Use of Emulations of the Immune System to Handle Constraints in Evolutionary Algorithms

Carlos A. Coello Coello* & Nareli Cruz Cortés†
CINVESTAV-IPN
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
†cnarelli@computacion.cs.cinvestav.mx

Abstract

In this paper, we propose an improved version of a constraint-handling scheme based on a model of the immune system. The approach is coupled to a genetic algorithm and used for global optimization. Our experiments indicate that the approach is relatively easy to implement, and that it is also efficient (computationally speaking).

1 Introduction

Genetic algorithms (GAs) have been quite successful in a wide variety of optimization problems [11, 17]. However, GAs, are an unconstrained search technique and, therefore, require an additional mechanism to incorporate constraints of any type (linear, nonlinear, equality and inequality) into the fitness function in order to guide the search properly. The approach most commonly used to incorporate constraints is the penalty function (mainly exterior), and there have been many successful applications of this approach in the literature [18, 2]. However, penalty functions have some well-known limitations [21], from which the most remarkable is the difficulty to define good penalty factors. These penalty factors are normally generated by trial and error, although their definition may severely affect the results produced by the GA [21].

In this paper, we propose an algorithm based on emulations of the immune system to handle the constraints of a problem being solved by a GA. The approach does not require the definitions of any penalty factors, it is conceptually simple and efficient, and it produces results that are competitive with those produced by best constraint-handling technique used for GAs known to date.

The paper is organized as follows. Section 2 defines the problem to be solved. In Section 3, some basic concepts related to the artificial immune system are introduced. The algorithm proposed is discussed in Section 4 and is validated with the examples from Section 5. Our comparison of results is presented in Section 6. The paper ends with our conclusions and a discussion of some possible paths of future research.

2 Problem Statement

In this paper, we will use GAs as a numerical optimization tool, and we will propose a constraint-handling technique that makes unnecessary the use of a penalty function, which is the most common approach used to incorporate constraints into the fitness function of a GA [21].

The problem that is of interest to us is the general non-linear programming problem in which we want to:

$$\text{Find } \vec{x} \text{ which optimizes } f(\vec{x}) \tag{1}$$

subject to:

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

$$h_j(\vec{x}) = 0, \quad j = 1, \ldots, p \tag{3}$$

where $\vec{x}$ is the vector of solutions $\vec{x} = [x_1, x_2, \ldots, x_r]^T$, n is the number of inequality constraints and p is the number of equality constraints (in both cases, constraints could be linear or non-linear). Only inequality constraints will be considered in this work.

3 The Immune System

Nature has long served as inspiration for many of the scientific and technological advancements of humankind. Humans, after all, can be seen as complex biological information processing systems capable of performing complex tasks even in the presence of uncertainty and lack of information.

We can consider three main information processing systems with a strong biological inspiration [3]: artificial neural networks (inspired on neural systems), evolutionary algorithms (inspired on the mechanism of "survival of the fittest"), and the artificial immune system (inspired on our immune system). Our current discussion will concentrate on the last of these systems.

From the information processing perspective, the immune system is seen as a parallel and distributed adaptive system [10]. It is capable of learning, it uses memory and is able of associative retrieval of information in recognition and classification tasks. Particularly, it leads 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 [3]. All these features of the immune system provide, in consequence, great robustness, fault tolerance, dynamism and adaptability [8]. These are the properties of the immune system that mainly attract researchers to try to emulate it in a computer.

The first conceptual and mathematical models of the biological immune system were developed mainly during the 1980s [19, 25, 26, 20].

There are certain computational models that emulate some specific process of the immune system with the purpose of understanding its nature [8]. For example, the model of specific response that the immune system presents in the presence of an invader (primary response), was designed by Forrest et al. [6] and Smith [24]. On the other hand, the first computational model that emulates the secondary response (memory) of the immune system was developed by Derek Smith [23].

Farmer et al. [5] were the first to suggest a way of representing the immune system in a computer. In their model, both antigens and detectors are represented as strings of symbols in a small alphabet, and the interactions among the strings represent molecular bonds. Recent examples of this model can be seen in the work of Detours & Perelson [4].

