Vol.2019-GI-42 No.4
2019/7/19

ニューロエボリューションを用いた連鎖型パズルゲーム AI の研究

杉江 矢[1]　橋本 剛[1]

概要：近年，ゲーム AI の汎用機械学習の研究が注目されているが，成功しているゲームはブロック崩しなど単純なものがほとんどである．テトリスは人間にとっては単純に感じるが，汎用機械学習の題材としては難しいことが報告されている．本稿では，連鎖要素を持つパズルゲームであるぷよぷよとパネルでポンを題材とし，ニューロエボリューションを用いた機械学習により，多くの連鎖を行う AI を作成することを目的とする．

キーワード：パズルゲーム，パネルでポン，ぷよぷよ，連鎖，ニューロエボリューション

Research of chained puzzle game AI using Neuroevolution

Nao Sugie[1]　Tsuyoshi Hashimoto[1]

Abstract: In recent years, research on general-purpose machine learning of game AI has attracted attention, but successful games are as simple as Breakout. Although Tetris feels simple to humans, it is reported that it is difficult as a subject of general-purpose machine learning. In this paper, we aim to create an AI that performs many chains by machine learning using neuro-evolution, applying Puyopuyo and Panel de Pon as a chained puzzle game.

Keywords: puzzle game, Panel de Pon, Puyopuyo, chain, Neuroevolution

1. はじめに

近年，ゲーム AI の汎用学習の研究が注目されている．汎用学習の中でも，DeepMind 社の DQN を用いた Atari ゲームの機械学習は有名である [1]．これは，ブロック崩しやピンボール等 49 種類の簡単なゲームの内，半数以上で人間よりも高いスコアを獲得することに成功している．しかし，汎用学習が成功しているゲームは単純なものばかりである．パズルゲームのテトリスは人間にとっては単純に感じるが，青木の汎用学習を用いた研究 [2] では約 8 ラインしか消去できておらず，人間や，学習でない AI と比較すると遠く及ばない結果である．同様に，代表的なパズルゲームであるぷよぷよも汎用学習させることは難しいと考えられる．そこで本稿では，「ぷよぷよ」と「パネルでポン」というゲームに注目する．

パズルゲームというジャンルは「パズル問題を作成する」，「パズル問題を解く」という 2 つの視点から研究の題材として取り扱われることが多い．ぷよぷよを例にすると，逆向き生成法を利用してパズル問題を作成する研究 [3] や，Nested Monte Carlo Search を用いてパズル問題を解く研究 [4] 等が行われている．このように，有名なパズルゲームは様々な手法で研究が行われているが，比較的知名度が低いパネルでポンの研究は少なく，特にパズル問題を解く研究は行われていない．

本研究ではパネルでポンにおいて重要な要素である「連鎖」に注目し，多くの連鎖を行う AI を作成することを目的とする．機械学習の手法として，高田のマリオで裏技を発見する AI の研究 [5] を参考にニューロエボリューションを用いる．また，この手法によりぷよぷよでも連鎖を学

[1] 松江工業高等専門学校
National Institute of Technology, Matsue College

習できるかどうか実験し，パネルでポンの学習結果と比較する．

本稿では，まず 2 章でパネルでポンとぷよぷよの概要を説明する．次に，3 章で学習手法であるニューロエボリューションを解説し，4 章でその実装について説明する．次に，パネルでポンとぷよぷよの実験内容についてそれぞれ 5 章，6 章で述べる．最後に 7 章で本研究のまとめ，8 章で今後の方針を記す．

2. パネルでポンとぷよぷよ

2.1 パネルでポン

2.1.1 概要

パネルでポンとは，1995 年 10 月 27 日に任天堂が発売したスーパーファミコン用のパズルゲームである（図 1）．その後は，ゲームボーイやニンテンドー DS 等のハードで発売されている他，海外にも展開している．日本国外版では主に“Puzzle League”や“Tetris Attack”の名称が使われている．

図 1 パネルでポン
Fig. 1 Panel de Pon

2.1.2 ルール

パネルでポンは，盤面の下部からせり上がってくるパネルを消していくパズルゲームである．パネルが盤面の最上部まで達してしまうとゲームオーバーになる．高得点を獲得するには，連鎖を効率良く行ったり，一度に多くのパネルを消したりする必要がある．一般的な落ち物パズルゲームと違い，パネルが消えている間にも他のパネルを動かすことができ，その間にさらにパネルを消すことができるアクション要素もある．

