強化学習を利用したオセロゲームの方策の自動生成

執筆者：杉田　尚士

指導教官：新納　浩幸

平成 14 年 3 月 1 日

目次

図目次

表目次

第１章　はじめに

1.1　概要

近年、コンピュータのめざましい発達により、従来では不可能と考えられてきた手法が実現可能となった．相互作用問題を解くのに利用する強化学習もその１つであり，人工知能などの研究に応用されてきた．

本研究の目的は，大規模な相互作用問題を解くことを通じて強化学習の可能性を探ることである．そのための題材としてオセロゲームの方策作成を選択した．オセロゲームを選択した理由は

- 状態数が非常に多い（$3^{64} \fallingdotseq 3.4 \times 10^{30}$の盤面が想定できる）
- 似たような盤面でも優位性が異なり，汎化が困難である
- 後述するマルコフ性を持ち，強化学習に適用しやすい問題である
- 勝ち負けがはっきりしたゲームなので評価がしやすい

などの理由からである．

1.2　本論文の構成

本論文は強化学習の基本的解法（第２章）に始まり，状態の汎化の手法（第３章），オセロゲームに強化学習を適用するにあたってのモデル化（第４章），方策作成の実験（第５章）および考察（第６章），実験結果に関するまとめ（第７章）と続く．

また，巻末にはプログラムソースを添付した．

第 2 章　強化学習

2.1　相互作用問題と強化学習

2.1.1　相互作用問題とは

ある目的を持ったエージェントが，その目的を達成するために，環境に対してある行動を行う．環境は，その行動に応答し，新たな状況を提示する．このエージェントと環境が相互作用を行っていく問題を相互作用問題と呼ぶ[1].

また，環境は行動に対し報酬を与える．報酬とはエージェントが時間経過の中で最大化しようとする特別な数値でエージェントの持つ目的と密に関係がある．強化学習とは，これら相互作用問題の枠組みの中で，報酬の総量が最大になるように，行動の指針である方策を作成することである．強化学習は，環境との相互作用から学習していく教師なし学習である．

詳細には，相互作用問題では離散的な時間ステップ t=0,1,2,3……の各々において相互作用を行う．各時間ステップ t においてエージェントは何らかの環境の状態 $s_t \in S$（可能な状態の集合）を受け取り，これに基づいて行動 $a_t \in A(s_t)$（s_t において選択できる行動の集合）を選択する．1ステップ後に，エージェントはその行動の結果として数値化された報酬 $r \in R$ を受け取り，新しい状態 s_{t+1}にいる事を知る．

また，各時間ステップにおいて，状態から可能な行動を選択する確率の写像である方策π_tがエージェントには実装されている．ここで，$\pi_t(s_t, a_t)$は s_t=s の時に a_t を選択する確率である．強化学習では受け取る報酬の総量が最大になるように方策πを設計する．エージェントと環境の相互作用の関係を図1

に示す．

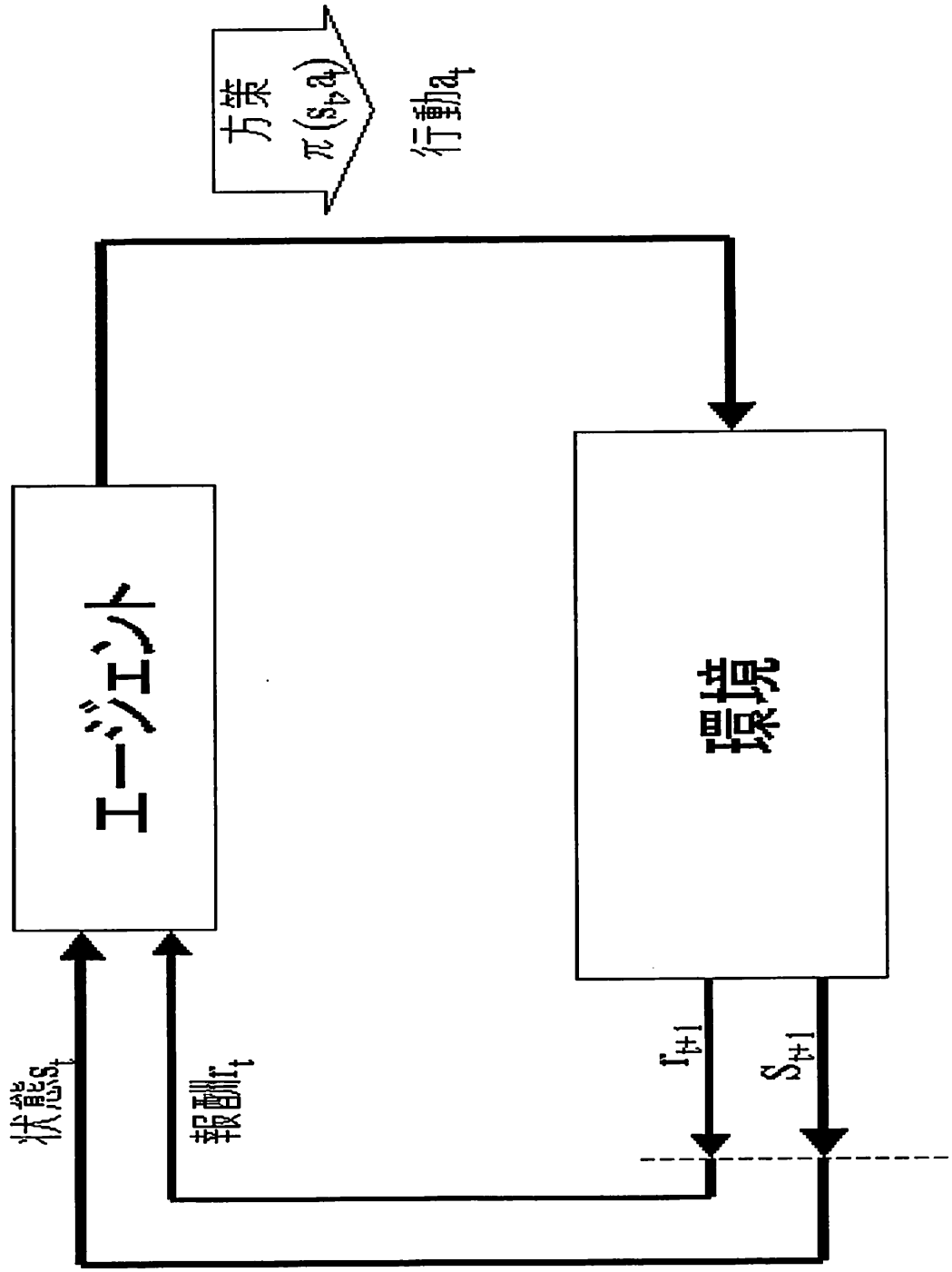

図 2.1　強化学習におけるエージェントと環境間の相互作用

2.1.2 相互作用問題の設計

現実にある問題を相互作用問題に適用するにあたって，いくつかの留意点がある．

まず，環境とエージェントの境界線である．エージェントとは意志決定および行動に関する部分であり，環境はそれ以外全てである．また，エージェントが任意で変更できないものはエージェントの外部要素，つまり環境であると考える．この考えから，報酬もエージェントの外部にあるものと見なす．なぜなら，報酬は行動に対する対価であり，エージェントはそれを操作できないからである．

次に，報酬の設定である．エージェントは報酬の総計を最大にするように意志決定を行うため，何を報酬にするかでエージェントの行動指針が決定される．もし，誤った報酬を設定してしまうと，誤った目的に向かって行動してしまう．また，部分目的に関し報酬を設定すると，最終目的を無視し，あるいは最終目的に反する行動をとってでも部分目標に向かって行動してしまう．報酬信号とはエージェントにどのように目的を達してほしいかではなく，何を達成してほしいかを伝える信号なのである．

2.1.3 収益と割引

エージェントは，相互作用問題の一連の流れ（以下タスク）での最終的な報酬の合計を最大化にするように学習を行う．そのための指針として，ある時間ステップ t の時に，将来得られる報酬の総和 R_t を収益として定義する．

ところで，タスクにはエピソードと呼ばれる部分系列に分解できる物とそうでない物がある．前者をエピソード的タスク，後者を連続タスクと呼ぶ．

エピソード的タスクには終端状態という特殊な状態が存在し，終端状態に達した場合は状態の初期化を行い，次のエピソードに移行される．

連続タスクの場合は，その終端状態が存在しない．そのため，最終時間ステップ $T=\infty$ の時にどのような方策を採っていたとしても収益が無限大になってしまうことが考えられるため，問題が生じる．このため，割引という概念を導入する．割引とは，未来で得る報酬が現在でどれほどの価値を持つかを表す．

$$R_t = r_{t+1} + \gamma r_{t+2} + \gamma r_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \qquad (2.1)$$

ここで，γは割引率と呼ばれるパラメータで$0 \leqq \gamma \leqq 1$である．

強化学習はR_tが最大となるようなa_tを選択するπを作成するのが目的となる．このとき，もしγが0ならば将来の報酬を考えずに即時報酬のみを最大化するような方策を作成する．γが大きくなるにつれて将来的な展望を考慮した方策となる．

2.1.4 マルコフ性

これまで出てきた状態とは，エージェントが利用可能である情報を指す．

状態は，エージェントが知ることのできない情報を持つ必要はない．むしろ，エージェントが知らないことの方が自然な場合はエージェントにその情報を与えるべきでない．一方，ある時間ステップtでエージェントが情報を得るとき，それ以前に得た全ての情報を知っている必要がある．そのため，過去の全ての履歴を保持する情報が望ましい．そのような情報をマルコフ性を持つと呼ぶ．逆にいうと，もしマルコフ性を持っているならば，ある状態s_tとそのときにとった行動a_tがあるときに，次の状態s_{t+1}はそれまでとった$s_1 \sim s_{t-1}$および$a_1 \sim a_{t-1}$に関係なく，s_tとa_tのみで決定される．完全なマルコフ性を持つ情報はなかなか存在しないが，不完全なマルコフ性でも，凶科学種を解くことは可能である．

2.1.5 関数の定義

実際に強化学習の手法を扱うにあたって，表現の簡略のためにいくつかの関数を定義する．

- 遷移確率

  任意の状態s，行動aが与えられた時に，次の状態s'に遷移する確率

$$P_{ss'}^{a} = \Pr\{s_{t+1} = s' \mid s_t = s, a_t = a\} \qquad (2.2)$$

・ 報酬の期待値

任意の状態 s，行動 a が与えられた時の，次の報酬 r_t の期待値

$$R^a_{ss'} = E\{r_{t+1} = s' \mid s_t = s, a_t = a, s_{t+1} = s'\} \tag{2.3}$$

・ 状態価値関数

エージェントが状態 s にいて方策 π に従うときの期待収益

$$V^\pi(s) = E_\pi\{R_t \mid s_t = s\}a = E_\pi\left\{\sum_{k=0}^{\infty}\gamma^k r_{t+k+1} \mid s_t = s\right\} \tag{2.4}$$

$$= \sum_a \pi(s,a)\sum_{s'} P^a_{ss'}\left[R^a_{ss'} + \gamma V^\pi(s')\right] \tag{2.5}$$

・ 行動価値関数

エージェントが状態 s にいて行動 a をとった後方策 π に従うときの期待収益

$$Q^\pi(s) = E_\pi\{R_t \mid s_t = s, a_t =\}a = E_\pi\left\{\sum_{k=0}^{\infty}\gamma^k r_{t+k+1} \mid s_t = s, a_t = a\right\} \tag{2.6}$$

$$= \sum_{s'} P^a_{ss'}\left[R^a_{ss'} + \gamma V^\pi(s')\right] \tag{2.7}$$

2.2 動的計画法

動的計画法は強化学習の古典的解法の１つであり，環境がマルコフ性を持ち，なおかつ環境の完全なモデルが与えられているときに適用できる．

動的計画法は，方策の評価と改善を繰り返し行うことによって最適方策を得る．

2.2.1 方策評価

任意の方策 π のもとでの状態価値関数 $V^\pi(s)$ を求めるには式(2.5)を用いればよい．しかし，式(2.5)をそのまま解くと計算に時間のかかる場合がある．そのため，反復解法を用いて $V^\pi(s)$ を求める．その方法を図(2.2)に示す．このアルゴリズムを反復方策評価と呼ぶ．

