3

Evolutionary Multi-Objective Algorithms

Aurora Torres, Dolores Torres, Sergio Enriquez,
Eunice Ponce de León and Elva Díaz
University of Aguascalientes,
México

1. Introduction

The versatility that genetic algorithm (GA) has proved to have for solving different problems, has make it the first choice of researchers to deal with new challenges. Currently, GAs are the most well known evolutionary algorithms, because their intuitive principle of operation and their relatively simple implementation; besides they have the ability to reflect the philosophy of evolutionary computation in an easy and quick way.

As time goes by, human beings are more sophisticated. Every time we demand better performance of the equipment and techniques in the solution of more complex problems; forcing problem-solvers to use non-exhaustive solution techniques, although this could means the loss of accuracy. Non conventional techniques provide a solution in a suitable time when other techniques can be extraordinarily slow. Evolutionary algorithms are metaheuristics inspired on Darwin's theory of the survival of the fittest. A feature shared by these algorithms is that they are population-based, so each population represents a group of possible solutions to the problem posed; and only will transcend to the next generation those individuals with the best performance. At the end of the evolutionary process, the population is formed by the better individuals only. In general, all metaheuristics have shown their efficiency in solving complex optimization problems with one goal, so having to work simultaneously with more than one target, and therefore having to determine not only one answer but a set of them; population-based metaheuristics like evolutionary algorithms seem to be the most natural technique to address this type of optimization.

This chapter presents the theoretical description of the multi-objective optimization problem and establishes some important concepts. Later the most well known algorithms that initially were used for solving this problem are presented. Among these algorithms excels the GA and some modifications to it. The chapter also briefly discusses the estimation of the distribution algorithm (EDA), which was also inspired on the GA. Subsequently, the drawing graphs problem is established and solved. This problem, like many other of real life is inherently multi-objective. The proposed solution to this problem uses a hybrid EDA combined with a hill-climbing algorithm, which handled three simultaneous objectives: minimizing the number of crossing edges in the graph (total number of crossing edges of the graph have to be minimized), minimizing the graph area (total space used by the graph have to be as small as possible) and minimizing the graph aspect ratio (the graph have to be in a perfect square Visualized area). This section includes the description of the used approach and a group of experimental results, as well as some conclusions and future work. Finally, the last section of this chapter is a brief reflection on the future of multi-objective optimization research. On it, we capture some concerns and issues that are relevant to the development of this area.

2. Multi-objective optimization

Optimization in both mathematics and computing, refers to the determination of one or more feasible solutions that corresponds to an extreme value (maximum or minimum), according to one or more objective functions. To find the extreme solutions of one or more objective functions can be applied in a wide range of practical situations, such as to minimize the manufacturing cost of a product, to maximize profit, to reduce uncertainty, and so on. The principles and methods of optimization are used in solving quantitative problems in disciplines such as physics, biology, engineering, economics, and others. The simplest optimization problems involve functions of a single variable and can be solved by differential calculus. When researchers work with optimization, we could find two main types: mono-objective optimization and multi-objective optimization (MOO), depending on the number of optimization functions. The optimization can be subject to one or several constraints. The constraints are conditions that limit the selection of the values variables can take. This area has been approached for different techniques and methods.

Probably, the main difficulty of modelling mono-objective problems consists on obtaining just one equation for the complete problem. This stage could be too complicated to reach (Collette & Siarry, 2002). Due to the difficulty of finding an equation for a problem where many factors can influence, multi-objective optimization gives a very important advantage. Nevertheless, multi-objective optimization let us use some equations for reaching more than one objective; this property adds complexity to the model. As complexity of problems is increased, it is necessary to use new tools; for example: lineal programming that was created to solve optimization problems that involve two or more entrance variables.

2.1 Global optimization

Global optimization is the process of finding the global maximum or minimum (it will depend on the problem to be solved), inside a space S. Formally, it could be defined as (Bäck, 1996):

Definition 1. Given a function $f(\vec{x}) : \Omega \subseteq S = \mathbb{R}^n \rightarrow \mathbb{R}$, $\Omega \neq \emptyset$, for $\vec{x} \in \Omega$ the value $f^* \triangleq f(\vec{x}^*) > -\infty$ is named the global minimum if and only if

$$\forall \vec{x} \in \Omega : f(\vec{x}^*) \leq f(\vec{x}) \tag{1}$$

This way, $\vec{x}$ is the global minimum, $f(\vec{x}^*)$ is the objective function and the set Ω is the feasible region inside the set S. The problem of determining the global minimum is called "*problem of global optimization*". When the problem to optimize is mono-objective, the solution is unique. But this is not the case of multi-objective optimization problems (MOOP), they usually give a group of solutions that satisfy all objectives presented in vectors. Then, the decision maker (the human with this work) selects one or more of that vectors which represent acceptable solutions of the problem according to their own point of view (Coello et al., 2002).

2.2 General multi-objective optimization problem

MOOP also called multi-criteria optimization, multi-performance or vector optimization problem, can be defined (in words) as the problem of finding a vector of decision variables which satisfies constraints and optimizes a vector function whose elements represent the objective functions (Osyczka, 1985). These functions form a mathematical description of performance criteria which are usually in conflict with each other. Hence, the term "*optimize*" means finding such a solution which would give the values of all the objective functions acceptable to the decision maker (Coello, 2001).

2.2.1 Decision variables

Decision variables are numeric values, which should be selected in a problem of optimization. These variables are represented for x_i where $i = 1,2,\dots,n$.

The vector of n decision variables $\vec{\mathrm{x}}$ is represented by:

$$\vec{\mathrm{x}} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \tag{2}$$

2.2.2 Constraints

Constraints imposed by the nature and environment of certain studied case, will be found in most of optimization problems. These conditions can be physical limitations, space or resistance obstacles, or restrictions in the time for the realization of a task, among others. So, certain solution is considered acceptable, if at least it satisfies these constraints. The constraints represent dependences between the parameters and the decision variables in the optimization problem. We can identify two different types of constraints; constraints of inequality:

$$\mathrm{g_i}(\vec{\mathrm{x}}) \leq 0 \qquad \mathrm{i} = 1,2,\dots,\mathrm{m} \tag{3}$$

and the equality constraints:

$$\mathrm{h_i}(\vec{\mathrm{x}}) = 0 \qquad \mathrm{i} = 1,2,\dots,\mathrm{p} \tag{4}$$

It is necessary to highlight that p should be smaller than n, because the number of equality constraints should be smaller than the number of decision variables, since if $p \geq n$ the problem is known as over constrained (Ramírez, 2007), and this means that will have more unknown variables than equations. Those constraints can be explicit (described by one algebraic expression), or implicit (in which case, an algorithm or method have to exist to calculate this constraints for any vector $\vec{v}$).

2.2.3 Objective functions

To know how good a solution is, it is necessary to have a criterion to evaluate it. This measure should be expressed as an algebraic function of the decision variables and it is known as objective function. It is possible that researches do not have this mathematical model, so, at least it is needed to have some mechanisms to determine the quality of the solutions, which can vary depending on the problem.

In many problems of the real world, objective functions are in conflict one to each other and even in the same problem some of them can be functions to minimize while the remaining ones have to be maximized. The vector of objective functions $\vec{f}(\ \vec{x}\ )$ is defined as follow:

$$\vec{f}(\vec{x}) = \begin{bmatrix} \vec{f}_1(\vec{x}) \\ \vec{f}_2(\vec{x}) \\ \vdots \\ \vec{f}_k(\vec{x}) \end{bmatrix} \tag{5}$$

The set where R denotes the real numbers by $\mathbb{R}^n$ is called Euclidian space of n dimensions. For the multi-objective optimization problem are considered two Euclidian spaces: the one of the decisions variables and the one of the objective functions. Each point in the first space represents a solution and it can be mapped in the space of the objective functions and then the quality of each solution can be determined. The general MOOP can be formally defined as:

Definition 2. Find the vector $\vec{x}^* = [x_1^*, x_2^*, \dots, x_n^*]^T$ which will satisfy the m inequality constraints:

$$g_i(\vec{x}) \leq 0 \quad i = 1,2,\dots,m \tag{6}$$

the p equality constraints

$$h_i(\vec{x}) = 0 \quad i = 1,2,\dots,p \tag{7}$$

and will optimize the vector function

$$\vec{f}(\vec{x}) = [f_1(\ \vec{x}\ ), f_2(\ \vec{x}\ ), \dots, f_k(\ \vec{x}\ )]^T \tag{8}$$

In other words, MOOP consists on determining the set of values for the decision variables $x_1^*, x_2^*, \dots, x_n^*$ which satisfy equations (6) and (7) and simultaneously optimize (8). Constraints given in (6) and (7) the feasible region of Ω and any point $\vec{x} \in \Omega$ is a feasible solution. The vector of functions $\vec{f}(\vec{x})$ map the group of feasible solutions Ω to the group of feasible objective functions. The k objective functions in the vector $\vec{f}(\vec{x})$ represent the criterion that can be expressed in different units. The restrictions $g_i(\vec{x})$ and $h_i(\vec{x})$ represent constraints applied to the decision variables. The vector $\vec{x}^*$ represents the group of optimal solutions.

2.3 Multi-objective optimization type of problems

In the area of multi-objective problems, three variants could be found; the first of them consists on minimizing the whole set of objective functions, the second consists on maximizing them and the third one is a mixture of minimization and maximization of the objective functions.

When we are in the third case, is very common that all the functions be transformed to their minimization version or maximization one, as it is preferred. So, the next equation can be used:

$$\max\left(\mathrm{f_i}(\vec{\mathrm{x}})\right) = -\min\left(-\mathrm{f_i}(\vec{\mathrm{x}})\right) \tag{9}$$

In the same way, inequality constraints (6) can be transformed multiplying by -1 and changing the sign of the inequality as follows:

$$-\mathrm{g_i}(\vec{\mathrm{x}}) \geq 0 \quad \mathrm{i} = 1,2,\dots,\mathrm{m} \tag{10}$$

2.4 The ideal vector

The ideal vector $\vec{\mathrm{f}}_\mathrm{i}^{\star}$ is formed as $\vec{\mathrm{f}}_\mathrm{i}^{\star} = \left[\vec{\mathrm{f}}_1^{\star}, \vec{\mathrm{f}}_2^{\star}, \cdots, \vec{\mathrm{f}}_\mathrm{k}^{\star}\right]^\mathrm{T}$, where $\mathrm{f_i}^{\star}$ denotes the optimal for the i-th objective function. If the objectives were not in conflict, then would exist a unique point $\vec{\mathrm{x}}^{\star}$ (in the space of the decision variables), but this situation is very exceptional in the real world.

