This document is published in:

Operations Research Letters (2011), 39 (2), pp. 150-154.

DOI:10.1016/j.orl.2011.01.002

MB-GNG: Addressing drawbacks in multi-objective optimization estimation of distribution algorithms

Luis Martí [a,*], Jesús García [a], Antonio Berlanga [a], Carlos A. Coello Coello [b], José M. Molina [a]

[a] Group of Applied Artificial Intelligence, Department of Informatics, Universidad Carlos III de Madrid. Av. de la Universidad Carlos III, 22. Colmenarejo 28270 Madrid, Spain
[b] Department of Computer Science, CINVESTAV-IPN, Av. IPN No. 2508, Col. San Pedro Zacatenco México, D.F. 07360, Mexico

A B S T R A C T

We examine the model-building issue related to multi-objective estimation of distribution algorithms (MOEDAs) and show that some of their, as yet overlooked, characteristics render most current MOEDAs unviable when addressing optimization problems with many objectives. We propose a novel model-building growing neural gas (MB-GNG) network that is specially devised for properly dealing with that issue and therefore yields a better performance. Experiments are conducted in order to show from an empirical point of view the advantages of the new algorithm.

Keywords:
Multi-objective optimization
Estimation of distribution algorithm
Model building
Growing neural gas

1. Introduction

Most human endeavors involve the creation of artifacts with properties that must be tuned to be as efficient as possible. This fact has prompted the creation of a number of interrelated research areas like optimization, mathematical programming, operational research and decision-making. Although these areas share some of their goals, each of them differs from the others in the approaches put forward by their respective communities and the characteristics of the problems they deal with.

Many real-world optimization problems involve more than one goal to be optimized. This type of problems is known as *multi-objective optimization problems* (MOPs). A MOP can be expressed as the problem in which a set of *objective functions* $f_1(\boldsymbol{x}), \ldots, f_M(\boldsymbol{x})$ should be jointly optimized;

$$\min \boldsymbol{F}(\boldsymbol{x}) = \langle f_1(\boldsymbol{x}), \ldots, f_M(\boldsymbol{x}) \rangle; \quad \boldsymbol{x} \in \mathcal{S}; \tag{1}$$

where $\mathcal{S} \subseteq \mathbb{R}^n$ is known as the *feasible set* and could be expressed as a set of restrictions over the decision set, $\mathbb{R}^n$. The image set of $\mathcal{S}$ produced by function vector $\boldsymbol{F}(\cdot)$, $\mathcal{O} \subseteq \mathbb{R}^M$, is called *feasible objective set* or *criterion set*.

The solution to this type of problem is a set of trade-off points. The optimality of a solution can be expressed in terms of the Pareto dominance relation.

Definition 1 (*Pareto Dominance Relation*)**.** For the optimization problem specified in (1) and having $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{S}$, $\boldsymbol{x}$ is said to dominate $\boldsymbol{y}$ (expressed as $\boldsymbol{x} \prec \boldsymbol{y}$) iff $\forall f_j, f_j(\boldsymbol{x}) \leq f_j(\boldsymbol{y})$ and $\exists f_i$ such that $f_i(\boldsymbol{x}) < f_i(\boldsymbol{y})$.

Definition 2 (*Non-Dominated Subset*)**.** In problem (1) and having the set $\mathcal{A} \subseteq \mathcal{S}$. $\hat{\mathcal{A}}$, the *non-dominated subset* of $\mathcal{A}$, is defined as

$$\hat{\mathcal{A}} = \left\{ \boldsymbol{x} \in \mathcal{A} \mid \nexists \boldsymbol{x}' \in \mathcal{A} : \boldsymbol{x}' \prec \boldsymbol{x} \right\}.$$

The solution of (1) is $\hat{\mathcal{S}}$, the non-dominated subset of $\mathcal{S}$. $\hat{\mathcal{S}}$ is known as the *efficient set* or *Pareto-optimal set* [4]. If problem (1) has certain characteristics, e.g., linearity or convexity of the objective functions or convexity of $\mathcal{S}$, the efficient set can be determined by mathematical programming approaches [4]. However, in the general case, finding the solution of (1) is an *NP*-complete problem [2]. In this case, heuristic or metaheuristic methods can be applied in order to have solutions of practical value at an admissible computational cost.

