

改良型免疫アルゴリズムによる 多峰性関数の大域的最大値の探索

中村秀明¹・松本 剛²・宮本文穂³

¹知能情報システム工学科

²日本電信電話株式会社(N T T)

³知能情報システム工学科

近年、遺伝的アルゴリズム(GA: Genetic Algorithm)の組合せ最適化問題や多峰性関数の最適化問題などへの適用が盛んに行われている。GAは多点探索と交叉オペレータにより最適解を得るため、状態空間を広く探索することが可能である。しかしながら、GAでは、解の探索が不十分な時点で集団の多様性が急速に失われることがある。この問題を解決するため、集団の多様性維持を目的とした手法がいくつか提案されている。

免疫アルゴリズム(IA: Immune Algorithm)は、生体のもつ免疫システムを真似た最適化手法であり、解の多様性を維持するとともに、複数の準最適解を得ることが可能である。本研究では、最適設計問題に、このIAを適用することを念頭に、IAの改良を行い、収束を早くするとともに、より多様性を持った解の探索を可能とした。

Key Words : genetic algorithm(GA), immune algorithm(IA), optimization, global maximum, diversity

1. はじめに

組合せ最適化問題では、考慮すべき組合せ数が、問題の規模に応じて指数関数的に増大するため、実用的な最適解が現実的な時間内に得られない場合が多い。特に、工学分野における最適設計問題などでは、制約条件が多かったり、設計パラメータが多数あるため、その組合せ数は、膨大な数となることが予想される。近年、このような組合せ最適化問題や多峰性関数の最適化問題などの解法として遺伝的アルゴリズム^{1),2)}(GA: genetic algorithm)が盛んに用いられている。GAは、生物の遺伝と進化のメカニズムを工学的にモデル化したものであり、生物の進化過程における遺伝的操作、すなわち、個体の遺伝子の選択、交叉、突然変異を模倣した最適化・探索手法である。GAでは、複数の探索点が相互協力的に解を探索していくため、比較的短時間で準最適解を得ることが可能であり、各個体の遺伝子情報と適応度のみがわかれば解の探索を行うことができることから、目的関数が微分不可能な評価関数である問題にも適用でき、GAを用いた設計支援システム³⁾など、多くのGAを適用した研究がなされている。

しかしながらGAでは、解の探索が不十分な時点

で、集団の多様性が喪失する現象がしばしば生じる。この現象は、一般に初期収束と呼ばれ、探索の初期段階で他の解に比べ優れた解が生成された場合、その解が集団中に多数残り、同一の遺伝子型の個体との間で交叉が行われやすくなり、解集団の多様性が急速に失われる現象である。この問題を解決するため、集団の多様性維持を目的とした手法がいくつか提案されている^{4),5),6),7)}。

一方、工学分野における通常の最適設計問題では、非常に幅広い設計空間の中から、最適な解を見つける必要があり、環境条件や製作や施工の手間など、目的関数の中に含めるこのできない要因が多数存在する。このような場合に、異なる特徴をもった複数の最適設計案が提示されれば、設計者は、その案をもとに設計を行うことが可能となる。このような問題に対しては、概ね1つの準最適解しか得られないGAでは不十分であると考えられる。

GAとは異なる最適化手法として、生体のもつ免疫システム^{8),9),10)}に基づいた最適化手法がいくつか提案されている。森ら¹¹⁾は、免疫システムに基づいた新しい最適化手法として免疫アルゴリズム(Immune Algorithm, IA)を提案している。このIAは、GAのsharing法に似た方法で初期収束を避けて

おり、解の多様性を維持するとともに、複数の準最適解を得ることが可能である。

本研究では、最適設計問題にこの I A を適用することを念頭に、I A の改良を行い、収束を早くするとともに、より多様性を持った解の探索を可能とした。

2. 改良型免疫アルゴリズム (Modified Immune Algorithm)

(1) 免疫システム (Immune System) ¹¹⁾

免疫には、生体を抗原から守るために免疫細胞が自己と非自己 (抗原) を識別する機能と、非自己である病原菌やウイルス、あるいは病気等のために変化した自己成分を排除する機能が備わっている。生体の免疫システムは生体内に侵入してくる抗原に対して、細胞遺伝子の再構成を行うことによりその抗原に対応する抗体を産生し、産生された抗体を増殖することにより抗原の排除を行う。中でも脊椎動物における免疫機構は、過去に排除したことのある抗原に対して有効であった抗体を記憶細胞に記憶しており、再び出現した抗原に対して直接有効な抗体をつくる機能を持つ。また、免疫システムは自己に対しても免疫性を示すことが知られている。同一種類の抗体が大量に産生されたとき、抗体産生を抑制する機能を持つサプレッサー細胞によりその抗体と同種類の抗体の産生を抑制し、バランスの崩れた状態から定常状態に戻そうとする機構を有している。

