勝率と評価値の歪みに基づく評価関数調整法
– 将棋における進行度差の評価 –

竹内聖悟† 林　芳樹††
金子知適† 川合　慧†

本稿では、勝率と評価値の歪みに基づいた評価関数の調整法を提案し、将棋を例題に、本手法の有効性を示す。評価関数の調整は強いプログラムの作成に不可欠であるが、どこに問題があるか発見することや評価値を適切にあたえることはゲームの知識が必要であり困難が多い。本研究では、評価関数に問題がある局面では勝ち易さを適切に評価できず、勝率と評価関数との関係に歪みが生じていることに着目し、条件毎に勝率と評価値のグラフを描くことにより評価関数の問題点をを発見することを提案する。

本手法を将棋において先手と後手の進行度差がある局面に対して用い、プレイヤ毎の進行度を評価しない評価関数には問題があることを示した。さらに、その問題を解決するため、進行度差を評価に含めた評価関数を設計し、値の自動的な調整を行った。そして、自己対戦によって調整後のプログラムの棋力の向上を確認し、本手法の有効性を示した。

Adjustment of Evaluation Functions Based on Relation Between Static Values and Win Ratios
– Evaluation of Safety Difference Between Both Kings in Shogi –

Shogo Takeuchi,† Yoshiki Hayashi,†† Tomoyuki Kaneko†
and Satoru Kawai †

In this paper, we present a new method for adjusting evaluation functions based on relation between static values and win ratios, and show its effectiveness. Accurate evaluation functions are important for strong game program, however, it is not easy to find out problems in existing evaluation functions. Incorrect evaluation functions assign incorrect prediction of win ratio for states in a certain condition. Therfore, we focus on relation between evaluation values and win ratio, and propose to plot them in a graph for each set of states. If an evaluation function works bad for states in a certain condition, a line of the relation for those states will be drawn apart from lines of the relation for other states.

We applied this method for Shogi, and showed that usual evaluation functions work bad for states where the difference between king safety of both players is large. Then, we constructed a new evaluation function considering the difference between king safety of both players and automatically adjusted its weights. Significant improvement on strength is confirmed in self-play, and showed effectiveness of this method.

1. はじめに

評価関数は、与えられた局面の勝ち易さを局面の特徴である評価項目とその重みで評価するもので、強いゲームプログラムを作るために良い評価関数が必要となる。良い評価関数を作るには、評価項目の選択と重みの調整が重要である。

† 東京大学大学院総合文化研究科
Department of General Systems Studies, Graduate School of Arts and Sciences, The University of Tokyo
{takeuchi,kaneko,kawai}@graco.c.u-tokyo.ac.jp
†† グーグル株式会社
Google Japan Inc.

これまで評価関数の調整は多くが人の手で行われており、囲碁や将棋のように複雑な形勢判断が必要とされるゲームにおいて自動調整が明確に成功を収めたという論文は少ない。手動での調整には、どの特徴をどの程度に評価すべきか、どこに問題点があるかの判断がゲームに強いプログラマーでなくては難しいという欠点がある。また、調整結果を確認するために調整前と調整後のプログラムで問題集の正答数の比較、自己対戦などの時間がかかる手法が必要となる上、問題点を解決できたかの確認が難しいことも多い。

評価関数に問題があるか、特徴の評価が適当かなどを容易に確認できるようになれば、評価関数の調整、問題点の解決が容易となりプログラムの強さを向上さ

せられる上に、調整結果の確認も容易になり、作業時間の短縮にもつながる。

評価関数に問題があるということは、局面の勝ち易さを適切に評価できていないということで、勝ち易さを勝率と考えると、勝率と評価値との間に歪みが生じていると言える。

そこで、本稿では評価値と勝率の歪みに基づき評価関数の問題点や評価項目の重みの不適切さを発見する手法を提案する。さらに将棋において、プレイヤー毎の進行度差を評価する枠組に対してその手法を利用して調整を行い、自己対戦により調整後の評価関数を利用したプログラムが調整前の評価関数を利用するプログラムに対して有意に勝ち越すことを確認し、本手法の有効性を示した。

以下に、本論文の構成を述べる。2 章では、評価関数に関する関連研究を述べ、3 章では調整の対象とした進行度について述べる。4 章では本手法を具体的に説明し、5 章で実験とその結果を示し、6 章で結論を述べる。

2. 関連研究

将棋プログラムの評価関数では、序盤は駒の損得、終盤では玉の危険度が重要と言われている。他にも玉との相対位置を考え、近付くほど評価を高くするという手法は多くのプログラムで用いられている[10)12)8)]。また、近年では絶対テーブルを利用する手法も用いられている[13)11)]。これらの評価項目の値はプログラマがその知識によって決定している。