A broad range of heuristic and metaheuristic approaches has been used to address MOPs [4]. Of these, multi-objective evolutionary algorithms (MOEAs) [5] have been found to be a competent approach in a wide variety of application domains. Their main advantages are ease of use, inherent parallel search and lower susceptibility to the shape or continuity of the image of the efficient set, compared with traditional mathematical programming techniques for multi-objective optimization [4].

There is a class of MOPs that are particularly appealing because of their inherent complexity: the so-called *many-objective problems*. These are problems with a relatively large number of objectives (normally, four or more). Although somewhat counter-intuitive and hard to visualize for a human decision maker, these problems are not uncommon in real-life engineering practice. For example, [14] details some relevant real problems of this type.

* Corresponding author.
E-mail addresses: lmarti@inf.uc3m.es (L. Martí), jgherrer@inf.uc3m.es (J. García), aberlan@ia.uc3m.es (A. Berlanga), ccoello@cs.cinvestav.mx (C.A. Coello Coello), molina@ia.uc3m.es (J.M. Molina).

The scalability issues of traditional MOEAs in these problems have triggered a sizable amount of research, aiming to provide alternative approaches that can properly handle many-objective problems and perform reasonably well.

Estimation of distribution algorithms (EDAs) are one such approaches [11]. EDAs have been hailed as a paradigm shift in evolutionary computation. They build a model of the population instead of applying evolutionary operators. This model is then used to synthesize new individuals. EDAs have been extended to the multi-objective optimization problem domain as multi-objective EDAs (MOEDAs).

Although MOEDAs have yielded some encouraging results, their introduction has not lived up to *a priori* expectations. This can be attributed to a number of different causes. We have recognized three of them, in particular, those derived from the incorrect treatment of population outliers; the loss of population diversity, and that too much computational effort is being spent on finding an optimal population model.

A number of works have dealt with the issues listed above, particularly with loss of diversity. Nevertheless, in our opinion, the community has failed to acknowledge that the underlying cause for all those problems could, perhaps, be traced back to the algorithms used for model building in EDAs.

In this paper we examine the model-building issue of current MOEDAs and show that some of its characteristics, which have been disregarded so far, render most current approaches unsuitable for tackling MOPs. We then propose a novel model-building algorithm, based on the growing neural gas (GNG) network. This model-building GNG (MB-GNG) is the main contribution of this paper. It has been devised with this particular problem in mind, and therefore addresses the problems of current approaches.

The remainder of this paper is organized as follows. Section 2 serves as a brief introduction to MOEDAs and the issues present in current model-building algorithms. After this, MB-GNG is described in Section 4. Then, in Section 5, a comparative study is carried out in order to establish from an experimental point of view the improvements introduced by MB-GNG with respect to similar algorithms. Finally, some conclusive remarks are put forward.

2. Multi-objective estimation of distribution algorithms

Estimation of distribution algorithms (EDAs) are population-based optimization algorithms that rely on machine learning methods. The introduction of machine learning implies that these new algorithms lose the straightforward biological inspiration of their predecessors. Nonetheless, they gain the capacity of scalably solving many challenging problems, in some cases significantly outperforming standard EAs and other optimization techniques.

Most multi-objective EDAs (MOEDAs) consist of a modification of existing EDAs whose fitness assignment function is substituted by one taken from an existing MOEA.

