MOEA/D Using Constant-Distance Based Neighbors Designed for Many-Objective Optimization

Hiroyuki Sato
Faculty of Informatics and Engineering,
The University of Electro-Communications
1-5-1 Chofugaoka, Chofu, Tokyo, JAPAN, 182-8585
Email: sato@hc.uec.ac.jp

***Abstract*—Several recent studies showed the effectiveness of MOEA/D for many-objective optimization. However, MOEA/D was originally proposed not for many-objective optimization but for multi-objective optimization. Therefore, its algorithm causes several problems in many-objective optimization. MOEA/D uses the neighbor mating, and neighbors are determined by the user-defined neighbors' size T. For each weight vector which determines a search direction in the objective space, MOEA/D calculates distances to all weights and find T nearest weights as its neighbors. However, the number of weights having the same distance is increased as the number of objectives is increased, and MOEA/D faces the difficulty to determine neighbors by the neighbors' size T in many-objective optimization. Also, especially for the extreme weights to search the extreme objective function values, weights far from them are included as their neighbors. It causes a negative effect on the search of the extreme objective values. To overcome these problems and enhance the search performance of MOEA/D by improving its algorithm appropriately for many-objective optimization, in this work we focus on the handling of neighbors and propose an improved MOEA/D including the constant-distance based neighbors and the tournament selection based on the scalarizing function values. We use many-objective knapsack problems with 2-8 objectives and compare the search performances of the conventional MOEA/D, the improved MOEA/D and NSGA-III. As the results, we show that the improved MOEA/D achieves higher search performance than the conventional MOEA/D and NSGA-III by improving the diversity of the obtained solutions in the objective space.**

I. Introduction

Evolutionary algorithms are particularly suited to solve multi-objective optimization problems (MOPs) because they can find a set of Pareto optimal solutions (POS) approximating the optimal trade-off among conflicting objective functions from the population in a single run of the algorithm [1]. NSGA-II [2] and SPEA2 [3] have been known as representative MOEAs. In these MOEAs, the selection incorporates elitism and it is biased by Pareto dominance ranking and a diversity preserving strategy in the objective space. Pareto dominance based selection has been successfully applied especially in MOPs which optimize two or three objectives. However, the ranking by Pareto dominance loses its effectiveness as the number of objectives is increased, severely deteriorating the search performance of MOEAs for many objectives optimization [4], [5]. Therefore, there is a growing interest in applying MOEAs to solve many-objective optimization problems (MaOPs) which optimize four or more objective functions simultaneously. To overcome this problem of MOEAs in MaOPs, several approaches have been studied [6]. On the basis of conventional MOEAs effective for problems with a small number of objectives, one approach aims to reduce the dimensionality of the objective space, so that conventional MOEAs known to perform well in low dimensional objective space can be utilized [7]–[9]. As another approach, user preferences are also being investigated, aiming to focus the solution search on a specific region of Pareto front [10], [11]. Contrary to these approaches, to approximate the entire Pareto front considering all objective functions in MaOPs, algorithms to improve the selection pressure and assign more fine-grained rank to solutions have been designed. They can be categorized into MOEAs based on an extension of Pareto dominance relation [5], [12], MOEAs based on an indicator [13], [14], and MOEAs based on the decomposition of the objective space [4], [15]–[17]. As one of the promising approaches for many-objective optimization, in this work we focus on the last decomposition approach and MOEA/D [16] as a representative MOEA employing this approach.

MOEA/D decomposes a MOP into a number of single-objective optimization problems [16]. Each single-objective optimization problem is defined by a scalarizing function using a weight vector and is used to search for a search direction in the multi-dimensional objective space. MOEA/D assigns one solution with each scalar optimization and simultaneously optimizes them to approximate the Pareto front by applying genetic operations to neighboring solutions. MOEA/D uses a scalar value aggregated from multiple objective function values instead of Pareto dominance as the criterion to compare solutions. Therefore, MOEA/D can easily determine the superiority or inferiority of solutions even in MaOPs. Recently, NSGA-III, an improved version of NSGA-II for solving MaOPs, has also introduced the concept of decomposing the objective space [17]. Actually, several recent studies showed the effectiveness of MOEA/D in MaOPs [18], [19]. However, MOEA/D was originally proposed not for MaOPs but for MOPs, and its search performance was verified on MOPs with 2-4 objectives in the original paper of MOEA/D [16]. Therefore, its algorithm causes several problems in many-objective optimization. Since MOEA/D is one of promising algorithm frameworks for solving MaOPs, in this work we try to improve its algorithm appropriate for many-objective optimization.

MOEA/D uses the neighbor mating which selects parents from neighbors in the objective space. It has been known that the neighbor mating is effective to improve the search performance of MOEAs in several MOPs [16], [20], [21]. The

978-1-4799-7492-4/15/$31.00

conventional MOEA/D uses the user-defined neighbors' size T to determine neighbors. For each weight vector, MOEA/D calculates Euclidean distances to all weight vectors and find T nearest weights as neighbors. However, the number of weights having the same distance is increased as the number of objectives (dimensions) is increased, and MOEA/D faces the difficulty to determine neighbors by the neighbors' size T in MaOPs. Also, especially for the extreme weight vectors to search the extreme objective values, weight vectors far from them are included as their neighbors. It causes a negative effect on the search of the extreme objective function values.

To overcome these problems and enhance the search performance of MOEA/D by improving its algorithm appropriate for many-objective optimization, in this work we focus on the handling of neighbors and propose an improved MOEA/D including the constant-distance based neighbors and the tournament selection based on the scalarizing function values. For each weight vector, the improved MOEA/D determines neighbors by the user-defined constant-distance δ in the weight space instead of neighbors' size T. The proposed handling of neighbors solves the aforementioned problems and equally treats weight vectors having the same distance in MaOPs. Also, for the extreme weight vectors, since the proposed handling of neighbors selects only weight vectors close to them as their neighbors, we can expect to encourage the solution search of the extreme objective values and improve the spread of solutions in the objective space. Next, to select parents from neighbors, we propose the tournament selection based on the scalarizing function values while the conventional MOEA/D randomly selects parents from neighbors. This method enhances the solution search of each scalarizing function by selecting appropriate parents from neighbors. To verify the effectiveness of the proposed methods, we compare the search performances of the conventional MOEA/D, the improved MOEA/D and NSGA-III [17]. This work focuses on combinatorial many-objective optimization and uses many-objective knapsack problems [3] with 2-8 objectives.

