将棋における Tree Strap に基づく評価関数の学習

宇賀神拓也[†1]　小谷善行[†2]

近年，コンピュータ将棋では 1 万を超える評価項目を持った評価関数を用いることが一般的となっている．大量の評価項目それぞれに対し適切な重みを決定することは人手では非常に困難である．本論文ではコンピュータチェスにおいて成功を収めた自己対局を用いて評価関数を学習する Tree Strap を将棋に適用し，評価関数の学習を行った．実験では駒の重み, 玉の安全度などを学習させ, 得られる評価関数について考察を行った．また学習前の評価関数との対局実験を行うことで Tree Strap の有用性を調べた．実験の結果，学習前の評価関数に対し 109 勝 26 敗 5 分と勝ち越したことからコンピュータ将棋においても有用な手法であることが分かった．

Learning Evaluation Function Based on Tree Strap in Shogi

Takuya UGAJIN[†1] and Yoshiyuki KOTANI[†2]

Recently, many Shogi programs use evalation function that has many features. However it is difficult to determin many parameters of evaluation functions by heuristic. In this paper we apply Tree Strap method which learns evaluation functions using self-games and is succeeded in chess programs to Shogi. We used the weight of pieces, pieces in hand, the safety of the King and so on as features . We examin the learning the parameter of piece, piece in hand, the safety degree of the King and so on as parameters and discussed about the learned evaluation functions. We discuss the the learning result.In addition we performed the match experiments between the program which uses the learned parameters and the program which uses the old parameters in order to examine the effectiveness of the method. The program uses the learned parameters 109 wins, 26 loss and 5 draws against the program uses the old parameters. The results of the experiments show that the method is also useful for Shogi programs.

1. はじめに

ゲームプログラムを作成する上で重要な要素のひとつとして, 評価関数がある．評価関数はゲームの性質を正確に表現する必要があり, 適切な評価項目を設定し, それぞれの評価項目がどの程度重要か重みを決める必要がある．将棋では 1 万を超える評価項目を持つ評価関数を用いたプログラムが成功をおさめ一般的に使用されている．評価項目の少ない評価関数であれば人手により適切な重みを設定することは可能であるが, 1 万を超える特徴それぞれに適切な重みを与えることは人手では事実上不可能である．したがって機械学習によって重みの学習をおこなうことは重要な課題であると考えられる．

近年のコンピュータ将棋ではプロの棋譜を用いた兄弟局面学習が成功を収めており, コンピュータ将棋選手権上位のプログラムのほとんどは兄弟局面学習により評価関数の学習をおこなっている[6)7)]．一方, プロの棋譜のような教師となる情報を用いない手法として TD 学習があげられる．将棋において TD 学習を用いて評価関数の学習をおこなった研究は Beal や薄井により行われているが強いプログラムにおいて実用された例は少ない[4)9)]．Veness は探索した各局面の評価値の範囲に評価関数を学習する Tree Strap を提案し, チェスにおいて TD-leaf よりもすぐれた結果を残した．また,Tree Strap により学習した評価関数を用いたプログラムはマスターレベルの強さであった[3)]．そこで、本論文では Tree Strap を将棋に適用し評価関数の重みの学習を行い有用性を調べた．

2. 関連研究

ゲームプログラムの評価関数の学習の研究としてはバックギャモンにおいて,TD 学習を用いた Tesauro や, オセロにおいて最小二乗法を用いた Buro の研究があげられる[1)5)]．将棋の評価関数の教師ありの学習例として兄弟局面学習が挙げられる．兄弟局面学習は教師

†1 東京農工大学大学院　工学府　情報工学専攻
Department of Computer and Information Sciences Graduate School of Engineering
†2 東京農工大学　工学研究院　先端情報科学部門
Institute of Engineering Tokyo University of Agriculture and Technology