(2) 改良した免疫アルゴリズムの概要

免疫アルゴリズム (I A) は、免疫システムの抗体産生機構とその自己調節機構を工学的にモデル化したアルゴリズムである。本研究では、多様性を維持し、より確実に複数の準最適解が得られるように従来の I A ¹¹⁾ の一部を改良して適用した。本研究で用いた I A の基本的なアルゴリズムの概要を図-1 に示す。

【Step 1】抗原の認識

抗原をシステムに認識させる。最適化問題において抗原は目的関数と制約条件にあたる。したがって、目的関数および制約条件をシステムに組み込み、それらを制御するパラメータがあれば入力情報として与えることにより抗原を認識させる。

【Step 2】抗体の遺伝子型の決定 (Coding)

対象とする問題の事象にあたる抗体の遺伝子要素を記号列で表すことにより、対象とする問題に I A を適用することができる。つまり、I A に適用するには、対象とする問題の入力データを記号列 (抗

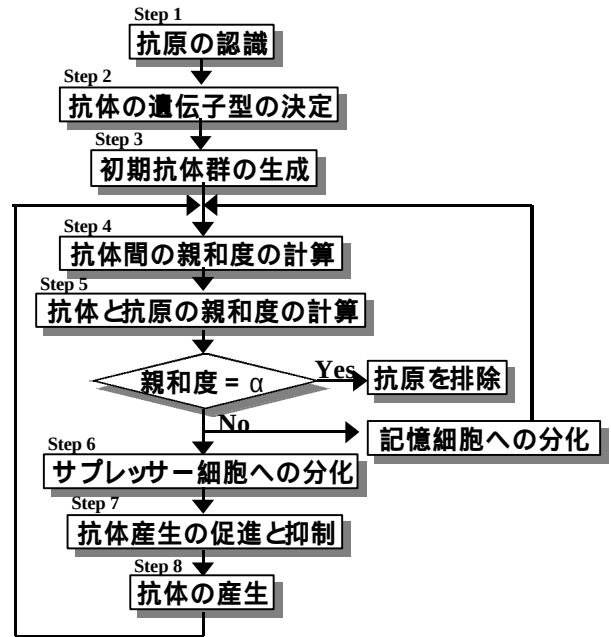


図-1 免疫アルゴリズム

体) へ変換する必要がある。通常、各抗体は 1 次元配列の遺伝子型であり、各要素を 2 進数 (0 または 1) で表現することが多い。

【Step 3】初期抗体群の生成

Step 2 で決められたモデルに基づいて、記号列を多数発生させる。このとき、記憶細胞により過去に有効であった抗体群を生成する。つまり、過去に有効であった抗体 (解候補) を保存した記憶細胞 (データベース) から抗体を読み込み、初期抗体群を形成する。その概念を図-2 に示す。記憶細胞が存在しないとき、あるいは初期抗体数に満たないときには、抗体の各遺伝子をランダムに決定し、遺伝子が異なる抗体を発生させることにより、初期抗体群を生成する。

【Step 4】抗体間の親和度の計算

全抗体について他の全ての抗体との親和度 (類似性) を測定する。抗体と抗体との親和度 $ay_{i,j}$ は次式により得られる。

$$ay_{i,j} = 1 / (1 + H_{i,j}) \quad (1)$$

ここで、 $H_{i,j}$ は抗体 i と抗体 j との距離であり、 $H_{i,j} = 0$ のとき、抗体 i と抗体 j は完全に一致。一般にはハミング距離が用いられる。

【Step 5】抗原と抗体間の親和度の計算

全抗体について抗原との親和度 (解の評価値) を計算する。抗原と抗体間の親和度 ax_i は次式により

得られる．

$$ax_i = opt_i \quad (0 \leq opt_i \leq a) \quad (2)$$

ここで， opt_i は抗原と抗体 i との結合力の強さの評価値であり，0 から a で規格化し， a が最適解になるように定義する．つまり，親和度 ax_i の値が a のとき，抗原の排除に成功した（これ以上最適解は存在しない）と考える．

