APPLICATION OF PARTICLE SWARM OPTIMIZATION TO THE MIXED DISCRETE NON-LINEAR PROBLEMS

Satoshi KitaYama, Koetsu Yamazaki, Masao Arakawa
Dept. of Human & Mechanical Systems Engineering, Kanazawa University
2-40-20, Kodatsuno, Kanazawa, 920-8667, Japan
kitagon@t.kanazawa-u.ac.jp

Abstract: Particle Swarm Optimization is applied to the mixed discrete non-linear problems (MDNLP). PSO is mainly a method to find a global or quasi-minimum for a non-convex optimization problem of continuous design variables. To handle the discrete design variables, penalty function is introduced. By using penalty function, it is possible to treat all design variables as the continuous design variables. Through typical structural optimization problem, the validity of proposed approach for MDNLP is examined.

Key words: Particle Swarm Optimization, Global Optimization, Mixed Discrete Non-Linear Problems, Penalty Function Approach

1. INTRODUCTION

Particle Swarm Optimization (PSO), which mimics the social behavior, is an optimization technique developed by Kennedy et al [1]. It has been reported that PSO is suitable for the non-convex function of the continuous design variables. Few researches of PSO have been reported, with regard to the discrete design variables problems [2], [3]. These researches handle the discrete design variables as the continuous design variables, directly. That is, firstly all design variables may be considered as the continuous design variables. After optimum is calculated, the round-off or cut-off techniques are used. However some problems are included into these approaches. (See Fig.1 (a), (b))

Fig.1 (a) shows a case by the round-off. Point A and B represents the discrete design variables. In this case, Point B is chosen as the neighborhood of x_L by the round-off. However, the objective function at Point B makes a change for the worse, compared with the objective function at Point A[4].

Another case shown in Fig.1 (b) is well known. That is, all constraints are not satisfied by the round-off or the cut-off. [5]

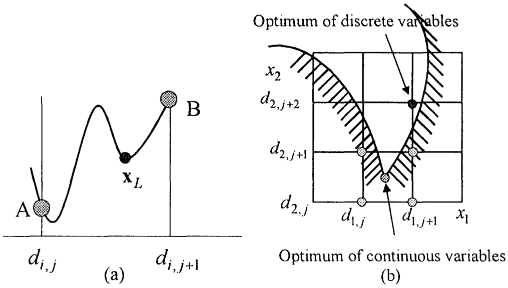

Fig.1 Optimum nature of discrete optimization

We consider that all design variables should be handled as the continuous design variables when we apply PSO to the discrete design variables problems. That is, the discrete design variables should be transformed into the continuous design variable by any methods [6].

In this paper, penalty function approach for the discrete design variables is used. By using penalty function, it is possible to handle the discrete design variables as the continuous design variables. The validity of proposed approach is examined through typical benchmark problem.

2. PARTICLE SWARM OPTIMIZATION

Particle Swarm Optimization (PSO), which is one of the meta-heuristic methods, is developed for the non-convex function of continuous design variables, and PSO does not utilize the gradient information of function like Genetic Algorithm (GA). In the PSO, each particle updates their position and velocity by a simple addition and subtraction of vector during search process, and finally some particles find global or quasi-optimum. Some models of PSO have been proposed. Among of them, most popular model may be called as g-best model [7].

The position and velocity of particle d are represented by $\boldsymbol{x}_d^k$ and $\boldsymbol{v}_d^k$, respectively. k represents k-th iteration. The position and velocity of particle d at k+1 iteration are calculated by the following equations.

$$\boldsymbol{x}_d^{k+1} = \boldsymbol{x}_d^k + \boldsymbol{v}_d^{k+1} \tag{1}$$

$$\boldsymbol{v}_d^{k+1} = w\boldsymbol{v}_d^k + c_1 r_1(\boldsymbol{p}_d^k - \boldsymbol{x}_d^k) + c_2 r_2(\boldsymbol{p}_g^k - \boldsymbol{x}_d^k) \tag{2}$$

In Eq.(2), w is called as inertia term. c_1 and c_2 are parameters. r_1 and r_2 are random number between [0,1]. c_1=c_2=2 are often used[7]. $\boldsymbol{p}_d^k$, which is called as p-best, represents the best position of particle d till k-th iteration, and p_g^k, which is called as g-best, represents the best position in the swarm till k-th iteration. The inertia term in Eq. (2) gradually decreases as follow.

$$w = w_{\max} - (w_{\max} - w_{\min}) / k_{\max} \times k \tag{3}$$

In Eq. (3), $w_{\max}$ and $w_{\min}$ represent the maximum and minimum value of inertia, respectively. $w_{\max} = 0.9$ and $w_{\min} = 0.4$ are generally recommended[8].

