

Introducing MONEDA: Scalable Multiobjective Optimization with a Neural Estimation of Distribution Algorithm

Luis Martí Jesús García Antonio Berlanga José M. Molina
GIAA, Dept. of Informatics, Universidad Carlos III de Madrid
Av. Universidad Carlos III 22, Colmenarejo 28270 Madrid, Spain
{lmarti,jgherrer}@inf.uc3m.es; {aberlan,molina}@ia.uc3m.es

ABSTRACT

In this paper we explore the model-building issue of multiobjective optimization estimation of distribution algorithms. We argue that model-building has some characteristics that differentiate it from other machine learning tasks. A novel algorithm called multiobjective neural estimation of distribution algorithm (MONEDA) is proposed to meet those characteristics. This algorithm uses a custom version of the growing neural gas (GNG) network specially meant for the model-building task. As part of this work, MONEDA is assessed with regard to other classical and state-of-the-art evolutionary multiobjective optimizers when solving some community accepted test problems.

Categories and Subject Descriptors

I.2.8 [Artificial Intelligence]: Problem Solving, Control Methods and Search; I.2.m [Artificial Intelligence]: Evolutionary Computing and Genetic Algorithms—*Multiobjective Evolutionary Algorithms*

General Terms

Algorithms, Experimentation, Performance

Keywords

Multiobjective Optimization, Estimation of Distribution Algorithms (EDAs), Growing Neural Gas (GNG)

1. INTRODUCTION

Multiobjective optimization problems (MOPs) have attracted a great deal of attention inside the evolutionary computation community. In those problems the optimizer must find one or more feasible solutions, that corresponds to the extreme values (either maximum or minimum) of two or more functions subject to a set of restrictions.

The application of evolutionary computation approaches to the above described problem has prompted the creation of

what has been called multiobjective optimization evolutionary algorithms (MOEAs) [6]. These algorithms have succeeded in yielding relevant results mainly because of their parallel global search and non-assumption of any particular shape of the underlying fitness landscape.

In spite of the success of MOEAs, two crucial issues arise when addressing high-complexity problems. The first one is inherited from single objective evolutionary algorithms and has to do with the non-intuitive nature of the evolutionary operators used. These operators in most cases are problem dependent and hard to grasp by the experimenter. The other issue has to do with scalability and what has been denominated as many-objective optimization problems. In a series of experimental studies, like [9, 18] and [6, pp. 414–419], among others, it has been shown that there is an exponential dependence between the dimension of the objective space and the amount of resources required to solve the problem correctly.

One of the possible strategies for addressing these issues is to resort to simpler and more efficient evolutionary approaches. Estimation of distribution algorithms (EDAs) [11] can be used for that objective. EDAs have been hailed as a landmark in the progress of evolutionary algorithms. They replace the application of evolutionary operators with the creation of a statistical model of the fittest elements of the population using a machine learning algorithm. This *model-building algorithm* is the key feature that differentiates EDAs from other evolutionary approaches. The constructed model is then sampled to produce new elements.

EDAs have been extended to the multiobjective domain prompting what has been denominated multiobjective EDAs (MOEDAs). Most MOEDAs have limited themselves to porting single objective EDAs to the multiobjective domain by incorporating features taken from MOEAs, in particular the fitness assignment functions. Although MOEDAs have proved themselves as a valid approach to MOPs, this fact hinders the achievement of a significant improvement regarding “standard” MOEAs in high-dimensional problems.

There are two fundamental research directions that might yield a substantial MOEDAs improvement: the fitness assignment and the model-building algorithms. The former has to do with the creation of more robust and less computationally complex fitness assignment functions and is shared with MOEAs. The other direction deals with the machine learning algorithms used for building the model of the fittest elements. So far, most MOEDAs (and their single-objective counterparts, for that matter) have used off-the-shelf ma-

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

GECCO’08, July 12–16, 2008, Atlanta, Georgia, USA.
Copyright 2008 ACM 978-1-60558-130-9/08/07...\$5.00.

chine learning algorithms. This was done without taking into account that the model-building problem has some specific requirements that differentiate it from the rest of the classical machine learning tasks. Overlooking those requirements introduces a handicap in the search process that might be the cause of the similar performance of MOEDAs and MOEAs.

In this paper we focus on this later line of research, as, to the best of our knowledge, it has not been properly explored yet. We provide an in-depth discussion of the model-building issue, its particularities and requirements. Relying on those arguments we propose a novel algorithm called multiobjective neural EDA (MONEDA). This algorithm uses a modified the growing neural gas (GNG) network [8] that meets the requirements of the model-building task.

The rest of this work is organized as follows. In the subsequent section we analyze the model-building problem. After that, we proceed to properly describe MONEDA and the model-building GNG network. The ensuing section is dedicated to assess MONEDA with regard to other classical and state-of-the-art evolutionary multiobjective optimizers when solving some community accepted test problems. Finally, some conclusive remarks are made, and future lines of work are sketched.

2. THEORETICAL BACKGROUND

A multiobjective optimization problem (MOP) can be expressed as

$$\left. \begin{array}{l} \text{minimize } \mathbf{F}(\mathbf{x}) = \langle f_1(\mathbf{x}), \dots, f_M(\mathbf{x}) \rangle, \\ \text{subject to } c_1(\mathbf{x}), \dots, c_R(\mathbf{x}) \leq 0, \\ \text{with } \mathbf{x} \in \mathcal{D}, \end{array} \right\} \quad (1)$$

where \mathcal{D} is known as the *decision space*. The functions $f_1(\mathbf{x}), \dots, f_M(\mathbf{x})$ are the *objective functions*. Their corresponding image set, \mathcal{O} , of \mathcal{D} produced by is named *objective space* ($\mathbf{F} : \mathcal{D} \rightarrow \mathcal{O}$). Finally, inequalities $c_1(\mathbf{x}), \dots, c_R(\mathbf{x}) \leq 0$ express the restrictions imposed to the values of \mathbf{x} .