となるプロの着手を実際に探索が指す方向に重みを学習させることにより評価関数の学習に成功した．また, 教師となる棋譜がないゲームおいても深い探索によって得られた着手を教師とする手法が柿木により提案され 5 五将棋において成功を収めている[11]．柿木の手法は 5 五将棋以外のゲームにおいても一定の成果を収めていることが築地によって示された[10]．一方教師なしの学習例として TD 学習がある．Beal は TD 学習を用いて駒価値の学習を行い, 概ね一般的な駒価値の学習を行うことができたとの報告がある[4]．さらに薄井は駒価値以外にも玉の安全度などを考慮した評価関数を学習させた[9]．また TD 学習を行う際, 評価関数を呼ぶのではなく探索を行った結果を利用する TD-leaf が Baxter により提案された[2]．Baxter はプログラムと人間を対局させることにより学習を行い, レーティングを 500 程度向上させることに成功した．しかし TD-leaf は探索を行った際の最善応手手順のみを用いており, その他の探索木の情報を捨てている．本手法では最善応手手順のみではなく探索のすべての部分木においても学習を行う．これにより既存の TD-leaf と比較してより多くの局面を学習対象とするため探索結果を無駄にせず効率の良い学習ができると考えられる．また, 探索中には棋譜などでは表れないような局面も表れることから, スパースネスの問題に対しても回避できると考えられる．

3. 学 習 手 法

3.1 Tree Strap

Tree Strap について述べる．また図 1, 図 2 に具体的な手順を示す．

本手法は前述の柿木の手法と同じように探索を行った結果を近似する評価関数の作成を目的とする．また学習の際, 最善応手手順以外にも探索木の各部分木についても学習対象とすることにより既存の TD-leaf と比較して最善応手手順以外の探索木を利用することで効率の良い学習を目指す．

具体的には,MinMax 法により探索を行った場合,root ノードおよび内部ノードでは leaf ノードで求めた評価値がのぼっていくため各ノードで探索によって得られた評価値を求めることができる．そして, 探索木の各ノードで評価関数を呼んだ際の評価値を探索によりのぼってきた評価値に近づけるよう重みを学習する．しかし, 一般的に使用される α β 探索を行った場合, 枝刈りが発生し各ノードの正確な値を示すことのできるノードは非常に少ない．だが, 各ノードの評価値の範囲を α 値, β 値から求めることはできるため評価関数がノードの評価値の範囲から外れた場合のみ, 重みの更新を行う．つまり探索木内部の各ノードで評価関数を呼び, ノードの評価値の上界より大きい場合, 上界と評価関数の値の誤差を小さくなるよう, 下界より小さい場合, 下界と評価関数との誤差を小さくなるよう重みの更新を行う．

時刻 t における目的関数は

$$\delta_t^a(s) = \begin{cases} a_{s_t}^D(s) - H(s) & \text{if } H(\mathrm{s}) > a_{s_t}^D(s) \\ 0 & \text{otherwise} \end{cases}$$

$$\delta_t^b(s) = \begin{cases} b_{s_t}^D(s) - H(s) & \text{if } H(\mathrm{s}) < b_{s_t}^D(s) \\ 0 & \text{otherwise} \end{cases}$$

$$OF = \sum_{s \in T_t} \delta_t^a(s)^2 + \delta_t^b(s)^2 \quad (1)$$

となる．ただし $a_{s_t}^D(s)$,$b_{s_t}^D(s)$ は局面 s を深さ D で探索した際の探索木 T の各ノード s の評価値の上界, 下界を示し, H(s) は局面 s での評価関数によって得られた評価値を示す．H(s) は w と $\phi(s)$ の線形和により表されている．目的関数は各ノードの探索により得られた評価値の範囲と各ノードで評価関数を呼んだ際の評価値の平均二乗誤差を示している．したがって目的関数を最小にすることにより探索を行った際の評価値と探索を行わない際の評価値の誤差が減ることになる．つまり探索を行うことなく探索を行った場合の評価値を得られるよう重みの学習を行うことになる．

3.2 重みの更新

目的関数を最小にするため最急降下法により重みの更新を行う．更新のため, 各重みの勾配を求める．重み w の勾配は

$$\frac{\partial OF}{\partial w} = -2 \sum_{s \in T_t} (\delta_t^a(s) + \delta_t^b(s))\phi(s) = \Delta w \quad (2)$$

により得られる．したがって, 重み w の更新は,

$$w = w + \alpha \Delta w \quad (3)$$

となる．ただし α は学習率である．

重みの更新は対局中に着手を選択するため探索を行うごとに更新を行うため一対局分の情報を保持する必要が無く学習を行うことができる．また最善応手手順のみではなく探索木それぞれの部分木についても学習が行われるため効率よく学習することができる．

