An Immune Algorithm for Protein Structure Prediction on Lattice Models

Vincenzo Cutello, Giuseppe Nicosia, *Member, IEEE*, Mario Pavone, and Jonathan Timmis, *Member, IEEE*

***Abstract*—We present an immune algorithm (IA) inspired by the clonal selection principle, which has been designed for the protein structure prediction problem (PSP). The proposed IA employs two special mutation operators, hypermutation and hypermacromutation to allow effective searching, and an aging mechanism which is a new immune inspired operator that is devised to enforce diversity in the population during evolution.**

When cast as an optimization problem, the PSP can be seen as discovering a protein conformation with minimal energy. The proposed IA was tested on well-known PSP lattice models, the HP model in two-dimensional and three-dimensional square lattices', and the functional model protein, which is a more realistic biological model.

Our experimental results demonstrate that the proposed IA is very competitive with the existing state-of-art algorithms for the PSP on lattice models.

***Index Terms*—Aging operator, clonal selection algorithms, functional model proteins, hypermacromutation operator, hypermutation operator, immune algorithms (IAs), protein structure prediction problem, two-dimensional HP model, three-dimensional HP model.**

I. Introduction

ARTIFICIAL Immune Systems (AISs) represent a field of biologically inspired computing that attempts to exploit theories, principles, and concepts of modern immunology to design immune system-based applications in science and engineering [1]–[3]. One role of the immune system (IS) is to protect the host organism against attacks from antigens (i.e., viruses and bacterias) and eliminate those cells that have been "infected." The IS provides an excellent example of a bottom up intelligent strategy [4], through which adaptation operates at the local level of cells and molecules, and useful behavior emerges at the global level: this is exemplified by the immune humoral and cellular responses.

AISs are proving to be a very general and applicable form of bio-inspired computing. A great deal of work has gone into developing algorithms that extrapolate basic immune processes such as clonal selection, negative and positive selection, danger theory, and immune networks [2]. To date, AIS have been applied to areas such as machine learning [5], [6], optimization [7]–[9], bioinformatics [9]–[11], robotic systems [12]–[14], decision support systems [15], network intrusion detection [16], [17], combinatorial optimization [18], [19], scheduling [20], anomaly detection [21], fault diagnosis [22], [23], computer security [24], data analysis [10], [25], [26], virus detection [27], and many other areas [1]–[3], [28]. The field of AIS appears not only to be a powerful computing paradigm, but potentially a prominent apparatus for improving the understanding of biological data and systems [9], [29].

When one designs any computational solution, the nature of the problem space should always be taken into account. This is especially true in an emerging area such as AIS, and we must avoid the *one size fits all* attitude that the authors of [30] warn us against. With this in mind, and in the context of the framework for AIS presented in [2], we introduce an immune algorithm (IA) based on the clonal selection principle [9]. We employ a new aging operator and specific mutation operators tailored for the *protein structure prediction problem* (PSP) in the HP model for two-dimensional (2-D) and three-dimensional (3-D) lattices, and in the functional model proteins. Given the primary sequence of a protein, the protein structure prediction problem requires the identification of its native (tertiary) conformation with minimum energy; while the protein folding problem requires information about the possible pathways to folding and unfolding. Since a protein's structure determines its biological function, it is very important to be able to predict the final spatial conformation of the proteins. This paper is concerned only with the static aspect, that is, how to predict the folded tertiary structure of a protein, given its sequence of amino acids through the use of lattice models.

This paper is structured as follows: Section II describes the protein structure prediction problem in Dill's model and in the functional model protein; Section III presents the IA, inspired by the clonal selection theory, for the protein structure prediction problem; Section IV details the characteristic dynamics of the implemented IA using an aging process; Section V describes the technique used to partition the landscape of the PSP, and the application of the aging process and memory B cells to improve the overall performance of the algorithm; Section VI reports the results for the 2-D HP model; Section VI-A describes previous related works, and draws comparisons between these and the proposed IA for the 2-D HP model; Section VII presents results for the 3-D HP model; Section VIII presents the results obtained for the functional model protein; Section IX provides a brief comparison between the IA and other biologically inspired algorithms; finally, concluding remarks are presented in Section X.

II. Lattice Models for the PSP

There are essentially five approaches to modeling the PSP: molecular dynamics [31], Monte Carlo methods [32], statistical mechanical models [33], [34], probabilistic road map-based

Manuscript received August 9, 2005; revised January 3, 2006.

V. Cutello, G. Nicosia, and M. Pavone are with the Department of Mathematics and Computer Science, University of Catania, 95125 Catania, Italy (e-mail: vctl@dmi.unict.it; nicosia@dmi.unict.it; mpavone@dmi.unict.it).

J. Timmis is with the Department of Computer Science and the Department of Electronics, University of York, Heslington, York YO10 5DD, U.K. (e-mail: jt517@ohm.york.ac.uk).

Digital Object Identifier 10.1109/TEVC.2006.880328

[35], [36], and lattice models [37], [38]. The first two techniques have been used to analyze the number and the characteristics of folding pathways; the second two techniques are useful tools for studying the folding landscape, while the final technique, whilst having a fundamental theoretical relevance, cannot be applied directly to real proteins. In this paper, we focus on lattice models, in particular, we use the well-known Dill's lattice model, the HP model [39], and the "shifted" HP model (also called functional model proteins) [40].

The HP model takes into account the *hydrophobic interaction* as the main driving force in protein folding. The HP model involves *attraction—interaction only*. The functional model proteins, unlike Dill's model, has a unique native fold with an energy gap between the native and the first excited state; the native state is not maximally compact, and thus presents cavities or potential *binding sites*, a key biological property required in order to investigate ligand binding. To include these properties, the functional model has *both attractive and repulsive interactions*.

A. The Dill's Model

Proteins are sequences of amino acids. In the standard Dill's model, each amino acid is represented as a bead, and connecting bonds are represented as lines. In this approach, the protein is composed of a specific sequence of only two types of beads, H (bead-Hydrophobic/non-polar) or P (bead-hydrophilic/Polar); that is, the 20 amino acids can be divided into two classes: H and P. This is usually called the HP model (or Dill model) [39], where the label P is used to represent hydrophilic amino acids because those amino acids are also polar. We reduce the alphabet from 20 characters to 2, where our protein sequences take the form of strings belonging to the alphabet $\{H, P\}^+$. Hydrophobic amino acids tend to come together to form a compact core that excludes water. Due to the fact that the environment inside cells is aqueous (primarily water), these hydrophobic amino acids tend to be on the inside of a protein, rather than on its surface. Hydrophobicity is one of the key factors that determines how the chain of amino acids will fold up into an active protein.

The whole conformation is embedded in a two or 3-D lattice, which simply divides space into amino acid-sized units. Bond angles only have limited discrete values, dictated by the structure of the lattice (for instance, square, triangular, or cubic [41], [42]). A lattice site may either be empty or contain one bead. In particular, on a 2-D square lattice, the HP model represents proteins as 2-D *self-avoiding walk chains* of ℓ beads on the lattice, i.e., two beads cannot occupy the same site of the lattice, and each bead occupies only one lattice site connected to its chain neighbors.

1) Protein Energy: For each conformation, one can evaluate the value of the energy function: this allows for the modeling of free energies of protein folds. The simplest form of energy function counts the number of H-H-contacts. Each H-H topological contact has energy value ϵ, while all other contact interaction types (H-P, P-H, P-P) have energy value δ. Two amino acids create H-H-contact if they are topological neighbors and they are not connected by a bond. The goal is to find a conformation with the lowest energy. In the HP model in general, the residues' interactions can be defined as follows: $e_{HH} = -|\epsilon|$ and $e_{HP} = e_{PH} = e_{PP} = \delta$. When $\epsilon = -1$ and $\delta = 0$, we have the typical interaction energy matrix for the standard HP model [39]; while for $\epsilon = -2$ and $\delta = 1$, we have the interaction energy matrix for the shifted HP model [43]. For the Dill's model, the native conformation is the one that maximizes the number of contacts H-H, i.e., the one that minimizes the free energy function.

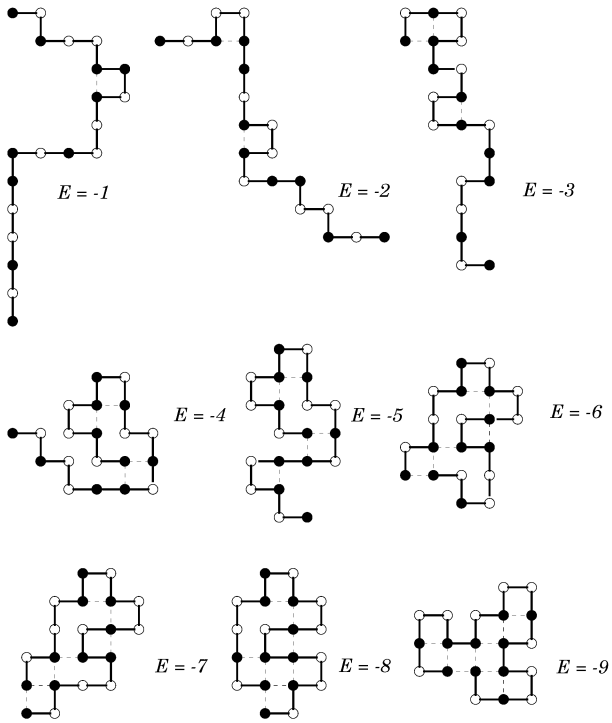

Fig. 1. Convergence process for the protein sequence No. 1, at different energy levels.

Regarding the functional model proteins (described in Section VIII), in order to find binding pockets and the required energy gap, the native conformation finds a tradeoff between the number of H-H contacts (i.e., the *attractive force*) and the non H-H contacts (i.e., the *repulsive forces*).

Fig. 1 shows snapshots of the IA during the convergence process, when applied to the Protein sequence *No. 1* (see Table I) at different energy levels: from poor conformations to the native conformation with Energy -9 (the H-H contacts are represented by dotted lines, and the hydrophobic residues by black circles).

2) 2-D Square Lattice Standard HP Benchmarks: For our experiments, we used the first nine instances of the *Tortilla 2-D HP Benchmarks*[1] (the first eight sequences are taken from [44], sequence 9 is taken from [45], the last three instances are taken from [41]) to test the searching capability of the designed IA. In Table I, E^* is the optimal or best-known energy value, H_i, P_i e $(\ldots)_i$ indicate i repetitions of the relative symbol or subsequence.

The 12 chosen HP instances are standard benchmarks used to test the searching ability of heuristics methods and blind search algorithms. These instances have been tested on more than 20 different algorithms (see Section VI-A and Table VII). Analyzing the HP model is very interesting and challenging for

[1] http://www.cs.sandia.gov/tech_reports/compbio/tortilla-hp-benchmarks.html.

TABLE I
2-D SQUARE LATTICE STANDARD HP BENCHMARKS FROM [41] AND [44]

No.	ℓ	*Protein Sequence*	E^*
1	20	$hphp_2h_2php_2hph_2p_2hph$	-9
2	24	$h_2p_2(hp_2)_6h_2$	-9
3	25	$p_2hp_2(h_2p_4)_3h_2$	-8
4	36	$p_3h_2p_2h_2p_5h_7p_2h_2p_4h_2p_2hp_2$	-14
5	48	$p_2h(p_2h_2)_2p_5h_{10}p_6(h_2p_2)_2hp_2h_5$	-23
6	50	$h_2(ph)_3ph_4p(hp_3)_2hp_4(hp_3)_2hph_4(ph)_3ph_2$	-21
7	60	$p_2h_3ph_8p_3h_{10}php_3h_{12}p_4h_6ph_2php$	-36
8	64	$h_{12}(ph)_2(p_2h_2)_2p_2h(p_2h_2)_2p_2h(p_2h_2)_2p_2(hp)_2h_{12}$	-42
9	20	$h_3p_2(hp)_2hp_2(hp)_2hp_2h$	-10
10	18	$php_2hph_3ph_2ph$	-9
11	18	$hphph_3p_3h_4p_2h_2$	-8
12	18	$h_2p_5h_2p_3hp_3hp$	-4

computer scientists, but is considered unsatisfactory for many biologists. Whilst proteins fold in nature in a matter of seconds, computational biologists have found that folding proteins to their minimum energy conformations is an unsolved optimization problem. The PSP on the HP model has been shown to be NP complete for 2-D [46] (NP-hardness is shown by a reduction from an interesting variation of the planar Hamilton cycle problem), and 3-D lattices [47] (NP-hardness is shown by a reduction from a variation of the bin packing problem).

