Journal of Computer Science and Cybernetics, V.33, N.3 (2017), 193–212
DOI 10.15625/1813-9663/33/3/11111

REVIEW

MULTI-OBJECTIVE EVOLUTIONARY ALGORITHMS: FOUNDATION, DEVELOPMENT AND OPEN ISSUES

LAM THU BUI

Le Quy Don Technical University; lambt@lqdtu.edu.vn

Abstract. Evolutionary computation (EC) has been a fascinating branch of computation inspired by a natural phenomenal of evolution. EC enables computer scientists to design effective algorithms dealing with difficult problems. This paper focuses on a special class problem called multi-objective optimization problems and evolutionary algorithms designed for it. We will overview the development of multi-objective evolutionary algorithms (MOEAs) over the years and problem difficulties and then indicate the open problems in this area. Our chief goal is to provide readers with reference material in the area of multi-objective evolutionary algorithms.

Keywords. Evolutionary computation, multi-objective evolutionary algorithms, optimization.

1. INTRODUCTION

Evolutionary algorithms (EAs) [3, 28, 40] have emerged as major heuristic search paradigms. With the usage of a population for the search in each iteration, EAs are naturally suitable for solving multi-objective problems, which often have multiple conflicting objectives. They have attracted significant attention from the research community over the last 30 years. We can observe this fact by the number of publications produced over time [13, 16, 17]. Obviously, in these problems, there is no single solution that is the best when measured on all objectives (note that the terms *solution, individual* and *point* are used interchangeably in this paper). Instead we usually find several trade-off solutions (called the *Pareto optimal set* (POS) to honor Vilfredo Pareto [44], or *Pareto optimal front* (POF) for the image of the vectors corresponding to these solutions). In that sense, the search for an optimal solution has fundamentally changed from what we see in the case of single-objective problems. The task of solving multi-objective optimization problems (MOPs) is called *multi-objective optimization.* An example is given in Figure 1 where we are considering a MOP, such as design, scheduling, or investment problems addressing two objectives: low cost and high performance.

Users practically need only one solution from the set of optimal tradeoff solutions. Therefore, solving MOPs can be seen as the combination of both searching and decision making processes [31]. There is a large set of techniques being proposed over years for MOPs. Generally speaking, there are four main approaches in the literature [42]. The first one does not use preference information (called *no-preference*). These methods solve a problem and give a solution directly to the decision maker. The second one is to find all possible solutions of the non-dominated set and to then use the user preference to determine the most suitable one (called *decision making after search*, or *posterior*). Meanwhile, the third approach is to

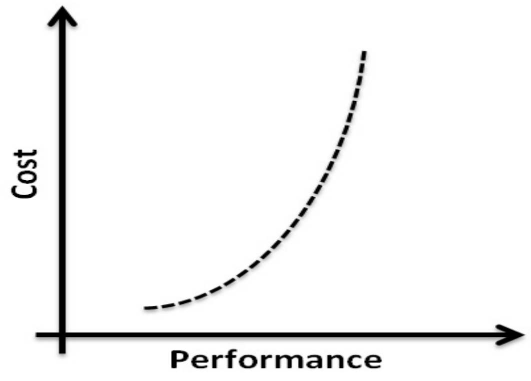

Figure 1. There are several trad-off optimal solutions (black dots) satisfying the problem of cost-performance

use preference before the optimization process; and hence it will result in only one solution at the end (called *decision making before search*, or *priori*). With this approach, the bias (from the user preference) is imposed all the time. The fourth approach (called *decision making during search*, or *interactive*) is to hybridize the second and third ones in which human decision-making is periodically used to refine the obtained trade-off solutions and thus to guide the search. In general, the second one is mostly preferred within the research community since it is less subjective than the other two.

This paper is dedicated to make an overview of multi-objective evolutionary algorithms (MOEAs) and then pinpoints a number of research open issues and problem difficulties. Therefore, it is organized as follows: the second section is for the common concepts and notations in multi-objective optimization using evolutionary algorithms. It is followed by descriptions of traditional multi-objective algorithms as well as MOEAs (the third and fourth sections respectively). The fifth and sixth sections are dedicated to the performance assessment and research issues. The paper is concluded in Section 7.

2. BACKGROUND

This section will define some common concepts that have been widely used in the literature as well as in this paper. Interested readers might refer to books of [15, 19, 23, 42, 10] for a more detailed description. Note that for evolutionary multi-objective optimization, in addition to the *decision variable space* (phenotype), we have the *decision space* or *search space* (genotype), while also the objectives form another space, called the *objective space.* However, this paper uses a continuous representation, so there is no distinction between genotype/phenotype. Therefore, the two spaces are identical.

Mathematically, in a k-objective optimization problem, a vector function $\overrightarrow{f}(\overrightarrow{x})$ of k objectives is defined as

$$\overrightarrow{f}(\overrightarrow{x}) = \begin{bmatrix} f_1(\overrightarrow{x}) \\ f_2(\overrightarrow{x}) \\ ... \\ f_k(\overrightarrow{x}) \end{bmatrix}, \tag{1}$$

in which $\overrightarrow{x}$ is a vector of decision variables in the n-dimensional space $\mathbb{R}^n$; n and k are not necessarily the same. A solution is assigned a vector $\overrightarrow{x}$ and therefore the corresponding objective vector $\overrightarrow{f}$. Therefore, a general MOP is defined as follows

$$\min f_i(\overrightarrow{x}) | \overrightarrow{x} \in D, \tag{2}$$

where $i = 1, 2, ..., k$ and $D \subseteq R^n$, called the *feasible search region.* All solutions (including optimal solutions) that belong to D are called *feasible solutions.*

In general, when dealing with MOPs, a solution x_1 is said to dominate x_2 if x_1 is better than x_2 when measured on all objectives. If x_1 does not dominate x_2 and x_2 also does not dominate x_1, they are said to be non-dominated. If we use $\preceq$ between x_1 and x_2 as $x_1 \preceq x_2$ to denote that x_1 dominates x_2 and $\lhd$ between two scalars a and b, as $a \lhd b$ to denote that a is better than b (similarly, $a \rhd b$ to denote that a is worse than b, and $a \ntriangleright b$ to denote that a is not worse than b), then the dominance concept is formally defined as follows.

Definition 1. $x_1 \preceq x_2$ if the following conditions are held:
1. $f_j(x_1) \ntriangleright f_j(x_2)$, $\forall j \in [1, 2, ..., k]$;
2. $\exists$j $\in [1, 2, ...,$ k] in which $f_j(x_1) \lhd f_j(x_2)$.

