Adaptation of Scalarizing Functions in MOEA/D: An Adaptive Scalarizing Function-Based Multiobjective Evolutionary Algorithm

Hisao Ishibuchi, Yuji Sakane, Noritaka Tsukamoto, 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,nojima}@cs.osakafu-u.ac.jp,
{sakane,nori}@ci.cs.osakafu-u.ac.jp
http://www.ie.osakafu-u.ac.jp/~hisaoi/ci_lab_e

Abstract. It is well-known that multiobjective problems with many objectives are difficult for Pareto dominance-based algorithms such as NSGA-II and SPEA. This is because almost all individuals in a population are non-dominated with each other in the presence of many objectives. In such a population, the Pareto dominance relation can generate no strong selection pressure toward the Pareto front. This leads to poor search ability of Pareto dominance-based algorithms for many-objective problems. Recently it has been reported that better results can be obtained for many-objective problems by the use of scalarizing functions. The weighted sum usually works well in scalarizing function-based algorithms when the Pareto front is convex. However, we need other functions such as the weighted Tchebycheff when the Pareto front is non-convex. In this paper, we propose an idea of automatically choosing between the weighted sum and the weighted Tchebycheff for each individual in each generation. The characteristic feature of the proposed idea is to use the weighted Tchebycheff only when it is needed for individuals along non-convex regions of the Pareto front. The weighted sum is used for the other individuals in each generation. The proposed idea is combined with a high-performance scalarizing function-based algorithm called MOEA/D (multiobjective evolutionary algorithm based on decomposition) of Zhang and Li (2007). Effectiveness of the proposed idea is demonstrated through computational experiments on modified multiobjective knapsack problems with non-convex Pareto fronts.

1 Introduction

Evolutionary multiobjective optimization (EMO) has been and will continue to be one of the most active research areas in the field of evolutionary computation. EMO algorithms have been successfully applied to various multiobjective optimization problems [3], [4]. Almost all well-known EMO algorithms such as NSGA-II [5] and SPEA [26] use the Pareto dominance relation together with a crowding measure for the fitness evaluation of each individual. In this sense, we call them Pareto dominance-based algorithms. Pareto dominance-based algorithms usually work very well for finding Pareto-optimal or near Pareto-optimal solutions along the Pareto front of

M. Ehrgott et al. (Eds.): EMO 2009, LNCS 5467, pp. 438–452, 2009.

an optimization problem with two or three objectives. The search ability of those algorithms is, however, severely deteriorated by the increase in the number of objectives as pointed out in the literature (See [18], [22] for early studies, and [12], [13] for a short review on evolutionary many-objective optimization studies). This is because almost all individuals in a population become non-dominated with each other under many objectives. When all individuals in a population are non-dominated, the Pareto dominance relation cannot generate any selection pressure toward the Pareto front. In this case, the fitness evaluation of each individual is based on only a crowding measure in each Pareto dominance-based algorithm. Thus their search ability is severely deteriorated by the increase in the number of objectives. It has been demonstrated in the literature [7]-[10], [16], [17] that better results can be obtained by the use of scalarizing functions for many-objective problems with more than three objectives.

Whereas Pareto dominance-based algorithms usually work well on two-objective and three-objective problems, it is not always the case in their applications to combinatorial optimization problems [6]. For example, it has been demonstrated in some studies [16], [17], [23] that Pareto dominance-based algorithms are not good at finding Pareto-optimal or near Pareto-optimal solutions around the edges of the Pareto front of a two-objective knapsack problem. A possible reason for this observation is that the Pareto dominance relation has a too large effect on the fitness evaluation for the two-objective knapsack problem especially in the early stage of evolution. This means that a crowding measure does not have a sufficient effect on the fitness evaluation for increasing the diversity of solutions. As a result, good solutions are obtained in a small region around the center of the Pareto front. It has been reported in the above-mentioned studies [16], [17] that much better results in terms of the diversity of solutions are obtained by scalarizing function-based algorithms.

As we have already explained, scalarizing function-based algorithms have several advantages such as the scalability to many-objective problems and high search ability for combinatorial optimization problems. Moreover, scalarizing function-based fitness evaluation needs much less computation load than Pareto dominance-based one especially for many-objective problems. That is, computational efficiency is another advantage of scalarizing function-based algorithms.

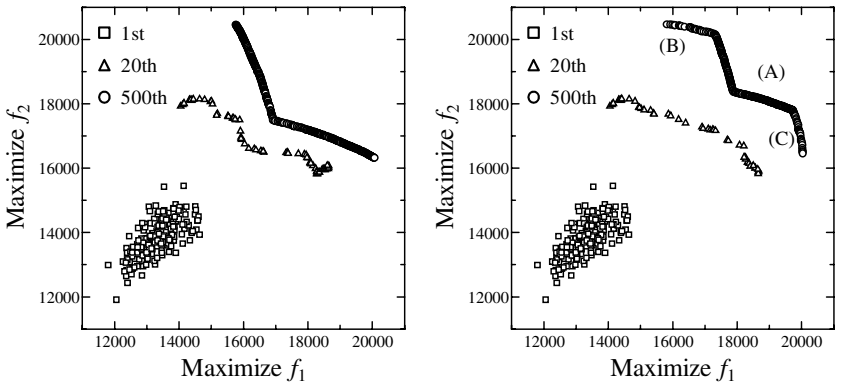
One promising approach for improving the search ability of EMO algorithms is the hybridization with local search [11], [14]-[17], [24]. Local search in such a hybrid EMO algorithm usually uses a scalarizing function (e.g., the weighted sum in multiobjective genetic local search [11] and an augmented achievement scalarizing function in a recent hybrid EMO algorithm [24]). It is easy to incorporate local search into scalarizing function-based algorithms whereas the implementation of local search in Pareto dominance-based algorithms is not always easy. High compatibility with local search is also another advantage of scalarizing function-based algorithms.