値を自動的に調整する手法もあり、そのうちの一つは、局面と理想の評価値を集めてそれを実現するように重みを調整する手法である。この手法はオセロにおいて成功を収めた。その手法とは、終局時の石の数の差を勝敗の差の度合と考え、それを元に終盤用の評価関数を調整し、中盤と序盤の評価関数を探索結果を元に調整する手法である[3)]。しかし、将棋ではオセロの終局時の石の数のような明確な勝敗の差を表すものは知られておらず、理想の評価値を得ることが難しいためこの手法を応用するのは難しい。

他に、理想の評価値を必要とせずに自動調整した例に、TD 法[4)] を用いて駒の価値を学習した例がある[2)]。しかし、これは駒の価値のみの学習であり、強いプログラムでの利用例もない。

3. 進行度

進行度とはゲームの進み具合を表すものである。多くのゲームではゲームの進み具合によって性質が変化することがあり、変化に応じて探索や評価関数を切替えることなどに進行度は用いられる。本研究で扱う将棋では、人間にとっては序盤中盤終盤の三つに大きく分けられる。序盤では駒得が重視され、終盤では駒得よりも玉の危険度や攻めの早さが重視されるようになり、中盤はそれらの中間となっている。多くの将棋プログラムでも進行度は用いられ[7)8)10)]、駒の位置や持駒の数、玉の危険度などが要素として用いられる。具体的な使われ方は、

$$\text{評価関数} = (1 - \text{進行度}) \times \text{序盤評価値} + \text{進行度} \times \text{終盤評価値} \qquad (1)$$

などとして、序盤用の駒得重視の評価関数と終盤用の危険度重視の評価関数との配分を決めるという方法がある。なお、式 (1) では進行度の範囲は 0 から 1 であり、ゲームが進行するにつれて値が大きくなるとした。このように進行度を利用することで、終盤に近付くにつれて終盤用の評価関数の割合が高くなるなど、ゲームの進行に対応した局面の評価が可能となる。

本研究で利用した GPS 将棋の評価関数は式 (1) の形を取っている。進行度には、自玉の近傍にある利きの数や持駒を用いているため、玉の危険度としてみなすことができる。

3.1 単純な進行度の問題点

全体で一つの進行度を利用するだけでは局面によっては問題が生じてしまうと言われている[5)]。例えば、駒得していても終盤では駒得の評価が下がるため、進行度を上げないようにと攻め駒をそっぽに向けたり持駒を捨てたりすることや、逆に駒損していると終盤ではその駒損が重視されなくなるため、進行度を上げようと無理に攻めたり自陣に攻め込ませたりするなど、問題がある。

3.2 プレイヤー別進行度

上記の問題は両プレイヤーで共通の進行度を使うことが原因と考えられるために、ゲーム全体の進行度に加えてプレイヤー毎に進行度を持たせる手法がある[5)]。また、IS 将棋でもプレイヤー毎に進行度を持ち、その大小と駒の位置とを組合せ、評価値を決める方法を取っている[☆]。例えば、自分が相手より危険なら、相手玉から遠い駒は攻めに間に合わないとして評価値を 0 とするような方法である。しかし、この手法は点数を細かく調整することができるという利点があるものの、調整すべきパラメータが多いため、具体的に適切な値を決定することが困難となってしまう。

☆ 棚瀬 (CSA 特別例会, GPW 2003)

3.3 GPS 将棋での進行度

本稿では、調整すべきパラメータの少ない枠組としてプレイヤー間の進行度差を評価する枠組を考案し、勝率と評価値の関係に基づいて調整を行う。

GPS 将棋[14] での全体進行度は、両プレイヤーの自玉の 5×3 近傍における敵からの利きの数と持駒の点数の和で計算される。次に、プレイヤー別進行度は、自玉の 5×3 近傍における敵からの利きの数と持駒の点数から計算される「攻めの進行度」と自玉の 5×3 近傍における味方からの利きの数から計算される「守りの進行度」によって構成される。いずれも 0 から 1 の値を取り、1 に近いほど、終盤に近い、自玉が危険、自玉が安全ということを意味する。

なお、進行度の実装における点数は、利きの数は 1/8 で、持駒はそれぞれ、歩 1/64, 香 1/16, 金銀桂 1/8, 飛角 3/16 となっている。合計が 1 以上の時は全て 1 としている。

4. 勝率と評価値の歪みに基づく評価関数の調整

本章では、勝率と評価値の歪みに基づく評価関数の調整について詳しく説明する。まず、勝率と評価値のデータを取得について述べ、次に、勝率と評価値の歪みの説明を行い、プレイヤ別の進行度差における歪みを調べ、最後に自動調整方法について述べる。

4.1 勝率と評価値

まず、実験で用いる勝率と評価値をいかにして得るかを説明する。評価値は棋譜中の各局面における評価値を利用し、勝敗は 10,000 ノードまで詰め将棋探索を行い、詰みを見つけた場合は詰ませる側の勝ち、詰みの無い場合は投了した側の負けとして勝敗を決めた。評価値に対する勝率は、評価値を 500 点毎に区切り、各区間毎に勝敗を数え上げて勝率を得た。

