エージェント指向自己適応遺伝アルゴリズム

村田佳洋[†] 柴田直樹[†] 伊藤 実[†]

遺伝アルゴリズム（Genetic Algorithm，以下 GA）の探索効率は，突然変異率や交叉率といったパラメータによって大きく左右される．しかし，多くのパラメータの調整を人手で行うのは困難である．そこで，パラメータを自動的に調整する様々な適応 GA が提案されている．従来の適応 GA のほとんどは少数のパラメータしか適応させられず，また，多数のパラメータを適応させる適応 GA であっても，そのほとんどが大きな計算量を必要としていた．本論文では，エージェント指向の手法によりメタ GA と環境分散型並列 GA を組み合わせ，多数のパラメータを同時に適応させつつ探索を行うエージェント指向自己適応遺伝アルゴリズムを提案する．評価実験を用いて，この手法により 4 つのパラメータが合理的な計算量で同時に適応させられることを示す．

Agent Oriented Self-Adaptive Genetic Algorithm

Yoshihiro Murata,[†] Naoki Shibata[†] and Minoru Ito[†]

Efficiency of Genetic Algorithms (GAs) depends largely on parameters such as crossover rate and mutation rate. In general, however, it is difficult to adjust those parameters manually. Although there are a few researches about adaptive GAs for adjusting multiple parameters, they require extremely large computation costs. In this paper, we propose a new algorithm based on multi agent techniques which combines existing meta-GA techniques and GA with distributed environment scheme. Through some simulations, we have confirmed that the proposed algorithm can adapt multiple parameters in reasonable computation costs.

1. はじめに

遺伝アルゴリズム[5]（Genetic Algorithm，以下 GA）は生物の進化を模した最適化アルゴリズムであり，広範な領域の問題に適用可能な手法として期待されている．

しかし，GA の探索効率は与えられるパラメータ（突然変異率，交叉率など）の組合せによって大きく左右されるという問題があり，多くのパラメータの調整を人手で行うのは困難である．そこで，パラメータを自動的に調整する，様々な適応 GA が提案されている[1),3),9),10)]．しかし，従来の適応 GA のほとんどは 1 ないし 2 つのパラメータしか適応させられず，多数のパラメータを適応させる適応 GA であっても，そのほとんどが大きな計算量を必要としていた．

適応 GA の最大の目的は，パラメータ適応の労力を軽減することである．その観点から，適応 GA を用いるためには，必要な予備知識や事前定義が少なければ少ないほど望ましいと考えられる．一般に，GA を適用するためには，適応度の評価方法と遺伝的演算子（交叉演算子や突然変異演算子）の定義を与えることが必要である．いくつかの適応 GA はそれに加えて多様性などに関する定義を必要とし，これらを利用して探索効率を高めている[9)]．しかし，これらを定義することは，問題によっては労力を必要とする．

本論文では，エージェント指向の手法により，適応 GA の 1 つであるメタ GA と島モデル GA の 1 つである環境分散型並列 GA[11)]（いずれも後述）を組み合わせ，パラメータを同時に適応させつつ探索を行うエージェント指向自己適応遺伝アルゴリズム（Agent oriented Self-Adaptive Genetic Algorithm，以下 A-SAGA）を提案する．A-SAGA は，適応度の評価方法と遺伝的演算子の定義が与えられれば，各島に与えられるパラメータがどのような組合せでも適用可能であるという特徴を持つ．本手法の有効性を，メタ GA との探索効率の比較実験により評価し，4 つのパラメータが合理的な計算量で同時に適応させられることを示す．

2. 関連研究

Goldberg らは，個体数や交叉率などのパラメータ

† 奈良先端科学技術大学院大学情報科学研究科
Graduate School of Information Science, Nara Institute of Science and Technology

の最適値を与えるために，one max problem に GA を適用した際に有効に働くパラメータの範囲を理論的に計算し[6),8)]，その範囲を示すコントロールマップと呼ばれるグラフを描いている．また，いくつかの選択手法の特性を分析している[7)]．しかし，これらの分析結果を利用して具体的なパラメータの値を得るためには，対象となる問題の特性を改めて定式化する必要があり，問題によってはこの定式化が困難な場合がある．この困難と我々の研究の関連性については 4.3.3 節において改めて考察する．

Bäck による適応 GA[1)] では，各個体の突然変異率がそれぞれの遺伝子上にコーディングされている．これにより，良い突然変異率を持った個体が生き残ることが期待される．しかし，高い突然変異率を持つ個体は致死個体となる可能性が高く，終盤には低い突然変異率を持つ個体ばかりになってしまう傾向がある．そのため，終盤の探索力が低くなることが報告されている[4)]．