【Step 6】記憶細胞とサブレッサー細胞への分化

探索過程で得られた有効な抗体を記憶細胞とサブレッサー細胞として産生する．ここで産生される記憶細胞が最適解の候補となる．

まず，全ての抗体の濃度を計算し，抗体の濃度 c_i がしきい値（ T_c ）を超えた抗体 i を記憶細胞に分化させる．ただし，全ての記憶細胞とその抗体の親和度を計算し，記憶細胞との親和度があらかじめ決めておいたしきい値（ T_{ac3} ）を超えた場合には，その記憶細胞との入れ換えを行う．また，記憶細胞の数には限りがあり，記憶細胞の総数が上限に達したときは，それまでに保存してきた記憶細胞と分化した記憶細胞との親和度を計算し，その中で最も親和度の高い記憶細胞と入れ換えを行う．ここでは，抗原との親和度が選択した記憶細胞よりも分化した記憶細胞の方が高い場合のみ入れ換えを行う．

次に，新しく分化した記憶細胞と同じ遺伝子を持つサブレッサー細胞を分化させる．ただし，同世代に分化する記憶細胞は1つ以下とし，候補の中で最も期待値の大きい抗体とする．

抗体 i の濃度 c_i は次式で与えられる．

$$c_i = \frac{\sum_{j=1}^N ac_{i,j}}{N} \quad (0 \leq c_i \leq 1) \quad (3)$$

$$ac_{i,j} = \begin{cases} 1 & ay_{i,j} \geq T_{ac1} \\ 0 & otherwise \end{cases} \quad (4)$$

ここで， T_{ac1} は類似度のしきい値であり， N は抗体の総数である．つまり，抗体 i と抗体 j がある程度の類似性（ T_{ac1} ）を持っているとき，同一種類の抗体とみなすことを示している．

【Step 7】抗体産生の促進と抑制

全抗体の中で抗原との親和度の低いものから $N/2$ 個を自然消滅させる．

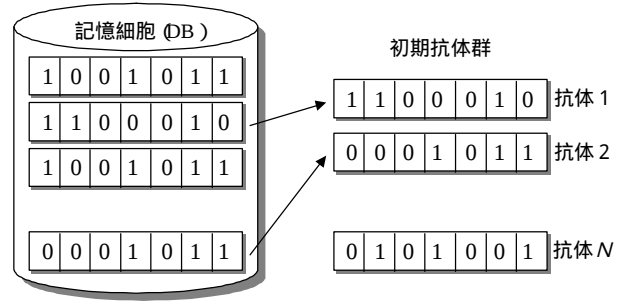


図-2 記憶細胞が存在したときの初期抗体群の生成

の操作で生き残った各抗体についてサブレッサー細胞との親和度を計算し，親和度がしきい値を超えた抗体を消滅させる．この操作により同じ記憶細胞の産生を回避することができる．

抗体の次世代に残る期待値を次式により与える．

$$e_i = \frac{ax_i \prod_{s=1}^S (1 - as_{i,s}^k)}{c_i \sum_{j=1}^{N'} ax_j} \quad (5)$$

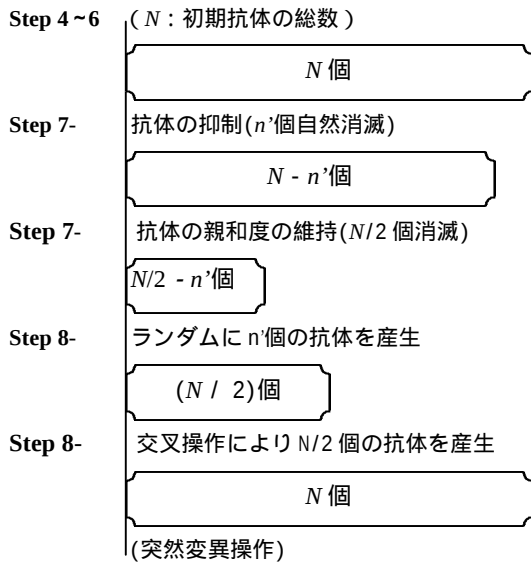
$$as_{i,s} = \begin{cases} ay_{i,s} & ay_{i,s} \geq T_{ac2} \\ 0 & otherwise \end{cases} \quad (6)$$

ここで， T_{ac2} は類似度のしきい値であり， N' は生き残っている抗体数， S はサブレッサー細胞の総数， k は産生抑制力 (Suppress power) である．

