

探索パラメータの調整に適した目的関数の調査

- モンテカルロ木探索将棋の探索パラメータの調整 -

竹 内 聖 悟^{†1} 金 子 知 適^{†2}

強いプレイヤーの作成のために探索パラメータの調整が重要である。探索パラメータの自動調整には課題が多く、本研究では、学習モデルを検討するために、適切な目的関数の調査を行う。

実験では、将棋を対象にモンテカルロ木探索の探索パラメータを調整し、目的関数の性能について評価を行った。複数の目的関数を使い、遺伝的アルゴリズムによるパラメータ調整を行った。その後、目的関数毎に得られたプレイヤーについて対戦と各目的関数の数値を計算し、対戦結果からレーティングを計算し、各目的関数とレーティングとの回帰分析などから、適切な指標について調査した。

調整結果のプレイヤーの性能や目的関数の値とレーティングの回帰分析の結果、調整時間から、目的関数として、多くのプレイアウトを行った指手との一致数を利用するものが適切であると結論付けた。

A Survey of Objective Functions for Tuning Search Parameters

- Tuning Search Parameters of Monte Carlo Tree Search in Shogi -

SHOGO TAKEUCHI^{†1} and TOMOYUKI KANEKO ^{†2}

Tuning search parameters is important for strong programs, and there are many problems for automatic tuning. In this research, we survey objective functions for tuning search parameters, in order to investigate the model for automatic tuning.

First, we tuned the parameters by Genetic Algorithms with various objective functions for Monte Carlo Tree Search player in Shogi. We performed tournaments and calculate rating for each player, to measure performance of players with tuned parameters. Finally we calculated the value of each objective function and run regression analysis. From the experimental results, we conclude that the accuracy is suitable for tuning search parameters.

1. はじめに

近年のコンピュータ将棋・囲碁の棋力の向上が目覚ましい。強いプログラムの作成には最新のゲーム木探索の手法を取り入れることが重要であるが、各手法は内部に多くのパラメータを持つため、それらを適切に調整する必要がある。機械学習は自動調整のための有力な手段であるが、万能ではなく、調整対象のパラメータに適切な学習モデル（教師や目的関数）を注意深く設定する必要がある。評価関数の例でも、オセロでは探索結果の最小二乗法による学習⁴⁾が、将棋では棋譜の指手の一致率を高める学習²³⁾が効果的であったように、対象とするゲームによって効果的な学習モデルが

異なることすらある。また、目的関数が微分可能である必要であるなどの制約も学習手法によっては存在する。このような背景で、各パラメータに対する適切な学習手法は、重要であるにも関わらず、現時点では分かっていないことが多い。

コンピュータ囲碁の分野においてモンテカルロ木探索が多くのプログラムに用いられ、大きな成果を挙げている¹⁸⁾。モンテカルロ木探索は General Game Playing¹¹⁾ など他のゲームへも応用されており、将棋への応用もいくつか試されている^{14),19),20)}。将棋へと応用を試みる利点としては、モンテカルロ木探索の並列化がアルファベータ木探索に比べ容易であることや、局面によっては、アルファベータ木探索よりも正確な評価を行うことが可能であることなどが挙げられる。一方、モンテカルロ木探索における探索パラメータは、強さのために重要でありながら、確立された調整方法はまだ存在しない。

本稿では、探索パラメータの調整にどのような学習モデルが適するかを明らかにするため、適切な報酬

^{†1} JST ERATO 湊離散構造処理系プロジェクト
JST ERATO Minato Discrete Structure Manipulation System Project
takeuchi@erato.ist.hokudai.ac.jp

^{†2} 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
kaneko@graco.c.u-tokyo.ac.jp

の与え方を調査する。これは最適化の文脈において、目的関数の設計に相当する。具体的には、目的関数を複数用意し、それぞれの目的関数毎に調整を行う。調整結果から目的関数の性能を見る他、目的関数の値とレーティングとの相関について解析を行い、調整にかかる時間などから適切な目的関数の調査を行う。題材として、将棋におけるモンテカルロ木探索プログラムを利用し、パラメータの調整では学習モデルの単純のために遺伝的アルゴリズムを利用する。

2. 関連研究

パラメータの自動調整についての関連研究について述べる。評価関数の重みの自動調整と探索パラメータの自動調整、探索ヒューリスティックについての選択を行った研究について述べ、調整に用いる遺伝的アルゴリズムについて説明を行う。

2.1 評価関数の重みの自動調整

保木は将棋プログラムを対象に、大規模な評価関数の重みの学習に成功している²⁴⁾。目的関数には棋譜の指し手との一致率（不一致率）を利用している。評価値を得る際に浅い探索を行う他、棋譜の指手が他の兄弟手よりも高い評価値を持つかどうかで一致率（不一致率）を計算している。