パネルでポンの特徴を以下に説明する．

- 盤面：横 6 マス縦 12 マスの大きさを持つ盤である．
- パネル：盤面 1 マスにつき 1 枚配置される．本研究で用いる 6 種類のパネルを図 2 に示す．同じ種類のパネルを縦か横に 3 枚以上並べることでそれらのパネルを消すことができる．
- カーソル：横 2 マス縦 1 マスの大きさで盤面上に表示され，カーソル内のパネルを入れ替えることができる．
- 入れ替え：カーソル内のパネルを入れ替える操作を表す（図 3）．入れ替えによってパネルが消滅する例を図 4 に示す．

図 2 6 種類のパネル
Fig. 2 Six types of panel

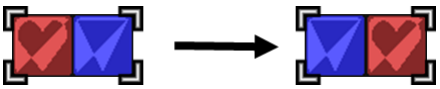


図 3 パネルの入れ替え
Fig. 3 Swap panel

図 4 入れ替えによるパネルの消滅
Fig. 4 Match panel

- 連鎖：パネルが消滅したとき，そのパネルより上にあるパネルは下に落ちてくる．そして，落ちてきたパネルが再び消滅する条件を満たしたとき，これを連鎖と呼ぶ．図 5 に 2 連鎖が発生した例を示す．

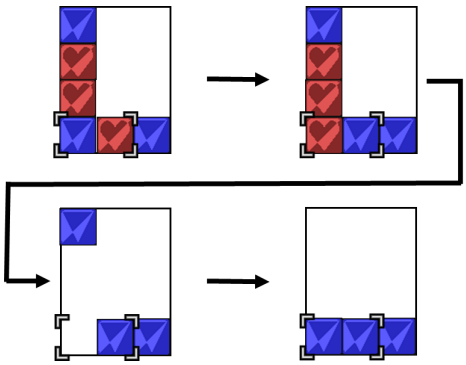


図 5 2 連鎖の例
Fig. 5 Two chain

2.1.3 ぷよぷよとの比較

このゲームがぷよぷよよりも単純だと考える理由は 2 つある．まず 1 つ目は，操作方法である．ぷよぷよでは，一度置いたぷよは動かせないため先を見越しながらぷよを置く必要がある．さらに，置くぷよを回転できる点も便利である一方，操作が複雑化する一因となっている．比べてパネルでポンは，横 2 マスのパネルならどこでも入れ替えることができ，それ以外の特殊な操作は無い．次に 2 つ目は，

パネル（ぷよ）の消し方である．ぷよぷよは，同じぷよを4つ繋げると消えるが，繋げ方の多様性が高いため，慣れないうちは思うように消せない状況になることが多い．一方パネルでポンは，縦か横に3つ以上繋げて消すというパズルゲームの中でも慣れやすいルールとなっている．

2.1.4 関連研究

パネルでポンのパズルモードを対象として，面白いパズル問題をコンピュータに作らせる研究がある [6]．パズルモードとは，盤面上にあらかじめパネルが配置されており，決められた手数ですべてのパネルを消すことが目的のモードである．アルゴリズムを用いてパズル問題を創作した後，力まかせ探索によってその問題が解けるかどうか調べている．しかし，手数とパネルの総数が増えるほど解の探索に莫大な時間がかかっており，手数が5手の問題でも3分近くかかるという結果が出ている．このことから、手数の少ないパズルは力まかせ探索でも解くことができるが，パネルの総数や手数が多い場合は，より効率的な手法を用いる必要があると考えられる．上の研究では数手で解けるパズル問題を作成することを目的としているが，本研究ではあらかじめパネルが配置されたパズルを解くことを目的とする．

2.2 ぷよぷよ

2.2.1 概要

ぷよぷよとは，コンパイルが発売したパズルゲームである（図6）．2個1セットで落ちてくる様々な色のぷよを積んでいき，同じ色のぷよを縦横4つ以上くっつけて消すことで得点が入る．ぷよが消滅したとき，そのぷよより上にあるぷよは下に落ちてくる．そして，落ちてきたぷよが再び消滅する条件を満たしたとき，パネルでポンと同様これを連鎖と呼ぶ．

