Genetic Programming and Evolvable Machines, 6, 163–190, 2005

Solving Multiobjective Optimization Problems Using an Artificial Immune System

CARLOS A. COELLO COELLO ccoello@cs.cinvestav.mx
NARELI CRUZ CORTÉS nareli@computacion.cs.cinvestav.mx
CINVESTAV-IPN, Evolutionary Computation Group, Depto. de Ingeniería Eléctrica, Sección de Computación, Av. Instituto Politécnico Nacional No. 2508, Col. San Pedro Zacatenco, México, D. F. 07300, Mexico

Submitted December 1, 2002; Revised December 15, 2003; Accepted January 16, 2004

Abstract. In this paper, we propose an algorithm based on the clonal selection principle to solve multiobjective optimization problems (either constrained or unconstrained). The proposed approach uses Pareto dominance and feasibility to identify solutions that deserve to be cloned, and uses two types of mutation: uniform mutation is applied to the clones produced and non-uniform mutation is applied to the "not so good" antibodies (which are represented by binary strings that encode the decision variables of the problem to be solved). We also use a secondary (or external) population that stores the nondominated solutions found along the search process. Such secondary population constitutes the elitist mechanism of our approach and it allows it to move towards the true Pareto front of a problem over time. Our approach is compared with three other algorithms that are representative of the state-of-the-art in evolutionary multiobjective optimization. For our comparative study, three metrics are adopted and graphical comparisons with respect to the true Pareto front of each problem are also included. Results indicate that the proposed approach is a viable alternative to solve multiobjective optimization problems.

Keywords: artificial immune system, multiobjective optimization, clonal selection

1. Introduction

Given that our own life depends on our immune system, it should be obvious why it is considered as one of the most important biological mechanisms that humans possess. In recent years, several researchers have developed computational models of the immune system that attempt to capture some of its most remarkable features such as its self-organizing capability [17, 25].

From the information processing perspective, the immune system can be seen as a parallel and distributed adaptive system [9, 19]. It is capable of learning, it uses memory and is capable of associative retrieval of information in recognition and classification tasks. Particularly, it learns to recognize patterns, it remembers patterns that it has been shown in the past and its global behavior is an emergent property of many local interactions [9]. All these features of the immune system provide, in consequence, great robustness, fault tolerance, dynamism and adaptability [17]. These are the properties of the immune system that mainly attract researchers to try to emulate it in a computer.

Most optimization problems naturally have several objectives to be achieved (normally conflicting with each other), but in order to simplify their solution, they are treated as if they had only one (the remaining objectives are normally handled as constraints). These problems with several objectives, are called "multiobjective" or "vector" optimization problems, and were originally studied in the context of economics. However, scientists and engineers soon realized that such problems naturally arise in all areas of knowledge.

Over the years, the work of a considerable number of operational researchers has produced a wide variety of techniques to deal with multiobjective optimization problems [31]. However, it was until relatively recently that researchers became aware of the potential of evolutionary algorithms (EAs) and other population-based heuristics in this area.

The first implementation of a multiobjective evolutionary algorithm (MOEA) dates back to the mid-1980s [39, 40]. Since then, a considerable amount of research has been done in this area, now known as evolutionary multiobjective optimization (EMO for short). The growing importance of this field is reflected by a significant increment (mainly during the last eight years) of technical papers in international conferences and peer-reviewed journals, books, special sessions at international conferences and interest groups on the Internet [7].[1]

The main motivation for using EAs (or any other population-based heuristics) in solving multiobjective optimization problems is because EAs deal simultaneously with a set of possible solutions (the so-called population) which allows us to find several members of the Pareto optimal set in a single run of the algorithm, instead of having to perform a series of separate runs as in the case of the traditional mathematical programming techniques [31]. Additionally, EAs are less susceptible to the shape or continuity of the Pareto front (e.g., they can easily deal with discontinuous and concave Pareto fronts), whereas these two issues are a real concern for mathematical programming techniques [3].

Despite the considerable amount of EMO research in the last few years, there have been very few attempts to extend certain population-based heuristics (e.g., cultural algorithms and particle swarm optimization). Particularly, the efforts to extend an artificial immune system to deal with multiobjective optimization problems have been practically inexistent until very recently. In this paper, we precisely provide one of the first proposals to extend an artificial immune system to solve multiobjective optimization problems (either with or without constraints). Our proposal is based on the clonal selection principle and is validated using several test functions and metrics, following the standard methodology adopted in the EMO community [7, 11].

2. Basic definitions

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

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

Then, $\vec{x}^*$ is the global minimum solution(s), f is the objective function, and the set Ω is the feasible region ($\Omega \in \mathcal{S}$), where $\mathcal{S}$ represents the whole search space.

Definition 2 (General multiobjective optimization problem (MOP). Find the vector $\vec{x}^* = [x_1^*, x_2^*, \ldots, x_n^*]^T$ which will satisfy the m inequality constraints:

$$g_i(\vec{x}) \geq 0 \quad i = 1, 2, \ldots, m \tag{2}$$

the p equality constraints

$$h_i(\vec{x}) = 0 \quad i = 1, 2, \ldots, p \tag{3}$$

and will optimize the vector function

$$\vec{f}(\vec{x}) = \left[f_1(\vec{x}), f_2(\vec{x}), \ldots, f_k(\vec{x})\right]^T \tag{4}$$

where $\vec{x} = [x_1, x_2, \ldots, x_n]^T$ is the vector of decision variables.

Having several objective functions, the notion of "optimum" changes, because in MOPs, the aim is to find good compromises (or "trade-offs") rather than a single solution as in global optimization. The notion of "optimum" that is most commonly adopted is that originally proposed by Francis Ysidro Edgeworth [15] and later generalized by Vilfredo Pareto [37]. Although some authors call *Edgeworth-Pareto optimum* to this notion (see for example [45]), it is normally preferred to use the most commonly accepted term: *Pareto optimum*. The formal definition is provided next.

Definition 3 (Pareto optimality). A point $\vec{x}^* \in \Omega$ is Pareto optimal if for every $\vec{x} \in \Omega$ and $I = \{1, 2, \ldots, k\}$ either,

$$\forall_{i \in I}(f_i(\vec{x}) = f_i(\vec{x}^*)) \tag{5}$$

or, there is at least one $i \in I$ such that

$$f_i(\vec{x}) > f_i(\vec{x}^*) \tag{6}$$

In words, this definition says that $\vec{x}^*$ is Pareto optimal if there exists infeasible vector $\vec{x}$ which would decrease some criterion without causing a simultaneous increase in at least one other criterion. The phrase "Pareto optimal" is considered to mean with respect to the entire decision variable space unless otherwise specified.

Other important definitions associated with Pareto optimality are the following:

Definition 4 (Pareto dominance). A vector $\vec{u} = (u_1, \ldots, u_k)$ is said to dominate $\vec{v} = (v_1, \ldots, v_k)$ (denoted by $\vec{u} \preceq \vec{v}$) if and only if u is partially less than v, i.e., $\forall i \in \{1, \ldots, k\},\ u_i \leq v_i \wedge \exists i \in \{1, \ldots, k\} : u_i < v_i$.

Definition 5 (Pareto optimal set). For a given MOP $\vec{f}(x)$, the Pareto optimal set ($\mathcal{P}^*$) is defined as:

$$\mathcal{P}^* := \{x \in \Omega \mid \neg\exists\, x' \in \Omega \quad \vec{f}(x') \preceq \vec{f}(x)\}. \tag{7}$$

Definition 6 (Pareto front). For a given MOP $\vec{f}(x)$ and Pareto optimal set $\mathcal{P}^*$, the Pareto front ($\mathcal{PF}^*$) is defined as:

$$\mathcal{PF}^* := \{\vec{u} = \vec{f} = (f_1(x), \ldots, f_k(x)) \mid x \in \mathcal{P}^*\}. \tag{8}$$

In general, it is not easy to find an analytical expression of the line or surface that contains these points and in most cases, it turns out to be impossible to do it. The normal procedure to generate the Pareto front is to compute the points Ω and their corresponding $f(\Omega)$. When there is a sufficient number of these, it is then possible to determine the nondominated points and to produce the Pareto front.

Pareto optimal solutions are also termed *non-inferior*, *admissible*, or *efficient* solutions [24]; their corresponding vectors are termed *nondominated*.

3. The immune system

One of the main goals of the immune system is to keep our organism healthy. The immune system is capable of recognizing and combating pathogens (infectious foreign elements). Molecular patterns expressed on those pathogens are called *antigens*. An antigen is any molecule that can be recognized by the immune system provoking its response. This immune response is specific to each antigen.

The cells called *lymphocytes* have a very important role in the immune system. There are two types of lymphocytes: B lymphocytes (or B cells) and T lymphocytes (or T cells). When an antigen is detected, those B cells that best recognize the antigen, will proliferate by cloning. Some of these new cloned cells will be differentiated into *plasma cells*, which are the most active *antibodies* secretors. Meanwhile, the other cloned cells will become *memory cells*. These new cloned cells suffer a high-rate somatic mutation (or hypermutation) that will promote their genetic variation. Subsequently, a mechanism of selective pressure will possibly result in the survival of cells with increased affinity. These mutations experienced by the clones are proportional to their affinity to the antigen. The highest affinity cloned cells experiment the lowest mutation rates, whereas the lowest affinity cloned cells have high mutation rates. Due to the random nature of this mutation process, some clones could be dangerous for the body and should therefore be eliminated.

Plasma cells are capable of secreting only one type of *antibodies*, which are relatively specific for the antigen. *Antibodies* are molecules that play the main role in the immune response. They are capable of adhering to the antigens in order to neutralize and mark them for elimination for other cells of the immune system.

These cloning and hypermutation processes are called the *clonal selection principle* [2].

Although the clonal selection principle operates on both B and T cells, we have only focused on the B cells process in order to provide a very simplified way to explain the immune response.