強化学習を使い評価関数の重みの学習を試みた例には、TD-Leaf²⁾や Bootstrapping¹⁷⁾がある。それぞれチェスプログラムを対象とし、インターネット上で人間と対戦し、一定の強さまで向上した。David-Tabibiらは遺伝的アルゴリズムを使い、チェスプログラムの評価関数の調整に成功した⁹⁾。調整対象は35パラメータで、目的関数として強いプログラムの評価値との差を利用している。

2.2 探索パラメータの自動調整

Multi-ProbCutは統計的に枝刈の閾値を決めており⁵⁾、これも探索パラメータの自動調整と見ることが出来る。同様に、実現確率探索では実現確率を棋譜から統計的に計算しており¹⁶⁾、これも自動調整とみなせる。

Björnssonらはチェスにおいて、探索延長のパラメータの自動調整を行った³⁾。問題集を正答するのにかかったノード数を目的関数とし、プレイヤーに問題集を解かせ、パラメータを実際に微増減させて問題集を解かせて、ノード数の変化を微分値とみなして調整を行った。対戦による調整では、前回の探索と同じ手を指しているのに評価値が悪くなった時に悪手を指したと検出し、その部分の探索延長パラメータを、ノード数に応じて増加させている。対戦による調整が良く、問題集は手調整と同程度であった。

David-Tabibiらは遺伝的アルゴリズムを用いてチェスプログラムの探索パラメータの自動調整を行った¹⁰⁾。適応度関数には Björnsson らと同様のものを使っている。結果として、人手で調整したプレイヤーと同レベルの強さとなり、大会でも優れた成績を残している。

モンテカルロ木探索の方策の学習では Minorization-Maximization 法を用いた調整⁸⁾や Simulation Balancing^{13),15)}が成功している。Simulation Balancing では目的関数として、たくさんのプレイアウトを行ったプレイヤーによる勝率と調整するプレイヤーによる勝率との二乗誤差を使用している。

また、モンテカルロ木探索の木探索部分における探索パラメータの調整を⁷⁾が行い、成功している。対戦を行い、その勝率を目的関数として、クロスエントロピー法を使い調整を行なっている。この際、パラメータそのものではなく、パラメータの分布を計算している点で違いがある。

2.3 探索ヒューリスティックの選択

Anantharaman は、チェスプログラム Deep Thought で使うヒューリスティックの選択を行った¹⁾。対戦には時間がかかりすぎるため、対戦以外の良い指標を調査し、各プレイヤーのレーティングを計測し、各指標との相関や誤差を計測し、指標の調査を行った。深く探索したプレイヤーとの指手の一致数、深く探索したプレイヤーの評価値との差を使うのが良いという結果を得ている。

2.4 遺伝的アルゴリズム

遺伝的アルゴリズムによる調整について説明を行う。生物の進化のメカニズムを真似たアルゴリズムで、最適化問題などに使われる¹²⁾。David-Tabibiらはチェスプログラムの評価関数と探索パラメータの調整を遺伝的アルゴリズムによって行い、成功している^{9),10)}。

ゲームプログラムのパラメータ調整においては、プレイヤーを個体とすると、各プレイヤーの持つパラメータが遺伝子に対応する。ある目的関数を最大化するように調整する場合、その目的関数が適応度関数となる、適応度が高いほど生き残りやすく、次の世代に子孫を残しやすくなる。次の世代の個体は、親のペアの遺伝子を交叉させ、突然変異を起こし、生成される。遺伝子のデータ表現や交叉や突然変異については様々な手法がある。

各世代に N 個体あるとし、 M 世代まで繰り返すとすると、手順は次のようになる。

- (1) ランダムに N 個体を作成
- (2) 各個体の適応度を計算
- (3) M 世代に達するか、適応度が十分に大きいなら

終了。そうでないなら、以下のように新しい子孫を作成

- 現世代から、適応度に応じた確率で、親となるペアを選択
- ペアの遺伝子を確率 p_c で交叉、確率 p_m で突然変異させる

(4) 古い世代を新しい世代で置き換え、(2)へ

遺伝的アルゴリズムの実装には以下を設計する必要がある。

- 遺伝子から個体への変換方法
- 交叉や突然変異などのオペレータ
- 洗濯方法
- 適応度の計算

本研究では、パラメータが遺伝子でプレイヤーが個体であるので変換方法は自明であり、交叉や突然変異、選択については後述の遺伝的アルゴリズムライブラリを使用しているため、不要である。さらに、適応度は今回調査を行う目的関数であるため、遺伝的アルゴリズムを行うのに必要とされるものはそろっている。

