情報処理学会論文誌　Vol. 51　No. 11　2021–2030 (Nov. 2010)

# 探索結果を利用した実現確率探索

佐藤佳州[†1,*1]　高橋大介[†1]

本論文では，探索結果に基づく実現確率探索を提案する．実現確率による探索打ち切りアルゴリズム（実現確率探索）は，コンピュータ将棋において注目を集めている探索法の 1 つであり，多くのトップレベルのプログラムがこのアルゴリズムをベースとした探索法を用いている．実現確率探索は探索深さの決定にプロの棋譜から求めた指し手の確率を用いることで，ありえそうな展開を深く探索するという特長を持つアルゴリズムである．この手法は，多くの場面において優れた結果を収めているものの，プロの棋譜から求めた確率を利用するため，プログラムの探索手法や評価関数の性質によらず確率がつねに固定である点に改善の余地があると考えられる．また，実現確率探索では「王手」「駒を取る手」といった表面的な特徴から確率を算出するが，これらの特徴に加え，探索中の履歴を考慮することでさらに性能を改善することができる可能性があると考えられる．本論文ではこれらの点について，プログラムの探索結果を利用し，評価関数や探索の深さに応じた確率を利用する，探索中に得られる情報を特徴として利用する，といった改良を行うことにより改善を試みた．実験の結果，提案手法は全幅探索や従来の実現確率探索に勝ち越すことに成功し，その有効性を示した．

# The Realization Probability Search Based on Search Results

Yoshikuni Sato[†1,*1] and Daisuke Takahashi[†1]

In this paper, we propose a realization probability search algorithm based on search results. The realization probability search is one of the search algorithms attracting much attention in the Computer-Shogi area, and it is used by many top-level programs. This method determines search depths according to probabilities of moves obtained from game records by professional players, and searches deeper for more probable moves. The realization probability search is an efficient algorithm, but it is considered that it still has room for improvement because probabilities of moves are fixed at any time. We improved the realization probability search by using the search results. In our method, probabilities change according to the rest of search depths and heuristics obtained while searching. In the result of our experiments, our program based on the proposed method is superior to existing methods.

## 1. はじめに

ゲームプログラミングの分野において，コンピュータ将棋はチェス，囲碁とならびさかんに研究されているテーマの 1 つである．コンピュータ将棋の棋力は，アルゴリズムの改良やハードウェアの進歩により，現在ではアマチュアトップに匹敵するレベルにまで達している．

将棋をはじめ，オセロ，チェスといった思考ゲームをコンピュータで実現する手法としては，探索と評価関数を用いるのが一般的である．効率の良い探索法の実現は難しい課題であり，将棋の場合，現在のコンピュータは 1 秒間に数十万〜数百万局面を探索するにもかかわらず，人間のトッププレイヤには到達していない．また，ハードウェアの進歩という点からみた場合，現在，CPU のクロックの向上は限界に達しつつある．今後，今までのようなハードウェアによる単純な性能の向上は難しく，ますます効率の良い探索法の研究が重要になると考えられる．

ゲーム木探索の手法としては，大きく分けて，全幅探索と選択探索が存在する．全幅探索はすべての手を探索対象とする手法，選択探索は知識による枝刈りを行い，指し手を絞って探索する手法である．将棋では合法手の多さから，以前は選択探索が主流であった．しかし，Bonanza[1] 登場以降，全幅探索も見直されつつあり，選択探索と全幅探索のどちらが優れているかという結論は今のところ得られていない．

選択探索のうち，注目を集めている手法の 1 つとして実現確率による探索打ち切りアルゴリズム[2),3)] がある．この手法は，プロの棋譜を基にした学習により指し手の探索深さを決定するというもので，ありえそうな展開をより深く探索できるという特長を持つ．実現確率による探索打ち切りアルゴリズムは世界コンピュータ将棋選手権でも多くのプログラムで採用され[3)–6)]，成功を収めている探索法の 1 つであるといえる．

実現確率による探索打ち切りアルゴリズムは非常に優れた探索法であるが，プロの棋譜から求めた確率を利用するため，プログラムの探索手法や評価関数の性質によらず，どのよう

†1 筑波大学大学院システム情報工学研究科
Graduate School of Systems and Information Engineering, University of Tsukuba
*1 現在，パナソニック株式会社先端技術研究所
Presently with Advanced Technology Research Laboratories, Panasonic Corporation

な状況でも確率が一定になってしまうといった点について，改善の余地があると考えられる．また，実際に探索を行っていくと，「王手」「駒を取る手」といった表面的な特徴だけでは分からないような指し手の善悪が分かってくる場合がある．このような場合，指し手の表面的な特徴に基づいた確率よりも，探索中に得られた情報を重視した方が高い性能を得ることができる可能性がある．本論文では，このような問題点をプログラムの探索結果を利用することにより改善した探索手法を提案する．

## 2. 関 連 研 究

### 2.1 実現確率による探索打ち切りアルゴリズム

実現確率による探索打ち切りアルゴリズム[2),3)]（以下，実現確率探索）は鶴岡によって提案された手法である．この探索法は多くの将棋プログラムで採用されており，さかんに研究が行われている[4),5),7),8)]．

一般的な探索法では，深さを探索の打ち切り条件とするのに対して，実現確率探索では深さの代わりに局面の実現確率を利用する．局面の実現確率は以下の式によって再帰的に定義される値であり，この値が大きい指し手をより深く探索する．

(局面の実現確率) = (親局面の実現確率) × (遷移確率)