Once the antigens have been eliminated with the help of the antibodies, the regulatory mechanisms of the immune system will cause the removal of exceeding cells and the convergence to a somewhat stable state, capable of reflecting the past experiences faced by the immune system. However, some cells remain circulating throughout the body as *memory cells*. When the immune system is later exposed to the same type of antigen (or a similar one), these memory cells are activated, presenting a better and more efficient response. This second encounter with the same antigen is called *secondary response*.

The algorithm proposed in this paper is based on the clonal selection principle previously described.

```
Repeat
    1. Select an antigen $\mathcal{A}$ from $PA$
       ($PA$ = Population of Antigens)
    2. Take (randomly) $R$ antibodies from $PS$
       ($PS$ = Population of Antibodies)
    3. For each antibody $r \in R$, match it against
       the selected antigen $\mathcal{A}$
       Compute its match score (e.g., using Hamming distance)
    4. Find the antibody with the highest match score
       Break ties at random
    5. Add match score of winning antibody to its fitness
Until maximum number of cycles is reached
```

Figure 1. Immune system model (fitness scoring) pseudocode.

4. Previous work

One of the applications in which the emulations of the immune system has been found useful is to maintain diversity in the population of a genetic algorithm (GA) used to solve multimodal optimization problems [18, 42, 43]. The proposal in this case has been to use binary strings to model both antibodies and antigens. Then, matching of an antibody and an antigen is determined if their bit strings are complementary (i.e., maximally different). The algorithm proposed in this case to compute fitness is shown in Figure 1 (this algorithm, assumes a population that includes antigens and antibodies both represented with binary strings) [43]. The main idea of this approach is to construct a population of antigens and a population of antibodies. Antibodies are then matched against antigens and a fitness value is assigned to each antibody based on this matching (i.e., maximize matching between antigens and antibodies). Finally, a conventional genetic algorithm is used to replicate the antibodies that better match the antigens present.

Smith et al. [43] show that fitness sharing emerges when their emulation of the immune system is used. Furthermore, this approach is more efficient (computationally speaking) than traditional fitness sharing [13], and it does not require additional information regarding the number of niches to be formed.

This same approach has been used to handle constraints in evolutionary optimization [22, 23] and has also been hybridized with a multiobjective evolutionary algorithm [8, 29]. However, the first direct use of the immune system to solve multiobjective optimization problems reported in the literature is the work of Yoo and Hajela [51]. This approach uses a linear aggregating function to combine objective function and constraint information into a scalar value that is used as the fitness function of a GA. Then, the best designs according to this value are defined as antigens and the rest of the population as a pool of antibodies. The simulation of the immune system is then done as in the previous work of the authors where the technique is used to handle constraints [23]. The algorithm is the following:

1. Select randomly a single antigen from the population of antigens.
2. From the population of antibodies, take a sample (randomly selected) without replacement (Yoo and Hajela [51] suggest three times the number of antigens).

3. Each antibody in the sample is matched against the selected antigen, and a match score (based on the Hamming distance measured on the genotype) is computed.
4. The antibody with the highest score is identified, and ties are broken at random.
5. The matching score of the winning antibody is added to its fitness value (i.e., it is "rewarded").
6. The process is repeated a certain number of times (typically three times the number of antibodies).

This approach is applied to some structural optimization problems with two objectives (a two-bar truss structure, a simply supported I-beam, and a 10-bar truss structure). The use of different weights allows the authors to converge to a certain (pre-specified) number of points of the Pareto front, since they make no attempt to use any specific technique to preserve diversity. In this study, the approach is not compared to any other technique.

More recently, Anchor et al. [1] used both lexicographic ordering and Pareto-based selection in an evolutionary programming algorithm that adopts an artificial immune system to detect (computer) virus attacks and hacker intrusions in general. In this work, however, emphasis is placed on the application rather than on the multiobjective aspects of the algorithm, since that is the main aim of this work. Therefore, the algorithm is not compared to other multiobjective optimization approaches.

It is also worth mentioning CLONALG, which is an artificial immune system based on the clonal selection principle that has been used for (single-objective and multimodal) optimization [36]. CLONALG uses two populations: one of antigens and another one of antibodies. When used for optimization, the main idea of CLONALG is to reproduce individuals with a high affinity, then apply mutation (or blind variation) and select the improved maturated progenies produced. Note that "affinity" in this case, is defined in terms of better objective function values rather than in terms of genotypic similarities (as, for example, in pattern recognition tasks). This implies that CLONALG does not really use antigens when solving optimization problems, but, instead, the closeness of each antibody to the global optimum (measured in relative terms with respect to the set of solutions produced so far) defines the rate of hypermutation to be used.

In a related paper, Nunes de Castro and Von Zuben [35] proposed an artificial immune system that combines CLONALG with immune network theory [26]. This algorithm is called aiNet (artificial immune network). In this approach, antibodies are part of an immune network and the decision about which one will be cloned (proportional to its objective function value), suppressed or maintained depends on the interaction established by the immune network. An affinity measure is also used in this case (for all the cells in the network) and a threshold is adopted to determine which cells are suppressed (i.e., cells whose affinities are below this threshold are eliminated). This approach has been used for multimodal optimization [33].

The approach introduced in this paper can then be considered as the first attempt to use an artificial immune system to solve the general multiobjective optimization problem. To validate our proposal, we adopt the conventional methodology of the evolutionary multiobjective optimization community, which includes a comparison with respect to other algorithms using several test functions and metrics.

5. The proposed approach

As indicated before, our algorithm has taken ideas from the *clonal selection principle*, modeling the fact that only the highest affinity antibodies will proliferate. Antibodies, in our case, are representented by binary strings that encode the decision variables of the problem to be solved. However, we do not use a population of antigens, but only Pareto dominance and feasibility to identify solutions that deserve to be cloned. Additionally, our approach uses mutation [20] (uniform mutation is applied to the clones and non-uniform mutation is applied to the "not so good" antibodies, as we will see later on). We also use a secondary (or external) population that stores the nondominated solutions found along the search process. Such secondary population is the elitist mechanism most commonly adopted in multiobjective optimization, and it allows us to move towards the true Pareto front of a problem over time [7]. The implementation details of the secondary population adopted in our algorithm are provided in Section 5.3.

Note that despite the fact that the algorithm presented next is based on our previous proposal reported in [5], several important aspects of such algorithm have been modified, including the elimination of certain parameters required in our previous version.

5.1. The algorithm

Our algorithm named Multiobjective Immune System Algorithm (MISA) is the following:

1. The initial population is randomly generated.
2. Initialize the secondary population so that it is empty.
3. Determine for each individual in the population, if it is (Pareto) dominated or not. For constrained problems, determine if an individual is feasible or not. Note that Pareto dominance is determined only among individuals of the same class (e.g., we only compare a feasible individual with respect to another feasible individual).
4. Determine which are the "best" antibodies, since we will clone them adopting the following criteria:

   - If the problem is unconstrained, then all the nondominated individuals are cloned. If the number of nondominated individuals is less than 5% of the population size, then we select dominated individuals until reaching a number of individuals equal to the 5% of the population size. Obviously, those individuals that are dominated by a lower number of individuals are favored.
   - If the problem is constrained, then we select all the nondominated individuals that are feasible. If the number of individuals selected is less than 5% of the population size, then we select additional individuals (until completing the 5% required) using the following criteria:
     - First, select individuals that are feasible, even if they are dominated. Preference is given to individuals that are dominated by a lower number of individuals.
     - If more individuals are needed, then select infeasible individuals, as long as they are nondominated. In this case, preference is given to individuals with a lower amount of constraint violation.

  - Finally, if more individuals are still required, the last case is to select individuals which are infeasible and dominated. In this case, preference is given to individuals that are dominated by a lower number of individuals.

  Note that in practice, the last two cases are normally not reached, but are included in our implementation for completeness.

5. Copy the best antibodies (obtained from the previous step) into the secondary population.
6. We determine for each of the "best" antibodies the number of clones that we wish to create using the following criteria:

   – First, we estimate the total number of clones to be produced by all the individuals selected. The total number of clones to be produced is of 600% the total population size (this percentage was empirically derived after performing an exhaustive set of experiments). This total number of clones is equally distributed among all the individuals selected (e.g., if 6 individuals are selected for cloning, we estimate that 100 copies of each of them will be produced).
   – This initial estimate of the number of clones to be produced for each individual is increased or decreased using the following rules:
     - *Case 1*: When the secondary population is not full, then we determine the euclidean distances of each individual selected with respect to all the other individuals selected. These distances are computed in objective function space. Then, we compute the average euclidean distance (with respect to the individuals selected for cloning). Using this average value as a reference (or threshold), we determine which region is more densely populated: either the region above the average value or the region below it. For each individual selected, we determine if it belongs to the most densely populated region or to the less densely populated region. If the individual belongs to the most densely populated region, its number of clones is decreased in 50%. Conversely, if it belongs to the less densely populated region, its number of clones is increased in 50%. Finally, if the individual has an euclidean distance equal to the average value, its number of clones remains unchanged.
     - *Case 2*: When the secondary population is full, there are three possible cases:

       (a) If the individual to be inserted into the secondary population is not allowed access either because it is already within it, or because it belongs to the most crowded region of objective function space, then the number of clones created from this antibody is zero.
       (b) When we have an individual that belongs to a cell whose number of solutions contained is below average (with respect to all the occupied cells in the secondary population), then the number of clones to be generated is duplicated.
       (c) When we have an individual that belongs to a cell whose number of solutions contained is above average (with respect to all the occupied cells in the adaptive grid—see Section 5.3 for details of the adaptive grid), then the number of clones to be generated is reduced by half. Note that this average value is used as a threshold to regulate the number of clones to be produced

(this number can be increased or decreased), in an analogous way to aiNet [33].

