5. 第23回世界コンピュータ将棋選手権自戦記

応専

― Bonanza選手権バージョンの紹介―

保木邦仁（電気通信大学）

Bonanzaについて

筆者がコンピュータ将棋に本格的に興味を持ち始めたのは2004年頃である．2005年に拙作のプログラムBonanzaをインターネットに公開，以降2006年からコンピュータ将棋協会が毎年5月に開催する選手権に出場し続けている．これまで上位プログラムの1つとして，選手権の熾烈な戦いを盛り上げてきた．今年（2013年）の5月で8回目の出場となり，戦績は優勝2回，2位1回，3位1回，4位1回，5位2回，9位（予選落ち）1回である．

上位プログラム群の実力は接近しているようで，Bonanzaの戦績には実力だけでなく運の要素も強く反映されている．2度優勝してはいるが，それぞれ一局ずつ，将棋の内容では負けていながら相手プログラムの不合理な動作により勝っている．決勝リーグは7回戦しかなく，そのうちの1局で星が黒から白にひっくり返ることの最終結果への影響は大きい．

2005年当初のBonanzaの作成方針は，局面評価関数の機械学習と，チェスで成功した探索法の利用であった．この方針は基本的には現在も変わっていないが，それでも選手権において毎年トップ争いを繰り広げるために，以後さまざまな工夫を凝らしてきた．本稿では，Bonanzaに施してきたこれらの工夫の一部を簡単に紹介する．

ハードウェアの工夫：分散並列環境の利用

ここ数年に選手権上位プログラムに起きた変化の1つとして，分散並列環境の利用が挙げられる．2008年までは各出場者は会場へ1台の計算機を運搬し選手権に参加していたものだった．しかし，2009年には小幡らの文殊が計算機3台を利用して選手権3位という好成績を得ている[1]．彼らは，合議法によって複数台の計算機を利用した．これは，当時ソースコードを公開したBonanzaの評価関数に乱数値を加え，これを複数プロセス走らせ，多数決を取ることにより質の良い指し手を得るものであった．それからたった3年後，2012年には金子・田中らのGPS将棋が約800台もの計算機を利用して選手権で優勝している[2]．彼らは，分散並列探索法により非常に多くの計算機を利用した．これは，将棋の大きな探索空間を部分木に分割し，この部分木を分散並列環境により手分けして探索して質の高い指し手を得るものであった．

筆者もこのような分散並列化の波に乗り遅れまいと頑張り，2010年からデスクトップマシン複数台の利用に挑戦している．2012年，2013年のBonanzaでは，あから2010の開発に携わった経験を活かし[3]，これら合議法と分散並列探索を組み合わせた分散並列化を行った．今年の選手権では約30台のIntel Xeonマシンを使用した．GPS将棋の約800台という数字と比較すると見劣りするが，1台あたりのマシン性能ではBonanzaの方が優っていた．選手権で採用されている25分切れ負けという短時間の対局では，通信オーバヘッドや集計処理の負荷が少ない少数精鋭型の分散並列環境は，それほど悪くなかったのではないかと感じている．

図-1に，今年の選手権で使用したBonanzaの構成を示す．多数決合議[1]，大規模分散並列化[2]，詰み探索専用プロセスとの通信により複数計算機の有効利用がなされていた．図中灰色の部分は選手権会場内にて実行されたプログラムであり，対局の継続