One important issue in the implementation of scalarizing function-based algorithms is the choice of an appropriate scalarizing function. For example, both the weighted sum and the weighted Tchebycheff were implemented and compared in a scalarizing function-based algorithm called MOEA/D (multiobjective evolutionary algorithm based on decomposition [25]). It is well-known that the weighted sum cannot appropriately handle multiobjective problems with non-convex Pareto fronts whereas the weighted Tchebycheff can handle them. Moreover, the weighted Tchebycheff has nice theoretical properties [20]. Thus the weighted Tchebycheff seems to be

a better choice in scalarizing function-based algorithms than the weighted sum. It was, however, clearly demonstrated in [25] that better results were obtained from the weighted sum than the weighted Tchebycheff in the application of MOEA/D to multiobjective knapsack problems. This observation may suggest the use of the weighted sum for multiobjective problems with convex Pareto fronts whereas its use is not appropriate for other problems with non-convex Pareto fronts.

From these discussions, the shape of the Pareto front seems to be a good indicator for choosing an appropriate scalarizing function. One difficulty in using this indicator is that the shape of the Pareto front is often unknown before we find a large number of Pareto-optimal solutions. Another possible difficulty is related to the use of the same scalarizing function for all individuals in every generation.

When the current population is far from the Pareto front, we can use the weighted sum even if its shape is non-convex. This situation is illustrated by a population of small squares in the bottom-left area of each plot in Fig. 1. This figure shows experimental results of MOEA/D with the weighted Tchebycheff on modified two-objective 500-item knapsack problems with non-convex Pareto fronts. In each plot, three populations at the first, 20th and 500th generations are shown by squares, triangles and circles, respectively. We can see that the population at the first generation is far from the Pareto front. Thus the weighted sum can be used in the early stage of evolution (e.g., in the first 20 generations). When the current population is close to the non-convex Pareto front (e.g., after the 20th generation), the use of the weighted Tchebycheff seems to be appropriate for all solutions in Fig. 1 (a). On the other hand, we need the weighted Tchebycheff only for solutions around the non-convex region (A) of the Pareto front in Fig. 1 (b). The weighted sum can be used throughout the execution of MOEA/D for the other solutions around the two convex regions (B and C) of the Pareto front in Fig. 1 (b). Motivated by these discussions, we propose an idea of automatically choosing between the weighted sum and the weighted Tchebycheff.



(a) A non-convex Pareto front. (b) A partially non-convex Pareto front.

Fig. 1. Experimental results of MOEA/D [25] with the weighted Tchebycheff on modified two-objective 500-item knapsack problems with non-convex Pareto fronts

This paper is organized as follows. In Section 2, we explain the two scalarizing functions (i.e., the weighted sum and the weighted Tchebycheff) used in MOEA/D. In Section 3, we explain MOEA/D. After that, we compare the two scalarizing functions with each other in Section 4 through computational experiments on multiobjective 500-item knapsack problems with two, four and six objectives. In Section 5, we explain how we can generate multiobjective knapsack problems with non-convex Pareto fronts by modifying the original ones in Zitzler and Thiele [26]. For example, we can generate the two knapsack problems in Fig. 1. In Section 6, we explain our idea for automatically choosing between the weighted sum and the weighted Tchebycheff for the use in MOEA/D. We examine the effectiveness of the proposed idea through computational experiments on modified multiobjective knapsack problems with non-convex Pareto fronts in Section 7. Finally we conclude this paper in Section 8.

2 Scalarizing Functions

Let us consider the following m-objective maximization problem:

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

where $\boldsymbol{f}(\boldsymbol{x})$ is the m-dimensional objective vector, $f_i(\boldsymbol{x})$ is the i-th objective to be maximized, and $\boldsymbol{x}$ is the decision vector. Since we use multiobjective knapsack problems in our computational experiments, we explain our idea for the multiobjective maximization problem in (1).

One of well-known and frequently-used scalarizing functions is the weighted sum with a non-negative weight vector $\boldsymbol{\lambda} = (\lambda_1, \lambda_2, ..., \lambda_m)$:

$$g^{WS}(\boldsymbol{x} \mid \boldsymbol{\lambda}) = \lambda_1 \cdot f_1(\boldsymbol{x}) + \lambda_2 \cdot f_2(\boldsymbol{x}) + \ \ ... \ + \lambda_m \cdot f_m(\boldsymbol{x})\,, \tag{2}$$

where λ_i is a non-negative weight for the i-th objective $f_i(\boldsymbol{x})$. We assume that the weight vector $\boldsymbol{\lambda}$ satisfies $\lambda_1 + \lambda_2 + ... + \lambda_m = 1$ and $\lambda_i \geq 0$ for $i = 1, 2, ..., m$.

Another well-known scalarizing function is the weighted Tchebycheff. Let Ω be a set of solutions. A reference point $\boldsymbol{z}^* = (z_1^*, z_2^*, ..., z_m^*)$ in the objective space can be specified by the best objective value of each objective in Ω as

$$z_i^* = \max\{ f_i(\boldsymbol{x}) \mid \boldsymbol{x} \in \Omega \},\ i = 1, 2, ..., m\,. \tag{3}$$

When Ω is the set of all feasible solutions, $\boldsymbol{z}^*$ is called the ideal vector.

The weighted Tchebycheff measures the distance from the reference point $\boldsymbol{z}^*$ to a solution $\boldsymbol{x}$ in the objective space as follows:

$$g^{TE}(\boldsymbol{x} \mid \boldsymbol{\lambda}, \boldsymbol{z}^*) = \max_{i=1,2,...,m} \{ \lambda_i \cdot \mid z_i^* - f_i(\boldsymbol{x}) \mid \}\,. \tag{4}$$

As in Zhang and Li [25], we specified the reference point $\boldsymbol{z}^*$ for each generation in our computational experiments as follows:

$$z_i^* = 1.1 \cdot \max\{ f_i(\boldsymbol{x}) \mid \boldsymbol{x} \in \Omega(t) \}, \ i = 1, 2, ..., m, \tag{5}$$

where $\Omega(t)$ is the secondary population at the t-th generation of MOEA/D [25]. The reference point $\boldsymbol{z}^*$ is updated whenever the best value in (5) is updated.

