A Micro Artificial Immune System

Juan Carlos Herrera-Lozada, Hiram Calvo, and Hind Taud

Abstract—**In this paper, we present a new algorithm, namely, a micro artificial immune system (Micro-AIS) based on the Clonal Selection Theory for solving numerical optimization problems. For our study, we consider the algorithm CLONALG, a widely used artificial immune system. During the process of cloning, CLONALG greatly increases the size of its population. We propose a version with reduced population. Our hypothesis is that reducing the number of individuals in a population will decrease the number of evaluations of the objective function, increasing the speed of convergence and reducing the use of data memory. Our proposal uses a population of 5 individuals (antibodies), from which only 15 clones are obtained. In the maturation stage of the clones, two simple and fast mutation operators are used in a nominal convergence that works together with a reinitialization process to preserve the diversity. To validate our algorithm, we use a set of test functions taken from the specialized literature to compare our approach with the standard version of CLONALG. The same method can be applied in many other problems, for example, in text processing.**

Index terms—**Artificial immune system, Clonal selection theory, micro algorithm, numerical optimization.**

I. Introduction

BIO-INSPIRED and Evolutionary algorithms have become very important within the area of artificial intelligence, because they have proved successful in solving certain complex problems of machine learning, classification and numerical optimization [1]. Such techniques are population based, in other words, they use a population of potential solutions enabling a wide exploration of the search space. The simplicity of an algorithm is one of the current trends in the field of evolutionary computation, although in the majority of cases, the performance is sacrificed in favor of a lower computational cost [2]. Due to the simultaneous manipulation of a large set of solutions, implementation of an algorithm requires a large space in data memory and generally high processing time. To reduce these factors, algorithms with extremely small populations are designed, and for most applications their performance is comparable with the standard population algorithms [3].

As the evolutionary algorithms, Artificial Immune System (AIS) has been successfully applied to a variety of optimization problems [4]. AIS is a computational intelligence paradigm inspired by the biological immune system which has found application in pattern recognition and machine learning. Different ways of AIS for optimization as the immune network theory and the clonal selection principle have been proposed and implemented by different researchers as explained in [5].

The main motivation of our research is to propose a simple and powerful algorithm which presents a reduced computational cost when using a micro population of individuals within a clonal proliferation scheme which is the central point in the functioning of artificial immune system.

Two novel mutation operators were designed and implemented. These operators accelerate the convergence by providing a uniform search to avoid getting into local optimum.

In this work we apply micro-AIS for numerical optimization, but the same method can be applied in many other problems, for example, in text processing. It is promising to apply bio-inspired algorithms (more specifically, genetic algorithms) in text processing tasks, see, for example, [6].

A. *Previous Work*

In [3] Goldberg introduced the concept of nominal convergence when he experimented with a simple genetic algorithm (GA) using a population of only 3 individuals. He found that these 3 chromosomes were sufficient to ensure convergence of the algorithm regardless of the size of them, aided by a process of elitism. Goldberg applied genetic operators in a nominal convergence which is controlled by two possible parameters: a specified number of generations or a degree of similarity among all chromosomes. At the end of the nominal convergence, the best individual is preserved and two individuals are randomly generated: they will form the new population.

In [7] Krishnakumar designed a GA with a population of 5 individuals and he named his algorithm *Micro Genetic Algorithm* (Micro-GA). Like Goldberg, Krishnakumar used elitism to preserve the best single strand found at the end of nominal convergence, as one of the individuals used for the next generation. When comparing the performance of the Micro-GA with a simple GA with a population of 50 individuals, better results were obtained on functions of only one objective and the GA with a reduced population converged faster. Krishnakumar's algorithm has achieved good results when it is used to solve optimization problems for high-dimensional functions [8].

Dozier *et al.* in [9] presented two heuristic-based micro genetic algorithms which quickly find solutions to constraints satisfaction problem. They experimented with different sizes of micro population and found that for a particular problem, a relatively small number of individuals in the genetic algorithm was sufficient.