B. The Functional Model Proteins

In the HP model, the interaction between two hydrophobic residues is $-|\epsilon|$, and zero for the other possible pairs (H-P, P-H, and P-P), that is, the HP model involves *one attraction interaction* (H with H) and *three neutral interactions* (H with P, P with P), so that the energy matrix of the HP model may be written

$$\mathbf{E} = \begin{pmatrix} -1 & 0 \\ 0 & 0 \end{pmatrix}. \tag{1}$$

There are many other folding codes, i.e., the number of different types of residues and the matrix of energies describing the interactions between different kinds of residues. One important folding code is the "shifted" HP model (or functional model proteins) [40]. This model has native folds that are not maximally compact, and presents cavities or potential *binding sites* which is a key biological property required in order to investigate ligand binding. To include these properties, the shifted HP model has two bead types, and both attractive and repulsive interaction. Thus, the shifted HP energy matrix is

$$\mathbf{E} = \begin{pmatrix} -2 & +1 \\ +1 & +1 \end{pmatrix}. \tag{2}$$

The presence of binding sites in the shifted HP model allows us to briefly discuss the biological relevance of the model. Such sites support a significant number of proteins that can be classified as functional, with the ground states having lower degeneracies and more cooperative folding than the regular HP model [48].

C. The Conformational Space Into Lattice

To embed a hydrophobic pattern $s \in \{H, P\}^\ell$ into a lattice, we have the following three methods [49].

1) *Cartesian Coordinate*: The position of residues is specified independently from other residues.
2) *Internal Coordinate*: The position of each residue depends upon its predecessor residues in the sequence. There are two types of internal coordinate: *absolute directions* where the residue directions are relative to the axes defined by the lattice, and *relative directions* where the residue directions are relative to the direction of the previous move.
3) *Distance Matrix*: The location of a given residue is computed by means of its distance matrix.

Krasnogor *et al.* [49] performed an exhaustive comparative study using evolutionary algorithms (EAs) with relative and absolute directions. The experimental results show that relative directions almost always outperform absolute directions over square and cubic lattice, while absolute directions have better performances when facing triangular lattices. Experimental evidence suggests internal coordinates with relative directions should be used. However, in general, it is difficult to assess the effectiveness of direction encoding on an EA's performance.

III. THE IMMUNE ALGORITHM

A. The Clonal Selection Principle

The theory of clonal selection [50], suggests that B and T lymphocytes that are able to recognize the antigen, will start to proliferate by cloning upon recognition of such antigen. When a B cell is activated by binding an antigen (and a second signal is received from T lymphocytes), many clones are produced in response, via a process called *clonal expansion*. The resulting cells can undergo *somatic hypermutation*, creating offspring B cells with mutated receptors. The higher the affinity of a B cell to the available antigens, the more likely it will clone. This results in a Darwinian process of variation and selection, called *affinity maturation*. The increase in size of these populations couples with the production of cells with longer than expected *lifetimes*, assuring the organism a higher specific responsiveness to that antigenic attack in the future. This gives rise to *immunological memory* which is demonstrated by the fact that, when the host is first exposed to the antigen, a *primary* response is initiated; in this phase the antigen is recognized and immune memory is developed. When the same antigen is encountered in the future, a *secondary* immune response is initiated. This results from the stimulation of cells already specialized and present as memory cells: a rapid and more abundant production of antibodies is observed. The secondary response can be elicited from any antigen that is similar, although not identical, to the original one that established the memory. This is known as cross-reactivity.

TABLE II
PSEUDOCODE OF THE IA

Immune Algorithm$(\ell, d, dup, \tau_B, c)$
$Nc := d * dup$;
$t := 0$;
$P^{(t)} :=$ Initial_Pop();
Evaluate$(P^{(0)})$;
while ($\neg$ Termination_Condition()) **do**
  $P^{(clo)} :=$ Cloning $(P^{(t)}, Nc)$;
  $P^{(hyp)} :=$ Hypermutation $(P^{(clo)}, c, \ell)$;
  Evaluate $(P^{(hyp)})$;
  $P^{(macro)} :=$ Hypermacromutation (P^{clo});
  Evaluate $(P^{(macro)})$;
  $(P_a^{(t)}, P_a^{(hyp)}, P_a^{(macro)}) :=$ Aging$(P^{(t)}, P^{(hyp)}, P^{(macro)}, \tau_B)$;
  $P^{(t+1)} := (\mu + \lambda)$-Selection $(P_a^{(t)}, P_a^{(hyp)}, P_a^{(macro)})$;
  $t := t + 1$;
end_while

B. The Clonal Selection Algorithm

The proposed IA (see Table II) employs two entity types: antigens (Ag) and B cells. The Ag models the hydrophobic-pattern of the given protein, that is a sequence $s \in \{H, P\}^{\ell}$, where ℓ is the protein length, i.e., the number of amino acid in the protein sequence. The B cell population, $P^{(t)}$, represents a set of candidate solutions in the current fitness landscape at each generation t. The B cell, or B cell receptor, is a sequence of directions $r \in \{F, L, R\}^{\ell-1}$ (with L = Left, R = Right and F = Forward), where each r_i, with $i = 2, \ldots, \ell - 1$, is a *relative direction* [49] with respect to the previous direction (r_{i-1}) (i.e., there are $\ell-2$ relative directions) and r_1 the nonrelative direction. Hence, we obtain an overall sequence r of length $\ell - 1$. The sequence r specifies a 2-D conformation which is suitable for computing the energy value of the hydrophobic-pattern of the given protein.

At each generation t, there is a B cell population $P^{(t)}$ of size d. The initial population, time $t = 0$, is randomly generated in such a way that each B cell of $P^{(0)}$, is a *self-avoiding* conformation. There are two main functions within the algorithm.

Evaluate(P) which computes the fitness function value of each B cell $\vec{x} \in P^{(t)}$; hence $f(\vec{x}) = -e$ is the energy of conformation coded in the B cell receptor $\vec{x}$; and *Termination_Condition()* which returns true if a solution is found, or a maximum number of fitness function evaluations $(T_{\max})$ is reached.

The implemented IA, like all IAs based on the clonal selection principle outlined above, is characterized by cloning of B cells with higher antigenic affinity, affinity maturation, and hypermutation of offspring B cells. Within our approach, we employ three immune operators: cloning, hypermutation and aging, and a standard evolutionary operator: the $(\mu+\lambda)$-selection operator.

1) Static Cloning Operator: The cloning operator [4], [18], simply clones each B cell *dup* times producing an intermediate population $P^{(clo)}$ of size $d \times dup = Nc$. Throughout this paper, we will refer to this as *static cloning operator*, as opposed to a *proportional cloning operator* (used in the pattern recognition version of CLONALG [8]), which clones B cells proportionally to their antigenic affinities. Experimental results for PSP using such an operator (not shown in this paper), show a frequent premature convergence during the population evolution. In fact, proportional cloning allows B cells with high affinity values to survive for many more generations, and the process can easily become trapped in local minima.

2) Hypermutation Operators: The hypermutation operators act on the B cell receptor of the clone population $P^{(clo)}$. The number of mutations M is determined by a specific function, *mutation potential*, with their being several mutation potentials in existence [51]. In the same research paper, some significant hypermutation operators are discussed and quantitatively compared with respect to their success rate and computational cost. The authors of the paper investigated the searching capability of the IAs based on clonal selection principle using static, proportional and inversely proportional hypermutation operators and hypermacromutation operator. Analyzing the parameter surface for each variation operator and the performance on a complex "toy problem," the trap functions, and the 2-D HP model, clarifies that few different and useful hypermutation operators exist, namely: inversely proportional hypermutation, static hypermutation, and hypermacromutation operators. It appears that making use of inversely proportional hypermutation and hypermacromutation can contribute to finding the best experimental results for the 2-D HP model. As a consequence of these results, we implemented the IA presented in this paper, with inversely proportional hypermutation operator and a hypermacromutation operator. The hypermutation and the hypermacromutation operators mutate the B cell receptors using different mutation potentials.

If during the mutation process, a constructive mutation occurs, the mutation procedure will move on to the next B cell. We call such an event: *Stop at the first constructive mutation* (FCM). We adopted such a mechanism to slow down (premature) convergence, thus allowing a more detailed search through the landscape. A different policy would make use of $M-$mutations (M-*mut*), where the mutation procedure performs all M mutations determined by the potential for the current B cell. With this policy, however, and for the problems which are faced in this paper, the implemented IA did not provide good results [51].

a) Inversely Proportional Hypermutation: The inversely proportional hypermutation operator, makes mutations inversely proportional to the fitness value. In particular, at each generation t, the operator will perform at most the following mutations:

$$M_i\left(f(\vec{x})\right) = \begin{cases} \left(\left(1 - \frac{E^*}{-1}\right) \times \beta) + \beta\right), & \text{if } f(\vec{x}) = 0 \\ \left(\left(1 - \frac{E^*}{f(\vec{x})}\right) \times \beta)\right), & \text{if } f(\vec{x}) > 0 \end{cases} \quad (3)$$

with $\beta = c \times \ell$, and E^* the current best fitness value or the best-known value. In this case, $M_i(f(\vec{x}))$ has the shape of an hyperbola branch.

In [51], the hypermutation operators obtained by varying the parameter c, were thoroughly tested. Studying the parameter surfaces of the trap functions and the PSP, the authors discovered that for the hypermutation operator, inversely proportional to the fitness function value [modeled by (3)], the best values

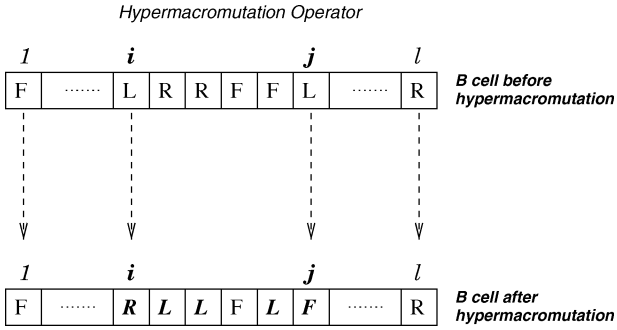

Fig. 2. Example of the hypermacromutation operator applied in the range $[i, j]$ (in bold face the values successfully mutated).

for the parameter c are located in the range $0.4, \ldots, 0.7$. In particular, for the sequences 1, 2, 3, 4, and 12 the best value for c is 0.4; for the sequences 5, 6, 9, 10, and 11 the best value is $c = 0.5$; for the sequence 7 the best value is $c = 0.6$; and for the sequence 8 the value is $c = 0.7$.

b) Hypermacromutation Operator: For the hypermacromutation operator [52] (previously introduced in [53] as "*macromutation operator*"), the number of mutations is determined by a simple random process which does not use functions depending upon constant parameters. Attempts are made to mutate each B cell receptor M times, whilst maintaining the self-avoiding property. The number of mutations M is at most $M_m(\vec{x}) = j - i + 1$, in the range $[i, j]$, with i and j being two random integers such that $(i + 1) \leq j \leq \ell$ (see Fig. 2). The number of mutations is independent from the fitness function f and any other parameter. The hypermacromutation operator for each B cell receptor, randomly selects a perturbation direction, either from left to right $(k = i, \ldots, j)$ or from right to left $(k = j, \ldots, i)$.

In general, the mutation operators perturb the B cells population $P^{(clo)}$, generating the new populations $P^{(hyp)}$ and $P^{(macro)}$, respectively. Each B cell is a feasible candidate solution of the HP model (for simplicity in 2-D using relative encoding, it is straightforward to extend to 3-D using other encoding schemes), making a self-avoiding walk chain on the lattice, $R = < r_1, r_2, \ldots, r_{\ell-1} > \in \{L, R, F\}^{\ell-1}$. Hence, given a protein conformation sequence R, the mutation operator randomly selects a direction r_k, $(1 \leq k \leq \ell - 1)$, or a subsequence $\bar{R} =< r_i, r_{i+1}, \ldots, r_{j-1}, r_j >$ $(i \geq 1$ and $j \leq \ell - 1)$; then, for each relative direction $D = r_k$, it randomly selects a new direction $D' \neq D \in \{F, L, R\}$. If the new conformation is again self-avoiding, then the operator accepts it, otherwise, the process is repeated using the last direction $D'' \neq D, D' \in \{F, LR\}$.

