# A SELF-ORGANIZING GENETIC ALGORITHM FOR PROTEIN STRUCTURE PREDICTION

**Vinicius Tragante do Ó[1], Renato Tinós[2]**
[1]Departement Computerwetenschappen, Katholieke Universiteit Leuven
Leuven - Belgium
vinicius.tragantedo@cs.kuleuven.be
[2]Grupo de Informática Biomédica, Departamento de Física e Matemática, FFCLRP, Universidade de São Paulo
Ribeirão Preto/SP - Brazil
rtinos@ffclrp.usp.br

## ABSTRACT

In the Genetic Algorithm (GA) with the standard random immigrants approach, a fixed number of individuals of the current population are replaced by random individuals in every generation. The random immigrants inserted in every generation maintain, or increase, the diversity of the population, what is advantageous to GAs applied to complex problems like the protein structure prediction problem. The rate of replaced individuals in the standard random immigrants approach is defined *a priori*, and has a major influence on the performance of the algorithm. In this paper, we propose a new strategy to control the number of random immigrants in GAs applied to the protein structure prediction problem. Instead of using a fixed number of immigrants per generation, the proposed approach controls the number of new individuals to be inserted in the generation according to a self-organizing process. Experimental results indicate that the performance of the proposed algorithm in the protein structure prediction problem is superior or similar to the performance of the standard random immigrants approach with the best rate of individual replacement.

## Keywords

Genetic Algorithms, Random Immigrants, Protein Structure Prediction, Self-Organization, Dynamic Optimization.

## 1. INTRODUCTION

The computation of three-dimensional structures of proteins from their amino acid sequence is one of the most important complex problems in molecular biology. This problem is of extreme importance, since the functionality of a protein is intimately related to its three-dimensional structure, which means that its tertiary structure determines its action. An eventual computational technique capable to predict the tertiary structures of long proteins would make it possible, for example, to develop new drugs with specific molecular structures capable of acting over toxic agents [Lehninger, 2005].

Nowadays, the best-known methods to determine an existing protein's tertiary structure are crystallography and nuclear magnetic resonance [Han & Kambert, 2001], but both methods are expensive and have limitations. Ideally, it would be possible to determine a protein structure based only on its amino acids sequence (*ab initio* approach). However, efficient computational techniques for the protein structure prediction problem are not available yet for proteins with a medium or large number of amino acids.

Genetic Algorithms (GAs), because of their intrinsic characteristics, seem to be suitable for the protein structure prediction problem, mainly in the *ab initio* approach. The main reason is because the protein structure prediction problem can be viewed as a optimization problem in which, given an amino acid sequence, the best structure among all possible structures, i.e., the one with the lowest value of a given energy function, must be found [Mitchell, 1996]. In a GA, a population of chromosomes, representing a series of candidate solutions to an optimization problem (also called individuals), evolves toward better solutions. Selection and reproduction operators are used, inspired in mechanisms employed in the evolution of biological species [Mitchell, 1996]. In GAs, solutions are generally represented by binary vectors, but other encodings such as real variable codification are also possible. The evolution usually starts from a population of randomly generated individuals, and, in each generation, the fitness of each individual in the population is evaluated; the best individuals are sent to the next generation (elitism), and the rest of the new population is formed by the recombination (crossover) of pairs of individuals, submitted to random mutations. The new population is then used in the next generation of the algorithm. Commonly, the algorithm ends when a maximum number of generations is reached.

GAs have been successfully applied to topics in which optimization is a requisite, such as attribute selection [Yang & Honavar, 1998], logistics [Taniguchi *et al.*, 1999], electrical systems [Fukuyama *et al.*, 1996], among others. In the protein structure prediction problem, GAs in the standard configuration have been applied, but without major success, mainly because of two problems. The first problem is the use of imprecise energy function to be minimized. The choice of the energy function in the protein structure prediction has a great impact on the optimization process because the modeling of the energy function is difficult as there are many interactions between atoms, which makes impractical to simulate every detail of the system with sufficient accuracy. For example, in [Schulze-Kremer, 1994], GAs were applied to the protein structure prediction problem and reached even lower energy levels than the protein in its native state; however, this was not enough to determine the real native

state. In [Schulze-Kremer, 1994], a small amount of torsion angles was used as a set of possible solutions for each torsion angle of each amino acid of the protein, based on the most common angles found in a protein database and separated by a 10° distance. Today, computing power has significantly increased, which allows us to use larger angle database sets and more complete force fields. However, the modeling of an efficient energy function to be minimized for large proteins is still an open problem, despite of some good results reached in specific domains.

The second major problem in the protein structure prediction task is the existence of a large number of local optima over a search space that is too large. In problems with a large solution space and with many local optima, even populations with several individuals are not sufficient to efficiently explore the search space. In the standard GA, the problem becomes more critical as the selection and crossover mechanisms cause the premature convergence of the population to local optima. In this way, as pointed in [Tragante & Tinós, 2009], the use of mechanisms to maintain the diversity of the population, and to minimize the premature convergence problem, is important in GAs. In [Tragante & Tinós, 2009], the Random Immigrants approach [Cobb & Grefenstette, 2003] was used to increase the diversity of the population in the protein structure prediction problem. The achieved results indicate that this strategy is efficient in reaching better energy levels than the standard GA for the protein structure prediction problem.

In the GA with the standard Random Immigrants approach, a fixed number of individuals of the current population are replaced by random individuals in every generation. The rate of replaced individuals is defined *a priori*, and has a major influence on the performance of the algorithm. If a small rate is employed, few random immigrants are inserted, what can result in population with small diversity. However, if a large number of individuals are replaced, the number of individuals of the population that explore the current best solutions is smaller, what can result in a lower convergence rate.

In order to keep the convergence not too fast nor too slow, the use of dynamic replacement rate was proposed for Dynamic Optimization Problems [Rohlfshagen & Bullinaria, 2006]. In [Yu *et al.*, 2008], where a GA with binary codification is used, individuals are replaced only if the new individuals keep the allele distribution. In this case, it is necessary to compute the allele distribution of the binary vectors in the population, which is very time consuming. In [Yu *et al.*, 2008], the new individuals are kept in a subpopulation in order to preserve the introduced diversity. The use of the subpopulation is based on [Tinós & Yang, 2007], where the worst individual of the current population and its neighbors are replaced by randomly-generated individuals and kept in a separate population of variable size. Individuals inside the subpopulation generate new individuals by mutation and crossover, which occurs only between parents inside the subpopulation. In [Tinós & Yang, 2007], the increase or decrease of the size of the subpopulation is self-organized, varying according to the diversity of the population, which is not explicitly computed. However, the number of fitness evaluations in each generation is not fixed and the replacement rate (number of individuals replaced by new random individuals) remains constant along the generations.

