NESTING DISCRETE PARTICLE SWARM OPTIMIZERS FOR MULTI-SOLUTION PROBLEMS

Masafumi Kubota and Toshimichi Saito
Faculty of Science and Engineering, Hosei University, 184-8584 Tokyo, Japan

Keywords: Swarm inteligence, Discrete particle swarm optimizers, Multi-solution problems.

Abstract: This paper studies a discrete particle swarm optimizer for multi-solution problems. The algorithm consists of two stages. The first stage is global search: the whole search space is discretized into the local sub-regions each of which has one approximate solution. The sub-region consists of subsets of lattice points in relatively rough resolution. The second stage is local search. Each subregion is re-discretized into finer lattice points and the algorithm operates in all the subregions in parallel to find all approximate solutions. Performing basic numerical experiment, the algorithm efficiency is investigated.

1 INTRODUCTION

The particle swarm optimizer (PSO) is a simple population-based optimization algorithm and has been applied to various systems (Wachowiak et al., 2004) (Garro et al., 2009) (Valle et al., 2008): signal processors, artificial neural networks, power systems, etc. The PSO has been improved and evolved in order to expand the search function for various problems (Engelbrecht, 2005): multi-objective problems, multi-solution problems (MSP), discrete PSO, hardware implementation, etc. The MSP is inevitable in real systems and several methods have been studied (Parsopoulos and Vrahatis, 2004). The discrete PSO operates in discrete-valued search space (Engelbrecht, 2005), (Sevkli and Sevilgen, 2010) and has several advantages on reliable operation, reproducible results, robust hardware implementation, etc. However, there exist not many works of digital PSO for the MSP.

This paper presents the nesting discrete particle swarm optimizers (NDPSO) for the MSP. The NDPSO consists of two stages. The first stage is the global search. The whole search space is discretized into the local subregions (LSRs) that consist of subsets of lattice points in relatively rough resolution. Applying the particles with ring topology, the NDPSO tries to find the LSRs each of which includes the first approximate solution (AS1) that is a solution candidate. The second stage is the local search where each LSR is discretized into finer lattice points. The NDPSO operates in all the LSRs in parallel and tries to find all the approximate solutions (AS2). The roughness in the global search is a key to find all AS2s and the parallel processing in the local search is basic for effective computation. Since the NDPSO adopts subdivision, we have used the word "nesting". The NDPSO may be regarded as a discrete version of the multi-population method (Yang and Li, 2010) with an eagle strategy (Yang and Deb, 2010). Performing basic numerical experiments, the algorithm capability is investigated.

2 NESTING DISCRETE PSO

For simplicity, we consider the MSP in 2-dimensional objective functions $F(\boldsymbol{x}) \geq 0$, $\boldsymbol{x} \equiv (x_1, x_2) \in R^2$ where the minimum (optimal) value is normalized as 0. F has multiple solutions $\boldsymbol{x}_s^i$, $i = 1 \sim M$: $F(\boldsymbol{x}_s^i) = 0$, $\boldsymbol{x}_s^i = (x_{s1}^i, x_{s2}^i) \in S_0$. The search space is normalized as the center at the original with width A: $S_0 \equiv \{\boldsymbol{x} | \ |x_1| \leq A, \ |x_2| \leq A\}$. As a preparation, we define several objects. The particle $\boldsymbol{\alpha}_i$ is described by its position $\boldsymbol{x}_i$ and velocity $\boldsymbol{v}_i$: $\boldsymbol{\alpha}_i = (\boldsymbol{x}_i, \boldsymbol{v}_i)$, $\boldsymbol{x}_i \equiv (x_{i1}, x_{i2})$, $\boldsymbol{v}_i \equiv (v_{i1}, v_{i2})$) and $i = 1 \sim N$. The position $\boldsymbol{x}_i$ is a potential solution. The personal best of the i-th particle, $pbest_i = F(\boldsymbol{x}_{pbest_i})$, is the best of $F(\boldsymbol{x}_i)$ in the past history, The local best, $lbest_i = F(\boldsymbol{x}_{lbest_i})$, is the best of the personal best $pbest_i$ for the i-th particle and its neighbors. For example, the neighbor means the i-th and both sides particles in the ring topology. The global best *gbest* is the best of the personal bests and is the solution of the present state. In the complete