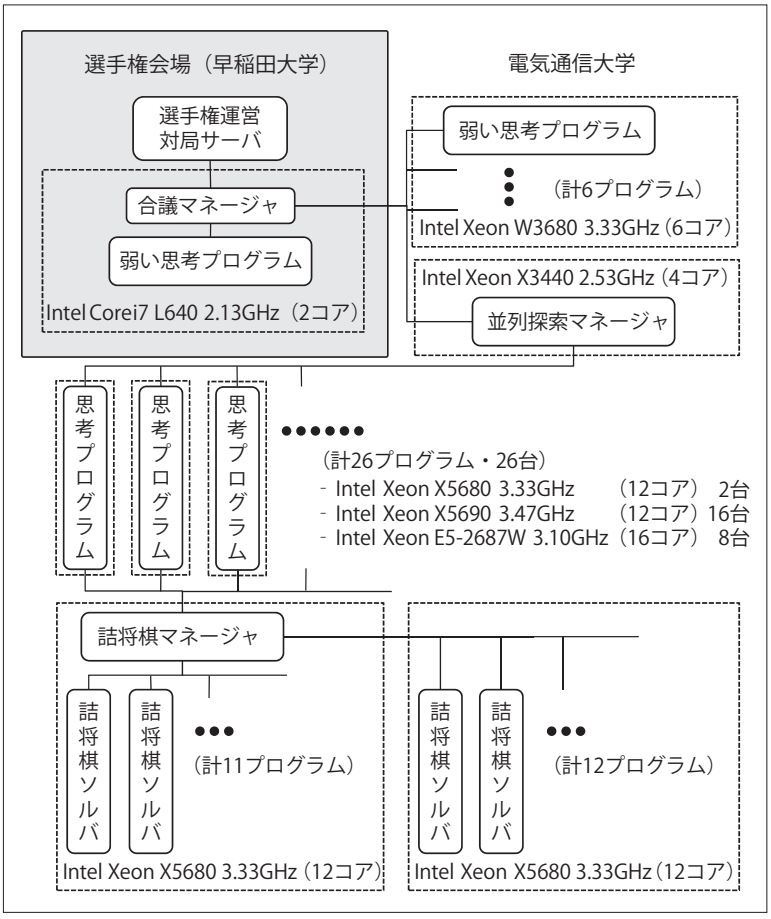

図 -1　第 23 回世界コンピュータ将棋選手権での Bonanza のシステム構成

に必要最低限な機能を備えていた．合議マネージャに接続されているプログラムは 7 個の弱い思考プログラムと 1 個の並列探索マネージャであった．各弱い思考プログラムは評価関数に小さな乱数が足されていて，探索結果にある程度のばらつきを持っていた．対局サーバに送信される指し手は，これら 8 プログラムの多数決により決定された．対戦相手の思考時間における予測読みもこの合議マネージャによりなされていた．

本構成における多数決合議が果たす役割は 2 つある．1 つめは，分散並列探索を行う一群の計算機（27 台）と，そのネットワーク接続に異常が起きたときのための異常終了回避である．票の重みは並列探索マネージャが 1.0，弱い思考プログラムが 0.1 であった．したがって，並列探索マネージャから指し手が返ってくる限りにおいては，これ単独で指し手が決定されることとなる．本番前の動作チェックでは並列探索マネージャから指し手が返ってこない動作異常が確認された．しかし，選手権当日は運営側の努力によりインターネット接続が安定していたこともあり，このような動作異常は発生しなかった．2 つめの役割は，思考時間の短縮・延長の決定である．弱い思考プログラムも含めて票がばらついていた場合には思考時間を延長し，一致した場合には思考時間を短縮した．

並列探索マネージャには，26 個の思考プログラムが接続されていた．各思考プログラムはメモリ共有並列探索を行い，並列探索マネージャから割り当てられた 1 つ以上の内部節点から続く部分木を探索した．これらの内部節点は現局面から一～三手の深さにあることがほとんどで，毎回思考の前処理として行う仮探索で有望と判断された応手系列を参考にして構成されていた．各思考プログラムはまた，担当する部分木のルートとその全子局面の詰みの有無を確認するために，詰将棋マネージャとも通信を行う．この詰将棋マネージャには，計 23 個の詰将ソルバが接続されていた．

評価関数の機械学習の工夫：より良い極小解の探求

コンピュータ将棋プログラムを作成するにあたり，形成の優劣判断を行う局面評価関数の生成が 1 つの課題となる．ゲームの複雑性を反映して，この関数に含まれるパラメタの数は数万～数千万にもなり，手調整による値の設定は困難な作業となる．機械学習による局面評価関数に含まれる一群のパラメタ調整は，選手権初参加の 2006 年から Bonanza の強さの秘訣となってきた．しかし，最近ではこの成功事例が比較的有名となり [4]，他プログラムも各自工夫を凝らした評価関数の機械学習を行うようになってきた．これは，プロ棋士の棋力を超えようとする近