3. MDNLP BY PARTICLE SWARM OPTIMIZATION

3.1 Problem definition

In general, the Mixed Discrete Non-Linear Problem (MDNLP) is described as follow. [6]

$$f(\boldsymbol{x}) \rightarrow \min \tag{4}$$

$$x_{i,L} \le x_i \le x_{i,U} \quad i = 1,2,\cdots,m \tag{5}$$

$$x_j \in D_j \quad D_j(d_{j,1}, d_{j,2}, \cdots, d_{j,q}) \quad j = 1,2,\cdots,n \tag{6}$$

$$g_k(\boldsymbol{x}) \le 0 \quad k = 1,2,\cdots,ncon \tag{7}$$

where $\boldsymbol{x}$ represents the design variables, which consist of the continuous and discrete design variables. $f(\boldsymbol{x})$is the objective function to be minimized, and $g_k(\boldsymbol{x})$ is the behavior constraints. $ncon$ is the number of behavior constraints. x_i represents the continuous design variables, and m is the number of continuous design variables. $x_{i,L}$ and $x_{i,U}$ denote the lower and upper bound of continuous design variables. On the other hand, x_j represents the discrete design variables, and n is the number of discrete design variables. D_j is the set of discrete values for the j-th discrete design variables. $d_{i,j}$ is the j-th discrete value for the i-the discrete design variables. q represents the number of discrete values. The lower and upper bound of the discrete design variables is given by $d_{j,1}$ and $d_{j,q}$, respectively.

3.2 Penalty function

In this paper, the following penalty function is used. [9]

$$\phi(\boldsymbol{x}) = \sum_{i=1}^{n} \frac{1}{2} \left[\sin \frac{2\pi \left\{ x_i - 0.25(d_{i,j+1} + 3d_{i,j}) \right\}}{d_{i,j+1} - d_{i,j}} + 1 \right] \tag{8}$$

where $d_{i,j}$ and $d_{i,j+1}$ represent the discrete design variables. x_i is the continuous design variables between $d_{i,j}$ and $d_{i,j+1}$. Then the augmented objective function is constructed as follow, by using above penalty function.

$$F(\boldsymbol{x}) = f(\boldsymbol{x}) + s\phi(\boldsymbol{x}) + r\sum_{k=1}^{ncon} \max[0, g_k(\boldsymbol{x})] \tag{9}$$

s in Eq.(9) denotes the penalty parameter for the penalty function of the discrete design variables, and r also denotes the penalty parameter for the penalty function of the behavior constraints. By using Eq. (8), it is possible to handle all design variables as the continuous design variables. MDNLP is transformed into the following problem.

$$F(\boldsymbol{x}) \rightarrow \min \tag{10}$$

$$x_{i,L} \le x_i \le x_{i,U} \quad i = 1,2,\cdots,m \tag{11}$$

$$d_{j,1} \le x_j \le d_{j,q} \quad j = 1,2,\cdots,n \tag{12}$$

In the following discussion design variables are supposed to be the discrete design variables for the simplicity. In the case of the mixed design variables, we discuss section 3.8.