3.3 学習の対象とするノード

本論文では静止探索部分のノードについては学習の対象とはしなかった．それは静止探索部分も学習の対象として実験を行ったが静止探索部分を学習の対象としないものと比較して実験結果についてちがいが見られなかったためである．また静止探索も含めて学習を行った場合一回の重みの更新の際に対象となるノードが多く, 学習に時間がかかるため学習の対象とはしなかった．

4. 実　　験

4.1 駒価値の学習

本手法の予備実験として, 駒価値のみの評価関数を作成し駒価値の学習を行った．学習条件を図 3 に示す．

(1) 以下の (2)-(5) を規定回数に達するまで繰り返す
(2) α β 探索を行い各ノードの評価値の範囲を格納
(3) 各ノードでの評価関数の値を計算
(4) 式 (3) に従い重みを更新
(5) 探索によって得られた最善手により次の局面へ進む

図 1 Tree Strap

各重みの初期値として表 1 の値を与え学習を行った．実験の結果, 得られた学習曲線を図 4 に示す．縦軸は重みを示し, 横軸は学習のための自己対局を行った回数である．

図よりいずれの条件においても香車の価値が高い傾向が見られる．また歩の価値が高く全体的に駒価値が低い値となっている．成った駒に関しては龍がもっとも重みが大きくなっているが，その他の駒の重みは大きな違いが見られなかった．これは成り駒は成っていない駒と比較して，出現する頻度が少ないことから学習の機会が少ないためであると考えられる．

表 1 駒価値の初期値

飛	角	金	銀	桂	香	歩
500	500	500	500	500	500	500
龍	馬	成銀	成桂	成香	と金	
500	500	500	500	500	500	

4.2 対 局 実 験

本手法の効果の確認のため学習前との対局実験を行った．実験前の評価関数は表 1 に示した値である．また, 実験条件を図 6 に示す．実験結果を表 2 に示す．表 2 より, 学習後の評価関数が学習前の評価関数に対して 109 勝 26 敗 5 分という結果を残した．したがって本手法を用いることにより学習前と比較して精度の高い評価関数を作成することができたことがわかった．

表 2 学習前と学習後の対局結果

対戦内容 (先手 VS 後手)	学習後の勝ち	負け	引き分け
学習前 VS 学習後	48	19	3
学習後 VS 学習前	61	7	2

4.3 評価関数の学習

駒価値のみではなく持ち駒の価値など複数の評価項目を持つ評価関数の学習を行った．学習条件は前述した条件と同じである．追加した評価項目を図 7 に示す．学習により得られた結果を図に示す．飛や角などの自由度は自由度が 1 や 2 のときに価値が大きくなっている．これは学習回数がまだ十分ではなく自由度が 5 や 6 などの場合が少ないため出現頻度の多い自由度が 1 や 2 のときの値が大きくなったと考えられる．また, 持ち駒の価値はいずれの駒も 1 枚持っている際の重みが大きく下がっている．これは Tree Strap は探索結果を利用して学習を行うため，終盤での水平線効果の手を学習して，駒を捨てる手の価値が高くなるように学習した結果であると考えられる．今回の実験では王手延長のみを行ったが，駒を取り返す場合などにも延長を行い，水平線効果を軽減する必要がある．

