TDMC(λ)に基づく評価関数の調整

大崎泰寛[†]　柴原一友[††]　但馬康宏[††]　小谷善行[††]

本稿では TD 学習に各非終端局面の勝率を導入させた強化学習法である TDMC(λ)によって評価関数のパラメータ調整を行った．思考ゲームにおける評価関数の自動調整の研究は古くから行われているが，自己対局によって収集された学習データによる TD(λ)の成功例の報告は十分でない．その原因の一つには，思考ゲームの報酬は終端局面で得られるゲームの勝敗ただ一つであり，各非終端局面には明確な報酬が存在しないことが挙げられる．したがって，TD(λ)は自己対局による学習データの勝敗に大きく依存していることが考えられる．そこで各非終端局面から多数のランダムシミュレートをすることで得られた勝率を代理的な報酬とみなして学習する TDMC(λ) と TD(λ)をオセロの環境において比較した．その結果，TDMC(λ)よって学習された評価関数は TD(λ)によって学習された評価関数に対して十分優れたパラメータの調整に成功した．

Learning Evaluation Function Based on TDMC(λ)

Yasuhiro OSAKI[†]　Kazutomo SHIBAHARA[††]　Yasuhiro TAJIMA[††]　Yoshiyuki KOTANI[††]

In this paper we present an adjustment of an Othello evaluation function using Temporal Difference Learning with Monte Carlo simulation(TDMC),which uses a combination of Temporal Difference Learning(TD) and winning probability in each non-terminal position. Studies on self-teaching evaluation functions as applied to logic games have been conducted for many years, however few successful results of employing TD have been reported. This is perhaps due to the fact that the only reward observable in logic games is their final outcome, with no obvious rewards present in non-terminal positions. TDMC(λ) attempts to compensate this problem by introducing winning probabilities, obtained through Monte Carlo simulation, as substitute rewards. Using Othello as a testing environment, TDMC(λ), in comparison to TD(λ), has been seen to yield better learning results.

1. はじめに

思考ゲームの研究において，コンピュータの思考ルーチンが最適な着手を選定する指針の一つである評価関数の機械学習がテーマにある．評価関数は与えられた局面から得られる様々な特徴とその重みから局面の形勢を数値で表すものであり，TD(Temporal Difference Learning)に代表される強化学習によってその精度向上の研究が取り組まれてきた[3][4]. TD 学習の目的は環境から与えられる報酬の総和を最大化することである．ところが思考ゲーム環境においては，エピソードの終端で得られる勝敗ただ一つを報酬として学習するため，学習データの推移や勝敗に依存し過ぎているという問題点があることから，自己対局によって得られた学習データから評価関数の学習を行うモデルには不向きであると考えられる．

本稿では, 非終端局面からモンテカルロシミュレーションによって得られた勝率を代理報酬としたTDMC(λ)(Temporal Difference Learning with Monte Carlo simulation)[10][11]を利用してオセロの評価関数のパラメータを調整した．モンテカルロシミュレーションの利点は，評価関数による形勢判断の困難な局面であっても乱数シミュレートによって近似的な勝率を算出できる点に加えて，勝率は評価関数の精度や構造には一切依存しないという点にある．非終端局面に勝率という代理的な報酬を与えることで期待代理報酬和の最大化を目標とした TD 学習へと拡張することを試みた．

本稿では，TDMC(λ)によって学習された評価関数と TD(λ)によって学習された評価関数をオセロで対局させることから学習成果の比較および分析を行った．そしてその結果，TD(λ)による評価関数に対して十分優れた評価関数の学習に成功した．

2. 思考ゲームにおける強化学習

強化学習の基本的な概念は「何を達成してほしいか」であり，決して「どのように達成してほしいか」ではない[1][2]．したがって，思考ゲーム環境において学習エージェントに与えられる目標はゲームの終端局面で環境から与えられる勝敗のみである．そしてその目標のことを報酬と呼ぶ．ゲーム木がすべて展開できるような探索空間ならば，すべての非終端局面にも終端局面から伝播した勝

[†] 東京農工大学大学院 工学府 情報工学専攻
Dept. of Computer and Information Sciences, Graduate school of Technology, Tokyo University of Agriculture and Technology

[††] 東京農工大学大学院 工学府
Department of Computer and Information Sciences, Tokyo University of Agriculture and Technology