In this work, a new self-organizing approach for the Random Immigrants strategy based on [Tinós & Yang, 2007] is proposed and applied to the protein prediction problem. In this approach, if the worst individual of the population is among the replaced individuals of the last generation, the replacement rate is increased. In the opposite case, the replacement rate is decreased. In this way, the use of the subpopulation is not needed, and this makes the algorithm much simpler. The number of fitness evaluations is fixed and the replacement rate changes along the generations, which allows modifying the diversity of the population. This property is important to the protein prediction problem, avoiding the premature convergence of the algorithm to local optima.

This article is organized as follows. In Section 2, the methodology, including GAs and protein angles, developed in this work is presented. The results obtained with three amino acid sequences applied in this work are presented in Section 3. Finally, Section 4 presents the conclusions and discussion for the work.

## 2. METHODS

The number of possible combinations for the torsion angles makes the protein structure prediction an NP-hard problem [Pierce & Winfree, 2002]. If we, for example, take a small protein that is four amino acids long and analyze all the possible combinations of angles, that will give us approximately $(3600 \times 3600)^4 = 2.8 \times 10^{28}$ combinations, if we consider a 0.1º interval for each angle, and considering only the main chain angles. Therefore, the combinatorial explosion of angle combinations makes it very difficult for any algorithm to find the optimal solution. On the other hand, if we simplify the range of angles (considering, for example, 1º of variation interval) then the angles are not accurate enough. Also, not all combinations are valid [Ramachandran & Sasisekharan, 1968] and should not be considered as solutions.

Based on this fact, angle database sets, formed by the torsion angles phi ($\phi$) and psi ($\psi$) for the main chain and the angles for the side chain placement, experimentally obtained by nuclear magnetic resonance or crystallography, can be used instead of using freely any combination of values. The side chain angles can vary from one to five angles, thus $\chi 1$ to $\chi 5$, depending on the amino acid (see Figure 1 for a graphical demonstration of these angles). These database sets aim at reducing the GA search space. With this in mind, a set of these angles, based on a large number of structures already determined by nuclear magnetic resonance or crystallography, was recorded into an angle database by the project CADB, Conformational Angle Database

[Sheik *et al.*, 2003]. The side chain also has its own database file, which is based on the Tuffery [Tuffery *et al.*, 1991] database (found in http://bioserv.rpbs.jussieu.fr/doc/Rotamers.html). This database was created starting from observations of protein structures determined by magnetic resonance and crystallography too, and ordered by the frequency of appearance in these structures. The codification of the solution (possible structure of the protein) by the chromosomes of the GA is based in these two database sets.

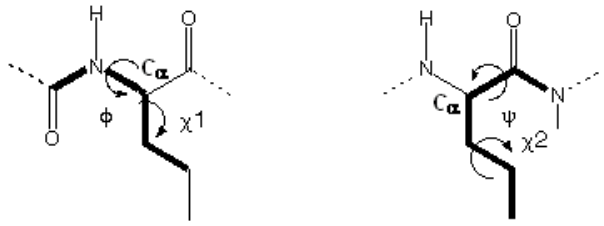


**Fig. 1. Angles used as inputs for the chromosome of each individual of the GA. Starting from Cα, which is the base carbon, we see ϕ and ψ angles which are the torsion angles from the main chain; and χ1 and χ2 angles, which are the angles from the side chain; depending on the amino acid, there can possibly be up to 5 χ angles.**

In this work, the chromosome of each individual is formed by the index of the main chain database of each amino acid and the index of the side chain database, as shown in Figure 2, which means that the chromosome size is 2*m*, where *m* is the size of the protein (number of amino acids). All angle values are saved into an extra vector, which is faster for retrieval of the data than searching the database sets each time a value is needed.

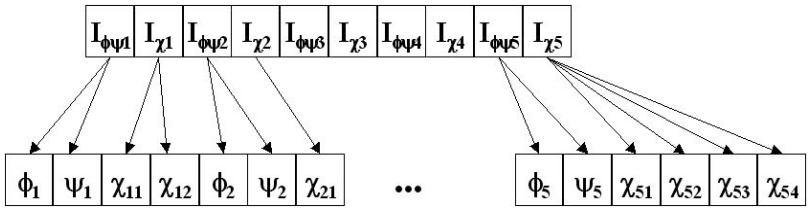


**Fig. 2. Graphical schema of a chromosome for a protein with 5 amino acids. Each amino acid is represented by two values, Iφ and Iψ, which are the indexes of the database set for the main chain and the side chain. An auxiliary vector stores the real values of the angles, in order not to seek the database index each time.**

The parameters of the GA were set as 1/(2*m*) for the mutation rate, what results in an average of one change of the index position for its upper or lower neighbor (the index changes +1 or –1 in its value) in each generation, and 0.8 for the crossover rate. The crossover uses 2 individuals that are chosen by tournament selection, in which, for each offspring, 2 individuals are randomly picked and the individual with the best fitness between them has 75% of being chosen for crossover. The crossover method chosen is the single-point crossover, in which a random point is chosen, the left side of this point is obtained from the first individual chosen and the right side of this point, from the second individual to generate the offspring 1, with the opposite method for the offspring 2, and these two individuals are automatically inserted in the next population after mutation (Figure 3). We inserted also an elitist process, by finding the two best individuals from the generation and automatically inserting them in the next population, without crossover or mutation. This procedure is adapted from the Schulze-Kremer approach [Schulze-Kremer, 1994], and has been used also in other works [Lima *et al.*, 2007], [Cui *et al.*, 1998].

When an individual is completely defined, the GA creates a file that contains the $\phi$, $\psi$, $\omega$ and $\chi$ angles of each amino acid and sends it to the molecular modeling package *Tinker* [Ponder *et al.*, 1998], which convert these torsion angles into a *pdb* or *xyz* file. The *pdb* file is the type of representation found in the PDB database (http://www.rcsb.org/), while the *xyz* file is a representation of each atom of the protein in its space coordinates. This step is necessary to evaluate the fitness of each candidate solution (individual), in the algorithm *Analyze*, also a component of the molecular modeling package Tinker. This program uses the file generated by the protein algorithm to verify the interactions that are present in the protein and informs the total energy of each individual, which is returned to the GA as the fitness of the individual. The purpose of the GA is to reduce this energy to the minimum possible value.