データの対象は将棋倶楽部 24[9] の棋譜 90,000 局で、区切った評価値区間に 1,000 局面以上あったものを対象とした。

本実験では GPS 将棋☆を使用しており、調整前の評価関数は式 (1) である。GPS 将棋では、序盤評価関数はほぼ駒得だけで、駒の関係[13] を利用するが評価値としては歩一枚程度となっている。終盤評価関数については駒と王の位置によって点数を増減するテーブルを利用するなどしている。

4.2 勝率と評価値の歪み

理想の評価関数は、局面の勝ち易さを適切に評価す

☆ OSL[14], revision 2602 を利用

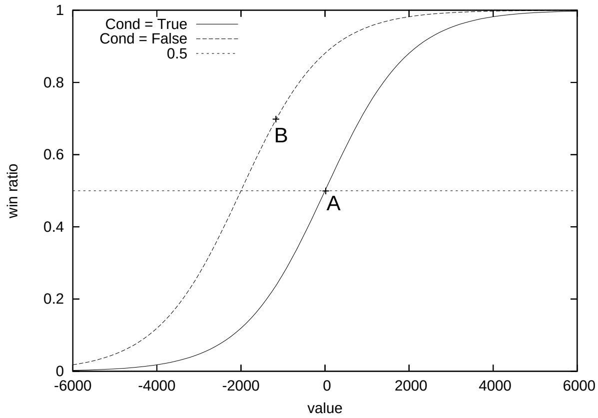

図 1 評価値と勝率の関係が不安定な例

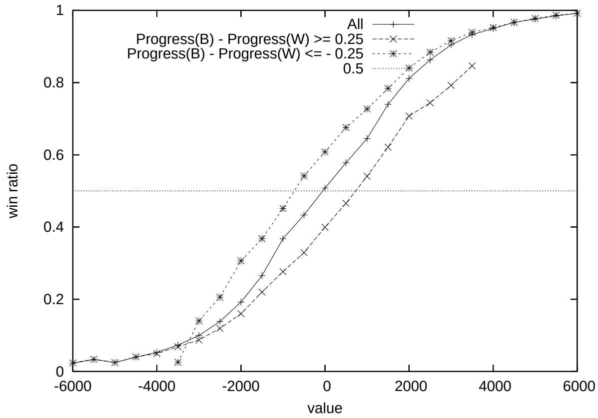

図 2 将棋倶楽部 24 の棋譜による評価値 - 勝率グラフ (調整前)

るものと考えられる。問題のある評価関数は局面の勝ち易さを適切に評価できないため、勝率と評価値との関係に歪みを生じていると考えた。

本稿で着目する勝率と評価値の歪みについて説明する。ゲームにおける探索の目的は勝率の高い局面を選択することである。仮に、x 軸に先手の評価値、y 軸に先手の勝率を条件 Cond 別にプロットした図 1 のような評価関数があるとすると、勝率の低い局面を選ぶことが起こりうる。例えば、探索途中に図 1 の点 A, B のどちらかを選択することを考えると、B の勝率が A よりも高いにもかかわらず、A の評価値が B よりも高いために勝率が低い A を選ぶということが起きてしまう。

このような評価関数を勝率と評価値の間に歪みがあると本稿では呼ぶ。逆に、理想的な評価関数では、状況や条件を変えても一本のグラフとなる。

4.3 プレイヤ別の進行度差

前章からプレイヤ別の進行度に差が生じた局面で問

題があると考えられるので、「先手と後手のプレイヤ別進行度に差が0.25以上ある」局面について、評価値500点毎にその勝率をプロットしたグラフが図2である。図中のProgress(B)は先手の進行度、Progress(W)は後手の進行度を表す。また、進行度は危険度なので、差が正であると先手が危険ということを意味している。

グラフを見ると分かるようにプレイヤ別の進行度に差が生じた時、評価値と勝率の関係に歪みが生じており、問題があると考えられる。なお、プレイヤ別進行度差の評価の枠組により評価関数は

$$\begin{aligned}\text{評価関数} = (1 - \text{全体進行度}) \times \text{序盤評価値} + \\ \text{全体進行度} \times (\text{終盤評価値} + \\ w_1 \times \text{攻めの進行度差} + \\ w_2 \times \text{守りの進行度差}) \qquad (2)\end{aligned}$$

となる。この式(2)中の重みを0とすれば、式(1)と一致する。プレイヤ別進行度差は主に玉周辺の利きの数からなり、終盤評価値に近いので同じように全体進行度に比例させた。

4.4 自動調整

評価値と新しい評価項目から勝率を予想するモデルを作り、棋譜から学習を行う。

手法は最小二乗法(Least Squares)によるものと最尤法(Maximum Likelihood Method)によるロジスティック回帰を用いる。変数yは局面の勝敗を表し、先手の勝ちなら1, 負けなら0と定義した。変数x_iは局面の評価項目を表し、全体をベクトルXとして次のようにまとめた。

$$X = (x_0, x_1, ..., x_n)$$

また、評価項目のサイズをn, データサイズをNとする。

4.4.1 最小二乗法(LS)

