2009－GI－21 (8)
2009／3／9

αβ探索における探索順序の自動学習

小幡 拓弥　　伊藤 毅志
電気通信大学 情報工学科

概要

αβ法によるゲーム木探索では、ノードの探索順序が探索効率を大きく左右する。そのため、効率の良い探索を行うためには、適切なムーブオーダリングが不可欠である。しかし、巨大で複雑なゲーム木に対し、あらゆる局面で最適なムーブオーダリングを行うパラメータを人手で設定することは非常に難しい。そこで本研究では、Bonanza Method[1]を応用した手法を用いて、そのパラメータを機械学習で最適化することを目指した。その結果、従来の手法であるSEEと比較して、概ね効率的なムーブオーダリングを行う関数を得ることができた。

Machine learning of move orderng in alpha-beta search

Abstract

In game tree search by using alpha-beta pruning, order of searching nodes certainly influences search efficiency. Therefore, proper move ordering is required to perform efficient search. However, it is very difficult to set up parameters of proper move ordering in huge and complex game tree by human power. In this research, we tried to optimize the parameters by using Bonanza method applied machine learning. As the result, we acquired the reasonable parameters to perform efficient move ordering as compared with traditional SEE method.

1. はじめに

将棋やチェスのような二人零和有限確定完全情報ゲームをコンピュータにプレイさせる最も基本的なアルゴリズムとして、αβ法がある。αβ法によるゲーム木探索では、探索するノードの順序が計算効率を大きく左右する。具体的には、最も良い順序で探索を行った場合に調べるノード数は、最も悪い順序で探索を行ったときのノード数の平方根に近い数で済む。そのため、効率の良い探索を行うためには、適切にムーブオーダリングを行う必要がある。

ムーブオーダリングの手法としては、代表的なものにSEE[2]や反復深化法[2]などがある。静的評価関数や探索アルゴリズムとの兼ね合いがある

ため、最適なムーブオーダリングの方法はプログラムごとに異なる。そのため、開発者は様々な手法を組み合わせるなどして、プログラムに合わせたムーブオーダリングの方法を設計している。

しかし、巨大で複雑なゲーム木に対し、あらゆるノードで最適なムーブオーダリングを行うパラメータを人手で設定することは、非常に難しい。そこで本研究では、Bonanza Method を応用した手法を用いて、このパラメータを機械学習で最適化することを目指した。本研究では、5 五将棋を題材として扱う。

2. 5 五将棋とは

5 五将棋は、通常より小さい 5×5 の盤を用いて行う将棋で、ミニ将棋とも呼ばれる。使用する駒は王、飛車、角、金、銀、歩が各 2 枚ずつで、下の図 1.のような初期配置で対局を開始する。駒が成れる升は、先後それぞれから見て一番奥の一段である。5 五将棋では千日手の扱いが本将棋とは異なり、千日手が成立した場合は先手の負けとなる。ただし連続王手による千日手は、本将棋と同じく王手をかけた側の負けになる。その他のルールは本将棋と同じである。

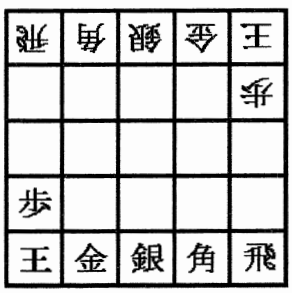

図.1 5 五将棋の開始局面

3. 効率の良い探索順序

αβ探索では、探索の結果が最も良い子ノードを最初に展開するようにして探索を行うのが、最も効率が良い。しかし、探索を行う前に探索結果を知ることはできないので、実際には探索結果を予測しながら次に展開するノードを決定する。この予測の精度が良いほどαβ枝刈りが効率的に発生し、探索量を削減することができる。

4. オーダリング関数

本研究では、次のようにして探索順序を決定する。探索中の各ノードで、全ての子ノードをある関数によって評価する。この関数は探索結果の良し悪しを予測する関数である。本稿ではこれをオーダリング関数と呼ぶことにする。オーダリング関数による評価値が高い子ノードから順に探索する。

オーダリング関数は、次の特徴から局面を評価する。

- 駒の価値
- 持ち駒の枚数による補正
- 駒の位置
- 先手玉に対する駒の位置
- 後手玉に対する駒の位置

これらの特徴にはそれぞれ点数がついており、ある局面に現れる特徴について、その点数を合計したものがその局面の評価値となる。この各特徴の点数を、学習によって最適化するのが本研究の目的である。

