2017 International Symposium on Nonlinear Theory and Its Applications,
NOLTA2017, Cancun, Mexico, December 4-7, 2017

A Co-Evolutional Particle Swarm Optimizer with Dynamic Re-Grouping Schemes

Ryosuke Kikkawa†, Hidehiro Nakano† and Arata Miyauchi†

†Faculty of Knowledge Engineering, Tokyo City University
1–28–1 Tamazutsumi, Setagaya-ku, Tokyo 158-8557, Japan
Email: g1681514@tcu.ac.jp, hnakano@tcu.ac.jp, miyauchi@ic.cs.tcu.ac.jp

Abstract—Particle Swarm Optimizer (PSO) is a kind of metaheuristic algorithms for solving optimization problems with continuous objective functions. PSO can be executed based on the simple dynamics of search particles. For solving high-dimensional optimization problems with the large number of design variables, Cooperative Particle Swarm Optimizer (CPSO) has been proposed. In CPSO, each sub-swarm searches partial solutions in each sub-space given by the division of search space. Integrating the partial solutions, CPSO can obtain solution candidates for the optimization problems. This paper proposes dynamical and deterministic grouping methods for the sub-swarms in CPSO. In the simulation experiments, the results for some benchmark problems are shown.

1. Introduction

Along with the recent increase in the scale and complexity of systems, demands for solving optimization problems faster in designing and constructing systems have been increasing. Since it takes a huge amount of time to find the exact optimal solution, metaheuristics has been studied extensively, which can obtain an approximate solution of the best solution within practical time. Metaheuristics is a heuristic method for optimization problems, and can be applied to various system design problems. Various metaheuristic algorithms have been proposed. One of them is Particle Swarm Optimization (PSO) [1]. PSO is a method designed to imitate the behavior of living creatures such as the flock of birds and fishes. In PSO, the organisms forming the flock are regarded as particles, and the particles find an optimal solution by searching the solution space. PSO has high convergence performance and can obtain a solution speedily. However, in high dimensional problems, PSO degrades the search performance and the solution accuracy may degrade in some cases. In order to improve the solution accuracy of the high dimensional problems, a method that introduces the concept of co-evolution into the optimization method has been proposed [2]. Co-evolution algorithms deal with high-dimensional problems by dividing the high-dimensional search space into multiple low dimensional partial search space. The Cooperative Particle Swarm Optimizer (CPSO) which adopts a co-evolutionary algorithm divides the particle group into a plurality of sub particle groups and searches for individual low dimensional search space. This method can realize to search for solutions without degrading performance against high dimensional problems [3]. Since CPSO is a method in which each sub particle group independently searches for each divided low dimensional partial search space, the search performance greatly deteriorates in a problem having dependency among design variables. In this problem, it is effective to cope with by changing dynamically combinations of design variables that compose each partial search space. Conventional research has proposed a method that uses stochastic rules to randomly recombine design variables of each partial search space. In this paper, we propose a method that uses deterministic grouping rules without stochastic factors. Some results of numerical experiments for some benchmark functions are shown.

2. CPSO-S_k

Particle Swarm Optimization (PSO) is one of the optimization methods designed to imitate the behavior of living creatures such as the flock of birds and fishes. In PSO, particles find an optimal solution by searching the solution space. Each particle has the best position information discovered in the past (pbest) and acts while sharing the best position information discovered in the whole swarm (gbest). Cooperative Particle Swarm Optimizer (CPSO) is a PSO variant that introduces in the concept of cooperative co-evolution. Cooperative co-evolution algorithm is an algorithm to cope with high dimensional problems by dividing the search space, and a solution is searched for divided partial search space based on the concept of co-evolution. In CPSO-S_k which is one of CPSO methods, an n dimensional design variables to be searched is divided into K sets of design variables (partial search space) of s dimensions, where $n = Ks$. Further, the particle group is divided into K sets of sub particle groups, and each set of design variables is independently op-