評価対象の方策πを入力
全ての$s \in S$に対して$V(s)=0$とする
繰り返し：
$\Delta \leftarrow 0$
各$s \in S$について：
$v \leftarrow V(s)$
$V(s) = \sum_a \pi(s,a) \sum_{s'} P^a_{ss'} \left[R^a_{ss'} + \gamma V^\pi(s') \right]$
$\Delta \leftarrow \max(\Delta, |v - V(s)|)$
$\Delta < \theta$（正の小さな値）ならば，繰り返しを終了
$V \approx V^\pi$を出力

図2.2　反復評価方策のアルゴリズム

2.2.2　方策改善

ある状態sで，$Q^\pi(s,a) \geqq V^\pi(s)$が成り立つならば，$s$の時には常に行動$a$をとった方が$\pi$に従うより良いと言える．従って，そのような$a$を選択していくことによって決定論的方策を求めることができる．

これより，ある方策とその価値関数を与えられたときに，全ての状態で，全ての可能な行動に関して，各状態で$Q^\pi(s,a)$が最良となる行動を選択するような方策π'を作成すれば，それは元の方策πより良い方策になると考えられる．この方策π'をグリーディ方策と呼び，次式で与えられる．

$$\pi' = \arg\max_a Q^\pi(s,a)$$

$$= \arg\max_a E\left\{ r_{t+1} + \gamma V^\pi(s_{t+1}) \mid s_t = t, a_t = a \right\} \tag{2.8}$$

$$= \arg\max_a \sum_{s'} P^a_{ss'} \left[R^a_{ss'} + \gamma V^\pi(s') \right] \tag{2.9}$$

ここでargmax_aは，それに続く式を最大にするようなaを与える．

2.2.3 方策反復

V^{π}を使って方策πを改善し,より優れた方策π'を得ることができたならば,続いてπ'から $V^{\pi'}$を計算して改善を行うことで，さらに優れた方策π"を得ることができる．このように，方策評価と方策改善を繰り返し行うことによって最適方策を導く方法を方策反復と呼ぶ．方策反復の完全なアルゴリズムを図 2.3 に示す．

1. 初期化

全ての $s \in S$ に対して $V(s) \in R$ と $\pi(s) \in A(s)$ を任意に初期化

2. 方策評価

$\Delta < \theta$（正の小さな数）になるまで繰り返し：

$\Delta \leftarrow 0$

各 $s \in S$ について：

$$v \leftarrow V(s)$$

$$V(s) = \sum_{a} \pi(s,a) \sum_{s'} P^{a}_{ss'} \left[R^{a}_{ss'} + \gamma V^{\pi}(s') \right]$$

$$\Delta \leftarrow \max\left(\Delta, \left| v - V(s) \right|\right)$$

3. 方策改善

polivy-stable←true

各 $s \in S$ について：

b←$\pi(s)$

$$\pi(s) \leftarrow \arg\max_{a} \sum_{s'} P^{a}_{ss'} \left[R^{a}_{ss'} + \gamma V^{\pi}(s') \right]$$

もし，b≠$\pi(s)$ならば *polivy-stable*←false

もし，*polivy-stable* ならば，終了；

それ以外は，2 から繰り返す

図 2.3　方策反復のアルゴリズム

2.2.4 価値反復

方策反復は各繰り返しステップの中で方策評価を必要としている．方策評価は状態空間に対するスイープ操作を何度も必要とする反復計算なので，計算量が増大してしまう．

価値反復は，方策評価と方策改善の２つのスイープ操作を，１回のスイープ操作の中に効果的に結合し計算量を減らしている．価値反復のアルゴリズムを図 2.4 に示す．

全ての $s \in S$ に対して $V(s) \in R$ を任意に初期化

繰り返し：

$\Delta \leftarrow 0$

各 $s \in S$ について：

$$v \leftarrow V(s)$$

$$V(s) = \sum_a \pi(s,a) \sum_{s'} P^a_{ss'} \left[R^a_{ss'} + \gamma V^\pi(s') \right]$$

$$\Delta \leftarrow \max\left(\Delta, \left| v - V(s) \right|\right)$$

$\Delta < \theta$（正の小さな数）ならば，繰り返しを終了

次式の決定論的方策 π を出力

$$\pi(s) = \arg\max_a \sum_{s'} P^a_{ss'} \left[R^a_{ss'} + \gamma V^\pi(s') \right]$$

図 2.4　価値反復のアルゴリズム

2.3 モンテカルロ法

モンテカルロ法とは，環境の完全な知識を必要とせず，経験を利用して価値関数の推定と最適方策の発見を行う．また，シミュレーション上の経験を用いて学習することができる．モンテカルロ法で解くことができるのはエピソード的タスクに限定される．

2.3.1 初回訪問 MC 法

方策πに従って状態 s を通過することによって得られるエピソードの集合が与えられたときに，πのもとでの $V(s)$を推定しようとする．エピソード群の各エピソードの中で初めて s が発生したとき（以下初回訪問）の収益の平均をとると，s への初回訪問回数が無限大に近づくにつれて収益の平均も $V^{\pi}(s)$に収束する．この方法により $V^{\pi}(s)$を推定する方法を初回訪問 MC 法と呼ぶ．初回訪問 MC 法のアルゴリズムを図 2.5 に示す．

初期化：

π←評価すべき方策

V←任意の状態価値関数

Returns(s)←空のリスト（全ての $s \in S$ に対して）

繰り返し：

(a) πを用いてエピソードを生成する

(b) エピソード中に出現する各状態 s について：

R←s の初回発生後の収益

R を *Returns(s)*に追加(append)する

$V(s)$←average(*Returns(s)*)

図 2.5　V^{π}推定のための初回訪問 MC 法

2.3.2 モンテカルロ ES 法

動的計画法においての方策反復の考え方を応用し，モンテカルロ法を用いて方策作成を行う．つまり，初回訪問 MC 法にて推定した $V(s)$を用いて新たな方策πを生成する．モンテカルロ法において最適方策を生成するためには，全ての $s \in S$ が無限回のエピソード中に無限回現れなくてはならない．モンテカルロ ES 法では，この条件を満たすために開始点探査を行う．開始点探査とは，全ての $s \in S$ がエピソードの開始点として選ばれる確率がゼロでないことを指定する．モンテカルロ ES 法のアルゴリズムを図 2.6 に示す．

全ての $s \in S$, $a \in A(s)$に対して初期化：

$Q(s,a)$←任意の値

Returns(s,a)←空のリスト

繰り返し：

(c)開始点探査とπを用いてエピソードを生成する

(d) エピソード中に出現する各 s,a の対について：

R←s,a の初回発生後の収益

Rを *Returns(s,a)*に追加(append)する

$Q(s,a)$←average(*Returns(s,a)*)

(e)エピソード中の各 s について

$$\pi(s) = \arg\max_a Q(s,a)$$

図 2.6　モンテカルロ ES 法のアルゴリズム

2.3.3　その他のモンテカルロ法

開始点探査の仮定は非現実的なので，他の手段によって全ての $s \in S$ を訪れる手法が必要となる．

1つ目は方策オン型手法と呼ばれ，全ての $s \in S$ と $a \in A(s)$に対して$\pi(s,a) > 0$であるソフトな方策を用いる．一般的には確率εで行動をランダムに選ぶε－グリーディ方策を用いる．

2つ目は方策オフ型手法と呼ばれ，生成された方策とは別の方策を用いてエピソードを生成する手法である．

図 2.7 に方策オン型手法であるεソフト方策オン型モンテカルロ制御アルゴリズム，図 2.8 に方策オフ型モンテカルロ制御アルゴリズムを示す．

全ての $s \in S$, $a \in A(s)$に対して初期化：

$Q(s,a)$←任意の値

$Returns(s,a)$←空のリスト

π←任意の εソフト方策

永久に繰り返し：

(a) πを用いてエピソードを１つ生成する

(b) エピソード中に出現する各 s,a の対について：

R←s,a の初回発生後の収益

Rを $Returns(s,a)$に追加(append)する

$Q(s,a)$←average($Returns(s,a)$)

(c) エピソード中の各 sについて

$$a^* = \arg\max_a Q(s,a)$$

すべての $a \in A(s)$について

$$\pi(s,a) \leftarrow \begin{cases} 1-\varepsilon+\varepsilon/|A(s)| & (a=a^*) \\ \varepsilon/|A(s)| & (a \neq a^*) \end{cases}$$

図 2.7　εソフト方策オン型モンテカルロ制御アルゴリズム

全ての $s \in S$, $a \in A(s)$に対して初期化：

$Q(s,a)$←任意の値

$N(s,a)$←0（$Q(s,a)$の分子）

$D(s,a)$←0（$Q(s,a)$の分母）

π←任意の決定論的方策

永久に繰り返し：

(a) 任意のソフト方策π'を選択し，それを用いて1個のエピソードを生成する：

$s_0, a_0, r_1, s_1, a_1, r_2, \cdots, s_{T-1}, a_{T-1}, r_T, s_T$

(b) τ←もっとも最近 $a_\tau \neq \pi(s_\tau)$となった時刻

(c) 時刻 τあるいはそれ以降にエピソード中に出現した各 s，a の対について：

$t \leftarrow t \geqq \tau$であるような s，a が最初に発生した時刻

$$w \leftarrow \prod_{k=t+1}^{T-1} \frac{1}{\pi'(s_k, a_k)}$$

$$N(s,a) \leftarrow N(s,a) + wR_t$$

$$D(s,a) \leftarrow D(s,a) + w$$

$$Q(s,a) \leftarrow N(s,a)/D(s,a)$$

(d) エピソード中の各 s について

$$\pi(s) = \arg\max_a Q(s,a)$$

図 2.8　方策オフ型モンテカルロ制御アルゴリズム

2.4 TD 学習

時間的差分学習（以下 TD 学習）は，強化学習の中でも比較的新しい考え方で，モンテカルロ法で用いた経験からの学習と，動的計画法で用いたブートストラップ手法を組み合わせた考え方である．

2.4.1 TD 予測

モンテカルロ法は，各訪問に対応する収益がわかるまで待ち，その値を $V(s)$ の目標値として利用する．つまり，モンテカルロ法での $V(s)$の目標値は R_t であり，$V(s)$の更新式は式(2.10)の通りである．

$$V(s_t) \leftarrow V(s_t) + \alpha[R_t - V(s_t)] \tag{2.10}$$

ここで R_tは時刻 t に対応する実際の収益値で，定数 αは 1 回の学習時の更新量を表すステップサイズ・パラメータである．モンテカルロ法では R_t を計測するため，1 回のエピソードを終了するまで $V(s)$の更新を行うことはできない．

一方，TD 予測では動的計画法同様，$V(s_{t+1})$を利用した値を用いて $V(s)$を更新するブートストラップ手法である．具体的には，TD 予測では $r_{t+1} + \gamma V(s_{t+1})$ を $V(s)$の目標値とする．TD 予測における $V(s)$の更新式は式(2.11)の通り．

$$V(s_t) \leftarrow V(s_t) + \alpha[r_{t+1} + \gamma V(s_{t+1}) - V(s_t)] \tag{2.11}$$

2.4.2 TD 学習

TD 予測を用いた方策生成は，動的計画法やモンテカルロ法同様，方策の価値を計算し，その価値から新たな方策を生成するという手法を用いる．また，方策を直接生成できるように $V(s)$ではなく $Q(s,a)$を予測するように変更する．

図 2.9 に方策オン型 TD 制御アルゴリズムである Sarsa，図 2.10 に方策オフ型 TD 制御である Q 学習のアルゴリズムを示す．

$Q(s,a)$を任意に初期化

各エピソードに対して繰り返し：

sを初期化

Qから導かれる方策（Qに対するεグリーディ方策など）

を用いてsでの行動aを選択する

エピソードの各ステップに対して繰り返し：

行動aを取り，r，s'を観測する

Qから導かれる方策（Qに対するεグリーディ方策など）

を用いてs'での行動a'を選択する

$$Q(s,a) \leftarrow Q(s,a) + \alpha\left[r + \gamma Q(s',a') - V(s,a)\right]$$

$s \leftarrow s'$; $a \leftarrow a'$;

sが終端状態ならば繰り返しを終了

図 2.9　Sarsa のアルゴリズム

$Q(s,a)$を任意に初期化

各エピソードに対して繰り返し：

sを初期化

エピソードの各ステップに対して繰り返し：

Qから導かれる方策（Qに対するεグリーディ方策など）

を用いてsでの行動aを選択する

行動aを取り，r，s'を観測する

$$Q(s,a) \leftarrow Q(s,a) + \alpha\left[r + \gamma \max_{a'} Q(s',a') - V(s,a)\right]$$