The concept defined in Definition 1 is sometimes referred to as *weak dominance.* For the *strict dominance* concept, solution x_1 must be strictly better than x_2 in all objectives. However, this paper follows the weak dominance concept as defined in Definition 1. Further, the relation between individuals is called the *dominance relation.*

Generally, there are several properties of binary relations that are related to the dominance relation; and they are listed as follows:

- **Irreflexive**: From Definition 1, clearly a solution does not dominate itself. Therefore, the dominance relation is not reflexive (or is irreflexive).
- **Asymmetric**: The dominance relation is asymmetric since $x_1 \preceq x_2$ does not imply $x_2 \preceq x_1$.
- **Antisymmetric**: The dominance relation is asymmetric, therefore it is not antisymmetric (recall that this property requires that if $x_1 \preceq x_2$ and $x_2 \preceq x_1$ then $x_1 = x_2$).
- **Transitive**: From Definition 1, we can see that if $x_1 \preceq x_2$ and $x_2 \preceq x_3$ then $x_1 \preceq x_3$. Therefore, the dominance relation is transitive.

Definition 2. A binary relation R is called

- partially ordered if it is reflexive, antisymmetric and transitive,
- strictly partially ordered if it is asymmetric and transitive.

Clearly, the dominance relation is a strict partial order relation (see Definition 2) since it is not reflexive and not antisymmetric. This is an important finding and it has been used in considering theoretical aspects of multi-objective optimization [23].

In dealing with MOPs, EAs use a population (set) of individuals during the optimization process. At the end, we usually have a set of individuals where no one individual is better than any other one in the set. This set is an approximation of the real optimal solutions for the problem.

In general, if an individual in a population is not dominated by any other individual in the population, it is called a non-dominated individual. All non-dominated individuals in a population form the non-dominated set (as formally described in Definition 3). Note that these definitions are equivalent to that from [19].

Definition 3. A set S is said to be the non-dominated set of a population P if the following conditions are held:

1. $S \subseteq P$*;*

2. $\forall s \in S, \nexists x \in P | x \preceq s$.

When the set P represents the entire search space, the set of non-dominated solutions S is called the *global Pareto optimal set.* If P represents a sub-space, S will be called the *local Pareto optimal set.* There is only one global Pareto optimal set, but there could be multiple local ones. However, in general, we simply refer to the global Pareto optimal set as the Pareto optimal set (POS). Although there are several conditions established in the literature for optimality [23, 42], however, for practical black-box optimization problems, these conditions generally can not be verified easily.

Finally, it is worthwhile to mention here two special objective vectors (assuming that the problem is minimization) that are related to the Pareto optimal set (see [23], p. 34). For the sake of simplicity, these vectors are also called '*solutions*'

- *Ideal solution*: This represents the lower bound of each objective in the Pareto optimal set. It can be obtained by optimizing each objective individually in the entire feasible objective space.

- *Nadir solution*: This contains all the upper bound of each objective in the Pareto optimal set. Obtaining the Nadir solution over the Pareto optimal set is not an easy task. One of the common approaches is to estimate the Nadir point by a pay-off table based on the Ideal solution.

3. TRADITIONAL MULTI-OBJECTIVE ALGORITHMS

We use the term '*traditional*' in this context to differentiate such methods from evolutionary ones. There is a handful of traditional methods, such as the method of global criterion, weighted-sum [18, 42], ϵ-constraint [29], weighted metric [42], and goal programming [50]. This section will only summarize several approaches that represent for different categories.

3.1. No-preference methods

There is no preference for the decision maker when receiving the solution of the optimization process. They can make the choice to accept or reject it. For this, the no-preference methods are suitable in the case that the decision maker does not have specific assumptions on the solution. The method of *global criterion* [42, 61] can be used to demonstrate this class of methods.

For this method, MOPs are transformed into single objective optimization problems by minimizing the distance between some reference points and the feasible objective region.

In the simplest form (using L_p-metrics), the reference point is the ideal solution and the problem is represented as follows

$$\text{minimize } \left(\sum_{i=1}^{k} |f_i(x) - z_i^*|^p\right)^{\frac{1}{p}}, \tag{3}$$

where z^* is the ideal vector, and k is the number of objectives.

When $p = \infty$, it is called a Tchebycheff problem with a Tchebycheff metric and is presented as follows

$$\text{minimize } \max_{i=1,..,k} |f_i(x) - z_i^*|. \tag{4}$$

From the equation, one can see that the obtained solutions depend very much on the choice of the p's value. Also, at the end the method will only give one solution to the decision maker.

3.2. Posteriori methods

For posteriori methods, a set of Pareto optimal solutions is presented to the decision maker and the most suitable one will be selected based on the decision maker's preference. Here, we overview the two most popular approaches: weighted sum and ϵ-constraint.

For the weighted-sum method, all the objectives are combined into a single objective by using a weight vector. The problem in Eq. 2 is now transformed as in Eq. 5.

$$\min f(\overrightarrow{x}) = w_1 f_1(\overrightarrow{x}) + w_2 f_2(\overrightarrow{x}) + ... + w_k f_k(\overrightarrow{x}) | \overrightarrow{x} \in D, \tag{5}$$

where $i = 1, 2, ..., k$ and $D \subseteq R^n$.

The weight vector is usually normalized such that $\sum w_i = 1$.

Although the weighted-sum method is simple and easy to use, there are two inherent problems. Firstly, there is the difficulty of selecting the weights in order to deal with scaling problems since the objectives usually have different magnitudes. Therefore, when combining them together, it is easy to cause biases when searching for trade-off solutions. Secondly, the performance of the method is heavily dependent on the shape of POF. Consequently, it cannot find all the optimal solutions for problems that have a non-convex POF.

To overcome the difficulty of non-convexity, the method of ϵ-constraint has been introduced, where only one objective is optimized while tranforming the others into constraints. The problem in Eq. 2 is now transformed as in Eq. 6. Again, the problem is now transformed into a single objective one

$$\min f_j(\overrightarrow{x}) | \overrightarrow{x} \in D, \tag{6}$$

subject to $f_i(\overrightarrow{x}) \leq \epsilon_i$ where $i = 1, 2, ..., k$, $i \neq j$ and $D \subseteq R^n$.

In this method, the ϵ vector is determined and uses the boundary (upper bound in the case of minimization) for all objectives i. For a given ϵ vector, this method will find an optimal solution by optimizing objective j. By changing ϵ, we will obtain a set of optimal solutions. Although, this method alleviates the difficulty of non-convexity, it still has to face the problem of selecting appropriate values for the ϵ vector, since it might be that case that for a given ϵ vector, there does not exist any feasible solution.

