The effects of asymmetric neighborhood assignment in the MOEA/D algorithm

Krzysztof Michalak

Department of Information Technologies, Institute of Business Informatics, Wroclaw University of Economics, Wroclaw, Poland

Abstract

The Multiobjective Evolutionary Algorithm Based on Decomposition (MOEA/D) is a very efficient multiobjective evolutionary algorithm introduced in recent years. This algorithm works by decomposing a multiobjective optimization problem to many scalar optimization problems and by assigning each specimen in the population to a specific subproblem. The MOEA/D algorithm transfers information between specimens assigned to the subproblems using a neighborhood relation.

In this paper it is shown that parameter settings commonly used in the literature cause an asymmetric neighbor assignment which in turn affects the selective pressure and consequently causes the population to converge asymmetrically. The paper contains theoretical explanation of how this bias is caused as well as an experimental verification. The described effect is undesirable, because a multiobjective optimizer should not introduce asymmetries not present in the optimization problem. The paper gives some guidelines on how to avoid such artificial asymmetries.

Keywords:
multiobjective optimization, evolutionary algorithms, MOEA/D algorithm, selective pressure

1. Introduction

The Multiobjective Evolutionary Algorithm Based on Decomposition (MOEA/D) proposed in [18] is a multiobjective evolutionary algorithm competitive to other well-known multiobjective optimization evolutionary algorithms (MOEAs) such as NSGA-II [6] or SPEA-2 [21]. The MOEA/D was found to outperform other algorithms not only in the case of benchmark problems [18, 11], but also in the case of combinatorial problems [15] and various applications to real-life problems [7, 10, 1, 17].

The class of multiobjective optimization problems (MOPs) can be formalized as follows:

$$\begin{aligned} &\text{minimize } F(x) = (f_1(x), \ldots, f_m(x)) \\ &\text{subject to } x \in \Omega, \end{aligned} \tag{1}$$

where:

Ω - the decision space,

m - the number of objectives.

Solutions can be compared to each other using the Pareto domination relation formally defined as follows [14, 3]. Given two points x_1 and x_2 in the decision space Ω for which:

$$\begin{aligned} F(x_1) &= (f_1(x_1), \ldots, f_m(x_1)) \\ F(x_2) &= (f_1(x_2), \ldots, f_m(x_2)) \end{aligned} \tag{2}$$

we say that x_1 *dominates* x_2 ($x_1 \succ x_2$) iff:

$$\begin{aligned} &\forall j \in \{1, \ldots, m\} : f_j(x_1) \leq f_j(x_2) \\ &\exists j \in \{1, \ldots, m\} : f_j(x_1) < f_j(x_2) \end{aligned} \tag{3}$$

A solution x is said to be *nondominated* (*Pareto optimal*) iff:

$$\neg \exists x' \in \Omega : x' \succ x. \tag{4}$$

Usually, there is no single $x_0 \in \Omega$ which minimizes all f_j, $j \in \{1, \ldots, m\}$ simultaneously, so it is not possible to simply choose one "the best" solution. Therefore, solving a multiobjective optimization problem usually means finding a *Pareto set* of nondominated solutions in the decision space Ω or a *Pareto front* of points in the objective space R^m that correspond to solutions from the Pareto set. Some of the algorithms, for example the commonly-used NSGA-II algorithm, use the notion of Pareto domination directly to decide which specimens are promoted to the next generation and which are replaced.

Conversely, the main idea employed in the MOEA/D algorithm is to decompose the multiobjective problem to a number of scalar subproblems. This task is done by assigning a weight vector $\lambda = (\lambda_1, \ldots, \lambda_m)$, $\lambda_j \geq 0$ for all $j \in \{1, \ldots, m\}$, $\sum_{j=1}^{m} \lambda_j = 1$ to each specimen in the population. The evaluation of a specimen is then performed using one of the aggregation approaches. Two the most commonly used aggregation approaches are the weighted sum approach and the Tchebycheff approach. In the weighted sum approach the following formulation of a scalar problem is used:

$$\begin{aligned} &\text{minimize } g^{ws}(x|\lambda) = \textstyle\sum_{j=1}^{m} \lambda_j f_j(x) \\ &\text{subject to } x \in \Omega, \end{aligned} \tag{5}$$

where:

λ - the weight vector assigned to the specimen.

The Tchebycheff approach uses a reference point $z^* = (z_1^*, \ldots, z_m^*)$ which is composed of the best attainable values along all the objectives: $z_j^* = min\{f_j(x) : x \in \Omega\}$. In case the

Email address: krzysztof.michalak@ue.wroc.pl (Krzysztof Michalak)

best possible values are not known in advance the MOEA/D algorithm offers a possibility of calculating the z^* value after each generation as a minimum of objectives of all the solutions found so far. The formulation of a scalar problem in the Tchebycheff aggregation approach is as follows:

$$\begin{aligned} &\text{minimize } g^{te}(x|\lambda, z^*) = \max_{1\le j\le m} \{\lambda_j |f_j(x) - z_j^*|\} \\ &\text{subject to } x \in \Omega, \end{aligned} \tag{6}$$

where:

λ - the weight vector assigned to the specimen,

z^* - the reference point.

In the MOEA/D algorithm the following way of generating weight vectors is used. First, a parameter H (a step size used for generating weight vectors) is set to a predefined value. Then, all weight vectors (i.e. vectors satisfying $\sum_{j=1}^{m} \lambda_j = 1$) are generated, such that:

$$\lambda_j \in \left\{ \frac{0}{H}, \frac{1}{H}, \ldots, \frac{H}{H} \right\}. \tag{7}$$

Therefore, the total number of weight vectors (and so the population size) is:

$$N = C_{H+m-1}^{m-1}. \tag{8}$$

For a bi-objective optimization problem ($m = 2$) the weight vector generation procedure described above results in a set of vectors $\left\{\lambda^{(i)}\right\}$, $i \in \{0, \ldots, H\}$, where:

$$\lambda^{(i)} = \left[\frac{i}{H}, \frac{H-i}{H} \right]. \tag{9}$$

Obviously:

$$\forall i \in \{0, \ldots, H\} : \lambda^{(i)} + \lambda^{(H-i)} = [1, 1] \ . \tag{10}$$

For example, if we set $H = 99$ we get a population size $N = 100$ with the weight vector set:

$$\left\{\lambda^{(0)}, \ldots, \lambda^{(H)}\right\} = \left\{[0, 1], [1/99, 98/99], \ldots, [1, 0]\right\}. \tag{11}$$

Another important concept used in the MOEA/D algorithm is subproblem neighborhood. The neighborhood of each weight vector $\lambda^{(i)}$ (and therefore also the corresponding specimen) is defined as a set $B(i)$ of T weight vectors that are closest to $\lambda^{(i)}$ with respect to the Euclidean distance calculated in the m-dimensional space of vector weights. The neighborhood $B(i)$ is used in MOEA/D to select parents for a new specimen that, if it generates a better value of the aggregated objective function (for example $g^{ws}(x|\lambda^{(i)})$ or $g^{te}(x|\lambda^{(i)}, z^*)$), may replace the old specimen assigned to the weight vector $\lambda^{(i)}$. Also, the newly found solution to the i-th subproblem may replace solutions corresponding to weight vectors from the $B(i)$ if it improves the aggregated objective for these solutions. Note, that by definition each vector $\lambda^{(i)}$ belongs to its own neighborhood $B(i)$. This is not just a side effect of using the Euclidean distance in the weight space to define the neighborhood, but it is also important for the working of the algorithm, because the current solution of a scalar subproblem is expected to significantly participate in the search for better solutions of this subproblem.