3.3 Characteristics of penalty function

The value of penalty function of Eq. (8) is small at the neighborhood of discrete value. On the other hand, the value of penalty function of Eq. (8) is large, turning from discrete value. When $\boldsymbol{p}_g^k$, which represents the best value of the objective function in the swarm till the k-the iteration, satisfies the following equation, the discrete value resides in the neighborhood of $\boldsymbol{p}_g^k$.

$$\phi(\boldsymbol{p}_g^k) \le \varepsilon \tag{13}$$

ε in Eq. (13) represents small positive value. As a result, we have to update the penalty parameter s for the discrete design variables in Eq. (9) until Eq. (13) is satisfied. In order to examine the effect of the penalty parameter s, let us consider a following simple problem.

$$f(x) = x^4 - \frac{8}{3}x^3 - 2x^2 + 8x \rightarrow \min \tag{14}$$

$$x = \{-1, 0, 1, 2\} \tag{15}$$

In this simple problem, objective $f(\boldsymbol{x})$ and augmented objective function $F(\boldsymbol{x})$ are shown in Fig. 3. Penalty parameter s in Fig. 3 is set as s=10.

From Fig.3, it is apparent that the augmented objective function becomes non-convex. As a result, to find optimum of the discrete design variables is transformed to finding global minimum of augmented objective function. Additionally, the discrete value is given by the point, at which the relative error becomes small between $f(\boldsymbol{x})$ and $F(\boldsymbol{x})$. Then we use the following equation as terminal criteria.

$$\frac{\left|F(p_g^k)-f(p_g^k)\right|}{\left|F(p_g^k)\right|}\le\varepsilon \tag{16}$$

Considered that PSO does not use the gradient information of function, it is difficult to satisfy Eq. (13) strictly. As a result, Eq. (16) is used.

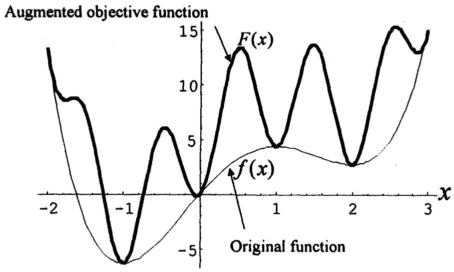

Fig. 3 Objective and augmented objective function

Behaviors of the augmented objective function for various penalty parameter s are shown in Fig. 4.

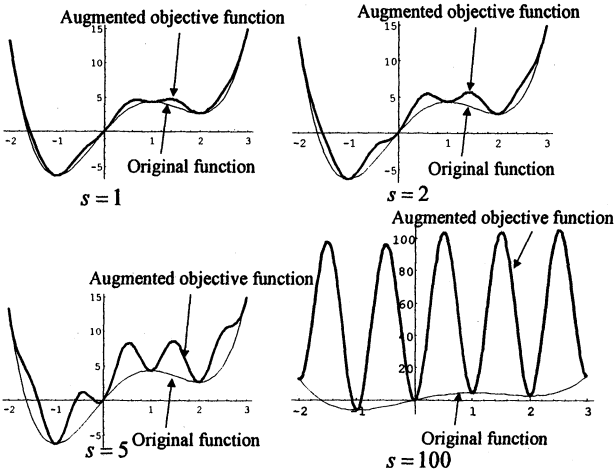

Fig.4 Augmented objective function for various penalty parameters

3.4 Initial penalty parameter s

Penalty parameter s in Eq. (9) is determined as follows. Initial search point $\boldsymbol{x}_d$ of particle d is determined randomly. Then the value of penalty function represented by Eq. (8) is calculated to each particle.

$$s_d = 1+\phi(\boldsymbol{x}_d) \qquad d=1,2,\cdots,agent \tag{17}$$

where s_d represents the penalty parameter of particle *d*. *agent* is the number of particles. Initial penalty parameter $s_{initial}$ is determined as follow.

$$s_{initial} = \min\{s_1, s_2, \cdots, s_{agent}\} \tag{18}$$

At the initial stage to search optimum, we actively transform the augmented objective function into non-convex function. And local minima are generated at the neighborhood of discrete value.

3.5 Update of penalty parameter

Suppose that s^k denotes penalty parameter s at k-th iteration. Following equation is used to update penalty parameter s.

$$s^{k+1} = s^k \times \exp(1 + \phi(\boldsymbol{p}_g^k)) \tag{19}$$

The behavior of the augmented objective function by updating penalty parameter s is shown in Fig.5. In Fig.5, solid line shows the augmented objective function at k-th iteration, and dot line shows the augmented objective function at k+1-th iteration.