図 6 ぷよぷよ
Fig. 6 Puyopuyo

3. 学習手法

3.1 マリオ AI

アクションゲームで裏技を発見する AI をニューロエボリューションを用いて作成する研究がある．この研究では，マリオの周囲情報を入力とし，マリオの次の行動を出力するニューラルネットワークを構築している．そして，マリオが敵を倒した時のスコアを報酬として，それを最大化させることで裏技を発見している．この研究を参考に，パネルでポンの連鎖数を報酬とすることによってそれを最大化できると考える．

3.2 ニューロエボリューション

ニューロエボリューションはニューラルネットワークに進化的アルゴリズムを組み合わせた強化学習の手法である．ニューロエボリューションのモデル図を図7に示す．今回はニューラルネットワークとして多層パーセプトロン，進化的アルゴリズムとして遺伝的アルゴリズムを用いる．多層パーセプトロンは，入力層・中間層・出力層からなる順伝播型ニューラルネットワークである [7]．学習が局所解に収束する可能性が高いという問題点があるが [8]，遺伝的アルゴリズムを用いてユニット間の重みを更新することによって，これを防ぐことができる．遺伝的アルゴリズムは，データを遺伝子で表現した個体を複数用意し，適応度の高い個体を優先的に選択して交叉と突然変異を繰り返しながら解を探索する．今回は，ニューラルネットワークのユニット間の重みを遺伝子とする．

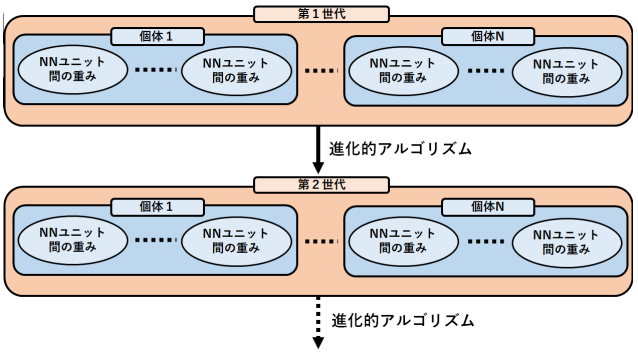


図 7 ニューロエボリューション
Fig. 7 Neuroevolution

4. 実装

本章では，使用したゲームプログラムと作成したエージェントについて説明する．

4.1 パネルでポン

4.1.1 仕様

alexirpan が Web 上で公開しているパネルでポンと同じルールのプログラム[*1]を使用し、これに学習部分のプログラムを追加した．

[*1] https://github.com/alexirpan/Tetris-Attack

4.1.2 エージェントの構築

ニューロエボリューションを実装するにあたって、NEAT-Python ライブラリ[*2]を使用した．このライブラリを用いてゲーム AI を学習させる際は，1 試行の終了条件やエージェントの報酬，ニューラルネットワークの入出力等を設定する必要がある．

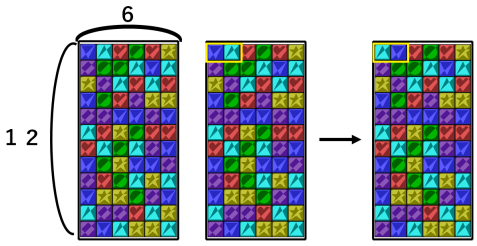
報酬は，多くの連鎖を行わせるため 1 回の試行が終了した時点での最大連鎖数を設定した．また，式 1 に示すスコアも連鎖数を増やすための指標になると考え，最大連鎖数の代わりにスコアを報酬とするエージェントも構築した．

$$\text{スコア} = \text{消したパネル数} \times \text{現在連鎖数}^2 \qquad (1)$$

ニューラルネットワークは，盤面情報を入力すると次に入れ替えるパネルを出力するように構築した，よって，図 8(a) のように全 72 のマス情報を入力とした．また，パネルを入れ替えることのできる全 60 通りを出力とした．パネル入れ替えの例の 1 つを図 8(b) に示す．

60 個の出力の中からルーレット選択により 1 つを選択し，それに対応するパネルを入れ替えた．なお，ルーレット選択とはそれぞれの値の大きさに比例する確率で 1 つを選択するアルゴリズムである．例えば，表 1 のような出力が与えられた場合，A が最も選ばれやすく、C は最も選ばれにくいことを意味する．

