方策勾配法とαβ探索を組み合わせた強化学習アルゴリズムの提案

森岡祐一[†1]　五十嵐治一[†2]

コンピュータ将棋において, プロ棋士らの棋譜から評価関数のパラメータを学習する Bonanza Method は広く用いられている. しかし, 我々はプロ棋士の棋譜の様な人間の専門知識を極力用いないで, 自己対戦だけで学習を行えないかと考えた. 本論文では強化学習の基本アルゴリズムである REINFORCE をベースとし, αβ探索と相性の良い学習アルゴリズム PGLeaf を提案する. また,5 五将棋において既存手法である TDLeaf(λ) 等との比較実験を行い, 提案手法の有効性を検証する事が出来た.

Reinforcement Learning Algorithm that Combines Policy Gradient Method with Alpha-Beta Search

YUICHI MORIOKA [†1] and HARUKAZU IGARASHI[†2]

The bonanza method, which is widely used in computer programs that play shogi (Japanese chess), is a supervised learning algorithm that uses the databases of the past games between professional shogi players. We propose a new learning algorithm called PGLeaf that reinforces an evaluation function of states merely by self-play without using the databases of past games. PGLeaf combines a reinforcement learning algorithm called REINFORCE and alpha-beta search. We compared PGLeaf and three traditional reinforcement learning algorithms, REINFORCE, TD(λ), and TDLeaf(λ), in learning experiments with 5x5 shogi, a small and simplified version of the game. Our experimental results verified the effectiveness of our proposed method.

1. はじめに

あから 2010 が女流棋士に, ボンクラーズが元名人に公開対局で勝利するなど, 最近のコンピュータ将棋の棋力向上には目覚しいものがある. しかし, 棋力がトッププロを上回る為には更なるブレイクスルーが必要であると考えられる. 一般に, コンピュータ将棋の棋力向上の為には高速化・探索の効率化・評価関数の強化が必要とされている. 本論文では, 評価関数の強化による強いプレイヤーの実現を目標とし, その為に自己対戦の経験を元にした強化学習[1] による評価関数のパラメータ調整を行う. コンピュータ将棋における評価関数のパラメータ調整手法としては,Bonanza Method[2] と呼ばれるプロ棋士らの棋譜を用いる教師あり学習の手法が広く用いられている.

一方, コンピュータチェスの分野では, 強化学習の一手法である TDLeaf(λ)[3] によって評価関数のパラメータ調整を行い,US Master と同程度まで棋力が向上したとの報告がある. しかし,TDLeaf(λ) をコンピュータ将棋に適用して良い結果を得たとの報告はまだされていない. 特に,TDLeaf(λ) では局面ごとの勝率予測の精度を高める事が目的であり, 報酬も勝敗を表す信号値に限定する必要があった. そこで, 方策勾配法（Policy Gradient Mehtod）をベースとして, 目的に応じて自由に報酬設定を行う事が可能な強化学習の手法 PGLeaf を提案する. 具体的には, 方策勾配法の基本アルゴリズムである REINFORCE[4][5] をベースに, 行動選択及びパラメータ調整に最善応手手順（Principal Variation）の末端局面である Principal Leaf を用いる事で, αβ探索と相性の良い評価関数のパラメータ獲得を目指す.

本研究では最終的には本将棋に提案手法を適用する事を目標としているが, 本論文では基礎データ収集を目的として 5 五将棋で学習を行った. これは, 合法手数・終局までの手数が共に少ない為, 学習・対局実験に必要な時間が短くてすむからである.

2. 先行事例

強化学習とは, エージェントと環境の相互作用によって学習を進めるアルゴリズムである[1]. エージェントは環境の中で様々な行動を選択し, どの様な方策（Policy）

†1 y.morioka.1979@gmail.com
†2 芝浦工業大学工学部情報工学科
Shibaura Institute of Technology
arashi50@sic.shibaura-it.ac.jp

に従って行動すればより多くの報酬を得られるかを学習するのが目的である.

強化学習のボートゲームへの応用で成功をおさめた例としては, サミュエルのチェッカープレイヤー[6], バックギャモンにおける TD-Gammon[7], チェスにおける KnightCap[3] 等がある. 一方, 将棋においては薄井ら[8] や Beal ら[9] の適用事例がある. しかし, これらの例はいずれも状態価値関数の計算を行う TD 法を用いている.

3. 主な強化学習手法

3.1 TD(λ) 及び TDLeaf(λ)

TD(λ)[1] とは評価値の時間差分を用いる学習法である. n 手目の局面 s_n の評価値 $V(s_n)$ と, それ以降の局面 $s_{n+i}(1 \leq i)$ の評価値 $V(s_{n+i})$ と報酬とから見積もられた $V(s_n)$ の目標値 $V'(s_n)$ との差を時間差分と呼び, この値を元に s_n の評価値を見積もる手法である.