II. MOEA/D

A. Weight Vectors and Scalarizing Functions

MOEA/D decomposes a MOP into a number of single-objective optimization problems. The single-objective optimization problems are formulated by scalarizing functions g using uniformly distributed weight vectors $\boldsymbol{\lambda}^i$ $(i = 1, 2, \ldots, N)$. Each element λ^i_j $(j = 1, 2, \ldots, m)$ is one of $\{0/H, 1/H, \ldots, H/H\}$ based on the decomposition parameter H, and C^{m-1}_{H+m-1} $(= N)$ kinds of weight vectors satisfying $\sum_{j=1}^{m} \lambda^i_j = 1.0$ are used for solution search. Each weight vector $\boldsymbol{\lambda}^i$ determines an unique search direction in the m-dimensional objective space, and MOEA/D uses a set of uniformly distributed weights to approximate the Pareto front. A similar idea was also proposed by Murata et al [22].

MOEA/D has several options of scalarizing functions g. This work uses the weighted Tchebycheff (TCH), the weighted sum (WS), the penalty-based boundary intersection (PBI) [16] and the inverted PBI (IPBI) [19] scalarizing functions. Each scalarizing function has its own characteristics, and it makes an impact on the search performance. A recent study reported that MOEA/D using IPBI scalarizing function achieved higher search performance than MOEA/Ds with other scalarizing functions and NSGA-III [17] in MaOPs.

Algorithm 1 The conventional size-based neighbors

Input: T : *The neighbors' size to determine neighbors*
Output: $\mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_N$: *Neighbor indices for each weight index*

```
for all i ∈ {1, 2, ..., N} do
    Id = {id_1, id_2, ..., id_N} = {1, 2, ..., N} : Weight Indices
    D = {d_1, d_2, ..., d_N} : Distances from λ^i to all weights
    B_i = {i_1, i_2, ..., i_T} : Neighbors' indices for weight index i
    for all j ∈ {1, 2, ..., N} do
        d_j = Calculate Euclidean distance(λ^i, λ^j)
    end for
    Sort indices Id in ascending order of distances D (Id, D)
    for all j ∈ {1, 2, ..., T} do
        i_j = id_j
    end for
end for
return B_1, B_2, ..., B_N
```

B. Algorithm

In the following, the algorithm of MOEA/D is described.

Step 1) Initialization:

Step 1-1) Calculate the Euclidean distances between any two weight vectors and find the T nearest weight vectors to each weight vector. For each $i \in \{1, 2, \ldots, N\}$, set $\mathcal{B}_i = \{i_1, i_2, \ldots, i_T\}$, where $\boldsymbol{\lambda}^{i_1}, \boldsymbol{\lambda}^{i_2}, \ldots, \boldsymbol{\lambda}^{i_T}$ are the T nearest weight vectors to $\boldsymbol{\lambda}^i$. (See **Algorithm 1**)

Step 1-2) Randomly generate the population $\boldsymbol{x}^1, \boldsymbol{x}^2, \ldots, \boldsymbol{x}^N$.

Step 2) Solution Search:

For each $i \in \{1, 2, \ldots, N\}$, perform the following procedure.

Step 2-1) Randomly choose two indexes k and l from $\mathcal{B}_i$, and then generate a new offspring $\boldsymbol{y}$ from parents $\boldsymbol{x}^k$ and $\boldsymbol{x}^l$ by applying genetic operators.

Step 2-2) For each index $j \in \mathcal{B}_i$, if $g(\boldsymbol{y}|\boldsymbol{\lambda}^j)$ is better than $g(\boldsymbol{x}^j|\boldsymbol{\lambda}^j)$, then the current solution $\boldsymbol{x}^j$ is replaced by the generated offspring $\boldsymbol{y}$, i.e. $\boldsymbol{x}^j = \boldsymbol{y}$.

Step 3) Stopping Criterion:

If the termination criterion is satisfied, then stop and pick the non-dominated solutions from the population $\boldsymbol{x}^1, \boldsymbol{x}^2, \ldots, \boldsymbol{x}^N$ as the output of the optimization. Otherwise, go to **Step 2**.

MOEA/D was originally proposed not for MaOPs but for MOPs, and its search performance was verified on MOPs with 2-4 objectives in the original paper of MOEA/D [16]. The decomposition approach of the objective space and the scalarizing approach of the objective function values to determine the superiority of solutions instead of Pareto dominance are effective for solving MaOPs. Actually, recent several studies showed that MOEA/D achieved high search performance in MaOPs [18], [19]. Currently, MOEA/D is one of promising algorithm frameworks for solving MaOPs. However, the algorithm of MOEA/D causes several problems in many-objective optimization.

C. Problem in the Handling of Neighbors for MaOPs

In this work, we focus on the handling of neighbors used in the conventional MOEA/D and discuss its two problems in the following.

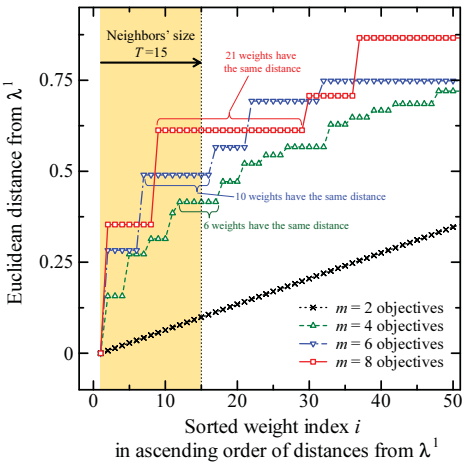


Fig. 1. Relation between distance from $\boldsymbol{\lambda}^1$ and sorted weight indices ($H = \{200, 9, 5, 4\}$ are used for $m = \{2, 4, 6, 8\}$ objectives, respectively)

The first problem is to face the difficulty to determine nearest neighbors by giving the user-defined neighbors' size T in MaOPs. As shown in **Step 1-1** of Section II-B, for each weight vector, the conventional MOEA/D finds T nearest neighbors in the weight space. The pseudocode of **Step 1-1** is shown in **Algorithm 1**. To find neighbors of weight $\boldsymbol{\lambda}^i$, first Euclidean distances from weight $\boldsymbol{\lambda}^i$ to all weights are calculated (lines 5-7). Next, the weight indices are sorted in ascending order of their distances (line 8), and the sorted weight indices from the first to T-th become neighbor indices $\mathcal{B}_i$ of weight $\boldsymbol{\lambda}^i$ (lines 9-11). For weight $\boldsymbol{\lambda}^1$, **Fig. 1** shows relations between sorted weight indices (horizontal axis) and their distances from $\boldsymbol{\lambda}^1$ (vertical axis) in $m = \{2, 4, 6, 8\}$ dimensional (objective) cases. In this figure, the decomposition parameters $H = \{200, 9, 5, 4\}$ are used for $m = \{2, 4, 6, 8\}$ objectives, respectively. In this example, we consider to find neighbors $\mathcal{B}_1$ of $\boldsymbol{\lambda}^1$ with neighbors' size $T = 15$. In $m = 2$ dimensional (objective) case, we can see that the distances from $\boldsymbol{\lambda}^1$ is monotonically increased. Therefore, there is no weight index having the same distance around $T = 15$. On the other hand, in $m = \{4, 6, 8\}$ dimensional (objective) cases, we can see that several indices have the same distance around $T = 15$. In MOEA/D, each dimension of m-dimensional space is decomposed by the equally-spaced interval $1/H$. Therefore, the number of weights having the same distance is increased as the dimension m is increased. If we determine neighbors by the neighbors' size T in high dimensional space, some of weight indices having the same distance are included in $\mathcal{B}_1$ and others are not included in $\mathcal{B}_1$ depending on the order of weight indices before sorting and the implementation of sorting algorithm. In MOEA/D, weight indices included in neighbors get the chance to update their solutions. Weight indices not included in neighbors lose the chance to update their solutions. Therefore, if neighbors are determined by neighbors' size as the conventional MOEA/D, the number of chances to update each solution becomes non-uniform especially in high dimensional objective space.