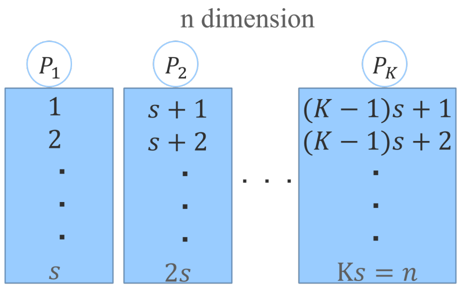

Figure 1: Division of design variables

timized. A conceptual diagram of division of design variables is shown in Fig.1. P_k represents the divided sub particle group, and each particle in each sub particle group has s design variables obtained by dividing n design variables. Each sub particle group optimizes only the allocated s design variables. By gathering K partial gbests from K sub particle groups and connecting them to a complete gbest, the overall result (solution) is obtained. The algorithm of CPSO-S_k is shown below.

1. Initialization
   Particle group is divided into K sub particle groups. Each sub particle group is assigned s design variables (partial search space). Let x^0_{ki} be the position of particle i in sub particle group k at iteration 0 and v^0_{ki} be the velocity. they are randomly initialized for all particles ki.

2. Updating pbest
   Calculate the evaluation value at the current position in the assigned partial search space of each particle, by applying the current position to the relevant part of gbest. Then, the position information of the best evaluation in each particle's search process is updates as pbest.

3. Updating gbest
   The position information of the best evaluation value is updated as gbest. Applying the pbest of each particle to the corresponding part of gbest, compare it with gbest before fitting. When updating gbest, only design variables of pbest's partial search space are updated.

4. Updating velocity
   Update velocity by Eq.(1).

$$\begin{aligned} v^{t+1}_{ki} = w \cdot v^t_{ki} + c_1 \cdot r_1 \cdot (pbest^t_{ki} - x^t_{ki}) \\ + c_2 \cdot r_2 \cdot (pbest^t_{ki} - x^t_{ki}) \end{aligned} \tag{1}$$

   where t represents the current iteration, w represents the inertial constant. c_1 and c_2 represent learning coefficients for *pbest* and *gbest*. r_1 and r_2 represent uniform random numbers of [0, 1].

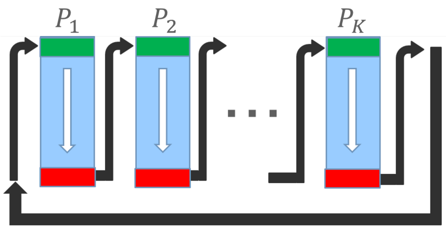

Figure 2: Proposed method 1

5. Updateing position

   Update position by Eq.(2).

$$x^{t+1}_{ki} = x^t_{ki} + v^{t+1}_{ki} \tag{2}$$

6. Condition judgment
   Repeat (2) to (5) until a predetermined number of times.

In CPSO-S_k, when there is a dependency between design variables in different partial search space, search performance deteriorates. Therefore, a method of dynamically recombining the design variables assigned to each sub particle group has been proposed. As a method for recombining design variables, a method named CPSO-rg that uses stochastic rules to randomly recombine the design variables of each partial search space has been proposed [4]. By using such a recombination, CPSO-rg can suppress the influence due to the dependency between the design variables.

3. Proposed method

In this paper, we propose a deterministic grouping method of design variables without using random factors. We propose the following two methods.

3.1. Method 1: Recombination by FIFO

In the proposed method 1, design variables of each partial search space are sequentially moved to adjacent partial search space one by one. A conceptual diagram of the design variable movement in the proposed method 1 is shown in Fig.2.

In this method, after the movement of s times, the design variable assigned to the head of a certain sub particle group moves to the head of the adjacent sub particle. Also, after the movement of Ks times, each design variable returns to the original position. By gradually shifting the design variables in the partial search space, we try to suppress the influence due to the dependency between the design variables.

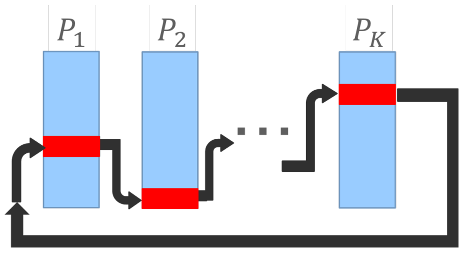

Figure 3: Proposed method 2

3.2. Method 2: Recombination by the difference of gbest

In the proposed method 2, design variables of each partial search space are moved one by one to the adjacent partial search space. The design variables to be moved are determined by the difference of gbest. A conceptual diagram of the design variable movement in the proposed method 2 is shown in Fig.3.