敗が与えられるが，本稿ではそのような思考ゲームを強化学習タスクとして扱っていない．したがって，未解明の思考ゲームでは，非終端局面自体には報酬は存在しない. Sutton らによる TD 学習[1]では，「環境から与えられる報酬の総和の最大化」を目標として時間ステップごとに学習が行われる．そのため思考ゲーム環境においては，学習エピソードの終端の勝敗が報酬となって各非終端局面に与えられる．これは学習が学習エピソードの終端の勝敗に大きく依存しているといえる．つまり異なる学習エピソードにまったく同じ局面を発見しても，学習エピソードの終端から伝播した報酬が異なれば正反対の学習を行うことになる．

そこで，著者らは各非終端局面にその局面の勝率を代理報酬として付与することを試みた．これは，モンテカルロシミュレーションによって得られた近似的な勝率の利用が近年のコンピュータ囲碁を大きく飛躍させたことに由来したものである．解明されていない思考ゲームでは，着手によって勝率を高めることが勝利に近づく有効な方策である．そして，各非終端局面に勝率という代理報酬を与えた学習を TDMC(λ)として提案した．

各非終端局面に代理報酬を与えることで，TDMC(λ)では各時間ステップにおいて「環境から与えられる代理報酬の総和の最大化」を目標に学習することが可能となった．学習データの終端の勝敗を教師として学習していた TD 学習に対して，局面情報の一種である勝率を教師として学習するため TDMC(λ)は従来手法と決定的に異なる．

また，各非終端局面における代理報酬はモンテカルロシミュレーションによって得られる勝率であるため

- それまでの着手列に依存しない．
- 評価関数の精度や構造に依存しない．
- 終端に近づくにつれて実際の勝敗に近づく．

という性質を持っている．

3 章では TDMC(λ)アルゴリズム※を説明する．

3. TDMC(λ)アルゴリズム

本章では TDMC(λ)アルゴリズムを説明する．モンテカルロシミュレーションによって得られた各非終端局面の勝率を代理報酬として TD 学習を行うのが TDMC(λ)の概要である．

評価関数を構成するパラメータの総数が M 個であるときそれぞれの重みを $w=(w_1,w_2,\ldots,w_M)$ と表す. また, 学習エピソードの終端ステップを T とした時間ステップ $t\in[1,T]$ における局面 P_t の特徴量を $x_t=(x_{t1},x_{t2},\ldots,x_{tM})$ と表したとき，局面 P_t の静的な評価値 $V_t(x_t,w)$ は

$$V_t(x_t,w)\coloneqq x_t\cdot w$$

と線形結合で表すことができる．

TD 学習では，時間ステップ t で得られた報酬 r_t と次局面の評価値 $V_t(x_{t+1},w)$ から，現在局面の評価値 $V_t(x_t,w)$ を更新する．

$$V_t(x_t,w)\coloneqq V_t(x_t,w)+\alpha[r_t+\gamma V_t(x_{t+1},w)-V_t(x_t,w)]$$

更新時の γ を正の割引率とし，α を正の学習率とした．ところが思考ゲームでは各非終端局面において報酬 r_t が存在しないため終端の勝敗を伝播する役割の $V_t(x_{t+1},w)$ の精度に依存していた．

そこで，本提案手法では各非終端局面の勝率を代理報酬として，期待される代理報酬和の最大化を目標に TD 学習を行う. 学習エピソード内の局面 P_t からモンテカルロシミュレーションを行って得られた勝率を代理報酬 r_t としたとき，時間ステップ t における収益 R_t を

$$R_t\coloneqq\gamma^0r_t+\gamma^1r_{t+1}+\gamma^2r_{t+2}+\ldots+\gamma^{T-t}r_T=\sum_{i=t}^{T}\gamma^{i-t}r_i \quad (1)$$

と表す．γ の値が大きければ，R_t は学習局面から遠い局面の代理報酬にも影響され，γ の値が小さければ近い局面の代理報酬に強く影響される．また, 式(1)のように, 時間ステップ t から学習エピソードの終了 T までのすべての代理報酬を利用する代わりに n ステップ以降を推定価値に代用したものを n ステップ収益と呼ぶ.

$$R_t^{(n)}\coloneqq\gamma^0r_t+\gamma^1r_{t+1}+\ldots+\gamma^{n-1}r_{t+n-1}+\gamma^nV_t(x_{t+n},w) \quad (2)$$

続いて, 実際の収益 R_t と n ステップ収益 $R_t^{(n)}$ に適格度 λ を導入することで複合的にバックアップさせた λ 収益 R_t^{λ} について示す.

