Multiobjective Optimization using Ideas from the Clonal Selection Principle

Nareli Cruz Cortés and Carlos A. Coello Coello

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
nareli@computacion.cs.cinvestav.mx,
ccoello@cs.cinvestav.mx

Abstract. In this paper, we propose a new multiobjective optimization approach based on the clonal selection principle. Our approach is compared with respect to other evolutionary multiobjective optimization techniques that are representative of the state-of-the-art in the area. In our study, several test functions and metrics commonly adopted in evolutionary multiobjective optimization are used. Our results indicate that the use of an immune system for multiobjective optimization is a viable alternative.

Category: **Artificial Immune Systems**

1 Introduction

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 [13]. However, it was until relatively recently that researchers realized of the potential of evolutionary algorithms (EAs) and other population-based heuristics in this area [7]. 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 [13]. 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 [7, 3]. Despite the considerable amount of research on evolutionary multiobjective optimization 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) [3]. 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 this area [3].

2 Basic Definitions

Definition 1 (Pareto Optimality:): *A point* $\boldsymbol{x}^* \in \Omega$ *(*Ω *is the feasible region) is* **Pareto optimal** *if for every* $\boldsymbol{x} \in \Omega$ *and* $I = \{1, 2, \ldots, k\}$ *either,*

$$\forall_{i \in I}(f_i(\boldsymbol{x}) = f_i(\boldsymbol{x}^*)) \tag{1}$$

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

$$f_i(\boldsymbol{x}) > f_i(\boldsymbol{x}^*) \tag{2}$$

□

Definition 2 (Pareto Dominance): *A vector* $\boldsymbol{u} = (u_1, \ldots, u_k)$ *is said to dominate* $\boldsymbol{v} = (v_1, \ldots, v_k)$ *(denoted by* $\boldsymbol{u} \preceq \boldsymbol{v}$*) 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 3 (Pareto Optimal Set): *For a given MOP* $\boldsymbol{f}(x)$*, the Pareto optimal set (*$\mathcal{P}^*$*) is defined as:*

$$\mathcal{P}^* := \{x \in \Omega \mid \neg\exists\ x' \in \Omega \ \ \boldsymbol{f}(x') \preceq \boldsymbol{f}(x)\}. \tag{3}$$

□

Definition 4 (Pareto Front:): *For a given MOP* $\boldsymbol{f}(x)$ *and Pareto optimal set* $\mathcal{P}^*$*, the Pareto front (*$\mathcal{PF}^*$*) is defined as:*

$$\mathcal{PF}^* := \{\boldsymbol{u} = \boldsymbol{f} = (f_1(x), \ldots, f_k(x)) \mid x \in \mathcal{P}^*\}. \tag{4}$$

□

3 The Immune System

One of the main goals 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). Those molecules that can be recognized by the immune system are called *antigens* that elicit an adaptive immune response.

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* [5]. The new cloned cells undergo high rate somatic mutations or *hypermutation*. The main roles of that mutation process are twofold: to allow the creation of new molecular patterns for antibodies, and to maintain diversity.

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 cloning 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 condition, 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.

4 Previous Work

The first direct use of the immune system to solve multiobjective optimization problems reported in the literature is the work of Yoo and Hajela [19]. 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 genetic algorithm. 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. Besides the limited spread of nondominated solutions produced by the approach, it is well-known that linear aggregating functions have severe limitations for solving multiobjective problems (the main one is that they cannot generate concave portions of the Pareto front [4]. In this study, the approach is not compared to any other technique.

There is an artificial immune system to solve machine learning problems, specifically pattern recognition tasks and multimodal optimization problems called *CLONALG*, it was proposed by De Castro and Von Zuben [6], and it is based on the clonal selection principle. This is the first approach that use that principle to solve multimodal optimization problems.

This paper is an extension of the work published in [2] with some important diferences. It can be considered as the first attempt to use an artificial immune system using the clonal selection principle to solve the general multiobjective optimization problem. That previous work follows the clonal selection principle very close. However in this proposal we do some changes which results in an improved performance of the algorithm, even when it does not follow the clonal selection principle exactly.

