2014 International Symposium on Nonlinear Theory and its Applications
NOLTA2014, Luzern, Switzerland, September 14-18, 2014

A Growing Insensitive Particle Swarm Optimizer for Identification of Multi-Solutions

Takuya Takemura, Takumi Sato, Kazuki Maruyama and Toshimichi Saito
Department of Electrical and Electronics Engineering, HOSEI University
Koganei-shi, Tokyo, 184-8584 Japan, tsaito@hosei.ac.jp

***Abstract*—This paper presents the growing insensitive particle swarm optimizer (GIPSO) for multi-solution problems. Especially, we consider the case where the number of solutions is unknown. The GIPSO uses ring-topology and has an insensitive parameter. The number of particles can increases and the swarm can grow. If parameter velues are selected suitably, the GIPSO can identify all the solutions and can clarify the number of solutions. Performing fundamental numerical experiments, we investigate the algorithm capability.**

I. Introduction

The particle swarm optimizer (PSO) is a population-based optimization method inspired by flocking behavior of living beings [1]. The living beings are modeled by a particle-swarm where each particle is characterized by position and velocity. The positions correspond to potential solutions and is evaluated by an objective function. The particles search desired optimal solution(s) based on inter-particle communication. The PSO does not require gradient information of objective function. The PSO is a simple in concept, is fast, and has been applied to various particle/potential engineering applications. The applications include design of circuits and neural networks [2]-[5].

For a simple single solution problem, the PSO is suitable for global search. Because it can find an optimal solution by few particles even if the search apace is vast. However, standard PSOs are not suitable for multi-solution problems (MSP [7]-[**?**]) where particles are often trapped into partial/local solutions.

In this paper, we present the growing insensitive particle swarm optimizer (GIPSO) for MSP. Especially, we consider the case where the number of solutions is unknown. The GIPSO is defined on a particle swarm of ring-topology and has an insensitive parameter. The number of particles can increases and the swarm can grow. If parameter velues are selected suitably, the GIPSO can identify all the (approximate) solutions and can clarify the number of solutions. Also, the GIPSO includes no random parameters: it is deterministic [9]-km2. Such a deterministic system is convenient in motion analysis and reproducibility performance evaluation. This system is distributive, and each particle motion is limited in lattice point. Performing fundamental numerical experiments, we investigate the algorithm capability in the MSP. Although this paper considers an elementary MSP, the results may be developed into a variety of engineering applications.

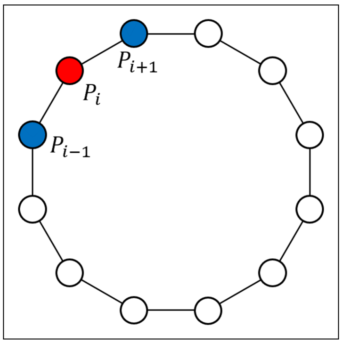

Fig. 1. Ring topology.

II. Ring-type Insensitive PSO

The objective function for the GIPSO is defined by

$$F_A : S_A \to R_+, S_A = \{(x_1, x_2)|X_L \leq x_i \leq X_R, i = 1, 2\} \quad (1)$$

where S_A is a two-dimensional search space and R_+ denotes positive reals. Let $x \equiv (x_1, x_2)$. Assuming F_A has plural minima, the multi solutions x_s^i are defined by

$$\begin{aligned} &F_A(x_s^i) = 0, \\ &x_s^i \equiv (x_{s1}^t, x_{s2}^t) \in S_A, i = 1 \sim N_A \end{aligned} \quad (2)$$

where $i = 1 \sim N_A$ and N_A is the number of solutions. Although we define the GIPSO for considers two-dimensional objective function F_A, the definition can be generalized by replacing x with n-dimensional vectors.

The GIPSO employs N^t particles at search step t. Note that the number of particles N^t is time-vaiant. let P^t denote the particle swarm at . Let P_i^t be the i-th particle. The i-th particle is characterized by its position x_i^t and velocity v_i^t where $i \sim N^t$. The update of the particle is based on the personal best ($Pbest_i$) and local best ($Lbest_i$) [1] for the objective function F_A. The $Pbest_i$ gives the best value in the past history of P_i. $Lbest_i$ is the best of the personal best in the neighbor of P_i. The neighbor particles are given depending on the topology of the particle swarms. We adopt the ring topology as shown in Fig. 1. For a particle, the both sides particles are the neighbors [2]. The GIPSO includes insensitive parameter that can make update of the $Lbest_i$ insensitive. In this paper, we assume

[1]In standard PSOs, the global best is used of the local best.
[2]In the compelte graph, all the particles are the neighbors.

that the number of solutions N_A is unknown. Our purpose is to identify positions of all the approximate solutions and to clarify the the number solutions. The GIPSO is defined by the following 7 steps.