$$\begin{aligned}R_t^{\lambda}&\coloneqq\lambda^0R_t^{(1)}+\lambda^1R_t^{(2)}+\ldots+\lambda^{T-t-2}R_t^{(T-t-1)}+\lambda^{T-t-1}R_t\\&=\sum_{n=1}^{T-t-1}\lambda^{n-1}R_t^{(n)}+\lambda^{T-t}R_t\end{aligned} \quad (3)$$

式(1)，(2)から得られた式(3)の λ 収益 R_t^{λ} を学習の目標値として，局面 P_t から得られる静的な評価値 $V_t(x_t,w)$ を近似させる．

学習エピソードにおいて，目標値と評価値の各時間ステップ t における 2 乗誤差は以下の式(4)によって示される．

$$OF_{TDMC}=\sum_{t=1}^{T}[R_t^{\lambda}-V_t(x_t,w)]^2 \quad (4)$$

目的関数(4)から，各パラメータの勾配を求める．

$$\frac{\partial OF_{TDMC}}{\partial w}=-2\sum_{t=1}^{T}[R_t^{\lambda}-V(x_t,w)]\nabla_wV(x_t,w)=\Delta w$$

※ 第 12 回ゲームプログラミングワークショップ 2007 にて著者らが発表した”TD(λ)-MC 法を用いた評価関数の強化学習”で提案した TD(λ)-MC アルゴリズムに改良を加えたものである.

そして，最急降下法によって目標値との誤差を減少させる方向にパラメータ列 w を更新する．

$$w \coloneqq w + \alpha \Delta w \tag{5}$$

式(5)から静的な評価値 $V_t(x_t, w)$ で，各局面の目標値 R_t^{λ} を表すよう近似する．更新した結果，各局面における評価値が目標値と一致した場合，ある局面から最も評価値の高い子局面を選択することが期待される最大の代理報酬和を獲得することになる．実際オセロでは，局面 P_t からモンテカルロシミュレーションごとに得られる終端局面が勝ちなら+1，負けなら-1，引き分けなら 0 として，それらの和を平均することで r_t を求めた．評価値 $V_t(x_t, w)$ と目標値 R_t^{λ} をそれぞれ *hyperbolic tangent* によって正規化して，パラメータの更新を行った．

以下図 1 に TDMC(λ)アルゴリズムを要約した．

- 評価関数を構成する M 個のパラメータの重み列を $w = (w_1, w_2, \ldots, w_M)$ とする．
- 学習データ群に対して，時刻 $t \in [1, T]$ における各局面 P_t 上の各パラメータの特徴量列を $x_t = (x_{t1}, x_{t2}, \ldots, x_{tM})$ とする．
- 時刻 $t = 1,2,\ldots,T-1$ において，各局面 P_t における静的な評価値 $V_t(x_t, w)$ を求める．

$$V_t(x_t, w) \coloneqq x_t \cdot w$$

- 時刻 $t = 1,2,\ldots,T-1$ において，各局面 P_t における w の勾配を求める．

$$\nabla_w V_t(x_t, w) = \left(\frac{\partial V_t(x_t, w)}{\partial w_1}, \frac{\partial V_t(x_t, w)}{\partial w_2}, \ldots, \frac{\partial V_t(x_t, w)}{\partial w_M} \right)$$

- 時刻 $t = 1,2,\ldots,T-1$ において，各局面 P_t における収益 R_t を求める．

$$R_t \coloneqq \gamma^0 r_t + \gamma^1 r_{t+1} + \gamma^2 r_{t+2} \ldots + \gamma^{T-t} r_T = \sum_{i=t}^{T} \gamma^{i-t} r_i$$

（割引率 $\gamma \in [0,1]$）

ただし，r_t は局面 P_t からモンテカルロシミュレーションを行って得られた局面 P_t の勝率の近似値とする．

- 時刻 $t = 1,2,\ldots,T-1$ において，各局面 P_t における n ステップ収益 $R_t^{(n)}$ を求める．

$$R_t^{(n)} \coloneqq \gamma^0 r_t + \gamma^1 r_{t+1} + \ldots + \gamma^{n-1} r_{t+n-1} + \gamma^n V_t(x_{t+n}, w)$$