The second problem is neighbors of the extreme weights to search the extreme value of each objective function. As an example, **Fig. 2** shows weight vectors in $m = 2$ dimensional weight space. **Fig. 2 (a)** shows neighbors $\mathcal{B}_8$ of weight $\boldsymbol{\lambda}^8$ and neighbors $\mathcal{B}_1$ of extreme weight $\boldsymbol{\lambda}^1$ to search the extreme

Algorithm 2 The proposed constant-distance based neighbors

Input: δ : *The constant-distance to determine neighbors*
Output: $\mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_N$: *Neighbor indices for each weight index*

1: **for all** $i \in \{1, 2, \ldots, N\}$ **do**
2: $\mathcal{D} = \{d_1, d_2, \ldots, d_N\}$: *Distances from* $\boldsymbol{\lambda}^i$ *to all weights*
3: $\mathcal{B}_i = \emptyset$: *Neighbors' indices for weight index* i
4: **for all** $j \in \{1, 2, \ldots, N\}$ **do**
5: d_j = Calculate Euclidean distance($\boldsymbol{\lambda}^i$, $\boldsymbol{\lambda}^j$)
6: **end for**
7: **for all** $j \in \{1, 2, \ldots, N\}$ **do**
8: **if** $d^j \leq \delta$ **then**
9: $\mathcal{B}_i = \mathcal{B}_i \cup j$
10: **end if**
11: **end for**
12: **end for**
13: **return** $\mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_N$

objective value in the conventional MOEA/D with neighbors' size $T = 15$. From this figure, we can see that two neighbors $\mathcal{B}_1$ and $\mathcal{B}_8$ are the same although their weights $\boldsymbol{\lambda}^1$ and $\boldsymbol{\lambda}^8$ have different search directions. In MOEA/D, neighbors are the range to select parents. That is, their two weights share the same solutions as information resource to search each of their different search directions. Especially for the extreme weight $\boldsymbol{\lambda}^1$ to search the extreme objective value, since solutions with weights far from $\boldsymbol{\lambda}^1$ have the chance to become parents, it may be a negative effect to search the extreme objective function value. As a result, the distribution of the obtained solutions is biased toward the central area of Pareto front, and we cannot expect to obtain widely spread solutions. Note that the second problem is not only for MaOPs but also for MOPs.

The above two problems in the conventional MOEA/D are caused by the mechanism to determine the neighbors of each weight by the user-defined neighbors' size T .

III. Improved MOEA/D using the Constant-Distance Based Neighbors

A. Concept

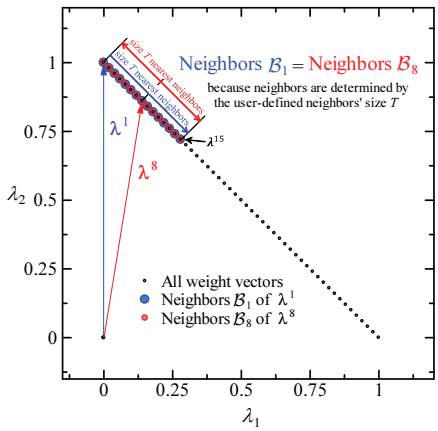
To overcome the aforementioned problems in the conventional MOEA/D and to enhance the search performance by improving its algorithm appropriate for many-objective optimization, in this work we propose an improved MOEA/D introducing the constant-distance based neighbors and the tournament selection using the scalarizing function values.

B. Constant-Distance Based Neighbors

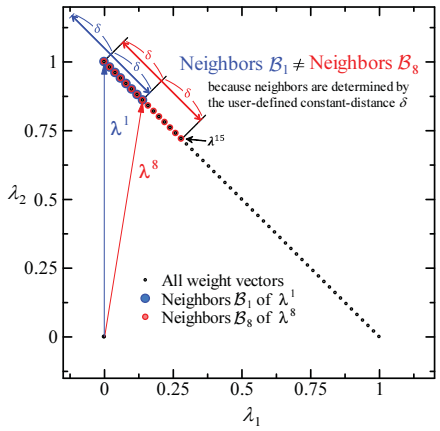
For each weight, the conventional MOEA/D assigns T nearest weights as its neighbors. In this case, neighbors' sizes of all weights are equivalent ($|\mathcal{B}_1| = |\mathcal{B}_2| = \ldots = |\mathcal{B}_N| = T$). On the other hand, the proposed method determines neighbors by a constant-distance δ. For each weight, its neighbors are all weights having distances equal to or shorter than δ. In this case, neighbors' sizes of all weights are not equivalent. δ is given by

$$\delta = \sqrt{2} \cdot \frac{h}{H}, \tag{1}$$

where, h is the user-defined parameter to control the distance δ, and its range is $h = [0, H]$. In the case of $h = H$, δ becomes the maximum, and all weights become neighbors for



(a) The size T based neighbors in the conventional MOEA/D [16]



(b) The constant-distance δ based neighbors in the improved MOEA/D

Fig. 2. All weight vectors, neighbors $\mathcal{B}_8$ of weight $\boldsymbol{\lambda}^8$, and neighbors $\mathcal{B}_1$ of weight $\boldsymbol{\lambda}^1$ to search the extreme objective value in $m = 2$ dimensional weight space (the decomposition parameter is set to $H = 50$)

all weights. δ is shortened by decreasing h, and the range of neighbors is shrunk. In the case of $h = 0$, δ becomes the minimum, the neighbor of $\boldsymbol{\lambda}^i$ is only itself ($\mathcal{B}_i = \{i\}$). In this special case, new offspring are generated by applying genetic operators to one solution. **Algorithm 2** shows the pseudocode to find neighbors based on the constant-distance.

