数理モデル化と問題解決　41－6
（２００２．　９．　２０）

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

奈良先端科学技術大学院大学 情報科学研究科
〒６３０－０１０１ 奈良県生駒市高山町８９１６番地の５
e-mail {yosihi-m, n-sibata, ito}@is.aist-nara.ac.jp

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 では少数のパラメータしか適応させないものがほとんどであった．また，多数のパラメータを適応させる適応 GA もいくつかあるが，その大部分が大きな計算量を必要としていた．本稿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，多数のパラメータの組み合わせ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した．また評価実験を行い，４つのパラメータを同時に合理的な計算量で適応させることができた．

Agent Oriented Self Adaptive Genetic Algorithm

Yoshihiro Murata, Naoki Shibata, Minoru Ito

Graduate School of Information Science
Nara Institute of Science and Technology
8916-5, Takayama, Ikoma, Nara 630-0101 Japan
e-mail {yosihi-m, n-sibata, ito}@is.aist-nara.ac.jp

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 [4](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 の探索効率は与えるパラメータ (突然変異率，交叉率など) 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][2][5][6].

従来の適応 GA では少数のパラメータしか適応させないものがほとんどであった．また多数のパラメータを適応させる適応 GA もいくつかあるが，その大部分が大きな計算量を必要としていた．

本稿では，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 [7] 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．また評価実験によりその性能を評価する．

2　提案手法

2.1　従来手法の問題点と解決法

これまでのほとんどの適応 GA は１ないし２つのパラメータだけを適応させており，多数のパラメータを同時に適応させるものは少なかった．またメタ GA は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．

メタ GA の計算量が大きい原因は，パラメータベクトル１つを評価するために１度 GA を走らせね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，GA1 回あたりの計算量を減らさねばならない．

ところが計算量を減らすと 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 の適したパラメータベクトルと元の GA の適したパラメータベクトルは一般には一致しない．

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

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. 繰り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

2.2 エージェント指向の手法の導入

我々は環境分散型並列 GA[7] とメタ 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．ここではこれを A-SAGAβ と呼ぶ．

A-SAGAβ では計算量を減らした GA をエージェントとみなし，それぞれに異なるパラメータベクトルを与える．エージェントは移民を通じて共同して解を探索する．

A-SAGAβ は環境分散型並列 GA とみなすことができ，各々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できる．

A-SAGAβ ではメタ GA を用いてパラメータベクトルを適応させるため，各エージェントにとって最適なパラメータベクトルが大きく異なる場合にはうまく適応できないことが予測される．A-SAGAβ はエージェントがお互いに対称になるようリング状に配置し，各エージェントの置かれる環境を同じにしている．

2.3 暫定アルゴリズム

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．エージェント毎に一定の評価回数が与えられるが n が異なるため，繰り返し回数も異なる．

A-SAGAβ は，複数のエージェントを用いて解の探索を行い，各エージェントの探索効率を評価する．探索効率は，エージェントの持つエリート個体[1]の適応度の上昇値の累積値で評価されるが，移民によってより良い個体が他エージェントから提供された場合の上昇値は換算しない．それを各エージェントの適応度として扱い，メタ GA により良いパラメータベクトルを探索する．

[1]個体群の中で最良の適応度を持つ個体．

2.4 予備実験とそれに対する考察

我々は A-SAGAβ の評価実験を行ったが，結果は芳しくなかった (3.1節)．

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 [5]．A-SAGAβ は探索の終盤において探索効率が高いエージェントよりも，探索の序盤において探索効率が高いエージェントを高く評価するために，序盤に探索効率が高いエージェントばかりになってしまうのではないかと考えられる．

2.5 改良後のアルゴリズム

前節の考察を踏まえて，我々は 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 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，探索過程に応じたパラメータ適応を行うことが可能になる．

A-SAGA には通常のメタ GA のパラメータ以外に以下のパラメータが与えられる．

E_m : 1 メタ GA 世代あたりの評価回数
G_i : 1 メタ GA 世代あたりの時代数
N_a : 1 時代あたりのエージェント数

A-SAGA のエージェントにはパラメータベクトル $\mathbf{v} = (e_\mathrm{p}, p_\mathrm{m}, p_\mathrm{c}, l)$ が与えられる．

e_p : エージェント内 GA の 1 世代毎の評価回数
p_m : 突然変異率
p_c : 交叉率
p_c : 線形スケーリング係数

A-SAGA では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へと引き継がれていく (図 3)．A-SAGAβ ではパラメータにより個体数が変化していたが，A-SAGA では常に最大の個体数が与えられる．ただし GA1 世代毎に評価されるのは 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 $\frac{E_\mathrm{m}}{G_\mathrm{i} N_\mathrm{a}}$ で一定である．エージェントのアルゴリズムを図 1に，A-SAGA のアルゴリズムを図 2に示す．