Specifically within the area of the artificial immune system, there are two main models in which most of the current work is based [3]: the *model of immune network* and the *negative selection algorithm*. Both will be briefly described next.

3.1 Model of immune network

This is a mathematical model of the immune system developed by Jerne [15]. It uses differential equations that simulate the dynamics of the linphocites (increase or decrement of the concentration of a certain set of linphocites'clones). This model is based on the idea that linphocites do not work in an isolated manner, but as an interconnected network that works in levels. Based on this model, Perelson [20] presented a probabilistic study of the idiotypical networks. This is a very formal study where the transition phases of the idiotypical network are discussed.

Jerne's model has received a lot of atention in the last few years [20, 25], and a considerable number of computational models have been derived from it for practical purposes [13, 14].

3.2 Negative selection algorithm

Forrest et al. [7] developed the negative selection algorithm for detection of changes. This algorithm is based on the discrimination principle that the immune system uses to know what is part of it and what is not. This algorithm generates detectors in a random manner, eliminating those that are not capable of recognizing themselves. Then, only those detectors able to identify invaders are kept.

This algorithm seems to have a great potential in applications related to change detection, due mainly to the fact that such detection is performed probabilistically by the algorithm. It is also a robust system because it looks for any alien action instead of just looking for a certain specific pattern of changes.

Besides the models previously mentioned, there are others used to simulate different aspects of the immune system such as its ability to detect patterns in a noise environment [6], its ability to discover and maintain diverse classes of patterns and its ability to learn effectively, even when not all the possible types of invaders had been previously presented to it.

4 The Proposed Approach

Our algorithm to handle constraints is based on the negative selection model of Forrest et al. [7] and is described next:

1. Generate randomly an initial population for the GA

2. If the initial population contains a mixture of feasible and infeasible, then we divide the population in two groups. The first will contain the infeasible individuals, which will be denominated "antibodies", and the second will contain the feasible individuals, which will be called "antigens".

3. If none of the individuals in the initial population is feasible, then we will use the magnitude of constraint violation of each individual as its fitness. Then, we will use as the "antigen" to the best individual in the population, where "best" means lower constraint violation. The "antibodies" are a randomly selected portion of the population. For the assignment of antibodies, is important to consider only a portion of the population, to avoid premature convergence. Under this scheme (i.e., no feasible individuals in the initial population), we run our simulation of the immune system based on minimum constraint violation and we verify at certain intervals if we have reached the feasible region. Whenever that happens, then we use the feasible individuals found as our antigens, and we proceed to divide the population as in the previous step.

4. The fitness of all the antibodies is set to zero.

5. The fitness of the antibodies is computed according to their similarity with a set of pre-determined antigens in the following way:

   - An antigen is randomly selected from the antigens population.
   - From the population of N antibodies, we randomly select a sample of size σ without replacement.
   - Each antibody in the sample is compared against the antigen selected, and we compute the result of the comparison, to which we will call Z (matching magnitude). Z represents a distance (normally but not necessarily Euclidean) measured at the genotype level (i.e., at the level of the chromosomic encoding).

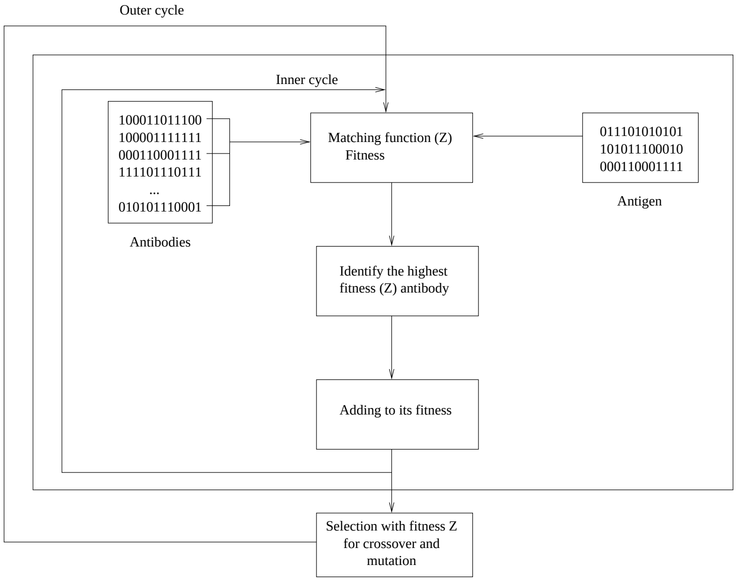

Figure 1: Our algorithm of the immune system.