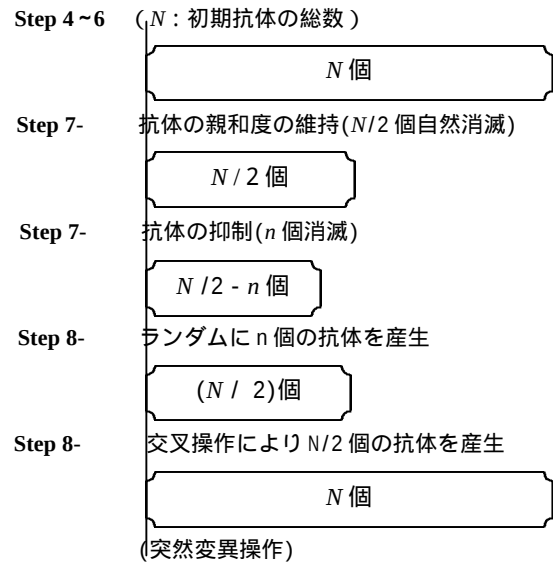
式(5)により期待値を与えることで，抗原との親和度が高い抗体ほど次世代に残る確率を高く，そしてサブレッサー細胞との親和度が高く濃度が高い抗体ほど次世代に残る確率を低くする．サブレッサー細胞に反応する抗体の産生を抑制することにより，同種類の記憶細胞の産生を防ぎ，複数の解の探索効率を高める．

【Step 8】抗体の産生

Step 7- で消滅した抗体に代わる抗体を産生する．乱数を用いて各遺伝子をランダムに決定することにより抗体を産生する．次に **Step 7** で残った抗体と新しく産生した抗体に対して，重複を許して $N/4$ 組の抗体のペアを選択する．ここで，確率的に期待値が高い抗体ほど選択されやすいものとする．ペアになった抗体に対して交叉を行い，新しく $N/2$ 個の抗体を産生する．産生した抗体に対して突然変異操作を行い，遺伝子を変化させる．ただし，交叉方法と突然変異確率は任意に設定する．ここで，



(1) 従来の I A



(2) 本研究で用いた I A

図-3 抗体数の推移

交叉と突然変異には G A¹⁾で行う交叉と突然変異と同じ操作を用いる。

Step 4 から Step 8 までをあらかじめ設定していた最終世代に達するまで行う。

(3) 従来の免疫アルゴリズムからの変更点

従来のアルゴリズムでは、Step 7 において全抗体に対してサプレッサー細胞により抑制を行った後に、親和度が低い N/2 個の抗体を自然消滅させていた。しかし、すでに得られている解候補の産生を防ぐには、全抗体からではなく、親和度の高い抗体の中から同一種類の抗体を消滅するだけで効果があると思われるため、本アルゴリズムでは、全抗体の中で親和度の低い N/2 個の抗体を消滅した後にサプレッサー細胞による抑制を行う手順に変更した。図-3 にその過程の抗体数の推移を示す。このアルゴリズムの変更により、サプレッサー細胞により消滅される抗体の数が少なくなるため、交叉を行う際に同種類の抗体を産生する可能性が高くなるとされる。つまり、サプレッサー細胞の抑制効果を適度に残しつつ、抗体は従来の I A と比較すると収束しやすく、記憶細胞を 1 つ産生するまでの世代数は小さくなることが期待でき、その分、一定世代数の中でより多種類の記憶細胞を産生することが期待される。

また、確実に抗原との親和度の高い抗体を記憶細胞（解候補）として残していくため、Step 6 における記憶細胞の産生時に 2 つの条件を付与した。1 つは記憶細胞の入れ換えを行うときに、単に一定の濃

度を越えた抗体を記憶細胞と入れ換えてしまうと、その抗体よりも条件に合った記憶細胞を消滅させてしまう可能性があり、それを防ぐために記憶細胞と抗体のうち抗原との親和度が高い方を記憶細胞として生存させる条件である。他の 1 つは、しきい値の概念である。濃度の高い抗体を記憶細胞に分化させる際に、それまでに産生された記憶細胞との親和度があらかじめ決めておいたしきい値よりも大きい場合、その抗体は記憶細胞と同一種類の細胞とみなし、記憶細胞との入れ換えを行う。これらの条件が存在しない場合、記憶細胞容量が小さいときに全ての記憶細胞が同傾向にある（記憶細胞間の距離が短い）結果に陥る可能性があり、多様性を持った解候補群を形成することが困難になる。

