H. Ishibuchi and Y. Nojima, “Optimization of scalarizing functions through evolutionary multiobjective optimization,” *Lecture Notes in Computer Science 4403: Evolutionary Multi-Criterion Optimization - EMO 2007*, pp. 51-65, Springer, Berlin, 2007.

Optimization of Scalarizing Functions through Evolutionary Multiobjective Optimization

Hisao Ishibuchi and Yusuke Nojima

Department of Computer Science and Intelligent Systems, Graduate School of Engineering, Osaka Prefecture University, 1-1 Gakuen-cho, Naka-ku, Sakai, Osaka 599-8531, Japan
hisaoi@cs.osakafu-u.ac.jp, nojima@cs.osakafu-u.ac.jp
http://www.ie.osakafu-u.ac.jp/~hisaoi/ci_lab_e

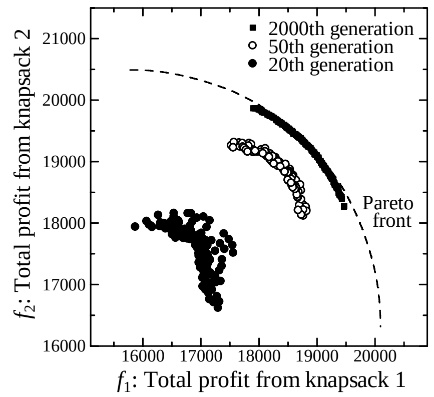
Abstract. This paper proposes an idea of using evolutionary multiobjective optimization (EMO) to optimize scalarizing functions. We assume that a scalarizing function to be optimized has already been generated from an original multiobjective problem. Our task is to optimize the given scalarizing function. In order to efficiently search for its optimal solution without getting stuck in local optima, we generate a new multiobjective problem to which an EMO algorithm is applied. The point is to specify multiple objectives, which are similar to but different from the scalarizing function, so that the location of the optimal solution is near the center of the Pareto front of the generated multiobjective problem. The use of EMO algorithms helps escape from local optima. It also helps find a number of alternative solutions around the optimal solution. Difficulties of Pareto ranking-based EMO algorithms in the handling of many objectives are avoided by the use of similar objectives. In this paper, we first demonstrate that the performance of EMO algorithms as single-objective optimizers of scalarizing functions highly depends on the choice of multiple objectives. Based on this observation, we propose a specification method of multiple objectives for the optimization of a weighted sum fitness function. Experimental results show that our approach works very well in the search for not only a single optimal solution but also a number of good alternative solutions around the optimal solution. Next we evaluate the performance of our approach in comparison with a hybrid EMO algorithm where a single-objective fitness evaluation scheme is probabilistically used in an EMO algorithm. Then we show that our approach can be also used to optimize other scalarizing functions (e.g., those based on constraint conditions and reference solutions). Finally we show that our approach is applicable not only to scalarizing functions but also other single-objective optimization problems.

1 Introduction

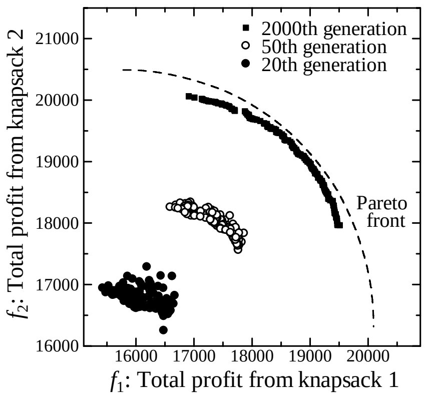
Evolutionary multiobjective optimization (EMO) is one of the most active research areas in the field of evolutionary computation. EMO algorithms have been successfully applied to various application areas involving multiple objectives [2]. In some cases, EMO algorithms can outperform single-objective evolutionary algorithms even when they are used to solve single-objective problems. It was reported in some studies on multiobjectivization [15], [18] that better results were obtained by transforming single-objective problems into multiobjective ones (see [15] for multiobjectivization).

Motivated by these studies on multiobjectivization, we examined the use of EMO algorithms to optimize the sum of multiple objectives in our former studies [8], [10]. We obtained promising results when we used NSGA-II [3] to optimize the simple sum fitness function for a two-objective 500-item (i.e., 2-500) knapsack problem of Zitzler & Thiele [19]. That is, NSGA-II outperformed its single-objective version in finding the optimal solution of the sum of the two objectives. This is because the use of NSGA-II helps escape from local optima.

Usually EMO algorithms are very good at finding Pareto-optimal or near Pareto-optimal solutions around the center of the Pareto front of a two-objective problem. EMO algorithms, however, are not always good at finding good solutions near the edge of the Pareto front. This is illustrated in Fig. 1 where NSGA-II was applied to the 2-500 knapsack problem [19] using two different settings. In Fig. 1 (a), standard parameter values were used (i.e., 0.8 crossover probability and 1/500 mutation probability). In this case, we observe a good convergence of solutions to the Pareto front. Actually NSGA-II outperformed its single-objective version in finding the optimal solution of the simple sum fitness function: $fitness(\mathbf{x}) = f_1(\mathbf{x}) + f_2(\mathbf{x})$. On the other hand, lower crossover and higher mutation probabilities were used in Fig. 1 (b) in order to increase the diversity of solutions. The increase in the diversity of solutions in Fig. 1 (b) was achieved at the cost of the deterioration in the convergence to the Pareto front. Experimental results in Fig. 1 suggest that the direct use of EMO algorithms is not a good choice for finding the optimal solution of a weighted sum fitness function with very different weight values such as $fitness(\mathbf{x}) = 0.1\, f_1(\mathbf{x}) + 0.9\, f_2(\mathbf{x})$.



(a) Crossover 0.8 and mutation 1/500.



(b) Crossover 0.2 and mutation 5/500.