Kubota M. and Saito T..
NESTING DISCRETE PARTICLE SWARM OPTIMIZERS FOR MULTI-SOLUTION PROBLEMS.
DOI: 10.5220/0003623102630266
In *Proceedings of the International Conference on Evolutionary Computation Theory and Applications* (ECTA-2011), pages 263-266
ISBN: 978-989-8425-83-6

graph, the $lbest_i$ is consistent with $gbest$.

First, the search space S_0 is discretized into $m_1 \times m_1$ lattice points as shown in Fig. 1 (a): $L_0 \equiv \{0_{11}, l_{02}, \cdots, l_{0m_1}\}$, $l_{0k} = -A + kd - d/2$, $d = 2A/m_1$ where $k = 1 \sim m_1$, $F(\boldsymbol{x})$ is sampled on the discretized search space D_0. The target is the AS1 defined by

$$F(\boldsymbol{p}) < C_1,\ \boldsymbol{p} \equiv (p_1, p_2) \in D_0 \qquad (1)$$

where C_1 is the criterion. The AS1 is used to make the LSRs each of which includes either solution. Let t_1 denote the discrete time in this stage and let k_1 denote the counter of AS1.

Step 1. Let $t_1 = 1$ and let $k_1 = 0$. Let N_1 particles form ring topology. The particle positions $\boldsymbol{x}_i$, $i = 1 \sim N$, are assigned on D_0 randomly following uniform distribution on D_0 (Fig. 1 (a)). $\boldsymbol{v}_i$, $pbest_i$ and $lbest_i$ are all initialized.

Step 2. The position and velocity are updated.

$$\begin{aligned} \boldsymbol{v}_i(t_1+1) &= \omega \boldsymbol{v}_i(t_1) + r_1(\boldsymbol{x}_{pbest_i} - \boldsymbol{x}_i(t_1)) \\ &\quad + r_2(\boldsymbol{x}_{lbest_i} - \boldsymbol{x}_i(t_1)) \\ \boldsymbol{x}_i(t_1+1) &= \boldsymbol{x}_i(t_1) + \boldsymbol{v}_i(t_1+1) \end{aligned} \qquad (2)$$

where r_1 and r_2 are random parameters in $[0, c_1]$ and $[0, c_2]$, respectively. If $\boldsymbol{x}_i(t_1+1)$ exceeds D_0 then it is re-assigned into D_0. The parameters ω, r_1, r_2, c_1 and c_2 are selected from lattice points to satisfy the condition $\boldsymbol{x}(t_1) in D_0$.

Step 3. The personal and local bests are updated:

$$\begin{aligned} \boldsymbol{x}_{pbesti} &= \boldsymbol{x}_i(t_1) \ \text{ if } \ F_1(\boldsymbol{x}_i(t_1)) < F_1(\boldsymbol{x}_{pbest_i}) \\ \boldsymbol{x}_{lbest_i} &= \boldsymbol{x}_{pbest_i} \ \text{ if } \ F_1(\boldsymbol{x}_{pbest_i}) < F_1(\boldsymbol{x}_{lbest_i}) \end{aligned} \qquad (3)$$

Step 4. If $F(\boldsymbol{x}_i) < C_1$ for some i (Fig. 1(b)) then $\boldsymbol{x}_i$ is declared as the k_1-th AS1 and the counter number is increased: $\boldsymbol{p}^{k_1} = \boldsymbol{x}_i$ and $k_1 = k_1 + 1$. The position $\boldsymbol{x}_i$ is declared as a tabu lattice point and is prohibited to revisit. $\boldsymbol{x}_i$ is reset to a lattice point (Fig. 1(c)). $\boldsymbol{v}_i$, $pbest_i$ and $lbest_i$ are all reset.

Step 5. Let $t_1 = t_1 + 1$, go to Step 2 and repeat until the maximum time step t_{max1}.