- We identify the antibody in the population that has the highest matching magnitud (Z). Ties are solved by randomly choosing a winner.
- The matching magnitude of the winner antibody is added to its fitness. The fitness of the other antibodies remains unaltered.
- The antibodies are returned to the population, and the process is repeated typically during three times the number of antibodies.

6. Based on the fitness computed in the previous step, the population of antibodies is reproduced in a traditional genetic algorithm (using crossover and mutation).
7. The process is repeated from the fourth step until convergence (i.e., when the mean and the maximum fitness in the population are practically the same) or until we reach a maximum number of iterations.
8. Individuals are returned to the external GA and we proceed in the conventional way.

This algorithm is illustrated in Figure 1. The amount of similarity Z of an antibody against an antigen is computed using:

$$Z = \sum_{i=1}^{L} t_i \tag{4}$$

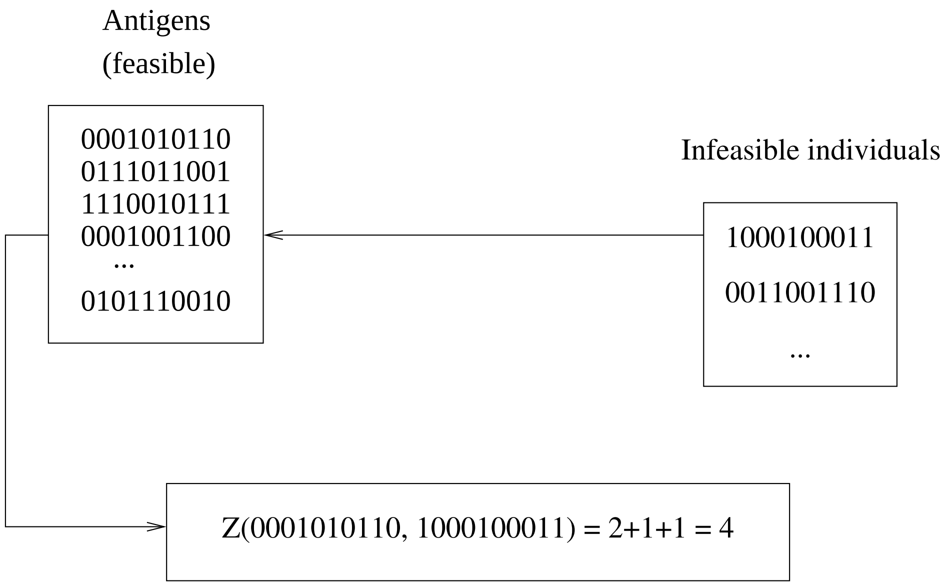

Figure 2: Matching measure Z.

where $t_i = 1$ if there is a matching at position $i = 1, \cdots, L$ (L is the length of the chromosome), or zero if there is not match. This is illustrated in Figure 2. A large Z value means a high matching between the two strings compared and, therefore, a high fitness value.

There are a few issues that need to be mentioned. First, the approach is really using a GA embedded inside another GA used to optimize a certain function. However, the GA that is run with the emulation of the immune system does not use the fitness function directly and, therefore, does not require to evaluate the objective function of the problem. Also, the implicit premise of the technique is that, under certain conditions, the reduction of genotypic differences between two individuals will produce, as consequence, a phenotypic similarity, which, in our case, will make that an infeasible individual approaches the feasible region[1].

The size of the sample σ of antibodies determines if the antibodies are *generalists* (i.e., able to distinguish a great variety of antigens) or *specialists* (i.e., able to distinguish only a small variety of antigens). If the value of σ (i.e., size of the sample of antibodies) is sufficiently large, then the antibodies tend to be specialists, since there is a larger probability of an antibody to match exactly a certain antigen.