3) Aging Operator: This operator is designed to generate diversity, in an attempt to avoid getting trapped in local minimum. Although it is an operator inspired by the observation in the immune system that there is an expected mean life for the B cell [54], the aging operator can be thought of as a general problem- and algorithm-independent operator.

The aging process attempts to capitalize on the immunological fact that B cells have a limited life span, and that memory B cells have a longer life span. Starting from this basic observation, the aging operator eliminates old B cells from the populations $P^{(t)}$, $P^{(hyp)}$ and $P^{(macro)}$. The parameter τ_B sets the maximum number of generations allowed for generated B cells to remain in the population. When a B cell is $\tau_B + 1$ old it is erased from the current population, no matter what its fitness value is. We call this strategy, *static pure aging*. During the cloning expansion, a cloned B cell inherits the age of its parent. After the hypermutation phase, a cloned B cell which successfully mutates, i.e., it obtains a better fitness value, will be considered to have age equal to 0. Thus, an equal opportunity is given to each "new genotype" to effectively explore the fitness landscape. We note that for τ_B greater than the maximum number of allowed generations, the IA works essentially without the aging operator. In such a limited case, the algorithm employs a strong elitist selection strategy.

The aging operator is implemented by extending the B cells data structure with a counter age, which is initialized as $age = 0$ at generation $t = 0$ and whenever a cloned B cell is successfully mutated. B cells are selected for survival, only if its life $age \leq \tau_B$. The age of each B cell is incremented by one for each of the d surviving B cells. If the surviving B cells are less than d (the population size), new B cells are randomly created (with $age = 0$) and are added by the *Elitist_Merge function* into the population.

Within the literature there is a similar mechanism of the aging process using evolution strategies (ES) [55], where the authors allow a life span, k, for each parent of a $(\mu+\lambda)$-ES or a (μ, λ)-ES. A parent older than k generations is not considered further in the selection process, leaving the new offspring to enter into the population at the next generation. This mechanism allows a more flexible variation of the selection scheme between the two extreme cases $k = 1$, that is (μ, λ)-ES, and $k = \infty$, that is $(\mu+\lambda)$-ES. As noted by the authors of the above cited paper, this mechanism has not been properly investigated and appears to be a "standalone" research work.

4) $(\mu + \lambda)$-Selection With Birth Phase and No Redundancy: A new population $P^{(t+1)}$, of d B cells, for the next-generation $t + 1$, is obtained by selecting the best B cells which "survived" the aging operator, from the populations $P^{(t)}$, $P^{(hyp)}$ and $P^{(macro)}$. No redundancy is allowed. Thus, each B cell receptor is unique, i.e., each genotype is different from all other genotypes in the current population $P^{(t)}$. If only $d' < d$ B cells survived, then the *Elitist_Merge* function creates $d - d'$ new B cells (*Birth phase*). Hence, the $(\mu+\lambda)$-selection operator (with $\mu = d$ and $\lambda = Nc$, or $\lambda = 2Nc$ if both variation operators are activated) reduces the offspring B cell population (created by cloning and hypermutation operators) of size $\lambda \geq \mu$ to a new parent population of size $\mu = d$. The selection operator identifies the d best elements from the offspring set and the old parent B cells, thus guaranteeing monotonicity in the evolution dynamics.

The properties of each immune operator are relatively well understood: the cloning operator explores the attractor basins and valleys of each candidate solution; the hypermutation operators introduce innovations by *exploring* the current population of B cells; the aging operator creates diversity during the search process. The selection evolutionary operator directs the search process toward promising regions of the fitness landscape and *exploits* the information coded within the current popula-

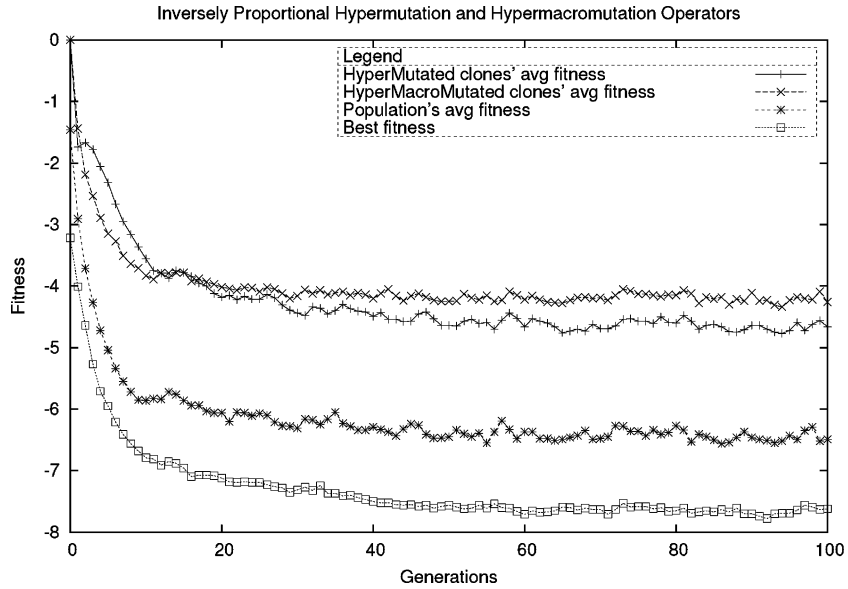

Fig. 3. Average fitness function values of $P^{(hyp)}$, $P^{(macro)}$, $P^{(t)}$ and the best B cell receptor on protein sequence $Seq2$, with parameter values $d = 10$, $dup = 2$, and $\tau_B = 5$.

tion. While selection is a universal problem- and algorithm-independent operator, hypermutation, and in general mutation and crossover operators are specific operators that focus on the structure of the given landscape.

Finally, it is worth noting that representation and mutation operators presented in this section use a discrete coding. They work on an alphabet of three letters $\{L, R, F\}^+$ for relative directions in 2-D square lattices, and on an alphabet of six letters $\{L, R, F, U, D, B\}^+$ (where U = Up, D = Down, and B = Backward) for relative directions in 3-D cubic lattice (see Section VII). Hence, the work described in this paper is applicable only in the context of discrete coding. In Table II, we outline the pseudocode of the proposed IA.

IV. IA Dynamics

In this section, we discuss the characteristic dynamics of the proposed IA. The population size is set to d and the maximum number of fitness function evaluations allowed is set to $T_{\max}$, for minimal values $d = 10$ and $T_{\max} = 10^5$, with $dup = 2$, $c = 0.4$ and $\tau_B = 5$. All the values reported in this section are averaged on 100 independent runs.

In Fig. 3, we show the average fitness values of populations $P^{(hyp)}, P^{(macro)}, P^{(t)}$ and the best fitness value when the IA faces the PSP instance $Seq2$, $hhpphpphpphpphpphpphpphh$, ($\ell = 24$ and minimum energy value known $E^* = -9$).

In this figure, we can see how the four curves decrease, almost monotonically, approximately in the first 20–40 generations, whereas in the remaining generations all four curves reach a steady-state dynamics. The small oscillations are due to the random nature of the overall process governing the hypermutation and the hypermacromutation operators. The higher the average fitness of the hypermutated and hypermacromutated clones, the higher is the diversity in the current population [19].

A. Maximum Information Gain Principle

To analyze the learning process, we use an entropy function, the *Information Gain*. This measures the quantity of information the system discovers during the learning phase [19]. To this end, we define the B cells distribution function $f_m^{(t)}$ as the ratio between the number, B_m^t, of B cells at time t with fitness function value m, and the total number of B cells

$$f_m^{(t)} = \frac{B_m^t}{\sum_{m=0}^{h} B_m^t} = \frac{B_m^t}{d}. \tag{4}$$

It follows that the information gain can be defined as:

$$K(t, t_0) = \sum_m f_m^{(t)} \log\left(f_m^{(t)} / f_m^{(t_0)}\right). \tag{5}$$

The gain is the amount of information the system has already learnt from the given problem instance with respect to the randomly generated initial population $P^{(t=0)}$ (the initial distribution). Once the learning process begins, the information gain increases monotonically until it reaches a final steady state (see Fig. 4). This is consistent with the idea of a *maximum information-gain principle* of the form $(dK/dt) \geq 0$.

Fig. 5 shows the information gain curves for $\tau_B = 1$ and $\tau_B = 5$. For $\tau_B = 5$ the IA learns a greater amount of information than for $\tau_B = 1$, in fact, in the inset plot, the standard deviation obtained with $\tau_B = 1$ is greater than $\tau_B = 5$.

In the x axis log plot 4, it is evident how the information gain is a more informative measure than the mean fitness. The standard deviation, the uncertainty over the population of a given generation (see the inset plot in Fig. 4), decreases quickly in the first ten generations. In fact, the IA converges to the global minimum in this temporal window. After this "threshold" the standard deviation suddenly increases, producing strong oscillations; that is, strong uncertainty regarding the current populations for $t > 10$.

The mean value is essentially constant during all generations. For example, in the first generation, the IA gains more information than in the second, because it generates more constructive mutations. Thus, the population at generation $t = 1$ ex-

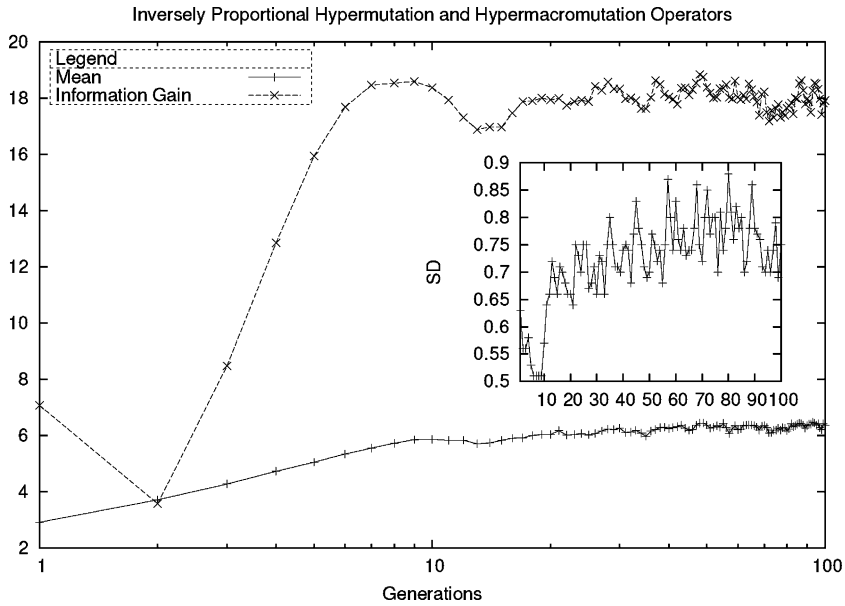

Fig. 4. Information gain, mean fitness versus generations of IA on protein sequence $Seq2$, with parameter values $d = 10$, $dup = 2$, and $\tau_B = 5$. Inset plot displays standard deviation.

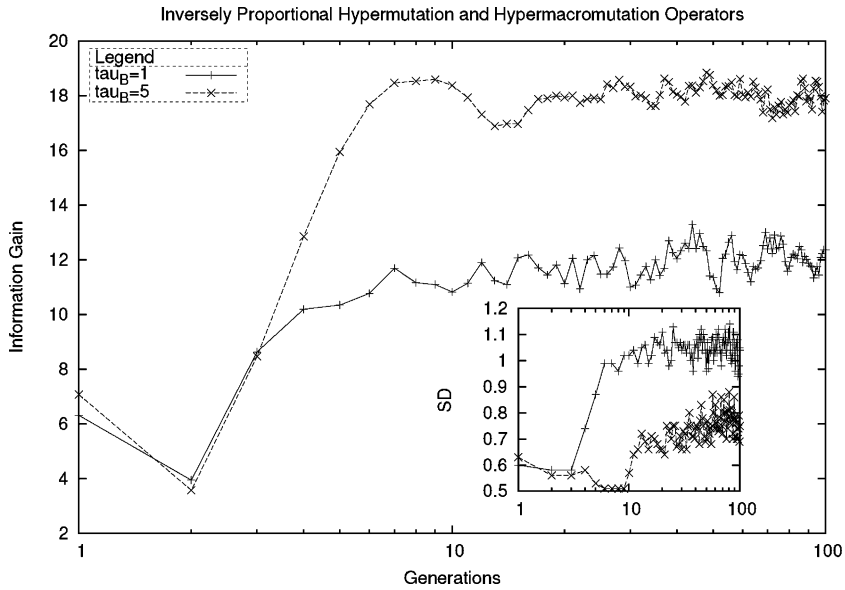

Fig. 5. Information gain and standard deviation versus generations on protein sequence $Seq2$ varying $\tau_B \in \{1, 5\}$.

tracts more informative building blocks than the population at the second generation.

