International Journal of Computational Intelligence Research.
ISSN 0973-1873 Vol.1, No.1 (2005), pp. 1-18

Handling of Overlapping Objective Vectors in Evolutionary Multiobjective Optimization

Hisao Ishibuchi[1], Kaname Narukawa[2] and Yusuke Nojima[3]

Department of Computer Science and Intelligent Systems,
Osaka Prefecture University, Sakai,
Osaka 599-8531, Japan
[1]E-mail: hisaoi@cs.osakafu-u.ac.jp, [2]E-mail: kaname@ci.cs.osakafu-u.ac.jp, [3]E-mail: nojima@cs.osakafu-u.ac.jp

Abstract: **Recently a number of evolutionary multiobjective optimization (EMO) algorithms have been proposed to find a variety of well-distributed Pareto-optimal or near Pareto-optimal solutions with a wide range of objective values. We focus on the handling of overlapping objective vectors in the objective space. First we show that there exist a large number of overlapping objective vectors in each population when EMO algorithms are applied to multiobjective combinatorial optimization problems with only a few objectives. We discuss the number of overlapping objective vectors from a viewpoint of the diversity-convergence balance in the objective space. Next we implement two strategies to handle overlapping objective vectors. One strategy is the removal of overlapping objective vectors (i.e., overlapping solutions in the objective space). In this strategy, overlapping objective vectors are removed during the generation update phase except for only a single solution among them. As a result, each solution in the next population has a different location in the objective space. The other strategy is the removal of overlapping decision vectors (i.e., overlapping solutions in the decision space) so that each solution in the next population has a different location in the decision space. In this strategy, the next population may include overlapping objective vectors because different solutions in the decision space are not necessarily different in the objective space. Finally we examine the effect of each strategy on the performance of EMO algorithms through computational experiments on multiobjective 0/1 knapsack problems.**

Keywords: Diversity preserving mechanisms, evolutionary multiobjective optimization (EMO), multiobjective combinatorial optimization, multiobjective genetic algorithms.

I. Introduction

Since Schaffer's work [1], a large number of evolutionary multiobjective optimization (EMO) algorithms have been proposed to find a variety of well-distributed Pareto-optimal or near Pareto-optimal solutions with a wide range of objective values (e.g., see Deb [2] and Coello *et al*. [3]). Whereas various diversity preserving mechanisms have been discussed in the design of EMO algorithms, the handling of overlapping objective vectors in the objective space has not been explicitly discussed. One reason for such little attention to overlapping objective vectors is that the performance of EMO algorithms has been often evaluated through computational experiments on multiobjective optimization problems with continuous decision variables. Since almost all EMO algorithms have diversity preserving mechanisms, many overlapping objective vectors are not likely to exist in each population when they are applied to multiobjective optimization problems with continuous decision variables and/or many objectives. On the other hand, the handling of overlapping objective vectors seems to be an important issue in the application of EMO algorithms to multiobjective combinatorial optimization problems with only a few objectives. In such an application, there may exist a large number of overlapping objective vectors in each population.

First we examine whether there exist a large number of overlapping objective vectors in each population through computational experiments on a number of test problems. As a representative EMO algorithm, we use the NSGA-II algorithm of Deb *et al*. [4] because it is one of the most well-known and frequently-used EMO algorithms in the literature. As test problems, we use Min-Ex in Deb [2], ZDT1, ZDT2 and ZDT3 in Zitzler *et al*. [5], multiobjective 0/1 knapsack problems in Zitzler and Thiele [6], multiobjective flowshop scheduling problems in Ishibuchi *et al*. [7], and multiobjective fuzzy rule selection problems in Ishibuchi and Yamamoto [8]. It is shown that a large number of solutions are overlapping with each other in the objective space in the application of the NSGA-II algorithm to combinatorial optimization problems while this is not the case in the application to function optimization problems. We further discuss experimental results from a viewpoint of the diversity-convergence balance in the objective space. More specifically, we monitor the number of overlapping objective vectors together with several performance indices that measure the convergence of solutions to the Pareto front and

the diversity of solutions. We also examine the effect of incorporating an additional diversity preserving mechanism into the NSGA-II algorithm on the number of overlapping objective vectors.

Next we implement two strategies to handle overlapping objective vectors. One strategy is the removal of overlapping objective vectors (i.e., overlapping solutions in the objective space). Overlapping solutions in the objective space are removed during the generation update phase except for a randomly chosen single solution among them. As a result, each solution in the next population has a different location in the objective space. The other strategy is the removal of overlapping decision vectors (i.e., overlapping solutions in the decision space). Duplicate copies are removed during the generation update phase so that each solution in the next population has a different location in the decision space (i.e., so that they are different from each other). It should be noted that overlapping objective vectors are not necessarily the same solution in the decision space. That is, overlapping objective vectors are not always overlapping with each other in the decision space. Thus each population may include overlapping objective vectors when we use the second strategy.

Effects of each strategy on the performance of the NSGA-II algorithm are examined through computational experiments on multiobjective 0/1 knapsack problems. We show that the two strategies slightly improve the performance of the NSGA-II algorithm. We also show that the NSGA-II algorithm is significantly improved by the two strategies when they are used together with the tournament selection scheme with large tournament size based on the weighted sum fitness function. That is, the removal of overlapping objective vectors plays an important role under the high selection pressure based on the weighted sum fitness function.

This paper is organized as follows. First we explain basic concepts of multiobjective optimization in Section II. Next we examine the existence of overlapping objective vectors in each population through computational experiments on a number of test problems in Section III. Then we examine the effects of the two strategies for the handling of overlapping objective vectors in Section IV. Finally we conclude this paper in Section V.

II. Multiobjective Optimization

A. Multiobjective Optimization Problem

Let us consider the following k-objective maximization problem:

$$\text{Maximize } \mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}),\ f_2(\mathbf{x}),\ ...,\ f_k(\mathbf{x})), \quad (1)$$

$$\text{subject to } \mathbf{x} \in \mathbf{X}, \quad (2)$$

where $\mathbf{f}(\mathbf{x})$ is an objective vector, $f_i(\mathbf{x})$ is the ith objective function to be maximized, $\mathbf{x}$ is a decision vector, and $\mathbf{X}$ is the feasible region in the decision space.

If there exists a solution $\mathbf{x}^*$ that satisfies the following relations, $\mathbf{x}^*$ is said to be the absolutely optimal solution of the multiobjective optimization problem in (1)-(2):

$$f_i(\mathbf{x}^*) = \max\{ f_i(\mathbf{x}) \mid \mathbf{x} \in \mathbf{X}\} \text{ for } i = 1, 2, ..., k. \quad (3)$$

In general, multiobjective optimization problems do not have such an absolutely optimal solution that is optimal with respect to all objectives. This is because different objectives usually conflict with each other in multiobjective optimization. Thus a different concept of optimal solutions, which is defined based on a dominance relation between two solutions, is often used in the field of multiobjective optimization.

Let $\mathbf{x}$ and $\mathbf{y}$ be two feasible solutions of the multiobjective maximization problem in (1)-(2). The solution $\mathbf{y}$ is said to dominate the solution $\mathbf{x}$ (i.e., $\mathbf{y}$ is better than $\mathbf{x}$) when the following relations hold:

$$^{\forall} i,\ f_i(\mathbf{x}) \le f_i(\mathbf{y}) \text{ and } ^{\exists} i,\ f_i(\mathbf{x}) < f_i(\mathbf{y}). \quad (4)$$

If there exists no feasible solution $\mathbf{y}$ that dominates $\mathbf{x}$, $\mathbf{x}$ is said to be a Pareto-optimal solution. In this case, $\mathbf{x}$ is optimal in the sense that $\mathbf{x}$ is not dominated by any other feasible solution. The set of all Pareto-optimal solutions is the Pareto-optimal solution set. We denote the Pareto-optimal solution set by S^* in this paper. The image of the Pareto-optimal solution set onto the objective space is called the Pareto front. That is, the Pareto front is the Pareto-optimal solution set in the objective space.

Let S be a set of feasible solutions. When no solution in S is dominated by any other solution in S, we call S a non-dominated solution set in this paper.

B. Performance Indices of Solution Sets

Whereas EMO algorithms try to find all Pareto-optimal solutions, it is not always easy to find true Pareto-optimal solutions of multiobjective optimization problems. Thus an EMO algorithm usually presents a non-dominated solution set S to the decision maker at the end of its execution where S can be viewed as an approximation for the Pareto-optimal solution set.

A number of performance indices have been proposed in the literature to evaluate non-dominated solution sets (e.g., see Deb [2] and Coello *et al*. [3]). Those performance indices have been used to compare different solution sets (i.e., to compare different EMO algorithms). There is, however, no performance index that can simultaneously measure various aspects of non-dominated solution sets (e.g., their diversity and their convergence to the Pareto-optimal solution set) as pointed out in some studies [9]-[11]. Moreover the use of only a single performance index is sometimes misleading. Thus we use several performance indices in order to evaluate both the diversity of non-dominated solution sets and their convergence to the Pareto-optimal solution set.

Let S and S^* be a non-dominated solution set and the Pareto-optimal solution set, respectively. The convergence of the non-dominated solution set S to the Pareto-optimal solution set S^* is measured by the following performance index called the generational distance [12]:

$$\mathrm{GD}(S) = \frac{1}{|S|} \sum_{\mathbf{x} \in S} \min\{\|\mathbf{f}(\mathbf{x}) - \mathbf{f}(\mathbf{y})\| : \mathbf{y} \in S^*\}, \quad (5)$$

where $\|\mathbf{f}(\mathbf{x}) - \mathbf{f}(\mathbf{y})\|$ is the Euclidean distance between the two solutions $\mathbf{x}$ and $\mathbf{y}$ in the objective space. The generational distance is the average distance from each solution in S to its nearest Pareto-optimal solution in S^* (see Fig. 1).

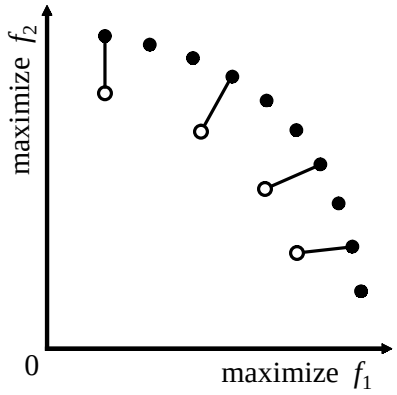

Figure 1. Illustration of the generational distance. Closed circles and open circles correspond to Pareto-optimal solutions in S^* and solutions in S, respectively.

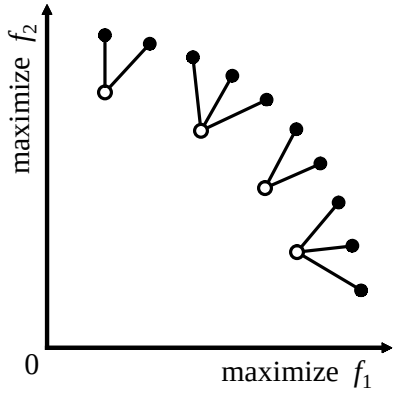

Figure 2. Illustration of the $\mathrm{D1_R}$ measure. Closed circles and open circles correspond to Pareto-optimal solutions in S^* and solutions in S, respectively.

Whereas the generational distance measures the proximity of the non-dominated solution set S to the Pareto-optimal solution set S^*, it does not always mean the quality of S. Let us consider an extreme case where S consists of only a single Pareto-optimal solution. In this case, the generational distance is zero (i.e., its best value). The diversity of this solution set, however, is minimum. That is, this solution set is very good in terms of the convergence but very poor in terms of the diversity.