The evaluation of the Tinker package depends on the force field chosen. We decided to use the Charmm .27 force field, since this force field is quite complete in terms of modeled interactions, and is also a popular force field used in other works. This force field seems to be a suitable approach for a close modeling of the existing interactions in a protein, because of its amplitude. This force field is formed by 7 components, explained below:

$$E_{tot} = E_{bs} + E_{ab} + E_{UB} + E_{id} + E_{ta} + E_{VdW} + E_{cc} \quad (1)$$

where:

- $E_{tot}$ : Total energy (fitness);

- $E_{bs}$ : Bond stretching energy, which measures the energy according to the distance of the bonding. If the bonding is compressed then the electrons overlap other electrons, and this energy is low;
- $E_{ab}$ : Angle bending energy, given by the sum of all interactions between angles of the structure, usually lower than bond stretching;
- $E_{UB}$ : Urey-Bradley energy, representing the interactions between pairs of atoms separated by two atomic bondings;
- $E_{it}$ : Improper dihedral energy, which is associated to deformations of improper torsion angles. It refers to atoms with sp2 hybridization, that generate deformations out of the plan;
- $E_{ta}$ : Torsional angle energy;
- $E_{VdW}$ : Van der Waals energy, which results from the interaction between two atoms, balancing from attraction to repulsion. Repulsion appears in short distances, when the interactions between electrons are strong. Attraction appears in the fluctuations in the charge distribution of the electrons;
- $E_{cc}$ : Charge-charge energy, represented by the Coulomb potential. It varies according to the distance between the atoms.

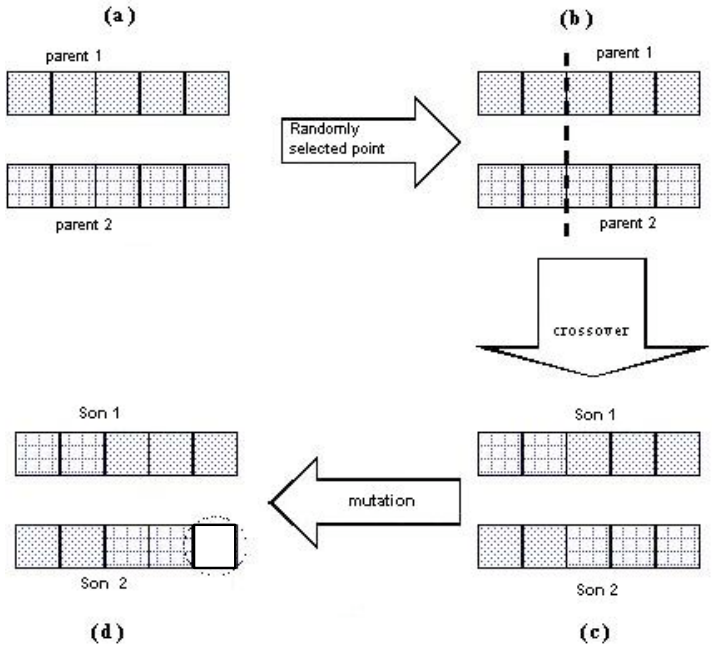


**Fig. 3. Example of the generation of two offspring from two selected parents (a). After the selection of the point of separation (b), a part from parent 1 and other part from parent 2 are recombined to generate offspring (c). The mutation operator may alter one or both individuals (d). Image adapted from [Linden, 2006].**

After the codification of this standard GA, two other strategies were codified, individually, in order to improve the diversity of the individuals over the generations. These procedures are called Random Immigrants and Simplified Self-Organized Random Immigrants, explained below.

The Random Immigrants approach [Cobb & Grefenstette, 1993] replaces a percentage of individuals from the population by new, randomly generated ones. Those replaced individuals are also randomly selected. It is possible to start changing individuals from a particular point or since the beginning of the execution of the algorithm. The pseudo-code below describes this procedure. By *reproduction* (line 8) it is meant the complete procedure, elitism, crossover, and mutation. *Fixed_rate* (line 3) means that it is up to be configured the percentage of individuals to be replaced. After inserting the new individuals, the new population is completed with individuals generated from crossover between individuals from the old population.

**Pseudo-code for the Random Immigrants procedure**

```
1  procedure generation()
2  begin
3    new_individuals <- fixed_rate
4    do
5      add(new_individual)
6    while (total_immigrants < new_individuals)
7    do
8      reproduction(population)
9    while (new_population not full)
10 end.
```

Extending the Random Immigrants approach, we propose a dynamic replacement rate, in which the algorithm would analyze the conditions and decide if the number of new generated individuals should be higher or lower than the number of new chromosomes in the previous generation. This strategy is called *Simplified Self-Organizing Random Immigrants* (SSORIGA), based on the original SORIGA presented in [Tinós & Yang, 2007]. In the original SORIGA algorithm, the worst individual and its neighbors are replaced by randomly-generated individuals. These individuals are kept in a subpopulation in order not to lose this genetic variety right on the following generation, because it is most probable that the fitness of these new individuals is worse than the existing ones. The subpopulation increases or decreases according to the number of new individuals created at a certain generation.

In SSORIGA, instead of creating a subpopulation, new individuals are simply inserted in the following generation, and then they will be evaluated, but as part of the normal population. Before generating the individuals for the next generation, the individual with the worst fitness in the population is identified. If the worst individual is among the immigrants generated in the current generation, then the amount of immigrants is increased for the following generation. Otherwise, the number of random immigrants is restarted from a fixed value. Here, the minimum amount of random immigrants in a generation is 2, and each consecutive time the worst individual is among the immigrants from the previous generation, this number is increased by 2, decreasing by one the number of the necessary crossovers to complete the population for the following generation. In case the number of random immigrants reaches 70% of the population size, the number of replaced individuals is restarted to 2 again.

The pseudo-code described below explains this idea. First, the worst individual is found (line 3). If this individual belongs to the recently-created individuals, then the number of new individuals will be increased in next generation (line 5). Otherwise, the number of new immigrants is reset for the following generation. After the computation of the number of new immigrants, the rest of the population is formed of crossovers among the existing population (line 13).