STEP 1(Initialization): Let search step $t = 0$ and let the initial number of particles is $N^t = N_0$. Particle positions x_i^t and velocities v_i^t are initialized where $i = 1 \sim N$. Personal bests and local bests are initialized: $\vec{x}_{pbest_i} = \vec{x}_{lbest_i} = \vec{x}_i^t$. The number of approximate solutions is initialized: $k = 0$. The number of areas of approximate solutions is initialized: $S = 0$.

STEP 2 (Approximate solutions): If the i-th particle position satisfies

$$F(\vec{x}_i^t) < C_A \tag{3}$$

then x_i^t is declared as an approximate solution. The approximate solution is labelled by a_k. (If this is the first approximate solution then $S = 1$).

STEP 3 (Area judgement): If a_k is not included in an area of existing approximate solutions then a new area is generated.

$$S \leftarrow S + 1 \text{ if } |\vec{a}_k - \vec{a}_j| > r \text{ for } j < k \tag{4}$$

where $|\cdot|$ denote the Euclidean distance and the parameter r decides the approximate solution area. We have used the descending sort algorithm in the judgement. Let $k \leftarrow k + 1$.

STEP 4 Personal and local bests are updated:

$$\begin{array}{ll} \vec{x}_{pbest_i}^t \leftarrow \vec{x}_i^t & \text{if } F(\vec{x}_i^t) < F(\vec{x}_{pbest_i}^t) \\ \vec{x}_{lbest_i}^t \leftarrow \vec{x}_{pbest_i}^t & \text{if } F(\vec{x}_{pbest_i}^t) < (\vec{x}_{lbest_i}^t) - \epsilon^t \end{array} \tag{5}$$

$$\epsilon^t \leftarrow \epsilon^0 \times \frac{t_{max} - t}{t_{max}}$$

where ϵ^t is an time-variant parameter which controls insensibility. ϵ^0 is an initial value and t_{max} is the maximum step in STEP 7. If ϵ^t is smaller then the update occurs frequently. The case $\epsilon^t = 0$ corresponds to normal PSOs. This insensitivity may be effective to avoid trap into local/partial solutions and to enlarge the swarm diversity.

STEP 5: Position and velocities are updated:

$$\begin{array}{l} \vec{v}_i^{t+1} \leftarrow w \times \vec{v}^t + c \times (\vec{x}_{lbest_i}^t - \vec{x}_i^t) \\ \vec{x}_i^{t+1} \leftarrow \vec{x}^t + \vec{v}_i^{t+1} \end{array} \tag{6}$$

where w and c are deterministic parameters. Note that our GIPSO includes no random parameters.

STEP 6 (Growing Swarm): At every Δt steps, the number of particle increases:

$$N^t \leftarrow N^t + \Delta N \text{ at } t = n\Delta t \tag{7}$$

where $\Delta t = t_{max}/M$ and The maximum particle number is $N_0 + M \times \Delta N$.

STEP7 Let $t \leftarrow t + 1$, return to **STEP2** and repeat until $t = t_{max}$.

After the algorith is terminated, S gives the number of approximate solution areas and a_i, $i = 1 \sim k$ gives the approximate solutions. Note that a_i in the comomn approximate solution area corresponds to one approximate solution of the MSP.

III. Numerical Experiments

We consider the following fundamental objective function:

$$\begin{array}{l} F_A(x) = \cos(2\pi x_1) + \cos(2\pi x_2) + 2 \\ x \equiv (x_1, x_2), \ -1.9 < x_1 < 1, \ -1.9 < x_2 < 1 \end{array} \tag{8}$$

As shown in the contour map in Fig. 2, this function has 9 solutions ($N_A = 9$). The pupose of MSP is identification of all the 9 approximate solutions. We apply the GIPSO to this MSP. For simplicity, we discretize the search space S_A onto $N_C \times N_C$ lattice points: the particle positions are discretize on the lattice points where the function is . After trial-and-errors, parameters are selected as the following:

$$\begin{array}{l} w = 0.7, \ c = 0.7. \ C_A = 0.02, \ N_0 = 10, \ t_{max} = 50 \\ \Delta t = 10 \ (M = 5), \ \Delta N = 20, \ \epsilon_0 = 0.5, \ N_C = 128. \end{array} \tag{9}$$

Performing numerical experiment, we have confirmed that the GIPSO can identify all the 9 approximate solutions. Figure 3 shows snapshots where nine red circles are approximate solution areas and blue points in them are approximate solutions.