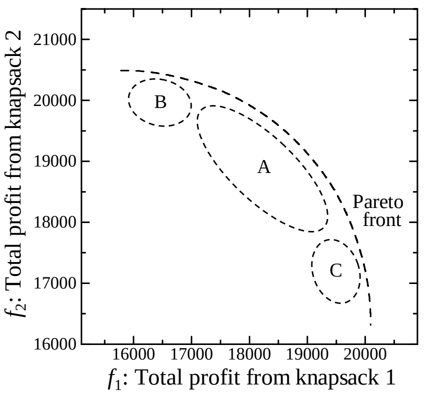
Fig. 1. Experimental results of NSGA-II on the 2-500 knapsack problem using two different settings of the crossover and mutation probabilities.

Another weakness of EMO algorithms is the difficulty in the handling of many objectives. Most EMO algorithms are based on Pareto ranking to evaluate the fitness of each solution. Pareto ranking-based EMO algorithms, however, do not work well on many-objective problems (e.g., see [6], [7], [14], [17]). This is because solutions rarely dominate other solutions in the presence of many objectives. Hughes [7] showed that multiple runs of single-objective optimizers outperformed a single run of EMO algorithms in their applications to many-objective problems. Similar results were also reported in Jaszkiewicz [14]. These results in the literature suggest that the use of EMO algorithms is not a good choice for finding the optimal solution of a scalarizing function generated from many objectives such as the simple sum fitness function of four objectives: $fitness(\mathbf{x}) = f_1(\mathbf{x}) + f_2(\mathbf{x}) + f_3(\mathbf{x}) + f_4(\mathbf{x})$.

The above-mentioned experimental results can be summarized as follows:

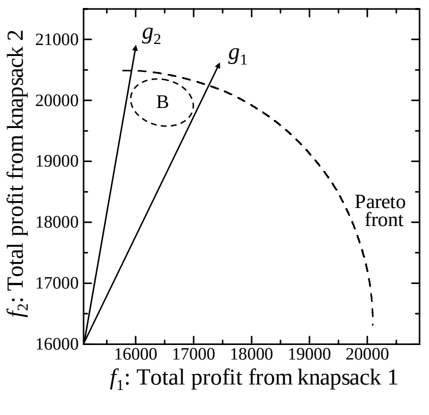
(1) EMO algorithms work well for optimizing a scalarizing function if the location of its optimal solution is near the center of the Pareto front of a two-objective optimization problem. For example, EMO algorithms can easily find good solutions in the region A in Fig. 2 (a) as shown in Fig. 1 (a).
(2) EMO algorithms do not always work well for optimizing a scalarizing function if the location of its optimal solution is near the edge of the Pareto front of a two-objective optimization problem. For example, EMO algorithms do not always easily find good solutions in the region B or C in Fig. 2 (a) as shown in Fig. 1 (b).
(3) EMO algorithms are not likely to work well for optimizing a scalarizing function if they are applied to a many-objective problem.

In this paper, we propose an idea of using an EMO algorithm to efficiently optimize a scalarizing function even in the last two cases: (2) and (3). We generate a new multiobjective problem to which an EMO algorithm is applied. The point is to specify multiple objectives, which are similar to but different from the given scalarizing function, so that the location of the optimal solution is near the center of the Pareto front of the generated multiobjective problem. Our idea is illustrated in Fig. 2 (b) where we generate two objectives g_1 and g_2 in order to efficiently find good solutions in the region B. Slow convergence of EMO algorithms in the case of many objectives is remedied by the use of similar objectives as we will show later in this paper.



21000
20000
19000
18000
17000
16000
16000
17000
18000
19000
20000
B
A
C
Pareto front
f_2: Total profit from knapsack 2
f_1: Total profit from knapsack 1

(a) Three regions in the objective space.



(b) Newly generated objectives: g_1 and g_2.

Fig. 2. Illustration of the proposed idea.

In this paper, we first demonstrate that the performance of EMO algorithms as single-objective optimizers of scalarizing functions highly depends on the choice of multiple objectives in Section 2. Based on this observation, we propose a specification method of multiple objectives for the optimization of a weighted sum fitness function in Section 3. Experimental results show that our approach works very well in the search for not only a single optimal solution but also a number of alternative solutions around the optimal solution. We also show that EMO algorithms work well as single-objective optimizers even in the case of many objectives. In Section 4, the effectiveness of our approach is compared with a hybrid EMO algorithm where a single-objective fitness evaluation scheme is probabilistically used in an EMO algorithm. Then we show that our approach is applicable not only to weighted sum fitness functions but also other scalarizing functions (e.g., those based on constraint conditions and reference solutions) and more general single-objective optimization problems in Section 5. Finally we conclude this paper in Section 6.

2 Optimization of Scalarizing Functions by EMO Algorithms

In this section, we examine the effectiveness of EMO algorithms as single-objective optimizers of scalarizing functions through computational experiments on multiobjective 0/1 knapsack problems in Zitzler & Thiele [19]. As a representative EMO algorithm, we use NSGA-II [3]. For comparison, we also use its single-objective version.

2.1 Scalarizing Functions

Let us consider the following k-objective maximization problem:

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

where $\mathbf{f}(\mathbf{x})$ is the k-dimensional objective vector, and $\mathbf{x}$ is the decision vector.

One of the frequently used scalarizing functions is the weighted sum fitness function with the non-negative weight vector $\mathbf{w} = (w_1, w_2, ..., w_k)$:

$$\mathit{fitness}(\mathbf{x}) = w_1 \cdot f_1(\mathbf{x}) + w_2 \cdot f_2(\mathbf{x}) + \ ...\ + w_k \cdot f_k(\mathbf{x})\,. \tag{2}$$

We assume that the weight vector $\mathbf{w}$ is normalized (i.e., the sum of the weight values is 1). The weight vector $\mathbf{w}$ in (2) is usually supposed to be given by human users.

The weighted sum fitness function with various weight vectors was successfully used to directly realize various search directions in multiobjective genetic local search (MOGLS) algorithms [9], [11], [12]. High performance of MOGLS of Jaszkiewicz [12] was reported [1], [13], [16]. The weighted sum fitness function was also used in hybrid or multi-stage EMO algorithms (e.g., see [8], [10], [16]).

When a reference vector $\mathbf{f}^* = (f_1^*, f_2^*, ..., f_k^*)$ is given as a desired point in the objective space, the distance from $\mathbf{f}^*$ can be used as a scalarizing function:

$$fitness(\mathbf{x}) = distance(\mathbf{f}^*, \mathbf{f}(\mathbf{x}))\,. \tag{3}$$

In this paper, we use the Euclidean distance. The incorporation of reference points into EMO algorithms was examined in Deb & Sundar [4].