[1]

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

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.
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 all the best antibodies (obtained from the previous step) into the secondary memory.
6. We determine for each of the “best” antibodies the number of clones that we want to create. We wish to create the same number of clones of each antibody, and also that the total number of clones created is equal to the 60% of the total population

[1] The exception is a recent approach reported in [1] in which both lexicographic ordering and Pareto-based selection are adopted in an evolutionary programming algorithm used to detect attacks with an artificial immune system for virus and computer intrusion detection. In this case, however, the paper is more focused on the application rather than on the approach and no proper validation of the proposed algorithms is provided.

used. However, if the secondary memory is full, then we modify this quantity doing 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 cloning of the best antibodies based on the information from the previous step. Note that the population size grows after the cloning process takes place. Then, we eliminate the extra individuals giving preference (for survival) to the new clones generated.
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 "worst" antibodies (which are all that not resulted selected as "best antibodies" in step 4). The initial mutation rate adopted is high and it is decreased over time linearly (from 0.9 to 0.3).
10. If the secondary memory is full, we apply crossover to a fraction of its contents (we proposed 60%). The new individuals generated that are nondominated with respect to the secondary memory will then be added to it.
11. After that cloning process ends, the population size is increased, then it is necessary to reset to its original value. We eliminate the excedent individuals but allowing to survive those nondominated individuals.
12. 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. 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.

Even when the multimodal optimization CLONALG version apply cloning and hypermutation process, there are some differences with respect to MISA as the mutation rate, number of clones created, selection strategy among others (and of course we are solving different kind of optimization problems).

Despite that our algorithm is taken operators from Evolutionary Algorithms (EA), as the crossover applied to the elements in the secondary memory (step 10), it is nos a EA because the use of cloning (asexual reproduction) to the main population, the mutation type and the change in the population size.

5.1 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 [11] (see Figure 1).

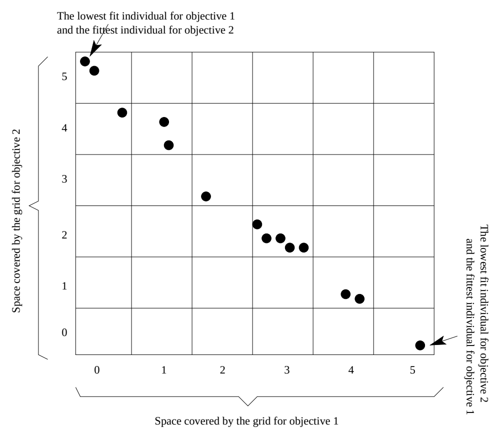

Fig. 1. An adaptive grid to hadle the secondary memory

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.
   – To determine if a certain antibody 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 antibody.

6 Experiments

In order to validate our approach, we used several test functions reported in the standard evolutionary multiobjective optimization literature [17, 3]. 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 (this is a conventional value used by the researchers of the multiobjective optimization area).

Number of iterations is determined by the number of fitness function evaluations requiered. These parameters produce a total of 12,000 fitness function evaluations. MISA was compared against the NSGA-II [9] and against PAES [11]. These two 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. Nondominated Sorting Genetic Algorithm *(NSGA-II)* was proposed by Srinivas and Deb (1994) [8, 9]. It is based on several layers of classifications of the individuals. The population is ranked on the basis of nondominnation: all nondominated individuals are classified into one category sharing a dummy fitness value. Then this first group of classified individuals is ignored and other layer of nondominated individuals is considered. The process continues until all individuals in the population are classified. Then a proportional selection technique is applied. This allows to search for nondominated regions. NSGAII uses elitism and a crowded comparison operator.

Pareto Archivied Evolution Strategy *(PAES)* proposed by Knowles and Corne [11] is a evolution strategy where a single parent generates one single offspring $(1 + 1)$, additionally it uses a historical archive that keeps nondominated solutions previously found, this archive is used as a reference set against which each mutated individual is being copared (as a tournament comparison). As a diversity mechanism it uses a geographically-based procedure which consists of a crowding procedure that divides objective space in a recursive manner. This geographically-based procedure indicates the number of solutions that reside in each grid location.

