Solving Multiobjective Optimization Problems using an Artificial Immune System

Carlos A. Coello Coello and Nareli Cruz Cortés
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
ccoello@cs.cinvestav.mx,
nareli@computacion.cs.cinvestav.mx

Abstract. In this paper, we propose an algorithm to solve multiobjective optimization problems (either constrained or unconstrained) using the clonal selection principle. Our approach is compared with respect to 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 than humans possess. In recent years, several researchers have developed computational models of the immune system that attempt to capture some of their most remarkable features such as its self-organizing capability (Hunt and Cooke, 1995; Forrest and Hofmeyr, 2000).

From the information processing perspective, the immune system can be seen as a parallel and distributed adaptive system (Frank, 1996; Dasgupta, 1999). It is capable of learning, it uses memory and is able 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 (Dasgupta, 1999). All these features of the immune system provide, in consequence, great robustness, fault tolerance, dynamism and adaptability (Forrest and Hofmeyr, 2000). 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 amount of operational researchers has produced an important number of techniques to deal with multiobjective optimization problems (Miettinen, 1998). However, it was until relatively recently that researchers realized of the potential of evolutionary algorithms (EAs) and other population-based heuristics in this area.

The first implementation of a multi-objective evolutionary algorithm (MOEA) dates back to the mid-1980s (Schaffer, 1984; Schaffer, 1985). Since then, a considerable amount of research has been done in this area, now known as evolutionary multi-objective 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 (Coello Coello et al., 2002).[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 (Miettinen, 1998). 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 (Coello Coello, 1999).

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

[1] The first author maintains an EMO repository which currently contains over 1000 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/`

the standard methodology adopted in the EMO community (Coello Coello et al., 2002).

2. Basic Definitions

Definition 1 (Global Minimum): *Given a function* $f : \Omega \subseteq \mathcal{R}^n \rightarrow \mathcal{R}$*,* $\Omega \neq \emptyset$*, 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

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

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 (1881) and later generalized by Vilfredo Pareto (1896). Although some authors call *Edgeworth-Pareto optimum* to this notion (see for example (Stadler, 1988)), 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 no feasible 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:*

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

□

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 (Horn, 1997); their corresponding vectors are termed *nondominated*.

3. The Immune System

The main goal of the immune system is to protect the human body from the attack of foreign (harmful) organisms. The immune system is capable of distinguishing between the normal components of our organism and the foreign material that can cause us harm (e.g., bacteria). These foreign organisms are called *antigens*.

The molecules called *antibodies* play the main role on the immune system response. The immune response is specific to a certain foreign organism (antigen). When an antigen is detected, those antibodies that best recognize an antigen will proliferate by cloning. This proccess is called *clonal selection principle* (Nunes de Castro and Von Zuben, 1999).

The new cloned cells undergo high rate mutations or *hypermutation* in order to increase their receptor population (called repertoire). These mutations experienced by the clones are proportional to their affinity to the antigen.

The highest affinity antibodies experiment the lowest mutation rates, whereas the lowest affinity antibodies have high mutation rates. After this mutation process ends, some clones could be dangerous for the body and should therefore be eliminated.

After these clonation and hypermutation processes finish, the immune system has improved the antibodies' affinity, which results on the antigen neutralization and elimination.

At this point, the immune system must return to its normal conditions, eliminating the excedent cells. However, some cells remain circulating throughout the body as memory cells. When the immune system is later attacked by 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 𝒜 from PA
             (PA = Population of Antigens)
       2. Take (randomly) R antibodies from PS
             (PS = Population of Antibodies)
       3. For each antibody r ∈ R, match it against
             the selected antigen 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 (Forrest and Perelson, 1991; Smith et al., 1992; Smith et al., 1993). 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) (Smith et al., 1993). 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. (1993) 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 (Deb and Goldberg, 1989), 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 (Hajela and Lee, 1995; Hajela and Lee, 1996) and has also been hybridized with a multi-objective evolutionary algorithm (Kurpati and Azarm, 2000; Cui et al., 2001). 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 (1999). 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 (Hajela and Lee, 1996). The algorithm is the following

1. Select randomly a single antigen from the antigens population.

2. From the population of antibodies, take a sample (randomly selected) without replacement (Yoo and Hajela (1999) 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. (2002) used both lexicographic ordering and Pareto-based selection in an evolutionary programming algorithm used to detect attacks with an artificial immune system for virus and computer intrusion detection. 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.

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 is based on the *clonal selection principle*, modeling the fact that only the highest affinity antibodies to the antigens will proliferate. Our algorithm uses the concept of Pareto dominance to generate nondominated vectors. Also, an external (or secondary) memory is used to store nondominated vectors found along the evolutionary process, in order to move towards the true Pareto front over time (this can be seen as a form of elitism in evolutionary multiobjective optimization (Coello Coello et al., 2002)). Note that despite the fact that the algorithm presented next is based on our proposal reported in (Coello Coello and Cruz Cortés, 2002), several aspects of such algorithm have been modified, including the elimination of certain parameters required in our previous version.

5.1. The Algorithm

Our algorithm is the following:

1. The initial population is created by dividing decision variable space into a certain number of segments with respect to the desired population size. Thus, we generate an initial population with a uniform distribution of solutions such that every segment in which the decision variable space is divided has solutions. This is done to improve the search capabilities of our algorithm instead of just relying on the use of a mutation operator. Note however, that the solutions generated for the initial population are still random, since we are only contraining their boundaries to make sure that their distribution is uniform along the available range of the decision variables.

2. Initialize the secondary memory 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.

4. Determine which are the best antibodies, since we will clone them adopting the following criterion:

   – If the problem is unconstrained, then all the nondominated individuals are cloned.

   – If the problem is constrained, then we have two further cases: a) there are feasible individuals in the population, and b) there are no feasible individuals in the population. For case b), all the nondominated individuals are cloned. For case a), only the nondominated individuals that are feasible are cloned (nondominance is measured only with respect to other feasible individuals in this case).

5. Copy the best antibodies (obtained from the previous step) into the secondary memory.

6. We determine for each of the “best” antobodies the number of clones that we wish to create. This is done such that the total number of clones created is equal to the 60% of the total population used. When the secondary memory is full, then we do the following:

   – If the individual to be inserted into the secondary memory is not allowed access either because it was repeated or because it belongs to the most crowded region of objective function space, then the number of clones created is zero.

- 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 memory), then the number of clones to be generated is duplicated.
- 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), then the number of clones to be generated is reduced by half.

7. We perform the clonation of the best antibodies based on the information from the previous step.

8. A mutation operator is applied to the clones in such a way that the number of mutated genes in each chromosomic string is equal to the number of decision variables of the problem. This is done to make sure that at least one mutation occurs per string, since otherwise we would have duplicates (the original and the cloned string would be exactly the same)

9. We apply a non-uniform mutation operator to the “best” antibodies found without guaranteeing that duplicates will not be generated (unlike the mutation operator of the previous step). The initial mutation rate adopted is high and it is decreased over time (from 0.9 to 0.3).

10. If the secondary memory is full, we apply crossover to a fraction of its contents. The new individuals generated that are nondominated with respect to the secondary memory will then be added to it.

11. We repeat this process from step 3 during a certain (predetermined) number of times.

Note that in the previous algorithm there is no distinction between antigen and antibody as in some of our previous work (e.g., (Coello Coello and Cruz Cortés, 2002)). In contrast, in this case all the individuals are considered as antibodies, and we only distinguish between “better” antibodies and “not so good” antibodies.

The reason for using an initial population with a uniform distribution of solutions over the allowable range of the decision variables is to sample the search space uniformly. This helps the mutation operator to explore the search space more efficiently.

We apply crossover to the individuals in the secondary memory once this is full so that we can reach intermediate points between them. Such information is used to improve the performance of our algorithm.