In order to make the LSRs each of which is desired to include the target solution, we select top K of the AS1s $\boldsymbol{p}^1, \cdots, \boldsymbol{p}^K$ such that $F(\boldsymbol{p}^1) \leq, \cdots, \leq F(\boldsymbol{p}^K) \leq C_1$. (If $k_1 < K$ then the top k_1 is used). Using the AS1s, we construct subsets (LSR candidates) successively for $i = 1$ to K: the i-th subset S_i is centered at $\boldsymbol{p}^1$ with area $(2m_a d)^2$: $S_i = \{\boldsymbol{x} \mid |x_1 - p_1^i| < a_1,\ |x_2 - p_2^i| < a_1\}$, $a_1 = m_a d$, $i = 1 \sim K$ where $\boldsymbol{p}_i = (p_1^i, p_2^i)$ is the i-th AS1 and m_a is an integer smaller sufficiently than m_1. The m_a determines area of S_i in square shape. Although this paper uses square-shaped subspaces for simplicity, a variety of shapes should be tried depending on objective problems. If two or more subsets overlaps then the subset centered at the smallest AS1 is survived and the other subsets are removed.

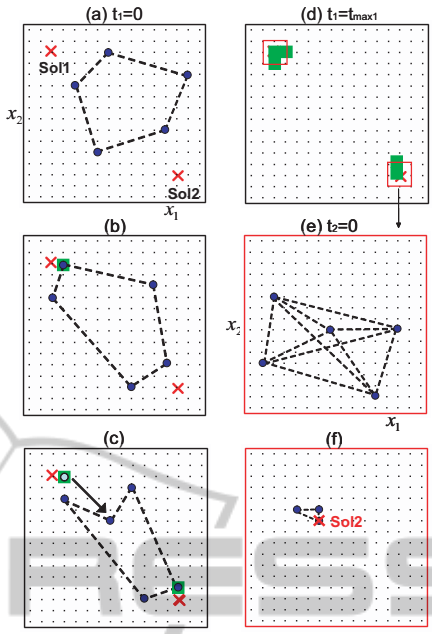

Figure 1: Particles movement for two solutions. (a) Initialization, (b) the first AS1, (c) position reset, (d) two LSRs. (e) initialization for local search, (f) AS2.

If we obtain more than M survived subsets, we select subsets centered at top M of AS1s. We then obtain M pieces of LSRs and reassign the notation $\boldsymbol{p}^i \equiv (p_1^1, p_2^i)$ to the AS1 in the i-th LSR. *If the LSRs include the target solutions $\boldsymbol{x}_s^i$, $i = 1 \sim M$ then the global search is said to be successful.*

Next, we discretize each LSR onto $m_2 \times m_2$ lattice points: $D_i = \{\boldsymbol{x} \mid x_1 - p_1^i \in L_i,\ x_2 - p_2^i \in L_i\}$, $L_i = \{l_{i1}, \cdots, l_{im_2}\}$, $l_{ik} = -(m_a + 1/2)d + kd_1$, $d_1 \equiv 2m_a d/m_2$ This D_i is the i-th discretized LSR. $F(\boldsymbol{x})$ is sampled on D_i. The target is the AS2 defined by

$$F(\boldsymbol{q}^i) < C_2,\ \boldsymbol{q}^i \equiv (q_1^i, q_2^i) \in D_i \qquad (4)$$

where C_2 ($< C_1$) is the criterion. The local search operates in parallel in D_i (Fig. 1 (e) (f)).

Step 1. Let t_2 denote the discrete time and let $t_2 = 1$. Positions of N_2 particles in complete graph are assigned randomly on D_i (Fig. 1 (e)). $\boldsymbol{v}_i$, $pbest_i$ and $gbest$ are all initialized.