- 時刻 $t = 1,2,\ldots,T-1$ において，収益 R_t および n ステップ収益 $R_t^{(n)}$ から λ 収益 R_t^{λ} を得る．

$$R_t^{\lambda} \coloneqq \sum_{n=1}^{T-t-1} \lambda^{n-1} R_t^{(n)} + \lambda^{T-t-1} R_t$$

（適格度 $\lambda \in [0,1]$）

- 各パラメータ w_j を以下の式から更新する．

$$w_j \coloneqq w_j + \alpha \sum_{t=1}^{T-1} [R_t^{\lambda} - V_t(x_t, w)] \nabla_{w_j} V_t(x_t, w)$$

（学習率 $\alpha \in (0,1]$）

図 1　TDMC(λ)アルゴリズム

4. 強化学習の実験環境

オセロにて，提案手法による評価関数のパラメータ調整を行った．以下，強化学習の実験環境について述べる．

4.1 評価関数の構造

評価関数は表 1 の 15 種類のパラメータを持つ．

表 1　評価関数の各パラメータ

特徴	数	詳細
マス	10	盤の中心を通る4つの軸に対して対称関係にあるマス
可能手	1	可能手の総数
開放度	1	自分の石のひっくり返される置き方の総数
確定石	1	今後ひっくり返されない自分の石の数
スコア	1	盤上の自分の石の数
手番	1	着手する手番

オセロの初期位置を図 2 に示す．先手番の黒の石が置かれていればマスの特徴量を +1 とし，白の石が置かれていれば -1 とし，いずれの石も置かれていなければ 0 とする．また，8×8 マスをそれぞれ独立したパラメータとせず，盤の中心を通る 2 本の対角線，垂直線，水平線に対して対称関係にあるマスを同じものと見なした．

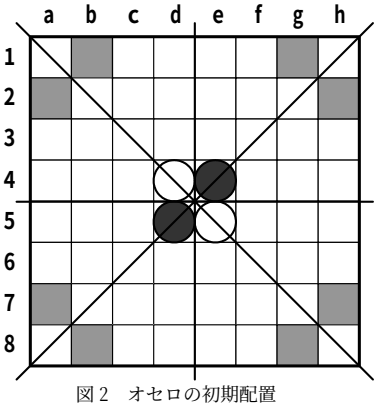

図 2　オセロの初期配置

例えば座標(b,1),(g,1),(a,2),(h,2),(a,7),(h,7),(b,8),(g,8) は対称関係にあるためすべて同じ特徴と見なした．

4.2 オセロの進行度

オセロは 8×8 の盤上から置いた石は取り除かれることがないので，パスを除けば最長で 60 手で対局が終了する．そこで，盤上の石の置き方から思考ルーチンに序盤・中盤・終盤の大局観を持たせ，それぞれ独立した重み列 w で局面を判断するようにした．

表 2　進行度の条件

進行度	条件
序盤	行1,行8,列1,列8のいずれかに石が置かれるまで
中盤	(a,1),(h,1),(a,8),(h,8)のうち,2つ以上同じ色の石が置かれるまで
終盤	終局まで

表 2 のように，ゲームの進行度に合わせて重み列 w を切り替えることで，序盤の学習結果が終盤の学習に影響を与えすぎないように配慮した．

4.3 自己対局による学習データ群の収集

ε-greedy に基づいた自己対局から学習データを収集した．着手は ε でランダムに子局面を選択し，1-ε で子局面のうち最も評価値の高いものを着手とする．ただし，最も評価値の高いものが 2 つ以上ある場合は，その中からランダムで着手を選択する．ε の値が小さいと評価関数の構造に依存した学習データへと偏向してしまうが，一方で大きすぎると評価関数の学習成果が学習データに反映されなくなってしまう．

このようにして得られた学習データから，提案手法を行って各パラメータを調整する．そして，更新された評価関数から再び自己対局を行って学習データを収集した．以下の図 3 に学習サイクルモデルを示す．

1　各パラメータを 0 に初期化する．
2　ε-greedy に基づいた自己対局を行う．
3　得られた学習データから強化学習を行う．
4　評価関数を更新する．
5　規定回数学習が行われるまで 2 に戻る．
6　終了

図 3　学習サイクルモデル

5. 実験結果

予備実験から割引率 γ と適格度 λ をそれぞれ 0.98 と定めた．また，学習率 α の初期値を 0.5 として，学習回数に応じて値を減少させていった．TDMC(λ)および TD(λ)から得られた評価関数による対局結果を以下に示す．