Another scalarizing function is based on the transformation of some objectives into inequality conditions. Let us assume that the minimum requirement level for each of the first (k – 1) objectives is given as an inequality condition:

$$f_i(\mathbf{x}) \geq \varepsilon_i \ \text{ for } \ i = 1, 2, ..., k-1\,. \tag{4}$$

The following scalarizing fitness function is usually formulated from the maximization problem of $f_k(\mathbf{x})$ with the (k – 1) inequality conditions in (4):

$$fitness(\mathbf{x}) = f_k(\mathbf{x}) - \alpha \sum_{i=1}^{k-1} \max\{0,\, \varepsilon_i - f_i(\mathbf{x})\}\,, \tag{5}$$

where α is the unit penalty with respect to the violation of the inequality conditions in (4). In computational experiments of this paper, we specified α as $\alpha = 1$.

2.2 NSGA-II and Its Single-Objective Version

NSGA-II is an elitist EMO algorithm with the $(\mu + \lambda)$-ES generation update mechanism. The outline of NSGA-II can be written as follows:

```
[NSGA-II]
Step 1: $P$ = Initialize($P$)
Step 2: While the stopping condition is not satisfied, do
Step 3:           $P'$ = Parent Selection($P$)
Step 4:           $P''$ = Genetic Operations($P'$)
Step 5:           $P$ = Generation Update($P \cup P''$)
Step 6: End while
Step 7: Return Non-dominated($P$)
```

In NSGA-II, each solution in the current population P is evaluated using Pareto ranking and a crowding measure in the following manner for parent selection in Step 3. First the best rank is assigned to all the non-dominated solutions in the current population. Solutions with the best rank are tentatively removed from the current population. Next the second best rank is assigned to all the non-dominated solutions in the remaining population. In this manner, ranks are assigned to all solutions in the current population. The rank of each solution is used as the primary criterion in parent selection. A crowding measure is used to compare solutions with the same rank as the secondary criterion in parent selection (for details, see [2], [3]).

A prespecified number of pairs of parent solutions are selected from the current population by binary tournament selection to form a parent population P' in Step 3.

An offspring solution is generated from each pair of parent solutions by crossover and mutation to form an offspring population *P''* in Step 4. The current population *P* and the offspring population *P''* are merged to form an enlarged population. Each solution in the enlarged population is evaluated by Pareto ranking and the crowding measure as in the parent selection phase. A prespecified number of the best solutions are chosen from the enlarged population as the next population *P* in Step 5. Usually the number of offspring solutions is the same as the population size (i.e., $\mu = \lambda$ in the $(\mu + \lambda)$-ES generation update mechanism).

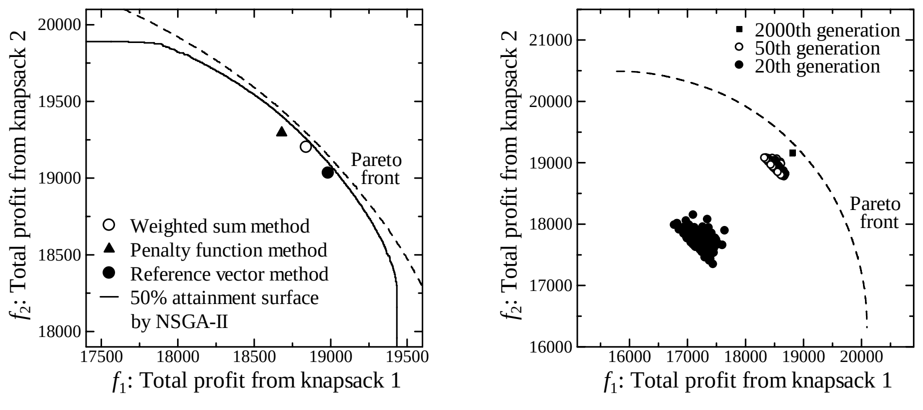
We can easily implement a single-objective version (single-objective genetic algorithm: SOGA) of NSGA-II by using a scalarizing fitness function for parent selection and generation update. Such an SOGA has the $(\mu + \lambda)$-ES generation update mechanism with $\mu = \lambda$. We compare NSGA-II with SOGA through computational experiments on multiobjective 0/1 knapsack problems in Zitzler & Thiele [19].

2.3 Computational Experiments

As in Fig. 1 (a), we applied NSGA-II to the 2-objective 500-item (i.e., 2-500) knapsack problem [19] using the following parameter specifications:

Population size: 200 (i.e., $\mu = \lambda = 200$),
Crossover probability: 0.8 (uniform crossover),
Mutation probability: 1/500 (bit-flip mutation) where 500 is string length,
Termination condition: 2000 generations.

Average results over 50 runs of NSGA-II are summarized as the 50% attainment surface [5] in Fig. 3 (a) where average results of SOGA are also shown for comparison. The weight vector, the reference vector and the minimum requirement level were specified in SOGA as $\mathbf{w} = (0.5, 0.5)$, $\mathbf{f}^* = (19250, 19250)$ and $\varepsilon_1 = 18750$ in Fig. 3 (a), respectively. We can observe that NSGA-II outperformed SOGA in Fig. 3 (a).



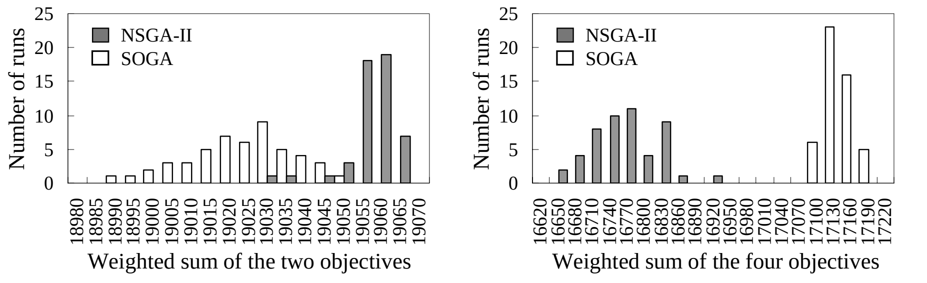
(a) Comparison between NSGA-II and SOGA. (b) SOGA with the weighted sum fitness.