The solution to this problem can be defined relying on the so-called *Pareto dominance relation*: having $\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{D}$, \mathbf{x}_1 is said to dominate \mathbf{x}_2 (expressed as $\mathbf{x}_1 \prec \mathbf{x}_2$) iff $\forall f_j, f_j(\mathbf{x}_1) \leq f_j(\mathbf{x}_2)$ and $\exists f_i$ such that $f_i(\mathbf{x}_1) < f_i(\mathbf{x}_2)$. The solution of (1) is a subset of \mathcal{D} that contains elements that are not dominated by other elements of \mathcal{D} . That subset, \mathcal{D}^* , is known as the *Pareto-optimal set* and its image in objective space is called *Pareto-optimal front*, \mathcal{O}^* .

Although MOPs have been addressed with a variety of methods, evolutionary algorithms (EAs) have proved themselves as a competent approach from both theoretical and practical points of view. This fact has led to what has been called multiobjective optimization evolutionary algorithms (MOEAs). Their success is due to the fact that EAs do not make any assumptions about the underlying fitness landscape. Therefore, it is believed they perform consistently well across various types of problems.

Estimation of distribution algorithms (EDAs), like EAs, are population-based optimization algorithms. However, in EDAs, the step where the evolutionary operators are applied is substituted by construction of a statistical model of the most promising subset of the population. This model is then sampled to produce new individuals that are merged with the original population following a given substitution policy. The introduction of machine learning techniques implies that these new algorithms lose the biological plausibility of

their predecessors. In spite of this, they gain the capacity of scalably solving many challenging problems, significantly outperforming standard EAs and other optimization techniques. Early EDAs were intended for combinatorial optimization but they have been extended to continuous domain (see [11] for a review).

Multiobjective optimization EDAs (MOEDAs) [17] are the extensions of EDAs to the multiobjective domain. Most of MOEDAs are a modification of existing EDAs whose fitness assignment strategy is substituted by one of the commonly used by MOEAs.

3. THE MODEL-BUILDING ISSUE

As it was already hinted in section 1, by analyzing different MOEDAs it can be deduced that there are two directions of improvement for their scalability issue. One has to do with the generation of better performing but less computationally complex fitness assignment functions that can guide the search process requiring smaller populations. The other deals with the creation of machine learning algorithms particularly suited for the model-building task. In a previous study [13] the authors compared the behavior of a set of model-building algorithms. It was found out that, in high-dimensional problems and under the same experimental conditions, robust algorithms, like Bayesian networks [16], were outperformed by less robust approaches like k -means algorithm or the randomized leader algorithm [4].

These results have prompted us to conclude that the model-building problem has its own peculiarities and requirements that, therefore, must be addressed with specially tailored approaches and not with off-the-shelf machine learning methods. Among those characteristics we have distinguished two essential ones: the handling of outliers in the model-building data set and the waste of an excess of resources on finding the optimal complexity of the model.

Classical algorithms handle outliers as not representative, noisy or bogus data. However, for model-building purposes outliers are essential, as they could represent unexplored areas of the Pareto-optimal front. If a population element has been included in the set of promising solutions it certainly should not be disregarded. Instead it should be preserved and perhaps even reinforced with regard to other elements located in more populated zones of the objective space. The proper handling outliers would also promote diversity on the Pareto front as it tends to equally sample more or less densely crowded portions of the Pareto-optimal front.

In most EDAs an excess of computing power is spent on finding the optimal model size. However, in model-building, as we had just hypothesized, there is no need of having an statistically correct model. For example, when doing a cluster-based model-building it is not necessary to find the correct amount of clusters. Here a certain degree of overkill regarding the size of the model could simplify the model creation step and therefore it would end up as a less computationally intensive process. Introducing a self-organizing approach capable of on-the-fly model construction might be the answer to this issue.

This discussion leads us towards the search of new model-building approaches that conform to the previous issues. This is a rather odd task, since most of the currently existing machine learning approaches have been conceived with a different and sometimes opposite set of objectives. There are two strategies to sort out this situation. One is to search for

algorithms that possess some known properties that make them particularly suitable, while the other is to synthesize an original algorithm from scratch. Although in the long term we find the second option as the most promising in this paper, we have opted for taking the first alternative as it seems more viable for initial studies.

4. MULTIOBJECTIVE NEURAL EDA

The multiobjective neural EDA (MONEDA) is a MOEDA that uses a modified growing neural gas (MB-GNG) network as its model-building algorithm. The MB-GNG network is a custom-made model-building algorithm devised to cope with the specifications of the task and that will be described later on.

The NSGA-II non-dominated sorting [6] was the scheme selected for fitness assignment. It was chosen because of its proven effectiveness and its relative low computational cost.

Although MONEDA intends to deal with the issues raised by the previous discussion, it was also designed with the following properties in mind:

- *scalability*: MONEDA is expected to outperform similar algorithms when solving many-objective problems;
- *elitism*: as it has proved itself a very advantageous property, and;
- *diversity preservation*: in spite of promoting the preservation of the most fitted solutions, it is also essential that the population remains as diverse as possible.

It should be noted that it could be questioned if MONEDA does or does not an estimation of a distribution. In either case we decided to keep the “EDA” tag as it implies a widely known evolutionary approach.