ルート局面の確率は 1.0 とし，遷移確率には指し手の選択される確率を用いる．指し手の選択確率は，指し手を特徴ごとに分類し，その特徴に対応する指し手がプロの棋譜中でどれくらいの割合で指されたかを算出することにより求める．たとえば，「相手の駒を取る手」や「成る手」「王手」といった特徴を持つ指し手は，実際のプロの棋譜において選択されることが多く，遷移確率は高くなる．実現確率探索では，このような指し手をありえそうな手と見なし，深く探索する．一方で，大きな駒損をする手などは，プロの棋譜中で選択されることは少なく，遷移確率は低くなり，結果的に浅く探索されることになる．

実現確率探索の例を**図 1** に示す．各ノードに付与された値は局面の実現確率を，枝に付与された値は指し手の遷移確率を表す．なお，図 1 では探索を打ち切る実現確率の閾値を 0.3 としている．

図 1 では左側のノードほど実現確率が高く，より起こりやすい展開といえる．実現確率探索では，図 1 のように起こりやすい展開をより深く探索していることが分かる．

なお，実際には実現確率は対数をとった形で扱うのが一般的である．局面の実現確率は指し手の遷移確率の積となるため，対数をとることで乗算を加算に変換でき，一般的な探索手法における深さのように扱うことができる．たとえば，実現確率が 0.25 の局面において，

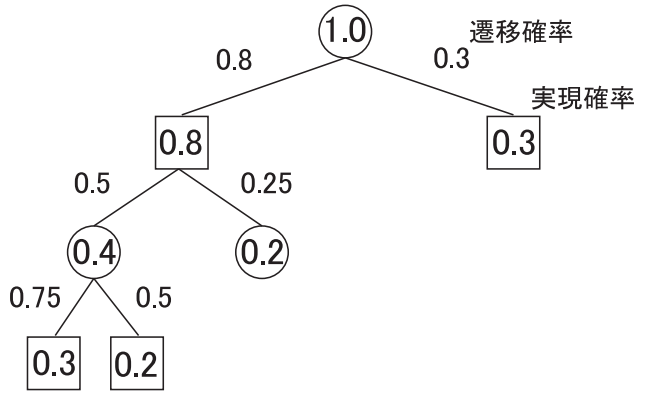


**図 1**　実現確率探索の例
Fig. 1　Example of the realization probability search.

遷移確率が 0.50 の指し手を指した場合，子局面の実現確率は $0.25 \times 0.50 = 0.125$ となる．この計算は，確率の対数[⋆1]をとり，符号を反転させると，$(-\log_2 0.25) + (-\log_2 0.50) = 3$ と変換できる．実際の探索中では，この実現確率の対数をとり符号を反転させた値が閾値に達した時点で探索を打ち切る．遷移確率の大きい指し手ほど，対数をとった値は小さくなるため，結果的に深く探索されることになる．以降，本論文中では，遷移確率および実現確率の対数をとり，符号を反転させた値を便宜的に「深さ」と呼ぶ．

### 2.2 ロジスティック回帰による実現確率探索

指し手の遷移確率については，以前は「王手」「駒を取る手」といった指し手の表面的な特徴を 1 つだけ利用して算出するのが一般的であったが，現在の「激指」[3)] ではロジスティック回帰を利用し，指し手についての様々な情報を総合的に使って確率を算出するといった手法を用いている．本論文でも指し手の選択確率の算出にはロジスティック回帰を利用した手法を用いる．

この手法では，$n$ 個の特徴が存在し，$i$（$1 \le i \le n$）番目の特徴の値を $x_i$ とすると，特徴の値 $(x_1, x_2, \ldots, x_n)$ を持つ指し手の遷移確率 $p$ は以下の式で表される．

$$p(x_1, x_2, \ldots, x_n) = \frac{1}{1 + \exp(-\sum_{i=1}^{n} w_i x_i)} \tag{1}$$

⋆1 対数の底には 2 を用いるのが一般的である．本論文でも対数の底は 2 とした．

ロジスティック回帰による実現確率探索では，プロの棋譜から各特徴が選択される割合を求める代わりに，式 (1) における各特徴の重み $w_i$ を学習により求める．学習の際には，プロの棋譜において実際に指された手を正例，指されなかった手を負例とすることで，各特徴の重み $w_i$ を推定する．

### 2.3　自動実現確率探索（ARPS）

自動実現確率探索（Automatic Realization-Probability Search，ARPS）[9] は橋本らによって提案された手法である．実現確率探索には指し手の確率を求めるために優れた棋譜を大量に要するといった問題点がある．ARPS ではこの問題を，コンピュータどうしの自動対局によって得られた棋譜を利用することにより解決している．手本となるべき棋譜のほとんどないゲームである「Lines of Actions」に適用し，通常の反復深化法に勝ち越すといった結果を得ている．

本論文中の提案手法もプログラムの探索結果を用いており，ARPS を発展させた形のアルゴリズムとなっている．

## 3. 提 案 手 法

### 3.1　実現確率探索の問題点

実現確率探索の基本的な考え方は，指し手の特徴に応じ，プロがその特徴を持つ手を選択した割合によって，探索深さを決定するというものである．この手法は多くの場合において優れた結果を得ているものの，必ずしも最善とはいえないと考えられる．

プロの棋譜を用いた実現確率探索の問題点として，探索の残り深さやプログラムの評価関数の性質によらず，確率がつねに固定であるといった点があげられる．たとえば探索の末端に近いノードで，10 手先まで読まなければ最善手と評価できない手を深く探索しても無駄になる可能性が高い．探索の残り深さが少ない局面では，単純に「駒を取る手」など直接的に得をする手の確率を高くし，逆に探索の残り深さが十分な場合には，深い読みが必要となる戦術的な手の確率も高くすることが望ましいと考えられる．また，評価関数と確率の相性が悪い場合には，（評価関数の性質上）良いと評価できない指し手を無駄に深く探索する，あるいは評価値が高い手に低い確率を割り当て再探索[⋆1]が頻繁に起こる，といった問題が生じる可能性がある．このような問題を回避するため，確率の値はプロの棋譜から求めた固定のものではなく，評価関数や探索の残り深さに応じたものにすることが望ましいと考えられる．