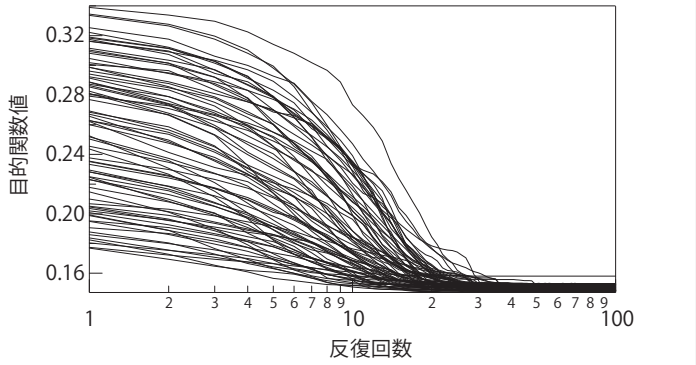

図 -2　13 個の駒価値の最適化約 80 セット．初期駒価値は一様乱数により与えられた．図中の表は，収束後に目的関数に最も小さい値を与えた駒価値と大きい値を与えた駒価値が示されている

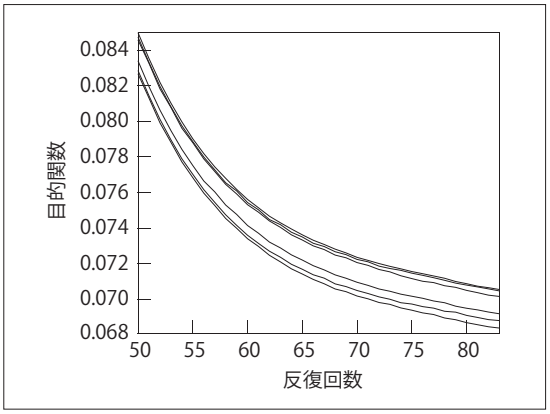

図 -3　乱数により生成された 6 つの異なる初期重みベクトルを用いた最適化の結果

年の将棋プログラムの強さ向上の原動力となっているように感じられる．

とにかく，評価関数の機械学習は Bonanza だけが恩恵をあずかっていたアドバンテージではなくなった．筆者もまた，選手権で良い成績をおさめるためには，評価関数の設計にさらなる工夫を凝らす必要があった．その中でも，2010 年頃から取り組んできた，特に効果が高かった方法をここで紹介する．

評価関数の機械学習では，探索結果とプロ棋士等の棋譜との指し手の不一致度を測る目的関数を設計し，これを最小化することによって評価関数内の重みベクトルを最適化する．この探索結果に依存する目的関数は当然凸関数などではなく，複数の極小解を持っている．

図 -2 は，駒価値の機械学習の初期値依存性を示す[5]．約 80 セットの初期駒価値は一様乱数により与えられた．これらの初期駒価値を用いた目的関数の反復ごとの減少の様子が図に示されている．おおよそ 50 回程度の反復で 13 個の駒価値が収束した．しかし，収束後の目的関数値はそれぞれ異なっていて，0.1475 から 0.1580 の範囲内でばらついている．最も小さい値に収束した駒価値は人間の直観と一致するが，大きい値に収束してしまった駒価値は明らかに不自然な値を持つ（歩の方が桂香よりも価値が高い）．この結果から，より良い極小解の発見は強いプログラムを作るために必要なことと考えられる．

図 -2 と同様な計算を，駒の位置関係も考慮に入れた全評価関数の機械学習に対しても行った．この評価関数は特徴ベクトルの線形重み和によりモデル化されていて，特徴ベクトルは駒価値に加えて，玉―駒の二駒の位置関係，玉―玉―駒や玉―駒―駒の 3 駒の位置関係から構成される．詳細は Bonanza のソースコードを参照されたい．