Espinoza らによる適応 GA[3)] では，個体それぞれが局所探索を行える．また，個体の生殖などの計算量と局所探索の計算量の割合を適応させ，評価回数ごとの探索効率の改善がなされている．

Krink らの適応 GA[10)] では，格子状空間の座標に応じて交叉率と突然変異率が定っており，この空間内に配置された個体が，より望ましいパラメータの組合せに対応する座標を目指して移動する．さらに，同じ格子の中に同時に存在できる個体数が制限されているため，個体の座標がある程度収束した後も交叉率と突然変異率の多様性が確保される．

メタ GA は，GA のパラメータの組合せ（以後，パラメータベクトル）を GA によって探索する手法である．しかし，メタ GA では，通常の GA の実行，パラメータベクトルの吟味，再度の通常の GA の実行という過程を繰返すため，評価回数が極端に大きくなる傾向があり，全体の計算量に多大な影響を及ぼす．たとえば，100 個体 100 世代の通常 GA は 100×100 の評価回数を必要とするが，メタ GA は，通常 GA の 1 度の実行で 1 回分の評価を行うので，10 個体 20 世代のメタ GA を実行すると，$10 \times 20 \times 100 \times 100 = 2{,}000{,}000$ の評価回数を必要とする．

Kee らによるメタ GA の改良法[9)] では，実際の問題に対して探索を行う前に，トレーニングのための事前探索を行う．事前探索において，まず，個体群の状態をいくつかの指標により分類する．次に，各個体群に対し，数十種類のパラメータベクトルに関して探索効率を調査する．これにより，個体群の状態に応じてどのパラメータの組合せの探索効率が高いかを調べる．実際の問題の探索では，個体群の状態に応じて，トレーニングにおいて最も探索効率が高かったパラメータベクトルを使用する．

並列 GA の 1 つとして島モデル GA（Island GA，以下 IGA）[2)] がある．IGA は，並列に実行させたいくつかの GA を島と見なし，一部の個体を別の島に移民することにより探索過程の情報を共有し，共同探索を行う．三木らの環境分散型並列 GA[11)] では，IGA の各島に対してそれぞれ異なったパラメータベクトルを与えることにより，良いパラメータベクトルが与えられた島において良い解を見つけることが期待できる．ただし，パラメータベクトルは利用者が与えるので，これは適応 GA ではない．

3. 提 案 手 法

3.1 従来手法の問題点

従来の適応 GA のほとんどは 1 ないし 2 つのパラメータしか適応させられず，多数のパラメータを適応させる適応 GA であっても，そのほとんどが大きな計算量を必要としていた．また，適応 GA の最大の目的は，パラメータ適応の労力を軽減することである．その観点から，適応 GA を用いるためには，必要な予備知識や事前定義が少なければ少ないほど望ましいと考えられる．一般に，GA を適用するためには，適応度の評価方法と遺伝的演算子の定義を与えることが必要である．いくつかの適応 GA はそれに加えて多様性などに関する定義を必要とし，これらを利用して探索効率を高めている．しかし，これらを定義することは，問題によっては労力を必要とする．

メタ GA は適応度の評価方法と遺伝的演算子の定義を与えるだけで多数のパラメータを同時に適応させることができるが，計算量が大きいという欠点があった．この原因は，1 つのパラメータベクトルを評価するために 1 度 GA を実行しなければならない点にある．評価するパラメータベクトルの数が多いほど，メタ GA は良いパラメータベクトルを見つける可能性が高まる．同じ計算量でパラメータベクトルをより多く評価しようとすると，GA 1 回ごとの計算量を減らさなければならない．しかし，その計算量を減らすと GA は十分な探索を行うことができず，良い解を導くことが難しい．さらに，計算量を減らした GA に適したパラメータベクトルと元の GA に適したパラメータベクトルが一致するとは限らないという問題がある．

3.2 提案手法における改良点

我々は，メタ GA と環境分散型並列 GA を組み合わ

せ，A-SAGA の前身であるアルゴリズムを開発した．以降，このアルゴリズムを A-SAGAβ と呼ぶ．単純にメタ GA を IGA 化した場合には，パラメータベクトル 1 つを評価するために GA を 1 度実行する必要がある．A-SAGAβ では，IGA における各島を 1 つのエージェントと見なし，それぞれに異なるパラメータベクトルを与えることにより複数のパラメータベクトルを同時に適応させる．このことにより，システム全体において 1 度 GA を実行するごとに複数のパラメータベクトルを同時に評価することができる．環境分散型並列 GA の特徴から，A-SAGAβ においてそれぞれのエージェントの計算量が少なくても，システム全体として良い解を導くことが期待できる．区別のために，各個体の持つ適応度のことを個体適応度，エージェントの評価のために与えられる適応度のことをエージェント適応度と呼ぶ．