4.1 Model-Building with Growing Neural Gas

Clustering algorithms have been used as part of the model-building algorithms of EDAs and MOEDAs. However, as we discussed in the previous section a custom-made algorithm might be one of the ways of achieving a significant improvement in this field.

As a foundation for our proposal we have chosen the growing neural gas (GNG) network [8]. GNG networks are intrinsic self-organizing neural networks based on the neural gas [15] model. This model relies in a competitive Hebbian learning rule [14]. It creates an ordered topology of inputs classes and associates a cumulative error to each. The topology and the cumulative errors are conjointly used to determine how new classes should be inserted. Among the vast amount of existing clustering methods we decided to base our approach on GNG because of its interesting properties, in particular:

- it has a fast convergence to low distortion errors and these errors are better than those yielded by “standard” algorithms like k -means clustering, maximum-entropy clustering and Kohonen’s self-organizing feature maps [15];
- its learning rule follows a stochastic gradient descent that follows an explicit energy surface [19];
- the network is sensitive to outliers [19], something undesirable in typical applications but suitable for model-building;

- the network grows to fit itself automatically to the complexity of the problem being solved, and;
- although it benefits from the topological ordering of the nodes; it does not suffer the problem associated to Kohonen networks, where a node can pull its neighbors to invalid or non representative locations of the input space.

Our model-building GNG (MB-GNG) is an extension of the original (unsupervised) GNG. MB-GNG creates a quantization of the input space using a modified version of the GNG algorithm and then computes the deviations associated to each node.

To the original GNG formulation we have added a cluster repulsion mechanism [20]. This enhancement fosters exploration of the input space as it makes each cluster to represent a distinctive zone of the space.

MB-GNG is a one layer network that defines each class as a local Gaussian density and adapts them using a local learning rule. The layer contains a set of nodes $\mathcal{C} = \{c_1, \dots, c_{N^*}\}$, with $N_0 \leq N^* \leq N_{\max}$. Here N_0 and N_{\max} represent the initial and maximal amount of nodes in the network.

A node c_i consists of a center, μ_i , deviations, σ_i , an accumulated error, ξ_i , and a set of edges that define the set of topological neighbors of c_i , \mathcal{V}_i . Each edge has an associated age, $\nu_{i,j}$.

The dynamics of a GNG network consists of three concurrent processes: network adaptation, node insertion and node deletion. The combined use of these three processes renders GNG training Hebbian in spirit [14].

The network is initialized with N_0 nodes with their centers set to randomly chosen inputs. A training iteration starts after an input \mathbf{x} is randomly selected from the training data set, Ψ . Then two nodes are selected for being the closest ones to \mathbf{x} . The *best-matching node*, c_b ,

$$b = \arg \min_{i=1, \dots, N^*} d(\mu_i, \mathbf{x}), \quad (2)$$

is the closest node to \mathbf{x} . Consequently, the *second best-matching node*, $c_{b'}$, is determined as

$$b' = \arg \min_{i=1, \dots, N^*; i \neq b} d(\mu_i, \mathbf{x}). \quad (3)$$

Here $d(\mathbf{a}, \mathbf{b})$ is a distance metric. For this work we have used $d(\cdot)$ defined as

$$d(\mathbf{a}, \mathbf{b}) = \|\mathbf{a} - \mathbf{b}\|. \quad (4)$$

If $c_{b'}$ is not a neighbor of c_b then a new edge is established between them $\mathcal{V}_b = \mathcal{V}_b \cup \{c_{b'}\}$ with zero age, $\nu_{b,b'} = 0$. If, on the other case, $c_{b'} \in \mathcal{V}_b$ the age of the corresponding edge is reset $\nu_{b,b'} = 0$.

At this point, the age of all edges is incremented in one. If an edge is older than the maximum age, $\nu_{i,j} > \nu_{\max}$, then the edge is removed. If a node becomes isolated from the rest it is also deleted.

Clustering error is then added to the best-matching node error accumulator,

$$\Delta \xi_b = d(\mu_i, \mathbf{x})^2. \quad (5)$$

After that, learning takes place in the best-matching node and its neighbors with rates ϵ_{best} and ϵ_{vic} , respectively. For c_b adaptation follows the rule originally used by GNG

$$\Delta \mu_b = \epsilon_{\text{best}} (\mathbf{x} - \mu_b). \quad (6)$$

However for c_b 's neighbors a cluster repulsion term [20] is added to the original formulation, that is, $\forall c_v \in \mathcal{V}_b$,

$$\Delta \boldsymbol{\mu}_v = \epsilon_{\text{vic}} (\mathbf{x} - \boldsymbol{\mu}_v) + \beta e \left(-\frac{d(\boldsymbol{\mu}_v, \boldsymbol{\mu}_b)}{\zeta} \right) \frac{\sum_{c_u \in \mathcal{V}_b} d(\boldsymbol{\mu}_u, \boldsymbol{\mu}_b) (\boldsymbol{\mu}_v - \boldsymbol{\mu}_b)}{|\mathcal{V}_b| d(\boldsymbol{\mu}_v, \boldsymbol{\mu}_b)}. \quad (7)$$

Here β is an integral multiplier that defines the amplitude of the repulsive force while ζ controls the weakening rate of the repulsive force with respect to the distance between the nodes' centers. This approach was already used as part of the robust GNG [19] and it has proved itself useful for obtaining a good spread of the clusters in the inputs' space. In the aforementioned work its stated that the adaptation rule is not sensitive with respect to its parameters. We have set them to $\beta = 2$ and $\zeta = 0.1$ as suggested in [19].