表 2 にニューロエボリューションの学習パラメータを示す．



(a) 入力　　(b) 出力例

図 8　ニューラルネットワークの入出力
Fig. 8　Input and output of neuroevolution

表 1　ルーレット選択の例
Table 1　Roulette selection

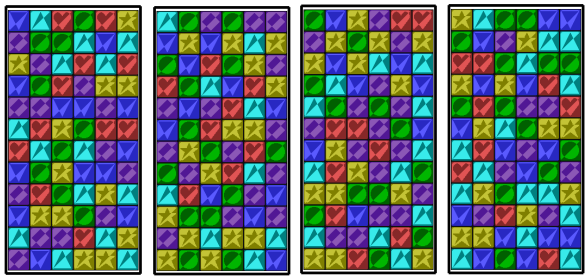
出力名	A	B	C	D
出力値	0.5	0.2	0.1	0.2
選ばれる確率 [%]	50	20	10	20

表 2　学習パラメータ
Table 2　Learning parameter

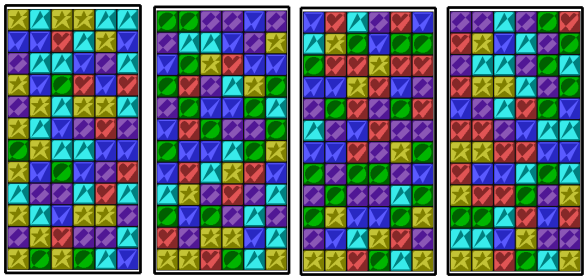
項目	値
入力ユニット数	72[個]
中間ユニット数	10[個]
出力ユニット数	60[個]
個体数	40[個体]
世代数	1000[世代]

4.1.3 盤面

初期盤面は擬似乱数を用いて 6 種類のパネルをランダムに配置して作成した．実験に使用した 8 種類の初期盤面を図 9 に示す．



(a) 盤面 1　(b) 盤面 2　(c) 盤面 3　(d) 盤面 4



(e) 盤面 5　(f) 盤面 6　(g) 盤面 7　(h) 盤面 8

図 9　初期盤面
Fig. 9　Initial board

4.2 ぷよぷよ

4.2.1 仕様

ニューロエボリューションの汎用性を検証するためぷよぷよも実装した．hMatoba が Web 上で公開しているぷよぷよと同じルールのプログラム[*3]を使用し，これに学習部分のプログラムを追加した．

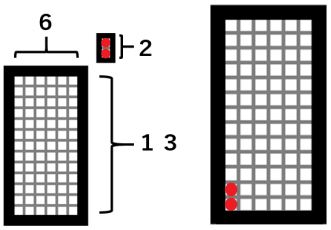
[*2] https://neat-python.readthedocs.io/en/latest

[*3] https://github.com/hMatoba/puyopuyo

4.2.2 エージェントの構築

エージェントは，パネルでポンと同様に最大連鎖数，またはスコアを報酬とした．なお，ぷよぷよのスコアは式 1 の「消したパネル数」を「消したぷよ数」に置き換えた式で求められる．図 10(a) のように盤面 78 マス＋次に置くぷよ 2 個の合計 80 個の情報をニューラルネットワークの入力として与えた．また、ぷよを置くことのできる 11 × 2 の合計 22 個を出力とした．ぷよを置くときの例の 1 つを図 10(b) に示す．

22 個の出力の中からルーレット選択により 1 つを選択し，それに対応する位置にぷよを置いた．表 3 にニューロエボリューションの学習パラメータを示す．



(a) 入力　　(b) 出力例

図 10 ニューラルネットワークの入出力
Fig. 10 Input and outout of neuroevolution

表 3 学習パラメータ
Table 3 Roulette selection

項目	値
入力ユニット数	80[個]
中間ユニット数	10[個]
出力ユニット数	22[個]
個体数	40[個体]
世代数	1000[世代]

4.2.3 盤面

パネルでポンと違い，盤面にぷよが何も無い状態から始め，擬似乱数により落ちてくるぷよの順番を固定した．擬似乱数のパターンはパネルでポン同様 8 つとした．

5. 実験 (パネルでポン)

5.1 実験内容

図 11 のような流れで，最大連鎖数を報酬とする実験と，スコアを報酬とする実験を行った．4.2 章で構築したエージェントを 4.3 章で作成した 8 種類の初期盤面で学習させた．そして，1 世代毎にその世代の中の最大連鎖数と最大スコアを記録した．