3 MOEA/D of Zhang and Li (2007)

MOEA/D (multiobjective evolutionary algorithm based on decomposition) was proposed by Zhang and Li [25] as a high-performance scalarizing function-based algorithm. We use MOEA/D because it is a simple but very powerful EMO algorithm. MOEA/D has a number of advantages over Pareto dominance-based algorithms such as the scalability to many-objective problems, high performance for combinatorial optimization problems, computational efficiency of fitness evaluation, and high compatibility with local search.

The main characteristic of MOEA/D is that a multiobjective problem is handled as a collection of a large number of single-objective problems. Each single-objective problem has a scalarizing function with a different weight vector. Each weight vector has a single individual in each population. This idea is similar to a cellular EMO algorithm of Murata et al. [21] where a different weight vector was assigned to each cell. In both algorithms, each individual in the current population was governed by a scalarizing function with a different weight vector. The decomposition of an original multiobjective problem into a number of multiobjective problems was also discussed in other studies (e.g., cone separation by Branke et al. [1]). High performance of MOEA/D has already been demonstrated for well-known test problems such as the ZDT and DTLZ series and knapsack problems in [25], flowshop scheduling problems in [2], and some difficult problems with complicated Pareto fronts [19].

MOEA/D uses a prespecified number of uniformly distributed weight vectors satisfying the following two conditions:

$$\lambda_1 + \lambda_2 + ... + \lambda_m = 1, \tag{6}$$

$$\lambda_i \in \left\{ 0, \frac{1}{H}, \frac{2}{H}, ..., \frac{H}{H} \right\}, \ i = 1, 2, ..., m, \tag{7}$$

where H is a user-definable positive integer. The number of weight vectors is calculated as $N = {}_{H+m-1}C_{m-1}$ [25]. For example, we have 101 weight vectors by specifying H as $H = 100$ for a two-objective problem: $\boldsymbol{\lambda}$ = (0, 1), (0.01, 0.99), ..., (1, 0).

Let us denote the generated N weight vectors as $\{\boldsymbol{\lambda}^1, \boldsymbol{\lambda}^2, ..., \boldsymbol{\lambda}^N\}$. Each weight vector $\boldsymbol{\lambda}^k$ has the nearest T weight vectors (including $\boldsymbol{\lambda}^k$ itself) as its neighbors where T is a user-definable positive integer. We denote the T neighbors of $\boldsymbol{\lambda}^k$ by $B(\boldsymbol{\lambda}^k)$, which can

be viewed as the neighborhood of size T for the weight vector $\boldsymbol{\lambda}^k$. The distance between two weight vectors is measured by the standard Euclidean distance.

The same neighborhood structure is used for individuals since each weight vector has a single individual. Let us denote the individual associated with the weight vector $\boldsymbol{\lambda}^k$ by $\boldsymbol{x}^k$. Then we denote the T individuals associated with the T weight vectors in $B(\boldsymbol{\lambda}^k)$ by $B(\boldsymbol{x}^k)$, which is referred to as the neighborhood of $\boldsymbol{x}^k$. We also call the T individuals in $B(\boldsymbol{x}^k)$ as the neighbors of $\boldsymbol{x}^k$. In MOEA/D, genetic operations for each individual are locally performed among its neighbors as in cellular algorithms.

Now we have N weight vectors. We also have the T neighbors in $B(\boldsymbol{\lambda}^k)$ for each weight vector $\boldsymbol{\lambda}^k$, $k = 1, 2, ..., N$. As in standard evolutionary algorithms, the first step of MOEA/D is to generate an initial population. It should be noted that the population size is the same as the number of the weight vectors (i.e., N). We randomly generate an initial individual for each weight vector. Next we generate an offspring for each weight vector by selection, crossover and mutation. When an offspring is to be generated for the weight vector $\boldsymbol{\lambda}^k$, a couple of parents are randomly selected among the T neighbors of $\boldsymbol{x}^k$ in $B(\boldsymbol{x}^k)$. Then an offspring is generated by crossover and mutation. Let us denote the generated offspring by $\boldsymbol{y}^k$. If the offspring $\boldsymbol{y}^k$ is better than the current individual $\boldsymbol{x}^k$, $\boldsymbol{x}^k$ is replaced with $\boldsymbol{y}^k$. The two individuals $\boldsymbol{x}^k$ and $\boldsymbol{y}^k$ are compared with each other by the scalarizing function with the weight vector $\boldsymbol{\lambda}^k$ (i.e., the weighted sum or the weighted Tchebycheff). The newly generated offspring $\boldsymbol{y}^k$ is also compared with all neighbors in $B(\boldsymbol{x}^k)$. This comparison is performed using the weight vector of each neighbor. If $\boldsymbol{y}^k$ is better than some neighbors, they are replaced with $\boldsymbol{y}^k$. The genetic operations (i.e., selection, crossover, mutation) and the comparison of the newly generated offspring with all neighbors in $B(\boldsymbol{x}^k)$ are performed for each individual $\boldsymbol{x}^k$ (i.e., $k = 1, 2, ..., N$) in the current population. We used the total number of examined solutions as the stopping condition in our computational experiments.

