

# 小さな将棋の解を得る手法の提案

柴原一友 但馬康宏 小谷善行  
東京農工大学工学教育部電子情報工学専攻

## 概要

我々は以前の研究で 3x3, 3x4 の将棋の解を得る研究を行った。本稿では 4x4 の将棋の解を求めることを目的とする。また、解を得る上で、単純な PDS 探索に加え、手順を限定することで探索性能を向上させる手法について提案する。また、提案した手法を 4x4 将棋に適用した結果、単純な PDS に比べ、全体で 10%、どちらの手法も解を得た場合でも 3% の性能向上が得られたことを示す。

## Approach to Solve Small Boards of Shogi

Kazutomo Shibahara Yasuhiro Tajima Yoshiyuki Kotani  
Tokyo University of Agriculture and Technology

## Abstract

We researched the solutions of 3x3 and 3x4 boards of Shogi. In this paper, we tried to solve 4x4 boards of Shogi. And we show new approaches that limit the scope of positions for performance gain. We show that the performance gain is 10% in all data, 3% in the data sets that was solved by normal PDS search and the method.

## 1. はじめに

筆者らは過去に、小さな将棋を解く研究を行った[1]。本稿では、4x4 の将棋の解を求めることを目的とする。4x4 の将棋は探索空間が膨大となり、単純な PDS 探索では難しい。そこで、本稿では小さな将棋に対し、棋譜や  $\alpha$   $\beta$  探索と組み合わせる方法についても提案し、その有効性を検証する。

## 2. 実験に用いたシステム

今回の実験では、[1]で使用した証明数探索 PDS を使用している。具体的には、PDS に対し、優越関係と証明駒、反証駒を導入したアルゴリズムである。PDS は、証明数・反証数とハッシュを利用した多重反復深化法になっている。アルゴリズムとしては、df-pn が PDS よりも優れているとされているが、今回の実装は見送っている。

## 3. 棋譜を用いた手順限定手法

### 3.1. アルゴリズム

前述のように、単純な PDS 探索では 3x3, 3x4 は解くことができても、4x4 を解くのは難しい。類似研究として、[2]では人間の棋譜を用い、末端からルートまでを逆にたどりながら証明数探索を用いて探索する方法が提案されている。さらに、通常のゲーム木探索に使用する評価関数を閾値として使用し、近似的な AND-OR 木の解を精練している。

本研究では、まず人間の用意する棋譜を 1 つ以上用意する。そのうちの一つの最終局面から証明数探索を実行し、証明できた場合はその親ノードの証明を行う。定められた制限時間を超過しても証明できなかった場合は、新たなルートを作成する。具体的には、用意した棋譜の中に同一の手順を持つ棋譜が存在した場合は、その棋譜に示された手で局面を進行する。もし存在しない場合は、駒価値だけの単純な評価関数を用いたゲーム木探索で得られた手で進行する。その局面においても証明数探索を実行し、制限時間内に証明できない場合はさらに延ばしていく。証明できた場合は、再び親のノードを証明するループへと戻る。棋譜に関しては用意しなくても探索は可能であるが、多く用意することで、より効率的に探索することができる。

### 3.2. 実験結果と考察

今回の実験では図1のような局面を使用した。棋譜に関しては今回は一つだけ使用している。また、駒が成ることが出来るのは敵陣の一段目だけである。実験の結果を表1に示す。

♞	♜	♝	♚
♙	♖	♗	♛

表1 実験結果

	探索ノード数	ハッシュ利用数	探索時間
提案手法、棋譜なし	102706	94554	45.718
提案手法、棋譜に正しい初手を与える	59893	54861	26.312
PDS	145114	132981	62.04

図1 実験に使用した局面

実験の結果、単純な PDS 探索よりも早く探索できることがわかる。ただし、棋譜なし適用に関しては、この局面で偶然起きた結果であるといえる。棋譜に対し、正しくない初手を与えた場合、探索時間は大幅に跳ね上がり、解が得られなかった。これは、提案手法がある一つの手順以下だけを重点的に調べるため、棋譜や $\alpha\beta$ 探索が解く上で最善な手を与えてくれない場合、探索量が膨れ上がる。しかし、複数の棋譜や条件を設定し、複数のコンピュータで動作させ、そのうちの一つでも解を得ることができれば、ゲームは解ける。優れた棋譜の集合を用意できれば、提案手法は単純な PDS では解くことの難しい問題を解けるだろう。人間が探索の様子を確認しながら、優良な棋譜集合を用意することで、問題を解くことに近づけると考えられる。

## 4. より汎用的な手順限定手法

### 4.1. アルゴリズム

提案手法は、優れた棋譜の集合が必要となり、手間をかけなければ解を得られない可能性が高い。そこで、この提案手法をより汎用的にする方法について考察する。PDS の行う最良優先探索は、人間から見て非効率的な部分も探索せざるを得ない。たとえば、開始局面が先手必勝であることを示すには、候補手のうち一つだけ先手必勝であることがわかればよい。しかし、最良優先探索はその他の、人間から見たら明らかに先のない手であっても最良優先探索であるがゆえに、探索しなければならない。一手だけでも限定して探索を行うことの有効性は先の実験結果からも明らかである。