その他の問題点としては，ある程度探索を行い，最善手となりそうな手が分かってきた場合にも，そのような情報は確率にはまったく反映されないといった点があげられる．また，実現確率探索の性能はどのような指し手の特徴を用いるかに大きく依存するが，指し手を詳細な特徴に分類することは容易ではないといった点も問題点の 1 つといえる．

### 3.2　探索結果を利用した実現確率探索

本論文では，プログラムの探索結果を学習データとして利用した実現確率探索を提案する．プロの棋譜の代わりに，プログラムの探索結果を利用するという考え方は，ARPS（自動実現確率探索）でも用いられている．しかし，ARPS では，プロの棋譜を入手できないようなゲームに対しての適用を目的としており，単純に従来の実現確率探索を上回ることは難しいと考えられる．提案手法では，探索結果を利用し評価関数との相性を考慮するほか，

- 探索中に得られる情報を特徴として利用する
- 探索の残り深さに応じた指し手の確率を用いる
- 学習データとして探索結果とプロの棋譜を併用する

といった手法により，従来の実現確率探索を上回る性能を得ることを目的とする．具体的な手順を学習部分と探索部分に分けて述べる．

**学習部**

各 $j$（$j$ は $1 \leq j \leq n$ を満たす整数）に対し，以下の (1)，(2) を実行する．なお本論文では $n = 7$ としている．

(1)　複数の訓練局面に対して，プログラムにより深さ $j$ の全幅探索を行い最善手を得る．これを深さ $j$ の学習データと呼ぶ（一般に異なる深さの探索では，異なる最善手が得られる）．

(2)　深さ $j$ の学習データを用いて，深さに応じた各特徴 $i$ の重み $w_i^j$ をロジスティック回帰により学習する．なお各訓練局面に対し最善手を正例，それ以外の手を負例とする．

ここで深さ $j$ は，1 から $n$ まで $n$ 通りに変えるため，(1) では $n$ セットの学習データが得られ，(2) では各学習データに対応して $n$ セットの重みが得られる．

**探索部**

提案手法の探索部の特徴は，指し手の遷移確率の算出に用いる式 (1) 中の各特徴の重み $w_i$ を，探索の残り深さに応じたものにすることである．具体的には，探索中の残り深

⋆1 実現確率探索では水平線効果を防ぐため，確率の低い手が最善手となった場合，確率を高くして同じ手を再度探索する．再探索が頻繁に起きると無駄が大きくなるため，再探索が起きる割合は小さいほうが望ましい．

さ $d$（実数）が $j-1$ 以上 $j$ 未満のとき（$j$ は $1 \leq j \leq n$ の整数）には，深さ $j$ の探索結果から求めた重み $w_i^j$ を用いる．なお，$d \geq n$ の場合には，つねに最大深さの探索結果から求めた重み $w_i^n$ を用いる（以下，提案手法 1），プロの棋譜から求めた重み $w_i$ を用いる（以下，提案手法 2）の 2 通りの手法を検討する．

以下で提案手法の特徴について具体的に説明していく．3.2.1 項，3.2.2 項で述べる探索中に得られる情報の利用，探索の残り深さに応じた確率の利用は提案手法 1，提案手法 2 で共通して行う手法である．3.2.3 項で述べるプロの棋譜の併用は提案手法 2 のみで行う．なお，以降本論文中で単に提案手法と表記した場合，提案手法 1，提案手法 2 の両方に共通する特徴であることを示す．

#### 3.2.1　探索中に得られる情報の利用

提案手法では，探索中に得られる情報として (1) killer moves であるか否か，(2) history heuristic の 2 点を指し手の特徴として用いる．Killer moves は兄弟ノードの最善手，history heuristic はある手が探索中で $\beta$ カットを起こした割合を示す．これらの値はコンピュータ将棋やコンピュータチェスでは，探索中の指し手の並べ替え（move ordering）などによく用いられている．Killer moves，history heuristic はともに探索中に変化する値であり，実際に探索を行わないプロの棋譜を学習データとした通常の実現確率探索では利用できない．提案手法では，これらの特徴を用いることで，指し手の確率を探索中に動的に変化させる．

この手法を用いる利点としては，まず表面的な特徴だけでは分からない指し手の善悪を確率に反映できる点があげられる．将棋の指し手を特徴に分類することは非常に複雑な作業であり，詳細に分類するのには限界がある．Killer moves や history heuristic といった実際に探索を行ったときの値を用いることで，「王手」「駒を取る手」といった単純な特徴では表現しきれなかった指し手の善悪を確率に反映することができると考えられる．

また，通常の実現確率探索では，たとえば「歩を取る手」の確率は，他に有力な手が存在する場合，存在しない場合にかかわらずつねに固定である．しかし，同じ「歩をとる手」でも，他に有力な手が存在しない場合には重要な手と考え深く探索し，存在する場合にはそれほど重要な手とは考えず浅く探索するのが自然といえる．Killer moves，history heuristic は，他の指し手との比較に基づく特徴であるため，このような特徴を用いることで，他の指し手も考慮した確率を求めることができると考えられる．

#### 3.2.2　探索の残り深さに応じた確率の利用

提案手法では，探索の残り深さに応じた確率を用いる．探索の残り深さが少ない場面では浅い探索結果を基にした確率を利用し，逆に探索の残り深さが深い場面では深い探索結果を基にした確率を利用する．このようにすることで，

- 残り深さが多い局面では，深い探索により最善手となりそうな指し手の確率が高くなる
- 残り深さが少ない局面では，浅い探索により最善手となりそうな指し手の確率が高くなる

