Comparisons between the Hybrid Taguchi-Genetic Algorithm and Genetic Algorithm

Koun-Tem Sun [1], Ching-Ling Lin[1], Hsin-Te Chan [1], Hong-Ming Kang [1], Man-Ting Ku [2]

[1]National University of Tainan, [2]Far East University

E-mail: sunkountem@gmail.com

Abstract: Hybrid Taguchi genetic algorithm can be used to solve the global continuous optimization problems. Aside from the global search capability of traditional genetic algorithm, it further combines Taguchi experimental method to explore the optimal feasibility of the offspring. Taguchi method is inserted between the crossover and mutation operations of the traditional genetic algorithm. Hybrid Taguchi genetic algorithm also seems to outperform the traditional genetic method in obtaining the optional or near optimal solutions because of its fast convergence ability and robustness.

Although the hybrid Taguchi genetic algorithm is more powerful than the traditional genetic one in the optimization of global continuous function, yet it still needs further investigation to conclude if it also offers better solution than the latter to the optimization of global discrete function.

Therefore, this study tries to compare the two algorithms in each individual's performance in the optimization of global discrete function. It aims to figure out whether the hybrid Taguchi genetic algorithm is better than traditional genetic algorithm or not.

Keywords: Hybrid Taguchi-Genetic algorithm, Genetic algorithm, Optimization problems.

INTRODUCTION

In 2004, Scholar Tsai et al.[1] combined the Taguchi's quality control mechanism[2] into Genetic Algorithm[3], and proposed a hybrid Taguchi-Genetic Algorithm (HTGA) to solve many optimization problems. The performances of the solved problems were much better than traditional Genetic Algorithm (GA)[4, 5]. However, the solved problems in the published papers were continuous functions. How about the optimization problems with discrete function and constraints? They had not been discussed and analysed.

In this paper, we will compare the HTGA and GA for solving the problems with continuous variables and discrete variables, respectively. In addition, we will point out the weakness of HTGA for solving the problems with discrete variables and constraints, why the performance of HTGA is not better than GA. Two kinds of optimization problems: the 0-1 knapsack problem and the traveling salesman problem with different sizes were used as the test examples for the HTGA and GA. Experimental results showed that GA obtained better solutions than HTGA. The HTGA is not suitable to solve the problems with discrete functions and constraints.

THEORY, EVIDENCE AND RESEARCH NEEDS

Taguchi Method: The Taguchi method is a robust design approach to improve the quality of a product by minimizing the effect of the causes of variation without eliminating the causes[6-9]. We will use the following simple example to illustrate the concept of Taguchi method.

If an optimization problem is defined as:

(1)

$$Max.\ f(x_1, x_2, ..., x_7) = \sum_{i=1}^{7} x_i, \quad \forall x_i, \quad x_i \in \{0, 1\}.$$

Each variable xi may be 0 or 1. One set (denoted as S1) of data is {0, 0, 1, 1, 0, 1, 0} and another set (denoted as S2) is {1, 1, 0, 0, 1, 0, 1}. How do we select the value of xi from these two sets to find the maximum value of function f(x)?

Based on the Taguchi method, two-level orthogonal array can be used. One level represents the set S1 (denoted as level 1), and another level represents the set S2 (denoted as level 2). Each variable corresponds to a factor in the orthogonal array. An orthogonal array is a fractional factorial matrix, which assures a balanced comparison of levels of any factor or interaction of factors. Table 1 shows a L_8 (2^7) orthogonal array, where each column represents the level of the factors in each run, and each column represents the factor that can be changed from each run. The suffix 8 represents the number of experimental runs, and the superscript 7 denotes the number of factors (columns) in the orthogonal array. The value 2 represents the number of levels for each factor. The general form of two-level orthogonal array is:

ISSN: 2222-7059(print), ISSN: 2222-7067(online), http://www.aspbs.com/aiem/
DOI: 10.7508/AIEM-V1-N1-10-18

Ln (2^{n-1}), where $n = 2^k$, and k is a positive integer which is greater than 1.