$s \leftarrow s'$;

sが終端状態ならば繰り返しを終了

図 2.10　Q 学習のアルゴリズム

第３章　汎化およびその手法

3.1　汎化

3.1.1　汎化とは

前章では，状態あるいは状態行動対の１つに対して１つのエントリーが対応するようなテーブル形式の価値関数を扱ってきた．この方式は，わかりやすくはあるがデータ量が膨大となってしまい，テーブルを格納するメモリの問題だけではなく，テーブルを正確に埋め尽くすだけの計算量が問題になってしまう．

汎化とは，その問題を解決する方法の１つで，状態空間内での限定された部分集合での経験をより広い状態空間での部分集合に対する近似として扱おうとする．特に強化学習では，以前に経験した状態をまだ経験していない状態へと一般化する手法である，

3.1.2　関数近似による汎化

関数近似とは目標関数（たとえば価値関数）から実例を取り出し，そこから関数全体の近似を作り出そうとする．また，学習時に教師データを使う教師あり学習である．

関数近似による汎化は強化学習に限らず様々な分野で研究されている汎化手法であり，他分野で研究されている手法を強化学習に容易に取り込むことが可能である．

ここでは，目標関数にニューラルネットワークを用いた関数近似を用いる．

3.2 ニューラルネットワーク

3.2.1 ニューラルネットワークとは

ニューラルネットワークとは，生物の神経系の特徴を模した数学的モデルであり，主に各ユニットが入力から出力まで全て順方向に配置されている構造を用いる[2]．そのようなニューラルネットワーク階層構造ニューラルネットワークという．階層構造ニューラルネットワークの構造を図 3.1 に示す．

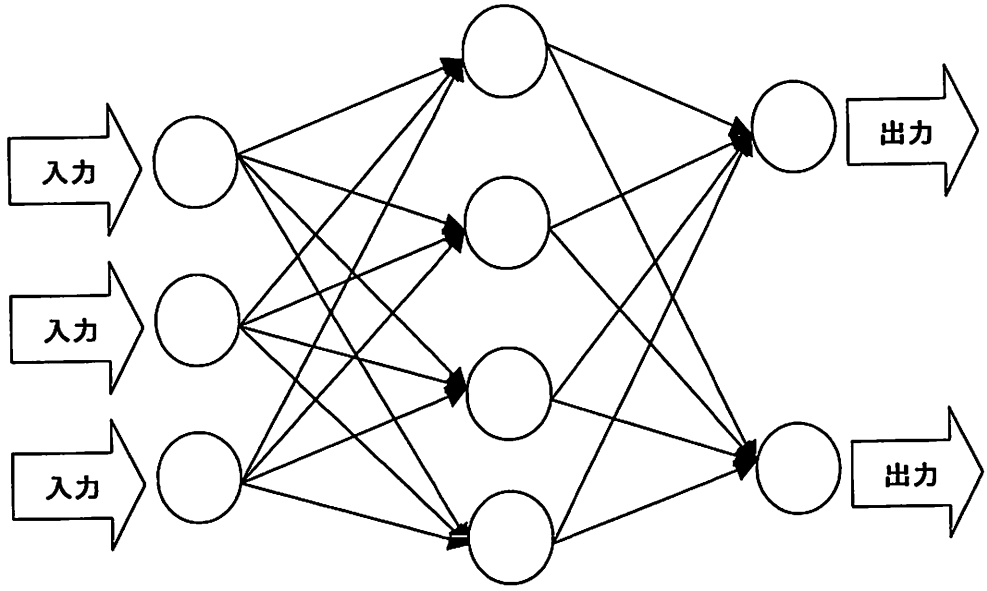

図 3.1　階層構造ニューラルネットワーク

ユニットにあるしきい値を越えた入力があった際に，各ユニットはパルスを放出すると考える．そのため，各ユニットからの出力にはシグモイド関数と呼ばれる以下の式(3.1)を適用し，出力が 0 か 1 になるようにする．

$$f(x) = \frac{1}{1+\exp(-x)} \tag{3.1}$$

ニューラルネットワークは，入力データとそれに対応する理想的と考えられる出力データを与え，実際の出力との差からユニット間の結合強度を変更する教師あり学習である．

3.2.2 逆誤差伝搬法

逆誤差伝搬法（BackPropagation 法，以下 BP 法）は，階層構造ニューラルネットワークの学習方法の中ではもっとも使われていると言われている手法である．

まず，対象とするニューラルネットワークの第 *s-1* 層第 *j* ユニットの出力を y_j，第 s 層第 i ユニットの出力を x_iの時に，各ユニットの入出力関係を以下の式で定義する．

$$x_i = h(z_i) \tag{3.2}$$

$$z_i = \sum_j w_{ij} y_j \tag{3.3}$$

ここで，$h(x)$は式(3.1)で与えられるシグモイド関数であり，w_{ij}は第 *s-1* 層第 *j* ユニットと，第 *s* 層第 *i* ユニットとの結合強度である．

また，第 *s* 層が出力層だとすると，x_iに対応する教師信号 d_iを用いて，出力と教師信号との誤差を次の式で求めることができる．

$$E(w) = \frac{1}{2}\sum_i (x_i - d_i)^2 \tag{3.4}$$

BP 法では，誤差関数 $E(w)$を最小化するため，$E(w)$の勾配ベクトルである式(3.5)を計算し，その逆方向に w を改良するといういわゆる最急降下法を用いる．

$$\frac{\partial E(w)}{\partial w_{ij}} = \frac{\partial E(w)}{\partial z_i}\frac{\partial z_i}{\partial w_{ij}} = \frac{\partial E(w)}{\partial x_i}\frac{\partial x_i}{\partial z_i}\frac{\partial z_i}{\partial w_{ij}} \tag{3.5}$$

ここで，式(3.2)より，$\frac{\partial z_i}{\partial w_{ij}} = y_j$，$\frac{\partial x_i}{\partial z_i} = h'(z_i) = h(z_i) * [1 - h(z_i)]$なので，表記の簡略化のため$\delta_i = \frac{\partial E(w)}{\partial x_{ip}} h'(z_i)$とすると，式(3.5)は以下の通りになる．

$$\frac{\partial E(w)}{\partial w_{ij}} = \delta_i y_j \tag{3.6}$$

δ_i について考える．s が出力層の場合は式(3.4)より $\frac{\partial E(w)}{x_i} = x_i - d_i$ となるので

$$\delta_i = (x_i - d_i)h'(z_i) \tag{3.7}$$

となる．第 s 層が出力層でない場合には，第 s 層第 i ユニットからの出力が変化すると，第 $s+1$ 層の各ユニット（添え字を k とする）への入力の総量 z_k が変化し，第 $s+1$ 層の各ユニットからの出力も変化する．その変化が第 $s+2$ 層以降のユニットにも伝わる．このことより

$$\frac{\partial E(w)}{\partial x_i} = \sum_k \frac{\partial E(w)}{\partial z_k}\frac{\partial z_k}{\partial x_i} \tag{3.8}$$

となる．$\frac{\partial E(w)}{\partial z_k}$ について考えると，第 $s+1$ 層第 k ユニットへの入力の総量 z_k が変化すると，そのユニットからの出力の総量 x_k を変化させ，$E(w)$を変化させる．それゆえ，

$$\frac{\partial E(w)}{\partial z_k} = \frac{\partial E(w)}{\partial x_k}\frac{\partial x_k}{\partial z_k} = \frac{\partial E(w)}{\partial x_k}h'(z_k) = \delta_k \tag{3.9}$$

となる．また，式(3.3)より $\frac{\partial z_k}{\partial x_i} = w_{ki}$ となるので，以上より s が出力層で内場合の δ_i は式(3.10)の通りになる．

$$\delta_i = h'(z_i)\sum_k \delta_k w_{ki} \tag{3.10}$$

式(3.7)，(3,10)より全ての δ を求められるので，式(3.6)を用いることによって各勾配ベクトルを求めることができる．結合ベクトル w_{ij} の更新式はステップ定数 α と勾配ベクトルを用いた以下の式で表現される．

$$w^{(k+1)} = w^{(k)} - \alpha \frac{\partial E(w)}{\partial w}\Big| w = w^{(k)} \tag{3.11}$$

3.2.3 強化学習への適用

ニューラルネットワークを用いて強化学習の関数近似による汎化を行うには，ニューラルネットワークの入力を状態，出力を価値関数の出力値，教師信号を強化学習での価値関数の目標値として設計すればよい．表 3.1 に強化学習の３つの手法それぞれの価値関数の目標値を示す．

表 3.1　価値関数の目標値

動的計画法	$E_\pi\left\{r_{t+1}+\gamma V^\pi(s_{t+1}) \mid s_t = s\right\}$
モンテカルロ法	R_t
TD 学習	$r_{t+1}+\gamma V_t(s_{t+1})$

しかし，ニューラルネットワークを用いた汎化を行う際にはいくつかの留意点がある．

まず，ニューラルネットワークではシグモイド関数を通して出力されるため，0 から 1 の値しか出力することができない．それゆえ，与える報酬を考慮しなければならない．たとえば，ステップごとに 1 あるいは-1 を与えるようなタスクは，ニューラルネットワークを用いた汎化には不向きである．

次に，入力層の設計だが，起こりえる状態を全て表現できるような物でなくてはならない．また，入力も 0 から 1 しか扱えない．そのため，離散的な状態を扱おうとするならば，状態数が n あるとすると $\log_2 n$ の入力ユニットを設計する必要がある．

最後に，中間層だが，理論上は非常に多い中間層を設計すれば精密な近似が可能なのだが，中間層が多すぎると計算量が増大して処理が遅くなってしまう．また，必要以上に中間層があると一度の学習での変化が少なくなり，近似解が求まるまでの学習量も増大してしまう．そのため，２重に時間がかかってしまう．適切な中間層の数を求めるには，いくつかの目安があるとはいえ，結局の所試行錯誤を行う以外にはない．

第4章　オセロゲーム

4.1　オセロゲームのルール

オセロゲームとは，2人で行う対戦型ボードゲームである．詳細なルールは以下の通り．

- 8マス四方の格子状平面上でゲームを行う．
- 2人のプレイヤーが黒と白に別れる．
- プレイヤーは自分の手順の時に自分の色のコマをゲーム盤面に1つ置く．
- 自分のコマで相手のコマを縦，横，斜め方向に挟んだとき，相手のコマをひっくり返して自分のコマにする．
- 1つも相手のコマをひっくり返せないときは，そこの場所にコマを置くことができない．
- どこにもコマを置けないときは，自分の順番がとばされる．
- 盤面が全てコマで埋まるか，お互いにどこにもコマが置けなくなった時点でゲームが終了する．
- ゲーム終了時，盤面に自分のコマが多い方が勝ちである．
- ゲームの初期状態は，左上を(0,0)と考えた座標で座標(3,3))，(4,4)に黒コマ，(3,4)，(4,3)に白コマを置く．

4.2 動的計画法による方策の生成

4.2.1 オセロゲームのモデル設計

オセロゲームの方策を動的計画法で生成するためのモデル設計を行う．まず，問題となるのはオセロゲームは盤面の状態が 3^{64}（１つのマスにつき何もない，黒コマがある，白コマがあるの３通り，それが６４マスある）$\fallingdotseq 3.4\times 10^{30}$ の状態があると予想される．そのため，動的計画法をそのまま適用するのは難しい．そのため，４マス四方の縮小された盤面について考える．この場合の状態数は $3^{16}\fallingdotseq 4.3\times 10^{7}$ であり，実現可能な数字である．また，初期状態はそれに併せて(1,1)，(2,2)に黒コマ，(1,2)，(2,1)に白コマが置かれた状態とする．表 4.1 に動的計画法を用いるためのオセロゲームのモデルを示す．