5.1 学習前の評価関数に対する対局結果

評価関数の各パラメータの重みを 0 に初期化した後，学習の規定回数を 1000 回として自己対局による学習データから TDMC(λ)によって評価関数のパラメータの更新を行った．このとき，自己対局時の乱数着手の割合 ε を 0.03 とした．また，学習データの各局面からモンテカルロシミュレーションを 1000 回して，その勝率を代理報酬とした．

TDMC(λ)によって学習された評価関数の学習成果を示すため，学習前の評価関数と対局実験を行った．最も評価値の高い着手が 2 つ以上ある場合は，その中からランダムで選択し，先手後手合わせて 1000 回対局した結果が以下の表 3 である．

表 3　TDMC(λ)と学習前の対局結果

TDMC(λ) Player color	TDMC(λ) Player wins	Draws	Untrained Player wins
Black	481	1	18
White	483	6	11
TOTAL	964	7	29

このように, 1000 回対局した結果 964 勝 29 敗 7 分と学習前の評価関数に対して十分勝ち越していることがわかった．また，表 3 から，先手後手に勝ち数の偏りはないといえる．

続いて，TDMC(λ)による強化学習同様，自己対局による学習データ群から TD(λ)よって評価関数の更新を行った．このとき, ε を 0.03 として自己対局を行った．ただし，TDMC(λ)と比較して一回の学習サイクルに要する時間が 1/5 程度であったため，自己対局を 5000 回行った．TD(λ)によって学習された評価関数の学習成果を示すため，学習前の評価関数に対して先手後手合わせて 1000 回対局した．

表 4　TD(λ)と学習前の対局結果

TD(λ) Player color	TD(λ) Player wins	Draws	Untrained Player wins
Black	479	2	19
White	478	3	19
TOTAL	956	5	38

表 4 のように，TDMC(λ)同様，TD(λ)による評価関数であっても学習前の評価関数に対して十分勝ち越していることがわかった．また，表 4 から先手後手に勝ち数の偏りはないといえる．

5.2　TD(λ)による評価関数に対する対局結果

続いて, TDMC(λ)による評価関数と TD(λ)による評価関数との対局実験を行った．お互い最も評価値の高い局面を着手し，2 つ以上ある場合は，その中からランダムで着手を選択した．ただし，深さ *d* 手目までは完全にランダムで可能手を選択するようにして，そこから先手後手を合わせて 1000 回対局した．対局結果を以下の表 5 に示した．

表 5　TDMC(λ) と TD(λ)の対局結果

d	TDMC(λ) Player wins	Draws	TD(λ) Player wins
4	880	0	120
6	817	10	173
8	794	20	186
10	782	18	200

ランダム着手の深さ *d* を 4，6，8，10 で比較したところ，そのすべての場合において TDMC(λ)による評価関数が TD(λ)による評価関数に勝ち越した．なお *d = 0* のとき，つまりお互い評価値が最大になるような着手を選択することで対局した結果，先手後手ともに TDMC(λ)による評価関数が勝った．

また，TDMC(λ)によって学習された評価関数の盤の重みを表 6 から表 8，図 4 から図 6 に示した．図 4 から図 6 では，色が濃くなるにつれて重みが増すものとする．

表 6 から序盤では行 1，行 8，列 a，列 h は学習対象ではないため, 初期加重が 0 のままであった．盤の 4 隅に自分の石を置くと勝率が上がるため，それらに石が置きやすくなるマス(c,3), (c,6), (f,3), (f,6)の値が最も高くなったと考えられる．一方で，(b,2)，(b,7)，(g,2)，(g,7)に石を置くと，4 隅が取られやすくなって勝率が下がるために値が負になるよう学習されている．

表 7 から中盤では石を置くと勝率が高まる盤の 4 隅の値が最も高くなった．一方で，盤の 4 隅の近傍が最も低い値となった．その原因は，それらのマスに石を置くと 4 隅が相手に取られて勝率が下がるためであると考えられる．

表 8 から終盤においても盤の 4 隅が最も高い値を示した．更に序盤，中盤では比較的値の低かった盤の中央の値が高くなった．これは，終盤に自分の石が盤の中央にあれば相手の石を多くひっくり返す機会が生まれるからであると考えられる．

