駒の関係を利用した将棋の評価関数の学習

金子知適[†] 田中哲朗[††]
山口和紀[†] 川合慧[†††]

将棋の評価関数として，駒（所有者，種類，位置を含む）の 2 項関係を評価するモデルと，棋譜を使った自動的な調整方法を提案する．将棋の評価関数では駒の損得を基本としたうえで，駒の働きや玉の危険度など様々な評価項目が用いられている．しかしそのような駒の損得以外の評価項目は設計が難しく，また自動的な値の調整も困難であった．本稿では，駒の関係という単純な評価項目を提案し，既存の評価項目の多くが駒の関係により表現可能であることを示す．さらに，棋譜で指された指手と指されなかった手の差分に注目した判別分析を行うことで，重みを自動的に調整することを提案する．実際に評価関数を作成したところ，形を評価する実験で良い結果が得られた．さらに，対戦でも提案した評価関数を使ったプログラムが有為に勝ち越し，提案手法の有効性が示された．

Learning of Evaluation Functions Based on Pairs of Pieces

Tomoyuki Kaneko,[†] Tetsuro Tanaka,[††] Kazunori Yamaguchi[†] and Satoru Kawai[†††]

New evaluation function in shogi, based on pairwise piece relations, and a method for automated tuning of their weights are presented. Existing evaluation functions treat a lot of specific aspects, as well as material balance, of the game. Most of such aspects can, we show, be handled by the proposed method. Then, we automatically adjust the weights of pairwise piece relations, by means of discriminant analysis of many game records. Our experiments showed that reasonably accurate evaluation functions for preferable configuration of pieces can successfully be constructed. Moreover, significant improvement on strength was confirmed in self-play.

1. はじめに

強いゲームプログラムを作るためには良い評価関数が必要であり，そのためには評価項目の設計と各項目の重みづけが必要である．評価項目とは局面の中で何に注目するかであり，将棋の場合，駒の損得や玉の危険度などが有効な項目として知られている[8]．

評価関数は複数の評価項目を考慮して総合的に有利・不利を判定する．一般的に用いられる線形結合の場合は，各評価項目の値に重みをかけた総和を評価値とする．各評価項目の重みは事前に決めておく必要がある．

複雑な評価関数では，最適な各評価項目の重みをプログラマが探すことは難しい．そのため，できるだけ評価項目を単純にし，重みの自動調整を行うことが望ましい．しかし，将棋の実用的な評価項目について，自動的な調整に成功した例は今までなかった．

本稿では，「駒の関係」という単純な評価項目を提案し，既存の評価項目の多くが駒の関係により表現可能であることを示す．さらに，棋譜に基づく自動的な重みの調整を提案する．実際に評価関数を作成したところ，従来は困難であった好形悪形の判定[6] で，十分な結果を得ることができた．またコンピュータ将棋選手権で上位のプログラムを用いた対戦でも，作成した評価関数を用いると有意に強くなることが確認された．

2. 関連研究

現在までの将棋プログラムの評価関数として最も基本的な評価項目は駒の損得である．駒の重みは，“激指” や YSS のものは公開されている[8),11]．TD 法[4] を用いて自動的な調整を試みた研究もある[2]．

将棋ではチェスなどと比べて駒の損得以外の評価項

† 東京大学大学院総合文化研究科
Graduate School of Arts and Sciences, The University of Tokyo
†† 東京大学情報基盤センター
Information Technology Center, The University of Tokyo
††† 放送大学
The University of the Air

目が重要である[15]．具体的には，駒の働きを相手玉に近づくほど高く評価する[9),11)]，玉の危険度として玉の周辺の利きや玉の自由度を使う[7]，落とし穴方式[11)]や一般的な規則で囲いを評価する[9)]などの方法が用いられている．なお，探索量が増えるほど評価関数は単純ですむといわれており[9]，実際に大駒を取られる可能性の評価は現在は重視されていない．ハードウェアの進歩や探索技術の進歩[5)]によりこの傾向は続くと思われるが，駒の損得のみで強いプログラムを作れるとは考えにくく，評価項目の設計は将来にわたって重要な課題である．これらの手法で必要になる各評価項目に与える重みはほとんどの場合，将棋プログラマにより経験的に決められている．著者らが最初に駒の関係に基づく評価関数の自動調整を提案した2003年の時点[12)]では，自動的に値を調整した例はなかった．

将棋以外のゲームではオセロ[3)]やバックギャモン[4)]などで，重みを自動的に調整して作成した評価関数が手で設計した評価関数の性能を上回っている．特にオセロ[3)]では約150万という多数の評価項目を用いて正確な評価関数を実現した．本研究で提案する手法では約500万で，今までの将棋で使われている手法よりも多く，正確な評価関数となることが期待できる．一方，オセロで正確な評価関数を作成できた一因は，大差か接戦かの差を終局時の石の数の差（−64から64）で判定できたことである[3]．しかし，将棋の棋譜では，大差か接戦かの差を表す有効な指標は見つかっていない．そこで，本稿の実験ではその問題を避けるため，目標とする概念を絞り好形の判定のみ学習させた．