表 4.1　動的計画法のためのオセロゲームのモデル

状態 s	盤面を数列化した物 何もない　：0 黒コマがある：1 白コマがある：2 最上位桁に，どちらの手順であるかを表記 黒の手順　：1 白の手順　：2 １７桁の３進数で表現 $10000000000000000_{(3)}$～$22222222222222222_{(3)}$
行動 a	コマを置いた場所の座標 (0,0)～(3,3)
報酬 r	ゲーム終了時に勝利した側に＋１ 敗北した側に－１ ゲーム中は常に０の報酬
方策 π	価値反復から求まる状態価値関数 $V(s)$によるグリーディ方策

また，オセロゲームは行う対戦型ゲームであり，自分の手順である状態の次の状態は必ず相手の手順である状態となる．また，相手にとって価値の高い状態は自分にとって価値の低い状態であると言える．そのため，価値反復

の計算時に用いる次の状態価値 $V(s_{t+1})$はそのまま用いず，－１を乗じた値を用いる事にした．

4.2.2 アルゴリズム

動的計画法では完全な状態の遷移を必要とするので，まず状態の遷移を書き出す必要がある．そのためのアルゴリズムは図 4.1 の通り．ルーチンＡを再帰的に呼び出すことによって，全ての状態遷移を書き出している．

```
状態を初期化
ルーチンＡ：
  現在の状態 p を記憶する
  全てのマス h について：
    h≠0 のとき　ルーチンＣを呼び出した後，終了
    h＝0 のとき
      ルーチンＢを呼び出す
      B=true ならば
        現在の状態，取った行動，遷移後の状態を書き出す
        遷移後の状態を現在の状態として，ルーチンＡを呼び出す
      B＝false ならば　盤面はそのまま相手の手順の状態へ移行
ルーチンＢ：
  現在の盤面 p とコマ h について
    h にコマを置いたときに
      ひっくり返る相手のコマがあれば true を返す
      １つもひっくり返せないならば false を返す
ルーチンＣ：
  現在の状態と終端状態であることを示すコード，
  どちらが勝ったかを書き出す
```

図 4.1　状態遷移書き出しのためのアルゴリズム

次に，書き出された状態遷移を元に価値反復による強化学習を行う．そのアルゴリズムを図 4.2 に示す．

全ての $V(s) \mid \forall s \in S$ を 0 で初期化

繰り返し：

$\Delta = 0$

各 $s \in S$ について：

s が終端状態ならば

黒の勝利の場合　$V(s)=1$

白の勝利の場合　$V(s)=-1$

引き分けの場合　$V(s)=0$

s が終端状態でなければ

$v \leftarrow V(s)$

$$V(s) \leftarrow (-1) \times \max_a \sum_{s'} V(s')$$

$$\Delta \leftarrow \Delta, |v - V(s)|$$

$\Delta < 0.01$ ならば，繰り返しを終了

次式の決定論的方策 π を出力

$$\pi(s) = (-1) \times \arg\max_a \sum_{s'} V(s')$$

図 4.2　価値反復によるオセロゲームの方策の出力

4.3 TD 学習による方策の生成

4.3.1 オセロゲームのモデル設計

オセロゲームの方策を TD 学習で生成するためのモデル設計を行う．盤面は本来のルール通り，8マス四方の物を扱う．表 4.2 に TD 学習のためのオセロゲームのモデルを示す．

表 4.2 TD 学習のためのオセロゲームのモデル

状態 s	盤面を数列化した物 何もない　：0 黒コマがある：1 白コマがある：－1 どちらの手順であるかを表記 黒の手順　：1 白の手順　：－1 要素数 65 の数列で表記
行動 a	コマを置いた場所の座標 (0,0)～(7,7)
報酬 r	ゲーム終了時に勝利した側に＋1 敗北した側に－1 ゲーム中は常に 0 の報酬
方策π	Q 学習から求まる状態価値関数 $V(s)$によるε－グリーディ方策

4.3.2 汎化用ニューラルネットワークの設計

前述の通り，オセロゲームの状態数は非常に多く，このままでは TD 学習に適用するのは難しい．そのため，ニューラルネットワークを用いた関数近似による汎化を行う．まず，ニューラルネットワークからは状態価値関数 $V(s)$ を求めることにして，行動価値関数 $Q(s,a)$は行動 a をとったときの次の状態 sを求めた後，$V(s)$を出力させることで代用する．次にニューラルネットワークそのものだが，ニューラルネットワークでは入力および出力は 0 から 1 しか扱うことができない．そこで，入力および出力ユニットを 2 つ 1 組として扱うことで－1 から 1 を扱えるようにする．表 4.3 に汎化に用いたニュー

ラルネットワークの構造を示す.

表 4.3　汎化用のニューラルネットワークの構造

入力層	130 ユニット	
	1~64	各対応マスに黒コマがあれば 1，なければ 0
	65	黒の手順の時は 1，そうでなければ 0
	66~129	各対応マスに白コマがあれば 1，なければ 0
	130	白の手順の時は 1，そうでなければ 0
中間層	40 ユニット	
出力層	2 ユニット	ゲーム終了時：
	1	黒が勝ったときには，教師データ 1 を入力
		そうでないときには，教師データ 0 を入力
	2	白が勝ったときには，教師データ 1 を入力
		そうでないときには，教師データ 0 を入力
		ゲーム終了時以外：
		教師データ $V(s_{t+1})$を入力
		教師データより，BP 法により学習を行う

4.3.3　アルゴリズム

方策$\pi(s)$を実際のゲーム用プログラムに組み込むのは困難なので，ゲーム用プログラムにはニューラルネットワークの計算ルーチンおよび結合強度 w，決定論的方策$\pi(s)$の計算ルーチンを組み込み，学習用プログラムには結合強度 wのみを出力させることにした.

前述の設計の元に，ニューラルネットワークを用いた関数近似による汎化を行った，Q学習を用いた強化学習のアルゴリズムを図 4.3，図 4.4 に示す. また，結合強度 w を用いた決定論的方策$\pi(s)$の出力アルゴリズムを図 4.5 に示す.

結合強度 w の初期化
n 回繰り返し：
　繰り返し：
　　状態 s の初期化
　　ルーチンＥを呼び出す
　　ルーチンＡを呼び出す
　　　Ａ＝*none* のとき　s'←盤面はそのまま相手の手順の状態
　　　Ａ≠*none* のとき　s'←s で a を取った遷移後の状態
　　s が終端状態ならば，繰り返しを終了

w を出力する

ルーチンＥ：
　s が終端状態：
　　黒の勝利時　d←*1*
　　白の勝利時　d←*-1*
　s が終端状態でない
　　ルーチンＡを呼び出す
　　　Ａ＝*none* のとき　s'←盤面はそのまま相手の手順の状態
　　　Ａ≠*none* のとき　s'←s に a を取った遷移後の状態
　　d←$V(s')$
　$V(s)$と d より w の学習を行う

図 4.3　Q学習を用いた強化学習のアルゴリズム　その１

ルーチンA：

全てのマスhについて：

ルーチンBを呼び出す

B=*true*ならば　そのときの行動を行列 $a(n)$に記憶

n＝0ならば　*none*を返す

n≠0ならば

確率0.1で $a(n)$の中から等確率で行動 a を選択

確率0.9で $argmax_a Q(a,s)$となる a を選択

a を返す

ルーチンB：

現在の盤面pとコマhについて

hにコマを置いたときに

ひっくり返る相手のコマがあれば *true* を返す

1つもひっくり返せないならば *false* を返す

図4.4　Q学習を用いた強化学習のアルゴリズム　その2

結合強度wの組み込み

状態 s の入力

各 $a \in A(s)$について：

$s' \leftarrow s$ で a を取った遷移後の状態

$V(s')$を w を組み込んだニューラルネットワークにて求める

$$\pi(s) = \arg\max_a \sum_{s'} V(s')$$

$\pi(s)$を出力

図4.5　決定論的方策π(s)の出力アルゴリズム

第5章　実験

5.1　動的計画法による方策の生成

5.1.1　プログラミング

表 4.1 で提示した設計，図 4.1，図 4.2 で提示したアルゴリズムを元に C++ 言語を用いてプログラムを作成した[3][4].

また，本実験では実際に人間と対戦することによって目に見える評価を行うため，Microsoft VisualBasic6.0 を用いて，作成された方策を組み込んだインタフェースの作成を行った[5][6]. この際出力される方策が VisualBasic では扱いきれないデータ量になることが予想されたため，Microsoft Access を用いたデータベースを利用した[7].

5.1.2　結果

4マス4方のオセロゲームの総状態数は 88764 通りだった. この状態遷移から価値反復による学習を行わせたところ，初期状態[10000012002100000]の価値関数が-1 であった. なお，状態を列挙するのに4時間ほどの時間がかかった.

インタフェースを用い，作成された方策との対戦を行ったところ，作成された方策が後手の場合は一度も勝てなかった. また，作成された方策が先手であっても，方策通りの戦法を取らないと勝てないため，作成された方策が最適方策であると言える.

5.2 TD 学習による方策の生成

5.2.1 プログラミング

表 4.2，表 4.3 で提示した設計，図 4.3，図 4.4 で提示したアルゴリズムを元に C++言語を用いてプログラムを作成した．このとき，１００万回のエピソードを生成し，その結果による学習を行った．また，作成した方策を客観的に評価するために，図 4.5 で提示したアルゴリズムによって出力される決定論的方策と完全なランダム方策を対戦させ，その勝率を計算した．

なお，こちらの実験でも実際に人間と対戦することによって目に見える評価を行うため，Microsoft VisualBasic6.0 を用いて，作成された方策を組み込んだインタフェースの作成を行った．

5.2.2 結果

作成された方策の，ランダム方策との対戦結果を表 5.1 に示す

手順	対戦回数	勝利	敗北	引き分け	勝率
先手	1000	565	390	45	0.59
後手	1000	687	282	31	0.71

表 5.1　ランダム方策との対戦結果

インタフェースを用い，実際に対戦したところ，人間が特に考えずにゲームを行うとそれなりの勝負になった．しかし，人間がある程度考えながらゲームを行うと，作成された方策が勝つことはできなかった．

なお，１００万回のエピソードによる学習を行うのに，約３０時間の時間を要した．

第6章　考察

動的計画法を用いてのオセロゲームの方策の自動生成は，実際に最適方策を生成することができた．場合にもよるが，状態数が数万程度であり，状態の遷移を完全に書き出すことが可能ならば，動的計画法は相互作用問題の解決に十分利用可能である．

一方，TD 学習およびニューラルネットワークによる汎化を行ったオセロゲームの方策の自動生成は，ランダム方策よりは改善されているものの，まだまだ最適と言えるレベルの方策は生成できなかった．その理由は

- 学習量が不十分である
- ニューラルネットワークの設計が不適切である
- 結合強度の初期設定が不適切である

などが考えられる．１つ目の問題は時間さえかければ解決される．２つ目，３つ目の問題は，複数回の実験を行い，試行錯誤の結果適切な値を見つけだすしかないと考えられる．

また，今回作成したプログラムで一部冗長な処理を行っているせいか，計算にかなりの時間がかかってしまった．

第７章　終わりに

強化学習の可能性を探るために，大規模な相互作用問題であるオセロゲームの方策を動的計画法，TD 学習の２通りの手法を用いて作成した．

動的計画法では縮小型のゲームとはいえ最適方策を作成できた．しかし，TD 学習では最適方策を作成することはできなかった．

TD 学習での最適方策を作成するには，複数回の実験を行い適切な設定を見つけだすことが重要である．そのためにももっと効率の良いアルゴリズム，プログラムを作成し，計算にかかる時間を短縮することが今後の課題である．

謝辞

本研究の遂行および論文の作成に置いて多大な御助言および御指導を賜りました新納 浩幸 教官（茨城大学工学部システム工学科）に深い感謝の意を表します．

最後に，御指導を頂きましたシステム工学科計算機応用学講座の教官の方々，本研究を進めるにあたり御助言，御協力を頂きました，高橋 篤史 氏（茨城大学大学院理工学研究科システム工学専攻），阿部 修也 氏（茨城大学大学院理工学研究科システム工学専攻），同研究室の田邊 繁 氏（茨城大学工学部システム工学科），進藤 修 氏（茨城大学工学部システム工学科）に深く感謝致します．