Fig. 3. Experimental results of NSGA-II and SOGA on the 2-500 knapsack problem.

In Fig. 3 (b), we show two intermediate and final populations during a single run of SOGA with the weighted sum fitness function with **w** = (0.5, 0.5). It should be noted that SOGA in Fig. 3 (b) was executed under the same parameter specifications as NSGA-II in Fig. 1 (a). From the comparison between these two figures, we can see that NSGA-II maintained a larger diversity of solutions. The decrease in the diversity of solutions during the execution of SOGA seems to be the main reason of the inferior performance of SOGA in Fig. 3 (a) in comparison with NSGA-II.

NSGA-II and SOGA are further compared with each other in Fig. 4 (a) for the 2-500 knapsack problem using the weighted sum fitness function with **w** = (0.5, 0.5). Fig. 4 (a) shows the distribution of obtained solutions from 50 runs of each algorithm. Whereas good solutions were almost always obtained by NSGA-II, the quality of the final solution by each run of SOGA seems to highly depend on the initial population.

In Fig. 3 (a) and Fig. 4 (a), NSGA-II outperformed SOGA when they were used to optimize the three scalarizing functions. The advantage of NSGA-II over SOGA, however, disappears as the increase in the number of objectives. In Fig. 4 (b), we show experimental results on the four-objective knapsack problem. The performance of NSGA-II as a single-objective optimizer was deteriorated in Fig. 4 by the increase in the number of objectives from two in Fig. 4 (a) to four in Fig. 4 (b). Pareto ranking-based EMO algorithms usually do not work well on many-objective problems.



(a) Results on the two-objective problem. (b) Results on the four-objective problem.

Fig. 4. Comparison between NSGA-II and SOGA. The 2-500 knapsack problem with **w** = (0.5, 0.5) in (a) and the 4-500 knapsack problem with **w** = (0.25, 0.25, 0.25, 0.25) in (b).

The advantage of NSGA-II over SOGA also disappears when the location of the optimal solution of a scalarizing function is near the edge of the Pareto front as we have already explained using Fig. 1. For example, NSGA-II did not work well on the 2-500 knapsack problem as a single-objective optimizer of scalarizing functions in the following cases: **w** = (0.1, 0.9), $\mathbf{f}^*$ = (16750, 20500) and ε_1 = 17000. In each case, the location of the optimal solution is close to the top-left edge of the Pareto front (see Fig. 1 for the spatial relation between the obtained solutions by NSGA-II and the Pareto front). NSGA-II had difficulties in efficiently searching for good solutions near the edge of the Pareto front in comparison with SOGA in these cases.

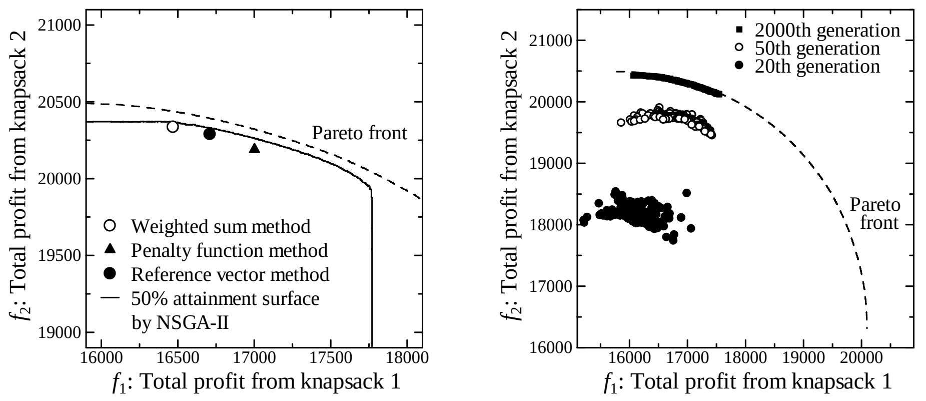
In the above-mentioned three difficult cases, the performance of NSGA-II as a single-objective optimizer was improved when we used the following two objectives:

$$g_1(\mathbf{x}) = 0.5 f_1(\mathbf{x}) + 0.5 f_2(\mathbf{x}) , \qquad (6)$$

$$g_2(\mathbf{x}) = -0.3 f_1(\mathbf{x}) + 1.3 f_2(\mathbf{x}) , \qquad (7)$$

where $f_1(\mathbf{x})$ and $f_2(\mathbf{x})$ are the original two objectives of the 2-500 knapsack problem. Average results over 50 runs of NSGA-II and SOGA are summarized in Fig. 5 (a) where NSGA-II was applied to the two-objective problem in (6) and (7). In Fig. 5 (a), NSGA-II outperformed SOGA in their applications to the optimization of the scalarizing functions in the above-mentioned three difficult cases. Fig. 5 (b) shows two intermediate and final populations during a single run of NSGA-II. The multiobjective search of NSGA-II was appropriately driven toward the desired region by the two objectives in (6) and (7) as we can see from the comparison of Fig. 5 (b) with Fig. 1 (a). As a result, NSGA-II found better solutions of the scalarizing functions than SOGA in Fig. 5 (a).

Experimental results in Fig. 5 suggest that the performance of NSGA-II as a single-objective optimizer highly depends on the specification of multiple objectives. In the next section, we propose a specification method of multiple objectives for the optimization of a weighted sum fitness function. The proposed idea can be used for other scalarizing functions as shown in Section 4.



(a) Comparison between NSGA-II and SOGA. (b) NSGA-II with the new two objectives.

Fig. 5. Experimental results of NSGA-II and SOGA on the 2-500 knapsack problem.

3 Handling of Weighted Sum Fitness Functions

When we use EMO algorithms to optimize a scalarizing function, it is essential to generate multiple objectives so that the location of the optimal solution is near the center of the Pareto front of the generated multiobjective problem. In this section, we show how we can generate such a multiobjective problem to optimize a weighted sum fitness function.