3.3. Priori methods

For these methods, the decision maker is requested to indicate his/her assumption about the preferences before starting the optimization process. Therefore, the issue is how to quantify the preference and incorporate it into the problem before the optimization process. Here, one obvious method is the weighted-sum where the weights can be used to represent the decision maker's preference.

However, here the methods of lexicographic ordering and goal programming [24, 33, 42] are used to demonstrate the use of priori preference. For a lexicographic method, the decision maker is asked to arrange the objective functions by relying on their absolute importance. The optimization process is performed individually on each objective following the order of importance. After optimizing with the most important objective (the first objective), if only one solution is returned, it is the optimal solution. Otherwise, the optimization will continue with the second objective and with a new constraint on the obtained solutions from the first objective. This loop might continue to the last objective.

For the method of goal programming, aspiration levels of the objective functions will be specified by the decision maker. Optimizing the objective function with a aspiration level is seen as a goal to be achieved. In its simplest form, goal programming can be stated as below

$$\text{minimization} \sum_{i=1}^{k} |f_i(x) - z_i|, \tag{7}$$

where z is the vector indicating the aspiration levels. A more general formulation of this equation can be derived by replacing $|f_i(x) - z_i|$ by $|f_i(x) - z_i|^p$.

3.4. Interactive methods

We are now looking at a special class called interactive methods. The decision maker is allowed to interact with the optimization program (or an analyst). In general, the interaction can be described step-by-step as follows (see [42]):

- **Step 1**: Find an initial feasible solution;
- **Step 2**: Interact with the decision maker, and;
- **Step 3**: Obtain a new solution (or a set of new solutions). If the new solution (or one of them) or one of the previous solutions is acceptable to the decision maker, stop. Otherwise, go to Step 2.

With the interaction between the program and the decision maker, as indicated in [42], many weaknesses from the above approaches can be alleviated. To date, there are many approaches using an interactive style, namely, GDF [27], Tchebycheff method [50] (pp. 419-450), Reference point method [59], NIMBUS [41]. Recently, interactive methods have also been incorporated with MOEAs (see [2] for an example). Here we use the reference direction approach to demonstrate this class of the methods. The approach is described as follows:

- **Step 1**: Present the decision maker information of the problem such as the ideal solution;

- **Step 2**: Request the decision maker to specify a reference point $\overline{z}$;
- **Step 3**: Minimize the achievement function (an example is given in Eq. 8) and obtain a Pareto optimal solution x and the corresponding z. Present z to the decision maker

$$\text{minimize} \max_{i=1,..,k} [w_i(f_i(x) - z_i)]; \tag{8}$$

- **Step 4**: Calculate a number of k other Pareto optimal solutions by minimizing the achievement function with perturbed reference points: $\overline{z}(i) = \overline{z} + de^i$, where $d = \|\overline{z} - z\|$ and e^i is the i^{th} unit vector for $i = 1, .., k$;
- **Step 5**: Present the alternatives to the decision maker. If he/she finds one of the $k+1$ solutions satisfied, the corresponding x is the final solution. Otherwise, go to Step 3.

From the above basic steps, it appears that the approach is very simple and practical. The preference is incorporated into the achievement function and therefore the problem becomes single objective. The perturbation of the reference point gives the decision maker more understanding of the Pareto optimal set.

4. MULTI-OBJECTIVE EVOLUTIONARY ALGORITHMS

4.1. Overview

Multi-objective evolutionary algorithms (MOEAs) have been around for a while. They are considered as a class of stochastic optimization techniques. Similar to other optimization algorithms, MOEAs are designed to find Pareto optimal solutions for a particular problem, but differ by using a population-based approach. This paper focuses on the class of dominance-based MOEAs. Although it is possible of not using dominance relation when designing MOEAs, such as VEGA [48], the majority of existing MOEAs employ the concept of dominance in their courses of action.

The optimization mechanism of MOEAs is quite similar with that of EAs, except for the use of the dominance relation (fitness assignment). In more detail, at each iteration, the objective values are calculated for every individual and are then used to determine the dominance relationships within the population, in order to select potentially better solutions for the production of the offspring population. This population might be combined with the parent population to produce the population for the next generation. Further, the existence of the objective space might give MOEAs the flexibility to apply some conventional supportive techniques such as niching.

Generally, MOEAs have to deal with two major inherent issues [19]: (1) how to get close to the Pareto optimal front. This is not an easy task, because converging to the POF is a stochastic process. (2) how to keep diversity among the solutions in the obtained set. These two issues have become common criteria for most current algorithmic performance comparisons [67]. A diverse set of solutions will give more options for decision makers, designers, etc. However, working on a set of solutions instead of only one, makes the measurement of the convergence of a MOEA harder, since the closeness of one individual to the optima does not act as a measure for the entire set.

Table 1. The common framework for multi-objective optimization evolutionary algorithms

```
Step 1: Initialize a population P;
Step 2: (optional):Select elitist solutions from P to create/update an external set FP
(For non-elitism algorithms, FP is empty);
Step 3: Create a mating pool from one or both of P and FP;
Step 4: Perform reproduction based on the pool to create the next generation P;
Step 5: Possibly combine FP into P;
Step 6: Go to step 2 if the termination condition is not satisfied.
```

To date, many MOEAs have been developed. Generally speaking, there are several ways to classify MOEAs. However, this paper follows the one used in [13] where they are classified into two broad categories including elitism and non-elitism[1]

4.2. Non-elitism approach

This approach does not explicitly keep the best solutions when it does selecting individuals for the next generation from the current population [19]. Instead, selected individuals from the current generation are used to exclusively generate solutions for the next generation by crossover and mutation operators as in EAs. The only difference from conventional EAs is that they use the dominance relation when assessing solutions. Instances of this category include MOGA [25], NPGA [32] and NSGA [19].

The authors in [13] classify all algorithms using this approach as instances of the *first generation* of MOEAs which implies simplicity. Although MOEAs are different from each other, the common steps of these algorithms can be summarized as shown below (Table 1). Note that steps 2 and 5 are used for elitism approaches that will be summarized in the next sub-section.

4.3. Elitism approach

In EC, elitism is a concept of preserving the best individuals from generation to generation. By this way, the system never loses the best individuals found during the optimization process. Elitism was used at quite an early stage of evolutionary computing (see [22]) for an example); and to date, it has been used widely with EAs. Elitism can be done by placing one or more of the best individuals directly into the population for the next generations, or by comparing the offspring individual with its parents and then the offspring will only be considered if it is better than the parent [51].