参考文献

[1] Ricard S.Sutton and Andrew G Barto (訳 三上 貞芳, 皆川 雅章) : "強化学習", 森北出版 (2000)

[2] 馬場 規夫, 小島 史男, 小澤誠一 : "ニューラルネットの基礎と応用", 共立出版 (1994)

[3] 内山 章夫, 河野 吉伸, 津村 栄一, 中村 隆一, 長谷川 洋介 : "学生のための C", 東京電気大学出版局 (1995)

[4] Kris Jamsa (訳 春木良且, 佐藤 東九男) : "Dr.Jamsa の C++超入門", 株式会社アスキー (1994)

[5] 玉井 浩 : "Visual Basic プログラミング" 株式会社 サイエンス社 (2000)

[6] 林 晴比古 : "新 Visual Basic 入門 ビギナー編", ソフトバンク パブリッシング株式会社 (1998)

[7] 松田 猛, 小高 郁 : "Visual Basic 6.0 300 の技", 技術評論社 (2000)

付録 A プログラムソースリスト

```
//////////////////////////////////////////////////////
        状態数の列挙
//////////////////////////////////////////////////////
#include <iostream.h>
#include <stdio.h>


void check(int[4][4],char,int);
int change(int[4][4],int,int,char);
void outdata(int[16],int[16],int,int,char);
void outerr(int[16],char);
void result(int[16],int[4][4]);
char pchange(char);
main(){

        int i,j,k;
        int mark[4][4];
        for(i=0;i<4;i++){
                for(j=0;j<4;j++){
                        mark[i][j] = 0;
                }
        }
        mark[1][1] = 1;
        mark[2][2] = 1;
        mark[1][2] = 2;
        mark[2][1] = 2;

        check(mark,'B',0);
```

```
        check(mark,'W',0);
        return(0);
}

void check(int mk[4][4],char player,int flag1){

        int a,b,c,x,y;
        int swap[4][4];
        int out1[16];
        int out2[16];
        int flag2,flag3;

        flag2 = 0;
        for(y=0;y<4;y++){for(x=0;x<4;x++){
        c = 0;
        for(a=0;a<4;a++){
                for(b=0;b<4;b++){
                        swap[a][b] = mk[a][b];
                        out1[c] = mk[a][b];
                        c++;
                }
        }

        flag3 = change(swap,x,y,player);

        c = 0;
        for(a=0;a<4;a++){
                for(b=0;b<4;b++){
                        out2[c] = swap[a][b];
                        c++;
                }
        }
        if (flag3 == 1){
                outdata(out1,out2,x,y,player);
                flag2 = 1;
                check(swap,pchange(player),0);
```

```
                }
        }}
        if(flag2 == 0){
                if(flag1 != 1){
                        outerr(out1,player);
                        check(swap,pchange(player),1);
                }else{
                        result(out1,swap);
                }
        }
}

int change(int mk[4][4],int x,int y,char p){
        int ret = 0;
        int pc,oc;
        if(p =='B'){pc = 1;oc = 2;}
        else{pc = 2;oc = 1;}

        if(mk[y][x] != 0) return(0);
        mk[y][x] = pc;
        if(x+2 <= 3) if(mk[y][x+1] == oc && mk[y][x+2] == pc){
                mk[y][x+1] = pc;ret = 1;};
        if(x+3 <= 3) if(mk[y][x+1] == oc && mk[y][x+2] == oc && mk[y][x+3] == pc){
                mk[y][x+1] = pc;mk[y][x+2] = pc;ret = 1;}
        if(x-2 >= 0) if(mk[y][x-1] == oc && mk[y][x-2] == pc){
                mk[y][x-1] = pc;ret = 1;};
        if(x-3 >= 0) if(mk[y][x-1] == oc && mk[y][x-2] == oc && mk[y][x-3] == pc){
                mk[y][x-1] = pc;mk[y][x-2] = pc;ret = 1;}
        if(y+2 <= 3) if(mk[y+1][x] == oc && mk[y+2][x] == pc){
                mk[y+1][x] = pc;ret = 1;};
        if(y+3 <= 3) if(mk[y+1][x] == oc && mk[y+2][x] == oc && mk[y+3][x] == pc){
                mk[y+1][x] = pc;mk[y+2][x] = pc;ret = 1;}
        if(y-2 >= 0) if(mk[y-1][x] == oc && mk[y-2][x] == pc){
                mk[y-1][x] = pc;ret = 1;};
        if(y-3 >= 0) if(mk[y-1][x] == oc && mk[y-2][x] == oc && mk[y-3][x] == pc){
                mk[y-1][x] = pc;mk[y-2][x] = pc;ret = 1;}
```

```
        if(x+2 <= 3 && y+2 <= 3) if(mk[y+1][x+1] == oc && mk[y+2][x+2] == pc){
                mk[y+1][x+1] = pc;ret = 1;};
        if(x+3 <= 3 && y+3 <= 3) if(mk[y+1][x+1] == oc && mk[y+2][x+2] == oc &&
mk[y+3][x+3] == pc){
                mk[y+1][x+1] = pc;mk[y+2][x+2] = pc;ret = 1;}
        if(x-2 >= 0 && y+2 <= 3) if(mk[y+1][x-1] == oc && mk[y+2][x-2] == pc){
                mk[y+1][x-1] = pc;ret = 1;};
        if(x-3 >= 0 && y+3 <= 3) if(mk[y+1][x-1] == oc && mk[y+2][x-2] == oc &&
mk[y+3][x-3] == pc){
                mk[y+1][x-1] = pc;mk[y+2][x-2] = pc;ret = 1;};
        if(x+2 <= 3 && y-2 >= 0) if(mk[y-1][x+1] == oc && mk[y-2][x+2] == pc){
                mk[y-1][x+1] = pc;ret = 1;};
        if(x+3 <= 3 && y-3 >= 0) if(mk[y-1][x+1] == oc && mk[y-2][x+2] == oc &&
mk[y-3][x+3] == pc){
                mk[y-1][x+1] = pc;mk[y-2][x+2] = pc;ret = 1;};
        if(x-2 >= 0 && y-2 >= 0) if(mk[y-1][x-1] == oc && mk[y-2][x-2] == pc){
                mk[y-1][x-1] = pc;ret = 1;};
        if(x-3 >= 0 && y-3 >= 0) if(mk[y-1][x-1] == oc && mk[y-2][x-2] == oc && mk[y-
3][x-3] == pc){
                mk[y-1][x-1] = pc;mk[y-2][x-2] = pc;ret = 1;};

        if(ret == 0) mk[y][x] = 0;
        return(ret);
}

void outdata(int o1[16],int o2[16],int x,int y,char p){
        int a;
        FILE *fp;
        fp = fopen("outdata.txt","a");
        for(a=0;a<16;a++) fprintf(fp,"%d",o1[a]);
        fprintf(fp," %d%d%c ",y,x,p);
        for(a=0;a<16;a++) fprintf(fp,"%d",o2[a]);
        fprintf (fp,"¥n");
        fclose(fp);
}
```

```
void outerr(int o1[16],char p){
	int a;
	FILE *fp;
	fp = fopen("outdata.txt","a");
	for(a=0;a<16;a++) fprintf(fp,"%d",o1[a]);
	fprintf(fp," ¥"%c¥" is passed.¥n",p);
	fclose(fp);
}

void result(int out[16],int data[4][4]){
	int b=0,w=0,a,c;
	FILE *fp;
	for(a=0;a<4;a++){
		for(c=0;c<4;c++){
			if(data[a][c] == 1) b++;
			else if(data[a][c] == 2) w++;
		}
	}
	fp = fopen("outdata.txt","a");
	for(a=0;a<16;a++) fprintf(fp,"%d",out[a]);
	fprintf(fp," result is B:%d W:%d.¥n",b,w);
	fclose(fp);
}

char pchange(char pc){
	if(pc == 'B') pc = 'W';
	else pc = 'B';
	return(pc);
}
```

```
//////////////////////////////////////////////////////////
        動的計画法による学習
//////////////////////////////////////////////////////////

#include <iostream.h>
#include <stdio.h>

void change(int[4][4],int,int,char,long);

long datact;

struct st{
        int data[4][4];
        long act[2][4][4];
        int actct;
        float cost;
        };
struct st state[100000];

main(){

        int i,j,k;
        long l;
        char p;
        char sp[17];
        FILE *fp;
        FILE *fp2;

        for(i=0;i<4;i++){
                for(j=0;j<4;j++){
                        state[0].data[i][j] = 0;
                }
        }
        l = 1;

        fp = fopen("data4.txt","r");
```

```
while((fscanf(fp,"%s",sp)) != EOF){
        i = 0;
        for(j=0;j<4;j++){
                for(k=0;k<4;k++){
                        switch(sp[i]){
                                case    '0':
                                        state[l].data[j][k] = 0;
                                        i++;
                                        break;
                                case    '1':
                                        state[l].data[j][k] = 1;
                                        i++;
                                        break;
                                case    '2':
                                        state[l].data[j][k] = 2;
                                        i++;
                                        break;
                        }
                }
        }
        l++;
}
fclose(fp);
datact = l;

for(l=0;l<datact;l++){
        state[l].actct = 0;
        state[l].cost = 0;
        for(i=0;i<2;i++){
                for(j=0;j<4;j++){
                        for(k=0;k<4;k++){
                                state[l].act[i][j][k] = 0;
                        }
                }
        }
}
```

```
        for(l=0;l<datact;l++){
                for(i=0;i<4;i++){
                        for(j=0;j<4;j++){
                                change(state[l].data,i,j,'B',l);
                                change(state[l].data,i,j,'W',l);
                        }
                }cout << l << endl;
        }

        fp2 = fopen("tst4.txt","w");
        for(l=0;l<datact;l++){
                fprintf(fp2,"%5ld ",l);
                for(i=0;i<4;i++){
                        for(j=0;j<4;j++){
                                fprintf(fp2,"%d",state[l].data[i][j]);
                        }
                }
                for(i=0;i<2;i++){
                        for(j=0;j<4;j++){
                                for(k=0;k<4;k++){
                                        if(state[l].act[i][j][k] != 0){
                                                if(i == 0) p = 'B';
                                                else p = 'W';
                                                fprintf(fp2,"                %d%d%c
> %5ld",j,k,p,state[l].act[i][j][k]);
                                        }
                                }
                        }
                }
        fprintf(fp2,"¥n");
        }

        fclose(fp2);
        return(0);
}
```

```
void change(int mark[4][4],int x,int y,char p,long a){
        int b,c;
        long d;
        int ret = 0;
        int pc,oc;
        int mk[4][4];
        int flag;
        if(p =='B'){pc = 1;oc = 2;}
        else{pc = 2;oc = 1;}
                for(b=0;b<4;b++){
                        for(c=0;c<4;c++){
                                mk[b][c] = mark[b][c];
                        }
                }

        if(mk[y][x] == 0){
        mk[y][x] = pc;
        if(x+2 <= 3) if(mk[y][x+1] == oc && mk[y][x+2] == pc){
                mk[y][x+1] = pc;ret = 1;};
        if(x+3 <= 3) if(mk[y][x+1] == oc && mk[y][x+2] == oc && mk[y][x+3] == pc){
                mk[y][x+1] = pc;mk[y][x+2] = pc;ret = 1;}
        if(x-2 >= 0) if(mk[y][x-1] == oc && mk[y][x-2] == pc){
                mk[y][x-1] = pc;ret = 1;};
        if(x-3 >= 0) if(mk[y][x-1] == oc && mk[y][x-2] == oc && mk[y][x-3] == pc){
                mk[y][x-1] = pc;mk[y][x-2] = pc;ret = 1;}