B. Searching Ability of Hypermutation and Hypermacromutation

To understand the searching ability of hypermutation operators when across a range of parameter values, we performed a set of experiments on the PSP instance, $Seq2$. The duplication parameter dup varies from 1 to 10 and the aging parameter τ_B is drawn from the set $\{1, 2, 3, \ldots, 15, 20, 25, 50, 100, 200, \infty\}$. We note that setting the parameter τ_B at a higher value than the possible number of generations is equivalent to giving the B cell an infinite life, in effect, we turn off the aging operator. As a function of dup and τ_B, we show the *success rate* (SR). The 3-D plots obtained are characteristic *parameter surfaces* for the given operator. We set the population size to a minimal value $d = 10$, to emphasize the property of each operator when working with few B cells (points) in the conformational space. This strategy provides a good measure of the "real" performance of single hypermutation procedures. In addition, the *Termination_Condition()* function is allowed at most $T_{\max} = 10^5$ fitness function evaluations and we performed for each value pair of the parameters 100 independent runs. Using the SR and AES (average number of evaluations to solution) values, our experimental protocol has the following three objectives:

1) to plot the characteristic parameter surface of each hypermutation operator;

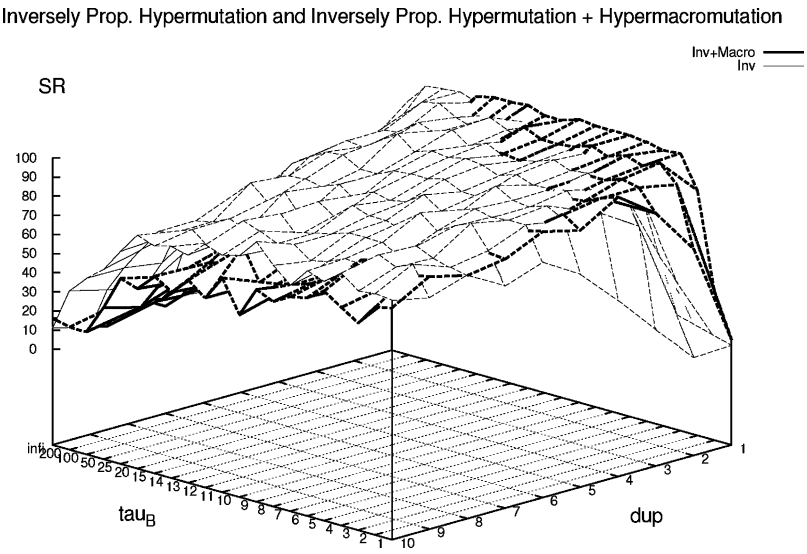

Fig. 6. SR versus parameter values dup and τ_B for the sequence 2. The surface parameters of the combination of inversely proportional hypermutation and hypermacromutation operators.

2) to analyze the joint effects of hypermacromutations and hypermutations;
3) to find the best settings of the parameter values for each operator and for their combination, such that, the best delimited region on the parameter surfaces maximizes the SR value and minimizes the AES value.

Fig. 6 shows the surface parameters of the inversely proportional hypermutation operator and the combination of inversely proportional hypermutation and hypermacromutation operators (surface in bold face): the hypermacromutation extends the region with high SR values, and in particular improves the region where the inversely proportional hypermutation operator alone performed poorly ($\tau_B \in \{1, \ldots, 20\}$ and $dup \in \{1, 2, 3\}$). The highest peak, $\mathrm{SR} = 98$, is obtained for $dup = 2$ and $\tau_B = 3$ with $\mathrm{AES} = 28270, 9$.

V. Partitioning the Funnel by Memory B Cells

Folding energy landscapes are funnel-like, which means that many conformations have high energy and few have low energy. More formally, protein conformations having high free energy (a single point on an energy landscape) have high conformational entropy and states having low free energy (native state and other deep minima) have low conformational entropy. Discrete models have this characteristic, in particular, in the HP model where the energy level is a funnel landscape [44]; for example, the seq. 1 of the benchmarks (see Table I) over 83,779,155 valid conformations have approximately 66×10^6 conformations with high "energy" (i.e., 0 or -1), and only four conformations in the native state with minimal energy $E^* = -9$ (see [44, Table I]). Starting from this simple topological observation, we can deduce that within the funnel landscape the hardest area to search is the middle region: it is typically rugged with many local minima.

For this reason, we partition the funnel landscape in three regions where in the rugged middle region we allocate B cells with a longer life span, called *memory B cells*.

If the native fold has energy value E^*, we have

$$E_{\text{level}} = -E^* + 1$$

energy levels, thus the boundary of the first partition and secondary partition are, respectively

$$B_{fp} = -(E_{\text{level}} * 0.67)$$

and

$$B_{sp} = -(E_{\text{level}} * 0.85).$$

Hence, the B cells with energy in the range

$$0 < E \leq B_{fp}$$

have a life span τ_B, the B cells with energy in the range

$$B_{fp} < E \leq B_{sp}$$

have a longer life span $\tau_{Bmem} > \tau_B$, while all the B cells with energy $E > B_{sp}$ have the same life span of B cells in the first partition τ_B.

Fig. 7 shows the partitioning of the funnel landscape of the PSP problem in three regions. The B cells either belong to the top region or to the bottom region and have life span τ_B, while the memory B cells belong to the middle region and have life span τ_{Bmem}.

Theoretical findings in [56] and experimental results undertaken by ourselves which are not reported in this paper, show that the hardest region to search is the middle. Typically, it is rugged containing many local minima. Therefore, we only apply

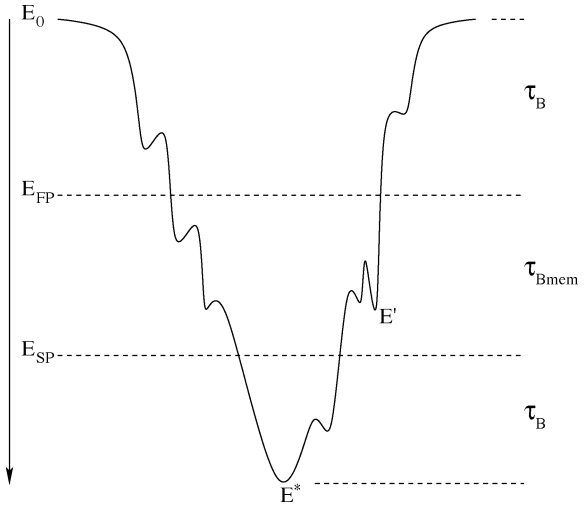

Fig. 7. Partitioning of the funnel landscape using memory B cells.

memory B cells to such a region. Conformations whose energy value is in the middle region, are allowed to mature.

VI. Results for the 2-D HP Model

In this section, we show the overall performances of the IA for the protein structure prediction problem for the 2-D HP model using the well-known tortilla benchmarks (see Table I). The experiments were performed with population size $d = 10$, duplication parameter $dup = 2$, and maximum number of fitness function evaluations $T_{\max} = 10^7$. All the experimental results reported in this section are averaged over 30 independent runs.

Table III shows the number of energy evaluations required on the best run to achieve the optimum, (or the best value found), by a genetic algorithm (GA) and a Monte Carlo approach as reported in [44], the multimeme algorithm [57], the estimation of distribution algorithms (EDA) [58], and the IA for eight instances in the 2-D HP model used in [41] (see [41, Table 6]) and in [59] (see [59, Table 3]) as test bed for the folding algorithms. Gaps in Table III indicate that the particular folding algorithm has not been tested on the respective protein instance. As we can see, the IA obtains the lowest number of energy evaluations, except for the protein instance 4 where the EMC approach reaches a lower number of energy evaluations, and for the protein instance 8, where the EDAs obtain a conformation with 41 topological contacts.

For the sake of completeness, we must note the generality of the MMA algorithm; it has been tested on the HP model and functional model proteins using various lattices, 2-D and 3-D square and triangular lattice, without any modification to the algorithm. It would appear that the algorithm performs robustly across all the models [41], [42], [57].

Tables IV and V show the results obtained by the IA using no memory B cells, and memory B cells. For each protein instance, and for each value of τ_B, the tables report the SR, AES, best found energy value (b.f.), mean and standard deviation values. In bold face, we show the best results reached for each instance of the tortilla benchmarks, sorted first by SR value, then by AES value. For $\mathrm{SR} = 0$, we sort them by the following ordered criteria: best found conformation, mean and standard deviation.

TABLE III
Comparisons of Energy Evaluations for the 2-D HP Model

No.	ℓ	E^*	GA	EMC	MMA	EDA	IA
1	20	-9	-9, 30492	-9, 9374	-9, 14621	-9, 4510	-9, **1925**
2	24	-9	-9, 30491	-9, 6929			-9, **2479**
3	25	-8	-8, 20400	-8, 7202	-8, 18736	-8, 13880	-8, **4212**
4	36	-14	-14, 301339	-14, **12447**	-14, 208233	-14, 113000	-14, 43416
5	48	-23	-22, 126547	-23, 165791	-22, 1155656	-23, 53995	-23, **37269**
6	50	-21	-21, 592887	-21, 74613	-21, 336763	-21, 118000	-21, **18919**
7	60	-35	-34, 208781	-35, 203729		-35, 473500	-35, **161448**
8	64	-42	-37, 187393	-39, 564809		**-41**, 595900	-39, **377404**

By observing the performance of the IA with no memory B cells (Table IV), we can see how the IA using $\tau_B = 5$ does substantially better in 8 out of 12 PSP instances, while the IA using a longer life span, $\tau_B = 10$, does better in the remaining four PSP instances. Both IA versions reach the best-known conformations with maximal success rate, $\mathrm{SR} = 100$, for the PSP instances: 1, 2, 3, 6, 9, 10, 11, 12; for the protein sequences 4 and 5, the IA with $\tau_B = 5$ obtains higher success rate values than with $\tau_B = 10$;, while for the protein sequences 7 and 8, the IA is not able to reach the best-known minima, and it appears to become trapped in local minima.

In Table V, we present the results obtained by the IA using memory B cells, with the following aging values $(\tau_B = 1, \tau_{Bmem} = 5)$, $(\tau_B = 5, \tau_{Bmem} = 10)$. We can see how the IA when using $(\tau_B = 1, \tau_{Bmem} = 5)$, performs better in 8 out of 12 PSP instances, reaches (for the first time) $\mathrm{SR} = 100$ for the protein sequence No. 4 and increases the success rate value for the protein sequence No. 5 obtaining $\mathrm{SR} = 56.67$.

When comparing the results obtained by the IA with memory B cells (Table V) and without memory B cells (Table IV) we are able to see that the IA with memory B cells with aging values $(\tau_B = 1, \tau_{Bmem} = 5)$, outperforms the IA without memory B cells. In fact, this version reaches the best-known conformations with maximal success rate, $\mathrm{SR} = 100$, for the PSP instances: 1, 2, 3, 4, 6, 9, 10, 11, 12; 9 out of 12 PSP instances; and for the protein sequence No. 5 obtains the highest success rate value, $\mathrm{SR} = 56.67$.

Therefore, with the IA using memory B cells, $\tau_B = 1$ and $\tau_{B_{mem}} = 5$ (see Table V), we find it is able to locate the best-known energy values with maximum success rate on 9 protein instances over 12. For protein sequence 5, IA obtains a $\mathrm{SR} = 56.67$, while for the instances 7 and 8 (the "hard instances"), the IA reaches only suboptimal energy values, respectively, -35, and -39 with high mean, and standard deviation values. To improve these protein instances we include in the IA, a special local search procedure known as the Long Range Move (as defined and used in [59]). This procedure tries to escape a local minimum by unfolding the candidate solutions when they are trapped in a local minimum.