For the first problem described in Section II-C, the proposed method equally handles weights having the same distance. The proposed method does not restrict the neighbor's size, and any weights having distances equal to or shorter than δ are included as neighbors. Therefore, the chances to select as parents and update solutions are equally assigned to each weight depending on δ. For the second problem described in Section II-C, as shown in **Fig. 2 (b)**, especially for the extreme weights to search the extreme objective values such as $\boldsymbol{\lambda}^1$, the proposed method can restrict weights in its neighbors. In this way, to search the extreme objective value directed by $\boldsymbol{\lambda}^1$, only solutions closer to $\boldsymbol{\lambda}^1$ become parents and provide information resources to search for the direction. Therefore, we can expect to encourage the solution search of extreme objective values.

Additionally, in the proposed method, we control the number of matings for each weight. In MOEA/D, weight vectors frequently included in the neighbors $\mathcal{B}_i$ $(i = 1, 2, \ldots, N)$ have many chances to update their solutions. Since extreme weight vectors such as $\boldsymbol{\lambda}^1$ shown in **Fig. 2** are located in the extreme areas of the weight space, the total number of the extreme weights included in $\mathcal{B}_i$ $(i = 1, 2, \ldots, N)$ are less than other weights, and the chances to update their solutions are also less than other weights. It has a negative effect to search the extreme objective values. To reduce the bias of the chances to update solutions for each weight, we introduce the chance values $\mathcal{C} = \{c_1, c_2, \ldots, c_N\}$ and the accumulated chance values $\mathcal{AC} = \{ac_1, ac_2, \ldots, ac_N\}$. c_i $(i = 1, 2, \ldots, N)$ is the total number of weight $\boldsymbol{\lambda}^i$ included in all neighbors $\mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_N$. The pseudocode to calculate chance values $\mathcal{C}$ is shown **Algorithm 3**. ac_i $(i = 1, 2, \ldots, N)$ is the accumulated chance value of c_i. In the solution search, when weight index i is focused to select parents, c_i is added to ac_i, i.e. $ac_i = ac_i + c_i$. For every parents selection and offspring generation, the focused weight index i is determined as the index having the minimum accumulated chance value

Algorithm 3 Calculation of Chance values

Input: $\mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_N$: *Neighbor indices for each weight index*
Output: $\mathcal{C} = \{c_1, c_2, \ldots, c_N\}$: *Chance values for all indices*

```
C = {c_1, c_2, ..., c_N} = {0, 0, ..., 0}
for all i ∈ {1, 2, ..., N} do
    for all j ∈ B_i do
        c_j = c_j + 1
    end for
end for
```

by the following equation.

$$i = \underset{j \in \{1,2,\ldots,N\}}{\operatorname{argmin}} \; ac_j. \tag{2}$$

Then, we select two parents from neighbor $\mathcal{B}_i$ of the focused weight index i and generate offspring. In this way, the proposed method reduce the bias of the chances to update solutions of each weight by introducing the chance values $\mathcal{C}$ and the accumulated chance values $\mathcal{AC}$.

C. Two Parent Selections from Neighbors

In this work, we prepare two options to select parents from neighbors $\mathcal{B}_i$ of the focused weight index i.

Random Selection (RS):

In the same manner as the conventional MOEA/D [16], we randomly choose two parent indices k and l from neighbors $\mathcal{B}_i$ of the focused weight index i.

g-Tournament Selection (gTS):

To select more appropriate parents for the focused weight index i, we propose g-tournament selection using the scalarizing function value $g(\boldsymbol{\lambda}^i|\cdot)$. To select the first parent, we randomly choose two parent candidate indices pc_1 and pc_2 from $\mathcal{B}_i$. Then we compare scalarizing function values for the focused weight index i of two candidate solutions $\boldsymbol{x}^{pc_1}$ and $\boldsymbol{x}^{pc_2}$. If $g(\boldsymbol{x}^{pc_1}|\boldsymbol{\lambda}^i)$ is better than $g(\boldsymbol{x}^{pc_2}|\boldsymbol{\lambda}^i)$, pc_1 becomes the first parent index. Otherwise, pc_2 becomes the first parent index. The second parent is also selected by the same procedure. Compared with the random selection mentioned above, g-tournament selection can select more appropriate parents for the focused weight

index i. We can expect to encourage the solution search for each search direction by g-tournament selection.

D. Algorithm

In the following, the algorithm of the improved MOEA/D is described. The proposed mechanisms described in Section III-B is included in **Step 1-1**, **-3**, **-4**, **Step 2-1**. The mechanisms described in Section III-C is included in **Step 2-2**. In the following of this paper, the improved MOEA/D using the random selection is described as the improved MOEA/D-RS. The improved MOEA/D using g-tournament selection is described as the improved MOEA/D-gTS.

Step 1) Initialization:

Step 1-1) Calculate the Euclidean distances between any two weight vectors and find the neighbor weight vectors within the user-defined constant-distance δ to each weight vector. For each $i \in \{1, 2, \ldots, N\}$, set $\mathcal{B}_i = \{i_1, i_2, \ldots\}$, where $\boldsymbol{\lambda}^{i_1}, \boldsymbol{\lambda}^{i_2}, \ldots$ are neighbor weight vectors where their distances from $\boldsymbol{\lambda}^i$ are within the user-defined constant-distance δ. (See **Algorithm 2**)

Step 1-2) Randomly generate the population $\boldsymbol{x}^1, \boldsymbol{x}^2, \ldots, \boldsymbol{x}^N$.

Step 1-3) Calculate the chance values $\mathcal{C} = \{c_1, c_2, \ldots, c_N\}$ for all weight indices. c_i $(i = 1, 2, \ldots, N)$ is the total number of index i included in all neighbors $\mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_N$. (See **Algorithm 3**)

Step 1-4) Initialize the accumulated chance values by zeros, i.e. $\mathcal{AC} = \{ac_1, ac_2, \ldots, ac_N\} = \{0, 0, \ldots, 0\}$.

Step 2) Solution Search:

Perform the following procedure N times.

Step 2-1) Find the focused weight index i with the minimum accumulated chance value sc_i from all weight indices by **Eq. (2)**, and $ac_i = ac_i + c_i$.

Step 2-2) Select two indexes k and l from $\mathcal{B}_i$ by the random selection (RS) or g-tournament selection (gTS), and then generate a new offspring $\boldsymbol{y}$ from parents $\boldsymbol{x}^k$ and $\boldsymbol{x}^l$ by applying genetic operators.

Step 2-3) For each index $j \in \mathcal{B}_i$, if $g(\boldsymbol{y}|\boldsymbol{\lambda}^j)$ is better than $g(\boldsymbol{x}^j|\boldsymbol{\lambda}^j)$, then the current solution $\boldsymbol{x}^j$ is replaced by the generated offspring $\boldsymbol{y}$, i.e. $\boldsymbol{x}^j = \boldsymbol{y}$.