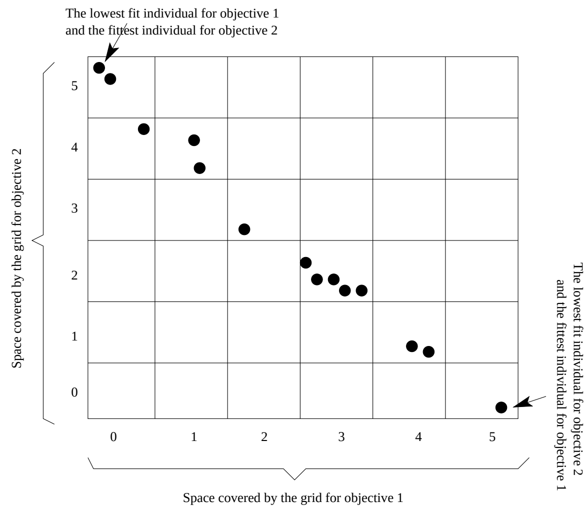

Figure 2. An adaptive grid to hadle the secondary memory

5.2. SECONDARY MEMORY

We use a secondary or external memory as an elitist mechanism in order to maintain the best solutions found along the process. The individuals stored in this memory 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 memory. Therefore, the external memory 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 use the adaptive grid proposed by Knowles and Corne (2000) (see Figure 2).

Ideally, the size of the external memory 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 memory. By enforcing this limit, our external memory 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 memory: 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 memory, determine the cell to which it belongs.

3. If the external memory is full, then determine which is the most crowded cell.

4. To determine if a certain antigen is allowed to enter the external memory, 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 antigen.

6. Experiments

In order to validate our approach, we used several test functions reported in the standard evolutionary multiobjective optimization literature (Deb, 1999; Van Veldhuizen and Lamont, 1999; Coello Coello et al., 2002). 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, which we call the multiobjective immune system 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 memory = 100 (we have eliminated several parameters adopted in our previous work (Coello Coello and Cruz Cortés, 2002)). 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 (Coello Coello and Cruz Cortés, 2002).

MISA was compared against the micro-genetic algorithm for multiobjective optimization (Coello Coello and Toscano Pulido, 2001), against the NSGA-II (Deb et al., 2000; Deb et al., 2002) and against PAES (Knowles and Corne, 2000). 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.

To allow a fair comparison, all the approaches performed the same number of fitness function evaluations as MISA and they all adopted the same size for their external memories.

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/vars, where vars = 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 (for the microGA) 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 compare our results:

– **Two Set Coverage (SC)**: This metric was proposed in (Zitzler et al., 2000), and it can be termed *relative coverage comparison of two sets*. Consider $X', X'' \subseteq X'$ as two sets of phenotype decision vectors. SC is defined as the mapping of the order pair (X', X'') to the interval $[0, 1]$.

$$SC(X', X'') \triangleq \frac{|\{a'' \epsilon X''; \exists a' \epsilon X' : a' \succeq a''\}|}{|X''|} \tag{9}$$

If all points in X' dominate or are equal to all points in X'', then by definition $SC = 1$. $SC = 0$ implies the opposite. In general, $SC(X', X'')$ and $SC(X'', X')$ both have to be considered due to set intersections not being empty. Of course, this metric can be used for both spaces (objective function or decision variable space), but in this case we applied it in objective function space. The advantage of this metric is that it is easy to calculate and provides a relative comparison based upon dominance numbers between generations or algorithms.

– **Spacing (S)**: This metric was proposed by Schott (1995) as a way of measuring the range (distance) variance of neighboring vectors in the Pareto front known. This metric is defined as:

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

where $d_i = \min_j(| f_1^i(\vec{x}) - f_1^j(\vec{x}) | + | f_2^i(\vec{x}) - f_2^j(\vec{x}) |), i, j = 1, \ldots, n$, $\overline{d}$ 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.

– **Generational Distance (GD)**: The concept of generational distance was introduced by Van Veldhuizen & Lamont (Van Veldhuizen and Lamont, 1998; Van Veldhuizen and Lamont, 2000) 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:

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

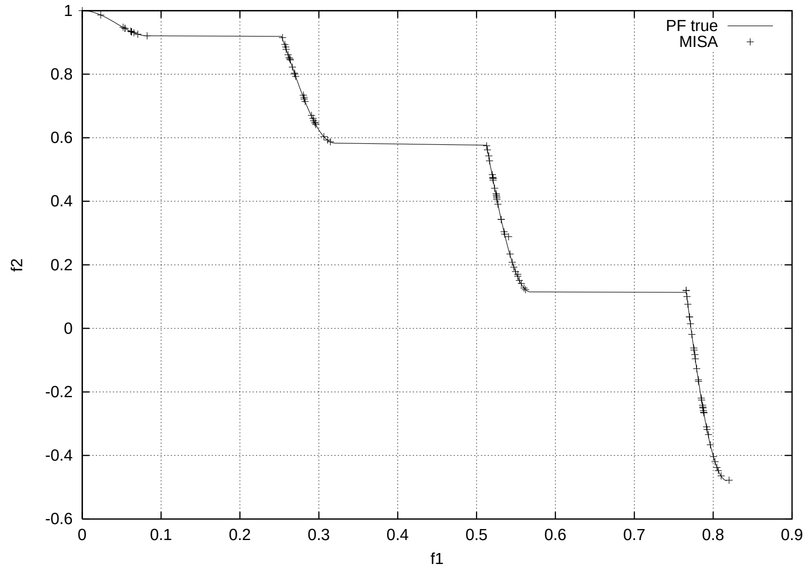

Figure 3. Pareto front obtained by MISA in the first example. The true Pareto front is shown as a continuous line (note that the horizontal segments are NOT part of the Pareto front and are shown only to facilitate drawing the front) and the Pareto front found by MISA is shown as crosses.

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 (Rudolph, 1998), Schott (1995), and Zitzler et al. (Zitzler et al., 2000).

EXAMPLE 1

Minimize: $F = (f_1(x,y), f_2(x,y))$, where

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

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

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

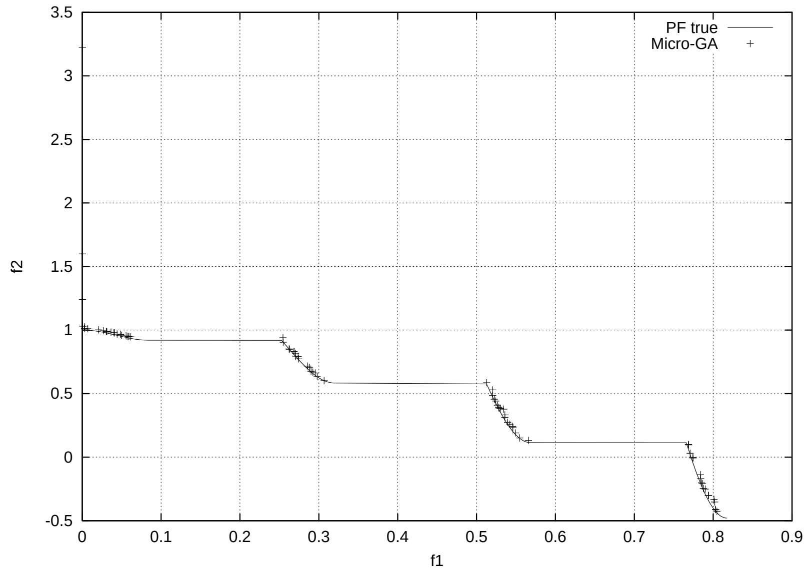

Figure 4. Pareto front obtained by the microGA in the first example.

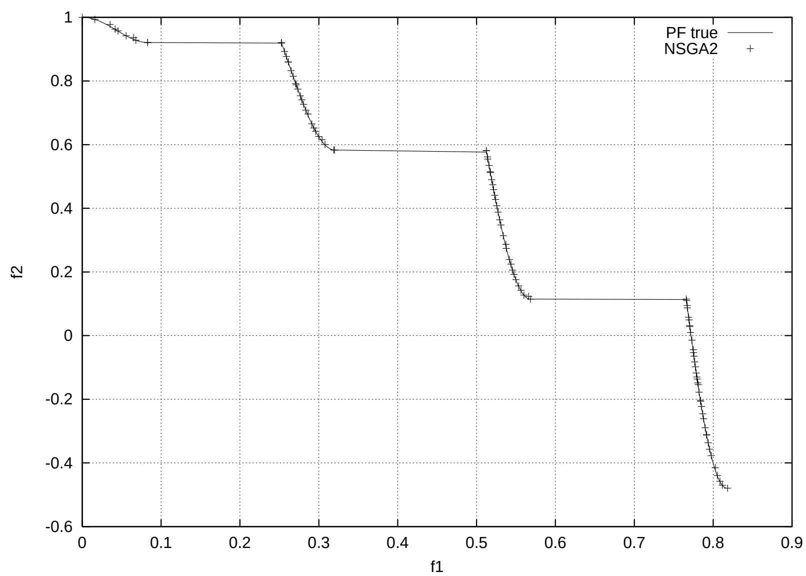

Figure 5. Pareto front obtained by the NSGA-II in the first example.

