 1 手あたり 50 回のシミュレーションで性能が上まっている．シミュレーション回数を増やすと，平均スコアも向上している．2000 回のシミュレーションのもとでの平均スコア 533 542 は，同じ評価関数を用いた expectimax 3-ply search の結果 (平均スコア 394 632[3]) を大きく上回るものであった．この結果は，筆者らの知る限り，PUCT アルゴリズムで 12800 回以上のシミュレーションを用いた AlphaZero プレイヤー [1] 以外のニューラルネットワークプレイヤより優れている．