Step 3) Stopping Criterion:

If the termination criterion is satisfied, then stop and pick the non-dominated solutions from the population $\boldsymbol{x}^1, \boldsymbol{x}^2, \ldots, \boldsymbol{x}^N$ as the output of the optimization. Otherwise, go to **Step 2**.

IV. Experimental Setup

A. Algorithms

To verify effects of the proposed constant-distance based neighbors and g-tournament selection, in this work we compare the search performances of the conventional MOEA/D [16], the improved MOEA/D-RS, the improved MOEA/D-gTS and NSGA-III [17]. Each of the three MOEA/Ds are combined with TCH, WS, PBI and IPBI scalarizing functions, respectively. For PBI and IPBI scalarizing functions, their parameter θ [19] are set to 0.1.

NSGA-III is an improved version of the representative NSGA-II [2]. NSGA-III is designed especially for solving MaOPs and also employs the decomposition approach of the objective space. The main difference between NSGA-II and III is the method to maintain the distribution of solutions in the objective space. NSGA-III uses the perpendicular distances between solutions and reference lines based on a predefined set of reference points (weight vectors) instead of the crowding distance used in NSGA-II. For the normalization of each objective value, the original NSGA-III uses m-dimensional hyper-plane constituted by m-kinds of extreme vectors in the candidate population $\mathcal{S}$ to select the parent population $\mathcal{P}$. Since there is the case that the hyper-plane cannot be uniquely determined, in this paper, we use the obtained ideal objective vector $\boldsymbol{z}$ and simply normalize each objective value by $f_j^n(\boldsymbol{x}) = (z_j - f_j(\boldsymbol{x}))/(z_j - \max_{\boldsymbol{y} \in \mathcal{S}} f_j(\boldsymbol{y}))$ $(j = 1, 2, \ldots, m)$. Therefore, note that the search performance of NSGA-III shown in this paper is not exactly the same as the original one.

B. Test Problem

In this work, we focus on combinatorial optimization, and use the many-objective knapsack problem (the MOKP) [3] as the test problem. MOKP is formulated by

$$\begin{cases} \text{Maximize} & f_j(\boldsymbol{x}) = \sum_{i=1}^{n} x_i \cdot p_{i,j} \\ \text{subject to} & g_j(\boldsymbol{x}) = \sum_{i=1}^{n} x_i \cdot q_{i,j} \leq c_j \end{cases} (j = 1, \ldots, m), \quad (3)$$

where, $p_{i,j}$ and $q_{i,j}$ $(j = 1, 2, \cdots, m)$ denote profit and weight of item i according to knapsack (objective) j. c_j is the capacity of knapsack j. The capacity of knapsack is given by $c_j = \phi \cdot \sum_{i=1}^{n} q_{i,j}$ based on the feasibility ratio ϕ. In this work , we use MOKPs with $m = \{2, 4, 6, 8\}$ objectives, $n = 500$ items (bits), and feasibility ratio $\phi = 0.5$. To deal with infeasible solutions we use the greedy repair [3]. Similar to [3], $p_{i,j}$ and $q_{i,j}$ are generated as random integers in the interval $[10, 100]$.

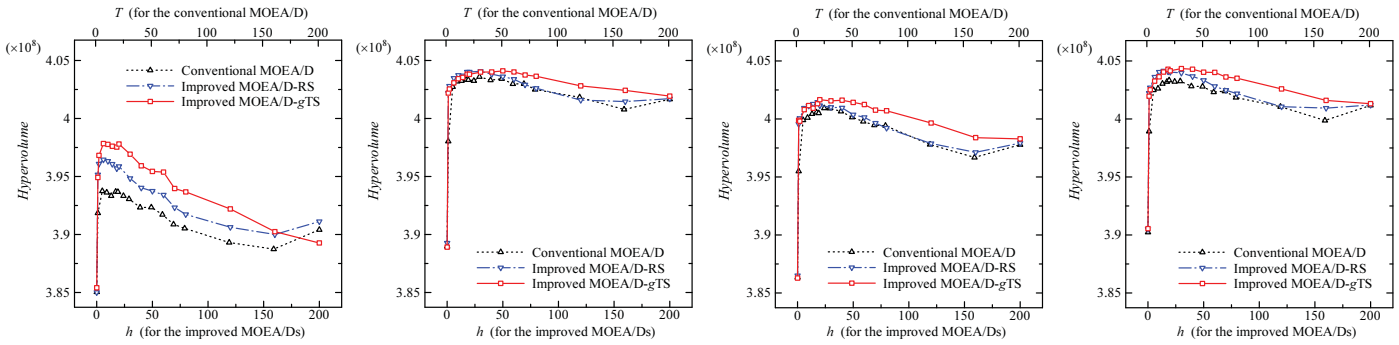
C. Metrics

The search performance is evaluated by Hypervolume (HV) [23]. HV is the m-dimensional volume of the region enclosed by the obtained non-dominated solutions and a dominated reference point $\boldsymbol{r}$ in the objective space. A High HV indicates a good non-dominated solutions in both the convergence toward Pareto front and the diversity to approximate a wide range of Pareto front.

To analyze contributing factors to the value of HV, in this work we independently evaluate the diversity and the convergence of the obtained non-dominated solutions in MOKPs. To measure the diversity of the obtained non-dominated solutions, we use Maximum Spread (MS) [23]. A high MS indicates a high diversity of the obtained non-dominated solutions in the objective space, i.e. a widely spread non-dominated solutions. Also, to measure the convergence of the obtained non-dominated solutions toward Pareto front, we use $Norm$ [24]. A High $Norm$ generally means a high convergence toward the true Pareto front.