In general terms, MOEDAs follow a common algorithmic scheme. At a given iteration t, a MOEDA has a population $\mathcal{P}_t$ of individuals, each one representing a point in the search space. In every iteration, $\mathcal{P}_t$ elements are ranked according to a given fitness assignment function. A subset $\mathcal{M}_t$, with the best elements of $\mathcal{P}_t$ is computed. The model-building algorithm relies on $\mathcal{M}_t$ to create a model of the best part of the population. This model is sampled in order to create new elements which are combined with $\mathcal{P}_t$ to create the population to be used in the next iteration, $\mathcal{P}_{t+1}$.

A given stopping criterion determines when the optimization process should be interrupted. When this happens, $\hat{\mathcal{P}}$, the non-dominated subset of $\mathcal{P}_t$, is returned as the solution.

Although there are different approaches for determining $\mathcal{M}_t$ and $\mathcal{P}_{t+1}$, MOEDAs are better characterized by their two main components, the fitness assignment function and the model-building algorithm.

Fitness functions have been mostly taken from MOEAs. It should be noted that the Pareto dominance-based approach proposed by the NSGA-II algorithm is, by far, the most popular in the current literature.

The model-building algorithm is the kernel of an EDA. There are two main types of methods for addressing this problem: those based on graphical models and those based on mixture distributions.

2.1. Graphical model MOEDAs

Most EDAs based on graphical models rely on Bayesian networks. From these, the Bayesian optimization algorithm (BOA) [11] is the specific approach that has been extrapolated to the multi-objective domain. The exhaustive synthesis of a Bayesian network from the algorithm's population is an NP-hard problem [7]. Therefore, these EDAs must employ heuristic alternatives for building their networks while keeping the computational cost under reasonable margins.

BOA-based MOEDAs combine the Bayesian model-building scheme with an already existing Pareto-based fitness assignment. This is the case of the multi-objective BOA (mBOA) that exploits the fitness assignment used in NSGA-II. Another algorithm based on hierarchical BOA (hBOA), called mhBOA, also uses the same form of fitness assignment but introduces clustering in the feasible objective set. A similar idea is proposed by combining the mixed BOA (mBOA) with SPEA2's selection scheme [5] to form the multi-objective mBOA (mmBOA). The multi-objective real BOA (MrBOA) [1] also extends a preexisting EDA, namely, the real BOA (rBOA). MrBOA combines the fitness assignment of NSGA-II with rBOA.

2.2. Mixture distribution MOEDAs

Another approach to modeling the subset with the best population elements is to apply a distribution mixture approach. Bosman and Thierens [3] proposed several variants of their multi-objective mixture-based iterated density estimation algorithm (MIDEA). They are based on their IDEA framework. They also introduced a novel Pareto-based and diversity-preserving fitness assignment function. The model construction is inherited from the single-objective version. The proposed MIDEAs considered several types of probabilistic models for both discrete and continuous problems.

MIDEAs do not provide a specific mechanism to ensure equal coverage of the Pareto-optimal front if the number of representatives in some parts of the front is much larger than the number of representatives in some other parts. The clustering algorithms applied for this task include the randomized leader algorithm, the k-means algorithm and the expectation maximization (EM) algorithm [15].

2.3. Other MOEDA approaches

There are some other approaches for model building. For example, the regularity model-based multi-objective estimation of distribution algorithm (RM-MEDA) [16] is based on the regularity property derived from the Karush–Kuhn–Tucker condition. Covariance matrix adaptation evolution strategies (CMA-ES) [9] have been also used in the multi-objective context. CMA-ES consists of a method for updating the covariance matrix of the multivariate normal mutation distribution used in an evolution strategy. They can be viewed as EDAs, as new individuals are sampled according to the mutation distribution.

3. Model-building in the multi-objective case

Regardless of the many efforts at providing usable model-building methods for EDAs, the nature of the problem itself has received relatively little attention. Generally, model-building algorithms are off-the-shelf machine learning methods that were originally intended for other classes of problems. On the other hand, the model-building problem has particular requirements that the above methods do not meet and may even go against.