In the domain of evolutionary multi-objective optimization, elitist MOEAs usually (but not necessarily) employ a concept of an external set (the archive) to store the non-dominated solutions after each generation. In general, when using the archive, there are two important aspects as follows:

- How to get the archive and the main population interacted? This is about how we use the archive during the optimization process; for example, one such way is to combine

[1]In [13], the author used the first and second generations of MOEAs. However, the classification actually relied on elitism

the archive with the current population to form the population for the next generation as in [65].

- How to update the archive? This is about the methodology to build the archive, one such method is by using the neighborhood relationship between individuals using crowded dominance [20], clustering [65], or geographical grid [39], while another method is by controlling the size of the archive through truncation when the number of non-dominated individuals are over a predefined threshold.

Obviously, the current archive might then be a part of the next generation; however, the way to integrate this archive may be different from one algorithm to another. In general, with elitism, the best individuals in each generation are always preserved, and this helps the algorithms to get closer to the POF; a proof of convergence for MOEAs using elitism can be found in [46]. By implementing elitism, the algorithmic design of MOEAs becomes diverse with dominance ranking[20], decomposition[62], direction[9]; and hence resutls in many algorithms to date. Algorithms such as PAES [39], SPEA2 [65], PDE [1], NSGA-II [20], MOPSO [14], MOEA/D [62], NSGA-III [21], and DMEA[9] are typical examples of this category.

4.4. Selected MOEAs

This section will summarize several approaches in the literature. These approaches are selected since they are the current state-of-the-art algorithms and have been widely used in the field.

4.4.1. Non-dominated sorting genetic algorithms version 2 (NSGA-II)

NSGA-II is considered as an elitism algorithm [19, 20] with a special elitism-preservation operation. Note that NSGA-II does not use an explicit archive; a population is used to store both elitist and non-elitist solutions for the next generation. However, for consistency, it is still considered as an archive. Firstly, the archive size is set equal to the initial population size. The current archive is then determined based on the combination of the current population and the previous archive. To do this, NSGA-II uses dominance ranking to classify the population into a number of layers, such that the first layer is the non-dominated set in the population, the second layer is the non-dominated set in the population with the first layer removed, the third layer is the non-dominated set in the population with the first and second layers removed and so on. The archive is created based on the order of ranking layers: the best rank being selected first. If the number of individuals in the archive is smaller than the population size, the next layer will be taken into account and so forth. If adding a layer makes the number of individuals in the archive exceed the initial population size, a truncation operator is applied to that layer using *crowding distance*.

The *crowding distance* D of a solution x is calculated as follows: the population is sorted according to each objective to find adjacent solutions to x; boundary solutions are assigned infinite values; the average of the differences between the adjacent solutions in each objective is calculated; the truncation operator removes the individual with the smallest *crowding distance*.

$$D(x) = \sum_{m=1}^{M} \frac{F_m^{I_x^m+1} - F_m^{I_x^m-1}}{F_m^{\max} - F_m^{\min}} \tag{9}$$

in which, F is the vector of objective values, and I_x^m returns the sorted index of solution x, according to objective m^{th}.

An offspring population of the same size as the initial population is then created from the archive, by using crowded tournament selection, crossover, and mutation operators. Crowded tournament selection is a traditional tournament selection method, but when two solutions have the same rank, it uses the crowding distance to break the tie.

4.4.2. Strength pareto evolutionary algorithm 2 (SPEA2)

SPEA2 is actually the elitist version of 'The Strength Pareto Evolution Algorithm' - SPEA [67]. For it, the initial population, representation and evolutionary operators are standard: uniform distribution, binary representation, binary tournament selection, single-point crossover, and bit-flip mutation. However, the distinctive feature of SPEA2 lies in the elitism-preserved operation.

An external set (archive) is employed for storing primarily non-dominated solutions. It is then combined with the current population to form the next archive that is then used to create offspring for the next generation. The size of the archive is fixed. It can be set to be equal to the population size. Therefore, there exist two special situations when filling solutions in the archive. If the number of non-dominated solutions is smaller than the archive size, other dominated solutions taken from the remainder part of the population are filled in. This selection is carried out according to a fitness value, specifically defined for SPEA. That is the individual fitness value defined for a solution x, is the total of the SPEA-defined strengths of solutions which dominate x, plus a density value.

The second situation happens when the number of non-dominated solutions is over the archive size. In this case, a truncation operator is applied. For that operator, the solution which has the smallest distance to the other solutions will be removed from the set. If solutions have the same minimum distance, the second nearest distance will be considered, and so forth. This is called the *k-th nearest distance rule.*

4.4.3. Direction-based multi-objective evolutionary algorithm (DMEA)

For DMEA [9], two types of directions are used including convergence direction (from a dominated solution to a non-dominated one) and spreading direction (between two non-dominated solutions) for generation of offspring along those directions. We call them as directions of improvement. A population of solutions is maintained and evolved over time under the guidance of directions of improvement. Further, an archive is maintained over time. This archive is not only used for contributing elite solutions for the next generation, but also for deriving the directions. In order to fill the populations of the next generation as well as the archive, DMEA gets solutions from the combined population (resulted from combining the current archive and the offspring). For the next generation, the non-dominated solutions using niching information in the decision space is filled in the first half of the population, while the other half is filled by using a weighted-sum technique for all remaining solutions in the combined population. The archive needs to be updated and the selection of non-dominated solutions for updating is also assisted by a niching mechanism, in which neighborhoods of rays in objective space serve as niches. These rays originate from the current estimate of the ideal point and emit uniformly into the hyper-quadrant that contains the current POF.

4.4.4. A multi-objective evolutionary algorithm based on decomposition(MOEA/D)

The authors proposed a new multi-objective evolutionary algorithm based on decomposition [62]. Basically, MOEA/D explicitly decomposes the MOP into N scalar optimization subproblems. A population of solutions is evolved in order to solve these subproblems simultaneously. The current population is composed of the best solution found so far (i.e. since the start of the run of the algorithm) for each subproblem. In this way, the elitism is preserved. MOEA/D defines the neighborhood relations among these subproblems based on the distances between their aggregation coefficient vectors.The optimal solutions to two neighboring subproblems should be very similar. Note that each subproblem (i.e., scalar aggregation function) is optimized in MOEA/D by using information only from its neighboring subproblems.