そこで、提案手法の改良方法として、限定する局面を最良優先探索的に行う方法が考えられる。複数の探索経路を同時に保持し、複数存在する探索手順のうち、最終的な解を得るうえで有望と思われる手順を展開する方法が考えられる。基準としては、その結果を調べることが最終的な結果を導くのにどれだけ貢献できるのかということや、その局面がどれだけ解を得やすい局面であるか、ということなどがあげられる。もっとも単純な判断基準としては、証明数や反証数がある。しかし、証明数や反証数はある程度の深さだけを見て得られる情報であり、有望な局面を示す基準とはいきれない。よりよい判断基準を導入することも検討すべきと考えられる。

改良手法の手順について現時点で考えている手順を次に示す。

1. ルート局面を PDS 探索する。ルート局面を展開局面リストに登録する。
2. ルート局面の解が得られた場合は終了する。
3. 展開局面リストの末端局面の候補手における未展開の手から、なんらかの判断基準（証明数や棋譜情報、候補手の解が得られている割合等）を用いて、展開すべき手を決定する。
4. 3. で展開した手順の末端局面を PDS 探索で調べる。
5. 解が得られた場合は3. に戻る。
6. 3. で展開した局面(手順)を展開局面リストに登録する。
7. 2. に戻る。

この方法には、展開局面リストの最適化や、展開すべき手の決定方法など、考えるべき点は多く存在する。探索経路の選定方法は最良優先探索的であり、再帰的な深さ優先探索で代替する方法も検討する必要がある。

#### 4.2. 展開の判定に使用する評価関数

末端リストからの展開基準としては、証明数、反証数が考えられるが、実験の結果、あまりうまくいかない印象を受けた。これは、証明数、反証数は一時的な値であり、展開を続ける上で頻りに値が変化するため、その時々々の最小値はばらばらになる可能性が高く、あまり上等な判断基準とはいえないからであると思われる。

そこで、末端リストの手を指した後の局面における $\alpha\beta$ 探索の結果を判断基準のひとつとして使用した。このプログラムで実装されている $\alpha\beta$ 探索は、取り合い探索等のアルゴリズムが導入されていないため、水平線効果が起きやすく、精度の高いものではない。この精度を上げることで、さらに性能が向上すると考えられる。現在は深さ4までの反復深化とすることで、水平線効果の発生を多少下げている。

また、末端リストの候補手のうち、解が判明している手の割合も評価として組み込んでいる（閉塞度と以降呼ぶ）。これは、展開局面リストの増加を防ぐ目的も兼ねている。つまり、解が得られやすそうな手を重点的に調べることで、手順の解を判明させ、展開局面リストから外すことで、展開局面リストの増加を防ごうというものである。同時に、その展開局面リストの途中局面においても同様の割合を調べている。評価値はこれらの値の線形和で算出している。この最適化は行っていない。

#### 4.3. 実験環境と結果

実験対象として、自陣一段目に王とその他の駒を配置した局面 1806 局面を対象に実験を行った。成る段に関しては、王が隅にいる場合だけ、1~3 段目の場合を調べ、他の場合は1 段目だけを調べている。実験の結果、解が得られたのは 624 局面であり、うち後手勝ちが 9 局面であった。後手勝ちの局面を表 2 に示す。3 段目に駒を移動させることで相手の行動を抑えることができるため、先手勝ちが圧倒的に多くなったものと考えられる。

提案手法における線形和の重みの異なる二つの設定と単純な PDS 探索の探索時間の平均を表 3 に示す。ここで、設定 2 は設定 1 よりも開始局面に近い局面を優先的に探索する。すべてのデータとは、解を得られた局面のうち、自明であるものと、提案手法の開始時のルート局面における探索で解が得られるほど簡単な局面を除いた 229 局面で行った。また、両方が解を得たデータの数は 148 局面である。

表 3 提案手法と単純な PDS の 4x4 将棋探索時間の比較

	PDS	設定 1	設定 2
すべてのデータ	32353.57	28863.69	28728.96
両方が解を得たデータ	19993.29	19460.44	19841.43