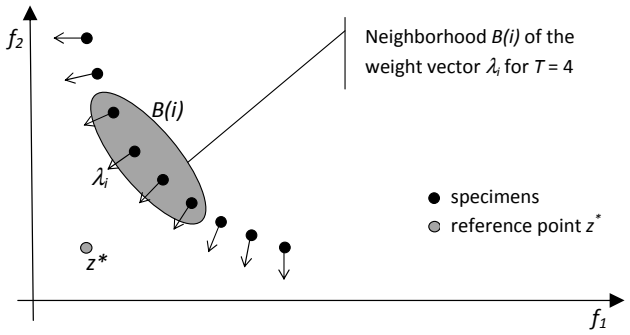

Figure 1: An overview of concepts involved in the working of the MOEA/D algorithm in the case of minimization in a bi-objective problem.

The overview of concepts involved in the working of the MOEA/D algorithm in the case of minimization in a bi-objective problem is presented in Figure 1.

From the definition of neighborhood $B(i)$ it is clear that when the neighborhood size T is set to an even number the neighborhood $B(i)$ is not symmetrically spread around the weight vector $\lambda^{(i)}$ (cf. Figure 1).

In this paper the effects of asymmetric neighborhood assignment are studied. The interest in such effects is motivated by the fact that in a number of papers in which the MOEA/D algorithm is studied even values of the neighborhood size T are used which may cause a tendency for the algorithm to converge asymmetrically. Table 1 presents values of the T parameter found in a number of papers in which the value of this parameter is specified. In the table bibliographic references are also given. In some papers, the value of the T parameter is not set directly, but instead it is defined as a fraction of the population size N. In [12] the parameters are set to $T = 0.1N$, $N = 100$ and in [9] the parameters are set to $T = 0.1N$, $N = 200$, 600 and 1000. Clearly, both parameter settings lead to even values of the parameter T. Unfortunately, many other papers do not mention the value of the T parameter.

Table 1: Values of the neighborhood size T found in the literature

Paper	Value of the T parameter
Multi-objective energy-efficient dense deployment in Wireless Sensor Networks using a hybrid problem-specific MOEA-D [10]	2
Community detection in networks by using multiobjective evolutionary algorithm with decomposition [8]	10
MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition [18]	10 for the Multiobjective Knapsack Problem (MOKP) 20 for continuous MOPs
Evolutionary computation of multi-band antenna using multi-objective evolutionary algorithm based on decomposition [7]	12
Multiobjective Optimization Problems with Complicated Pareto Sets, MOEA-D and NSGA-II [11]	20
Comparison between MOEA/D and NSGA-II on the Multi-Objective Travelling Salesman Problem [15]	25
Using computational intelligence for large scale air route networks design [1]	30
Decomposition-based modern metaheuristic algorithms for multi-objective optimal power flow - A comparative study [13]	30

2. Weight vector neighborhood properties

Weight vectors $\lambda^{(i)}$, $i \in \{1, \ldots, N\}$ determine the importance of objectives f_j, $j \in \{1, \ldots, m\}$ in each of the scalar subproblems. The overall importance of the objectives can therefore be compared by calculating a vector which contains the average weights that the objectives have in all the neighborhoods defined for all the subproblems:

$$\bar{\lambda} = \frac{1}{(H+1)T} \sum_{i=0}^{H} \sum_{\lambda \in B(i)} \lambda. \tag{12}$$

For neighborhoods generated for a bi-objective optimization problem neighborhoods $B(i)$ have the form of:

$$B(i) = \left\{\lambda^{(l_i)}, \ldots, \lambda^{(h_i)}\right\}, \tag{13}$$

where:
$l_i = i - \left\lfloor \frac{T-1}{2} \right\rfloor$,
$h_i = i + \left\lceil \frac{T-1}{2} \right\rceil$.

Because weight vector indices must be from the set $0, \ldots, H$ when $l_i < 0$ we set $B(i) = \{\lambda^{(0)}, \ldots, \lambda^{(T-1)}\}$ and when $h_i > H$ we set $B(i) = \{\lambda^{(H-T+1)}, \ldots, \lambda^{(H)}\}$. Considering the above we get:

$$B(i) = \begin{cases} \{\lambda^{(0)}, \ldots, \lambda^{(T-1)}\} & \text{for } i \leq \left\lfloor \frac{T-1}{2} \right\rfloor \\ \{\lambda^{(i-\lfloor \frac{T-1}{2} \rfloor)}, \ldots, \lambda^{(i+\lceil \frac{T-1}{2} \rceil)}\} & \text{otherwise} \\ \{\lambda^{(H-T+1)}, \ldots, \lambda^{(H)}\} & \text{for } i > H - \left\lceil \frac{T-1}{2} \right\rceil \end{cases} \tag{14}$$

Next, we can write Equation (14) separately for even and odd values of the neighborhood size T and substitute $B(i)$ in Equation (12).

1° $T = 2k$, $k \in \mathbb{N}$

$$B(i) = \begin{cases} \{\lambda^{(0)}, \ldots, \lambda^{(T-1)}\} & \text{for } i \leq k-1 \\ \{\lambda^{(i-k+1)}, \ldots, \lambda^{(i+k)}\} & i \in \{k, H-k\} \\ \{\lambda^{(H-T+1)}, \ldots, \lambda^{(H)}\} & \text{for } i > H-k \end{cases} \tag{15}$$

