ソフトマックス戦略と実現確率による深さ制御を用いたシンプルなゲーム木探索方式

原悠一[†1]　五十嵐治一[†1]　森岡祐一　山本一将[†2]

ソフトマックス戦略に基づくシンプルな探索方式を提案し，コンピュータ将棋へ適用した実験結果を報告する．本探索方式では探索木中のノードの評価値は子ノードの評価値を選択確率で重み付けした期待値であり，再帰的に定義される．選択確率は選択先のノードの評価値を目的関数とするボルツマン分布を用いる．探索は実現確率を良さの度合いとする最良優先探索であり，深さの制御には実現確率の閾値を用いた反復深化を用いる．各ノードへの実現確率はルートノードからの選択確率の積で定義する．したがって，将棋の有効な指し手に関するヒューリスティクスは使用せず，最終的には局面評価関数だけに依存する．本発表ではこの探索方式の詳細と評価実験の結果を報告する．

キーワード：コンピュータ将棋，ソフトマックス戦略，実現確率，ボルツマン分布

A Simple Game-Tree Search Algorithm Based on Softmax Strategy and Search Depth Control Using Realization Probablity

YUICHI HARA[†1]　HARUKAZU IGARASHI[†1]　YUICHI MORIOKA
KAZUMASA YAMAMOTO[†2]

Abstract: We propose a new simple game-tree search algorithm based on a softmax strategy and report our experimental results with it after applying it to computer shogi. In this algorithm, each node's value in a search tree is defined by the expectation of the values of its child nodes. The selection probabilities of the child nodes are used as weights when calculating the expectations. A child node's selection probability is given by the Boltzmann distribution function, including the node's value as its objective function. A game tree is searched by the best-first search algorithm where the realization probability is used as the node's goodness for reaching a goal. The realization probability is also used as a threshold in iterative deepening to control a search's depth. We defined a node's realization probability as the product of the selection probabilities from the root node to the node itself. That means the realization probabilities depend on the values of the leaf nodes, not on the heuristics based on the statistics of moves in shogi as in the traditional research. Only a state evaluation function is necessary for calculating the realization probabilities. The details of our new search algorithm and experimental results are shown in this short report.

Keywords: Computer shogi, Softmax strategy, Realization probability, Boltzmann distribution function

1　はじめに

近年，コンピュータ将棋の棋力向上がめざましい．プロ棋士との対戦でも大きく勝ち越すようになってきた．この要因の一つとして探索技術の進化が挙げられる．コンピュータ将棋の探索法は，コンピュータチェスの影響を受けて選択探索から全幅探索へと移り変わってきている．選択探索，全幅探索どちらも Min-Max 戦略をベースとしているが，前者は探索木中で展開するノードをそのゲームに特有な知識を用いて制限し，より深く読むことを目標としている．一方，後者は，基本的には全ノードにより探索漏れをなくすることを目標としているが，各種の枝刈りを用いてαβ法の効率化を図っている．現在は，この 2 つの探索法をミックスし，さらにマルチスレッドやマルチコアを利用した並列化処理の枠組みに乗せて高速化を図るのが大きな流れと言える．

しかし，この流れを振り返ってみると，様々な仮定・近似・予測に基づく枝刈りが複数組み合わされており，かなり処理が複雑化している．その上，将棋に特有なヒューリスティクスも探索アルゴリズムに相当埋め込まれてしまっている場合もあり，将棋以外の他のゲーム木探索に適用できるかという汎用性の点でも問題があるように思われる．

たとえば，選択探索の中でも注目を集めた探索手法の一つとして，実現確率による探索打ち切りアルゴリズム[1]（以下，実現確率探索）がある．この手法はあり得そうな指し手を優先してより深く探索するという特徴を持っているが，実現確率の計算に将棋特有の知識を用いている．すなわち，指し手を特徴量により表現し，プロ棋士の棋譜データ集から学習した特徴量の重みを用いてその手が選択される確率（“激指”の遷移確率に対応）を計算し，その積により実現確率を計算している．この特徴量は局面評価関数とは別の関数であり，将棋の知識を用いて設計されている．

2　提案する探索方式の基本方針

極力，シンプルな探索方式とする．すなわち，αβ法による後向き枝刈りや，全幅探索で行われているような後向き枝刈りを予測・期待する ProbCut[2] や Null Move

†1 芝浦工業大学工学部情報工学科
Shibaura Institute of Technology
†2 (株) コスモ・ウェブ
Cosmoweb Co., Ltd.