3. 改良型免疫アルゴリズムによる多峰性関数の最大値の探索

I A の多点探索問題に対する有効性を検証するために、G A および従来の I A と本研究で改良を行った I A を用いて次式で定義される多峰性関数 $F1(x)$, $F2(x)$ を対象としてシミュレーションを行った。

$$F1(x) = \sin^6(5\pi x)$$

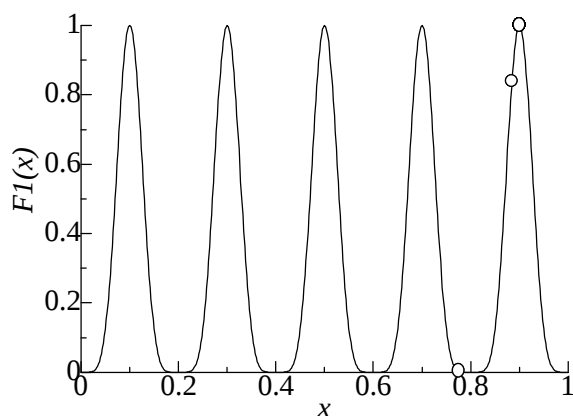
(7)

$$F2(x) = \exp\left[-2\log(2)\left\{\frac{(x-0.1)}{0.8}\right\}^2\right]\sin^6(5\pi x)$$

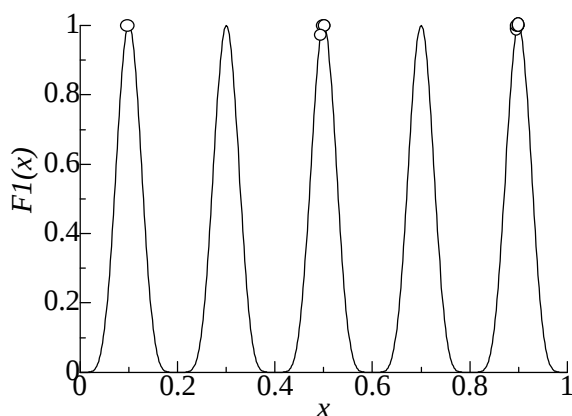
(8)

)

ここで、関数 $F1(x)$ は高さが等しい 5 つのピーク

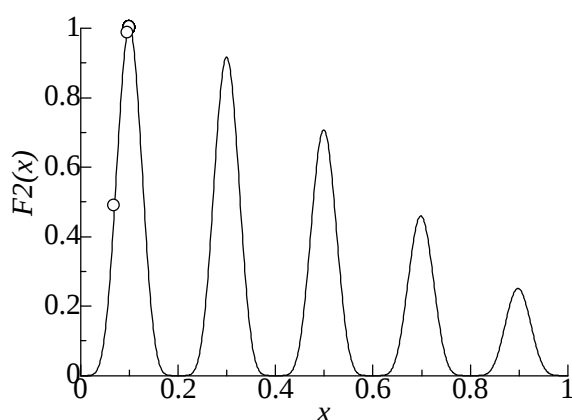


(a) G A によって進化した個体

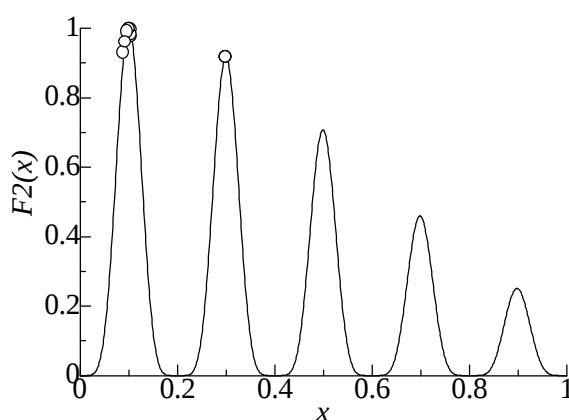


(b)従来の I A によって得られた記憶細胞

図-4 関数 $F1(x)$ に対するシミュレーション結果



(a) G A によって進化した個体



(b)従来の I A によって得られた記憶細胞

図-5 関数 $F2(x)$ に対するシミュレーション結果

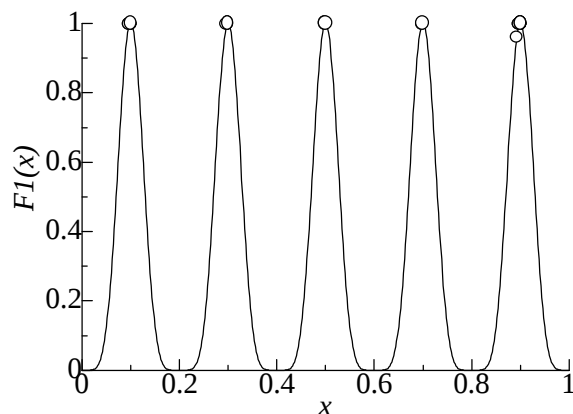
をもつ関数であり，関数 $F2(x)$ は高さが異なる 5 つのピークをもつ関数である．