Table 1. L8 (27) Orthogonal array for 7 factors {A, B, … G}

Experiment index	Factors						
	A	B	C	D	E	F	G
	Column index						
	1	2	3	4	5	6	7
1	1	1	1	1	1	1	1
2	1	1	1	2	2	2	2
3	1	2	2	1	1	2	2
4	1	2	2	2	2	1	1
5	2	1	2	1	2	1	2
6	2	1	2	2	1	2	1
7	2	2	1	1	2	2	1
8	2	2	1	2	1	1	2

A value SNR(ηi) is used as the quality characteristic of choice for the experiment i. There are three types SNRs available depending on the goals: normal-the-best (NTB), smaller-the-best (STB), and larger-the-best (LTB). For Eq. (1), we will apply the larger-the-best SNR values for deciding to select level one or level two for each factor. For example, factors A ~ G are set to level 1 (S1) in the experiment 1. The value of xi, i=1~7, would be {0, 0, 1, 1, 0, 1, 0}, and gets the value of function f to be 3. The value will be substituted into SNR formula[2]. After running eight experiments, we can obtain eight ηi values. Each ηi corresponds to the experiment i. Then, for each factor, if the summation of ηi at level 1 is less than that of level 2, then the value of this factor is set to the value of level 2. For example, considered the factor A, we sum η i , i =1~4, for level 1, and sum the values ηi , i =5 ~ 8 for level 2. If the summation of level 1 is less than that of level 2, then the value of factor A is set to level 2 (=1). The larger the numerical value of the level, the more favorable the situation is [10].

Hybrid Taguchi-Genetic Algorithm (HTGA): The HTGA is similar to GA[11-13], but incorporates the Taguchi method with the crossover operation of GA. In the crossover operation of HTGA, each gene in the offspring chromosome is decided by the SNR values computed by the Taguchi method. Two parent chromosomes are thought as two levels S1 and S2. Then, the genes in chromosome S1 are [0, 0, 1, 1, 0, 1, 0] and genes in chromosome S2 are [1, 1, 0, 0, 1, 0, 1]. When the fitness function is equal to Eq. (1), f (A) = 3 and f (B) =4. We can simplify the SNR formula for experiment k as:

$$\eta_k = \sum_{i=1} x_i^2 . \quad (2)$$

Combine the Taguchi method and crossover operation for the offspring chromosome shown in Table 2. The value in Table 2 is the gene in chromosome S1 (level 1) or chromosome S2 (level 2) which are selected from the orthogonal array defined in Table 1.

For factor A, the summation of level 1 ηk is:

$$E_{f1} = \sum_{k=1,\,2,3,\,4} \eta_k = 9 + 9 + 9 + 9 = 36 \quad (3)$$

For factor A, the summation of level 2 ηk is:

$$E_{f2} = \sum_{k=5,6,7,8} \eta_k = 25 + 1 + 25 + 25 = 76 \quad (4)$$

We can also obtain the value for factor B.

For factor B, the summation of level 1 ηk is:

$$E_{f1} = \sum_{k=1,2,5,6} \eta_k = 9 + 9 + 25 + 1 = 44 \quad (5)$$

For factor B, the summation of level 2 ηk is:

$$E_{f2} = \sum_{k=3,4,7,8} \eta_k = 9 + 9 + 25 + 25 = 68 \quad (6)$$

Based on the same procedure, we can compute the value of Ef1 and Ef2 for factors C~G (shown in Table 2). The goal of the function f is to find the maximum value. By applying the larger-the-best approach of Taguchi method to find the better factor for each factor, we can obtain the optimal level for each factor by select the larger value of E_{f1} or E_{f2}. In this way, we can obtain each gene of offspring chromosome by selecting the gene with optimal level (chromosome). For example, the first and second genes would be selected from chromosome 2 (optimal level = 2), the first and second genes of offspring chromosome would be 11. The 3rd to 7th genes could be also selected by the same procedure, and obtain the genes: 11111. The full chromosome of offspring would be 1111111, it is also the optimal solution.

Table 2. A better offspring generated from two parent chromosomes s1 and s2 by using Taguchi method

Experiment index	Factors							fitness f	η_k
	A	B	C	D	E	F	G		
	1	2	3	4	5	6	7		
1	0	0	1	1	0	1	0	3	9
2	0	0	1	0	1	0	1	3	9
3	0	1	0	1	0	0	1	3	9
4	0	1	0	0	1	1	0	3	9
5	1	0	0	1	1	1	1	5	25
6	1	0	0	0	0	0	0	1	1
7	1	1	1	1	1	0	0	5	25
8	1	1	1	0	0	1	1	5	25
E_{f1}	36	44	68	68	44	68	44		
E_{f2}	76	68	44	44	68	44	68		
Optimal level	2	2	1	1	2	1	2		
Offspring chromosome	1	1	1	1	1	1	1		

CONTINUOUS VARIABLES

Three optimization problems[1] with continuous variables are used to demonstrate the performance of HTGA and GA.

