3C-3

コンピュータ将棋への TD(λ)法の適用:Bonanza の評価関数パラメータ値

五十嵐 治一† 山本 一将‡
芝浦工大† 芝浦工大‡

1. はじめに

近年，コンピュータ将棋の実力はプロ棋士に迫るものがある[1]．この一因となっているのが，将棋ソフト Bonanza で提案された評価関数の自動学習[2]である．現在の将棋ソフトにおいては，Bonanza と同様，評価関数中のパラメータをプロ棋士の棋譜データベースを利用した教師付学習により自動調整することが主流となっている．

一方，評価関数の自動学習に関しては，強化学習の一種である TD(λ)法を用いた試みがなされてきた[3][4]．TD(λ)法はバックギャモンでは大成功を収めたが[5]，残念ながら将棋ではそれほど良い結果は報告されていない．我々は，この原因として，評価関数中のパラメータが多い（例，Bonanza ver.4.1.3 では約 9000 万個）ため，全くのゼロの状態から強化学習を適用させて適切なパラメータ値を得るのは難しいのではないかと考えた．そこで，すでに公開されている Bonanza のパラメータ値を初期値として，TD(λ)法により Bonanza の評価関数を強化することを試みている．本稿はこの試みの理論的枠組みを解説する．

2. 強化学習とコンピュータ将棋

2.1 TD(λ)法の概略

方策πによる状態 s での状態価値関数 $V^{\pi}(s)$，

$$V^{\pi}(s) \equiv E_{\pi}\left[R_t \middle| s_t = s\right] = E_{\pi}\left[\sum_{k=0}^{L-t-1} \gamma^k r_{t+k+1} \middle| s_t = s\right] \quad (1)$$

を，パラメータωを含む関数 $V(s;\omega)$ で近似する．ただし，$\gamma \in (0,1]$は割引率，L はエピソードの最終時刻である．そこで，次の平均二乗誤差，

$$MSE(\omega) \equiv \sum_{\sigma \in \Omega_\sigma} P(\sigma;\omega) \sum_{s} \left[V^{\pi}(s) - V(s;\omega)\right]^2 \quad (2)$$

に対して最急勾配法を用いる．ただし，$P(\sigma;\omega)$はエピソードσの生成確率で，$V^{\pi}(s)$はλ収益

$$R_t^{\lambda} \equiv (1-\lambda) \sum_{n=1}^{L-t-1} \lambda^{n-1} R_t^{(n)} + \lambda^{L-t-1} R_t \quad (3)$$

で近似する．$R_t^{(n)}$は n ステップ収益であり，

$$R_t^{(n)} \equiv r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \cdots + \gamma^{n-1} r_{t+n} + \gamma^n V(s_{t+n}) \quad (4)$$

$$= \sum_{k=1}^{n} \gamma^{k-1} r_{t+k} + \gamma^n V(s_{t+n}) \quad (5)$$

により定義されている．(3)を(2)の $V^{\pi}(s)$ へ代入し，(2)の右辺をωで微分すれば，確率的降下法により各時刻 t ごとの前方観測的な更新式，

$$\omega_{t+1} = \omega_t + \alpha\left[R_t^{\lambda} - V(s_t;\omega_t)\right] \nabla_w V(s_t;\omega_t) \quad (6)$$

を得る．ただし，$\alpha(>0)$ は学習係数であり，$P(\sigma;\omega)$のω依存性は無視した．しかし，毎時刻ごとに学習を行うには，過去の情報だけを用いる後方観測的な更新式

$$\omega_{t+1} = \omega_t + \alpha \delta_t e_t \quad (7)$$

の方が都合がよい．ここで，δ_tは TD 誤差，

$$\delta_t = r_{t+1} + \gamma V(s_{t+1};\omega_t) - V(s_t;\omega_t) \quad (8)$$

である．e_tは適格度トレースの列ベクトル，

$$e_t = \gamma\lambda e_{t-1} + \nabla_w V(s_t;\omega_t) \quad (9)$$

であり，時刻ごとに加えて行けばよい．(7)の学習則は(6)の学習則と等価である[6].

2.2 予測勝利確率の関数近似

将棋において自己の t 回目の手番の局面を状態 s_t とし，終局時(t=L)における勝敗を z（勝てば z =1，負ければz =0）で表す．時刻は手番ごとに 1 ステップずつ経過するものとする．ここで，各時刻 t に与える報酬 r_t を，t=L においては r_t=z，それ以外の時刻では r_t=0 とする．このとき，(1)でγ=1 とおいた状態価値関数 $V^{\pi}(s)$は，局面 s において学習プログラムが勝利する確率の予測値 $P^{\pi}(s) \equiv E_{\pi}[z|s_t=s]$ （以下，予測勝利確率）と解釈できる．また，ωの更新は学習プログラムの手番ごとに行うものとする．

時刻 t における予測勝利確率 $P^{\pi}(s)$の近似関数を $P_t(s_t;\omega_t)$と表す．TD-Gammon[5]では階層型のニューラルネットワークモデル（ωはニューロン間の結合重み）が用いられたが，将棋では以下のシグモイド関数が用いられている[3][4].

$$P_t(s_t;\omega_t) = 1 \Big/ \left(1 + e^{-E(s_t;\omega_t)/\tau}\right) \quad (10)$$

Application of TD(λ) to a Shogi Program: Improvement of Evaluation Function Parameters of Bonanza
†Harukazu Igarashi, Shibaura Inst. of Tech.
‡Kazumasa Yamamoto, Shibaura Inst. of Tech.