In order to measure not only the convergence but also the diversity, Czyzak and Jaszkiewicz [13] used the following performance index called the $\mathrm{D1_R}$ measure (see Fig. 2):

$$\mathrm{D1_R}(S) = \frac{1}{|S^*|} \sum_{\mathbf{y} \in S^*} \min\{\|\mathbf{f}(\mathbf{x}) - \mathbf{f}(\mathbf{y})\| : \mathbf{x} \in S\}. \quad (6)$$

As shown in Fig. 2, the $\mathrm{D1_R}$ measure is the average distance from each Pareto-optimal solution $\mathbf{y}$ in S^* to its nearest solution in S.

The diversity of the non-dominated solution set S can be more directly measured by the sum of the range of objective values for each objective function (see Fig. 3):

$$\mathrm{Spread}(S) = \sum_{i=1}^{k} [\max_{\mathbf{x} \in S} \{f_i(\mathbf{x})\} - \min_{\mathbf{x} \in S} \{f_i(\mathbf{x})\}]. \quad (7)$$

This measure is similar to the maximum spread of Zitzler [14].

In order to measure both the diversity and the convergence, we can also use the hypervolume measure [15] that calculates the volume of the dominated region by the non-dominated solution set S in the objective space. The hypervolume measure is illustrated in Fig. 4.

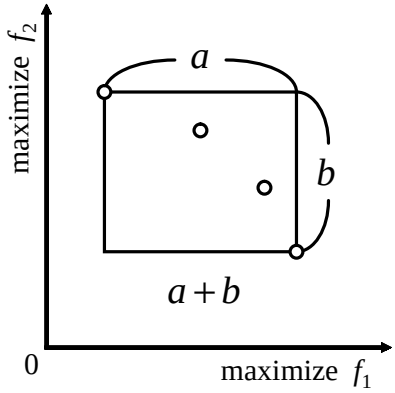

Figure 3. Illustration of the spread measure. Open circles correspond to solutions in the non-dominated solution set S.

As shown in Fig. 4, the boundary of the dominated region in the objective space is called the attainment surface [16]. From multiple attainment surfaces obtained by iterative executions of an EMO algorithm for a multiobjective optimization problem, we can calculate the 50% attainment surface as a kind of their average result. For details of the calculation of the 50% attainment surface, see Fonseca and Fleming [16] and Deb [2].

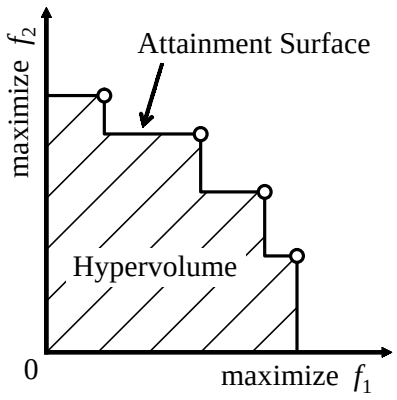

Figure 4. Illustration of the hypervolume measure. Open circles correspond to solutions in the non-dominated solution set *S*.

C. NSGA-II Algorithm

Recently developed EMO algorithms share some common features such as fitness evaluation based on the dominance relation in (4), diversity preserving and elitism. The NSGA-II algorithm of Deb *et al.* [4], which is one of the most well-known and frequently-used EMO algorithms, also has these features. In this sense, it is a typical EMO algorithm. In this paper, we use the NSGA-II algorithm due to its typicality as well as its simplicity, popularity and high search ability.

The basic framework of the NSGA-II algorithm can be written as follows:

[NSGA-II]
Step 1: P = Initialize(P)
Step 2: While the stopping condition is not satisfied, do
Step 3: P' = GeneticOperations(P)
Step 4: P = Replace($P \cup P'$)
Step 5: End while
Step 6: Return(P)

In Step 1, the population P is initialized. The initialization is usually performed randomly. In Step 3, an offspring population P' is generated from the current population P by selection, crossover and mutation. The standard binary tournament selection scheme is usually used to choose a pair of parent solutions. The size of the offspring population P' is usually the same as that of the current population P. In Step 4, the best solutions are chosen from the merged population ($P \cup P'$) to construct the next population P as in the $(\mu + \lambda)$-ES generation update scheme. Elitism is realized in Step 4 by choosing the best solutions from the current and offspring populations.

Each solution in the current population is evaluated in Step 2 in the following manner to select a pair of parent solutions using the dominance relation in (4) and a crowding measure. First, Rank 1 is assigned to all non-dominated solutions in the current population. Solutions with Rank 1 are tentatively removed from the current population. Next, Rank 2 is assigned to all non-dominated solutions in the reduced current population. Solutions with Rank 2 are tentatively removed from the reduced current population. This ranking procedure is iterated until all solutions are tentatively removed from the current population (i.e., until ranks are assigned to all solutions). The ranking procedure is illustrated in Fig. 5 where solutions with higher ranks (i.e., smaller ranks) are viewed as better solutions.

Diversity preserving is realized in the NSGA-II algorithm by calculating a crowding measure for each solution among those with the same rank after the ranking procedure is completed. Roughly speaking, the crowding measure is the sum of distances from adjacent solutions over each objective (see Fig. 6 where $a + b$ is assigned to a solution with Rank 3). More specifically, two adjacent solutions of each solution are found among those with the same rank for each objective. Then the distance between these two adjacent solutions is calculated with respect to the corresponding objective. Finally the calculated distance for each objective is summed up over the k objectives. An infinitely large value is assigned to extreme solutions with the minimum or maximum objective value of at least one objective among solutions with the same rank. The calculation of the crowding measure is illustrated in Fig. 6.

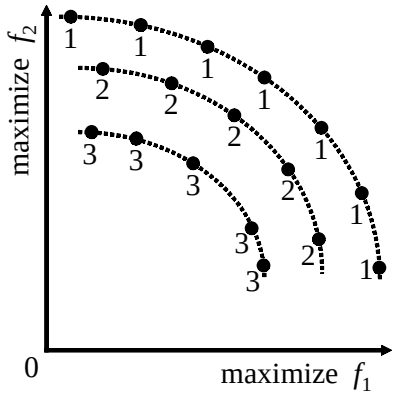

Figure 5. Ranking in the NSGA-II algorithm.

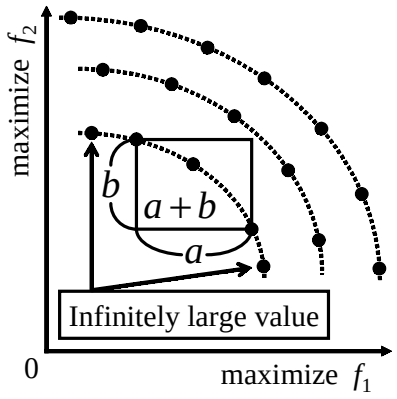

Figure 6. Calculation of the crowding measure in the NSGA-II algorithm.

In the binary tournament selection scheme in Step 3, two solutions are randomly selected from the current population. If the two solutions have different ranks, the better solution with the higher rank (i.e., smaller rank) is chosen as a parent solution. If they have the same rank, the better solution with a larger value of the crowding measure is chosen as a parent solution. By iterating this tournament selection scheme, a pair of parent solutions are selected. An offspring solution is generated from the selected pair of parent solutions by crossover and mutation in our implementation of the NSGA-II algorithm. The selection, crossover and mutation are iterated until a pre-specified number of offspring solutions are generated.

In Step 4 of the NSGA-II algorithm, each solution in the merged population is evaluated by the ranking procedure and the crowding measure in the same manner as in Step 3 for parent selection. A pre-specified number of the best solutions are selected from the merged population based on the rank of each solution and its value of the crowding measure. In the generation update phase in Step 4 as well as the parent selection phase in Step 3, solutions with higher ranks are viewed as better solutions. Among solutions with the same rank, larger values of the crowding measure are viewed as better.

The convergence of solutions to the Pareto front is realized by choosing better solutions with respect to their ranks in the parent selection phase and the generation update phase. The diversity of solutions is maintained by choosing better solutions with respect to the crowding measure among solutions with the same rank. For more details of each step of the NSGA-II algorithm, see Deb [2] and Deb *et al.* [4].

III. Examination of Overlapping Objective Vectors

A. Test Problems

As test problems, we use Min-Ex in Deb [2], ZDT1, ZDT2 and ZDT3 in Zitzler *et al.* [5], multiobjective 0/1 knapsack problems in Zitzler and Thiele [6], multiobjective flowshop scheduling problems in Ishibuchi *et al.* [7], and multiobjective fuzzy rule selection problems in Ishibuchi and Yamamoto [8]. We briefly explain each test problem in this subsection.

Deb [2] used the following two-objective minimization problem called Min-Ex to illustrate characteristics of a number of EMO algorithms:

$$\text{Minimize } f_1(\mathbf{x}) = x_1 \text{ and } f_2(\mathbf{x}) = (1 + x_2)/x_1, \tag{8}$$

$$\text{subject to } 0.1 \le x_1 \le 1 \text{ and } 0 \le x_2 \le 5. \tag{9}$$

We represent each decision variable by a binary string of length 30 using standard binary coding.

Zitzler *et al.* [5] framed six test problems (ZDT1 to ZDT6) to examine the performance of a number of EMO algorithms. We use the first three test problems (ZDT1 to ZDT3), which have 30 continuous decision variables in the unit interval [0, 1]. That is, the decision space of these test problems is the 30-dimensional unit hyper-cube $[0, 1]^{30}$. All the six test problems in [5] can be written in the following form:

$$\text{Minimize } f_1(\mathbf{x}) \text{ and } f_2(\mathbf{x}) = g(\mathbf{x}) \cdot h(f_1(\mathbf{x}), g(\mathbf{x})). \tag{10}$$

In all the first three test problems (ZDT1 to ZDT3), $f_1(\mathbf{x})$ and $g(\mathbf{x})$ are defined as follows:

$$f_1(\mathbf{x}) = x_1 \text{ and } g(\mathbf{x}) = 1 + \frac{9}{n-1}\sum_{i=1}^{n} x_i, \tag{11}$$

where n is the number of decision variables (i.e., 30). The first three test problems in [5] are different from each other in the definition of the function $h(f_1, g)$ as follows:

$$\text{ZDT1: } h(f_1, g) = 1 - \sqrt{f_1/g}, \tag{12}$$

$$\text{ZDT2: } h(f_1, g) = 1 - (f_1/g)^2, \tag{13}$$

$$\text{ZDT3: } h(f_1, g) = 1 - \sqrt{f_1/g} - (f_1/g)\cdot\sin(10\pi f_1). \tag{14}$$

Zitzler and Thiele [6] used nine multiobjective 0/1 knapsack problems, each of which has two, three or four objectives and 250, 500 or 750 items, to evaluate the performance of a number of EMO algorithms. Each test problem with k knapsacks (i.e., k objectives and k constraints) and n items is written as follows:

$$\text{Maximize } \mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), ..., f_k(\mathbf{x})), \tag{15}$$

$$\text{subject to } \sum_{j=1}^{n} w_{ij} x_j \le c_i, \quad i = 1, 2, ..., k, \tag{16}$$

$$\text{where } f_i(\mathbf{x}) = \sum_{j=1}^{n} p_{ij} x_j, \quad i = 1, 2, ..., k. \tag{17}$$

In this formulation, $\mathbf{x}$ is an n-dimensional binary vector, p_{ij} is the profit of item j according to knapsack i, w_{ij} is the weight of item j according to knapsack i, and c_i is the capacity of knapsack i. Each solution $\mathbf{x}$ is handled as a binary string of length n. We denote the k-objective n-item test problem as the k-n knapsack problem. Multiobjective 0/1 knapsack problems have been used frequently to examine the performance of EMO algorithms in the literature (e.g., Jaszkiewicz [17], [18], Knowles & Corne [19], Mumford [20], and Zydallis & Lamont [21]).

