遺伝的プログラミングにおける木の深さと突然変異の与える影響

Influence of Tree Depth and Mutation on Genetic Programming

○非　渡辺 章人（同志社大院）　　正　廣安 知之（同志社大生命医科）
正　三木 光範（同志社大工）　　非　横内 久猛（同志社大生命医科）

Akihito WATANABE, Graduate School of Engineering, Doshisha University
Tomoyuki HIROYASU, Doshisha University, Tatara Miyakodani 1-3, Kyo-Tanabe, Kyoto
Mitsunori MIKI, Doshisha University, Tatara Miyakodani 1-3, Kyo-Tanabe, Kyoto
Hisatake YOKOUCHI, Doshisha University, Tatara Miyakodani 1-3, Kyo-Tanabe, Kyoto

Genetic Programming (GP) is one of evolutionary computation methods and GP can handle structural data such as tree structure, and graph structure. One of the problems of GP is surplus chromosome growth which is called a bloat. In this paper, searching mechanism of GP is discussed. Through the numerical examples of two types of Symbolic Regression problems, the performance of crossover and influence of mutation rate were researched. In these researches, the influences of tree depth and mutation rate on search solution were discussed. As result, it was found out that GP doesn't search interpolation by crossover in complex problem.

Key word: Genetic programing, Genetic Algorithm, mutation

1　はじめに

遺伝的プログラミング（Genetic Programming：GP)[1)] は，1992年にStanford大学のJohn Kozaらにより提案された進化論的計算手法であり，遺伝的アルゴリズム（Genetic Algorithms：GA)の遺伝子型を構造的な表現（木構造，グラフ構造）で扱えるように拡張したものである．ロボットの行動の規則生成や，画像フィルタの設計などといった実問題の解決を試みた場合，木構造，グラフ構造を決定しなければならず，GPの利用は非常に有効であると考えられる．

GPを利用する際の問題点としてブロートがあげられる．GPにおいて通常用いられる交叉は個体に対して破壊的な作用を持っていると考えており，イントロンの指数的成長であるブロートの発生を引き起こすとされている[2)]．このブロートの発生を抑制する方法として，Poliら[3)] による一様交叉やLandon[4)] による相同性交叉オペレータなど，交叉手法についての改良が多く提案されている．

本稿では，GAでは非常に探索の際に重要な交叉がなぜ破壊的な作用の振る舞いを行うのか，突然変異は解探索において，どのような操作を行うのかについて，再検討を行う．具体的には，GPのパラメータである生成木の深さと突然変異率の観点から，ブロート抑制の検討を行う．なお，生成木の深さとは，初期個体生成時の木の深さと突然変異木の深さの上限を指し示す．また，研究の第一段階として，対象問題に関数同定問題を用い，生成木の深さと突然変異率が探索にもたらす影響について調査を行う．

2　進化計算における探索戦略

2.1　開発と探査

進化計算に限らず最適化においては，開発(exploitation)と探査(exploration)が重要である．全探索に必要な計算コストよりも少なく最適点を決定するためには，大局的に最適点の存在を保証し，かつ，精度の良い解を決定する必要があるからである．大局的な解を開発しながら，精度の良い解の探査が必要なのである[5)]．しかしながら，計算コストを一定にした場合，この開発と探査はトレードオフの関係となる．すなわち，開発を重視すると精度の高い解がえられず，精度を重視すると，大局的な解の保証が得られにくい．最適化手法においては，これらの開発と探査のバランスをうまくとることが重要である．遺伝的アルゴリズム(Genetic Alogrithm: GA)や遺伝的プログラミング(Genetic Programming：GP)といった進化計算手法では，これらの開発と探査の探索メカニズムを自動的に切り替え良い解を探索している．