Conversely, if the value of σ is small, then the antigens tend to be generalists. When a sample of size σ is equal or slightly larger than the number of antigens, then both generalists and specialists individuals exist in the popula-

[1] This can be debatable, but there is some theory that sustains this assumption [9, 22]. Also, the results obtained by Hajela & Lee [12] and us, are empirical evidence of this assumption.

tion. Generalists antibodies contain fragments of information that encode good solutions.

Our algorithm is an extension of the proposal of Hajela & Lee [12] in which, among other things, they always assumed an initial population containing a mixture of feasible and infeasible individuals, and considered only binary encoding. Although the results reported in this paper were produced using binary representation, our algorithm can be used with any other representation (appropriate matching functions have been defined for other encodings, such as integer representation). We have also performed a comprehensive empirical study of the performance of the algorithm when used to solve several standard benchmark problems reported in the literature[2]. Our study has included an analysis of the role of σ in the performance of the algorithm. To the best of our knowledge, this study is the first of its type for this sort of constraint-handling technique based on the immune system.

5 Examples

Several examples were taken from the specialized literature to evaluate the algorithm proposed. Only three will be shown here due to space limitations. The three test functions selected were taken from Michalewicz's benchmark [18], which is a well-known test suite to evaluate the performance of new approaches to incorporate constraints into an evolutionary algorithm.

5.1 Example 1

Minimize:

$$f(x) = 5.3578547x_3^2 + 0.8356891x_1x_5 + 37.293239x_1 - 40792.141 \tag{5}$$

subject to:

$$g_1(x) = 85.334407 + 0.0056858x_2x_5 + 0.0006262x_1x_4 - 0.0022053x_3x_5 - 92 \leq 0 \tag{6}$$

$$g_2(x) = -85.334407 - 0.0056858x_2x_5 - 0.0006262x_1x_4 + 0.0022053 \leq 0 \tag{7}$$

$$g_3(x) = 80.51249 - 0.0071317x_2x_5 + 0.0029955x_1x_2 + 0.0021813x_3^2 - 110 \leq 0 \tag{8}$$

$$g_4(x) = -80.51249 - 0.0071317x_2x_5 + 0.0029955x_1x_2 - 0.0021813x_3^2 + 90 \leq 0 \tag{9}$$

$$g_5(x) = 9.300961 + 0.0047026x_3x_5 + 0.0012547x_1x_3 + 0.0019085x_3x_4 - 25 \leq 0 \tag{10}$$

$$g_6(x) = -9.300961 - 0.0047026x_3x_5 - 0.0012547x_1x_3 - 0.0019085x_3x_4 + 20 \leq 0 \tag{11}$$

where $78 \leq x_1 \leq 102$, $33 \leq x_2 \leq 45$ y $27 \leq x_i \leq 45$ para $3 \leq i \leq 5$.

The global optimum of this problem is located at $x^* = (78, 33, 29.995256, 45, 36.775812)$ where $f(x^*) = -30665.539$. Two constraints are active at the optimum: g_1 and g_6.

[2]The approach proposed by Hajela & Lee [12] was only tested with some engineering design problems.

5.2 Example 2

Minimize:

$$f(x) = (x_1 - 10)^3 + (x_2 - 20)^3 \tag{12}$$

subject to:

$$g_1(x) = -(x_1 - 5)^2 - (x_2 - 5)^2 + 100 \leq 0 \tag{13}$$

$$g_2(x) = (x_1 - 6)^2 + (x_2 - 5)^2 - 82.81 \leq 0 \tag{14}$$

where $13 \leq x_1 \leq 100$ y $0 \leq x_2 \leq 100$. The global optimum is located at $x^* = (14.095, 0.84296)$, where $f(x^*) = -6961.81388$. Both constraints are active at the optimum.