In order to compare the camability of GIPSO, we apply the other three algorithms to the same MSP:

RPSO: PSO on the ring topology. It is defined by removing growing swarm in Step 6 and insensitive parameter in Step 4.

IPSO: PSO with insensitivity. It is defined by removing growing swarm in Step 6.

GPSO: PSO with growing swarm. It is defined by removing insensitive parameter in Step 4.

These three algorithm can not identify all the nine approximate solutions as suggested in Figs. 4 to 6.

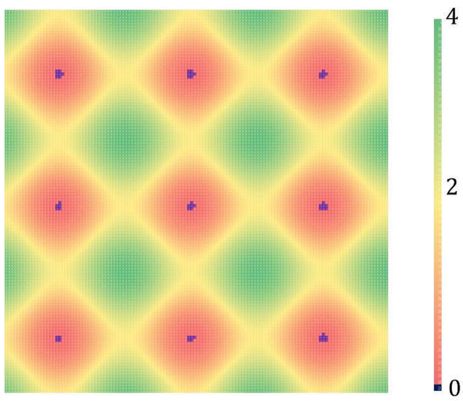

Fig. 2. Contour map of the MSP objective function.

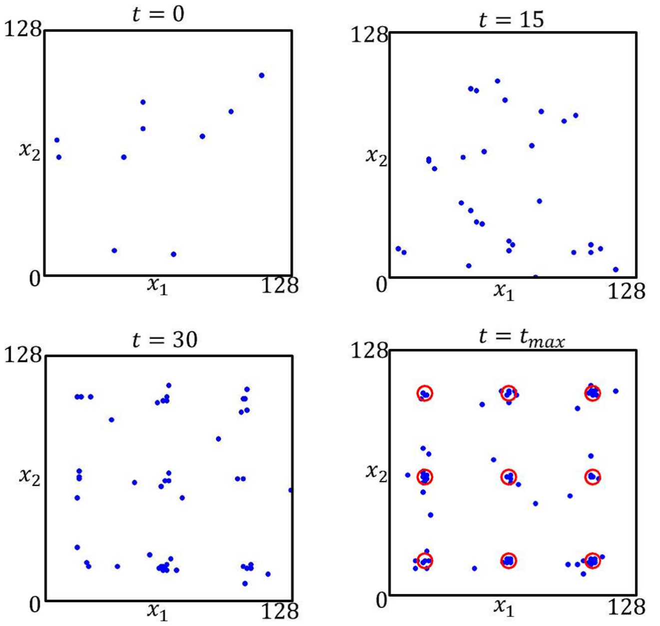

Fig. 3. Snapshots in search process of GIPSO. Red circles denote identified solution areas.

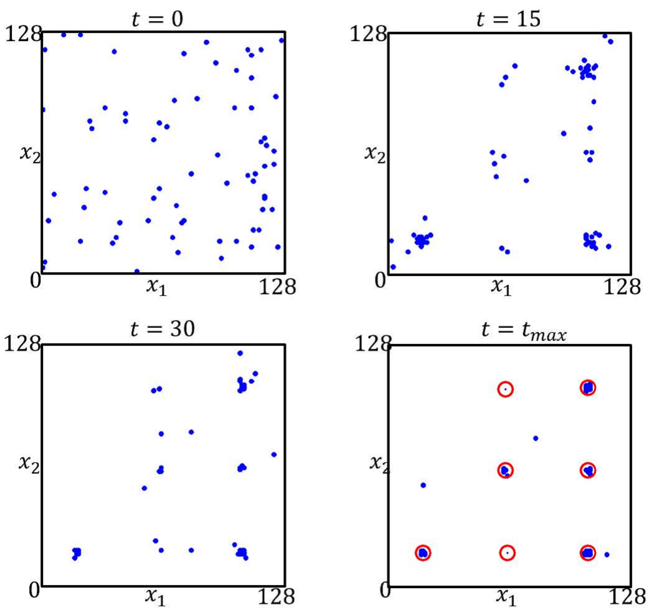

Fig. 4. Snapshots of RPSO ($N = 60$, $\epsilon^t = 0$)