5.3　ε の値による対局結果の変化

自己対局によって学習データを収集するとき，ε の割合でランダム着手をすることが学習データの偏向を防いでいる. この ε の値によって対局結果がどのように変化するかを調べた．TDMC(λ)による評価関数を ε の値に合わせて学習させた．また，TD(λ)による評価関数は 5.2 と同じものを用いた．10 手目までは完全にランダムで可能手を選択するようにして，そこから先手後手合わせて 1000 対局行った結果を以下の表 9 に示した．

表 9
TDMC(λ)の ε の変化に応じた対局結果

TDMC(λ)'s ε	TDMC(λ) Player wins	Draws	TD(λ) Player wins
0.01	781	11	208
0.03	782	18	200
0.05	757	13	230
0.10	767	12	221
0.30	721	14	265
1.00	712	12	276

表 9 から，ε の値が低いほうが TD(λ)による評価関数に対する勝ち数が多いことがわかった．この理由は，ランダム着手が発生した手前の学習局面 P_t から得られた収益 R_t が不正確であるからだと考えられる．なぜなら収益 R_t は学習局面から近い局面の勝率ほど割引率 γ の影響が少ないからである．続いて，TDMC(λ)，TD(λ)ともに ε に合わせて学習させた場合，対局結果がどのように変化するかを調べた．10 手目までは完全にランダムで可能手を選択するようにして，そこから先手後手合わせて 1000 対局行った結果を以下の表 10 に示した．

表 6　序盤の盤上の各マスの重み

Row＼Col	a	b	c	d	e	f	g	h
1	0	0	0	0	0	0	0	0
2	0	-0.02231	0.055829	0.020036	0.020036	0.055829	-0.02231	0
3	0	0.055829	0.10126	-0.10927	-0.10927	0.10126	0.055829	0
4	0	0.020036	-0.10927	-0.10155	-0.10155	-0.10927	0.020036	0
5	0	0.020036	-0.10927	-0.10155	-0.10155	-0.10927	0.020036	0
6	0	0.055829	0.10126	-0.10927	-0.10927	0.10126	0.055829	0
7	0	-0.02231	0.055829	0.020036	0.020036	0.055829	-0.02231	0
8	0	0	0	0	0	0	0	0

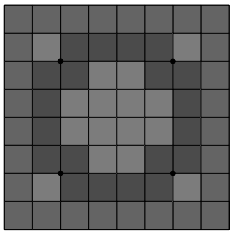

図 4　序盤の重み

表 7　中盤の盤上の各マスの重み

Row＼Col	a	b	c	d	e	f	g	h
1	6.327108	-3.32813	0.339068	-2.00512	-2.00512	0.339068	-3.32813	6.327108
2	-3.32813	-1.52928	-1.8755	-0.18176	-0.18176	-1.8755	-1.52928	-3.32813
3	0.339068	-1.8755	1.069396	0.624148	0.624148	1.069396	-1.8755	0.339068
4	-2.00512	-0.18176	0.624148	0.105396	0.105396	0.624148	-0.18176	-2.00512
5	-2.00512	-0.18176	0.624148	0.105396	0.105396	0.624148	-0.18176	-2.00512
6	0.339068	-1.8755	1.069396	0.624148	0.624148	1.069396	-1.8755	0.339068
7	-3.32813	-1.52928	-1.8755	-0.18176	-0.18176	-1.8755	-1.52928	-3.32813
8	6.327108	-3.32813	0.339068	-2.00512	-2.00512	0.339068	-3.32813	6.327108

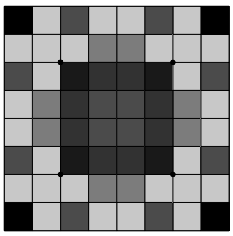

図 5　中盤の重み

表 8　終盤の盤上の各マスの重み

Row＼Col	a	b	c	d	e	f	g	h
1	5.500621	-0.17812	-2.58948	-0.59007	-0.59007	-2.58948	-0.17812	5.500621
2	-0.17812	0.968038	-2.16084	-2.01723	-2.01723	-2.16084	0.968038	-0.17812
3	-2.58948	-2.16084	0.490617	-1.07055	-1.07055	0.490617	-2.16084	-2.58948
4	-0.59007	-2.01723	-1.07055	0.734863	0.734863	-1.07055	-2.01723	-0.59007
5	-0.59007	-2.01723	-1.07055	0.734863	0.734863	-1.07055	-2.01723	-0.59007
6	-2.58948	-2.16084	0.490617	-1.07055	-1.07055	0.490617	-2.16084	-2.58948
7	-0.17812	0.968038	-2.16084	-2.01723	-2.01723	-2.16084	0.968038	-0.17812
8	5.500621	-0.17812	-2.58948	-0.59007	-0.59007	-2.58948	-0.17812	5.500621

