遺伝的プログラミング：人工蟻の探索

T00K049F 鶴田 寿男

1. 遺伝的アルゴリズム(GA)

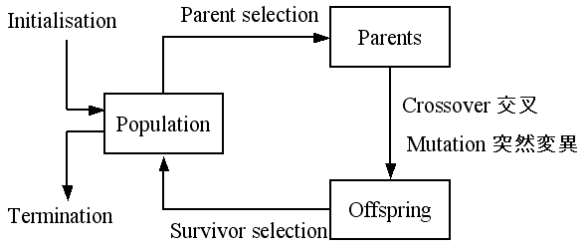

図 1. GA の一般的概念

GA とは図 1 に示すように，ある集団に交叉や突然変異といった遺伝的オペレータを適用することで，設定した適合度に近くなるように集団全体を徐々に進化させていく，という手法である．

2. 遺伝的プログラミング(GP)

GP は 1990 年頃に GA の中から一つの分野として確立した．

GP では，グラフ構造(とくに木構造)を扱えるように GA の手法を拡張する．これによりプログラムを進化させることが可能になる．

GP における基本的な人工知能の進化，ふるまいを学習するため人工蟻の探索を行う．

3. 人工蟻の探索(Santa Fe Trail)

これは人工生命の練習問題であり，与えられたフィールドの中で人工蟻が自分の限られたエネルギーの範囲内で出来るだけ効率的に餌を探索する．詳しいルールは以下のとおり．

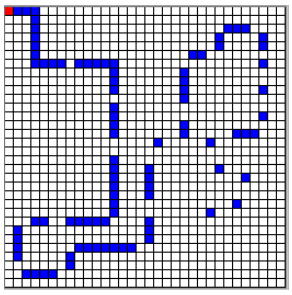

図 2. Santa Fe Trail

A) 蟻は動作(前進，向きを変える)を行うたびにエネルギーを 1 単位消費する．また壁を越えて進むことは出来ない．

B) 蟻はセンサーを持ち自分の前方に餌があるか否かがわかる．

C) 餌の有るマスに進んだ場合餌の所持数を＋1 し，マス上の餌を消す．

蟻のプログラムを GP で生成するために以下のように非終端記号と終端記号を定める．

IFF [2]…目前に餌があるとき第 1 引数，ないとき第 2 引数を実行する
PROG2 [2]…第 1，2 引数を順に実行
PROG3 [3]…第 1，2，3 引数を順に実行
R…右に 90 度向きを変える
L…左に 90 度向きを変える
F…一歩前に進む

適合度は以下のように評価する．

i. energy=400,food=0 とする．
ii. プログラムにしたがって蟻を動かす．F,L,R という終端記号を評価したら，energy=energy-1 とする．
iii. もし蟻の位置の餌があれば，food=food+1 としその位置の餌を消す．
iv. energy が 0 なら，food を適合度として返す．
v. ii.へ戻る．

4. 実験の条件

実験の基本条件は以下の通り．

集団数 100,世代数 500,交叉確率 80%, 突然変異率 1%,トーナメントサイズ 2,エリートサイズ 5

5. 選択

中間発表までのルーレット選択を用いた実験では，最適解を得たのは 50 回中 1 回，80 以上の値も 5 回とあまり良い成績とはいえなかった．

これはルーレット選択のみでの試行では集団の中でも最も優れた固体を次世代へ残さず取りこぼしてしまうという事例が起こりえるためと考えられる．よって今回はエリート選択を取り入れる．

エリート選択を取り入れたルーレット選択とトーナメント選択を 100 回ずつ実験し，よりよい結果を得た手法を用いることにする．

結果エリート＋トーナメントを基本条件として採用した．

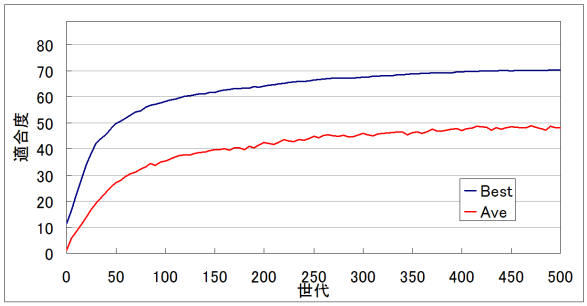

図 3. 基本条件における 100 回試行の平均

この手法では 100 回中 15 回最適解を得ることが出来ている．

6. 頑強性とブロート

訓練例とは異なる環境においても優れた動作をするプログラムを頑強であるという．

GP のプログラムは探索過程でその長さが増大する傾向がある．この現象をブロートという．大きすぎるプログラムは実行時間や計算コストの増大をまねく，また一般にこのようなプログラムは訓練例に対して過適応（Overfitting）となりやすく，頑強性を失いやすい．