MOEA/D has a secondary population (i.e., an archive population) for storing non-dominated solutions. The secondary population is updated by newly generated offspring throughout the execution of MOEA/D. The secondary population is used just for storing non-dominated solutions. That is, no individual in the secondary population is used in the genetic operations for generating new offspring. This means that the secondary population has no effect on the search behavior of MOEA/D. The use of non-dominated solutions in the secondary population usually significantly improves the search ability of EMO algorithms (e.g., [11], [26]). In MOEA/D, the "replace-if-better" strategy is used for all individuals. This replacement strategy can be viewed as a kind of elitism. Thus MOEA/D has high search ability without utilizing non-dominated solutions in the secondary population as parents in the genetic operations for generating new offspring.

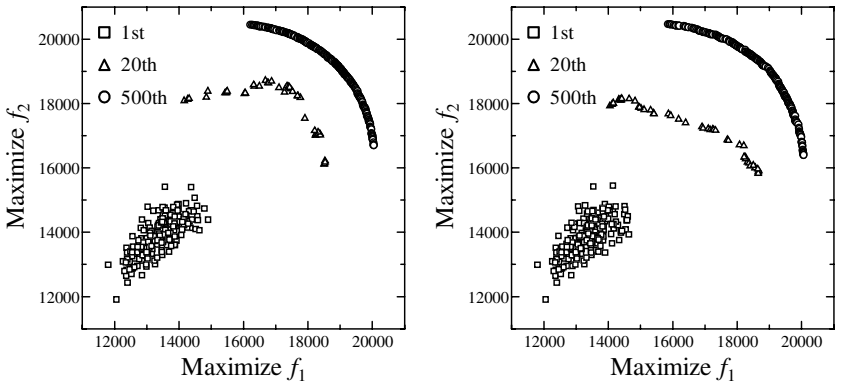
4 Comparison of the Two Scalarizing Functions in MOEA/D

In this section, we compare the weighted sum and the weighted Tchebycheff with each other through computational experiments. We used the two-objective and four-objective knapsack problems in Zitzler and Thiele [26]. We also generated a six-objective 500-item knapsack problem. We denote these test problems by the number of objectives

and the number of items as the 2-500, 4-500 and 6-500 problems. In our computational experiments, we used the following parameter specifications:

Population size (which is the same as the number of weight vectors):
200 (2-500), 220 (4-500), and 252 (6-500),
Parameter H for generating weight vectors:
199 (2-500), 9 (4-500), and 5 (6-500),
Stopping condition (i.e., the total number of examined solutions):
100,000 (2-500), 150,000 (4-500), and 200,000 (6-500),
Crossover probability: 0.8 (Uniform crossover),
Mutation probability: 1/500 (Bit-flip mutation),
Neighborhood size T (i.e., the number of neighbors): 10.

Our computational experiments were performed in the same manner as in Zhang and Li [25]. For example, we used the same greedy repair method as in [25]. We show experimental results of a single run of MOEA/D with the weighted sum in Fig. 2 (a) and the weighted Tchebycheff in Fig. 2 (b). In each plot, we show all individuals in the current and secondary populations at each generation.



(a) The weighted sum. (b) The weighted Tchebycheff.

Fig. 2. Experimental results of a single run of MOEA/D with the weighted sum and the weighted Tchebycheff on the 2-500 knapsack problem of Zitzler and Thiele [26]

In Fig. 2, we cannot observe any clear difference in the final results at the 500th generation between Fig. 2 (a) with the weighted sum and Fig. 2 (b) with the weighted Tchebycheff. There exists, however, a clear difference in the intermediate results at the 20th generation between the two plots in Fig. 2. That is, we can see that the faster convergence speed was achieved by the weighted sum in Fig. 2 (a) than the weighted Tchebycheff in Fig. 2 (b) in the application of MOEA/D to the 2-500 test problem.

In Figs. 3-5, we compare the two scalarizing functions with each other using the hypervolume measure. Each figure shows the histogram of the values of the hypervolume measure obtained by 100 runs of MOEA/D with each scalarizing function. Whereas

there is no clear difference between the two scalarizing functions for the 2-500 problem in Fig. 3 (also see Fig. 2), much better results were obtained by the weighted sum for the 4-500 problem in Fig. 4 and for the 6-500 problem in Fig. 5.

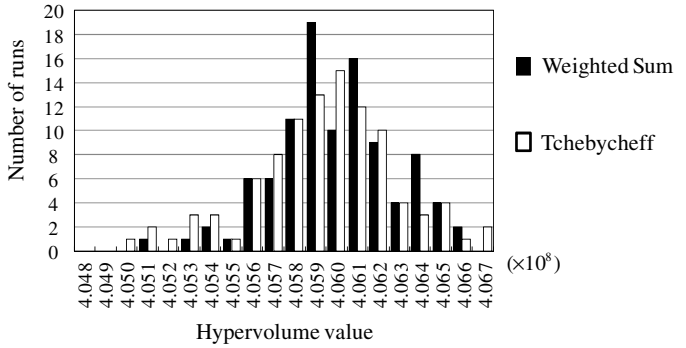


Fig. 3. Distribution of 100 values of the hypervolume measure (2-500 problem)

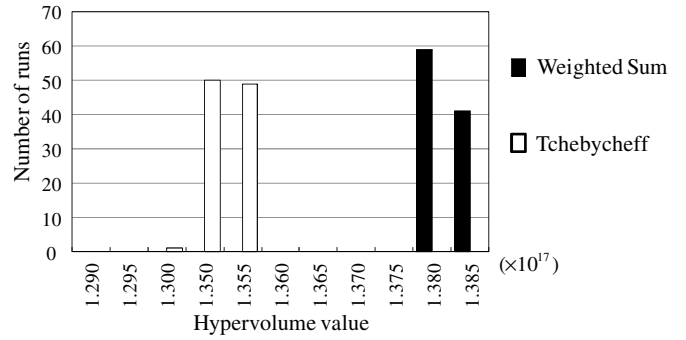


Fig. 4. Distribution of 100 values of the hypervolume measure (4-500 problem)