シミュレーションの条件としては，関数 $F1(x)$ ， $F2(x)$ に対する両シミュレーションにおいて，一つの個体および抗体の遺伝子型を 12 ビット，個体数および抗体数 30，交叉確率 1.0，突然変異確率 0.01，交叉操作には一点交叉，突然変異操作には任意 1 ビットの反転を用いた．I A では，各しきい値を $T_{ac1} = 0.50$ ， $T_{ac2} = 0.50$ ， $T_{ac3} = 0.50$ とし，サプレッサーパワーを 1.5，記憶細胞（解候補）容量を 20 抗体とした．また，関数 $F1(x)$ に対するシミュレーションでは $T_c = 0.5$ ，世代数 200 とし，関数 $F2(x)$ に対するシミュレーションでは $T_c = 0.6$ ，世代数 100 と設定した．関数 $F1(x)$ および関数 $F2(x)$ に対するシミュレーションの結果をそれぞれ図-4 と図-5 に示す．なお，ここでは x に対する関数の値が大きなものほどよい解と設定する．

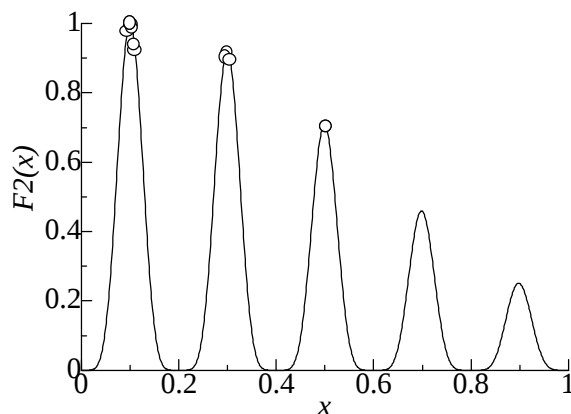
図-4(a)および図-5(a)に示す図中の マーク

は，それぞれ関数 $F1(x)$ ， $F2(x)$ に対して G A によって最終世代までに進化した個体の適応度をプロットしたものであり，図-4(b)および図-5(b)に示す図中の マークは，それぞれ関数 $F1(x)$ ， $F2(x)$ に対して従来の I A によって最終世代までに得られた記憶細胞の親和度をプロットしたものである．

図-4(a)に示すように，関数 $F1(x)$ は 5 つの最適解を有する関数であるにも関わらず，G A では大部分の個体は 1 つの最適解付近に収束していることがわかる．また，図-5(a)に示すように，大きさの異なる極大値を 5 つ有する関数 $F2(x)$ に対しては，大部分の個体が唯一の最適解付近に収束していることがわかる．なお，収束していない個体は，突然変異の影響を受けたものと考えられる．それに対して従来の I A を用いると，図-4(b)と図-5(b)に示すように，両関数に対してサプレッサー細胞の働きにより 1 つの局所解に陥るのではなく幅広い範囲で解の探索を行うため，複数個の極大値付近を示す記憶細胞が得られている．特に関数 $F2(x)$ に対しては，最適



(a)関数 F1 に対する記憶細胞



(b)関数 F2 に対する記憶細胞

図-6 改良した I A により得られた記憶細胞

解はもちろんのこと、やや最適解より小さな値を持つ極大値をも抽出していることがわかる。しかしながら、関数 $F1(x)$ の 5 つ最適解すべてを求められていないのは、シミュレーションに対して抗体数が 30 と少なかったことと世代数が小さかった可能性が考えられる。

このシミュレーションで示されたように、複数の最適解を有する関数を目的関数とする最適化問題に対し、従来の I A では抗体数および世代数に比較的小さい数を設定するとすべての点を求めることは困難であると思われる。そのため、すべての最適解を求めるには抗体数と世代数がある程度大きな値を取る必要があるが、対象とする問題によっては、評価関数の計算に多くの計算時間を必要とする場合があり、解を得るまでに莫大な時間を要することになる。そこで、抗体数と世代数がある程度小さくしたときにも確実に複数の解を得ることを目的として改良したのが改良型 I A（以下、本 I A）である。

そこで、本 I A の有効性を示すために同様のシミュレーションを行った。シミュレーションは式 (7) および式 (8) で示した関数 $F1(x)$ および関数 $F2(x)$ を対象とし、前節の従来の I A で用いた条件と同じ条件で行った。本 I A により得られた結果を図-6 に示す。