Substituting $B(i)$ from Equation (15) in Equation (12) we get the vector of average weights:

$$\bar{\lambda} = \frac{1}{(H+1)T} \sum_{i=0}^{H} \sum_{\lambda \in B(i)} \lambda \tag{16}$$

$$= \frac{1}{(H+1)T} \left(\sum_{i=0}^{k-1} \sum_{\lambda \in B(i)} \lambda + \sum_{i=k}^{H-k} \sum_{\lambda \in B(i)} \lambda + \sum_{i=H-k+1}^{H} \sum_{\lambda \in B(i)} \lambda \right) \tag{17}$$

$$= \frac{1}{(H+1)T} \left(\sum_{i=0}^{k-1} \sum_{j=0}^{T-1} \lambda^{(j)} + \sum_{i=k}^{H-k} \sum_{j=i-k+1}^{i+k} \lambda^{(j)} + \sum_{i=H-k+1}^{H} \sum_{j=H-T+1}^{H} \lambda^{(j)} \right) \tag{18}$$

$$= \frac{1}{(H+1)T} \left(k \sum_{j=0}^{T-1} \lambda^{(j)} + \sum_{i=k}^{H-k} \sum_{j=i-k+1}^{i+k} \lambda^{(j)} + k \sum_{j=H-T+1}^{H} \lambda^{(j)} \right) \tag{19}$$

$$= \frac{1}{(H+1)T} \left(k \sum_{j=0}^{T-1} \lambda^{(j)} + \sum_{i=k}^{H-k} \sum_{j=i-k+1}^{i+k} \lambda^{(j)} + k \sum_{j=0}^{T-1} \lambda^{(H-j)} \right) \tag{20}$$

$$= \frac{1}{(H+1)T} \left(k \sum_{j=0}^{T-1} \left(\lambda^{(j)} + \lambda^{(H-j)} \right) + \sum_{i=k}^{H-k} \sum_{j=i-k+1}^{i+k} \lambda^{(j)} \right) \tag{21}$$

Using Equation (10) we get:

$$\bar{\lambda} = \frac{k}{(H+1)}[1,1] + \frac{1}{(H+1)T}\left(\sum_{i=k}^{H-k}\sum_{j=i-k+1}^{i+k}\lambda^{(j)}\right) \tag{22}$$

$$= \frac{k}{(H+1)}[1,1] + \frac{1}{(H+1)T}\left(\sum_{i=k}^{H-k}\sum_{j=i-k+1}^{i+k}\left[\frac{j}{H},\frac{H-j}{H}\right]\right) \tag{23}$$

$$= \frac{1}{2H(H+1)}\left[H^2+2H-2k+1, H^2+2k+1\right] \tag{24}$$

Since $H \geq T = 2k$ (the neighborhood size is smaller than the size of population) so $H - k \geq k$ and therefore from Equation (24) we get that the average weight that the first objective has in all the neighborhoods defined for all the subproblems is not smaller than the weight for the second one. The equality holds only when $N - 1 = H = T = 2k$ i.e. when the neighborhood size is almost equal to the size of the entire population, which is rarely, if ever, used. In all the other cases (i.e. $T < H$) the average weight that the first objective has is larger than the weight the second objective has. The difference between the average weights for both objectives $\bar{\lambda}_1 - \bar{\lambda}_2$ equals:

$$\bar{\lambda}_1 - \bar{\lambda}_2 = \frac{H^2+2H-2k+1}{2H(H+1)} - \frac{H^2+2k+1}{2H(H+1)} \tag{25}$$

$$= \frac{H-2k}{H(H+1)} \tag{26}$$

Therefore, for a fixed value of the parameter $T = 2k$ the difference gets smaller with the increasing H.

2° $T = 2k+1, k \in \mathbb{N}$

$$B(i) = \begin{cases} \{\lambda^{(0)}, \ldots, \lambda^{(T-1)}\} & \text{for } i \leq k \\ \{\lambda^{(i-k)}, \ldots, \lambda^{(i+k)} & i \in \{k+1, H-k-1\} \\ \{\lambda^{(H-T+1)}, \ldots, \lambda^{(H)}\} & \text{for } i \geq H-k \end{cases} \tag{27}$$

Substituting $B(i)$ from Equation (27) in Equation (12) we get the vector of average weights:

$$\bar{\lambda} = \frac{1}{(H+1)T}\sum_{i=0}^{H}\sum_{\lambda\in B(i)}\lambda \tag{28}$$

$$= \frac{1}{(H+1)T}\left(\sum_{i=0}^{k}\sum_{\lambda\in B(i)}\lambda + \sum_{i=k+1}^{H-k-1}\sum_{\lambda\in B(i)}\lambda + \sum_{i=H-k}^{H}\sum_{\lambda\in B(i)}\lambda\right) \tag{29}$$

$$= \frac{1}{(H+1)T}\left(\sum_{i=0}^{k}\sum_{j=0}^{T-1}\lambda^{(j)} + \sum_{i=k+1}^{H-k-1}\sum_{j=i-k}^{i+k}\lambda^{(j)} + \sum_{i=H-k}^{H}\sum_{j=H-T+1}^{H}\lambda^{(j)}\right) \tag{30}$$

$$= \frac{1}{(H+1)T}\left((k+1)\sum_{j=0}^{T-1}\lambda^{(j)} + \sum_{i=k+1}^{H-k-1}\sum_{j=i-k}^{i+k}\lambda^{(j)} + (k+1)\sum_{j=H-T+1}^{H}\lambda^{(j)}\right) \tag{31}$$

$$= \frac{1}{(H+1)T}\left((k+1)\sum_{j=0}^{T-1}\lambda^{(j)} + \sum_{i=k+1}^{H-k-1}\sum_{j=i-k}^{i+k}\lambda^{(j)} + (k+1)\sum_{j=0}^{T-1}\lambda^{(H-j)}\right) \tag{32}$$

$$= \frac{1}{(H+1)T}\left((k+1)\sum_{j=0}^{T-1}\left(\lambda^{(j)} + \lambda^{(H-j)}\right) + \sum_{i=k+1}^{H-k-1}\sum_{j=i-k}^{i+k}\lambda^{(j)}\right) \tag{33}$$

Using Equation (10) we get:

$$\bar{\lambda} = \frac{k+1}{(H+1)}[1,1] + \frac{1}{(H+1)T}\left(\sum_{i=k+1}^{H-k-1}\sum_{j=i-k}^{i+k}\lambda^{(j)}\right) \tag{34}$$

$$= \frac{k+1}{(H+1)}[1,1] + \frac{1}{(H+1)T}\left(\sum_{i=k+1}^{H-k-1}\sum_{j=i-k}^{i+k}\left[\frac{j}{H},\frac{H-j}{H}\right]\right) \tag{35}$$

$$= \frac{k+1}{(H+1)}[1,1] + \frac{H-2k-1}{2(H+1)}[1,1] \tag{36}$$

$$= \frac{H+1}{2(H+1)}[1,1] \tag{37}$$

$$= \left[\frac{1}{2},\frac{1}{2}\right] \tag{38}$$

From Equation (38) we get that when the neighborhood size T is an odd number the average weights that both objectives have in all the neighborhoods defined for all the subproblems are the same. Therefore, we might expect that the first objective will be treated by the algorithm as more important than the second one when the neighborhood size T is even, but the objectives should be treated equally in the case when T is odd. However, it needs to be verified if the inequality in average weights significantly influences the symmetry of the final approximation of the Pareto front produced by the algorithm.