Find the Mini Value of

$$\{\sum_{i=1}^{30} x_i^2,\ \ x_i \in [-100,100],\ x_i \in R$$

Fig 1 shows that the convergence speed and solution quality of HTGA are similar to those of GA.

Find the Mini Value of

$$\{\sum_{i=1}^{30}|x_i| + \Pi_{i=1}^{30}|x_i|,\ \ x_i \in [-10,10],\ x_i \in R$$

Fig 2 shows that the convergence speed and solution quality of HTGA are much better than those of GA.

Find the Mini Value of

$$\{\sum_{i=1}^{30}(\sum_{j=1}^{i} x_j)^2,\ x_i \in [-100,100], x_i \in R$$

Fig 3 shows that the solution quality of HTGA is also better than that of GA. However, the convergence speed is slower than GA before 900-generation.

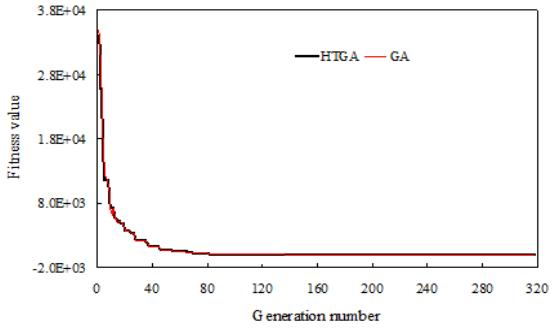


Figure 1. The best solution of GA and HTGA in each generation

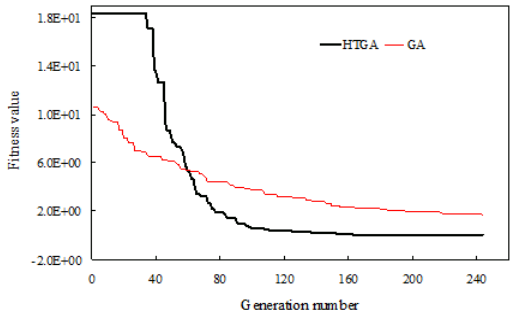


Figure 2. The best solution of GA and HTGA in each generation

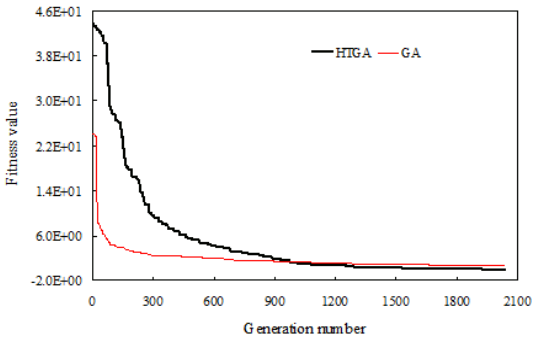


Figure 3. The best solution of GA and HTGA in each generation

In general, the HTGA has a very remarkable performance for solving optimization problem with continuous variables. However, the performance of HTGA for solving the optimal problems with discrete variables has not been evaluated still now. Then, the following section will apply two kinds of optimization problems with discrete variables and constraints to test the performance of HTGA and GA.

DISCRETE VARIABLES WITH CONSTRAINTS

Two kinds of optimization problems with discrete variables and constraints were applied to evaluate the performances of HTGA and GA. These two problems can be easily transformed into other optimization problems[14-16], and can be regarded as the typical of optimization problems with discrete variables and constraints.

0-1 Knapsack Problem: The 0-1 knapsack problem is an NP-hard problem[16-19]. This problem can be stated as the following. A person goes into a store with n objects. Each object i is worth ci dollars and weighs wi pounds, where ci and wi are integers. The person wants to take as valuable a load as possible, but he can carry at most W pounds in his/her knapsack, for some integer W. Which objects should be taken? It can be formulated as the following equations:

$$\text{(7)} \qquad \begin{aligned} &Max.\sum_{i=1}^{n} c_i x_i \\ &s.t.\sum_{i=1}^{n} w_i x_i \le W, \ \text{where } x_i \in \{0,1\} \text{ and } c_i, w_i \text{ and } W \in N. \end{aligned}$$

When an object is selected into the knapsack, the value of xi =1, else xi = 0.