If the current iteration is an integer multiple of T_+ and N^* is smaller than N_{max} then a new neuron is inserted to the network. First, the node with largest error, c_e , is selected the node. Then the worst node among its neighbors, $c_{e'}$, is located. Then N^* is incremented and the new node, c_{N^*} , is inserted between the two nodes,

$$\boldsymbol{\mu}_{N^*} = 0.5 (\boldsymbol{\mu}_e + \boldsymbol{\mu}_{e'}). \quad (8)$$

The edge between c_e and $c_{e'}$ is removed and two new edges connecting c_{N^*} with c_e and $c_{e'}$ are created. The accumulated errors are reduced in a rate $0 \leq \delta_I \leq 1$ by letting

$$\xi_e = \delta_I \xi_e, \quad \xi_{e'} = \delta_I \xi_{e'}. \quad (9)$$

The error of the newly created node is computed as

$$\xi_{N^*} = 0.5 (\xi_e + \xi_{e'}). \quad (10)$$

Finally, the errors of all nodes are decreased by a factor δ_G ,

$$\xi_i = \delta_G \xi_i, \quad i = 1, \dots, N^*. \quad (11)$$

Stopping the learning of GNG is a non-trivial issue shared by the rest of clustering algorithms and all reiterative heuristic algorithms. As we are interested to cover the inputs space as much as possible we will stop if, after a learning epoch, the standard deviation of the accumulated errors is smaller than a certain threshold, ρ ,

$$\sqrt{\frac{1}{N^*} \sum_{i=1}^{N^*} (\xi_i - \bar{\xi})^2} < \rho. \quad (12)$$

This means that we will stop when the outliers are as better represented as possible.

After training has ended the deviations, $\boldsymbol{\sigma}_i$, of the nodes must be computed. For this task we employ the unbiased estimator of the deviations detail in the following algorithm

Set $\mathbf{s}_1, \dots, \mathbf{s}_{N^*} = \mathbf{0}$ and $n_1, \dots, n_{N^*} = 0$.

for all $\mathbf{x} \in \Psi$ **do**

Determine the closest node, c_c to \mathbf{x} .

$\mathbf{s}_c = \mathbf{s}_c + (\mathbf{x} - \boldsymbol{\mu}_c)^2$.

$n_c = n_c + 1$.

Compute the deviations as $\boldsymbol{\delta}_i = \sqrt{\frac{\mathbf{s}_i}{n_i}}$.

4.2 MONEDA Algorithm

MONEDA maintains a population of individuals, \mathcal{P}_t , with t as the current iteration. The algorithm's workflow is similar to other EDAs. It starts from a random initial population \mathcal{P}_0 of n_{pop} individuals. It then proceeds to sort the individuals using the NSGA-II fitness assignment function.

The NSGA-II fitness first ranks the individuals according the dominance relations established between them. Individuals with the same domination rank are then compared using a local crowding distance.

The first step consists in classifying the individuals in a series of categories $\mathcal{F}_1, \dots, \mathcal{F}_L$. Each of these categories store individuals that are only dominated by the elements of the previous categories,

$$\forall \mathbf{x} \in \mathcal{F}_i : \exists \mathbf{y} \in \mathcal{F}_{i-1} \text{ such that } \mathbf{y} \prec \mathbf{x}, \text{ and} \\ \nexists \mathbf{z} \in \mathcal{P}_t \setminus (\mathcal{F}_1 \cup \dots \cup \mathcal{F}_{i-1}) \text{ that } \mathbf{z} \prec \mathbf{x}; \quad (13)$$

with \mathcal{F}_1 equal to \mathcal{P}_t^* , the set of non-dominated individuals of \mathcal{P}_t .

After all individuals are ranked a local crowding distance is assigned to them. The use of this distance primes individuals more isolated with respect to others. The assignment process goes as follows,

for all category sets \mathcal{F}_l , having $f_l = |\mathcal{F}_l|$ **do**
for all individuals $\mathbf{x}_i \in \mathcal{F}_l$ **do**
 $d_i = 0$.
for all objective functions $m = 1, \dots, M$ **do**
 $\mathbf{I} = \text{sort}(\mathcal{F}_l, m)$ (generate index vector).
 $d_{I_1}^{(l)} = d_{I_{f_l}}^{(l)} = \infty$.
for $i = 2, \dots, f_l - 1$ **do**
Update the remaining distances as

$$d_i = d_i + \frac{f_m(\mathbf{x}_{I_{i+1}}) - f_m(\mathbf{x}_{I_{i-1}})}{f_m(\mathbf{x}_{I_1}) - f_m(\mathbf{x}_{I_{f_l}})}.$$

Here the sort (\mathcal{F}, m) function produces an ordered index vector \mathbf{I} with respect to objective function m .

Having the individual ranks and their local distances they are sorted using the crowded comparison operator, stated as: An individual \mathbf{x}_i is better than \mathbf{x}_j if:

- \mathbf{x}_i has a better rank: $\mathbf{x}_i \in \mathcal{F}_k$, $\mathbf{x}_j \in \mathcal{F}_l$ and $k < l$, or;
- if $k = l$ and $d_i > d_j$.