最近，保木が棋譜に基づく評価関数の自動調整を提案しており，また実際に強いプログラムBonanzaの作成に成功している[17]．保木の手法の主なアイデアは棋譜で指された指手の評価値を指されなかった指手より高く評価されるように評価関数を調整することで，その考え方は本稿で提案する兄弟モデルと共通する．一方，主な相異点は，保木の手法では訓練例の各局面に対して探索を行い最善応手手順後の局面を学習に用いることである．探索を行うとより正確な評価関数が得られる可能性がある一方で，探索それ自体の計算や数値計算において勾配が使えなくなる☆影響から，計算時間が長くかかるというデメリットがある☆☆．提案手法では共役勾配法を用いるため，3.2節の環境のPCでせいぜい1週間程度で実用的な評価関数の作成に成功している．しかも，評価項目の数は保木の手法では1万程度なのに対し，提案手法では100万を超えるものを扱っている．

3. 駒の関係に基づく評価関数

3.1 評価項目

まず本研究で提案する評価関数の評価項目を説明する．提案する評価関数では，駒の関係に重みを割り当てる．駒の関係とは，盤上の2つ駒の p，q に対する両者の関係 $R_{p,q}$ で，各駒について位置，種類，所有者を考慮して，$R_{p,q} = ((\text{位置}_p, \text{種類}_p, \text{所有者}_p), (\text{位置}_q, \text{種類}_q, \text{所有者}_q))$ とする（同じ駒の場合（$p = q$）も含む）．たとえば，先手7三銀，後手8二飛の2つの駒の関係 R は，$R = ((7\text{三}, \text{銀}, \text{先手}), (8\text{二}, \text{飛}, \text{後手}))$ と表す．局面 s の評価値は，局面上に成り立っている関係に対する重み $value(R_{p,q})$ の総和とする：

$$\text{評価値}\ (s) = \sum_{R_{p,q} \in s} value(R_{p,q}). \qquad (1)$$

このモデルは現在よく使われている基本的な手法を包含している．駒の価値のみの評価関数は提案する方式の特殊な場合で，$p = q$ のときに $value(R_{p,q})$ を駒の価値とし他は0とした場合に相当する．敵玉への迫り方を考慮した評価関数も同様に，$value(R_{p,q})$ が p も q も玉でない場合ならば0である場合に相当する．飛車は敵陣にいる方が価値が高い[7]といった1つの駒と位置のみの評価も，同じ駒の関係（$R_{p,p}$）を調整することで組み込むことが可能である．

一方，大駒などの遠くまで届く利きは，さえぎる駒が関係するため直接には表現できない☆☆☆．したがって，飛車の利きが玉の側に届いているかなどの概念を表すためには，別の評価項目を併用する必要がある．また，囲いについても直接認識するわけではなく，2つの駒に分解して重みをつける．しかし後で述べる実験の範囲では，囲いは十分に認識されており個別に取り扱う必要は認められなかった．

3.2 局面評価の速度と必要なメモリ

評価に要する時間は，現在の局面で成り立つ関係の重みを表から求める時間である．平手戦の場合40枚の駒があり，1局面では $\left(\sum_{i=1}^{40} i\right) = 820$ の関係が存在する．新規に計算する場合はすべての駒の関係を調べるため820回の表引きが必要となる．探索においては差分計算が可能であり，その場合は指手で変化した関係のみを調べるため，表を引く回数は80回（単純な移動の場合），約160回（駒を取る場合）となる．単純な駒の損得の計算よりはコストが大きいが，条件判

☆ 評価関数が変わると最前応手手順が変わりうるため．

☆☆ 2006年のゲームプログラミングワークショップの質疑では，3カ月程度とのこと．

☆☆☆ 短い利きについては位置だけで決まるため表現されている．

表 1 1 局面あたりの評価にかかる時間（cycles）
Table 1 Number of cycles for evaluation of a position.