ここで，$E(s;\omega)$は局面 s の静的評価関数である．ただし，文献[3],[4]ではτは用いられていない．これは，(10)で τ=1と設定したことと等価である．予備実験として Bonanza 同士の対戦を行った結果，Bonanza ver.4.1.3 の予測勝利確率を(10)のシグモイド関数で近似する場合は，τ〜1000 程度であれば良いことが分かった．

3. Bonanza の利用

3.1 Bonanza とは

Bonanza は，保木邦仁氏により開発された将棋ソフトである．探索アルゴリズムはチェスやオセロ等で一般的に用いられている手法に基づいた全幅探索であるが，将棋用に bitboard を用いた盤面構造の取り扱い，膨大な個数のパラメータを含む静的評価関数の使用，教師付学習によるパラメータ値の決定法を提案し，2006 年の世界コンピュータ将棋選手権で優勝している．現在，このソースコードと評価関数中のパラメータ値は Web 上で公開されている[7].

3.2 評価関数

Bonanza ver.4.1.3 の評価関数 E_B(s)は(11)のように表される．

$$E_B(s;w)=\sum_{j=1}^{N} w_j\left[x_j(s^1)-x_j(s^2)\right] \qquad (11)$$

ただし，関数 x_jは特徴量 j が局面に現れているときに 1，それ以外は 0 をとる．Bonanza ver.4.1.3 では，各駒の価値と，２種類の３駒の位置関係（①自分の王 1 駒と，相手の王を除く２駒の計３駒，②自分と相手の王の２駒と，自分の 1 駒の計３駒）を局面 s の特徴量と考え，評価関数は各特徴量の線形和で表されている．なお，２駒の位置関係は①の中に含まれている．

また，(11)の右辺において，s^1/s^2 は局面 s における先手/後手側から見た駒配置である．(11)の定義から，先手側が優勢であるときには E_B>0 となる．したがって，3.3 で述べる学習則を用いる際には，学習プログラムが先手であるときには，$E(s)=E_B(s)$とし，後手であるときにはマイナス符号を付けて $E(s)= -E_B(s)$として用いれば良い．

3.3 本研究で用いた学習則

本研究ではωの更新式として(7)を用いる．ただし，δ_t と e_t には $V(s;\omega)$ の代わりに(10)の $P_t(s_t;\omega_t)$を用いて次のようにした．

$$\delta_t = r_{t+1}+P_t(s_{t+1};\omega_t)-P_t(s_t;\omega_t) \qquad (12)$$

$$e_t = \lambda e_{t-1}+\nabla_w P_t(s_t;\omega_t) \qquad (13)$$

$$= \lambda e_{t-1}+\left[1-P_t(s_t;\omega_t)\right]P_t(s_t;\omega_t)\partial E/\partial\omega_t \qquad (14)$$

4. 学習における問題点と考察

4.1 探索量の低下

学習実験を行うには，Bonanza ver.4.1.3 の探索エンジンと評価関数を利用できる．また，公開されているパラメータωの値を学習時の初期値ω_0として用いることも可能である．

ここで，学習時には学習プログラム側の探索量の低下に留意する必要がある．対局中は自己の手番ごとに(7)によりωの値を更新させ，次の手番でそのωを用いることになる．したがって，手番ごとに学習の計算時間を消費してしまう．さらに，通常，学習プログラムは評価値計算を実数型で行う必要があり，整数型でこの計算を行っている元の Bonanza と比べて処理速度が低下する．我々の予備実験では，単位時間当たりの探索量が 3 割程度は減ってしまうことが分かっている．したがって，学習のために棋力の低下が生じてしまうので，学習時の対戦相手との棋力差をコントロールしたい場合は，何らかの工夫が必要であろう．

4.2 近似関数中のパラメータτの値

Bonanza の予測勝利確率が最適かどうかは自明ではない．したがって，予測勝利確率の近似関数 $P_t(s_t;\omega_t)$中のパラメータτの値は1000ではなく，様々に変えて学習実験を行う必要がある．

4.3 学習の目的

2章で述べた TD(λ)法では，学習の目的が予測誤差の最小化であり，勝率の最大化ではない．精度の高い予測勝利確率を与える評価関数を学習により獲得し，それを用いて探索を行うことが棋力向上へ貢献するとの期待からである．そこで，勝率の最大化自体を学習の目的として，TD(λ)法以外の強化学習法，例えば方策勾配法などの適用も現在検討中である．

文献

[1] 伊藤毅志他，“ミニ特集：コンピュータ将棋の不遜な挑戦”，情報処理，vol.51, no.8, pp.986-1022(2010).

[2]保木邦仁，“局面評価の学習を目指した探索結果の最適制御”， 第 11 回ゲーム・プログラミングワークショップ (GPW2006), pp.78-83(2006).

[3] 佐々木宣介，飯田弘之，“将棋種の歴史的変遷の解析”，情報処理学会論文誌， vol.43, no.10, pp.2990-2997(2002).

[4] 薄井克俊，鈴木豪，小谷善行，“TD 法を用いた評価関数の学習”，第 4 回ゲーム・プログラミングワークショップ (GPW1999), pp.31-38(1999).

[5] G.J.Tesauro, “TD-Gammon, a self-teaching backgammon program, achieves master-level play,” Neural Computation, vol.6, no.2, pp.215-219(1994).

[6] R.S. Sutton and A.G. Barto, Reinforcement Learning, Chapter 7, The MIT Press, 1998.

[7] Bonanza のソースコードの入手先， http://www.geocities.jp/bonanza_shogi/