We compare each component of the design variables in gbest at iteration t and that at the previous iteration $t-1$. The equation for finding the difference of gbest is shown below.

$$Dif_j = |gbest_j^t - gbest_j^{t-1}| \quad (3)$$

where $gbest_j^t$ is the j-th component of the design variables in gbest. The design variables with the smallest Dif from each partial search space move to each adjacent sub particle group. In each partial search space, the smallest value of Dif means that the design variable has weak influence in the partial search space. By moving this design variable to other partial search space, the search is continued while gathering the design variables with the dependency in the same partial search space.

4. Experiment and consideration

We compare the performance by using benchmark functions for the two proposed methods, and the two conventional methods CPSO-S_k and CPSO-rg. We use three benchmark functions defined by CEC 2013 Benchmark Function[5]. Also, we define one original benchmark function. In this experiments, all the optimal solutions for the benchmark functions are normalized to 0 for the simplicity of comparison. Experimental parameters are shown in Table 1. The results of the experiments are shown in Table 2, and Figs.4-7. From the results, the performance of CPSO-S_k is the highest in Rotated Rosenbrock function. The Rotated Rosenbrock function has a dependency between design variables, but it is a unimodal function. Therefore, it seems that performance degradation due to not performing recombination in CPSO-S_k did not occur. Although the proposed method has lower performance than CPSO-S_k, the performance is almost equal to that of CPSO-rg which performs random grouping. Since the Rotated Rosenbrock function is a function having a strong dependency on the adjacent design variables, it is considered that the effect was not exhibited by the recombination method. We then introduce the original benchmark function named Random Rotated Rosenbrock Function, in which the order of the design variables are randomly shuffled. In this benchmark function, the proposed method 1 exhibited higher performance than the existing methods. In the Rastrigin function, it is understood that the performance of the proposed method 2 is the best. A design variable with a small difference in gbest provides to the stagnation of the solution search in the partial search space. Therefore, it is considered that the movement between the sub particle groups by the proposed method 2 produced an effect like the re-initialization processing, and the effect provided good influence in the Rastrigin function. In the Rotated Rastrigin function, the performance of the proposed method 1 is good, and the performance of the proposed method 2 is lower than those of the conventional methods. In the proposed method 1, since all design variables necessarily move between sub particle groups, the performance is improved in the Rotated Rastrigin function having dependencies with many variables. On the other hand, in the proposed method 2, since all design variables do not necessarily move, it is considered that the performance is not improved.

Table 1: Setting parameters

	CPSO
Particles	100
Iterations	50000
Dimension n	100
Inertial constant w	0.729
Learning coefficient C_1, C_2	1.4955
Sub particle groups K	5
Trials	10

