Dipartimento di Scienze Economiche, Matematiche e Statistiche

Università degli Studi di Foggia

Dynamic Objectives Aggregation in MultiobjectiveEvolutionary Optimization

Gabriella Dellino, Mariagrazia Fedele e Carlo Meloni

Quaderno n. 06/2008

"Esemplare fuori commercio per il deposito legale agli effetti della legge 15 aprile 2004 n. 106"

Quaderno riprodotto al
Dipartimento di Scienze Economiche, Matematiche e Statistiche
nel mese di marzo 2008 e
depositato ai sensi di legge

Dipartimento di Scienze Economiche, Matematiche e Statistiche, Largo Papa Giovanni Paolo II, 1, 71100 Foggia (Italy), Phone +39 0881-75.37.30, Fax +39 0881-77.56.16

Dynamic Objectives Aggregation in Multi-objective Evolutionary Optimization

Gabriella Dellino[1], Mariagrazia Fedele[2], and Carlo Meloni[3]*

[1] Dipartimento di Matematica
Università di Bari, Via E. Orabona, 4 - 70125 Bari, Italy.
[2] Dipartimento di Scienze Economiche, Matematiche e Statistiche
Università di Foggia, Largo Papa Giovanni Paolo II, 1 - 71100 Foggia, Italy.
[3] Dipartimento di Elettrotecnica ed Elettronica
Politecnico di Bari, Via E. Orabona, 4 - 70125 Bari, Italy.
*Corresponding author: meloni@deemail.poliba.it

Abstract. *Several approaches for solving multi-objective optimization problems entail a form of scalarization of the objectives. This paper proposes a study of different dynamic objectives aggregation methods in the context of evolutionary algorithms. These methods are mainly based on both weighted sum aggregations and curvature variations. A comparison analysis is presented on the basis of a campaign of computational experiments on a set of benchmark problems from the literature.*

Keywords: *Multi-objective optimization, Evolutionary algorithms, Aggregate objective functions*

1 Introduction

A multi-objective nonlinear optimization problem with m objectives (or criteria) can be stated as follows: $min\, f(x) = (f_1(x), f_2(x), \ldots, f_m(x))$, where $x = (x_1, \ldots, x_n)$ is a decision vector which, possibly, has to satisfy a certain number of constraints. We denote the image of the feasible region X with $Z = f(X)$ and the criteria vectors belonging to Z are called *feasible criteria vectors*. The goal of a multi-objective optimizer is to achieve a set of *Pareto optimal solutions*. Since every Pareto point is of potential interest, the target is to *capture* the whole Pareto front. There are several methods for solving multi-objective optimization problems; a classical approach entails a form of scalarization of the criteria vector. Repeated applications of these methods are performed to achieve an estimation of the Pareto front. The aggregate objective function methods transform a multi-criteria optimization problem into a scalar problem using free parameters to be set; for every set of parameter values, the scalar optimization problem is solved to seek a Pareto solution. Hence, the original problem is transformed as follows: $\min_{x \in X} G(x) = F(f_1(x), \ldots, f_m(x))$, with $F : Z \subset \mathbb{R}^m \to \mathbb{R}$. The main issue of a scalarization approach is determining whether the transformed problem and the original one are equivalent. In order to provide the decision maker the chance to choose among all optimal points, an aggregate

function should be able to capture any existing Pareto solution. It is possible to prove that any Pareto optimal point can be *captured* if there is an appropriate aggregate function [14], where a point x is called *capturable* if it is a local optimum of the obtained scalar problem. Therefore, a main issue of this approach is the determination of an appropriate function structure able to provide all the optimal points according to free parameter values.
In this work, an experimental comparative study of different Dynamic Objectives Aggregation Methods (DOAMs) in the context of evolutionary optimization algorithms is proposed. The study is conducted on a set of benchmark problems from the literature. Section 2 presents methods based on both weighted sum aggregations, and curvature variations. In Section 3 the experimental setting is described. In Section 4, the analysis of results is reported, and some conclusions are drawn.