        if(y+2 <= 3) if(mk[y+1][x] == oc && mk[y+2][x] == pc){
                mk[y+1][x] = pc;ret = 1;};
        if(y+3 <= 3) if(mk[y+1][x] == oc && mk[y+2][x] == oc && mk[y+3][x] == pc){
                mk[y+1][x] = pc;mk[y+2][x] = pc;ret = 1;}
        if(y-2 >= 0) if(mk[y-1][x] == oc && mk[y-2][x] == pc){
                mk[y-1][x] = pc;ret = 1;};
        if(y-3 >= 0) if(mk[y-1][x] == oc && mk[y-2][x] == oc && mk[y-3][x] == pc){
                mk[y-1][x] = pc;mk[y-2][x] = pc;ret = 1;}
```

```
        if(x+2 <= 3 && y+2 <= 3) if(mk[y+1][x+1] == oc && mk[y+2][x+2] == pc){
                mk[y+1][x+1] = pc;ret = 1;};
        if(x+3 <= 3 && y+3 <= 3) if(mk[y+1][x+1] == oc && mk[y+2][x+2] == oc &&
mk[y+3][x+3] == pc){
                mk[y+1][x+1] = pc;mk[y+2][x+2] = pc;ret = 1;}
        if(x-2 >= 0 && y+2 <= 3) if(mk[y+1][x-1] == oc && mk[y+2][x-2] == pc){
                mk[y+1][x-1] = pc;ret = 1;};
        if(x-3 >= 0 && y+3 <= 3) if(mk[y+1][x-1] == oc && mk[y+2][x-2] == oc &&
mk[y+3][x-3] == pc){
                mk[y+1][x-1] = pc;mk[y+2][x-2] = pc;ret = 1;};
        if(x+2 <= 3 && y-2 >= 0) if(mk[y-1][x+1] == oc && mk[y-2][x+2] == pc){
                mk[y-1][x+1] = pc;ret = 1;};
        if(x+3 <= 3 && y-3 >= 0) if(mk[y-1][x+1] == oc && mk[y-2][x+2] == oc &&
mk[y-3][x+3] == pc){
                mk[y-1][x+1] = pc;mk[y-2][x+2] = pc;ret = 1;};
        if(x-2 >= 0 && y-2 >= 0) if(mk[y-1][x-1] == oc && mk[y-2][x-2] == pc){
                mk[y-1][x-1] = pc;ret = 1;};
        if(x-3 >= 0 && y-3 >= 0) if(mk[y-1][x-1] == oc && mk[y-2][x-2] == oc && mk[y-
3][x-3] == pc){
                mk[y-1][x-1] = pc;mk[y-2][x-2] = pc;ret = 1;};
        }
        if(ret == 1){
                state[a].actct++;
                for(d=0;d<datact;d++){
                        flag = 1;
                        for(b=0;b<4;b++){
                                for(c=0;c<4;c++){
                                        if(mk[b][c] != state[d].data[b][c]) flag = 0;
                                }
                        }
                        if(flag == 1) state[a].act[pc-1][y][x] = d;
                }
        }
        else mk[y][x] = 0;
}
```

```
//////////////////////////////////////////////////////
        ニューラルネットを用いての汎化を利用したTD学習による学習
//////////////////////////////////////////////////////

#include <iostream.h>
#include <stdlib.h>
#include <time.h>
#include <fstream.h>

//結合要素
float w1[130][40];
float w2[40][2];
//盤面の状態
int plate[8][8];
int play;
struct dim{
        int x;
        int y;
};
struct nlout{
        float a;
        float b;
};
dim temp[60];
int tempcount;

//関数のプロトタイプ宣言
void reset(void);//盤面の初期化
int changeplay(int);//プレイヤーの変更
int changeplate(int[8][8],int,dim);//石をひっくり返す
void ablepoint(void);//石を置ける場所のチェック
dim setact(void);//行動ａの決定
nlout result(void);//勝敗の判定
float sig(float);//シグモイド関数
float dsig(float);//シグモイド関数の微分
nlout neural(int[8][8],int);//ニューラルネットによるｒの算出
```

```
void training(int);//ニューラルネットの学習
void dataout(char[64]);//データ出力

void main(void){//メインプログラム
        int a,b;
        int i,j;
        dim d;
        int flag1,flag2;
        srand((unsigned)time(NULL));

//結合強度の初期化
        for(i=0;i<130;i++){for(j=0;j<40;j++){
                w1[i][j] = rand() / 65536.0;
        }}
        for(i=0;i<40;i++){for(j=0;j<2;j++){
                w2[i][j] = rand() / 65536.0;
        }}

        for(a=0;a<100;a++){
                for(b=0;b<10000;b++){
                        reset();
                        flag1 = 0;
                        flag2 = 0;
                        training(flag1);
                        do{
                                ablepoint();
                                if(tempcount == 0){
                                        if(flag2) flag1 = 1;
                                        else flag2 = 1;
                                }else{
                                        flag2 = 0;
                                        d = setact();
                                        changeplate(plate,play,d);
                                }
                                play = changeplay(play);
                                training(flag1);
```

```
                    }while(flag1 == 0);
                }
        cout << "count " << a+1 << "0000" << endl;
        dataout(("dataout.txt"));
        }
}

void reset(void){//盤面の初期化
        int a,b;
        for(a=0;a<8;a++){
                for(b=0;b<8;b++){
                        plate[a][b] = 0;
                }
        }
        plate[3][3] = 1;
        plate[3][4] = -1;
        plate[4][3] = -1;
        plate[4][4] = 1;
        play = 1;
}

int changeplay(int a){//プレイヤーの変更
        if(a == 1) return(-1);
        else return(1);
}

int changeplate(int p[8][8],int pl,dim pt){//石をひっくり返す
        int pc = pl;
        int oc = changeplay(pl);
        int rt = 0;
        p[pt.x][pt.y] = pl;

        if(pt.x+2 < 8)
                if(p[pt.x+1][pt.y] == oc
                && p[pt.x+2][pt.y] == pc){
                        p[pt.x+1][pt.y] = pc;
```

```
                    rt = 1;}
if(pt.x+3 < 8)
        if(p[pt.x+1][pt.y] == oc
        && p[pt.x+2][pt.y] == oc
        && p[pt.x+3][pt.y] == pc){
                p[pt.x+1][pt.y] = pc;
                p[pt.x+2][pt.y] = pc;
                rt = 1;}
if(pt.x+4 < 8)
        if(p[pt.x+1][pt.y] == oc
        && p[pt.x+2][pt.y] == oc
        && p[pt.x+3][pt.y] == oc
        && p[pt.x+4][pt.y] == pc){
                p[pt.x+1][pt.y] = pc;
                p[pt.x+2][pt.y] = pc;
                p[pt.x+3][pt.y] = pc;
                rt = 1;}
if(pt.x+5 < 8)
        if(p[pt.x+1][pt.y] == oc
        && p[pt.x+2][pt.y] == oc
        && p[pt.x+3][pt.y] == oc
        && p[pt.x+4][pt.y] == oc
        && p[pt.x+5][pt.y] == pc){
                p[pt.x+1][pt.y] = pc;
                p[pt.x+2][pt.y] = pc;
                p[pt.x+3][pt.y] = pc;
                p[pt.x+4][pt.y] = pc;
                rt = 1;}
if(pt.x+6 < 8)
        if(p[pt.x+1][pt.y] == oc
        && p[pt.x+2][pt.y] == oc
        && p[pt.x+3][pt.y] == oc
        && p[pt.x+4][pt.y] == oc
        && p[pt.x+5][pt.y] == oc
        && p[pt.x+6][pt.y] == pc){
                p[pt.x+1][pt.y] = pc;
```

```
                    p[pt.x+2][pt.y] = pc;
                    p[pt.x+3][pt.y] = pc;
                    p[pt.x+4][pt.y] = pc;
                    p[pt.x+5][pt.y] = pc;
                    rt = 1;}
if(pt.x+7 < 8)
          if(p[pt.x+1][pt.y] == oc
          && p[pt.x+2][pt.y] == oc
          && p[pt.x+3][pt.y] == oc
          && p[pt.x+4][pt.y] == oc
          && p[pt.x+5][pt.y] == oc
          && p[pt.x+6][pt.y] == oc
          && p[pt.x+7][pt.y] == pc){
                    p[pt.x+1][pt.y] = pc;
                    p[pt.x+2][pt.y] = pc;
                    p[pt.x+3][pt.y] = pc;
                    p[pt.x+4][pt.y] = pc;
                    p[pt.x+5][pt.y] = pc;
                    p[pt.x+6][pt.y] = pc;
                    rt = 1;}

if(pt.x-2 >= 0)
          if(p[pt.x-1][pt.y] == oc
          && p[pt.x-2][pt.y] == pc){
                    p[pt.x-1][pt.y] = pc;
                    rt = 1;}
if(pt.x-3 >= 0)
          if(p[pt.x-1][pt.y] == oc
          && p[pt.x-2][pt.y] == oc
          && p[pt.x-3][pt.y] == pc){
                    p[pt.x-1][pt.y] = pc;
                    p[pt.x-2][pt.y] = pc;
                    rt = 1;}
if(pt.x-4 >= 0)
          if(p[pt.x-1][pt.y] == oc
          && p[pt.x-2][pt.y] == oc
```

```
            && p[pt.x-3][pt.y] == oc
            && p[pt.x-4][pt.y] == pc){
                    p[pt.x-1][pt.y] = pc;
                    p[pt.x-2][pt.y] = pc;
                    p[pt.x-3][pt.y] = pc;
                    rt = 1;}
if(pt.x-5 >= 0)
            if(p[pt.x-1][pt.y] == oc
            && p[pt.x-2][pt.y] == oc
            && p[pt.x-3][pt.y] == oc
            && p[pt.x-4][pt.y] == oc
            && p[pt.x-5][pt.y] == pc){
                    p[pt.x-1][pt.y] = pc;
                    p[pt.x-2][pt.y] = pc;
                    p[pt.x-3][pt.y] = pc;
                    p[pt.x-4][pt.y] = pc;
                    rt = 1;}
if(pt.x-6 >= 0)
            if(p[pt.x-1][pt.y] == oc
            && p[pt.x-2][pt.y] == oc
            && p[pt.x-3][pt.y] == oc
            && p[pt.x-4][pt.y] == oc
            && p[pt.x-5][pt.y] == oc
            && p[pt.x-6][pt.y] == pc){
                    p[pt.x-1][pt.y] = pc;
                    p[pt.x-2][pt.y] = pc;
                    p[pt.x-3][pt.y] = pc;
                    p[pt.x-4][pt.y] = pc;
                    p[pt.x-5][pt.y] = pc;
                    rt = 1;}
if(pt.x-7 >= 0)
            if(p[pt.x-1][pt.y] == oc
            && p[pt.x-2][pt.y] == oc
            && p[pt.x-3][pt.y] == oc
            && p[pt.x-4][pt.y] == oc
            && p[pt.x-5][pt.y] == oc
```

```
        && p[pt.x-6][pt.y] == oc
        && p[pt.x-7][pt.y] == pc){
                p[pt.x-1][pt.y] = pc;
                p[pt.x-2][pt.y] = pc;
                p[pt.x-3][pt.y] = pc;
                p[pt.x-4][pt.y] = pc;
                p[pt.x-5][pt.y] = pc;
                p[pt.x-6][pt.y] = pc;
                rt = 1;}

if(pt.y+2 < 8)
        if(p[pt.x][pt.y+1] == oc
        && p[pt.x][pt.y+2] == pc){
                p[pt.x][pt.y+1] = pc;
                rt = 1;}
if(pt.y+3 < 8)
        if(p[pt.x][pt.y+1] == oc
        && p[pt.x][pt.y+2] == oc
        && p[pt.x][pt.y+3] == pc){
                p[pt.x][pt.y+1] = pc;
                p[pt.x][pt.y+2] = pc;
                rt = 1;}
if(pt.y+4 < 8)
        if(p[pt.x][pt.y+1] == oc
        && p[pt.x][pt.y+2] == oc
        && p[pt.x][pt.y+3] == oc
        && p[pt.x][pt.y+4] == pc){
                p[pt.x][pt.y+1] = pc;
                p[pt.x][pt.y+2] = pc;
                p[pt.x][pt.y+3] = pc;
                rt = 1;}