7. We perform the cloning of the best antibodies based on the information obtained from the previous step.
8. A mutation operator is applied to the clones. In order to determine the number of bits to be mutated in each individual, we rank them using the same criteria from step 4 (i.e., individuals that are both feasible and nondominated are given the highest possible rank). The individual with the highest rank is mutated in N positions, where N is the number of decision variables (the specific bits to be mutated are randomly chosen). When we move down one position in our hierarchy (i.e., when we run out of feasible nondominated individuals, and we have selected individuals that are feasible but dominated by a low number of individuals), the number of bits to be mutated is increased by one. This process continues until reaching the lowest possible rank as described in step 4. Note that we are using an antigenic affinity measure corresponding to Pareto dominance and feasibility of an individual. In our case, however, there is no explicit antigen population to be recognized, but only a set of objective functions to be optimized and a set of constraints to be satisfied. It is also worth mentioning that the hypermutation rate for each antibody is proportional to its affinity value as corresponding to the ranks described in step 4. This scheme follows the suggestions from [36] for problems with multiple optima.
9. We apply a non-uniform mutation operator to the “not so good” antibodies found. In this case, the initial mutation rate adopted is high and it is decreased linearly over time (from 0.6 to $1/L$, where L is the total number of bits of each individual).
10. The population size returns to its original value by selecting as many individuals as the size required using the same criteria described in step 4.
11. We repeat this process from step 3 during a certain (predetermined) number of times.

Information extracted from the secondary (external) population is used to guide some aspects of the algorithm, such as the number of clones created from each antibody. This is possible because the external population divides the Pareto front into regions, and its filtering mechanism avoids too crowded regions (or cells) of the grid used to store this secondary population.

The clonal selection principle seems to be very suitable to solve multiobjective optimization problems, mainly because of the following reasons:

- It is a population-based approach, which is important in order to allow the generation of several elements from the Pareto optimal set in one run.
- It allows to perform a local search in different directions along the Pareto front (i.e., the exploitation stage) by using the cloning process. At the same time, the mutation operator applied to the worst antibodies performs the exploration through the whole search space, trying to get closer to the true Pareto front of the problem. This balance between exploration and exploitation has been recently targeted as a very promising research area [7] and therefore our interest in adopting the clonal selection principle for multiobjective optimization.

- The use of an affinity measure is an interesting alternative to maintain diversity (instead of using fitness sharing) in the primary population of our algorithm.

5.2. Biological inspiration

Since the value of using immunological concepts in our algorithm may be questioned, we have decided to discuss briefly the components of our approach with respect to a traditional artificial immune system. For this discussion, we used the framework provided in [32], according to which, an artificial immune system, being a biologically inspired algorithm, requires are least three basic elements:

- A representation for the components of the system.
- A set of mechanisms to evaluate the interaction of individuals with the enviroment and with each other.
- Procedures of adaptation that govern the dynamics of the system (i.e., how does it behave over time).

According with these previous criteria, we can say the following about the components of our algorithm:

- Antibodies are encoded as binary strings, which represent potential solutions to the problem. As in CLONALG (when used for optimization), we do not use a population of antigens [36].
- The interaction between individuals with the enviroment is defined by the objective functions, and the relationship between individuals (antibodies) is defined by Pareto dominance relationships and by feasibility. Also, we use an affinity measure in the main population to control the amount of hypermutation to be applied to each antibody. Additionally, there is an interaction with the secondary population to decide (based on crowding criteria) how many clones to produce.
- The procedures of adaptation of our algorithm are: cloning and mutation.

We can then say that our algorithm is an artificial immune system based solely on immune-inspired concepts, derived from clonal selection theory and immune network theory.

5.3. Secondary population

We use a secondary or external population as an elitist mechanism in order to maintain the best solutions found along the process. The individuals stored in this population are all nondominated not only with respect to each other but also with respect to all of the previous individuals who attempted to enter the external population. Therefore, the external population stores our approximation to the true Pareto front of the problem.

In order to enforce a uniform distribution of nondominated solutions that cover the entire Pareto front of a problem, we implemented the adaptive grid proposed by Knowles and Corne [28] (see Figure 2). The grid is called "adaptive" because its boundaries (defined in

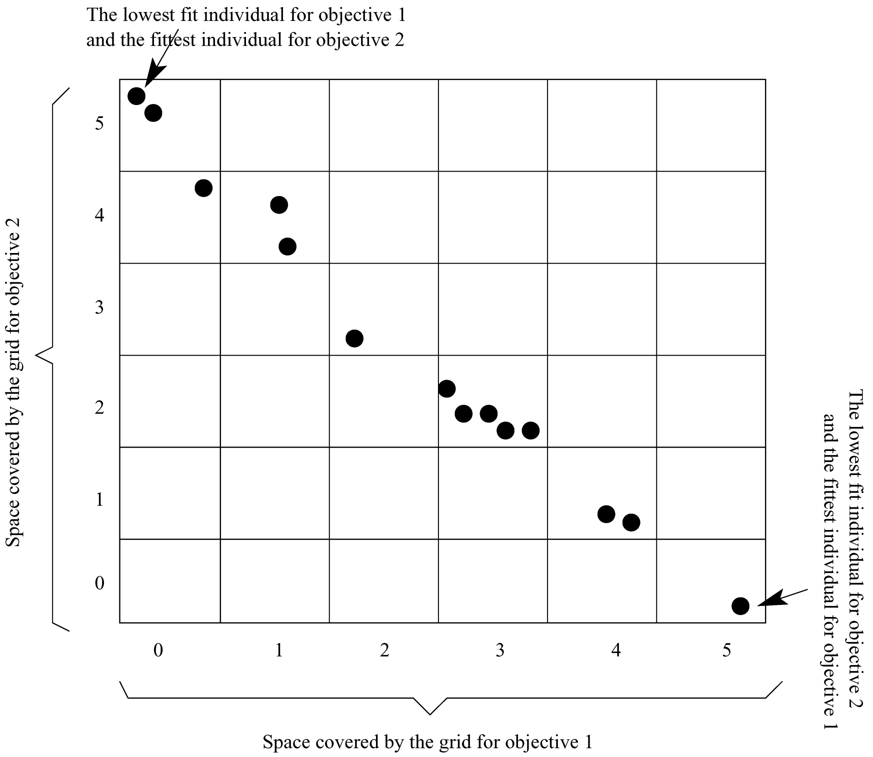
The lowest fit individual for objective 1
and the fittest individual for objective 2
Space covered by the grid for objective 2
5
4
3
2
1
0
0
1
2
3
4
5
Space covered by the grid for objective 1
The lowest fit individual for objective 2
and the fittest individual for objective 1

Figure 2. An adaptive grid to handle the secondary population.

terms of the maximum and minimum values available so far for each objective function) change over time, "adapting" to the new values that appear within the secondary population.

Ideally, the size of the external population should be infinite. However, since this is not possible in practice, we must set a limit to the number of nondominated solutions that we want to store in this secondary population. By enforcing this limit, our external population will get full at some point even if there are more nondominated individuals wishing to enter. When this happens, we use an additional criterion to allow a nondominated individual to enter the external population: region density (i.e., individuals belonging to less densely populated regions are given preference).

The algorithm for the implementation of the adaptive grid is the following:

1. Divide objective function space according to the number of subdivisions set by the user.
2. For each individual in the external population, determine the cell to which it belongs.
3. If the external population is full, then determine which is the most crowded cell.
4. To determine if a certain antibody is allowed to enter the external population, do the following:

   – If it belongs to the most crowded cell, then it is not allowed to enter.

  - Otherwise, the individual is allowed to enter. For that sake, we eliminate a (randomly chosen) individual that belongs to the most crowded cell in order to have an available slot for the incoming antibody.

6. Experiments

In order to validate our approach, we used several test functions reported in the standard evolutionary multiobjective optimization literature [7, 10, 48]. In each case, we generated the true Pareto front of the problem (i.e., the solution that we wished to achieve) by enumeration using parallel processing techniques. Then, we plotted the Pareto front generated by our algorithm MISA.

The results indicated below were found using the following parameters for MISA: Population size = 100, number of grid subdivisions = 25, size of the external population = 100 (we have eliminated several parameters adopted in our previous work [5]). These parameters produce a total of 12,000 fitness function evaluations, which is a much lower number than the 138,000 fitness function evaluations reported in our previous work [5].

MISA was compared against the micro-genetic algorithm for multiobjective optimization (microGA) [6], against the NSGA-II [12, 14] and against PAES [28]. These three algorithms were chosen because they are representative of the state-of-the-art in evolutionary multiobjective optimization and their codes are in the public domain.

The Pareto Archived Evolution Strategy (PAES) was proposed by Knowles and Corne [28]. PAES consists of a $(1 + 1)$ evolution strategy (i.e., a single parent that generates a single offspring) in combination with a historical archive that records some of the nondominated solutions previously found. This archive is used as a reference set against which each mutated individual is being compared. An interesting aspect of this algorithm is the mechanism used to maintain diversity which consists of a crowding procedure that divides objective space in a recursive manner. Each solution is placed in a certain grid location based on the values of its objectives (which are used as its “coordinates” or “geographical location”). A map of such grid is maintained, indicating the number of solutions that reside in each grid location. Since the procedure is adaptive, no extra parameters are required (except for the number of divisions of the objective space).

The Nondominated Sorting Genetic Algorithm II (NSGA-II) was proposed by Deb et al. [12, 14] and is a revised version of the NSGA [44]. The NSGA is based on several layers of classifications of the individuals. Before selection is performed, the population is ranked on the basis of nondomination: all nondominated individuals are classified into one category (with a dummy fitness value, which is proportional to the population size, to provide an equal reproductive potential for these individuals). The NSGA-II is more efficient (computationally speaking) than the NSGA, it uses elitism and a crowding comparison operator that keeps diversity without specifying any additional parameters. The NSGA-II uses $(\mu + \lambda)$-selection instead of a secondary population (as PAES) as its elitist mechanism.