2 Dynamic Objectives Aggregation Methods

The aim of this section is the introduction of the evolutionary dynamic objectives aggregation methods to solve multi-objective optimization problems. An aggregate objective function method transforms a multi-objective optimization problem into a scalar optimization problem. The most common and widely used aggregate function is the weighted sum. Although it has been shown that weighted sum is unable to deal with multi-objective optimization problems with a concave Pareto front, in [6, 7] it is investigated the possibility to capture also the points on concave Pareto front by using a dynamic weighted aggregation combined with evolution strategies. In [8] the phenomenon of *global convexity* is introduced in order to explain the potential success of dynamic weighted aggregation. However, no analytical characterization is given in order to identify a global convex problem, therefore the discussion is based on an observed behaviour rather than theoretical analysis.
From an implementation point of view, classical methods that scalarize multiple objectives, perform repeated applications in order to achieve a set of non-dominated solutions. While dynamic weighted aggregation provides in a single run an entire front of non-dominated solutions. At this aim, these procedures generally use an archive to store the non-dominated solutions obtained during the search process. Empirical results in the literature show that the evolutionary dynamic weighted sum is able to provide the entire non-dominated front in one run of the evolutionary algorithm capturing in some cases the points on concave parts of the Pareto frontier. On the other hand, the method based on the increase of the aggregate function curvature seems to be able to capture the points on concave regions of the front where the weighted sum fails. The rationale behind an integration of the two methods can be summarized observing that by increasing the curvature it may be possible to reach the concave part of the front and by dynamically varying the weights it may be possible to move close to the concave Pareto frontier.
The remainder of the section is devoted to introduce the algorithmic approaches

investigated in our computational study: i) dynamic weighted sum methods, and ii) dynamic curvature variations methods.

2.1 Dynamic weighted sum methods

The most widely used aggregate function is the weighted sum; the corresponding aggregate optimization problem can be stated as:

$$\min_x \sum_{i=1}^{m} w_i f_i(x), \tag{1}$$

where w_i is a non-negative weight and $\sum_{i=1}^{m} w_i = 1$. For every choice of the weights vector w, the problem (1) yields an optimal Pareto point. It is well-known that a weakness of this aggregate function is the failure to capture the points on a concave Pareto fronts. In fact, it is possible to prove that every point captured by $\sum_{i=1}^{m} w_i f_i$ is in a convex region of the non-dominated frontier. In [6] the dynamic weighted aggregation method combined with evolution strategies has been studied and it has been shown that this method is able to reach the entire Pareto frontier in one run and the points in concave regions as well. This procedure is based on the dynamic aggregation approach. While conventionally the scalarization function weights are fixed during optimization, the main idea on which the method is based is that the weights systematically change during evolution; so the function to be minimized dynamically changes. In this way the optimizer moves close to the frontier, once it achieves a non-dominated solution. Several ways of changing weights have been proposed: randomly, switching between 0 and 1, periodically. In the first case, the weights are generated from a uniform random distribution changing in each generation. The second way of varying the weights is realized by switching them from zero to one abruptly and viceversa. Literature results suggest that the weights should vary gradually and periodically. In particular, a gradual and continuous change is needed to keep the points on a convex front: an abrupt switch of the search direction does not allow the optimizer to move close to the front, storing non-dominated points. During the evolution the population goes through the Pareto frontier, and therefore an archive is needed to record all the Pareto solutions encountered. Although it has been extensively shown that the conventional weighted sum is unable to provide the Pareto solutions on concave regions, the dynamic weighted sum method succeeds in obtaining non-dominated solutions in concave regions as well, traversing the frontier dynamically. Empirical results highlight the important role of law of the varying weights [6, 7]. Since the incorporation of chaos in population-based optimization algorithms has been shown to enhance the searching ability [2, 5, 12], in this study it is proposed to introduce the chaotic behaviour in the dynamic weights generation as well.

2.2 Dynamic curvature variation methods

In order to overcome the drawbacks of the weighted sum scalarization function, several aggregate functions have been introduced in the literature. In particular,

to enhance the capability of objective functions to capture also the points on a concave Pareto front, in [14] it is suggested to increase the curvature of the aggregate function. The corresponding scalar optimization problem can be stated as follows:

$$\min_x \sum_{i=1}^{m} w_i (f_i(x))^t, \tag{2}$$

where t is a positive real number. It is found that varying all the free parameters (i.e. weights and exponents), it is possible to achieve all the points on the Pareto frontier. This aggregate function is also investigated in [10, 11], where it has been proved that applying the t-th power to the objective functions the convexification of non-dominated frontier can be achieved in an appropriate equivalent objective space. The main problem is again the choice of a function structure enable to provide all the Pareto solutions for some values of the parameters used in aggregate function. Assuming that the aggregate objective function and the Pareto frontier satisfy certain differentiability requirements, the necessary and sufficient condition for an admissible aggregate objective function to capture the Pareto points has been developed by Messac [14]. Although these conditions are inapplicable if the Pareto frontier is not known — as it is in real applications — Messac suggested the use of an aggregate function (2) whose curvature can be increased by setting free parameters with the aim to enhance the capability of objective functions to capture also the points on concave Pareto front. This t-th power approach is also investigated in [10]. For sufficiently large values of t, the efficient frontier in the $[f_1^t, ..., f_m^t]$ space is guaranteed to be convex under certain conditions. Therefore, the weighted sum of the t-th power of the objectives is able to solve the problem in the $[f_1^t, ..., f_m^t]$ space. In [4] the properties of the weighted t-th power approach are summarized: i) the optimal solutions of the t-th power problem (2) are efficient solutions of the multi-objective problem; ii) for every efficient solution there exists a $\widehat{t} > 0$ such that for all $t \geq \widehat{t}$ the t-th power aggregate function captures that solution. This result guarantees the existence of a t-th power aggregate function that is able to capture all the Pareto front. Therefore this is an important theoretical support for this work in which different rules to change the values of t are considered in addition to those concerning the weights w_i.

3 Computational Experiments

In this section the experimental setting is illustrated. The evolutionary algorithms involved in the test and their configurations are described in Subsection 3.1. Another subsection reports on the different DOAMs considered in the experiments. In order to evaluate and compare the effectiveness of the proposed methods, a suite of test problems is employed as will be described in Subsection 3.3.

3.1 Evolutionary algorithms and their configurations

To test the method proposed in this work, the standard genetic algorithm included in the Matlab's Genetic Algorithm and Direct Search Toolbox [13] has been used. This algorithm enables to solve single-objective optimization problems and can be easily adapted to work with both constraints and dynamic objectives aggregation. Some parameters values need to be specified, before the algorithm execution: we adopted a stochastic uniform selection operator, a scattered crossover function with probability 0.7 and a Gaussian mutation function with probability 0.3; the number of best individuals that will survive to the next population has been fixed to 2 and the stopping criterion is based on the maximum number of generations to be produced. Several settings have been considered for the genetic algorithm, by varying the population size (in the set $\{25, 50, 100\}$) and the performed iterations (100, 500 or 1000); thus, 9 overall different configurations of the genetic algorithm are used.
The considered DOAMs require to solve single-objective optimization problems, given by the dynamic weighted sum of the objectives we are really interested in. To keep all the non-dominated solutions provided by the evolutionary process, the algorithm is equipped with an archive in which they are dynamically stored. The use of the archive requires a capacity control: a domination and crowding analysis is conducted on the elements that are proposed to be enclosed in it. The evolutionary optimizer at each iteration proposes the feasible elements contained in its current population to the archive which is updated, removing all dominated solutions. A well-known multi-objective genetic algorithm (MOGA), NSGA-II — extensively described by Deb in [3] — has also been used, aiming to compare the solutions obtained with the proposed method with those provided by a *native* MOGA. The algorithm configurations described before have also been applied to NSGA-II, except for the dominance management which is implicitly guaranteed by the algorithm itself.

3.2 The set of DOAMs

Several DOAMs have been used in the campaign of experiments conducted in this work, involving both the variation of the weights (only) and the combined variation of the weights and the exponents of (2). For the sake of simplicity, in order to describe the methods, a bi-objective problem is considered. In this case, the aggregate function corresponding to the k-th generation can be stated as follows:

$$G(x, k) = w_1(k) f_1^t(x) + w_2(k) f_2^t(x), \tag{3}$$

where the expressions of w_1 and w_2 and the value of t depend on the adopted variation law; clearly, for $t = 1$ the simpler weighted sum aggregated function is obtained. The weights w_i can be dynamically modified according to a rule $R(k)$ described by a specific function of k:

$$w_1(k) = R(k), \qquad w_2(k) = 1 - w_1(k). \tag{4}$$

We consider different rule functions: *one*, *switch*, *sin*, *triangle*, *rand*, *chaos*. The first refers to the case of fixed unitary weights (i.e. the aggregate function is simply given by the sum of the objectives). In the second case w_1 periodically changes from 0 to 1, with a given period $T = 200$ (in terms of the number of algorithm's generations). Similarly, a periodical changing of the weights can be obtained also according to a sin or triangle wave in the successive adopted rules. The *rand* rule, at each iteration k, generates a random value in $(0, 1)$ for w_1. Whereas, the last rule applies a chaotic variation law to the weights. A logistic equation — which is extensively used to describe a chaotic system [1,2] — is employed as follows:

$$w_1(k+1) = \mu w_1(k)(1 - w_1(k)), \qquad w_2(k) = 1 - w_1(k). \tag{5}$$

This equation shows chaotic behaviour when $\mu = 4$ and $w_1(0) \notin \{0, 0.25, 0.5, 0.75, 1\}$. Clearly, some other well-known chaotic maps could also be employed instead of the logistic one to generate the weights in the aggregate objective functions [2]. In order to let the curvature of the aggregate function vary during the evolution process four possible strategies are proposed for the variation of the exponent. In all the cases considered in the following, the exponent value ranges between $t = 1$ and $t = 4$, retaining that greater values of t would not provide further improvements in the optimization results achieved so far. A first scheme (*one*) considers only fixed unitary exponents. The second scheme (*step*) establishes to increment the exponent value every $N/4$ iterations, N being the maximum number of generations that can be produced. An adaptive scheme (*adapt*) has also been tested, according to which the exponent value is incremented when there is no improvement in the optimization process for a given number of iterations, which has been fixed to $\Delta = 0.05\,N$. According to both these strategies, the exponent value is always a positive integer number; the last strategy (*cont*) considered in this work let it range among the (positive) real numbers, i.e. the interval $(0, N)$ has been mapped into the interval $(1, 4)$ such that the exponent t can vary continuously in this range. Combining the aforementioned weights-exponents strategies, 24 different DOAMs are obtained. Hereinafter, each of them will be denoted indicating the strategies as an ordered pair (e.g. *chaos-step* represents the strategy with the chaotic rule for the weights and the *step* for exponents, respectively).

3.3 Test Problems

The computational test of the methods has been conducted on a set of benchmark problems, characterized by different specific features in the Pareto front, so that the general results obtained would not depend on the particular test problem chosen. Problems P_1-P_7 are discussed by Y. Jin et al. in [6–8]: in P_2-P_5 it is assumed that $x_i \in [0, 1]$ for all $i = 1, \ldots, n$; while in P_1, P_6, and P_7 there are no restrictions on the range of the decision variables. The problem P_1 is defined by the following equations:

$$f_1 = \frac{1}{n}\sum_{i=1}^{n} x_i^2; \qquad f_2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - 2)^2 \tag{6}$$

and produces a uniform Pareto front. P_2 is described by:

$$f_1 = x_1 \tag{7}$$

$$g(x_2, \ldots, x_n) = 1 + \frac{9}{n-1} \sum_{i=2}^{n} x_i \tag{8}$$

$$f_2 = g\,(1 - \sqrt{f_1/g}). \tag{9}$$

having a convex Pareto front. Because of the interest in studying problems showing non-convex or discontinuous Pareto front, some instances belonging to this class have been considered. The following problem, P_3, has a concave Pareto front and is defined as follows:

$$f_1 = x_1 \tag{10}$$

$$g(x_2, \ldots, x_n) = 1 + \frac{9}{n-1} \sum_{i=2}^{n} x_i \tag{11}$$

$$f_2 = g\,(1 - (f_1/g)^2). \tag{12}$$

The fourth problem, P_4, has been obtained through combining — in some sense — P_2 and P_3:

$$f_1 = x_1 \tag{13}$$

$$g(x_2, \ldots, x_n) = 1 + \frac{9}{n-1} \sum_{i=2}^{n} x_i \tag{14}$$

$$f_2 = g\,(1 - \sqrt[4]{f_1/g} - (f_1/g)^4). \tag{15}$$

Its Pareto front is neither purely convex nor purely concave. The following problem, P_5, is characterized by a Pareto front consisting of a number of separated convex parts.

$$f_1 = x_1 \tag{16}$$

$$g(x_2, \ldots, x_n) = 1 + \frac{9}{n-1} \sum_{i=2}^{n} x_i \tag{17}$$

$$f_2 = g\,(1 - \sqrt{f_1/g} - (f_1/g)\, sin(10\pi f_1)) \tag{18}$$

Problem P_6 is defined through the following equations:

$$f_1 = 1 - exp\left\{-\sum_{i=1}^{n}\left(x_i - \frac{1}{\sqrt{n}}\right)^2\right\} \tag{19}$$

$$f_2 = 1 - exp\left\{-\sum_{i=1}^{n}\left(x_i + \frac{1}{\sqrt{n}}\right)^2\right\} \tag{20}$$

showing a concave Pareto front. Another problem P_7, is obtained in [8] extending one of the test function proposed in literature [15] to the two-dimensional case:

$$f_1 = exp(-x_1) + 1.4\, exp(-x_1^2) + exp(-x_2) + 1.4\, exp(-x_2^2) \tag{21}$$

$$f_2 = exp(x_1) + 1.4\, exp(-x_1^2) + exp(x_2) + 1.4\, exp(-x_2^2) \tag{22}$$

The resulting Pareto front is continuous and non-convex. Even if the problem is considered an easy task for evolutionary optimizers [8], the region that defines the Pareto front in the parameter space is disconnected; so, it could be an interesting problem to be studied. In the following, two other benchmark problems, P_8-P_9, are considered, because of the particular shape of their feasible region and/or Pareto front. These problems are described in [16]. Problem P_8 is referred to as the TNK problem. The objectives are very simple, and defined by

$$f_1 = x_1, \qquad f_2 = x_2, \tag{23}$$

where

$$x_1 \in [0, \pi], \qquad x_2 \in [0, \pi].$$

The constraints are

$$g_1 = -x_1^2 - x_2^2 + 1 + 0.1\, cos(16 arctg(x_2/x_1)) \leq 0, \tag{24}$$
$$g_2 = (x_1 - 0.5)^2 + (x_2 - 0.5)^2 \leq 0.5. \tag{25}$$

Problem P_9 is the so-called Poloni test problem: the objective functions are defined as

$$f_1 = 1 + (a - b)^2 + (c - d)^2, \qquad f_2 = (x_1 + 3)^2 + (x_2 + 1)^2, \tag{26}$$

where the parameters introduced in the expression of f_1 are:

$$\begin{aligned} a &= 0.5 sin(1) - 2.0 cos(1) + 1.0 sin(2) - 1.5 cos(2) \\ b &= 0.5 sin(x_1) - 2.0 cos(x_1) + 1.0 sin(x_2) - 1.5 cos(x_2) \\ c &= 1.5 sin(1) - 1.0 cos(1) + 2.0 sin(2) - 0.5 cos(2) \\ d &= 1.5 sin(x_1) - 1.0 cos(x_1) + 2.0 sin(x_2) - 0.5 cos(x_2) \end{aligned}$$

The variables ranges are: $x_1 \in [-\pi, \pi]$; and $x_2 \in [-\pi, \pi]$.

4 Results

In order to test and compare the different DOAMs on the considered benchmark problems, we adopt as main performance indicator the hyperarea ratio (HR) [9]. As secondary indicators we report the number of non-dominated elements (ND) and the spacing (S). For each experiment three different runs have been executed initializing algorithms with random populations.
Table 1 contains the average results on 3 runs for each of the 9 algorithms configurations described in Subsection 3.1, so each entry is the average on 27 experimental values. The collected results show that methods based on periodical changes of weights often achieve relative good performances with respect to other DOAMs as well as to a state-of-the-art (*native*) multi-objective optimizer. This behaviour seems to be reinforced by the use of exponents variation. Nonetheless, it is noticeable the competitive performance of the DOAMs based

on a *chaos* rule in terms of HR, ND, and S. On the other hand, strategies based on the *switch* rule (no matter what strategy is adopted for exponents) almost always give relative poor results.
Although each average is composed of a large number of data points, it is necessary to carry out a statistical analysis to assess if the observed differences in the average values are indeed statistically significant. Since in non-parametric testing a lot of information is lost because the data have to be ranked and the differences in the values are transformed into a rank value. We consider parametric ANOVA analysis and non-parametric Friedman rank-based test. These analyses assume only HR as response variable.
Figures 1 and 2 show the means plot in the Friedman and ANOVA analysis, respectively. The analyses are conducted on the algorithm factor considering 24 different DOAMs and NSGA-II with three repetition for each experiment which is characterized by the problem and by the algorithm configuration (a total of 81 combinations are considered). Thus, in our Friedman analysis, for every experiment 75 ranks are obtained, assigning a larger rank to better results. For both tests we use a confidence interval of 95% and adopt the Tukey's HSD intervals. As it can be seen, the non-parametric test is less powerful presenting much wider intervals (in which overlapping intervals indicate a non-statistically significant difference on the average performance of the algorithms) and neglecting the differences in the response variables. These analyses clearly confirm the negative assessment on *switch* strategies and the promising behaviour of chaos-based DOAMs which are often the best strategy (e.g. see DOAM13 in Figure 1) being able to dominate the performance of different candidates (see figures 1 and 2). This result seems to support the increasing research interest about the introduction of some form of chaotic behaviour in stochastic optimizers [1, 2, 5, 12].
These experimental results are of interest also in different contexts such as the development of multi-objective optimization algorithms starting from well-established evolutionary single-objective optimizers, in the design of compact (and fast) local search procedures, in surrogate-based optimization, and in landscape approximation of costly functions. However, the observations based on the encouraging results from the conducted experiments indicate that different aspects deserve further research efforts. Mainly, it seems useful to extend the experimental campaign: i) on a wider set of problems (also from real applications); ii) to include different quality indicators; iii) to compare DOAMs and other state-of-the-art MOGAs; iv) to better enlighten the effects of the interactions of weights and exponents based rules; and v) to include other chaotic DOAMs in order to deeply investigate their effectiveness.