表 2 後手勝ちの局面

駒配置	成段
桂王桂金	1
銀王桂飛	1
王角桂金	1
王桂桂金	2,3
王桂桂銀	2,3
王角桂桂	2,3

表 4 解の判明した 624 局面に対するそれぞれの解けた局面数

	解けた局面数	解けなかった局面数
PDS	603	21
提案手法(設定1)	565	59

#### 4.4. 考察

表を見ると、すべてのデータに関しては平均 10%程度の探索速度向上が見られた。ただし、これは一方が解を得られなかった場合も含まれている。提案手法は探索の打ち切りが平均して通常の PDS よりも早く設定されている。解を得られなかった数は、単純な PDS 探索は 21 個なのに対し、設定 1 は 59 個であった。解の判明した 624 局面に対して、それぞれの手法が解いた数を表 4 に示す。また、現時点ではハッシュのガーベジコレクションは実行していない上に、リハッシュは一定回数で打ち切れ、以降の情報は保存されない設定となっている。これらのことも提案手法に有利に働いている可能性がある。

また、両方が解を得たデータについてみると、提案手法の方が平均 3%ほど短縮されていることがわかる。これにより、提案手法は証明数探索の性能を向上することができているといえる。局面によって

は 1/4 に削減できているものもあつたり、またその逆の場合も存在する。それほど大きく探索性能が向上しなかった原因として、探索経路の候補手を選び出すのに使用された  $\alpha\beta$  探索が簡易的な探索であるため、効果的な経路を保存できなかつたことが考えられる。実験結果を見ると、探索量が増える場合は大幅に増えることが多く存在した。現在使用している探索は、単純な駒価値だけで深さ 4 の取り合い探索なしの探索である。先の実験で示したように、一手限定されることで探索の性能は大幅に向上する。この性能を上げることでより高い性能で解を得ることができると考えている。また、経路の評価も最適化されていないため、見当違いな経路を高く評価している可能性も考えられる。これらの改善も重要である。

## 5. おわりに

提案手法を 4x4 の将棋に適用させ、棋譜の集合次第で、単純な PDS 探索よりも速く解を得ることができることを示した。また、提案手法の改良方法について提案、実装し、その実験を行った。

今後の展望としては、取り合い探索の強化や評価関数の調整、df-pn 探索の導入、メモリ使用の効率化、その他の 4x4 将棋への適用などがあげられる。

## 参考文献

- [1] 後藤智章, 柴原一友, 乾伸雄, 小谷善行: 小さな将棋の解, Game Programming Workshop 2003
- [2] J. Schaeffer, Y. Bjornsson, N. Burch, A. Kishimoto, M. Muller, Solving Checkers, IJCAI2005
- [3] V. Allis, Searching for solutions in Games and Artificial Intelligence. PhD thesis, Department of Computer Science, University of Limburg, 1994.
- [4] Herik, H. J. van den, J. Uiterwijk, and J. van Rijswijck: Games Solved Now and in the Future, Artificial Intelligence Journal 134:pp.277-312. (2002).
- [5] D.E.Knuth and R.W. Moore : An Analysis of Alpha-Beta Pruning, Artificial Intelligence 6(4), pp.293-326 (1975).
- [6] Ayumu Nagai: A new AND/OR Tree Search Algorithm Using Proof Number and Disproof Number, Complex Games Lab Workshop, pp.40-45, 10 November (1998).
- [7] 脊尾昌宏: 詰将棋を解くアルゴリズムにおける優越関係の効率的な利用について, 第 5 回 ゲームプログラミングワークショップ, pp.129-136 (1999).
- [8] 長井 歩, 今井 浩: df-pn アルゴリズムの詰将棋を解くプログラムへの応用, ジャーナル アブストラクト Vol.43 No.06 - 020, 情報処理学会 (2002).
- [9] Louis V. Allis, Maarten van der Meulen, and H.Jaap van den Herik: Proof-Number Search, Report CS 91-01, University of Limburg, Maastricht, Netherlands, 1991. Also available at Artificial Intelligence, Vol.66, pp. 91-124 (1994).
- [10] D. M. Breuker, H. J. van den Herik, J. W. H. M. Uiterwijk, and L. V. Allis: A Solution to the GHI Problem for Best-First Search, Theoretical Computer Science 252(1-2): 121-149 (2001).

付録 4x4 将棋の実験結果の例

駒配置	通常の PDS		提案手法設定 1		提案手法設定 2	
	勝敗	ハッシュ 時間(s)	勝敗	ハッシュ 時間(s)	勝敗	ハッシュ 時間(s)
飛玉銀金	先手	35735 10359	先手	216300 71168	先手	216235 71156
歩玉金桂	先手	167036 27141	先手	98156 14529	不明	193753 30572
金玉金香	先手	118827 23930	先手	51458 10495	先手	51477 10507
桂玉金香	先手	245755 55248	先手	129787 25208	先手	129744 25211
金玉角飛	先手	19741 6208	先手	66196 20683	先手	66171 20675
桂玉角飛	先手	45325 12709	先手	133787 38030	先手	134229 38115
歩玉金飛	先手	78191 20021	先手	49025 12796	先手	61099 16125
歩玉銀飛	先手	28823 7420	先手	131971 34753	先手	132601 34872
桂玉銀飛	先手	76442 20948	先手	37811 10713	先手	37708 10680