5.3 Example 3

Maximize:

$$f(x) = (100 - (x_1 - 5)^2 - (x_2 - 5)^2)/100 \tag{15}$$

subject to:

$$g_1(x) = (x_1 - p)^2 + (x_2 - q)^2 + (x_3 - r)^2 - 0.0625 \leq 0 \tag{16}$$

where $0 \leq x_i \leq 10$ para $1 \leq i \leq 3$ y $p, q, r = 1, 2, \ldots, 9$. The feasible region of the search space for this problem consists of 9^3 disjoint spheres. A point (x_1, x_2, x_3) is considered feasible if and only if there are p, q and r such that the above inequality holds. The global optimum of this problem is located at $x^* = (5, 5, 5)$, where $f(x^*) = 1$.

6 Comparison of Results

To have a better idea of the degree of difficulty of each of the test functions selected, we computed the metric ρ, suggested by Koziel & Michalewicz [16] to determine how difficult is to reach the feasible region of a certain constrained optimization problem. We determined $\rho = \mid F \mid / \mid S \mid$ experimentally, by generating one million random points in the search space $\mathcal{S}$ and determining whether or not they lied or not within the feasible region $\mathcal{F}$.

Table 1 shows the values of ρ obtained for each of the three test functions chosen to validate our approach. In Table 1, n refers to the number of variables, LI are the linear inequalities, NE are the nonlinear equalities, and NI are the nonlinear inequalities.

In all the experiments reported in this paper, our genetic algorithm used binary representation, stochastic remainder selection, one-point crossover (crossover rate of 0.75), and uniform mutation (mutation rate of 0.05). We used a population size of 90 and ran our genetic algorithm for 5000 generations (i.e., we performed 450000 fitness function evaluations), except for the first example in

Test Function	n	Type of f	ρ	LI	NE	NI
1	5	quadratic	52.1230%	0	0	6
2	2	cubic	0.0066%	0	0	2
3	3	quadratic	0.0018%	0	0	2

Table 1: Computation of ρ for the test functions selected.

		Best Result		Mean Result		Worst Result	
TF	**optimal**	**immune**	**KM**	**immune**	**KM**	**immune**	**KM**
1	-30665.539	-30664.8	-30664.5	-30632.4	-30655.3	-30493.7	-30645.9
2	-6961.814	-6961.7	-6952.1	-6950.7	-6342.6	-6819.0	-5473.9
3	1.000	1.000	0.999999857	1.000	0.999134613	1.000	0.991950498

Table 2: Comparison of the results for the three test functions selected.

which the maximum number of generations was set to 10000 (i.e., 900000 fitness function evaluations).

The summary of results is shown in Table 2. We compared our results with the homomorphous mappings of Koziel & Michalewicz (KM) [16], which is the best constraint-handling technique known to date. Koziel & Michalewicz [16] performed $70 \times 20000 = 1400000$ fitness function evaluations to solve each of the examples previously described.

For the first example, the best solution known is the global optimum. It can be seen that for our genetic algorithm, we adopted an encoding with six digits after the decimal point. We used a value of σ of one third of the number of antigens in the population. Two other choices of σ were also tried (e.g., we used σ equals to the number of antibodies or the number of antigens depending on the number of feasible individuals in the population). However, our experiments indicated that the performance of the algorithm was not significantly affected by the choice of either of these values of σ.

7 Conclusions and Future Work

We have presented a constraint-handling approach based on emulations of the immune system (particularly, using the negative selection model) that was incorporated into a genetic algorithm used for global optimization. The approach seems to be competitive with state-of-the-art constraint-handling techniques, since it produces reasonably good results at a relatively low computational cost. However, more work is necessary to validate the sensitivity of the algorithm to its parameters (i.e., a more detailed statistical analysis over a large set of test functions). Furthermore, we are interested in extending this algorithm to deal with constrained multiobjective optimization problems [1].

8 Acknowledgments

The first author acknowledges partial support from CONACyT through the NSF-CONACyT project number 32999-A and from CINVESTAV through project JIRA'2001/08. The second author acknowledges support from CONACyT through a scholarship to pursue graduate studies in Computer Science at the Sección de Computación of the Electrical Engineering Department at CINVESTAV-IPN.

References

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

[2] Carlos A. Coello Coello and Alan D. Christiansen. A Simple Genetic Algorithm for the design of reinforced concrete beams. *Engineering with Computers*, 13(4):185–196, 1997.