A set $\hat{\mathcal{P}}_t$ containing the best $\lfloor \alpha |\mathcal{P}_t| \rfloor$ elements is extracted from the sorted version of \mathcal{P}_t ,

$$|\hat{\mathcal{P}}_t| = \lfloor \alpha |\mathcal{P}_t| \rfloor. \quad (14)$$

A MB-GNG network is then trained using $\hat{\mathcal{P}}_t$ as training data set. In order to have a controlled relation between size of $\hat{\mathcal{P}}_t$ and the maximum size of the network, N_{max} , these two sizes are bound by the rate $\gamma \in (0, 1]$,

$$N_{\text{max}} = \lceil \gamma |\hat{\mathcal{P}}_t| \rceil. \quad (15)$$

The resulting Gaussian kernels are sampled to produce an amount $\lfloor \omega |\mathcal{P}_t| \rfloor$ of new individuals. Each one of these individuals substitute a randomly selected ones from the section of the population not used for model-building $\mathcal{P}_t \setminus \hat{\mathcal{P}}_t$. The set obtained is then united with best elements, $\hat{\mathcal{P}}_t$, to form the population of the next iteration \mathcal{P}_t .

Iterations are repeated until a given stopping criterion is met. The output of the algorithm is the set of non-dominated solutions of \mathcal{P}_t , \mathcal{P}_t^* . The outline of MONEDA is presented in algorithmic form on Algorithm 1.

Algorithm 1 Algorithmic representation of MONEDA.

MB-GNG parameters: $N_0, \nu_{\max}, \epsilon_b, \epsilon_v, \delta_I, \delta_G$ and ρ .**MONEDA parameters:** $n_{\text{pop}}, \alpha, \gamma$ and ω .Let $t = 0$.Randomly generate the initial population P_0 with z individuals.**repeat**Sort \mathcal{P}_t individuals with regard to the crowded comparison operator.Extract first $\alpha |\mathcal{P}_t|$ elements the sorted \mathcal{P}_t to $\hat{\mathcal{P}}_t$.Train MB-GNG network with $\hat{\mathcal{P}}_t$ training data set and $N_{\max} = \lfloor \gamma |\hat{\mathcal{P}}_t| \rfloor$.Sample $\lfloor \omega |\mathcal{P}_t| \rfloor$ from the MB-GNG.Substitute randomly selected individuals of $\mathcal{P}_t \setminus \hat{\mathcal{P}}_t$ with the new individuals to produce P'_t . $P_{t+1} = \hat{\mathcal{P}}_t \cup P'_t$. $t = t + 1$.**until** end condition is metDetermine the set of non-dominated individuals of \mathcal{P}_t , \mathcal{P}_t^* .**return** P_t^* as the algorithm's solution.

5. EXPERIMENTS

MONEDA and its underlying hypothesis must be properly assessed from an experimental point of view in order to complement the theoretical discussion issued above. However, because of the length restrictions imposed to this contribution we have had to restrain ourselves present a limited set of results.

The experiments consist on the application of a set of known and competent evolutionary multiobjective optimizers and MONEDA to some community-accepted test problems. The algorithms applied are naïve MIDEA [3], mrBOA [1], RM-MEDA [21], MOPED [5], NSGA-II [6] and SPEA2 [6]. The DTLZ3 and DTLZ6 problems scalable multiobjective test problems [7] were used. These problems were selected for their scalability, their known and easily measurable Pareto-optimal front, and their multiple suboptimal fronts. Each problem was configured with 3, 6 and 9 objective functions. In all cases the dimension of the decision space was fixed to 15.

Tests were carried out under the PISA experimental framework [2]. Algorithms' implementations were adapted from the ones provided by their respective authors with the exception of NSGA-II and SPEA2, that were already distributed as part of the framework, and MOPED and MONEDA that were implemented from scratch.

The unary additive epsilon indicator [10] and the Pareto-optimal front coverage indicator [4] were used to assess the performance. The first indicator measures how close the set of solutions is to the optimal one while the second indicator gauges how well distributed are the solutions along the optimal set. Both indicators should be minimized.

For each problem/dimension pair each algorithm was executed 30 times. As assessing the progress of the algorithms in higher dimensions is a complicate matter the MGBM multiobjective optimization cumulative stopping criterion [12] was used.

Figure 1 shows the progress of each algorithm for the different problem setups with regard to the two indicators. In each case the data series are shown until the first of the

30 runs stopped. The statistical description of the results yielded by the algorithms can be inspected on Fig. 2.

In the three dimensional configurations our approach performed similarly to the rest of the algorithms. This was an expected outcome since our MOEDA uses an already existent fitness function and its model-building algorithm is meant to provide a significant advantage in more extreme situations.

However, as the tests proceed into higher dimensions, it becomes evident that MONEDA outperforms the rest of the optimizers applied. MONEDA not only yields better final results but they have a very low variance and also fewer iterations are required. This means that it performed consistently well across the different runs. The results not only show that MONEDA's solutions are close to the optimal but also they manage to evenly cover the Pareto-optimal front.

This might lead us to hypothesize that thanks to its novel treatment of the outliers in the model-building data set our approach manages to overcome the difficulties that hampers the rest of the methods. Although very interesting, the results presented here rise the question of how if they are conditioned by the particularities of the problems solved. They must be further investigated to understand if the low dispersion of the error indicators can only be obtained in the problems solved or if it can be extrapolated to other problems. Regretfully, the briefness of this communication does not allow us to deal with these issues in more depth.