5. PERFORMANCE ASSESSMENTS

It is commonsense that performance metrics are usually used to compare algorithms in order to form an understanding of which one is better and in what aspects. However, it is hard to define a concise definition of algorithmic performance, especially in the case of MOEAs. In general, when doing comparisons, a number of criteria are employed [67]:

- How close the obtained non-dominated set is to the Pareto optimal front?
- How good is the distribution of solutions within the set (in most cases, uniform)?
- How spreading is the obtained non-dominated front? i.e., for each objective, a wide range of values should be covered by the non-dominated solutions.

Based on these criteria, a number of performance metrics has been developed. There have been a number of efforts to develop platforms for performance assessments including most popular metrics such as the PISA system [4]. This section will provide a summary of the most popular ones of these metrics.

5.1. Metrics evaluating closeness to POF

The first obvious metric is the error rate, ER, introduced by Veldhuizen [55]. It is calculated by the percentage of solutions that are not in POF

$$ER = \frac{\sum_{i=1}^{N} e_i}{N}, \tag{10}$$

where N is the size of the obtained set and $e_i = 1$ if the solution i is not in the POF, otherwise $e_i = 0$. The smaller the ER, the better the convergence to the POF. However, this metric does not work in the case when all the solutions of two compared sets are not in the

POFs. In this case, a threshold is employed, such that if the distance from a solution i to the POF is greater than the threshold, $e_i = 1$, otherwise $e_i = 0$.

The second metric is the generation distance, *GD*, which is the average distance from the set of solutions found by evolution to POF [55]

$$GD = \frac{\sqrt{\sum_{i=1}^{N} d_i^2}}{N}, \tag{11}$$

where d_i is the Euclidean distance (in objective space) from solution i to the nearest solution in POF. If there is a large fluctuation in the distance values, it is also necessary to calculate the variance of the metric. Finally, the objective values should be normalized before calculating the distance.

5.2. Metrics evaluating diversity among obtained non-dominated solutions

The spread metric is also an important one in performance comparisons. One of its instances is introduced by Schott [49], called the *spacing* method

$$S = \frac{\sqrt{\frac{1}{N}\sum_{i=1}^{N}(\overline{d} - d_i)^2}}{\overline{d}}, \tag{12}$$

where $d_i = \min_{j=1->N} \sum_{m=1}^{M} |f_m^i - f_m^j|$ and f_m is the m^{th} objective function. N is the population size and M is the number of objectives. The interpretation of this metric is that the smaller the value of S, the better the distribution in the set. For some problems, this metric might be correlated with the number of obtained solutions. In general, this metric focuses on the distribution of the Pareto optimal set, not the extent of the spread.

In [20], the authors proposed another method to alleviate the problem of the above spacing method. The spread of a set of non-dominated solutions is calculated as follows

$$\Delta = \frac{\sum_{i=1}^{M} d_i^e + \sum_{i=1}^{N} |d_i - \overline{d}|}{\sum_{i=1}^{M} d_i^e + N\overline{d}}, \tag{13}$$

where d_i can be any distance measure between neighboring solutions and $\overline{d}$ is the mean value of these distances. d_i^e is the distance between extreme solutions of the obtained non-dominated set and the true Pareto optimal set. Δ ranges from 0 to 1. If it is close to 1, the spread is bad.

5.3. Metrics evaluating both closeness and diversity

Note that all of the above metrics focus on a single criterion only. This section summarizes the others that take into account both closeness and diversity. The first one is the inverse generation distance (IGD). Given the obtained solution set P, the first-norm equation for IGD is as follows

$$IGD = \frac{\sum_{i=1}^{N} d_i}{N}, \tag{14}$$

where d_i is the Euclidean distance (in objective space) from solution i in POS to the nearest solution in P, and N is the size of POS. In order to get a good value for IGD (ideally zero),

P needs to cover all parts of POS. Note that this method only focuses on the solution that is closest to the solution in POS indicating that a solution in P might not take part in this calculation.

The second one is the hyper-volume ratio [66], one of the most widely accepted by the research community of MOEAs. To calculate the hyper-volume, an area of objective space covered by the obtained POF is measured, called the hyper-area. Note that calculating the hyper-volume is a time-consuming process (although recently several attempts have been given to speed up this process [57, 58]). In general, for two sets of solutions, whichever has the greater value of hyper-volume will be the best. However, when using hyper-volume it is sometime difficult to understand the quality of the obtained POF in comparison with the true POF.

As recommended in [15, 55], it is considered better to use the hyper-volume ratio, *HR*, that is measured by the ratio between the hypervolumes of hyperareas covered by the obtained POF and the true POF, called H_1 and H_2 respectively. *HR* is calculated as in Eq. 15. For this metric, the greater the value of *HR*, the better the convergence that the algorithm has

$$HR = \frac{H_1}{H_2}. \tag{15}$$

It is important to define the reference point for the calculation of the hyper-volume (for example, it can be the origin [55]). However, generally it is dependent on the area of the objective space that is visited by all comparing algorithms. In this paper, as suggested elsewhere [19], the reference point is the one associated with all the worst values of objectives found by all the algorithms under investigation.

The third metric uses a statistical comparison method. It was first introduced by Fonesca and Fleming [26]. For experiments of MOEAs, which generate a large set of solutions, this metric is often the most suitable, as their data can easily be assessed by statistical methods. Knowles and Corne [38] modified this metric and instead of drawing parallel lines, all lines originate from the origin. The basic idea is described with an example as follows: suppose that two algorithms (*A1, A2*) result in two non-dominated sets: *P1* and *P2*. The lines that join the solutions in $P1$ and $P2$ are called attainment surfaces. The comparison is carried out in the objective space. In order to do the comparison, a number of lines are drawn from the origin (assuming a minimization problem), such that they intersect with the surfaces. The comparison is then individually done for each sampling line to determine which one outperforms the other. Each intersection line will then yield to a number of intersection points. In this case, statistical tests are necessary to determine the percentage an algorithm outperformed the other in each section. For both of these methods, the final results are two numbers that show the percentage of the space where each algorithm outperforms the other.

5.4. Statistical testing

Because of the stochastic nature, we cannot rely on the results obtained by MOEAs and from only one run tested on a particular problem. Therefore, it is necessary that every algorithm involved in the comparison needs to be tested on the problem for a number of independent runs (equivalent to using different random seeds, i.e 30 runs). By applying the above metrics (except the one using attainment surfaces), at the end, a set of numerical values was obtained for each algorithm. All comparisons will be done on these sets. This helps to reduce the effect of randomness. Further, statistical tests need to be carried out on the results from these runs in order to make fair comparisons and judgments. An amount of 30 runs was employed across all experiments since it is commonly used in the literature of evolutionary multi-objective optimization [14, 52, 67].