関数 $F1(x)$ に対して従来の I A では、図-4 からわかるように最終世代までに 5 つの最適解のうち 3 つあるいは 4 つの最適解付近の記憶細胞しか生成できなかった。しかし、図-6(a) に示されているように、同じ条件下にも関わらず、本 I A を用いると 5 つすべての最適解付近の記憶細胞が生成されている。また、大きさの異なる局所解を複数有する関数 $F2(x)$ に対しては、図-6(b) に示すように最適解だけでなく、従来の I A に比べると本 I A の方がある

程度低い局所解まで探索していることから、柔軟性をもって多点探索が行われていることがわかる。以上のことから、本 I A は従来の I A よりも多様性をもって多点探索が行われており、従来の I A よりも多点探索を行う問題を扱うには有効であるといえる。

また、G A と I A の大きな違いの 1 つであるサブレッサー細胞の影響を検証するために、本シミュレーションで得られた多様度の推移を図-7 に示す。多様度¹¹⁾とは、産生した抗体の多様性を測定する尺度であり、多様度は次の手順で得られる。 M 個の遺伝子をもつ N 個の抗体（個体）により抗体（個体）群が構成されており、抗体のとりうる記号が S 個存在する。このとき抗体（個体）の遺伝子座 j の情報エントロピー $H_j(N)$ は次式で表される。

$$H_j(N) = \sum_{i=1}^S (-p_{ij} \log p_{ij}) \quad (9)$$

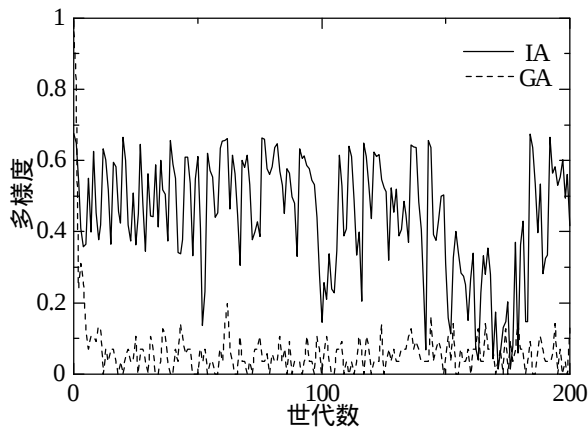
ここで、

$$p_{ij} = \frac{\text{遺伝子座 } j \text{ に出現した } i \text{ 番目の記号の総数}}{N}$$

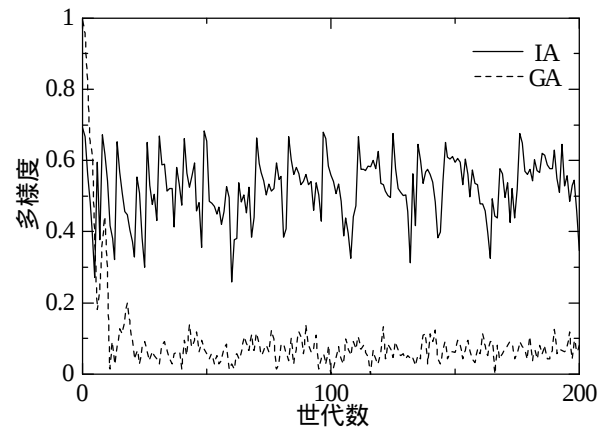
したがって、抗体（個体）群の多様性の平均エントロピー $H(N)$ は次式で表され、抗体（個体）群の多様性を測定することができる。

$$H(N) = \frac{1}{M} \sum_{j=1}^M H_j(N) \quad (10)$$

ここでは、G A と改良した I A（本 I A）による結果を比較した。なお、個体数および抗体数は 100 個としたときの結果も示す。個体数・抗体数が 30 のときと 100 のときの両シミュレーションにおいても、



(1) 個体数・抗体数 30



(2) 個体数・抗体数 100

図-7 GAと本IAの多様度の推移

GAでは手法特有の初期収束の早さにより個体のほとんどが早い世代で収束するために多様性が低くなっている。それに対して、本IAではサプレッサー細胞の働きにより常に新しい抗体を産生することにより多様性を維持していることがわかる。ここで、抗体数が30のときに150世代目あたりから180世代目あたりにかけて多様度が低くなっているのは、ほとんどの解候補を検出したことにより新しいサプレッサー細胞が産生されなくなり、サプレッサー細胞により消滅される抗体の数が減少したために、新しく産生される抗体の数が少なくなったためと思われる。この結果からサプレッサー細胞により多点探索能力が維持されていると考えられる。