5. 学習手法

学習には、Bonanza Method を応用した手法を用いた。Bonanza Method は、2006 年に保木氏によって提案された、静的評価関数を学習させるための手法である。深さ 1 手+静止探索の探索によって得られる手が、エキスパートの棋譜の手と一致するように学習を行う。

Bonanza Method では、静的評価関数を学習するという将棋の問題を、数学的な問題に置き換え

て解く。保木氏の論文[1]を参考に、その手法を説明する。

まず、エキスパートの指し手との一致度を測る目的関数を設計する。これを最小にするベクトル v を求める。

$$J(P_0,\ldots,P_{N-1},v)=$$

$$\sum_{i=0}^{N-1} l(P_i,v)+\lambda[M_1(v)-M_0]+wM_2(v) \quad (1)$$

ここで、P_i は教師データ中に現れる局面、v は静的評価関数のパラメータで L 次元ベクトルである。右辺の第 2 項、第 3 項は拘束条件とペナルティで、不適切な極小点を排除したり、オーバーフィッティングを回避する働きがある。

$l(P,v)$ は次のように表される。

$$l(P,v)=\sum_{m=1}^{M} T[\xi(p_m,v)-\xi(p_{m=0},v) \quad (2)$$

p_m は局面 P を合法手 m によって進めた局面で、教師データ中で指された手を $m=0$ とする。$\xi(p_m,v)$ は、p_m の探索の結果としての評価値を表す。$T(x)$ は評価値の差を指し手の一致度に変換する関数である。

目的関数の最小化は、$J(P_0,\ldots,P_{N-1},v)$ の勾配ベクトルを用いて行う。$J(P_0,\ldots,P_{N-1},v)$ の勾配ベクトル $\Delta J(P_0,\ldots,P_{N-1},v)$ は、次のように表される。

$$\Delta_v J(P_0,\ldots,P_{N-1},v)=$$

$$\sum_{i=0}^{N-1}\sum_{m=1}^{M-1}\frac{dT(x)}{dx}\Delta_v[f(p_{i,m}^{leaf},v)-f(p_{i,m=0}^{leaf},v)]$$

$$+\lambda\Delta_v M_1(v)+w\Delta_v M_2(v) \quad (3)$$

ここで、探索の結果としての最善応手列が v 近傍で単一と仮定し、$\Delta_v\xi(p_{i,m},v)=\Delta_v f(p_{i,m}^{leaf},v)$ とする。ただし、$f(P,v)$ は局面 P の静的評価関数による評価値で、$p_{i,m}^{leaf}$ は局面 $p_{i,m}$ を最善応手の末端まで進めた局面である。この勾配ベクトルを用いて、次のように v を更新する。

$$v_l^{new}=v_l^{old}-h\,sign[\frac{\delta J(P_0,\ldots,P_{N-1},v)}{\delta v_l}] \quad (4)$$

h は 1 回の更新での v の変化量、$sign(x)$ は x の符号を返す関数である。

オーダリング関数の学習に Bonanza Method を応用した理由は、2 つある。1 つは、オーダリング関数と静的評価関数は非常に似ているということである。局面に対し評価値を与えるという点で同じであるため、静的評価関数のパラメータを学習するのと同様に、オーダリング関数のパラメータも学習できると考えた。もう 1 つは、兄弟ノードの比較によって学習を行うという Bonanza Method の性質のためである。探索順序を決定することは、兄弟ノードを探索結果の良さそうな順に並べることであるため、この性質はオーダリング関数の学習に上手く働くと考えた。

本研究で用いた手法は、本来の Bonanza Method と 2 つの点で異なる。1 つは、用いる教師データがエキスパートの棋譜ではなく、プログラム自身による探索結果であるということである。Bonanza Method ではエキスパートの指し手を正解の手とみなすが、オーダリング関数は探索結果を予測するものであるため、プログラム自身の探索結果が正解の手となる。もう 1 つ異なる点は、Bonanza Method では深さ 1 手+静止探索が

教師データの手に一致するように学習を行うが、本研究の手法では一手読みで一致するように学習をするという点である。つまり、式(2)の $\xi(p_{i,m}, v)$ を探索の評価値ではなく、局面の静的な評価値に変更する。これは、オーダリング関数では探索を利用できないためである。

6. 学習結果と評価

教師データとして、深さ 5 手+静止探索の自己対戦によって収集した棋譜から重複を排除した 4988 局を用いた。これらの棋譜中に登場する局面のうち、投了局面を除いた全局面が教師データである。全て自己対戦の棋譜であるため、棋譜中の指し手は、そのプログラム自身による探索結果ということになる。