Pruning[3]などの枝刈り手法は基本的に行わない．さらに，選択探索で行われている将棋の知識に基づく前向き枝刈りも行わない．これによって，処理の並列化が容易になると期待している．

次に，探索は実現確率を良さの度合いとする最良優先探索法を採用する．ただし，“激指”で行われているような指し手の特徴量は用いない．局面評価関数だけを用いる．局面評価関数は，どのゲームでもそれほど無理なく設計や学習ができると思われるので，これを用いても探索法としての汎用性は失われないと考えられる．また，探索深さの制御には，読みの深さではなく実現確率の値による探索打ち切りを用いた反復深化法を用いる．各ノードへの実現確率はルートノードからの選択確率の積で定義する．

さらに，ノードの選択確率は，選択先の子ノードの評価値を目的関数とするボルツマン分布を用いる（Softmax 戦略）．ノードの評価値は末端ノード（leaf）ではそのノードの局面評価値であり，末端でなければその子ノードの評価値を選択確率で重み付けした期待値である．したがって，ノード単位にそのノードの評価値計算を個々に行えばよく，粒度が小さい並列度の高い並列化が期待できる．

3　探索方式の詳細

3.1　Bolzmann 分布を利用した指し手の選択確率

Softmax 戦略に基づいて指し手の選択確率の算出には Bolzmann 分布と呼ばれる確率分布関数を利用する[4].

$$\pi(a|s) = exp(E_a(a;s)/T)/Z \quad (1)$$

$$Z \equiv \sum_{x\in A(s)} exp(E_a(x;s)/T) \quad (2)$$

$E_a(a;s)$は局面 s における指し手 a の評価値，$A(s)$は局面 s における合法手の集合，T は温度パラメータである．ここで，指し手 a の評価値を指した後の局面ノード $v = v(a;s)$ の“ノード評価値” $E_s(v)$で置き換える．すなわち，

$$\pi(a|s) = exp(E_s(v(a;s))/T)/Z \quad (3)$$

$$Z \equiv \sum_{x\in A(s)} exp(E_s(v(x;s))/T) \quad (4)$$

とする．ただし，一般にノード評価値$E_s(s)$はsが末端局面(leaf)でなければ次のように再帰的に定義されている．

$$E_s(s) = \sum_{x\in A(s)} \pi(x|s)\, E_s(v(x;s)) \quad (5)$$

もし，s が末端ノードであれば局面評価関数$E_s^{end}(s;\omega)$が呼ばれる．ωは局面評価関数中の特徴量の重みである．したがって，(5)で定義されたノード評価値と，通常の局面評価関数$E_s^{end}(s;\omega)$による局面自身の静的評価値とは異なることに注意する．

3.2　実現確率による探索打ち切りアルゴリズム

実現確率探索では読みの深さではなく，局面の実現確率の値を探索の打ち切り条件とする．局面の実現確率の値は以下の式 によって再帰的に定義され，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）＝
（直前の局面の実現確率）×（選択確率）　(6)

ただし，ルート局面の実現確率は 1.0 とする．このように，実現確率を探索の閾値とすることで，実際にありえそうな展開だけを重点的に深く探索できる点が実現確率探索の大きな特徴である．実現確率探索の例を図 1 に示す．

なお，実際に探索を行う際は，扱いやすい範囲の数値とするために実現確率は対数をとった形で扱う．すなわち，局面の実現確率は指し手の選択確率の積のため，対数をとることで乗算を加算に変換する．たとえば，実現確率が 0.25 の局面において，選択確率が 0.50 の指し手を指した場合，子局面の実現確率は0.25×0.50となる．この場合は，$(-\log_2 0.25) + (-\log_2 0.50)$へと変換される．実際の探索中では，実現確率の対数をとり符号を反転させた値が閾値 θ に達した時点で探索を打ち切る．

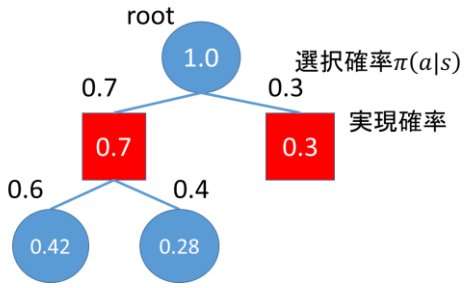


図 1 実現確率探索の例

Figure 1　Examples of realization probabilities.