References

1. Bucolo M., Caponetto R., Fortuna L., Frasca M., Rizzo A., *Does chaos work better than noise?*, IEEE Circuits and Systems Magazine, Vol. 2, No. 3, pp. 4–19, 2002.

Table 1. Average values of HR (in %), ND, and S achieved by the algorithms for each problem.

Algorithm	Description	P_1			P_2			P_3			P_4			P_5			P_6			P_7			P_8			P_9		
		HR	ND	S	HR	ND	S	HR	ND	S	HR	ND	S	HR	ND	S	HR	ND	S	HR	ND	S	HR	ND	S	HR	ND	S
DOAM1	chaos-one	74	25.37	0.36	90	10.67	0.42	85	6.26	0.17	81	7.00	0.29	86	6.04	0.21	21	16.79	0.14	100	328.19	3.41	23	4.70	0.18	91	224.79	0.21
DOAM2	one-one	74	24.85	0.38	91	7.59	0.45	83	4.33	0.29	81	7.04	0.25	83	5.19	0.40	21	15.29	0.14	100	152.93	1.5e56	21	4.89	0.12	89	126.25	0.63
DOAM3	rand-one	75	25.41	0.29	91	8.52	0.32	84	4.96	0.35	81	6.33	0.29	84	5.26	0.56	21	16.13	0.18	100	310.37	3.68	22	4.63	0.14	91	212.92	0.27
DOAM4	switch-one	73	23.52	0.53	88	3.48	0.18	82	2.22	0.13	81	5.78	0.29	83	3.78	0.42	22	14.96	0.14	100	474.59	1.7e11	21	4.48	0.18	82	36.63	1.56
DOAM5	sin-one	75	24.93	0.31	91	8.07	0.24	85	4.44	0.18	81	6.37	0.25	86	5.07	0.33	22	19.17	0.11	100	371.56	8.98	22	4.67	0.16	91	160.58	1.18
DOAM6	triangle-one	76	24.15	0.33	90	6.11	0.37	85	5.22	0.18	81	6.74	0.27	85	5.74	0.36	23	20.46	0.10	100	327.00	4.82	21	4.48	0.16	91	166.29	1.29
DOAM7	chaos-step	75	25.07	0.36	90	8.04	0.30	84	5.11	0.32	81	6.41	0.25	85	6.44	0.36	21	17.46	0.16	100	329.52	5.31	21	4.11	0.14	91	244.50	0.23
DOAM8	one-step	75	25.00	0.38	90	7.78	0.28	84	4.04	0.30	81	7.48	0.31	83	5.41	0.55	21	15.08	0.18	100	141.93	3.54	22	5.00	0.13	91	158.46	0.27
DOAM9	rand-step	76	24.70	0.31	91	6.93	0.40	84	4.33	0.29	81	6.67	0.27	85	5.22	0.36	21	17.04	0.12	100	315.37	3.78	21	4.37	0.16	91	231.00	0.26
DOAM10	switch-step	75	23.52	0.46	89	3.67	0.16	82	2.59	0.23	80	5.48	0.35	84	3.85	0.34	21	15.92	0.12	100	475.74	1.8e10	22	4.59	0.15	83	38.71	1.60
DOAM11	sin-step	75	24.63	0.32	91	7.22	0.28	84	4.11	0.17	81	7.15	0.29	85	5.44	0.27	23	20.83	0.15	100	373.44	9.37	21	4.30	0.18	91	176.75	0.68
DOAM12	triangle-step	75	25.44	0.28	91	8.33	0.31	84	4.67	0.31	81	7.52	0.27	85	5.33	0.32	24	20.21	0.11	100	325.22	3.70	22	4.15	0.12	91	184.75	0.60
DOAM13	chaos-adapt	74	24.85	0.32	91	8.96	0.20	84	4.89	0.21	81	7.15	0.26	85	5.78	0.28	22	17.96	0.11	100	339.44	3.90	22	4.48	0.14	91	243.42	0.41
DOAM14	one-adapt	76	25.93	0.32	91	7.41	0.25	84	3.89	0.25	81	7.44	0.28	83	5.33	0.60	20	16.75	0.14	100	144.70	4.08	22	4.78	0.13	91	142.42	0.27
DOAM15	rand-adapt	73	25.33	0.31	90	6.07	0.30	85	4.70	0.14	81	6.78	0.31	83	6.15	0.42	21	16.79	0.12	100	319.78	3.92	21	4.15	0.12	91	218.13	0.24
DOAM16	switch-adapt	73	24.26	0.59	88	3.04	0.13	83	3.00	0.18	80	4.74	0.37	84	4.07	0.32	21	15.63	0.14	100	478.63	1.3e12	21	4.11	0.16	85	39.46	0.96
DOAM17	sin-adapt	75	25.30	0.27	91	6.67	0.13	85	4.85	0.14	81	6.74	0.32	86	5.48	0.41	24	21.21	0.08	100	375.22	11.70	20	4.30	0.14	91	171.42	0.89
DOAM18	triangle-adapt	76	25.15	0.38	91	8.52	0.25	84	4.56	0.32	82	7.44	0.23	85	6.26	0.31	23	19.13	0.14	100	332.15	4.19	22	4.59	0.19	91	196.75	0.91
DOAM19	chaos-cont	75	24.93	0.30	90	8.52	0.25	84	5.22	0.41	81	7.11	0.29	85	6.63	0.27	21	17.08	0.13	100	287.00	4.55	21	4.15	0.12	91	210.83	0.26
DOAM20	one-cont	75	25.22	0.33	91	7.93	0.23	84	5.07	0.37	81	7.11	0.31	85	5.74	0.40	21	15.54	0.20	100	93.52	3.27	22	4.48	0.15	91	148.50	0.29
DOAM21	rand-cont	75	23.89	0.38	91	7.07	0.31	83	4.19	0.31	81	6.78	0.30	85	6.11	0.43	22	17.17	0.14	100	256.85	4.45	23	4.56	0.15	91	184.96	0.27
DOAM22	switch-cont	74	23.85	0.48	89	3.74	0.26	82	3.04	0.10	80	5.48	0.32	83	3.96	0.12	22	16.38	0.13	100	482.89	2.9e10	21	4.30	0.11	80	32.25	1.17
DOAM23	sin-cont	75	24.81	0.33	91	7.63	0.36	84	5.48	0.58	81	7.15	0.25	85	5.89	0.32	23	19.25	0.10	100	290.78	5.85	22	4.48	0.17	91	162.50	1.08
DOAM24	triangle-cont	76	25.59	0.33	90	7.30	0.42	84	5.85	0.30	81	7.19	0.22	85	5.89	0.49	23	20.17	0.11	100	258.74	3.49	22	4.33	0.13	91	171.58	1.20
NSGA-II	multi-obj	65	26.52	0.07	90	4.19	0.17	84	2.59	0.15	81	4.96	0.21	86	4.11	0.25	18	16.92	0.03	100	411.33	1.5e11	16	2.48	0.12	91	440.42	0.16