Step 2. The position and velocity are updated.

$$\begin{aligned} \boldsymbol{v}_i(t_2+1) &= \omega \boldsymbol{v}_i(t_2) + r_1(\boldsymbol{x}_{pbest_i} - \boldsymbol{x}_i(t_2)) \\ &\quad + r_2(\boldsymbol{x}_{gbest} - \boldsymbol{x}_i(t_2)) \\ \boldsymbol{x}_i(t_2+1) &= \boldsymbol{x}_i(t_2) + \boldsymbol{v}_i(t_2+1) \end{aligned} \qquad (5)$$

where r_1 and r_2 are random parameter in $[0, c_1]$ and $[0, c_2]$, respectively. If $\boldsymbol{x}_i$ exceeds D_i then it is re-assigned into D_i. The parameters ω, c_1 and c_2 are selected to satisfy $\boldsymbol{x}_i \in D_i$.

Step 3. The personal and global best are updated:

$$\begin{aligned} \boldsymbol{x}_{pbesti} &= \boldsymbol{x}_i(t_2) \ \text{ if } \ F_1(\boldsymbol{x}_i(t_2)) < F_1(\boldsymbol{x}_{pbest_i}) \\ \boldsymbol{x}_{gbest} &= \boldsymbol{x}_{pbest_i} \ \text{ if } \ F_1(\boldsymbol{x}_{pbest_i}) < F_1(\boldsymbol{x}_{gbest}) \end{aligned} \qquad (6)$$

Step 4. If $F(\boldsymbol{x}_{gbest}) < C_2$ for some i then we obtain one AS2 and the algorithm is terminated.
Step 5. Let $t_2 = t_2 + 1$, go to Step 2 and repeat until the maximum time step t_{max2}.

If the object is the AS2 only, the particles are often trapped into either solution or local minima and are hard to solve the MSP. Our NDPSO tries to suppress the trapping by global search for the AS1 with discretization (sampling) of the objective function.

3 NUMERICAL EXPERIMENTS

In order to investigate the algorithm capability, we have performed basic numerical experiments for the Himmelblau function with four solutions as illustrated in Fig. 2 (a):

$$F_H(\boldsymbol{x}) = (x_1^2 + x_2 - 11)^2 + (x_1 + x_2^2 - 7)^2$$
$$\min(F_H(\boldsymbol{x}_s^i)) = 0,\ i = 1 \sim 4, \quad (7)$$

$$\boldsymbol{x}_s^1 = (-2.805118, 3.131312) \equiv \text{Sol1},$$
$$\boldsymbol{x}_s^2 = (3, 2) \equiv \text{Sol2},$$
$$\boldsymbol{x}_s^3 = (-3.779310, -3.283185) \equiv \text{Sol3},$$
$$\boldsymbol{x}_s^4 = (3.584428, -1.848126) \equiv \text{Sol4}$$
$$S_0 = \{\boldsymbol{x} \mid |x_1| \le 6,\ |x_2| \le 6\}.$$

We have selected m_1, C_1 and t_{max1} as control parameters and other parameters are fixed after trial-and-errors: $N_1 = N_2 = 20$, $\omega = 0.7$, $c_1 = c_2 = 1.4$, $K = 30$, $m_1/(2m_a) = 8$, $m_2 = 32$, $t_{max2} = 50$ and $C_2 = 0.04$. Fig. 2 (b) to (f) and Fig. 3 show a typical example of the global search process for $m_1 = 64$, $C_1 = 5$ and $t_{max1} = 50$. The NDPSO find the first AS1 at $t = 4$ and find the other AS1s successively. At time limit t_{max1}, the NDPSO can construct all the four LSRs successfully. Each LSR has 8^2 ($8 = 2m_a = m_1/8$) lattice points. Fig. 2 (g), (h) and Fig. 4 show the local search process where the NDPSO can find all the approximate solutions.