メタ GA では，個体群の中で最良の個体適応度を持つ個体（以後，エリート個体と呼ぶ）の適応度をパラメータベクトルの評価として与えている．しかし，A-SAGA では，同じ手法を用いてもうまく働かない．その理由は，各エージェントの持つエリート個体がまわりのエージェントからの移民に依存して与えられ，パラメータベクトルに対する評価が独立に与えられないためである．A-SAGAβ では，エージェント適応度はエリート個体の個体適応度の上昇値の累積値で評価する．ただし，エリート個体の個体適応度は，より良い個体適応度を持つ個体が他の島から移民してきた場合にも上昇するが，この上昇はそのエージェント自身の探索効率の高さによるものではないので，移民による個体適応度の上昇はエージェント適応度に反映させない．

A-SAGAβ では，各エージェントは，システム全体の利得の直接的な最大化を目的とするのではなく，それぞれの得る利得の最大化を目的とする点に自律性を見い出している．また，複数の利己的なエージェントを用いてシステム全体の利得の最大化を試みる点から，A-SAGAβ はマルチエージェントシステムであると見なしている．

A-SAGAβ では，エージェントにパラメータベクトル $\mathbf{v}_\beta = (n, p_\mathrm{m}, p_\mathrm{c}, l)$ が与えられる．ここで，

n : 個体数
p_m : 突然変異率
p_c : 交叉率
l : 線形スケーリング係数

である．各エージェントは通常の GA により探索を行う．エージェントごとに一定の評価回数が与えられるが，n が異なるため，繰返し回数も異なる．

3.3 予備実験とそれに対する考察

A-SAGAβ の評価実験を行ったが，期待していたほどの性能は得られなかった（4.2 節）．

探索の序盤において探索効率が高いエージェントを序盤高効率エージェント，探索の終盤において探索効率が高いエージェントを終盤高効率エージェントと呼ぶ．A-SAGAβ において，この両者が混在した場合と，環境分散型並列 GA と同様の効果が働き，移民を通じて助け合うことにより全体として良い探索を行うことができる．しかし，より個体適応度が高いエリート個体が移民によって得られた場合には，その個体適応度の上昇値をエージェント適応度に換算しない．したがって，序盤高効率エージェントと終盤高効率エージェントが混在した場合，探索の序盤には序盤高効率エージェントだけが，終盤には終盤高効率エージェントだけが個体適応度を得る傾向がある．一般に，GA において探索効率の高いパラメータは探索過程において変化する[9]．また，探索の序盤において世代ごとの適応度の上昇値は大きいが，終盤においてこれは小さくなる．したがって，A-SAGAβ では，エージェント適応度を比較したとき，序盤高効率エージェントの方が終盤高効率エージェントよりも大きなエージェント適応度を得る傾向がある．

A-SAGAβ では，エージェント適応度によってエージェント，および，それが持つパラメータベクトルを評価している．その結果，低いエージェント適応度しか持たない終盤高効率エージェントは淘汰され，序盤高効率エージェントばかりになってしまう．このことからシステム全体の性能が低下し，期待した性能が得られなかったと考えられる．

3.4 エージェント指向自己適応遺伝アルゴリズム

前節の考察を踏まえて，A-SAGAβ の探索過程を分割し，それぞれ別のエージェント群により探索するエージェント指向自己適応遺伝アルゴリズム（A-SAGA）を開発した．ここで，探索過程を分割する単位を時代と呼ぶ．これにより，探索過程に応じた適応が可能となる．A-SAGA の各エージェントのアルゴリズムを**図 1** に，A-SAGA 全体のアルゴリズムを**図 2** に示す．これらのアルゴリズム内におけるループを区別するため，図 1 において項目 3. から 8. までを 1 度繰返すことを 1 エージェント内世代，図 2 において項目 3. から 9. までを 1 度繰返すことを 1A-SAGA 世代と呼ぶ．

A-SAGA では，通常のメタ GA のパラメータ以外に以下のパラメータを与える．

1. 個体群受け取り
2. 個体群評価
3. 個体群選択
4. 個体群交叉
5. 個体群突然変異
6. 個体群評価
7. 累積評価回数が規定値に達していれば個体移民
8. 累積評価回数が規定値に達していなければ 3 へ
9. 個体群引き渡し

図 1 A-SAGA エージェントアルゴリズム
Fig. 1 The algorithm of the agents of A-SAGA.