[3] Dipankar Dasgupta and Nii Attoh-Okine. Immunity-Based Systems: A Survey. In *IEEE International Conference on Systems, Man and Cybernetics*, Orlando, Florida, October 12-15 1997.

[4] V. Detours and A. S. Perelson. Explaining High Alloreactivity as a Quantitative Consequence of Affinity-Driven Thymocyte Selection. *Proc. Natl. Acad. Sci.*, 22:5153–5158, 1999.

[5] J. D. Farmer, N. H. Packard, and A. S. Perelson. The Immune System, Adaptation, and Machine Learning. *Physica D*, 22:187–204, 1986.

[6] S. Forrest, B. Javornik, R. Smith, and A.S. Perelson. Using genetic algorithms to explore pattern recognition in the immune system. *Evolutionary Computation*, 1(3):191–211, 1993.

[7] S. Forrest, A.S. Perelson, L. Allen, and R. Cherukuri. Self-nonself discrimination in a computer. In *IEEE Symposium on Research in Security and Privacy*, pages 202–212, Oakland, CA, May 16-18 1994. Springer-Verlag.

[8] Stephanie Forrest 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.

[9] Stephanie Forrest 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.

[10] Steven A. Frank. *The Design of Natural and Artificial Adaptative Systems.* Academic Press, New York, 1996.

[11] David E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison-Wesley Publishing Co., Reading, Massachusetts, 1989.

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

[13] John E. Hunt 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.

[14] A. Ishiguru, Y. Watanable, and Y. Uchikawa. Fault Diagnosis of Plant Systems Using Immune Networks. In *Proceedings of the 1994 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI'94)*, Las Vegas, October 2-5 1994.

[15] N. K. Jerne. The immune system. *Scientific American*, 229(1):52–60, 1973.

[16] Slawomir Koziel and Zbigniew Michalewicz. Evolutionary Algorithms, Homomorphous Mappings, and Constrained Parameter Optimization. *Evolutionary Computation*, 7(1):19–44, 1999.

[17] Zbigniew Michalewicz, Dipankar Dasgupta, R. Le Riche, and Marc Schoenauer. Evolutionary algorithms for constrained engineering problems. *Computers & Industrial Engineering Journal*, 30(4):851–870, September 1996.

[18] Zbigniew Michalewicz and Marc Schoenauer. Evolutionary Algorithms for Constrained Parameter Optimization Problems. *Evolutionary Computation*, 4(1):1–32, 1996.

[19] Ronald R. Mohler, Carlo Bruni, and Alberto Gandolfi. A System Approach to Immunology. In *Proceedings of the IEEE*, volume 68, pages 964–990, 1980.

[20] Alan S. Perelson. Immune network theory. *Immunological Reviews*, 10:5–36, 1989.

[21] Jon T. Richardson, Mark R. Palmer, Gunar Liepins, and Mike Hilliard. Some guidelines for genetic algorithms with penalty functions. In J. David Schaffer, editor, *Proceedings of the Third International Conference on Genetic Algorithms*, pages 191–197, George Mason University, 1989. Morgan Kaufmann Publishers.

[22] Alice E. Smith and David M. Tate. Genetic Optimization Using a Penalty Function. In Stephanie Forrest, editor, *Proceedings of the Fifth International Conference on Genetic Algorithms*, pages 499–503, San Mateo, California, July 1993. University of Illinois at Urbana-Champaign, Morgan Kaufmann Publishers.

[23] Derek Smith. Towards a Model of Associative Recall in Immunological Memory. Technical Report 94-9, University of New Mexico, Albuquerque, NM, 1994.

[24] R. Smith and A. S. Perelson. Searching for Diverse, Cooperative Populations with Genetic Algorithms. *Evolutionary Computation*, 2(1):127–149, 1993.

[25] Frank T. Vertosick and Robert H. Kelly. Immune network theory: a role for parallel distributed processing? *Immunology*, 66:1–7, 1989.

[26] Richard G. Weinand. Somatic mutation, affinity maturation and antibody repertoire: A computer model. *Journal of Theoretical Biology*, 143(3):343–382, 1990.