遺伝的アルゴリズム(GA)による関数最適化の一手法

4T−6

広瀬 哲也　丸山 勉　小長谷 明彦
NEC C&C研究所 コンピュータシステム研究部

1 はじめに

遺伝的アルゴリズムを用いる上で重要なポイントは，問題のマッピング，すなわち(1)遺伝子構造の決定,(2)個体に適用する遺伝オペレータ(子孫の生成方法),(3)個体の評価関数である.

遺伝的アルゴリズムを用いて関数最適化を行なう研究は，これまでも多くなされてきた．しかし従来手法では，ビット列による"遺伝子構造","交叉","突然変異"といった考えに捕らわれていて，問題となる関数空間と遺伝的アルゴリズムで扱う空間が一致せず，最適解を効率良く見つけられない場合があった．本稿では，関数最適化に特化した遺伝オペレータの開発を行ない，それを用いる事によって，より速くより良い解を求められる事を示す.

2 従来のGAによる関数最適化

関数の最適化手法として，従来からGAを用いた方法が研究されている．従来の手法では，ビット列で表された個体に対して交叉，突然変異を行ない選択を繰り返す．交叉は各々の個体のビット列の1部分を確率的に選択し，それらを組み合わせて新たな個体を作る，突然変異はある個体のビット列の1部を確率的に変化させ新たな個体を作る操作であった.

このような方法では新たに生成される個体は前の世代に大きく依存するため，探索点が偏りがちになる問題がある．以下，例として個体を3ビットの2進数で表現した場合について述べる.

値"3"と値"5"の2つの個体を考えると，その個体表現はそれぞれ"011","101"である．この2つの個体から，交叉によって新に生成される可能性のある個体は，"001","011","101","111"で,"000","010","100","110"の個体が生成される確率は0である．すなわち，この2つの個体を元にした場合，値"0"，値"2"，値"4"，値"6"の点については探索できない.

同様に，突然変異を個体のビット列の内，ある1ビットのみを確率的に変化させる操作とすると，値"3"の個体からは，値"0"，値"4"，値"5"，値"6"の点については探索できない．突然変異を個体のビット列の内，任意のビットを確率的に変化させる操作とする場合は，探索点の偏りは無くなるが，複数の乱数を生成する必要があり処理速度の面から探索効率を下げることになる.

このように従来方法では，探索点が偏ってしまい，交叉/突然変異の関数空間での意味が不明確である．その結果，十分探索が行なわれる前に収束してしまい，探索能力，探索効率が十分でないという問題があった.

3 本手法における遺伝オペレータ

本手法では，各個体はビット列ではなく変数値そのままで表現し，交叉を近傍探索，突然変異を大域探索と位置づけ，以下のような遺伝オペレータを用いる.

突然変異 選択された親から，全探索空間内にランダムに子供を生成する．(図1)

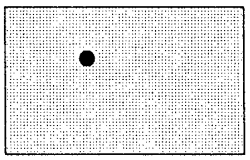

図1: 突然変異による探索空間

交叉 まず複数の親を選択する．選択された複数の個体間の距離dに応じて，親の位置を中心とした距離g(d)の近傍空間内に子供を生成する．但し，g(d)は単調増加関数とする．(図2)

突然変異は，ある一定確率で全探索空間を偏りなく探索可能である(大域探索)．また，交叉は初期の段階では比較的広い空間を探索し，探索が進むにつれ(収束が進むにつれ)それまでに残った比較的評価値の良い個体の近傍を偏りなく探索する(近傍探索).

このような2つの遺伝オペレータを用いることで，偏り無くしかも効率よく探索することが可能となる.

A Genetic Argorithm for Function Optimization
Tetsuya HIROSE, Tsutomu MARUYAMA, and Akihiko KONAGAYA
C&C Research Laboratories, NEC Corporation

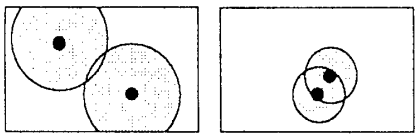

図 2: 交叉による探索空間

4 評価

関数最適化でよく使われている DeJong の 5 つの評価関数 [1] を用いて，従来手法との比較評価を行なった．従来手法としては，1 点交叉 /1 ビット突然変異と n 点交叉 /n ビット突然変異の 2 つの方法を評価した．但し，n は変数の数とし各変数ごとに 1 点交叉 /1 ビット突然変異が起こる．また，ビット列のコード化は binary/gray の 2 つの方法を用いた．