1. エージェント群の初期化
2. 個体群の初期化
3. 次の時代のエージェントへの個体群引き継ぎ
4. エージェント群による探索
5. 最後の時代でなければ 3 へ
6. エージェント群選択
7. エージェント群交叉
8. エージェント群突然変異
9. 繰返し回数が規定値に達していなければ 2. へ

図 2 A-SAGA アルゴリズム
Fig. 2 The algorithm of A-SAGA.

E_{m} : 1A-SAGA 世代ごとの評価回数
G_{i} : 1A-SAGA 世代ごとの時代数
N_{a} : 1 時代ごとのエージェント数

A-SAGA の各エージェントに対してパラメータベクトル $\mathbf{v} = (e_{\mathrm{p}}, p_{\mathrm{m}}, p_{\mathrm{c}}, l)$ を与える．

e_{p} : 1 エージェント内世代ごとの評価回数
p_{m} : 突然変異率
p_{c} : 交叉率
l : 線形スケーリング係数

各時代には別のエージェント群が配置される．最初の時代のエージェントだけが個体群の初期化を行い，その個体群が次の時代のエージェントに引き継がれる（**図 3**）．A-SAGAβ ではパラメータにより個体数が変化したが，A-SAGA ではつねに最大の個体数が与えられる．ただし，1 エージェント内世代ごとに評価されるのは e_{p} 個体のみで，各エージェントに与えられる評価回数の合計は $E_{\mathrm{m}}/G_{\mathrm{i}}N_{\mathrm{a}}$ で一定である．

A-SAGAβ において，E_{m} の評価回数と N_{a} のエージェントを用いて探索する場合，各エージェントにはそれぞれ $E_{\mathrm{m}}/N_{\mathrm{a}}$ の評価回数が与えられる．A-SAGA においてこれをさらに G_{i} の時代に分割する場合，各時代には N_a のエージェントが配置され，各エージェントにはそれぞれ $E_{\mathrm{m}}/N_{\mathrm{a}} \cdot G_{\mathrm{i}}$ の評価回数が与えられる．

ここでエージェントは自分の配置された時代だけを認識し，次の時代のエージェントに個体群を引き渡す以外には他の時代のエージェントに干渉することはない．また，全体をメタ的に解釈するエージェントは存在せず，すべてのエージェントは同じ能力を持っている．別の時代に配置されたエージェント同士は隔離され，時代ごとに別々のメタ GA を用いてパラメータ探索を行う．このことから，序盤高効率エージェントと終盤高効率エージェントが比較されることはない．同時に時代の数と同じだけのメタ GA を動作させることになるが，エージェント適応度を与えるために必要となる評価回数の合計は同じであるために，評価のためにかかる計算量も同じである．

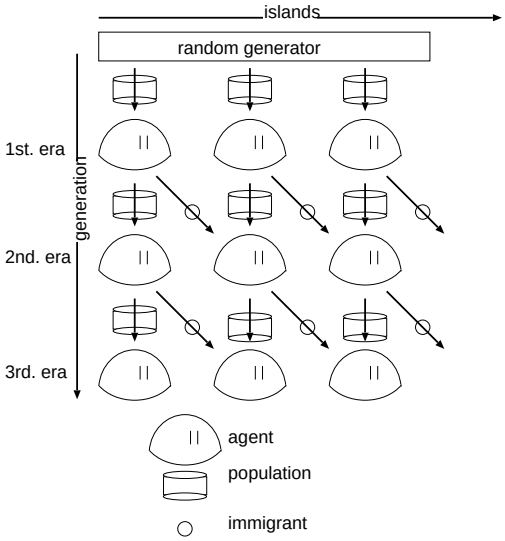

図 3 A-SAGA 概念図
Fig. 3 The conceptual model of A-SAGA.