In this paper we argue that the model-building problem has not been properly identified. For this reason, it has been treated like other previously existing problems overlooking that this problem has particular requirements. This matter did not show up as clearly in single-objective EDAs. Thanks to the extension to the multi-objective domain this issue has become more evident, as we will debate in the remainder of this section.

There are at least three properties of current model-building approaches that hinder their performance, in particular,

1. the incorrect treatment of data outliers;
2. the loss of population diversity; and
3. the excess of computational effort devoted to finding an optimal population model.

The data outliers' issue is a good example of the defective understanding of the nature of the model-building problem, and is, in our opinion the cornerstone for reaching a better understanding of the problem.

In machine-learning practice, outliers are handled as noisy, inconsistent or irrelevant data. Therefore, outlying data is expected to have little influence on the model or it is just disregarded. However, this behavior is not appropriate for model building. In this case, it is known beforehand that all elements in the data set should be taken into account, as they represent newly discovered or candidate regions of the search space and, therefore, must be explored. Therefore, these instances should be at least equally represented by the model and perhaps even reinforced.

Another weakness of most MOEDAs (and most EDAs, for that matter) is the loss of population diversity. This is a point that has already been made, and some proposals for addressing the issue have been laid out [1,13]. This loss of diversity can be traced back to the above outliers' issue of model-building algorithms. The repetitive application of an algorithm that disregards outliers tends to generate more individuals in areas of the search space that are more densely represented. Although there have been some proposals to circumvent this problem, we take the view that the ultimate solution is the use of an appropriate algorithm.

The third issue to be dealt with is the computational resources wasted on finding an optimal description for the subpopulation being modeled. In the model-building case, optimal model complexity can be sacrificed in the interests of a faster algorithm. This is because the only constraint is to have a model that is sufficiently, but not necessarily optimal in terms of complexity, in order to represent the data in a correct manner.

In conclusion, we can deduce that understanding the nature of the model-building problem and the application of suitable algorithms seem to point the way forward in this area.

4. Model-building growing neural gas

Clustering algorithms [15] 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.

The growing neural gas (GNG) network [8] has been chosen as a starting point after surveying the literature for suitable candidates. GNG networks are intrinsic self-organizing neural networks based on the neural gas [10] model. The term "neural gas" refers to the behavior of the center of the nodes during the adaptation process, which distribute themselves like a gas within an imaginary container defined by the bounds implicitly given by the dataset on which the network is being trained.

Among the vast number of existing clustering methods we decided to base our approach on GNG because of its interesting properties, in particular:

- the network has been shown to be sensitive to outliers [12], something undesirable in typical applications but suitable for model-building;
- the network adapts its topology automatically to meet the complexity of the problem being solved;
- 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 [10];
- although it benefits from the topological ordering of the nodes it does not suffer the problem associated to Kohonen networks, and;
- the introduction of a novel cluster repulsion mechanism fosters the exploration of the input space.

Our model-building GNG (MB-GNG) is an extension of the original GNG. It introduces a cluster repulsion term that fosters a better spread of the clusters along the training dataset, as explained in [12].

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 $\mathcal{C} = \{c_1, \ldots, c_{N^*}\}$, with $N_0 \leq N^* \leq N_{\max}$. Here N_0 and $N_{\max}$ represent initial and maximal number of nodes in the network. The network receives inputs $\boldsymbol{x} \in \mathcal{I}$ in its input set, $\mathcal{I}$. In our case this input set is the decision set: $\mathcal{I} = \mathbb{R}^n$.

A node c_i describes a local multivariate Gaussian density that consists of a center, $\boldsymbol{\mu}_i$, and a standard deviations vector, $\boldsymbol{\sigma}_i$. It also has 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, ν_{ij}.