評価なし	駒の損得	駒の関係
98.2	106.3	1,078.4

断がないため，プログラマがリストアップした形を調べるような複雑な評価関数よりは速いと期待できる．

実際に，著者らが開発に参加している将棋ライブラリ[14]で速度を測定して結果を**表 1** にまとめた．所要時間は終盤の局面から 3 手の全探索で約 300 万局面を評価したときの平均である．「評価なし」は，指手生成と局面変更のみを行った場合で，利きやハッシュコードの管理などは行っていない．「駒の損得」と「駒の関係」はそれぞれの評価値を計算した場合である．測定には AthlonMP 1.7 GHz の PC を用いた．駒の関係の評価を行うと駒の損得の場合と比べて約 10 倍の時間がかかるが，それでもまだ 1 秒間に約 160 万局面を評価できる速度である．一般のプログラムはハッシュコードの管理や指手の並べ替えなど評価関数以外の処理も含めて 1 秒間に 30 万局面程度の探索速度であることを考えると，評価関数単体の速度としては問題ないと考られる．

駒の種類は所有者と成不成をあわせて 28 種類で，駒の位置は盤上と駒台で 82 種類であるため，重みを 1 バイトで表現して単純な表を作ると大きさは $(28{\cdot}82)^2 =$ 約 5 メガバイト強，重みを 2 バイトにしてもその倍ですむように現実的な範囲である．なお，$R_{p,q} = R_{q,p}$ であり，また同じ場所には 1 つの駒しかないという制約があるため，可能な関係の種類は 2,578,852 である．先手と後手の対称や左右の対称を考慮すればさらに減るが，今回の実験では簡単のために区別した．

4. 好形の学習

続いて，前節で提案した評価関数について，各関係の重みを自動的に調整する方法を提案する．

ここでは好形悪形の評価を題材とする．好形を評価する評価関数は，駒の損得のない指手が多くある局面で形の良さを細かく判断するものである．定跡を外れてから駒組みを行うためには形の評価が必要となる．

重みを自動調整するためには，目標とする概念と学習モデルにあわせた訓練例を用意する必要がある．本稿では，判別分析[13]を採用し，「局面 a は局面 b より良い（悪い）」という優越関係を棋譜から作って訓練例として使用した．以下それぞれ詳しく説明する．

4.1 好形の定義と訓練例の作成

まず，人間が多く指した手は好形であると仮定して，訓練例は棋譜から作成した．ただし，駒の損得のある指手は，形が良いから指しているとは考えにくいために対象としなかった．駒得をする指手は形が悪くても指す可能性があり，また駒損の手を指す際は詰みや将来の駒得などが関係するためである．実験を単純にするために，勝敗やプレイヤの強さなどは考慮していない．

続いて，棋譜の指手から判別分析に必要な訓練例を作成する．以下に述べる 2 つが自然なモデルとして考えられるため，本稿では 2 種類の実験を行った．

4.1.1 親子モデル

親子モデルでは，棋譜の指手の前後を比較したときに，「指した後は指す前よりも手番のプレイヤにとって良くなる」という仮定から優越関係を作る．有効な手を指している限りは妥当な仮定と思われるが，指さなかった手はモデルに入らないため学習されない．

具体的には，手を指す前の局面 s と指した後の局面 t を比較して，s が先手番の場合，

$$\text{評価値}\ (t) - \text{評価値}\ (s) > 0$$

という訓練例とし，後手番の場合は

$$\text{評価値}\ (t) - \text{評価値}\ (s) < 0$$

という訓練例とする（本稿では評価値は先手が有利のときに正，後手が有利のときに負とする）．

4.1.2 兄弟モデル

兄弟モデルでは，「棋譜で指された手はその局面の他の候補手よりも良い」という仮定から優越関係を作る．指さなかった手を明示的にモデルに入れることで悪い形も認識できると期待できるが，指さなかった手はすべて悪手という仮定は強すぎる可能性もある．

具体的には，ある局面（s）で，棋譜の指手の後の局面（t_0）と他の候補手の後の局面（$t_1, \ldots, t_n$）があったとする．s が先手番の場合，

$$\text{評価値}\ (t_0) - \text{評価値}\ (t_i) > 0$$

という訓練例を各 i（$0 < i \leq n$）について作る．後手の場合は，親子モデルと同様に不等号を逆にする．

4.2 判 別 分 析

最後に判別分析[16]を用いて評価関数の重みを決める．判別分析とは統計の手法で 2 群の判別に用いられるが，ここでは訓練例として与えられた不等式をなるべく満たすような重みを求める手法として使用する．実際の計算では線形回帰に変形して重みを求めた．すなわち，X を訓練例を表す行列，w を重みのベクトル，y を教師値のベクトルとして，

$$X^t X w = X^t y \tag{2}$$

を解いて w を求める．このとき，教師値 y は，正例（左辺が正の訓練例）の場合 1，負例（左辺が負の訓練例）の場合 -1 とした．行列 X の各行ベクト

ル $x[k]$ は各訓練例の左辺と対応し，2 つの局面で成り立つ関係の差分を表す．訓練例 k で比較している局面 2 つを k_0，k_1 とすると，$x_i[k] = r_i[k_0] - r_i[k_1]$ となる．ただし，各関係 $R_{p,q}$ に番号 i を割り当て，その重み $value(R_{p,q})$ を w_i と表す．$r_i[k]$ は，関係 i と局面 k の対応を表す変数で，その値は関係が存在すれば 1，そうでなければ 0 とする．たとえば，関係 ((7 三, 銀, 先手), (8 二, 飛, 後手)) の番号を α とすると，$r_\alpha[k] = 1$ となるのは，局面 k の 8 二に後手の飛車が，7 三に後手の銀が存在する場合である．評価値は式 (1) の形なので，$x[k]$ は重み w との内積を取ると訓練例 k の左辺となる：

$$\begin{aligned} w \cdot x[k] &= \sum_i w_i x_i[k] \\ &= \sum_i w_i (r_i[k_0] - r_i[k_1]) \\ &= \text{評価値 } (k_0) - \text{評価値 } (k_1). \end{aligned}$$