3. 調査と実験

調査と実験は以下の手順で行う。調査対象の目的関数の有用性を調べるため、適応度関数として用い、遺伝的アルゴリズムによる調整を行う。各目的関数毎に得られたプレイヤーについて、対戦や各目的関数の値の計測を行い、性能評価を行う。得られた結果と調整に要した時間などから適切な目的関数について考察を行う。

遺伝的アルゴリズムを用いる理由は、学習モデルや目的関数に設ける制約を少なくするためである。また、チェスにおいて成功した例があり、有望な手法と考えられるからである¹⁰⁾。今回の調査対象は将棋とし、モンテカルロ木探索プログラムを用いる。モンテカルロ木探索はアルファベータ探索のプログラムに現在のところ敵わないが、応用が模索されている。

本研究では、レーティングとの相関、推定誤差、計算時間の面から適切な目的関数を調査する。強いプレイヤーの作成が目的であるが、「強さ」は対戦によって測ることが一般的で、レーティングによって計測される。対戦には長時間を要するため、より短時間のものが望ましい。遺伝的アルゴリズムによる調整では、ランダムに初期値を生成し、適応度の高いもの同士の組合せから新しいパラメータを作ることになる。この調整が正しく動作するならば、レーティングを近似するような適応度関数を使うことで強いプレイヤーの作成を行う事ができる。そこで、目的関数とレーティングに

ついて回帰分析を行い、相関や推定誤差を見ることで適切な目的関数の調査を行うことができる。これは、Anantharaman による性能評価指標の研究¹⁾と同様の手法であるが、プレイヤーの数の多さ、レーティングの計測を実際に 1,000 局以上行った点や回帰分析後に検定を行なった点などより正確なデータを取るようになっている。

3.1 実験設定

遺伝的アルゴリズムにおいては、David-Tabibi らの実験¹⁰⁾に従い、10 個体 50 世代とし、変異率を 0.05、交叉率を 0.75 として調整を行った。なお、遺伝的アルゴリズムには、マサチューセッツ工科大学の Matthew Wall による遺伝的アルゴリズムのライブラリ GALib^{*1}を使用した。今回は、遺伝子のデータ表現はパラメータベクトルをビットで表現したものとし、遺伝子のペアについて、長さ l とすると、 i ($i = 1, 2, \dots, l$) 番目のビットの前後で遺伝子を入れ替える一点交叉、突然変異は遺伝子のビットをランダムに反転させる方法を用いている。

実験用の将棋プログラムには、探索パラメータの調整に用いるテストプログラムとして GPS 将棋^{*2}(ver.2724, Open Shogi Library^{*3} ver.4446)を利用したモンテカルロ木探索プレイヤーを作成し、対戦相手の対照プログラムに Bonanza^{*4} ver.6.0 を用いた。なお、強さを揃えるため、参照プログラムの探索ノード数を 10,000 ノードに限定した。手調整のモンテカルロ木探索プレイヤーでは勝率が 0.53 程度となり、対戦相手として適当であると考えた。モンテカルロ木探索プレイヤーの実装については、評価関数を用いたプレイアウト打ち切り手法²⁰⁾を採用している他、Sato ら¹⁴⁾にならい、キラームーブやヒストリーヒューリスティックを適用している。例えば、ある深さにおける過去の最善手を覚えておき、モンテカルロ木探索の木探索時にその手と一致する場合は UCB1 のバイアス項を定数倍するという事を行っている。

調整するパラメータは、10 パラメータあり、それぞれの値域と調整の際のステップ、手調整による値を表 1 にまとめた。手調整のプログラムについては、パラメータを変えての対戦テストや問題集の正答数の比較などから試行錯誤でパラメータを決定した。各パラメータについて説明すると、p.o. 深さはプレイアウト

*1 <http://lancet.mit.edu/ga/>

*2 <http://gps.tanaka.ecc.u-tokyo.ac.jp/gpsshogi/>

*3 <http://gps.tanaka.ecc.u-tokyo.ac.jp/gpsshogi/pukiwiki.php?OpenShogiLib>

*4 http://www.geocities.jp/bonanza_shogi/

表 1 パラメータ			
	値域	ステップ	手調整
p.o. 深さ	[0, 64]	2	100
ノード展開	[1, 32]	1	1
rating weight	[1, 4]	1	4
PB	[0.0, 3.1]	0.1	0.5
FPU	[0.0, 3.1]	0.1	0.9
UCB	[0.0, 3.1]	0.1	1.0
SEE-penalty	[-1.55, 1.55]	0.1	1.0
Killer	[0.0, 3.1]	0.1	1.1
Bigram-Killer	[0.0, 3.1]	0.1	1.25
HH	[0.0, 3.1]	0.1	1.1
HH tics	[0.0, 0.62]	0.02	0.05