**Pseudo-code for the SSORIGA procedure**

```
1  procedure generation()
2  begin
3    worst <- find_worst_individual (population)
4    if (worst among new individuals) then
5      increase_number (new_individuals)
6    else
7      reset_number (new_individuals)
8    end_if
9    do
10     add(new_individual)
11   while (total_immigrants < new_individuals)
12   do
13     reproduction(population)
14   while (new_population not full)
15 end.
```

## 3. RESULTS

The proposed algorithm was tested using three proteins obtained from the PDB bank (http://www.rscb.org/pdb): Crambin (code 1CRN), Met-Enkephalin (code 1PLW), and Drosophila Engrailed Homeodomain (code 1ENH). In the first generation, individuals are randomly initialized, by picking random positions of the database sets for each amino acid of each individual. The process is repeated $n$ times, $n$ being the number of individuals per generation. For the proteins Crambin (PDB code 1CRN), which is 46 amino acids long and has been used in other works [Gabriel *et al.*, 2007] [Pedersen & Moult, 1996], and Drosophila Engrailed Homeodomain (PDB code 1ENH), 55 amino acids long and used in [Lima, 2006], 500 generations of 100 individuals were evolved in order to reach the final results; and for the protein Met-Enkephalin (PDB code 1PLW), 5 amino acids long and also used in other works [Nicosia & Stracquadanio, 2008] [Bindewald *et al.*, 1998] [Kaiser *et al.*, 1997], 50 generations of 100 individuals were evolved. Ten different random seeds were used for each protein and each algorithm. The algorithms were developed using Java technology.

All approaches were also tested using both sorted and unsorted main chain database sets. In the first case, the main chain database set for each amino acid is sorted first by the $\phi$ angle, from –180º to 180º. In case of two equal $\phi$ angles, then the $\psi$ angle was also used for sorting. The purpose of this approach is to allow a smooth variation in the fitness landscape via

mutation, since each mutation will perform only slight changes that may be enough to improve the results. This is possible because a mutation will change the index of the angles from the individual to its upper or lower neighbor in the dataset.

The Charmm .27 force field was used, altering the parameter "dielectric" to 78.7, in order to simulate the effect of the water in the protein. Statistical analyses were made in order to compare the performance of the proposed algorithm to the performance of standard GA and the GA with Random Immigrants with different replacement rates. Next sections show the results for each protein.

## 3.1 Crambin

With the Random Immigrants approach, replacement rates of 2%, 6%, 10% and 30% (which were defined from previous tests) of the population for the sorted databases and 2% and 6% for the unsorted database were tested. With the SSORIGA, the replacement rate starts with 2% and changes according to the algorithm previously explained. Table 1 shows the results for these approaches. The presented results (best, worst, median, and mean fitness and standard deviation obtained from all 10 runs) are relative to the best individual found in the optimization process. Notice that higher replacement rates, such as 30%, do not necessarily imply better performance, mainly because there are few generations for the assimilation of the new characteristics introduced by the random immigrants. The "native state", which means the energy found for the conformation of the protein found at PDB, is presented at the end of the table with the purpose of comparison with results obtained.

From the results, one can observe that the SSORIGA approach reaches practically the same result as the Random Immigrants approach, which is proven by a student's T test, which gave nearly 98% of similarity between these two methods, when using the 10% replacement parameter and using the sorted database set; however, an average of only 4.45% of individuals per generation were replaced with SSORIGA. When using the unsorted database set, T test results in a probability of 63.2% of similarity between the methods, considering the 6% replacement rate, but the average replacement rate for the SSORIGA approach was 4.55%, which is also better than the fixed parameter approach. Another important result is that, despite of fixing the maximum number of new individuals per generation in 70%, the highest reached replacement rate was 32%.

One can observe that both methods reach much better results than the Standard GA approach. The high standard deviation for the Standard GA was in general caused by the premature convergence of the population in different local optima, as the initial population was randomly generated with different random seeds.

## 3.2 Met-Enkephalin

For the protein Met-Enkephalin, all approaches reached lower energy potentials than the protein in its native state, a result already reported in the literature [Mitchell, 1996]. Table 2 presents the results found using each of the approaches employed in this work. From Table 2, it is possible to notice that SSORIGA reached the best average of all methods, but the student's T test was not able to identify clearly this difference: the score for this test was 0.866 when comparing SSORIGA with the unsorted database set against the 10% replacement Random Immigrants strategy, which means that there is a high probability that these samples are different. When tested against the other replacement rates, values decrease to 0.485 (6% replacement) and 0.001 (30% replacement); this last result proves the efficiency of SSORIGA over the fixed 30% replacement rate. If we consider the sorted database set, there is a 90.8% probability of equal results between SSORIGA and the Random Immigrants approach, with 10% replacement rate, according to a T test; when tested against 6% and 30%, respectively, T test rates decrease to 0.733 and 0.011; what definitely states that the 30% replacement approach is not a suitable strategy.

The average replacement rate for SSORIGA was 3.255% when using the unsorted database, which is much lower than the fixed tested rate parameters. The maximum number of replacements on one generation was 18 individuals, using the unsorted database. When using the unsorted database, the average replacement rate was 3.464%, and the maximum individuals replaced were 26 in one generation; none of the methods reached the 70% replacement rate, which was the maximum value stated for the algorithm. Both Random Immigrants and SSORIGA approaches also reached better results than the Standard GA approach, with a score of only 0.014 on T test when comparing SSORIGA against the Standard GA, with the unsorted database set.

**Table 1. Fitness of the best individuals, average fitness and standard deviation at the end of the execution of each approach for the protein Crambin**

| Algorithm | Best Fitness | Worst Fitness | Average Fitness | Median | Std. Deviation |
|---|---|---|---|---|---|
| RandIm. 2% (sort) | 561.596 | 594.184 | 574.746 | 581.137 | 11.978 |
| RandIm. 2% (unsort) | 559.022 | 637.158 | 590.557 | 589.029 | 30.941 |
| RandIm. 6% (sort) | 506.252 | 559.859 | **525.767** | **525.340** | 16.054 |
| RandIm. 6% (unsort) | 517.040 | 555.950 | 538.819 | 540.134 | 11.936 |
| RandIm. 10% (sort) | 519.987 | 569.479 | 538.219 | 533.379 | 18.476 |
| RandIm. 30%(sort) | 661.803 | 774.967 | 735.586 | 747.037 | 43.491 |
| SSORIGA (sort) | 527.095 | **550.101** | 538.387 | 539.519 | 8.359 |
| SSORIGA (unsort) | **503.559** | 559.338 | 535.086 | 533.946 | 20.984 |
| Standard GA (sort) | 695.754 | 1042.387 | 831.733 | 817.469 | 110.018 |
| Standard GA (unsort) | 626.908 | 1474.926 | 816.237 | 763.061 | 247.777 |
| Native state | 465.538 | | | - | |