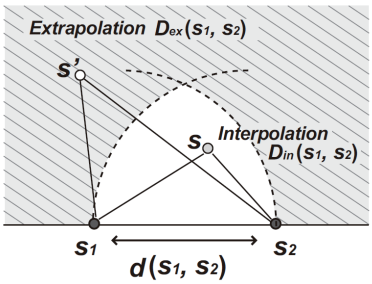

Fig. 1　Definition of Inter and Outer Region

2.2　内挿と外挿

GAの代表的な探索のメカニズムに交叉があげられる．交叉の設計においては，子個体を生成する際に，形質遺伝，および獲得を可能にすることが非常に重要である．Sakumaは解空間に距離尺度を導入し，親個体から構築される空間内に子個体生成が行われるような交叉を内挿交叉，外部に子個体生成が行われるような交叉を外挿交叉と定義している[6)]．Fig.1に内挿領域，外挿領域の概念図を示す．点線で囲まれている領域がs1，s2 に関する内挿領域，斜線の領域が外挿領域を示している．なお，対象問題によって距離の定義は異なる．ビットの問題の場合はハミング距離，TSP の場合は異なるエッジの数などが用いられる．

遺伝子型の近傍の設計と内挿交叉の設計が良好に行われている場合，良好な近接する親個体からは，交叉により良好な子個体が生成され，かつ，探査が十分に行われることとなる．親個体が離れている場合には，開発を重視した探索が行われる．すなわち，良好な遺伝子型の近傍の設計と内挿交叉の設計を行うことで，多様性のある初期個体を用意し，淘汰圧を低くすることで，十分な開発と探査が行え

ることとなる．内挿交叉だけでは，開発が十分でない場合には，突然変異と外挿交叉を用意して開発を行う必要がある．一方で，遺伝子型の近傍の設計と内挿交叉の設計のいずれかに不備がある場合には，探査は十分行われず，高い精度の解は得ることができない．

3 遺伝的プログラミング（Genetic Programming：GP)

3.1 GPのアルゴリズム

GPのアルゴリズムはGAと同様であり，以下に示すものとなってる．

STEP 1 初期解候補の生成

あらかじめ設定された数だけランダムに個体を生成する．生成した個体の数のことを母集団サイズ（Population Size）や単に個体数（Number of Individuals）と呼び，ここで生成した個体の集団を初期母集団とする．

STEP 2 評価

予め定められた適合度関数により，各個体の適合度を計算する．

STEP 3 選択

STEP 2で求めた評価値に基づき，次世代へ残す個体を選ぶ．手法としては，トーナメント選択やルーレット選択がある．なお，本稿ではトーナメント選択を利用している．

STEP 4 交叉

親1と親2の交叉点をランダムに選び，それぞれ交叉点に応じた部分木同士で交叉させる．

STEP 5 評価

STEP2と同様にして，各個体の適合度を計算する．

STEP 6 突然変異

個体の中より，ランダムに突然変異点を選び，その点に応じた部分木と突然変異木を入れ替える．ただしルートノードは突然変異の対象外である．全個体のうち何割の遺伝子を変異するかは突然変異率（Mutation Rate）によって定める．

STEP 7 終了判定

終了判定には，以下のような終了条件がある．

- 実行すべき最大世代にまで達したとき
- 目的とする個体が得られたとき

3.2 ブロート

ブロートとは，探索が進むにつれてプログラムサイズが増大する現象のことである．ブロートの発生は，探索の停滞や探索時間，およびメモリ消費量の増大をもたらす[7]．これはプログラムサイズが大きくなると，探索空間が広がることやプログラムの評価時間が長くなるからである．

4 GP における木の深さと突然変異の調査

本研究では，対象問題に関数同定問題を用い，生成木の深さと突然変異率による，解の適応度やプログラムサイズへの影響について調査を行う．なお，生成木の深さとは，初期個体生成時の木の深さと突然変異木の深さの上限を指し示す．

4.1 実験に用いるテスト問題

テスト問題は，GPの代表的なベンチマーク問題であるSymbolic Regression問題（Simple Symbolic Regression問題，Complex Symbolic Regression 問題）[1]とする．Symbolic Regression問題は構文的イントロンが発生しない問題，すなわち，すべてのノードが解の評価に影響を及ぼす問題である．以下に，問題について説明する．