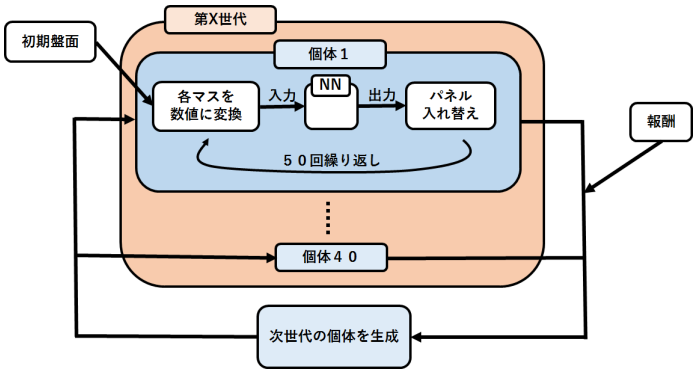


図 11 実験の流れ
Fig. 11 Experiment flow

5.2 実験結果

パネルでポンの最大連鎖数とスコア結果の一部を図 12，13，14，15 に示す．なお，横軸のデータ数が非常に多いため，スコアのグラフは移動平均（区間 20）を用いて，連鎖数のグラフは 10 個おきにデータを抽出してデータ数を減らしている．図に示していない盤面 5，6，7，8 は盤面 3 のように値が上昇していたが，盤面 1，2 は盤面 4 のように値が上昇しなかった．8 つの盤面の全データは[*4]に置く．また，盤面 3 において 4 連鎖が発生している様子を図 16 に示す．

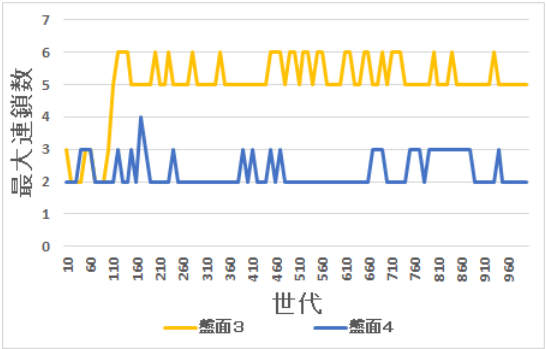


図 12 報酬を連鎖数としたパネルでポンの最大連鎖数（10 世代おきにデータ抽出）
Fig. 12 Maximum number of chains with reward as chain number