if(pt.y+5 < 8)
        if(p[pt.x][pt.y+1] == oc
        && p[pt.x][pt.y+2] == oc
        && p[pt.x][pt.y+3] == oc
        && p[pt.x][pt.y+4] == oc
```

```
        && p[pt.x][pt.y+5] == pc){
                p[pt.x][pt.y+1] = pc;
                p[pt.x][pt.y+2] = pc;
                p[pt.x][pt.y+3] = pc;
                p[pt.x][pt.y+4] = pc;
                rt = 1;}
if(pt.y+6 < 8)
        if(p[pt.x][pt.y+1] == oc
        && p[pt.x][pt.y+2] == oc
        && p[pt.x][pt.y+3] == oc
        && p[pt.x][pt.y+4] == oc
        && p[pt.x][pt.y+5] == oc
        && p[pt.x][pt.y+6] == pc){
                p[pt.x][pt.y+1] = pc;
                p[pt.x][pt.y+2] = pc;
                p[pt.x][pt.y+3] = pc;
                p[pt.x][pt.y+4] = pc;
                p[pt.x][pt.y+5] = pc;
                rt = 1;}
if(pt.y+7 < 8)
        if(p[pt.x][pt.y+1] == oc
        && p[pt.x][pt.y+2] == oc
        && p[pt.x][pt.y+3] == oc
        && p[pt.x][pt.y+4] == oc
        && p[pt.x][pt.y+5] == oc
        && p[pt.x][pt.y+6] == oc
        && p[pt.x][pt.y+7] == pc){
                p[pt.x][pt.y+1] = pc;
                p[pt.x][pt.y+2] = pc;
                p[pt.x][pt.y+3] = pc;
                p[pt.x][pt.y+4] = pc;
                p[pt.x][pt.y+5] = pc;
                p[pt.x][pt.y+6] = pc;
                rt = 1;}

if(pt.y-2 >= 0)
```

```
            if(p[pt.x][pt.y-1] == oc
            && p[pt.x][pt.y-2] == pc){
                        p[pt.x][pt.y-1] = pc;
                        rt = 1;}
if(pt.y-3 >= 0)
            if(p[pt.x][pt.y-1] == oc
            && p[pt.x][pt.y-2] == oc
            && p[pt.x][pt.y-3] == pc){
                        p[pt.x][pt.y-1] = pc;
                        p[pt.x][pt.y-2] = pc;
                        rt = 1;}
if(pt.y-4 >= 0)
            if(p[pt.x][pt.y-1] == oc
            && p[pt.x][pt.y-2] == oc
            && p[pt.x][pt.y-3] == oc
            && p[pt.x][pt.y-4] == pc){
                        p[pt.x][pt.y-1] = pc;
                        p[pt.x][pt.y-2] = pc;
                        p[pt.x][pt.y-3] = pc;
                        rt = 1;}
if(pt.y-5 >= 0)
            if(p[pt.x][pt.y-1] == oc
            && p[pt.x][pt.y-2] == oc
            && p[pt.x][pt.y-3] == oc
            && p[pt.x][pt.y-4] == oc
            && p[pt.x][pt.y-5] == pc){
                        p[pt.x][pt.y-1] = pc;
                        p[pt.x][pt.y-2] = pc;
                        p[pt.x][pt.y-3] = pc;
                        p[pt.x][pt.y-4] = pc;
                        rt = 1;}
if(pt.y-6 >= 0)
            if(p[pt.x][pt.y-1] == oc
            && p[pt.x][pt.y-2] == oc
            && p[pt.x][pt.y-3] == oc
            && p[pt.x][pt.y-4] == oc
```

```
            && p[pt.x][pt.y-5] == oc
            && p[pt.x][pt.y-6] == pc){
                    p[pt.x][pt.y-1] = pc;
                    p[pt.x][pt.y-2] = pc;
                    p[pt.x][pt.y-3] = pc;
                    p[pt.x][pt.y-4] = pc;
                    p[pt.x][pt.y-5] = pc;
                    rt = 1;}
if(pt.y-7 >= 0)
            if(p[pt.x][pt.y-1] == oc
            && p[pt.x][pt.y-2] == oc
            && p[pt.x][pt.y-3] == oc
            && p[pt.x][pt.y-4] == oc
            && p[pt.x][pt.y-5] == oc
            && p[pt.x][pt.y-6] == oc
            && p[pt.x][pt.y-7] == pc){
                    p[pt.x][pt.y-1] = pc;
                    p[pt.x][pt.y-2] = pc;
                    p[pt.x][pt.y-3] = pc;
                    p[pt.x][pt.y-4] = pc;
                    p[pt.x][pt.y-5] = pc;
                    p[pt.x][pt.y-6] = pc;
                    rt = 1;}

if(pt.x+2 < 8 && pt.y+2 < 8)
            if(p[pt.x+1][pt.y+1] == oc
            && p[pt.x+2][pt.y+2] == pc){
                    p[pt.x+1][pt.y+1] = pc;
                    rt = 1;}
if(pt.x+3 < 8 && pt.y+3 < 8)
            if(p[pt.x+1][pt.y+1] == oc
            && p[pt.x+2][pt.y+2] == oc
            && p[pt.x+3][pt.y+3] == pc){
                    p[pt.x+1][pt.y+1] = pc;
                    p[pt.x+2][pt.y+2] = pc;
                    rt = 1;}
```

```
if(pt.x+4 < 8 && pt.y+4 < 8)
        if(p[pt.x+1][pt.y+1] == oc
        && p[pt.x+2][pt.y+2] == oc
        && p[pt.x+3][pt.y+3] == oc
        && p[pt.x+4][pt.y+4] == pc){
                p[pt.x+1][pt.y+1] = pc;
                p[pt.x+2][pt.y+2] = pc;
                p[pt.x+3][pt.y+3] = pc;
                rt = 1;}
if(pt.x+5 < 8 && pt.y+5 < 8)
        if(p[pt.x+1][pt.y+1] == oc
        && p[pt.x+2][pt.y+2] == oc
        && p[pt.x+3][pt.y+3] == oc
        && p[pt.x+4][pt.y+4] == oc
        && p[pt.x+5][pt.y+5] == pc){
                p[pt.x+1][pt.y+1] = pc;
                p[pt.x+2][pt.y+2] = pc;
                p[pt.x+3][pt.y+3] = pc;
                p[pt.x+4][pt.y+4] = pc;
                rt = 1;}
if(pt.x+6 < 8 && pt.y+6 < 8)
        if(p[pt.x+1][pt.y+1] == oc
        && p[pt.x+2][pt.y+2] == oc
        && p[pt.x+3][pt.y+3] == oc
        && p[pt.x+4][pt.y+4] == oc
        && p[pt.x+5][pt.y+5] == oc
        && p[pt.x+6][pt.y+6] == pc){
                p[pt.x+1][pt.y+1] = pc;
                p[pt.x+2][pt.y+2] = pc;
                p[pt.x+3][pt.y+3] = pc;
                p[pt.x+4][pt.y+4] = pc;
                p[pt.x+5][pt.y+5] = pc;
                rt = 1;}
if(pt.x+7 < 8 && pt.y+7 < 8)
        if(p[pt.x+1][pt.y+1] == oc
        && p[pt.x+2][pt.y+2] == oc
```

```
            && p[pt.x+3][pt.y+3] == oc
            && p[pt.x+4][pt.y+4] == oc
            && p[pt.x+5][pt.y+5] == oc
            && p[pt.x+6][pt.y+6] == oc
            && p[pt.x+7][pt.y+7] == pc){
                    p[pt.x+1][pt.y+1] = pc;
                    p[pt.x+2][pt.y+2] = pc;
                    p[pt.x+3][pt.y+3] = pc;
                    p[pt.x+4][pt.y+4] = pc;
                    p[pt.x+5][pt.y+5] = pc;
                    p[pt.x+6][pt.y+6] = pc;
                    rt = 1;}

if(pt.x-2 >= 0 && pt.y+2 < 8)
            if(p[pt.x-1][pt.y+1] == oc
            && p[pt.x-2][pt.y+2] == pc){
                    p[pt.x-1][pt.y+1] = pc;
                    rt = 1;}
if(pt.x-3 >= 0 && pt.y+3 < 8)
            if(p[pt.x-1][pt.y+1] == oc
            && p[pt.x-2][pt.y+2] == oc
            && p[pt.x-3][pt.y+3] == pc){
                    p[pt.x-1][pt.y+1] = pc;
                    p[pt.x-2][pt.y+2] = pc;
                    rt = 1;}
if(pt.x-4 >= 0 && pt.y+4 < 8)
            if(p[pt.x-1][pt.y+1] == oc
            && p[pt.x-2][pt.y+2] == oc
            && p[pt.x-3][pt.y+3] == oc
            && p[pt.x-4][pt.y+4] == pc){
                    p[pt.x-1][pt.y+1] = pc;
                    p[pt.x-2][pt.y+2] = pc;
                    p[pt.x-3][pt.y+3] = pc;
                    rt = 1;}
if(pt.x-5 >= 0 && pt.y+5 < 8)
            if(p[pt.x-1][pt.y+1] == oc
```

```
        && p[pt.x-2][pt.y+2] == oc
        && p[pt.x-3][pt.y+3] == oc
        && p[pt.x-4][pt.y+4] == oc
        && p[pt.x-5][pt.y+5] == pc){
                p[pt.x-1][pt.y+1] = pc;
                p[pt.x-2][pt.y+2] = pc;
                p[pt.x-3][pt.y+3] = pc;
                p[pt.x-4][pt.y+4] = pc;
                rt = 1;}
if(pt.x-6 >= 0 && pt.y+6 < 8)
        if(p[pt.x-1][pt.y+1] == oc
        && p[pt.x-2][pt.y+2] == oc
        && p[pt.x-3][pt.y+3] == oc
        && p[pt.x-4][pt.y+4] == oc
        && p[pt.x-5][pt.y+5] == oc
        && p[pt.x-6][pt.y+6] == pc){
                p[pt.x-1][pt.y+1] = pc;
                p[pt.x-2][pt.y+2] = pc;
                p[pt.x-3][pt.y+3] = pc;
                p[pt.x-4][pt.y+4] = pc;
                p[pt.x-5][pt.y+5] = pc;
                rt = 1;}
if(pt.x-7 >= 0 && pt.y+7 < 8)
        if(p[pt.x-1][pt.y+1] == oc
        && p[pt.x-2][pt.y+2] == oc
        && p[pt.x-3][pt.y+3] == oc
        && p[pt.x-4][pt.y+4] == oc
        && p[pt.x-5][pt.y+5] == oc
        && p[pt.x-6][pt.y+6] == oc
        && p[pt.x-7][pt.y+7] == pc){
                p[pt.x-1][pt.y+1] = pc;
                p[pt.x-2][pt.y+2] = pc;
                p[pt.x-3][pt.y+3] = pc;
                p[pt.x-4][pt.y+4] = pc;
                p[pt.x-5][pt.y+5] = pc;
                p[pt.x-6][pt.y+6] = pc;
```

```
                    rt = 1;}

if(pt.x+2 < 8 && pt.y-2 >= 0)
        if(p[pt.x+1][pt.y-1] == oc
        && p[pt.x+2][pt.y-2] == pc){
                p[pt.x+1][pt.y-1] = pc;
                rt = 1;}
if(pt.x+3 < 8 && pt.y-3 >= 0)
        if(p[pt.x+1][pt.y-1] == oc
        && p[pt.x+2][pt.y-2] == oc
        && p[pt.x+3][pt.y-3] == pc){
                p[pt.x+1][pt.y-1] = pc;
                p[pt.x+2][pt.y-2] = pc;
                rt = 1;}
if(pt.x+4 < 8 && pt.y-4 >= 0)
        if(p[pt.x+1][pt.y-1] == oc
        && p[pt.x+2][pt.y-2] == oc
        && p[pt.x+3][pt.y-3] == oc
        && p[pt.x+4][pt.y-4] == pc){
                p[pt.x+1][pt.y-1] = pc;
                p[pt.x+2][pt.y-2] = pc;
                p[pt.x+3][pt.y-3] = pc;
                rt = 1;}
if(pt.x+5 < 8 && pt.y-5 >= 0)
        if(p[pt.x+1][pt.y-1] == oc
        && p[pt.x+2][pt.y-2] == oc
        && p[pt.x+3][pt.y-3] == oc
        && p[pt.x+4][pt.y-4] == oc
        && p[pt.x+5][pt.y-5] == pc){
                p[pt.x+1][pt.y-1] = pc;
                p[pt.x+2][pt.y-2] = pc;
                p[pt.x+3][pt.y-3] = pc;
                p[pt.x+4][pt.y-4] = pc;
                rt = 1;}
if(pt.x+6 < 8 && pt.y-6 >= 0)
        if(p[pt.x+1][pt.y-1] == oc
```

```
            && p[pt.x+2][pt.y-2] == oc
            && p[pt.x+3][pt.y-3] == oc
            && p[pt.x+4][pt.y-4] == oc
            && p[pt.x+5][pt.y-5] == oc
            && p[pt.x+6][pt.y-6] == pc){
                    p[pt.x+1][pt.y-1] = pc;
                    p[pt.x+2][pt.y-2] = pc;
                    p[pt.x+3][pt.y-3] = pc;
                    p[pt.x+4][pt.y-4] = pc;
                    p[pt.x+5][pt.y-5] = pc;
                    rt = 1;}
if(pt.x+7 < 8 && pt.y-7 >= 0)
            if(p[pt.x+1][pt.y-1] == oc
