

遺伝的プログラミングを用いたゲームの局面評価関数の生成

Generation of an Evaluation Function for a Game
using Genetic Programming 22-03

堀之内 剛史

河口 信夫

稲垣 康善

Tsu Yoshi Horinouchi

Nobuo Kawaguchi

Yasuyoshi Inagaki

名古屋大学工学部情報工学科

Dept. of Information Eng., Nagoya Univ.

Abstract: In a game playing program they usually make use of the game tree search based on an evaluation function for selecting a move. Since it is difficult to get the function well tuned to the game, they have taken some heuristic approaches. Genetic Programming(GP) provides a way to genetically breed various functions. In this paper, we propose a GP approach to this problem. As the examples base, we take the set of pairs of the game's positions, one of which is known to be superior to the other. Our approach is applied to the game, called tic-tac-toe, successfully.

1 はじめに

思考型ゲームのプログラミングにおいて、指手をゲーム木探索によって決定するのは一般的な手法である。この場合、探索の末端の局面で、読みを行わずに局面の形勢を数値化する評価関数を用いる。強いプログラムを得るには、適切な評価関数が望まれるが、対象がある程度複雑なゲームの場合、解析的に適切な評価関数を求めることは計算量的にも困難であり、発見的な手法に頼らざるをえない。

一方、確率的探索手法の一つである遺伝的プログラミング (Genetic Programming, GP) を用いて、様々な関数生成が試みられている [1]。また [2] では、グラフ配置問題において、ユーザが美しいと感じるグラフと美しくないと感じるグラフからなる事例の集合から、遺伝的プログラミングによりユーザの嗜好を反映した評価関数の生成を試みている。

本稿では、遺伝的プログラミングを用いて、ゲームの局面評価関数を事例から生成する手法を提案し、この手法を tic-tac-toe と呼ばれるゲームに適用した実験結果を報告する。この手法で用いる事例は、相対的に優劣のついた対象ゲームの局面の組である。一般的に、人は二つの対象を比較する能力に優れている点に注目する。

まず 2 章では、準備として、遺伝的プログラミングとゲーム木探索の基本的概念について述べる。3 章では、評価関数生成手法の枠組みについて述べて、tic-tac-toe ゲームにその手法を適用する。4 章では、実験結果を示

し、獲得した評価関数に対する解析を試みる。

2 準備

2.1 遺伝的プログラミング

GP は遺伝的アルゴリズム (Genetic Algorithms, GA) をプログラム生成に適用したものである [1]。ランダムに生成されたプログラム群 (母集団) に対して、適応度関数をもとに、選択淘汰、遺伝的操作 (交叉、突然変異など) による世代進化を行わせることによって、最終的に目的のプログラムを得るという手法である。

GP で生成するプログラムの形式は、取り扱い易さの点から Lisp の S 式などを選ぶ。S 式を構成する関数記号、アトム、そして母集団中のプログラムが目的のプログラムにどれほど近いかを表す適応度関数を定め、GA の手法を用いて、関数を生成する。

2.2 ゲーム木探索の基本的概念

二人零和有限完全情報確定ゲームでは、すべての可能性を探索し尽くすことによって最善手を選択できる。しかし、対象が複雑なゲームになるとそれが困難であるので、ゲーム木を作成して、最善手を探索する。この探索の拠り所は、ゲーム木の末端の局面に対する評価値であり、局面の評価関数により定まる。一般的に、評価関数は、局面から得られる様々な情報 (以下、局面パラメータと呼ぶ) から形勢を判断して、先手が有利な局面であ

表 1: GP の実行パラメータ

母集団サイズ	300
終了世代数	100
初期母集団中の S 式の最大深さ	5
世代 1 以降の S 式の最大深さ	10
突然変異で生成する部分木の最大深さ	3
交叉による生成割合	0.6
突然変異による生成割合	0.3
選択のみによる生成割合	0.1
事例の数	150

本手法では、多くの事例を正しく評価できる関数であるほど、より適切な評価関数であると仮定し、GP システムは、与えた事例ベース中のできる限り多くの事例(無論、すべての事例に対して正しく評価できることが望ましい)を正しく評価する関数を生成する。