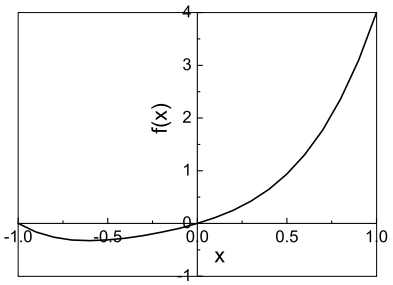

Fig. 2 Simple Symbolic Regression

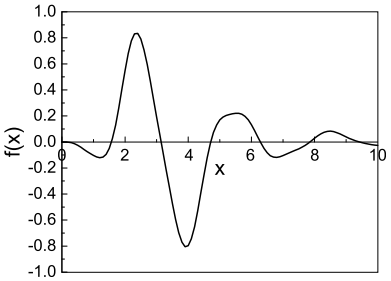

Fig. 3 Complex Symbolic Regression

4.1.1 Simple Symbolic Regression

Symbolic Regression問題とは，n組の入出力データから未知の関数f_{obj}を同定する問題である[1]．

Simple Symbolic Regression問題は，その中でも式(1)に示す単純な関数f_{obj}を同定する問題である[1]．その関数f_{obj}の概形をFig.2に示す．

$$f_{obj}(x) = x^4 + x^3 + x^2 + x \tag{1}$$

非終端記号は$\{+, \times, -, \%, sin, cos, exp, rlog\}$，終端記号は$\{x\}$である．

評価関数E_{val}式(2)は，-1から1の間を0.1刻みにした21個の入力に対する出力誤差の絶対値の総和であり，その総和が0.01以下（$E_{val} \leq 0.01$）の場合，探索が成功したとする最小化問題である．$prog$は生成されたプログラムである．

$$E_{val} = \sum_{i=0}^{20} |prog(x_i) - f_{obj}(x_i)| \tag{2}$$

4.1.2 Complex Symbolic Regression

Complex Symbolic Regression問題は，Symbolic Regression問題の中でも式(3)に示す複雑な関数f_{obj}を同定する問題である[8]．その関数f_{obj}の概形をFig.3に示す．

$$f_{obj}(x) = x^3 cos(x) sin(x) e^{-x} (sin^2(x) cos(x) - 1) \tag{3}$$

非終端記号は$\{+, \times, -, \%, sin, cos, exp, rlog\}$，終端記号は$\{x, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0\}$である．

評価関数E_{val}式(4)は，0から10の間を0.1刻みにした101個の入力に対する出力誤差の絶対値の総和であり，その総和が2以下の場合（$E_{val} \leq 2$），探索が成功したとする最小化問題である．

Table 1　Parameter of GP

	Simple Symbolic Regression	Complex Symbolic Regression
M	400	400
G	250	1000
P_c	0.9	0.9
K	2	4
E	1	1
D_{max}	17	17

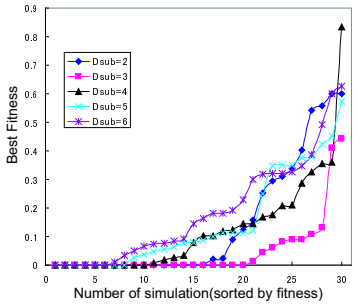

Fig. 4　Influence of tree depth on fitness in Simple Symbolic Regression

$$E_{val} = \sum_{i=0}^{100} |prog(x_i) - f_{obj}(x_i)| \tag{4}$$

4.2　実験概要

実験で利用するGPの固定パラメータを表1に示す．ここで，Mは個体数，P_cは交叉率，Kはトーナメントサイズ，Eはエリート個体数，D_{max}は最大の深さである．

また，木の深さと突然変異が解に与える影響を調べる為に，以下の2点について実験を行った．

- 木の深さが解に与える影響
- 突然変異が解に与える影響

なお，対象問題は4.1節で示したSimple Symbolic Regression，およびComplex Symbolic Regressionとする．

4.3　実験結果

4.3.1　木の深さが解に与える影響

突然変異率(P_m)を0.1に固定し，生成木・突然変異木の最大の深さ(D_{submax})を2，3，4，5，6に変更して解への影響を調べた．Simple Symbolic Regressionにおいて30試行，Complex Symbolic Regressionにおいて10試行，行った時に得られた解の最良解を昇順に並べたものをFig.4，Fig.5に，また，最良解のプログラムサイズを昇順に並べたものをFig.6，Fig.7に示す．