From the statistical point of view, there are a number of concepts that can be used to compare the sets, including the mean, standard deviation, and median. However, the confidence on using these concepts in comparison is questionable. In general, the final decision on the performance of algorithms will be made after completing a statistical testing. Among a diverse collection of testing methods, the paired two-sample *t-test* [47] is quite popular. The null hypothesis is the equality of the two sets. The significant level is always set at 0.05. Thus, if the confident interval achieved from a test is greater or equal to 95%, the difference between the two sets is considered to be significant.

6. RESEARCH ISSUES AND PROBLEM DIFFICULTIES

Research in the area of MOEAs has attracted lots of attention and has been very fruitful. However, still we have rooms to improve and implement new algorithms. Here, we try to briefly overview all major research issues in design an evolutionary multi-objective search.

- **Representation**: Representing the problem's solution is the center of the optimization story. For MOEAs, it decides how we design crossover and mutation operators. The volume of the search space is dependent on the solution's representation. It also is the foundation for other concepts to be stemmed from such as genotype-phenotype mapping, redundancy, or causality [45]. Further, the representation should be suitable for the nature of the problems which might have variables in the discrete, real-valued or mixed formats. A recent example can be seen in Shayan et al. [36].

- **Fitness assignment**: Fitness is a measure of how good a solution is within a population. In single-objective optimization problems, it can be easily derived from the solution's objective values. However, in the case of multi-objective ones, the relation between solutions is not totally ordered. Therefore, fitness assignment is not straightforward as in single objective any more (such as using the objective value). For example, NSGA-II takes into account both the dominance-based rank of solutions and their crowding distance when considering their fitness.

- **Population diversity**: MOEAs are usually expected to return a set of trade-off solutions. Therefore, diversity is essential for them. Further, as EAs, the lost of diversity can make MOEAs easily get trapped in local POFs. An example of keeping diversity can be seen as the use of fitness sharing (see NSGA [19] or DMEA [9].

- **Elitism**: As shown above, elitism is an important aspect of developing MOEAs. The questions of how to apply and how to exploit elitism are always essential in design of multi-objective searches. A detailed discussion was given in Bui et al [8] for the usage of elitism in the context of MOEAs.

- **Population sizing**: For EAs, setting an appropriate population size is very critical. Although, population sizing is relatively related to the problem scale, there is no universal definition over the area of evolutionary computation. However, there exist a number of work have been done for genetic algorithms as well as for MOEAs in certain situations.

Further, there have been a number of possible challenges that MOEAs might face in solving MOPs. By recognizing these challenges, the algorithmic designs will be benefited accordingly. Note that there are several problem difficulties for conventional optimization that do not matter to MOEAs, such as convexity, concavity, discontinuity, or differentiability. Although they are not always separated, but for the purpose of reference materials, they can be categorized as follows:

- **Multi-modality**: Problems with multi-modality usually have multiple local optima. This causes the algorithm to become easily attracted to a basin associated with a local optima. This causes one of the major issues that faces the application of optimization algorithms to many real life problems. Solving these problems is the main objective of a class of optimization algorithms called global optimization [53]. In multi-objective optimization, multi-modality of a problem is understood in the sense that the problem has many local Pareto optimal fronts and only one global POF.

- **Dynamic and uncertain environments**: The world is not static; in fact it is changing from time to time with different levels and at almost all aspects. That is why dynamism happening on many real-world optimization problems, such as timetabling, routing, or path-planning [11]. For a detailed example, we can consider the problem of school timetabling. Since changes of class room, or the absence of teachers can happen at any time, the school timetabling system need to be well aware and adaptive in order to be able to deal with these changes. In general, there are several aspects when dealing with dynamic environments including the frequency of change and the severity of change [5]. Further, the changes might also come form the uncertainly of decision maker [6].

  Regarding noisy environments, a noise effect is inevitable in many real-world problems. Sources of noise can vary depending on the way data is obtained such as the sensors, and actuators, or because of the stochastic elements pertaining in some problems such as multi-agent simulations. In the presence of noise, the optimization process might be misled by including inferior solutions. If the distribution of noise is symmetric around zero, such population-based approaches as evolutionary algorithms can maintain well their progress since the population acts as a noise filter [7, 43].

- **Scalability**: This challenge has a long history associated with optimization. Originally, it was considered in the decision space, where by increasing the number of variables, the search space became larger, more complicated and more vulnerable to the effect of numerical rounding errors [56, 64]. With the development of multi-objective optimization, scalability is also considered in the objective space with more than two objectives [30, 34, 37, 63], which we call many-objective optimization.

- **Expensive objective-value evaluation**: Each objective-value evaluation will take a large amount of time. So this difficulty does not cause errors in approaching the optima. However, it sometimes makes the optimization process become impossible, since it has to spend too much time on objective evaluation efforts. Here, we mention two popular approaches, which have been using estimated objective functions [35] and employing distributed computation [12, 54].

Note that these challenges are main themes in the research community of evolutionary computation in both theoretical and experimental senses. The consideration of these aspects reveals a key rule that the success of an algorithm is heavily dependent on the study of the search problems with regards to the no-free lunch theorem [60].

7. CONCLUSION

This paper provided an overview of different aspects of evolutionary multi-objective optimization including mathematical foundations, design, and performance assessment. It is worthwhile to note that the popular contribution in this area is centered around the algorithmic design of a new MOEA for a particular class of problems. With a special characteristic of using populations for searching, MOEAs have been a main research stream in evolutionary computation over the last three decades with a large volume of publications, dedicated highquality journals, and textbooks. They have been applied to solve effectively real world problems ranging from mechanical design and network design to planning and scheduling. However, there are still many problems opened for further investigation and analysis including issues in algorithmic design and application domains. Hence, the paper is expected to serve the wide audience of researchers a reference source.

ACKNOWLEDGMENT

This research is funded by Vietnam National Foundation for Science and Technology Development (NAFOSTED) under grant number 102.01-2015.12.

REFERENCES

[1] H. A. Abbass, R. Sarker, and C. Newton, "PDE: A Pareto frontier differential evolution approach for multiobjective optimization problems," in *Proceedings of the Congress on Evolutionary Computation*, vol. 2. Seoul Korea: IEEE Service Center, 2001, pp. 971–978.

[2] H. A. Abbass, "An economical cognitive approach for bi-objective optimization using bliss points, visualization, and interaction," *Soft Computing*, vol. 10, no. 8, pp. 687–698, 2006.