4. 実　　験

4.1 実験条件

A-SAGA の探索効率を評価するために，式 (1) に示す Rastrigin 関数と 51 都市巡回セールスマン問題（Traveling Salesman Problem，以下 TSP）eil51 [12] を用いて実験を行った．いずれも最小化問題である．ただし，10 変数の Rastrigin 関数を用い，各変数を 16 ビットのグレイコードで表現した．TSP において，突然変異は 2 都市の交換を用いた．最適解の個体適応度は，Rastrigin 関数が 0，TSP eil51 が 426 であることが知られている．

$$f(x_1, x_2, \ldots, x_n) = 10n + \sum_{i}^{n} (x_i^2 - 10\cos(2\pi x_i)) \quad (1)$$

$$-5.12 < x_i \leq 5.12$$

探索効率の向上を計測するために，それぞれの A-SAGA 世代で求められたエリート個体の個体適応度を計測した（実験 1）．20A-SAGA 世代まで 2000 試行の実験を行った．なお，定義から $G_{\mathrm{i}} = 1$ のときの A-SAGA は A-SAGAβ と同じ性能となる．実験では，以下のパラメータを用いた．

E_{m} : 20000
G_{i} : 1, 10, 20
N_{a} : 10

比較対象として，同じ評価回数を与えたメタ GA に対して同様の計測を行った．筆者らの知る限り，適応度の評価方法と遺伝的演算子の定義を与えるだけで多数のパラメータを同時に適応させることができるという特徴を持つ適応 GA は，メタ GA だけである．

メタ GA は複数のパラメータを同時に適応させることができるが，パラメータを適応させるためのパラメータを手作業で与えなければならない．公正な比較を行うために，メタ GA に与えるパラメータは，あらかじめ手作業により最適化して効率を高めた．このメタ GA を調整メタ GA と呼ぶ．調整メタ GA は 10 回 IGA を実行させるごとにそれらの結果を元に交叉や突然変異を行う．調整メタ GA に与えるパラメータを選ぶために，以下の中から総当たりで調べ，最も探索効率の良かったものの周辺をさらに手作業で調査した．

- メタ GA の突然変位率：0，0.1，0.2，0.3，0.4，0.5，0.6，0.7，0.8，0.9，1 の 11 段階
- メタ GA の交叉率：0，0.2，0.4，0.6，0.8，1 の 6 段階
- メタ GA の線形スケーリング係数：1，2，3，4，5，6，7，10，20，50，100，200，500 の 13 段階

A-SAGA は，探索過程を時代で分割することによって，探索過程に応じたパラメータベクトルの適応がなされることを狙っている．これを確かめるため，20A-SAGA 世代までパラメータベクトルを適応させたとき，エージェントが時代ごとにどのようなパラメータベクトルを持っているかを計測した（実験 2）．

4.2 実験結果

図 4，**5** はパラメータ適応による探索効率の上昇を示すグラフである（実験 1）．これらのグラフにおいて，$G_{\mathrm{i}} = 10, 20$ の結果は，ほぼ重なっている．横軸が A-SAGA 世代，縦軸が各 A-SAGA 世代におけるエリート個体の個体適応度である．調整メタ GA の 1 世代は 10A-SAGA 世代に相当する．**図 6** は，図 5 における $G_{\mathrm{i}} = 1, 10, 20$ の結果の違いを明確にするために，A-SAGA の結果のみを抜粋したものである．

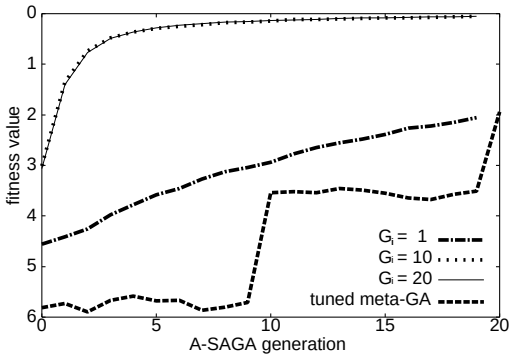

図 4 Rastrigin 個体適応度
Fig. 4 Rastrigin individual fitness.

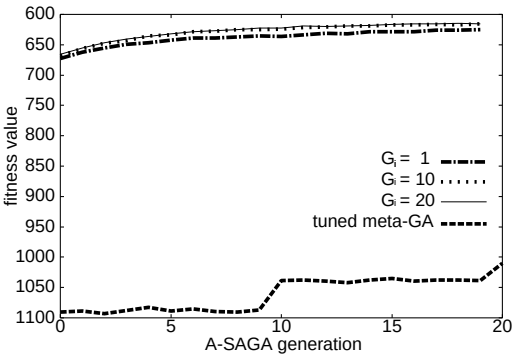

図 5 TSP eil51 個体適応度
Fig. 5 TSP eil51 individual fitness.

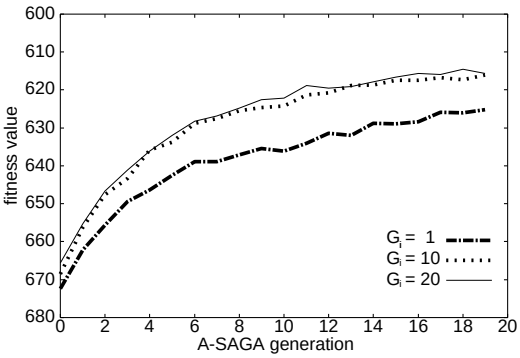

図 6 TSP eil51 個体適応度 (2)
Fig. 6 TSP eil51 individual fitness (2).