を行う最大深さを表し、ノード展開は、ノードを何回訪問した時にノードを展開するかの閾値を表す。rating weight は指手の rating に関する重みで Progressive Bias⁶⁾ に関係し、PB は手選択時の UCB1 式に加えられる Progressive Bias に対する重みとなっている。FPU は各ノードの初期 UCB1 値、UCB は UCB1 式のバイアス項に対する重みで、SEE は静的駒交換値の値に基づくペナルティで歩一枚以上の駒損する時にかけられるものである。Killer がキラームープ、Bigram Killer は 2 手組みでのキラームープ、HH はヒストリーヒューリスティックに基づいてバイアス項に加える値と、HH tics はヒストリーヒューリスティックの値の刻みに関連する値となっている。UCB, Killer, Bigram Killer, HH は全てバイアス項にかかるが、Bigram Killer, Killer, HH, UCB の順に優先度がある。

調整に用いた目的関数は、関連研究で挙げた調整手法で使われている目的関数のうちモンテカルロ木探索への適用が容易であったものを採用した。各目的関数で特に記述しない限り、各プレイヤーのプレイアウト数 3,000、局面数 1,000 で実験している。二乗誤差と一致数、コストでは、教師例を作成する必要がある。これらは棋譜からランダムに取り出した局面に対して、手調整のプレイヤーで 100,000 プレイアウトを行なって得た勝率や指手を教師としている。

- 対戦：対照プログラムと 100 戦した勝数
- 問題集：「らくらく次の一手」^{21),22)} 432 問の問題集の正答数
- 二乗誤差：教師の勝率と各プレイヤーの勝率との二乗誤差
- 一致数：教師の指手と各プレイヤーの指手が一致した数
- コスト：教師の指手を、各プレイヤーが選ぶまでにかかるプレイアウト数（最大で 3,000 プレイアウト）

なお、適応度関数は最大化するものなので、二乗誤差

表 3 パラメータの調整結果

	対戦	問題集	二乗誤差	一致数	コスト
p.o. 深さ	46	6	34	0	46
ノード展開	1	2	1	2	1
rating weight	4	4	2	1	4
PB	1.8	2.1	2.2	2.6	0.9
FPU	1.4	1.7	1.1	0.8	1.1
UCB	1.4	2.6	0.6	0.2	0.6
SEE-penalty	1.45	1.45	1.55	1.55	1.35
Killer	1.2	1.2	2.3	2.8	2.4
Bigram-Killer	0.9	1.5	3.1	2.3	1.8
HH	1.2	2.5	1.1	2.6	2.3
HH tics	0.22	0.46	0.58	0.32	0.52

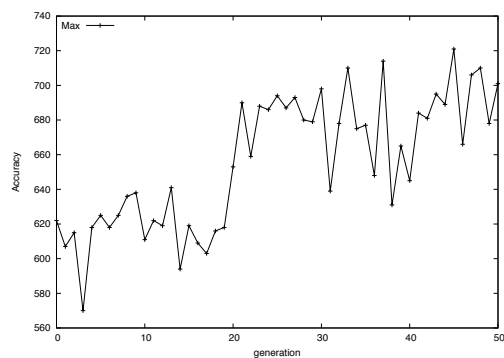


図 1 一致数による GA の遷移 (各世代における最大値)

とコストについては最大値からそれぞれの値を引いたものを適応度関数として利用した。

上記の目的関数と関連研究について述べる。対戦はモンテカルロ木探索囲碁の探索パラメータの調整⁷⁾、チェスの延長パラメータの調整³⁾、強化学習^{2),17)} で使われている。問題集とコストそれぞれは使われていないが、問題集を使った探索コストはチェスにおいて遺伝的アルゴリズムを使った探索パラメータの調整に使われている¹⁰⁾。深い探索結果との二乗誤差はモンテカルロ木探索囲碁の方策の学習に使われている^{13),15)}。深い探索結果との平均誤差はチェスの遺伝的アルゴリズムを使った評価関数の調整に使われている⁹⁾。深い探索結果との一致数は Anantharaman の結果に対応する¹⁾。

実験には CPU: Core i7 970 3.20GHz, メモリ 24GB のマシンを利用した。

3.2 調整結果

調整にかかった時間と、各目的関数の平均値や値域などの情報を表 2 にまとめた。二乗誤差と一致率、コストでは教師の計算にそれぞれ 6 時間かかっているが、これらのデータは共有が可能であるので、表中では括弧内に追加している。対戦を 100 局と少なめに設