といったような残り深さに応じて適切な確率を用いることができると考えられる．

また，通常の実現確率探索では，残り確率から考えて意味のない手は生成しないといったことを手動で行っている．たとえば激指では，送りの手筋は，残りの確率が少ないときには生成していない[10]．これは，送りの手筋は駒をいったん損して，取り返すところまで読めなければならないため，確率が少ない局面では無駄になると考えられるためである．提案手法を用いれば，このような残り深さに応じた確率の算出・枝刈りも自動でできる可能性があると考えられる．

#### 3.2.3　プロの棋譜の併用

提案手法では指し手の確率を求めるために，プログラムの探索結果を作成しなければならない．多くの局面について探索を行う必要があるため，非常に時間のかかる処理になり，探索深さも限られたものになるという問題がある．この問題を改善するため，探索の残り深さが多い場合，学習用のデータとしてプロの棋譜を併用することを検討する．この手法は，探索深さを十分に増やした場合には，理想的にはプロの指し手とプログラムの指し手がほぼ一致する，という考えに基づくものである．

提案手法 1（プロの棋譜を併用しない）の利点としては，ARPS と同様，人間の知識がないゲームにも適用できる点があげられるのに対し，提案手法 2（プロの棋譜を併用する）の利点としては，探索の残り深さが十分な場合に高い精度の確率を得ることができると期待できる点があげられる．

## 4. 実　　装

### 4.1　実装，実験に用いたプログラム

提案手法の実装，実験には Bonanza Version 4.1.1 [⋆1]，および著者が開発中のプログラム「棋理」を用いた．Bonanza，棋理はともに全幅探索を採用したプログラムであり，それぞれアマチュアトップレベル，アマチュア三段程度の強さを持つプログラムである[⋆2]．本研究

⋆1 Bonanza はソースコードが公開されている（http://www.geocities.jp/bonanza_shogi/）．
⋆2 世界コンピュータ将棋選手権や floodgate [11] での成績，次の一手問題の正答数から推定．

では Bonanza，棋理の探索部分に通常の実現確率探索，提案手法をそれぞれ実装し，全幅探索も含めた性能の比較を行う．

なお，Bonanza，棋理では探索中で以下の枝刈りを行っている．

- Null move pruning
- Futility pruning
- LMR（Late Move Reductions）※棋理は history pruning

このうち，null move pruning，futility pruning は全幅探索，実現確率探索，提案手法のすべてで行い，LMR，history pruning については，全幅探索のみで行う．これは，LMR，history pruning は単純に実現確率探索と組み合わせた場合には効果が得られなかったためである．

### 4.2　用いた特徴，学習データ

実現確率探索の指し手の分類に用いた特徴は以下のとおりである．

- 駒の損得
- 駒を取る手
- 直前に動いた駒を取る手
- 成る手
- 逃げる手
- 当たりをかける手
- 王手
- 王手を防ぐ手（玉の移動，合駒）
- 玉との相対位置テーブルの値の増減
- 指し手の移動元（絶対位置）
- 指し手の移動先（絶対位置）
- 指し手の移動元（玉との相対位置）
- 指し手の移動先（玉との相対位置）
- 移動元の周囲 $3 \times 3$ の駒の配置のパターン
- 移動先の周囲 $3 \times 3$ の駒の配置のパターン
- Killer moves
- History heuristic

これらの特徴のうち，当たりをかける手，逃げる手，history heuristic の値は 0〜1.0 の連続値[⋆1]，駒の損得，相対位置テーブルの値の増減は −1.0〜1.0 の連続値，その他の特徴は 1，0 で表現している．

各特徴の重みは LIBLINEAR [12] を用いて求めている．LIBLINEAR とは SVM やロジスティック回帰といったデータの分類に関する計算を高速に行うライブラリである．本実験では，LIBLINEAR version 1.5 を用いた．学習の際には L2 正則化を行っている[⋆2]．

指し手の確率の算出には以下の学習データを用いた．

- プロの棋譜 1,000 局（通常の実現確率の算出に用いる）
- 上記の棋譜に現れるすべての局面について，深さ 1〜7 の全幅探索を行った結果

なお，指し手はすべて生成したうえで，特徴に分類する．確率が低い手についても知識による前向き枝刈りは行わないため全幅探索に近い形の実装になっている．また，1 手で消費する深さ（指し手の遷移確率をとり符号を反転させた値）は通常は 0〜無限大の実数となるが，本実験では 0.75〜3 に制限している．この範囲は実験的に決定したため，さらに検討の余地はあると考えられる．

## 5. 実験結果

### 5.1　実験環境

本研究の実験環境を**表 1** に示す．実験結果はすべて 1CPU で行ったものである．

自己対局は，定跡で 16 手目まで進めた局面（同一局面を除く）から開始し，先後を入れ替えて 250 セット，計 500 局の対局を行った．思考の打ち切り条件は特に断りがない限り，1 手 500,000 ノードとした．この条件ではノード数で探索を打ち切るため，思考時間は同一ではない[⋆3]．今回の実験では，相手玉を詰ますか，評価値が一定以上になった場合を勝利条件としている．勝利条件の評価値は Bonanza は 1,000 点，棋理は 1,500 点とした．また，本実験では千日手のほか，手数が 256 手を超えた場合も引き分け扱いとしている．

なお，事前の学習部分のみ**表 2** の環境を用いている．これは，LIBLINEAR が学習時に大量のメモリ（本実験では 5 GB 程度）を使用するためである．表 2 の環境において，ある探索深さ $j$ における特徴の重み $w_i^j$ を学習するのに要する時間は 20 分程度である．