**Table 2. Fitness of the best individuals, average fitness and standard deviation at the end of the execution of each approach for the protein Met-Enkephalin**

| Algorithm | Best Fitness | Worst Fitness | Average Fitness | Median | Std. Deviation |
|---|---|---|---|---|---|
| RandIm. 2% (sort) | 43.420 | 48.399 | 46.577 | 46.781 | 1.284 |
| RandIm. 2% (unsort) | 44.602 | 47.821 | 46.618 | 46.706 | 0.899 |
| RandIm. 6% (sort) | 44.86 | 47.844 | 46.439 | 46.365 | 0.979 |
| RandIm. 6% (unsort) | 43.404 | **47.160** | **45.737** | **45.786** | 1.246 |
| RandIm. 10% (sort) | 44.847 | 47.862 | 46.160 | 46.131 | 0.848 |
| RandIm. 30%(sort) | 46.426 | 48.819 | 47.907 | 47.930 | 0.670 |
| SSORIGA (sort) | **42.819** | 48.306 | 46.229 | 46.706 | 1.640 |
| SORIGA (unsort) | 43.876 | 47.864 | 46.077 | 45.898 | 1.267 |
| Standard GA (sort) | 45.599 | 48.852 | 47.107 | 46.689 | 1.092 |
| Standard GA (unsort) | 46.203 | 49.641 | 47.598 | 47.152 | 1.223 |
| Native State | 345.978 | | | - | |

## 3.3 Drosophila Engrailed Homeodomain

The Drosophila Engrailed Homeodomain is the largest protein tested, 55 amino acids long. For this protein, the best results were reached with the Random Immigrants with a 6% replacement rate, and the student's T test proved this algorithm was better than the Standard GA with less than 1% probability of sampling error. We did not test the 30% replacement rate based on the previous results, that showed that such a high replacement rate is only time-consuming and is not useful for converging to a point, since too many new characteristics are included each generation and these characteristics are replaced too quickly, without sufficient time for the improvement of the performance of the population.

Although the best average has not been reached by SSORIGA, the best individual of all seeds was found using this algorithm. Results from SSORIGA are also statistically better than the Standard GA results, with 1.7% chance of sampling error with the sorted database and 6.9% for the unsorted database. The average replacement rate for SSORIGA was 4.45% per generation when using the sorted database, and 4.55% for the unsorted database, showing that the number of new individuals is smaller than the fixed rates of 6% that reached the best results. Table 3 presents the results obtained for the protein Drosophila Engrailed Homeodomain.

**Table 3. Fitness of the best individuals, average fitness and standard deviation at the end of the execution of each approach for the protein Drosophila Engrailed Homeodomain**

| Algorithm | Best Fitness | Worst Fitness | Average Fitness | Median | Std. Deviation |
|---|---|---|---|---|---|
| RandIm. 2% (sort) | 795.085 | 1408.256 | 1047.238 | 1013.552 | 183.626 |
| RandIm. 2% (unsort) | 704.036 | 1662.950 | 1196.289 | 1056.478 | 458.129 |
| RandIm. 6% (sort) | 691.593 | **739.464** | **713.582** | **713.836** | 12.922 |
| RandIm. 6% (unsort) | 673.558 | 839.261 | 728.646 | 722.505 | 60.821 |
| RandIm. 10% (sort) | 746.979 | 1085.296 | 868.154 | 848.293 | 101.005 |
| SSORIGA (sort) | 732.863 | 1113.551 | 909.881 | 893.529 | 156.665 |
| SSORIGA (unsort) | **639.502** | 934.531 | 759.303 | 754.978 | 89.934 |
| Standard GA (sort) | 1446.176 | 4722.807 | 3721.321 | 3391.329 | 2794.986 |
| Standard GA (unsort) | 1077.668 | 17467.372 | 4290.645 | 2727.655 | 5047.524 |
| Native State | 345.978 | | | - | |

## 4. DISCUSSION

Fig. 4 shows the average replacement rate for all seeds when predicting the Crambin structure using the unsorted database set for SSORIGA. Experiments were the diversity level of the population was recorded indicated that the replacement rate indirectly changes according to the diversity of the population: when the diversity of the population is small, more individuals are generated, i.e., the replacement rate is increased; after the periods where the replacement rate is higher, the diversity increases, which causes smaller replacement rates. This process is self-organized and does not depend on an external evaluation. In this way, it is not necessary to alter the replacement rates by measuring the diversity of the population, what is a very time consuming task if the algorithm has to analyze the diversity of the chromosomes.

The self-organization behavior is caused by changing the number of immigrants in the next generation according to the identification of the worst individual in the previous generation. In the initial generations, the diversity of the population is high and, thus, the probabilities of generating the worst individual in the main population and in the subpopulation of new random individuals are close. In this way, the replacement rate is, in general, small. The same occurs in other points of the evolution when the diversity level is high. When the population reaches local optima, the diversity level of the population becomes small, and most individuals of the main population have similar values of fitness, which are higher than the values of fitness of the random immigrants. In this way, the probability of generating the worst individual from the random immigrants is higher, what can cause an increase in the replacement rate and, as a consequence, in the diversity level of the population. Increasing the number of random immigrants can be useful to the main population escape from the local optima as genes of the random immigrants can be inserted in the individuals of the main population.

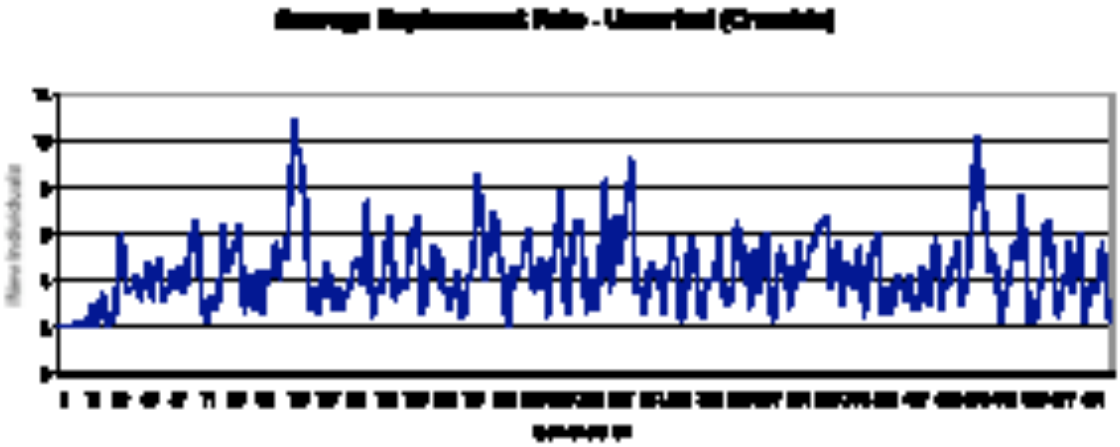

**Fig. 4. Graphical schema of the average individual replacement rate per generation for Crambin, using unsorted database.**

Fig. 5 shows the average replacement rate for all seeds in relation to the Crambin, when using the sorted database set. It is possible to see that more points over the minimum are found after the execution of half part of the generations, when the diversity tends to be lower. In this sense, the algorithm increases the replacement rate in order to keep the diversity in a reasonable pattern.