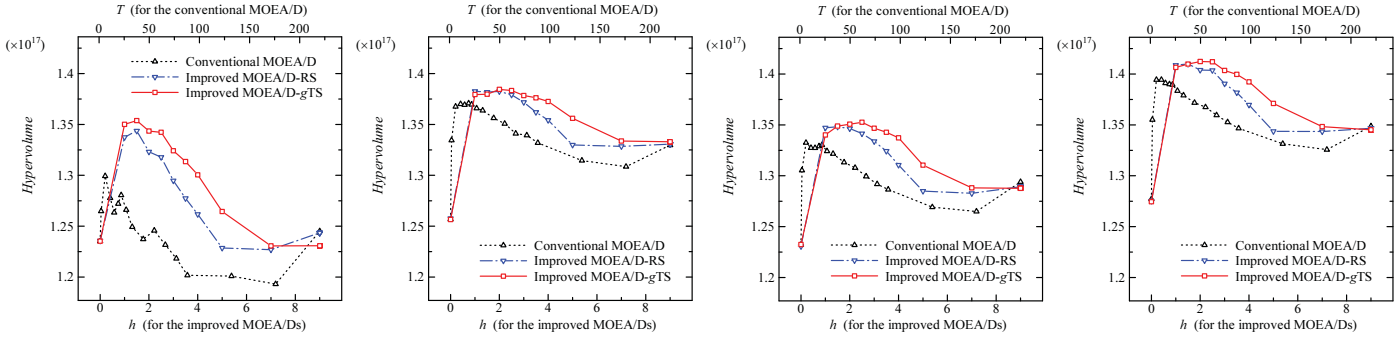
D. Parameters

We adopt CCG_{UX} crossover [21] which is an extension of uniform crossover. Generally, in uniform crossover, the exchange ratio of each gene is $\alpha = 0.5$. CCG_{UX} controls the exchange ratio α. The previous work [21] showed that CCG_{UX} with a very small α improved the search performance of MOEAs especially in MaOPs. On the basis of the results [21], this work uses $\alpha = 0.05$. Also, the crossover rate is set to 1.0. We also adopt bit-flipping mutation with a mutation rate



(a) Tchebycheff (TCH) (b) Weighted Sum (WS) (c) PBI (d) IPBI

Fig. 3. Results of HV in MOKP with $m = 2$ objectives when T for the conventional MOEA/D and h for the improved MOEA/Ds are varied, respectively



(a) Tchebycheff (TCH) (b) Weighted Sum (WS) (c) PBI (d) IPBI

Fig. 4. Results of HV in MOKP with $m = 4$ objectives when T for the conventional MOEA/D and h for the improved MOEA/Ds are varied, respectively

$4/n$. The termination criterion is set to 10^4 generations. To calculate HV, the reference point is set to $\boldsymbol{r} = \{0, 0, \ldots, 0\}$. The decomposition parameters to generate weight vectors in MOEA/D are set to $H = \{200, 9, 5, 4\}$ for $m = \{2, 4, 6, 8\}$ objectives, respectively. Therefore, the population sizes are $N = \{201, 220, 252, 330\}$ for $m = \{2, 4, 6, 8\}$, respectively. MOEA/Ds and NSGA-III use the same weight vectors and the population size. In the following experiments, we show the average performance of 30 runs.

V. Experimental Results and Discussion

A. HV by Varying Parameters to Handle Neighbors

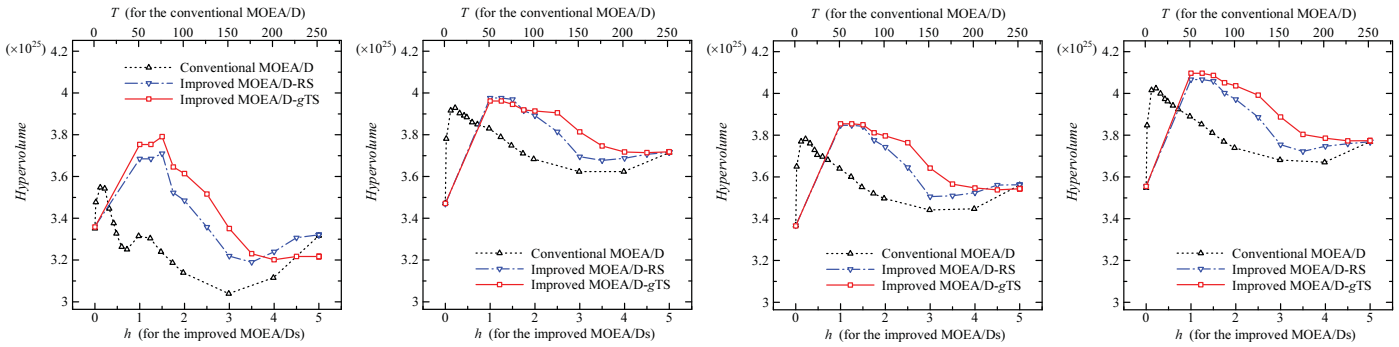
We compare the search performances of the conventional MOEA/D and the two improved MOEA/Ds when the neighbors' size T and the distance parameter h is varied, respectively. **Figs. 3-6** show results of HV at the final generations in MOKPs with $m = \{2, 4, 6, 8\}$ objectives. In each graph, the horizontal top axis indicates neighbors' size T for the conventional MOEA/D, and the horizontal bottom axis indicates the distance parameter h for the two improved MOEA/Ds. A small T and h indicate a small selection range of parents and update range of solutions. A large T and h indicate a large selection and update range. Each **(a)-(d)** show the results of MOEA/Ds combined with TCH, WS, PBI and IPBI scalarizing functions, respectively.

First, for both the conventional MOEA/D and the improved MOEA/Ds, we can see that HV is improved by decreasing T and h from their maximum values, respectively. However, HV is decreased by setting too small T and h. In all MOKPs and all algorithm combinations with scalarizing functions, there are the appropriate parameters T^* and h^* to maximize HV. As many researches showed before [16], [21], we can see that the neighbor mating is effective for solving MOKPs.

To verify the effectiveness of the proposed constant-distance based neighbors, next we compare the maximum values of HV obtained by the conventional MOEA/D and the improved MOEA/Ds. From the results, we can see that the maximum values of HV obtained by the two improved MOEA/Ds are higher than the one obtained by the conventional MOEA/D in all MOKPs and all algorithm combinations with four scalarizing functions. These results reveal that the search performances of MOEA/Ds with any scalarizing functions are improved by employing the proposed constant-distance based neighbors.

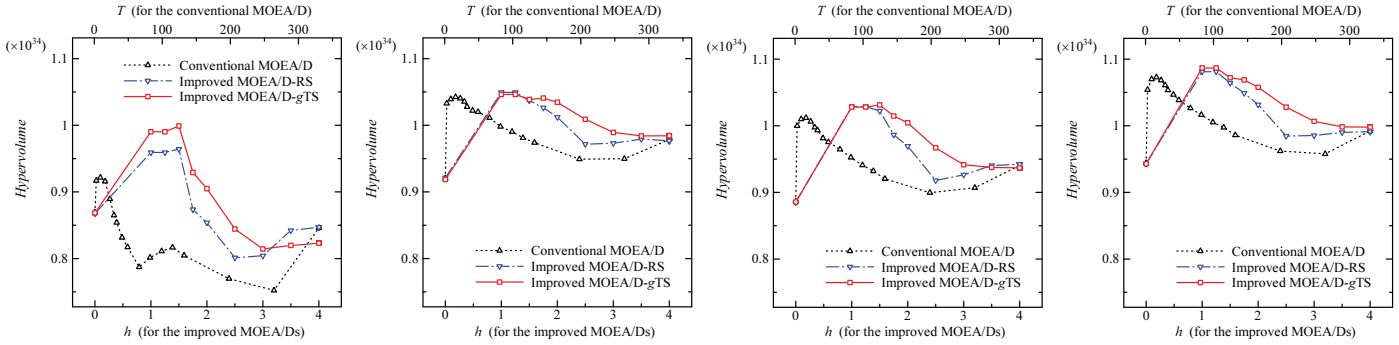
To verify the effectiveness of the proposed g-tournament selection, next we compare the maximum values of HV obtained by the improved MOEA/D-RS and the improved MOEA/D-gTS. When we combine with TCH, PBI and IPBI scalarizing functions, the improved MOEA/D-gTS achieves higher HV than the improved MOEA/D-RS. On the other hand, when we combine with WS scalarizing function, the improved MOEA/D-RS achieves higher HV than the improved MOEA/D-gTS. These results reveal that the search performance of the improved MOEA/D is further improved when the algorithm is combined with TCH, PBI and IPBI scalarizing functions.