When an EMO algorithm is applied to the multiobjective 0/1 knapsack problem in (15)-(17), genetic operations often generate infeasible solutions that do not satisfy the constraint conditions in (16). We use a repair method based on a maximum profit/weight ratio as suggested by Zitzler and Thiele [6]. When an infeasible solution is generated, a feasible solution is created by removing items (i.e., by

changing the corresponding values in the binary string **x** from 1 to 0) in the ascending order of the maximum profit/weight ratio defined as

$$q_j = \max\{p_{ij}/w_{ij} : i = 1,2,...,k\}, \quad j = 1,2,...,n. \tag{18}$$

In Ishibuchi *et al*. [7], [22], multiobjective flowshop scheduling problems were used to examine the performance of some EMO and memetic EMO algorithms. Each solution of a flowshop scheduling problem with *n* jobs is represented by a permutation of the given *n* jobs $\{J_1, J_2, ..., J_n\}$. Two-objective test problems in [7], [22] are written as

$$\text{Minimize} \quad f_1(\mathbf{x}) = \max\{C_i \mid i = 1,2,...,n\}, \tag{19}$$

$$\text{Minimize} \quad f_2(\mathbf{x}) = \max\{\max\{C_i - d_i, 0\} \mid i = 1,2,...,n\}, \tag{20}$$

where C_i and d_i are the completion time and the due-date of the *i*th job J_i, respectively. The first objective is to minimize the makespan (i.e., the maximum completion time) while the second objective is to minimize the maximum tardiness.

Three-objective flowshop scheduling problems in [7], [22] have the following objective in addition to the two objectives in (19) and (20):

$$\text{Minimize} \quad f_3(\mathbf{x}) = \sum_{i=1}^{n} C_i. \tag{21}$$

Each flowshop scheduling problem in [7] has 20 machines and 20, 40, 60 or 80 jobs. We denote the *k*-objective test problem with *n* jobs as the *k*-*n* flowshop scheduling problem.

As test problems, we also use multiobjective fuzzy rule selection problems of Ishibuchi and Yamamoto [8], [23]. They used EMO and memetic EMO algorithms for multiobjective design of fuzzy rule-based classification systems in the following manner.

First a pre-specified number of candidate fuzzy rules were generated from training patterns for each class. For details of fuzzy rule generation, see the textbook on fuzzy data mining by Ishibuchi *et al*. [24]. Then non-dominated rule sets were found as subsets of the generated candidate fuzzy rules with respect to the following three objectives:

$$\text{Maximize} \quad f_1(\mathbf{x}), \text{ minimize} \quad f_2(\mathbf{x}), \text{ and minimize} \quad f_3(\mathbf{x}), \tag{22}$$

where **x** denotes a subset of the candidate fuzzy rules (i.e., **x** is a set of selected rules), $f_1(\mathbf{x})$ is the number of correctly classified training patterns by the selected rules in **x**, $f_2(\mathbf{x})$ is the number of the selected rules, and $f_3(\mathbf{x})$ is the total number of antecedent conditions of the selected rules.

Let *N* be the total number of generated candidate fuzzy rules (i.e., *N*/*M* is the number of generated candidate fuzzy rules for each class where *M* is the number of classes). Then each solution of the three-objective rule selection problem in (22) is represented by a binary string of length *N*.

In addition to the three-objective problem in (22), we also use the following two-objective fuzzy rule selection problem with the first two objectives in (22):

$$\text{Maximize} \quad f_1(\mathbf{x}), \text{ and minimize} \quad f_2(\mathbf{x}). \tag{23}$$

We denote the *k*-objective test problem with *N* candidate fuzzy rules as the *k*-*N* fuzzy rule selection problem.

B. Results on Multiobjective Function Optimization

In this subsection, we report our experimental results on the four multiobjective function optimization problems with continuous decision variables: Min-Ex, ZDT1, ZDT2 and ZDT3. The NSGA-II algorithm was applied to each test problem 50 times using the following parameter values:

Population size: 100,
Crossover probability: 0.8 (One-point crossover),
Mutation probability: 1/*N* (*N* is the string length),
Stopping condition: 3000 generations.

In Fig. 7, we show the average number of different objective vectors at each generation for each test problem. From this figure, we can see that the average number of different objective vectors was always more than 80% of the population size for all the four test problems. This means that each population did not include many overlapping solutions in our computational experiments on the four test problems of multiobjective function optimization with continuous decision variables.

Experimental results in Fig. 7 suggest that the existence of overlapping solutions is not a serious issue in the application of EMO algorithms to multiobjective function optimization with continuous decision variables. This may be the reason why this issue has not been explicitly discussed in many studies on EMO algorithms in the literature.

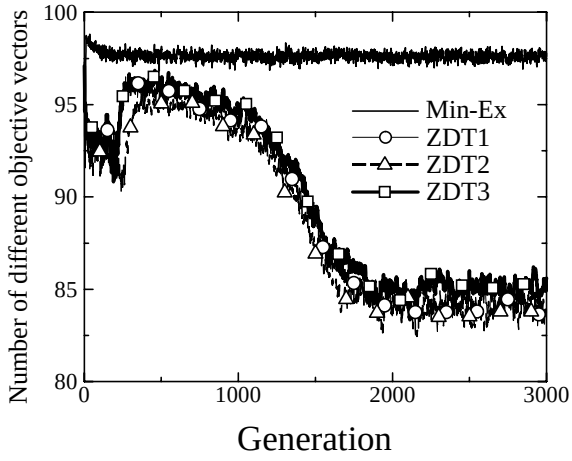

Figure 7. Average number of different objective vectors at each generation for the multiobjective function optimization problems.

C. Results on Multiobjective Knapsack Problems

We applied the NSGA-II algorithm to the three two-objective knapsack problems (i.e., 2-250, 2-500 and 2-750 knapsack problems) 50 times using the same parameter values as the computational experiments in Fig. 7 for the multiobjective function optimization problems. Fig. 8 shows the average number of different objective vectors at each generation for each two-objective knapsack problem.

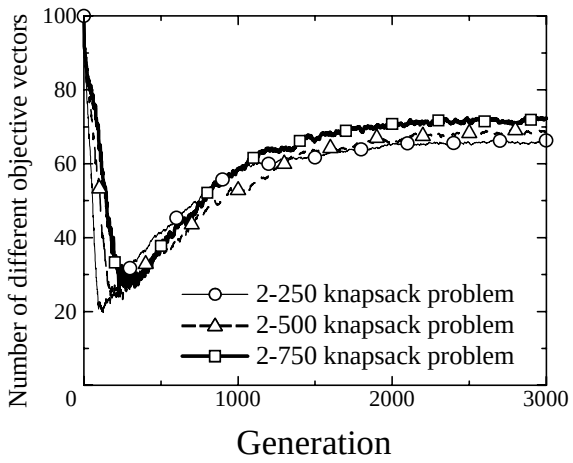

Figure 8. Average number of different objective vectors at each generation for the two-objective knapsack problems.

From Fig. 8, we can see that the number of different objective vectors rapidly decreased to about 20% of the population size during the first 500 generations. That is, each population included a lager number of overlapping objective vectors especially in the early stage of evolution.

We also examined the effect of increasing the number of objectives on the number of overlapping objective vectors through computational experiments on the 500-item knapsack problems with two, three and four objectives (i.e., 2-500, 3-500 and 4-500 knapsack problems). We used the same parameter values as in the previous computational experiments in Fig. 8. Experimental results are summarized in Fig. 9. From Fig. 9, we can see that the increase in the number of objectives leads to the increase in the number of different objective vectors (i.e., the decrease in the number of overlapping objective vectors).

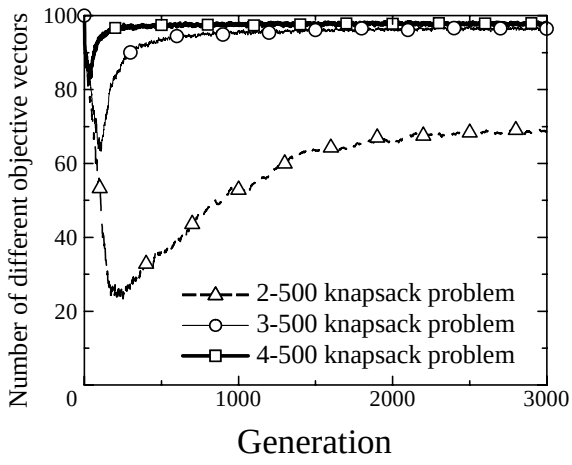

Figure 9. Average number of different objective vectors at each generation for the 500-item knapsack problems with two, three and four objectives.

D. Results on Multiobjective Flowshop Scheduling

We applied the NSGA-II algorithm to the two-objective flowshop scheduling problems with 40 and 80 jobs (i.e., 2-40 and 2-80 flowshop scheduling problems) using the following parameter values:

Population size: 100,
Crossover probability: 0.8 (Two-point order crossover),
Mutation probability: 0.8 per string (Shift mutation),
Stopping condition: 3000 generations.

These parameter values are almost the same as those in the previous computational experiments in Fig. 8 and Fig. 9. For genetic operations such as order crossover and shift mutation, see [7], [22], [25]. Various genetic operations for flowshop scheduling were compared with each other in Murata *et al.* [25].

Experimental results are summarized in Fig. 10. The number of different objective vectors was decreased to about 20% of the population size in the first 1000 generations. This means that many solutions in each population were overlapping with each other in the objective space.

In the same manner as Fig. 10, we also applied the NSGA-II algorithm to the three-objective flowshop scheduling problems with 40 and 80 jobs (i.e., 3-40 and 3-80 flowshop scheduling problems). Experimental results are summarized in Fig. 11. From the comparison between Fig. 10 and Fig. 11, we can see that the increase in the number of objectives leads to the increase in the number of different objective vectors (i.e., the decrease in the number of overlapping objective vectors). The same observation was obtained for the multiobjective knapsack problems in the previous subsection.

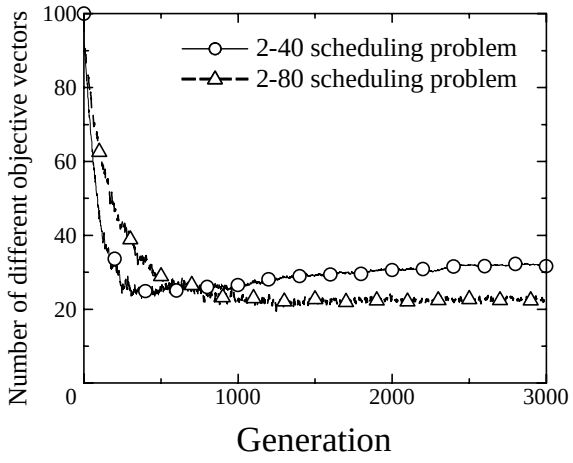

Figure 10. Average number of different objective vectors at each generation for the two-objective flowshop scheduling problems with 40 and 80 jobs.