表 2 パラメータ調整の要約					
	対戦	問題集	二乗誤差	一致数	コスト
平均値	26.855	216.544	3565.94	587.61	16211.70
最小値	1	75	3071.77	277	3154
最大値	61	287	3824.29	721	19575
値域	[0, 100]	[0, 432]	[0, 4000]	[0, 1000]	[0, 30000]
時間	187.5	13	57 (+6)	44.5 (+6)	25 (+6)

定したが、それでも他の手法と比べ長い時間がかかっている。局面数に違いのない二乗誤差と一致数で 10 時間の差が生じているが、理由は不明である。コストは一致数と同じ局面で正解と一致したら探索をやめるので、一致数よりも短いのは予想された通りの結果である。

パラメータの調整結果を表 3 に示す。パラメータはノード展開の閾値と SEE のペナルティを除いていずれもばらついた値となっている。ノード展開についてはプレイアウト数が少ないため、木を早く展開することが強さにつながったと考えられる。SEE のペナルティについては駒損する手にペナルティをかけるのは自然であり、かつ最大値の 1.55 に近い値となっており、より大きな値を取ろうとして最大値周辺の値に落ち着いた可能性が考えられる。

一致数を目的関数とした調整の途中経過を図 1 に示す。途中経過の図は各世代の最大値をプロットしたものである。なお、いずれの目的関数でもグラフは概ね右肩上がりとなっていた。

続いて、性能評価を行うため、得られたプレイヤーと対照プログラムとの対戦、得られたプレイヤー同士の対戦を行った。得られたプレイヤー同士では定跡の関係で同じゲームを行うことがあったため、実際の棋譜から 30 手進めた 500 局面を使い、先後を入れ替え 1,000 局の対局を行った。対照プログラムとの対戦では、1,000 局のうち重複は 10 ゲームに未達だったため、重複を許したまま 1,000 局の対戦を行った。対戦結果を表 4 にまとめた。結果から、勝率の高い順に並べると対戦と二乗誤差、一致数、手調整は同程度で、次にコスト、問題集となっている。調整にかかった時間を考えると、二乗誤差と一致率が目的関数として適当に考えられる。

3.3 解 析

対戦結果からレーティングを計測し、各目的関数の値との回帰分析を行い、適切な目的関数についての解析を行う。

レーティングと目的関数の値を計測した。レーティングの計測には、Rémi Coulom の Bayeselo^{*1} を利用した。Bayeselo により、Bayesian によるレーティング

表 5 レーティング		
	Bayeselo	Elostat
手調整	41.891	32.473
対戦	48.247	38.007
問題集	-137.473	-108.414
二乗誤差	36.141	28.312
一致数	43.521	33.829
コスト	-14.329	-10.587
参照	-17.997	-13.62

表 6 各プレイヤーによる目的関数の値

プレイヤー	対戦	問題集	二乗誤差	一致数	コスト
手調整	0.536	280	3745.33	716	17663
対戦	0.568	281	3664.35	691	17736
問題集	0.3355	287	3541.61	582	16742
二乗誤差	0.55	267	3789.41	699	19118
一致数	0.581	273	3766.01	700	19062
コスト	0.5485	245	3730.02	706	19119

と Elostat によるレーティングの両方が得られるので併記したが、Bayesian Elo Rating と Elostat との間に値の差はあるが傾向に違いは見られないため、以下では Bayesian Elo Rating の結果を用いる。また、二乗誤差と一致数、コストでは調整に用いた 1,000 局面とは別の 1,000 局面を使って計測した。対戦については前述の対戦時に行った参照プログラムとの 1,000 局の勝率を使い、問題集は代わりとして適当なものがなかったため、同じものを利用している。レーティングを表 5 に、目的関数の計測結果を表 6 にまとめた。なお、レーティングの 95%信頼限界は ± 8.9 程度であった。

レーティングの大小は表 4 から予想した結果と一致している。目的関数の値については、一致数を除き、各目的関数で調整した結果が一番高い値となっており、遺伝的アルゴリズムによって適応度関数が高くなるように調整が行われていることがわかる。ただ、問題集やコストはレーティングでは他のものに劣るので、強いプレイヤーの作成という観点では適切な結果とは言いがたい。調整されたプレイヤーが過学習のようになっているか、目的関数が適切でないことが考えられる。