In [59], one of the key features of the improved ant colony optimization method is the Long Range Move (LRM). This

TABLE IV
RESULTS OF THE IA WITH NO MEMORY B CELLS FOR THE 2-D HP MODEL

Protein			$\tau_B = 5$					$\tau_B = 10$				
No.	ℓ	E^*	*SR*	*AES*	*b. f.*	*mean*	σ	*SR*	*AES*	*b. f.*	*mean*	σ
1	20	-9	100	26008.13	-9	-9	0	100	**24274.03**	-9	-9	0
2	24	-9	100	**27923.1**	-9	-9	0	100	47394.47	-9	-9	0
3	25	-8	100	**138035.37**	-8	-8	0	100	138620.9	-8	-8	0
4	36	-14	**93.33**	3332912.9	-14	-13.93	0.25	73.33	3728668.4	-14	-13.73	0.44
5	48	-23	**30**	5071455.3	-23	-22.23	0.56	23.33	2322230	-23	-22.20	0.48
6	50	-21	100	**826399.57**	-21	-21	0	100	1069777.77	-21	-21	0
7	60	-36	0	//	-35	**-34.17**	**0.37**	0	//	-35	-33.87	0.5
8	64	-42	0	//	-39	-36.83	1.27	0	//	-39	**-37.83**	1.49
9	20	-10	100	**21488.67**	-10	-10	0	100	44694.13	-10	-10	0
10	18	-9	100	92524.2	-9	-9	0	100	**80802.77**	-9	-9	0
11	18	-8	100	**56949.7**	-8	-8	0	100	61782.4	-8	-8	0
12	18	-4	100	84239.63	-4	-4	0	100	**68289.6**	-4	-4	0

TABLE V
RESULTS OF THE IA WITH MEMORY B CELLS FOR THE 2-D HP MODEL

Protein			$\tau_B = 1$ and $\tau_{Bmem} = 5$					$\tau_B = 5$ and $\tau_{B_{mem}} = 10$				
No.	ℓ	E^*	*SR*	*AES*	*b. f.*	*mean sol.*	*st. dev.*	*SR*	*AES*	*b. f.*	*mean sol.*	*st. dev.*
1	20	-9	100	23710	-9	-9	0	100	**20352.4**	-9	-9	0
2	24	-9	100	69816.7	-9	-9	0	100	**39959.9**	-9	-9	0
3	25	-8	100	**269513.9**	-8	-8	0	100	282855.7	-8	-8	0
4	36	-14	**100**	2032504	-14	-14	0	73.33	4569496.3	-14	-13.73	0.44
5	48	-23	**56.67**	6403985.3	-23	-22.47	0.67	6.67	4343279	-23	-21.47	0.62
6	50	-21	100	**778906.4**	-21	-21	0	100	1135818.9	-21	-21	0
7	60	-36	0	//	-35	-33.73	0.68	0	//	-35	**-34.5**	**0.5**
8	64	-42	0	//	**-39**	-36.13	1.28	0	//	-38	-35.1	1.25
9	20	-10	100	**18085.8**	-10	-10	0	100	18473.6	-10	-10	0
10	18	-9	100	**69210**	-9	-9	0	100	130342	-9	-9	0
11	18	-8	100	**41724.2**	-8	-8	0	100	50151.2	-8	-8	0
12	18	-4	100	87494.5	-4	-4	0	100	**74426.5**	-4	-4	0

local search, as noted by the authors, mimics the folding process of the real proteins, where a moving residue will typically push its neighbors in the chain to different positions. The first step of the procedure selects a direction in a given conformation uniformly at random. The second step of the procedure randomly changes the direction, and then modifies the directions of the remaining residues probabilistically. The local search starts a "*chain reaction*" that loops until a self-avoiding path condition is held. Practically, this procedure allows a given conformation to fold and unfold moves to escape local minima in the multiple-minima funnel landscape. Since the above cited procedure is obviously time consuming, as in [59], we apply it to the best conformation in the current population of the algorithm.

TABLE VI
RESULTS OF THE IA WITH LRM FOR THE 2-D HP MODEL, PROTEIN INSTANCES 5, 7, AND 8

Protein			$\tau_B = 1$ and $\tau_{Bmem} = 5$				
No.	ℓ	E^*	*SR*	*AES*	*b. f.*	*mean sol.*	*st. dev.*
5	48	-23	20	5870105.2	-23	-22.20	**0.401**
7	60	-36	0	//	-35	**-35**	0
8	64	-42	3.33	8078548	**-42**	-39.3	0.641

In Table VI, we report the results of the IA using the long range move (IA WITH LRM), for the hard instances, protein sequences 5, 7, and 8. By inspecting the table, we note that for instance 5 the IA obtains poor results, while for sequence 7, although the IA does not reach the best-known energy value (-36), the algorithm with the long range move always reaches the suboptimal energy value -35, which is the best result obtained for the IA. It is worthy of note, that the longest protein sequence the IA with LRM reaches the best-known energy value -42 with mean -39.2 and $\sigma = 0.641$.

A. Comparison With State-of-Art Folding Algorithms

In this section, we briefly present related works for the 2-D HP model. We present this here, as it allows us to show comparative results of our proposed approach, and current state-of-art algorithms that are used on this problem.

The first application of EAs on the HP model was in [44]: a GA is employed, and conformations are changed by a mutation operator that follows the conventional Monte Carlo steps (*MCmutation operator*), and by a crossover operator. The authors found that the GA is superior to conventional Monte Carlo methods (MC, LONG MC, and MULTIPLE MC).

In [60], there is an improved version of the simple GA (SG) using a new crossover operator (*systematic crossover*), the SGA-S: it couples the best candidate solutions, tests every possible crossover point, and selects the two best conformations for the next generation. In addition, the authors implemented a new search strategy, the SGA with systematic crossover and pioneer search, SGA-SC-P, which tried to prevent the population from becoming too homogeneous, moving to a new search region every ten generations.

A famous metaheuristic for combinatorial optimization is the *Memetic Algorithm*. Memetic algorithms are EAs that include in the evolutionary cycle a local search procedure [61]. The *Multimeme algorithm* (MMA) [41], [61] is a memetic algorithm that self-adaptively selects from a set of local searchers, which heuristic to use during the search process for different instances. The MMA has been used on different protein structure models with results competitive with other techniques [41].

To improve the performance of the GA, in [62] the authors proposed a hybrid algorithm of GA and tabu search (TS) and novel crossover operator borrowed from the TS. The introduction of the TS approach improves the overall performances of the GA and shows that in all the instances, the hybrid algorithm GTS works better than a GA alone.

Another Monte Carlo method is the EMC [63] that works with populations of candidate solutions which are optimizated by Monte Carlo simulation. This hybrid algorithm found the best-known structure for protein sequence 8, with energy $E^* = -42$.

The contact interactions algorithm, CI, may be regarded as an extension of the standard MC method which is improved by the strategy of cooperativity. The major innovation of the CI approach [45] is that criteria for acceptance of new conformations are not based on the energy of the entire protein, but on the fact that cooling factors associated with each residue define regions of the model protein with higher or lower mobility. Hence, the CI is not a blind general purpose algorithm, it uses a heuristic based on effects of an H-H contact on the mobility of the residues in different portions of a protein. The CI algorithms, with a fixed *starting temperature* of $c_k = 0.3$ (CI, $c_k = 0.3$), or with different starting temperature (CI), proved to be very efficient to localize energy minima.

Among the best folding algorithms there is the *tabu search* strategy [65]. This incorporates problem domain knowledge into the algorithm, for instance the conformational motifs, during the search process for finding low energy conformations.

The core-directed chain growth (CG) [64] is a very efficient algorithm that has found optimal and best-known conformations for protein instances 1–6 and 8. It is an ad hoc heuristic that approximates the hydrophobic core of discrete proteins.

The EDA [58] is a suitable class of nondeterministic search procedures for the HP model. EDA constructs an explicit probability model of the candidate solutions selected and captures relevant interactions among the variables of the given protein instance. The experimental results have proven the effectiveness of the EDA approach to face lattice models for the standard HP benchmarks and the functional model proteins.

The state-of-art algorithm for 2-D HP problem is an improved ant colony optimization (ACO) algorithm, IMPROVED (IACO) [59]. The improvements over the previous ACO algorithm [67], are the following: *long range moves* for chain reconfigurations when the protein conformation is very compact (described in Section VI); *improving ants* that take the global best solution found so far and apply a randomized greedy local search to it; and *selective local search* that performs the critical operation of the local search phase only on promising low energy conformations. Moreover, in [59], the authors report poor performance of a local search procedure (ONLY L), modest performances of an (IMPROVED GA), and good performance of Pruned Enriched Rosenbluth Method (PERM) [66]. The Grassberg and co-workers' algorithm is a Monte Carlo method which is among the best-known algorithms for the 2-D HP model, and is a bias chain growth algorithm. PERM found the best solution for the protein sequence No. 7, $E^* = -36$.

In Table VII, we report the comparisons with the state-of-art algorithms for the 2-D HP model. The reported energy values are the lowest obtained by each method. Gaps in the table indicate that a particular algorithm has not been tested on the respective protein sequence. The shown results suggest that the proposed IA using the aging operator and memory B cells, and the IA with LRM are comparable to and, in many protein instances, outperform the best algorithms.

TABLE VII
IA VERSUS THE STATE-OF-ART ALGORITHMS FOR THE 2-D HP MODEL

Sequence		1	2	3	4	5	6	7	8	9
E^*		-9	-9	-8	-14	-23	-21	-36	-42	-10
IMPROVED ACO	[59]	-9	-9	-8	-14	-23	-21	-36	-42	
IA with LRM		**-9**	**-9**	**-8**	**-14**	**-23**	**-21**	**-35**	**-42**	**-10**
CG	[64]	-9	-9	-8	-14	-23	-21	-35	-42	
Tabu Search	[65]	-9	-9	-8	-14	-23	-21		-42	
EDA	[58]	-9	-9	-8	-14	-23	-21	-35	-41	
CI	[45]	-9	-9	-8	-14	-23	-21	-35	-40	
IA		**-9**	**-9**	**-8**	**-14**	**-23**	**-21**	**-35**	**-39**	**-10**
EMC	[63]	-9	-9	-8	-14	-23	-21	-35	-39	
GTS	[62]	-9	-9	-8	-14	-23	-21	-35	-39	
PERM	[59], [66]	-9	-9	-8	-14	-23	-21	-36	-38	
MMA	[41]	-9		-8	-14	-22	-21	-36	-38	
CI $c_k = 0.3$	[45]	-9	-9	-8	-14	-22	-21	-34	-38	
Improved GA	[59]	-9	-9	-8	-14	-23	-21	-34	-37	
GA	[44]	-9	-9	-8	-14	-22	-21	-34	-37	
SGA-SC-PS	[60]	-9			-14	-23			-37	
ACO	[67]	-9	-9	-8	-14	-23	-21	-34	-32	-10
SGA-SC	[60]	-9			-14	-22			-33	
LONG MC	[44]	-9	-9	-8	-13	-20	-21	-33	-35	
SGA	[60]	-9			-14	-20			-32	
Only LS	[67]	-9	-9	-8	-14	-21	-20	-33	-33	-10
MULTIPLE MC	[44]	-9	-9	-7	-13	-19	-20	-32	-32	
MC	[44]	-8	-8	-7	-12	-18	-19	-31	-31	

VII. RESULTS FOR THE 3-D HP MODEL

The protein structure prediction problem with $d = 2$, and making use of a square lattice, captures the protein folding problem in the 2-D HP model [39]. Analogously for $d = 3$ and using a cubic lattice, we have the 3-D HP model [39].

In the 3-D cubic lattice, each point has six different neighbors and five available locations. We use two different schemes of moves, (absolute and relative directions), to represent and embed a protein in the lattice. The relative and absolute encoding were described in Section III-B: the *residues directions* are relative to the direction of the previous move, whilst in the *absolute directions*, encoding the residues direction is relative to the axes defined by the lattice.

Both for the absolute and relative coding, not all moves provide a feasible conformation. In our work, we force the self-avoidance constraint such that each set of moves will correspond to a feasible sequence (feasible conformation).