3.3　選択確率と実現確率の計算

深さ d の局面s^dにおいて，指し手a^dにより生成される局面をs^{d+1}とする（図 2）．(5)より，深さ d におけるノード評価値$E_s(s^d)$は，

$$E_s(s^d) = \sum_{a^d} \pi(a^d|s^d) \cdot E_s(s^{d+1}) \quad (7)$$

と書ける．したがって，(3),(4)より$E_s(s^{d+1})$が求まるたびに選択確率$\pi(a^d|s^d)$は更新される．また，ノードの“実現確率”はルートノードからそのノードまでの経路上における選択確率の積で定義する．

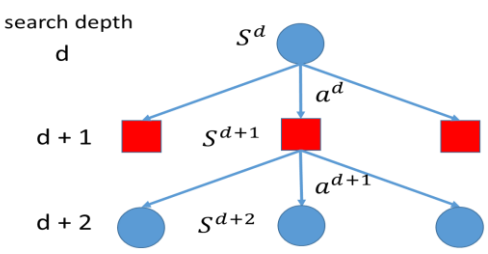


図 2　ノードと指し手の関係（●ノードが自手番）

Figure 2　Nodes and moves.

3.4 実現確率の閾値を用いた反復深化

探索には実現確率の値をノードの良さとする最良優先探索を採用する．したがって，最良のノードを優先して展開する．ただし，実現確率に閾値を設定し，その閾値を段階的に減らす反復深化を行う．すなわち，実現確率の値（%）が閾値以下となれば探索木の展開を中断し，バックトラックを行う．ただし，実現確率の閾値は 3.2 で述べたように対数をとって符号を反転させた値θを用いる．たとえば，閾値θ_1を$-\log_2 0.1$（実現確率:10%），閾値θ_2を$-\log_2 0.01$（実現確率:1%）とした場合，図 2 のように探索木の展開を行う．

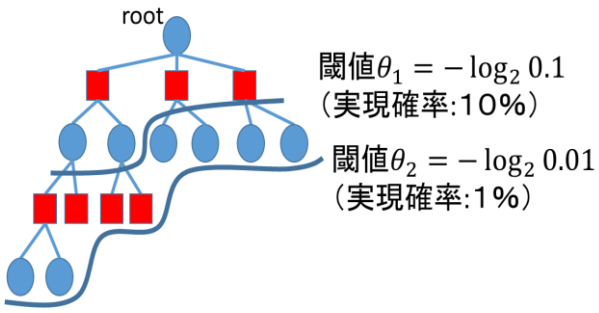


図 3　実現確率の閾値を用いた反復深化

Figure 3　Iterative deepening with thresholds of the realization probabilities.

この探索手法の特徴として，温度パラメータ T を下げると深さ優先探索に近づき，T を上げると幅優先探索に近づく．したがって，有力な手が少数に絞られる場合には T を下げて深く読み進み，逆に有力手が多い難解な局面では T を上げて読み抜けを防ぐなどの処理が可能である．

4　P-GPP 方式の並列処理の実装について

4.1 P-GPP における兄弟局面間での順位と実現確率

“GPS 将棋”[5]の非共有メモリ環境での並列化手法である P-GPP[6][7]（pipeline-game position parallerization）を参考に並列処理の方法を考える．P-GPP は廉価なネットワークで接続されたグリッドコンピューティング環境を前提としたゲーム木探索の並列化手法である．マスタ計算機は内部にゲーム木を保持し，その葉にワーカ計算機を対応付けることで分担局面を決定し，非同期に探索を行う．この際，実現する確率の高い局面を優先的にゲーム木に追加することで効率的な分担ができる．