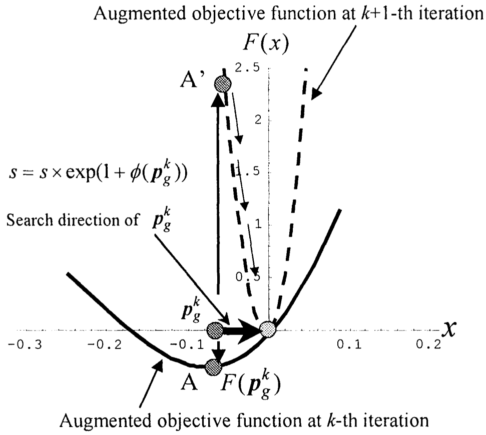

Fig. 5 Update of penalty parameter

As shown in Fig.5, an augmented objective function at k+1-th iteration becomes hard non-convex function, compared with an augmented objective function at k-th iteration. For example, A in Fig.5 corresponds to the point $\boldsymbol{p}_g^k$ at k-th iteration. By updating penalty parameter s, $\boldsymbol{p}_g^k$ corresponds to the point A' on the dot line, which represents the augmented objective function at k+1-th iteration. As a result, it is expected that $\boldsymbol{p}_g^k$ moves to the direction in Fig. 5. Finally, it is also expected that $\boldsymbol{p}_g^k$ will satisfy Eq. (16).

3.6 Initialization of penalty parameter

When Eq. (16) is satisfied, the discrete value around the neighborhood of $\boldsymbol{p}_g^k$ resides. Then in order to find another discrete value, initial penalty parameter obtained by Eq. (18) is utilized. This is because the augmented objective function becomes hard non-convex function, by updating penalty parameter s. As a result, it is considered that $\boldsymbol{p}_g^k$ fail to escape from local minimum. For such occasions, we relax the augmented objective function by initializing penalty parameter when $\boldsymbol{p}_g^k$ which satisfies Eq. (16) can be obtained. As a result, it is expected that $\boldsymbol{p}_g^k$ can escape from local minimum.

3.7 Difference between traditional and proposed method

In this paper, the penalty function for the discrete design variables is the same as Shin et. al.[9]. However, its approach is very different. Shin et. al. searched optimum by handling all design variables as the continuous design variables at the initial search stage. In this stage, penalty parameter s in Eq. (9) is set as zero. Then the penalty function for the discrete design variables was introduced after optimum was obtained. This is because the augmented objective function becomes non-convex when penalty parameter s is introduced at the initial search stage.

On the other hand, we actively introduce the penalty parameter s at the initial search stage. Obviously the augmented objective function becomes non-convex function. However, this is not serious problem because PSO is applied to this non-convex augmented objective function. We also introduced the new update scheme of penalty parameter s, which was expressed by Eq. (19). In the past researches [5], [9], the constant positive number is used to update the penalty parameter. This means the update scheme of penalty parameter depends on the problem. However, the penalty parameter s always changes in our approach. This is because the value of the penalty function $\phi(\boldsymbol{p}_g^k)$ is utilized. Finally, we newly introduced the initialization of the penalty parameter s. By initializing the penalty parameter s, it is possible to relax the augmented objective function. As a result, it is expected that $\boldsymbol{p}_g^k$ can escape from local minimum.

3.8 In the case of mixed design variables

In case of mixed design variables, component of $\boldsymbol{p}_g^k$ can be expressed as follow.

$$\boldsymbol{p}_g^k = (\boldsymbol{x}^{cont}, \boldsymbol{x}^{discrt})^T \tag{20}$$

where $\boldsymbol{x}^{cont}$ represents the components of the continuous design variables, and $\boldsymbol{x}^{discrt}$ also represents the components of the discrete design variables. Then, the components of the continuous design variables $\boldsymbol{x}^{cont}$ in $\boldsymbol{p}_g^k$ are neglected when we consider the terminal criteria given by Eq. (16). That is, only the components of the discrete design variables $\boldsymbol{x}^{discrt}$ in $\boldsymbol{p}_g^k$ are considered when we consider the terminal criteria by Eq. (16).