Manuscript received March 12, 2011. Manuscript accepted for publication June 6, 2011.

Juan Carlos Herrera-Lozada and Hiram Calvo are with Centro de Investigación en Computación, Instituto Politécnico Nacional, México D. F., 07738, Mexico (e-mail: jlozada@ipn.mx, hcalvo@cic.ipn.mx).

Hind Taud is with Centro de Innovación y Desarrollo Tecnológico en Cómputo, Instituto Politécnico Nacional, México D. F., 07700, Mexico (e-mail: htaud@ipn.mx).

Coello and Toscano designed a Micro-GA for solving the multi-objective optimization problem [10], providing criteria for the management of constraints, besides proposing a scheme of Pareto dominance with a geographical location to maintain the diversity and uniformly distribution of the solutions on the Pareto front. This algorithm works with a population of 4 individuals and uses a secondary memory that stores potential solutions throughout the search. This approach was widely used to successfully solve various engineering problems as discussed in [11] and [12].

Recently, Fuentes and Coello in [13] designed a micro algorithm for PSO (Particle Swarm Optimization) to solve optimization problems of one objective and constraints satisfaction. They use 5 particles (individuals) helped by a nominal convergence.

With regard to artificial immune systems with small population, there are no studies reported in the literature. There are, however, certain similarities with the works cited above:

1. Population size of 3 to 5 individuals.
2. Nominal convergence is required as well as a reinitialization process.
3. Elitism is necessary to preserve at least the best individual obtained at the end of the nominal convergence.

II. Artificial Immune System

De Castro and Von Zuben developed the Clonal Selection Algorithm (CLONALG) on the basis of clonal selection theory of the immune system [14, 15]. Clonal Selection is based on the way in which both B-cells and T-cells adapt in order to match and kill the foreign cells. This algorithm can perform pattern recognition and adapt to solve multimodal optimization tasks. The block diagram of CLONALG is shown in Fig. 1. This algorithm is described as follows:

(1) Generate (randomly) a set (*P*) of candidate solutions or antibodies, composed of the memory cells (*M*) and the remaining population (*Pr*), ($P = Pr + M$);

(2) Select the *n* best antibodies (P_n), based on affinity;

(3) Clone these *n* best antibodies in proportion to their affinity using $N_c = \sum_{i=1}^{n} round\left(\frac{\beta \cdot N}{i}\right)$ where N_c is the total number of clones generated for each of the antigens like objective function, β is a multiplying factor, *N* is the total number of antibodies, and $round\,(.)$ is the operator that rounds its argument toward the closest integer. Each term of this sum corresponds to the clone size of each selected antibody, e.g., for N=100 and $\beta = 1$, the antibody with highest affinity will produce 100 clones; the antibody with the second highest affinity produces 50 clones, and so on, giving rise to a temporary set of clones (C);

(4) Apply a hypermutation to the temporary clones. The degree of mutation is inversely proportional to the affinity. The maturated antibodies are generated (C*);

(5) Re-select the best elements from C* to compose the memory set M. Some members of P can be replaced by other improved members of C*;

(6) Replace *d* antibodies by novel ones to introduce the diversity concept. The probability to be replaced is inversely proportional to the affinity of the previous remaining population (Pr).

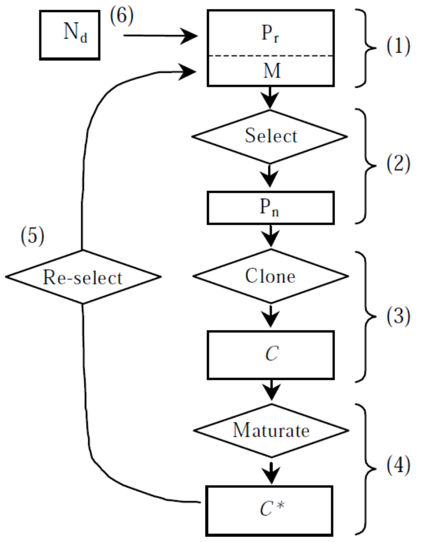

Fig. 1. Block diagram of the clonal selection algorithm CLONALG by De Castro and Von Zuben.