The Micro Genetic Algorithm for Multiobjective Optimization was proposed by Coello Coello and Toscano Pulido [6]. A micro-genetic algorithm is a GA with a small population and a reinitialization process. The way in which the microGA works is the following. First, a random population is generated. This random population feeds the population memory, which is divided in two parts: a replaceable and a non-replaceable portion. The non-replaceable portion of the population memory never changes during the entire run and is meant to provide the required diversity for the algorithm. In contrast, the replaceable portion changes after each cycle of the microGA. The population of the microGA at the beginning of each of its cycles is taken (with a certain probability) from both portions of the population memory so that there is a mixture of randomly generated individuals (non-replaceable portion) and evolved individuals (replaceable portion). During each cycle, the microGA undergoes conventional genetic operators. After the microGA finishes one cycle, two nondominated vectors are chosen[2] from the final population and they are compared with the contents of the external population (this population is initially empty). If either of them (or both) remains as nondominated after comparing it against the vectors in this external population, then they are included there (i.e., in the external population). This is the historical archive of nondominated vectors. All dominated vectors contained in the external population are eliminated. The microGA uses then three forms of elitism: (1) it retains nondominated solutions found within the internal cycle of the microGA, (2) it uses a replaceable memory whose contents is partially "refreshed" at certain intervals, and (3) it replaces the population of the microGA by the nominal solutions produced (i.e., the best solutions found after a full internal cycle of the microGA).

In general, modern MOEAs (Multiobjective Evolutionary Algorithms) share three main similarities (this applies to MISA as well): (1) they all use Pareto ranking, which means that individuals in the population are ranked based on the concept of Pareto dominance (i.e., nondominated individuals are given the highest rank), (2) they use some form of elitism that allows to retain solutions that are globally (rather that only locally) nondominated (i.e., with respect not only to the current population, but also to all the previous populations), and (3) they all use a mechanism to preserve diversity.

What distinguishes one MOEA of another is the specific mechanism used for each of these 3 components. For example, the NSGA-II ranks the population using layers, whereas the microGA ranks the entire population at once.

Regarding MISA, it differs from the other MOEAs adopted in our comparative study in several aspects. First of all, MISA does not use any recombination operator and it adopts an affinity measure to control the amount of hypermutation applied to each antibody. Also, no current MOEA replicates (i.e., clones) solutions as MISA does. Finally, none of the MOEAs adopted for this study interact with the secondary population to affect the reproduction capabilities of an individual. In contrast, MISA decides how many clones to produce from a certain antibody based on how crowded is the region to which it belongs in the secondary population. These (apparently small) differences distinguish one type of MOEA from another. Interestingly, it has been found (empirically) that such changes can be translated into a significant difference in terms of performance and therefore the current extended proliferation of new MOEAs [7].

To allow a fair comparison among the approaches used, we adopted a criterion normally used in evolutionary multiobjective optimization [7]: we made sure that all the algorithms performed the same number of fitness function evaluations as MISA, and we also adopted the same size for their secondary populations (when applicable). Although this sort of practice does not necessarily allow the best possible performance for each of the MOEAs adopted, we made sure that the combination of parameters chosen for each of the algorithms compared was appropriate for the approach to have a reasonably good performance. This can be corroborated by checking the original sources of each of the methods compared.

In the following examples, the NSGA-II was run using a population size of 100, a crossover rate of 0.75, tournament selection, and a mutation rate of $1/N$, where N = number of decision variables of the problem. PAES was run using a mutation rate of $1/L$, where L refers to the length of the chromosomic string that encodes the decision variables. For the microGA we used an internal population size of 4 individuals, 15 subdivisions of the adaptive grid, a maximum number of generations of 750, a crossover rate of 0.8 and a mutation rate of $1/L$ as in the NSGA-II and PAES.

Despite the graphical comparisons performed, the three following metrics were adopted to allow a quantitative comparison of results:

- *Error Ratio (ER)*: This metric was proposed by Van Veldhuizen [46] to indicate the percentage of solutions (from the nondominated vectors found so far) that are not members of the true Pareto optimal set:

  $$ER = \frac{\sum_{i=1}^{n} e_i}{n}, \tag{9}$$

  where n is the number of vectors in the current set of nondominated vectors available; e_i = 0 if vector i is a member of the Pareto optimal set, and $e_i = 1$ otherwise. It should then be clear that $ER = 0$ indicates an ideal behavior, since it would mean that all the vectors generated by our algorithm belong to the Pareto optimal set of the problem.
- *Spacing (S)*: This metric was proposed by Schott [41] as a way of measuring the range (distance) variance of neighboring vectors in the known Pareto front. This metric is defined as:

  $$S \triangleq \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(\overline{d} - d_i)^2}, \tag{10}$$

  where $d_i = \min_j(\mid f_1^i(\vec{x}) - f_1^j(\vec{x}) \mid + \mid f_2^i(\vec{x}) - f_2^j(\vec{x}) \mid), i, j = 1, \ldots, n, \overline{d}$ is the mean of all d_i, and n is the number of vectors in the Pareto front found by the algorithm being evaluated. A value of zero for this metric indicates all the nondominated solutions found are equidistantly spaced.
- *Inverted generational distance (GD)*: The concept of generational distance was introduced by Van Veldhuizen and Lamont [47, 49] as a way of estimating how far are the elements in the Pareto front produced by our algorithm from those in the true Pareto front of the problem. This metric is defined as:

  $$GD = \frac{\sqrt{\sum_{i=1}^{n} d_i^2}}{n} \tag{11}$$

  where n is the number of nondominated vectors found by the algorithm being analyzed and d_i is the Euclidean distance (measured in objective space) between each of these and the nearest member of the true Pareto front. It should be clear that a value of $GD = 0$ indicates that all the elements generated are in the true Pareto front of the problem. Therefore, any other value will indicate how "far" we are from the global Pareto front of our problem. Similar metrics were proposed by Rudolph [38], Schott [41], and Zitzler et al. [52]. In our case, and following the suggestion of a reviewer, we implemented an "inverted" generational distance metric in which we use as a reference the true Pareto front, and we compare each of its elements with respect to the front produced by an algorithm. This intends to reduce some of the problems that occur with the generational distance metric when an algorithm generates very few nondominated solutions.

Example 1. Our first example is a bi-objective problem proposed by Deb [10]:
Minimize: $F = (f_1(x, y), f_2(x, y))$, where

$$
\begin{aligned}
f_1(x, y) &= x,\\
f_2(x, y) &= (1 + 10y)\\
&\quad \times \left[1 - \left(\frac{x}{1 + 10y}\right)^{\alpha} - \frac{x}{1 + 10y}\sin(2\pi q x)\right]
\end{aligned}
$$

and $0 \leq x, y \leq 1, q = 4, \alpha = 2$.

In this case, both decision variable space and objective function space are disconnected (the true Pareto front of the problem consists of 4 Pareto curves).

The comparison of results between the true Pareto front of this example and the Pareto front produced by MISA, the microGA, the NSGA-II and PAES are shown in Figures 3 and 4. Note that the Pareto front is disconnected (it consists of four Pareto curves). All the Pareto fronts shown in this paper correspond to the mean result (produced by each algorithm) with respect to the inverted generational distance metric. Therefore, they indicate the average behavior of each algorithm with respect to closeness to the true Pareto front of each problem.

The values of the three metrics for each algorithm are presented in Tables 1, 2, and 3, respectively.

Regarding error ratio, PAES had the best average value, closely followed by the NSGA-II. MISA ranked third with respect to this metric, but had the lowest standard deviation.

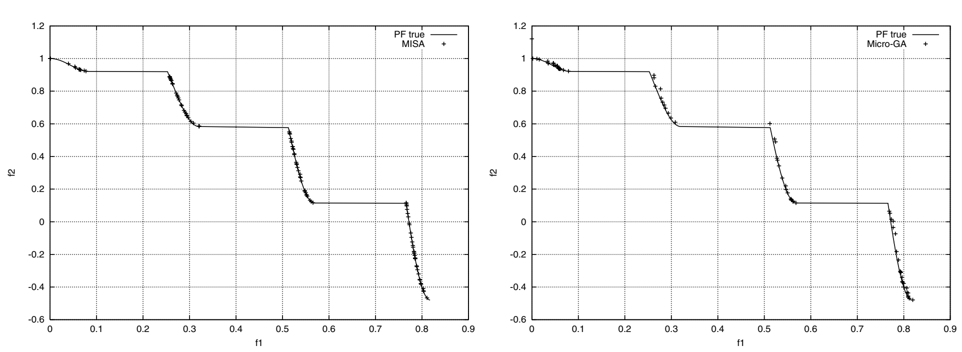

Figure 3. Pareto fronts produced by MISA (left) and the microGA (right) for the first test function. The true Pareto front is shown as a continuous line (note that the horizontal segments are NOT part of the Pareto front and are shown only to facilitate drawing the front).

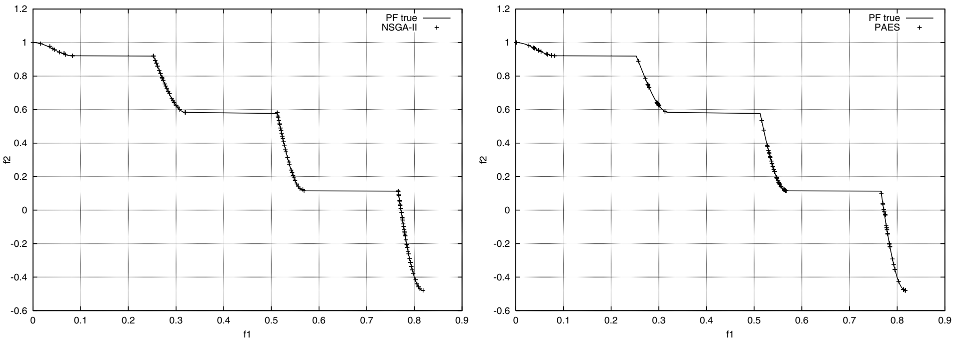

Figure 4. Pareto fronts produced by the NSGA-II (left) and PAES (right) for the first test function.