4. ALGORITHM

The algorithm of PSO for MDNLP is described.
(STEP1) Number of particles, and maximum search iteration k_{max} are determined. Set k=1. (k represents iteration) Set initial position and velocity of each particle, randomly.
(STEP2) Calculate the value of penalty function by Eq. (8) for each particle.
(STEP3) Calculate the penalty parameter s for each particle, using Eq. (17).
(STEP4) Initial penalty parameter $s_{initial}$ is determined by using Eq. (18).
(STEP5) Calculate the augmented objective function for each particle.
(STEP6) Calculate g-best and p-best.
(STEP7) Check the terminal criteria by Eq. (16). If Eq. (16) is satisfied, the penalty parameter s is initialized. Otherwise the penalty parameter s is updated by Eq. (19).
(STEP8) The inertia term is updated by Eq. (3).
(STEP9) The velocity and position is updated by using Eq.(1) and Eq.(2).
(STEP10) Iteration is increased as $k = k+1$.
(STEP11) Check the iteration. If $k \le k_{max}$, go to STEP 2. Otherwise, the algorithm will terminate by considering p_g^k as optimum.

5. OPTIMUM DESIGN OF PRESSURE VESSEL

Let us consider the optimum design of pressure vessel as shown in Fig.6. This problem is one of the most famous MDNLP, and many researches have been done [10-15]. Several results are shown in Table 1. From Table 1, it is considered that it is very difficult to find optimum solution though this problem consists of only 4 design variables.

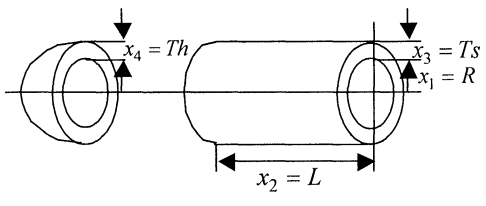

Fig.6 Optimum design of pressure vessel

Design variables are 1) Radii R (continuous design variables), 2) Length L (continuous design variables), 3) Thickness Ts (discrete design variables), and 4) Thickness Th (discrete design variables). Objective function is to minimize the cost, and is given as follow.

$$f(\boldsymbol{x}) = 0.6224x_1x_2x_3 + 1.7781x_1^2x_4 + 3.1661x_2x_3^2 + 19.84x_1x_3^2 \to \min \quad (21)$$

On the other hand, behavior and side constraints are given as follows.

$$25 \le x_1 \le 150 \quad (22)$$

$$25 \le x_2 \le 240 \quad (23)$$

$$0.0625 \le x_3, x_4 \le 1.25 \quad (24)$$

$$g_1(\boldsymbol{x}) = 0.0193 x_1 / x_3 - 1 \le 0 \quad (25)$$

$$g_2(\boldsymbol{x}) = 0.00954 x_1 / x_4 - 1 \le 0 \quad (26)$$

$$g_3(\boldsymbol{x}) = x_2 / 240 - 1 \le 0 \quad (27)$$

$$g_4(\boldsymbol{x}) = \left(1296000 - \frac{4}{3}\pi x_1^3\right) \Big/ \pi x_1^2 x_2 - 1 \le 0 \quad (28)$$

x_3 and x_4 are discrete design variables, and are integer multiples of 0.00625 inch. Behavior constraints from Eq. (25) to Eq. (28) are handled as penalty function as shown in Eq. (9). $r = 1.0 \times 10^8$ is used.

The number of particle is set as 100, and the number of maximum search iteration is set as 5000. 10 trials are performed with different random seed. The best result during 10 trials is shown in Table 1.

Table 1 Comparison of results

	Sandgren (11)	Qian (12)	Kannan (10)	Hsu (13)	Lewis (14)	Kitayama	Arakawa (15)
R[inch]	47.000	58.312	58.291	N/A	38.760	38.684	38.858
L[inch]	117.701	44.522	43.690	N/A	223.299	224.096	221.402
Ts[inch]	1.125	1.125	1.125	N/A	0.750	0.750	0.750
Th[inch]	0.625	0.625	0.625	N/A	0.375	0.375	0.375
g1	-0.194	0.000	0.000	N/A	-0.003	-0.004	0.000
g2	-0.283	-0.110	-0.110	N/A	-0.014	-0.016	-0.011
g3	-0.510	-0.814	-0.818	N/A	-0.070	-0.066	-0.078
g4	0.054	-0.021	-1.109	N/A	-1.519	0.000	0.000
Objective[$]	8129.800	7238.830	7198.200	7021.670	5980.950	5875.254	5850.770