The Max function in equation (7) is the objective function, and is used to find the maximum valuable load from n objects. The lower parts in equation (7) are the constraints that the capacity of knapsack is restricted to W.

The chromosome is an n-gene length 0-1 string. Each bit i represents the i-th object which is selected (=1) or not (=0) into the knapsack. We use the symbol xi to represent the state of the i-th bit in the chromosome. Each chromosome is also a solution of 0-1 knapsack problem if it satisfied the constraints. The genes with bit 1 in the chromosome would be randomly selected and mutate to 0 for violating the constraints (capacity of knapsack ≦ W). The execution flow of GA/HTGA for solving the 0-1 knapsack problem is shown in Fig 4.

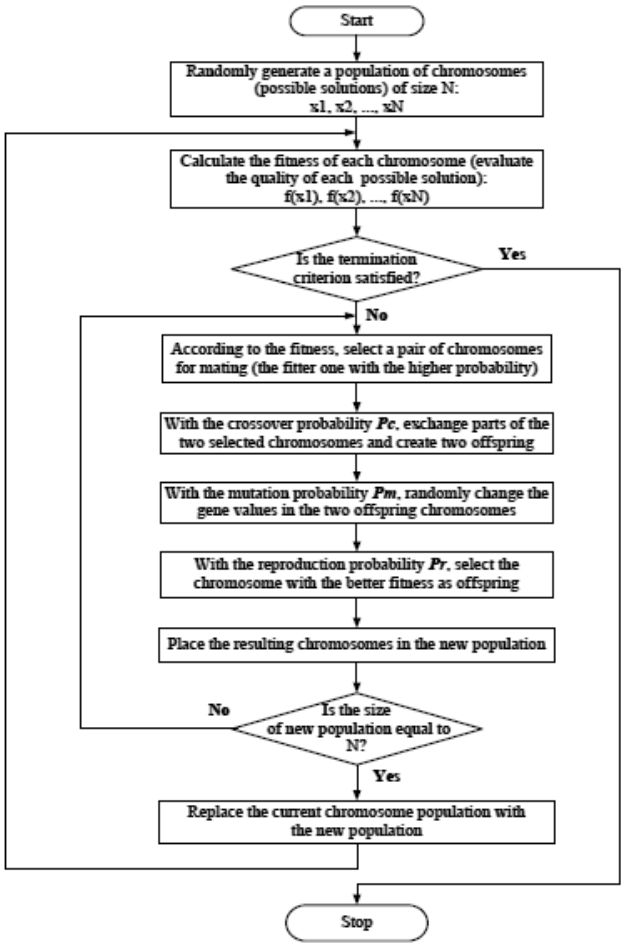


Figure 4. The execution flow of GA/HTGA implementation

The HTGA method incorporated the Taguchi method into the crossover operation of GA for creating offspring. The selection operation is the roulette wheel selection. The parameters of GA/HTGA used in this experiment are shown in Table 3.

Four different sizes of knapsack problems were selected and randomly ran 5 times for each size. The parameters of these problems were defined in Table 4.

Table 3. The parameters of GA/HTGA for solving the 0-1 knapsack problems

Parameters	Values
Population Size (N)	100
Pc	0.7
Pm	0.02
Iteration number	10000

Table 4. The parameters of the 0-1 knapsack problems

Parameters	Values
Object (n)	100, 200, 300, 400
Cost range (ci)	1~1000
Weight range (wi)	1~1000
Capacity bound (W)	10000, 30000, 40000, 50000

Experimental results (shown in Table 5) show that the performances of GA were better than HTGA which were different from the results obtained for optimization problems with continuous variables. The Taguchi method used in GA could not improve the performance of GA for optimization problems with discrete variables.