3. Influence of the neighborhood size on solutions found by the algorithm

In Section 2 it was shown that for even values of the neighborhood size T the average value of weights belonging to all neighborhoods is different for each of the two objectives. This effect is not present when the neighborhood size T is an odd number. This theoretical result is not enough to conclude that the asymmetry of neighborhood assignment influences the characteristics of the approximation of the Pareto front found by the algorithm. The final result depends not only on information transfer between specimens within neighborhoods, but also on the working of genetic operators (mutation and crossover) and also on possible tendencies introduced by the objective functions if the problem itself is not symmetric. This complexity makes it very hard to predict theoretically how the algorithm will behave when optimizing a given problem.

In order to investigate the influence of neighborhood assignment on the symmetry of the approximation of the Pareto fronts found by the algorithms an experimental verification was performed. To remove any possible influence of the optimization problem itself as well as the influence of the initial population distribution the experimental setup was carefully prepared to avoid any asymmetries other than those in the neighborhood assignment.

The optimization problem

There are many benchmark problems used in the scientific community for investigating multiobjective optimization problems, for example the ZDT test problem set [20]. Unfortunately, such benchmark problems have asymmetries that would make it difficult to distinguish potential effects of the uneven

selective pressure. Therefore, a dedicated test problem is proposed in this paper.

A geometric representation of the test problem used in experiments is as follows. First, the dimensionality d of the search space is selected. The search space is a d-dimensional cube $\Omega = [0,1]^d$. In this cube two points are located: one with all coordinates equal to $\frac{1}{4}$ and the other one with all coordinates equal to $\frac{3}{4}$:

$$A = \left[\frac{1}{4}, \ldots, \frac{1}{4}\right] \in R^d \tag{39}$$

$$B = \left[\frac{3}{4}, \ldots, \frac{3}{4}\right] \in R^d \tag{40}$$

For a given point $x_0 \in [0,1]^d$ the two objectives are defined as distances to points A and B respectively. Clearly, the problem of minimizing both distances is a bi-objective problem with contradictory objectives. Formally, the problem can be stated as follows:

$$\begin{array}{l} \text{minimize } F(x) = (\|x - A\|, \|x - B\|) \\ \text{subject to } x \in [0,1]^d \quad , \end{array} \tag{41}$$

which is equal to:

$$\begin{array}{l} \text{minimize } F(x) = (f_1(x), f_2(x)) \\ \text{subject to } x \in [0,1]^d \quad , \end{array} \tag{42}$$

where:

$$f_1(x) = \sqrt{\sum_{i=1}^{d} (x_i - A_i)^2} \tag{43}$$

$$f_2(x) = \sqrt{\sum_{i=1}^{d} (x_i - B_i)^2} \tag{44}$$

Clearly, the Pareto set of this optimization problem is the line segment connecting the points A and B in the decision space $[0,1]^d$. The length of this segment is $\frac{\sqrt{d}}{2}$. Therefore, the Pareto front is a line segment in the objective space R^2 with the equation:

$$y = -x + \frac{\sqrt{d}}{2}, \quad x \in \left[0, \frac{\sqrt{d}}{2}\right] \quad . \tag{45}$$

All of the points on this line segment are attainable, because positioning the solution anywhere on the AB line segment is allowed. The proposed test problem has several advantages presented below.

- The Pareto front is symmetric with respect to the $y = x$ line. This ensures that the objectives themselves do not introduce any preference for neither f_1 nor for f_2.
- The problem is invariant with respect to any reordering of dimensions of the decision space Ω. This ensures that no preference for any of the variables can cause an asymmetry in the final Pareto front.

Table 2: Algorithm parameters

Parameter name	Value
Number of generations (N_{gen})	250
Population size (N)	100
Weight vector step size ($H = N - 1$)	99
Crossover probability (P_{cross})	1.0
Crossover distribution index (η_{cross})	20
Mutation probability (P_{mut})	0.033
Mutation distribution index (η_{mut})	20
Decomposition method	Tchebycheff

- The transformation $m : R^d \to R^d$ defined as follows:

$$m(x) = x' \text{ s.t. } \forall i \in \{1, \ldots, d\} : x'_i = 1 - x_i \tag{46}$$

can be used to transform any solution x_0 with objective values $[f_1(x_0), f_2(x_0)]$ to another solution $x'_0 = m(x_0)$ with objective values $[f_1(x'_0) = f_2(x_0), f_2(x'_0) = f_1(x_0)]$. This property can be used to generate an initial population placed symmetrically with respect to $y = x$ line in the objective space R^2.

The algorithm

In the experiments the Multiobjective Evolutionary Algorithm Based on Decomposition (MOEA/D) was used. The algorithm was implemented according to steps detailed in [18]. For genetic operators the Simulated Binary Crossover (SBX) [4] and the polynomial mutation operator [5] were chosen, because these operators are designed to deal with real-valued problems and are commonly used in the literature for this task.

Test problem dimensionality was set to $d = 30$. The parameters of the algorithm were set in the same way as in [18] for continuous MOPs with the exception of the neighborhood size T which was changed in order to determine its influence on the symmetry of the final approximation of the Pareto front. Values of the algorithm parameters are summarized in Table 2.

Initial populations that were used for each run of the algorithm were built by first creating $N/2$ specimens with 30 coordinate values drawn from a uniform distribution on $[0,1]$. Then, the other $N/2$ specimens were initialized by transforming the coordinates of the specimens in the first half of the population using the transform $m(\cdot)$ given as Equation (46). Such initialization procedure ensures that the initial population is located symmetrically with respect to the $y = x$ line in the objective space.

The measurement of the symmetry of the Pareto front

The MOEA/D algorithm was run for $N_{gen} = 250$ generations. The algorithm stores an archive EP which contains the best approximation of the Pareto front found so far. At the end of each generation the symmetry of the approximation of the Pareto front that was found up to that generation was measured. Coordinates of the specimens in the EP archive were transformed by a 45° degree clockwise rotation:

$$x' = \frac{x}{\sqrt{2}} + \frac{y}{\sqrt{2}} \quad (47)$$

$$y' = -\frac{x}{\sqrt{2}} + \frac{y}{\sqrt{2}} \quad (48)$$

after which the Pareto front is oriented vertically with the center moved from $\left[\frac{\sqrt{d}}{4}, \frac{\sqrt{d}}{4}\right]$ to $\left[\frac{\sqrt{d}}{2\sqrt{2}}, 0\right]$. The average y' coordinate calculated over the EP set obtained at a generation g was used as a measure of location of the approximation of the Pareto front $S(g)$ obtained in that generation. The higher the value of the $S(g)$ measure the more is the approximation of the Pareto front offset towards the second objective.

Statistical verification of the results

In the experiments 100 independent runs were performed for each neighborhood size T with the weights assigned according to the procedure described in Section 2. As described in Section 2, when T is even such weights assignment causes the average weight corresponding to the first objective in all the neighborhoods defined for all the subproblems to be larger than the average weight corresponding to the second objective. The same number of runs was done with weight vectors sorted in the reversed order which resulted in the second objective having larger average weight. Values of the location measure were recorded for weight vectors sorted in one direction as $S(g)$, $g \in \{1, \ldots, N_{gen}\}$ and for weight vectors sorted in the opposite direction as $S'(g)$, $g \in \{1, \ldots, N_{gen}\}$.

