A Method to Acquire Multiple Satisfied Solutions

TOMOHIRO YOSHIKAWA[†1] KOUKI MARUYAMA[†1]

Abstract: In general, the main purpose of Genetic Algorithm (GA) is to acquire a solution with the highest evaluation value in a single-objective problem or Pareto solutions with various evaluation values in a multi-objective problem. However, in engineering problems, the acquisition of multiple satisfied solutions satisfying certain conditions is often more strongly desired than acquiring a single best solution. In addition, to help set design choices, satisfied solutions should satisfy different design variable patterns from one another. There are multiple objective functions and rather than being maximized/minimized these are intended to approximate certain target values. These multiple objective functions can be unified into a single-objective function by summing up the errors from the target values. Through this unification of objective functions, computing resources for searching can be assigned in terms of the diversity in the design variable space rather than the objective space. In this paper, a method for acquiring multiple satisfied solutions by GA in many constrained multi-objective optimization problems is proposed. The proposed method is applied to a real-world problem and compared with Island model to investigate its performance.

Keywords: Satisfied Solutions, Design Variables Pattern, Unification of Objectives, Genetic Algorithm, Island Model

1. Introduction

In addition to improving the performance of computers, Genetic Algorithm (GA) is actively applied to engineering problems. GA is an optimization method that imitates the evolution of creatures. In general, the main purpose of GA is to acquire a solution with the highest evaluation value in a single-objective problem or Pareto solutions with various evaluation values in a multi-objective problem. In both cases, evaluation values are the highest priority, and the variety of individuals is considered in the objective space. However, in engineering problems, the acquisition of multiple satisfied solutions satisfying certain conditions is often more strongly desired than acquiring a single best solution [1]. In addition, to help set design choices, satisfied solutions should satisfy different design variable patterns from one another.

Because of the characteristics of GA, when applying it to the acquisition of satisfied solutions and after a satisfied solution is acquired, searches of the population are intensively performed very close to the acquired solution because individuals with slight differences from the satisfied solution could also be satisfied solutions. As a result, many satisfied solutions in the design variable space that are very similar to the first one are often acquired. These similar solutions usually have no practical meaning. Many methods that can maintain the diversity of design variables have been proposed [2-4]. However, these methods aim to prevent solutions from converging to local solutions by maintaining the diversity of design variables, rather than acquiring various types of satisfied solutions.

In the case of applying GA to multi-objective optimization problems, searches are performed to acquire various and uniform solutions in the objective space [5]. In this case, the diversity of design variables is generally not considered. Thus, various solutions are acquired in the objective space rather than in the design variable space. In general, different solutions in the objective space have different design variables. However, there is no guarantee that solutions have different design variables.

In contrast, there are multiple objective functions, which approximate certain target values rather than being maximized/minimized. These multiple objective functions can be unified into a single-objective function by summing up the errors from the target values. Through this unification of objective functions, computing resources for searching can be assigned in terms of the diversity in the design variable space rather than the objective space.

In this study, a method for acquiring multiple satisfied solutions in unified single-objective optimization problems using GA is proposed. To investigate the effectiveness of the proposed method, an experiment is conducted. In the experiment, the proposed method is applied to a two-objective optimization problem with many constraints [6] and compared with Island model [4] which is one of the most representative methods to maintain the diversity of design variables.

2. Unification of Objectives

2.1 Multi-objectives optimization problems

There are multiple objective functions. When these functions are not to be maximized/minimized but rather approximated to certain target values, they can be unified into a single objective function by summing up the errors from the target values in each objective function. The formulas for calculating this unification of objectives are shown in eqs. (1), (2), and (3).

$$\min F = \sum_{i=1}^{m} |\hat{f_i}| \tag{1}$$

$$\hat{f_i} = \frac{\bar{f_i}}{\bar{f}_i^{max}} \quad (i = 1,2,\ldots,m) \tag{2}$$

$$\bar{f_i} = \begin{cases} \max(|f_{ti} - f_i| - th_i, 0) & (a) \\ \max(f_{ti} - f_i, 0) & (b) \\ \max(f_i - f_{ti}, 0) & (c) \end{cases} \tag{3}$$