なお，式 (2) を解く際には，直接解法は使えず，反復解法を用いる必要がある．これは X^tX が，次数が重みの数（> 200 万）の正方行列となりメモリに載らないためである．本稿の実験では共役勾配法[1] を用いた．一般に $Aw = b$ を解いて w を反復解法で求める過程では，ベクトル（p とする）に対する積 $q = Ap$ を計算する必要が生じる．今回の場合 $A = X^tX$ であり，q の各要素を求める計算は以下のように変形できる．

$$\begin{aligned} q_i &= \sum_j (a_{ij} p_j) = \sum_j (x[i] \cdot x[j]\, p_j) \\ &= \sum_j \sum_k (x_i[k] x_j[k])\, p_j \\ &= \sum_k \sum_j (x_i[k] x_j[k]\, p_j) \end{aligned}$$

なお，行列 A の各要素を a_{ij} と表記した．この変形を用いると，行列各訓練例 k を巡回しながら q_i に $\sum_j (x_i[k]\, x_j[k]\, p_j)$ を足すことで $q = Ap$ を計算できる．判別分析にかかる時間は，この積の計算がほとんどである．各訓練例 k において，$x_i[k]$ が 0 の場合はそれが関係する積の計算を省略できるため，計算時間は $x_i[k]$ の 0 でない個数の二乗に比例する．$x_i[k]$ は局面で成り立つ関係の差分であるため，似た局面どうしの場合 0 であるものが多い．兄弟モデルでは，親子モデルのほぼ 2 倍の差分があり，計算時間は 4 倍近くなる．

5. 実験結果

親子モデルと兄弟モデルに基づいて 2 つの評価関数を作り，得られた評価関数の評価を行った．

訓練例としては，将棋倶楽部 24[10] の棋譜集から駒の損得のない局面を抽出して用いた．駒の損得の有無は，取り合い探索を行ったときに評価値が変わらないことで判定した．取り合い探索で用いる駒の価値としては “激指” で用いられている値[8] を参考に**表 2** の値（盤上の駒も持ち駒にも同じ点）を用いた．

表 2 駒の価値
Table 2 Value of pieces.

歩	香	桂	銀	角	飛
100	400	400	550	800	950
馬	龍	金・他の成駒			
1,150	1,300	600			

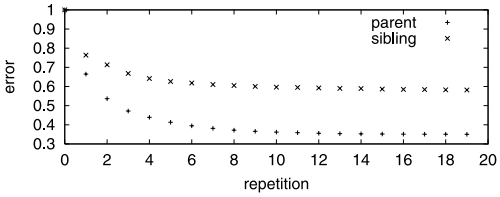


図 1 共役勾配法の収束の状況
Fig. 1 Convergence of CG.

棋譜の数は親子モデルの場合 20 万棋譜，兄弟モデルの場合は 4 万棋譜（棋譜あたりの訓練例が多いため）を用いた．訓練例の数はそれぞれ，約 1,500 万と約 9 億となった．学習の安定性を考慮して，訓練例の中で 20 局面以上に現れた関係のみを用いた．親子モデルの場合，全部で約 258 万種類の関係の中で約 106 万のみが対象となった．本来は別に測定するべきだが，実験の都合で兄弟モデルでも同じ関係に重みをつけた．

図 1 に収束の様子を示す．反復ごとに少数のテスト例での回帰の誤差をプロットしている．親子モデル（parent）の方が誤差が小さく，兄弟モデル（sibling）の方が難しい制約が多いことを示唆している．親子モデルは 36 回反復させ，兄弟モデルは実験時間の都合で 20 回で打ち切った．それぞれのモデルの上位 3 つの重みを調べたところ**表 3** の上半分のように不自然な関係が現れた．いずれも出現回数が少ないため出現回数 200 回以上の条件で調べ直したところ，表の下半分のように利きをつける関係などが現れた．出現数の閾値には調整の余地が残っているといえる．

最後に重みが $[-127, 127]$ の範囲でほぼ山型となるようにそれぞれ 128 倍，160 倍にスケーリングして離散化した．そのヒストグラムを**図 2** に掲載する．見やすさのために 4 つ刻みでプロットした．離散化後に 0 でない重みがついた関係はそれぞれ約 97 万と約 99 万，範囲を超えた重みはどちらも約 500 であった．

5.1 指手による評価値の上昇の確認

テスト用のデータとして，訓練例とは別の 1 万試合

表 3 重みの上位 3 つ（上：無制限，下：出現数 200 以上）
Table 3 Features with best three weigths (upper: unrestricted, lower: more than 200 occurrence).