The most accepted notion of optimum in the multi-objective environment was formulated by Francis Ysidro Edgeworth in 1881 and generalized after by Vilfredo Pareto in 1896.

2.5 Pareto – optimality

The concept of Pareto Optimum (also called Efficiency of Pareto, in honour of his discoverer, Vilfredo Pareto), is a concept of the economy with application in that discipline and in social sciences and engineering.

According to Pareto, a specific situation X is superior or preferable to other situation Y when the pass from Y to X supposes an improvement for all the members of the society, or an improvement for some, without the other ones be harmed. In other words, in economy and political economy, the concept of "*Optimum of Pareto*" simply indicates a situation in which cannot improve the situation of somebody without making worse the others' situation.

As already was said, the concept was born in economics, but its scope covers any situation with more than one objective to optimize.

Pareto optimality

We say that a vector of decision variables $\vec{\mathrm{x}}^* \in S$ is Pareto optimal if there is not another $\vec{\mathrm{x}} \in S$ such that $\mathrm{f_i}(\vec{\mathrm{x}}) \leq \mathrm{f_i}(\vec{\mathrm{x}}^*)$ for all i = 1, ... k and $\mathrm{f_j}(\vec{\mathrm{x}}) < \mathrm{f_j}(\vec{\mathrm{x}}^*)$ for at least one j. In other words, this definition establishes $\vec{\mathrm{x}}^*$ is Pareto optimal if there no exists a feasible vector of decision variables $\vec{x} \in S$ which would decrease some criterion without causing a simultaneous increase in at least one other criterion. Unfortunately, this concept almost always gives not a single solution, but rather a set of solutions called the Pareto optimal set. The vectors $\vec{\mathrm{x}}^*$ corresponding to the solutions included in the Pareto optimal set are called non-dominated ones. The plot of the objective functions whose non-dominated vectors are in the Pareto optimal set that is called the Pareto front (Coello, 2011).

2.6 Pareto dominance

Formally, it is said that a vector $\overline{u} = [\mathrm{u_1}, \mathrm{u_2}, \dots, \mathrm{u_k}]^\mathrm{T}$ dominates a vector $\overline{v} = [\mathrm{v_1}, \mathrm{v_2}, \dots, \mathrm{v_k}]^\mathrm{T}$ if and only if $\overline{u}$ is partially less than v. In other words:

$$\forall i \in \{1,2,\dots,k\}, u_i \le v_i \wedge \exists i \in \{1,2,\dots,k\} : u_i < v_i \qquad (11)$$

And it is denoted by: $\vec{u} \preceq \vec{v}$.

Considering a MOOP $\vec{f}(\vec{x})$, then the Pareto optimal set P^* is defined as:

$$\mathcal{P}^* = \{\vec{x} \in \Omega \mid \neg \exists\ \vec{y} \in \Omega\ \vec{f}(\vec{y}) \preceq \vec{f}(\vec{x})\} \qquad (12)$$

2.7 Pareto front

The Pareto Front concept is defined formally as follow:

Considering a MOOP $\vec{f}(\vec{x})$ and a Pareto optimal set P^*; the Pareto Front $\mathcal{PF}^*$ is defined as

$$\mathcal{PF}^* = \{\vec{f} = [f_1(\vec{x}),\ f_2(\vec{x}), \dots, f_k(\vec{x})]^T \mid \vec{x} \in \mathcal{P}^*\} \qquad (13)$$

Figures 1, 2, 3 and 4 show some Pareto fronts for two objective functions (f1 and f2). In all mentioned figures, the front is the set of points marked with a line. Figure 1 for example, presents the case in which both objective functions are minimized.

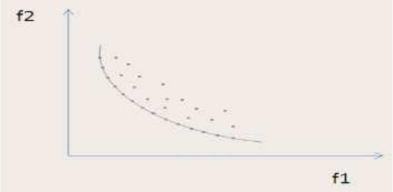


Fig. 1. Pareto front for the minimization of two objective functions (f1 and f2)

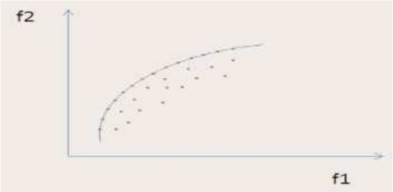


Fig. 2. Pareto front for the minimization of f1 and the maximization of f2

Figure 2 shows the Pareto front for the minimization of function f1 and the maximization of function f2. As the reader can see, the front is formed by the solutions that are bigger on f2 but smaller on f1.

In figure 3, it is presented the Pareto front for the maximization of the two objective functions. Here the solutions on the front are those with the biggest value on function f1 and the biggest value on f2 too.

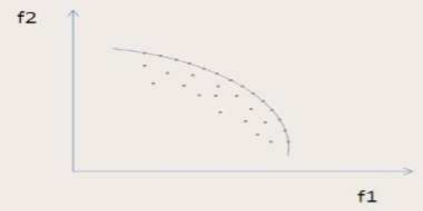


Fig. 3. Pareto front for the maximization of f1 and the maximization of f2

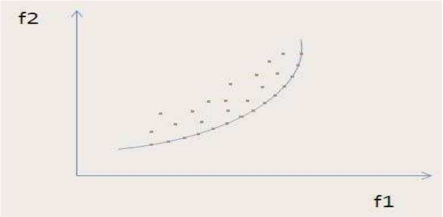


Fig. 4. Pareto front for the maximization of f1 and the minimization of f2

Finally, figure 4 shows the shape of the Pareto front when f1 is maximized while f2 is minimized. In this figure it can be seen that the Pareto front is formed by solutions that exhibit a high fitness on f1 but low fitness on f2.

Normally, it is impossible to find a mathematical expression that allows us to determine the whole set of points conforming the $\mathcal{PF}^*$. To determine this group, usually are calculated the $\vec{f}$ of an enough number of points in Ω (feasible region). If the number of points calculated is appropriate, then can be determined which solutions are not dominated ones and this way the Pareto front can be obtained. Not dominated solutions don't have any relationship to each other, on the fact they are members of the Pareto optimal. This set corresponds to the non dominated solutions that conform the Pareto front.

According with the definition of Pareto optimal, to get the solutions, it is necessary to make a commitment among the functions, in other words, improving an objective will be reflected as the deterioration of another. This is one of the main concepts in multi-objective optimization. The commitment is subjected to questions in some cases, maybe not in the totality of cases. But we could generate better results in terms of quality and smaller cost, only changing the formulation of the problem (Zeleny, 1997).

2.8 Strong and weak Pareto dominance

Besides the Pareto optimality concept, there are some other concepts very important in MOOP, two of them are called: weak Pareto dominance and strong Pareto dominance. A vector is a weak Pareto optimal if does not exist another vector in which all components in the objective functions space are better. Formally it can be defined as: A solution $\vec{x}^* \in \Omega$ is a weakly not dominated solution if does not exist another solution $\vec{x} \in \Omega \mid f_i(\vec{x}) < f_i(\vec{x}^*)$, for $i = 1,2,\dots,k$.

The concept of strong Pareto dominance could be summarized as follows: A solution $\vec{x}^* \in \Omega$ is a strongly not dominated solution if does not exist another solution $\vec{x} \in \Omega \mid f_i(\vec{x}) \leq f_i(\vec{x}^*)$, for $i = 1,2,\dots,k$ and also exists at least a value $j \mid f_j(\vec{x}) < f_j(\vec{x}^*)$

3. Multi-objective evolutionary algorithms

Although apparently the only source of motivation for using evolutionary algorithms to solve multi-objective problems arises from a single source (Goldberg 1989), this field has become very wide in recent years. As discussed in the introduction to this chapter, the parallel nature of evolutionary algorithms make them a tool with great potential when trying to find a group of solutions on an optimization problem.

This section will discuss the first multi-objective optimization algorithms (MOAs) used, passing from those that handle the problem as if it were a single objective problem, to those that make use of EDAs. EDAs are particularly important in this chapter, because towards the end of it, the problem of graph drawing is addresses by this type of metaheuristics.

The field of both mono-objective and multi-objective optimization has been benefited from a significant number of classical techniques, but quantity of new techniques have been recently included. A particularly successful approach is the application of evolutionary computation. Because this chapter deals with the solution of multi-objective problems with heuristic tools, we will start describing the general operation of an evolutionary algorithm.

An evolutionary algorithm begins with the creation (initialization) of a population of individuals (possible solutions to the problem) "*Pt*", usually created by a random procedure or knowledge-driven problem-information. Thereafter, the algorithm performs an iterative process that evaluates the quality of each individual in the population and starts a process of transformation of the current population by certain operators. The most common operators are selection, crossover, mutation and elitism. The iterative process stops when one or more predetermined criteria are met. Figure 5 shows the general procedure of an evolutionary algorithm. In this figure each apostrophe represents a new transformation of the current population, while "*t*" indicates the generation number.