もっとも単純な手法として、勝敗を特徴の線形モデルで表し、最小二乗法によりパラメータの調整を行う。

$$f(X) = \sum_{i=1}^{n} w_i x_i$$

目的関数(Objective Function)は、勝敗の予想関数fと実際の勝敗yとの二乗が最小となるように

$$OF_{LS} = \min \sum_{j=1}^{N} (y_j - f(X_j))^2 \qquad (3)$$

として表される。(3)式で表される目的関数が最小化されたときのパラメータが調整結果である。

4.4.2 最尤法を用いたロジスティック回帰(MLM)

次に、最尤法を用いたロジスティック回帰について説明を行う。まず、勝敗の予想をロジスティック式で表す。

$$g(X) = \frac{1}{(1 + \exp(-\sum_{i=1}^{n} w_i X_i))}$$

この式の定義域は$[-inf, inf]$で値域は$[0, 1]$となる。次に、上式と実際の勝敗との尤度を求めると、

$$\mathit{likelihood}(X, y) = g(X)^y (1 - g(X))^{(1-y)}$$

のようになる。よって、全局面での尤度をかけ合わせたものを最大化すれば、最も尤もらしくなるので、目的関数は

$$OF_{MLM} = \max \prod_{j=1}^{N} g(X_j)^{y_j} (1 - g(X_j))^{(1-y_j)}$$

となる。ここで、両辺の対数をとると、

$$\begin{aligned}\log(OF_{MLM}) = \max \sum_{j=1}^{N} \{y_j \log(g(X_j)) + \\ (1 - y_j) \log(1 - g(X_j))\} \quad (4)\end{aligned}$$

となる。対数を取っても大小関係は変化しないので、(4)式を最大化すると目的関数を最大化することになり、この時のパラメータが調整結果として得られる。

5. 実　　験

5.1 調整結果

進行度差の評価は、最小二乗法、最尤法によるロジスティック回帰を行い、手調整した結果との比較を行った。なお、これらのデータ処理はオープンソースの統計解析システムR[1]を利用して行った。また、訓練例は4.2節で勝率と評価値のグラフを描いたときに利用したデータ、将棋倶楽部24[9]の棋譜集90,000局の棋譜から得た9,178,021局面を対象とした。条件別のデータなどは表1にまとめた。

表 1　訓練例のデータ

データ	局面数	勝ち数	勝率	平均評価値
全体	9,178,020	4,658,800	50.76	7.25
進行度差 $\geq$ 0.25	747,041	211,193	28.27	-1,060.68
進行度差 $\leq$ -0.25	778,101	562,826	72.33	1,080.21

表 2　進行度差の調整結果

調整	攻めの進行度差	守りの進行度差
最小二乗法	- 141	101
最尤法	- 153	97
手調整	- 125	50

調整の結果を表2にまとめた。手調整によるものと比較すると守りの進行度差の評価が高い傾向がある

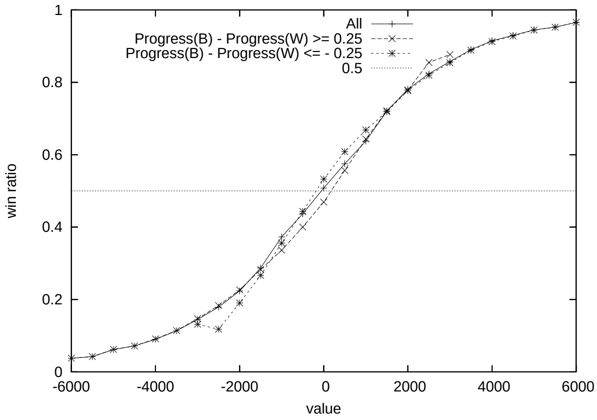

図 3　将棋倶楽部 24 の棋譜による評価値 - 勝率グラフ (最小二乗法による調整後)

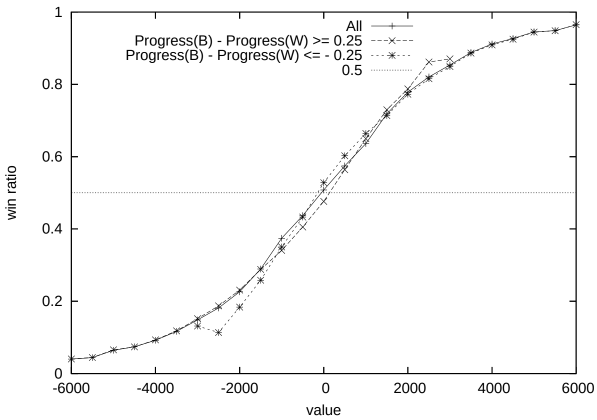

図 4　将棋倶楽部 24 の棋譜による評価値 - 勝率グラフ (最尤法による調整後)

が、オーダーが大きくはずれることもなくいずれも同じような結果となったと言える。なお、守りの進行度差の評価が高い理由として、詰め将棋を 10,000 ノード呼んでいることが考えられる。人間なら見逃すような詰みを発見して勝ちだと判断するため、守りを重視している可能性がある。