In the two improved MOEA/Ds, when h is increased from h^* maximizing HV, we can see the decrease of HV obtained by the improved MOEA/D-gTS is more gradual than the one obtained by the improved MOEA/D-RS. This is because the selection pressure to select appropriate parents for each search direction works in the improved MOEA/D-gTS even h



(a) Tchebycheff (TCH) (b) Weighted Sum (WS) (c) PBI (d) IPBI

Fig. 5. Results of HV in MOKP with $m = 6$ objectives when T for the conventional MOEA/D and h for the improved MOEA/Ds are varied, respectively



(a) Tchebycheff (TCH) (b) Weighted Sum (WS) (c) PBI (d) IPBI

Fig. 6. Results of HV in MOKP with $m = 8$ objectives when T for the conventional MOEA/D and h for the improved MOEA/Ds are varied, respectively

is increased and the selection range of parents is expanded. Although h is an user-defined parameter and it needs to be tuned, these results reveal that g-tournament selection can reduce the sensitivity of h for the search performance.

Finally, we can see that the highest HV is achieved by the improved MOEA/D-gTS with IPBI scalarizing function in all MOKPs used in this work. In the previous work [19], the conventional MOEA/D with IPBI achieved higher HV than the ones with other scalarizing functions in MOKPs with 2-8 objectives. These results in this work reveal that the search performance is further improved by combining IPBI scalarizing function with the improved MOEA/Ds.

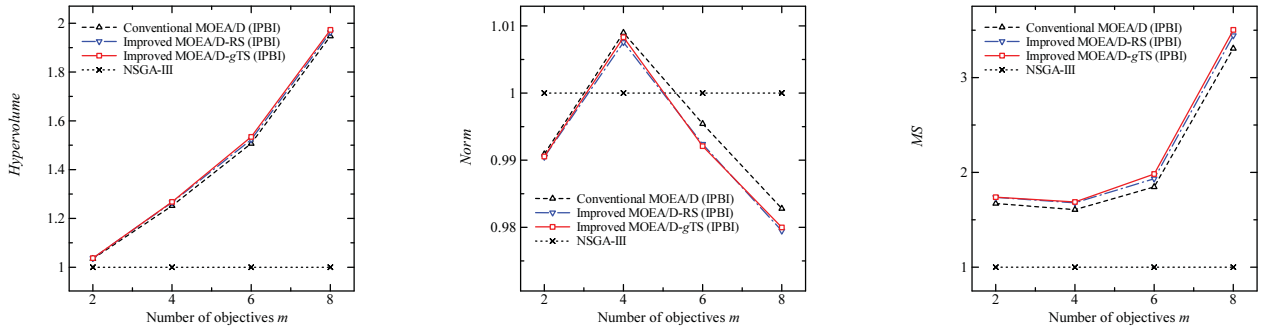
B. Analysis of Contribution Factors to HV

A scalar value of HV involves the contributions of the convergence and the diversity of the obtained solutions. In this section, we analyze the contributing factors to HV improvement by the improved MOEA/Ds. We focus only on combinations with IPBI scalarizing function which achieves the highest HV in all MOKPs in the previous section. **Fig. 7** shows the results of HV, $Norm$ and MS achieved by the conventional MOEA/D using T^*, the two improved MOEA/Ds using h^* and NSGA-III. All the results are normalized by the result of NSGA-III.

From the results of HV in **Fig. 7 (a)**, we can see that the three MOEA/Ds achieve higher HV than NSGA-III. As reported in [19], especially in problems with the difficulty to approximate a wide range of Pareto front such as MOKPs, MOEA/D achieves better HV. In all MOKPs used in this work, the two improved MOEA/Ds achieve higher HV than the conventional MOEA/D, and the improved MOEA/D-gTS achieves higher HV than the improved MOEA/D-RS. From the results of $Norm$ in **Fig. 7 (b)**, we can see that NSGA-III achieves the highest $Norm$ in $m = \{6, 8\}$ objective problems. This results clearly show that highest convergence of NSGA-III in MaOPs. On the other hand, the two improved MOEA/Ds show lower $Norm$ than the conventional MOEA/D. From the results of MS in **Fig. 7 (c)**, the two improved MOEA/Ds achieve higher MS than the conventional MOEA/D. Also, MS obtained by the improved MOEA/D-gTS is higher than the one obtained by the improved MOEA/D-RS. These results reveal that HV improvement of the improved MOEA/Ds comes form the improvement of the diversity of solutions in the objective space. Also, these results clarify that both the constant-distance based neighbors and g-tournament selection contribute to improving the diversity of the obtained solutions.

VI. Conclusions

To enhance the search performance of MOEA/D by improving its algorithm appropriate for many-objective optimization, in this work we focused on the handling of neighbors and proposed the improved MOEA/D introducing the constant-distance based neighbors and g-tournament selection. We used combinatorial many-objective knapsack problems and compared the search performances of the conventional MOEA/D, the improved MOEADs-RS, the improved MOEA/D-gTS and NSGA-III. As the results, we showed that the improved MOEA/Ds using the proposed constant-distance based neighbors with any scalarizing functions achieved higher search performance than the conventional MOEA/D in all MOKPs with 2-8 objectives. Also, we showed that HV is further improved by employing g-tournament selection in the improved MOEA/Ds with TCH, PBI and IPBI scalarizing functions. In



(a) HV (Comprehensive) (b) $Norm$ (Convergence) (c) MS (Diversity)

Fig. 7. Results of HV, MS and $Norm$ when the number of objectives m is varied

all MOKPs used in this work, the improved MOEA/D-gTS using IPBI scalarizing function achieved the highest HV. Also, we showed that both the constant-distance based neighbors and g-tournament selection contribute to improving HV by increasing the diversity of solutions in the objective space. Although the conventional MOEA/D using IPBI achieved high search performance in the latest report [19], this work revealed that the search performance is further improved by combining IPBI with the improved MOEA/D framework.