⋆1 当たりをかける手，逃げる手は駒の価値を考慮しているため連続値になっている．
⋆2 学習の際に用いたパラメータは「-s 0 -B 0.05」である．
⋆3 実現確率探索の探索速度は実装に依存する部分が大きい．本実験ではアルゴリズムの有効性を検証するため，ノード数による比較を行った．

**表 1**　実験環境
Table 1　Experimental environment.

| | |
|---|---|
| CPU | Xeon X5355 2.66 GHz |
| メモリ | 2 GB |

**表 2**　実験環境（学習時）
Table 2　Experimental environment (in learning).

| | |
|---|---|
| CPU | Core i7 920 2.66 GHz |
| メモリ | 12 GB |

**表 3**　1 秒間に探索できるノード数
Table 3　Number of nodes that can be searched in one second.

| 手法 | 1 秒間に探索できるノード数（nps） | |
|---|---|---|
| | Bonanza | 棋理 |
| 全幅探索 | 149,243 | 338,868 |
| 実現確率探索 | 120,340 | 282,189 |
| 提案手法 1 | 111,624 | 280,156 |
| 提案手法 2 | 111,881 | 279,822 |

### 5.2　探 索 速 度

**表 3** に探索法ごとに 1 秒間に探索できるノード数（nps）を計測した結果を示す．なお，ここで示す nps の値は静止探索中のノードもカウントしたものである．

棋理の場合，提案手法の探索速度は，全幅探索と比較し，約 17%の速度低下となっている．ただし，指し手の特徴の分類やロジスティック回帰など時間がかかる処理が増えていることを考えると，速度低下としては小さいといえ，十分に実用的な範囲内であると考えられる．一方，Bonanza の場合の速度低下は約 25%と棋理よりも大きくなっている．これは Bonanza では，データ構造として bitboard を採用しており，利き情報を持っていないため，指し手を特徴に分類するコストが大きくなっているためだと考えられる．

ただし，本実験の実装では，特徴に分類する部分の高速化は行っていないため，実装次第では，より全幅探索に近い探索速度を出すことが可能だと考えられる．

### 5.3　探索中に得られる情報を特徴として利用した場合の効果

探索中に得られる情報（killer moves，history heuristic）を利用した場合の効果を検証した．**表 4** は探索中に得られる情報を利用した場合と利用しない場合で対局を行った結果である（*が付いているものは有意水準 5%の二項検定で有意）．学習データとしては深さ 7 の探索結果を用いている．

**表 4**　探索中に得られる情報を特徴として利用する効果
Table 4　Effect of using information obtained while searching.

| プログラム | 勝敗（勝率） |
|---|---|
| Bonanza | 289 勝 206 敗　5 分（0.583）* |
| 棋理 | 266 勝 203 敗 31 分（0.567）* |

**表 5**　図 2 における指し手の確率
Table 5　Probability of moves in Fig.2.

| オーダー | 探索開始時 | | 探索中 | |
|---|---|---|---|---|
| | 指し手 | 確率 | 指し手 | 確率 |
| 1 | ▲ 2 五歩 | 0.555 | ▲ 2 五桂 | 0.594 |
| 2 | ▲ 2 五桂 | 0.422 | ▲ 2 五歩 | 0.395 |
| 3 | ▲ 4 六銀 | 0.255 | ▲ 4 六銀 | 0.373 |
| 4 | ▲ 3 五歩 | 0.193 | ▲ 3 五歩 | 0.259 |
| 5 | ▲ 1 八飛 | 0.192 | ▲ 1 八飛 | 0.218 |
| 6 | ▲ 5 八飛 | 0.178 | ▲ 4 六歩 | 0.145 |
| 7 | ▲ 2 八飛 | 0.173 | ▲ 2 四角 | 0.142 |
| 8 | ▲ 4 八飛 | 0.166 | ▲ 1 四歩 | 0.140 |
| 9 | ▲ 1 四歩 | 0.165 | ▲ 4 八飛 | 0.118 |
| 10 | ▲ 2 四角 | 0.142 | ▲ 4 六角 | 0.102 |

実験の結果，Bonanza，棋理のいずれを用いた実験においても，探索中の情報を利用したプログラムが勝ち越した．この結果から，探索中に得られる情報を利用することは有効であると考えられる．

以下に探索中の情報を特徴として利用した場合の例を示す．**表 5** は学習データとして深さ 7 の探索結果を利用した場合の **図 2** をルート局面とした場合の Bonanza の指し手の遷移確率（上位 10 手）を示したものである．

表中の探索開始時は，探索中に得られる情報（history heuristic）[⋆1]がまったくない状態での確率を表している．また，探索中はある程度探索を行い（本実験では 200,000 ノード探索時点とした），history heuristic の情報が得られた状態での確率を表している．表中の確率が高い指し手ほど，より深く探索されることになる．

図 2 の局面では，探索開始時には▲ 2 五歩の確率が高くなっている．これは，▲ 2 五歩が頻出手であることに加え，相手の銀にあたりをかける手にもなっていることが理由と考えられる．ただし，この局面では 1 七桂の移動先がなくなってしまうため，あまり良い手とは

⋆1 ルートでは killer moves は利用していない．

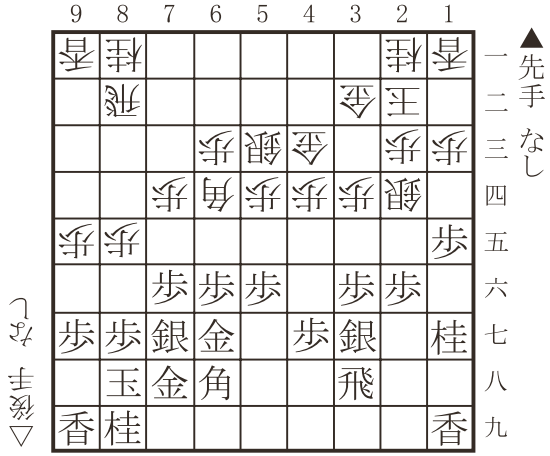


**図 2**　探索中に得られる情報を特徴として利用する効果（例）
Fig. 2　Example of using information obtained while searching.

**表 6**　残り深さに応じた確率を用いる効果
Table 6　Effect of using probability according to the remaining depth.

| プログラム | 勝敗（勝率） |
|---|---|
| Bonanza | 262 勝 233 敗　5 分（0.529） |
| 棋理 | 256 勝 210 敗 34 分（0.549）* |