調整後の評価値と勝率のグラフが図 3, 4, 5 である。いずれも調整前の図 2 と比べて一本の線に近づいており、評価値と勝率の関係が改善されていることが見て取れる。

5.2　自 己 対 戦

進行度差の評価によるプログラムの棋力向上を確認するため、調整前のプログラムと調整後のプログラムとで 40 局の自己対戦を行った。開始局面は棋譜の 30 手目の局面を利用し、持時間は 25 分とした。結果は表 3 の通りである。表中 none は進行度差を評価しないプログラムであり、LS は最小二乗法によって調整したもの、MLM は最尤法で調整したもの、hand は手調整を行ったものである。

調整後のプログラムは調整前のプログラムに対して有意に勝ち越しており、図によって確認したように、調整による棋力向上が確認できた。調整したプログラム同士の対戦結果を見ると、有意ではないが強さの順に手調整, MLM, LS となっているように思われる。また、有意な差はないので、人間の手による調整と同程度の調整を行うことができたと言える。

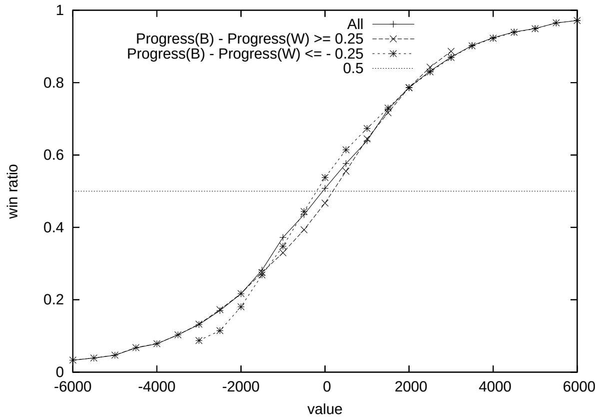

図 5　将棋倶楽部 24 の棋譜による評価値 - 勝率グラフ (手調整後)

表 3　自己対戦

	none	LS	MLM	hand
none	-	11 - 29	13 - 27	11 - 29
LS	29 - 11	-	19 - 20 - 1	16 - 24
MLM	27 - 13	20 - 19 - 1	-	19 - 21
hand	29 - 11	24 - 16	21 - 19	-

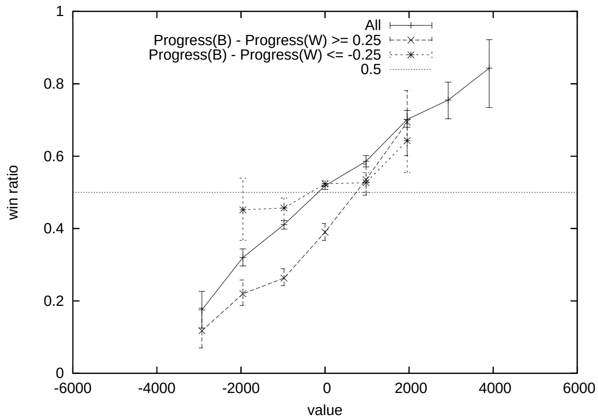

図 6　59 期順位戦の棋譜による評価値 - 勝率グラフ (調整前)

5.3　プロの棋譜の利用

将棋倶楽部 24 の棋譜はアマチュアによるものだが、

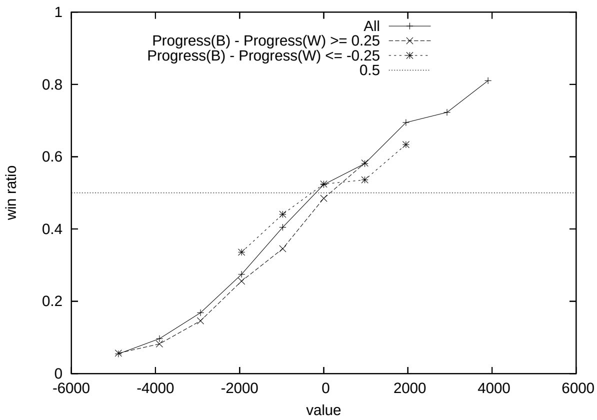

図 7　59 期順位戦の棋譜による評価値 - 勝率グラフ (最小二乗法による調整結果を利用)

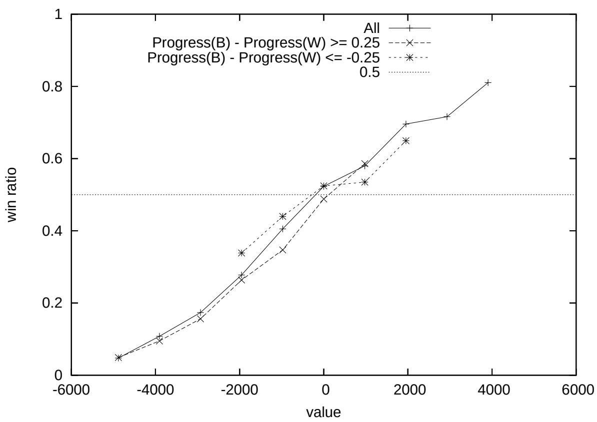

図 8　59 期順位戦の棋譜による評価値 - 勝率グラフ (最尤法による調整結果を利用)