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

$$b = \arg\min_{i=1,\ldots,N^*} d\left(\boldsymbol{\mu}_i, \boldsymbol{x}\right),$$

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

$$b' = \arg\min_{i=1,\ldots,N^*; i\neq b} d\left(\boldsymbol{\mu}_i, \boldsymbol{x}\right).$$

Here $d(\boldsymbol{a}, \boldsymbol{b})$ is a metric, in our case, the Euclidean one.

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

At this point, the age of all edges is incremented by one. If an edge is older than the maximum age, $\nu_{ij} > \nu_{\max}$, then the edge is removed. If a node becomes isolated from the rest of the nodes because no edges connecting it remain, it is also deleted.

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

$$\Delta\xi_b = d\left(\boldsymbol{\mu}_i, \boldsymbol{x}\right)^2.$$

After that, learning takes place in the best-matching node and its neighbors with rates ϵ_{best} and ϵ_{vic} ($\epsilon_{\text{best}} > \epsilon_{\text{vic}}$), respectively.

These two rates control the movement of the centers of the nodes involved towards the current input $\boldsymbol{x}$.

For c_b, adaptation follows the rule originally used by GNG,

$$\Delta\boldsymbol{\mu}_b = \epsilon_{\text{best}}\left(\boldsymbol{x} - \boldsymbol{\mu}_b\right).$$

However, for the neighbors of c_b, a cluster repulsion term is added to the original formulation. This repulsion term avoids the meaningless concentration of nodes in the data space and therefore, promotes a proper representation of the data set with fewer nodes.

Following that, the learning rule for those nodes can be expressed as, $\forall c_v \in \mathcal{V}_b$,

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

This approach was already used as part of the robust GNG [12] and it has proven itself useful for obtaining a good spread of the clusters in the inputs' space. In the aforementioned work, it was stated that the adaptation rule is not sensitive to its parameters. We have set them to $\beta = 2$ and $\zeta = 0.1$ as suggested in [12].

After a given number, T_+, of dataset iterations have taken place, it can be assumed that there is enough information stored in the error accumulators, ξ_i. This information is used to determine where to add new nodes to the network. In particular, if the current iteration is an integer multiple of T_+ and the network has not reached its maximum size ($N^* < N_{\max}$) then a new node is inserted in the network.

First, the node with the largest error, c_e, is selected. 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,

$$\boldsymbol{\mu}_{N^*} = 0.5\left(\boldsymbol{\mu}_e + \boldsymbol{\mu}_{e'}\right); \qquad \xi_{N^*} = 0.5(\xi_e + \xi_{e'}).$$

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 decreased

$$\xi_e = \delta_I \xi_e, \qquad \xi_{e'} = \delta_I \xi_{e'},$$

by a rate $0 \leq \delta_I \leq 1$. Finally, the errors of all nodes are decreased by a factor δ_G,

$$\xi_i = \delta_G \xi_i, \quad i = 1, \ldots, N^*.$$

The algorithm stops after a learning epoch if the standard deviation of the accumulated errors is smaller than a certain threshold, ρ,

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

This means that it will stop when the outliers are as well 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 normal estimator of the deviations. The local Gaussian densities resulting from the described algorithm can be combined to synthesize the Gaussian mixture with parameters $\boldsymbol{\Theta}$,

$$P\left(\boldsymbol{x}|\boldsymbol{\Theta}\right) = \frac{1}{N^*}\sum_{i=1}^{N^*} P\left(\boldsymbol{x}|\boldsymbol{\mu}_i, \boldsymbol{\sigma}_i\right).$$

Each Gaussian density is formulated as

$$P\left(\boldsymbol{x}|\boldsymbol{\mu}_i, \boldsymbol{\sigma}_i\right) = \frac{1}{(2\pi)^{n/2}|\boldsymbol{\Sigma}_i|^{1/2}} \times \exp\left(-\frac{1}{2}\left(\boldsymbol{x} - \boldsymbol{\mu}_i\right)^\top \boldsymbol{\Sigma}_i^{-1}(\boldsymbol{x} - \boldsymbol{\mu}_i)\right),$$

with the covariance matrices $\boldsymbol{\Sigma}_i$ defined as a diagonal matrix with its non-zero elements set to the values of the deviations $\boldsymbol{\sigma}_i$. The Gaussian mixture can be used by the EDA to generate new individuals. These new individuals are created by sampling $P\left(\boldsymbol{x}|\boldsymbol{\Theta}\right)$.