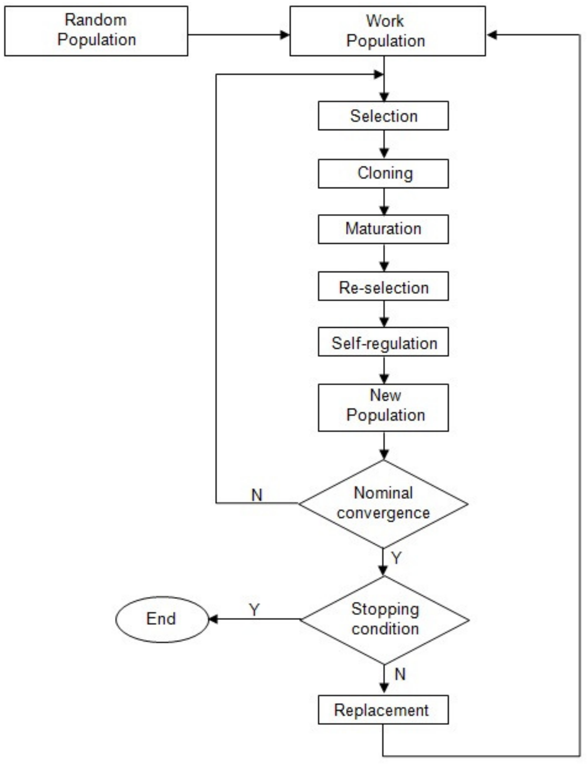

Fig. 2. Micro-AIS.

III. Micro Artificial Immune System

Fig. 2 shows our algorithm. Our methodology is based on the methodology proposed by Goldberg in [3]: the variation operators are applied to a small population (randomly generated) to achieve nominal convergence. Subsequently, a new population should be generated by transferring the best individuals of the population obtained after the convergence to the new one. The remaining individuals are randomly generated.

The proposed algorithm works as follows:

(1) Generate randomly a population of 5 antibodies (individuals). In the initial generation, these antibodies are copied directly to the working population and nominal convergence is controlled by the number of generations, in our case equal to 10.

(2) Use selection based on ranking. The antibody with the highest affinity will be the best individual. In our algorithm we named this individual as *BestAb*.

(3) Perform the cloning of the antibodies using $N_C = \sum_{i=1}^{n} (n - (i-1))$, where N_C is the number of clones to be generated for each antibody, n is the total number of antibodies of the population and i is the current antibody starting from the antibody with the highest affinity (*BestAb*).

(4) Consider a population of 5 antibodies and generate a population of 15 clones: *BestAb* antibody gets 5 clones; the second ranking antibody gets 4 clones and so on until the worst antibody that gets a single clone.

(5) Perform the maturation of clones using mutation process. The probability of mutation is set at the beginning of nominal convergence for each group of clones obtained from the same antibody. This probability is determined in proportion to the affinity of each antibody and decreases uniformly in each generation, so the group of clones obtained from *BestAb* mutates less than other groups of clones that have been generated from the remaining antibodies. The single clone that we got from the worst antibody has the highest possibility to mutate. For this purpose we use

$$prob\_mutation(i) = \frac{Aff(i)}{\sum_{i=1}^{n} Aff(i)}$$

where i is the antibody that will set the mutation probability for the group of clones that were obtained from himself and n is the total antibody population. To decrease the mutation probability uniformly in each generation, within the nominal convergence we used

$$if\ prob \le \frac{prob\_mutation(i)}{generation}\quad then\ \ to\ apply\ mutation$$

where $random\ prob \in [0,1]$ and *generation* variable is the current generation within nominal convergence considering $int\ generation \in [1,10]$. Note that we should not divide by zero.