The worst overall performer (i.e., in terms of average value and standard deviation) was the microGA. With respect to inverted generational distance, MISA had the best performance (both in terms of average value and standard deviation), followed by the microGA. Regarding spacing (i.e., uniform distribution of solutions), MISA and the NSGA-II both found the best results with very similar average values (0.0082 for MISA vs. 0.0076 for the NSGA-II). PAES and the microGA ranked second and third, respectively. The worst performer with respect to this metric was the microGA. Thus, in terms of the metrics, there is no clear winner.

Table 1. Error Ratio for Example 1.

	MISA	MicroGA	NSGA-II	PAES
Average	0.42800	0.70832	0.14250	0.13300
Best	0.34000	0.53000	0.08000	0.06000
Worst	0.51000	0.93333	0.29000	0.26000
Std. dev.	0.04641	0.15826	0.05035	0.05121

Table 2. Spacing for Example 1.

	MISA	MicroGA	NSGA-II	PAES
Average	0.00816	0.18410	0.00760	0.03229
Best	0.00722	0.02114	0.00657	0.01246
Worst	0.00904	0.50707	0.00920	0.25179
Std. dev.	0.00057	0.15078	0.00065	0.05240

Table 3. Inverted Generational Distance for Example 1.

	MISA	MicroGA	NSGA-II	PAES
Average	0.00030	0.00068	0.00105	0.00304
Best	0.00029	0.00057	0.00052	0.00012
Worst	0.00033	0.00100	0.00308	0.02520
Std. dev.	0.00001	0.00013	0.00057	0.00758

Since metrics can sometimes be misleading in multiobjective optimization, it is always important to rely on graphical comparisons (whenever possible). When analyzing the Pareto front produced by an approach, it is important to identify two main things: (1) if the solutions are placed on the true Pareto front (which is indicated as a continuous line in this case), and (2) how uniform is the distribution of solutions along the Pareto front. In Figures 3 and 4, we can see that the NSGA-II clearly produced the best Pareto fronts, followed by MISA. In contrast, the microGA and PAES both missed several portions of the true Pareto front. Thus, we conclude that for this test function the NSGA-II had the best overall performance, followed by MISA. PAES and the microGA ranked third and fourth, respectively.

Example 2. Our second example is a two-objective optimization problem proposed by Schaffer [39] that has been used by several researchers [44]:

$$\text{Minimize } f_1(x) = \begin{cases} -x & \text{if } x \leq 1 \\ -2 + x & \text{if } 1 < x \leq 3 \\ 4 - x & \text{if } 3 < x \leq 4 \\ -4 + x & \text{if } x > 4 \end{cases} \tag{12}$$

$$\text{Minimize } f_2(x) = (x - 5)^2 \tag{13}$$

and $-5 \leq x \leq 10$.

The comparison of results between the true Pareto front of this example and the Pareto front produced by MISA, the microGA, the NSGA-II and PAES are shown in Figures 5 and 6. The values of the three metrics for each algorithm are presented in Tables 4, 5, and 6, respectively.

In this case, the microGA had the best performance (both in terms of best average value found and in terms of lowest standard deviation) with respect to error ratio (followed by MISA). With respect to spacing, MISA had the best performance (both in terms of best

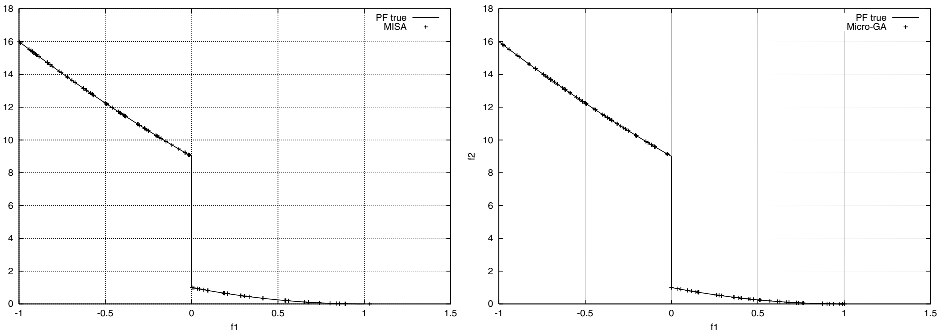

Figure 5. Pareto fronts produced by MISA (left) and the microGA (right) for the second test function. The true Pareto front is shown as a continuous line (note that the vertical segment is NOT part of the Pareto front and is shown only to facilitate drawing the front).

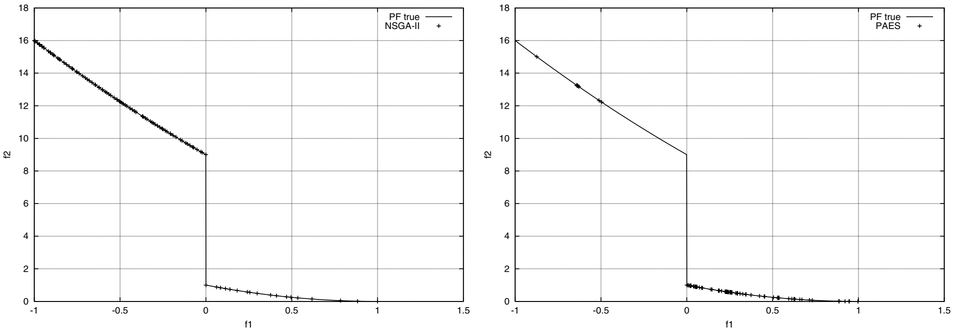

Figure 6. Pareto fronts produced by the NSGA-II (left) and PAES (right) for the second test function.

average value found and in terms of lowest standard deviation), followed by the NSGA-II. Finally, with respect to inverted generational distance, the NSGA-II had the best performance (both in terms of best average value found and in terms of lowest standard deviation) closely followed by MISA. Thus, with respect to the metrics used, we can see that PAES is clearly the worst performer, and MISA seems to be the most suitable candidate for best performer since it was the second best both with respect to error ratio and with respect to inverted generational distance. It is also interesting to see that MISA had the best spacing in

Table 4. Error Ratio for Example 2.

	MISA	MicroGA	NSGA-II	PAES
Average	0.00800	0.00550	0.01050	0.01550
Best	0.00000	0.00000	0.00000	0.00000
Worst	0.04000	0.02000	0.04000	0.10000
Std. dev.	0.01005	0.00605	0.01099	0.02645

Table 5. Spacing for Example 2.

	MISA	MicroGA	NSGA-II	PAES
Average	0.00816	0.05443	0.04422	0.17232
Best	0.00722	0.04726	0.03906	0.01003
Worst	0.00904	0.05932	0.05035	0.53812
Std. dev.	0.00057	0.00460	0.00292	0.13835

Table 6. Inverted Generational Distance for Example 2.

	MISA	MicroGA	NSGA-II	PAES
Average	0.00173	0.00218	0.00097	0.03568
Best	0.00136	0.00216	0.00088	0.00376
Worst	0.00225	0.00219	0.00111	0.23518
Std. dev.	0.00030	0.00001	0.000006	0.06748

this case, since this contrasts with the (typically not so good) distributions that the previous version of the algorithm normally obtained [5].

Graphically, we can seen that all the algorithms converged to the true Pareto front of the problem. However, PAES had problems with the distribution of points from the top section of the Pareto front. The NSGA-II produced a lot of points on the top portion of the front, but it had some small holes on the lower portion. MISA had a few holes both at the top and bottom portions, but it managed to produce most of the front. The microGA had less solutions on the top part of the Pareto front than the NSGA-II, but its distribution on the lower portion of the front looks better than the NSGA-II. Thus, from the graphical comparisons, it is difficult to determine a clear winner, but, except for PAES, all the other algorithms seem to have had a reasonably good performance in this problem.

Example 3. The third example is the three-objective function problem proposed by Viennet [50]:

$$\text{Minimize}: \ F = (f_1(x, y), f_2(x, y), f_3(x, y))$$

where

$$\begin{aligned} f_1(x, y) &= \frac{(x-2)^2}{2} + \frac{(y+1)^2}{13} + 3, \\ f_2(x, y) &= \frac{(x+y-3)^2}{175} + \frac{(2y-x)^2}{17} - 13, \\ f_3(x, y) &= \frac{(3x-2y+4)^2}{8} + \frac{(x-y+1)^2}{27} + 15 \end{aligned}$$

and $-4 \leq x, y \leq 4$, $y < -4x + 4$, $x > -1$, $y > x - 2$.

The comparison of results between the true Pareto front of this example and the Pareto front produced by MISA, the microGA, the NSGA-II and PAES are shown in Figures 7

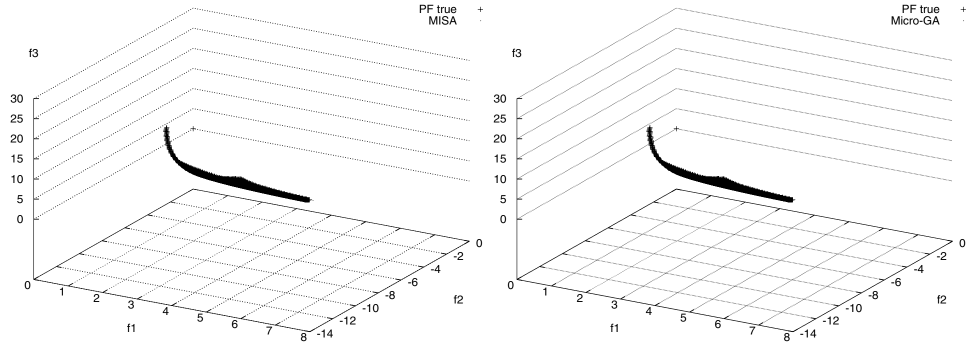

Figure 7. Pareto fronts produced by MISA (left) and the microGA (right) for the third test function. The true Pareto front is shown as dots.