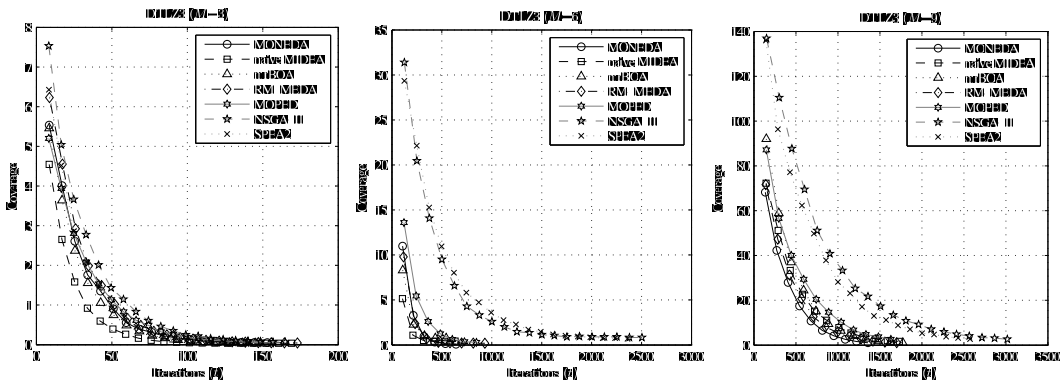
6. FINAL REMARKS

In this paper we have explored the model-building issue of MOEDAs. We argued that model-building has some characteristics that differentiate it from other machine learning tasks. Specifically, we contended that in this problem outliers must not be disregarded but instead they should be preserved, as they represent regions of the suboptimal solutions space that have not been properly explored. In addition to that, we reflected on the excess of computation capacity wasted by MOEDAs in finding the optimal model size and we hypothesized that a self-organizing approach would yield better results in terms of resource consumption, even if a certain degree of overkill is included.

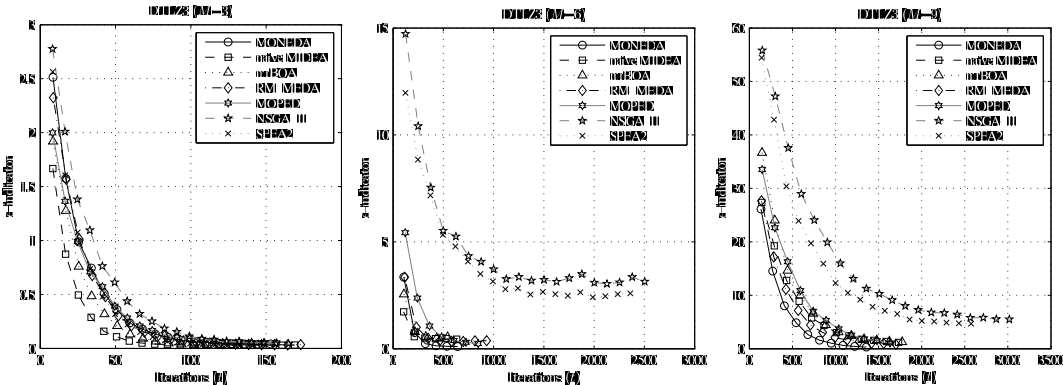
Using those postulates as requirements for a model-building algorithm we proposed a novel algorithm, which we called MONEDA. MONEDA uses a custom-made model-building GNG (MB-GNG) specially intended for this task. In the subsequent experimental studies MONEDA's performance was contrasted with a set of multiobjective optimizers. In those tests MONEDA outperformed the rest of the methods in terms of proximity to the Pareto-optimal front, diversity, and speed of convergence.

In spite of the promising results obtained so far, further studies are necessary. More problems, of both continuous and discrete nature, should be addressed to establish if the outcome of the experiments shown here can be generalized. The detailed analysis of the execution of MONEDA will also be helpful in understanding the actual impact of the modified model-building strategy we put forward here in an iteration-wise scale.

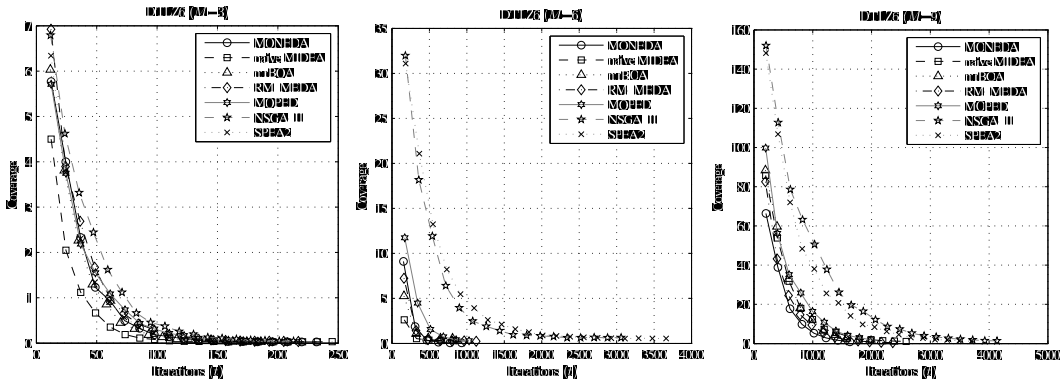
From a theoretical perspective some points still need to be explored. For example, a computational complexity study is necessary in order to grasp the resource consumption of MONEDA when advancing into higher dimensions. On the other hand, more model-building approaches that follow our



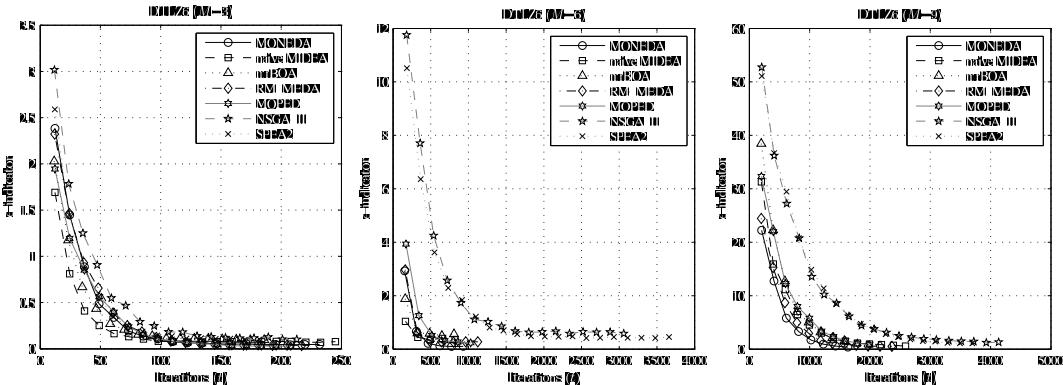
(a) DTLZ3, $M = 3$, coverage. (b) DTLZ3, $M = 6$, coverage. (c) DTLZ3, $M = 9$, coverage.