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. Note that all these three algorithms (NSGAII, PAES and MISA) use a secondary or external memory as elitist mechanism. Besides the graphical comparisons performed, the three following metrics were adopted to compare our results:

– **Error Ratio (ER)**: This metric was proposed by Van Veldhuizen [16] to indicate the percentage of solutions (from the nondominated vectors found so far) that are not members of the true Pareto optimal set:

$$ER = \frac{\sum_{i=1}^{n} e_i}{n}, \tag{5}$$

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 [15] as a way of measuring the range (distance) variance of neighboring vectors in the Pareto front known. This metric is defined as:

$$S \triangleq \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(\overline{d} - d_i)^2}\ , \tag{6}$$

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

– **Generational Distance (GD)**: The concept of generational distance was introduced by Van Veldhuizen & Lamont [18] 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:

$$GD = \frac{\sqrt{\sum_{i=1}^{n} d_i^2}}{n} \tag{7}$$

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.

In all the following examples, we performed 20 runs of each algorithm. The graphs shown in each case were generated using the average performance of each algorithm with respect to generational distance.

Example 1

Our first example is a two-objective optimization problem proposed by Schaffer [14]:

$$\text{Minimize } f_1(x) = \begin{cases} -x & \text{if } x \le 1 \\ -2+x & \text{if } 1 < x \le 3 \\ 4-x & \text{if } 3 < x \le 4 \\ -4+x & \text{if } x > 4 \end{cases} \quad (8)$$

$$\text{Minimize } f_2(x) = (x-5)^2 \quad (9)$$

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

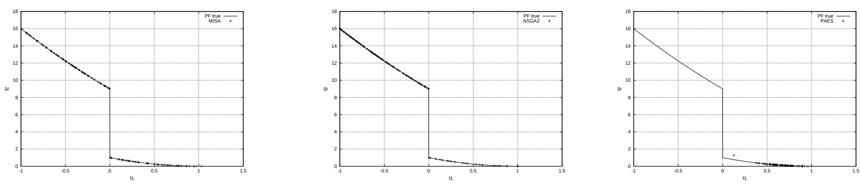

Fig. 2. Pareto front obtained by MISA (left), the NSGA-II (middle) and PAES (right) in the first example. The true Pareto front of the problem is shown as a continuous line (note that the vertical segment is NOT part of the Pareto front and is shown only to facilitate drawing the front).

The comparison of results between the true Pareto front of this example and the Pareto front produced by MISA, the NSGA-II, and PAES are shown in Figure 2. The values of the three metrics for each algorithm are presented in Tables 1 and 2.

	Spacing			**GD**		
	MISA	NSGA-II	PAES	MISA	NSGA-II	PAES
Average	0.236345	0.145288	0.268493	0.000375	0.000288	0.002377
Best	0.215840	0.039400	0.074966	0.000199	0.000246	0.000051
Worst	0.256473	0.216794	1.592858	0.001705	0.000344	0.034941
Std. Dev.	0.013523	0.079389	0.336705	0.000387	0.000022	0.007781
Median	0.093127	0.207535	0.137584	0.000387	0.000285	0.000239

Table 1. Spacing and Generational Distance for the first example.

In this case, MISA had the best average value with respect to generational distance. The NSGA-II had both the best average spacing and the best average error ratio. Graphically, we can see that PAES was unable to find most of the true Pareto front of the problem. MISA and the NSGA-II were able to produce most of the true Pareto front and their overall performance seems quite similar from the graphical results with a slight advantage for MISA with respect to closeness to the true Pareto front and a slight advantage for the NSGA-II with respect to uniform distribution of solutions.

	MISA	NSGA-II	PAES
Average	0.410094	0.210891	0.659406
Best	0.366337	0.178218	0.227723
Worst	0.445545	0.237624	1.000000
Std. Dev.	0.025403	0.018481	0.273242
Median	0.410892	0.207921	0.663366

Table 2. Error ratio for the first example.