Fig.4より，Simple Symbolic Regressionにおいて，$D_{submax} = 3$の時，より多くの最適解を得ている．それに引き続き，$D_{submax} = 2$の時，より多くの最適解を得ている．また，Fig.5より，Complex Symbolic Regressionについては，$D_{submax} = 3$の時のみ最適解に至る試行が発生しなかったが，その他の試行についてはあまり差が生じなかった．

Fig.6より，Simple Symbolic Regressionにおいて，$D_{submax} = 2, 3$の時，他の試行と比較してプログラムサイズが小さいことが分かる．また，Fig.7より，Complex Symbolic Regressionについては，D_{submax}の大小によって，プログラムサイズに差異が生じていないことが分かる，

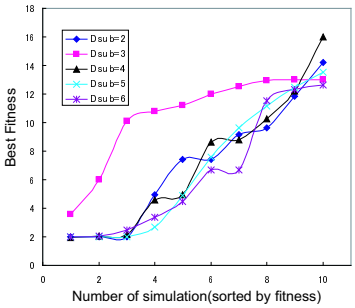

Fig. 5　Influence of tree depth on fitness in Complex Symbolic Regression

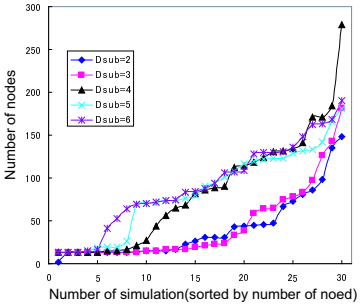

Fig. 6　Influence of tree depth on program size in Simple Symbolic Regression

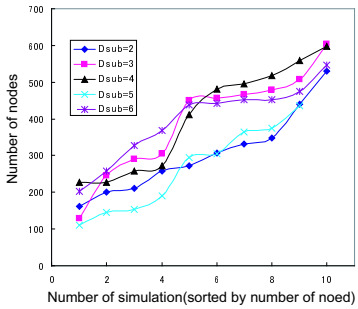

Fig. 7　Influence of tree depth on program size in Complex Symbolic Regression

4.3.2　突然変異が解に与える影響

生成木・突然変異木の最大の深さ(D_{submax})を4に固定し，突然変異率(P_m)を0.01，0.05，0.1，0.2，0.4に変更して解への影響を調べた．Simple Symbolic Regressionにおいて30試行，Complex Symbolic Regressionにおいて10試行，行った時に得られた解の最良解を昇順に並べたものをFig.8，Fig.9に，また，最良解のプログラムサイズを昇順に並べたものをFig.10，Fig.11に示す．

Fig.8より，Simple Symbolic Regressionにおいて，P_mが増減しても，評価値にあまり影響を及ぼしていないことが分かる．また，Fig.9より，Complex Symbolic Regressionについては，突然変異率の高い試行において良い結果が出ていることが分かる．

Fig.10より，Simple Symbolic Regressionにおいて，P_mの高い試行においてプログラムサイズが大きくなっていることが分かる．また，Fig.11より，Complex Symbolic