From 100 pairs of $S(g)$ and $S'(g)$ values independently collected for each generation the differences $\Delta(g) = S'(g) - S(g)$ were calculated. The average of $\Delta(g)$ values from independent runs for a given g indicates if the change in weight vectors ordering influences the location of the approximation of the Pareto front obtained in generation g or not. The interpretation of values of $\Delta(g)$ is as follows:

$\Delta(g) > 0$	The ordering of weight vectors favors the first objective
$\Delta(g) = 0$	The ordering of weight vectors does not favor any of the objectives
$\Delta(g) < 0$	The ordering of weight vectors favors the second objective

To verify if the average value of $\Delta(g)$ is significantly different from 0, statistical tests were performed. First, the hypothesis that the distribution of $\Delta(g)$ values is normal was verified using the Kolmogorov-Smirnov test [2]. In all the cases there was no reason to reject the normality hypothesis. This is not surprising since the $\Delta(g)$ values are differences of $S'(g)$ and $S(g)$ values, and these are averages of the y' coordinate values of many solutions stored in the EP set.

To verify if $\Delta(g)$ values are significantly different from 0 a Student's t-test [16] was performed for each generation for the null hypothesis that the mean value of $\Delta(g)$ is actually 0. If the p-value obtained from this test was not higher than 0.05 the $\Delta(g)$ for a given generation g was assumed to be significantly different from 0. The interpretation of such situation is that when weight vectors are sorted in one direction the approximation of the Pareto front is significantly offset compared to when the weight vectors were sorted in the opposite direction.

4. Results

The experiments were aimed at finding out if the asymmetry in the neighborhood assignment influences the location of the approximation of the Pareto front attained by the algorithm. Therefore, various values for the neighborhood size T were tested. In the experiments the parameter T was set to all values in the range $[2, 20]$ and $T = 25$ and $T = 30$. This choice of values includes parameters presented in Table 1 and includes both even and odd values. For each value of the parameter T two series of 100 runs of the algorithm were done, one series with the ordering of weights presented in Section 2 and another series with the ordering reversed. The difference between the average location of points in the obtained Pareto fronts was calculated for each generation number $g \in \{1, \ldots, N_{gen}\}$ and the statistical significance of the results was verified as discussed in Section 3. Figures 2-22 present the average differences in locations of points plotted against the generation number g. Shades of gray are used to indicate the statistical significance: the values plotted in dark gray are statistically significant at the $\alpha = 0.05$ level. Values plotted in light gray are those for which the p-value for a hypothesis that the mean value of the displacement is actually 0 was higher than 0.05.

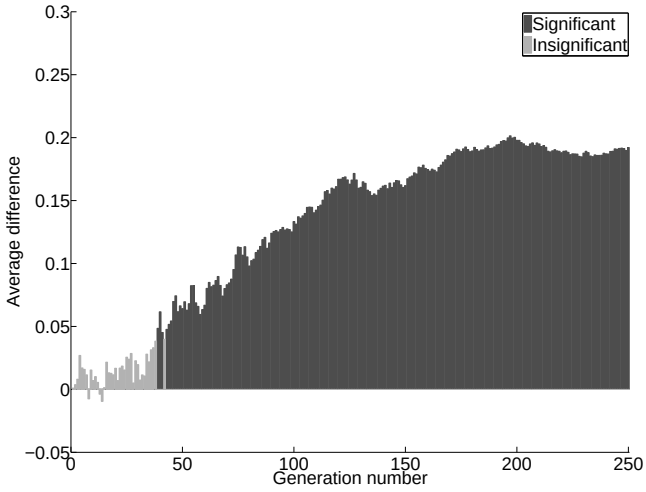

Figure 2: The average difference between locations of the Pareto front with normal and reversed weight vectors ordering for the neighborhood size $T = 2$.

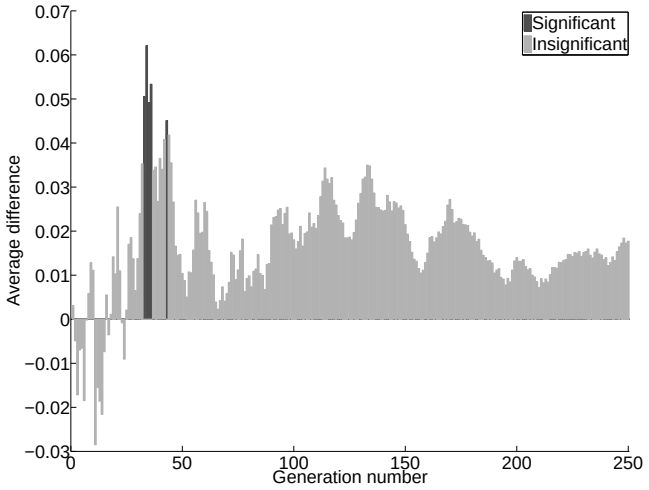

Figure 3: The average difference between locations of the Pareto front with normal and reversed weight vectors ordering for the neighborhood size $T = 3$.

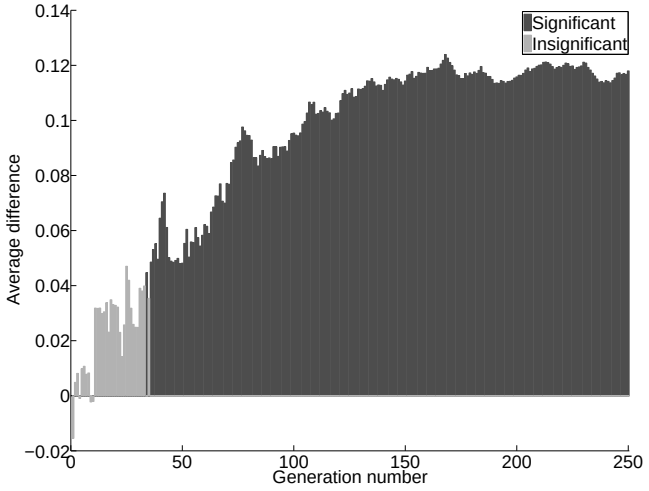

Figure 4: The average difference between locations of the Pareto front with normal and reversed weight vectors ordering for the neighborhood size $T = 4$.

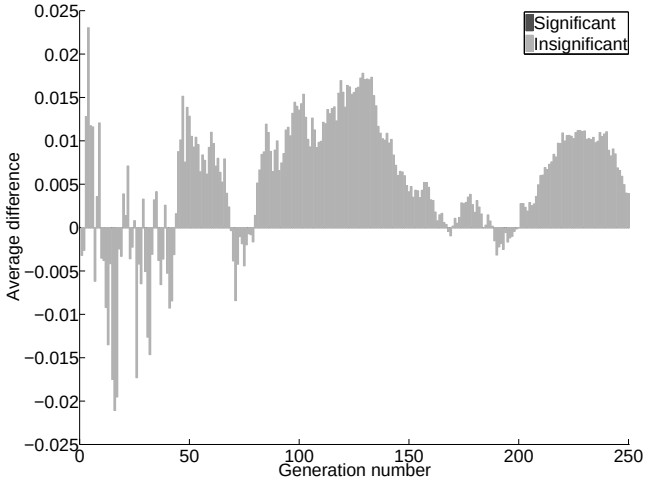

Figure 5: The average difference between locations of the Pareto front with normal and reversed weight vectors ordering for the neighborhood size $T = 5$.