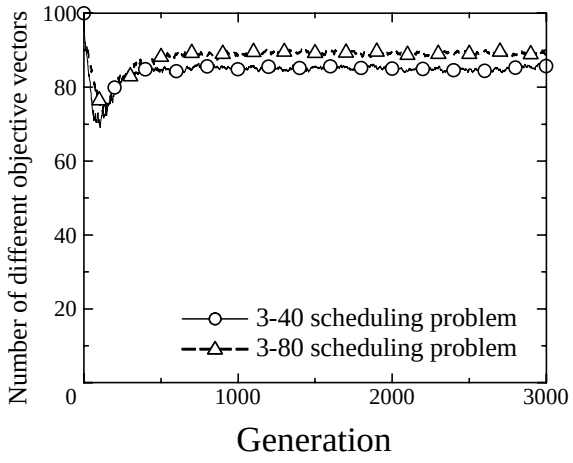

Figure 11. Average number of different objective vectors at each generation for the three-objective flowshop scheduling problems with 40 and 80 jobs.

E. Results on Multiobjective Fuzzy Rule Selection

In computational experiments on fuzzy rule selection problems, we used the wine data set in the UC Irvine Machine Learning Repository. This data set is a three-class pattern classification problem with 13 continuous attributes and 178 training patterns. First we generated a pre-specified number of candidate fuzzy rules for each of the three classes of the wine data set. The number of candidate fuzzy rules for each class was specified as 50 and 100 (i.e., 150 and 300 in total). Then the NSGA-II algorithm was applied to the two-objective fuzzy rule selection problem 50 times for each of the two sets of the generated candidate fuzzy rules. We used the following parameter values:

Population size: 100,
Crossover probability: 0.8 (One-point crossover),
Mutation probability:
0.1 for the mutation form 1 to 0,
1/N for the mutation from 0 to 1 (N is the string length),
Stopping condition: 3000 generations.

These parameter values are almost the same as those in the previous computational experiments on the other test problems. We biased the mutation probability in order to efficiently decrease the number of selected fuzzy rules in each solution (for details, see [8], [23], [24]). We also used a domain-specific heuristic trick to remove unnecessary fuzzy rules from each solution as in [8], [23], [24].

Experimental results are summarized in Fig. 12 where the number of different objective vectors was very small (i.e., about 5% of the population size). This means that almost all solutions in each population were overlapping with each other in the objective space during the execution of the NSGA-II algorithm on the two-objective fuzzy rule selection problems for the wine data set.

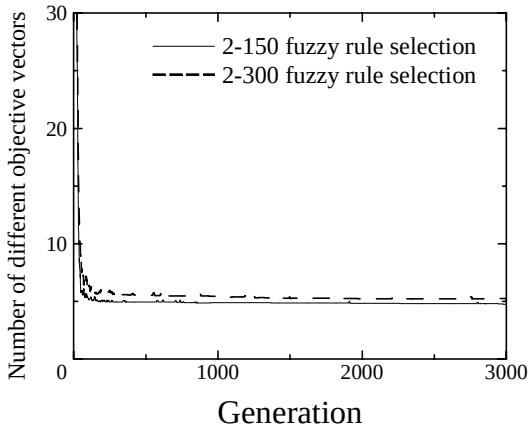

Figure 12. Average number of different objective vectors at each generation for the two-objective fuzzy rule selection problems with 150 and 300 candidate fuzzy rules.

In the same manner as Fig. 12, the NSGA-II algorithm was applied to the three-objective fuzzy rule selection problems with the 150 and 300 candidate fuzzy rules for the wine data set. Experimental results are summarized in Fig. 13. Whereas the number of different objective vectors increased from Fig. 12 (i.e., from about 5% to about 12% of the population size), it was still very small in Fig. 13. That is, almost all solutions were still overlapping with each other in the objective space even in the case of the three objectives.

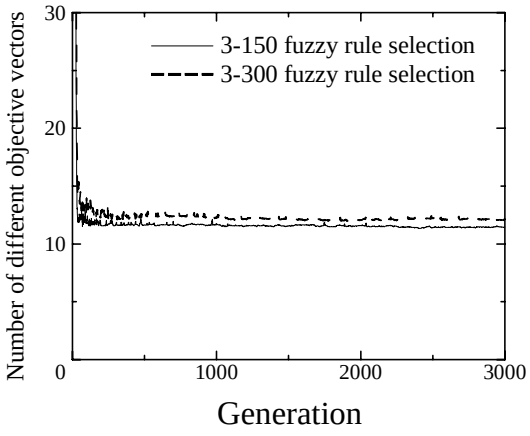

Figure 13. Average number of different objective vectors at each generation for the three-objective fuzzy rule selection problems with 150 and 300 candidate fuzzy rules.

F. Relation between Diversity and Convergence

We have already shown that a large number of overlapping objective vectors were included in each population when the NSGA-II algorithm was applied to some multiobjective combinatorial optimization problems with only a few objectives. We have also shown that each population did not include many overlapping objective vectors in the case of multiobjective function optimization with continuous decision variables. In this subsection, we further examine the number of overlapping objective vectors from the viewpoint of the balance between diversity and convergence in multiobjective evolution.

As shown in Ishibuchi and Narukawa [26], mutation operations generally increase the diversity of solutions while crossover operations improve the convergence of solutions to the Pareto front. In order to examine the effect of crossover and mutation on the number of overlapping solutions, we performed computational experiments on the 2-500 knapsack problems using different parameter specifications of the crossover and mutation probabilities.

We compared the following three specifications of the crossover and mutation probabilities with each other: (0.8, 1/500), (0.8, 4/500) and (0.2, 4/500). The first combination of the crossover and mutation probabilities is the same as the previous computational experiments in Fig. 8. The second combination uses a high mutation probability while the third one uses a low crossover probability together with a high mutation probability.

In the same manner as in Fig. 8, the NSGA-II algorithm was applied to the 2-500 knapsack problem 50 times. In Fig. 14, we show the average number of different objective vectors. From Fig. 14, we can see that the use of a low crossover probability and a high mutation probability increased the number of different objective vectors. This suggests that such a combination of the crossover and mutation probabilities has a positive effect on the diversity of solutions.

In Fig. 15, we show an intermediate population at the 1000th generation in a single run of the NSGA-II algorithm for each of the three parameter specifications. Fig. 15 visually demonstrates the effects of a low crossover probability and a high mutation probability on the diversity of solutions.

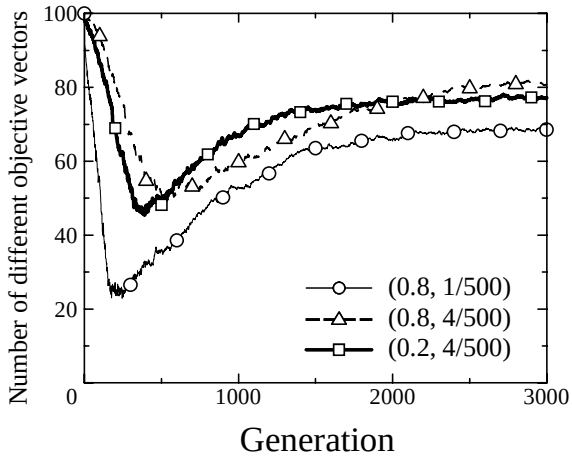

Figure 14. Average number of different objective vectors at each generation for the three parameter specifications on the 2-500 knapsack problem.

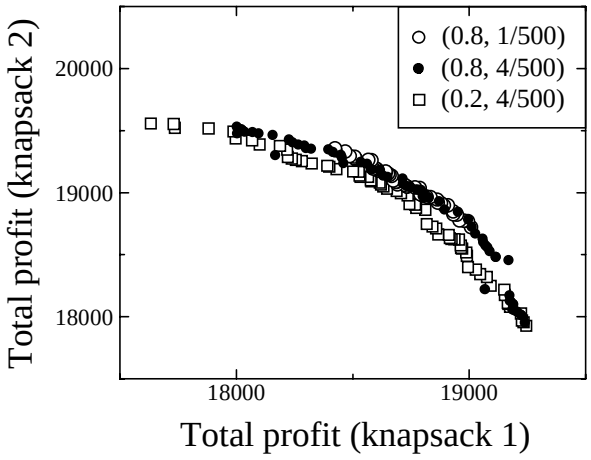

Figure 15. All solutions in the 1000th generation of a single run of the NSGA-II algorithm with each of the three parameter specifications.

In order to examine the relation between the number of different objective vectors and the other performance indices of non-dominated solution sets explained in Section II, we monitored the generational distance, the $D1_R$ measure, the spread measure and the hypervolume measure for the non-dominated solution set at each generation of each run in our computational experiments in Fig. 14. Average results over 50 runs are summarized in Figs. 16-19.

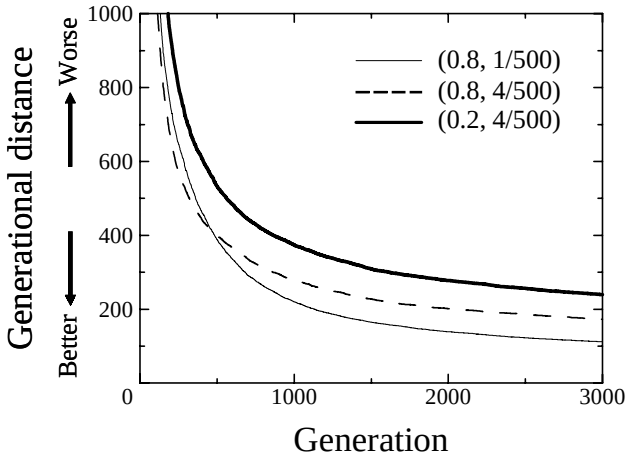

Figure 16. Generational distance at each generation for the three parameter specifications on the 2-500 knapsack problem.

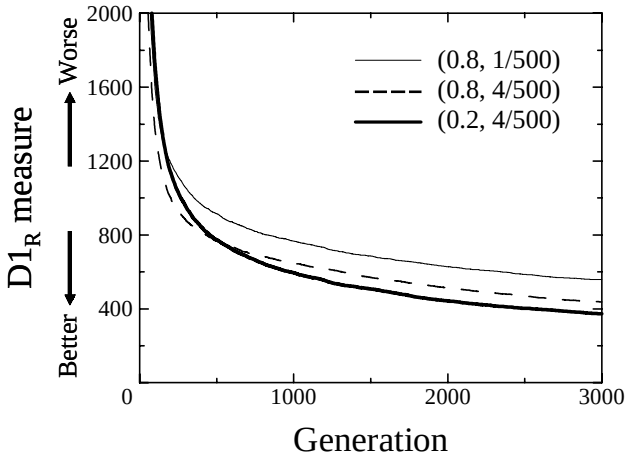

Figure 17. $D1_R$ measure at each generation for the three parameter specifications on the 2-500 knapsack problem.

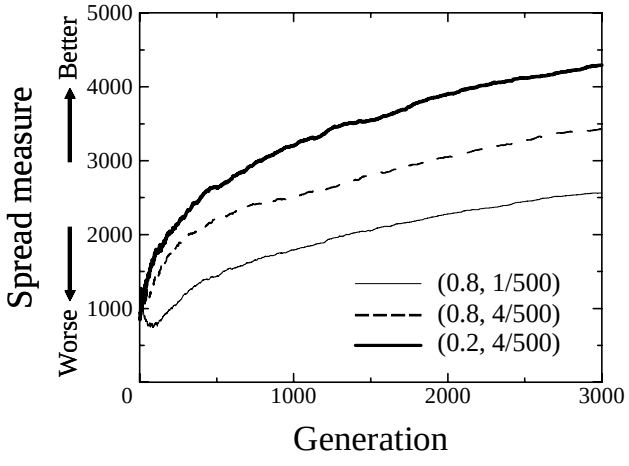

Figure 18. Spread measure at each generation for the three parameter specifications on the 2-500 knapsack problem.