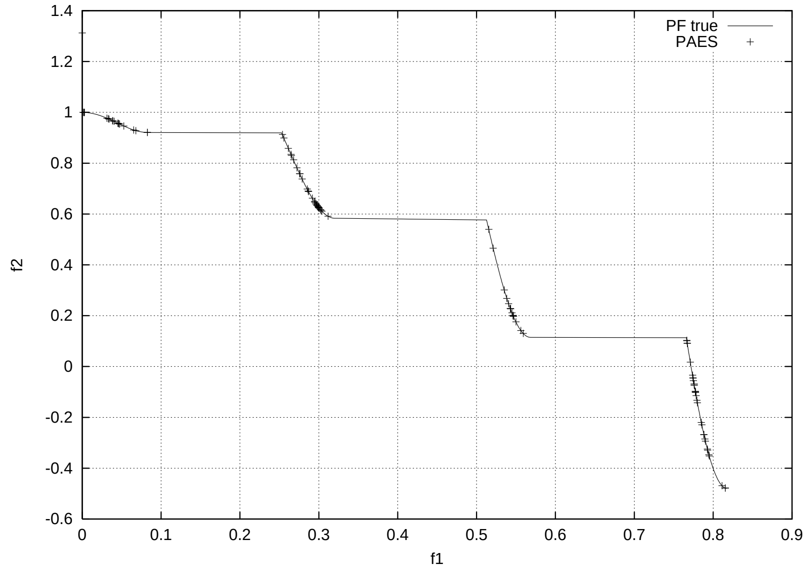

Figure 6. Pareto front obtained by PAES in the first example.

Table I. Spacing for example 1

	MISA	MicroGA	NSGA-II	PAES
Average	0.008296750	0.186035000	0.007596200	0.032293900
Best	0.007123000	0.035300000	0.006569000	0.012456000
Worst	0.009568000	0.507064000	0.009198000	0.251791000
Std. Dev.	0.000690148	0.144612383	0.000646588	0.052402532

PAES are shown in Figures 3, 4, 5 and 6, respectively. Note that the Pareto front is disconnected (it consists of four Pareto curves). The graphs previously indicated show the average behavior of each algorithm with respect to the generational distance metric.

The values of the three metrics for each algorithm are presented in Tables I, II, and III, respectively. In this case, the four algorithms compared have a similar average value for the generational distance, PAES being the best (MISA placed third with respect to this metric). Note, however that the differences are minor and all the algorithms performed well with respect to this metric.

Regarding coverage, MISA performed better with respect to the microGA and with respect to PAES. With respect to the NSGA-II, MISA showed a very similar behavior regarding coverage.

Table II. Coverage for example 1

	MISA & μGA	μGA & MISA	MISA & NSGA-II
Average	0.6865	0.3790	0.3565
Lowest	0.2700	0.2500	0.3000
Highest	1.0000	0.5800	0.4200
Std. Dev.	0.3557	0.0932	0.0338
	NSGA-II & MISA	MISA & PAES	PAES & MISA
Average	0.3280	0.3265	0.2490
Lowest	0.3000	0.2800	0.1400
Highest	0.3700	0.3900	0.4400
Std. Dev.	0.0221	0.0327	0.0944

Table III. Generational Distance for example 1

	MISA	MicroGA	NSGA-II	PAES
Average	0.00028705	0.0003192	0.0002532	0.000178
Best	0.00023	0.000229	0.000233	0.000117
Worst	0.000376	0.000655	0.000275	0.000237
Std. Dev.	0.000033	0.0001450	0.0000109	0.0000329

Regarding spacing, MISA and the NSGA-II both found the best results with very similar values. The microGA and PAES ranked second and third, respectively.

Since metrics can sometimes be misleading in multiobjective optimization, it is always important to rely on graphical comparisons (whenever possible). In this case, we can see that the NSGA-II and MISA show a similar behavior regarding closeness to the true Pareto front and spread. In contrast, the microGA produces a few solutions that do not belong to the true Pareto front of the problem and that the selection mechanism of the algorithm could not eliminate. PAES does not have an appropriate spread of solutions and also has solutions that do not belong to the true Pareto front.

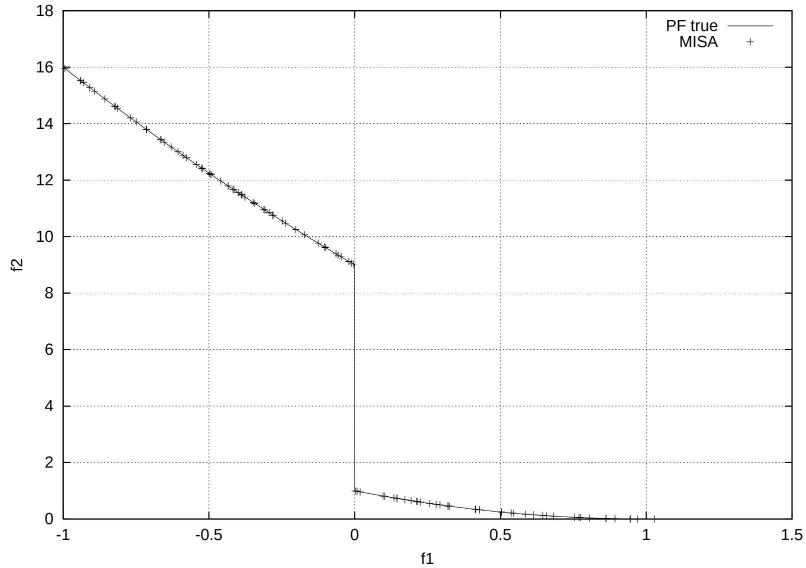

Figure 7. Pareto front obtained by MISA in the second example. The true Pareto front of the problem is shown as a continuous line (note that the vertical segment is NOT part of the Pareto front and is shown only to facilitate drawing the front) and the Pareto front found by MISA is shown as crosses.

Then, in conclusion, for the first example, we can say that MISA and the NSGA-II tied in the first place regarding best overall performance, with PAES and the microGA in second and third place, respectively.

EXAMPLE 2

Our second example is a two-objective optimization problem proposed by Schaffer (1984) that has been used by several researchers (Srinivas and Deb, 1994):

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

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

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 7, 8, 9 and 10, respectively. The values of the three metrics for each algorithm are presented in Tables IV, V, and VI, respectively.

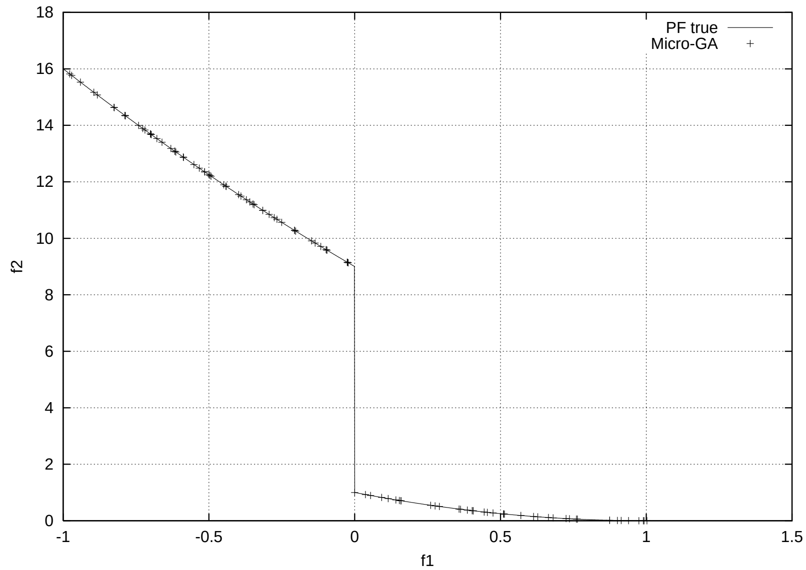

Figure 8. Pareto front obtained by the microGA in the second example.