P-GPP は非共有メモリ環境での並列化手法であるが，今回の並列化の実装は共有メモリ環境で行った．そのため，ワーカ計算機に特定局面を分担させるのではなく，スレッドに特定局面を分担させる．

また，P-GPP では各着手の選択確率をマスタゲーム木における兄弟ノード間での評価値の順位によって定めている．評価値の順位と選択確率との関係は，ワーカプログラムによって熟練者の既存棋譜中の局面をあらかじめ探索し，熟練者の着手をワーカが第何位の手と評価したかの統計を取ることで算出する[6]．しかし，本提案手法では Bolzmann 分布と呼ばれる確率分布関数を利用している．P-GPP では選択確率の算出に評価値の兄弟ノード間の順位のみを利用しているにすぎないが，本提案手法では兄弟ノードの評価値のより詳細な大小関係を考慮して選択確率の値に反映させている．また，既存棋譜などの将棋の専門知識を用いる必要がないという利点もある．

4.2 スレッドの割り当て

提案手法ではゲーム木を構築し，ゲーム木の持つ各ノードに対しスレッドを割り当てる．各スレッドは非同期に探索木を展開し，割り当てられたノードの評価値を求める．ノードにスレッドを割り当てたゲーム木の例を図 4 に示す．図 4 の例では，6 つのノードそれぞれに 1 つずつスレッドが割り当てられている．”others”と書かれているノードは，すでに分担されている手以外の全ての兄弟ノードである．

両プレイヤーの着手により盤面が変化したとき，着手後の局面をルートノードとする部分木に含まれないノードは，もう探索する必要のない局面である．そこで，部分木に含まれないノードに割り当てられたスレッドを回収し，新たに展開されたノードへ再割り当てを行う．このように，部分木を壊さずに保存するというアイディアは“optimistic pondering”と“tree pipeline”という手法としてチェスで提案されている[8]．この一例を図 5 に示す．図 5a において手 a_1 が指された場合のゲーム木を図 5b に示す．スレッド C のノードが新たなルートノードとなる．スレッド D・E・F は探索を継続する必要がなくなるので，手 a_1 以下の部分木のノードに再割り当てされる．

また，P-GPP 方式の並列化効果は，現局面よりも先の局面をゲーム木に追加し，予め探索しておくことから得られる．したがって，追加した局面が実際に指される可能性が

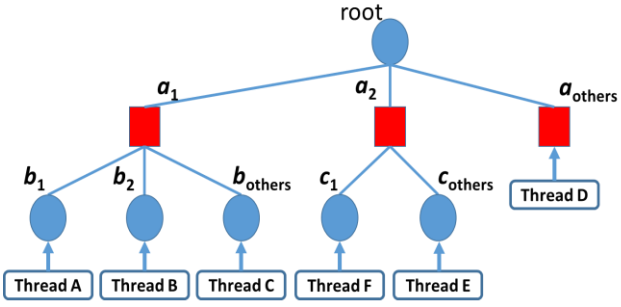
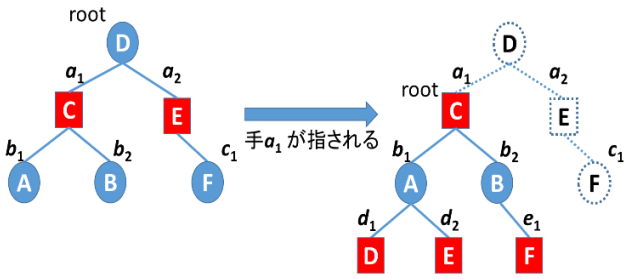


図 4　スレッドを割り当てたゲーム木

Figure 4　A serach tree where threads are assigned to nodes.



(a) (b)

図 5　ゲーム木が展開される様子

Figure 5　Expansion of a game tree.