種類	駒 1	駒 2	出現数	重み
親子	△ 1 二馬	△ 7 九龍	23	2.56
	△ 2 二歩	▲ 2 五成銀	56	2.27
	△ 1 九玉	▲ 7 八香	34	2.19
兄弟	△ 7 九成銀	△ 9 八成香	30	2.66
	△ 6 二馬	△ 6 四馬	24	2.61
	▲ 3 六龍	▲ 6 三成桂	25	2.32
親子	△ 2 六飛	▲ 2 七歩	4,011	1.01
(≥ 200)	▲ 7 九銀	▲ 8 八銀	39,650	0.90
	△ 7 六飛	▲ 7 七歩	1,533	0.84
兄弟	△ 2 五玉	▲ 2 六歩	419	1.05
(≥ 200)	△ 2 四香	▲ 2 五歩	765	1.04
	△ 3 八馬	▲ 3 九金	378	1.03

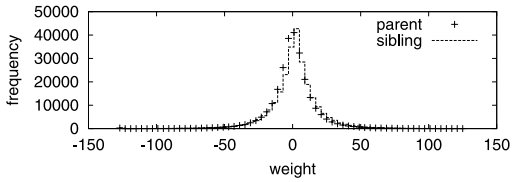


図 2 スケール調整後の重みの分布のヒストグラム
Fig. 2 Histogram of weights after scaling.

表 4 指手の前後での評価値の上昇
Table 4 Ratio that evaluation value of successive position is greater than that of previous position.

手数	対象 (%)	上昇率 (%)	
		親子	兄弟
0–20	98.4	99.9	93.2
20–40	90.7	98.3	87.2
40–60	65.8	96.1	82.4
60–80	49.8	96.9	81.1
80–100	46.3	97.2	80.2
100–	46.2	96.4	79.0
全体	67.5	97.8	85.6

について，棋譜の指手の中で指手の前後で駒の損得が変わらない局面を訓練例と同様に選び，約 76 万局面を用意した．これらに対して指手の前後で評価値が指した手番からみて上昇したかどうかを調べたところ，**表 4** の結果となった．表で対象とは，対象となった指手の割合を示し，その率の低下は中盤から終盤へと進むにつれて駒の損得がある指手が増えるいう将棋の性質を示している．

親子モデルでは，全体で 97.8%とほぼモデルどおりの学習ができたといえる．一方，兄弟モデルでもかなりの確率で指した後の評価値が上昇しているが，親子間の比較はモデルに入っていないため，親子モデルには及ばない．さらに手数ごとに細かく見ると，成績は手数が進むごとに両モデルとも成績が悪くなり，兄弟モデルの方が，より低下が大きい．

表 5 1 手読みによる指手の順位づけ
Table 5 Move ordering by one-ply search.

手数	候補手の数		順位	
	合法手	候補手	親子	兄弟
0–20	35.0	30.5	12.2	1.7
20–40	44.6	27.2	8.3	2.3
40–60	68.9	17.9	5.7	2.0
60–80	109.3	12.9	4.1	1.8
80–100	133.2	10.3	3.3	1.7
100–	161.3	9.1	2.9	1.9
全体	75.7	21.1	7.3	1.9

5.2 1 手読みの評価

同じテストデータに対して，棋譜の指手が他の候補手よりも高く評価されるかどうかを調べた．その結果を**表 5** に掲載する．まず合法手をすべて作成し，その中で駒の損得がない手を候補手とした．指手が進むにつれ合法手の中の候補手の割合は減ってゆくが，全体としては 4 分の 1 程度（21.1/75.7）であった．それらの候補手を，指した後の局面の評価値順に並べ，棋譜の指手の順位を調べた．兄弟モデルでは平均約 2 番目でほぼモデルどおり学習できたといえる．もし駒の損得のみの評価関数を使うと候補手 21 手は同じ値となり，その中からランダムに選ぶとすると平均順位は 10 番目程度になるため，大幅な向上といえる．親子モデルでも棋譜の指手は平均約 7 番目とそれなりに高く評価されているが，親子モデルでは兄弟の比較はモデルに入っていないため兄弟モデルには及ばない．親子モデルでは手数にほぼ関係なく候補手の数の約 3 分の 1 程度の順位であるが，兄弟モデルによる順位は候補手の数の影響が小さい．