TABLE VIII
RESULTS OF THE IA FOR THE 3-D HP MODEL

	Absolute Encoding						Relative Encoding					
	F-EA			IA			F-EA			IA		
Seq.	Best	Mean	σ	Best	Mean	σ	Best	Mean	σ	Best	Mean	σ
1	-11	-10.32	0.61	-11	**-11**	**0**	-11	-9.84	0.86	-11	**-10.90**	**0.32**
2	-13	-10.90	0.98	-13	**-13**	**0**	-11	-10.00	0.87	**-13**	**-12.22**	**0.65**
3	-9	-7.98	0.71	-9	**-9**	**0**	-9	-8.64	0.69	-9	**-8.88**	**0.48**
4	-18	-14.38	1.26	-18	**-16.76**	**1.02**	-18	-13.72	1.41	-18	**-16.08**	**1.02**
5	-25	-20.80	1.61	**-29**	**-25.16**	**0.45**	-28	-18.90	2.08	-28	**-24.82**	**0.71**
6	-23	-20.20	1.50	-23	**-22.60**	**0.40**	-22	-19.06	1.46	**-23**	**-22.08**	**1.43**
7	-39	-34.18	2.31	**-41**	**-39.28**	**0.24**	-38	-32.28	3.09	**-41**	**-39.02**	**0.50**
8	-39	-33.01	2.49	**-42**	**-39.08**	**0.95**	-36	-30.84	2.55	**-42**	**-39.07**	**1.20**

By inspecting the experimental results for all the considered instances, it is worthy to note that the IA (working with feasible solutions) locates the known minimum value. For all instances, the located mean value is lower than the results obtained in [68], where EAs working on feasible-space were employed. For several sequences presented in [69], (as shown in Table VIII), we have found new, best—lowest energy values for 3-D protein sequences 5, 7, and 8 (results reported in bold face in Table VIII).

For our experiments, the IA was set with standard parameter values: $d = 10$, $dup = 2$, as described in [51], B cells have the aging parameter $\tau_B = 5$ and memory B cells $\tau_{B_m} = 10$. For the experimental protocol, we adopted the same values used in [69]: 50 independent runs and a maximum number of evaluations equal to 10^5. In [68], the author does not use the SR and AES values as quality metrics, but the following parameters: Best found solution (Best), mean and standard deviation (σ).

In addition, we designed an IA which made use of a penalty strategy and a repair-based approach as reported in [68], which obtained similar experimental results to [68] (not shown). Such an IA proved to be very efficient for both absolute and relative encoding, and allowed us to find energy minima not found by other EAs working in feasible space and described in literature [68].

VIII. RESULTS FOR THE FUNCTIONAL MODEL PROTEINS

Table IX shows the benchmarks for the functional model proteins into 2-D square lattice [41], [43], the instance number, protein sequence, optimal conformation, and the minimal energy values. Each instance of the benchmarks[2] has a unique native fold conformation minimal energy value, E^*, and an energy gap between E^* and the first excited state (best suboptimal). In Fig. 8, we report the native fold of all the protein sequences, each native fold has at least one binding site, or binding pocket

[2]http://www.cs.nott.ac.uk/ñxk/HP-PDB/2dfmp.html

TABLE IX
2-D SQUARE LATTICE FUNCTIONAL MODEL INSTANCES [43], WHERE EACH PROTEIN SEQUENCE HAS 23 MONOMERS

No.	*Protein Sequence and Optimal Conformation*	E^*
1	$php_2hp_2h_4p_2hp_2hphp_2h_2$ $FRLLRLLRFRFLLRLLRLRLLR$	-20
2	$php_2hp_2h_5p_4hp_2hp_2h$ $RLRRLRRFLLRLFRRFLRRLRR$	-17
3	$hphphph_3p_2hp_3hph_2p_2h_2$ $FRLRLRLRFRRLFRFLRLFRRL$	-16
4	$h_3ph_3p_2h_2p_3hphp_2h_4$ $RRLLFRFLLRFLFLRLRLLFLR$	-20
5	$php_6hph_2phph_4phph$ $RLFFRFRFRLLFRFRRLFLLRL$	-17
6	$h_2php_2hp_4hp_4hp_3h_3$ $FLLRLLRLFLFRFLFLRLFLLR$	-13
7	$phph_2ph_6p_2h_3ph_5$ $FLRFFRRFLRLRLLFFFLFLLR$	-26
8	$hphp_3h_4php_4hphph_3$ $FRLRFRFRFLLRFLFLRLFLLR$	-16
9	$phph_2ph_2ph_2phphphp_5h$ $FFRRLLFRLRLLFLRLRFFLFL$	-15
10	$hphphp_5h_2p_3hphphph_2$ $RLRLRLFLFLFRFLFRLFLFLL$	-14
11	$php_2h_3php_2hph_2php_5h$ $LLRRFRLLRLLRLLRRLRFRFR$	-15

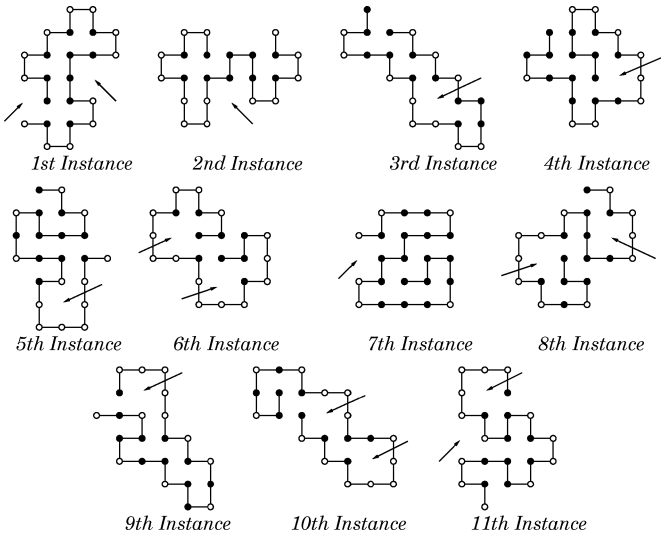

Fig. 8. Native fold for 2-D square lattice functional model instances.

(which are illustrated in figure by arrows pointing to the binding site(s) of each functional model instance).

In the first experiment, we compared the performance of the IA with and without elitist aging (see Table X) using the standard parameter values: $d = 10$, $dup = 2$, $c = 0.4$, $\tau_B = 5$. In this section, all experimental results reported, were obtained by the IA with inversely proportional hypermutation and hypermacromutation operators. All the values are averaged on 30 independent runs.

TABLE X
ELITIST AGING VERSUS PURE AGING IN THE FUNCTIONAL MODEL PROTEINS IN TERMS OF (SUCCESS RATE, AVERAGE NUMBER OF EVALUATIONS TO SOLUTION)

No.	E^*	Elitist Aging	Pure Aging
1	-20	(100, 45563.1)	(**100**, **32847.73**)
2	-17	(100, 40389.47)	(**100**, **17526.73**)
3	-16	(3.33, 1194332)	(**56.67**, **2403985.3**)
4	-20	(100, 168227.83)	(**100**, **128015.1**)
5	-17	(100, 17667.9)	(**100**, **12095.33**)
6	-13	(83.33, 2732830)	(**100**, **332938.5**)
7	-26	(40, 127819.83)	(**100**, **584179.8**)
8	-16	(100, 149415.4)	(**100**, **38262.6**)
9	-15	(10, 377108)	(**100**, **281720.8**)
10	-14	(100, 587811)	(**100**, **104155.4**)
11	-15	(93.33, 174583.82)	(**100**, **27743.7**)

Analogously to the HP model instances, in the functional model proteins, the pure aging procedure outperforms the elitist aging in term of SR and AES on all functional model instances. The IA with pure aging obtains a $\mathrm{SR} = 100$ for all the instances excluding the functional protein sequence 3, where the algorithm reaches $\mathrm{SR} = 56.67$. This confirms the optimal searching ability and diversity generation of the pure aging strategy.

In Table XI, we report the experimental results obtained when using memory B cells. As described previously, we partition the funnel landscape energy levels. For example, sequences 1 and 4 have $E^* = -20$, it follows that there are 21 energy levels, thus all the B cells with energy value E in the range $0 < E \leq -13$ or with energy $E > 17$ will have life span equal to τ_B, while the B cells with energy $-13 < E \leq -17$ will be considered as memory B cells with a life span of τ_{Bmem}. Table XI presents the best experimental results obtained using the aging values $(\tau_B = 3, \tau_{Bmem} = 5)$, $(\tau_B = 4, \tau_{Bmem} = 8)$, and $(\tau_B = 5, \tau_{Bmem} = 10)$; the best values of SR and AES are shown in bold face. From these results, it is clear that $\tau_B = 5$, and $\tau_{Bmem} = 10$ appear to be the optimal choice for partitioning the funnel landscapes of the protein instances.

When comparing the results obtained by the IA, when adopting a pure aging strategy and $\tau_B = 5$ with the IA using memory B cells with $\tau_B = 5$, and $\tau_{Bmem} = 10$, the algorithm performs slightly better without the memory B cells. We believe this is due to the length of the protein sequences of the functional model proteins, which have 23 residues only. Hence, for short protein sequences ($\ell < 36$), the IA without memory B cells performs better than the IA with memory B cells, both for the HP model and for the functional model proteins.

TABLE XI
IA PERFORMANCES USING MEMORY B CELLS IN TERMS OF (SUCCESS RATE, AVERAGE NUMBER OF EVALUATIONS TO SOLUTION) FOR VARIOUS PAIRS OF (τ_B, τ_{Bmem})

No.	E^*	$(3,5)$	$(4,8)$	$(5,10)$
1	-20	(100, 253393)	(100, 60664.8)	(**100**, **38586.63**)
2	-17	(100, 41189.7)	(100, 258387)	(**100**, **28434.9**)
3	-16	(16.67, 31311300)	(36.67, 2448820)	(**43.33**, **2583300.8**)
4	-20	(100, 568485)	(100, 261439.1)	(**100**, **130849**)
5	-17	(**100**, **16726.3**)	(100, 17586.13)	(100, 20834.46)
6	-13	(96.67, 1083238.6)	(100, 711828.33)	(**100**, **483126.76**)
7	-26	(100, 1171346.4)	(100, 936008.9)	(**100**, **588057.5**)
8	-16	(100, 107131)	(100, 54432.7)	(**100**, **42562.53**)
9	-15	(100, 506368)	(**100**, **75273.8**)	(93.33, 907962.4)
10	-14	(100, 226564.87)	(100, 141515.23)	(**100**, **100085.43**)
11	-15	(**100**, **43327.2**)	(100, 82361.1)	(100, 71903.1)

TABLE XII
COMPARISON OF BEST RUNS FOR MMA [41], EDAS [58], AND IA, WITH AND WITHOUT MEMORY B CELLS FOR THE FUNCTIONAL MODEL PROTEINS

No.	E^*	MMA	EDA	IA without memory	IA with memory
1	-20	15170	12650	**3372**	3643
2	-17	61940	2750	**578**	1488
3	-16	132898	**35900**	319007	100234
4	-20	66774	15900	**4955**	20372
5	-17	53600	20950	**1047**	1956
6	-13	32619	5420	**2828**	7482
7	-26	114930	19450	**10061**	37841
8	-16	28425	10350	**1818**	1937
9	-15	25545	4950	**3845**	10399
10	-14	111046	8950	**2847**	3462
11	-15	52005	2950	3176	**1007**

However, for long protein sequences $(\ell \geq 36)$, partitioning the funnel landscape with the memory B cells appears to be a good strategy for effectively searching the rugged landscape in the middle of the funnel.

Finally, in Table XII, we present the number of energy evaluations required by the best run to locate the optimum or a suboptimum energy value. We compare the performances of the IA with and without memory B cells, with the state-of-art algorithms for the functional model proteins: the MMA [41] and the EDAs [58]. Both versions of the IA, outperform the MMA and EDA on all the test bed except for the protein instance 3 where EDA reaches a lower number of energy evaluations. In particular, the IA without memory B cells obtains the best results on 9 instances over 11.

We propose that more effective metrics to assess the overall performances are the success rate values and the average number of evaluations to solution. The number of fitness function evaluations required by the best run for a given instance are less significant for showing the overall performance of the randomized algorithms.

IX. IA VERSUS EAs

It is worth a little time to highlight the contribution of our algorithms to the artificial immune systems discipline. The proposed IA makes use of a new hypermutation operator, the hypermacromutation operator, that extends the region of the parameter surface with high SR values, and in particular improves the region where the inversely proportional hypermutation operator alone, performed poorly. This simple random process, which does not use functions dependent upon constant parameters, improves the overall performance of the IA.

The second innovation of the IA is the aging operator, which is used to generate and maintain diversity in the population. As shown in the plots and in the tables, the aging operator and memory B cells with a longer life span, are the key features of the proposed approach that we feel can be inserted in any EA. In fact, as a selection operator, the aging operator is a general, problem- and algorithm-independent operator.