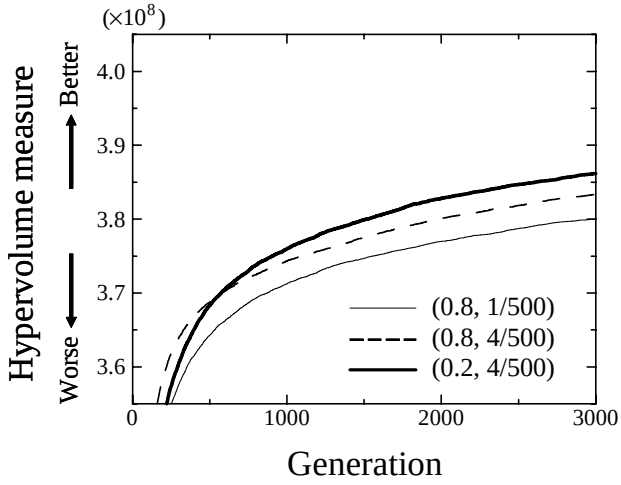

Figure 19. Hypervolume measure at each generation for the three parameter specifications on the 2-500 knapsack problem.

From Figs. 16-19, we can see that the use of a low crossover probability and a high mutation probability (i.e., (0.2, 4/500) in each figure) improved the spread measure and degraded the generational distance. That is, such a parameter specification increased the diversity of solutions but degraded the convergence of solutions to the Pareto front. This is visually shown in Fig. 15 by depicting an intermediate population of a single run for each of the three parameter specifications.

The $D1_R$ measure and the hypervolume measure were also improved by the use of a low crossover probability and a high mutation probability as a result of the increase in the diversity of solutions. This is because these two measures are related to both the diversity and the convergence.

During the initial stage of evolution where the number of different objective vectors rapidly decreased in the case of (0.8, 1/500) in Fig. 14, the convergence was improved significantly in Fig. 16 while the diversity was not improved in Fig. 18. During the last 2000 generations (i.e., from the 1000th generation to the 3000th generation) where the number of different objective vectors gradually increased in Fig. 14, the diversity of solutions was also gradually improved in Fig. 18.

From these observations, we can see that the number of different objective vectors was related to the diversity of solutions in our computational experiments on the 2-500 knapsack problem.

G. Incorporation of Diversity Preserving Mechanism

In the previous subsection, we improved the diversity of solutions by the use of a low crossover probability and a high mutation probability. Here we use a similarity-based mating scheme of Ishibuchi and Shibata [27]-[29]. They proposed an idea of recombining similar solutions [27] and extended it to the recombination of extreme and similar solutions [28], [29].

Their similarity-based mating scheme is shown in Fig. 20 where one parent (say Parent A) is chosen from α candidates and its mate (say Parent B) is chosen from β candidates.

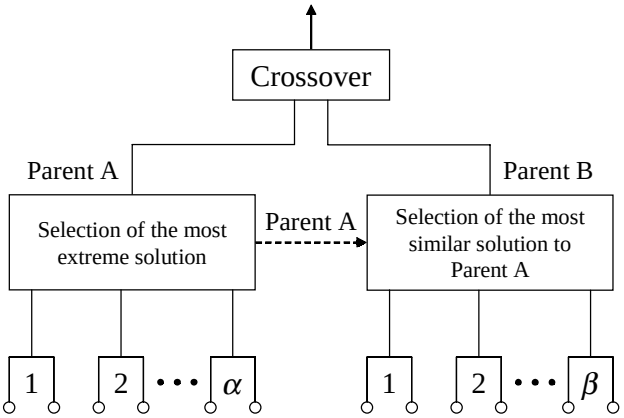

Figure 20. Similarity-based mating scheme [27], [28].

In the similarity-based mating scheme, first α candidates are selected from the current population by iterating the standard binary tournament selection procedure α times. Next the average objective vector of the selected α candidates is calculated in the objective space. The most dissimilar one from the average objective vector among the α candidates is chosen as the first parent (Parent A in Fig. 20). The dissimilarity (and the similarity) is measured as the Euclidean distance in the objective space. Then β candidates are selected by iterating the standard binary tournament procedure β times. Finally the most similar one to Parent A among the β candidates is chosen as its mate (i.e., Parent B in Fig. 20).

In this manner, extreme and similar parents are recombined by the similarity-based mating scheme. The tendency to choose extreme and similar parents can be controlled by the values of the two parameters α and β.

We incorporated the similarity-based mating scheme into the NSGA-II algorithm. That is, the binary tournament selection procedure of the NSGA-II algorithm is iterated to select α and β candidates in the modified NSGA-II algorithm. All the other parts except for the parent selection scheme are the same as the original NSGA-II algorithm.

We applied the modified NSGA-II algorithm to the 2-500 knapsack problem using the three specifications of the two parameters α and β as follows: $(\alpha, \beta) = (1, 1), (5, 5), (10, 10)$. It should be noted that the modified NSGA-II algorithm with (1, 1) is exactly the same as the original NSGA-II algorithm. This is because the binary tournament selection is executed only once to choose a single parent. The larger the values of these two parameters are, the modified NSGA-II algorithm has the stronger tendency to choose extreme and similar parents.

Our computational experiments were performed in the same manner as the previous computational experiments in Fig. 8. That is, the crossover and mutation probabilities were specified as 0.8 and 1/500, respectively. Experimental results are summarized in Figs. 21-25 (i.e., the number of different objective vectors in Fig. 21, the generational distance in Fig. 22, the $D1_R$ measure in Fig. 23, the spread measure in Fig. 24, and the hypervolume measure in Fig. 25).

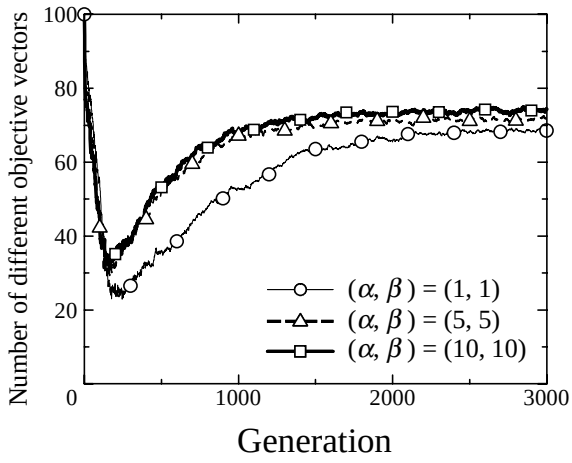

Figure 21. Average number of different objective vectors at each generation of the modified NSGA-II algorithm on the 2-500 knapsack problem.

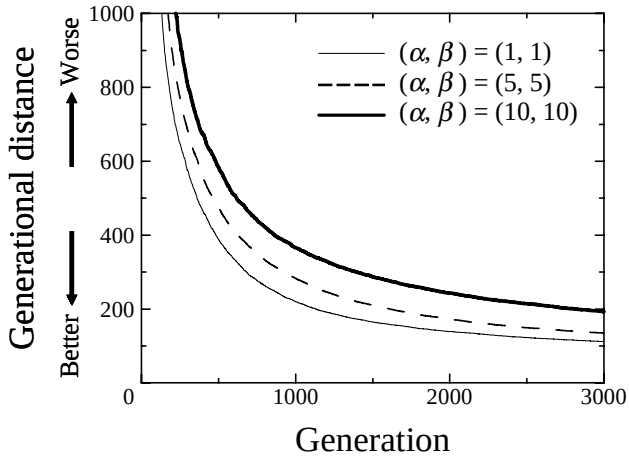

Figure 22. Generational distance at each generation of the modified NSGA-II algorithm on the 2-500 knapsack problem.

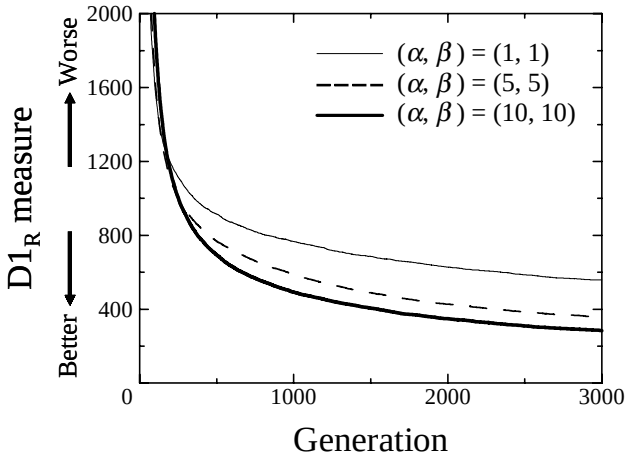

Figure 23. $D1_R$ measure at each generation of the modified NSGA-II algorithm on the 2-500 knapsack problem.

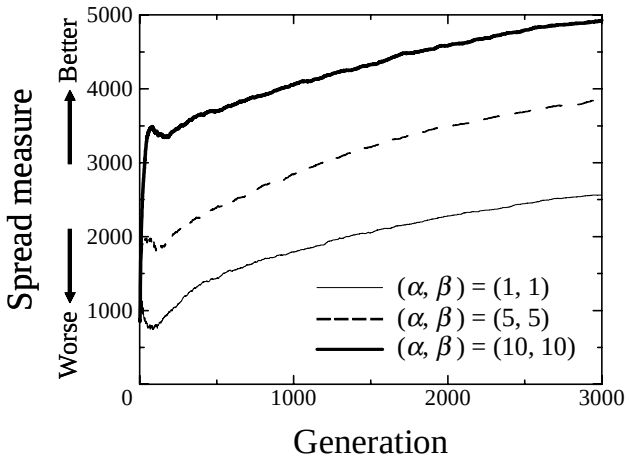

Figure 24. Spread measure at each generation of the modified NSGA-II algorithm on the 2-500 knapsack problem.

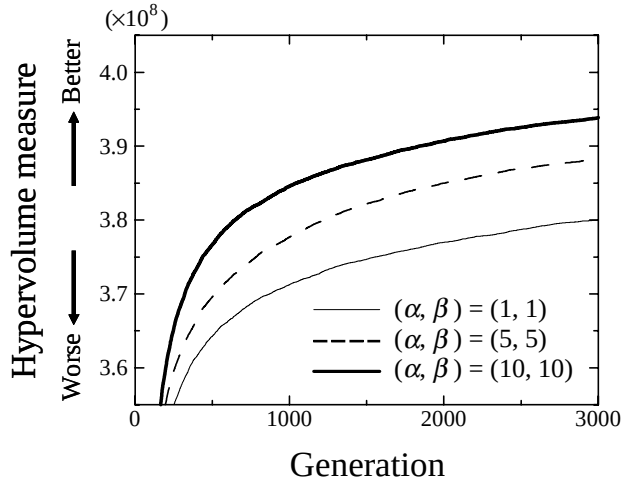

Figure 25. Hypervolume measure at each generation of the modified NSGA-II algorithm on the 2-500 knapsack problem.

From these results, we can see that the similarity-based mating scheme increased the number of different objective vectors (Fig. 21), improved the diversity of solutions (Fig. 24), and degraded the convergence of solutions (Fig. 22). The $D1_R$ measure (Fig. 23) and the hypervolume measure (Fig. 25) were improved by the similarity-based mating scheme as a result of the improvement in the diversity of solutions.