As future work, we will verify the effectiveness of the improved MOEA/D in continuous MaOPs. Also, we will study an automatic control of the distance parameter h by considering an adaptive control of T in the conventional MOEA/D [25].

ACKNOWLEDGMENT

This work was supported by JSPS KAKENHI Grant Number 26730129.

REFERENCES

[1] K. Deb, *Multi-objective optimization using evolutionary algorithms*, John Wiley & Sons, 2001.

[2] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, "A fast elitist multi-objective genetic algorithm: NSGA-II," *IEEE Trans. on EC*, Vol. 6, pp.182–197, 2002.

[3] E. Zitzler, L. Thiele, "Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach," *IEEE Trans. on EC*, Vol. 3 (4), pp. 257–271, 1999.

[4] E. J. Hughes, "Evolutionary many-objective optimisation: Many once or one many?" *Proc. of 2005 IEEE Congress on Evolutionary Computation (CEC2005)*, pp. 222-227, 2005.

[5] H. Sato, H. Aguirre, K. Tanaka, "Self-controlling dominance area of solutions in evolutionary many-objective optimization," *Proc. of the 8th Intl. Conf. on Simulated Evolution and Learning (SEAL 2010)*, Springer, LNCS, Vol. 6457, pp. 455–465, 2010.

[6] H. Ishibuchi, N. Tsukamoto, Y. Nojima, "Evolutionary many-objective optimization: A Short Review," *Proc. of 2008 IEEE Congress on Evolutionary Computation (CEC2008)*, pp. 2424-2431, 2008.

[7] K. Deb, K. Saxena, "Searching for Pareto-optimal solutions through dimensionality reduction for certain large-dimensional multi-objective optimization problems," *Proc. of 2006 IEEE Congress on Evolutionary Computation (CEC 2006)*, pp.3353–3360, 2006.

[8] D. Brockhoff, E. Zitzler, "Are all objectives necessary? On dimensionality reduction in evolutionary multiobjective optimization," *Proc. of the 9 th Parallel Problem Solving from Nature - PPSN IX*, Springer, LNCS, Vol. 4193, pp.533–542, 2006.

[9] A. L. Jaimes, C. A. C. Coello, D. Chakraborty, "Objective reduction using a feature selection technique," *Proc. of 2008 Genetic and Evolutionary Computation Conference (GECCO 2008)*, pp. 674–680, 2008.

[10] K. Deb, J. Sundar, "Preference point based multi-objective optimization using evolutionary algorithms," *Proc. of 2006 Genetic and Evolutionary Computation Conference (GECCO 2006)*, pp. 635–642, 2006.

[11] A. Auger, J. Bader, D. Brockhoff, E. Zitzler, "Articulating user preferences in many-objective problems by sampling the weighted hypervolume," *Proc. of 2009 Genetic and Evolutionary Computation Conference (GECCO 2009)*, pp. 555–562, 2009.

[12] H. Ishibuchi, N. Tsukamoto, Y. Nojima, "Iterative approach to indicator-based multiobjective optimization," *Proc. of 2007 IEEE Congress on Evolutionary Computation (CEC2007)*, pp. 3697–3704, 2007.

[13] E. Zitzler, S. Kunzili, "Indicator-based selection in multiobjective search," *Proc. of the 8th Intl. Conf. on Parallel Problem Solving from Nature (PPSN-VIII)*, Springer, LNCS, Vol. 3242, pp.832–842, 2004.

[14] N. Beume, B. Naujoks, M. Emmerich, "SMS-EMOA: Multiobjective selection based on dominated hypervolume," *European Journal of Operational Research*, Vol. 181, No. 3, pp.1653–1669, 2007.

[15] H. Ishibuchi, Y. Nojima, "Optimization of scalarizing functions through evolutionary multiobjective optimization," *Proc. of the 4th Intl. Conf. on Evolutionary Multi-Criterion Optimization (EMO2007)*, Springer, LNCS, Vol. 4403, pp.51–65, 2007.

[16] Q. Zhang, H. Li, "MOEA/D: A Multi-objective evolutionary algorithm based on decomposition," *IEEE Trans. on EC*, Vol. 11, No. 6, pp. 712–731, 2007.

[17] K. Deb, H. Jain, "An evolutionary many-objective optimization algorithm using reference-point based non-dominated sorting approach, Part I: Solving problems with box constraints," *IEEE Trans. on EC*, Volume:PP, Issue:99, pp.1–23, 2013.

[18] H. Ishibuchi, N. Akedo, Y. Nojima, "A study on the specification of a scalarizing function in MOEA/D for many-objective knapsack problems," *Proc. of Learning and Intelligent Optimization 7 (LION 7)*, Springer, LNCS, Vol. 7997, pp. 231–246, 2013.

[19] H. Sato, "Inverted PBI in MOEA/D and its Impact on the Search Performance on Multi and Many-Objective Optimization," *Proc. of 2014 Genetic and Evolutionary Computation Conference (GECCO 2014)*, pp.645-652, 2014.

[20] S. Watanabe, T. Hiroyasu, M. Miki, "Neighborhood Cultivation Genetic Algorithm for Multiobjective Optimization Problems," *Proc. of Genetic and Evolutionary Computation Conference (GECCO2002)*, pp. 458–465, 2002.

[21] H. Sato, H. Aguirre, K. Tanaka, "Variable Space Diversity, Crossover and Mutation in MOEA Solving Many-objective Knapsack Problems," *Annals of Mathematics and Artificial Intelligence*, Springer, Volume 68, Issue 4, pp. 197-224, 2013.

[22] T. Murata, H. Ishibuchi, and M. Gen, "Specification of Genetic Search Directions in Cellular Multi-Objective Genetic Algorithm," *Proc. of EMO 2001*, LNCS, Vol. 1993, pp. 82-95, 2001.

[23] E. Zitzler, *Evolutionary algorithms for multiobjective optimization: Methods and Applications*, PhD thesis, Swiss Federal Institute of Technology, Zurich, 1999.

[24] M. Sato, H. Aguirre, K. Tanaka, "Effects of δ-similar elimination and controlled elitism in the NSGA-II multiobjective evolutionary algorithm," *Proc. of 2006 IEEE Congress on Evolutionary Computation (CEC 2006)*, pp. 1164–1171, 2006.

[25] S. Z. Zhao, P. N. Suganthan, Q. Zhang, "Decomposition-based multiobjective evolutionary algorithm with an ensemble of neighborhood sizes," *IEEE Trans. on EC*, Vol. 16, Issue 3, pp. 442–446, 2012.