[*4] http://www.matsue-ct.ac.jp/home/hashimoto/Panel

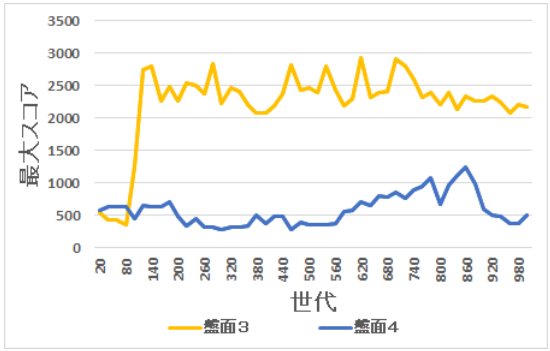


図 13　報酬を連鎖数としたパネルでポンの最大スコア（移動平均，区間 20）

Fig. 13　Maximum score with reward as chain number

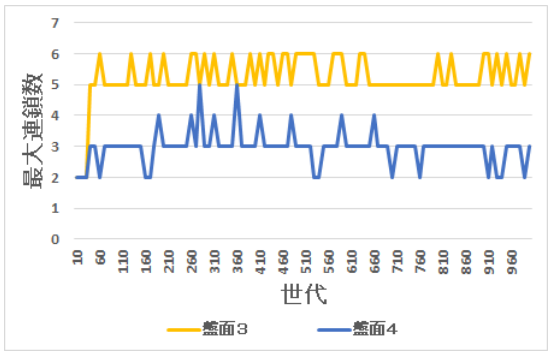


図 14　報酬をスコアとしたパネルでポンの最大連鎖数（10 世代おきにデータ抽出）

Fig. 14　Maximum number of chains with reward as score

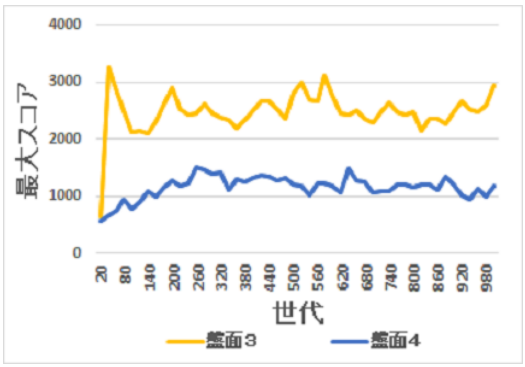


図 15　報酬をスコアとしたパネルでポンの最大スコア（移動平均，区間 20）

Fig. 15　Maximum score with reward as score

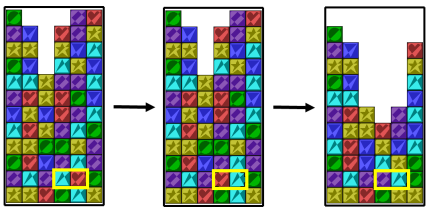


図 16　4 連鎖の様子

Fig. 16　Four chain

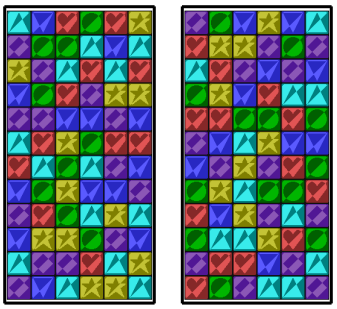
5.3　考察

図 12 から，盤面では最大連鎖数が途中から上昇していることがわかる．逆に，盤面 4 は値が振動しているだけで成長していない．これは，盤面 4 が盤面 3 と比べて，連鎖を発生させるまでの手数が多かったため難しかったのではないかと考えられる．

図 12 から最大 6 連鎖で学習が収束している．しかし人間は 10 連鎖以上できるため，今回の手法では人間を超えるほどの結果は得られなかった．この原因としてニューラルネットワークの入出力が多すぎたのではないかと考えられる．入出力が多い程，ニューラルネットワークの構造は複雑になるため，処理に時間がかかったり理想の解を得られにくかったりする．

また，ニューロエボリューションでは 1 回の試行が終わった後にエージェントが報酬を受け取っている．よって，50 回行ったパネルの入れ替えのうち，どの入れ替えによって連鎖が発生したのか学習できていない．この問題を解決するには，Q 学習のような，ある状態毎に行動の評価値を記録するといった手法が有効である．しかし，パネルでポンは状態数が非常に多いため，盤面をスケールダウンさせるなどして状態数を減らす必要がある．

パネルの色を数値に変換しているため，パネルの色がずれると入力値もずれてしまうという点も問題であると考えられる．例えば図 17(a) と図 17(b) は本質的には同じ盤面のため、同じ出力が得られることが理想だが，実際は異なる出力になる．



(a) 盤面 A　　(b) 色がずれた盤面 A

図 17　本質的に同じ盤面

Fig. 17　Essentially the same board

6.　実験 (ぷよぷよ)

6.1　実験内容

5.1 章と同様に図 18 のような流れで，最大連鎖数を報酬とする実験と，スコアを報酬とする実験を行った．そして，1 世代毎にその世代の中の最大連鎖数と最大スコアを記録した．

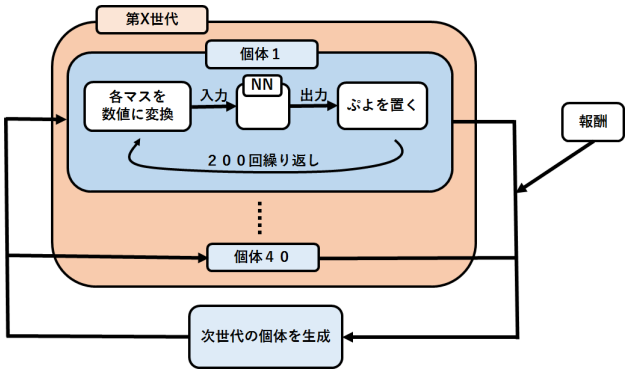
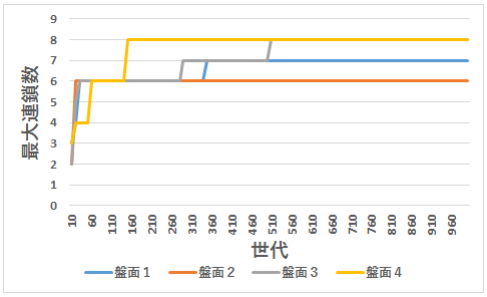