3.1 Weighted Sum Fitness Function of Two Objectives

Our task in this subsection is to optimize the weighted sum of two objectives:

$$fitness(\mathbf{x}) = w_1 \cdot f_1(\mathbf{x}) + w_2 \cdot f_2(\mathbf{x})\,. \tag{8}$$

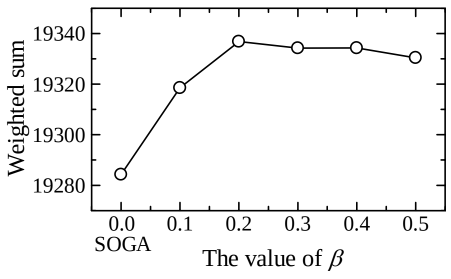
Two objectives can be newly generated by changing the weight vector as follows:

$$g_1(\mathbf{x}) = (w_1 + \beta)\, f_1(\mathbf{x}) + (w_2 - \beta)\, f_2(\mathbf{x})\,, \tag{9}$$

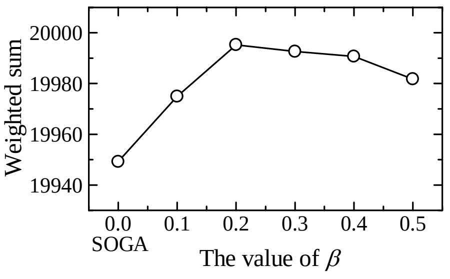
$$g_2(\mathbf{x}) = (w_1 - \beta)\, f_1(\mathbf{x}) + (w_2 + \beta)\, f_2(\mathbf{x})\,. \tag{10}$$

For example, the weight vectors of the newly generated two objectives are specified as (0.2, 0.8) and (0.4, 0.6) from the weight vector **w** = (0.3, 0.7) when $\beta = 0.1$.

Using various specifications of β, we applied NSGA-II to the two-objective problem in (9) and (10) generated from the weighted sum fitness function with **w** = (0.3, 0.7) for the 2-500 knapsack problem. Average results over 50 runs of NSGA-II are summarized in Fig. 6 (a). It should be noted that NSGA-II with $\beta = 0$ is the same as SOGA because $g_1(\mathbf{x})$ and $g_2(\mathbf{x})$ become the same as the original weighted sum fitness function. We also show experimental results for the case of **w** = (0.1, 0.9) in Fig. 6 (b). We can observe in Fig. 6 that the multiobjectivization by (9) and (10) clearly improved the quality of the obtained solutions. We can also observe that the performance of NSGA-II as a single-objective optimizer was not sensitive to the value of β.



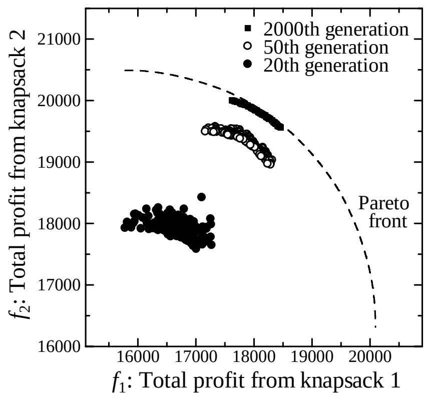
(a) Average results with **w** = (0.3, 0.7).



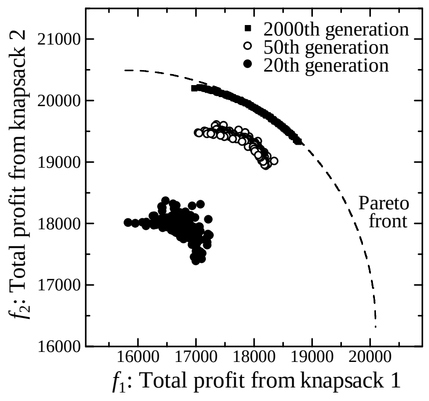
(b) Average results with **w** = (0.1, 0.9).

Fig. 6. Weighted sum optimization by NSGA-II for the 2-500 knapsack problem.

The multiobjectivization by (9) and (10) is effective in the search not only for a single optimal solution but also for multiple good solutions around the optimal solution. In each plot of Fig. 7, we show two intermediate and final populations during a single run of NSGA-II for each of the two cases: $\beta = 0.3$ and $\beta = 0.5$. In both cases, the weight vector of the original weighted sum fitness function was specified as **w** = (0.3, 0.7). In Fig. 7, multiple solutions were obtained along the Pareto front of the original 2-500 knapsack problem. Moreover, the spread of the finally obtained solution set in each plot depended on the value of β. These observations suggest that the multiobjectivization by (9) and (10) can drive the population toward an appropriate search region and adjust its diversity. This means that the proposed idea has a potential usefulness as an approach to the focused search by EMO algorithms.



(a) $\beta = 0.3$ with $\mathbf{w} = (0.3, 0.7)$.



(b) $\beta = 0.5$ with $\mathbf{w} = (0.3, 0.7)$.

Fig. 7. Behavior of NSGA-II with the newly generated two objectives from the 2-500 problem.

3.2 Weighted Sum Fitness Function of Many Objectives

The proposed idea in the previous subsection can be easily generalized to the case of more than two objectives. Let us consider the weighted sum of three objectives with the weight vector $\mathbf{w} = (w_1, w_2, w_3)$. Three objectives can be generated by changing the weight vector $\mathbf{w} = (w_1, w_2, w_3)$ toward the three directions: (1, 0, 0), (0, 1, 0) and (0, 0, 1). More specifically, the weight vectors of the three objectives are generated in the following manner:

$$\mathbf{w}_A = \mathbf{w} + \beta \cdot \frac{\mathbf{a}}{||\mathbf{a}||}, \quad \mathbf{w}_B = \mathbf{w} + \beta \cdot \frac{\mathbf{b}}{||\mathbf{b}||}, \text{ and } \quad \mathbf{w}_C = \mathbf{w} + \beta \cdot \frac{\mathbf{c}}{||\mathbf{c}||}, \tag{11}$$

where $||\mathbf{a}||$ denotes the length of the vector, and **a**, **b**, **c** are specified as follows:

$$\mathbf{a} = (1, 0, 0) - \mathbf{w}, \quad \mathbf{b} = (0, 1, 0) - \mathbf{w}, \text{ and } \quad \mathbf{c} = (0, 0, 1) - \mathbf{w}. \tag{12}$$