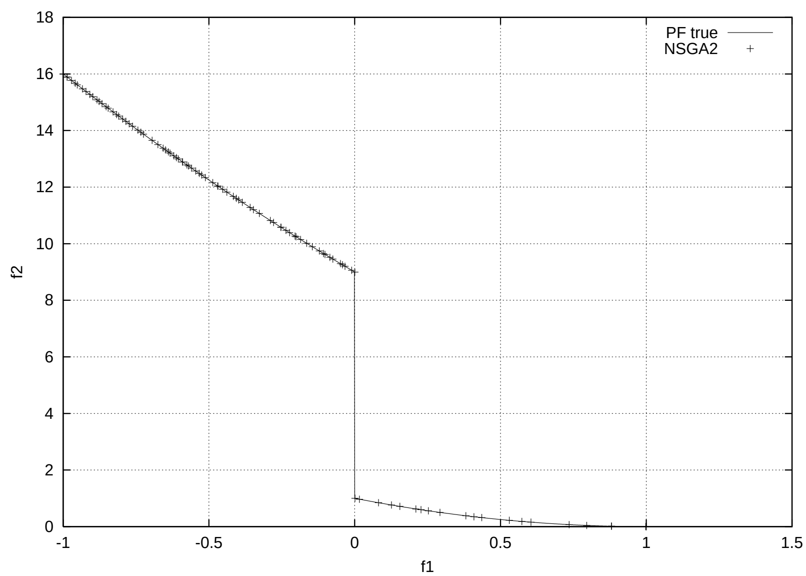

Figure 9. Pareto front obtained by the NSGA-II in the second example.

Table IV. Spacing for example 2

	MISA	MicroGA	NSGA-II	PAES
Average	0.227902850	0.065564200	0.146915350	0.243692100
Best	0.205998000	0.050730000	0.044937000	0.010033000
Worst	0.248307000	0.160646000	0.207402000	1.599812000
Std. Dev.	0.013566168	0.024662209	0.065184527	0.346439517

Table V. Coverage for example 2

	MISA & μGA	μGA & MISA	MISA & NSGA-II
Average	0.5965	0.5915	0.7175
Lowest	0.5700	0.5800	0.5800
Highest	0.6300	0.6100	1.0000
Std. Dev.	0.0232	0.0123	0.1904
	NSGA-II & MISA	MISA & PAES	PAES & MISA
Average	0.7900	0.5980	0.3565
Lowest	0.7500	0.5700	0.0000
Highest	0.8200	0.6300	0.7900
Std. Dev.	0.0238	0.0228	0.2740

Table VI. Generational Distance for example 2

	MISA	MicroGA	NSGA-II	PAES
Average	0.00023	0.00024	0.00028	0.00018
Best	0.00020	0.00021	0.00025	0.00005
Worst	0.00026	0.00026	0.00030	0.00027
Std. Dev.	0.00002	0.00003	0.00001	0.00008

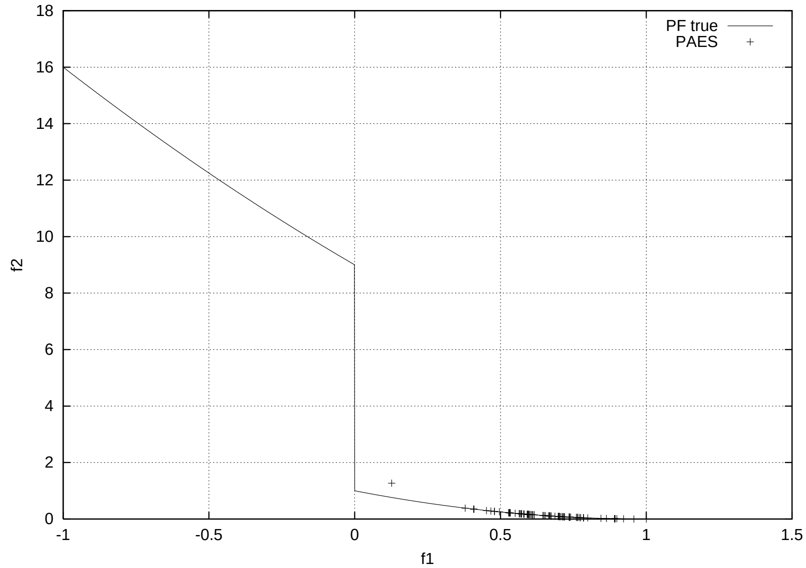

Figure 10. Pareto front obtained by PAES in the second example.

In this case, PAES had the best performance with respect to the generational distance metric, while MISA, the NSGA-II and the microGA all had a slightly poorer performance than PAES. However, all the algorithms compared are very close to the ideal behavior with respect to this metric.

Regarding coverage, MISA is in a tie both with respect to the microGA and with respect to the NSGA-II. With respect to PAES, MISA has a better performance (more of its solutions dominate those generated by PAES).

With respect to spacing, the microGA showed the best performance, followed by the NSGA-II. PAES and MISA ranked last with respect to this metric with very similar values between them. However, graphically, we can see that MISA, the microGA and the NSGA-II all provided a reasonable approximation of the Pareto front, with a very good distribution of points along it. In contrast, PAES could not generate any of the two extrema of the Pareto front.

In general, we can say that for this example PAES showed the worst average performance and that MISA, the microGA and the NSGA-II they all showed a good average performance. It is also worth noting that the microGA provided the most uniform distribution of solutions along the Pareto front.

EXAMPLE 3

The third example is the three-objective function problem proposed by Viennet (1995):

Minimize: $F = (f_1(x, y), f_2(x, y), f_3(x, y))$

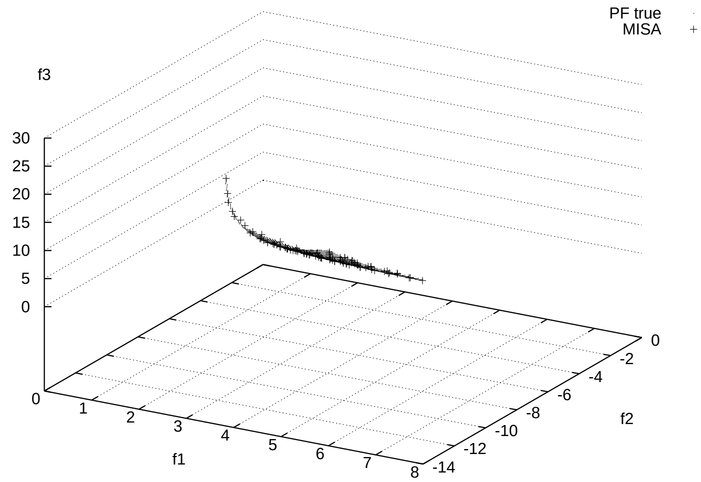

Figure 11. Pareto front produced by MISA in the third test function. The true Pareto front of the problem is shown as dots and the Pareto front found by MISA is shown as crosses.

where:

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

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 11, 12, 13 and 14, respectively. The values of the three metrics for each algorithm are presented in Tables VII, VIII, and IX, respectively.

In this example, both PAES and MISA had the best performance regarding generational distance, followed by the NSGA-II. The microGA had the poorest performance with respect to this metric.

Regarding coverage, MISA outperformed the microGA, but it was outperformed both by PAES and by the NSGA-II.

With respect to spacing, the microGA had the best performance, followed by the NSGA-II. PAES and MISA ranked last with respect to this metric, both obtaining very similar values. Note however, that due to the poor generational

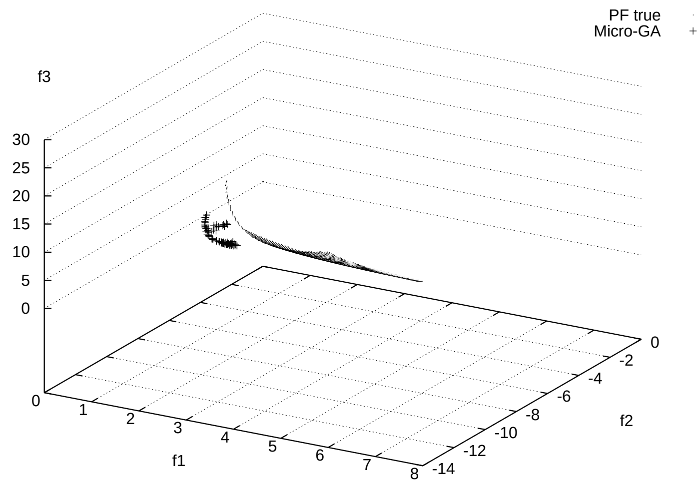

Figure 12. Pareto front obtained by the microGA in the third example.