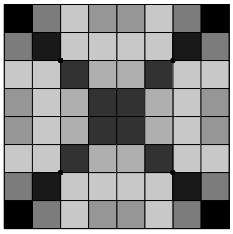

図 6　終盤の重み

表 10　ε の変化に応じた対局結果

ε	TDMC(λ) Player wins	Draws	TD(λ) Player wins
0.01	816	7	177
0.03	782	18	200
0.05	758	19	223
0.10	700	15	285
0.30	719	16	265
1.00	781	10	209

表 10 から，ε の値に関わらずすべての場合において TDMC(λ)による評価関数が TD(λ)による評価関数に対して 7 割以上勝ち越していることがわかった. 最も勝ち数が多かったのは, *ε=0.01* のときで, 1000 対局中 816 勝であった.

5.2, 5.3 より，可能手から完全にランダムで着手を選択する深さ *d* や自己対局時における乱数着手の割合 ε について比較してきたが, そのすべてにおいて TDMC(λ)による評価関数が TD(λ)による評価関数に対して大きく勝ち越す結果となった.

では，どうして TDMC(λ)は TD(λ)よりも優れた評価関数を学習できたのだろうか．以下に 2 つの考察を示した.

6. 考察

- 表 10 より両者の学習時における *ε=1.00* のとき，つまり完全にランダムに学習データを収集する場合と ε の値が低い場合との間の勝ち数に多く差が生じなかったことから，TDMC(λ) は TD(λ) と比べて学習データのランダム性に大きく依存しないことが考えられる．つまり，例え有利な局面からでたらめな着手をしたことで結果的に負けたとしても，非終端局面における勝率から学習を行うために TD(λ)よりも終端の勝敗の影響を受けないのである．このことから初期化された評価関数による自己対局と学習を交互に行うような学習サイクルにおいて，学習データの精度が求められるために TD(λ) よりも TDMC(λ) の方がより適していたと考えられる.

- TD(λ)が学習データの終端の勝敗を報酬としているのに対し，TDMC(λ)では各非終端局面の勝率を代理報酬としているため，各学習局面における目標値は 2 つの学習アルゴリズムでは異なるが，ゲームが進行するにつれて両者の目標値は似た値をとる．なぜなら，終端に近づくにつれ，モンテカルロシミュレーションによる勝率は実際の勝敗にほぼ一致した値を返すからである．このことから，TDMC(λ)と TD(λ)では終盤の評価関数のパラメータはとても良く似ることが予想される. 以下の図 7, 図 8 は 5.2 にて TDMC(λ) および TD(λ)から学習された評価関数における手番の重みに対する各パラメータの重みの比率を示したものである.

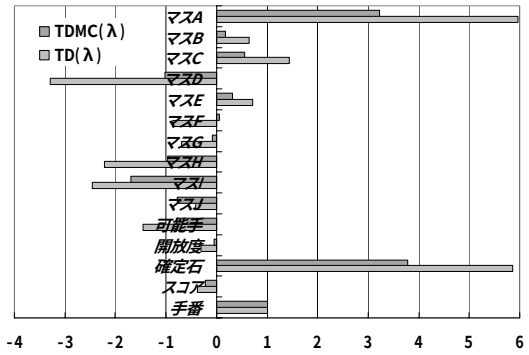

図 7　中盤の手番の重みに対する各パラメータの比率

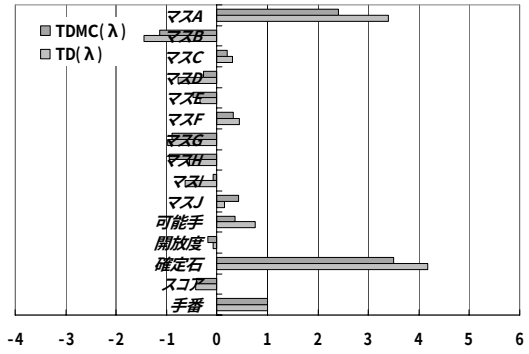

図 8　終盤の手番の重みに対する各パラメータの比率

図 8 より TDMC(λ)と TD(λ)の終盤の各パラメータの比率はよく似たものとなっている．一方で，図 7 より TDMC(λ)と TD(λ)の中盤の各パラメータの比率は両者の間に差が生じていることがわかる．マス B※，マス D※，マス G※，開放度（自分の石がひっくり返される置き方の総数）は 3 倍以上差があり，特にマス F※は値の符合が異なっている．また中盤において比率の高いパラメータであるマス A※，および確定石（もうひっくり返されない石の総数）は中盤では 1.5 倍程度の開きがあった．図 7 より，両者の対局結果の差は中盤の評価関数の重みの差が関係していることがわかった．