Obviously, the implemented algorithm can be applied to any other combinatorial and numerical optimization problem apart from the protein structure prediction problem using suitable representations and variation operators [4], [9], [18], [69].

A. IA Versus Other Clonal Selection Algorithms

A well-known clonal selection algorithm in the AIS literature, is CLONALG [8], [69]. This algorithm employs fitness values for proportional cloning, inversely proportional hypermutation, and a birth operator to introduce diversity in the current population along with a mutation rate to flip a bit of a B cell. Extended versions of this algorithm use a threshold value to clone the best cells in the present population.

CLONALG maintains two populations: a population of antigens Ag and a population of antibodies Ab (indicated with $P^{(t)}$). The individual antibody, Ab, and antigen, Ag, are represented by string attributes $m = \{m_L, \ldots, m_1\}$, that is, a point in an L-dimensional real-valued shape space S, $m \in S^L \subseteq \Re^L$. The Ab population is the set of current candidate solutions, and the Ag is the environment to be recognized. The algorithm loops for a predefined maximum number of generations (N_{gen}). In the first step, affinity values (fitness function values) are determined for all $\text{Ab}'s$ in relation to the Ag. Then, it selects n Abs that are to be cloned independently and proportionally to their antigenic affinities, thus generating the clone population $P^{(clo)}$. The higher the affinity, the higher the number of clones generated for each of the n $\text{Ab}'s$ with respect to the following function:

$$N_c = \sum_{i=1}^{n} \lfloor (\beta \times n)/i \rfloor \tag{6}$$

where β is a multiplying factor. Each term of the sum corresponds to the clone size of each Ab. The hypermutation operator performs an affinity maturation process inversely proportional to the fitness values, generating the matured clone population $P^{(hyp)}$. After having computed the antigenic affinity of the population $P^{(hyp)}$, CLONALG randomly creates d new antibodies that will replace the d lowest fit $\mathrm{Ab}'s$ in the current population.

Clearly, both CLONALG and the IA are inspired by the clonal selection principle. Hence, they have many similarities, but there are some significant differences (for a comparative study see [69]). We begin with the common features. Both algorithms employ a population of Ag's to represent the input, whilst the population of immune entities (Ab's or B cells) are the candidate solutions to the given computational problem. Both algorithms use hypermutation operators inversely proportional to the fitness values.

Considering their differences, while in the IA the cloning operator selects all the immune entities for cloning. In CLONALG, it is possible to use a threshold value to clone only the best cells in the current population. In CLONALG, the cloning operator is proportional to the fitness values depending on a multiplying factor (see (6)). However, in the IA, the cloning operator makes use of static cloning: each immune entity will produce dup clones. Hence, CLONALG uses proportional cloning, and the IA uses static cloning. The underlying notion of employing static cloning, is to give each point of the given search space equal opportunity to explore its neighborhood; proportional cloning provides a bias to each point of the search space based on its fitness function value. This bias could be useful or not, depending of course on the computational problem being addressed.

To produce diversity in the population, at each generation CLONALG uses a birth operator which introduces d new immune entities. The IA, however, uses an aging operator modeled by an expected life time parameter τ_B. Additionally, CLONALG uses memory B cells as an implicit "memory mechanism," an archive of the best candidate solutions, while the IA version for the partition of the landscape uses memory B cells with a longer mean life parameter τ_{Bmem}, with τ_{Bmem} greater than the mean life of standard B cells τ_B to allow a search of the rugged regions of the landscape.

The selection scheme used by CLONALG to decide which immune entities will go to the next generation uses elitism, while the IA uses a standard $(\mu+\lambda)$—selection operator without an elitist strategy.

Finally, as termination condition, CLONALG typically uses a fixed number of generations, while the IA can use three termination conditions: a fixed number of generations, a maximum number of fitness function evaluations, or the maximum information-gain principle $(dK/dt) \geq 0$ [19]. Though the last condition does not avoid the possibility of being trapped in local minima solutions, this termination condition measures the quantity of information the IA discovers during the convergence process.

X. Conclusion

This paper has proposed a novel IA for the protein structure prediction problem. Within this paper, we have assessed the overall performance of the IA in terms of solution quality and average number of evaluations to solution (metric, we feel is more robust than run-time values). We have made use of various discrete protein structure models, and have compared the results with the present state-of-art algorithms when applied to each model.

As future work, we intend to tackle the prediction of 3-D structures for actual proteins [70] using the designed IA. As in [70], we plan to consider the parallelization of the folding algorithm in order to reduce execution time and resource expenditure.

The IA uses a new hypermutation operator, the hypermacromutation operator, which does not use functions depending upon constant parameters, and extends the region of the parameter surface with high success rate values. This operator contributes to the overall improvement in terms of performance of the IA.

A second innovation of the IA is the aging operator, which is used to generate and maintain diversity in the population. As demonstrated by our results, the aging operator and memory B cells with a longer life span are the key features of the proposed approach, that we propose could be inserted in any EA.

With regards to the actual computational results on the PSP problem, we found that for short protein sequences $(\ell < 36)$, the IA without memory B cells performs better than the IA with memory B cells both for the HP model and for the functional model proteins. For long protein sequences $(\ell \geq 36)$, partitioning the funnel landscape with the memory B cells is a good strategy to search more effectively the rugged landscape in the middle of the funnel.

Acknowledgment

The authors would like to express their gratitude to the anonymous reviewers for their helpful comments on the manuscript. G. Nicosia would like to thank the Computing Laboratory, University of Kent, Canterbury, U.K., for their kind support.

References

[1] D. Dasgupta, *Artificial Immune Systems and Their Applications*. Berlin, Germany: Springer-Verlag, 1999.

[2] L. N. de Castro and J. Timmis, *Artificial Immune Systems: A New Computational Intelligence Paradigm*. London, U.K.: Springer-Verlag, 2002.

[3] Y. Ishida, *Immunity-Based Systems: A Design Perspective*. Heidelberg, Germany: Springer-Verlag, 2004.

[4] V. Cutello and G. Nicosia, "The clonal selection principle for in silico and in vitro computing," in *Recent Developments in Biologically Inspired Computing*, L. N. de Castro and F. J. von Zuben, Eds. Hershey, PA: Idea Group Publishing, 2004.

[5] J. E. Hunt and D. E. Cooke, "Learning using an artificial immune system," *J. Netw. Comput. Appl.*, vol. 19, pp. 189–212, 1996.

[6] G. Nicosia, F. Castiglione, and S. Motta, "Pattern recognition by primary and secondary response of an artificial immune system," *Theory in Biosciences*, vol. 120, no. 2, pp. 93–106, 2001.

[7] T. Fukuda and M. T. K. Mori, "Parallel search for multimodal function optimization with diversity and learning of immune algorithm," in *Artif. Immune Syst. Their Appl.*, D. Dasgupta, Ed. Berlin, Germany: Springer-Verlag, 1999.

[8] L. N. de Castro and F. J. V. Zuben, "Learning and optimization using the clonal selection principle," *IEEE Trans. Evol. Comput.*, vol. 6, pp. 239–251, Jun. 2002.

[9] G. Nicosia, "Immune algorithms for optimization and protein structure prediction," Ph.D. dissertation, Dept. Math. Comput. Sci., Univ. Catania, Catania, Italy, 2004.

[10] G. B. Bezerra, L. N. de Castro, and F. J. V. Zuben, "A hierachical immune network applied to gene expression data," in *Proc. 3rd Int. Conf. Artif. Immune Syst.*, G. Nicosia, V. Cutello, P. Bentley, and J. Timmis, Eds., Catania, Italy, Sep. 2004, pp. 14–27.

[11] V. Cutello, G. Narzisi, and G. Nicosia, "A multi-objective evolutionary approach to the protein structure prediction problem," *J. Royal So. Interface*, vol. 3, no. 6, pp. 139–151, Feb. 2006.

[12] S. Ichikawa, A. Ishiguro, S. Kuboshiki, and Y. Uchikawa, "A method of gait coordination of hexapod robots using immune networks," *J. Artif. Life Robotics*, vol. 2, pp. 19–23, 1998.

[13] S. Singh and S. Thayer, "Kilorobot search and rescue using an immunologically inspired approach," in *Distributed Autonomous Robotic Systems*. Berlin, Germany: Springer-Verlag, June 2002, vol. 5.

[14] L. Kesheng, Z. Jun, C. Xianbin, and W. Xufa, "An algorithm based on immune principle adopted in controlling behavior of autonomous mobile robots," *Comput. Eng. Appl.*, vol. 5, pp. 30–32, 2000.

[15] D. Dasgupta, "An artificial immune system as a multiagent decision support system," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, San Diego, CA, Oct. 1998, pp. 3816–3820.

[16] J. Kim and P. Bentley, "Towards an artificial immune system for network intrusion detection: An investigation of clonal selection with negative selection operator," in *Proc. IEEE Int. Congr. Evol. Comput.*, Seoul, Korea, May 2001, pp. 1244–1252.

[17] D. Dasgupta and F. A. Gonzalez, "An immunity-based technique to characterize intrusions in computer networks," *IEEE Trans. Evol. Comput.*, vol. 6, pp. 281–291, Jun. 2002.

[18] V. Cutello and G. Nicosia, "An immunological approach to combinatorial optimization problems," in *Proc. 8th Ibero-American Conf. Artif. Intell.*, Seville, Spain, Nov. 2002, pp. 361–370.

[19] V. Cutello, G. Nicosia, and M. Pavone, "A hybrid immune algorithm with information gain for the graph coloring problem," in *Proc. LNCS on Genetic and Evol. Comput. Conf.*, Chicago, IL, Jul. 2003, vol. 2723, pp. 171–182.

[20] E. Hart and P. Ross, "The evolution and analysis of a potential antibody library for use in job-shop scheduling," in *New Ideas in Optimization*, D. Corne, M. Dorigo, and F. Glover, Eds. London, U.K.: McGraw-Hill, 1999.

[21] D. Dasgupta and N. S. Majumdar, "Anomaly detection in multidimensional data using negative selection algorithm," in *Proc. IEEE World Congr. Comput. Intell., Congr. Evol. Comput.*, Honolulu, HI, May 2002, pp. 1039–1044.

[22] D. Bradley and A. M. Tyrrell, "Hardware fault tolerance: An immunological solution," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, Nashville, TN, Oct. 2000, pp. 107–112.

[23] P. J. C. Branco, J. A. Dente, and R. V. Mendes, "Using immunology principles for fault detection," *IEEE Trans. Ind. Electron.*, vol. 50, no. 2, pp. 362–373, 2003.

[24] S. Forrest, A. S. Perelson, L. Allen, and R. Cherukuri, "Self-nonself discrimination in a computer," in *Proc. IEEE Symp. Research in Security and Privacy*, Oakland, CA, May 1994, pp. 202–212.

[25] D. Dasgupta, Y. Cao, and C. Yang, "An immunogenetic approach to spectra recognition," in *Proc. Int. Conf. Genetic and Evol. Comput. Conf.*, Orlando, FL, Jul. 1999, pp. 149–155.

[26] J. Timmis and M. Neal, "A recourse limited artificial immune system for data analysis," *Knowledge Based Systems*, vol. 14, no. 3–4, pp. 121–130, 2001.

[27] S. Hedberg, "Combating computer viruses: Ibm's new computer immune system," *IEEE Parallel and Distributed Technology: Systems & Applications*, vol. 4, no. 2, pp. 9–11, 1996.

[28] G. Nicosia, V. Cutello, P. Bentley, and J. Timmis, in *Proc. 3rd Int. Conf. Art. Immune Syst.*, 2004.

[29] S. Stepney, R. Smith, J. Timmis, and A. Tyrrell, "Towards a conceptual framework for artificial immune systems," in *Proc. LNCS on Artif. Immune Syst.*, G. Nicosia, V. Cutello, P. Bentley, and J. Timmis, Eds., Catania, Italy, Sep. 2004, vol. 3239, pp. 53–64.

[30] A. Freitas and J. Timmis, "Revisiting the foundations of artificial immune systems: A problem oriented perspective," in *Proc. LNCS on Artificial Immune Syst.*, J. Timmis, P. Bentley, and E. Hart, Eds., Edinburgh, U.K., Sep. 2003, vol. 2787, pp. 229–241.

[31] M. Levitt, "Protein folding by restrained energy minimization and molecular dynamics," *J. Mol. Biol.*, vol. 170, pp. 723–764, 1983.