An Evolutionary Optimization Procedure

```
t=0;
Initialization ($P_t$);
do
    Evaluation($P_t$);
    $P_t$' = Selection ($P_t$);
    $P_t$'' = Variation ($P_t$');
    $P_{t+1}$ = Elitism ($P_t$, $P_t$'');
while (Termination($P_t$,$P_{t+1}$));
```

Fig. 5. General Evolutionary Optimization Procedure (Deb, 2008)

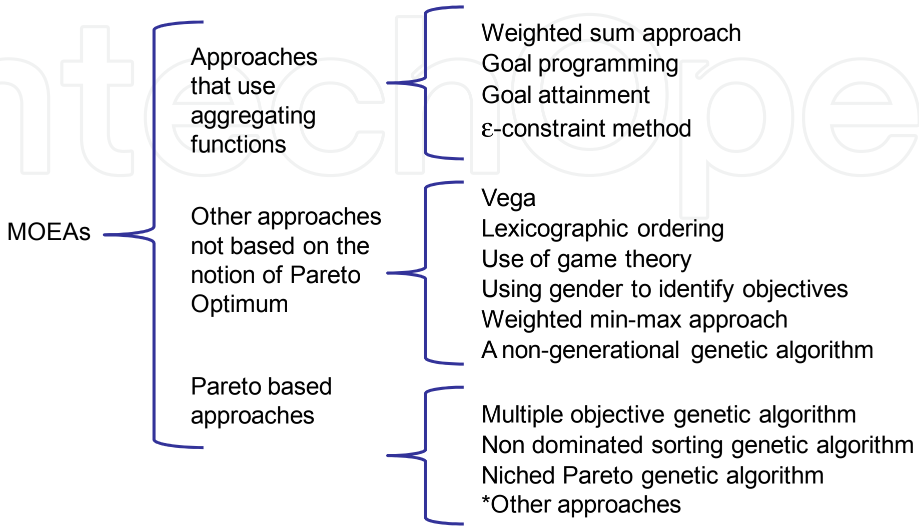

MOEAs
Approaches that use aggregating functions
Weighted sum approach
Goal programming
Goal attainment
ε-constraint method
Other approaches not based on the notion of Pareto Optimum
Vega
Lexicographic ordering
Use of game theory
Using gender to identify objectives
Weighted min-max approach
A non-generational genetic algorithm
Pareto based approaches
Multiple objective genetic algorithm
Non dominated sorting genetic algorithm
Niched Pareto genetic algorithm
*Other approaches

Fig. 6. Classification of Multi-Objective Evolutionary Algorithms

Even though the evolutionary multi-objective optimization field is very young (less than twenty years), it is already considered as a well-established research and application area; according to Deb (Deb, 2008) there are hundreds of doctoral theses on this topic, and are dozens of books devoted to it too.

Some of the reasons why evolutionary algorithms (EAs) have become so popular are:

1. EAs do not require any derivative information
2. EAs are relatively simple to implement
3. EAs are flexible and have a wide-spread of applicability (Deb, 2008)

Marler and Arora (Marler and Arora, 2004) propose a general classification of all multi-objective optimization methods according to the decision maker (DM) intervention. These researchers distinguished the next categories:

- Methods with a priori articulation of preferences
- Methods with a posteriori articulation
- Methods with no articulation of preferences.

The first category focuses on those methods where the user (DM) can specify certain preferences since the beginning of the process; which may be articulated in terms of goals, levels of importance of the objective functions, etc. The second category refers to the group of methods that begin the search for the Pareto set without additional information, but as the search process progresses, the method has to be assisted by the introduction of some preferences provided by the DM. Finally, when the DM is not able to define specifically what he prefers, it is necessary to employ methods that do not require any articulation of preferences. These methods are those that make up the third category of Marler and Arora. For more details see (Marler and Arora, 2004).

Speaking more specifically about multi-objective evolutionary algorithms (MOEAs), we can find another widely accepted classification. This classification groups them as follows:

- Those algorithms that do not incorporate the concept of Pareto optimality in their selection mechanism.
- Those algorithms that rely in the population according to whether an individual is dominated or not.

Considering this last classification and the one used by Coello (Coello, 1999), main multi-objective evolutionary algorithms can be grouped in the way shown in Figure 6.

In this chapter we will use mainly the latter classification, because our interest is in those techniques that come from the evolutionary computation. Since explaining all the algorithms of the previous classification would be very extensive, we will focus on discussing only the most used of them.

3.1 Approaches that use aggregative functions

The most commonly used methods for solving multi-objective problems, also called "*basic methods*" (Miettinen, 2008) are those who handle problems as if they were single-objective problems. These methods consist on the transformation of the problem so that they can be solved by optimizing a single objective function. The tendency to transform a multi-objective problem to the form of a single-objective one, responds to the fact that single-objective optimization techniques are better known than those that include optimization based on several functions. The intuitive nature of these techniques, besides the fact that GAs use scalar fitness, makes aggregative functions the first option for solving multi-objective problems. Aggregative functions are combinations either linear or nonlinear of all objectives into a single one. Although there are some drawbacks in using arithmetic combinations of objectives, these techniques have been used extensively since the late sixties, when Rosenberg published his work (Rosenberg, 1967). Even though Rosenberg did not use a multi-objective technique, his work showed that it was feasible to use evolutionary search techniques to handle multi-objective problems. The two techniques that best represent this kind of approaches are: Weighted Sum Method and ε-Constraint Method.

Readers interested on techniques in this group, can consult "*A comprehensive Survey of Evolutionary-Based Multi-objective Techniques*" (Coello, 1999).

3.1.1 Weighted sum method

The goal of this method is constituted by the sum of all objectives of the problem, using different coefficients for each one. The coefficients used represent the level of importance assigned to each of the objectives. So the optimization problem becomes a problem of scale optimization as follows:

$$\text{minimize } \sum_{i=1}^{k} w_i f_i(\vec{x}) \tag{14}$$

Where $w_i \geq 0$ is the weighting coefficient that represents the relative importance of the i-th objective. It is usually assumed that

$$\sum_{i=1}^{k} w_i = 1 \tag{15}$$

The normalization above takes place because the results obtained by this technique may have significant variations to small changes in the coefficients and avoids that different magnitudes confuse the method. Very often it is need to perform a set of experiments before determining the best combination of weights. When the decision maker has some a priori knowledge about the problem, it is feasible and beneficial to introduce this information in modelling. At the end of the process is the decision maker the one who should make the most appropriate solution according to his experience and intuition. There are several variations of this method, for example, adding constant multipliers to scale objectives in a better way. This was the first method used for the generation of non inferior solutions for multi-objective optimization (Coello 1998), perhaps because it was implied by Kuhn and Tucker in their seminar work on numerical optimization (Kuhn and Tucker, 1951). Computationally speaking, this method is efficient and it has proven to have the ability of generating non-dominated solutions which are often used as a starting point for other techniques; nevertheless, its main drawback is the enormous complexity to determine the appropriate weights when there is no information about the problem. In the case that there is no information about the problem, the literature suggests using simple linear combinations of the objectives to adjust the weights iteratively. In general this technique is not suitable in the presence of search spaces non-convex (Ritzel et al., 1994), because the alteration of the weights can produce jumps between several vertex, leaving undetected intermediate solutions.

3.1.2 ε-constraint method

The operating principle of this method is to optimize only one objective at a time, leaving the rest of them as constraints that must be limited by certain permitted levels ε_j. The objective that is optimized, is the one considered as the principal or most important f1. ε_j levels are then altered to generate the Pareto optimal entire set. This method can be formulated as follows:

$$\text{minimize } f_l(\vec{x}) \tag{16}$$

$$subject\ to\ f_j(\vec{x}) \leq \varepsilon_j\ for\ all\ j = 1, \ldots, k, j \neq l \tag{17}$$

where $l \in \{1,\ldots.,k\}$ and ε_j are upper bounds for the objectives $(j \neq l)$. The search stops when the decision maker finds a satisfactory solution. This method was introduced by Haimes et al in (Haimes et al., 1971). It is possible that this procedure should be repeated for different values of the index l. In order to obtain a set of appropriate values of ε_j is very common to use independent GAs or other techniques for optimizing each objective function. The main weakness of this method is related to its huge consumption of time, however, its relative ease, has made it very popular especially in the community of GAs.

3.2 Other approaches not based on the notion of Pareto optimum

Although techniques mentioned in the previous sub-section have proven to be useful for solving multi-objective optimization problems, we must not forget that they do it as if it were a problem with a single objective. The search for other alternatives resulted in the development of the techniques in the second category according to Figure 6. Techniques in this category introduced two very important elements: the use the populations and the use of special handling of objectives. To illustrate this group of techniques, the Vector Evaluated Genetic Algorithm (VEGA) and the lexicographic ordering are going to be discussed. VEGA is so important because it was the first GA used as a tool for solving MOOP. On the other hand, during the decade of the 80's and early 90's, the MOEAs were characterized by the use of aggregative techniques (already discussed), target vector optimization and lexicographic ordering; so, it would be illustrative to review this last one.

3.2.1 Vector Evaluated Genetic Algorithm (VEGA)

The first multi-objective genetic algorithm was implemented by Schaffer (Schaffer, 1984), and it was inspired on the "*simple GA*" (SGA). After making some modifications to the first implementation, Schaffer named it "*Vector Evaluated Genet Algorithm*" (Schaffer, 1985). Schaffer proposed the creation of one sub-population per each objective function of the problem on each generation of the algorithm. So, assuming a population size of *N* for a problem with *k* objective functions, *k* subsets (sub-populations) of size *N/k* should be generated; then the *k* sub-populations must be shuffled together to obtain the new population of size *N*. Finally, the GA will apply classical operators. Figure 7 shows the selection scheme of VEGA.

The main weakness of this algorithm comes from the fact that it promotes the conservation of solutions with very good performance in only one of the k objectives of the problem, by eliminating the solutions that have what Schaffer called "*middling*" performance (acceptable performance in all objective functions). The problem mentioned is known in genetics like "*speciation*", and it is obviously undesirable in solving multi-objective problems because it goes against the goal of finding compromise solutions.

In more general terms, the performance of this method is compared with the linear combination of objectives, where the weights depend on the distribution of the population in each generation as demonstrated by Richardson et al (Richardson et al., 1989). Therefore this technique has not the ability to produce Pareto optimal solutions in the presence of non-convex search spaces.