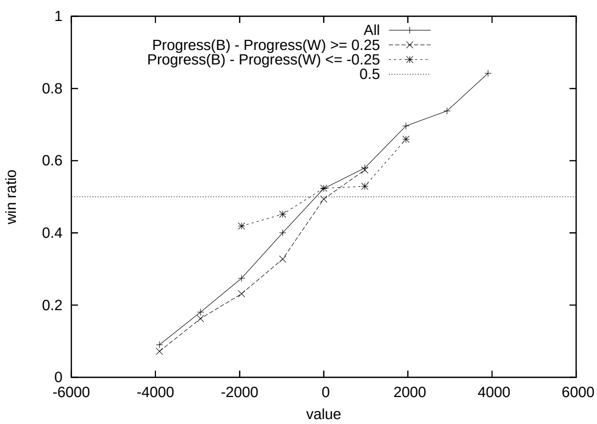

図 9　59 期順位戦の棋譜による評価値 - 勝率グラフ (手調整のパラメータを利用)

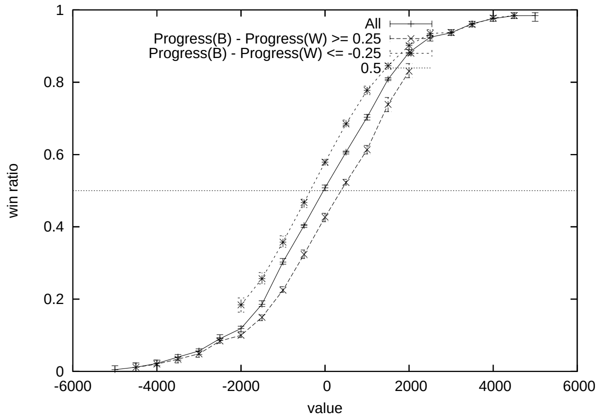

図 10　将棋倶楽部 24 の棋譜による評価値 (静止探索) - 勝率グラフ (調整前)

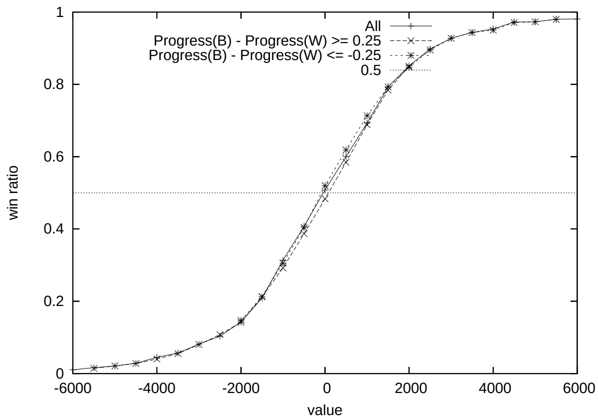

図 11　将棋倶楽部 24 の棋譜による評価値 (静止探索) - 勝率グラフ (手調整の結果を利用)

進行度差の評価がプロの棋譜でも有効であることを確認するため、プロ棋士の棋譜 (59 期順位戦: 全 603 局) を対象に 1,000 点毎に 100 局面以上ある評価値について勝率をプロットした。調整前が図 6、表 2 の調整結果を利用したものが図 7, 8, 9 となっている。図中のエラーバーは、勝敗の具合が勝率 p の二項分布で表されると仮定した時の 95% 信頼区間である。なお、調整後の図では各グラフが接近しエラーバーが重なるため、見やすい調整前の図にだけエラーバーをつけた。

サンプル数は少ないが、プロの棋譜においてもグラフが一本化しており、問題点の改善が確認できる。

5.4 静止探索の利用

多くの将棋プログラムでは評価の際に、評価関数の値をそのまま得るのではなく、より正確な評価を得るために静止探索を行っている。本手法においても静止探索の評価値を利用して調整することで、より良い結

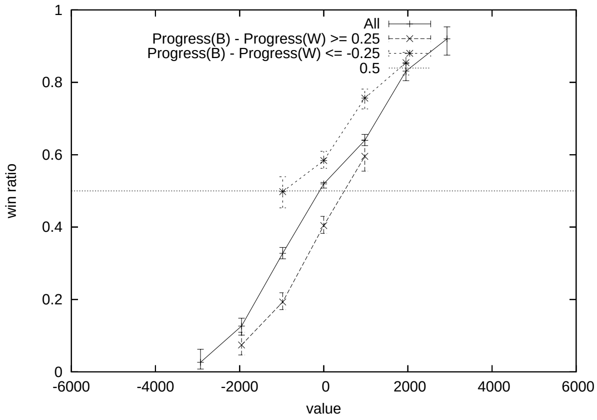

図 12　59 期順位戦の棋譜による評価値 (静止探索) - 勝率グラフ (調整前)