[32] D. G. Covell, "Folding protein α-carbon chains into compact forms by Monte Carlo methods," *Proteins: Struct. Funct. Genet.*, vol. 14, no. 4, pp. 409–420, 1992.

[33] E. Alm and D. Baker, "Prediction of protein-folding mechanisms from free-energy landscapes derived from native structures," in *Proc. Nat. Acad. Sci. USA*, 1999, vol. 96, no. 20, pp. 11305–11310.

[34] V. Muñoz and W. A. Eaton, "A simple model for calculating the kinetics of protein folding from three dimensional structures," in *Proc. Natl. Acad. Sci. USA*, 1999, vol. 96, no. 20, pp. 11311–11316.

[35] M. A. Apaydin, D. L. Brutlag, C. Guestrin, D. Hsu, and J.-C. Latombe, "Stochastic roadmap simulation: An efficient representation and algorithm for analyzing molecular motion," in *Proc. Annu. Int. Conf. Comput. Molecular Biol.*, Washington, DC, Apr. 2002, pp. 12–21.

[36] N. M. Amato, K. A. Dill, and G. Song, "Using motion planning to map protein folding landscapes and analyze folding kinetics of known native structures," *J. Comp. Biol.*, vol. 10, no. 3, pp. 239–255, 2003.

[37] K. F. Lau and K. A. Dill, "A lattice statistical mechanics model of the conformational and sequence spaces of proteins," *Macromolecules*, vol. 22, pp. 3986–3997, 1989.

[38] K. A. Dill, S. Bromberg, K. Yue, K. M. Fiebig, D. P. Thomas, and H. S. Chan, "Principles of protein folding: A perspective from simple exact models," *Protein Science*, vol. 4, pp. 561–602, 1995.

[39] K. A. Dill, "Theory for the folding and stability of globular proteins," *Biochemistry*, vol. 24, no. 6, pp. 1501–1509, 1985.

[40] B. P. Blackburne and J. D. Hirst, "Evolution of functional model proteins," *J. Chem. Phys.*, vol. 115, no. 4, pp. 1935–1942, 2001.

[41] N. Krasnogor, B. P. Blackburne, E. K. Burke, and J. D. Hirst, "Multimeme algorithms for protein structure prediction," in *Proc. Int. Conf. Parallel Problem Solving from Nature (PPSN VII)*, Granada, Spain, Sep. 2002, pp. 769–778.

[42] N. Krasnogor, "Towards robust memetic algorithms," in *Recent Advances in Memetic Algorithms*, W. E. H. N. Krasnogor and J. E. Smith, Eds. Berlin, Germany: Springer, 2004.

[43] J. D. Hirst, "The evolutionary landscape of functional model proteins," *Protein Engineering*, vol. 12, no. 9, pp. 721–726, 1999.

[44] R. Unger and J. Moult, "Genetic algorithms for protein folding simulations," *J. Mol. Biol.*, vol. 231, no. 1, pp. 75–81, 1993.

[45] L. Toma and S. Toma, "Contact interactions method: A new algorithm for protein folding simulations," *Protein Science*, vol. 5, pp. 147–153, 1996.

[46] P. Crescenzi, D. Goldman, C. Papadimitriou, A. Piccolboni, and M. Yannakakis, "On the complexity of protein folding," *J. Comp. Biol.*, vol. 5, no. 3, pp. 423–466, 1998.

[47] B. Berger and T. Leighton, "Protein folding in the hydrophobic-hydrophilic model is NP complete," *J. Comp. Biol.*, vol. 5, pp. 27–40, 1998.

[48] H. S. Chan and K. A. Dill, "Comparing folding codes for proteins and polymers," *Proteins: Struct., Funct., Genet.*, vol. 24, pp. 335–344, 1996.

[49] N. Krasnogor, W. E. Hart, J. Smith, and D. A. Pelta, "Protein structure prediction with evolutionary algorithms," in *Proc. Genetic Evol. Comput. Conf.*, Orlando, FL, Jul. 1999, pp. 1596–1601.

[50] F. M. Burnet, *The Clonal Selection Theory of Acquired Immunity*. Cambridge, U.K.: Cambridge Univ. Press, 1959.

[51] V. Cutello, G. Nicosia, and M. Pavone, "Exploring the capability of immune algorithms: A characterization of hypermutation operators," in *Proc. 3rd Int. Conf. Artif. Immune Syst.*, G. Nicosia, V. Cutello, P. Bentley, and J. Timmis, Eds., Catania, Italy, Sep. 2004, pp. 263–276.

[52] ——, "An immune algorithm with hyper-macromutations for the 2d hydrophilic-hydrophobic model," in *Proc. Congr. Evol. Comput.*, Portland, OR, Jun. 2004, pp. 1074–1080.

[53] N. Krasnogor, "Studies on the theory and design space of memetic algorithms," Ph.D. dissertation, Univ. West of England, Bristol, U.K., 2002.

[54] P. E. Seiden and F. Celada, "A model for simulating cognate recognition and response in the immune system," *J. Theor. Biology*, vol. 158, pp. 329–357, 1992.

[55] H. P. Schwefel and G. Rudolph, "Contemporary evolution strategies," in *Proc. 3rd Int. Conf. Advances in Artif. Life*, F. Moràn, A. Moreno, J. J. Merelo, and P. Chac'on, Eds., 1995, pp. 893–907.

[56] S. S. Plotkin and J. N. Onuchic, "Understanding protein folding with energy landscape theory," *Quart. Rev. Biophy.*, vol. 35, no. 2, pp. 111–167, 2002.

[57] D. A. Pelta and N. Krasnogor, "Multimeme algorithms using fuzzy logic based memes for protein structure prediction," in *Recent Advances in Memetic Algorithms*, W. E. H. N. Krasnogor and J. E. Smith, Eds. Berlin, Germany: Springer-Verlag, 2004.

[58] R. Santana, P. Larrañaga, and J. A. Lozano, "Protein folding in 2-dimensional lattices with estimation of distribution algorithms," in *Proc. 5th Int. Symp. Biol. Medical Data Analysis*, Barcelona, Spain, Nov. 2004, pp. 388–398.

[59] A. Shmygelska and H. H. Hoos, "An improved ant colony optimization algorithm for the 2d hp protein folding problem," in *Proc. 16th Canad. Conf. Artif. Intell.*, Halifax, Canada, Jun. 2003, pp. 400–417.
[60] R. Konig and T. Dandekar, "Improving genetic algorithms for protein folding simulations by systematic crossover," *Biosystems*, vol. 50, no. 1, pp. 17–25, 1999.
[61] N. Krasnogor and J. Smith, "A tutorial for competent memetic algorithms: Model, taxonomy, and design issues," *IEEE Trans. Evol. Comput.*, vol. 9, no. 5, pp. 474–488, Oct. 2005.
[62] T. Jiang, Q. Cui, G. Shi, and S. Ma, "Protein folding simulations of the hydrophobic-hydrophilic model by combining tabu search with genetic algorithms," *J. Chem. Phys.*, vol. 119, no. 8, pp. 4592–4596, 2003.
[63] F. Liang and W. H. Wong, "Evolutionary Monte Carlo for protein folding simulations," *J. Chem. Phys.*, vol. 115, no. 7, pp. 3374–3380, 2001.
[64] T. C. Beutler and K. A. Dill, "A fast conformational search strategy for finding low energy structures of model proteins," *Protein Science*, vol. 5, no. 10, pp. 2037–2043, 1996.
[65] M. Milostan, P. Lukasiak, K. A. Dill, and J. Blażewicz, "A tabu search strategy for finding low energy structures of proteins in HP-model," in *Proc. Annu. Int. Conf. Comput. Molecular Biol.*, Berlin, Germany, Apr. 2003, pp. Poster No.5-108.
[66] H. Hsu, V. Mehra, W. Nadler, and P. Grassberger, "Growth algorithms for lattice heteropolymers at low temperatures," *J. Chem. Phys.*, vol. 118, no. 1, pp. 444–451, 2003.
[67] A. Shmygelska, R. Anguirre-Hernandez, and H. H. Hoos, "An ant colony optimization algorithm for the 2d HP protein folding problem," in *Proc. Int. Workshop Ant Algorithms*, Brussels, Belgium, Sep. 2002, pp. 40–52.
[68] C. Cotta, "Protein structure prediction using evolutionary algorithms hybridized with backtracking," in *Artificial Neural Nets Problem Solving Methods*, ser. Lecture Notes in Computer Science, J. Mira and J. Álvarez, Eds. Berlin, Germany: Springer-Verlag, 2003, vol. 2687, pp. 321–328.
[69] V. Cutello, G. Narzisi, G. Nicosia, and M. Pavone, "Clonal selection algorithms: A comparative case study using effective mutation potentials," in *Lecture Notes in Computer Science*, Banff, Canada, Aug. 2005, vol. 3627, Proc. 4th Int. Conf. Artif. Immune Syst., pp. 13–28.
[70] D. A. V. Veldhuizen, J. B. Zydallis, and G. B. Lamont, "Considerations in engineering parallel multiobjective evolutionary algorithms," *IEEE Trans. Evol. Comput.*, vol. 7, no. 2, pp. 144–173, Apr. 2003.

Vincenzo Cutello received the Laurea degree and the Ph.D. degree in mathematics from the University of Catania, Catania, Italy, in 1984 and 1989, respectively, and the M.S. and Ph.D. degrees in computer science from the Courant Institute of Mathematical Sciences, New York University, New York, in 1989 and 1991, respectively.

Since 2001, he has been a Full Professor of Computer Science at the University of Catania. He is currently Chairman of the undergraduate program in Applied Computer Science, and Director of the Interdisciplinary Research Center on Applied Computer Science, University of Catania. He is author and coauthor of more than 100 papers in international journals and conference proceedings. His research activities are currently focused on the design and analysis of evolutionary algorithms, decision procedures, fuzzy logic, biological inspired computing, and artificial immune systems.

Giuseppe Nicosia (M'01) received the Laurea degree and the Ph.D. degree in computer science from the University of Catania, Catania, Italy, in 2000 and 2005, respectively.

Since 2001, he has been Grant-Holder of Cineca Supercomputing Center, Bologna, Italy, in the area of High-Performance Computing. In 2004, he was a Visiting Research Assistant in the Computing Laboratory, Kent University, Canterbury, Kent, U.K. Since October 2006, he has been an Associate Professor of Computer Science at the University of Catania. He is currently involved in the design and development of optimization algorithms for circuit design problems, a joint research project supported by the University of Catania and STMicroeletronics. He is coauthor of more than 40 papers in international journals and conference proceedings, and three coedited books on artificial immune systems. He has chaired various international conferences and workshops in the field of artificial immune systems. His primary research interests are design and analysis of artificial immune systems and immune algorithms, optimization, protein bioinformatics, artificial life, and various aspects of unconventional model of computation. His current research interest lies in the design of hybrid evolutionary and immunological algorithms for dynamic environment and constrained multiobjective optimization problems.

Mario Pavone received the M.Sc. and Ph.D. degrees in computer science from the University of Catania, Catania, Italy, in 1999 and 2004, respectively.

He is currently visiting the IBM-KAIST Bio-Computing Research Center, Korea Advanced Institute of Science and Technology (KAIST), Korea, doing research in computer science. His research interests include biologically inspired computing, including artificial immune systems, combinatorial and numerical optimization, and computational biology, with particular reference on protein structure prediction, multiple sequence alignment, gene regulatory network, and gene expression data.

Jonathan Timmis (M'02) is a Reader at the University of York, York, U.K., in a joint appointment with the Department of Computer Science and Department of Electronics. He has published over 60 papers on artificial immune system related research. He has worked on immune inspired approaches to real-time fault detection in ATM machines, machine learning, optimization, web mining, robot control, software testing, and theoretical aspects of immune inspired systems. His primary research interest is in the computational abilities of the immune, neural and endocrine systems and how they relate to computer science and engineering.

Dr. Timmis is the cofounder of the International Conference on Artificial Immune Systems (ICARIS). He is principle investigator for the EPSRC academic network on artificial immune systems, ARTIST, and co-investigator on new EPSRC funded project exploring the use of immunological modeling techniques for the development of novel immune inspired algorithms for bioinformatics (EP/D501377/1). He is heavily involved with the Grand Challenges for Computer Science in the U.K. and now serves on the GC-7 committee.