二人零和有限完全情報確定ゲームにおいて、任意の可能な局面は次のいずれかの集合に属する。

- S_1 : 先手勝ちの局面の集合
- S_2 : その後の指し手で先手が最善を尽くせば必ず先手が勝つ局面の集合
- S_3 : お互いに最善を尽くせば引き分けとなる局面の集合
- S_4 : その後の指し手で後手が最善を尽くせば必ず後手が勝つ局面の集合
- S_5 : 後手勝ちの局面の集合

tic-tac-toe においては、局面の対称性を考慮しない場合、5478 の局面が考えられ、 S_1 から S_5 に属する局面の数は、それぞれ 626, 2310, 1068, 1158, 316 である。

3.2 節において定義した評価関数 f の構造から、 S_1 , S_5 に属する局面に関しては、それぞれ十分大きな値、十分小さな値が返される。したがって、GP で生成する関数 g は、 S_2 から S_4 に属する局面のみ正しく評価すればよい、事例として用いる局面の組 (φ, ψ) は、

$$\varphi \in S_i, \psi \in S_{i+1} \quad (2 \leq i \leq 3)$$

を満たすものをランダムに選択する。

3.4 適応度関数

母集団中のプログラムが、目的のプログラムにどれほど近いかを表す値が適応度である。適応度関数は、多くの事例に対して正しく判断するプログラムほど、より高い適応度が与えられるように定義しなければならぬ。事例の数を n , i 番目の事例を (φ_i, ψ_i) , 局面 φ に対してプログラムの返す評価値を $f(\varphi)$ として、適応度関数 (*fitness*) を以下のように定める。

$$fitness = \frac{1}{1 + \sum_{k=1}^n \delta(k)}$$

$$\delta(i) = \begin{cases} 0, & f(\varphi_i) > f(\psi_i) \\ 1, & f(\varphi_i) \leq f(\psi_i) \end{cases}$$

4 実験

4.1 遺伝的操作・実行パラメータ

本実験では、Koza の手法に従って、以下に示す操作を用いている。それらの内容の詳細は [1] を参照されたい。

選択 適応度比例選択方式
交叉 部分木の交換
突然変異 部分木の削除・生成

本実験で用いた GP の実行パラメータを表 1 に示す。

4.2 生成した関数の評価

実験で、8 つの評価関数 $A \sim H$ を得た。そこで、これらの評価関数の良さを知るために、次の様な思考実験を行った。

GP によって生成された評価関数を f とする。与えられた局面 P で可能な指手 m について、その指手によって到達する局面を関数 f で評価して、その値が最大になる指手 m を局面 P で選択する。これを一手先読み f 戦略という。また、与えられた局面 P である指手 m を指して到達する局面に対して、相手は、可能な手の中から、それによって到達する局面の f の評価値が最小値になるように指手を選択するであろうから、その値を最大にするように指手 m を選択する戦略を、二手先読み f 戦略という。

さて、先手が一手先読み f 戦略で指手を決め、後手はすべての可能な指手を考えることにして、ゲームの木を作成し、そのゲーム木の終端節点を調べて、勝型、引分型、負型の数を数える(負型の節点が 0 であれば、先手は一手先読み f 戦略で負けることがないことがわかる)。同様に、先手が二手先読み f 戦略、後手が一手先読み f 戦略、二手先読み f 戦略を採用した場合のゲームの木を作成し、上と同様に、勝型、引分型、負型の節点の数を数え上げる。

$A \sim H$ の関数についてこの思考実験を行った。結果をまとめたのが表 2 である。

表 2: 生成した関数の評価

関数	失敗事例	一手先読み			二手先読み		
		勝	分	負	勝	分	負
A	3	先手 86	10	3	119	5	0
		後手 358	71	28	476	107	8
B	5	先手 72	4	0	89	5	0
		後手 378	70	9	504	119	0
C	6	先手 85	11	3	87	8	0
		後手 300	60	73	381	306	10
D	3	先手 72	4	0	124	4	0
		後手 378	70	9	381	312	0
E	7	先手 72	4	0	137	11	0
		後手 290	121	42	524	99	0
F	5	先手 72	4	0	107	5	0
		後手 348	93	16	524	99	0
G	6	先手 68	8	0	132	15	0
		後手 331	110	48	464	319	0
H	19	先手 108	17	8	154	2	24
		後手 368	99	106	544	82	155