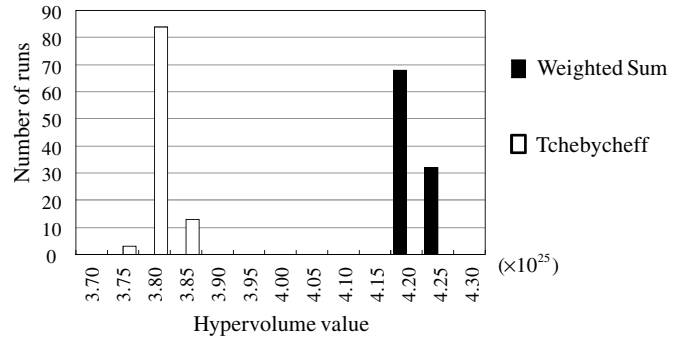
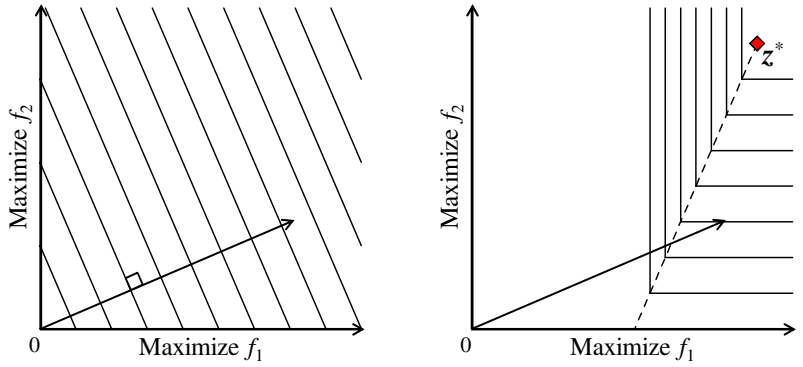


Fig. 5. Distribution of 100 values of the hypervolume measure (6-500 problem)

The deterioration in the performance of the weighted Tchebycheff by the increase in the number of objectives can be explained by the shape of contour lines of each scalarizing function. In the case of the weighted sum, a contour line is a line, plane or hyper-plane (see Fig. 6 (a) where the weight vector is assumed to be (0.7, 0.3)). The objective space is divided into two subspaces by a contour line. One subspace is a better region than the contour line while the other is a worse region. If a current solution is on the contour line, it is replaced with a newly generated offspring only when the offspring is in the better region. Roughly speaking, the size of the better region can be viewed as being 1/2 of the objective space independent of the number of objectives. On the other hand, the size of the better region is $(1/2)^m$ of the m-dimensional objective space in the case of the weighted Tchebycheff (see Fig. 6 (b)). This means that the probability of the replacement of a current individual with a newly generated offspring exponentially decreases with the number of objectives. As a result, the search ability of MOEA/D with the weighted Tchebycheff function was deteriorated by the increase in the number of objectives in Figs. 3-5.



(a) The weighted sum. (b) The weighted Tchebycheff.

Fig. 6. Contour lines of each scalarizing function

5 Test Problems with Non-convex Pareto Fronts

As shown in the previous section, better results are often obtained by the weighted sum especially for many-objective problems with convex Pareto fronts than the weighted Tchebycheff. The weighted sum, however, cannot appropriately handle multiobjective problems with non-convex Pareto fronts. In order to further examine the behavior of MOEA/D with each scalarizing function, we generated test problems with non-convex Pareto fronts from the three knapsack problems in the previous section. Our idea is to pull the convex Pareto front toward the inside of the feasible region so that its shape becomes non-convex.

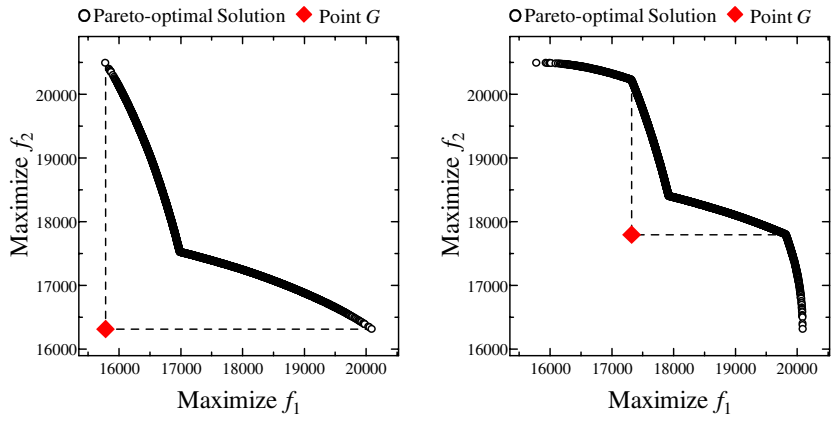
Let G be a point in the feasible region of the objective space (see Fig. 7). We denote the location of the point G in the objective space by $(g_1, g_2, ..., g_m)$. Using this point, the value of each objective is modified for each individual $\boldsymbol{x}$ as

$$f_i(\boldsymbol{x}) := f_i(\boldsymbol{x}) - \min\{0, \min_{j=1,2,...,m}\{\alpha \cdot (f_j(\boldsymbol{x}) - g_j)\}\}, \ i = 1, 2, ..., m, \tag{8}$$

where α is a user-definable positive parameter. By the modification in (8), all individuals dominating the point G are pulled toward the inside of the feasible region (i.e., toward the point G). Other individuals are not modified. The parameter α specifies the amount of the modification. In Fig. 7, the value of α is specified as $\alpha = 0.6$.

One important issue in (8) is the specification of the point G. We generated two types of test problems: Type C and Type D. In Type C, the point G is specified so that G is completely dominated by the Pareto front (see Fig. 7 (a)). As a result, the entire Pareto front becomes non-convex. On the other hand, the point G is specified so that G is dominated by a half of the Pareto front (see Fig. 7 (b)) in Type D. As a result, only the center region of the Pareto front becomes non-convex.