5.3 定跡の評価

別の実験として，序盤の定跡の手をどの程度高く評価できるかどうかを矢倉を題材に調べた．局面で定跡の指手が何番目に高く評価されたかと，定跡と一致しなかった場合に最善と評価された手を**表 6** にまとめた．兄弟モデルでは比較的よく定跡に一致し，また一致しなかった場合も無難な手を選択しているので実戦で効果があると期待できる．一方，親子モデルの方は，人間があまり指さないような指手が多く序盤には向かないと思われる．

5.4 悪形の回避

続いて，学習した評価関数が，一般的に悪形といわれている避けるべき形を回避できるかどうかを実験した．悪形の例題として，Web で公開されている「将棋プログラム KFEnd」の一部の「悪形チェック」[6] の中で取り上げられている局面を用いた．文献 6) では 30

表 7 各モデルの評価関数による悪形の例題の評価
Table 7 Evaluation of bad-shape problems with both evaluation functions.

	例題 1	例題 2	例題 3	例題 3'	例題 4
	指手 評価	指手 評価	指手 評価	指手 評価	指手 評価
親子モデル					
悪手	▲3六飛 9 ▲4六飛 61 ▲2七飛 21 ▲2五歩 30	▲7七桂 5 ▲7七金 −8	▲1八飛 99 (▲2八銀) 52	▲1八飛 88 (▲2八銀) 115	▲3九王−41 ▲2六歩 −3
普通の手	▲1六歩 55 ▲4六歩 41	▲7七銀−22	▲2六歩 7	▲2六歩 121	▲5六歩 19 ▲4六歩 26
兄弟モデル					
悪手	▲3六飛−15 ▲4六飛−13 ▲2七飛−17 ▲2五歩−41	▲7七桂−25 ▲7七金−42	▲1八飛 3 (▲2八銀)−21	▲1八飛 −38 (▲2八銀)−38	▲3九王−25 ▲2六歩−40
普通の手	▲1六歩 38 ▲4六歩 −5	▲7七銀 9	▲2六歩 −9	▲2六歩 59	▲5六歩 2 ▲4六歩 9

表 6 矢倉定跡の評価
Table 6 Evaluation of positions appear in Yagura opening.

定跡	親子		兄弟	
	順位	最善手	順位	最善手
▲7六歩	24	4八王	0	
△8四歩	8	6二王	1	3四歩
▲6八銀	5	4八王	2	2六歩
△3四歩	8	6二王	0	
▲7七銀	2	7七角	1	6六歩
△6二銀	9	6二王	0	
▲2六歩	11	3八飛	1	6六歩
△4二銀	4	5二王	5	8五歩
▲4八銀	7	4八王	1	2五歩
△3二金	12	5二王	1	5四歩
▲7八金	15	3九金	1	5六歩
△5四歩	16	7一金	0	
▲5六歩	12	3九金	3	2五歩
△4一王	6	7一金	0	
▲6九王	6	3九金	0	
△5二金	16	7一金	0	
▲5八金	13	3九金	0	
△7四歩	3	3一王	0	
▲3六歩	3	9八香	1	2五歩
△3三銀	11	1二香	2	6四歩
▲7九角	15	9八香	0	
△3一角	9	3一王	1	6四歩
▲6六歩	8	4六角	7	3五歩
△4四歩	12	7三銀	2	6四角

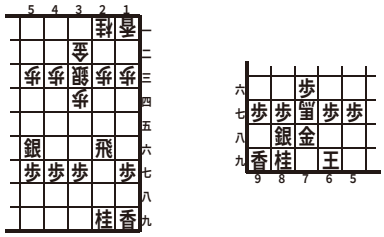


図 3 悪形の例題：1（左），2（右）
Fig. 3 Bad-shape problems: 1 (left), 2 (right).

種類の項目があげられているが，評価関数で評価するために，具体的な局面があるものを取り上げた．

表 7 に，学習した評価関数による評価結果を載せる．数値は指手を指す前後の評価値の増減であり，大きい方が先手有利という評価である．

例題 1（**図 3** 左）で，▲3六飛あるいは▲4六飛は「自歩先の飛」であり攻める態勢を作り難くしてしまう悪形，また▲2五歩は「自らの飛筋を止める歩を打つ」悪形とされている．一方，▲1六歩，▲4六歩は比較のために著者が加えた候補手である．親子モデルでは，悪形の▲4六飛に普通の手以上の評価がされている．一方，兄弟モデルのほうは，最高点は普通の手の▲1六歩であり妥当な評価と思われる．

例題 2（図 3 右）は，△7七角成と角を交換しにこられた図で，▲7七同金や同桂は壁形の悪形であり，▲7七同銀が普通と解説されている．この問題については，兄弟モデルは▲7七同銀に良い評価をし，また悪形になる手をはっきり悪く評価できたが，親子モデルは逆の結果となった．原因を調べるために，評価値に影響する関係の重みを抜き出して**表 8** に掲載した．親子モデルと兄弟モデルを比較するとほぼ似たような重みがついているが，▲7七金に対する重みが親子モデルでは高すぎることが問題のようである．

例題 3（**図 4** 左）では，▲1八飛は自陣の端に移動する悪形で，さらに▲2八銀とするとさらに悪形であ

表 8 例題 2 の指手に関係する重みの抜粋
Table 8 Weights related to bad-shape problem (2).