(d) DTLZ3, $M = 3$, ϵ -indicator. (e) DTLZ3, $M = 6$, ϵ -indicator. (f) DTLZ3, $M = 9$, ϵ -indicator.



(g) DTLZ6, $M = 3$, coverage. (h) DTLZ6, $M = 6$, coverage. (i) DTLZ6, $M = 9$, coverage.



(j) DTLZ6, $M = 3$, ϵ -indicator. (k) DTLZ6, $M = 6$, ϵ -indicator. (l) DTLZ6, $M = 9$, ϵ -indicator.

Figure 1: Algorithms' progress across iterations solving the DTLZ3 and DTLZ6 problems.

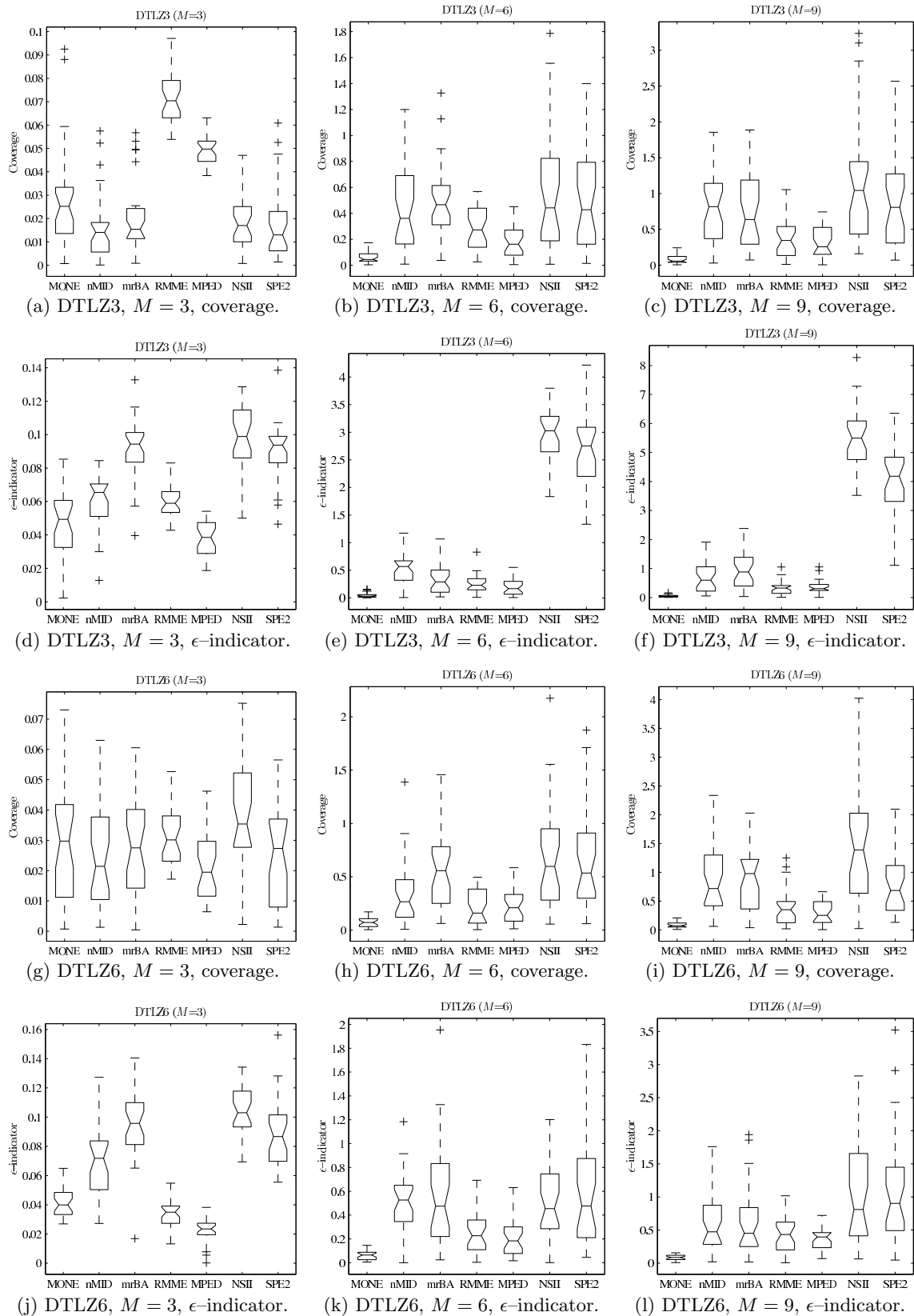


Figure 2: Box plots of the indicators obtained at the final iteration when solving the DTLZ3 and DTLZ6 problems.

premises should be tested. Such models could be found by doing a more exhaustive survey on machine learning algorithms or by proposing completely new ones. If such approaches show themselves as efficient as the one described here, it would help to corroborate the working hypothesis of this contribution.

Acknowledgments

This work was supported by projects MADRINET, TEC2005-07186-C03-02, SINPROB and TSI2005-07344-C02-02. The authors wish to thank the anonymous reviewers for the encouraging comments.