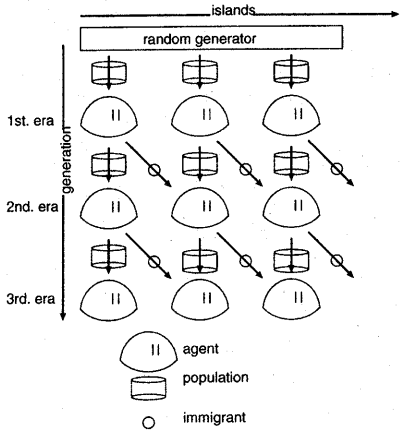

図 3: A-SAGA 概念図

3 実験

我々は A-SAGA の探索効率を評価するために，(1) 式に示す Rastrigin 関数と 51 都市巡回セールスマン問題 (Traveling Salesman Problem，以下 TSP)eil51[8] を用いて実験を行った．いずれも最小化問題である．Rastrigin 関数は 10 変数のものを用い，各変数は 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．

$$f(x_1, x_2, \ldots, x_n) = 10n + \sum_{i}^{n}(x_i^2 - 10\cos(2\pi x_i)) \quad (1)$$
$$-5.12 < x_i \leq 5.12$$

我々は探索効率の向上を計測するために，メタ GA 世代毎の合計の評価回数を一定にして A-SAGA を走らせ，それぞれの世代で求められたエリート個体の適応度を計測した (実験 1)．第 20 メタ GA 世代まで 2000 試行の実験を行った．なお，A-SAGA は $G_i = 1$ のときは A-SAGAβ と同じ性能となる．実験においてパラメータはいかのものを用いた．

E_m : 20000
G_i : 1, 10, 20
N_a : 10

A-SAGA では，各メタ GA 世代毎に個体群が初期化されて最初の時代のエージェント群に与えられ，時代毎に異なるエージェント群に引き継がれていく．従って時代の数，G_i の数だけパラメータベクトルが変化することになる．

我々は，第 20 メタ GA 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毎にどのようなパラメータの組み合わせを持っているのかを計測した (実験 2)．

3.1 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)．横軸がメタ GA 世代，縦軸が各メタ GA 世代におけるエリート個体の個体適応度である．

図 6,7,8,9は第 20 メタ GA 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．時代毎に平均をとっているため，$G_i = 1$ のときは値が一定である．図 6,8は p_m，図 7,9は e_p のパラメータの平均を示す．

図 4, 6, 7は Rastringin 関数による実験結果であり，図 5,8,9は TSP eil51 による実験結果である．

3.2 考察

図 4を見ると，いずれの時代においてもメタ GA 世代を経る毎に探索効率は向上しているが，G_i が 10,20 の場合の向上が早い (これらの結果はグラフ上で重なっている)．同様に図 5を見ると，図 4の場合ほど顕著ではないがやはり G_i が 10,20 であるときに探索効率の向上が早い．このことから探索過程を時代で分割することの効果があることがわかる．

図 4,5において，メタ GA 世代が 0，つまり全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 G_i が 10,20 の場合の探索効率が高い．最初にランダムな環境が与えられるために，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 $G_i \times N_a$) が多いほうが大きい．そのために環境分散型並列 GA[7] と同様の効果が出ているのではないかと考えられる．

図 6では，G_i が 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．これは局地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であると考えられる．従来の適応 GA では探索終盤には突然変異率が極めて低くなってしまっていた [3]．

図 8においては，探索が進むにつれて突然変異率が低下している．これはここで用いている突然変異オペレータが 2 都市の交換であるために，探索が進むと 2 都市の交換で改善することができる個体が減少していくことが原因であると考えられる．

図 7と図 9はは良く似た傾向を示している．まず最初に値が大きいが即座に激減し，その後徐々に上昇している．まず最初に値が大きい理由は，ランダムに得られた少数の個体だけを元に探索するよりも，すでにランダムに与えられたの個体の中から有望である個体を探したほうが効率が良いためであると考えられる．次に値が激減するのは，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を行うためであると考えられる．しかしそれだけでは即座に局地解に陥ってしまうために，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て初期収束を避けているのではないかと考えられる．