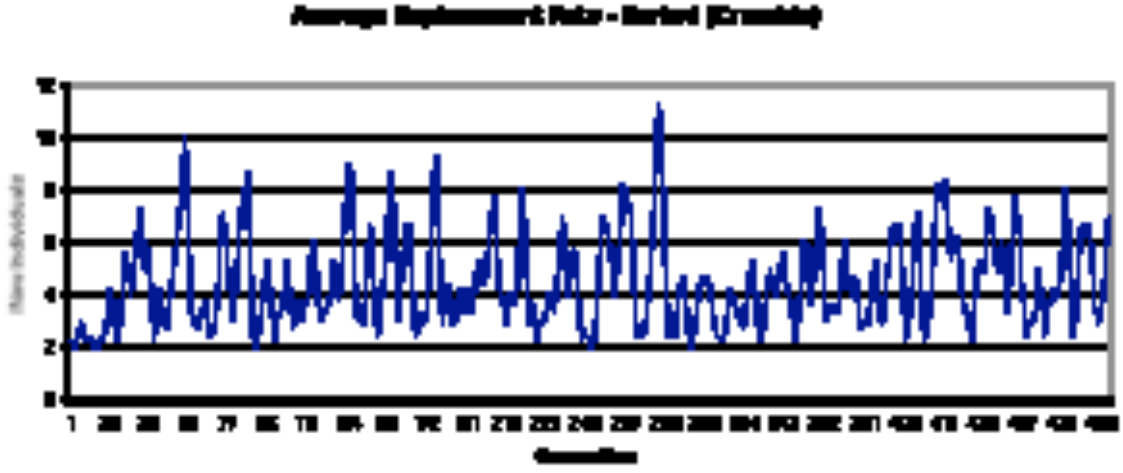

**Fig. 5. Graphical schema of the average individual replacement rate per generation for Crambin, using sorted database.**

Fig. 6 shows the average substitution rate for all seeds for the Met-Enkephalin protein using the unsorted database set, while Fig. 7 shows the same for the sorted database set. From the graphics, we notice a slight increasing rate, because each

generation presents better results, and finding even better results becomes more and more difficult. That is when rates need to be increased, in order to raise the diversity in the population and to find better results. This is exactly the expected result, which means that SSORIGA approach is an efficient method for generating dynamic replacement rates, according to the conditions that the problem is facing along generations.

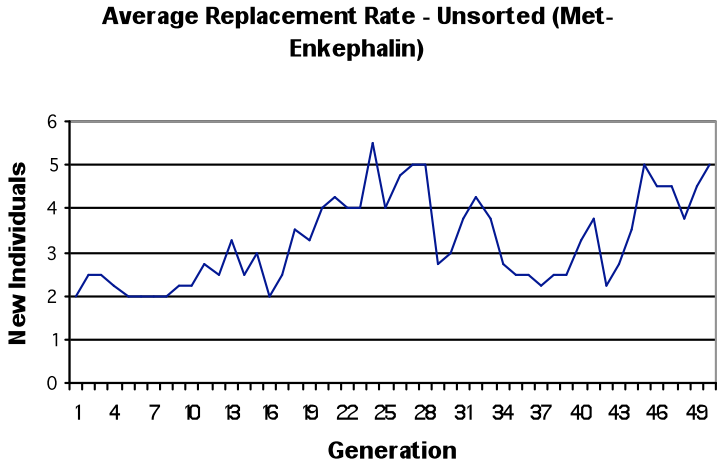


**Fig. 6. Graphical schema of average individual replacement rate per generation (Met-Enkephalin), unsorted dataset.**

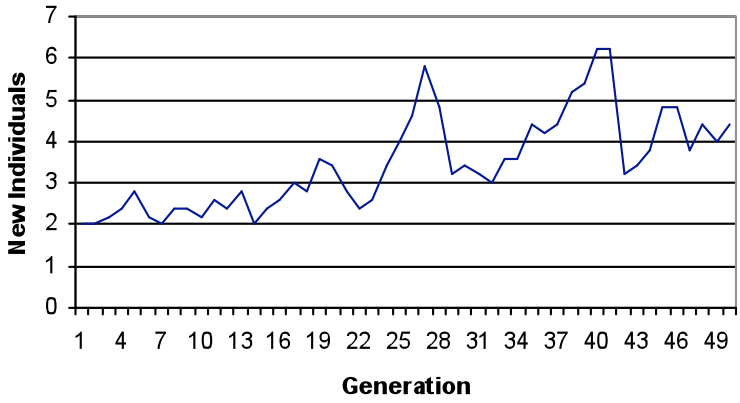


**Fig. 7. Graphical schema of average individual replacement rate per generation for Met-Enkephalin, sorted dataset.**

Similarly, in figures 8 and 9, it is possible to see the curve of replacement of individuals for the Drosophila Engrailed Homeodomain, using the unsorted (Fig. 8) and the sorted (Fig. 9) datasets. These graphs present a similar structure, with regular peaks followed by a dramatic reduction in the number of replacements, when this high number of new individuals is enough to keep the diversity for a number of generations.