3.3.1 回 帰 分 析

目的関数の適切さについて考えると、強いプレイヤーの作成を目的としているのでレーティングを推定できるようなものが望ましいことになる。そこで、各目的関

*1 <http://remi.coulom.free.fr/Bayesian-Elo/>

表 4 対戦結果: 1,000 局の勝率で、引分を 0.5 勝として計算している

	対照	手調整	対戦	問題集	二乗誤差	一致数	コスト
手調整	0.536	-	0.499	0.6875	0.510	0.514	0.5735
対戦	0.568	0.501	-	0.708	0.501	0.508	0.584
問題集	0.3355	0.3125	0.292	-	0.324	0.3275	0.357
二乗誤差	0.550	0.490	0.499	0.676	-	0.5	0.5635
一致数	0.581	0.486	0.487	0.6725	0.5	-	0.607
コスト	0.5485	0.4265	0.416	0.643	0.4365	0.393	-

表 7 レーティングと各指標の相関

	対戦	問題集	二乗誤差	一致数	コスト
相関係数	0.95233	0.19220	0.83299	0.92067	0.58234
c0	-388.842	256.102	-2457.95	-911.542	-764.018
c1	753.787	-0.92995	-0.66402	1.34032	0.42052
誤差	24.753	79.626	44.893	31.672	65.961
t 値	6.2434	0.39169	3.0111	4.7172	1.4327

表 10 中央値のプレイヤーを含めたレーティングと各指標の相関

	対戦	問題集	二乗誤差	一致数	コスト
相関係数	0.98834	0.78256	0.90266	0.95890	0.71113
c0	-367.113	-977.81	-2949.56	-1411.91	-725.58
c1	915.452	3.88715	0.81794	2.20952	0.04253
誤差	24.643	100.769	69.654	45.924	113.795
t 値	19.474	3.7709	6.2926	10.139	3.0344

数を説明変数、レーティングを被説明変数として回帰分析を行い、結果を表 7 にまとめた。レーティングは目的関数と $c1, c0$ から $Rating = c1 * Objective Function + c0$ として推定される。表中の「誤差」はモデルの標準偏差である。 t 値は $c1 = 0$ という帰無仮説における t 値であり、問題集とコストは帰無仮説を有意水準 5% で棄却できず、これらによる回帰式は適切でなかったということになる。

誤差について見ると、対戦によるレーティングの近似と一致数によるレーティングの近似とは大きな差はなく、対戦結果自身がレーティングの計算に使われていることを考えると、一致数による近似は計算時間からも目的関数として望ましいものと考えられる。

3.3.2 プレイヤを増やしての実験

データを増やすことで精度を高めることを考え、遺伝的アルゴリズムによる調整結果で得られたプレイヤーから、各目的関数の値が中央値となっているプレイヤーを追加し、レーティングの計測と目的関数の値の測定を行った。なお、時間に限りがあるため、レーティング計測の対戦は中央値のプレイヤー同士の対戦、中央値のプレイヤーと参照プログラム、手調整プレイヤーとの対戦のみ行なった。対戦結果を表 8 に、測定結果を表 9、回帰分析の結果を表 10 にそれぞれまとめた。この場合、いずれの目的関数も帰無仮説 $c1 = 0$ を有意水準 5% で棄却する。プレイヤーを増やしたことで問題集と

表 11 局面数とプレイアウト数を変えた場合の相関

	局面数	1k	2k	4k	1k	1k
	プレイアウト	3k	3k	3k	6k	12k
相関係数		0.9589	0.9258	0.9479	0.9559	0.9377
誤差		45.924	61.186	51.542	47.524	56.224
t 値		10.139	7.3470	8.9303	9.7669	8.0986

コストの結果は大きく改善しているように見える。

3.3.3 プレイアウト数や局面数の増加

プレイアウト数や局面数による影響を見るために、それぞれを 2 倍、4 倍として、各プレイヤーを使って一致数を計測し、回帰分析を行った。結果を表 11 にまとめた。いずれも有意な改善は見られなかった。局面数についてはまだ数が不足していることは考えられる。

4. 考 察

一致数と二乗誤差はレーティングとの相関が高く、調整にかかる時間も対戦より短くなるので目的関数として有用であると期待される。モデルの推定誤差について考えると、対戦を除くと一致数が最小である。さらに対戦結果はレーティング計算に使われていることを考えると、対戦による誤差は小さめに見積もられている可能性が高く、一致数の誤差は本来の対戦による誤差にかなり近いと考えられる。また、調整時間を考えると、対戦の 4 分の 1 程度と十分に小さく、解析時やレーティング計測時の対戦は 1,000 局行なっているため、40 分の 1 となる。時間については他の目的関数でも同様ではあるが、回帰分析の結果と合わせ、目的関数として一致数が有用であると考えられる。

