成長する粒子群最適化法
Growing Particle Swarm Optimizer

宮川 英士 †, 斎藤 利通 †
Eiji Miyagawa, Toshimichi Saito
† 法政大学 工学部 情報電気電子工学科

Abstract— This paper presents an improved version of PSO having grow-and-reduce structure. When a particle is trapped into a local optimum, a new particle is born at a position away from the trap and is connected to some/all of existing particles. If a particle can not escape from the trap, the particle is deleted in order to suppress excessive swarm grows. Performing basic numerical experiments, the algorithm performance is investigated.

Keywords— **Particle Swarm Optimization, Optimization, Swarm topology, Expand-and-reduce ability.**

1 まえがき

Particle Swarm Optimization (PSO) は、様々な最適化問題を解くのに用いられている最適化アルゴリズムである [?]。本論文では、成長-収縮能力を持った新たな PSO アルゴリズムを紹介する [3]。このアルゴリズムでは、粒子の追加によるネットワークの成長を行う事でより多くの探索空間で解の探索を行える。また、過度のネットワークの拡張を防ぐために粒子の削除も導入した。粒子の削除を行わなずに追加を続けて行っていった場合、ネットワークの過度な拡張がされ計算時間の増加に繋がってしまう。この拡張-収縮能力を持たせた PSO に基本的な 3 つのネットワーク構造である全結合とリング構造、木構造を適用しアルゴリズムの基本特性を基本的な数値実験を通して検証する。

ネットワークの成長や収縮能力は、Self-Organizing maps(SOM) を含むいくつかの学習アルゴリズムにおいて重要な手法である [2]。

2 成長-収縮型アルゴリズム

Kennedy らが提案した基本的な PSO は、数多くの粒子が多次元空間上を動き回る最適化アルゴリズムである。粒子同士が自身の探索結果を交信し、よりよい解へと収束し最適解を探索する。しかし、山が複数存在する多峰性関数では準最適解へと群全体が収束して捕らえられてしまう事がある。PSO には収束した場所が最適解か準最適解を判別する機能がないため、一度捕らわれてしまうと脱出は困難である。

ここで、粒子の追加と削除を行う PSO の探索アルゴリズムを提案する。この成長-収縮型の粒子群最適化法 (GRPSO) では、粒子群がどこかへ収束した場合に他の探索空間で解の探索を行う事が出来る。そのため、適切なパラメータの設定なしに準最適解から脱出できる可能性がある。以下で関数 F の最小値を求める問題における GRPSO の探索アルゴリズムを定義する。ここでは、粒子群のネットワーク構造には木構造を用いた。

Step 1: 粒子数 N を設定し、各粒子を探索空間内にランダムに配置。位置ベクトル $\vec{x}_i(t)$ と速度ベクトル $\vec{v}_i(t)$、カウンター $c_i(t)$ を $t = 0$ で初期化する。

Step 2: 各粒子について、探索結果内で最良の評価値 $pbest_i(t)$ と $F(x_i(t))$ の値を比較:
if $F(x_i(t)) < pbest_i(t)$ then
(a) $pbest_i(t) = F(x_i(t))$ and (b) $x_{pbest_i(t)} = x_i(t)$
$pbest_i(t)$ の値と $pbest_i(t-1)$ を比べ、改善率によってカウンター $c_i(t)$ の値を変更:

$$\begin{array}{ll} c_i(t) = c_i(t) + 1 & \text{if } \ |F(x_i(t)) - pbest_i| < \epsilon \\ c_i(t) = 0 & \text{if } \ |F(x_i(t)) - pbest_i| \geq \epsilon \end{array} \tag{1}$$

ただし、ϵ は十分に小さな数とする。

Step 3 (expansion): カウンター c_i の値が増加しているならば図 1 (a) に示すように粒子群が準最適解または最適解へ収束している可能性がある。カウンター c_i の値がしきい値 T_{int} を超えた時、そのカウンターを持つ i 番目の粒子の近傍に新たな粒子を追加する。図 1 (b) ∼ (c) に示すように、追加粒子と交信し合う事で収束地点以外の情報を得る事が出来る。i 番目の粒子が捕らわれている時、

$$x_{new}(t) = x_i(t) + r \quad \text{if} \ \ c_i(t) > T_{int}. \tag{2}$$

ただし r は乱数とし、各粒子番号は以下のように再設定される: $j = j + 1$ for $i < j$ and $x_{new} = x_{i+1}$。Let $N = N + 1$. カウンターの値も全て初期化される : $c_i(t) = 0$ for all i。この粒子の追加は、準最適解からの脱出に効果的である。

Step 4 (reduction): 新しい粒子の追加をしてから次の粒子の追加までに $gbest(t)$ の値に変化がない時、次に粒子を追加する際に同時に粒子の削除を行う (図 1 (d)) 。粒子の削除は、過度のネットワークの拡張による計算時間の肥大化を防ぐ。