駒 1	駒 2	親子	兄弟
▲ 8 八銀	▲ 7 八金	8	7
▲ 7 七銀	▲ 7 八金	3	7
▲ 7 七金	▲ 8 八銀	9	−7
▲ 7 七金	▲ 8 九桂	23	2
▲ 7 八金	▲ 8 八銀	8	7

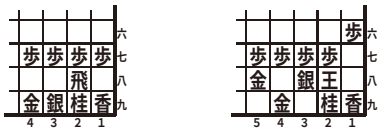


図 4 悪形の例題：3（左），4（右）
Fig. 4 Bad-shape problems: 3 (left), 4 (right).

る．▲ 2 八銀は▲ 1 八飛の後で指す手のため，表では括弧をつけて区別した．この例題では悪形を作る指手が，著者が考えた普通の手である▲ 2 六歩と比べて良い点になってしまっている．そこで，初期局面の盤面全体を使って同じ候補手を試したところ（例題 3'），▲ 2 六歩が最も良い手と評価された．したがって例題 3 に正解するためには，元の局面の外にある駒との関係が重要であると解釈できる．

例題 4（図 4 右）で，▲ 3 九玉は囲いを崩す悪形と解説されている．また▲ 2 六歩は，原文にはこの図との対応はないが，「自玉の頭の歩突き」に相当する悪形と判断し，著者が加えた．▲ 4 六歩▲ 5 六歩は普通の手の例として比較のために著者が加えた．どちらのモデルもまともな判断をしている．

総合して，どちらのモデルも悪手を避けるためには有効であり，特に兄弟モデルではすべての例題で悪形を避けることができた．兄弟モデルの評価関数は，実験全体を通しても良い成績であったため，兄弟モデルに基づいた学習は効果的であったと結論できる．

5.5 自己対戦

最後に，評価関数の有用さを評価するために，兄弟モデルで作成した評価関数を組み込んだプログラムと組み込まなかったプログラムで対戦を行った．プログラムは第 15 回世界コンピュータ将棋選手権で決勝リーグに進んだ GPS 将棋を用いた．GPS 将棋の評価関数は，序盤用と終盤用の 2 つの評価関数を進行度により内分をとるオーソドックスな作りに，駒の独占や先手と後手の危険度の開き具合などの調整を加えたものである．序盤用の評価関数は駒得と駒の関係を合計したもので（比率は表 2 と図 2 に掲載した値と同じである），終盤用には玉に接近した駒を高く評価する表を用いている．また，探索は実現確率探索[5] を基本に末端では KFEnd 風の静止探索を行っている．

表 9 自己対戦の結果（駒の関係あり 対 駒の関係なし）
Table 9 Result of self-play (program enhanced by piece pair evaluation function vs. original one).

開始局面の手数	勝	負	引分
10	33	7	0
30	32	8	0

対局は，第 15 回世界コンピュータ将棋選手権と同様に 25 分切れ負けのルールで行った．59 期順位戦の 20 棋譜の 10 手または 30 手進んだ局面を用意し，そこから先手後手を入れ替えて 2 局ずつ行い，合計 40 局行った．結果を**表 9** に，駒の関係の評価関数を用いたプログラムからみた勝敗で示す．どちらの場合でも有為に勝ち越しており，また，評価関数は 5.3 節の実験であるように駒の関係を用いた定跡の代わりに陣形を組むことに有効に働くと思われるが，30 手まで進めてから対局を行っても 10 手までででもほぼ同様の結果であることから，戦いにおいても有効であることが示された．

6. おわりに

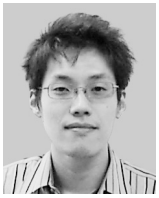
本稿では駒の関係を利用した局面の評価方法を提案し，既存の評価項目の多くが駒の関係により表現可能であることを示した．また，好形の判定を題材に，そのような評価関数の重みを自動的に調整する方法として，棋譜の判別分析を提案した．実験の結果，提案手法を用いて重みを自動調整した評価関数が，定跡や人間の指手をかなりの程度模倣し，また悪形を避けることに有効であることが確認できた．棋譜から訓練例を作る手法として，指手の前後の局面の比較をもとに学習する親子モデルと指手と他の候補手を比較して学習する兄弟モデルの 2 つの手法を比較したところ，後者の方が良い成績であった．