$(+4.88 (+ (+ (- (+ (- (-1.24 \text{ tn}) (+ \text{ cm pe})) (-4.71 \text{ ce})) (- (- (* (- \text{ tn } -2.81) -3.32) (* \text{ pm } (* \text{ pm ce}))) (+ \text{ ce } \text{tn}))) (+ (- (+ (- (+ \text{ cm pm})) (+ (* (-1.93 \text{ tn}) \text{ pm}) \text{tn}) \text{ce})) (+ (- \text{ cm pe}) (* \text{ pm ce}))) (* 0.32 1.25)) (+ (- \text{ cm pe}) (- (- \text{ ce } -2.42) (* (+ \text{ cm pm}) -2.08)))) (* (- (- \text{ omne } (* (- \text{ pm } (* \text{ cm tn})) \text{ce})) (- \text{ omne } \text{tn})) (+ (+ \text{ omnu } 3.53) (+ (* \text{ pm ce}) (- (* (* \text{ ce omnu } \text{ce}) \text{tn}))))))$

図 2: 関数 B の Lisp S 式表現

4.3 考察

図 2 は GP で生成した評価関数の一例 (関数 B) である (定数については小数第三桁以下を四捨五入して表示している). この関数で一手先読み f 戦略を用いると, 9 敗という結果であるが, これらの敗戦譜を調べ上げると, すべて図 3 の局面で敗着手を指していることがわかった. この局面における最善手は a の区画に×を打つことであるが, 関数 B は, 局面 (1),(2) に対して正しい形勢の比較ができない. これは, 本実験で与えた局面パラメータのもとでは, 局面 (1),(2) の形勢も正しく判断する評価関数は, GP で生成するのが困難であったと考えられる.

しかしながら, 二手先読み f 戦略を用いれば, 図 3 の局面でも正しい指手を選択することができた. 8 つの関数のうち 5 つの関数については, 負けることがないとい

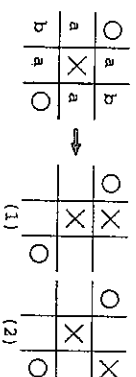


図 3: 敗着手を指した局面

う結果になった.

関数 H は正しく評価できない事例の数が 19 にも及んでいるが, これは GP の探索が局所解に収束したためと思われる. GP で生成する解を高いレベルで安定させるためには, 母集団のサイズを大きくして多様性を維持するなど, 実行パラメータを適切に定める必要がある.

5 おわりに

本稿では, ゲーム局面の形勢を判断する評価関数を, GP を用いて生成する手法を提案した. tic-tac-toe というゲームに対して, 二手先読み f 戦略を用いれば必ず負けない評価関数を生成することができた.

今後は, より複雑なゲームを対象とすること, 評価関数を意味的に解析することが課題である.

複雑なゲームを対象とした場合, 本実験のように事例を与えることは困難である. 現在, 次の二つの事例獲得の方法を検討している.

1. 対象ゲームのエクスパートが事例ベースを与える方法
2. 人またはコンピュータとの実践から事例を獲得する方法

評価関数の意味的な解析は, 評価関数が図 2 のような表現形式のままでは, 困難である. 本実験では, 評価関数に用いられる関数記号は和, 差, 積の三種類であるため, 人にとってわかりやすい多項式の形式に変換できる. GP で生成した評価関数の部分構造は, 対象ゲームの性質と深く関係があると考えられる. 評価関数の意味的な解析は, 対象ゲームの新たな性質の発見につながると思われる.

参考文献

- [1] Koza, J. R., Genetic Programming: A paradigm for genetically breeding populations of computer programs to solve problems, Report No. STAN-CS-90-1314, Dept. of Computer Science, Stanford Univ., 1990.
- [2] 増井 俊之, グラフ配置評価関数の進化的獲得, 「進化的アルゴリズム 2」 北野 宏明編, 産業図書, pp.127-144, 1995.