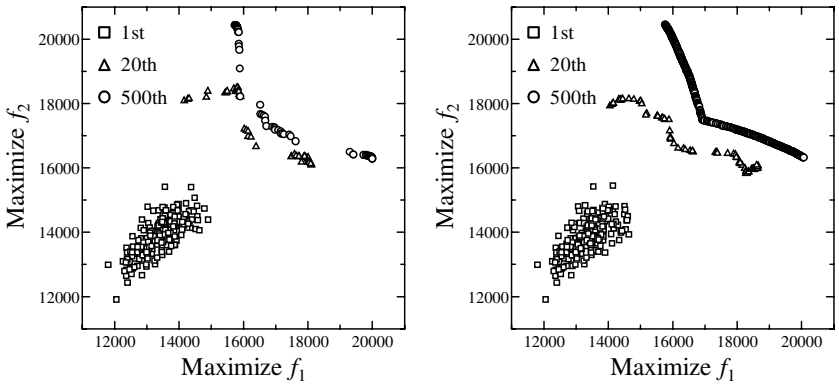
When the Pareto front of an original multiobjective problem is unknown, we can roughly estimate its range by applying an EMO algorithm to the problem and/or independently optimizing each objective. Then we can specify the point G based on the estimation of the range of the Pareto front. Due to the page limitation, we only report experimental results on the two test problems in Fig. 7 while we generated other test problems with non-convex Pareto fronts from the 4-500 and 6-500 problems.



(a) A newly generated test problem 2-500C. (b) A newly generated test problem 2-500D.

Fig. 7. Two test problems generated from the 2-500 knapsack problem (α=0.6)

In the same manner as in the previous section, we applied MOEA/D to the 2-500C problem in Fig. 7 (a). Experimental results are shown in Fig. 8. Fig. 8 (a) clearly demonstrates that the weighted sum cannot appropriately handle non-convex Pareto



(a) The weighted sum. (b) The weighted Tchebycheff.

Fig. 8. Experimental results of MOEA/D on the 2-500C test problem

fronts. All individuals in the main population converged to the two edges of the Pareto front whereas some other solutions were stored in the secondary population.

6 Our Idea for Automatically Choosing a Scalarizing Function

Our experimental results on many-objective problems with convex Pareto fronts in Fig. 4 and Fig. 5 clearly demonstrated the high search ability of MOEA/D with the weighted sum. On the other hand, experimental results in Fig. 8 on the two-objective problem with the non-convex Pareto front showed the advantage of the weighted Tchebycheff. These observations motivated us to devise a mechanism for automatically choosing between the weighted sum and the weighted Tchebycheff. Our idea is to use the weighted Tchebycheff only for individuals along non-convex regions of the Pareto front. In order to implement this idea, we have to detect non-convex regions of the Pareto front during the execution of MOEA/D.

Since the weighted sum cannot find any Pareto-optimal solutions along non-convex regions of the Pareto front, different weight vectors may have the same individual. In other words, a single individual can be the best solution with respect to the weighted sum with different weight vectors near a non-convex region of the Pareto front. Based on these discussions, we propose the following mechanism for automatically choosing between the weighted sum and the weighted Tchebycheff.

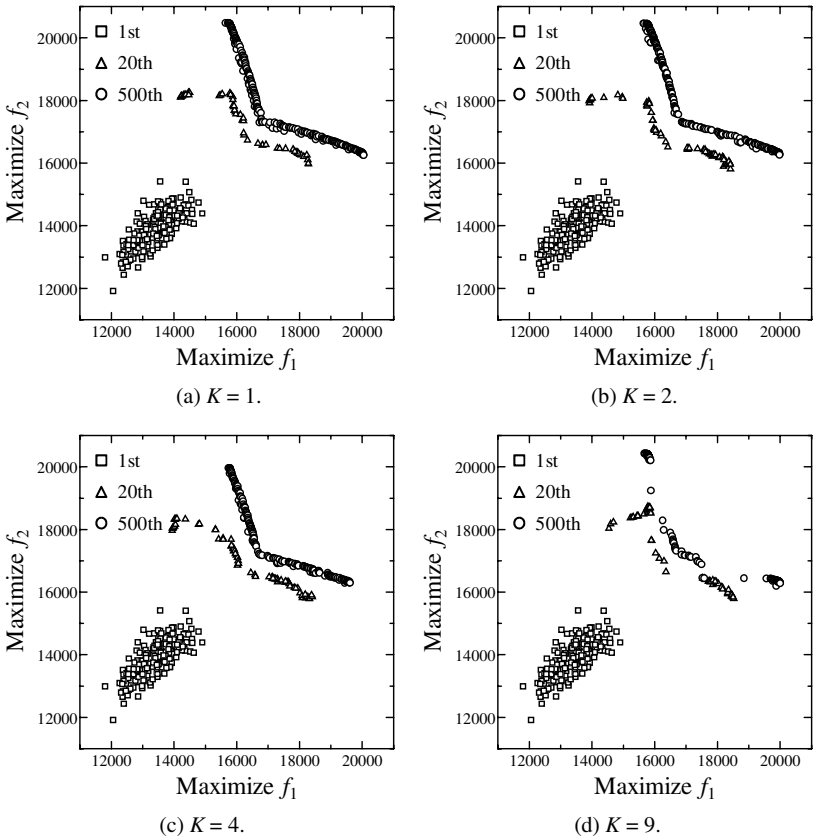
Proposed Idea: If a current individual $\boldsymbol{x}^k$ and at least K neighbors in its neighborhood $B(\boldsymbol{x}^k)$ have the same objective vector, we use the weighted Tchebycheff together with the weight vector $\boldsymbol{\lambda}^k$. Otherwise we use the weighted sum together with $\boldsymbol{\lambda}^k$.