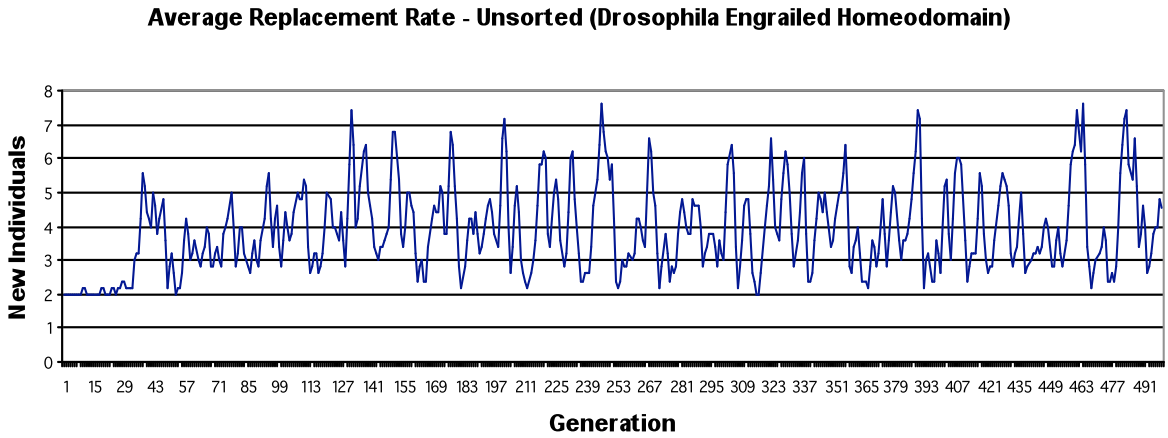


**Fig. 8. Graphical schema of average individual replacement rate per generation for Drosophila Engrailed Homeodomain, unsorted dataset.**

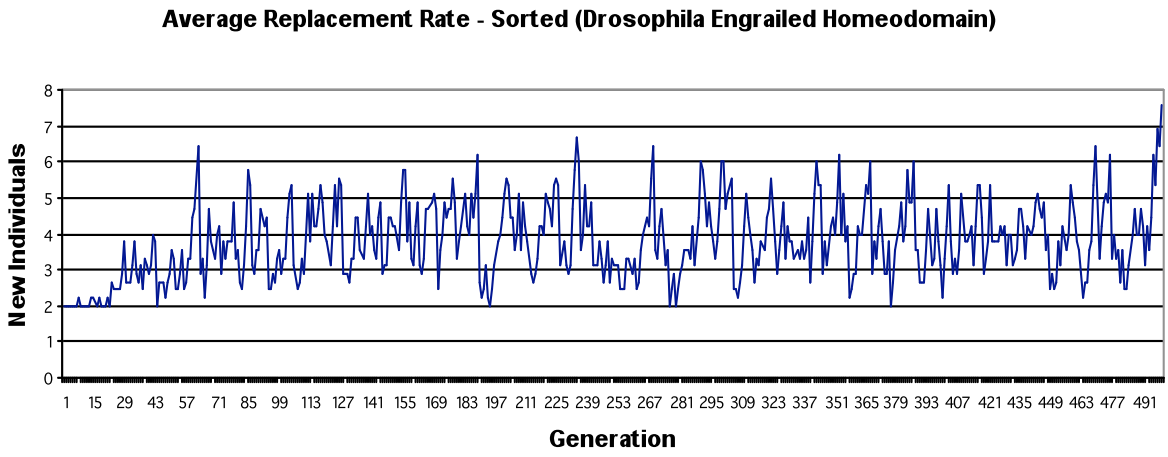


**Fig. 9. Graphical schema of the average individual replacement rate per generation for Drosophila Engrailed Homeodomain, sorted dataset.**

Figure 10 exemplifies the replacement rate for each generation for a single seed. It is possible to see that many times the replacement rates return to the minimum rate, but the maximum rate is never reached, because there is no need for such a high replacement rate, the diversity becomes sufficient with a not so high replacement rate.

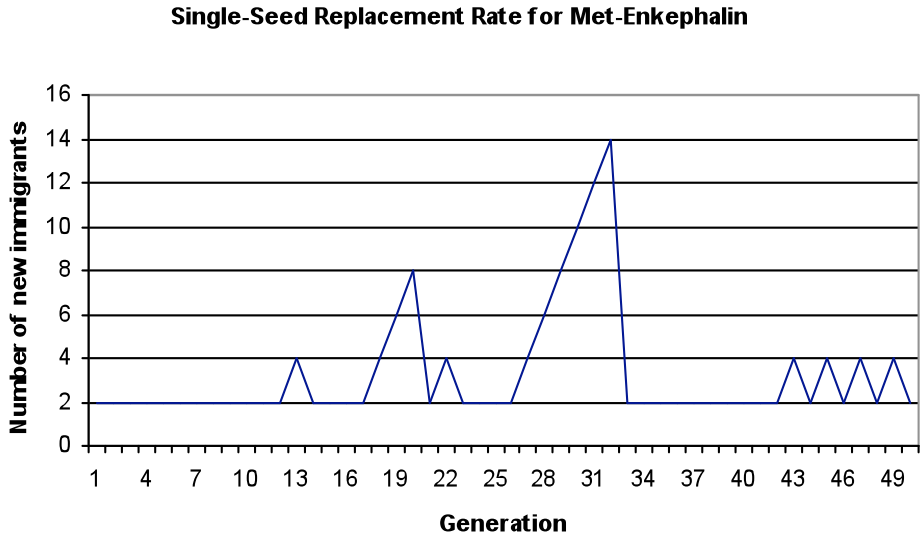


**Fig. 10. Graphical schema of the individual replacement rate per generation for a single seed, predicting structure of the protein Met-Enkephalin, using the unsorted database.**

It is important to notice that the algorithm found different replacement rates for each of the test cases, showing that it is ready for different optimization problems. Besides that, it is able to adjust the insertion of new individuals among generations, starting with low rates, when diversity is high, and increasing them as diversity begins to reduce.

## 5. CONCLUSIONS

In this work we described a new approach for the Random Immigrants strategy on Genetic Algorithms, based on the dynamic replacement of individuals over the generations, according to the conditions of the environment, called SSORIGA. Results showed that the efficiency of the algorithm is statistically comparable to the original Random Immigrants approach with the best replacement rate found, but with fewer replacements of individuals. This makes the algorithm run faster

The SSORIGA algorithm was able to reach lower energy levels than the previous genetic algorithms tested in many cases, and also was capable of keeping the diversity of characteristics in the population with a smaller number of new individuals per generation, allowing these new alleles to have the chance of being spread if they are worth it. This is very important, in the sense that it is less likely for the individuals to be kept in local optima, and more possibilities of the search space can be explored.

However, experimental comparisons using graphical tools for building protein structures showed that these final results still lack of similarity with the native structures, mainly because of the force field employed. Interactions such as the

hydrophilic/hydrophobic effect must be considered in order to improve the constraints and characteristics related to the process of folding, and then even better results can be reached.