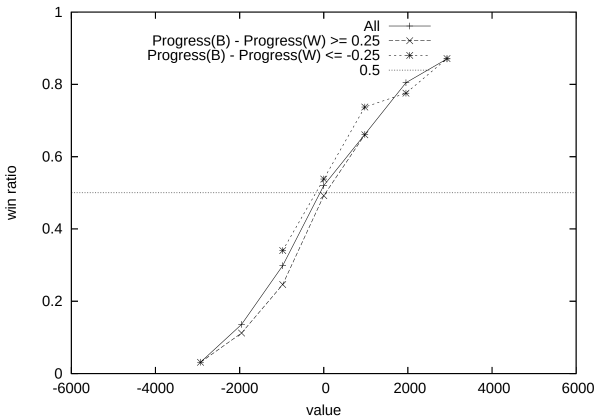

図 13　59 期順位戦の棋譜による評価値 (静止探索) - 勝率グラフ (手調整による結果を利用)

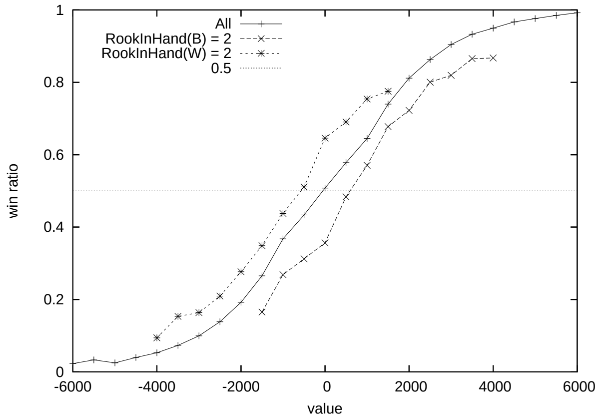

図 14　将棋倶楽部 24 の棋譜による評価値 - 勝率グラフ (持駒に飛車が 2 枚)

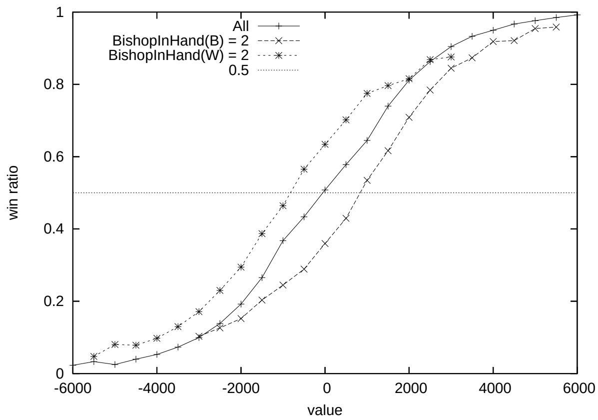

図 15　将棋倶楽部 24 の棋譜による評価値 - 勝率グラフ (持駒に角行が 2 枚)

果が得られる可能性がある。

実際に、プレイヤ別進行度差について実験を行った。将棋倶楽部 24 の棋譜 30,000 局を対象として評価値と勝率をプロットし、調整前の図 10、表 2 の手による調整結果を利用した図 11 を得た。次に、59 期順位戦の棋譜 603 局を対象とし、調整前の図 12、手による調整後の図 12 を得た。なお、GPS 将棋の静止探索は KFEnd[6) を参考に脅威を考慮した探索で、探索深さは 8 である。

どちらの結果も調整結果を利用したグラフの方が一本に近付いており、問題点の改善が確認できる。また、静止探索を行わない場合のグラフ (図 6) と比較するとエラーバーの範囲が狭くなっているなど評価値が正確になっていると考えられる。

しかし、計算時間を考えると、静止探索を行わない場合、詰将棋探索なしで 90,000 局を扱うのにおよそ数時間程度である一方、静止探索を行った場合、詰将棋探索なしで 30,000 局の棋譜を扱うのにおよそ 2 日かかる。さらに、評価関数に変更がある度に探索木に変化が起こるので再探索が必要となり、計算コストの点で静止探索を行うのはあまり実用的ではない。

調整を行うとすると、PV をたどり、たどった先の局面と評価値を利用することも考えられる。

5.5　他の問題点の発見

本手法がプレイヤ別の進行度差以外でも有効であることを示すため、他の条件に関しても実験を行った。同じ種類の駒を複数枚持駒として持っている局面に対して評価値 - 勝率グラフを描いた。これは、同じ駒を複数枚持っていても使いきれないので二枚目以降は点数を減らすというアイデアに基づいている。飛車を 2 枚持っている時の結果は図 14, 角行を 2 枚持っている時の結果は図 15, 桂馬を 3 枚以上持っている時は