6. まとめ

本研究では、遺伝的アルゴリズムと改良した免疫アルゴリズムの概要について述べ、改良した免疫アルゴリズムの最適設計問題に対する有効性を検証するため、多峰性関数を対象としたシミュレーションを行った。その結果、多峰性関数に対し、GAでは1つの最適解を確実に求めるものの、最適解が複数個存在するときにはそのうちの1つの最適解のみしか抽出できない結果となった。それに対し、免疫アルゴリズムでは、最適解が複数存在するときには複数の最適解を抽出することができ、さらに値の異なる局所解が複数個存在する関数に対しても、ある程度大きい値を持つ局所解を検出することができた。さらに改良した免疫アルゴリズムを用いると、従来の免疫アルゴリズムよりも確実に多くの複数の最適解、あるいは局所解を検出することができた。

実際の最適設計問題では、多くの要因を含んだ複数の目的関数が存在し、さらには環境条件や景観、

あるいは製作・施工条件等の数値で表すことのできない項目も考慮しなければならないため、設計者の意図を反映できるように設計を行うには複数個の設計案を提示する必要がある。したがって、改良した免疫アルゴリズムは、遺伝的アルゴリズムや従来の免疫アルゴリズムと比較すると設計問題に対して非常に有効であると思われる。そのような観点から考えるとGAは設計問題を解くには不十分であるが、1つの目的をもって設計したいときや早急に1つの実用解のみが必要であるときには、特有の初期収束により1つの実用解を高速に得ることが可能であるGAが有効であるといえる。したがって、効率よく設計案を提示するには、状況により免疫アルゴリズムと遺伝的アルゴリズムを使い分ける必要があると思われる。

参考文献

- 1) 荻原将人：ニュー・ロ・ファジィ・遺伝的アルゴリズム，産業図書，1994.
- 2) 石田良平，村瀬治比古，小山修平：遺伝的アルゴリズムの基礎と応用，森北出版株式会社，1997.
- 3) 千々岩浩巳，三原徹治，太田俊昭：GAによる鋼管杭基礎構造の最適配置決定法に関する研究，土木学会論文集，No.549/ -37, pp.97-105, 1996.10
- 4) D.E. Goldberg: *Genetic algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, 1989.
- 5) 坂和正敏：遺伝的アルゴリズム，朝倉書店，1995.
- 6) 市川，石井：遺伝子分布評価に基づく遺伝的アルゴリズムの多様性維持：計測自動制御学会論文集，Vol.30, No.10, pp.1242-1250, 1994.
- 7) 森，吉田，喜多，西川：遺伝的アルゴリズムにおける熱力学的選択ルールの提案，システム制御情報学会論文誌，Vol.9, No.2, pp.82-90, 1996.
- 8) 北野宏明：遺伝的アルゴリズム2，産業図書，1995.
- 9) J.D Farmer, N.H. Packard and A. S. Perelson: *The Immune System, Adaptation, and Machine Learning*, Physica 22D, pp.187-204, 1986.

- 10) 和田健之介，田中伸一，和田桂子：進化システム論
遺伝的アルゴリズムの基礎(5)，Computer Today, 51,
No.9, pp.82-87, 1992.
- 11) 森 一之，築山 誠，福田豊生：免疫アルゴリズムに
よる多峰性関数最適化，電気学会論文誌，C117, No.5,
pp.593-597, 1997.

(1998. 5. 15 受理)

SEARCH FOR GLOBAL MAXIMUM OF A FUNCTION OF MANY VARIABLES BY USING MODIFIED IMMUNE ALGORITHM

Hideaki NAKAMURA, Tsuyoshi MATSUMOTO and Ayaho MIYAMOTO

Genetic Algorithms(GAs) are one of the search procedures for combinatorial optimization problems. Since GAs are based on multi-point search and use crossover operator, GA have an excellent global search ability. However, in GAs, the diversity of the population sometimes decreases rapidly in the process of searching. In order to overcome this drawback, some researchers propose new algorithms for maintaining the diversity of the population.

Immune Algorithms(IAs) are the optimization technique, which imitate the immune system. IAs are able to get pulural semi-oprimum solution and maintain the diversity of the population. We improve IAs in order to converge rapidly and maintain the diversity of the population more better so that we could apply IAs to optimization design problem. Results of numerical examples show that modified IA performs better than former IAs.