高いほど効果が高い．そこで，ノードへスレッドを割り当てる際には，ノードを実現確率により評価し，実現確率の合計が最大となるようにスレッドを割り当てる．すなわち，ノード集合をVとし，ノード$v \in V$に到達する確率(vの実現確率)をp_vとすると，ノードの実現確率の合計値$\sum_{v \in V} p_v$ が最大になるようにスレッドを割り当てる方式が考えられる．

5　実験

提案した探索手法を実装し，従来の Min-Max 戦略に基づくコンピュータ将棋プログラムとの対局実験を行った．ただし，提案手法に基づくプログラムには，静的局面評価関数として Bonanza6.0 の評価関数を使用し，末端ノードでの局面評価には静止探索を用いた．ただし，今回は本プログラムも対局相手も並列処理の実装は行っていない．実験条件を下記に示す．

【実験条件】

・対局相手：“芝浦将棋 Jr”．ただし，2015 年第 3 回将棋電王トーナメント出場時のバージョン(28 チーム中 16 位)．詳細は文献[9]を参照のこと．
・温度パラメータ：T=60，80，100 と固定．
・総対局数：300 局．先後入替え．
・思考時間：1 手 10 秒に固定．
・閾値の制御：閾値 θ を 0.1 から 0.1 ずつ増加．
・Ponder：なし．

対局実験の結果を表 1 に示す．なお，勝率は引き分けを除いて計算した値である．

表 1　対局結果

Table 1 Results of experiments.

T	勝	負	分	勝率
60	106	188	6	36.1%
80	127	167	6	43.2%
100	94	200	6	32.0%

温度パラメータ T=80 のときの勝率が最も高かった．これは，T=60 のときは T=80 のときより深さ優先探索に近づき，狭く深い探索を行ったため，読み抜けが発生して勝率が下がったと考えられる．逆に，T=100 のときは T=80 のときより幅優先探索に近づき，広く浅い探索を行ったため，深い探索が行えず，勝率が下がったと考えられる．

6　まとめ

本研究では，ソフトマックス戦略に基づく実現確率を用いたシンプルな探索方式を提案した．提案手法を実装したプログラムと従来の Min-Max 戦略に基づくコンピュータ将棋プログラムとの対局実験を行った結果，勝ち越すことはできなかった．しかし，Min-Max 戦略で行われている枝刈りなどの高速化手法を全く取り入れていない割には高い勝率を得ている．

今後は，探索中に出現する局面の状態に応じて温度パラメータの調整を行うことにより，生成される探索木の制御を試みる．また，本論文で述べた並列処理方式を実際に実装し，評価実験を行いたい．さらに，ソフトマックス戦略下における局面評価関数の学習についても研究を進めていく予定である．

参考文献

[1] 鶴岡慶雄，横山大作，丸山孝志，近山隆，“局面の実現確率に基づくゲーム木探索アルゴリズム”，第 6 回ゲーム・プログラミングワークショップ，2001, p. 17-24,2001.
[2] Buro, M.. ProbCut: An Effective Selective Extension of the Alpha-Beta Algorithm. ICCA JOURNAL, vol. 18, no. 2, 1995, p. 71-76.
[3] Beal, D. F.. Experiments with the Null move. Advances in Computer Chess 5, 1999, p. 65-79.
[4] 五十嵐治一，森岡祐一，山本一将．方策勾配法による局面評価関数とシミュレーション方策の学習．情報処理学会研究報告，2013, vol. 2013-GI-30, no. 6, p.1-8.
[5] 金子知適，田中哲朗．多数の計算機を活用したゲーム木探索技術の進歩”．情報処理，2013, vol.54, no.9, p.914-922.
[6] Yokoyama, S., Kaneko, T., Tanaka, T. Parameter-Free Tree Style Pipeline in Asynchronous Parallel Game-Tree Search. Advances in Computer Games. Volume 9525 of the series Lecture Notes in Computer Science, 2015, p. 210-222.
[7] 横山秀，金子知適．評価値を用いて展開制御したゲーム木に基づく並列探索．　第 20 回ゲーム・プログラミングワークショップ，　2015, p. 46-53.
[8] Himstedt, K.. Gridchess: Combining Optimistic Pondering with the Young Brothers Wait Concept. ICGA Journal, 2012, vol.35, no.2, p. 67-79.
[9] 和田悠介他．「芝浦将棋 Jr.」とは．http://denou.jp/tournament2015/img/PR/ShibauraShogiJr.pdf，(参照 2016-09-22)．