報酬の重みに変化を与えた強化学習

2019SS051 中西重斗
指導教員：佐々木美裕

1　はじめに

最近のオセロや将棋の AI はとても強くなっている．そこには, 強化学習の考え方が用いられており, プロの人をも打ち負かすほど強い．一方で, 勝ち方が安定しておらず, 強い手を打つ場合と弱い手を打つ場合が存在する．本研究では, 報酬の重みに変化を与えることによって, なるべく強い手のみを学習させることを考える．

2　強化学習について

強化学習とは, 機械学習のひとつである．強化学習では,「状態」,「行動」,「報酬」,「エージェント」,「環境」の要素がある．学習主体のエージェントと制御対象の環境が存在する．行動とは, エージェントが観測する現在の環境がどのような状況であるかを示す．報酬とは, エージェントがある行動を起こした場合に得られる報酬のことを示す．エージェントは環境に対して行動し, 環境は, エージェントの行動を評価して報酬を与える．エージェントは, 得た報酬を元に行動を改善する．これらを繰り返すことによって, 報酬を最大化する行動をとるように学習を進める手法のことである [1] [5].

強化学習の使用例としては, ブロック崩しや将棋などのゲーム AI, 客の待ち時間を制御するエレベーターの制御システム, 車が密集している交差点のような難しい条件下における自動運転システムなどがある [3].

3　問題の説明

プログラムを用いた強化学習では, どのような勝ち方に対しても報酬の重みを最大にして学習させるため, 勝ち方に偏りが出やすくなる．そこで, 報酬の重みに変化を与えて, 対戦にどれだけ影響を与えるかを分析するために, 強化学習の一つである Q 学習を用いて三目並べとオセロを Python で実装する．学習をしたデータとランダムに石を置く対戦相手を用意し, 置き方や勝った割合などにどのような影響を与えるかを表や図で表す.

4　学習の説明

パラメータは表 1 のように設定する.

表 1　パラメーター

学習率	0.1
割引率	0.99
ε	0.1〜1
報酬	−100〜100

全ての状態 S_t に対して, 行動 A_t を選んだ先の報酬和の推定値を Q 値と呼び, $Q(S_t, A_t)$ と表す.

Q 値の更新には (1) の式を使用する．オセロでは, 報酬の重みを (2) を使用して変化を与える.

$$Q'(S_t, A_t) = Q(S_t, A_t) + \alpha \{R_t + \gamma \max_{A_{t+1}} Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)\} \tag{1}$$

ここで, γ は学習率, α は割引率である.

オセロの最終局面での石の数に応じた報酬を与えるために, 勝った時と引き分けた時は (2) の式, 負けた時は (3) の式で報酬の重みを計算をした.

$$R_w \times \frac{D_m}{D_o + D_m} \tag{2}$$

$$R_l \times \frac{D_o}{D_o + D_m} \tag{3}$$

ここで, R_wは, 勝った時の報酬 , R_lは, 負けた時の報酬 , D_oは, 相手の石の数 , D_mは, 自分の石の数 である.

また, 学習が進むにつれてランダムに手を選ぶ確率を下げるほうが良いとされているため, 探索手法の一つである ϵ-greedy 法を用いた [2].

ϵ-greedy 法は, 0〜1 の間に乱数を生成し, その値が ϵ 以下であればランダムに行動を選択する．ϵ より大きい場合は, Q 値の大きい行動を選択する.

深さ優先の全探索を行うためにアルゴリズムとして minimax 法を採用した．minimax 法では, 必ず最善手を指すようになるので, 戦略がどれほど改善できたかの指標とする.

以下に学習の流れを記述する (図 2) [4].
ステップ 1：先手は, 後手が打った後に Q 値を更新する.
ステップ 2：後手は, 先手が打った後に Q 値を更新する.
ステップ 3：試合が終了するまで, 報酬は 0 を与える.
ステップ 4：もし, 試合が終了した場合, 結果に応じて報酬を与える.
三目並べとオセロでは, 1〜4 のステップを 1 万回, 10 セット繰り返す．しかし, オセロでは試合の途中でパスが発生するため, パスの場合は Q 値の更新をしない．工夫を行い 6 × 6 オセロの強化学習を行なったが, 10 日間で 1 割ほどの学習であったため, 本研究では, 4 × 4 オセロの学習を採用した.