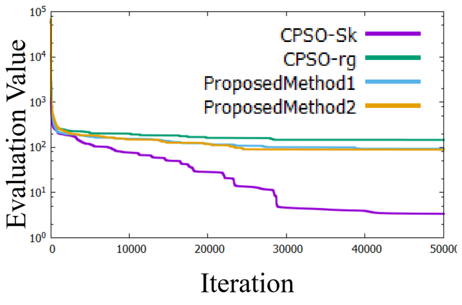

Figure 4: Rotated Rosenbrock

Table 2: Experimental results

	Average	Best	Worst
Rotated Rosenbrock Function (uni-modal with dependency)			
CPSO-S_k	3.403×10^0	2.033×10^{-2}	9.797×10^1
CPSO-rg	1.463×10^2	8.362×10^{-3}	2.625×10^2
PM1	9.351×10^1	3.989×10^1	1.923×10^2
PM2	8.930×10^1	1.028×10^{-2}	2.166×10^2
Random Rotated Rosenbrock Function (uni-modal with dependency)			
CPSO-S_k	8.465×10^1	1.100×10^{-3}	2.786×10^2
CPSO-rg	1.463×10^2	8.362×10^{-3}	2.625×10^2
PM1	6.292×10^1	1.041×10^{-3}	1.481×10^2
PM2	1.189×10^2	4.016×10^0	2.139×10^2
Rastrigin Function (multi-modal without dependency)			
CPSO-S_k	1.855×10^2	1.412×10^2	2.576×10^2
CPSO-rg	1.860×10^1	1.293×10^1	2.785×10^2
PM1	4.238×10^1	2.387×10^1	5.074×10^1
PM2	1.990×10^{-1}	0.000	9.994×10^{-1}
Rotated Rastrigin Function (multi-modal with dependency)			
CPSO-S_k	8.241×10^2	5.750×10^2	1.004×10^3
CPSO-rg	6.849×10^2	4.765×10^2	1.034×10^3
PM1	7.478×10^2	5.531×10^2	1.050×10^3
PM2	1.042×10^3	6.059×10^2	1.456×10^3

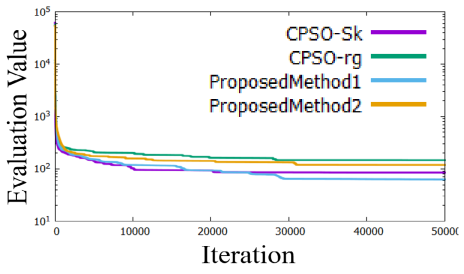

Figure 5: Random Rotated Rosenbrock

5. Conclusion

In this paper, we proposed a grouping method that does not rely on stochastic rules in CPSO and compared with the conventional methods. From the results in the simulation experiments, it was confirmed that performance was improved by the two proposed methods in plural benchmark functions. In the future we consider not only to move variables in the partial search space but also to verify methods such as adjustment of recombining intervals and dynamically changing the number of sub particle groups.

References

[1] J.Kennedy, " Particle Swarm Optimization, " Internation Conference on Neural Networks, Vol.4, pp.1942-1948 (1995)

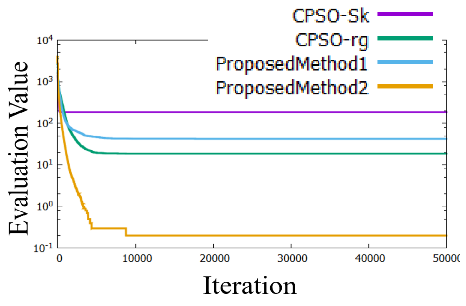

Figure 6: Rastrigin

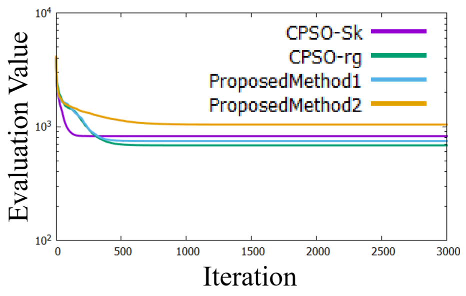

Figure 7: Rotated Rastrigin

[2] M. Potter and K. D. Jong, "A cooperative coevolutionary approach to function optimization," in Proceedings of the Third Conference on Parallel Problem Solving from Nature, pp. 249-257, Springer-Verlag (1994)

[3] F. van den Bergh and A. Engelbrecht, "A cooperative approach to particle swarm optimization," IEEE Transactions on Evolutionary Computation, vol. 8, no. 3, pp. 225-239 (2004)

[4] Z. Yang, K. Tang, and X. Yao, "Large scale evolutionary optimization using cooperative coevolution," Information Sciences, vol. 178, pp. 2986-2999, August (2008)

[5] J. J. Liang and B. Y. Qu and P. N. Suganthan and Alfredo G. Hernndez-Daz, "Problem Definition and Evaluation Criteria for the CEC 2013 Special Session on Real Parameter Optimization" Technical Report 201212, Computational Intelligence Laboratory, Zhenzhou University, Zhenzyou China And Technical Report,Nanyang Technological University, Singpore (2013)