6. CONCLUSIONS

In this paper PSO is applied to MDNLP. Penalty function has been introduced, in order to handle the discrete design variables. To utilize penalty function, it is possible to handle all design variables as the continuous design variables. The augmented objective function becomes non-convex function, by introducing penalty function. Considering that PSO is naturally suitable for the non-convex function of the continuous design variables, the penalty function approach may be valid. Additionally, we proposed how to determine the penalty parameter for the penalty function of the discrete design variables. Through typical benchmark problem, the validity has been examined.

ACKNOWLEDGEMENT

Authors would like to say many thanks to Prof. H. Yamakawa, (Waseda University in Japan), and Prof. H. Nakayama (Konan University in Japan), who gave us the useful and contributive comments.

REFERENCES

1. Kennedy, J., Eberhart, R.C., "*Swarm Intelligence*", Morgan Kaufmann Publishers, (2001).
2. Venter, G., Sobieski, J.S., "*Particle Swarm Optimization*", 9-th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization Conference, (2002), AIAA2002-1235.
3. Venter G., Sobieski J.S., "*Multidisciplinary Optimization of a Transport Aircraft Wing using Particle Swarm Optimization*", 9-th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization Conference, (2002), AIAA2002-5644.
4. Rao, S.S, "Engineering Optimization Theory and Practice", Wiley InterScience, (1996).
5. Fu, J.F, Fenton, R.G., Cleghorn, W.L, "*A Mixed Integer-Discrete Continuous Programming and its Application to Engineering Design Optimization*", Engineering Optimization, (1991), Vol. 17, pp.263-280.
6. Arora J.S, Huang M.W., "*Methods for Optimization of Nonlinear Problems with Discrete Variables: A Review*" ,Structural Optimization, (1994), 8, pp.69-85.
7. Parsopoulos, K.E., Vrahatis, M.N., "*Recent Approaches to Global Optimization Problems through Particle Swarm Optimization*", Natural Computing, (2002), Vol.1, pp.235-306.
8. Fukuyama, Y., " *A Particle Swarm Optimization for Reactive Power and Voltage Control Considering Voltage Security Assessment*", IEEE Trans. Power Syst., (2000), Vol.15-4, pp-1232-1239.
9. Shin, D.K, Gurdal, Z, Griffin O.H., "*A Penalty Approach for Nonlinear Optimization with Discrete Design Variables*", Engineering Optimization, (1990), Vol.16, pp-29-42.
10. Kannan, B.K., Kramer, S.N., "*An Augmented Lagrange Multiplier Based Method for Mixed Integer Discrete Continuous Optimization and Its Applications to Mechanical Design*", Journal of Mechanical Design, (1994), Vol.116, pp.405-411.
11. Sandgren,E., "*Nonlinear and Discrete Programming in Mechanical Design Optimization*", Journal of Mechanical Design, (1990), Vol.112, pp.223-229.
12. Qian, Z., Yu, J., Zhou, J., "*A Genetic Algorithm for Solving Mixed Discrete Optimization Problems*", DE-Vol.65-1, Advances in Design Automation, (1993), 1, pp.499-503.
13. Hsu Y.H., Sun, T.L.,Leu, LH., "*A Two Stage Sequential Approximation Method for Non-linear Discrete Variable Optimization*", ASME/DETC/DAC MA, (1995), pp.197-202
14. Lewis, K., and Mistree, F., "*Foraging-Directed Adaptive Liner Programming: An Algorithm for Solving Nonlinear Mixed Discrete/Continuous Design Problems*", ASME/DETC/DAC, (1996), Paper No. 1601, in CD-ROM.
15. Arakawa, M., Hagiwara, I., "*Nonlinear Integer Discrete and Continuous Optimization Using Adaptive Range Genetic Algorithm*", ASME/DETC/DAC, (1997), Paper No. 3728, in CD-ROM.