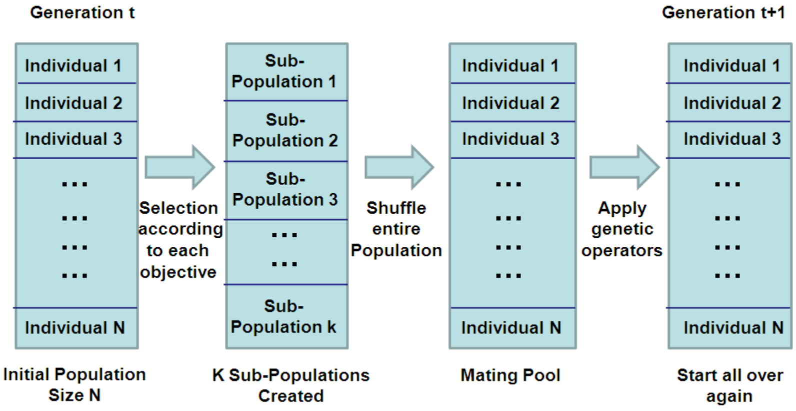


Fig. 7. Scheme of VEGA selection

3.2.2 Lexicographic ordering

This method, which is commonly grouped with the methods that articulate some preferences a priori according with the Marler and Arora's classification (Marler and Arora, 2004), or the named as a priori methods (Miettinen, 2008), begins with the arrangement of all objective functions according to their relative importance. Subsequently, the most important objective function is minimized subject to the original constraints. Then, we formulate a similar problem with the second most important objective function and an extra restriction. This procedure is repeated until the k objectives have been considered. The first problem to be solved, assuming that f_1 is the most important objective, has the following form:

$$\text{minimize } f_1(\vec{x}) \tag{18}$$

$$subject\ to\colon\ g_j(\vec{x}) \leq 0 \quad j = 1,2 \dots, m \tag{19}$$

By solving (5) and (6), we obtain $\overrightarrow{x_1}^*$ and $f_1^*=f(\overrightarrow{x_1}^*)$, and then, the next problem is formulated:

$$\text{minimize } f_2(\vec{x}) \tag{20}$$

$$subject\ to\colon\ g_j(\vec{x}) \leq 0 \quad j = 1,2 \dots, m \tag{21}$$

$$f_1(\vec{x}) = f_1^* \tag{22}$$

Once the problem in (7), (8) and (9) is solved, $\overrightarrow{x_2}^*$ and $f_2^*=f(\overrightarrow{x_2}^*)$ are obtained. This procedure is then repeated over and over, until all objective functions have been taken into account. The final solution obtained $\overrightarrow{x_k}^*$ is considered the best solution of the problem.

The greatest strength of this method lies in its simplicity, and its greatest weakness comes from the high level of dependence of their performance with the order of importance chosen for each objective function. Because this method takes into account one objective at a time, it tends to promote only certain goals, when there are others in the problem, making the process to converge to a particular area of the Pareto front.

3.3 Pareto based approaches

As the reader may have observed, all techniques discussed so far produce Pareto front members implicitly, because they do not use the Pareto-optimality concept as a search mechanism, nevertheless there are also a set of methods that employ the definition of Pareto-optimality to conduct the search for solutions. In 1989 Goldberg suggested the use of a fitness function based on the concept of Pareto-optimality to deal with the problem of speciation identified by Schaffler. Goldberg's proposal was to find the set of individuals that are Pareto non-dominated by the rest of the population and assign them the rank 1, then removing them from contention, and then find a new set of non-dominated individuals and rank them as 2, and so forth. This technique is named Pareto ranking.

The main weakness of this method is that there is not yet an efficient algorithm to check non-dominance in a set of feasible solutions (Coello, 1996). As the size of population and the number of objective functions grow up, efficiency of algorithms is worse; however, Pareto ranking is the most appropriate method to generate an entire Pareto front in a single run of the GA (Coello, 1999). Several algorithms that use Pareto based approaches have been developed; next subsections will discuss some of them.

3.3.1 Multiple Objective Genetic Algorithm (MOGA)

A scheme in which the rank of an individual depends on the number of individuals from a certain population, by which it is dominated, was proposed by Fonseca and Fleming (Fonseca and Fleming, 1993). For example, lets suppose generation *t*, all non-dominated individuals are assigned rank 1, while dominated ones are assigned a rank of $(1+p_i^{(t)})$ where $p_i^{(t)}$ is the number of solutions that dominates the solution x_i. The individual x_i in the generation t, can be assigned the next rank.

$$rank(x_i, t) = 1 + p_i{}^{(t)} \tag{23}$$

Fitness assignment is performed in the following way (Fonseca and Fleming, 1993).

1. Population is sort by the assigned rank
2. Fitness is assigned to individuals by interpolating from the best (rank 1) to the worst (rank n). Interpolation is usually linear but it can be non linear.
3. The fitness of individuals with the same rank is averaged, so all of them will be sampled at the same rate.

A potential weakness of this algorithm is the premature convergence produced by a large selection pressure because of blocked selected fitness (Goldberg and Deb, 1991). To avoid this, Fonseca and Fleming used niche-formation method to distribute the population over the Pareto-optimal region; however instead of performing sharing on the parameters values, they used sharing on the objective function values.

This algorithm has been widely accepted and used because of its efficiency and relatively easy implementation. As other Pareto ranking techniques, this algorithm is highly dependent of an appropriate selection of the sharing factor, but Fonseca and Fleming developed a methodology to compute this factor for their approach (Fonseca and Fleming, 1993).

3.3.2 Non-dominated Sorting Genetic Algorithm (NSGA)

The NSGA was proposed by Srinivas and Deb (Srinivas and Deb, 1993). This method is characterized in that the fitness assignment is performed by a rank of dominance. It does not work with a functional value, but with a dummy fitness.

In the first step of this method, the population is ranked based on non-domination. All non-dominated individuals are put into a category with a dummy fitness proportional to population size. Then, this group of classified individuals is ignored and another layer of non-dominated individuals is considered. This process continues until all individuals in the population have been classified. Because individuals of the first front have the highest value of fitness, they will be copied more times than the rest of the population. This method allows the search of non-dominated regions with quick convergence results. The efficiency of this method lies in the way a group of objectives is replaced by a dummy function using a non-dominated sorting procedure. According with Srinivas and Deb, with this approach maximization and minimization with any number of objectives can be handled (Srinivas and Deb, 1994). Among other researchers, Coello has reported that this approach is less efficient than the MOGA, and more sensitive to the value of the sharing factor.

3.3.3 Niched Pareto Genetic Algorithm (NPGA)

A tournament selection scheme based on Pareto dominance was proposed by Horn and Nafpliotis (Horn and Nafpliotis, 1993). The main idea of this approach is to use tournament selection based on Pareto dominance with respect to a subset of the population (typically around 10 individuals). In case of ties (when both competitors were either dominated or non-dominated), the decision is made by fitness sharing in both, fitness function space and in the decision variables space.

3.4 Other approaches

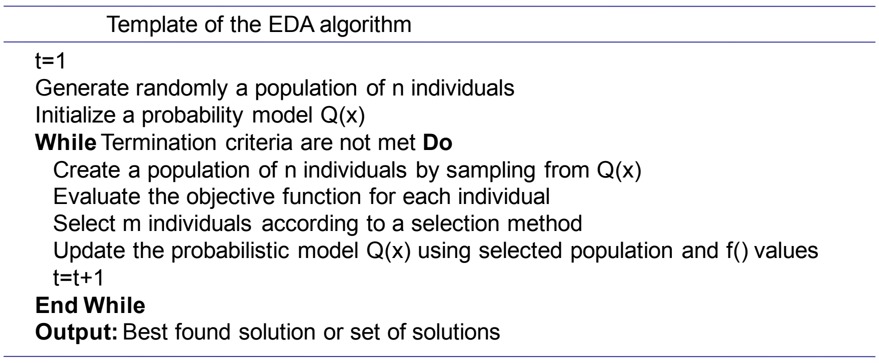
Evolutionary algorithms have proved to be very efficient in solving several multi-objective optimization problems, because they have good ability of global exploration and fast convergence speed, all due to the use of nature-inspired operators (crossover, mutation, selection). However, they also have been criticized for the little use made of the information about the problem, the high random component they possess and the large number of evaluations of the problem they use. Some of these problems are being addressed through proposals such as EDAs and Scatter Search, in which operators are deterministic or employ techniques that reduce the number of evaluations.

Another recent trend to address the weaknesses of evolutionary algorithms is combining them with classical optimization methods or other metaheuristics. This type of technique has been used successfully in single-objective optimization, leading to what is called "*memetic algorithms*" (Moscato, 1999).

In this section, the general idea behind the EDA is discuss, because it is the technique used in solving the problem of drawing graphs. Section 4.2 of this chapter describes the used algorithm called "*Hybrid multi-objective optimization estimation of distribution algorithm*". This algorithm is a hybridized EDA with Hill Climbing.

The main idea behind EDAs is to use the probability distribution of the population in the reproduction of the new offspring. EDAs are a natural outgrowth of GA in which statistical information of the population is used to build a probability distribution. Then, this distribution is used to generate new individuals by sampling. Because probability distribution replaces Darwinian operators, this kind of algorithm is classified as non-Darwinian evolutionary algorithm.

The general procedure of the EDA can be sketched as shown in figure 8.



```
Template of the EDA algorithm
t=1
Generate randomly a population of n individuals
Initialize a probability model Q(x)
While Termination criteria are not met Do
  Create a population of n individuals by sampling from Q(x)
  Evaluate the objective function for each individual
  Select m individuals according to a selection method
  Update the probabilistic model Q(x) using selected population and f() values
  t=t+1
End While
Output: Best found solution or set of solutions
```

Fig. 8. Estimation of the Distribution Algorithm (Talbi, 2009)

EDAs are classified according to the level of variable-interaction they use in their probabilistic model:

- Univariate: This class of EDAs suppose that there is not interaction among problem-variables.
- Bivariate: This class of EDAs suppose that there is interaction between two variables.
- Multivariate: In this class of EDAs, the probabilistic distribution models the interaction among more than two variables.

Although initially EDAs were intended for combinatorial optimization, now they have been extended to the continuous domain. Nowadays the application field of EDAs not only addresses mono-objective optimization issues, but it has been created a discipline related to their application on multi-objective problems. The group of EDAs applied to multi-objective optimization is called *"multi-objective optimization EDAs"* (MOEDAs) (Marti, 2008). Most of the actual MOEAs are modified single-objective EDAs whose fitness assignments are replaced by multi-objective assignments.

According to some researchers, there are several aspects that are crucial in the implementation of multi-objective solutions when MOEDAs are used; some of them are:

- Fitness assignment: Since several objectives have to be taken into account; this aspect is very important and more complex than in single-objective optimization.
- Diversity preservation: In order to reach a good coverage of the Pareto front, population diversity is critical.