いえない．提案手法の場合，探索中の確率では，表 5 から▲ 2 五歩の確率は下がり，逆にこの局面ではよりありえそうな▲ 2 五桂などの確率が高くなっていることが分かる．このように探索中の情報を利用することで，表面的な特徴だけでは表現することが難しい，状況に応じた確率を求めることが可能になると考えられる．

### 5.4　探索の残り深さに応じた確率を用いる効果

探索の残り深さに応じた確率を用いることによる効果を検証した．**表 6** は，深さ 1〜7 の探索結果を利用し，探索の残り深さに応じて確率を使い分けた場合（提案手法 1）と残り深さによらずつねに深さ 7 の探索結果を学習データとした確率を用いた場合の対局結果である（*が付いているものは有意水準 5%の二項検定で有意）．

実験の結果，Bonanza，棋理のいずれを用いた実験においても，探索の残り深さに応じた確率を用いた場合が勝ち越した．この結果から，探索の残り深さに応じた確率を用いることは有効であると考えられる．

**表 7**　図 3 における指し手の確率
Table 7　Probability of moves in Fig.3.

| オーダー | 深さ 1 の探索結果から求めた確率 | | 深さ 7 の探索結果から求めた確率 | |
|---|---|---|---|---|
| | 指し手 | 確率 | 指し手 | 確率 |
| 1 | ▲ 7 八銀 | 0.649 | ▲ 9 八香 | 0.679 |
| 2 | ▲ 5 五歩 | 0.368 | ▲ 3 六歩 | 0.241 |
| 3 | ▲ 3 六歩 | 0.333 | ▲ 4 六銀 | 0.233 |
| 4 | ▲ 7 八金 | 0.276 | ▲ 9 六歩 | 0.201 |
| 5 | ▲ 9 八香 | 0.192 | ▲ 1 六歩 | 0.177 |
| 6 | ▲ 1 六歩 | 0.156 | ▲ 5 五歩 | 0.149 |
| 7 | ▲ 4 六歩 | 0.097 | ▲ 7 八銀 | 0.137 |
| 8 | ▲ 6 八角 | 0.085 | ▲ 6 五歩 | 0.093 |
| 9 | ▲ 7 八飛 | 0.073 | ▲ 7 八金 | 0.087 |
| 10 | ▲ 8 六角 | 0.072 | ▲ 7 五歩 | 0.078 |

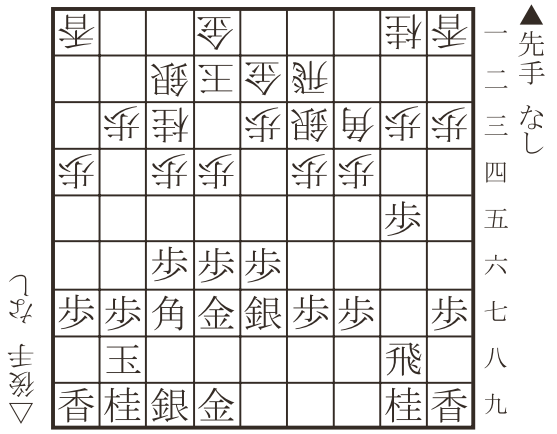


**図 3**　深さに応じた学習データを用いる効果（例）
Fig. 3　Example of using probability according to the remaining depth.

以下に探索の残り深さに応じた確率を用いた例を示す．**表 7** は提案手法を用いた場合の**図 3** の局面における指し手の確率である．深さ 1，および深さ 7 の探索結果を学習データとした場合の指し手の確率をそれぞれ上位 10 手を示している．

図 3 は居飛車対振り飛車の序盤戦である．先手はここから穴熊，美濃囲い（左美濃）の

**表 8**　プロの棋譜を学習データとして併用する効果
Table 8　Effect of using both search results and records of professional players.

| プログラム | 勝敗（勝率） |
|---|---|
| Bonanza | 258 勝 239 敗　3 分（0.519） |
| 棋理 | 232 勝 214 敗 54 分（0.520） |

どちらに進めるか，主に 2 通りに分かれる．穴熊の場合，囲いきってしまえば評価は非常に高いが手数がかかる，美濃囲いは穴熊に比べ囲いきるまでの手数が少なくて済むという特徴があるといえる．この囲いの特徴を考えると，

- 玉を囲いきる十分な深さまで探索できる場合には，美濃囲いよりも穴熊のほうが評価値が高くなる
- 残り深さが少ない場合，穴熊は囲いきるまで読みきることができず，深く探索したとしても最善手として選択されない可能性が高い

といったことが考えられ，残り深さによっては穴熊に囲う手を延長することは無駄になってしまうと考えられる．

表 7 をみると，深さ 7 の探索結果から求めた確率では穴熊を目指す▲ 9 八香の確率が高くなっており，深さ 1 の探索結果から求めた確率では美濃囲いを目指す▲ 7 八銀の確率が高くなっていることが分かる．このように，提案手法では残り深さが少ない場合には深い読みを入れなければ正しい判断ができないような指し手の確率は低くなっており，枝刈りのような効果を果たしていると考えられる．