            && p[pt.x+2][pt.y-2] == oc
            && p[pt.x+3][pt.y-3] == oc
            && p[pt.x+4][pt.y-4] == oc
            && p[pt.x+5][pt.y-5] == oc
            && p[pt.x+6][pt.y-6] == oc
            && p[pt.x+7][pt.y-7] == pc){
                    p[pt.x+1][pt.y-1] = pc;
                    p[pt.x+2][pt.y-2] = pc;
                    p[pt.x+3][pt.y-3] = pc;
                    p[pt.x+4][pt.y-4] = pc;
                    p[pt.x+5][pt.y-5] = pc;
                    p[pt.x+6][pt.y-6] = pc;
                    rt = 1;}

if(pt.x-2 >= 0 && pt.y-2 >= 0)
            if(p[pt.x-1][pt.y-1] == oc
            && p[pt.x-2][pt.y-2] == pc){
                    p[pt.x-1][pt.y-1] = pc;
                    rt = 1;}
if(pt.x-3 >= 0 && pt.y-3 >= 0)
            if(p[pt.x-1][pt.y-1] == oc
            && p[pt.x-2][pt.y-2] == oc
            && p[pt.x-3][pt.y-3] == pc){
```

```
                    p[pt.x-1][pt.y-1] = pc;
                    p[pt.x-2][pt.y-2] = pc;
                    rt = 1;}
if(pt.x-4 >= 0 && pt.y-4 >= 0)
          if(p[pt.x-1][pt.y-1] == oc
          && p[pt.x-2][pt.y-2] == oc
          && p[pt.x-3][pt.y-3] == oc
          && p[pt.x-4][pt.y-4] == pc){
                    p[pt.x-1][pt.y-1] = pc;
                    p[pt.x-2][pt.y-2] = pc;
                    p[pt.x-3][pt.y-3] = pc;
                    rt = 1;}
if(pt.x-5 >= 0 && pt.y-5 >= 0)
          if(p[pt.x-1][pt.y-1] == oc
          && p[pt.x-2][pt.y-2] == oc
          && p[pt.x-3][pt.y-3] == oc
          && p[pt.x-4][pt.y-4] == oc
          && p[pt.x-5][pt.y-5] == pc){
                    p[pt.x-1][pt.y-1] = pc;
                    p[pt.x-2][pt.y-2] = pc;
                    p[pt.x-3][pt.y-3] = pc;
                    p[pt.x-4][pt.y-4] = pc;
                    rt = 1;}
if(pt.x-6 >= 0 && pt.y-6 >= 0)
          if(p[pt.x-1][pt.y-1] == oc
          && p[pt.x-2][pt.y-2] == oc
          && p[pt.x-3][pt.y-3] == oc
          && p[pt.x-4][pt.y-4] == oc
          && p[pt.x-5][pt.y-5] == oc
          && p[pt.x-6][pt.y-6] == pc){
                    p[pt.x-1][pt.y-1] = pc;
                    p[pt.x-2][pt.y-2] = pc;
                    p[pt.x-3][pt.y-3] = pc;
                    p[pt.x-4][pt.y-4] = pc;
                    p[pt.x-5][pt.y-5] = pc;
                    rt = 1;}
```

```
        if(pt.x-7 >= 0 && pt.y-7 >= 0)
                if(p[pt.x-1][pt.y-1] == oc
                && p[pt.x-2][pt.y-2] == oc
                && p[pt.x-3][pt.y-3] == oc
                && p[pt.x-4][pt.y-4] == oc
                && p[pt.x-5][pt.y-5] == oc
                && p[pt.x-6][pt.y-6] == oc
                && p[pt.x-7][pt.y-7] == pc){
                        p[pt.x-1][pt.y-1] = pc;
                        p[pt.x-2][pt.y-2] = pc;
                        p[pt.x-3][pt.y-3] = pc;
                        p[pt.x-4][pt.y-4] = pc;
                        p[pt.x-5][pt.y-5] = pc;
                        p[pt.x-6][pt.y-6] = pc;
                        rt = 1;}

        return(rt);
}

void ablepoint(void){//石を置ける場所のチェック
        dim i,j;
        tempcount = 0;
        int dummyplate[8][8];
        for(i.x=0;i.x<8;i.x++){
                for(i.y=0;i.y<8;i.y++){

                        for(j.x=0;j.x<8;j.x++){
                                for(j.y=0;j.y<8;j.y++){
                                        dummyplate[j.x][j.y] = plate[j.x][j.y];
                                }
                        }

                        if(dummyplate[i.x][i.y] == 0){
                                if(changeplate(dummyplate,play,i)){
                                        temp[tempcount] = i;
                                        tempcount++;
```

```
                    }
                }
            }
        }
}

dim setact(void){//行動ａの決定
        dim i,j;
        int k;
        nlout l;
        float m;
        float max;
        int dummyplate[8][8];
        if((rand() * 10.0 / 32768.0) > 1.0){
                for(j.x=0;j.x<8;j.x++){
                        for(j.y=0;j.y<8;j.y++){
                                dummyplate[j.x][j.y] = plate[j.x][j.y];
                        }
                }
                changeplate(dummyplate,play,temp[0]);
                l = neural(dummyplate,changeplay(play));
                if(play == 1) m = l.a - l.b;
                else if(play == -1) m = l.b - l.a;
                max = m;
                i = temp[0];
                for(k=1;k<tempcount;k++){
                        for(j.x=0;j.x<8;j.x++){
                                for(j.y=0;j.y<8;j.y++){
                                        dummyplate[j.x][j.y] = plate[j.x][j.y];
                                }
                        }
                        changeplate(dummyplate,play,temp[k]);
                        l = neural(dummyplate,changeplay(play));
                        if(play == 1) m = l.a - l.b;
                        else if(play == -1) m = l.b - l.a;
                        if(m >= max){
```

```
                                max = m;
                                i = temp[k];
                        }
                }
        }else{
                i = temp[(int)(rand() * (float)tempcount / 32768.0)];
        }
        return(i);
}

nlout result(void){//勝敗の判定
        int black = 0;
        int white = 0;
        int i,j;
        nlout out;
        for(i=0;i<8;i++){
                for(j=0;j<8;j++){
                        if(plate[i][j] == 1) black++;
                        else if(plate[i][j] == -1) white++;
                }
        }
        if(black > white){out.a = 1.0;out.b = 0.0;}
        else if(black < white){out.a = 0.0;out.b = 1.0;}
        else {out.a = 0.0;out.b = 0.0;}
        return(out);
}

float sig(float a){//シグモイド関数
        return(1.0 / (1.0 + exp(-1.0 * a)));
}

float dsig(float a){//シグモイド関数の微分
        return(a * (1.0 - a));
}

nlout neural(int p[8][8],int pl){//ニューラルネットによるｒの算出
```

```
int i,j;
float x,y;
float x1[40];
float y1[2];
float data[130];
nlout out;

for(i=0;i<8;i++){
        for(j=0;j<8;j++){
                if(p[i][j] == 1) data[i*8 + j] = 1.0;
                else data[i*8 + j] = 0.0;
        }
}
if(pl == 1) data[64] = 1.0;
else data[64] = 0.0;

for(i=0;i<8;i++){
        for(j=0;j<8;j++){
                if(p[i][j] == -1) data[i*8 + j + 65] = 1.0;
                else data[i*8 + j + 65] = 0.0;
        }
}
if(pl == -1) data[129] = 1.0;
else data[129] = 0.0;

for(i=0;i<40;i++){
        x = 0;
        for(j=0;j<130;j++){
                x += w1[j][i] * data[j];
        }
        x1[i] = sig(x);
}
for(i=0;i<2;i++){
        y = 0;
        for(j=0;j<40;j++){
                y += x1[j] * w2[j][i];
```

```
                }
                y1[i] = sig(y);
        }
        out.a = y1[0];
        out.b = y1[1];
        return(out);
}

void training(int f){//ニューラルネットの学習
        float delta1[2];
        float delta2[40];
        nlout edu,ans,err;
        float x,y;
        float x1[40];
        float y1[2];
        float a = 0.1;
        dim d,e;
        int dummyplate[8][8];
        int i,j,k;
        float data[130];

//教師信号 edu の指定
        if(f == 1) edu = result();
        else{
                for(e.x=0;e.x<8;e.x++){
                        for(e.y=0;e.y<8;e.y++){
                                dummyplate[e.x][e.y] = plate[e.x][e.y];
                        }
                }
                ablepoint();
                if(tempcount != 0){
                        d = setact();
                        changeplate(dummyplate,play,d);
                        edu = neural(dummyplate,changeplay(play));
                }else{
                        edu = neural(dummyplate,changeplay(play));
```

```
            }
        }

//出力信号 ans の計算

        for(i=0;i<8;i++){
                for(j=0;j<8;j++){
                        if(plate[i][j] == 1) data[i*8 + j] = 1.0;
                        else data[i*8 + j] = 0.0;
                }
        }
        if(play == 1) data[64] = 1.0;
        else data[64] = 0.0;

        for(i=0;i<8;i++){
                for(j=0;j<8;j++){
                        if(plate[i][j] == -1) data[i*8 + j + 65] = 1.0;
                        else data[i*8 + j + 65] = 0.0;
                }
        }
        if(play == -1) data[129] = 1.0;
        else data[129] = 0.0;

        for(i=0;i<40;i++){
                x = 0;
                for(j=0;j<130;j++){
                        x += w1[j][i] * data[j];
                }
                x1[i] = sig(x);
        }
        for(i=0;i<2;i++){
                y = 0;
                for(j=0;j<40;j++){
                        y += x1[j] * w2[j][i];
                }
                y1[i] = sig(y);
```

```
        }
        ans.a = y1[0];
        ans.b = y1[1];

//誤差 err の計算
                err.a = edu.a - ans.a;
                err.b = edu.b - ans.b;

//学習部分
        delta1[0] = dsig(ans.a) * err.a;
        delta1[1] = dsig(ans.b) * err.b;
        for(i=0;i<40;i++){
                w2[i][0] = w2[i][0] + a * delta1[0] * x1[i];
                w2[i][1] = w2[i][1] + a * delta1[1] * x1[i];
        }
        for(i=0;i<30;i++){
                delta2[i] = 0;
                for(j=0;j<2;j++){
                        delta2[i] += dsig(x1[i]) * delta1[j] * w2[i][j];
                }for(j=0;j<130;j++){
                        w1[j][i] = w1[j][i] + a * delta2[i] * data[j];
                }
        }
}

void dataout(char filename[64]){//データ出力
        int i,j;
        ofstream file_object(filename);
        for(i=0;i<130;i++){for(j=0;j<40;j++){
                file_object << "w1[" << i << "][" << j << "] = " << w1[i][j] <<";"<< endl;
        }}
        for(i=0;i<40;i++){for(j=0;j<2;j++){
                file_object << "w2[" << i << "][" << j << "] = " << w2[i][j] <<";"<< endl;
        }}
}
```