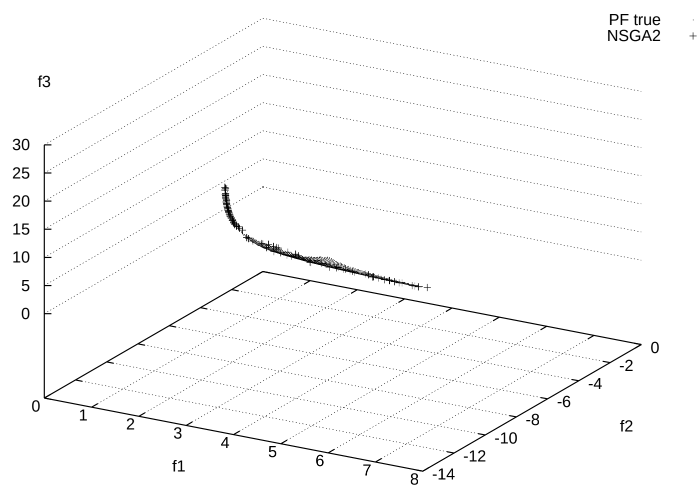

Figure 13. Pareto front obtained by the NSGA-II in the third example.

Table VII. Spacing for example 3

	MISA	MicroGA	NSGA-II	PAES
Average	0.472922000	0.292722474	0.372761632	0.482349632
Best	0.392846000	0.274215000	0.300136000	0.318394000
Worst	0.530847000	0.306928000	0.499717000	0.656009000
Std. Dev.	0.037310539	0.007727605	0.083564152	0.064430704

Table VIII. Coverage for example 3

	MISA & μGA	μGA & MISA	MISA & NSGA-II
Average	0.7165	0.3895	0.4240
Lowest	0.5900	0.0500	0.2300
Highest	0.8300	0.8200	0.7200
Std. Dev.	0.0601	0.2651	0.1334
	NSGA-II & MISA	MISA & PAES	PAES & MISA
Average	0.7265	0.6975	0.8390
Lowest	0.3700	0.5200	0.5500
Highest	0.9800	0.8800	0.9500
Std. Dev.	0.2050	0.1103	0.1089

Table IX. Generational Distance for example 3

	MISA	MicroGA	NSGA-II	PAES
Average	0.00328	0.11266	0.08282	0.00279
Best	0.00272	0.11089	0.06733	0.00256
Worst	0.00452	0.11494	0.10106	0.00313
Std. Dev.	0.00045	0.00116	0.00853	0.00021

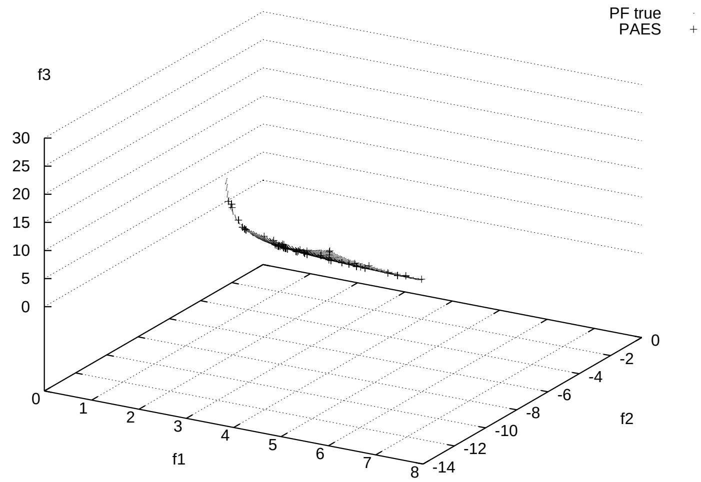

Figure 14. Pareto front obtained by PAES in the third example.

distance of the microGA, its good distribution of solutions is pointless, since many of these points do not lie on the true Pareto front of the problem.

Graphically, we can see that the NSGA-II, MISA and PAES produce a reasonably good approximation of the true Pareto front, and only the microGA shows a poor performance. In general, we can conclude that both the NSGA-II and MISA had a good average behavior on this problem (both in terms of closeness to the true Pareto front and in terms of spread of solutions), although the values of the different metrics do not indicate a clear overall winner.

EXAMPLE 4

The fourth example was proposed by Kita (1996):

Maximize $F = (f_1(x, y), f_2(x, y))$

where:

$$\begin{aligned} f_1(x, y) &= -x^2 + y, \\ f_2(x, y) &= \frac{1}{2}x + y + 1 \end{aligned}$$

$x, y \geq 0$, $0 \geq \frac{1}{6}x + y - \frac{13}{2}$, $0 \geq \frac{1}{2}x + y - \frac{15}{2}$, $0 \geq 5x + y - 30$.

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 15, 16, 17 and 18, respectively. The values of

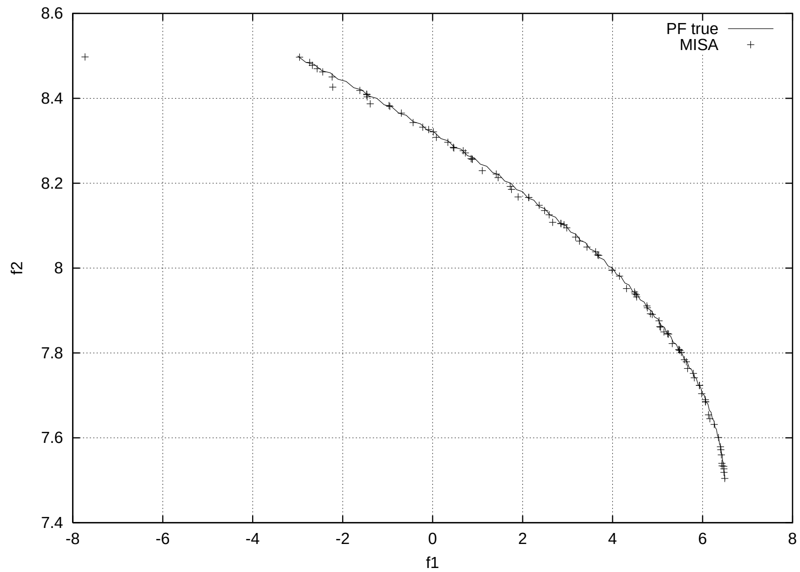

Figure 15. Pareto front obtained by MISA in the fourth test function. The true Pareto front of the problem is shown as a continuous line and the Pareto front found by MISA is shown as crosses.

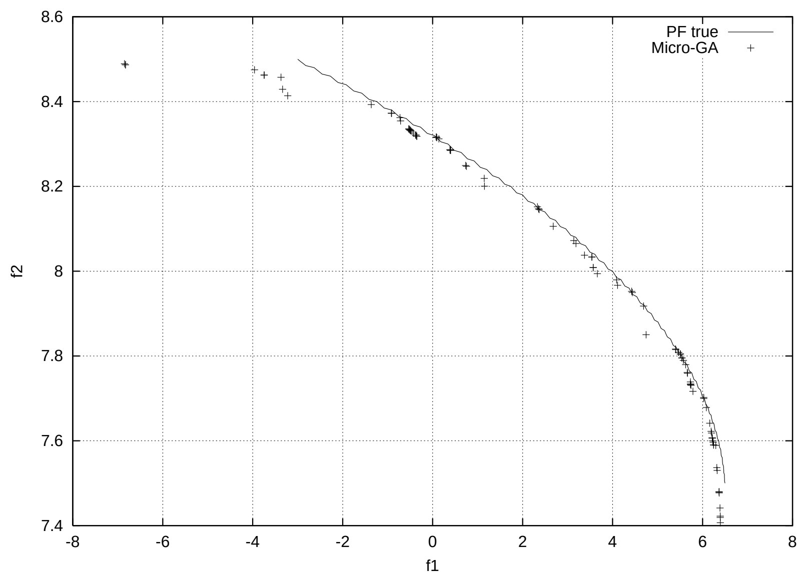

Figure 16. Pareto front obtained by the microGA in the fourth example.