- Elitism: Elitism is the mechanism used to preserve non dominated solution through successive generations of the algorithm.

With these aspects in mind, next section will discuss the implementation of the proposed solution to the graph drawing problem.

4. An application of a multi-objective optimization hybrid estimation of distribution algorithm for graph drawing problem

Graph drawing problems are a particular class of combinatorial optimization problems whose goal is to find plane layout of an input graph in such a way that certain objective functions are optimized. A large number of relevant problems in different domains can be formulated as graph layout problems. Among these problems are optimization of networks for parallel computer architectures, VLSI circuit design, information retrieval, numerical analysis, computational biology, graph theory, graphical model visualization, scheduling and archaeology. Most interesting graph drawing problems are NP-hard and their decisional versions are NP-complete (Garey and Johnson, 1983), but, for most of their applications, feasible solutions with an almost optimal cost are sufficient. As a consequence, approximation algorithms and effective heuristics are welcome in practice (Díaz et al., 2002).

Visualization of complex conceptual structures is a support tool used on several engineering and scientific applications. A graph is an abstract structure used to model information. Graphs are used to represent information that can be modeled as connections between variables, and so, to draw graphs to put information in an understandable way. The usefulness of graphs visualization systems depends on how easy is to catch its meaning, and how fast and clear is to interpret it. This characteristic can be expressed through of aesthetic criteria (Sugiyama, 2002) as the edges' crossing minimization, the reduction of drawing area and the minimization of aspect ratio, the minimization of the maximum length of an edge, among others.

In our approach the three first objectives are used and we can make a multi-objective optimization formulation for the graph drawing problem. On the one hand, to enhance the legibility of the graph drawing is very important to keep as low as possible the number of crosses, as well as to keep a good aspect ratio in the draw. Another point is to maintain symmetric the drawing region (same drawing height and width). It is very desirable too, to keep the drawing area small. This last requirement avoids the waste of screen space. These objectives are in conflict with each other. To reach the minimum crossing edges in the graph drawing is frequently needed a bigger area. At the same time, for minimizing the aspect ratio of the graph is needed to draw the nodes in a symmetrically delimited region. The reduction of the used area increases the number of crosses because as closer the edges are, there is less space to do the crossing edges minimization. Besides, area reduction of the sketching also affects the symmetrical delimitation of the region used by the graph. The aspect ratio minimization is affected by the crossing edges minimization due that just to get a node outside the defined area contributes to the imbalance of the symmetry reached until that moment. So, the reduction of the drawing area affects directly the aspect ratio of the graph because generally this kind of reduction is not symmetric. A first approach of the multi-objective optimization problem for these three objectives for graph drawing could be found in (Enriquez et al., 2011).

4.1 Formulation of the multi-objective optimization for graph drawing problem

At the beginning, we have a graph given by its edges, that is, a pair of vertices. To each vertex is assigned a pair of coordinates. All coordinates of the vertices of the graph are randomly generated in the cartesian plane. If any two vertices have the same coordinates then new coordinates are randomly generated for one of them. The candidate solution is represented as a vector of pairs of coordinates. The input information, i.e., the list of edges of the graph is used by the algorithm to draw the edges in the best manner in order to fulfill a tradeoff between all considered objective functions.

In this chapter the following in conflict objectives have been considered:

- Minimization of the number of crossing edges in the graph: The total number of crossing edges of the graph has to be minimized (f1).
- Minimization of the graph area: to minimize the total space used by the graph (f2).
- Minimization of the graph aspect ratio: the graph has to be visualized in an approximate square area (f3).

The vector of the objective functions is denoted by F=(f1,f2,f3). The first function f1 is calculated as follows:

To draw a line between two vertices, $v_1(x_1, y_1)$ and $v_2(x_2, y_2)$ we use the following equation:

$$y - y_1 = \frac{y_2 - y_1}{x_2 - x_1}(x - x_1) \tag{24}$$

and solve the equation system for knowing if the two lines corresponding to edges have an intersection point. The function f1 sums the number of intersection points between edges of this drawing.

$$a_1 x + b_1 y = c_1 \tag{25}$$

$$a_2 x + b_2 y = c_2 \tag{26}$$

The second function f2 is defined as the area of the rectangle containing the graph drawing. The following formula is used:

$$S = (x_{max} - x_{min}) \cdot (y_{max} - y_{min}) \tag{27}$$

where $x_{min}$and x_{max} are the least and greatest values on the abscise axis, and y_{min} and y_{max} are the least and greatest values on the vertical axis. S is the value of the function f2.

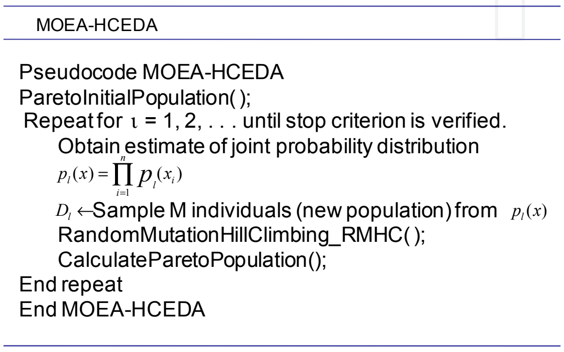
Finally, the f3 function is obtained as a ratio of $(x_{max} - x_{min})$ on $(y_{max} - y_{min})$ or vice versa, depending on which was the least. f3 is the value of this ratio, and it is knowing as aspect ratio.

We use the Pareto front approach for the multi-objective optimization problem (Coello and López, 2009), (Deb, 2001) and we give the final Pareto front and also give as more promissory solution, that solution closest to the origin, because it resumes all objective tradeoffs. The distance to origin is calculated evaluating the Euclidean distance using the standardized values of the objectives of the problem.

4.2 Hybrid multi-objective optimization estimation of distribution algorithm

This section presents a description of the components of the proposed algorithm, which is built of three main components. One of them the Univariate Marginal Distribution Algorithm (UMDA) (Mühlenbein et al., 1998) adapted for multi-objective optimization problems is used for exploration of the search space, and the second component the Random Mutation Hill Climbing (RMHC) algorithm is used for the exploitation. Finally, a component for calculating the Pareto front is used.

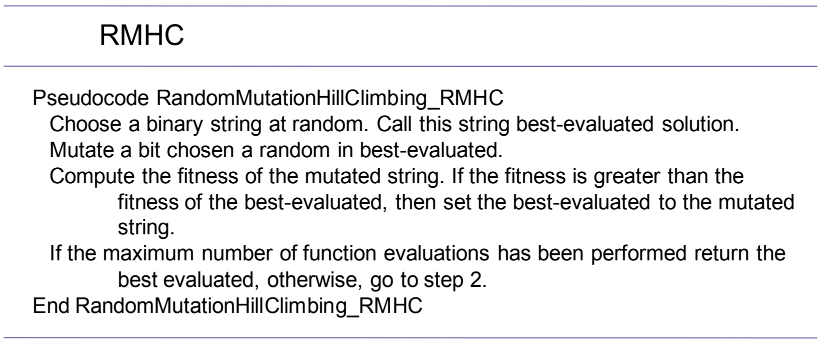
The pseudocode of the multi-objective optimization evolutionary hill climbing estimation of distribution algorithm (MOEA-HCEDA) is as shows in figure 9.



```
MOEA-HCEDA

Pseudocode MOEA-HCEDA
ParetoInitialPopulation( );
 Repeat for ι = 1, 2, . . . until stop criterion is verified.
    Obtain estimate of joint probability distribution
    $p_l(x) = \prod_{i=1}^{n} p_l(x_i)$
    $D_l$ ←Sample M individuals (new population) from $p_l(x)$
    RandomMutationHillClimbing_RMHC( );
    CalculateParetoPopulation();
End repeat
End MOEA-HCEDA
```

Fig. 9. Pseudocode of MOEA-HCEDA



```
RMHC

Pseudocode RandomMutationHillClimbing_RMHC
  Choose a binary string at random. Call this string best-evaluated solution.
  Mutate a bit chosen a random in best-evaluated.
  Compute the fitness of the mutated string. If the fitness is greater than the
          fitness of the best-evaluated, then set the best-evaluated to the mutated
          string.
  If the maximum number of function evaluations has been performed return the
          best evaluated, otherwise, go to step 2.
End RandomMutationHillClimbing_RMHC
```

Fig. 10. Pseudocode of RMHC

ParetoInitialPopulation( ): In the first step a random population with size 2*size of population is generated. After that, the first Pareto front is obtained using the dominance solution. The first approximated Pareto front is saved in D_0.

RandomMutationHillClimbing( ): In Random Mutation Hill Climbing (Mitchell et al., 1994), a string is chosen randomly and its fitness is evaluated. The string solution is mutated randomly choosing a single locus, and the new solution is evaluated. If mutation leads to an equal or higher fitness, the new string solution replaces the old. This procedure is iterated until the optimum has been found or a maximum number of function evaluations have been performed. The algorithm RMHC works as figure 10 shows.

CalculateParetoPopulation( ): In the first step, the last approximated Pareto front saved in D_{l-1} is joined with the recently generated population and saved in D_l. In the second step the new approximated Pareto front is calculated from $D_{l-1} \cup D_l$. The new approximated Pareto front is saved in D_{l-1}.

UMDA is a particular case of EDAs, introduced by Mühlenbein (Mühlenbein et al., 1998), where the variables are totally independent. The n-dimensional joint probability is a product of n univariate probability distributions (Larrañaga & Lozano, 2002).

Example:

$$p_l(x) = \prod_{i=1}^{n} p_l(x_i) \tag{28}$$

The joint probability distribution of each generation is estimated using the individuals $p_l(x)$ selected. The joint probability distribution factorizes as the product of independent univariate distributions.

4.3 Dominance index to evaluate solutions in Pareto front

This section describes how to define a measure of quality (dominance index) for each solution stored in the Pareto front. The objective of this dominance index is to order the elements of the Pareto front.

Definition. Dominance index of a solution $\vec{x}$: Let $\mathcal{P}_r, \mathcal{P}_s$ be two approximate Pareto fronts and let $r(\mathcal{P}_r)$ be the number of elements of $\mathcal{P}_r$ and $s(\mathcal{P}_s)$ the number of elements of $\mathcal{P}_s$. The dominance index of a solution $\vec{x}$ is defined as the number of times $n(\vec{x})$ that a solution $\vec{x}_r \in \mathcal{P}_r$ dominates solutions $\vec{x}_s \in \mathcal{P}_s$, divided by $s(\mathcal{P}_s)$.