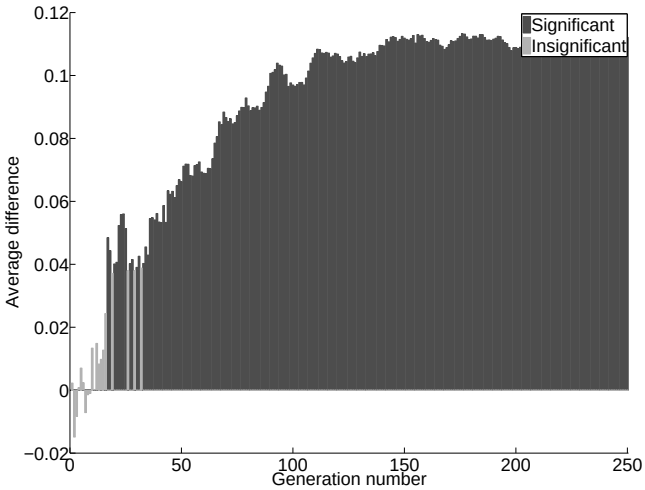

Figure 6: The average difference between locations of the Pareto front with normal and reversed weight vectors ordering for the neighborhood size $T = 6$.

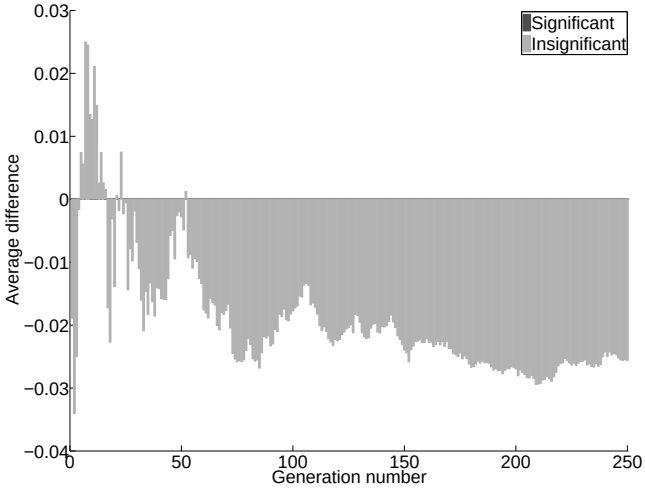

Figure 7: The average difference between locations of the Pareto front with normal and reversed weight vectors ordering for the neighborhood size $T = 7$.

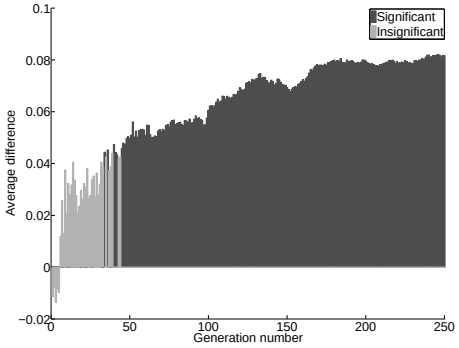

Figure 8: The average difference between locations of the Pareto front with normal and reversed weight vectors ordering for the neighborhood size $T = 8$.

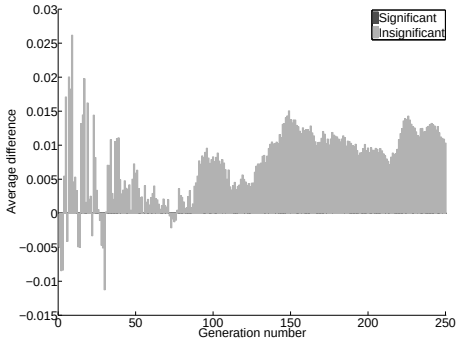

Figure 9: The average difference between locations of the Pareto front with normal and reversed weight vectors ordering for the neighborhood size $T = 9$.

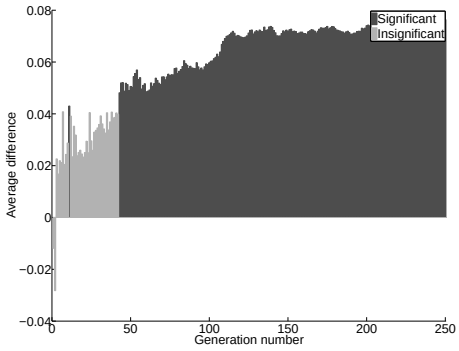

Figure 10: The average difference between locations of the Pareto front with normal and reversed weight vectors ordering for the neighborhood size $T = 10$.

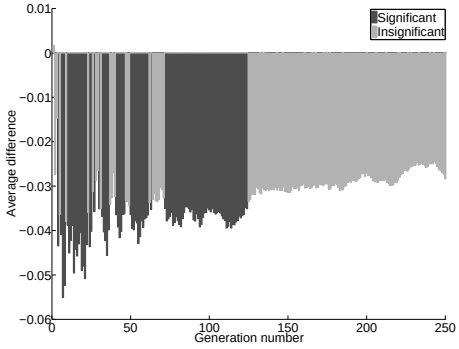

Figure 11: The average difference between locations of the Pareto front with normal and reversed weight vectors ordering for the neighborhood size $T = 11$.

As it can be seen from the graphs in the case of even values of the neighborhood size T the population eventually (after sufficiently many generations) converges to an approximation of the Pareto front that is offset in a direction affected by the ordering of the weight vectors. Statistical verification of the sample of results from 100 runs shows that the average displacement of the Pareto front is significantly different from 0. In the case of odd values of the parameter T the displacement was never found to be statistically significant after $N_{gen} = 250$ generations.

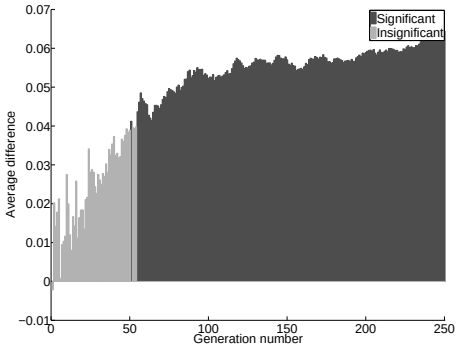

Figure 12: The average difference between locations of the Pareto front with normal and reversed weight vectors ordering for the neighborhood size $T = 12$.