その他の条件は全て，ルーレット＋エリート選択，個体数 20, 交叉確率 1.0, 突然変異確率 0.01 として，100 回の試行を行ないその平均を評価した．その結果，全ての関数で本手法の有効性が示された．(実験では，$g(d) = d + \epsilon$ を用いた．)

一部の結果を図 3に示す．F'(x) は，各関数を，形を変えず最小値を 0.0 にしたもので，これを評価関数として用いた．Rnd.avg/Rnd.best が本手法での世代毎の個体の平均値 / 最良値で，最良値の線を矢印で示した．同様に，1p.best/np.best は従来手法の 1 点交叉 /n 点交叉で個体の最良値である．コード化については，F1, F2, F4 では違いはほとんどなく，F3 では binary, F5 では gray の方が評価が良かった．図 3では，それぞれ評価の良い方の結果を示してある．これによると，従来手法では非最適解に収束してしまうのに対し，本手法では個体の多用性を保ちつつより速くより良い解に到達していることが解る．

5 まとめ

以上最適化手法として GA を用いる場合，ビット列の" 交叉"，" 突然変異" といった考えに捕らわれず，問題の持つ探索空間を効率良く探索できるようなマッピング，遺伝オペレータを適用することが必要であることを，関数最適化を例として実際に示した．最後に、本研究において数々の助言を頂きました、コンピュータシステム研究部の皆様に深く感謝いたします．

参考文献

[1] D.E.GOLDBERG: *GENETIC ALGORITHMS in Search*, Addison Wesley

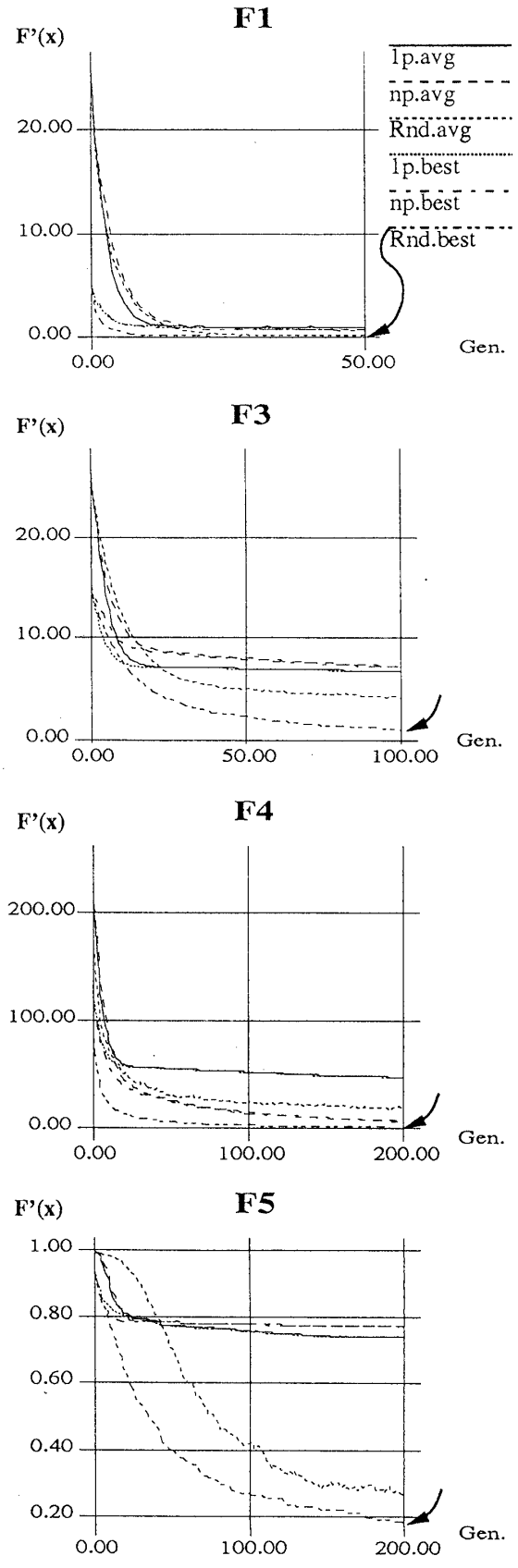

図 3: DeJong 関数の評価結果