4.4 Quality index to evaluate Pareto front performance

Based on the definition of dominance index of a solution $\vec{x}$, the quality index of Pareto front is constructed. Given two Pareto fronts, a relative evaluation of the first front $\mathcal{P}_r$ with respect to the second $\mathcal{P}_s$ can be given as follows:

Let $\vec{x}_i^{(r)}$ be one solution of the first Pareto front and let $n_i = n(\vec{x}_i^{(r)})$ be the number of times $\vec{x}_i^{(r)}$ dominates elements of the second Pareto front $\mathcal{P}_s$. To normalize this quantity in the dominance index definition, it is divided by the number of solutions of the second front $s(\mathcal{P}_s)$. The quantity obtained is the quality index to evaluate the solution $\vec{x}_i^{(r)}$.

Definition. Quality Index of the first Pareto front with respect to the second: Let now $\sum n_i$ be the sum of the number of times all the solutions of the first Pareto front dominate the solutions of the second front. To normalize this quantity, it is divided by the number of solutions in $\mathcal{P}_r$ front. This last quantity can be considered a relative quality index of the first Pareto front with respect to the second.

4.5 Experimental design

In a previous paper a factorial experiment was performed (Enriquez et al., 2011) where the best combination of factors found was: number of generations equal to 500 and population size equal to 150. These parameters were the ones that reached the best results of the algorithm. Seven graphs were selected from the papers (Rossete, 2000),(Branke, et al., 1997),(Eleoranta and Mäkinen, 2001), (Hobbs and Rodgers, 1998), (Rossete and Ochoa, 1998) to use them as benchmarks, but only the results of the composite graph (Enriquez et al., 2010) is commented in this chapter because this graph is the biggest one. It is a no planar graph with a total of 40 vertices and 69 edges. A total of ten runs for the combination of factors (500,150) were executed, each run has an output that is an approximation to the Pareto Front. The evaluation of the convergence to the Pareto front was performed with the quality index.

4.6 Results and discussion

The results of this experiment appear on table 1, figures 11, 12, 13, 14, 15, 16, and 17. Table 1 shows the best graphs obtained for ten repetitions of MOEA-HCEDA algorithm. For each of the best solution, the table shows run, graph number, total number of edges intersected, area size and aspect ratio. A distance to origin is used to evaluate the best solution obtained on each repetition. This distance is calculated evaluating the Euclidean distance using the standardized values of the three objectives of the problem. The optimal Pareto value is obtained in the graph 267 of the 5th repetition. The results show the average number crossing is 16.1, average area is 106318.6, and average aspect ratio is 1.0632.

		Objective Functions			
RUN	**GRAPH NUMBER**	**NCROSS**	**AREA**	**ASPECT RATIO**	**DISTANCE TO ORIGIN**
1	55	17	106446	1.079617834	0.679992707
2	115	18	89951	1.04778157	0.489841704
3	130	22	111132	1.058641975	0.702735094
4	230	10	128520	1.111764706	0.465940815
5	267	9	91506	1.003311258	0.343328661
6	303	12	155298	1.185082873	0.815347591
7	317	18	79520	1.014285714	0.368875792
8	420	16	93852	1.063973064	0.458813972
9	500	22	101661	1.064724919	0.559167084
10	520	17	105300	1.00308642	0.44348268
Total Average:		**16.1**	**106318.6**	**1.063227033**	

		The best solution obtained			
RUN	**GRAPH NUMBER**	**NCROSS**	**AREA**	**ASPECT RATIO**	**DISTANCE TO ORIGIN**
5	267	9	91506	1.0033113	0.343328661

Table 1. Best solution on each run

Figure 11 shows the average for ten runs of the Pareto front quality index printed on each generation of the algorithm, a convergent curve is showed. The results of the experiments showed that the algorithm converges to an optimal Pareto front.

Figures 12, 13, 14, 15, 16, and 17 show the evolution of graphs corresponding to run 5. Figure 12 shows the graph 16 of the generation 1. This graph has 412 edges crossing, 285270 total area and 1.01698 aspect ratio. Figure 13 shows the graph 2555 in the generation 100. This graph is better than the graph 16 because the edges crossing decrease to 29, total area decreases to 116620 and aspect ratio decreases to 1.0088. Figure 14 shows the graph 5822 of the generation 200. This graph is better in two objectives compared to 16th and 2555th graphs because the edges crossing decrease to 24, total area decreases to 110500 but the aspect ratio increases to 1.0461. Figure 15 shows the graph 10028 of the generation 300. This graph is better in two objectives than the other three graphs because the edges crossing decrease to 14, total area decreases to 109525 and the aspect ratio newly decreases to 1.0369. Figure 16 shows the graph 13924 of the generation 400. This graph is better in two objectives than the other four graphs because the total area decreases to 102700 and the aspect ratio decreases to 1.0284, the edges crossing is manteined in 14 crosses. Figure 17 shows the graph 17470 of the generation 500. This graph is the best in all objectives because the edges crossing decreases to 9, total area decrease to 91506 and aspect ratio decreases to 1.0033.

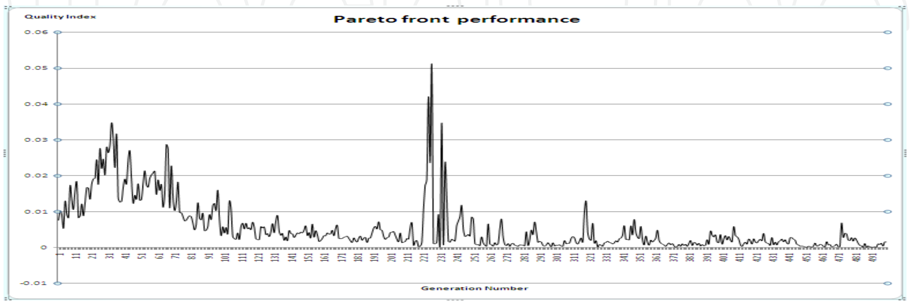
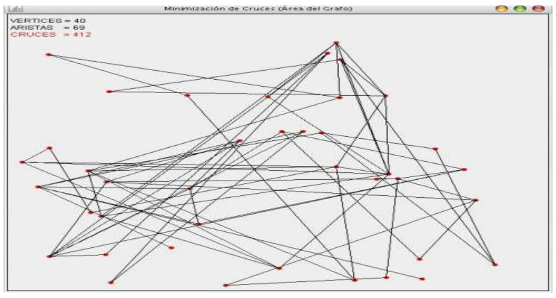
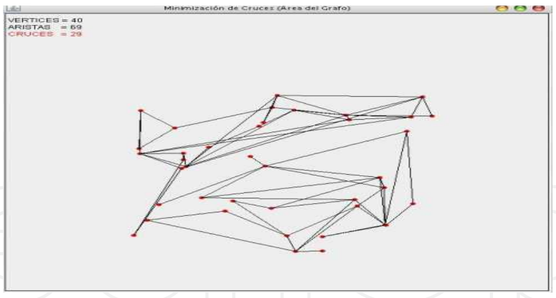


Fig. 11. Quality index for Pareto front comparison.



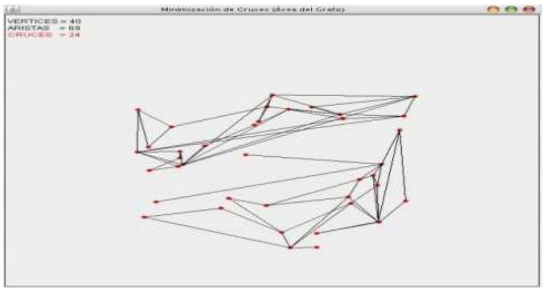
VERTICES = 40
ARISTAS = 69
CRUCES = 412

Fig. 12. Generation 1, graph 16.



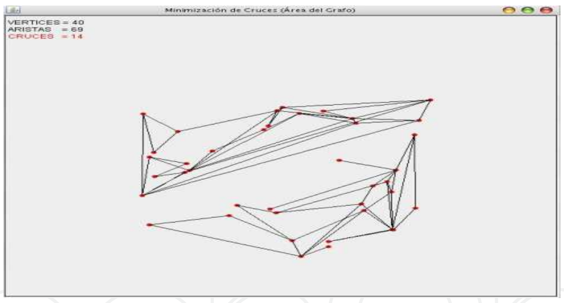
VERTICES = 40
ARISTAS = 69
CRUCES = 29

Fig. 13. Generation 100, graph 2555



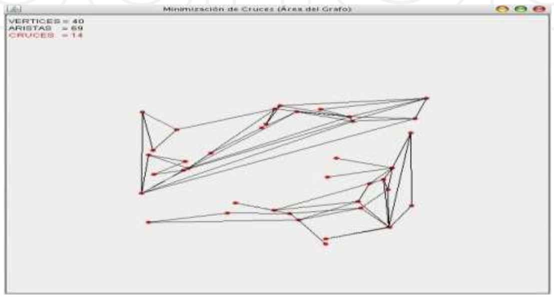
VERTICES = 40
ARISTAS = 69
CRUCES = 24

Fig. 14. Generation 200, graph 5822.



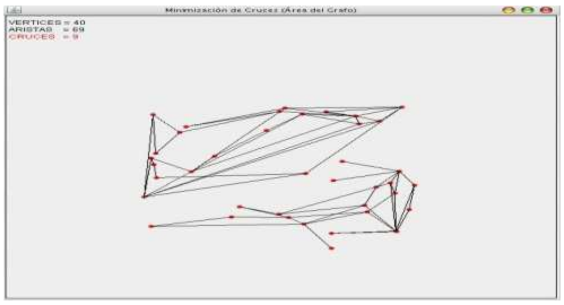
VERTICES = 40
ARISTAS = 69
CRUCES = 14

Fig. 15. Generation 300, graph 10028.



VERTICES = 40
ARISTAS = 69
CRUCES = 14

Fig. 16. Generation 400, graph 13924.