For the variation of each of the clones, we present two operators that are rather simple and mostly exploit the search space to perform different step sizes in the process of mutation. Several aspects have been considered to implement these operators: the number of clones, the current generation within nominal convergence and the permissible range of values of the decision variables. We use the following two mutation operators, with a 50% probability, which act on each decision variable of a clone (in our scheme, the entire solution vector is mutated):

$$x' = x + \frac{(\alpha \cdot range \cdot generation)}{N_c}$$

and

$$x' = x + \frac{(\alpha \cdot range)}{(generation \cdot N_c)}$$

where x' is the mutated decision variable, x is the decision variable to mutate, α is a uniform random number where $random\ \alpha \in [0,1]$, *generation* is the current generation within the nominal convergence and N_c is the total number of clones. The value of α is computed for each decision variable of the clone.

In case of the 5 clones derived from *BestAb*, $range \in [LB, UB]$ is a random number between the lower bound (LB) and the upper bound (UB) of decision variables and it is a constant value for all the dimension of the clone, in other words, it has the same value for all decision variables of the clone.

For the remaining clones which were obtained from the other 4 antibodies, *range* is any value (decision variable) from *BestAb* antibody which is chosen randomly.

The first operator using in the mutation generates step sizes larger than the second operator.

(6) Make another selection based on ranking. This time, we sort the 15 clones with respect to their affinity. We must select the two best clones (elitism) and the new population is completed with 3 other clones selected randomly from the population of mature clones. The remaining clones will be eliminated, providing a self-regulation within the nominal convergence.

(7) When nominal convergence is achieved (while working with 10 generations), we keep the two best clones, and other 3 antibodies are generated randomly to complete the new working population and the nominal convergence starts again until the algorithm achieves the stop condition.

IV. Experimental Setup

In order to validate the proposed approach, we used the multivariate functions presented in [16]. These functions are listed in Appendix A. All selected test functions have 30 variables (dimensions) and an optimum value at zero, except for *f08* with an optimum at -12569.5. For all cases we used a population of 5 individuals and nominal convergence in 10 generations. The general stop criterion of the algorithm varied depending on the problem to be solved. For the experiments, we used a 2.66 GHz Quad Core PC with 2MB. Table I shows the results for 20 runs of the algorithm.

TABLE I
RESULTS OBTAINED WITH MICRO-AIS

Function	External cycle	Nominal Convergence	Best	Worst	Mean
f01	1000	10	0.0	0.000022	0.000009
f02	1000	10	0.0	0.000017	0.000008
f03	1000	10	0.0	0.000002	0.000001
f04	1000	10	0.0	0.000012	0.000005
f05	1000	10	0.0	0.000028	0.000012
f06	2000	10	0.0	0.000032	0.000015
f07	2000	10	0.0	0.000027	0.000013
f08	2000	10	-12569.5	-12569.57	-12569.496
f09	2000	10	0.0	0.000033	0.000013
f10	2000	10	0.0	0.000011	0.000007
f11	2000	10	0.0	0.000013	0.000004

TABLE II
CLONALG VS. MICRO-AIS

	Ab (antibodies)	Clones	Nominal Convergence	External cycle	Evaluations to objective function	Time (seconds)
CLONALG						
f01	50	256	0	1000	1,280,000	47.2
f05	70	312	0	1000	21,840,000	78.6
f07	70	312	0	1200	26,208,000	103.7
Micro-AIS						
f01	5	15	10	1000	750,000	14.8
f05	5	15	10	1000	750,000	14.2
f07	5	15	10	2000	1,500,000	48.3

To validate the performance of our algorithm with respect to the standard version of CLONALG, we compared it with some of the above mentioned functions under equal conditions. The main results are related with the number of evaluations of the objective function and convergence time. Table II lists these results for 20 runs of both algorithms. We implemented the adaptations to CLONALG for using multivariate functions. For CLONALG we used a multiplication factor $\beta = 1$ and the number of antibodies listed in Table II.