$\bar{f}_i^{max}$is the maximum value of $\bar{f_i}$ in all $\bar{f_i}$. When a function aims to keep the error from the target value f_{ti}, within an allowable

[†1] Graduate School of Engineering, Nagoya University

value th_i, (a) is selected as $\overline{f_\iota}$. When a function aims to obtain a larger value than the target f_{ti}, (b) is selected as $\overline{f_\iota}$. When a function aims to obtain smaller value than the target f_{ti}, (c) is selected as $\overline{f_\iota}$. An individual for which F is equal to 0 is a satisfied solution, which indicates that all functions satisfy the given conditions.

2.2 Many-constrained

In optimization problems with many constraints, the unification below can be applied [7, 8].

$$\min F = \sum_{i=1}^{m} |\widehat{f_\iota}| + \sum_{i=1}^{l} |\widehat{g_\iota}| \qquad (4)$$

$$\widehat{g_\iota} = \frac{\overline{g_\iota}}{\overline{g}_i^{max}} \quad (i = 1,2,\dots,l) \qquad (5)$$

Here, $\widehat{f_\iota}$ is same as in eq. (2) in 2.A, $\overline{g_\iota}$ is the amount by which the i-th constraint is violated, and $\overline{g}_i^{max}$ is the maximum value of all $\overline{g_\iota}$. In this case, an individual for which F is equal to 0 is a feasible and satisfied solution, as in constrained multi-objective optimization problems.

3. Proposed Method

3.1 Flow of proposed methods

First, initial individuals are generated randomly, then "neighbors" are defined. When $r\_neighbor > d_{xy}$ is true, *x* and *y* are defined as mutual neighbors, where *x* and *y* are individuals, $r\_neighbor$ is the neighbor range, which is input in advance, and d_{xy} is the distance between *x* and *y* in the design variable space. After defining neighbors, one child (C) is generated. Then, the child's neighbors ($\boldsymbol{C}_n$) are defined. When $\boldsymbol{C}_n$ contains at least one satisfied solution, the child is not evaluated, and the generation of a child is repeated. When $\boldsymbol{C}_n$ does not contain satisfied solutions, the child is evaluated. After the evaluation, the population is selected according to the flow. In the flow, $|\boldsymbol{A}|$ is the number of individuals in $\boldsymbol{A}$, where $\boldsymbol{A}$ is a set of some individuals. Furthermore, $f(x) > f(y)$ denotes that *y*'s evaluation value is better than that of *x*, and $x \leftarrow y$ indicates that the information of *x* is updated, including *y*. After the selection, the generation of a child is repeated. This process is repeated until the end condition is satisfied.

3.2 Feature of proposed method

- Distributing computing resources dynamically
  - — When a satisfied solution exists among the neighbors for a new child, the child is deleted without evaluation to assign computing resources to search other areas (see Fig. 1(a)).
- Sequential update
  - — Like MOEA/D [9], a good child with a high fitness value can become a parent immediately. Thus, high convergence can be expected.
- Defining neighbors in the design variable space
  - — Defining neighbors using the neighbor radius in the design variable space can enact a group search. The use of group search leads to diversity being maintained, and sometimes results in high convergence [10] in each group search.
  - — We can adjust the granularity of the distance between acquired satisfied solutions in the design variable space. When the neighbor range is large, the distance between satisfied solutions is expected to be large, and *vice versa*.
- Neighborhood crossover
  - — High convergence can be expected because of neighborhood crossovers [11, 12].
- Mechanism for maintaining diversity in the design variable space

When the number of neighbors ($|\boldsymbol{C}_n|$) for a new child (C) is greater than the maximum neighbor population (n_{max}) and $f(C)$ is better than $f(C_n^{nad})$, the child replaces C_n^{nad}, and the information on the worst individual is updated (C_n^{nad}: the worst individual among the child's neighbors, see Fig. 1(b)). When the number of neighbors ($|\boldsymbol{C}_n|$) for a new child (C) is less than n_{max} and the number of neighbors $|\boldsymbol{I}_n^{max}|$ is greater than n_{max}, the child replaces $\boldsymbol{I}_{n_{nad}}^{max}$, and the information on the worst individual is updated (see Fig. 1(c)).