This method can be directly generalized to the case with more objectives. We applied NSGA-II to the weighted sum fitness function with $\mathbf{w} = (0.8, 0.1, 0.1)$ for the 3-500 knapsack problem. We also applied SOGA to the same problem. When we used NSGA-II, we generated a new three-objective problem using (11) and (12) with $\beta =$ 0.2. The distribution of the obtained solutions by 50 runs of each algorithm is shown in Fig. 8 (a). We can see from Fig. 8 (a) that NSGA-II outperformed SOGA in their applications to the optimization of the weighted sum of the three objectives. It should be noted that NSGA-II did not work well for the same task as a single-objective optimizer when it was applied to the original 3-500 knapsack problem.

We also performed the same computational experiments on the weighted sum fitness function with **w** = (0.25, 0.25, 0.25, 0.25) for the 4-500 knapsack problem. Experimental results are shown in Fig. 8 (b). As in Fig. 8 (a), NSGA-II outperformed SOGA in Fig. 8 (b). The comparison between Fig. 4 (b) and Fig. 8 (b) clearly demonstrates the effect of the proposed idea on the performance of NSGA-II as a single-objective optimizer of the weighted sum fitness function. It should be noted that the difficulty of Pareto ranking-based EMO algorithms in the handling of many objectives was remedied by the use of similar objectives in our approach as shown in Fig. 8.

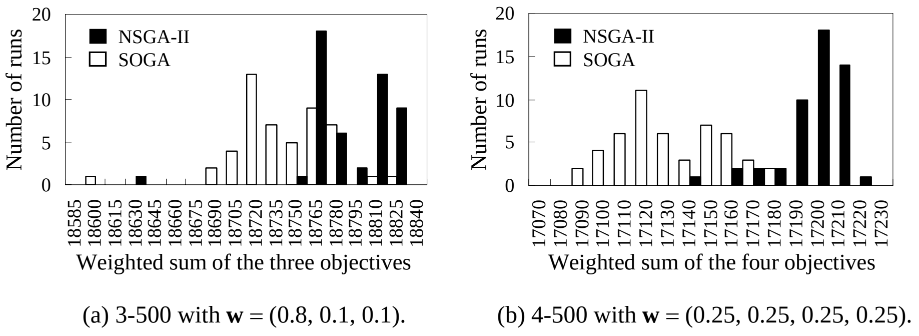


Fig. 8. Weighted sum optimization by NSGA-II using the proposed approach with $\beta = 0.2$.

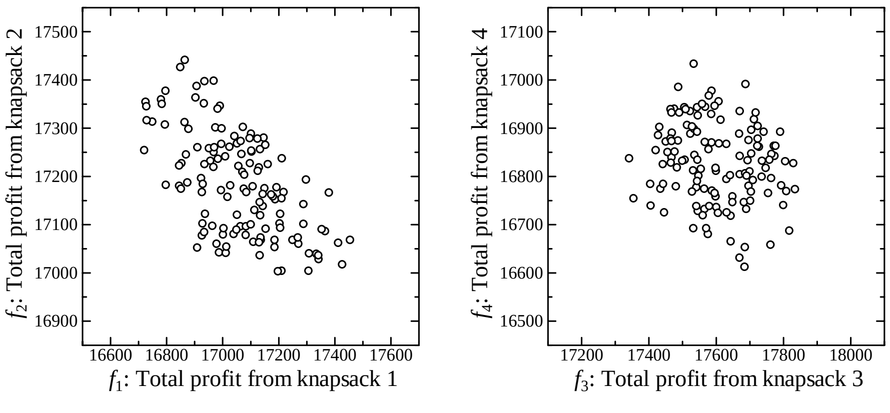


Fig. 9. Obtained solutions by a single run of NSGA-II on the modified 4-500 problem.

As shown in Fig. 8 (b), NSGA-II worked very well as a single-objective optimizer to search for the optimal solution of the weighted sum of the four objectives of the 4-500 knapsack problem. It also has the ability to find multiple non-dominated solutions as an EMO algorithm. In Fig. 9, we show the obtained non-dominated solution set by a single run of NSGA-II on the modified 4-500 knapsack problem, which was generated from the 4-500 knapsack problem with **w** = (0.25, 0.25, 0.25, 0.25) using (11) and (12) with $\beta = 0.2$. Each plot in Fig. 9 shows the projection of the obtained non-dominated solution set from the original four-dimensional objective space onto a two-dimensional space. From Fig. 9, we can see that a number of non-dominated solutions were obtained by a single run of NSGA-II.

4 Application of a Hybrid EMO Algorithm

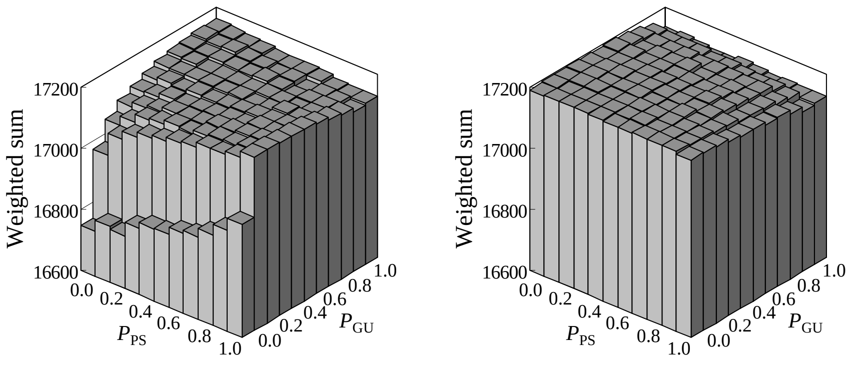
In our former studies [8], [10], we proposed a hybrid EMO algorithm where a weighted sum fitness function was probabilistically used in NSGA-II. We introduced two probabilities P_{PS} and P_{GU}, which specified how often the weighted sum fitness function was used for parent selection and generation update in the hybrid EMO algorithm, respectively. In this section, we compare our approach (i.e., application of NSGA-II to modified multiobjective problems) with the hybrid EMO algorithm.