Fig 5 showed one of the performance curves of GA and HTGA for the 0-1 knapsack problem with n = 400 objects. The fitness value is defined as the negative of the cost for easy representation. The solution of GA is much better than that of HTGA. The fitness function is shown as follow:

$$\text{fitness} = Min.(-\sum_{i=1}^{n} c_i x_i) \tag{8}$$

Table 5. The parameters of the 0-1 knapsack problems

	100		200		300		400	
Methods	Average	SD	Average	SD	Average	SD	Average	SD
GA	29685.2	66.1	59792.0	73.8	87407.0	213.1	114938.8	68.2
HTGA	29669.6	73.8	59490.8	70.1	84050.4	42.2	100408.0	188.2

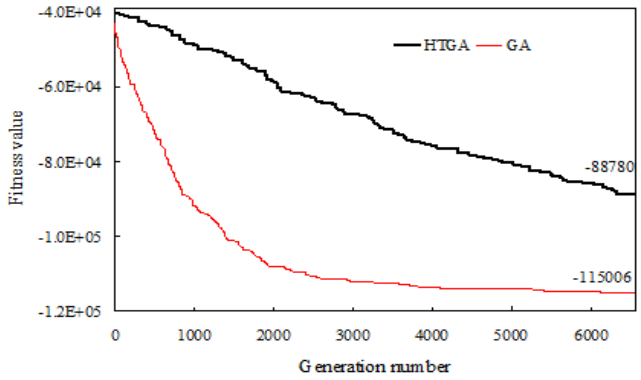


Figure 5. The performance curves for GA and HTGA for 0-1 knapsack problems

Traveling Salesman Problem: The traveling salesman problem is also an NP-hard problem [16, 17]. This problem can be stated as the following. A salesman wants to visit n cities and finishes at the city he starts from. Each city can be visited exactly once, and one city is visited at each time. There is a nonnegative cost c (i, j) to travel from city i to city j. The salesman wants to find a shortest tour for visiting n cities and returns to the starting city. It can be formulated as the following equations:

$$Min.[\sum_{i=1}^{n-1} c(x_i, x_{i+1}) + c(x_n, x_1)], \quad 1 \le i \le n-1. \tag{9}$$
$$s.t.\ x_i \neq x_j, \forall i \neq j,\ 1 \le i, j \le n,$$

where x_i is the index of visited city at time i.

We can use an n-gene length chromosome to represent a tour. Suppose we have five cities numbered from 1 to 5. In a chromosome, the order of the integers represents the order in which the cities will be visited by the salesman. For example:

[1][5][3][2][4] represents the route 1→ 5 → 3 → 2 → 4 →1

The GA crossover operation used[12], and the execution flow of GA for solving the traveling salesman problem is similar to Fig 4. The parameters of GA (HTGA) used in this experiment were shown in Table 3.

Four different sizes of traveling salesman problems were selected and randomly generated 5 times for each size. The parameters of the problem were defined in Table 6.

Table 6. The parameters of the traveling salesman problem

Parameters of Traveling Salesman Problems					
Parameters	Values				
City (n)	10	20	30	40	50
Cost range (c(i, j))	[0, 10]	[0, 20]	[0, 30]	[0, 40]	[0, 50]

Experimental results (see Table 7) show that the performances of GA were better than HTGA which were different from the results obtained for optimization problems with continuous variables. Fig 6 (a) and (b) show the solutions of TSP by using HTGA and GA. The Taguchi method used in GA (HTGA) could not improve the performance of GA for optimization problems with discrete variables.

Table 7. The experimental results of HTGA/GA for traveling salesman problem

Methods	Problem size (n)				
	10	20	30	40	50
GA	30.9	76.9	148.4	214.6	302.7
HTGA	30.9	76.9	154.5	221.8	308.9