Example 2

The second example was proposed by Kita [10]: Maximize $F = (f_1(x,y), f_2(x,y))$ where: $f_1(x,y) = -x^2 + y$, $f_2(x,y) = \frac{1}{2}x + y + 1$, $x, y \geq 0$, $0 \geq \frac{1}{6}x + y - \frac{13}{2}$, $0 \geq \frac{1}{2}x + y - \frac{15}{2}$, $0 \geq 5x + y - 30$.

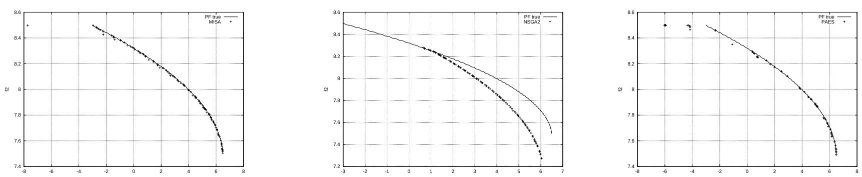

Fig. 3. Pareto front obtained by MISA (left), the NSGA-II (middle) and PAES (right) in the second example. The true Pareto front of the problem is shown as a continuous line.

The comparison of results between the true Pareto front of this example and the Pareto front produced by MISA, the NSGA-II and PAES are shown in Figure 3. The values of the three metrics for each algorithm are presented in Tables 3, and 4.

	Spacing			**GD**		
	MISA	NSGA-II	PAES	MISA	NSGA-II	PAES
Average	0.905722	0.815194	0.135875	0.036707	0.049669	0.095323
Best	0.783875	0.729958	0.048809	0.002740	0.004344	0.002148
Worst	1.670836	1.123444	0.222275	0.160347	0.523622	0.224462
Std. Dev.	0.237979	0.077707	0.042790	0.043617	0.123888	0.104706
Median	0.826587	0.173106	0.792552	0.019976	0.066585	0.018640

Table 3. Spacing and Generational Distance for the second example.

In this case, MISA had again the best average value for the generational distance. The NSGA-II had the best average error ratio and PAES had the best average spacing value. Note however from the graphical results that the NSGA-II missed most of the true

Pareto front of the problem. PAES also missed some portions of the true Pareto front of the problem. Graphically, we can see that MISA found most of the true Pareto front and therefore, we argue that it had the best overall performance in this test function.

	MISA	NSGA-II	PAES
Average	0.007431	0.002703	0.005941
Best	0.000000	0.000000	0.000000
Worst	0.010000	0.009009	0.009901
Std. Dev.	0.004402	0.004236	0.004976
Median	0.009901	0.0000	0.009901

Table 4. Error ratio for the second example.

Example 3

Our third example is a two-objective optimization problem defined by Kursawe [12]:

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

$$\text{Minimize } f_2(\boldsymbol{x}) = \sum_{i=1}^{n} \left(|x_i|^{0.8} + 5 \sin(x_i)^3 \right) \tag{11}$$

where: $-5 \leq x_1, x_2, x_3 \leq 5$

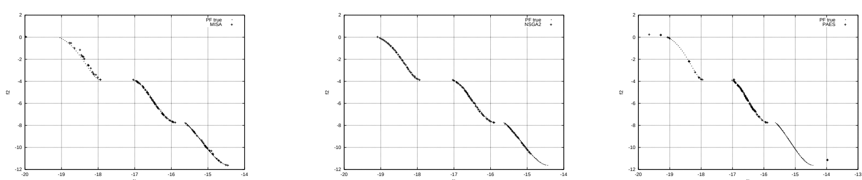

Fig. 4. Pareto front obtained by MISA (left), and the NSGA-II (middle) and PAES (right) in the third example. The true Pareto front of the problem is shown as a continuous line.