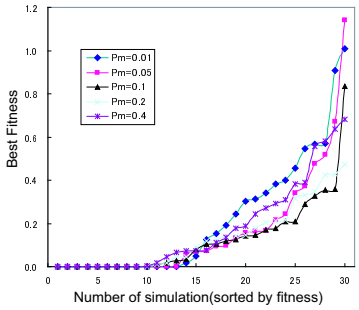

Fig. 8 Influence of mutation rate on fitness in Simple Symbolic Regression

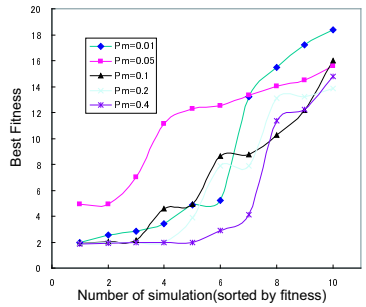

Fig. 9 Influence of mutation rate on fitness in Complex Symbolic Regression

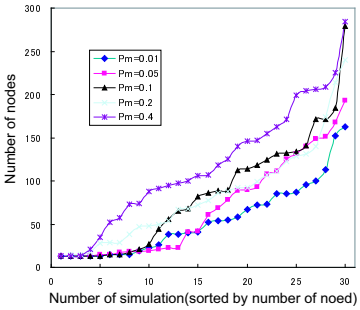

Fig. 10 Influence of mutation rate on program size in Simple Symbolic Regression

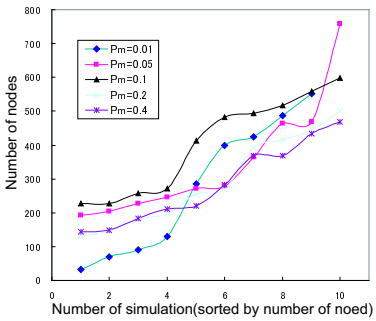

Fig. 11 Influence of mutation rate on program size in Complex Symbolic Regression

Regressionについては，P_mの増減がプログラムサイズにあまり影響を及ぼしていないことが分かる．

5 GPにおける木の深さと突然変異の与える影響

前節までの実験において，Simple Symbolic RegressionにおいてはD_{submax}が小さい時，解の精度が高まっている．一方で，Complex Symbolic Regressionにおいては，D_{submax}の大小は，解の精度に対してあまり影響を及ぼさず，P_mを大きくすることによって，プログラムサイズを保ったまま，解の精度を高めることができた．

Simple Symbolic Regression問題においては同定する関数が同じの部分木$(x \times x)$の集合であるため，最適解は比較的木の深さが短いものとなる．このような場合には，比較的短い部分木を用意し，突然変異の操作，すなわち開発戦略により最適解を決定することが可能である．この場合，交叉を行うと木の深さが長くなるために，突然変異による探索が有効であると考えられ，また，比較的短い部分木を用意することが重要である．

Complex Symbolic Regression問題の最適解は木の構造は複雑なものとなると考えられる．そのため，Complex Symbolic Regression問題では，最適解を得ることは難しい．木構造の一部のみを別の木に入れ替える操作を近傍と定義するならば，Complex Symbolic Regression問題の近傍を考慮すると明らかなように，最適解の近傍に位置する解は，適合度が高いものにならない．そのため，交叉によっても，突然変異によっても開発のみでは最適解の発見が難しい．これは先に述べた，良好な遺伝子型の近傍の設計と内挿交叉の設計が行われていない．そのため，内挿交叉が突然変異的な役割である開発のみの探索を行っている．さらに，P_mが高くなるとプログラムサイズが大きくなっていたが，これは，Simple Symbolic Regressionにおいて突然変異の影響が最適解の導出ではなく，イントロンの発生をもたらしているためと考えられる．

これまでの研究を振り返っても，GPにおける内挿交叉の設計は，開発探索の能力を向上させることに重点を置いていることがわかる．GPにおいては，部分木交換交叉[1)]が基本的な交叉であるが，GAと同様に，一様交叉や一点交叉[3)]も存在する．それに対して，家族組換え[9)]といったような特殊な交叉も提案されている．これは，動物の多くの種が，生き残るであろう子孫よりもはるかに多い数の子孫を産むことをモデル化したもので，家族サイズNを定義し多くの子個体を生成する．しかしながら，近傍がうまく設計できていない問題では，基本的には，突然変異を多く行わせることと同意であると言える．そのため，イントロンの存在は非常に重要であると言える．ある解ではイントロンとして存在するが，交叉により別の解に移動した場合に，突然変異の挿入木と同様の操作となり重要な部分木となるからである．そのため，GPにおける木の深さをむやみに短くする探索は最終的に良い解を得ることができないこととなる．相同性交叉[4)]はノードの位置と機能を考慮した交叉手法であり，ある位置からある位置までの木の機能がなくとも木全体の機能が同一の場合には，その部分木を削除することにより木の深さを削減している．しかしながら，あまりこの操作を行うと，内挿交叉による開発能力が削減してしまう可能性もある．