問題集とコストの結果について考察する。

4.1 問題集

問題集の正答数による比較について考察を行う。他の目的関数と比較し、調整結果のレーティングは低く、少ないプレイヤーでは帰無仮説を棄却できず、回帰分析による近似は適切でないなど最も悪い結果となっている。目的関数としては、局面と正解手が与えられた時にプレイヤーと正解手との一致数を使っており、一致数と同じものであり、一致数との違いについて考える。まず、局面数について考えると問題集は半分以下であり、目的関数の計測時点での誤差が大きくなっている

表 8 中央値プレイヤの対戦結果: 1,000 局の勝率で、引分を 0.5 勝として計算している

	対照	手調整	対戦	問題集	二乗誤差	一致数	コスト
対戦	0.256	0.2255	-	0.4705	0.466	0.5545	0.5605
問題集	0.313	0.2235	0.5295	-	0.4616	0.598	0.639
二乗誤差	0.2695	0.2125	0.534	0.5385	-	0.624	0.6295
一致数	0.205	0.130	0.4455	0.402	0.376	-	0.537
コスト	0.147	0.1495	0.4395	0.361	0.3705	0.463	-

表 9 中央値のプレイヤを含めた目的関数
(表下部は中央値のプレイヤ)

プレイヤ	Rating	対戦	問題集	二乗誤差	一致数	コスト
手調整	152.537	0.536	280	3745.33	716	17663
対戦	159.116	0.568	281	3664.35	691	17736
問題集	-26.611	0.3355	287	3541.61	582	16742
二乗誤差	147.004	0.55	267	3789.41	699	19118
一致数	154.382	0.581	273	3766.01	700	19062
コスト	96.532	0.5485	245	3730.02	706	19119
対戦	-139.754	0.256	217	3593.45	595	13479
問題集	-109.844	0.313	218	3577.9	601	16914
二乗誤差	-101.398	0.2695	244	3533.87	593	18566
一致数	-200.485	0.205	194	3308.42	563	10990
コスト	-224.582	0.147	236	3302.65	541	16068

可能性が考えられる。

他に、教師の作成やプログラムの性質に起因すると考えられる。例えば、モンテカルロ木探索の特徴として長い手順の読み切りが不得意点があるが、問題集における正答はその局面において 1 手だけある長い手順で良くなる手であることが多く、棋譜からランダムに取り出した局面に比べて偏っている傾向がある。一方の一致数では、多くのプレイアウトを行うモンテカルロ木探索プレイヤの指手が答となり、モンテカルロ木探索プレイヤの得意な手が選ばれやすくなると考えられる。このように、問題集を目的関数として用いた場合にうまくいかない理由として、局面と正解の性質の違いが原因の 1 つとして考えられる。

4.2 探索コスト

コストについての考察を行う。問題集ほどではないが、対戦や一致数、二乗誤差に比べるとレーティングは低く、最初の解析では帰無仮説を棄却できなかったなど悪い結果となっている。正解手は一致数と同じ教師を使い調整を行なっているため、コストの計測方法に起因するのではないかと考えられる。チェスにおいて同様の目的関数を使っていた場合¹⁰⁾、対象はアルファベータ木探索であり、今回はモンテカルロ木探索を使っている。モンテカルロ木探索ではプレイアウト数が少ないうちは全ての手を試しており、正確性のないままに最善手として選ぶことがある。ごく少ないプレイアウトで精度の低いまま正解する方が得られる報酬が多いということになる。また、2,900 プレイアウトでの正解と 3,000 プレイアウトでの不正解との目

的関数の値の差が非常に小さく、正解による利得が少ないことなども原因として考えられる。プレイアウトの上限をより大きな値にすると、上限ぎりぎりでの正解の数は相対的に減るので、性能が改善する可能性がある。

5. おわりに

探索パラメータの自動調整のために、適切な学習のモデルの選択が必要であり、本研究では適切な目的関数の調査を行った。目的関数として、対戦や問題集、値の二乗誤差、指手の一致数、正解するのに要する探索コストを利用した。将棋を対象にモンテカルロ木探索プログラムの探索パラメータの調整を行い、各目的関数で調整したプレイヤを得た。各プレイヤの性能評価と、目的関数によるレーティングの近似を行うなどの解析を行い、計算時間とレーティングとの相関や近似の誤差から、このケースでの適切な目的関数としては、指手の一致数が良いという結論に至った。

今後の課題としては、より詳細なデータとして、局面数やプレイアウト数の違いの影響や評価関数の学習で使われたような兄弟手の比較のような他の目的関数についての調査、調整にかかる時間が一緒になるよう局面数などを変更した場合の調整などが考えられる。また、今回の手法をアルファベータ木探索の将棋やチェス、モンテカルロ木探索囲碁へと応用すること、より多くのパラメータを調整すること、今回得られた結果を元に、遺伝的アルゴリズムでない探索パラメータの自動調整への応用などが考えられる。