図 7，**8**，**9**，**10** は，20A-SAGA 世代目において得られた各エージェントのパラメータの平均を示す（実験 2）．横軸は時代の推移である．時代ごとに平均をとっているため，$G_{\mathrm{i}} = 1$ のときは値が一定である．図 7，9 は p_{m}，図 8，10 は e_{p} のパラメータの平均を示す．

図 4，7，8 は Rastrigin 関数による実験結果であり，図 5，9，10 は TSP eil51 による実験結果である．

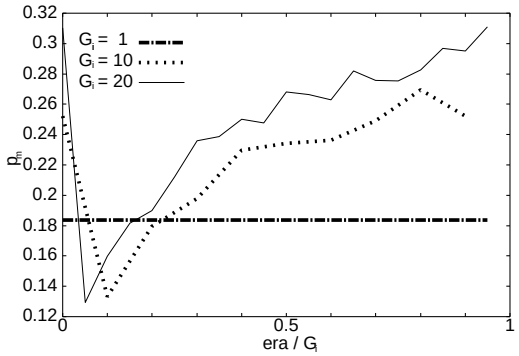

図 7 Rastrigin p_{m}
Fig. 7 Rastrigin p_{m}.

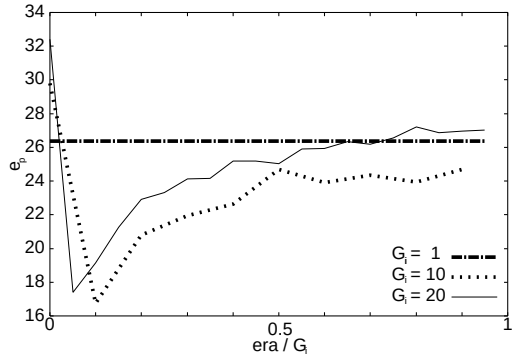

図 8 Rastrigin e_{p}
Fig. 8 Rastrigin e_{p}.

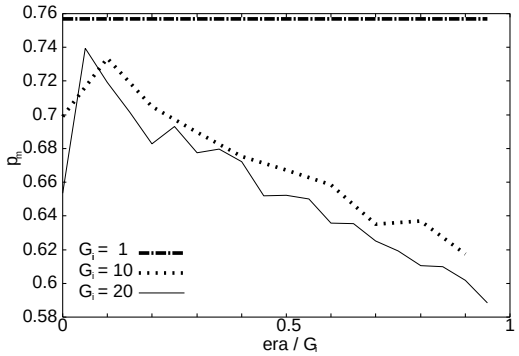

図 9 TSP eil51 p_{m}
Fig. 9 TSP eil51 p_{m}.

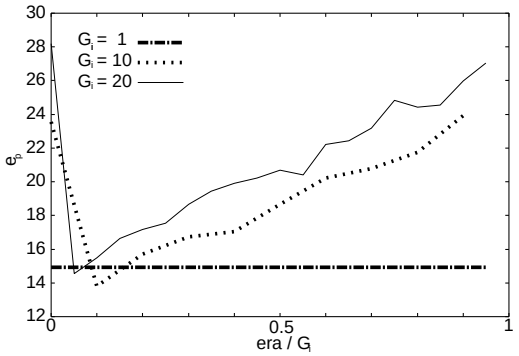

図 10 TSP eil51 e_{p}
Fig. 10 TSP eil51 e_{p}.

4.3 考　　察

4.3.1 実験 1 に関する考察

図 4 から，いずれの時代においても A-SAGA は，A-SAGA 世代を経るごとに探索効率が向上することが分かる．特に，$G_{\mathrm{i}} = 10, 20$ の場合に探索効率が高い（これらの結果はグラフ上で重なっている）．しかし，$G_{\mathrm{i}} = 1$ の場合，20A-SAGA 世代目で調整メタ GA に逆転されている．このことから，時代で分割することが有効に働いていると考えられる．調整メタ GA に与えたパラメータは，突然変異率が 0.01，交叉率が 1，線形スケーリング係数が 100 であった．線形スケーリング係数が高い理由は，調整メタ GA では 2 世代分しかパラメータベクトルの探索が行えないからであると考えられる．