TDLeaf(λ) 法[3] は TD(λ) 法と α β 探索を組み合わせたアルゴリズムである. 両者の違いは手 a を指した直後の局面の評価値を更新するか, 指した後に探索した PV 末端局面での評価値を更新するかという事である. この違いにより,TD(λ) 法での更新則と α β 探索を自然に繋ぎ合わせる事が可能になり, 深い探索結果を評価関数の学習に反映させる事が可能となった.

3.2 方策勾配法（REINFORCE）

3.1 節の手法は, 正確な評価関数の学習を目的としている. 方策勾配法はこれらの手法とは異なり, 価値関数の精度向上だけではなく, 報酬を最大化する方策の獲得を目標としている. また,TD 法及びその派生アルゴリズムは環境がマルコフ決定過程（MDP）である事を前提としているが, 方策勾配法は非 MDP でも動作する[10].

方策勾配法の基本アルゴリズムである REINFORCE では, まず下記の式で方策 π の勾配を推定する.

$$\nabla_\theta \widehat{J(\theta)} = \frac{1}{M}\sum_{m=1}^{M}\left(\gamma^{T_m-1}r_m - b*\right)g(m) \tag{1}$$

$$b* = \frac{\sum_{m=1}^{M}\gamma^{T_m-1}r_m g(m)^2}{\sum_{m=1}^{M} g(m)^2} \tag{2}$$

$$g(m) = \sum_{t=1}^{T_m}\nabla_\theta log\pi(a_{m,t}|s_{m,t};\theta) \tag{3}$$

M はパラメータを修正する間隔（エピソード数）,T_m は m 回目の対局における行動回数,$\pi(a_{m,t}|s_{m,t};\theta)$ は状態 $s_{m,t}$ において行動 $a_{m,t}$ を選択する確率である. γ は割引率と呼ばれるパラメータであり,0 より大きく 1 以下の値を設定する.

次に,M エピソードが経過するたびに, 以下の式で方策のパラメータ θ を更新する.

$$\theta = \theta + \alpha\nabla_\theta \widehat{J(\theta)} \tag{4}$$

α は学習率であり, 小さな正の数を設定する.

4. 提 案 手 法

4.1 概　　要

Baxter ら (1998) の実験から, 強いコンピュータゲームプレイヤーの構築には, 評価関数のパラメータ学習時にも先読みが有効だと考えられる. そこで, 方策勾配法と α β 探索を組み合わせる PGLeaf（Policy Gradient with Principal Leaf）を提案する. 通常の方策勾配法とは異なり,PGLeaf では評価値の計算及び方策勾配の計算に, α β 探索を行った PV 末端局面を用いる.

本論文では, 方策としてボルツマン分布によるソフトマックス方策[1] を用いる. ソフトマックス方策とはランダムに指し手を選択する方策であるが, 指し手の評価値が大きい手ほどより高い確率で選択される手法である.

4.2 学習の流れ

PGLeaf における学習の流れを以下に示す。

i) 価値関数のパラメータをごく小さな乱数で初期化する.

ii) 以下 (a)〜(b) を繰り返す.

(a) 自己対戦で M 局対局する.

(b) 方策勾配を元にパラメータ修正を行う.

4.3 パラメータ修正方法

まず,$evl(s,a;\theta)$ を「状態 s で行動 a を選択した後の局面から, 事前に設定した深さまで探索して得た PV 末端局面での, パラメータ θ を用いて計算した評価値」として下記の様に定義する.

$$evl(s,a;\theta) = \sum_{n=1}^{N}(\theta_n \cdot \phi_n(\tilde{s};s,a)) \tag{5}$$

なお, 式中の θ は「評価関数のパラメータベクトル」,$\phi(\tilde{s};s,a)$ は「局面 s で手 a を指した後の局面から α β 探索を行なって得た PV 末端の局面 $\tilde{s}$ の特徴量ベクトル」,N は「特徴量ベクトルの次元数」である.

次に, 方策は次のボルツマン分布を用いる.

$$\pi(a|s;\theta) = \frac{exp\left(\frac{evl(s,a;\theta)}{T}\right)}{\sum_{a'\in A} exp\left(\frac{evl(s,a';\theta)}{T}\right)} \tag{6}$$

なお, 式中の A は局面 s における合法手の集合,T は温度パラメータである.

次に,(3) の右辺の方策勾配を計算する. ここでは, 方策パラメータは線形近似のパラメータ θ のみとし, 温

度 T は定数とする.

$$\nabla_\theta log\left(\pi\left(a|s;\theta\right)\right) = \frac{1}{T}\left(\phi(s,a) - \sum_{a'\in A}\left(\pi\left(a'|s;\theta\right)\phi\left(s,a'\right)\right)\right) \quad (7)$$

勾配の推定及びパラメータ更新則は式 (1) 及び (4) に従って行う.