In the proposed idea, K is a user-definable integer. In an extreme case of $K = 0$, the weighted Tchebycheff is always used. In the other extreme case with an infinitely large value of K, the weighted sum is always used.

7 Effectiveness of Our Idea

We applied MOEA/D with the proposed idea to the test problems in Fig. 7. We used four values 1, 2, 4, 9 as the threshold parameter K. In Fig. 9 and Fig. 10, we show experimental results on the 2-500C problem in Fig. 7 (a) and the 2-500D problem in Fig. 7 (b), respectively. We can see from Fig. 9 and Fig. 10 that the proposed idea works well for finding Pareto-optimal or near Pareto-optimal solutions along the non-convex regions of the Pareto fronts. The effect of the specification of the threshold parameter K is also clearly demonstrated in Fig. 9 and Fig. 10.

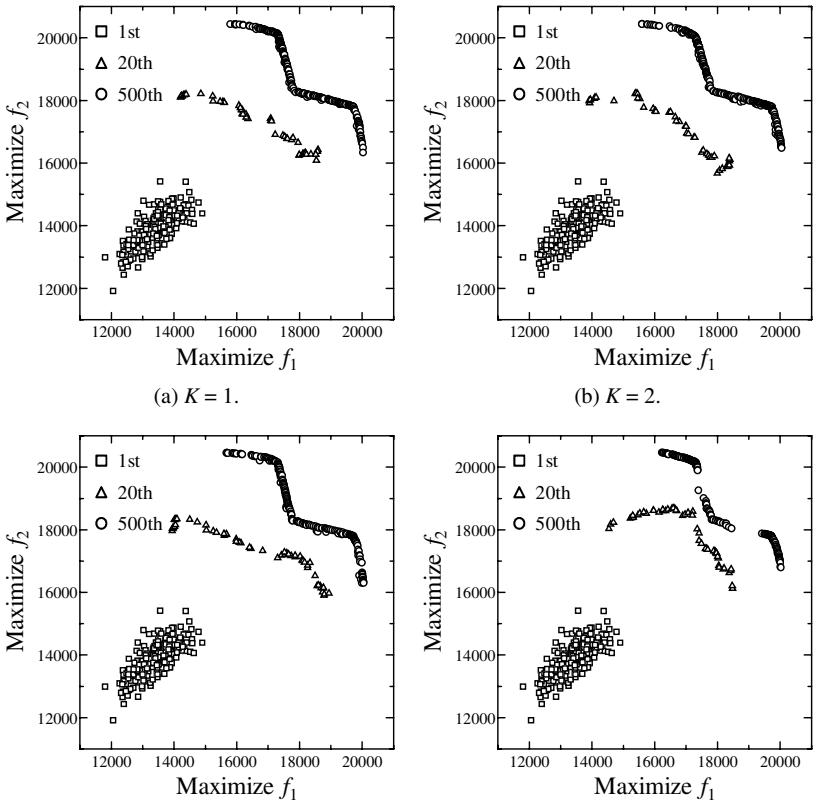
We also examined the performance of the proposed idea for the 2-500, 4-500, 6-500 and their C versions using the hypervolume measure as in Fig. 5. The weighted sum and the weighted Tchebycheff worked well for many objectives and non-convex Pareto fronts, respectively. Intermediate results between these two scalarizing functions were almost always obtained from the proposed idea.



(a) $K = 1$. (b) $K = 2$.

(c) $K = 4$. (d) $K = 9$.

Fig. 9. Experimental results of MOEA/D with the proposed idea on the 2-500C test problem



(a) $K = 1$. (b) $K = 2$.

(c) $K = 4$. (d) $K = 9$.

Fig. 10. Experimental results of MOEA/D with the proposed idea on the 2-500D test problem

8 Conclusions

We proposed an idea of automatically choosing between the weighted sum and the weighted Tchebycheff in a scalarizing function-based EMO algorithm MOEA/D [25]. To explain the motivation behind the proposal, we demonstrated that the search ability of MOEA/D with the weighted Tchebycheff is severely deteriorated by the increase in the number of objectives. We also visually demonstrated that the weighted sum cannot appropriately handle non-convex Pareto fronts. Then we explained how we can decide whether the weighted Tchebycheff is needed or not. This decision is based on the detection of non-convex regions of the Pareto front. In the proposed idea, the use of the weighted Tchebycheff is recommended when a single individual has the best weighted sum value among its neighbors for multiple weight vectors. The effectiveness of this idea was demonstrated through computational experiments.

We presented our idea in a very simple form. So it has a number of issues to be further discussed in future studies. One issue is the location of the reference point z^*

for the weighted Tchebycheff (we specified it by (5) in this paper). When only a part of the Pareto front is non-convex as in the 2-500D problem in Fig. 7 (b), it may be a good idea to locate the reference point z^* close to the non-convex region in order to concentrate the multiobjective search by the weighted Tchebycheff on the non-convex region. Another issue is the frequency of the change between the weighted sum and the weighted Tchebycheff. It seems that too frequent changes may have a negative effect on the search of MOEA/D. One critical issue is the possibility that the two scalarizing functions drive the current individual to totally different areas in the objective space even when they have the same weight vector. In this case, frequent changes may severely deteriorate the search ability of MOEA/D with our idea.

One possible remedy for such a negative effect of changing between the two scalarizing functions is to calibrate them so that they drive the current individual to the same area when they have the same weight vector. Another possible remedy is to use both the weighted sum and the weighted Tchebycheff as an integrated function. Actually we examined the use of the augmented weighted Tchebycheff in MOEA/D in our preliminary computational experiments. We observed high sensitivity of the search ability of MOEA/D with the augmented weighted Tchebycheff to the specification of the weight value for the augmented term. Different problems needed different specifications of the weight value. The point is how to adjust the weight for each problem. The proposed idea for detecting non-convex regions can be used in such an adjustment mechanism. The proposed idea can be also used for automatically specifying an appropriate objective function for each solution during the execution of local search in hybrid EMO algorithms (i.e., multiobjective memetic algorithms: MOMAs).