IV. Removal of Overlapping Objective Vectors

In the previous section, we demonstrated that a large number of overlapping objective vectors were included in each population when the NSGA-II algorithm was applied to multiobjective combinatorial optimization problems with only a few objectives. We also demonstrated that the number of overlapping objective vectors was decreased by diversity preserving efforts such as the use a higher mutation probability and the incorporation of the similarity-based mating scheme. In this section, we examine a more direct way for diversity preserving: removal of overlapping objective vectors.

A. Two Strategies for Removing Overlapping Solutions

We remove overlapping objective vectors during the generation update phase so that each solution in the next population has a different location in the objective space. That is, overlapping objective vectors are removed from the merged population before the next population is constructed.

In order to keep the population size constant (i.e., to prevent the next population from becoming smaller than the population size), we first generate an initial population with no overlapping objective vectors. In this case, the number of different objective vectors in the merged population is always larger than or equal to the population size. This means the size of the merged population is always larger than or equal to the population size even after overlapping objective vectors are removed. Thus we can always construct the next population of the pre-specified size with no overlapping objective vectors.

We also examine a similar strategy where overlapping decision vectors (i.e., overlapping solutions in the decision space) are removed from the merged population. Thus each solution in the next population has a different location in the decision space. This means that each solution is different from each other (i.e., no duplicate copies of solutions are included in the next population). When we use this strategy, we first generate an initial population with no duplicate copies.

B. Effects of Removing Overlapping Solutions

Through computational experiments on the 2-500 knapsack problem, we examined the effect of the two strategies on the performance of the NSGA-II algorithm. As in the previous section, we applied the NSGA-II algorithm to the 2-500 knapsack problem 50 times using the following parameter values:

Population size: 100,
Crossover probability: 0.8 (One-point crossover),
Mutation probability: 1/*N* (*N* is the string length),
Stopping condition: 3000 generations.

We also applied two variants of the NSGA-II algorithm to the 2-500 knapsack problem 50 times. One variant has the removal strategy of overlapping objective vectors and the other variant has the removal strategy of overlapping decision vectors. The average number of different objective vectors at each generation is shown in Fig. 26 for the second variant (i.e., the removal of overlapping decision vectors). Since overlapping objective vectors are removed in the first variant, the number of different objective vectors is always the same as the population size. In the second variant where overlapping decision vectors are removed in the decision space, each population can include overlapping objective vectors because different solutions in the decision space are not necessarily different in the objective space. Even so, the number of different objective vectors is almost always the same as the population size in Fig. 26. That is, the diversity of solutions in each population was significantly improved by the two removal strategies in terms of the number of different objective vectors.

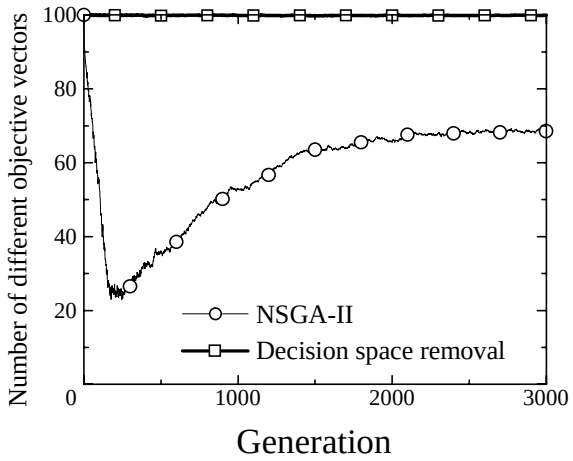

Figure 26. Average number of different objective vectors at each generation of the NSGA-II algorithm and its two variants on the 2-500 knapsack problem.

The NSGA-II algorithm and its two variants are compared with each other in terms of the four performance indices in Figs. 27-30. Whereas the two removal strategies clearly improved the diversity of solutions in each population in terms of the number of different objective vectors in Fig. 26, their effects on the four performance indices in Fig. 27-30 were not clear. They slightly improved the diversity of solutions (compare Fig. 29 with Fig. 24 for evaluating the effects of the removal strategies on the diversity of solutions). It is also interesting to note that the removal strategies did not degrade the convergence of solutions to the Pareto front (see Fig. 27).

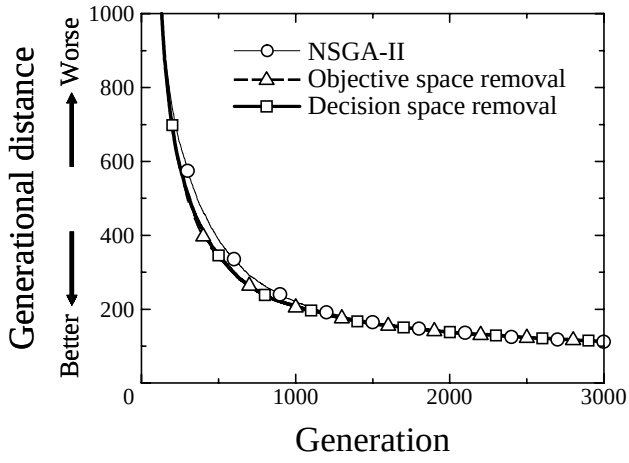

Figure 27. Generational distance at each generation of the NSGA-II algorithm and its two variants on the 2-500 knapsack problem.

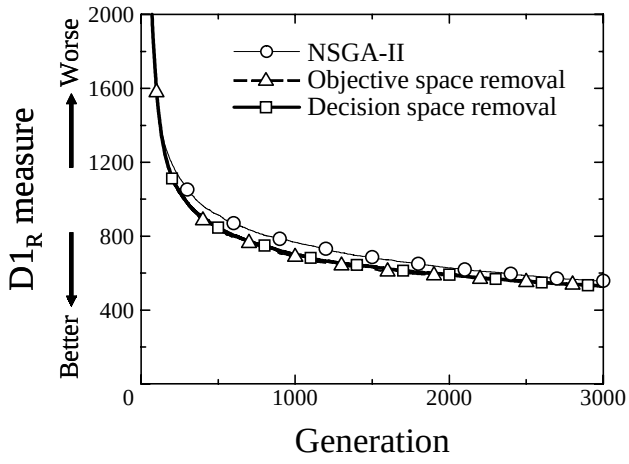

Figure 28. $D1_R$ measure at each generation of the NSGA-II algorithm and its two variants on the 2-500 knapsack problem.

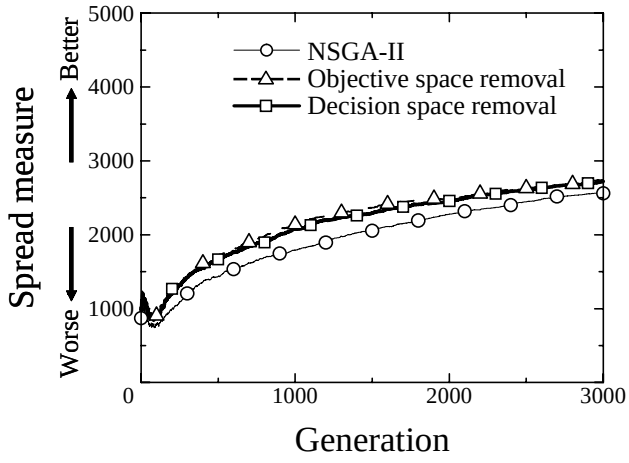

Figure 29. Spread measure at each generation of the NSGA-II algorithm and its two variants on the 2-500 knapsack problem.

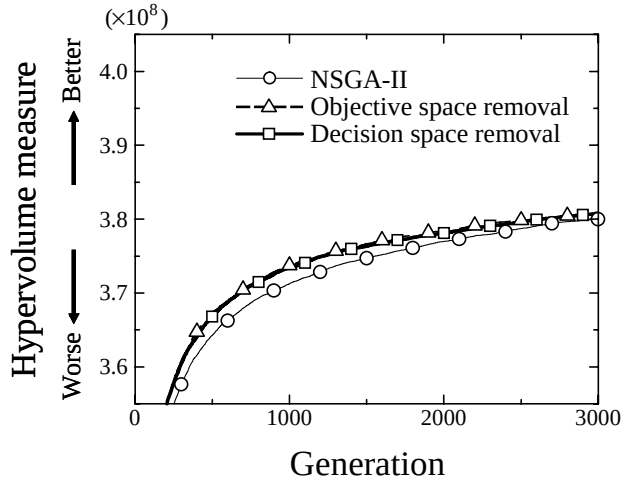

Figure 30. Hypervolume measure at each generation of the NSGA-II algorithm and its two variants on the 2-500 knapsack problem.

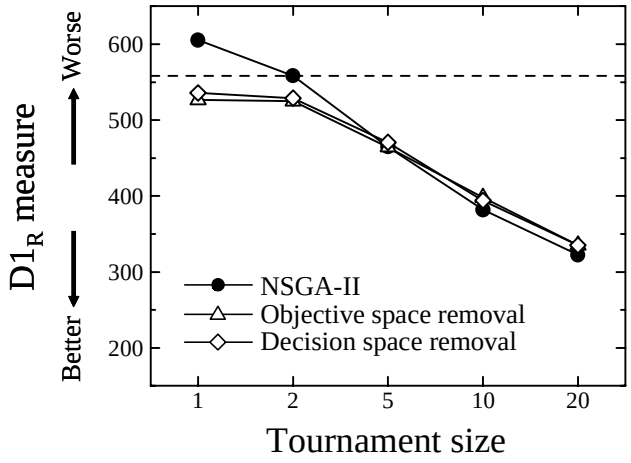

Figure 32. $D1_R$ measure at the 3000th generation of the NSGA-II algorithm and its two variants on the 2-500 knapsack problem.

C. Use of Higher Selection Pressure

Whereas the two removal strategies clearly increased the diversity of solutions in each population in terms of the number of different objective vectors, their effects on the four performance indices were not clear in our computational experiments in the previous subsection. In this subsection, we examine their effects under higher selection pressure.

We performed computational experiments on the 2-500 knapsack problem using the NSGA-II algorithm and its two variants under various specifications of the tournament size for parent selection. We examined five specifications of the tournament size: 1, 2, 5, 10, 20. Except for the tournament size, we used the same parameter values in the previous subsection. Average results at the 3000th generation over 50 runs are summarized in Figs. 31-34. In each figure, the dotted line shows the performance of the original NSGA-II algorithm with the standard parameter values.

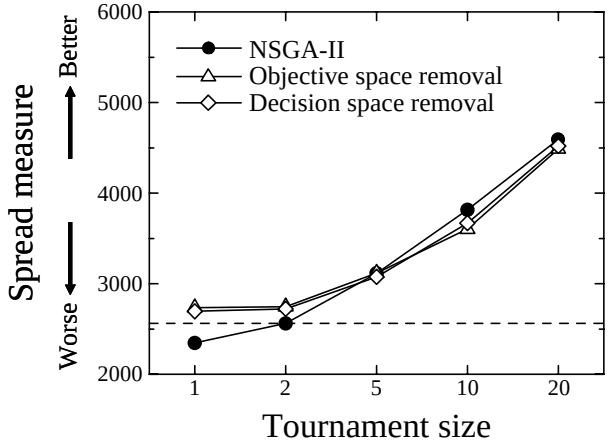

Figure 33. Spread measure at the 3000th generation of the NSGA-II algorithm and its two variants on the 2-500 knapsack problem.