We evaluate the global search by success rate (SR) that means rate of finding all the LSRs in 100 trials for different initial states. Table 1 shows the SR of global search for m_1 and C_1. For $m_1 = 32$, 64 and 128, the LSR has 4^2, 8^2 and 16^2 lattice points, respectively in the global search. The LSR is divided into $m_2 \times m_2$ lattice points for the local search. We can see that C_1 is important for finding LSRs. As C_1 increases the SR increases and tends to saturate. For smaller C_1, the NDPSO operates like standard analog PSO in principle and tends to trap local solution/minimum. For larger C_1, the NDPSO has possibility to find a suitable AS1 before the trapping. As m_1 increases, the resolution becomes higher and the SR tends to increase; however, the computation cost also increases

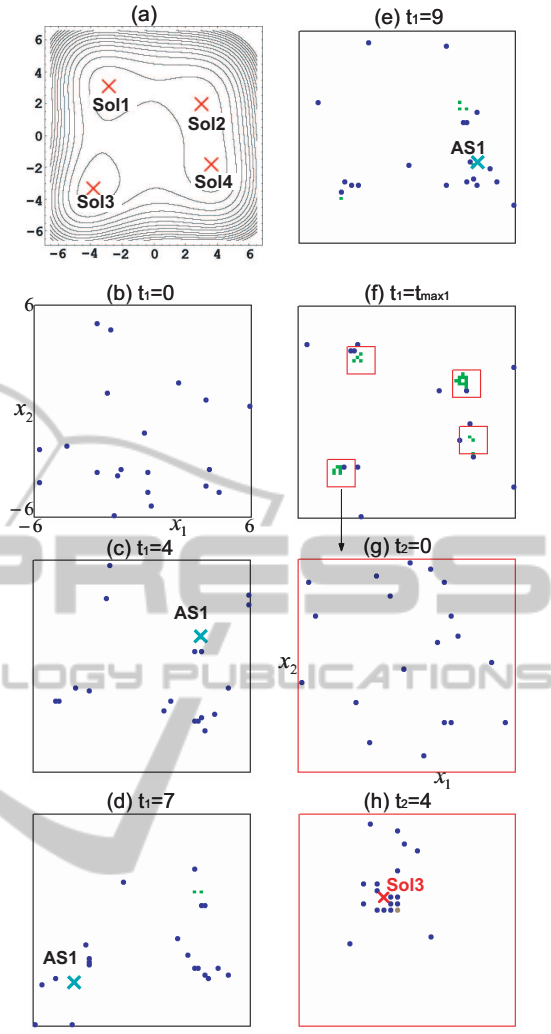

Figure 2: Particles movement of F_H. (a) contour map, (b) initialization, (c) to (e) global search process, (f) four LSRs (g) initialization for local search, (h) AS2.

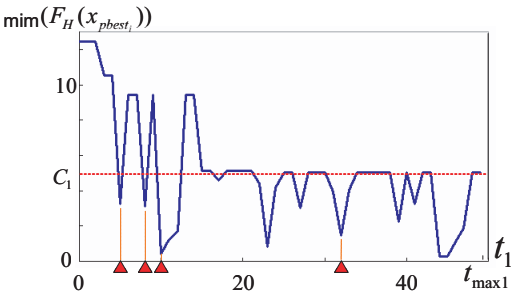

Figure 3: Global search process of F_H. $\text{mim}(F_H(x_i))$ means the minimum of F_H at time t_1.

of course: there exists a trade-off between the SR and computation cost.

We evaluate the local search by the SR and the average number of iteration (#ITE) of finding all the

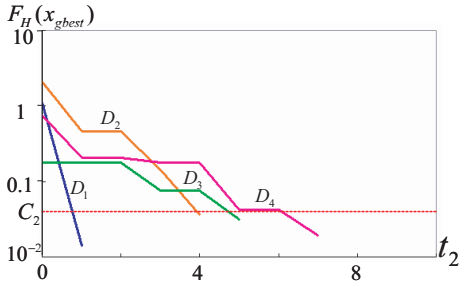

Figure 4: Local search process of F_H.

AS2s in 100 trials for different initial states. Table 2 shows the SR/#ITE of local search after the successful global search. We can see that the NDPSO can find all the AS2 speedily.

Table 1: SR of global search of F_H for $t_{max1} = 50$.