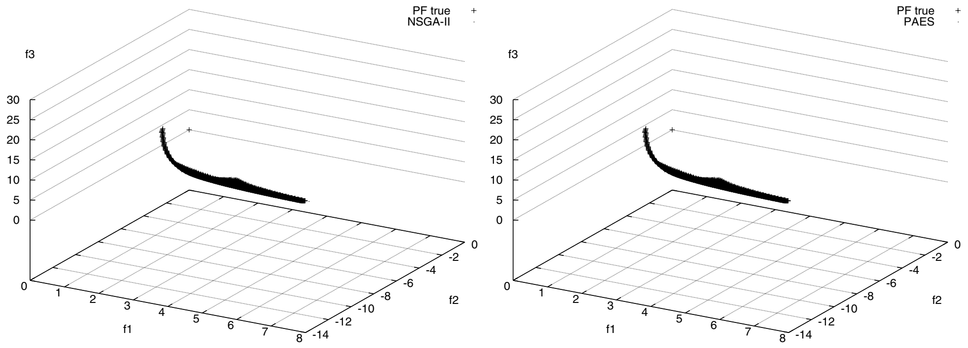

Figure 8. Pareto fronts produced by the NSGA-II (left) and PAES (right) for the third test function.

and 8. The values of the three metrics for each algorithm are presented in Tables 7, 8, and 9, respectively.

In this example, PAES had the best error ratio, followed by MISA. The NSGA-II had the worst performance with respect to this metric. Regarding spacing, the NSGA-II had the best performance, closely followed by PAES. MISA had the worst performance with respect to spacing in this case. Finally, with respect to inverted generational distance, the microGA had the best performance, closely followed by MISA and the two other algorithms. Thus,

Table 7. Error Ratio for Example 3.

	MISA	MicroGA	NSGA-II	PAES
Average	0.08700	0.11500	0.25550	0.04750
Best	0.02000	0.05000	0.10000	0.00000
Worst	0.18000	0.15000	0.45000	0.10000
Std. dev.	0.03262	0.02417	0.12361	0.03654

Table 8. Spacing for Example 3.

	MISA	MicroGA	NSGA-II	PAES
Average	0.46113	0.23300	0.13655	0.17736
Best	0.33834	0.14796	0.08508	0.11659
Worst	0.51544	0.36788	0.17795	0.53240
Std. dev.	0.04905	0.06195	0.02557	0.09875

Table 9. Inverted Generational Distance for Example 3.

	MISA	MicroGA	NSGA-II	PAES
Average	0.01216	0.01201	0.01382	0.01452
Best	0.01176	0.01179	0.01250	0.01280
Worst	0.01317	0.01332	0.01690	0.02370
Std. dev.	0.00050	0.00034	0.00121	0.00246

in terms of the metrics, MISA seems to be the most suitable candidate to be considered the best performer. Note however that all the algorithms converged very close to the true Pareto front of the problem. This can be corroborated by looking at the graphical results, since all the algorithms produced similar approximations of the Pareto front of this problem.

Example 4. The fourth example was proposed by Kita et al. [27]:

$$\text{Maximize } F = (f_1(x, y), f_2(x, y))$$

where

$$\begin{aligned} f_1(x, y) &= -x^2 + y, \\ f_2(x, y) &= \frac{1}{2}x + y + 1 \\ x, y \geq 0, \quad 0 &\geq \frac{1}{6}x + y - \frac{13}{2}, \quad 0 \geq \frac{1}{2}x + y - \frac{15}{2}, \quad 0 \geq 5x + y - 30. \end{aligned}$$

This problem has a concave Pareto front.

The comparison of results between the true Pareto front of this example and the Pareto front produced by MISA, the microGA, the NSGA-II and PAES are shown in Figures 9 and 10. The values of the three metrics for each algorithm are presented in Tables 10, 11, and 12, respectively.

In this case, MISA had the best performance in terms of the error ratio metric (both in terms of the best average value found and in terms of the lowest standard deviation). PAES ranked second with respect to error ratio. Regarding spacing, the NSGA-II ranked first and PAES ranked second. MISA had the third best average value with respect to spacing. Finally, with respect to inverted generational distance, MISA had the best performance (both in terms of the best average value found and in terms of the lowest standard deviation) and the microGA ranked second. Thus, in terms of the metrics, MISA is the best algorithm for this problem.

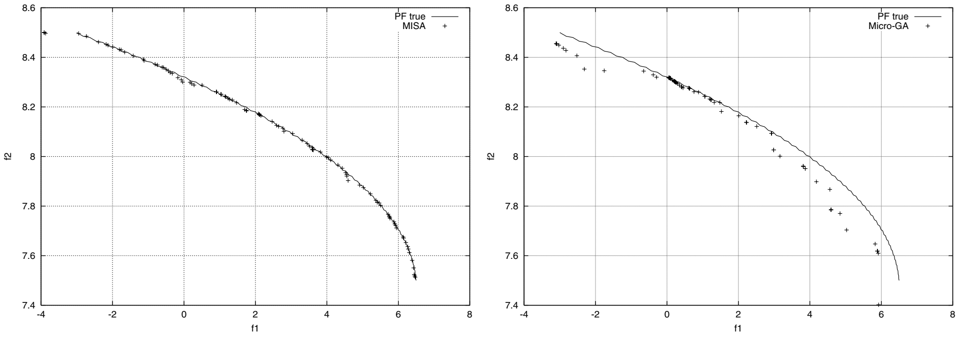

Figure 9. Pareto fronts produced by MISA (left) and the microGA (right) for the fourth test function. The true Pareto front is shown as a continuous line.

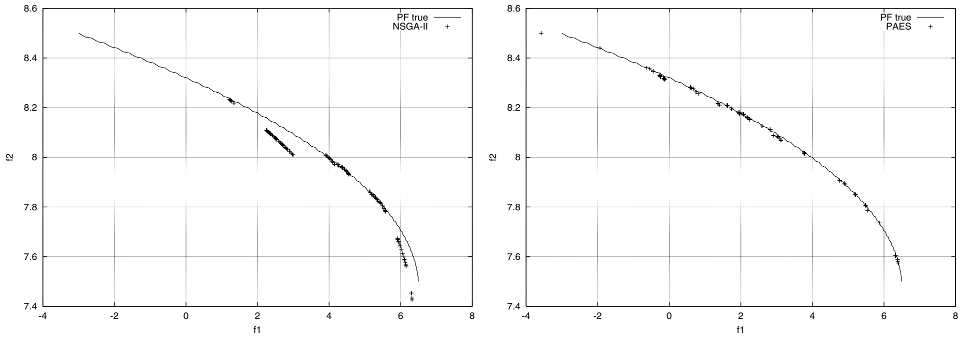

Figure 10. Pareto fronts produced by the NSGA-II (left) and PAES (right) for the fourth test function.

In this problem, however, the graphical results give a much better indicative of the performance of each algorithm. We can clearly see that the NSGA-II had a very good distribution of the solutions that it found, but it missed most of the Pareto front. Thus, the NSGA-II clearly had the worst performance for this problem, despite its good spacing values. Having a good distribution of solutions becomes irrelevant when the algorithm does not converge to the true Pareto front of the problem. Both the microGA and PAES also missed a considerable portion of the true Pareto front. MISA clearly had the best overall

Table 10. Error Ratio for Example 4.

	MISA	MicroGA	NSGA-II	PAES
Average	0.34000	0.80188	0.85409	0.65450
Best	0.34000	0.60000	0.69091	0.50000
Worst	0.34000	0.93182	1.00000	0.80000
Std. dev.	0.00000	0.09263	0.10091	0.07870

Table 11. Spacing for Example 4.

	MISA	MicroGA	NSGA-II	PAES
Average	0.17955	0.20139	0.02201	0.12157
Best	0.05932	0.04476	0.00724	0.04166
Worst	0.71776	0.94324	0.03504	0.22073
Std. dev.	0.21842	0.23828	0.00742	0.04998

Table 12. Inverted Generational Distance for Example 4.

	MISA	MicroGA	NSGA-II	PAES
Average	0.00497	0.01500	0.07066	0.01801
Best	0.00354	0.01064	0.01306	0.00859
Worst	0.00630	0.02597	0.16318	0.02356
Std. dev.	0.00092	0.00452	0.04473	0.00456

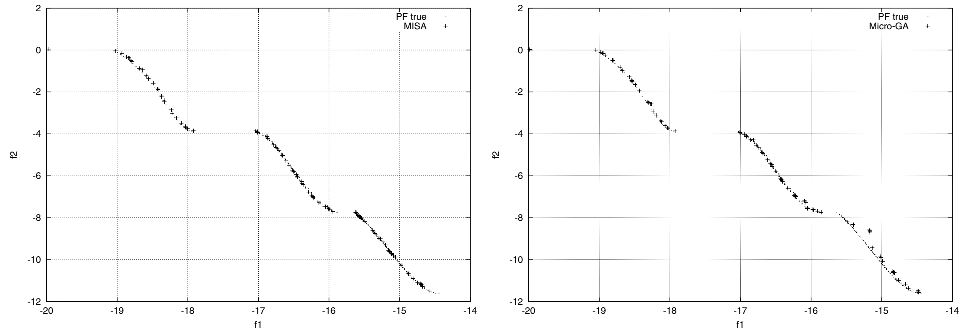

Figure 11. Pareto fronts produced by MISA (left) and the microGA (right) for the fifth test function. The true Pareto front is shown as a continuous line.

performance (both in terms of the graphical comparisons and in terms of the metrics) for this problem.