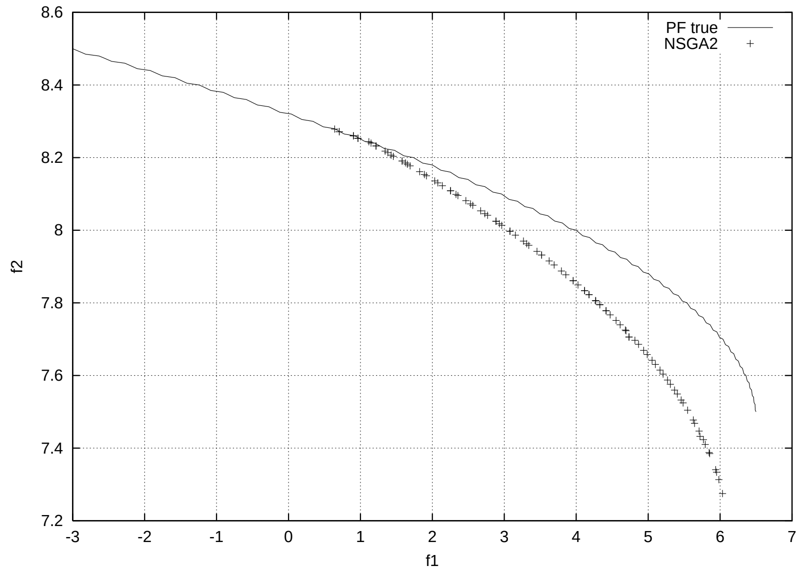

Figure 17. Pareto front obtained by the NSGA-II in the fourth example.

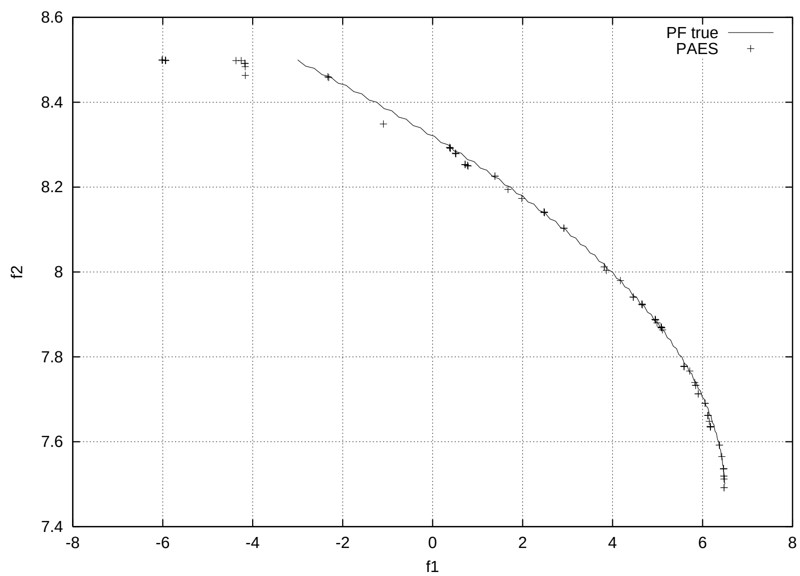

Figure 18. Pareto front obtained by PAES in the fourth example.

Table X. Spacing for example 4

	MISA	MicroGA	NSGA-II	PAES
Average	0.381157200	0.1720526	0.13789	0.121572950
Best	0.052625000	0.0575230	0.02157	0.041657000
Worst	1.310158000	1.1874970	1.48835	0.220725000
Std. Dev.	0.470287797	0.2413053	0.32169	0.049984296

Table XI. Coverage for example 4

	MISA & μGA	μGA & MISA	MISA & NSGA-II
Average	0.9425	0.7150	0.7950
Lowest	0.5800	0.3600	0.5200
Highest	1.0000	1.0000	1.0000
Std. Dev.	0.1168	0.2212	0.1612
	NSGA-II & MISA	MISA & PAES	PAES & MISA
Average	0.7275	0.4025	0.7175
Lowest	0.0600	0.2600	0.1900
Highest	1.0000	0.5000	1.0000
Std. Dev.	0.2879	0.0801	0.2531

the three metrics for each algorithm are presented in Tables X, XI, and XII, respectively.

In this case, MISA had the best performance regarding generational distance, followed by PAES and the microGA. The NSGA-II had the worst performance with respect to this metric.

Regarding coverage, MISA outperformed the microGA, and it got a tie with respect to the NSGA-II. PAES, however, outperformed MISA with respect to this metric.

With respect to spacing, PAES and the NSGA-II had the best performance, followed by the microGA. MISA had the worst performance regarding this metric.

Despite the values of the metrics, the graphical representation of the results clearly indicates that MISA produced the best results. The NSGA-II

Table XII. Generational Distance for example 4

	MISA	MicroGA	NSGA-II	PAES
Average	0.00394	0.00657	0.08349	0.00482
Best	0.00248	0.00101	0.00385	0.00201
Worst	0.00473	0.01405	0.67845	0.0728
Std. Dev.	0.00066	0.00369	0.16440	0.00132

missed about half of the true Pareto front and both PAES and the microGA were unable to generate large portions of the Pareto front as well.

EXAMPLE 5

Our fifth example is a two-objective optimization problem defined by Kursawe (1991):

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

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

where:

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

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 19, 20, 21 and 22, respectively. The values of the three metrics for each algorithm are presented in Tables XIII, XIV, and XV, respectively.

In this case, MISA had again the best performance with respect to the generational distance metric, followed by the NSGA-II, PAES and the microGA, respectively. Note however that the differences among the four approaches are minor.

Regarding coverage, MISA outperformed the microGA and the NSGA-II. However, PAES outperformed MISA with respect to this metric.

With respect to spacing, PAES showed the best performance, followed by MISA. The microGA and the NSGA-II ranked last with respect to this metric.

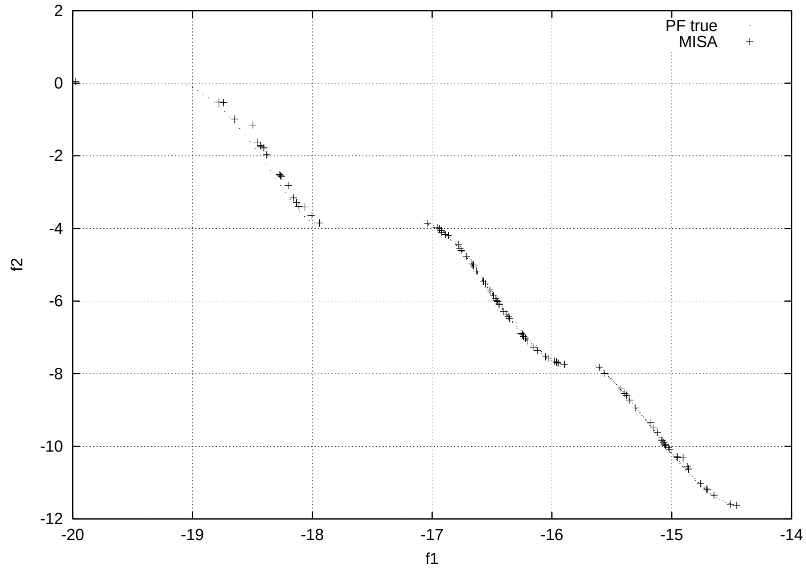

Figure 19. Pareto front produced by MISA in the fifth test function. The true Pareto front of the problem is shown as a continuous line (note that the horizontal segment is NOT part of the Pareto front and is shown only to facilitate drawing the front) and the Pareto front found by MISA is shown as crosses.

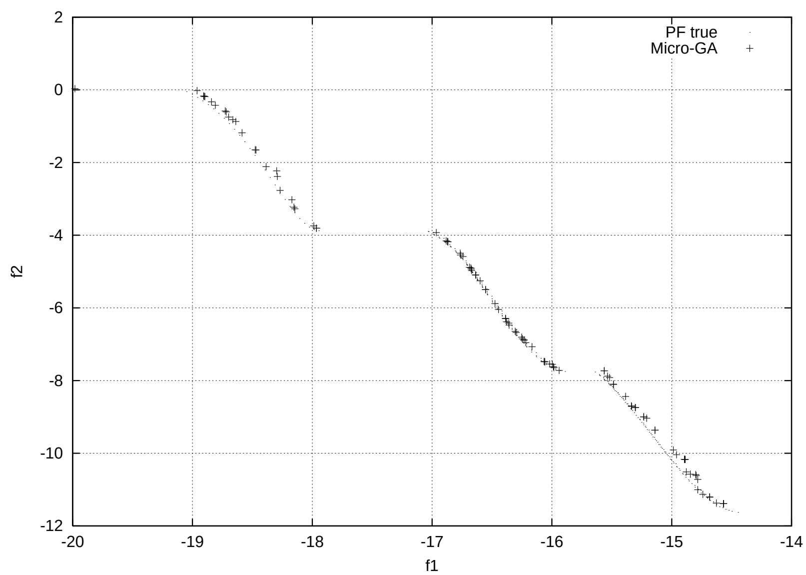

Figure 20. Pareto front obtained by the microGA in the fifth example.