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

For this test function, MISA had again the best average generational distance (this value was, however, only marginally better than the average value of the NSGA-II). The NSGA-II had the best average spacing value and the best average error ratio. However, by looking at the graphical results, it is clear that the NSGA-II missed the last (right

	Spacing			GD		
	MISA	NSGA-II	PAES	MISA	NSGA-II	PAES
Average	3.188819	2.889901	3.019393	0.004152	0.004164	0.009341
Best	3.177936	2.705087	2.728101	0.003324	0.003069	0.002019
Worst	3.203547	3.094213	3.200678	0.005282	0.007598	0.056152
Std. Dev.	0.007210	0.123198	0.133220	0.000525	0.001178	0.013893
Median	3.186680	2.842901	3.029246	0.004205	0.003709	0.004468

Table 5. Spacing and Generational Distance for the third example.

lowerhand) portion of the true Pareto front, although it got a nice distribution of solutions along the rest of the front. PAES missed almost entirely two of the three parts that make the true Pareto front of this problem. Therefore, we argue in this case that MISA was practically in a tie with the NSGA-II in terms of best overall performance, since MISA covered the entire Pareto front, but the NSGA-II had a more uniform distribution of solutions.

	MISA	NSGA-II	PAES
Average	0.517584	0.262872	0.372277
Best	0.386139	0.178218	0.069307
Worst	0.643564	0.396040	0.881188
Std. Dev.	0.066756	0.056875	0.211876
Median	0.504951	0.252476	0.336634

Table 6. Error ratio for example 3

Even when it is necessary more statistical analysis (more test functions and analysis of varianza), in general, we can see that MISA provides competitive results with respect to the two 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 (several other test functions were adopted but not included due to space limitations), particularly with respect to the generational distance metric.

7 Conclusions and Future Work

We have introduced a new multiobjective optimization approach based on the clonal selection principle. The approach was found to be competitive with other algorithms representative of the state-of-the-art in the area. Our main conclusion is that the sort of artificial immune system proposed in this paper is a viable alternative 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. Also, it is desirable to refine the mechanism to maintain diversity that our approach currently has, since that is its main current weakness.

Ackowledgements

The first 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. The second author gratefully acknowledges support from CONACyT through project 34201-A.

References

1. Kevin P. Anchor, 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.
2. Carlos A. Coello Coello 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.
3. Carlos A. Coello Coello, 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.
4. Indraneel Das and John Dennis. A Closer Look at Drawbacks of Minimizing Weighted Sums of Objectives for Pareto Set Generation in Multicriteria Optimization Problems. *Structural Optimization*, 14(1):63–69, 1997.
5. Leandro N. de Castro and Jonathan Timmis. *Artificial Immune Systems: A New Computational Intelligence Approach*. Springer, London, 2002.
6. Leandro Nunes de Castro and F. J. Von Zuben. Learning and Optimization Using the Clonal Selection Principle. *IEEE Transactions on Evolutionary Computation*, 6(3):239–251, 2002.
7. Kalyanmoy Deb. *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley & Sons, Chichester, UK, 2001. ISBN 0-471-87339-X.
8. Kalyanmoy Deb, 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.

[2] The algorithm proposed here is rather simple to implement, but in any case, our source code is available upon request via email.

9. Kalyanmoy Deb, 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.
10. Hajime Kita, 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.
11. Joshua D. Knowles and David W. Corne. Approximating the Nondominated Front Using the Pareto Archived Evolution Strategy. *Evolutionary Computation*, 8(2):149–172, 2000.
12. Frank Kursawe. 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.
13. Kaisa M. Miettinen. *Nonlinear Multiobjective Optimization*. Kluwer Academic Publishers, Boston, Massachusetts, 1998.
14. J. David Schaffer. *Multiple Objective Optimization with Vector Evaluated Genetic Algorithms*. PhD thesis, Vanderbilt University, 1984.
15. Jason 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, Massachusetts, May 1995.
16. David 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, Ohio, May 1999.
17. David A. Van Veldhuizen 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.
18. David A. Van Veldhuizen 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.
19. J. Yoo and P. Hajela. Immune network simulations in multicriterion design. *Structural Optimization*, 18:85–94, 1999.