Example 5. Our fifth example is a two-objective optimization problem defined by Kursawe [30]:

$$\text{Minimize } f_1(\vec{x}) = \sum_{i=1}^{n-1}\left(-10\exp\left(-0.2\sqrt{x_i^2 + x_{i+1}^2}\right)\right) \tag{14}$$

$$\text{Minimize } f_2(\vec{x}) = \sum_{i=1}^{n}(|x_i|^{0.8} + 5\sin(x_i)^3) \tag{15}$$

where

$$-5 \leq x_1, x_2, x_3 \leq 5 \tag{16}$$

This problem has a disconnected Pareto front that consists of three segments.

The comparison of results between the true Pareto front of this example and the Pareto front produced by MISA, the microGA, the NSGA-II and PAES are shown in Figures 11 and 12. The values of the three metrics for each algorithm are presented in Tables 13, 14, and 15, respectively.

Table 13. Error Ratio for Example 5.

	MISA	MicroGA	NSGA-II	PAES
Average	0.43600	0.60261	0.25450	0.36650
Best	0.31000	0.52000	0.18000	0.06000
Worst	0.58000	0.70408	0.38000	0.88000
Std. dev.	0.08088	0.06640	0.05326	0.21656

Table 14. Spacing for Example 5.

	MISA	MicroGA	NSGA-II	PAES
Average	0.10906	0.12571	0.06134	0.19530
Best	0.06274	0.09505	0.04645	0.06556
Worst	0.14373	0.14264	0.09914	0.49154
Std. dev.	0.01592	0.01577	0.01284	0.10921

Table 15. Inverted Generational Distance for Example 5.

	MISA	MicroGA	NSGA-II	PAES
Average	0.00466	0.01133	0.01663	0.02450
Best	0.00433	0.00698	0.00680	0.00648
Worst	0.00558	0.02298	0.03447	0.09593
Std. dev.	0.00031	0.00538	0.00773	0.02443

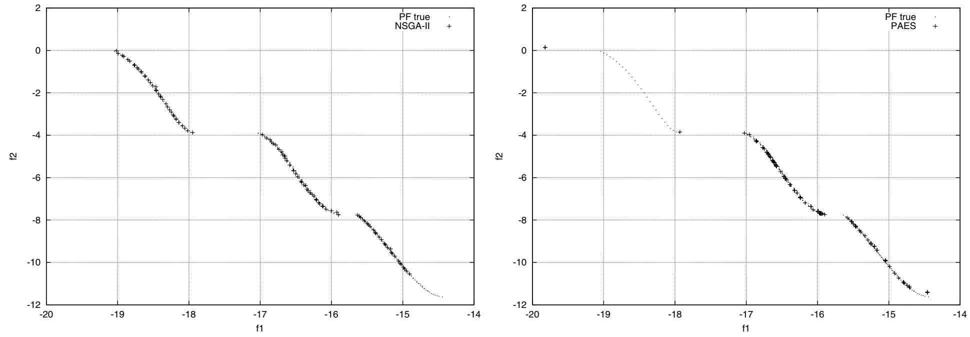

Figure 12. Pareto fronts produced by the NSGA-II (left) and PAES (right) for the fifth test function.

In this example, the NSGA-II had the best performance in terms of the error ratio metric (both in terms of the best average value found and in terms of the lowest standard deviation). PAES ranked second and MISA ranked third. Regarding spacing, the NSGA-II had again the best performance, and the microGA ranked second, closely followed by MISA. Finally, with respect to the inverted generational distance metric, MISA had the best performance (both in terms of the best average value and in terms of the lowest standard deviation), followed by the microGA. Based on the metrics, there is no clear winner.

Graphically, it can be clearly seen that PAES had the worst performance for this example (as indicated by the generational distance metric). The NSGA-II had a good distribution of points, but missed the lower right handside portion of the Pareto front. The microGA also had problems to generate this same portion of the Pareto front. In contrast, MISA was able to produce this portion of the Pareto front. Thus, based on the graphical results, MISA seems to be the best overall performer for this problem.

6.1. Discussion of results

From the results obtained, we could draw the following (preliminary) conclusions about the performance of our approach:

- MISA was able to produce most of the true Pareto front of all the problems adopted for our comparative study and it was the best performer in several of them.
- The best performance of the algorithm with respect to inverted generational distance (closeness to the true Pareto front) was observed in concave and disconnected Pareto fronts. In fact, in the only problem with a concave Pareto front, MISA was the only algorithm capable of converging to the true Pareto front.
- MISA ranked first in terms of spacing in one problem and practically tied with the NSGA-II in one more. It was the worst performer with respect to spacing in only one problem (the problem with 3 objectives).
- When dealing with a 3-objective problem, all the algorithms were able to converge very close to the true Pareto front, but their distribution of solutions was, in general terms, poor (MISA being the worst performer). Despite its bad performance in terms of spacing, MISA ranked second with respect to both error ratio and inverted generational distance in the 3-objective problem.

In general, we can see that MISA provided competitive results with respect to the three other algorithms against which it was compared. Although it did not always ranked first when using the three metrics adopted, in all cases it produced reasonably good approximations of the true Pareto front of each problem under study, including those with a concave or a disconnected Pareto front.

7. Conclusions and future work

We have presented a new multiobjective optimization algorithm based on the *clonal selection principle*. The approach is able to produce results similar or better than those generated by other three algorithms that are representative of the state-of-the-art in evolutionary multiobjective optimization. Our approach uses an affinity measure to control the amount of mutation to be applied to the antibodies. Affinity in this case, is defined in terms of nondominance and feasibility. This affinity measure, combined with the secondary population are used to distribute nondominated solutions in a uniform way. The approach proposed also uses a very simple mechanism to deal with constrained test functions, and our results indicate that such mechanism, despite its simplicity, is effective in practice.

From this limited comparative study, we conclude that artificial immune systems based on the clonal selection principle can be effectively used to solve multiobjective optimization problems in a relatively simple way.[3] We also believe that, given the features of artificial immune systems, an extension of this paradigm for multiobjective optimization (such as the one proposed here) may be particularly useful to deal with dynamic functions and that is precisely part of our future research.

As part of our future work, we are also considering the possibility of using spatial data structures to store and retrieve nondominated solutions in a more efficient way [21].

Finally, the use of alternative mechanisms to handle constraints (see for example [4]) may also improve the performance of our algorithm.

Acknowledgments

We thank the comments of the anonymous reviewers that greatly helped us to improve the contents of this paper. The first author gratefully acknowledges support from CONACyT through project 34201-A. The second author acknowledges support from CONACyT through a scholarship to pursue graduate studies at the Computer Science Section of the Electrical Engineering Department at CINVESTAV-IPN.

Notes

1. The first author maintains an EMO repository which currently contains over 1400 bibliographical entries at: http://delta.cs.cinvestav.mx/˜ccoello/EMOO, with mirrors at http://www.lania.mx/˜ccoello/EMOO/ and http://www.jeo.org/emo/
2. This is assuming that there are two or more nondominated vectors. If there is only one, then this vector is the only one selected.
3. The algorithm proposed here is rather simple to implement, but in any case, our source code is available upon request to the first author via email.

References

1. K. P. Anchor, J. B. Zydallis, G. H. Gunsch, and G. B. Lamont, "Extending the computer defense immune system: Network intrusion detection with a multiobjective evolutionary programming spproach," in First International Conference on Artificial Immune Systems (ICARIS'2002), J. Timmis and P. J. Bentley (Eds.), University of Kent at Canterbury, UK, Sept. 2002, pp. 12–21. ISBN 1-902671-32-5.
2. F. M. Burnet, "Clonal selection and after," in Theoretical Immunology, G. I. Bell, A. S. Perelson, and G. H. Pimgley Jr. (Eds.), Marcel Dekker Inc., 1978, pp. 63–85.
3. C. A. Coello Coello, "A comprehensive survey of evolutionary-based multiobjective optimization techniques," Knowledge and Information Systems. An International Journal, vol. 1, no. 3, pp. 269–308, 1999.
4. C. A. Coello Coello, "Theoretical and numerical constraint handling techniques used with evolutionary algorithms: A survey of the state of the art," Computer Methods in Applied Mechanics and Engineering, vol. 191, no. 11/12, pp. 1245–1287, 2002.
5. C. A. Coello Coello and N. Cruz Cortés, "An approach to solve multiobjective optimization problems based on an artificial immune system," in First International Conference on Artificial Immune Systems (ICARIS'2002), J. Timmis and P. J. Bentley (Eds.), University of Kent at Canterbury: UK, Sept. 2002, pp. 212–221. ISBN 1-902671-32-5.
6. C. A. Coello Coello and G. Toscano Pulido, "Multiobjective optimization using a micro-genetic algorithm," in Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'2001), L. Spector, E. D. Goodman, A. Wu, W.B. Langdon, H.-M. Voigt, M. Gen, S. Sen, M. Dorigo, S. Pezeshk, M. H. Garzon, and E. Burke (Eds.), Morgan Kaufmann Publishers: San Francisco, CA, 2001, pp. 274–282.
7. C. A. Coello Coello, D. A. Van Veldhuizen, and G. B. Lamont, "Evolutionary algorithms for solving multi-objective problems," Kluwer Academic Publishers, New York, May 2002, ISBN 0-3064-6762-3.
8. X. Cui, M. Li, and T. Fang, "Study of population diversity of multiobjective evolutionary algorithm based on immune and entropy principles," in Proceedings of the Congress on Evolutionary Computation 2001 (CEC'2001), IEEE Service Center: Piscataway, New Jersey, May 2001, vol. 2, pp. 1316–1321.
9. D. Dasgupta (Ed.), Artificial Immune Systems and Their Applications, Springer-Verlag: Berlin, 1999.
10. K. Deb, "Multi-objective genetic algorithms: Problem difficulties and construction of test problems," Evolutionary Computation, vol. 7, no. 3, pp. 205–230, Fall 1999.