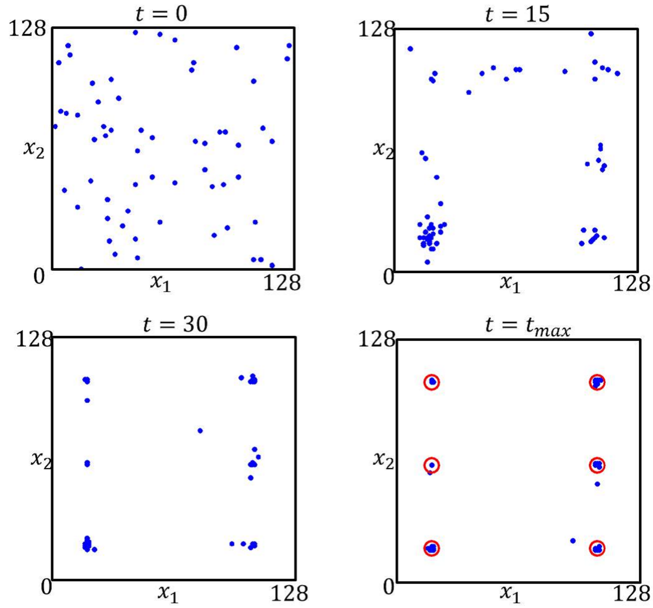

Fig. 5. Snapshots of IPSO ($N = 60$)

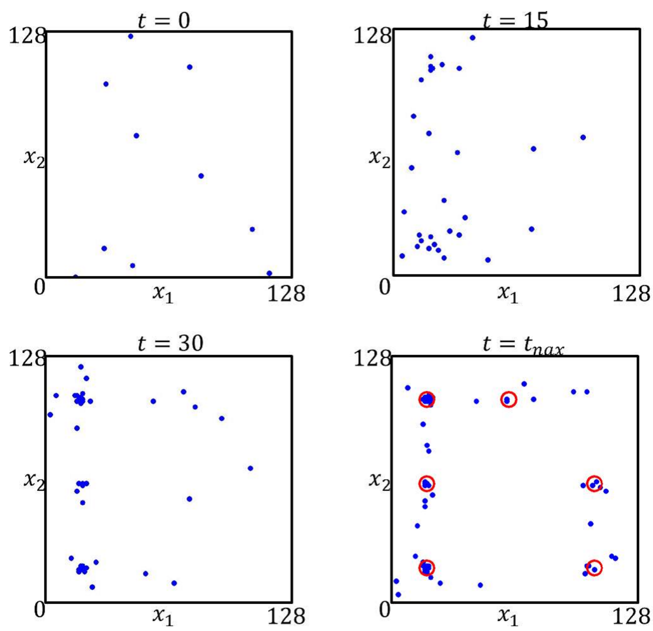

Fig. 6. Snapshots of GPSO ($\epsilon^t = 0$)

IV. CONCLUSIONS

GIPSO is presented and is applied to a fundamental MSP in this paper. In the MSP, the number of solutions is assumed to be unknown. The GIPSO is defined on the ring-type particle swarm with growing structure and has insensitive parameter. In the numerical experiment, the GIPSO has identified all the solutions.

Our future problems include optimization of topology and parameters, generalization of algorithm, application to large-scale problems, application to complex MSPs, and engineering applications.

REFERENCES

[1] A. P. Engelbrecht, Fundamentals of computational swarm intelligence, Willey (2005)

[2] S. -T. Hsieh, T. -Y. Sun, C. -L. Lin, and C. -C. Liu, Effective learning rate adjustment of blind source separation based on an improved particle swarm optimizer, IEEE Trans. Evol. Comput., 12(2), pp. 242–251 (2008)

[3] R. A. Vural, T. Yildirim, T. Kadioglu, and A. Basargan, Performance evaluation of evolutionary algorithms for optimal filter design, IEEE Trans. Evol. Comput., 16(1), pp. 135–147 (2012)

[4] H. Qin, J. W. Kimball and G. K. Venayagamoorthy, Particle swarm optimization of high-frequency transformer, Proc. IEEE/IECON, pp. 2908–2913 (2010)

[5] H. Matsushita and T. Saito, Application of particle swarm optimization to parameter search in dynamical systems, NOLTA, IEICE. E94-N, 10, pp. 458–471 (2011)

[6] Ott, E.: Chaos in Dynamical Systems. Cambridge Univ. Press, (1993)

[7] K. E. Parsopoulos and M. N. Vrahatis, On the computation of all global minimizers through particle swarm optimization, IEEE Trans. Evol. Comput., 8(3), pp. 211–224 (2004)

[8] X. Li, Niching without niching parameters: particle swarm optimization using a ring topology, IEEE Trans. Evol. Comput., 14(1), pp. 150–169 (2010)

[9] R. Sano, T. Shindo, K. Jin'no and T. Saito, PSO-based multiple optima search systems with switched topology, Proc. IEEE/CEC, pp. 3301–3307 (2012)

[10] K. Maruyama and T. Saito, Deterministic Particle Swarm Optimizers with Collision for Discrete Multi-solution Problems, Proc. IEEE/SMC, pp. 1335–1340 (2013)

[11] K. Maruyama and T. Saito, Collision Particle Swarm Optimizers and Exploring Periodic Points, NOLTA, IEICE (2014) in press