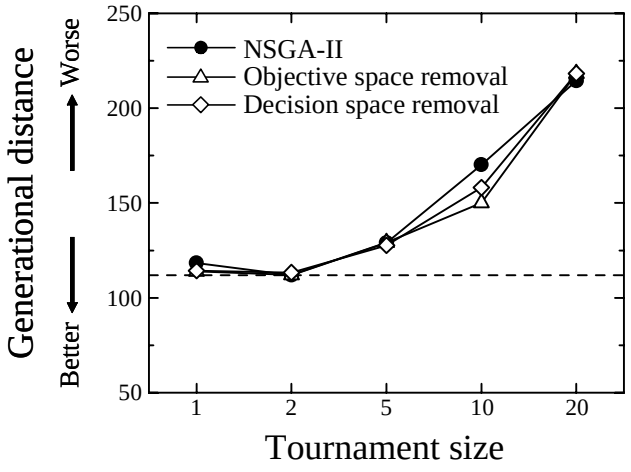

Figure 31. Generational distance at the 3000th generation of the NSGA-II algorithm and its two variants on the 2-500 knapsack problem.

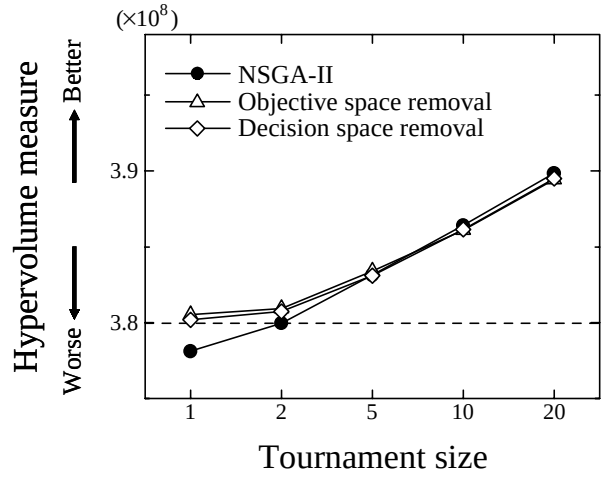

Figure 34. Hypervolume measure at the 3000th generation of the NSGA-II algorithm and its two variants on the 2-500 knapsack problem.

From Figs. 31-34, we can see that the increase in the tournament size improved the three performance indices in Figs. 32-34 related to the diversity of solutions whereas it degraded the generational distance in Fig. 31. We can also see from Figs. 31-34 that the two removal strategies did not have a large effect on any performance indices.

Finally we examined the use of the following weighted sum in the parent selection phase of the NSGA-II algorithm:

$$fitness(\mathbf{x}) = w_1 f_1(\mathbf{x}) + w_2 f_2(\mathbf{x}) + \cdots + w_k f_k(\mathbf{x}), \quad (24)$$

where $\mathbf{w} = (w_1, w_2, ..., w_k)$ is a weight vector:

$$w_1 + w_2 + \cdots + w_k = 1 \quad \text{and} \quad w_i \geq 0 \quad \text{for } i = 1, 2, ..., k. \quad (25)$$

The weight vector **w** is randomly specified whenever a pair of parent solutions are to be selected. That is, two parent solutions are selected based on the common weight vector. The next pair of parent solutions are selected using a different weight vector, which is randomly specified after a previous pair of parent solutions are selected. The weighted sum fitness function with a random weight vector was used in multiobjective genetic local search algorithms (i.e., multiobjective memetic algorithms) in Ishibuchi and Murata [30], Jaszkiewicz [31], [32] and Ishibuchi *et al*. [33]. It should be noted that the ranking and the crowding measure of the NSGA-II algorithm are still used in the generation update phase. The weighted sum fitness function is used only in the parent selection phase.

In the same manner as the above-mentioned computational experiments, we applied the modified NSGA-II algorithm and its two variants with the weighted sum fitness function to the 2-500 knapsack problem 50 times. Experimental results are summarized in Figs. 35-38 in the same manner as Fig. 31-34. The dotted line in each figure shows the performance of the original NSGA-II algorithm.

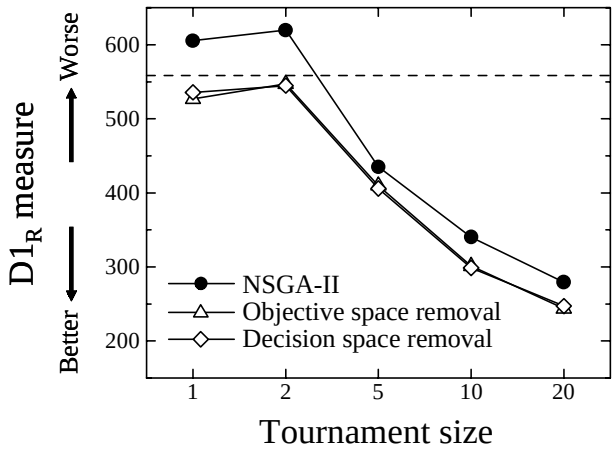

Figure 36. $D1_R$ measure at the 3000th generation of the NSGA-II algorithm and its two variants with the weighted sum fitness function.

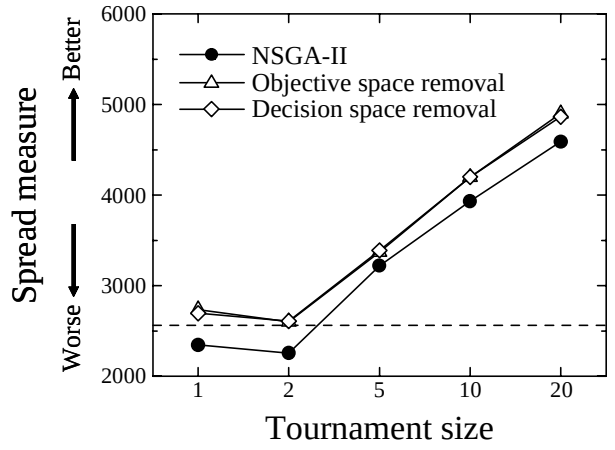

Figure 37. Spread measure at the 3000th generation of the NSGA-II algorithm and its two variants with the weighted sum fitness function.

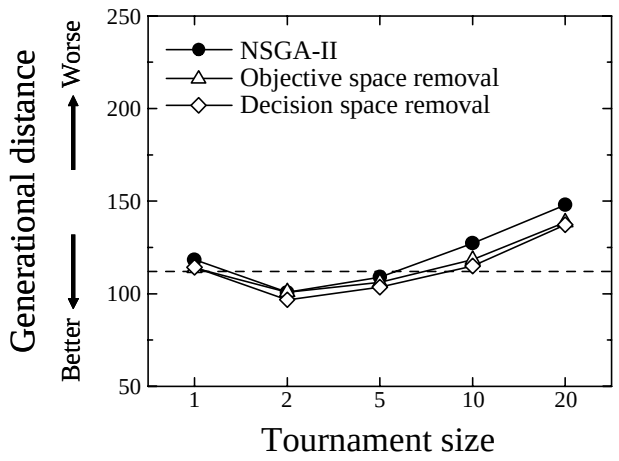

Figure 35. Generational distance at the 3000th generation of the NSGA-II algorithm and its two variants with the weighted sum fitness function.

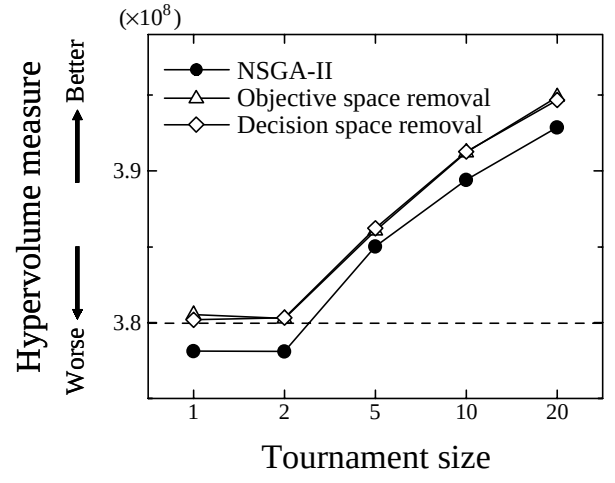

Figure 38. Hypervolume measure at the 3000th generation of the NSGA-II algorithm and its two variants with the weighted sum fitness function.

It is clear in Figs. 35-38 that the removal strategies had a positive effect on all the four performance indices. It is also clear that the two variants of the modified NSGA-II algorithm with the weighted sum fitness function outperformed the original NSGA-II algorithm (the dotted line in each figure) in many cases.

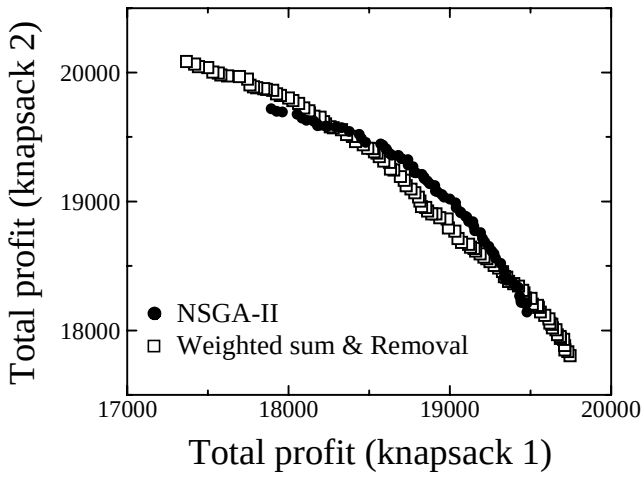

Figure 39. A final solution set at the 3000th generation of each algorithms.

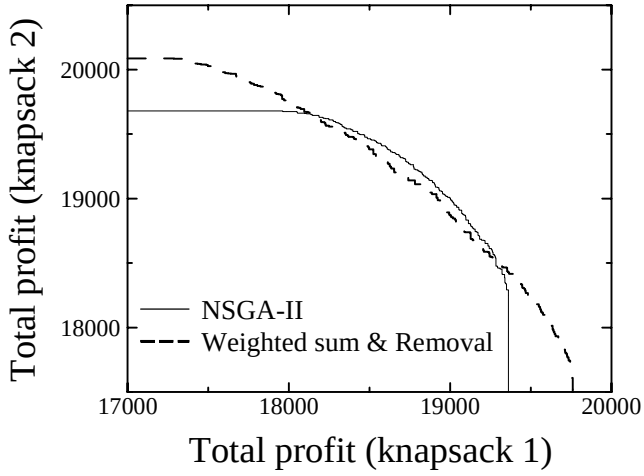

Figure 40. The 50% attainment surface at the 3000th generation over 50 runs of each algorithm.

In order to visually demonstrate the effect of the use of the weighted sum fitness function and the removal strategies on the performance of the NSGA-II algorithm, the following two algorithms are compared with each other in Fig. 39 and Fig. 40. One is the original NSGA-II algorithm with the standard parameter specification (NSGA-II in Fig. 39 and Fig. 40), and the other is the modified NSGA-II algorithm with the weighted sum fitness function and the removal strategy in the objective space (Weighted sum & Removal in Fig. 39 and Fig. 40). Whereas the binary tournament selection scheme was used in the original NSGA-II algorithm, the tournament size was specified as 20 in its variant in Fig. 39 and Fig. 40. Fig. 39 shows a final solution set obtained by a single run of each algorithm. On the other hand, Fig. 40 shows the 50% attainment surface over 50 runs of each algorithm. From these figures, we can see that the use of the weighted sum fitness function and the removal strategy significantly increased the diversity of solutions while slightly degrading the convergence of solutions to the Pareto front.