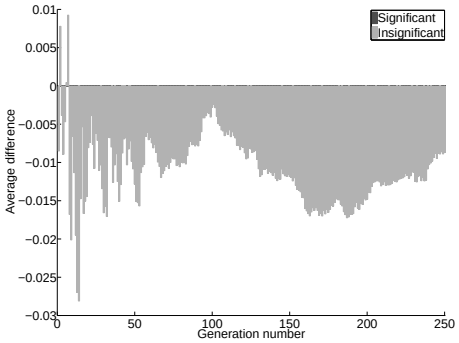

Figure 13: The average difference between locations of the Pareto front with normal and reversed weight vectors ordering for the neighborhood size $T = 13$.

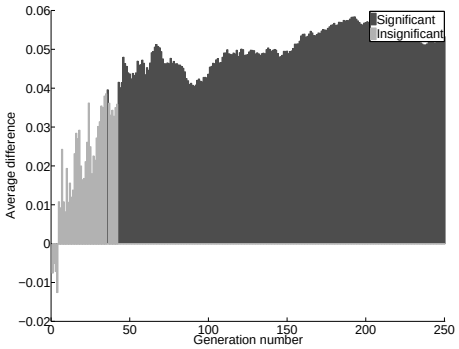

Figure 14: The average difference between locations of the Pareto front with normal and reversed weight vectors ordering for the neighborhood size $T = 14$.

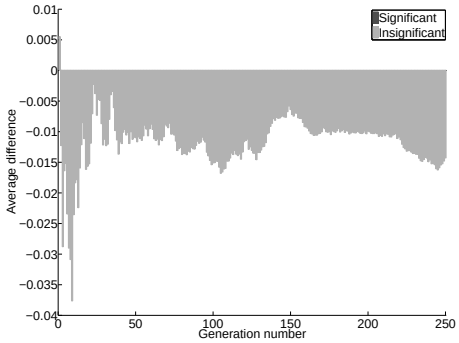

Figure 15: The average difference between locations of the Pareto front with normal and reversed weight vectors ordering for the neighborhood size $T = 15$.

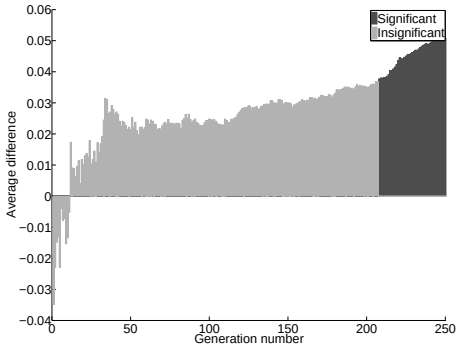

Figure 16: The average difference between locations of the Pareto front with normal and reversed weight vectors ordering for the neighborhood size $T = 16$.

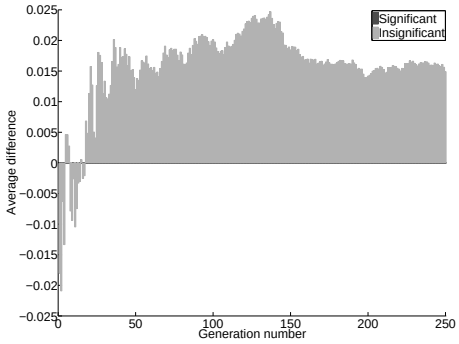

Figure 17: The average difference between locations of the Pareto front with normal and reversed weight vectors ordering for the neighborhood size $T = 17$.

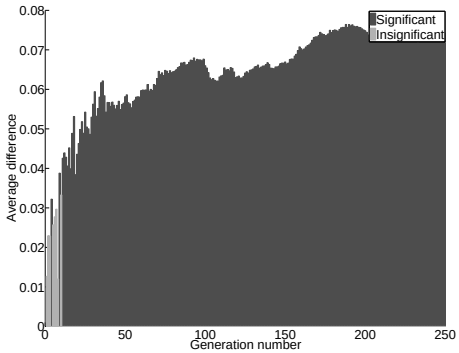

Figure 18: The average difference between locations of the Pareto front with normal and reversed weight vectors ordering for the neighborhood size $T = 18$.

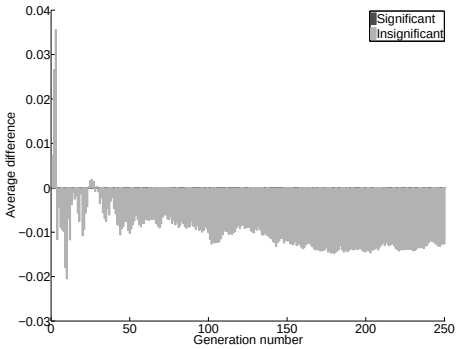

Figure 19: The average difference between locations of the Pareto front with normal and reversed weight vectors ordering for the neighborhood size $T = 19$.

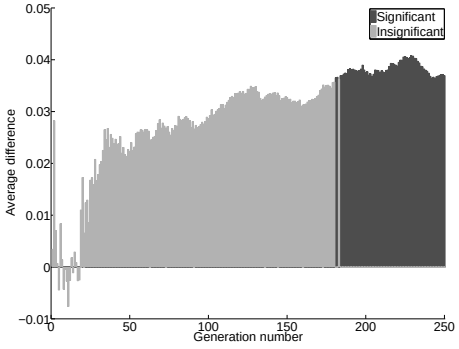

Figure 20: The average difference between locations of the Pareto front with normal and reversed weight vectors ordering for the neighborhood size $T = 20$.

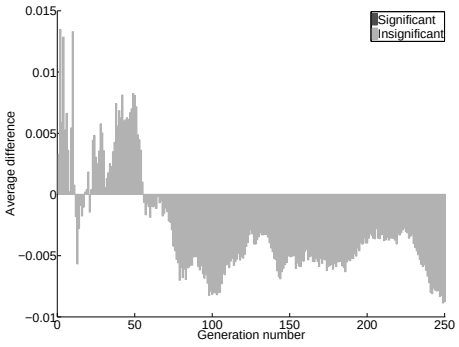

Figure 21: The average difference between locations of the Pareto front with normal and reversed weight vectors ordering for the neighborhood size $T = 25$.

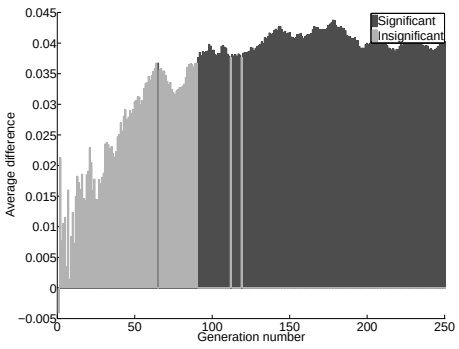

Figure 22: The average difference between locations of the Pareto front with normal and reversed weight vectors ordering for the neighborhood size $T = 30$.

5. Conclusion and further work

One of the very effective evolutionary multiobjective optimization algorithms MOEA/D works by decomposing the multiobjective problem into many scalar ones using aggregation of objectives. This aggregation is controlled by values of coordinates of weight vectors assigned to subproblems. Each specimen in the population is assigned a task of optimizing one scalar subproblem and the neighborhood relation defined over the set of weight vectors is used for parent selection and for distributing solutions to neighbor subproblems.