7. おわりに

本稿では，各非終端局面に勝率という代理報酬を導入させた新しい強化学習である TDMC(λ)の優れた学習成果を従来手法と比較することによって示した．自己対局によって収集した学習データから学習する場合において学習データの勝敗に依存し過ぎるという TD(λ)の問題点を勝率の導入によって改善したことがパラメータの調節の成功要因に挙げられる．期待される代理報酬の総和の最大化という目標で各非終端局面において強化学習をすることが，未解明の思考ゲームにおいて有効であるという点が本稿の最も主張するところである．

今後の課題には，CEC2006COMPETITION[7][8]で競われたような盤上のマス 64 種類をパラメータとした評価関数を構成し，TD(λ)や共進化学習(CEL)から学習された評価関数と TDMC(λ)による評価関数との対局結果を比較する必要がある．

また，熟達したオセロプレイヤ同士の対局から作られた棋譜から TDLeaf(λ)[4]によって学習された評価関数に対して，TDMC(λ)による評価関数は勝ち越すことが可能かを検証する必要がある．

更には UCT アルゴリズム[5][6]において探索木から離れた後のエピソードを完了されるための方策であるデフォルト方策[9]に，TDMC(λ)によってオフライン学習させた評価関数を用いた場合や，初めて訪問した局面の信頼上限の初期値やプレイアウト回数の事前推定に役立つことが考えられる．

※　マス A～マス J は以下の図に示された特徴である．

	a	b	c	d	e	f	g	h
1	A	I	B	D	D	B	I	A
2	I	J	H	G	G	H	J	I
3	B	H	C	E	E	C	H	B
4	D	G	E	F	F	E	G	D
5	D	G	E	F	F	E	G	D
6	B	H	C	E	E	C	H	B
7	I	J	H	G	G	H	J	I
8	A	I	B	D	D	B	I	A

参考文献

[1]Richard Sutton, Andrew Barto,"Introduction to Reinforcement Learning", *MIT Press*,1998.

[2]Richard Sutton,"Learning to predict by the methods of temporal differences", *Machine Learning* (1988 -(3)):pp.9-44.

[3]Gerald Tesauro,"Temporal difference learning and TD-gammon", *Communications of the ACM* (1994 -38(3)):pp.58-68.

[4]Jonathan Baxter, Andrew Tridgell, Lex Weaver,"Experiments in Parameter Learning Using Temporal Differences", *ICGA Journal* (1998 June):pp.84-99.

[5]Sylvain Gelly, Yizao Wang, Remi Munos, Olivier Teytaud, "Modification of UCT with Patterns in Monte-Carlo Go", *RR-6062-INRIA* (2006):pp.1-19.

[6]Remi Coulom,"Computing Elo Ratings of Move Patterns in the Game of Go", *Proc. 9th International Conference on Computer Games* (2007):pp.113-124.

[7]Edward P. Manning,"Temporal Difference Learning of an Othello Evaluation Function for a Small Neural Network with Shared Weights",*IEEE Symposium on Computational Intelligence and Games* (2007):pp.216-223.

[8]Simon M.Lucas, Thomas P.Runarsson,"Temporal Difference Learning Versus Co-Evolution for Acquiring Othello Position Evaluation", *IEEE Symposium on Computational Intelligence and Games* (2006):pp.52-59.

[9]Gelly Sylvain, Silver David,"Combining online and offline knowledge in UCT", *ICML Proc. 24th international conference on Machine learning* (2007):pp.273-280

[10]大崎 泰寛, 柴原 一友, 但馬 康宏, 小谷 善行, “TD(λ)-MC 法を用いた評価関数の強化学習”,*第 12 回ゲームプログラミングワークショップ*(2007):pp.36-43.

[11]大崎 泰寛, 柴原 一友, 但馬 康宏, 小谷 善行, “モンテカルロシミュレーションを用いた強化学習法の提案”, *ゲーム情報学研究会* (2008):pp.37-44.