One extreme case of the hybrid EMO algorithm with $P_{\mathrm{PS}} = P_{\mathrm{GU}} = 0.0$ is exactly the same as NSGA-II since the weighted sum fitness function is never used. Another extreme case with $P_{\mathrm{PS}} = P_{\mathrm{GU}} = 1.0$ is the same as SOGA since the weighted sum fitness function is always used. The balance between single-objective and multiobjective search can be adjusted between the two extreme cases using the two probabilities.

In our computational experiments in this section, we examined the following 11x11 combinations of the two probabilities:

Probability P_{PS}: 0.0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0,
Probability P_{GU}: 0.0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0.

We applied the hybrid EMO algorithm to the 4-500 knapsack problem to optimize the weighted sum fitness function with the weight vector **w** = (0.25, 0.25, 0.25, 0.25). This weighted sum fitness function was used for parent selection with the probability P_{PS} and generation update with the probability P_{GU} in the hybrid EMO algorithm. When the weighted sum fitness function was not used, the multiobjective fitness evaluation scheme in NSGA-II was invoked to evaluate each solution based on the original four objectives in the 4-500 knapsack problem. Average results over 50 runs are summarized in Fig. 10 (a). The bottom-left bar with $P_{\mathrm{PS}} = P_{\mathrm{GU}} = 0.0$ shows the result of NSGA-II while the top-right bar with $P_{\mathrm{PS}} = P_{\mathrm{GU}} = 1.0$ shows the result of SOGA. Whereas the performance of NSGA-II was very poor in Fig. 10 (a), it was significantly improved by the probabilistic use of the weighted sum fitness function. Better results than SOGA were obtained by the hybrid EMO algorithm in the top-left corner with $P_{\mathrm{PS}} = 0.0$ and $P_{\mathrm{GU}} = 1.0$ in Fig. 10 (a). We also applied the hybrid EMO algorithm to the same problem after modifying the 4-500 knapsack problem using the proposed approach with $\beta = 0.2$. Experimental results were shown in Fig. 10 (b) where good results were obtained even when the weighted sum fitness function was not used (i.e., the bottom-left bar with $P_{\mathrm{PS}} = P_{\mathrm{GU}} = 0.0$). That is, the hybridization is not necessary in Fig. 10 (b) where we modified the 4-500 knapsack problem by the proposed approach. Moreover, we can observe that better results were obtained in Fig. 10 (b) after the modification of the 4-500 problem than Fig. 10 (a).



(a) Original 4-500 problem. (b) Modified 4-500 problem.

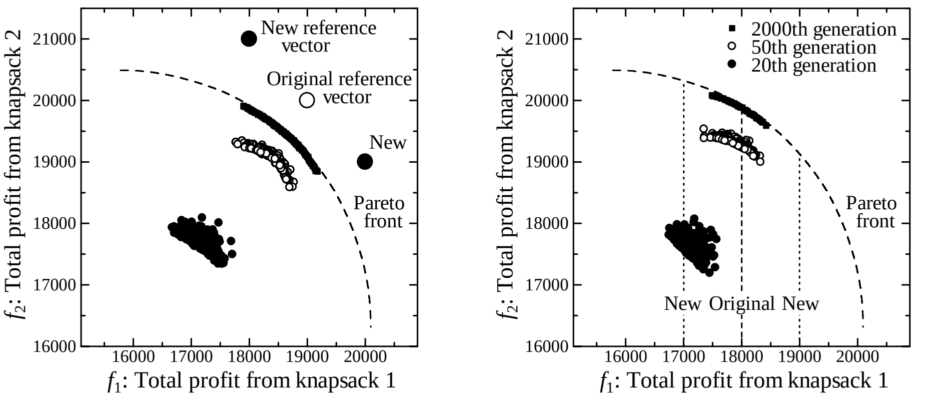
Fig. 10. Effect of our approach on the performance of the hybrid EMO algorithm.

5 Handling of Other Scalarizing Fitness Functions

The basic idea of our approach is to generate multiple objectives, which are similar to but different from the given scalarizing function, so that the location of its optimal solution is near the center of the Pareto front of the generated multiobjective problem. This idea can be also implemented for other scalarizing functions.

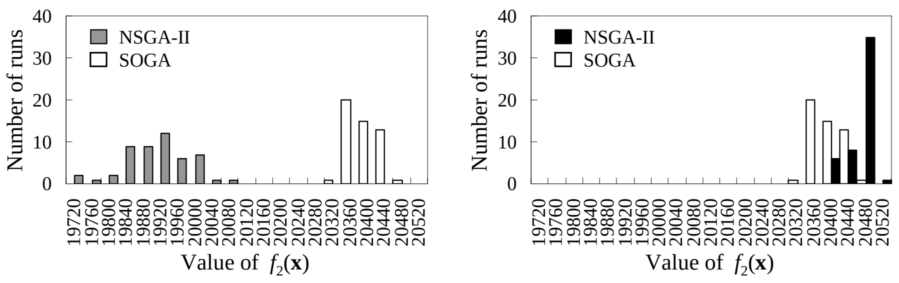
For example, let us assume that we have a reference vector (19000, 20000) for the 2-500 knapsack problem. In this case, we can generate two objectives by specifying two reference vectors around (19000, 20000). Experimental results with newly generated two reference vectors (18000, 21000) and (20000, 19000) are shown in Fig. 11 (a). On the other hand, when we have a minimum requirement level (e.g., 18000) for the first objective of the 2-500 knapsack problem, we can generate two objectives by specifying two minimum requirement levels around 18000 (e.g., 17000 and 19000). Experimental results with the newly generated minimum requirement levels 17000 and 19000 are shown in Fig. 11 (b). We can observe in Fig. 11 that the search of NSGA-II was appropriately directed by the newly generated multiple objectives. We can also observe that good alternative solutions were obtained around the optimal solution of the original scalarizing function in each plot in Fig. 11.