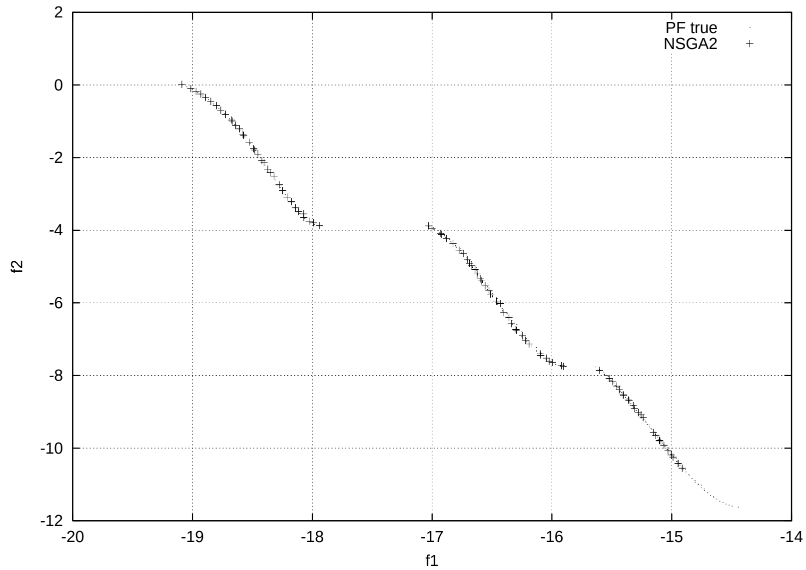

Figure 21. Pareto front obtained by the NSGA-II in the fifth example.

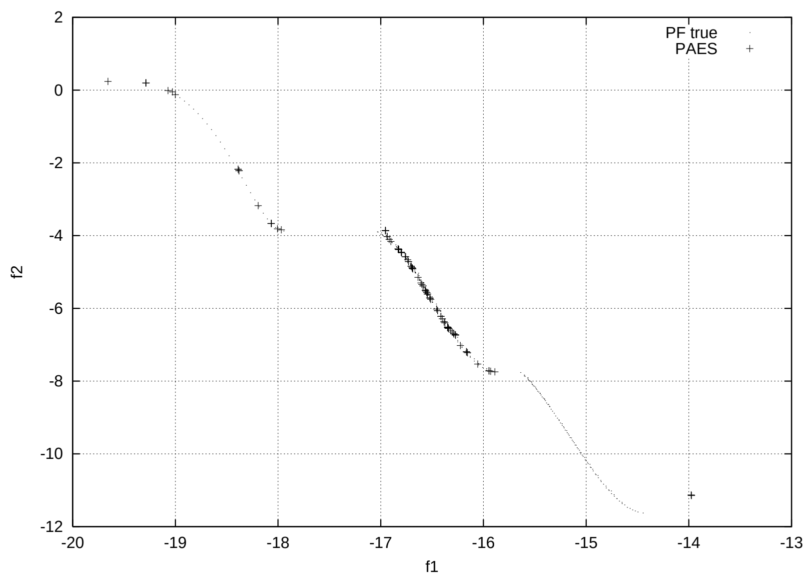

Figure 22. Pareto front obtained by PAES in the fifth example.

Table XIII. Spacing for example 5

	MISA	MicroGA	NSGA-II	PAES
Average	2.90986	3.232291	3.339190050	0.1953046
Best	2.8839250	2.9090340	3.019572000	0.0655610
Worst	3.200155000	3.784747000	3.728558000	0.491542
Std. Dev.	0.069326	0.3313995	0.250941502	0.109211

Table XIV. Coverage for example 5

	MISA & μGA	μGA & MISA	MISA & NSGA-II
Average	0.5605	0.2185	0.5790
Lowest	0.3200	0.0500	0.2200
Highest	0.9800	1.0000	0.8800
Std. Dev.	0.2313	0.1968	0.2407
	NSGA-II & MISA	MISA & PAES	PAES & MISA
Average	0.3925	0.1395	0.3580
Lowest	0.2600	0.0000	0.0700
Highest	1.0000	0.7400	1.0000
Std. Dev.	0.1532	0.1912	0.2425

Table XV. Generational Distance for example 5

	MISA	MicroGA	NSGA-II	PAES
Average	0.00358	0.00544	0.00429	0.00506
Best	0.00276	0.00440	0.00372	0.00231
Worst	0.00577	0.00717	0.00508	0.01924
Std. Dev.	0.00077	0.00114	0.00032	0.00399

Again, despite the values of the metrics, the graphical representation of the results shows that both MISA and the NSGA-II provide the best approximations of the Pareto front for this problem. However, in both cases a small segment of the true Pareto front is not generated. The microGA shows a good spread of points, but it misses a segment of considerable size of the true Pareto front. Finally, PAES has a very poor performance, since it only generates well one of the three portions by which the true Pareto front was composed of.

In general, we can see that MISA provides competitive results with respect to the three other algorithms with respect to 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. The approach proposed 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.

Therefore, we conclude that artificial immune systems can be effectively used to solve multiobjective optimization problems in a relatively simple way.[2] 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.

We are also interested in analyzing alternative mechanisms to maintain diversity and to achieve better distributions of solutions along the Pareto front, which is a current weakness of the approach proposed in this paper. We are also considering the possibility of using spatial data structures to store and retrieve nondominated solutions in a more efficient way (Habenicht, 1982).

[2] 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

Acknowledgements

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.

References

Anchor, Kevin P, Jesse B. Zydallis, Gregg H. Gunsch, and Gary B. Lamont. Extending the Computer Defense Immune System: Network Intrusion Detection with a Multiobjective Evolutionary Programming Approach. In Jonathan Timmis and Peter J. Bentley, editors, *First International Conference on Artificial Immune Systems (ICARIS'2002)*, pages 12–21. University of Kent at Canterbury, UK, September 2002. ISBN 1-902671-32-5.

Coello Coello, Carlos A. A Comprehensive Survey of Evolutionary-Based Multiobjective Optimization Techniques. *Knowledge and Information Systems. An International Journal*, 1(3):269–308, August 1999.

Coello Coello, Carlos A and Nareli Cruz Cortés. An Approach to Solve Multiobjective Optimization Problems Based on an Artificial Immune System. In Jonathan Timmis and Peter J. Bentley, editors, *First International Conference on Artificial Immune Systems (ICARIS'2002)*, pages 212–221. University of Kent at Canterbury, UK, September 2002. ISBN 1-902671-32-5.

Coello Coello, Carlos A. and Gregorio Toscano Pulido. Multiobjective Optimization using a Micro-Genetic Algorithm. In Lee Spector, Erik D. Goodman, Annie Wu, W.B. Langdon, Hans-Michael Voigt, Mitsuo Gen, Sandip Sen, Marco Dorigo, Shahram Pezeshk, Max H. Garzon, and Edmund Burke, editors, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'2001)*, pages 274–282, San Francisco, California, 2001. Morgan Kaufmann Publishers.

Coello Coello, Carlos A, David A. Van Veldhuizen, and Gary B. Lamont. *Evolutionary Algorithms for Solving Multi-Objective Problems*. Kluwer Academic Publishers, New York, May 2002. ISBN 0-3064-6762-3.

Cui, Xunxue, Miao Li, and Tingjian 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)*, volume 2, pages 1316–1321, Piscataway, New Jersey, May 2001. IEEE Service Center.

Dasgupta, Dipankar, editor. *Artificial Immune Systems and Their Applications*. Springer-Verlag, Berlin, 1999.