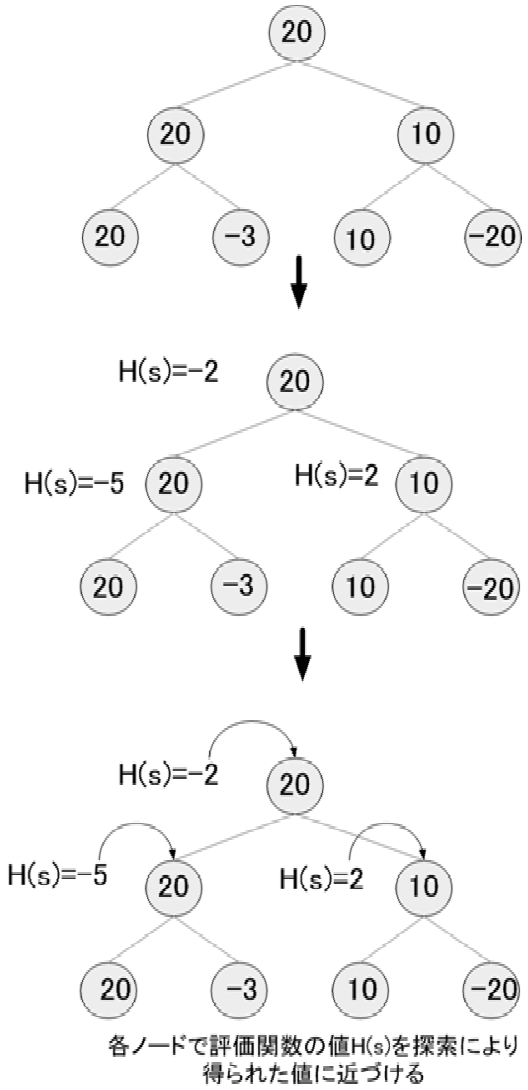

図 2 Tree Strap の図

玉の周りの利きの価値は利きが勝っている際に価値が高くなり，利きが負けてい際，価値が低くなった．これは玉の周りが安全な場合価値が高く，危険になっている場合価値が低くなっていると考えられるため正

- 学習に用いる探索は通常の $\alpha\beta$ 探索を用いて王手延長のみ行った
- 通常の探索では null move pruning など特殊な探索は行わない
- 静止探索は深さ 4 に制限した
- 学習率 α は 0.0001 で開始し学習回数に応じて減衰させた
- 評価関数には小さな乱数を加えた
- 探索を行う前に詰め探索を呼び, 詰みを発見した場合それ以降の局面の学習は行わない
- 探索中は詰め探索は行わない
- 初手から 20 手は定跡の中からランダムに着手を選択し, 学習は行わない

図 3 学習条件

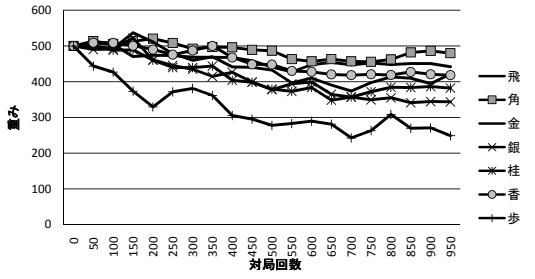

図 4 深さ 3 での学習曲線 (成っていない駒)

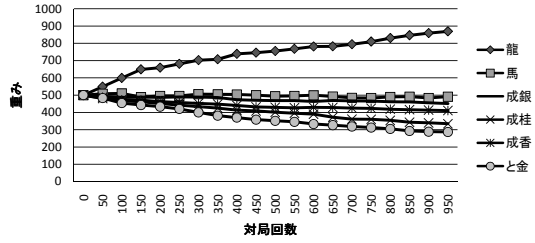

図 5 深さ 3 での学習曲線 (成った駒)

- 対局回数は先後入れ替えて合計 100 局
- 探索は深さ 1 でそれぞれ行った
- 初手から 20 手までは定跡の中からランダムに着手を選択する

図 6 実験条件

しく学習できていると考えられる.

- 持ち駒の枚数による得点
- 歩切れかどうか
- 王の周り 25 近傍の利きの勝ち負け
- 飛，龍の自由度 (縦・横)
- 角，馬の自由度 (前・後)
- 香の自由度
- 王の自由度
- 王が何段目か

図 7 追加した評価項目条件

表 3 飛，龍の自由度

自由度	0	1	2	3	4
縦	11.1	178.5	177.5	155.8	116.1
横	49.5	128.7	89.5	201.4	144.3
自由度	5	6	7	8	
縦	88.8	24.1	49.8	26.0	
横	95.9	72.5	32.9	13.1	

表 4 角，馬の自由度

自由度	0	1	2	3	4
前	150.4	169.6	50.3	88.6	-7.6
後	149.1	203.9	131.4	91.4	29.5
自由度	5	6	7	8	
前	26.3	22.1	7.5	-4.6	
後	-20.6	-36.9	-32.8	-12.6	