11. K. Deb, Multi-Objective Optimization using Evolutionary Algorithms, John Wiley & Sons: Chichester, UK, 2001. ISBN 0-471-87339-X.
12. K. Deb, S. Agrawal, A. Pratab, and T. Meyarivan, "A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II," in Proceedings of the Parallel Problem Solving from Nature VI Conference, M. Schoenauer, K. Deb, G. Rudolph, X. Yao, E. Lutton, J. J. Merelo, and H.-P. Schwefel (Eds.), Springer: Paris, France, 2000, pp. 849–858. Lecture Notes in Computer Science No. 1917.
13. K. Deb and D. E. Goldberg, "An investigation of niche and species formation in genetic function optimization," in Proceedings of the Third International Conference on Genetic Algorithms, J. D. Schaffer (Ed.), George Mason University, Morgan Kaufmann Publishers: San Mateo, California, June 1989, pp. 42–50.
14. K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA–II," IEEE Transactions on Evolutionary Computation, vol. 6, no. 2, pp. 182–197, 2002.
15. F. Y. Edgeworth, Mathematical Physics. P. Keagan: London, England, 1881.
16. C. M. Fonseca and P. J. Fleming, "Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization," in Proceedings of the Fifth International Conference on Genetic Algorithms, S. Forrest (Ed.), Morgan Kauffman Publishers: San Mateo, CA, 1993, pp. 416–423.
17. S. Forrest and S. A. Hofmeyr, "Immunology as information processing," in Design Principles for the Immune System and Other Distributed Autonomous Systems, L. A. Segel and I. Cohen (Eds.), Santa Fe Institute Studies in the Sciences of Complexity, Oxford University Press, 2000, pp. 361–387.
18. S. Forrest and A. S. Perelson, "Genetic algorithms and the immune system," in Parallel Problem Solving from Nature, H.-P. Schwefel and R. Männer (Eds.), Lecture Notes in Computer Science, Springer-Verlag: Berlin, Germany, 1991, pp. 320–325.
19. S. A. Frank, The Design of Natural and Artificial Adaptive Systems. Academic Press: New York, 1996.
20. D. E. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley Publishing Company: Reading, MA, 1989.
21. W. Habenicht, "Quad trees: A data structure for discrete vector optimization problems," in Lecture Notes in Economics and Mathematical Systems, vol. 209, pp. 136–145, 1982.
22. P. Hajela and J. Lee, "Constrained genetic search via schema adaptation. An immune network solution," in Proceedings of the First World Congress of Stuctural and Multidisciplinary Optimization, N. Olhoff and G. I. N. Rozvany (Eds.), Pergamon: Goslar, Germany, 1995, pp. 915–920.
23. P. Hajela and J. Lee, "Constrained genetic search via schema adaptation. An immune network solution," Structural Optimization, vol. 12, pp. 11–15, 1996.
24. J. Horn, "Multicriterion secision making," in Handbook of Evolutionary Computation, T. Bäck, D. Fogel, and Z. Michalewicz (Eds.), IOP Publishing Ltd. and Oxford University Press, 1997, vol. 1, pp. F1.9:1–F1.9:15.
25. J. E. Hunt and D. E. Cooke, "An adaptative, distributed learning systems based on the immune system," in Proceedings of the IEEE International Conference on Systems, Man and Cybernatics, 1995, pp. 2494–2499.
26. N.K. Jerne, "Towards a network theory of the immune system," Annals of Immunology (Inst. Pasteur), vol. 125C, pp. 373–389, 1974.
27. H. Kita, Y. Yabumoto, N. Mori, and Y. Nishikawa, "Multi-objective optimization by means of the thermodynamical genetic algorithm," in Parallel Problem Solving from Nature—PPSN IV, H.-M. Voigt, W. Ebeling, I. Rechenberg, and H.-P. Schwefel (Eds.), Lecture Notes in Computer Science, Springer-Verlag: Berlin, Germany, Sept. 1996, pp. 504–512.
28. J. D. Knowles and D. W. Corne, "Approximating the nondominated front using the pareto archived evolution strategy," Evolutionary Computation, vol. 8, no. 2, pp. 149–172, 2000.
29. A. Kurpati and S. Azarm, "Immune network simulation with multiobjective genetic algorithms for multidisciplinary design optimization," Engineering Optimization, vol. 33, pp. 245–260, 2000.
30. F. Kursawe, "A variant of evolution strategies for vector optimization," in Parallel Problem Solving from Nature. 1st Workshop, PPSN I, H. P. Schwefel and R. Männer (Eds.), vol. 496 of Lecture Notes in Computer Science, Springer-Verlag: Berlin, Germany, Oct. 1991, pp. 193–197.
31. K. M. Miettinen, Nonlinear Multiobjective Optimization, Kluwer Academic Publishers: Boston, MA, 1998.
32. L. Nunes de Castro and J. Timmis, Artificial Immune Systems: A New Computational Intelligence Approach, Springer: London, 2002.
33. L. Nunes de Castro and J. Timmis, "An artificial immune network for multimodal function optimization," in Proceedings of the 2002 IEEE Congress on Evolutionary Computation (CEC'2002), Honolulu, Hawaii, IEEE, vol. 1, pp. 699–704, May 2002.

34. L. Nunes de Castro and F. J. Von Zuben, “Artificial immune systems: Part I—Basic theory and applications,” Technical Report TR-DCA 01/99, FEEC/UNICAMP, Brazil, Dec. 1999.
35. L. Nunes de Castro and F. J. Von Zuben, “aiNet: An artificial immune network for data analysis,” in Data Mining:A Heuristic Approach, H. A. Abbass, R. A. Sarker and C. S. Newton (Eds.), Idea Group Publishing, USA, 2001, pp. 231–259, Chap. XII.
36. L. Nunes de Castro and F. J. Von Zuben, “Learning and optimization using the clonal selection principle,” IEEE Transactions on Evolutionary Computation, vol. 6, no. 3, pp. 239–251, 2002.
37. V. Pareto, Cours D’Economie Politique, volume I and II, F. Rouge, Lausanne, 1896.
38. G. Rudolph, “On a multi-objective evolutionary algorithm and its convergence to the pareto set,” in Proceedings of the 5th IEEE Conference on Evolutionary Computation, IEEE Press: Piscataway, New Jersey, 1998, pp. 511–516.
39. J. D. Schaffer, Multiple Objective Optimization with Vector Evaluated Genetic Algorithms, PhD thesis, Vanderbilt University, 1984.
40. J. D. Schaffer, “Multiple objective optimization with vector evaluated genetic algorithms,” in Genetic Algorithms and their Applications: Proceedings of the First International Conference on Genetic Algorithms, Lawrence Erlbaum: Hillsdale, New Jersey, 1985, pp. 93–100.
41. J. R. Schott, “Fault tolerant design using single and multicriteria genetic algorithm optimization,” Master’s thesis, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, MA, May 1995.
42. R. E. Smith, S. Forrest, and A. S. Perelson, “Searching for diverse, cooperative populations with genetic algorithms,” Technical Report TCGA No. 92002, University of Alabama, Tuscaloosa, AL, 1992.
43. R. E. Smith, S. Forrest, and A. S. Perelson, “Population diversity in an immune system model: Implications for genetic search,” in Foundations of Genetic Algorithms, L. D. Whitley (Ed.), Morgan Kaufmann Publishers: San Mateo, CA, 1993, vol. 2, pp. 153–165.
44. N. Srinivas and K. Deb, “Multiobjective optimization using nondominated sorting in genetic algorithms,” Evolutionary Computation, vol. 2, no. 3, pp. 221–248, Fall 1994.
45. W. Stadler, “Fundamentals of multicriteria optimization,” in Multicriteria Optimization in Engineering and the Sciences, W. Stadler (Ed.), Plenum Press: New York, 1988, pp. 1–25.
46. D. A. Van Veldhuizen, Multiobjective Evolutionary Algorithms: Classifications, Analyses, and New Innovations. PhD thesis, Department of Electrical and Computer Engineering. Graduate School of Engineering. Air Force Institute of Technology, Wright-Patterson AFB, OH, May 1999.
47. D. A. Van Veldhuizen and G. B. Lamont, “Multiobjective evolutionary algorithm research: A history and analysis,” Technical Report TR-98-03, Department of Electrical and Computer Engineering, Graduate School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH, 1998.
48. D. A. Van Veldhuizen and G. B. Lamont, “MOEA test suite generation, design & use,” in Proceedings of the 1999 Genetic and Evolutionary Computation Conference. Workshop Program, A. S. Wu (Ed.), Orlando, FL, July 1999, pp. 113–114.
49. D. A. Van Veldhuizen and G. B. Lamont, “On measuring multiobjective evolutionary algorithm performance,” in 2000 Congress on Evolutionary Computation, IEEE Service Center: Piscataway, New Jersey, July 2000, vol. 1, pp. 204–211.
50. R. Viennet, C. Fontiex, and I. Marc, “New multicriteria optimization method based on the use of a diploid genetic algorithm: Example of an industrial problem,” in Proceedings of Artificial Evolution (European Conference, selected papers), J. M. Alliot, E. Lutton, E. Ronald, M. Schoenauer, and D. Snyers (Eds.), Springer-Verlag: Brest, France, Sept. 1995, pp. 120–127.
51. J. Yoo and P. Hajela, “Immune network simulations in multicriterion design,” Structural Optimization, vol. 18, pp. 85–94, 1999.
52. E. Zitzler, K. Deb, and L. Thiele, “Comparison of multiobjective evolutionary algorithms: Empirical results,” Evolutionary Computation, vol. 8, no. 2, pp. 173–195, 2000.