4. Experiment

In this study, an experiment was conducted. In the experiment, an engineering problem in the real world [6] was considered. This is a constrained two-objective optimization problem. The problem comprises 222 design variables, 54 constraints, and two-objective functions (f_1 is minimized, and f_2 is maximized). In constrained optimization problems, feasible solutions are defined as those that satisfy all constraints [13]. In this problem, satisfied solutions are also defined as those that satisfy certain conditions (evaluation values are less than target values or greater than target values) among feasible solutions. In the experiment, the conditions were set to $f_1 \leq 3.0$ and $f_2 \geq 34$. These values are those introduced as the evaluation values of the solution designed by a human in the benchmark problem [6].

4.1 Problem settings

In the experiment, the searches using Island model and the proposed method with the unification of the objective functions and constraints described in Section II were compared. In the proposed method, plural groups are generated by defining neighbors based on neighbor range, which gives similar feature for maintaining diversity in the design variable space. Thus Island model was compared with the proposed method.

4.2 Experimental conditions

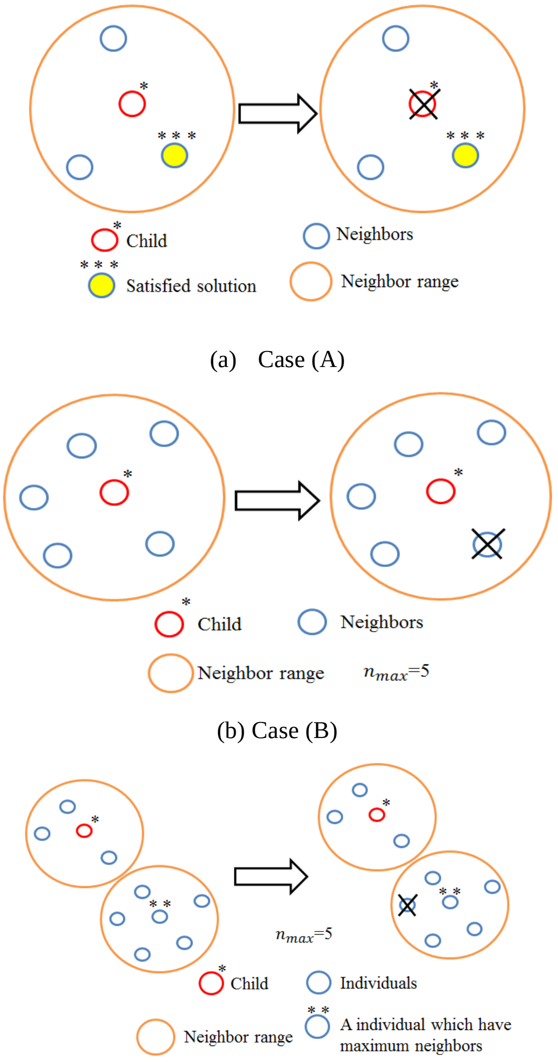
In the searches using Island model and the proposed method, the numbers of individuals, evaluations, and trials were 100, 30,000, 21, respectively. The initial population for both methods was the same in every trial. In the search using the proposed method, the maximum neighbor population n_{max} was 15, the neighbor range $r\_neighbor$ was 22.2 using the Manhattan distance, and the crossover rate with neighbors P_r was 0.7. In the search using Island model, the number of islands was 5, 10, 15. From the result of the pre-experiment, there was no migration.