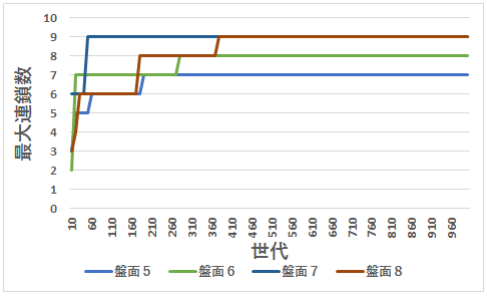
図 18　実験の流れ
Fig. 18　Experiment flow

6.2　実験結果

ぷよぷよの最大連鎖数とスコア結果を図 19，20，21，22 に示す．なお，横軸のデータ数が非常に多いため，スコアのグラフは移動平均（区間 20）を用いて，連鎖数のグラフは 10 個おきにデータを抽出してデータ数を減らしている．また，5 連鎖が発生している様子を図 23 に示す．

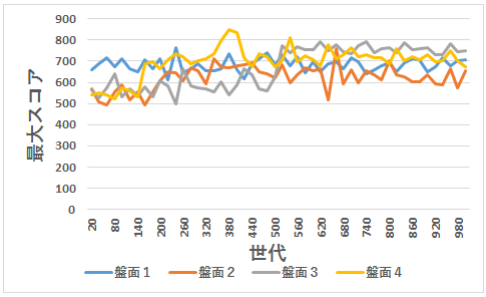


(a) 盤面 1〜4

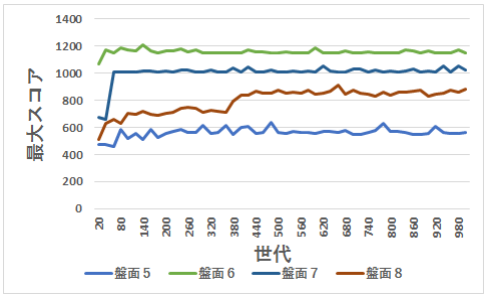


(b) 盤面 5〜8

図 19　報酬を連鎖数としたぷよぷよの最大連鎖数（10 世代おきにデータ抽出）
Fig. 19　Maximum number of chains with reward as chain number

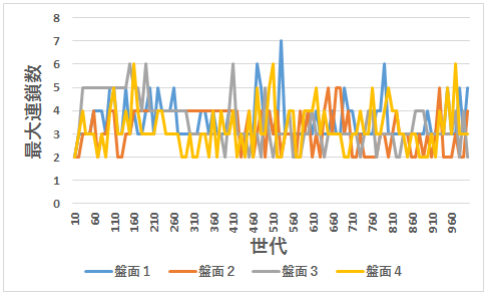


(a) 盤面 1〜4

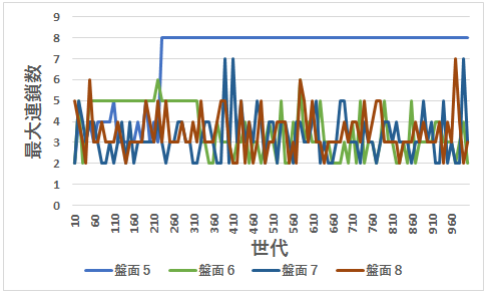


(b) 盤面 5〜8

図 20　報酬を連鎖数としたぷよぷよの最大スコア（移動平均，区間 20）
Fig. 20　Maximum score with reward as chain number

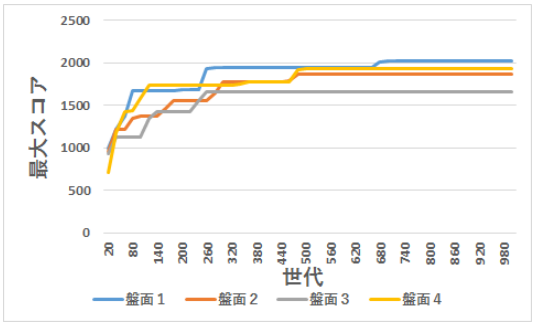


(a) 盤面 1〜4

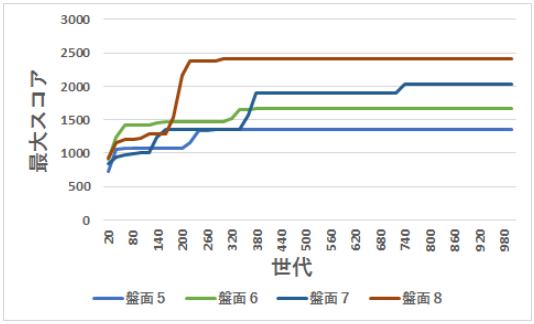


(b) 盤面 5〜8

図 21　報酬をスコアとしたぷよぷよの最大連鎖数（10 世代おきにデータ抽出）
Fig. 21　Maximum number of chains with reward as score