7. REFERENCES

- [1] C. W. Ahn. *Advances in Evolutionary Algorithms. Theory, Design and Practice*. Springer, 2006. ISBN 3-540-31758-9.
- [2] S. Bleuler, M. Laumanns, L. Thiele, and E. Zitzler. PISA—A Platform and Programming Language Independent Interface for Search Algorithms. In C. M. Fonseca, P. J. Fleming, E. Zitzler, K. Deb, and L. Thiele, editors, *Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003*, pages 494–508, Faro, Portugal, April 2003. Springer. Lecture Notes in Computer Science. Volume 2632.
- [3] P. A. Bosman and D. Thierens. The Naïve MIDEA: A baseline multi-objective EA. In C. A. Coello Coello, A. Hernández Aguirre, and E. Zitzler, editors, *Evolutionary Multi-Criterion Optimization. Third International Conference, EMO 2005*, pages 428–442, Guanajuato, México, March 2005. Springer. Lecture Notes in Computer Science Vol. 3410.
- [4] P. A. N. Bosman. *Design and Application of Iterated Density-Estimation Evolutionary Algorithms*. PhD thesis, Institute of Information and Computing Sciences, Universiteit Utrecht, Utrecht, The Netherlands, 2003.
- [5] M. Costa, E. Minisci, and E. Pasero. An hybrid neural/genetic approach to continuous multi-objective optimization problems. In B. Apolloni, M. Marinaro, and R. Tagliaferri, editors, *Italian Workshop on Neural Neural Nets (WIRN)*, volume 2859 of *Lecture Notes in Computer Science*, pages 61–69. Springer, 2003.
- [6] K. Deb. *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley & Sons, Chichester, UK, 2001. ISBN 0-471-87339-X.
- [7] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler. Scalable test problems for evolutionary multiobjective optimization. In A. Abraham, L. Jain, and R. Goldberg, editors, *Evolutionary Multiobjective Optimization: Theoretical Advances and Applications*, Advanced Information and Knowledge Processing, pages 105–145. Springer Verlag, 2004.
- [8] B. Fritzsche. A growing neural gas network learns topologies. In G. Tesauero, D. S. Touretzky, and T. K. Leen, editors, *Advances in Neural Information Processing Systems*, volume 7, pages 625–632. MIT Press, Cambridge, MA, 1995.
- [9] V. Khare, X. Yao, and K. Deb. Performance Scaling of Multi-objective Evolutionary Algorithms. In C. M. Fonseca, P. J. Fleming, E. Zitzler, K. Deb, and L. Thiele, editors, *Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003*, pages 376–390, Faro, Portugal, April 2003. Springer. Lecture Notes in Computer Science. Volume 2632.
- [10] J. Knowles, L. Thiele, and E. Zitzler. A tutorial on the performance assessment of stochastic multiobjective optimizers. TIK Report 214, Computer Engineering and Networks Laboratory (TIK), ETH Zurich, 2006.
- [11] P. Larrañaga and J. A. Lozano, editors. *Estimation of Distribution Algorithms. A new tool for Evolutionary Computation*. Genetic Algorithms and Evolutionary Computation. Kluwer Academic Publishers, Boston/Dordrecht/London, 2002.
- [12] L. Martí, J. García, A. Berlanga, and J. M. Molina. A cumulative evidential stopping criterion for multiobjective optimization evolutionary algorithms. In *GECCO '07: Proceedings of the 9th annual conference on Genetic and evolutionary computation*, pages 911–911, New York, NY, USA, 2007. ACM Press.
- [13] L. Martí, J. García, A. Berlanga, and J. M. Molina. Model-building algorithms for multiobjective EDAs: Directions for improvement. In Z. Michalewicz, editor, *IEEE Conference on Evolutionary Computation (CEC), part of 2008 IEEE World Congress on Computational Intelligence (WCCI 2008)*. IEEE Press, 2008.
- [14] T. M. Martinetz. Competitive Hebbian learning rule forms perfectly topology preserving maps. In *International Conference on Artificial Neural Networks (ICANN'93)*, pages 427–434, Amsterdam, 1993. Springer-Verlag.
- [15] T. M. Martinetz, S. G. Berkovich, and K. J. Shulten. Neural-gas network for vector quantization and its application to time-series prediction. *IEEE Transactions on Neural Networks*, 4:558–560, 1993.
- [16] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, San Francisco, CA, 1988.
- [17] M. Pelikan, K. Sastry, and D. E. Goldberg. Multiobjective estimation of distribution algorithms. In M. Pelikan, K. Sastry, and E. Cantú-Paz, editors, *Scalable Optimization via Probabilistic Modeling: From Algorithms to Applications*, Studies in Computational Intelligence, pages 223–248. Springer-Verlag, 2006.
- [18] R. C. Purshouse and P. J. Fleming. On the evolutionary optimization of many conflicting objectives. *IEEE Transactions on Evolutionary Computation*, 11(6):770–784, 2007.
- [19] A. K. Qin and P. N. Suganthan. Robust growing neural gas algorithm with application in cluster analysis. *Neural Networks*, 17(8–9):1135–1148, 2004.
- [20] H. Timm, C. Borgelt, C. Doring, and R. Kruse. An extension to possibilistic fuzzy cluster analysis. *Fuzzy Sets and Systems*, 147(1):3–16, October 2004.
- [21] Q. Zhang, A. Zhou, and Y. Jin. RM-MEDA: A regularity model-based multiobjective estimation of distribution algorithm. *IEEE Transactions on Evolutionary Computation*, 12(1):41–63, 2008.