### 5.5　プロの棋譜を併用する効果

探索の残り深さが多い場合に，プロの棋譜を併用した確率を用いる効果を検証した．**表 8** に提案手法 1（プロの棋譜を併用しない場合）との対局結果を示す（*が付いているものは有意水準 5%の二項検定で有意）．

いずれもプロの棋譜を併用した場合が勝ち越しているものの，有意な結果を得ることはできなかった．ただし，プロの棋譜を併用した確率に切り替える深さを調整することなどにより，性能をさらに改善できる余地は残されていると考えられる．また，本実験の思考時間は 1 手 500,000 ノードと比較的短い設定となっているため，残り深さが多い部分でプロの棋譜を併用した効果が現れにくかった可能性もあると考えられる．

### 5.6　自己対局による性能評価

自己対局により提案手法と全幅探索，および通常の実現確率探索との性能を比較した．**表 9** に全幅探索との対局結果，**表 10** に通常の実現確率探索との対局結果を示す（*が付いているものは有意水準 5%の二項検定で有意）．

**表 9**　全幅探索との対局結果
Table 9　Results of games with the program using the brute-force search.

| 手法 | 勝敗（勝率） | |
|---|---|---|
| | Bonanza | 棋理 |
| 実現確率探索 | 287 勝 207 敗　6 分（0.580）* | 279 勝 172 敗 49 分（0.618）* |
| 提案手法 1 | 325 勝 172 敗　3 分（0.653）* | 289 勝 164 敗 47 分（0.637）* |
| 提案手法 2 | 312 勝 178 敗 10 分（0.636）* | 291 勝 153 敗 56 分（0.655）* |

**表 10**　通常の実現確率探索との対局結果
Table 10　Results of games with the program using the realization probability search.

| 手法 | 勝敗（勝率） | |
|---|---|---|
| | Bonanza | 棋理 |
| 提案手法 1 | 290 勝 209 敗　1 分（0.581）* | 262 勝 218 敗 20 分（0.545）* |
| 提案手法 2 | 279 勝 218 敗　3 分（0.561）* | 243 勝 211 敗 46 分（0.535） |

実験結果から，全幅探索，通常の実現確率探索との対局において，提案手法 1，提案手法 2 がともに勝ち越していることが分かる．

探索結果のみを学習データとした提案手法 1 についても，従来の実現確率探索を上回る結果を得た．今回の実験では，学習データとして，深さ 7 までの探索結果しか用いていないため，より深い部分までの探索結果を利用することで，さらに性能を改善できる可能性がある．また，この結果からプロの棋譜などの優れた学習データを得ることが難しいゲームにおいても，従来の実現確率探索と同等以上の性能を得ることができる可能性があると考えられる．

## 6.　学習に用いる探索結果の深さによる勝率の変化

**図 4** に，指し手の確率の学習に用いる探索結果の深さを変えることによる勝率の変化を示す．勝率は，提案手法 1 において学習に用いる探索結果の深さを変えた場合の，通常の実現確率探索との対局結果を示している．

実験結果から，深い部分までの探索結果を用いることで勝率が向上していることが分かる．本実験では，深さ 7 までの探索結果しか用いていないが，さらに深い部分までの探索結果を利用することで性能を改善できる可能性もあると考えられる．

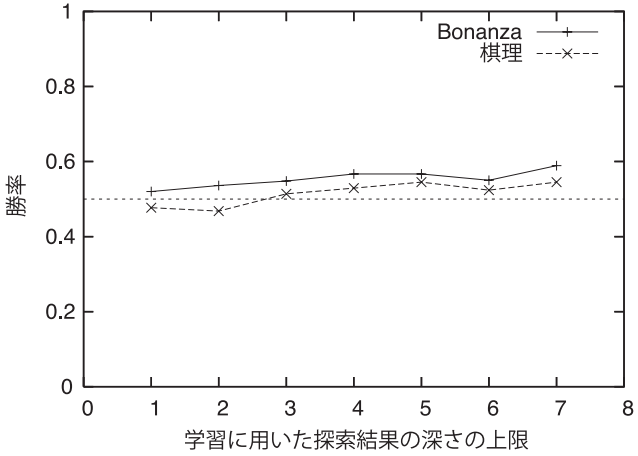


**図 4**　学習に用いる探索結果の深さを増やすことによる勝率の変化
Fig. 4　Improvement of winning ratio according to using deeper search results.

**表 11**　異なるプログラムの探索結果から求めた確率を用いた場合の対局結果
Table 11　Results of games using probability based on results of other programs.

| プログラム | 勝敗（勝率） |
|---|---|
| Bonanza | 220 勝 275 敗　5 分（0.444）* |
| 棋理 | 218 勝 244 敗 38 分（0.471） |

## 7. プログラム（評価関数）と確率の依存関係

提案手法では，プログラムの探索結果を用いて確率を算出するため，評価関数の性質によって，性能が変化することが考えられる．強いプログラムで求めた探索結果を用いた方が高い性能が得られるのか，あるいは弱くても自分のプログラムの探索結果を用いた方が高い性能が得られるのかといった点について，検証を行った．

**表 11** は提案手法 1 において，異なるプログラムから求めた確率（Bonanza の場合には棋理から求めた確率，棋理の場合には Bonanza から求めた確率）を用いた場合の，自分の探索結果から求めた確率を用いた場合（通常の提案手法 1）との対局結果である．

表 11 から，異なるプログラムから求めた確率を利用した場合，自分の探索結果を利用した場合に負け越していることが分かる．この実験結果から，プログラムによって良い性能が得られる確率には違いがあると考えられる．

## 8. 今後の課題

### 8.1 特徴の取捨選択，高速化

本論文の実験では，ノード数を固定にした場合には，提案手法が他の手法を大きく上回る勝率を得た．しかし，特に Bonanza の場合には速度低下が約 25%と大きく，そのままでは実用的には十分な性能向上が得られない可能性がある．思考時間を固定にした場合にも十分な性能が得られるよう，特徴の取捨選択や高速化などについて検討を行う必要があると考えられる．