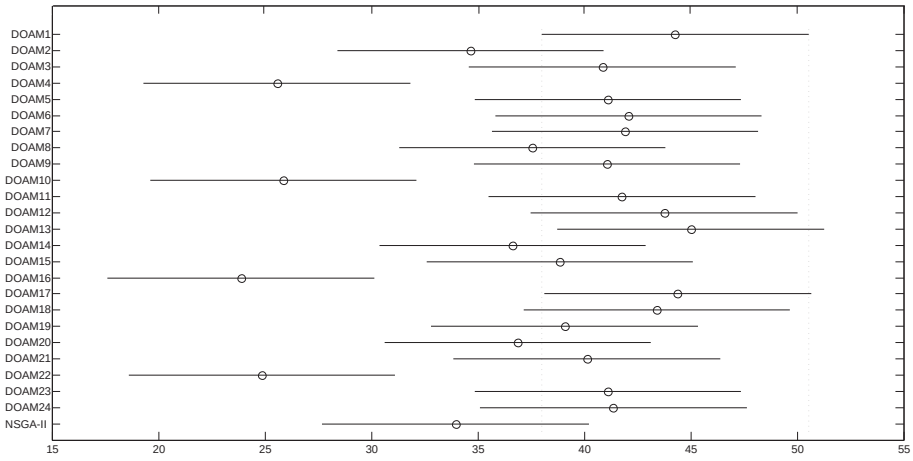

Fig. 1. Means plot and Tukey's HSD confidence intervals ($\alpha = 0.05$) resulting from the Rank-based Friedman analysis on HR.