5　学習結果

本研究では, 報酬の重みに変化を与えて強化学習をしたデータを用いて, ランダムに手を打つ対戦相手（以下, ランダム CPU）と戦わせた.
表 2 では, 勝った場合の報酬と引き分けた場合の報酬の重みに変化を与えてランダム CPU と 10 万回戦った結果である．引き分けた場合の報酬の重みを大きくした場合の方

表 2　報酬の重みの変化とその結果

報酬		割合 (%)		
勝った時	引き分けた時	勝った時	引き分けた時	負けた時
100	0	88.7	11.0	0.2
90	10	92.1	7.6	0.0
80	20	88.2	8.7	3.0
70	30	91.4	8.4	0.3
60	40	88.3	8.4	3.3
50	50	92.6	6.7	0.8
40	60	92.4	7.0	0.5
30	70	94.3	3.4	2.4
20	80	91.0	8.4	0.6
10	90	93.3	5.6	1.1
0	100	94.2	5.8	0.0

がおおよそ勝った割合が 9 割となった．勝った場合の報酬の重みを大きくした場合では，引き分けた場合の報酬の重みを大きくしたときよりも引き分けた割合の方が高い結果となった．

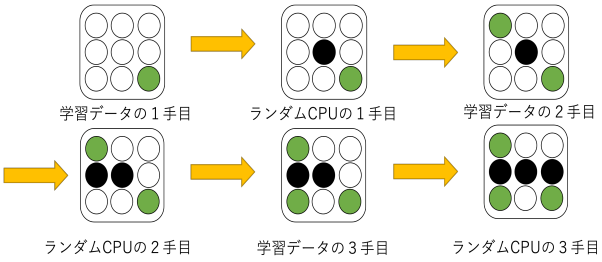

図 1　負けた場合の報酬の重みを大きくした場合の様子

負けた場合の報酬の重みを大きくした場合の打ち手は図 1 のようになった．学習済みデータは，負けた場合の報酬の重みを大きくして学習させたものである．ランダム CPU との対戦では，学習済みデータが相手の不利にならないような位置に石を置いていることがわかる．

表 3　対戦相手のランダムと戦わせた結果

	先手の石の数	後手の石の数
random	7.39	8.79
minimax	11.4	14.52
reward-normal	7.76	10.79
reward-change	8.58	12.42

表 3 は，先手と後手 それぞれでランダム CPU と 100 回ずつ戦ったときの最終局面の石の平均である． ここで，reward-normal は，報酬の重みに変化を与えた場合，reward-change は，報酬の重みを固定した場合を示す．

図 2 と図 3 は，学習した 4 × 4 オセロのデータでランダム CPU と 100 回戦わせた結果である．

6　おわりに

三目並べでは，先手後手関係なく最善手を指すと，引き分けになる．したがって，引き分けた場合の報酬の重みを

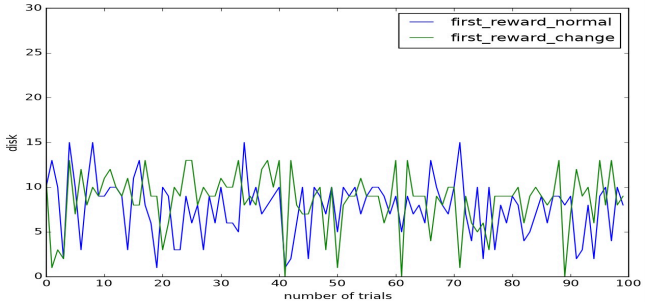

図 2　先手で報酬の重みに変化を与えた場合と与えない場合の比較

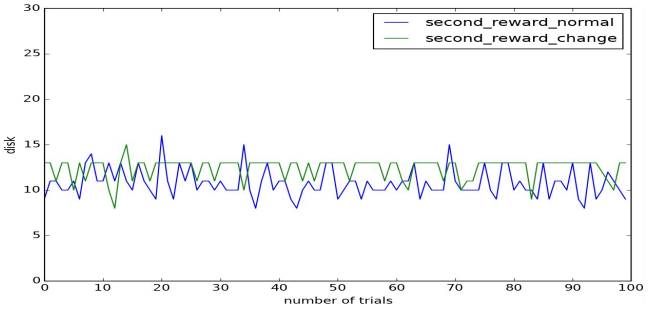

図 3　後手で報酬の重みに変化を与えた場合と与えない場合の比較

大きくしたが，勝った場合の報酬の重みを大きくしたときと比べて，勝った割合，引き分けた割合，負けた割合にそれほど大きな変化が見られなかった．また，負けたときの報酬の重みを大きくした場合，ランダム CPU に対して意図的に負けるような手を指すようになった．

4 × 4 オセロでは，報酬の重みに変化を与えた場合，石の平均の数が先手で 0.82，後手で 1.63 上昇した．先手では，報酬の重みに変化を与えた方が，石の数が極端に少ない場合が多くなった (図 2)．また，後手では，石の数が安定しており，最終局面でより多くの石を得る戦略をとったと見られる (図 3)．

参考文献

[1] 伊藤 真．「強化学習」を学びたい人が最初に読む本．日経 BP，2021 年．

[2] 近藤 康毅．ゼロから作る Deep Learning 4．オーム社，2022 年．

[3] AISmily．強化学習とは？手法や AI ロボットなどの活用事例を紹介，閲覧日： 2022 年 09 月 25 日．https://aismiley.co.jp/ai_news/reinforcement-learning-mechanism-and-examples/

[4] 橋本 洋志，牧野浩二，Python コンピューターシミュレーション入門 人文・自然・社会科学の数理モデル．オーム社，2021 年．

[5] 岡野原太輔 ディープラーニングを支える技術 2 ニューラルネットワーク最大の謎．技術評論社，2022 年．