参 考 文 献

- 1) Thomas S. Anantharaman. Confidently selecting a search heuristic. *ICCA Journal*, Vol. 14, No. 1, pp. 3–16, 1991.
- 2) Jonathan Baxter, Andrew Tridgell, and Lex Weaver. Learning to play chess using temporal differences. *Machine Learning*, Vol. 40, No. 3, pp. 243–263, 2000.
- 3) Yngvi Björnsson and T. Anthony Marsland. Learning extension parameters in game-tree search. *Inf. Sci.*, Vol. 154, No. 3-4, pp. 95–118, 2003.
- 4) M. Buro. Improving heuristic mini-max search by supervised learning. *Artificial Intelligence*, Vol. 134, No. 1–2, pp. 85–99, January 2002.
- 5) Michael Buro. Experiments with multi-probcut and a new high-quality evaluation function for othello. In H. Iida H. J. vanden Herik, editor, *Games in AI Research*, pp. 77–96. Van Spijk, 2000.
- 6) Guillaume M. J-B. Chaslot, Mark H.M. Winands, H. Jaap Van Den Herik, Jos W. H.M. Uiterwijk, and Bruno Bouzy. Progressive strategies for monte-carlo tree search. *New Mathematics and Natural Computation (NMNC)*, Vol. 4, No. 03, pp. 343–357, 2008.
- 7) Chaslot, Mark H. Winands, István Szita, and Jaap H. vanden Herik. Parameter tuning by the cross-entropy method. In *Proceedings of EWRL*. Springer, 2008.
- 8) Rémi Coulom. Computing elo ratings of move patterns in the game of go. In *Computer Games Workshop*, Amsterdam / The Netherlands, 2007.
- 9) Omid David-Tabibi, Moshe Koppel, and Nathan Netanyahu. Expert-driven genetic algorithms for simulating evaluation functions. *Genetic Programming and Evolvable Machines*, pp. 1–18, 2010. 10.1007/s10710-010-9103-4.
- 10) Omid David-Tabibi, Moshe Koppel, and Nathan S. Netanyahu. Genetic algorithms for automatic search tuning. *ICGA Journal*, Vol. 33, No. 2, pp. 67–79, 2010.
- 11) Hilmar Finnsson and Yngvi Björnsson. Simulation-based approach to general game playing. In *Proceedings of the Twenty-Third National Conference on Artificial Intelligence (AAAI-2008)*, pp. 259–264, 2008.
- 12) John H. Holland. *Adaptation in natural and artificial systems*. MIT Press, Cambridge, MA, USA, 1992.
- 13) Shih-Chieh Huang, Rémi Coulom, and Shun-Shii Lin. Monte-Carlo Simulation Balancing in Practice. In *International Conference on Computers and Games*, 2010.
- 14) Yoshikuni Sato and Reijer Grimbergen. A shogi program based on monte-carlo tree search. *ICGA Journal*, Vol. 33, No. 2, pp. 80–92, 2010.
- 15) David Silver and Gerald Tesauro. Monte-carlo simulation balancing. In *International Conference on Machine Learning*, pp. 119–952, 2009.
- 16) Yoshimasa Tsuruoka, Daisaku Yokoyama, and Takashi Chikayama. Game-tree search algorithm based on realization probability. *ICGA Journal*, Vol. 25, No. 3, pp. 145–152, 2002.
- 17) Joel Veness, David Silver, William Uther, and Alan Blair. Bootstrapping from game tree search. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pp. 1937–1945, 2009.
- 18) 美添一樹. モンテカルロ木探索 – コンピュータ囲碁に革命を起こした新手法. 情報処理, Vol. 49, No. 6, pp. 686–693, 2008.
- 19) 関栄二, 三輪誠, 近山隆. モンテカルロ将棋における方策の学習. ゲームプログラミングワークショップ 2011 論文集, pp. 104–107, 2011.
- 20) 竹内聖悟, 金子知適, 山口和紀. 将棋における, 評価関数を用いたモンテカルロ木探索. ゲームプログラミングワークショップ 2010 論文集, pp. 86–89, 2010.
- 21) 日本将棋連盟書籍 (編). ラクラク次の一手 基本手筋集. 日本将棋連盟, 2003.
- 22) 日本将棋連盟書籍 (編). ラクラク次の一手 2 基本手筋集. 日本将棋連盟, 2003.
- 23) 保本邦仁. コンピュータ将棋における全幅探索と futility pruning の応用. 情報処理, Vol. 47, No. 8, pp. 884–889, 2006. ミニ小特集 コンピュータ将棋の新しい動き.
- 24) 保本邦仁. 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.