全評価関数の最適化は駒価値だけのものよりも計算に時間がかかるので，**図 -3** には 6 本の最適化の結果のみが示されている．異なる初期重みベクトルは乱数を用いて生成された．図には，異なる初期値からスタートした反復計算が，異なる目的関数値に収束していく様子が示されている[6]．**図 -4** で示されるように，良い重みベクトルでの目的関数値は 0.05 を下回る．初期重みベクトルを乱数により与えて，最適化を多数回行い最も良い値に収束するベクトルを発見するような方法は，ベクトルの次元がある程度大きくなると効率が悪くなるようだ．

図 -4 も同様に異なる初期重みベクトルからスタートした反復計算結果の比較を示すが，初期重みベクトルを用意する方法に工夫がなされている．ここでは，重みベクトルすべてを一度に最適化しない．特徴の出現頻度の順番（1. 駒価値，2. 二駒関係に関する特徴，3. 三駒関係に関する特徴）を考慮し，出現頻度の高いものから順番に最適化を行っていく．図 -4 実線の初期重みベクトルは，前処理として最

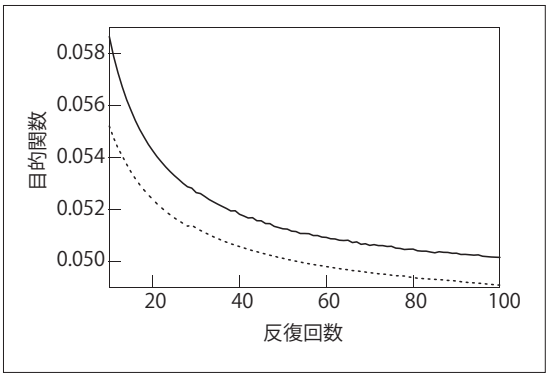

図 -4　2 つの異なる初期重みベクトルを用いた最適化の結果．駒価値のみを前処理として最適化し，ほかの重みはすべて 0 とした初期重みベクトルを用いた反復計算が実線で示される．一方，駒価値に加えて二駒の位置関係に関する重みも前処理として最適化し，ほかの重みはすべて 0 とした初期重みベクトルを用いた反復計算が破線で示される

適化された駒価値と，ほかはすべて 0 の値を持ったものである．一方，破線の初期重みベクトルは，駒価値に加えて二駒の位置関係に関する重みも前処理として最適化し，ほかは 0 の値を持ったものである．破線で示された反復計算が最も値の小さい極小点を発見した．出現頻度の高い順番に最適化を行うのが，より良い極小点を発見するコツのようである．

参考文献

1) 伊藤毅志，小幡拓弥，杉山卓弥，保木邦仁：将棋における合議アルゴリズム―多数決による手の選択，情報処理学会論文誌，Vol.52, No.11, pp.3030-3037 (Nov. 2011).
2) 金子知適，田中哲朗：最善手の予測に基づくゲーム木探索の分散並列実行，情報処理学会論文誌，Vol.53, No.11 (Nov. 2012).
3) Hoki, K., Kaneko, T., Yokoyama, D., Obata, T., Yamashita, H., Tsuruoka, Y. and Ito, T. : A System-Design Outline of the Distributed-Shogi-System Akara 2010, 14th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing, pp.466-471 (2013).
4) 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).
5) Hoki, K. and Kaneko, T. : The Global Landscape of Objective Functions for the Optimization of Shogi Piece Values with Game-Tree Search, Advances in Computer Games 13, Lecture Notes in Computer Science 7168, Springer, pp.184-195 (2011).
6) 保木邦仁，金子知適：Minimax 探索最適化の関数形，ゲームプログラミングワークショップ 2010 論文集，pp.67–70 (2010).

(2013 年 7 月 1 日受付)

保木邦仁（正会員）　hoki@cs.uec.ac.jp

2003 年東北大学理学研究科化学専攻修了（理学博士）．同年よりトロント大学化学科博士研究員．2006 年より東北大学にて研究員，助手として教育研究に従事．2010 年より電気通信大学特任助教．