[3] T. Back, *Evolutionary Algorithms in Theory and Practice.* New York: Oxford University Press, 1996.

[4] S. Bleuler, M. Laumanns, L. Thiele, and E. Zitzler, "Pisa: A platform and programming language independent interface for search algorithms," in *Evolutionary Multi-Criterion Optimization*, ser. Lecture Notes in Computer Science, vol. 2632. Springer, 2003, pp. 494–508.

[5] J. Branke, *Evolutionary optimization in dynamic environments.* Massachusetts USA: Kluwer Academic Publishers, 2002.

[6] L. T. Bui, H. A. Abbass, M. Barlow, and A. Bender, "Robustness against the decision-maker's attitude to risk in problems with conflicting objectives," *IEEE Transactions on Evolutionary Computation*, vol. 16, no. 1, pp. 1–19, 2012.

[7] L. T. Bui, H. A. Abbass, and D. Essam, "Fitness inheritance for noisy evolutionary multi-objective optimization," in *Proceedings of The 7th Annual Conference on Genetic and Evolutionary Computation.* ACM, 2005, pp. 779–785.

[8] ——, "Local modelsan approach to distributed multi-objective optimization," *Computational Optimization and Applications*, vol. 42, no. 1, pp. 105–139, 2009.

[9] L. T. Bui, J. Liu, A. Bender, M. Barlow, S. Wesolkowski, and H. A. Abbass, "Dmea: a direction-based multiobjective evolutionary algorithm," *Memetic Computing*, vol. 3, no. 4, pp. 271–285, 2011.

[10] L. T. Bui and S. Alam, *Multi-Objective Optimization in Computational Intelligence: Theory and Practice: Theory and Practice.* IGI Global, 2008.

[11] L. T. Bui, Z. Michalewicz, E. Parkinson, and M. B. Abello, "Adaptation in dynamic environments: a case study in mission planning," *IEEE Transactions on Evolutionary Computation*, vol. 16, no. 2, pp. 190–209, 2012.

[12] E. Cantuz-Paz, *Efficient and Accurate Parallel Genetic Algorithms.* Boston, MA, USA: Kluwer, 2001.

[13] C. A. C. Coello, "Evolutionary multi-objective optimization: A historical view of the field," *IEEE Computational Intelligence Magazine*, vol. 1, no. 1, pp. 28–36, 2006.

[14] C. A. C. Coello, G. T. Pulido, and M. S. Lechuga, "Handling multiple objectives with particle swarm optimization." *IEEE Transactions one Evolutionary Computation*, vol. 8, no. 3, pp. 256–279, 2004.

[15] C. A. C. Coello, D. A. V. Veldhuizen, and G. B. Lamont, *Evolutionary Algorithms for Solving Multi-Objective Problems.* New york USA: Kluwer Academic Publishers, 2002.

[16] C. Coello, "EMOO repository," http://delta.cs.cinvestav.mx/ ccoello/EMOO/, [Accessed: 30/12/2017], 2017.

[17] C. A. C. Coello, "Recent results and open problems in evolutionary multiobjective optimization," in *Proceedings of Theory and Practice of Natural Computing*, ser. Lecture Notes in Computer Science, vol. 10687. Springer, 2017, pp. 3–21.

[18] J. L. Cohon, *Multi-objective Programming and Planning.* New York, USA: Academic Press, 1983.

[19] K. Deb, *Multiobjective Optimization using Evolutionary Algorithms.* New York: John Wiley and Son Ltd, 2001.

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

[21] K. Deb and H. Jain, "An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part i: Solving problems with box constraints." *IEEE Trans. Evolutionary Computation*, vol. 18, no. 4, pp. 577–601, 2014.

[22] K. A. DeJong, "An analysis of the behavior of a class of genetic adaptive systems." Ph.D. dissertation, University of Michigan, Ann Arbor, 1975.

[23] M. Ehrgott, *Multicriteria Optimization.* Berlin, Germany: Springer; 2 edition, 2005.

[24] P. C. Fishburn, "Lexicographic orders, utilities, and decision rules: A survey," *Management Science*, vol. 20, no. 11, pp. 1442–1471, 1974.

[25] C. Fonseca and P. Fleming, "Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization," in *Proceedings of the Fifth International Conference on Genetic Algorithms, San Mateo, California.* Morgan Kauffman Publishers, 1993, pp. 416–423.

[26] ——, "On the performance assessement and comparision of stochastic multiobjective optimizers," in *Parallel Problem Solving from Nature - PPSN IV, Lecture Notes in Computer Science*, H.-M. Voigt, W. Ebeling, I. Rechenberg, and H.-P. Schwefel, Eds. Berlin Germany: Springer Verlag, 1996, pp. 584–593.

[27] A. M. Geoffrion, J. S. Dyer, and A. Feinberg, "An interactive approach for multi-criterion optimization, with an application to the operation of an academic department," *Management Science*, vol. 19, no. 4, pp. 357–368, 1972.

[28] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning.* Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1989.

[29] Y. Y. Haimes, . S. Lasdon, and D. A. Wismer, "On a bicriteriion formulation of the problem of integrated system identification and system optimization," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 1, no. 3, pp. 296–297, 1971.

[30] Z. He and G. G. Yen, "Many-objective evolutionary algorithms based on coordinated selection strategy," *IEEE Transactions on Evolutionary Computation*, vol. 21, no. 2, pp. 220–233, 2017.

[31] J. Horn, "Multicriteria decision making," in *Handbook of Evolutionary Computation*, T. Back, D. B. Gogel, and Z. Michalewicz, Eds. Institute of Physics Publishing, 1997.

[32] J. Horn, N. Nafpliotis, and D. Goldberg, "A niched pareto genetic algorithm for multiobjective optimization," in *Proceedings of the First IEEE Conference on Evolutionary Computation*, vol. 1. IEEE World Congress on Computational Intelligence, Piscataway, New Jersey, 1994, pp. 82–87.

[33] J. P. Ignizio, "Generalized goal programming: An overview," *Computer and Operations Research*, vol. 10, no. 4, pp. 277–289, 1974.

[34] S. Jiang and S. Yang, "A strength pareto evolutionary algorithm based on reference direction for multiobjective and many-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 21, no. 3, pp. 329–346, 2017.

[35] Y. Jin, "A comprehensive survey of fitness approximation in evolutionary computation," *Soft Computing*, vol. 9, no. 1, pp. 3–12, 2005.