In this paper it is shown by theoretical analysis that for neighborhoods of even size the assignment of weight vectors causes one of the objectives to have larger average weight assigned that the other objective. In the experiments this effect was shown to influence the location of the final approximation of the Pareto front found by the algorithm.

The simplest way of avoiding the asymmetry introduced by the neighborhood assignment is to choose an odd value of the neighborhood T. Other factors may also influence this effect. For example, from Equation (26) it can be concluded that with the increase of the value of the H parameter (and therefore also the population size) the difference between the average weights assigned to the objectives gets smaller if the neighborhood size T does not change. Also, when the value of the T parameter approaches the size of the entire population the asymmetry should get smaller. The latter is of little importance though, because in most cases the size of the neighborhood is significantly smaller than the population size. Another effect that was found in some implementations of the MOEA/D algorithm is that due to rounding errors weight vectors equally distant from a given vector λ_i may be considered as placed at slightly different distances. While this is erroneous, it might actually diminish the effects discussed in this paper because these rounding errors serve as random disturbances that prevent the neighborhood asymmetries occurring around different λ_i's from building up in one direction. A practical recommendation arising from this observation might be to implement the MOEA/D algorithm in such a way that when a choice must be made between equally distant neighbors it would be made randomly. However, so far no implementation using this approach was found in the literature, including the reference implementation made available by the authors of the MOEA/D algorithm [19]. Therefore, in cases when the exact internal working of a given implementation of the MOEA/D algorithm is unknown or when it is hard to change the implementation to prevent the possible asymmetry (e.g. in the case of various toolboxes or libraries) the best option is to specify an odd number for the neighborhood size T.

In this paper the effects of an asymmetric neighborhood assignment on the working of the MOEA/D algorithm were discussed in the case of a bi-objective problem. Further work may include a generalization of the presented analysis to problems with more objectives. This task is, however, significantly harder because in such case the asymmetries may push the convergence of the algorithm in many different directions. This may make the effect harder to measure and verify statistically. Theoretical description of such situation is also more complicated than the one presented in the paper. An interesting direction would also be to determine the influence of neighborhood assignment in the case of problems with asymmetric Pareto fronts, but it can be expected that such effects are much harder to detect.

References

[1] Cai, K., Zhang, J., Zhou, C., Cao, X., & Tang, K. (2012). Using computational intelligence for large scale air route networks design. *Applied Soft Computing*, *12*, 2790–2800.

[2] Corder, G. W., & Foreman, D. I. (2009). Nonparametric statistics for non-statisticians: A step-by-step approach. chapter Chapter 2: Testing Data for Normality. (pp. 12–37). New Jersey: Wiley.

[3] Deb, K. (2001). *Multi-Objective Optimization Using Evolutionary Algorithms*. New York, NY, USA: John Wiley & Sons, Inc.

[4] Deb, K., & Agarwal, R. (1995). Simulated binary crossover for continuous search space. *Complex Systems*, *9*, 115–148.

[5] Deb, K., & Goyal, M. (1996). A combined genetic adaptive search (GeneAS) for engineering design. *Computer Science and Informatics*, *26*, 30–45.

[6] Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, *6*, 182–197.

[7] Ding, D., Wang, H., & Wang, G. (2011). Evolutionary computation of multi-band antenna using multi-objective evolutionary algorithm based on decomposition. In *Proceedings of the Second international conference on Information Computing and Applications* ICICA'11 (pp. 383–390). Berlin, Heidelberg: Springer-Verlag.

[8] Gong, M., Ma, L., Zhang, Q., & Jiao, L. (2012). Community detection in networks by using multiobjective evolutionary algorithm with decomposition. *Physica A: Statistical Mechanics and its Applications*, *391*, 4050–4060.

[9] Jan, M. A., & Khanum, R. A. (2013). A study of two penalty-parameterless constraint handling techniques in the framework of moea/d. *Applied Soft Computing*, *13*, 128–148.

[10] Konstantinidis, A., & Yang, K. (2011). Multi-objective energy-efficient dense deployment in wireless sensor networks using a hybrid problem-specific MOEA/D. *Applied Soft Computing*, *11*, 4117–4134.

[11] Li, H., & Zhang, Q. (2009). Multiobjective optimization problems with complicated pareto sets, MOEA/D and NSGA-II. *IEEE Transactions on Evolutionary Computation*, *13*, 284–302.

[12] Mashwani, W. K., & Salhi, A. (2014). Multiobjective memetic algorithm based on decomposition. *Applied Soft Computing*, *21*, 221–243.

[13] Medina, M. A., Das, S., Coello Coello, C. A., & Ramirez, J. M. (2014). Decomposition-based modern metaheuristic algorithms for multi-objective optimal power flow - A comparative study. *Engineering Applications of Artificial Intelligence*, *32*, 10–20.

[14] Miettinen, K. (1999). *Nonlinear Multiobjective Optimization* volume 12 of *International Series in Operations Research and Management Science*. Kluwer Academic Publishers, Dordrecht.

[15] Peng, W., Zhang, Q., & Li, H. (2009). Comparison between MOEA/D and NSGA-II on the multi-objective travelling salesman problem. In C.-K. Goh, Y.-S. Ong, & K. C. Tan (Eds.), *Multi-Objective Memetic Algorithms* (pp. 309–324). Springer Publishing Company, Incorporated. (1st ed.).

[16] Press, W. H., Teukolsky, S. A., Vetterling, W. T., & Flannery, B. P. (2007). Numerical recipes 3rd edition: The art of scientific computing. chapter Chapter 14.2: Do Two Distributions Have the Same Means or Variances. (pp. 726–729). New York, NY, USA: Cambridge University Press. (3rd ed.).

[17] Waldock, A., & Corne, D. (2011). Multiple objective optimisation applied to route planning. In *Proceedings of the 13th annual conference on Genetic and evolutionary computation* GECCO '11 (pp. 1827–1834). New York, NY, USA: ACM.

[18] Zhang, Q., & Li, H. (2007). MOEA/D: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, *11*, 712–731.

[19] Zhang, Q., & Li, H. (2012). MOEA/D implementation dated 2012.08.22. `http://dces.essex.ac.uk/staff/qzhang/moead/moeadmatlab.rar`. Accessed 2013.05.20.

[20] Zitzler, E., Deb, K., & Thiele, L. (2000). Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary Computation*, *8*, 173–195.

[21] Zitzler, E., Laumanns, M., & Thiele, L. (2002). SPEA2: Improving the Strength Pareto Evolutionary Algorithm for Multiobjective Optimization. In K. Giannakoglou et al. (Eds.), *Evolutionary Methods for Design, Optimisation and Control with Application to Industrial Problems (EUROGEN 2001)* (pp. 95–100). International Center for Numerical Methods in Engineering (CIMNE).