This work was partially supported by Grant-in-Aid for Scientific Research for Scientific Research (B) (20300084).

References

1. Branke, J., Schmeck, H., Deb, K., Reddy, M.: Parallelizing Multi-Objective Evolutionary Algorithms: Cone Separation. In: Proc. of 2004 Congress on Evolutionary Computation, pp. 1952–1957 (2004)
2. Chang, P.C., Chen, S.H., Zhang, Q., Lin, J.L.: MOEA/D for Flowshop Scheduling Problems. In: Proc. of 2008 Congress on Evolutionary Computation, pp. 1433–1438 (2008)
3. Coello, C.A.C.: Evolutionary Multi-Objective Optimization: A Historical View of the Field. IEEE Computational Intelligence Magazine 1, 28–36 (2006)
4. Deb, K.: Multi-Objective Optimization Using Evolutionary Algorithms. John Wiley & Sons, Chichester (2001)
5. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II. IEEE Trans. on Evolutionary Computation 6, 182–197 (2002)
6. Gandibleux, X., Ehrgott, M.: 1984-2004 – 20 Years of Multiobjective Metaheuristics. But What About the Solution of Combinatorial Problems with Multiple Objectives? In: Coello Coello, C.A., Hernández Aguirre, A., Zitzler, E. (eds.) EMO 2005. LNCS, vol. 3410, pp. 33–46. Springer, Heidelberg (2005)
7. Hughes, E.J.: Multiple Single Objective Sampling. In: Proc. of 2003 Congress on Evolutionary Computation, pp. 2678–2684 (2003)
8. Hughes, E.J.: Evolutionary Many-Objective Optimization: Many Once or One Many? In: Proc. of 2005 Congress on Evolutionary Computation, pp. 222–227 (2005)

9. Hughes, E.J.: MSOPS-II: A General-Purpose Many-Objective Optimiser. In: Proc. of 2007 Congress on Evolutionary Computation, pp. 3944–3951 (2007)
10. Ishibuchi, H., Doi, T., Nojima, Y.: Incorporation of Scalarizing Fitness Functions into Evolutionary Multiobjective Optimization Algorithms. In: Runarsson, T.P., Beyer, H.-G., Burke, E.K., Merelo-Guervós, J.J., Whitley, L.D., Yao, X. (eds.) PPSN 2006. LNCS, vol. 4193, pp. 493–502. Springer, Heidelberg (2006)
11. 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, 392–403 (1998)
12. Ishibuchi, H., Tsukamoto, N., Hitotsuyanagi, Y., Nojima, Y.: Effectiveness of Scalability Improvement Attempts on the Performance of NSGA-II for Many-Objective Problems. In: Proc. of 2008 Genetic and Evolutionary Computation Conference, pp. 649–656 (2008)
13. Ishibuchi, H., Tsukamoto, N., Nojima, Y.: Evolutionary Many-Objective Optimization: A Short Review. In: Proc. of 2008 Congress on Evolutionary Computation, pp. 2424–2431 (2008)
14. 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, 204–223 (2003)
15. Jaszkiewicz, A.: Genetic Local Search for Multi-Objective Combinatorial Optimization. European Journal of Operational Research 137, 50–71 (2002)
16. 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, 402–412 (2002)
17. Jaszkiewicz, A.: On the Computational Efficiency of Multiple Objective Metaheuristics: The Knapsack Problem Case Study. European Journal of Operational Research 158, 418–433 (2004)
18. Khara, V., Yao, X., Deb, K.: Performance Scaling of Multi-objective Evolutionary Algorithms. In: Fonseca, C.M., Fleming, P.J., Zitzler, E., Deb, K., Thiele, L. (eds.) EMO 2003. LNCS, vol. 2632, pp. 376–390. Springer, Heidelberg (2003)
19. Li, H., Zhang, Q.: Multiobjective Optimization Problems with Complicated Pareto Sets, MOEA/D and NSGA-II. IEEE Trans. on Evolutionary Computation (in press); available from IEEE Xplore
20. Miettinen, K.: Nonlinear Multiobjective Optimization. Kluwer Academic Publishers, Boston (1998)
21. Murata, T., Ishibuchi, H., Gen, M.: Specification of Genetic Search Directions in Cellular Multi-objective Genetic Algorithms. In: Zitzler, E., Deb, K., Thiele, L., Coello Coello, C.A., Corne, D.W. (eds.) EMO 2001. LNCS, vol. 1993, pp. 82–95. Springer, Heidelberg (2001)
22. Purshouse, R.C., Fleming, P.J.: Evolutionary Many-Objective Optimization: An Exploratory Analysis. In: Proc. of 2003 Congress on Evolutionary Computation, pp. 2066–2073 (2003)
23. Sato, H., Aguirre, H.E., Tanaka, K.: Local Dominance and Local Recombination in MOEAs on 0/1 Multiobjective Knapsack Problems. European Journal of Operational Research 181, 1708–1723 (2007)
24. Sindhya, K., Deb, K., Miettinen, K.: A Local Search Based Evolutionary Multi-Objective Optimization Approach for Fast and Accurate Convergence. In: Rudolph, G., Jansen, T., Lucas, S., Poloni, C., Beume, N. (eds.) PPSN 2008. LNCS, vol. 5199, pp. 815–824. Springer, Heidelberg (2008)
25. Zhang, Q., Li, H.: MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition. IEEE Trans. on Evolutionary Computation 11, 712–731 (2007)
26. Zitzler, E., Thiele, L.: Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach. IEEE Trans. on Evolutionary Computation 3, 257–271 (1999)