今後，強いプログラムを作るには，詰む確率や玉の危険度を評価し，さらに，それらがどの程度の駒損と釣り合うかを調整することが重要である．適切な教師例を用意できれば，駒の関係でそれらを表せると考えている．また評価関数以外にも，駒の関係という着眼点は探索の深さの制御（実現確率[5]）などでも有効と思われる．将来，精度を上げるためには，駒のペアの代わりにトリプレットに基づく評価を行い，「龍の利きを遮る金底の歩」などの概念を表すことも考えられる．しかし表の要素数は $(28 \cdot 82)^3$ となり 12 ギガバイトを超え，また訓練に必要なデータや時間が大幅に増えるため現時点では難しい．

参 考 文 献

1) Barrett, R., Berry, M., Chan, T.F., Demmel, J., Donato, J., Dongarra, J., Eijkhout, V., Pozo, R., Romine, C. and der Vorst, H.V.: *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods, 2nd Edition*, SIAM, Philadelphia, PA (1994).
2) Beal, D.F. and Smith, M.C.: First Results from Using Temporal Difference Learning in Shogi, *CG98*, Tsukuba, Japan, pp.113–125, Springer-Verlag (1998).
3) Buro, M.: Improving heuristic mini-max search by supervised learning, *Artificial Intelligence*, Vol.134, No.1–2, pp.85–99 (2002).
4) Tesauro, G.: Programming backgammon using self-teaching neural nets, *Artificial Intelligence*, Vol.134, No.1–2, pp.181–199 (2002).
5) Tsuruoka, Y., Yokoyama, D. and Chikayama, T.: Game-tree Search Algorithm based on Realization Probability, *ICGA Journal*, Vol.25, No.3, pp.145–153 (2002).
6) 有岡雅章：悪形チェック（Inside KFEnd），Web (2000). http://plaza9.mbn.or.jp/%7ekfend/inside_kfend/bad_shape.html
7) 柿木義一：将棋プログラム K3.0 の思考アルゴリズム，コンピュータ将棋の進歩，松原 仁（編），chapter 1, pp.1–22, 共立出版 (1996).
8) 鶴岡慶雅：将棋，情報処理，特集ゲーム情報学，Vol.44, No.9, pp.900–904 (2003).
9) 鶴岡慶雅：将棋プログラム「激指」，アマ 4 段を超えるコンピュータ将棋の進歩 4，松原 仁（編），chapter 1, pp.1–17, 共立出版 (2003).
10) 久米 宏：将棋倶楽部 24 万局集，ナイタイ出版 (2002).
11) 山下 宏：YSS—そのデータ構造，およびアルゴリズムについて，コンピュータ将棋の進歩 2，松原 仁（編），chapter 6, pp.112–142, 共立出版 (1998).
12) 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 (2003).
13) 奥野忠一，久米 均，芳賀敏郎，吉沢 正：改訂版多変量解析法，日科技連 (1981).
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 (2003).
15) 棚瀬 寧：IS 将棋のアルゴリズム，コンピュータ将棋の進歩 3，松原 仁（編），chapter 1, pp.1–14, 共立出版 (2000).
16) 田中 豊，垂水共之，脇本和昌：パソコン統計解析ハンドブック，共立出版 (1984).
17) 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).

(平成 19 年 1 月 24 日受付)
(平成 19 年 5 月 9 日採録)



金子 知適（正会員）

1997 年東京大学教養学部卒業．2002 年東京大学大学院総合文化研究科博士課程修了．博士（学術）．2002 年東京大学院総合文化研究科助手．思考ゲーム，知識処理に興味を持つ．

田中 哲朗（正会員）

1965 年生まれ．1987 年東京大学工学部計数工学科卒業．1992 年東京大学大学院博士課程修了．博士（工学）．東京大学工学部助手，東京大学教育用計算機センター助教授を経て，現在は東京大学情報基盤センター准教授．日本ソフトウェア科学会，ACM 各会員．平成 15 年度情報処理学会論文賞受賞．

山口 和紀（正会員）

1978 年東京大学理学部数学科卒業．1981 年東京大学理学部助手．1985 年理学博士．1989 年筑波大学電子情報工学系講師．1992 年東京大学教養学部助教授．1999 年東京大学情報基盤センター教授．2007 年東京大学総合文化研究科教授．モデリング全般に興味を持つ．ACM 会員．

川合 慧（正会員）

1967 年東京大学理学部物理学科卒業．1981 年東京大学理学部情報科学科助教授．1984 年東京大学教育用計算機センター助教授．1988 年東京大学教養学部教授．1996 年東京大学総合文化研究科教授．2007 年放送大学教授，理学博士．研究分野：グラフィクス，プログラミング，ユーザインタフェース．電子情報通信学会，ソフトウェア科学会，ACM 各会員．