V. Concluding Remarks

In this paper, we first clearly demonstrated that each population included a large number of overlapping objective vectors when the NSGA-II algorithm of Deb *et al*. [4] was applied to multiobjective combinatorial optimization problems with only a few objectives. The number of overlapping objective vectors was small when it was applied to multiobjective function optimization problems with continuous decision variables.

Next we demonstrated that the number of overlapping objective vectors was decreased by the use of a lower crossover probability and a higher mutation probability. We also showed that the similarity-based mating scheme [28], [29] had a similar effect on the number of overlapping objective vectors. Such diversity preserving efforts improved the diversity of solutions and degraded the convergence of solutions to the Pareto front.

Then we examined two removal strategies: removal of overlapping solutions in the objective space and the decision space. We did not observe any clear performance improvement of the NSGA-II algorithm by the removal strategies with respect to the four performance indices. All the four performance indices were, however, clearly improved by the use of the weighted sum fitness function with high selection pressure (i.e., with large tournament size) in the NSGA-II algorithm together with the removal strategies. This may suggest that a good diversity-convergence balance was realized by the use of the weighted sum fitness function with high selection pressure and the removal strategies.

References

[1] J. D. Schaffer, “Multiple objective optimization with vector evaluated genetic algorithms,” *Proc. of 1st International Conference on Genetic Algorithms and Their Applications* (Pittsburgh, USA, July 24-26, 1985) pp. 93-100.

[2] K. Deb, *Multi-Objective Optimization Using Evolutionary Algorithms*, John Wiley & Sons, Chichester, 2001.

[3] C. A. Coello Coello, D. A. van Veldhuizen, and G. B. Lamont, *Evolutionary Algorithms for Solving Multi-Objective Problems*, Kluwer, Boston, 2002.

[4] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. on Evolutionary Computation*, vol. 6, no. 2, pp. 182-197, April 2002.

[5] E. Zitzler, K. Deb, and L. Thiele, "Comparison of multiobjective evolutionary algorithms: Empirical results," *Evolutionary Computation*, vol. 8, no. 2, pp. 125-148, Summer 2000.

[6] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach," *IEEE Trans. on Evolutionary Computation*, vol. 3, no. 4, pp. 257-271, November 1999.

[7] H. Ishibuchi, T. Yoshida, and T. Murata, "Balance between genetic search and local search in memetic algorithms for multiobjective permutation flowshop scheduling," *IEEE Trans. on Evolutionary Computation*, vol. 7, no. 2, pp. 204-223, April 2003.

[8] H. Ishibuchi and T. Yamamoto, "Fuzzy rule selection by multi-objective genetic local search algorithms and rule evaluation measures in data mining," *Fuzzy Sets and Systems*, vol. 141, no. 1, pp. 59-88, January 2004.

[9] J. D. Knowles and D. W. Corne, "On metrics for comparing non-dominated sets," *Proc. of 2002 Congress on Evolutionary Computation* (Honolulu, USA, May 12-17, 2002) pp. 711-716.

[10] T. Okabe, Y. Jin, and B. Sendhoff, "A critical survey of performance indices for multi-objective optimization," *Proc. of 2003 Congress on Evolutionary Computation* (Canberra, Australia, December 8-12) pp. 878-885.

[11] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. G. da Fonseca, "Performance assessment of multiobjective optimizers: An analysis and review," *IEEE Trans. on Evolutionary Computation*, vol. 7, no, 2, pp. 117-132, April 2003.

[12] D. A. Van Veldhuizen, *Multiobjective Evolutionary Algorithms: Classifications, Analyses, and New Innovations*, Ph. D dissertation, Air Force Institute of Technology, Dayton, 1999.

[13] P. Czyzak and A. Jaszkiewicz, "Pareto-simulated annealing – A metaheuristic technique for multi-objective combinatorial optimization," *Journal of Multi-Criteria Decision Analysis*, vol. 7, no. 1, pp. 34-47, January 1998.

[14] E. Zitzler, *Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications*, Ph. D dissertation, Swiss Federal Institute of Technology, Zurich, Shaker Verlag, Aachen, 1999.

[15] E. Zitzler and L. Thiele, "Multiobjective optimization using evolutionary algorithms – A comparative case study," *Proc. of 5th International Conference on Parallel Problem Solving from Nature* (Amsterdam, September 27-30, 1998) pp. 292-301.

[16] C. M. Fonseca and P. J. Fleming, "On the performance assessment and comparison of stochastic multiobjective optimizers," *International Conference on Parallel Problem Solving from Nature* (Berlin, September 22-26, 1996) pp. 584-593.

[17] A. Jaszkiewicz, "Comparison of local search-based metaheuristics on the multiple objective knapsack problem," *Foundations of Computing and Decision Sciences*, vol. 26, no. 1, pp. 99-120, 2001.

[18] A. Jaszkiewicz, "On the performance of multiple-objective genetic local search on the 0/1 knapsack problem - A comparative experiment," *IEEE Trans. on Evolutionary Computation*, vol. 6, no. 4, pp. 402-412, August 2002.

[19] J. D. Knowles and D. W. Corne, "A comparison of diverse approaches to memetic multiobjective combinatorial optimization," *Proc. of 2000 Genetic and Evolutionary Computation Conference Workshop Program: WOMA I* (Las Vegas, USA, July 8-12, 2000) pp. 103-108.

[20] C. L. Mumford, "Comparing representations and recombination operators for the multi-objective 0/1 knapsack problem," *Proc. of 2003 Congress on Evolutionary Computation* (Canberra, Australia, December 8-12, 2003) pp. 854-861.

[21] J. B. Zydallis and G. B. Lamont, "Explicit building-block multiobjective evolutionary algorithms for NPC problems," *Proc. of 2003 Congress on Evolutionary Computation* (Canberra, Australia, December 8-12, 2003) pp. 2685-2695.

[22] H. Ishibuchi and T. Murata, "A multi-objective genetic local search algorithm and its application to flowshop scheduling," *IEEE Trans. on Systems, Man, and Cybernetics - Part C: Applications and Reviews*, vol. 28, no. 3, pp. 392-403, August 1998.

[23] H. Ishibuchi and T. Yamamoto, "Effects of three-objective genetic rule selection on the generalization ability of fuzzy rule-based systems," *Lecture Notes in Computer Science 2632* (*Proc. of 2nd International Conference on Evolutionary Multi-Criterion Optimization*, Faro, Portugal, April 8-11, 2003) pp. 608-622, Springer, Berlin, April 2003.

[24] H. Ishibuchi, T. Nakashima, and M. Nii, *Classification and Modeling with Linguistic Information Granules: Advanced Approaches to Linguistic Data Mining*, Springer, Berlin, 2004.

[25] T. Murata, H. Ishibuchi, and H. Tanaka, "Genetic algorithms for flowshop scheduling problems," *Computer and Industrial Engineering*, vol. 30, no. 4, pp. 1061-1071, October 1996.

[26] H. Ishibuchi and K. Narukawa, "Recombination of similar parents in EMO algorithms," *Lecture Notes in Computer Science 3410* (*Proc. of 3rd International Conference on Evolutionary Multi-Criterion Optimization*, Guanajuato, Mexico, March 9-11, 2005) pp. 265-279, Springer, Berlin, March 2005.

[27] H. Ishibuchi and Y. Shibata, "An empirical study on the effect of mating restriction on the search ability of EMO algorithms," *Lecture Notes in Computer Science 2632* (*Proc. of 2nd International Conference on Evolutionary*

Multi-Criterion Optimization, Faro, Portugal, April 8-11, 2003) pp. 433-447, Springer, Berlin, April 2003.

[28] H. Ishibuchi and Y. Shibata, "A similarity-based mating scheme for evolutionary multiobjective optimization," *Lecture Notes in Computer Science 2723* (*Proc. of 2003 Genetic and Evolutionary Computation Conference*, Chicago, USA, July 12-16, 2003) pp. 1065-1076, Springer, Berlin, July 2003.

[29] H. Ishibuchi and Y. Shibata, "Mating scheme for controlling the diversity-convergence balance for multiobjective optimization," *Lecture Notes in Computer Science 3102* (*Proc. of 2004 Genetic and Evolutionary Computation Conference*, Seattle, USA, June 26-30, 2004) pp. 1259-1271, Springer, Berlin, June 2004.

[30] H. Ishibuchi and T. Murata, "A multi-objective genetic local search algorithm and its application to flowshop scheduling," *IEEE Trans. on Systems, Man, and Cybernetics - Part C: Applications and Reviews*, vol. 28, no. 3, pp. 392-403, August 1998.

[31] A. Jaszkiewicz, "Genetic local search for multi-objective combinatorial optimization," *European Journal of Operational Research*, vol. 137, no. 1, pp. 50-71, February 2002.

[32] A. Jaszkiewicz, "On the performance of multiple-objective genetic local search on the 0/1 knapsack problem - A comparative experiment," *IEEE Trans. on Evolutionary Computation*, vol. 6, no. 4, pp. 402-412, August 2002.

[33] H. Ishibuchi, T. Yoshida, and T. Murata, "Balance between Genetic Search and Local Search in Memetic Algorithms for Multiobjective Permutation Flowshop Scheduling," *IEEE Trans. on Evolutionary Computation*, vol. 7, no. 2, pp. 204-223, April 2003.

Author Biographies

Hisao Ishibuchi was born in 1963 in Kumamoto, Japan. He received the B.S. and M.S. Degrees in precision mechanics from Kyoto University, Kyoto, Japan, in 1985 and 1987, respectively, and the Ph.D. degree from Osaka Prefecture University, Osaka, Japan, in 1992.

Since 1987, he has been with Osaka Prefecture University, where he is currently a Professor at the Department of Computer Science and Intelligent Systems. He was a Visiting Research Associate at the University of Toronto, Toronto, ON, Canada, from August 1994 to March 1995 and from July 1997 to March 1998. His research interests include fuzzy rule-based classification, evolutionary multiobjective optimization, evolutionary game, and data mining.

Dr. Ishibuchi received the best paper award at GECCO 2004. He is currently an Associate Editor for *International Journal of Computational Intelligence Research*, *IEEE Trans. on Fuzzy Systems*, *IEEE Trans. on Systems, Man, and Cybernetics - Part B*, and *Mathware & Soft Computing*. He is also a fuzzy technical committee member of the IEEE Computational Intelligence Society.

Kaname Narukawa was born in 1980 in Osaka, Japan. He received the B.S. and M.S. Degrees in industrial engineering from Osaka Prefecture University, Osaka, Japan, in 2004 and 2005, respectively.

Currently he is a Ph. D. Student at the Department of Computer Science and Intelligent Systems, Osaka Prefecture University. His research interests include evolutionary multiobjective optimization, hybrid multiobjective optimization and memetic algorithms.

Yusuke Nojima was born in 1976 in Toyama, Japan. He received the B.S. and M.S. Degrees in mechanical engineering from Osaka Institute of Technology, Osaka, Japan, in 1999 and 2001, respectively, and the Ph.D. degree in computer science from Kobe University, Hyogo, Japan, in 2004.

Since 2004, he has been with Osaka Prefecture University, where he is currently a Research Associate at the Department of Computer Science and Intelligent Systems. His research interests include fuzzy rule-based classification, evolutionary multiobjective optimization, and intelligent robotic systems.