Nunes de Castro, Leandro and Fernando José Von Zuben. Artificial Immune Systems: Part I - Basic Theory and Applications. Technical Report TR-DCA 01/99, FEEC/UNICAMP, Brazil, December 1999.

Deb, Kalyanmoy. Multi-Objective Genetic Algorithms: Problem Difficulties and Construction of Test Problems. *Evolutionary Computation*, 7(3):205–230, Fall 1999.

Deb, Kalyanmoy, Samir Agrawal, Amrit Pratab, and T. Meyarivan. A Fast Elitist Non-Dominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGA-II. In Marc Schoenauer, Kalyanmoy Deb, Günter Rudolph, Xin Yao, Evelyne Lutton, Juan Julian Merelo, and Hans-Paul Schwefel, editors, *Proceedings of the Parallel Problem*

Solving from Nature VI Conference, pages 849–858, Paris, France, 2000. Springer. Lecture Notes in Computer Science No. 1917.

Deb, Kalyanmoy and David E. Goldberg. An Investigation of Niche and Species Formation in Genetic Function Optimization. In J. David Schaffer, editor, *Proceedings of the Third International Conference on Genetic Algorithms*, pages 42–50, San Mateo, California, June 1989. George Mason University, Morgan Kaufmann Publishers.

Deb, Kalyanmoy, Amrit Pratap, Sameer Agarwal, and T. Meyarivan. A Fast and Elitist Multiobjective Genetic Algorithm: NSGA–II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, April 2002.

Edgeworth, Francis Ysidro. *Mathematical Physics*. P. Keagan, London, England, 1881.

Fonseca, Carlos M. and Peter J. Fleming. Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion and Generalization. In Stephanie Forrest, editor, *Proceedings of the Fifth International Conference on Genetic Algorithms*, pages 416–423, San Mateo, California, 1993. Morgan Kauffman Publishers.

Forrest, Stephanie and Steven A. Hofmeyr. Immunology as Information Processing. In L.A. Segel and I. Cohen, editors, *Design Principles for the Immune System and Other Distributed Autonomous Systems*, Santa Fe Institute Studies in the Sciences of Complexity, pages 361–387. Oxford University Press, 2000.

Forrest, Stephanie and Alan S. Perelson. Genetic algorithms and the immune system. In Hans-Paul Schwefel and R. Männer, editors, *Parallel Problem Solving from Nature*, Lecture Notes in Computer Science, pages 320–325. Springer-Verlag, Berlin, Germany, 1991.

Frank, Steven A. *The Design of Natural and Artificial Adaptive Systems*. Academic Press, New York, 1996.

Habenicht, W. Quad trees: A data structure for discrete vector optimization problems. In *Lecture Notes in Economics and Mathematical Systems*, Vol. 209, pages 136–145, 1982.

Hajela, P. and J. Lee. Constrained Genetic Search via Schema Adaptation. An Immune Network Solution. In Niels Olhoff and George I. N. Rozvany, editors, *Proceedings of the First World Congress of Stuctural and Multidisciplinary Optimization*, pages 915–920, Goslar, Germany, 1995. Pergamon.

Hajela, P. and J. Lee. Constrained Genetic Search via Schema Adaptation. An Immune Network Solution. *Structural Optimization*, 12:11–15, 1996.

Horn, Jeffrey. Multicriterion Decision making. In Thomas Bäck, David Fogel, and Zbigniew Michalewicz, editors, *Handbook of Evolutionary Computation*, volume 1, pages F1.9:1–F1.9:15. IOP Publishing Ltd. and Oxford University Press, 1997.

Hunt, John E. and Denise E. Cooke. An adaptative, distributed learning systems based on the immune system. In *Proceedings of the IEEE International Conference on Systems, Man and Cybernatics*, pages 2494–2499, 1995.

Kita, Hajime, Yasuyuki Yabumoto, Naoki Mori, and Yoshikazu Nishikawa. Multi-Objective Optimization by Means of the Thermodynamical Genetic Algorithm. In Hans-Michael Voigt, Werner Ebeling, Ingo Rechenberg, and Hans-Paul Schwefel, editors, *Parallel Problem Solving from Nature—PPSN IV*, Lecture Notes in Computer Science, pages 504–512, Berlin, Germany, September 1996. Springer-Verlag.

Knowles, Joshua D. and David W. Corne. Approximating the Nondominated Front Using the Pareto Archived Evolution Strategy. *Evolutionary Computation*, 8(2):149–172, 2000.

Kurpati, A. and S. Azarm. Immune Network Simulation with Multiobjective Genetic Algorithms for Multidisciplinary Design Optimization. *Engineering Optimization*, 33:245–260, 2000.

Kursawe, Frank. A Variant of Evolution Strategies for Vector Optimization. In H. P. Schwefel and R. Männer, editors, *Parallel Problem Solving from Nature. 1st Workshop, PPSN I*, volume 496 of *Lecture Notes in Computer Science*, pages 193–197, Berlin, Germany, oct 1991. Springer-Verlag.

Miettinen, Kaisa M. *Nonlinear Multiobjective Optimization*. Kluwer Academic Publishers, Boston, Massachusetts, 1998.

Pareto, Vilfredo. *Cours D'Economie Politique*, volume I and II. F. Rouge, Lausanne, 1896.

Rudolph, Günter. On a Multi-Objective Evolutionary Algorithm and Its Convergence to the Pareto Set. In *Proceedings of the 5th IEEE Conference on Evolutionary Computation*, pages 511–516, Piscataway, New Jersey, 1998. IEEE Press.

Schaffer, J. David. *Multiple Objective Optimization with Vector Evaluated Genetic Algorithms*. PhD thesis, Vanderbilt University, 1984.

Schaffer, J. David. Multiple Objective Optimization with Vector Evaluated Genetic Algorithms. In *Genetic Algorithms and their Applications: Proceedings of the First International Conference on Genetic Algorithms*, pages 93–100, Hillsdale, New Jersey, 1985. Lawrence Erlbaum.

Schott, Jason R. Fault Tolerant Design Using Single and Multicriteria Genetic Algorithm Optimization. Master's thesis, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, Massachusetts, May 1995.

Smith, Robert E., Stephanie Forrest, and Alan S. Perelson. Searching for diverse, cooperative populations with genetic algorithms. Technical Report TCGA No. 92002, University of Alabama, Tuscaloosa, Alabama, 1992.

Smith, Robert E., Stephanie Forrest, and Alan S. Perelson. Population Diversity in an Immune System Model: Implications for Genetic Search. In L. Darrell Whitley, editor, *Foundations of Genetic Algorithms 2*, pages 153–165. Morgan Kaufmann Publishers, San Mateo, California, 1993.

Srinivas, N. and Kalyanmoy Deb. Multiobjective Optimization Using Nondominated Sorting in Genetic Algorithms. *Evolutionary Computation*, 2(3):221–248, Fall 1994.

Stadler, W. Fundamentals of multicriteria optimization. In W. Stadler, editor, *Multicriteria Optimization in Engineering and the Sciences*, pages 1–25. Plenum Press, New York, 1988.

Van Veldhuizen, David A. and Gary 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, Ohio, 1998.

Van Veldhuizen, David A. and Gary B. Lamont. MOEA Test Suite Generation, Design & Use. In Annie S. Wu, editor, *Proceedings of the 1999 Genetic and Evolutionary Computation Conference. Workshop Program*, pages 113–114, Orlando, Florida, July 1999.

Van Veldhuizen, David. A. and Gary B. Lamont. On Measuring Multiobjective Evolutionary Algorithm Performance. In *2000 Congress on Evolutionary Computation*, volume 1, pages 204–211, Piscataway, New Jersey, July 2000. IEEE Service Center.

Viennet, Rémy, Christian Fontiex, and Ivan Marc. New Multicriteria Optimization Method Based on the Use of a Diploid Genetic Algorithm: Example of an Industrial Problem. In J. M. Alliot, E. Lutton, E. Ronald, M. Schoenauer, and D. Snyers, editors, *Proceedings of Artificial Evolution (European Conference, selected papers)*, pages 120–127, Brest, France, September 1995. Springer-Verlag.

Yoo, J. and P. Hajela. Immune network simulations in multicriterion design. *Structural Optimization*, 18:85–94, 1999.

Zitzler, Eckart, Kalyanmoy Deb, and Lothar Thiele. Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. *Evolutionary Computation*, 8(2):173–195, Summer 2000.