ここで，A-SAGA 世代が 0，つまり，まったくパラメータベクトルを適応させていない場合においても調整メタ GA より A-SAGA の方が性能が良く，また，A-SAGA においては G_{i} が大きい方が探索効率が良い．この理由として，最初にランダムな環境が与えられるため，その中でたまたま良いパラメータを持つエージェントが存在する機会は，エージェントの数（ここでは $G_{\mathrm{i}} \times N_{\mathrm{a}}$）が多いほうが多く，そのため環境分散型並列 GA と同様の効果が出ていると考えられる．

図 5 から，A-SAGA はいずれの場合においても調整メタ GA に比べて探索効率が高い．最適解の個体適応度 426 には及ばないが，人間がパラメータ調整を行った予備実験の IGA が示した個体適応度 602 に近い値を示している．

図 6 から，$G_{\mathrm{i}} = 10, 20$ の場合の方が $G_{\mathrm{i}} = 1$ の場合よりも探索効率が高いことが分かる．Rastrigin 関数の場合と同様，時代で分割することが有効に働いていると考えられる．

4.3.2 実験 2 に関する考察

図 7 では，$G_{\mathrm{i}} = 10, 20$ のとき，探索が進むにつれて突然変異率が上昇している．この理由は，局所解から抜け出すために，探索終盤には高い突然変異率が必要であるためと考えられる．一方，Bäck の適応 GA では，探索終盤には突然変異率がきわめて低くなっており，探索効率が悪くなっていた[4]．

図 9 では，探索が進むにつれて突然変異率が低下し

ている．この理由は，ここで用いている突然変異オペレータがランダムで選ばれた 2 都市の交換であることが原因と考えられる．この突然変異オペレータで経路を改善できる個体は序盤には豊富であるが，いったん，この経路の改善がなされた場合，同じ 2 都市を再度交換した場合は必ず経路の改悪になる．したがって，この突然変異オペレータで改善すればするほど改善できる個体が減少し，それにつれて必要な突然変異率も低下していると考えられる．

図 7 と図 9 を比べると，パラメータ適応の様子が異なる．このことから，性質の違う問題に対しても，A-SAGA はパラメータを適応させることにより探索効率を高めていると考えられる．

図 8 と図 10 は良く似た傾向を示している．e_{p} は大きな初期値から急減し，その後徐々に上昇している．初期値が大きい理由は，初期個体群はランダムに与えられるために，少数の個体だけを用いて探索するよりも，多数の個体の中から有望な個体を探した方が効率が良いためと考えられる．初期値から e_{p} が急減する理由は，そこで見つけられた最良の個体の周辺のみを繰り返し探索するためと考えられる．その後 e_{p} が上昇する理由は，最良個体周辺の探索だけでは局所解に陥ってしまうため，その後は徐々に他の個体を取り込むことによって多様性を確保し，初期収束を避けているためと考えられる．

4.3.3 その他の考察

メタ GA では，適切なパラメータベクトルを用いて短く探索したときよりも，悪いパラメータベクトルを用いているが長く探索したときの方が良い結果が得られる場合がある．たとえば，図 5 において，$E_{\mathrm{m}} \times$ A-SAGA 世代数 $= 20{,}000 \times 20 = 400{,}000$ の評価回数を費やして，平均 615 の個体適応度を得た．しかし，$E_{\mathrm{m}} = 40{,}000$ とすると，たった 1A-SAGA 世代で平均 600 弱の個体適応度が得られた．A-SAGA の改善法としては，E_{m} を自動的に決定するアルゴリズムの開発があげられる．

我々は，実験により得られたパラメータを，Goldberg らのコントロールマップ[8] の概念に当てはめて評価するため，手作業によりコントロールマップに関する具体的な数値を得ようと試みた．しかし，10 変数 Rastrigin 関数に島モデル GA を適用して評価実験を行ったところ，コントロールマップに関する数値を得ることができなかった．具体的には，GA では淘汰圧が強すぎると cross-competitive（淘汰圧が高過ぎて多様性が失われてしまう状態）になってしまうと述べられているが，最適なスケーリング係数（淘汰圧を決定する）が非常に大きな値になり，値を大きくすることによって性能が下がる境界を観測できなかった．このことにはいくつかの要因があると考えられる．

- Goldberg らの研究では島モデルでない GA が分析されていたが，我々は島モデル GA を用いた．この場合では，それぞれの島において局所解に収束したとしても，異なる局地解に収束したのであれば相互作用を通じて最適解を導く場合があると考えられる．
- Goldberg らの研究では，最適解を見つけられるか否かだけに注目し，局所最適解の存在しない one max problem が用いられていた．しかし，我々は，準最適解であっても適応度が十分高ければそれを評価し，どれだけ高い適応度の平均値を得ることができるかを評価していた．つまり，評価基準が異なっているため，単純な比較ができない．