VERTICES = 40
ARISTAS = 69
CRUCES = 9

Fig. 17. Generation 500, graph 17470

4.7 Conclusions and future work

The main contributions of this application is the test of the hybrid MOEA-HCEDA algorithm and the quality index based on the Pareto front used in the graph drawing problem. The Pareto front quality index obtained on each generation of the algorithm showed a convergent curve. The results of the experiments showed that the algorithm converges. A graphical user interface was constructed providing users with a tool for a friendly and easy to use graphs display. The automatic drawing of optimized graphs makes it easier for the user to compare results appearing in separate windows, giving the user the opportunity to choose the graph design which best fits their needs.

To continue this research, the hybridization MOEA-HCEDA with others algorithms, for example using other types of EDAs is a next objective. The testing of the algorithms using others more complex benchmarks and, the comparison of the results between different variants is a very challenging and interesting task for future work. The graphical presentation can be friendlier and dispose other facilities as, for example, the printing of the results.

5. Future directions for research

Although there are many versions of evolutionary algorithms that are tailored to multi-objective optimization, theoretical results are apparently not yet available. Rudolph (1999) has shown that results known from the theory of evolutionary algorithms in case of single objective optimization do not carry over to the multi-objective case.

Assuming that the evolutionary algorithms are Markov processes, and that the fitness functions are partially ordered, Rudolph presented some theoretical results about the convergence of multi objective algorithms. In particular some properties of the operators have to be checked to establish the algorithm convergence. This theoretical analysis shows that a special version of an evolutionary algorithm converges with probability 1 to the Pareto set for the test problem under consideration, but this tools are not used frequently.

Although, there exist a number of multi-objective GA implementations and there exist a number of GA applications to multi-objective optimization problems, there not exists systematic study to speculate what problem features may cause a multi-objective GA to face difficulties. The systematic testing in a controlled manner on various aspects of problem difficulties is not so deeply addressed. Specifically, multi-modal multi-objective problems, deceptive multi-objective problems, multi-objective problems having convex, non-convex, and discrete Pareto-optima fronts, and non-uniformly represented Pareto-optimal fronts are not presented and systematically analyzed.

Although some studies have compared different GA implementations (Zitzler and Thiele, 1998), they all have presented a specific problem without an analysis about the complexity of the test problems. The test functions suggested until now in the literature provide various degrees of complexity but are not enough. The construction of test problems has been done without enough knowledge of how multi-objective GAs work. Thus, it will be worthwhile to investigate how existing multi-objective GA implementations work in the context of different test problems. It is intuitive that as the number of objectives increase, the Pareto-optimal region is represented by multi-dimensional surfaces. With more objectives, multi-objective GAs must have to maintain more diverse solutions in the non-dominated front in each iteration. Whether GAs are able to find and maintain diverse solutions, as demanded by the search space of the problem with many objectives would be a matter of interesting study. Whether population size alone can solve this scalability issue or a major structural change (implementing a better niching method) is imminent would be the outcome of such a study. Constraints can introduce additional complexity in the search space by inducing infeasible regions in the search space, thereby obstructing the progress of an algorithm towards the global Pareto-optimal front. Thus, creation of constrained test problems is an interesting area which should get emphasis in the near future. With the development of such complex test problems, there is also a need to develop efficient constraint handling techniques that would be able to help GAs to overcome hurdles caused by constraints. Some such methods are in progress in the context of single-objective GAs and with proper implementations they should also work in multi-objective GAs. Most multi-objective GAs that exist to date, work with the non-domination principle. It is a question if all solutions in a non-dominated set need not be members of the true Pareto optimal front, although some of them could be. This means that all non-dominated solutions found by a multi-objective optimization algorithm may not necessarily be Pareto-optimal solutions. Thus, while working with such algorithms, it is wise to check the Pareto-optimality of each of such solutions (by perturbing the solution locally or by using weighted-sum single-objective methods originating from these solutions). In this regard, it would be interesting to introduce special features (such as elitism, mutation, or other diversity-preserving operators), the presence of which may help us to prove convergence of a GA population to the global Pareto-optimal front. Some such proofs exist for single-objective GAs (Davis and Principe, 1991; Rudolph, 1994) and a similar proof may also be attempted for multi-objective GAs. Elitism is a useful and popular mechanism used in single-objective GAs. Elitism ensures that the best solutions in each generation will not be lost. They are directly carried over from one generation to the next and what is important is that these good solutions get a chance to participate in recombination with other solutions in the hope of creating better solutions. In the context of single-objective optimization, there is only one best solution in a population. But in multi-objective optimization, all non-dominated solutions of the first level are the best solutions in the population. There is no way to distinguish one solution from the other in the non-dominated set. Then if we like to introduce elitism in multi-objective GAs, should we carry over all solutions in the first non-dominated set to the next generation! This may mean copying many good solutions from one generation to the next, a process which may lead to premature convergence to non-Pareto-optimal solutions. How elitism should be defined in this context is an interesting research topic. In this context, an issue related to comparison of two populations also raises some interesting questions.

There are two goals in a multi-objective optimization—convergence to the true Pareto-optimal front and maintenance of diversity among Pareto-optimal solutions. A multi-objective GA may have found a population which has many Pareto-optimal solutions, but with less diversity among them. How would such a population be compared with respect to another which has a fewer number of Pareto-optimal solutions but with wide diversity? The practitioners of multi-objective GAs must have to settle for an answer for these questions before they would be able to compare different GA implementations or before they would be able to mimic operators in other single-objective GAs, such as CHC (Eshelman, 1990) or steady-state GAs (Syswerda, 1989). As it is often suggested and used in single-objective GAs, a hybrid strategy of either implementing problem-specific knowledge in GA operators or using a two-stage optimization process of first finding good solutions with GAs and then improving these good solutions with a domain-specific algorithm would make multi-objective optimization much faster than GAs alone.

Test functions test an algorithm's capability to overcome a specific aspect that a real-world problem may have. In this respect, an algorithm which can overcome more aspects of problem difficulty is naturally a better algorithm. This is precisely the reason why so much effort is spent on doing research in test function development. As it is important to develop better algorithms by applying them on test problems with known complexity, it is also equally important that the algorithms are tested in real-world problems with unknown complexity. Fortunately, most interesting engineering design problems are naturally posed as finding trade-offs among a number of objectives. Among them, cost and reliability are two objectives which are often the priorities of designers. This is because, often in a design, a solution which is less costly is likely to be less reliable and vice versa. In handling such real-world applications using single-objective GAs, often, an artificial scenario is created. Only one objective is retained and all other objectives are used as constraints. For example, if cost is retained as an objective, then an extra constraint restricting the reliability to be greater than 0.9 (or some other value) is used. With the availability of efficient multi-objective GAs, there is no need to have such artificial constraints (which are, in some sense, user-dependent). Moreover, a single run of a multi-objective GA may provide a number of Pareto-optimal solutions, each of which is optimal in one objective with a constrained upper limit on other objectives (such as optimal in cost for a particular upper bound on reliability). Thus, the advantages of using a multi-objective GA in real-world problems are many and there is the need for some interesting application case studies which would clearly show the advantages and flexibilities in using a multi-objective GA, as opposed to a single-objective GA.

We believe that more such mentioned studies are needed to understand better the working principles of a multi-objective GA. An obvious outcome of such studies would be the development of new and improved multi-objective GAs.

6. References

Bäck, T. (1996). Evolutionary Algorithms in Theory and Practice. Oxford University Press, ISBN 0-19-509971-0 New York, New York.

Branke, J., Bucher, F., and Schmeck, H. (1997). Using Genetic Algorithms for Drawing Undirected Graphs. In Allen, J. (Ed.), *Proceedings of 3rd Nordic Workshop on Genetics Algorithms and their Applications*, pp. 193-205 (1997).

Coello, C. (1996).*An Empirical Study of Evolutionary Techniques for Multiobjective Optimization. Engineering Design*. Doctoral thesis. Department of Computer Science, Tulane University, New Orleans, Louisiana, USA, April 1996.

Coello, C. (1999). A Comprehensive Survey of Evolutionary-Based Multiobjective Optimization Techniques. In Knowledge and Information Systems. Vol. 1, No.3 pp. 269-308., Aug.1999. Eberhart.

Coello, C., Van Veldhuizen, D., and Lamont, G. (2002). Evolutionary Algorithms for Solving Multi-Objective Problems. Kluwer Academic Publishers, ISBN 0-3064-6762-3 New York, New York.

Coello, C. (2011). Evolutionary Multiobjective Optimization, In: *Data Mining and Knowledge Discovery,* W. Pedrycz, (Ed.), pp. 444-447, John Wiley & Sons, ISSN 1942-4795, N.J, USA.

Coello, C. and López, A. (2009). Multi-Objective Evolutionary Algorithms: A Review of the State-of-the-Art and some of their Applications in Chemical Engineering. In *Multi-Objective Optimization Techniques and Applications in Chemical Engineering*. Rangaiah G. (ed), pp. 61-90, World Scientific, ISBN: 9812836519. Singapore.

Collette, Y. and Siarry, P. (2002). Multiobjective optimization: principles and case studies. Groupe Eyrolles , ISBN 978-3-540-40182-7, France.

Davis, T. and Principe, J. (1991). A simulated annealing like convergence theory for the simple genetic algorithm. Proceeding of the Fourth International Conference on Genetic Algorithms. ISBN 1-55860-208-9. pp. 174-181. San Diego, CA, USA, July 1991.

Deb, K. (2001). *Multi-Objective Optimization Using Evolutionay Algorithms*, John Wiley & Sons, Ltd, ISBN 0-471-87339-X. New York, N.Y, 2001.

Deb, K. (2008). Introduction to Evolutionary Multiobjective Optimization. In *Multiobjective Optimization. Interactive and Evolutionary Approaches*. Branke, J., Deb, K., Miettinen, K. and Slowinski, R.(Ed). Springer-Verlag Berlin Heidelberg, Germany.

Díaz, J., Petit, J., and Serna, M.(2002) A Survey of Graph Layout Problems. *ACM Computing Surveys*. Vol. 34, No. 3, (September 2002), pp. 313–356. New York, N.Y, USA.