V. CONCLUSIONS AND FUTURE WORK

In this paper we presented a new micro algorithm based on clonal selection theory for solving numerical optimization problem. Since the model of the artificial immune system does not include a crossover operator, cloning (set to 15 clones in our case) and mutation represent the main challenges for maintaining diversity.

Two mutation operators were designed in our approach showed excellent solutions with a low computational cost. These operators were used without modifications in all selected test functions. As shown in the results listed in Tables I and II, the Micro-AIS converges faster than CLONALG and uses less data memory. The nominal convergence and elitism of 40% of the population (considering only 5 antibodies) are of great importance to ensure the proper functioning of the algorithm.

Future work is aimed at the following four aspects:

- Find faster mutation operators,
- Design versions for handling constraints and muti-objective optimization,
- Develop possible hardware architectures, and
- Develop applications in different areas (for example, in text processing) and experiment with them.

REFERENCES

[1] D. Ashlock, *Evolutionary Computation for Modeling and Optimization*, Springer, 2005.

[2] M. Munetomo and Y. Satake, “Enhancing Model-building Efficiency in Extended Compact Genetic Algorithms,” in *ICSMC '06. IEEE International Conference on Systems, Man and Cybernetics, 2006*, Volume 3, Oct. 8-11, 2006, pp. 2362-2367.

[3] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison Wesley, Reading, MA, 1989.

[4] L. Nunes de Castro and J. Timmis, *Artificial Immune Systems: A new Computational Intelligence Approach*, Springer, 2002.

[5] D. Dasgupta, “Advances in artificial immune systems,” *Computational Intelligence Magazine*, IEEE, vol 1, issue 4, pp. 40-49, Nov. 2006.

[6] A. Gelbukh, G. Sidorov, D. Lara-Reyes, and L. Chanona-Hernandez, “Division of Spanish Words into Morphemes with a Genetic Algorithm,” *Lecture Notes in Computer Science*, 5039, Springer-Verlag, pp. 19-26, 2008.

[7] K. Krishnakumar, “Micro-genetic algorithms for stationary and non-stationary function optimization” in *SPIE Proceedings: Intelligent Control and Adaptive systems*, 1989, pp. 289-296.

[8] G. Alvarez, *Can we make genetic algorithms work in high-dimensionality problems?* Stanford Exploration Project (SEP) report 112, 2002.

[9] G. Dozier, J. Bowen and D. Bahler, “Solving Small and Large Scale Constraint Satisfaction Problems Using a Heuristic-Based Microgenetic Algorithm,” in *Proceedings of the First IEEE Conference on Evolutionary Computation (ICEC'94),* Z. Michalewicz, J. D. Schaffer, H.-P. Schwefel, D. B. Fogel and H. Kitano (eds), 1994, pp. 306-311.

[10] G. Toscano, Carlos. A. Coello, “A Micro-Genetic Algorithm for multiobjective optimization,” in *First International Conference on*

Evolutionary Multi-criterion Optimization, Lecture Notes in Computer Science, vol. 1993, Springer, 2001, pp. 126-140.

[11] Y. Ming and L. Cheng, "Application of Micro Genetic Algorithm to Optimization of Time-Domain Ultra-Wide Band Antenna Array," in *Microwave and Millimeter Wave Technology, 2007, ICMMT '07, International Conference*, April 2007, pp. 1-4.

[12] J. Mendoza, D. Morales, R. López, J. Vannier, and C. A. Coello, "Multiobjective Location of Automatic Voltage Regulators in a Radial Distribution Network Using a Micro Genetic Algorithm," *IEEE Transactions on Power Systems*, vol. 22, issue 1, pp. 404-412, Feb. 2007.

[13] J. C. Fuentes and C. A. Coello, "Handling Constraints in Particle Swarm Optimization Using a Small Population Size," *Lecture Notes in Computer Science, MICAI 2007: Advances in Artificial Intelligence*, vol. 4827, Springer, 2007.