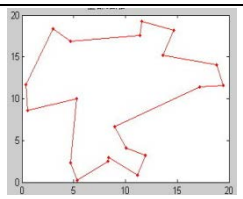
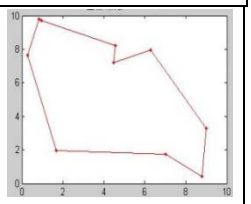
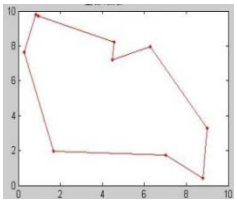
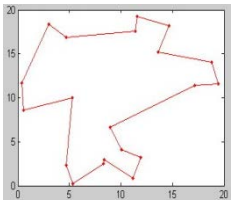
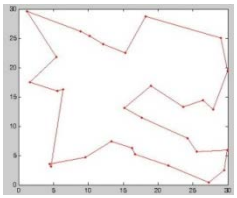
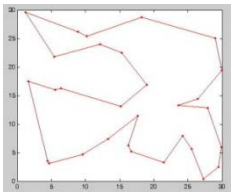
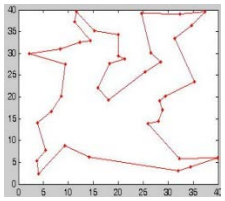
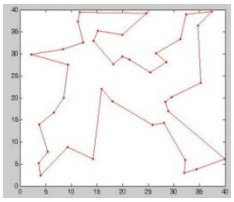
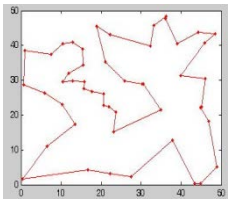
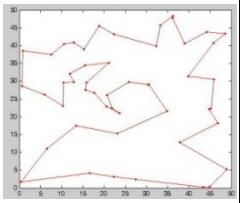
City no. (n)	GA	HTGA
10	 Tour length =30.9	 Tour length =30.9
20	 Tour length =76.9	 Tour length =76.9
30	 Tour length =146.1	 Tour length =164.3
40	 Tour length =218.8	 Tour length =228.4
50	 Tour length =300.5	 Tour length =309.4
	(a)	(b)

Figure 6. One of solutions for each size n (10 ~ 50) by using (a) GA and (b) HTGA

CONCLUSION

Two typical optimization problems with discrete variables and constraints are used to evaluate the performance of HTGA and GA. We can find that the Taguchi method in HTGA can't find better results than GA. The Taguchi method became a restriction for finding the optimal solution. The following example shows the weakness of HTGA during solving an optimization problem with discrete variables and constraints.

1) Example: Suppose a 0-1 knapsack problem with 5 objects, and each object i is worth ci dollars and weighs wi pounds, where ci = {2, 3, 4, 5, 9} and wi = {3, 3, 4, 4, 8}, respectively. The capacity of knapsack is 10 (W=10) pounds. Find the maximum cost under the constraint (i.e., capacity of the knapsack less than or equal to 10).

2) Sol: When we applied the GA/HTGA to solve this problem, if two chromosomes were obtained as following: C1: [0, 0, 1, 1, 0] and C2: [1, 1, 0, 0, 0]. The off-spring generated by the HTGA and GA would be:

a) HTGA method: We can define the levels 1 and 2 for factors A, B, C, D and E are {0, 0, 1, 1, 0} and {1, 1, 0, 0, 0}, respectively. Factors F and G are dummy (useless) and can be defined as all 0s. Based on the orthogonal array (defined in Table 1), the genes for each experiment can be obtained as Table 8. The fitness is defined as the cost of selected objects if it is satisfied the constraint (i.e., the total weight of selected objects is less than or equal to 10), and the value of ηk is set to the square of fitness.

In experiment 7, three objects were selected (B, C, D) and the total weight of these objects is 11 beyond the capacity (=10) (violating the constraints). Then, experiment 7 is an invalid solution and is not considered by the HTGA. We can obtain the values of E_{f1} and E_{f2} by the steps of Table 2, and a better offspring [0, 1, 1, 1, 0] is generated but it is invalid. HTGA can't consider to approaching the optimal solution and satisfying constraints simultaneously. It is the major weakness of this method.

Table 8. The genes of each experiment and an invalid offspring generated from two parent chromosomes C1 and C2 by using Taguchi method

Experiment index	Factors							fitness f	η_k
	A	B	C	D	E	F	G		
	1	2	3	4	5	6	7		
1	1	2	3	4	5	6	7	9	81
2	0	0	1	1	0	0	0	4	16
3	0	0	1	0	0	0	0	8	64
4	0	1	0	1	0	0	0	3	9
5	0	1	0	0	0	0	0	7	49
6	1	0	0	1	0	0	0	2	4
7	1	0	0	0	0	0	0	NA*	NA*
8	1	1	1	1	0	0	0	9	81
E_{f1}	1	1	1	0	0	0	0		
E_{f2}	170	150	178	194	pass	pass	pass		
Optimal level	134	154	126	110					
Offspring chromosome	1	2	1	1	1, 2	0	0		

(NA*: means an invalid solution)

b) GA method: Based on GA, the chromosomes C1: [0, 0, 1, 1, 0] and C2: [1, 1, 0, 0, 0] may generate two off-spring C1': [0, 0, 1, 0, 0] and C2': [1, 1, 0, 1, 0] by setting the cut-point between the 3rd and the 4th gene in the crossover operation. Then, the optimal solution [1, 1, 0, 1, 0] (max cost = 10) is found (=C2').