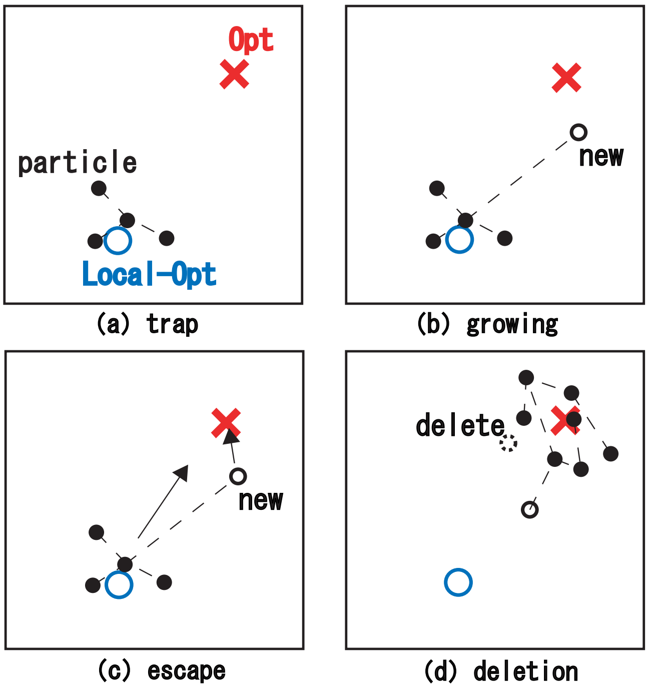

図 1: 粒子の追加と脱出

Step 5: 各粒子について、自身とその両近傍の粒子の探索結果内で最良の評価値 $lbest_i(t)$ と $F(x_i(t))$ の値を比較:
if $F(x_i(t)) < lbest_i(t)$ then
(a) $lbest_i(t) = F(x_i(t))$ and (b) $x_{lbest_i(t)} = x_i(t)$
Step 6: 各粒子の速度ベクトル $v_i(t)$ を変更:

$$v_i(t) = Wv_i(t)+\rho_1(x_{pbest_i(t)}-x_i(t))+\rho_2(x_{lbest_i(t)}-x_i(t)) \quad (3)$$

ただし ρ_1、ρ_2 は一様乱数とする。W は慣性項で、今回は以下の式で与えられるものとする。

$$W = W_{max} - \frac{W_{max}-W_{min}}{t_{max}} \times t \quad (4)$$

Step 7: 次式に従って粒子の移動を行う:

$$x_i(t+1) = x_i(t) + v_i(t).$$

Step 8: Let $t = t+1$。時刻 t が最大探索回数 t_{max} に至るまで、STEP2 に戻り探索を行う。

3 数値実験

前章で提案した GRPSO の性能を検証するために基本的な数値実験を行った。ベンチマーク問題として、準最適解を複数持つ複雑な多峰性関数である Griewangk's function に対して GRPSO の適用を行った。

$$f(x) = 1 + \sum_{i=1}^{m} \frac{x_i^2}{4000} - \prod_{i=1}^{m}(\cos(\frac{x_i}{\sqrt{i}})) \quad (5)$$
$$\min(f(x)) = f(0,\dots,0) = 0$$

上記の関数を 30 次元 ($m = 30$)、粒子の初期配置を悪い状態で探索を行った。100 回探索を繰り返して得た結果を表 1 に示す。粒子のネットワーク構造は全結合とリング構造、木構造を適用した。

表 1: Griewangk's function に対する結果。SR=成功率、#ITL=探索回数、#link=リンク数、#PCL=粒子数

Methods	SR	#ITL	#Link	#PCL
完全グラフ	0	500	780	40.0
完全 GRPSO	50	364	488	42.9
リング	0	500	40.0	40.0
リング GRPSO	76	243	29.0	39.6
木	0	500	39.0	40.0
木 GRPSO	100	130	27.9	39.2

成長構造を持たない PSO では初期位置に大きく依存してしまい、最適解の探索が困難だった。しかし GRPSO では追加粒子の情報を使うことで、最適解を高い確率で探し当てている。また、木構造を適用した GRPSO では全ての探索で最適解へと収束した。

4 むすび

粒子群最適化法に成長-収縮能力を持たせたアルゴリズムを提案した。本アルゴリズムは成長機能によって準最適解から効果的に脱出が行え、収縮能力により計算時間を抑圧する事が出来る。また、今回の数値実験では木構造が最適解探索に効果的であった。本論文はまだ初期段階であり、以下に示すような課題が挙げられます。:

1) 各パラメータの役割の分析、
2) 効果的な粒子のネットワーク構造の分析、
3) 各パラメータの自動設定、
4) より複雑なベンチマーク問題への適用、
5) 実用的な問題への適用。

参考文献

[1] J. Kennedy and R. Eberhart, Particle Swarm Optimization, Proc of IEEE/ICNN, pp.1942-1948, 1995

[2] T. Oshime, T. Saito and H. Torikai, ART-based parallel learning of growing SOMs and its application to TSP, LNCS, 4232, Springer, Neural Information Processing, I, pp. 1004-1011, 2006

[3] E. Miyagawa and T. Saito, Particle Swarm Optimizers with Grow-and-Reduce Structure, Proc of CEC, pp.3975-3980, 2008