(a) 盤面 1〜4



(b) 盤面 5〜8

図 22 報酬をスコアとしたぷよぷよの最大スコア（移動平均，区間 20）
Fig. 22 Maximum score with reward as score

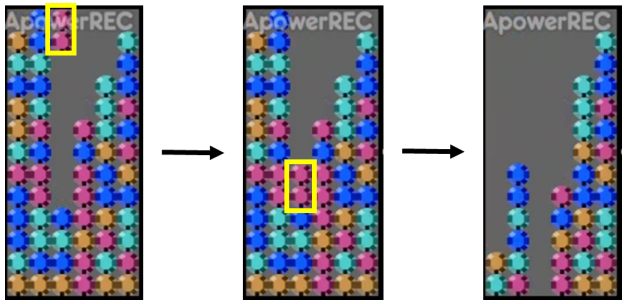


図 23 5 連鎖の様子
Fig. 23 Five chain

6.3 考察

図 19 から，最大連鎖数が上昇していることがわかる．また，図 22 から，スコアが上昇していることがわかる．しかし，図 21 を見ると，盤面 5 を除いて値が振動しているだけである．すなわち，スコアを報酬としてもスコアが上昇するだけで，連鎖を積極的に行わなかったことを意味する．

図 19 から、6〜9 連鎖で学習が収束しているが，大会に出場するような人間や，アルゴリズムによる AI には及ばない結果となった．

研究当初は操作方法やパネルの消し方の単純さの観点から，パネルでポンの方がぷよぷよより連鎖しやすいと考えていたが，逆の結果となった．これは、パネルでポンの方がニューラルネットワークの出力数が多い点が関係している可能性がある．すなわち，2.3 章で述べた，パネルでポンはどこでも入れ替えられるという利点が，逆に機械には選択肢が多過ぎて学習することが難しくなってしまったと考えられる．

7. まとめ

近年注目されている汎用学習は，単純なゲームでしか成功しておらず，連鎖要素のある複雑なパズルゲームの AI は人間がアルゴリズムを考えて作成することが一般的である．そこで，本研究では汎用学習の一つであるニューロエボリューションにより，2 種類のパズルゲームで連鎖を行う AI を作成した．実験の結果，学習の成果が見られないような盤面もあったが，パネルでポンは最大 6 連鎖，ぷよぷよは最大 9 連鎖まで行わせることができ，人間の初心者を超えるレベルの成果を挙げた．

8. 今後の方針

本研究では，固定の盤面に対して実験を行った．今後は，対戦モード等実際のゲームと同様の条件で連鎖が行えるように手法の改良について検討，実装したい．また，パネルでポンとぷよぷよ以外のパズルゲームにも汎用学習による検証を行いたい．

謝辞 本研究は JSPS 科研費 JP17K00514 の助成を受けたものです．

参考文献

[1] Volodymyr Mnih et al : Playing Atari with Deep Reinforcement Learning, ICML 2016, pp. 1-19（2016）
[2] 青木勢馬，橋本剛：テトリスを題材にしたスケールダウンを利用した学習手法の開発，ゲームプログラミングワークショップ 2017 論文集，pp. 99-103(2017)
[3] 高橋竜太郎，池田心：連鎖構成力向上のためのぷよぷよの問題作成，研究報告ゲーム情報学，pp. 1-7(2018)
[4] 齋藤晃介，三輪誠，鶴岡慶雅，近山隆：Nested Monte Carlo Search のぷよぷよへの適用，ゲームプログラミングワークショップ 2013 論文集，pp. 134-137(2013)
[5] 高田亮介，橋本剛：無限 1UP を題材としたアクションゲームの裏技を発見する自己学習手法の提案，研究報告ゲーム情報学，pp. 1-7(2018)
[6] 山崎隆介，Reijer Grimbergen：連鎖型パズルゲームにおけるパズル問題の自動創作，ゲームプログラミングワークショップ 2013 論文集，pp.118-121(2013)
[7] Christopher M.Bishop : Neural Networks for Pattern Recognition, Oxford University Press(1995)
[8] Geoffrey E.Hinton, et al. : Reducing the Dimensionality of Data with Neural Networks, Science 313(2006)