From the above example, we can clearly realize that the HTGA may not to approaching the optimal solution and satisfying constraints simultaneously. Experimental results showed that the performances of HTGA are better than GA for problems with continuous variables, but worse than GA for problems with discrete variables and constraints. Combinatory optimization problems are very difficult to find the optimal solutions within reasonable time consumption. In the future, a new technique which may be similar to Taguchi method would be developed and incorporated into GA to solve the optimization problem with discrete variables and constraints

REFERENCES

[1] Tsai, J. T., Liu, T. K., and Chou, J. H., Hybrid Taguchi-genetic algorithm for global numerical optimization, *IEEE transactions on evolutionary computation*, Vol. 8, pp. 365-377, 2004.

[2] Taguchi, G., Chowdhury, S., and Taguchi, S, Robust engineering, *New York: McGraw-Hill*, 2000.
[3] Gen, M. and Cheng, R., Genetic algorithms and engineering design, *New York: Wiley*, 1997.
[4] Chou, J. H., Chen, S. H., and Li, J. J., Application of Taguchi-genetic method to design optimal grey-fuzzy controller of a constant turning force system, *Journal of materials processing technology*, Vol. 105, pp. 333-343, 2000.
[5] Hsieh, C. H., Chou, J. H., and Wu, Y. J., Taguchi-MHGA method for optimizing Takagi-Sugeno fuzzy gain-scheduler., In: *Proceedings 2000 automatic control conference*, pp. 523-528, 2000.
[6] Mitra, A., The Taguchi method., *Wiley Interdisciplinary Reviews: Computational Statistics*, Vol. 3, No. 5,pp. 472-480, 2011.
[7] M. S. Gök, Y. Küçük, O. Gencel, V. Koç and W. Brostow, Application of Taguchi Method to Study Abrasive Wear Behavior of Ceramic-Coated Specimens with Plasma Technique, *Mechanics of Advanced Materials and Structures*, Vol.18, No. 6, pp. 389-395, 2011.
[8] Braun, C., Faville, J., Geib, T., and Buehler, D., Acoustic Noise Assessment of Gasoline Direct Injection Fuel Injectors Using Taguchi Methods, *SAE International*, doi:10.4271/2011-01-1216, 2011.
[9] Rabeah M., Mohsen J., Ghasem D. Najafpour and Naser S.,Applying the Taguchi Method for Optimized Fabrication of Lactalbumin Nanoparticles as Carrier in Drug Delivery and Food Science, *lamiIranica Journal of Energy & Environment*, Vol. 2, No. 1,pp. 87-91, 2011.
[10]Wu, Y., Taguchi methods for robust design, *New York: ASME*, 2000.
[11]Gen, M. and Cheng, R., Genetic algorithms and engineering design, *New York: Wiley*,1997.
[12]Goldberg, D. E., Genetic algorithms in search, optimization & machine learning, *Mitchell: Addison Wesley*,1989.
[13]J Grefenstette, Optimization of Control Parameters for Genetic Algorithms, IEEE Trans. Syst, Man Cybern, Vol. 16, pp. 122-28, 1986.
[14]Horowitz, E., Sahni, S., and Rajasekaran, S. Computer algorithms, *NJ: Computer Science Press*, 1998.
[15]Mitchell, M., An introduction to genetic algorithms, *MIT press*, 1996.
[16]Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C., Introduction to algorithms, *Cambridge: MIT Press*, 2009.
[17]Mitchell, M., An introduction to genetic algorithms, *MIT press*, 1996.
[18]Martello, S. and Toth, P., Knapsack problems, *Jonh Wiley*, UK,1990.
[19]Lin, W. Y., Lee, W. Y., and Hong, T. P., Adapting crossover and mutation rates in genetic algorithms, *Journal of Information Science and Engineering*, Vol.19, pp. 889-903, 2003.