＊ブロートの例

(* (+ 4 0) 1) ｜ (+ (+ (+ 1 1) 1) 1)

また以下のような条件下で交叉は効率的でなくなると考えられる．

1. 遺伝子操作が局所的である．
2. 偏りのある探索オペレータである．

これらを考慮し頑強性を高めるよう以下のような交叉を実験し比較する．

サイズ依存型交叉

このオペレータでは交叉で交換する部分木のサイズ(ノード数)を出来る限り等しくする．

これにより各世代での急激なプログラムサイズの増加を防ぐことが出来，また破壊的な交叉が起きないようになると考えられる．

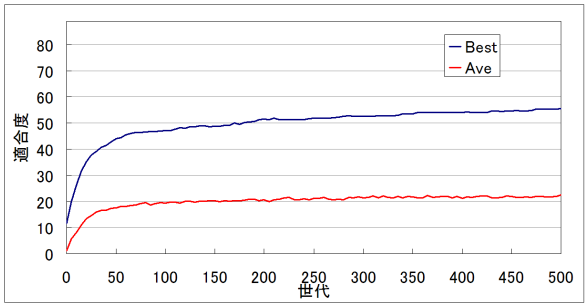

図 4．サイズ依存交叉：100 回試行の平均

結果，適合度は図 3. の基本条件による実験よりかなり落ちてしまった．最適解を得た事例も 2 回のみであった．一方ノード数においては基本の条件における実験が最終的に平均 63.49 個であったのに対し，21.43 個と非常に少なくブロートを抑えることには成功していることがわかる．

ノード数が少ないためこの交叉で得られた個体は他の環境に対しても良い成績を上げることが出来る．

以下に得られた 2 つの式を示す．

(IFF(F)(PROG2(R)(IFF(PROG3(F)(F)(F))(PROG3(L)(L)(IFF(F)(PROG2(R)(F)))))))

ノード数 17

(IFF(F)(PROG2(R)(IFF(F)(PROG3(R)(R)(PROG2(IFF(F)(R))(IFF(F)(F)))))))

ノード数 16

7. 局所解からの脱出

様々な GP においてある程度成長したあと数十世代に渡って適合度が成長しないという事例が多かった．このような現象は探索にスピードを重視する場合かなり大きな問題となる．

これは集団全体が成長するに従って良いものが残りやすいために偏りが起きてしまうことが主な原因であると考えられる．パラメータや交叉を調整することで偏りを抑えることは出来るが限界がある．

そのため根本的な改善策として，局所解に陥っていることを認識するシステムを設計し，実験を行う．

ある一定の世代間で適合度に変化が無い場合、停滞している(局所解に陥っている)と仮定し，それに応じて処理を行う．

今回は同じ適合度が 25 世代に渡って繰り返された場合に局所解に陥っていると仮定し，ランダムに生成した 20 固体を組み入れる．この操作はエリートを残した後に行うこととする．

ランダムな固体を入れることで多様性を持たせ，またブロートを抑える働きも期待できる．

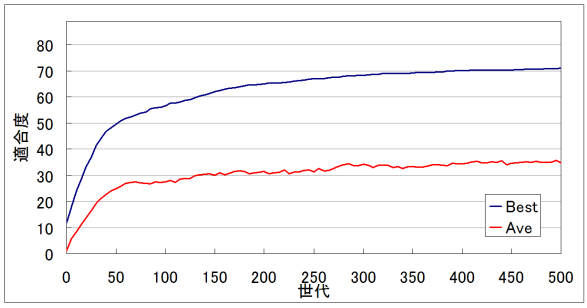

図 5．局所解からの脱出：100 回試行の平均

結果, 100 回試行の各世代の最適固体の平均においてわずかではあるが良い成果が得られた．最適解を得た回数も 19 回となり 15 回に比べよい成績を示した．

またノード数の増加は基本の条件に比べおよそ三分の二に抑えられていることが確認できた．

8. 考察

サイズ依存交叉は頑強性において良い結果を得ることは出来たが，成長が遅いため問題の条件に応じて初期世代のサイズを意図的に高めるなどの調整を行う必要がある．

今回は局所解に陥っていると仮定する条件と停滞していると判断した場合の処理を単純に定めたが，より複雑な統計量を判定に用いることや，判定条件を複数にする，突然変異率を操作する，など様々な拡張によりさらなる効果が期待できる．

文献

[1] A.E.Eiben and J.E.Smith,Introduction to Evolutionary Computing, pp.101-110,Springer(2003).

[2] 伊庭斉志，遺伝的プログラミング入門, pp.10-31,177-243,東京大学出版会(2001).