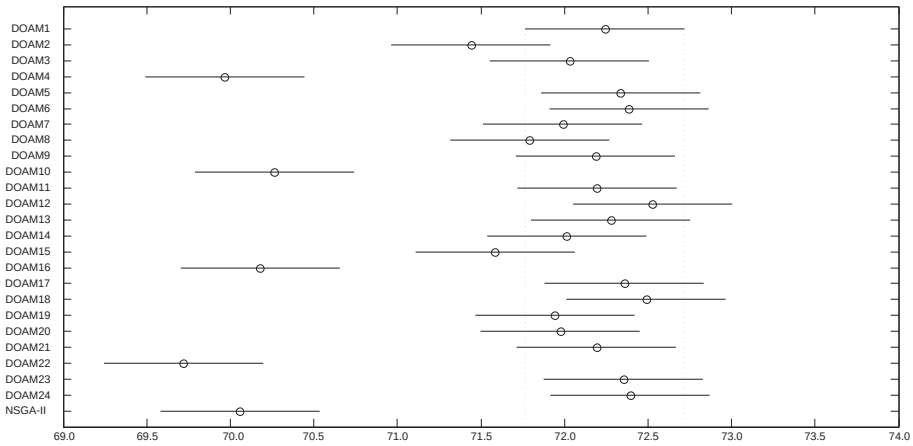

Fig. 2. Means plot and Tukey's HSD confidence intervals ($\alpha = 0.05$) resulting from the ANOVA analysis on HR.

2. Caponetto R., Fortuna L., Fazzino S. and Xibilia M.G., *Chaotic Sequences to Improve the Performance of Evolutionary Algorithms*, IEEE Transactions On Evolutionary Computation, Vol. 7, No. 3, pp. 289–304, June 2003
3. K. Deb, A. Pratap, S. Agarwal and T. Meyarivan, A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II, in *IEEE Transactions on Evolutionary Computation*, Vol. 6, No. 2, pp. 182–197, 2002.
4. Ehrgott, M. and Wiecek, M., *Multiobjective programming.* In J. Figueira, S. Greco, and M. Ehrgott, editors, Multicriteria Decision Analysis: State of the Art Surveys, pp. 667-722. Kluwer Academic Publishers, Boston, 2005.
5. Greenwood G.W., *Chaotic behavior in evolution strategies*, Physica D, 109, pp. 343-350, 1997.
6. Jin Y., Olhofer M., and Sendhoff B., *Dynamic weighted aggregation for evolutionary multi-objective optimization: why does it work and how?*. In Proceedings of the Genetic and Evolutionary Computation Conference, pp. 1042–1049, San Francisco, USA, 2001.
7. Jin Y., Okabe T., and Sendhoff B., *Adapting weighted aggregation for multiobjective evolution strategies.* Lecture Notes in Computer Science, Vol. 1993, Springer, pp. 96–110, Zurich, Switzerland, 2001.
8. Jin Y., *Effectiveness of weighted aggregation of objectives for evolutionary multiobjective optimization: methods, analysis and applications.*, unpublished manuscript, 2002.
9. Knowles J.D., Thiele L., Zitzler E., *A tutorial on the performance assessment of Stochastic Multi-objective optimizers*, TIK-report n.214, ETH Zurich, 2006.
10. Li D., *Convexification of a noninferior frontier*, Journal of Optimization Theory and Applications, Vol. 88, No. 1, pp. 177–196, January 1996.
11. Li D., Biswal M.P., *Exponential transformation in convexifying a noninferior frontier and exponential generating method*, Journal of Optimization Theory and Applications, Vol. 99, No. 1, pp. 183–199, October 1998.
12. Lu H., Zhang H., Ma L., *A new optimization algorithm based on chaos*, Journal of Zhejiang University SCIENCE A, Vol. 7, No. 4, pp. 539–542, 2006.
13. The MathWorks Inc., *Genetic Algorithm and Direct Search Toolbox*, Natick, Massachussetts, 2004.
14. Messac A., Sukam C.P., Melachrinoudis E., *Aggregate objective functions and Pareto frontiers: required relationships and practical implications*, Optimization and Engineering, Vol. 1, pp. 171–188, 2000.
15. Messac A., Sundararaj G.J., Tappeta R.V. and Renaud J.E., *Ability of objective functions to generate points on nonconvex pareto frontiers*, AIAA Journal, Vol. 38, No. 6, pp. 1084–1091, June 2000.
16. Rigoni E., Poles S., *NBI and MOGA-II, two complementary algorithms for Multi-Objective optimizations.* In J. Branke, K. Deb, K. Miettinen, and R. E. Steuer, editors, Practical Approaches to Multi-Objective Optimization, Dagstuhl Seminar Proceedings, No. 4461, 2005.