	$C_1 = 3$	$C_1 = 5$	$C_1 = 10$	$C_1 = 20$	$C_1 = 30$
$m_1 = 32$	20	36	59	79	84
$m_1 = 64$	32	53	74	92	94
$m_1 = 128$	46	61	80	91	95

Table 2: SR/#ITE of local search of F_H for $t_{max1} = 50$.

	$C_1 = 3$	$C_1 = 5$	$C_1 = 10$	$C_1 = 20$	$C_1 = 30$
$m_1 = 32$	100/5.30	100/5.60	100/5.46	100/5.55	100/5.57
$m_1 = 64$	100/5.01	100/5.34	100/5.41	100/5.43	100/5.49
$m_1 = 128$	100/5.08	100/5.20	100/5.36	100/5.42	100/5.55

Table 3: SR of global search of F_H for $C_1 = 5$.

	$m_1 = 32$	$m_1 = 64$	$m_1 = 128$
$t_{max1} = 10$	8	8	8
$t_{max1} = 30$	31	40	47
$t_{max1} = 50$	36	53	61
$t_{max1} = 100$	-	57	70
$t_{max1} = 200$	-	60	71
$t_{max1} = 400$	-	-	71
$t_{max1} = 800$	-	-	72

Table 3 shows the SR in global search for t_{max1} and m_1. The SR increases as t_{max1} increases. For $m_1 = 64$ and 128, the SR saturates and $t_{max1} = 50$ (or 100) is sufficient for reasonable results. The parameter t_{max1} can control the SR and computation costs.

4 CONCLUSIONS

The NDPSO is presented and its capability is investigated in this paper. Basic numerical experiments are performed and the results suggest the following.

1. The parameters m_1 and C_1 can control roughness in the global search that is important to find all the LSRs successfully. Higher resolution encourages trapping and suitable roughness seems to exist.
2. Parallel processing of the local search in LSRs is basic for efficient search. If LSRs can be constructed, the AS2s can be found speedily and steadily.
3. The discretization is basic to realize reliable and robust search in both software and hardware.

Future problems are many, including analysis of search process, analysis of role of parameters, comparison with various PSOs (Engelbrecht, 2005) (Miyagawa and Saito, 2009) and application to practical problems (Valle et al., 2008) (Kawamura and Saito, 2010).

REFERENCES

Engelbrecht, A. P. (2005). *Fundamentals of computational swarm intelligence*. Willey.

Garro, B. A., Sossa, H., and Vazquez, R. A. (2009). Design of artificial neural networks using a modified particle swarm optimization algorithm. In *Proc. IEEE-INNS Joint Conf. Neural Netw.*, pages 938–945.

Kawamura, K. and Saito, T. (2010). Design of switching circuits based on particle swarm optimizer and hybrid fitness function. In *Proc. Annual Conf. IEEE Ind. Electron. Soc.*, pages 1099–1103.

Miyagawa, E. and Saito, T. (2009). Particle swarm optimizers with growing tree topology. *IEICE Trans. Fundamentals*, E92-A:2275–2282.

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

Sevkli, Z. and Sevilgen, F. E. (2010). Discrete particle swarm optimization for the orienteering problem. In *Proc. IEEE Congress Evol. Comput.*, pages 1973–1944.

Valle, Y., Venayagamoorthy, G. K., Mohagheghi, S., Hernandez, J.-C., and Harley, R. G. (2008). Particle swarm optimization: basic concepts, variants and applications in power systems. *IEEE Trans. Evol. Comput.*, 12(2):171–195.

Wachowiak, M. P., Smolikova, R., Zheng, Y., and Zurada, J. M. (2004). An approach to multimodal biomedical image registration utilizing particle swarm optimization. *IEEE Trans. Evol. Comput.*, 8(3):289–301.

Yang, S. and Li, C. (2010). A clustering particle swarm optimizer for locating and tracking multiple optima in dynamic environments. *IEEE Trans. Evol. Comput.*, 14(6):959–974.

Yang, X.-S. and Deb, S. (2010). Eagle strategy using levy walk and firefly algorithms for stochastic optimization. *Nature Inspired Cooperative Strategies for Optimization*, 284:101–111.