Other approaches that can be considered are the use of different subpopulation and niching strategies, and the insertion of different structures as random immigrants, based on similarity with existent structures obtained from the Protein Database. It sounds plausible that conserved domains with a similar amino acid structure can be used as a model for the protein being searched, so the new individual can be partly created from a real structure and the different amino acids can have their positions selected form the data sets, forming a new structure that is not absolutely random, and can improve the result approximating by similarity. These ideas will be considered for the next steps of this work.

## 6. ACKNOWLEDGMENTS

This work was supported in part by *Coordenação de Aperfeiçoamento de Pessoal de Nível Superior* (CAPES) and *Fundação de Amparo à Pesquisa do Estado de São Paulo* (FAPESP), both Brazilian Funding Agencies. The authors also thank Nima Taghipour and Wannes Meert, from Katholieke Universiteit Leuven, and Fernando Luis Barroso da Silva, from FCFRP-Universidade de São Paulo, for the useful suggestions on the work. Vinicius Tragante do Ó is supported by Project G.0413.09 "Learning from data originating from evolution" funded by the Research Foundation - Flanders (FWO-Vlaanderen).

## 7. REFERENCES

Bindewald, E., Hesser, J., Manner, R. Implementing genetic algorithms with sterical constrains for protein structure prediction. *Parallel Problem Solving from Nature* (PPSN V), 959–967, Amsterdam, Netherlands, 1998.

Cobb, H.G., Grefenstette, J.J.. Genetic algorithms for tracking changing environments, *Proceedings of the 5th Int. Conf. on Genetic Algorithms*, Morgan Kaufmann, 523-530, 1993.

Cui, Y., Chen, R., Wong, W. Protein Folding Simulation with Genetic Algorithm and Supersecondary Structure Constraints. In: *PROTEINS: Structure, Function, and Genetics*, 31: 247–257, 1998.

Fukuyama, Y., Chiang, H., Miu, K.. Parallel genetic algorithm for service restoration in electric power distribution systems. *International Journal of Electrical Power and Energy Systems*, 18(2):111–119, 1996.

Gabriel, P., Lima, T., Delbem, A., Faccioli, R., Silva, I. Pure ab initio evolutionary approach to protein structure prediction. *Proceedings of the International Symposium on Mathematical and Computation Biology. BIOMAT 2007, Armação dos Búzios,* Brazil, 2007.

Han, J. & Kambert, M. *Data Mining: Concepts and Techniques*. Morgan Kaufmann, San Francisco, CA, 2001.

Kaiser Jr., C.E., Lamont, G.B., Merkle, L.D., Gates Jr., G.H., Patcher, R. Polypeptide structure prediction: Real-valued versus binary hybrid genetic algorithms. *Proceedings of the ACM Symposium on Applied Computing (SAC)*, 279–286, San Jose, CA, 1997.

Lehninger, A.L., Nelson, D.L., Cox, M.M. *Principles of Biochemistry* 4 ed., Freeman, New York, 2005.

Lima, T. *Algoritmos Evolutivos para Predição de Estruturas de Proteínas*. Master's thesis (Institute of Mathematical and Computational Sciences), Universidade de São Paulo, Brasil, 2006.

Lima, T., Gabriel, P., Delbem, A., Faccioli, R., Silva, I. Evolutionary algorithm to ab initio protein structure prediction with hydrophobic interactions. *IEEE Congress on Evolutionary Computation*, 2007 (CEC 2007), 612–619, 2007.

Linden, R. *Algoritmos Genéticos*. Ed. Brasport, Brazil, 2006.

Mitchell, M. *An Introduction to Genetic Algorithms*. Bradford Books, 1996.

Nicosia, G., Stracquadanio, G. Generalized Pattern Search Algorithm for Peptide Structure Prediction. *Biophysical Journal*, 95: 4988–4999, 2008.

Pedersen, J. Moult, J. Genetic algorithms for protein structure prediction. *Current Opinion in Structural Biology*, 6(2): 227–231, 1996.

Pierce, N.A., Winfree, E. Protein Design is NP-hard. *Protein Engineering*, 15 (10): 779-782, 2002.

Ponder, J. *et al. TINKER: Software Tools for Molecular Design*. Department of Biochemistry and Molecular Biophysics, Washington University School of Medicine, St. Louis, MO, 1998.

Ramachandran, G.N., Sasisekharan, V. Conformation of polypeptides and proteins. Advances in Protein Chemistry 23: 283–437, 1968.

Rohlfshagen, P., Bullinaria, J.A. Alternative Splicing in Evolutionary Computation: Adaptation in Dynamic Environments. *Proceedings of the 2006 IEEE Congress on Evolutionary Computation (CEC 2006)*, 2006.

Schulze-Kremer, S., Tiedemann, U.: Parameterizing genetic algorithms for protein folding simulation. *Proceedings of the 27 th Annual Hawaii Inernational. Conference on System Sciences*, 345–354, 1994.

Sheik, S., Ananthalakshmi, P., Bhargavi, G., Sekar, K. CADB: Conformation Angles DataBase of proteins. *Nucleic Acids Research,* 31(1):448–451, 2003.

Taniguchi, E., Noritake, M., Yamada, T., Izumitani, T. Optimal size and location planning of public logistics terminals. *Transportation Research Part E*, 35(3): 207–222, 1999.

Tinós, R., Yang, S. A self-organizing random immigrants genetic algorithm for dynamic optimization problems. *Genetic Programming and Evolvable Machines*, 8(3): 255-286, 2007.

Tragante, V., Tinós, R. Controle da Diversidade da População em Algoritmos Genéticos Aplicados na Predição de Estruturas de Proteínas". *Scientia (Unisinos),* 20(2): 83-93, 2009.

Tuffery, P., Etchebest, C., Hazout, S., Lavery, R.. A new approach to the rapid determination of protein side chain conformations. *Journal of Biomolecular Structure Dynamics*, 8(6):1267–89, 1991.

Yang, J., Honavar, V. Feature subset selection using a genetic algorithm. *In: Feature Extraction, Construction and Selection: a data mining perspective,* 117-136. Kluwer, 1998.

Yu, X., Tang, K., Yao, X. An Immigrants Scheme Based on Environmental Information for Genetic Algorithms in Changing Environments. *Proceedings of the 2008 IEEE Congress on Evolutionary Computation (CEC 2008)*, Hong Kong, 2008.