先に述べたように，進化計算においては，開発と探査のバランスが必須である．そのためには，内挿交叉により探査が行える仕組みが必要であり，それを実現するためには，遺伝子型における近傍の設計がきわめて重要であると言え，現在GPが適用されている多くの問題では，この近

傍設計がきちんと行われておらず，GPによる探索は，開発によるものがほとんどであり，探査により精度を高める探索であると言い難い．

6 結論

遺伝的プログラミング(GP)は木構造で表現できるルールや学習を進化計算手法により決定するアルゴリズムである．決定された木構造は，与えられた問題に対して，より精度が高く，かつできるだけ木構造の深さが短いことが求められる．最適化においては，開発と探査が重要である．大局的な解を開発し，精度の高い解を探索する必要があるが，計算コストが一定の場合，それらを行うことにはトレードオフの関係がある．そのため，両者のバランスをうまくとりながら最適解を決定する必要がある．遺伝的アルゴリズム(GA)やGPにおいては次の探索点候補を決定するメカニズムとして，交叉と突然変異といった遺伝的操作があげられる．交叉はさらに親個体の内挿を探索する内挿交叉と親個体の外挿を探索する外挿交叉とに分けられる．外挿交叉と突然変異は，主に開発の探索を行う操作であると考えられ，内挿交叉によって探査が行われる．

本稿では，GPにおける探索のメカニズムを再調査するために，まず，突然変異に着目して，突然変異率と突然変異の際に与える木の深さと解の木構造の木の深さとの関係を調査した．

Symbolic Regressionを対象問題として調査したところ，問題が複雑になると，突然変異による探索でなければ良い解が得られないことがわかった．これは，問題に対する遺伝子型を決定する際に，内挿交叉によって探査が行われていないことを示している．

一方，最適解の木構造が単純な問題においては，開発を重視したGPによって良好な解を得ることが可能である．そのような場合には，短い木を代入するような突然変異操作によって探索が可能であり，そのような探索を効率よく行うメカニズムの設計が重要となる．

例えば医療画像のフィルタなどを設計する場合には，用意されたテストセットに対して良好な値を示すだけでなく，未知のセットに対しても対応できることが必須となる．そのためには，ある程度の木の深さを用意する必要であろう．このような木構造を決定するためには，十分に近傍を設計すること，初期個体の検討を行うことが必要である．

参考文献

1) John R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, 1992.
2) 伊庭斉志. 遺伝的プログラミング入門. 東京大学出版会, 2001.
3) Riccardo Poli and W. B. Langdon. On the search properties of different crossover operators in genetic programming. In *University of Wisconsin*, pp. 293–301. Morgan Kaufmann, 1998.
4) W. B. Langdon. Size fair and homologous tree genetic programming crossovers. *Proc. of Genetic and Evolutionary Computation Conference(GECCO99)*, 1999.
5) J.-M. Renders and S.P. Flasse. Hybrid methods using genetic algorithms for global optimization. *IEEE Transactions on Systems, Man, and Cybernetics, Part B,*, Vol. 26, No. 2, 2002.
6) Jun Sakuma et al. Extrapolation-directed crossover for job-shop scheduling problems:complementary combination with jox. *Proc. of Genetic and Evolutionary Computation Conference(GECCO2000)*, pp. 973–980, 2000.
7) Wolfgang Banzhaf, Peter Nordin, Robert E. Keller, and Frank D. Francone. *Genetic Programming – An Introduction; On the Automatic Evolution of Computer Programs and its Applications*. Morgan Kaufmann, dpunkt.verlag, January 1998. （伊庭 斉志,新田 徹.訳：遺伝的プログラミング，科学技術出版(2000)）.
8) R. P. Salustowicz and J. Schmidhuber. Probabilistic incremental program evolution: Stochastic search through program space. In M. van Someren and G. Widmer, editors, *Machine Learning: ECML-97*, Vol. 1224, pp. 213–220. Springer-Verlag, 1997.
9) W. A. Tackett. Mining the genetic program. *IEEE Expert*, Vol. 10(3), pp. 28–38, 1995.