Eloranta, T. and Mäkinen E. (2001). *TimGA* - A Genetic Algorithm for Drawing Undirected Graphs. *Divulgaciones Matemáticas,* Vol. 9 No.2, (October, 2001), pp. 155-171, ISSN 1315-2068.

Enríquez, S., Ponce de León, E., Díaz, E., Padilla, A. (2010).A Hybrid Evolutionary Algorithm for the Edge Crossing Minimization Problem in Graph Drawing. In *Advances in computer Science and Engineering*, Vol. 45, pp. 27-40. ISSN 1870-4069

Enríquez, S., Ponce de León, E., Díaz, E., Padilla, A, Torres, D., Torres, A., Ochoa A. (2011). An Evolutionary Algorithm for Graph Drawing with a Multiobjective Approach. In *Logistics Management and Optimization through Hybrid Artificial Intelligent Systems*, Carlos A. Ochoa, Carmen Chira, Arturo Hernández y Miguel Basurto, IGI Global, ISBN 13 :9781466602977

Eshelman, L. (1991). The CHC Adaptive Search Algorithm: How to Have Safe Search When Engaging. In *Nontraditional Genetic Recombination*. Rawlins G. (ed.), pp. 265-283. Proceedings of the First Workshop on Foundations of Genetic Algorithms. ISBN 1-55860-170-8, San Mateo, CA. USA.

Fonseca, C. and Fleming P. (1993). Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion and Generalization. In *Proceeding of the Fifth International Conference on Genetic Algorithms.* Forrest, S. (Ed). Morgan Kauffman Publishers. pp. 416-423, San Mateo, California, 1993. Urbana-Champaign.

Garey, M. and Johnson, D. (1983). Crossing number is NP-complete. In SIAM Journal on Algebraic and Discrete Methods, Vol. 4 (March, 1982), pp. 312–316, ISSN 0895-4798.

Goldberg, D. (1989). Genetic Algorithms in Search Optimization & Machine Learning. Addison-Wesley.

Goldberg, D. and Den, K. (1991). A comparison of selection schemes used in genetic algorithms. In *Foundations of Genetic Algorithms* Rawlins, G. (Ed). Morgan Kaufmann. pp. 69-93., San Mateo, California, 1991.

Haimes, Y., Lasdon, L., Wismer, D. (1971). On a Bicriterion Formulation of the Problems of Integrated System Identification and System Optimization . IEEE. *Transaction on systems*, Man and Cibernetics Vol. 1. pp. 296-297.

Horn, J. and Nafpliotis, N. (1993). *Multiobjective Optimization using the Niched Pareto Genetic Algorithm*. Technical Report IlliGAl Report 93005, University of Illinois at Urbana-Champaign, Urbana, Illinois, USA, 1993.

Hobbs, M. and Rodgers, P. (1998). Representing Space: A Hybrid Genetic Algorithm for Aesthetic Graph Layout. In *FEA'98, Proceedings of 4th Joint Conference on Information Sciences, JCIS'98,* The Fourth Joint Conference on Information Sciences, Vol. 2, pp. 415-418.

Kuhn, H. and Tucker, A. (1951). Nonlinear programming. In *Proceedings of the Second Berkely Symposium on Mathematical Statistics and Probability.* Neyman, J. (ed). pp. 481-492. University of California Press. Berkely, California.

Larrañaga, P. and Lozano, J.(2002). *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation*. Kluwer Academic Publishers, ISBN 0-7923-7466-5. Norwell, Massachusetts, USA.

Marler, R. and Arora, J. (2004). Survey of Multi-Objective Optimization Methods for Engineering. In *Structural and Multidisciplinary Optimization*, Vol. 26, No. 6, pp. 369-395.

Marti, L., Garcia, J., Berlanga, A. and Molina, J. (2008). Model-building algorithms for multiobjective EDAs: Directions for improvement, *Evolutionary Computation, 2008. CEC 2008. (IEEE World Congress on Computational Intelligence).*ISBN 978-1-4244-1823-7, June, 2008.

Miettinen, K. (2008). Introduction to Noninteractive Approaches. In *Multiobjective Optimization. Interactive and Evolutionary Approaches*. Branke, J., Deb, K., Miettinen, K. and Slowinski R. (Eds). Springer-Verlag Berlin Heidelberg. Germany.

Mitchell, M., Holland, J. and Forrest, S. (1994). "When Will a Genetic Algorithm Outperform Hill Climbing?”, *Advances in Neural Information Processing Systems*, Vol. 6, pp. 51-58.

Moscato, P. (1999). Memetic algorithms: a short introduction, In *New Ideas in Optimization,* Corne, D., Dorigo, M. and Glover, F., pp. 219-234, McGraw-Hill, ISBN 0-07-709506-5, UK Maidenhead, UK, England.

Mühlenbein, H., Mahnig, T. and Ochoa, A.(1998). Schemata Distributions and Graphical Models. *In Evolutionary Optimization, Journal of Heuristic*, Vol. 5, No. 2, pp. 215-247.

Ramírez, M. (2007). *Técnicas Evolutivas Multiobjetivo Aplicadas en el Diseño de Rutas en Vehículos Espaciales*. Doctoral thesis, México, DF.

Richardson, J., Palmer, M., Liepins, G. and Hilliard, M. (1989). Some guidelines for genetic algorithms with penalti functions. In *Proceedings of the Third International Conference on Genetic Algorithms*. Schaffer J. (ed), Morgan Kaufmann Publishers. pp. 191-197, George Mason University.

Ritzel, B., Eheart, W. and Ranjithan S. (1994). Using genetic algorithms to solve a multiobjective groundwater pollution containment problem. *Water Resources Research*, doi:10.1029/93WR03511. Vol. 30 No. 5. pp. 1589-1606. May 1994.

Rosenberg, R. (1967). *Simulation of genetic populations with biochemical properties*. PhD thesis, University of Michigan, Ann Harbor, Michigan, 1967.

Rossete, A. and Ochoa, A. (1998). Genetic Graph Drawing. *In Proceedings of 13th International Conference of Applications of Artificial Intelligence in Engineering, AIENG'98*, Adey, R. A., Rzevski, G., Nolan, P. (Ed.) Galway, Computational Mechanics Publications, pp. 37-40.

Rossete, A. (2000). Un Enfoque General y Flexible para el Trazado de Grafos, Doctoral Thesis, Faculty of Industrial Enginering, CEIS, La Habana, Cuba.

Rudolph, G. (1994). Convergence Analysis of Canonical Genetic Algorithms. *In Transactions on Neural Networks IEEE*, Vol. 5, No. 1, (January, 1994). pp. 96-101, ISSN: 1045-9227.

Rudolph, G. (1999). *On a multi-objective evolutionary algorithm and its convergence to the Pareto set.* Technical Report No. CI-17/98. Dortmund, Germany: Department of Computer Science/XI, University of Dortmund.

Schaffer, J. (1984). *Some Experiments in Machine Learning Using Vector Evaluated Genetic Algorithms*. Doctoral Thesis, Nashville, TN; Vanderbilt University.

Schaffer, J. (1985). Multiple Objective Optimization with Vector Evaluated Genetic Algorithms. In *Proceedings of the First International Conference on Genetic Algorithms and their Applications*, pp. 93-100. July, 1985, Pittsburgh, Pa.

Srinivas, N. and Deb, K. (1993). *Multiobjective Optimization using nondominated sorting*. Technical report. Genetic algorithms, Department of Mechanical Engineering, Indian Istitute of Technology, Kanput, India, 1993.

Srinivas, N. and Deb, K. (1994). Multiobjective Optimization using Nondominated Sorting. In *Genetic Algorithms. Evolutionary Computation*, Vol. 2, No. 3, pp 221-248.

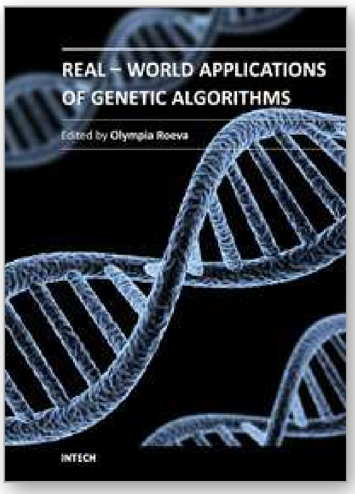
Sugiyama, K. (2002). *Graph Drawing and Applications for Software and Knowledge Engineers*, Japan Advanced Institute of Science and Technology, Vol. 11, p. 218.

Syswerda, G. (1989). Uniform Crossover in Genetic Algorithms. In Proceeding of the 3rd International Conference on Genetic Algorithms, pp. 2-9, ISBN:1-55860-066-3. San Francisco, CA, USA. 1989.

Talbi, E. (2009). *Methaheuristics. From Design to Implementation*. John Wiley & Sons, Inc, Publication, ISBN 978-0-470-27858-1, New Jersey.

Zeleny, M. (1997). Towards the Tradeoffs-Free Optimality in MCDM, In *Multicriteria Analysis*, J. Climaco, (Ed). , pp. 596-601, Springer-Verlag, ISBN 9783540620747, Berlin, Heidleberg.

Zitzler, E. and Thiele, L. (1998). Multiobjective optimization using evolutionary algorithms—A comparative case study. In *Proceedings of the 5th International Conference on Parallel Problem Solving from Nature*. pp. 292-301, ISBN:3-540-65078-4, Springer-Verlag, London, UK. 1998.



Real-World Applications of Genetic Algorithms
Edited by Dr. Olympia Roeva

ISBN 978-953-51-0146-8
Hard cover, 376 pages
Publisher InTech
Published online 07, March, 2012
Published in print edition March, 2012

The book addresses some of the most recent issues, with the theoretical and methodological aspects, of evolutionary multi-objective optimization problems and the various design challenges using different hybrid intelligent approaches. Multi-objective optimization has been available for about two decades, and its application in real-world problems is continuously increasing. Furthermore, many applications function more effectively using a hybrid systems approach. The book presents hybrid techniques based on Artificial Neural Network, Fuzzy Sets, Automata Theory, other metaheuristic or classical algorithms, etc. The book examines various examples of algorithms in different real-world application domains as graph growing problem, speech synthesis, traveling salesman problem, scheduling problems, antenna design, genes design, modeling of chemical and biochemical processes etc.

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821