Our approach is applicable not only to the optimization of scalarizing function but also to other optimization problems. For example, let us consider the maximization of $f(\mathbf{x})$. If we have another objective $g(\mathbf{x})$, we can generate two objectives as $f(\mathbf{x}) + w \cdot g(\mathbf{x})$ and $f(\mathbf{x}) - w \cdot g(\mathbf{x})$. In this case, the choice of $g(\mathbf{x})$ is not so important because its effect can be adjusted by the weight w. The direct use of $f(\mathbf{x})$ and $g(\mathbf{x})$ as two objectives is not a good strategy for optimizing $f(\mathbf{x})$ because the optimal solution of $f(\mathbf{x})$ is located at the edge of the Pareto front of the two-objective problem with $f(\mathbf{x})$ and $g(\mathbf{x})$. In Fig. 12, we show experimental results on the optimization of $f_2(\mathbf{x})$ of the 2-500 knapsack problem. We used $f_1(\mathbf{x})$ and $f_2(\mathbf{x})$ as two objectives in Fig. 12 (a), which is not a good strategy. On the other hand, we used $f_2(\mathbf{x}) + 0.3\, f_1(\mathbf{x})$ and $f_2(\mathbf{x}) - 0.3\, f_1(\mathbf{x})$ in Fig. 12 (b), which is a good strategy as multiobjectivization.



(a) Results based on two reference vectors. (b) Results based on two requirement levels.

Fig. 11. Experimental results on the 2-500 knapsack problem using other scalarizing functions.



(a) Use of the original 2-500 problem. (b) Use of the modified 2-500 problem.

Fig. 12. Optimization of $f_2(\mathbf{x})$ of the 2-500 knapsack problem by NSGA-II.

6 Conclusions

In this paper, we proposed an idea of using an EMO algorithm to optimize a scalarizing function. Our approach generates multiple objectives, which are similar to but different from the given scalarizing function, so that the location of the optimal solution of the scalarizing function is near the center of the Pareto front of the generated multiobjective problem. The effectiveness of our approach was examined through various computational experiments using NSGA-II. Experimental results showed that the performance of NSGA-II as a single objective optimizer highly depends on the choice of multiple objectives. One interesting observation is that NSGA-II worked very well even when it was applied to a four-objective 0/1 knapsack problem generated by our approach (whereas NSGA-II usually does not work well for many-objective problems). This is because our approach generates similar objectives.

This work was partially supported by Grant-in-Aid for Scientific Research on Priority Areas (18049065) and for Scientific Research (B) (17300075).

References

1. Colombo, G., Mumford, C. L.: Comparing Algorithms, Representations and Operators for the Multi-Objective Knapsack Problem. Proc. of 2005 Congress on Evolutionary Computation (2005) 2241-2247
2. Deb, K.: Multi-Objective Optimization Using Evolutionary Algorithms. John Wiley & Sons, Chichester (2001)
3. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II. IEEE Trans. on Evolutionary Computation 6 (2002) 182-197
4. Deb, K., Sundar, J.: Reference Point Based Multi-Objective Optimization Using Evolutionary Algorithms. Proc. of Genetic and Evolutionary Computation Conference (2006) 635-642
5. Fonseca, C. M., Fleming, P. J.: On the Performance Assessment and Comparison of Stochastic Multiobjective Optimizers. Lecture Notes in Computer Science, Vol. 114: PPSN IV. Springer, Berlin (1996) 584-593
6. Hughes, E. J.: Multiple Single Objective Sampling. Proc. of 2003 Congress on Evolutionary Computation (2003) 2678-2684
7. Hughes, E. J.: Evolutionary Many-Objective Optimization: Many Once or One Many? Proc. of 2005 Congress on Evolutionary Computation (2005) 222-227
8. Ishibuchi, H., Doi, T., Nojima, Y.: Incorporation of Scalarizing Fitness Functions into Evolutionary Multiobjective Optimization Algorithms. Lecture Notes in Computer Science, Vol. 4193: PPSN IX. Springer, Berlin (2006) 493-502
9. Ishibuchi, H., Murata, T.: 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 28 (1998) 392-403
10. Ishibuchi, H., Nojima, Y., Doi, T.: Application of Multiobjective Evolutionary Algorithms to Single-Objective Optimization Problems. Abstract Booklet of 7th International Conference on Multi-Objective Programming and Goal Programming (2006) 4 pages
11. Ishibuchi, H., Yoshida, T., Murata, T.: Balance between Genetic Search and Local Search in Memetic Algorithms for Multiobjective Permutation Flowshop Scheduling. IEEE Trans. on Evolutionary Computation 7 (2003) 204-223
12. Jaszkiewicz, A.: Genetic Local Search for Multi-Objective Combinatorial Optimization. European Journal of Operational Research 137 (2002) 50-71
13. Jaszkiewicz, A.: On the Performance of Multiple-Objective Genetic Local Search on the 0/1 Knapsack Problem - A Comparative Experiment. IEEE Trans. on Evolutionary Computation 6 (2002) 402-412
14. Jaszkiewicz, A.: On the Computational Efficiency of Multiple Objective Metaheuristics: The Knapsack Problem Case Study. European Journal of Operational Research 158 (2004) 418-433
15. Knowles, J. D., Watson, R. A., Corne, D. W.: Reducing Local Optima in Single-Objective Problems by Multi-Objectivization. Lecture Notes in Computer Science, Vol. 1993: EMO 2001. Springer, Berlin (2001) 269-283
16. Mumford, C. L.: A Hierarchical Solve-and-Merge Framework for Multi-Objective Optimization. Proc. of 2005 Congress on Evolutionary Computation (2005) 2241-2247
17. Purshouse, R. C., Fleming, P. J.: Evolutionary Many-Objective Optimization: An Exploratory Analysis. Proc. of 2003 Congress on Evolutionary Computation (2003) 2066-2073
18. Watanabe, S., Sakakibara K.: Multi-Objective Approaches in a Single-Objective Optimization Environment. Proc. of 2005 Congress on Evolutionary Computation (2005) 1714-1721
19. Zitzler, E., Thiele, L.: Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach. IEEE Trans. on Evolutionary Computation 3 (1999) 257-271