### 8.2 優れた棋譜を得ることが難しいゲームにおける提案手法の有効性

提案手法 1 は，プロの棋譜を用いる必要がないため，ARPS と同様，優れた棋譜が得にくいゲームへの適用も可能である．本論文ではゲームの対象としてコンピュータ将棋を用いたが，それ以外の特にプロの棋譜などの優れた学習データが入手しにくいようなゲームにおける提案手法の有効性についても検証していきたいと考えている．

また，ARPS では，単純な反復深化法との比較では優れた結果を得ているが，コンピュータ将棋の全幅探索はここ数年で大きく進歩しており，現在実験を行うと異なる結果が得られることも考えられる．そういったこともふまえ，提案手法，ARPS，全幅探索の比較を行い各手法の有効性について検証したいと考えている．

### 8.3 他の探索パラメータの自動チューニング

現在のゲームプログラムの探索部分は非常に複雑で，様々な探索アルゴリズムや枝刈りの手法を組み合わせたものになっているのが一般的である．そのため，用いられている手法のそれぞれについてパラメータを調整する必要がある．探索パラメータの値は手動で，実験的に調整されることも多いが，同時に複数のパラメータを適切に制御するのは非常に困難である．

本論文では，探索結果から指し手の確率を求める手法を提案したが，そのほかに，枝刈りのマージンなどの探索中のパラメータについても調整を行うことができる可能性があると考えられる．

## 9. おわりに

本論文では，プログラムの探索結果を利用した実現確率探索を提案した．実験結果から，探索中に得られる情報を特徴として利用することや深さに応じた学習データから求めた確

率を利用するといった改良により，実現確率探索の性能を改善することができることを示した．自己対局の結果では，提案手法は全幅探索や従来の実現確率探索を上回る結果を得ることに成功し，有効な探索手法となりうることを示した．

探索手法は評価関数とならびプログラムの強さを決定する重要な要素である．評価関数が機械学習により成功を収めつつあるのに対し，探索手法はまだ比較的手動で調整している部分が多いといえる．現在，計算機のハードウェアの性能は非常に高性能になっており，コンピュータ将棋の分野でも行うことのできる手法の幅は大きく広がっている．このような背景を活かし，提案手法のような高いマシンパワーを活かしたチューニングを行うことで，探索手法の性能も大きく改善できる可能性があると考えられる．

## 参 考 文 献


1) 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).
2) Tsuruoka, Y., Yokoyama, D. and Chikayama, T.: Game-Tree Search Algorithm Based On Realization Probability, *ICGA Journal*, Vol.25, pp.145–152 (2002).
3) 鶴岡慶雅：最近のコンピュータ将棋の技術背景と激指，情報処理，Vol.49, No.8, pp.982–986 (2008).
4) 棚瀬　寧：棚瀬将棋の技術背景，情報処理，Vol.49, No.8, pp.987–992 (2008).
5) 金子知適：最近のコンピュータ将棋の技術背景と GPS 将棋，情報処理，Vol.50, No.9, pp.878–886 (2009).
6) 橋本　剛：コンピュータ将棋 TACOS のアルゴリズム，コンピュータ将棋の進歩 5，pp.33–67 (2005).
7) 竹歳正史，橋本　剛，梶原羊一郎，長嶋　淳，飯田弘之：コンピュータ将棋における実現確率探索の研究，第 7 回ゲーム・プログラミングワークショップ，pp.87–92 (2002).
8) 三輪　誠，横山大作，近山　隆：指し手の履歴の抽出に基づくカテゴリの拡張，第 11 回ゲーム・プログラミングワークショップ，pp.64–69 (2006).
9) 橋本　剛，長嶋　淳，作田　誠，Uiterwijk, J.，飯田弘之：実現確率探索のゲーム全般への応用—Lines of Action を題材にして，第 7 回ゲーム・プログラミングワークショップ，pp.81–86 (2002).
10) 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，pp.1–17 (2003).
11) コンピュータ将棋連続対局場所（floodgate）．http://wdoor.c.u-tokyo.ac.jp/shogi/floodgate.html
12) Fan, R.-E., Chang, K.-W., Hsieh, C.-J., Wang, X.-R. and Lin, C.-J.: LIBLINEAR: A Library for Large Linear Classification, *Journal of Machine Learning Research*, Vol.9, pp.1871–1874 (2008).


(平成 22 年 1 月 25 日受付)
(平成 22 年 9 月 17 日採録)

**佐藤　佳州**（正会員）

1985 年生．2008 年筑波大学第三学群情報学類卒業．2010 年同大学大学院システム情報工学研究科博士前期課程修了．現在，パナソニック株式会社先端技術研究所．人工知能に関する研究に従事．

**高橋　大介**（正会員）

1970 年生．1991 年呉工業高等専門学校電気工学科卒業．1993 年豊橋技術科学大学工学部情報工学課程卒業．1995 年同大学大学院工学研究科情報工学専攻修士課程修了．1997 年東京大学大学院理学系研究科情報科学専攻博士課程中退．同年同大学大型計算機センター助手．1999 年同大学情報基盤センター助手．2000 年埼玉大学大学院理工学研究科助手．2001 年筑波大学電子・情報工学系講師．2004 年同大学大学院システム情報工学研究科講師．2006 年同助教授，2007 年同准教授．博士（理学）．並列数値計算アルゴリズムに関する研究に従事．1998 年度情報処理学会山下記念研究賞，1998 年度，2003 年度情報処理学会論文賞各受賞．日本応用数理学会，ACM，IEEE，SIAM 各会員．