表 5 玉，香の自由度

自由度	0	1	2	3	4
玉	-10.2	-4.7	-45.1	68.7	55.1
香	-24.6	59.6	51.4	-13.9	-28.5
自由度	5	6	7	8	
玉	-43.3	-7.5	-9.4	-3.6	
香	-24.4	-10.0	-0.5	-2.7	

表 6 持ち駒の枚数による得点

持ち駒の枚数	1	2	3	4	5
飛	-460.16	-25.76			
角	-442.80	-45.01			
金	-133.60	-31.97	0.67	-0.27	
銀	-105.74	-29.63	4.26	0.35	
桂	-53.03	34.17	8.90	0.26	
香	-69.66	-11.82	-1.8	0.84	
歩	-67.32	-60.00	-37.26	-4.2	20.26

5. おわりに

本論文では Tree Strap を将棋に適用し駒価値の学習を行った．また玉の安全度や駒の自由度について学

表 7 玉のまわり 25 近傍の利きの勝ち負け (勝ち)

49.3	59.0	-49.8	10.1	11.8
41.5	2.0	-30.8	15.2	50.3
17.8	16.3	玉	-13.9	-7.4
20.2	-74.2	-70.9	-89.0	-24.0
-13.1	-20.5	-37.8	-27.9	-34.2

表 8 玉のまわり 25 近傍の利きの勝ち負け (負け)

-64.3	-33.9	9.3	-13.1	-51.5
-51.7	-5.3	-1.5	-4.2	-57.9
-3.8	-10.6	玉	4.5	-6.0
-61.7	-5.3	-1.3	-2.7	-53.4
-33.4	-39.4	-39.6	-42.5	-41.8

表 9 玉のまわり 25 近傍の利きの勝ち負け (引き分け)

18.1	12.3	44.9	24.5	34.5
12.8	40.1	36.2	10.3	2.3
-15.1	27.3	玉	26.9	4.5
-25.0	-9.0	-38.6	-13.6	-54.8
-43.9	-45.3	-42.7	-43.8	-41.6

習を行い実験の結果, 本手法が, 将棋においても適用できることを示した．また対局実験を行うことにより学習前の評価関数に勝ち越したことから本手法が有用に動作することが分かった．今後の予定として, 将棋で近年一般的に用いられている大規模な評価項目を持つ評価関数においても適用が可能であるか調査を行う予定である．また, 既存の TD 学習や TD-leaf との比較も行う必要があると考えられる．

参 考 文 献

1) Gerald Tesauro: Temporal Difference Learning and TD-Gammon, Communications of the ACM, Vol.38, pp.58-68, 1995.
2) Jonathan Baxter, Andrew Tridgell, Lex Weaver: Experiments in Parameter Learning using Temporal Differences, ICCA Journal, Vol.21, No.2, pp.84-99, 1998.
3) J.Veness, D.Silver, W.Uther, A.Blair: Bootstrapping from Game Tree Search, Advances in Neural Information Processing Systems 22, pp.1937-1945, 2009.
4) Donald F. Beal and Martin C. Smith: First Results from Using Temporal Difference Learning in Shogi, Proceedings of the First International Conference on Computers and Games, p.113-125, 1998.
5) M. Buro: Experiments with Multi-ProbCut and a New High-Quality Evaluation Function for Othello , Games in AI Research, 2000
6) 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 2006, pp.78-83, 2006.
7) 金子知適, 山口和紀: 将棋の棋譜を利用した, 大規模な将棋の評価関数の学習, 第 13 回ゲームプログラミングワークショップ 2008, pp.152-159,2008.
8) 金子知適: 兄弟節点の比較に基づく評価関数の調整, 第 12 回ゲームプログラミングワークショップ 2007, pp.9-16,2008.
9) 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.
10) 築地毅, 小谷善行: 自己対局による兄弟局面学習における汎用的制御の有効性, 第 14 回ゲームプログラミングワークショップ 2009, pp.127-134,2009.
11) 柿木義一: 5五将棋における評価関数の自動学習, エンターテイメントと認知科学研究ステーション第 5 回招待講会，2008.
12) 小谷善行: コンピュータ将棋の頭脳, サイエンス社，2007.