5. 比 較 実 験

PGLeaf・REINFORCE・TDLeaf(λ)・TD(λ) の四つのアルゴリズムを,5 五将棋（将棋盤の大きさを縦横 5 マスとし, 駒の種類・数を減らした将棋の派生ゲーム)[11] を用いて比較する.

5.1 学習時の設定

報酬の設定は以下の二通りとする.

MDP 的な環境 勝った側に+1, 負けた側に-1, 引き分けなら 0 を終局時に与える.

非 MDP 的な環境 勝った側に+1+囲いの駒のボーナス, 負けた側に-1+囲いの駒のボーナス, 引き分けなら双方 0 を終局時に与える.

「囲いの駒のボーナス」は, 初期局面から終局局面までの間に常に盤上にあり, かつ終局時に王将の 8 近傍に存在した金将・銀将 1 枚あたり+0.1 を報酬に加算するものとする. 従って, 終局面だけではなく初期局面からの履歴に依存するので報酬のマルコフ性が無く, 非 MDP 的な環境である.

学習は自己対戦を用いて行い, 先手エージェントと後手エージェントが同一の評価関数を共有する条件下で行う. メタパラメータは予備実験において最適であった値を用いて, 下記の通り設定した.

探索深さ PGLeaf 及び TDLeaf(λ) で全幅探索 1 手＋静止探索 4 手

学習率 全てのアルゴリズム・報酬設定で 0.001

γ 全てのアルゴリズム・報酬設定で 1

温度 PGLeaf 及び REINFORCE は 0.05,TDLeaf(λ) 及び TD(λ) は 0.03

M PGLeaf 及び REINFORCE で 100

λ TD(λ) 及び TDLeaf(λ) で 0.7

評価関数は下記の評価項目を用いた線形の関数とした. パラメータ数は合計 8033 個である.

- 駒割
- 盤上の駒の位置
- 盤上の二つの駒の相対位置
- 飛び利きを遮る駒の種類
- 駒の自由度
- 王将の移動可能なマスの数

評価関数のパラメータは駒の価値を以下の値に設定し, それ以外のパラメータはごく小さな乱数で初期化した.

表 1 駒の価値

	歩	銀	金	角	飛車
生駒	0.02	0.08	0.10	0.16	0.20
成駒	0.10	0.10	-	0.26	0.30

5.2 対局実験時の設定

学習後に棋力比較のための対局実験を行った。実験条件を以下のように設定した。

対局数 各 400 局

探索アルゴリズム 学習時と同じ探索ルーチンを用い, 反復深化を行った

思考時間 1 手 1 秒で打ち切り

探索深さ 反復深化を行い, 思考時間内で可能な限り先読みを行った

乱数 同一手順での対局の出現を抑制する為, 平均 0, 分散 0.001 の正規乱数を評価値に加算する

5.3 MDP 環境の場合

上記 4 種類のアルゴリズムの学習途中の 1000 局,1 万局,10 万局の時点でのパラメータを用いて総当たり戦を行った場合の勝率は下記の通りとなった. 各マスの数値は, 上の見出しに書いたアルゴリズムに対する, 左のアルゴリズムの勝率である. なお, 優位水準 1 %で強くなっていると判断出来る結果は, 表中では太字で示した.

表 2 1000 局終了後の評価実験結果

	PL	RE	TL	TD
PGLeaf	-	**77.0**	24.3	42.0
REINFORCE	23.0	-	10.5	16.8
TDLeaf(λ)	**75.7**	**89.5**	-	**67.8**
TD(λ)	**58.0**	**83.2**	31.2	-

表 3 1 万局終了後の評価実験結果

	PL	RE	TL	TD
PGLeaf	-	**88.3**	44.1	**88.8**
REINFORCE	11.7	-	12.3	52.6
TDLeaf(λ)	55.9	**87.7**	-	**89.3**
TD(λ)	11.2	47.4	10.7	-

表 4 10 万局終了後の評価実験結果

	PL	RE	TL	TD
PGLeaf	-	**76.2**	**56.2**	**87.9**
REINFORCE	23.8	-	16.0	31.3
TDLeaf(λ)	43.8	**84.0**	-	**87.0**
TD(λ)	12.1	**68.7**	13.0	-

次に, 上記 4 アルゴリズムの学習前・1000 局終了後・

1 万局終了後・10 万局終了後の各パラメータを,5 五将棋に対応した思考エンジンである ssp [*1] と対局させた結果を以下に示す. 対局の条件は総当たり戦と同じとし, 勝率（%）の推移を図 1 に示す.