5. Experimental study

A comprehensive comparative study is essential in order to verify that the novel MB-GNG actually yields a substantial improvement. In these experiments we compare MB-GNG and some of the alternative model-building algorithms under a common EDA framework like the one presented in Section 2. The model-building algorithms that also take part in the tests are: Bayesian networks, as used in MrBOA; randomized leader algorithm, k-means algorithm and EM algorithm, as described for MIDEAs; $(1 + \lambda)$-CMA-ES as described in [9]; and the original GNG. Furthermore, the NSGA-II and SPEA2 MOEAs were also involved in the experiments in order to provide a baseline for the comparisons. The performance of each algorithm is assessed in terms of approximation to the Pareto-optimal front and the computational cost of each one.

Six well-known community-accepted problems are addressed: the WFG4 to WFG9 problems [5]. These problems pose different classes of challenges, like local-optima, parameter-bias, uni- and multi-modality, etc., to the optimizer. See the corresponding paper for a full description of each one. Each problem is configured with 3, 5, 7 and 9 objective functions. For all cases, the decision space dimension was set at 15. All the algorithms were executed 30 times for each problem/dimension pair.

The quality of the solutions is determined by the use of the hypervolume indicator [5]. This is the only indicator that has the properties of a metric and the only to be strictly Pareto monotonic. However, it must be noted that there are other alternatives.

Statistical hypotheses tests have to be applied to establish the validity of the results. For this, we perform a Kruskal–Wallis test [6] with the indicator values yielded by each algorithm's run for each problem/dimension combination. In the context of these experiments, the null hypothesis for the test was that all algorithms were equally capable of solving the problem. If the null hypothesis was rejected, which was the case in all the experimental instances, the Conover–Inman procedure [6, pp. 288–290] was applied in a pairwise manner to determine if the results of one algorithm were significantly better than those of the other. A significance level, α, of 0.05 was used for all the tests.

Understanding these results in a one-by-one basis is rather cumbersome as it implies cross-examining and comparing the results presented separately and would require larger amounts of space and effort. That is why we decided to adopt a more integrative representation.

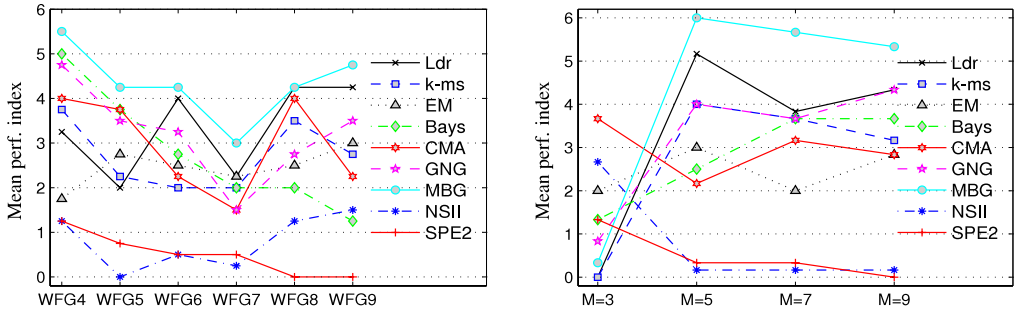
That is, for a given set of algorithms $A_1, \ldots, A_K$, a set of P test problem instances $\Phi_{1,m}, \ldots, \Phi_{P,m}$, configured with m objectives, the function $\delta(\cdot)$ is defined as

$$\delta\left(A_i, A_j, \Phi_{p,m}\right) = \begin{cases} 1 & \text{if } A_i \gg A_j \text{ solving } \Phi_{p,m} \\ 0 & \text{in other case,} \end{cases}$$

where the relation $A_i \gg A_j$ defines if A_i is significantly better than A_j when solving the problem instance $\Phi_{p,m}$, as computed by the statistical tests previously described.