4.3 Results

The results for Island model are shown in Tables 2 to 4. Table 2 shows the number of islands which succeeded in acquiring satisfied solutions and the number of satisfied solutions. The number of satisfied solutions decreased as the number of islands increased (see Table 2). It is thought that the number of individuals for one island decreased as the number of islands increased. Thus the number of islands which could acquire satisfied solutions decreased because of low convergence by small number of individuals. Table 3 shows the distance between satisfied solutions in each island in a trial whose number of acquired satisfied solutions was the median in 21 trials in the case that the number of islands was 5. In this trial, 4 islands could acquire satisfied solutions, and each island was named “island 1” to “island 4.” The distances between satisfied solutions in each island were very small, which shows that very similar satisfied solutions were acquired (see Table 3). It is thought that when a satisfied solution was acquired on a certain island, very similar solutions also became satisfied solutions and were acquired by intensively searching around the satisfied solution. In practice, these similar satisfied solutions are regarded to be only one satisfied solution. Thus, in this trial, the substantial number of acquired satisfied solutions were 4, which was the number of islands which succeed in acquiring satisfied solutions. In Island model, because there is no mechanism for controlling the distance between islands, it is possible that the distance between islands become small and the diversity of the design variable space cannot be maintained. This tendency has seen by migration in Experiment 1. The distance between islands which could acquire satisfied solutions (one island is regarded as one satisfied solution) is shown in Table 4. The distance between islands was calculated as the distance between their center of gravity of satisfied solutions in each island. The distance between islands was sufficiently large (see Table 4). The influence of the number of islands to the distance between satisfied solutions was very small. In other words, regardless of the number of islands, it is considered that the distance between islands would be around 50 in Manhattan distance in Island model without migration. Although it is possible to acquire satisfied solutions in Island model, it is difficult to adjust the distance between islands, that is, satisfied solutions expressly.

The results for the proposed method are shown in Table 5 and 6. Table 5 shows the number of satisfied solutions acquired by the proposed method. The smaller the neighbor range was, the more satisfied solution were acquired (see Table 5). Table 6 shows the distance between satisfied solutions. It was confirmed that the granularity of the distance between satisfied solutions was adjustable by changing the neighbor range $r\_neighbor$.

It was confirmed that in Island model without migration, diverse satisfied solutions were also acquired. However, because Island model does not explicitly give the distance between islands, it is difficult to adjust the granularity of the satisfied solutions, while the granularity of the satisfied solutions can be adjusted in the proposed method by changing the neighbor range. The evaluation value of satisfied solutions in island model is shown in Table 7 and in the proposed method is shown in Table 8. There was no big difference in both methods (see Table7 and 8).



(a) Case (A)

(b) Case (B)

(c) Case (C)

Figure 1: Mechanism for maintaining diversity

Table 2: Number of islands which succeeded in acquiring satisfied solutions and number of satisfied solutions using Island model (no migration)

	Number of islands		
	5	10	15
Number of islands which succeed in acquiring satisfied solutions	4	2.6	1.3
Number of satisfied solutions	80	25.7	8.4

Table 3: Distance between satisfied solutions in each island (Manhattan distance)

	island 1	island 2	island 3	island 4
Min.	0.05	0.05	0.05	0.05
Max.	1.70	1.45	2.35	1.50
Ave.	0.61	0.41	0.75	0.71

Table 4: Distance between islands (Manhattan distance)

	Number of islands		
	5	10	15
Min.	42.42	47.27	45.53
Max.	52.18	50.69	48.26
Ave.	48.15	49.04	47.05

Table 5: Number of satisfied solutions
in the proposed method

	Neighbor range		
	4.4	8.8	22.2
Number of satisfied solutions	14.24	4.95	3.9

Table 6: Distance between satisfied solutions
in the proposed method (Manhattan distance)

	Neighbor range		
	4.4	8.8	22.2
Min.	5.03	16.66	43.69
Max.	25.05	38.69	50.6
Ave.	14.78	31.14	47.54

Table 7: Evaluation values of satisfied solutions
in Island model

		Number of islands		
		5	10	15
f1	Ave.	2.993	2.993	2.994
	Std.	0.004	0.004	0.002
f2	Ave.	34.03	34.04	34.08
	Std.	0.12	0.12	0.14

Table 8: Evaluation values of satisfied solutions
in the proposed method

		Neighbor range		
		4.4	8.8	22.2
f1	Ave.	2.994	2.993	2.993
	Std.	0.005	0.005	0.004
f2	Ave.	34.08	34.01	34.01
	Std.	0.20	0.02	0.02