以上のことから，文献 8) のモデルは我々の実験結果に当てはまらないと考えられる．

5. おわりに

本論文では，エージェント指向の手法によりメタ GA と環境分散型並列 GA を組み合わせた A-SAGA を提案した．A-SAGA は探索過程に応じて，既存のほとんどの適応 GA を上回る 4 つのパラメータを同時に比較的少ない計算量で適応させることができた．今後の課題として，E_{m} を自動的に決定するアルゴリズムの開発，エージェントに対する GA 以外の進化アルゴリズムの利用，他のエージェントの行動も視野に入れて知的に振る舞うエージェントを開発などがあげられる．

参考文献

1) Bäck, T.: *Self-adaptation in genetic algorithms*, F.J. and Varela, P.B. (Eds.), *Proc. 1st European Conference on Artificial Life*, pp.263–271 (1992).
2) Cant'u-Paz, E.: A Survey of Parallel Genetic Algorithms, *Calculateurs Pralleles, Reseaux et Systems Repartis*, Paris, Hermes, Vol.10, No.2, pp.141–171 (1998).
3) Espinoza, F., Minsker, B.S. and Goldberg, D.: A Self-Adaptive Hybrid Genetic Algorithm, *Proc. Genetic and Evolutionary Computation Conference*, San Francisco, Morgan Kaufmann Publishers, p.759 (2001).
4) Glicman, M.R. and Sycara, K.: Reasons for Premature Convergence of Self-Adapting Mutation Rates, *Proc. Congress on Evolutionary*

Computation, pp.62–69 (2000).
5) Goldberg, D.: *Genetic Algorithm in Search, Optimization, and Machine Learning*, Addison-Wesley, Reading, MA (1989).
6) Goldberg, D.: Sizing populations for serial and parallel genetic algorithms, Schaffer, J.D. (Ed.), *Proceedings of the Third International Conference on Genetic Algorithms*, Morgan Kaufmann Publishers, pp.70–79 (1989).
7) Goldberg, D. and Deb, K.: A comparative analysis of selection schemes used in genetic algorithms, Rawlins, G.J.E. (Ed.), *Foundations of Genetic Algorithms*, Morgan Kaufmann Publishers, pp.69–93 (1991).
8) Goldberg, D., Deb, K. and Dirk, T.: Toward a better understanding of mixing in genetic algorithms, *Journal of the Society of Instrument and Control Engineers*, Vol.32, No.1, pp.10–16 (1993).
9) Kee, E., Airey, S. and Cye, W.: An Adaptive Genetic Algorithm, *Proc. Genetic and Evolutionary Computation Conference*, pp.391–397 (2001).
10) Krink, T. and Ursem, R.K.: Parameter Control Using the Agent Based Patchwork Model, *Proc. Congress on Evolutionaly Computation*, pp.77–83 (2000).
11) Miki, M., Hiroyasu, K., Kaneko, M and Hatanaka, I.: A Parallel Genetic Algorithm with Distributed Environment Scheme, *IEEE Proceedings of Systems, Man and Cybernetics Conference SMC '99*, pp.695–700 (1999).
12) http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/

(平成 14 年 10 月 8 日受付)
(平成 14 年 12 月 2 日再受付)
(平成 15 年 1 月 7 日採録)

村田　佳洋（学生会員）

1975 年生．2000 年奈良先端科学技術大学院大学情報科学研究科修士課程修了，現在博士課程在学中．遺伝的アルゴリズム，エージェント技術等の研究に従事．

柴田　直樹（正会員）

1974 年生．2001 年大阪大学大学院基礎工学研究科情報数理系専攻博士後期課程修了．2001 年 4 月より奈良先端科学技術大学院大学情報科学研究科助手．エージェント技術，遺伝的アルゴリズム，形式的設計検証等の研究に従事．

伊藤　　実（正会員）

1977 年，1979 年，1983 年にそれぞれ大阪大学基礎工学部卒業，基礎工学研究科博士前期課程修了，博士後期課程修了．1979 年より大阪大学基礎工学部助手．1986 年より大阪大学基礎工学部講師．1989 年より大阪大学基礎工学部助教授．1993 年 4 月より現在，奈良先端科学技術大学院大学情報科学研究科教授．関係データベース理論，オブジェクト指向データベースのアプリケーション，DNA プローブ等の研究に従事．ACM，IEEE 各会員．