[36] S. Kavakeb, T. T. Nguyen, Z. Yang, and I. Jenkinson, "Evolutionary fleet sizing in static and uncertain environments with shuttle transportation tasks-the case studies of container terminals [application notes]," *IEEE Computational Intelligence Magazine*, vol. 11, no. 1, pp. 55–69, 2016.

[37] V. Khare, X. Yao, and K. Deb, "Performance scaling of multi-objective evolutionary algorithms," in *Proceedings of the Evolutionary Multi-objective Optimization Conference*, ser. Lecture Notes in Computer Science, vol. 2632. Springer, 2003, pp. 376–390.

[38] J. Knowles and D. Corne, “Approximating the nondominated front using the pareto archibed evoltion strategy,” *Evolutionary Computation*, vol. 8, no. 2, pp. 149–172, 2000.

[39] ——, “Approximating the nondominated front using the pareto archived evolution strategy,” *Evolutionary Computation*, vol. 8, no. 2, pp. 149–172, 2000.

[40] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*, 3rd ed. London, UK: Springer-Verlag, 1996.

[41] K. Miettinen, “On the metholodlogy of multi-objective optimization with applications,” Ph.D. dissertation, Report No. 60, University of Jyväskylä, Department of Mathematics, 1994.

[42] ——, *Nonlinear Multiobjective Optimization.* Boston, USA: Kluwer Academic Publishers, 1999.

[43] V. Nissen and J. Propach, “On the robustness of population-based versus point-based optimization in the presence of noise,” *IEEE Transactions on Evolutionary Computation*, vol. 2, no. 3, pp. 107–119, 1998.

[44] V. Pareto, *Cours d'èconomie politique professè à l'universitè de Lausanne.* F. Rouge, Laussanne, 1896, vol. 1,2.

[45] F. Rothlauf and D. E. Goldberg, *Representations for Genetic and Evolutionary Algorithms.* Physica-Verlag, 2002.

[46] G. Rudolph and A. Agapie, “Convergence properties of some multi-objective evolutionaryalgorithms,” in *Proceedings of the Congress on Evolutionary Computation.* IEEE Press, 2000, pp. 1010–1016.

[47] R. P. Runyon, K. A. Coleman, and D. Pittenger, *Fundamentals of Behavioral Statistics.* Boston, MA: McGraw-Hill, 1996.

[48] J. Schaffer, “Multiple objective optimization with vector evaluated genetic algorithms,” in *Genetic Algorithms and their Applications:Proceedings of the First International Conference on Genettic Algorithms*, Hillsdale, New Jersey, 1985, pp. 93–100.

[49] J. Schott, “Fault tolerant design using single and multicriteria genetic algorithm optimization.” Master's thesis, Department of Aeronaustics and Astronautics, Massachusets Institute of Technology, 1995.

[50] R. E. Steuer, *Multiple Criteria Optimization: Theory, Computation, and Applications.* John Wiley & Sons, Inc., 1986.

[51] R. Storn and K. Price, “Differential evolution - a simple and efficient adaptive scheme for global optimization over continuous spaces. technical report tr-95-012,” ICSI, Tech. Rep., 1995.

[52] K. C. Tan, Y. J. Yang, and C. K. Goh, “A distributed cooperative coevolutionary algorithm for multi-objective optimization,” *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 5, pp. 527–549, 2006.

[53] A. Torn and A. Zilinskas, *Global Optimization.* Springer-Verlag, 1989.

[54] D. A. V. Veldhuizen, J. B. Zydallis, and G. B. Lamont, “Considerations in engineering parallel multiobjective evolutionary algorithms,” *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 2, pp. 144–173, 2003.

[55] D. Veldhuizen, "Multiobjective evolutionary algorithms: Classifications, analyses, and new innovation," Ph.D. dissertation, Department of Electrical Engineering and Computer Engineering, Airforce Institue of Technology, Ohio, 1999.

[56] S. Watanabe, M. Ito, and K. Sakakibara, "A proposal on a decomposition-based evolutionary multiobjective optimization for large scale vehicle routing problems," in *Evolutionary Computation (CEC), 2015 IEEE Congress on.* IEEE, 2015, pp. 2581–2588.

[57] L. While, L. Bradstreet, L. Barone, and P. Hingston, "Heuristics for optimising the calculation of hypervolume for multi-objective optimization problems," in *IEEE Congress on Evolutionary Computation*, vol. 3. IEEE Press, 2005, pp. 2225–2232.

[58] R. L. While, P. Hingston, L. Barone, and S. Huband, "A faster algorithm for calculating hypervolume." *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 1, pp. 29–38, 2006.

[59] A. P. Wierzbiki, *Optimization techniques, Part 1.* Lecture Notes in Control and Information Sciences 22, Springer-Verlag, Berlin, 1980, ch. A Metholodlogical Guide to Multiobjective Optimization, pp. 99–123.

[60] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 67–82, 1997.

[61] M. Zeleny, *Multiple Criteria Decision Making.* University of South Carolina Press, 1973, ch. Compromise Programming, pp. 262–301.

[62] Q. Zhang and H. Li, "Moea/d: A multiobjective evolutionary algorithm based on decomposition," *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 6, pp. 712–731, 2007.

[63] X. Zhang, Y. Tian, and Y. Jin, "A knee point-driven evolutionary algorithm for many-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 19, no. 6, pp. 761–776, 2015.

[64] H. Zille, H. Ishibuchi, S. Mostaghim, and Y. Nojima, "Mutation operators based on variable grouping for multi-objective large-scale optimization," in *Computational Intelligence (SSCI), 2016 IEEE Symposium Series on.* IEEE, 2016, pp. 1–8.

[65] E. Zitzler, M. Laumanns, and L. Thiele, "SPEA2: Improving the strength pareto evolutionary algorithm for multiobjective optimization," in *Evolutionary Methods for Design Optimization and Control with Applications to Industrial Problems*, K. C. Giannakoglou, D. T. Tsahalis, J. Periaux, K. D. Papailiou, and T. Fogarty, Eds. International Center for Numerical Methods in Engineering (Cmine), 2001, pp. 95–100.

[66] E. Zitzler and L. Thiele, "Multi-objective optimization using evolutionary algorithms - a comparative case study," in *Parallel Problem Solving from Nature*, ser. Lecture Notes in Computer Science, vol. 1498. Springer, 1998, pp. 292–304.

[67] E. Zitzler, L. Thiele, and K. Deb, "Comparision of multiobjective evolutionary algorithms: Emprical results," *Evolutionary Computation*, vol. 8, no. 1, pp. 173–195, 2000.

Received on January 07, 2018
Revised on February 05, 2018