[14] L. Nunes de Castro and F. J. Von Zuben, "The clonal selection algorithm with engineering applications," in *Proceedings of Genetic and Evolutionary Computation Conference, Workshop on AISAA*, July 2000, pp. 36-37.

[15] L. Nunes de Castro and F. J. Von Zuben, "Learning and optimization using the clonal selection principle," *IEEE Trans. Evol. Comput.*, vol. 6, no. 3, pp. 239-251, Jun. 2002.

[16] E. Mezura, J. Velázquez, and C. A. Coello, "A comparative study of differential evolution variants for global optimization," in *ACM, GECCO 2006*, pp. 485-492.

APPENDIX A

Multivariate functions for the experimental setup, taken from [15].

f01 – Sphere Model

$$f_1(x) = \sum_{i=1}^{30} (x_i)^2$$

$-100 \leq x_i \leq 100$
min $(f_1) = f_1(0, \ldots, 0) = 0$

f02 – Schwefel's Problem

$$f_2(x) = \sum_{i=1}^{30} |x_i| + \prod_{i=1}^{30} |x_i|$$

$-10 \leq x_i \leq 10$
min $(f_2) = f_2(0, \ldots, 0) = 0$

f03 – Schwefel's Problem

$$f_3(x) = \sum_{i=1}^{30} \left(\sum_{j=1}^{i} x_j \right)^2$$

$-100 \leq x_i \leq 100$
min $(f_3) = f_3(0, \ldots, 0) = 0$

f04 – Schwefel's Problem

$$f_4(x) = max_i \{ [x_i], 1 \leq i \leq 30 \}$$

$-100 \leq x_i \leq 100$
min $(f_4) = f_4(0, \ldots, 0) = 0$

f05 – Generalized Rosenbrock's Function

$$f_5(x) = \sum_{i=1}^{29} \left| 100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2 \right|$$

$-30 \leq x_i \leq 30$
min $(f_5) = f_5(1, \ldots, 1) = 0$

f06 – Step Function

$$f_6(x) = \sum_{i=1}^{30} (\lfloor x_i + 0.5 \rfloor)^2$$

$-100 \leq x_i \leq 100$
min $(f_6) = f_6(0, \ldots, 0) = 0$

f07 – Quartic Function with Noise

$$f_7(x) = \sum_{i=1}^{30} i x_i^4 + random[0,1)$$

$-1.28 \leq x_i \leq 1.28$
min $(f_7) = f_7(0, \ldots, 0) = 0$

f08 – Generalized Schwefel's Problem

$$f_8(x) = \sum_{i=1}^{30} \left(x_i \sin\left(\sqrt{|x_i|} \right) \right)$$

$-500 \leq x_i \leq 500$
min $(f_8) = f_8(420.9687, \ldots, 420.9687) = -12596.5$

f09 – Generalized Rastrigin's Problem

$$f_9(x) = \sum_{i=1}^{30} \left[x_i^2 - 10\cos(2\pi x_i) + 10 \right]$$

$-5.12 \leq x_i \leq 5.12$
min $(f_9) = f_9(0, \ldots, 0) = 0$

f10 – Ackley's Function

$$f_{10}(x) = -20e\left(-0.2\sqrt{\frac{1}{30}\sum_{i=1}^{30} x_i^2} \right) - e\left(\frac{1}{30}\sum_{i=1}^{30}\cos(2\pi x_i) \right) + 20 + e$$

$-32 \leq x_i \leq 32$
min $(f_{10}) = f_{10}(0, \ldots, 0) = 0$

f11 – Generalized Griewank's Function

$$f_{11}(x) = \frac{1}{4000}\sum_{i=1}^{30} x_i^2 - \prod_{i=1}^{30}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$$

$-600 \leq x_i \leq 600$
min $(f_{11}) = f_{11}(0, \ldots, 0) = 0$