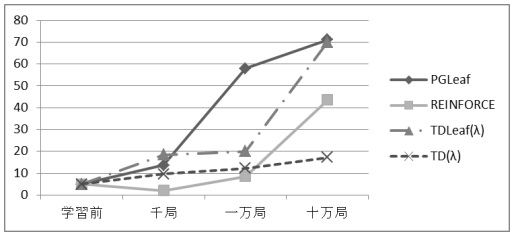

図 1 MDP 的環境での対 ssp の勝率推移

5.4 非 MDP 環境の場合

MDP の場合と同様に ssp との対局実験を行った結果, 図 2 の通りとなった.

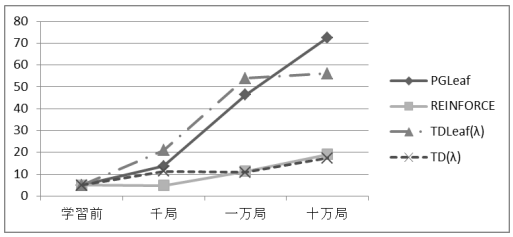

図 2 非 MDP 的環境での対 ssp の勝率推移

6. 考　　察

MDP 環境での 4 アルゴリズムの総当たり戦（表 2〜表 4）においては,PGLeaf は REINFORCE に対して高い勝率を示し,PV 末端局面を学習に用いた優位性が伺える. また,PGLeaf と TDLeaf(λ) との対局では学習初期は勝率が低いものの, 最終的には PGLeaf は TDLeaf(λ) に対して僅かながらも高い勝率を示した. 一方, 対 ssp での勝率（図 1）を見ると,1 万局終了後の勝率は PGLeaf が飛び抜けて高く, 学習の収束が速い事を示唆している.

非 MDP 環境では 1 万局終了時までは TDLeaf(λ) が PGLeaf をやや上回るが,10 万局終了後の対局では PGLeaf が TDLeaf(λ) より明らかに強くなっていた（図 2）. これは, 非 MDP 環境でも動作する方策勾配法の特徴をよく表していると考えられる. また, 対 ssp の勝率を通して MDP 環境（図 1）と非 MDP 環境（図 2）での学習結果を比較すると,TDLeaf(λ) は非 MDP 環境の方が弱くなっているのに対し,PGLeaf は非 MDP 環境でも変化は無く, これも非 MDP 環境に強い方策勾配法の特徴をよく表している.

7. 今後の課題

本論文ではデータ採取の容易さから 5 五将棋における評価関数パラメータの学習を行った. その結果,MDP 環境では PGLeaf は TDLeaf(λ) よりも学習の収束が速く, 非 MDP 環境では最終的な棋力が大きく上回る結果となった. 今後は本将棋でも同様な学習実験を行う予定である.

その際, 攻守のバランスを取る為にも, 上手く攻めた場合にボーナスを与える等の報酬設定を上手く行う事が重要である.

参　考　文　献

1) Richard S.Sutton and Andrew G.Barto : *Reinforcement Learning An Introduction*, The MIT Press, Massachusetts(1998). 和訳 三上 貞芳, 皆川 雅章 :「強化学習」, 森北出版 (2000).
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, pp.78-83(2006).
3) Jonathan Baxter, Andrew Tridgell, Lex Weaver : *TDLeaf(λ): Combining Temporal Difference Learning with Game-Tree Search.*, In Proceedings of the 9th Australian Conference on Neural Networks (ACNN-98), pp.39–43(1998)
4) Williams, R. J. : *Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning*, Machine Learning, 8, pp.229–256(1992).
5) 八谷 大岳, 杉山 将 :「強くなるロボティック・ゲームプレイヤーの作り方 実践で学ぶ強化学習」, 毎日コミュニケーションズ (2008).
6) Samuel A. L : *Some studies in machine learning using the game of checkers.* IBM Journal on Research and Development, 3, pp.221–229(1959)
7) Tesauro G. J : *Practical issues in temporal difference learning.* Machine Learning, 8, pp.257–277(1992).
8) 薄井 克俊, 鈴木 豪, 小谷 善行 :「TD 法を用いた将棋の評価関数の学習」, ゲームプログラミングワークショップ ’99, pp.31–38(1999).
9) Donald F. Beal, Martin C. Smith : *Temporal difference learning applied to game playing and the results of application to shogi*, Theoretical Computer Science, Vol.252, pp.105-119(2001)
10) 五十 嵐治一, 石原 聖司, 木村昌臣 :「非マルコフ決定過程における強化学習—特徴的適正度の統計的性質—」, 電子情報通信学会論文誌 D, Vol.J90-D, No.9, pp.2271-2280 (2007).
11) http://ja.wikipedia.org/wiki/5 五将棋

⋆1 http://www.geocities.jp/shogi_depot/#7