学習の結果得られた関数を実際の探索で評価した結果が、表 1.である。第 2 回 UEC 杯 5 五将棋大会の棋譜 24 局に登場する全局面(886 局面)に対し、深さ 11 手+静止探索で探索を行わせ、SEE を用いた場合と比較した。

	SEE	学習後	割合
時間[sec]	87,684.8	59,076.1	67.4%
ノード[万 N]	12,448,675	9,703,356	77.9%

表 1. 学習した関数の SEE との比較

表 2.表 3.は、学習効果の偏りを調べるため、対局開始からの手数毎に分けて集計したものである。

	局面数	探索時間[sec]		割合
		SEE	学習後	(学習後/SEE)
全局面	886	87684.8	59076.1	67.4%
1 手目～10 手目	240	8977.8	4384.4	48.8%
11 手目～20 手目	227	11513.8	9239.0	80.2%
21 手目～30 手目	190	28223.0	15527.4	55.0%
31 手目～40 手目	117	18377.7	13768.5	74.9%
41 手目～50 手目	78	16442.9	10335.8	62.9%
51 手目～	34	4146.56	5821.11	140.4%

表 2.対局開始からの手数毎の探索時間

	局面数	探索ノード数[万 Nodes]		割合
		SEE	学習後	(学習後/SEE)
全局面	886	12,448,675	9,703,356	77.9%
1 手目～10 手目	240	2,598,795	740,394	28.5%
11 手目～20 手目	227	2,483,516	1,506,968	60.7%
21 手目～30 手目	190	2,256,835	1,873,533	83.0%
31 手目～40 手目	117	2,231,546	2,405,636	107.8%
41 手目～50 手目	78	1,963,992	1,712,244	87.2%
51 手目～	34	1,297,293	1,081,279	83.3%

表 3. 対局開始からの手数毎の探索ノード数

7. 考察

評価の結果から、学習した関数は SEE と比べて概ね効率的な探索を行えるといえる。手数毎にみていくと、特に序盤に効率化の効果が大きい。考えられる理由としては、開始局面が固定という将棋のルール上、序盤の局面のバリエーションが少ないことが挙げられる。教師データに似たような局面が多く登場するため、それらの局面が重点的に学習されている可能性がある。また、実際の探索においても教師データと同じ、または似た局面が多く登場するため、中終盤に比べて学習の効果が発揮されていたものと思われる。序盤に効率化の効果が大きいのは単に序盤に対してオーバーフィッティングされたためなのか、それとも中終盤の教師データも十分な量を用意すれば序盤の効率を落とさずに中終盤も効率化できるのか、今後の調査で明らかにしていきたい。

表 2.と表 3.を見比べると、探索時間の効率化の度合いと探索ノード数の効率化の度合いが一定でないことがわかる。これはオーダリング関数の実装に依存する問題である。実験に用いたプログラムでは、静的評価関数の呼び出しにかかる計算コストは局面に依らずほぼ一定である。一方で、オーダリング関数の呼び出しにかかる計算コストは、局面に依存して変動する。そのため、静的評価関数を用いてムーブオーダリングを行う SEE を用いた場合と比べて、学習したオーダリング関数を用いた場合は単位時間当たりの探索ノード数にばらつきがある。

今回の学習で教師データに用いたのは、深さ 5 手+静止探索の自己対戦の棋譜である。つまり、深さ 5 手＋静止探索の探索結果を予測するオーダリング関数を学習したことになる。これでは、あらゆる局面に対し最適なムーブオーダリングを行うことはできない。ムーブオーダリングを行うノードの深さに応じたオーダリング関数を用意するという方法が上手く働くかどうか、今後調査していくつもりである。また、序、中、終盤などの進行度に応じて使い分けるということも考えている。

教師データの収集には、様々な方法が考えられる。今回は自己対戦の棋譜を用いたが、自己対戦の棋譜では、登場する局面がおそらく偏っている。より多様な局面に対応するためには、教師データも多様な局面を探索したものである必要がある。例えば、ランダムに生成した局面を探索させ、そ

れを教師データとするという方法が考えられる。しかし、実際の対局中に現れる可能性の低いものを教師データに含めるべきか否かは、まだ調査を要する問題である。教師データの収集方法についての調査も、今後の課題である。

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] 小谷善行： “コンピュータ将棋の頭脳” (2007)