Relying on $\delta(\cdot)$, the performance index $P_{p,m}(A_i)$ of a given algorithm A_i when solving $\Phi_{p,m}$ is then computed as

$$P_{p,m}\left(A_i\right) = \sum_{j=1:j\neq i}^{K} \delta\left(A_i, A_j, \Phi_{p,m}\right).$$



(a) $\overline{P}_p$ (), mean values for each problem. (b). $\overline{P}_m$ (), mean values for each dimension.

Fig. 1. Mean values of the performance index across the different problems and objective space dimensions. Algorithms involved are the leader algorithm (Ldr), the *k*-means (k-ms), EM, Bayesian networks (Bays), CMA-ES (CMA), GNG and the model-building GNG (MBG). NSGA-II (NSII) and SPEA2 (SPE2) are also included for comparison reasons.

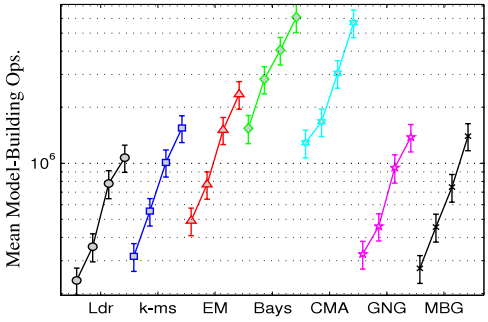


Fig. 2. Progression of the mean floating-point CPU operations used by the model-building algorithms as the objective space dimension increases. For each algorithm four points are plotted corresponding to $M = \{3, 5, 7, 9\}$. See Fig. 1 for a description of the acronyms.

This index intends to summarize the performance of each algorithm with regard to its peers.

Fig. 1 exhibits the results computing the performance indexes. Fig. 1(a) represents the mean performance indexes yielded by each algorithm when solving each problem in all of its configured objective dimensions,

$$\overline{P}_p\,(A_i) = \frac{1}{|\mathcal{M}|}\sum_{m\in\mathcal{M}} P_{p,m}\,(A_i)\,; \quad \mathcal{M} = \{3, 5, 7, 9\}\,.$$

It is worth noticing that GNG and, particularly, MB-GNG have better overall results than the other algorithms. It is somewhat unexpected that the randomized leader and the *k*-means algorithms do not have a very good overall performance for some problems, like WFG5 and WFG7 for the randomized leader and WFG8 and WFG9 for *k*-means. A possible hypothesis is that these results may be biased by the three-objective problems, where there are sizable differences compared with the results of the other dimensions.

This situation is clarified in Fig. 1(b), which presents the mean values of the index computed for each dimension,

$$\overline{P}_m\,(A_i) = \frac{1}{P}\sum_{p=1}^{P} P_{p,m}\,(A_i)\,.$$

In this representation, it becomes noticeable that MB-GNG outperforms the other approaches in dimensions larger than three, with the exception of the nine objectives case, in which the original GNG outperforms MB-GNG.

Another key issue when dealing with high-dimensional problems is the computational cost of the algorithms. One simple way of inspecting this point is to compute the number of CPU operations dedicated to model building in each case. Fig. 2 summarizes these results. NSGA-II and SPEA2 are not included in the analysis, since they do not perform any model building. The main conclusion in this case is that MB-GNG requires more or less the same amount of resources to yield better results than the other approaches.