5. Conclusion

In this study, the unification of objective functions in multi-objective optimization problems and many-constraint optimization problems was introduced. This paper proposed a method for acquiring multiple satisfied solutions in unified single-objective optimization problems. To investigate the effectiveness of the proposed method, an experiment was conducted. In the experiment, a 54 constraint two-objective optimization problem was considered, and the proposed method and Island model were compared. The results showed that both Island model and the proposed method could acquire diverse satisfied solutions in the design variable space. The results also showed that the proposed method could adjust the granularity of the distance between acquired satisfied solutions in the design variable space while Island model could not. In the experiment, although satisfied solutions could be acquired in all trials, there was one trial in which only one satisfied solution could be acquired. Because the purpose of this study was to acquire various satisfied solutions in engineering problems, multiple satisfied solutions should be acquired in all trials. A study of the appropriate neighbor range is necessary. Further studies are also needed in order to make the proposed method more suitable for engineering problems.

Reference

[1] K.Deb, “Evolutionary Algorithms for Multi-Criterion Optimization in Engineering Design,” Evolutionary Algorithms in Engineering and Computer Science 2, pp. 135-161, 1999.

[2] L.Nguyen, L,Bui, H.Abbass, “A new niching method for the direction-based multi-objective evolutionary algorithm,” IEEE Symposium on Computational Intelligence in Multi-Criteria Decision-Making, pp. 1-8, 2013.

[3] H,Toshio, N.Kenta, “Structural Morphogenesis by the Genetic Algorithms Considering Diversity of Solution” J.Struct. Constr. Eng. AIJ, No.614, pp35-43, 2007.

[4] D.Whitely, R.Soraya, H.Robert B, "Island model genetic algorithms and linearly separable problems.", In: AISB International Workshop on Evolutionary Computing. Springer, Berlin, Heidelberg. pp. 109-125, 1997

[5] H.Li, Q.Zhang, “Multiobjective Optimization Problems With Complicated Pareto Sets, MOEA/D and NSGA-II,” IEEE Transactions on Ecolutionary Computation, Vol. 13, No. 2, pp. 284-302, 2009.

[6] T.Kohira, H.Kemmotsu, A.Oyama, T.Tatsukawa, “Proposal of Simultaneous Design Optimization Benchmark Problem of Multiple Car Structures Using Response Surface Method,” The Japanese Society for Evolutionary Computation, 2017. http://ladse.eng.isas.jaxa.jp/benchmark/index.html.

[7] K.Deb, “An efficient constraint handling method for genetic algorithms,” Computer Methods in Applied Mechanics and Engineering, Vol. 186, No. 2, pp. 311-338, 2000.

[8] E.Mezura-Montes, “Constraint-Handling in Evolutionary Optimization,” Springer, 2009.

[9] Q.Zhang, H.Li, “MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition,” IEEE Transactions on Evolutionary Computation, Vol. 11, No. 6, pp. 712-731, 2007.

[10] D.Whitley, S.Rana, R.B.Heckendorn, “Island model genetic algorithms and linearly separable problems,” AISB International Workshop on Evolutionary Computing, Vol. 1305, pp.109-125, 1997.

[11] S.Kikuchi, T.Suzuki, “The Effect of Neighborhood Crossover in Evolutionary Search Methods for Landscape Photograph Geocoding Support,” IPSJ SIG Technical Report, Vol. 2010-FI-98, No. 10, 2010.

[12] S.Watanabe, T.Hiroyasu, M.Miki, “NCGA: Neighborhood Cultivation Genetic Algorithm for Multi-Objective Optimization Problems,” Proceedings of the Genetic and Evolutionary Computing Conference, pp. 458-465, 2002.

[13] E.Mezura-Montes, C.A.C.Coello, “Constraint-handling in nature-inspired numerical optimization: Past present and future,” Swarm and Evolutionary Computation, Vol. 1, No. 4, pp. 173-194, 2011.

[14] JB.Kruskal, M.Wish, “Multidimensional scaling” Vol.11, Sage, 1978.