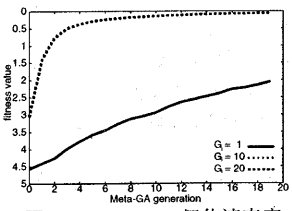

図 4: Rastringin 個体適応度

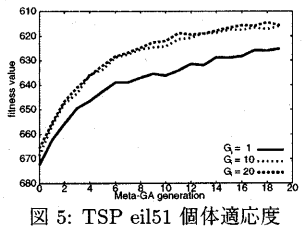

図 5: TSP eil51 個体適応度

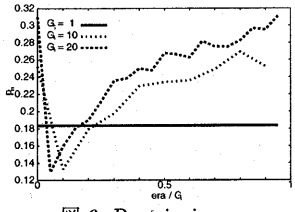

図 6: Rastringin p_m

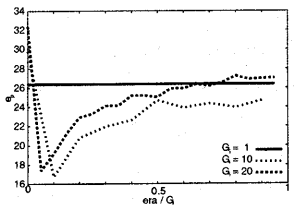

図 7: Rastringin e_p

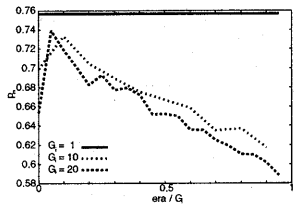

図 8: TSP eil51 p_m

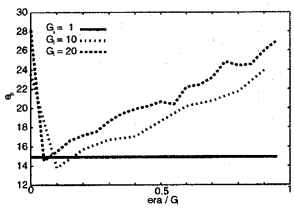

図 9: TSP eil51 e_p

メタ GA には，良いパラメータを見つけるよりも，悪いパラメータのままで長く探索したほうが良い結果が得られる場合がある．我々の A-SAGA にも同様であり，E_m の値をどうするかは人間の手に委ねられている．例えば図 5 において，$E_m \times$ メタ GA 世代数 $= 20000 \times 20 = 400000$ の評価回数を費やして，平均 615 の個体適応度を得ている．しかし $E_m = 40000$ とすると，たった 1 メタ GA 世代で平均 600 弱の個体適応度を得ることができた．我々の今後の課題として，探索過程を監視して E_m を自動的に決定するアルゴリズムの開発が挙げられる．

4 おわりに

我々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．今後の課題としてはメタ GA などの従来の手法と A-SAGA の性能を比較検討すること，E_m を自動的に決定するアルゴリズムを開発すること，エージェントに対して GA 以外のアルゴリズムの利用を許すこと，他のエージェントの行動も視野に入れて知的に振舞うエージェントを開発することなどが挙げられる．

参考文献

[1] T. Bäck, "Self-adaptation in genetic algorithms", In F.J.Varela, P. B., editor, Proceedings of 1st European Conference on Artificial Life, pp. 263–271, 1992.

[2] F. Espinoza, B. S. Minsker, and D. Goldberg, "A Self-Adaptive Hybrid Genetic Algorithm", Proceedings of the Genetic and Evolutionary Computation Conference, ,San Francisco, Morgan Kaufmann Publishers, 2001.

[3] M. R. Glicman, K. Sycara, "Reasons for Premature Convergence of Self-Adapting Mutation Rates", Proceedings of the Congress on Evolutionary Computation, pp. 62–69, 2000.

[4] D. Goldberg, "Genetic Algorithm in Search, Optimization, and Machine Learning", Addison-Wesley, Reading, MA, 1989.

[5] E. Kee, S. Airey and W. Cye, "An Adaptive Genetic Algorithm", Proceedings of the Genetic and Evolutionary Computation Conference, pp 391–397, 2001.

[6] T. Krink and R. K. Ursem, "Parameter Control Using the Agent Based Patchwork Model", Proceedings of the Congress on Evolutionaly Conputation, pp. 77–83, 2000.

[7] M. Miki, K. Hiroyasu, M. Kaneko and I. Hatanaka, "A Parallel Genetic Algorithm with Distributed Environment Scheme", IEEE Proceedings of Systems, Man and Cybernetics Conference SMC'99, pp. 695–700, 1999.

[8] http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/