Acknowledgements

The authors wish to thank the referee and the associate editor assigned to this paper for their comments and suggestions. They helped to substantially improve the paper. They also wish to thank Prof. Elisenda Molina for her assistance in the preparation of the manuscript. LM, JG, AB and JMM were supported by projects CICYT TIN2008-06742-C02-02/TSI, CICYT TEC2008-06732-C02-02/TEC, SINPROB, CAM CONTEXTS S2009/TIC-1485 and DPS2008-07029-C02-02. CACC was supported by CONACyT project 103570.

References

[1] C.W. Ahn, R.S. Ramakrishna, Multiobjective real-coded Bayesian optimization algorithm revisited: Diversity preservation, in: GECCO'07: Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation, ACM Press, New York, NY, USA, 2007, pp. 593–600.
[2] T. Bäck, Evolutionary Algorithms in Theory and Practice, Oxford University Press, New York, Oxford, 1996.
[3] P.A.N. Bosman, D. Thierens, The balance between proximity and diversity in multiobjective evolutionary algorithms, IEEE Transactions on Evolutionary Computation 7 (2003) 174–188.
[4] J. Branke, K. Miettinen, K. Deb, R. Słowiński (Eds.), Multiobjective Optimization, in: Lecture Notes in Computer Science, vol. 5252, Springer-Verlag, Berlin/Heidelberg, 2008.
[5] C.A. Coello Coello, G.B. Lamont, D.A. Van Veldhuizen, Evolutionary algorithms for solving multi-objective problems, in: Genetic and Evolutionary Computation, 2nd ed., Springer, New York, 2007.
[6] W.J. Conover, Practical Nonparametric Statistics, 3rd ed., John Wiley & Sons, New York, 1999.
[7] P. Dagum, M. Luby, Approximating probabilistic inference in Bayesian belief networks is NP-hard, Artificial Intelligence 60 (1993) 141–153.
[8] B. Fritzke, A growing neural gas network learns topologies, in: G. Tesauro, D.S. Touretzky, T.K. Leen (Eds.), in: Advances in Neural Information Processing Systems, vol. 7, MIT Press, Cambridge, MA, 1995, pp. 625–632.
[9] C. Igel, N. Hansen, S. Roth, Covariance matrix adaptation for multi-objective optimization, Evolutionary Computation 15 (2007) 1–28.
[10] T.M. Martinetz, S.G. Berkovich, K.J. Shulten, Neural-gas network for vector quantization and its application to time-series prediction, IEEE Transactions on Neural Networks 4 (1993) 558–560.
[11] M. Pelikan, K. Sastry, D.E. Goldberg, Multiobjective estimation of distribution algorithms, in: M. Pelikan, K. Sastry, E. Cantú-Paz (Eds.), Scalable Optimization via Probabilistic Modeling: From Algorithms to Applications, in: Studies in Computational Intelligence, Springer-Verlag, 2006, pp. 223–248.
[12] A.K. Qin, P.N. Suganthan, Robust growing neural gas algorithm with application in cluster analysis, Neural Networks 17 (2004) 1135–1148.
[13] J. Shapiro, Diversity loss in general estimation of distribution algorithms, in: Parallel Problem Solving from Nature (PPSN IX), pp. 92–101.
[14] T.J. Stewart, O. Bandte, H. Braun, N. Chakraborti, M. Ehrgott, M. Göbelt, Y. Jin, H. Nakayama, S. Poles, D. Di Stefano, Real-world applications of multiobjective optimization, in: J. Branke, M. Kaisa, K. Deb, R. Słowiński (Eds.), Multiobjective Optimization, in: Lecture Notes in Computer Science, vol. 5252, Springer-Verlag, Berlin/Heidelberg, 2008, pp. 285–327.
[15] R. Xu, D. Wunsch II, Clustering, in: IEEE Press Series on Computational Intelligence, illustrated ed., Wiley, IEEE Press, New York, 2008.
[16] Q. Zhang, A. Zhou, Y. Jin, RM-MEDA: A regularity model-based multiobjective estimation of distribution algorithm, IEEE Transactions on Evolutionary Computation 12 (2008) 41–63.