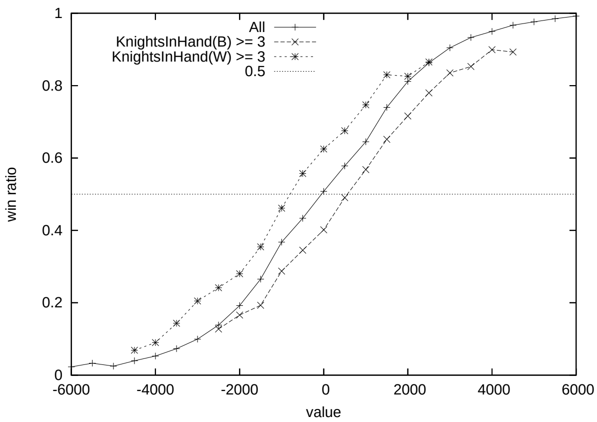

図 16 評価値 - 勝率グラフ (持駒に桂馬が 3 枚以上)

図 16 のようになり、進行度差の時と同様に評価値と勝率の間に歪みが生じていることがわかる。よって、同じ種類の駒を複数枚持駒としている条件で評価値を調整すると良い可能性があると考えられる。図中の RookInHand(B) は B = 先手 (Black) の持駒にある飛車 (Rook) の枚数を表す。W は後手 (White)、Bishop は角行, Knight は桂馬を表す。

6. おわりに

従来、評価関数の問題点を発見するにはゲームの知識が必要であり、困難であった。本稿では、評価関数に問題があるということは勝ち易さを適切に評価できず、勝率と評価関数との関係に歪みが生じていると考え、勝率と評価値のグラフを描くことにより評価値の問題点を発見する手法を提案した。実際に将棋を対象に、プレイヤ間に進行度差がある局面に対して本手法を用い、評価関数に問題があることを発見した。

その後、発見された問題を解決するため、進行度差を評価するという枠組で評価関数の調整を行い、棋力向上を確認するために自己対戦を行った。調整後のプログラムと調整前のプログラムとの自己対戦の結果、調整後のプログラムが有意に勝ち越し、本手法の有効性を示すことができた。

提案した手法は評価関数の問題点を発見する手法としても有効であることを示した。本手法は勝率と評価値に基づいているため、将棋に限らず多くのゲームへの応用が期待できる。

今後の課題としては、条件発見の自動化が挙げられる。今回実験の対象とした、プレイヤ別の進行度に大きな差がある、という条件は人間の知識から得られた。条件の発見を自動化できれば、評価関数の調整がますます容易になり、さらなるゲームプログラムの実力向上が期待できる。

参考文献

1) The R project for statistical computing. Web. http://www.r-project.org/.
2) D. Beal and M. Smith. First Results from using Temporal Difference Learning in Shogi. In H.J.van den Herik and H.Iida, editors, *Computers and Games: Proceedings CG'98. LNCS 1558*, pp. 113–125. Springer Verlag, Berlin, 1999.
3) M. Buro. Improving heuristic mini-max search by supervised learning. *Artificial Intelligence*, Vol. 134, No. 1–2, pp. 85–99, January 2002.
4) Gerald Tesauro. Temporal dfference learning and TD-Gammon. *Communications of the ACM*, Vol.38, No. 3, pp. 58–68, March 1995.
5) あきすての. きのあ将棋. Web, 2004. (http://www12.ocn.ne.jp/%7ekinoa/).
6) 有岡雅章. 将棋プログラム KFEnd における探索. 松原仁（編）, コンピュータ将棋の進歩 4, pp. 18–40. 共立出版, 2003.
7) 柿木義一. 将棋プログラム K3.0 の思考アルゴリズム. 松原仁（編）, コンピュータ将棋の進歩, 第 1 章, pp. 1–22. 共立出版, 1996.
8) 鶴岡慶雅. 将棋プログラム「激指」. 松原仁（編）, アマ 4 段を超えるコンピュータ将棋の進歩 4, 第 1 章, pp. 1–17. 共立出版, 2003.
9) 久米宏. 将棋倶楽部 24 万局集. ナイタイ出版, 2002.
10) 山下宏. Yss-「コンピュータ将棋の進歩 2」以降の改良. 松原仁（編）, コンピュータ将棋の進歩 5, 第 1 章, pp. 1–32. 共立出版, 2005.
11) 橋本剛. 将棋プログラム tacos のアルゴリズム. 松原仁（編）, コンピュータ将棋の進歩 5, 第 2 章, pp. 33–67. 共立出版, 2005.
12) 金沢伸一郎. 金沢将棋のアルゴリズム. 松原仁（編）, コンピュータ将棋の進歩 3, 第 2 章, pp. 15–26. 共立出版, 2000.
13) 金子知適, 田中哲朗, 山口和紀, 川合慧. 駒の関係を利用した将棋の評価関数. 第 8 回ゲームプログラミング ワークショップ, pp. 14–21, November 2003.
14) 田中哲朗, 副田俊介, 金子知適. 高速将棋ライブラリ OpenShogiLib の作成. 第 8 回ゲームプログラミング ワークショップ, November 2003.