Differential Evolution for Protein Structure Prediction Using the HP Model

J. Santos and M. Diéguez

Computer Science Department, University of A Coruña, Spain
{jose.santos,martin.dieguez}@udc.es

Abstract. We used Differential Evolution (DE) for the problem of protein structure prediction. We employed the HP model to represent the folding conformations of a protein in a lattice. In this model the nature of amino acids is reduced considering only two types: hydrophobic residues (H) and polar residues (P), which is based on the recognition that hydrophobic interactions are a dominant force in protein folding. Given a primary sequence of amino acids, the problem is to search for the folding structure in the lattice that minimizes an energy potential. This energy reflects the fact that the hydrophobic amino acids have a propensity to form a hydrophobic core. The complexity of the problem has been shown to be NP-hard, with minimal progress achieved in this category of ab initio folding. We combined DE with methods to transform illegal protein conformations to feasible ones, showing the capabilities of the hybridized DE with respect to previous works.

1 Brief Introduction to the Problem of Protein Structure Prediction

Proteins only attain their sophisticated catalytic activities by folding into complex 3D structures. The amino acid hydrophobicity plays an important role in defining this folding process: for reasons of thermodynamics, most proteins fold with hydrophobic side-chains pointing inwards to form a hydrophobic interior and present a hydrophilic surface to the watery cytoplasm of the cell. Anfinsen [1] showed that a protein in its natural environment folds into a unique three dimensional structure, the native structure, independent of the starting conformation. The thermodynamic hypothesis states that the native conformation of the protein is the one with lowest Gibbs free energy. That native state of a protein plays an essential role in the functionality of a protein. As experimental determination of the native conformation is still difficult and time consuming, much work has been done to forecast the native conformation computationally.

There is an extensive research done on the direct prediction of the final protein structure conformations (secondary and tertiary structures). In the case of secondary structure prediction (local regular elements such as helixes and strands), machine learning methods such as neural nets and support vector machines, can achieve up 80% overall accuracy in globular proteins [7].

In the case of the prediction of the final tertiary structure, the methods range from comparison methods with resolved structures to the “ab initio” prediction. In the first case, the search space is pruned by the assumption that the target protein adopts a structure close to the experimentally determined structure of another homologous protein. But the output of experimentally determined protein structures -by time-consuming and relatively expensive X-ray crystallography or NMR spectroscopy- is lagging far behind the output of protein sequences. Because of this, the most difficult *ab initio* prediction is a challenge in bioinformatics. It uses only the information from the amino acid sequence of the primary structure [21]. In such prediction there are models that simplify the complexity of the interactions and the nature of the amino acid elements, like the models that locate these in a lattice, or detailed atomic models like the Rosetta system [20]. Nevertheless, as Zhao [24] indicates, such detailed atomic models would not be able to explore more than small changes that occur over very small timescales and they involve many parameters and approximations. For this reason, simplified or minimalist models are employed. The use of a reduced alphabet of amino acids is based on the recognition that the binary pattern of hydrophobic and polar residues is a major determinant of the folding of a protein.

In the HP model [6] the elements of the chain can be of two types: H (hydrophobic residues) and P (polar residues). The sequence is assumed to be embedded in a lattice that discretizes the space conformation and can exhibit different topologies such as 2D square or triangular lattices, or 3D cubic or diamond lattices. The interaction between two H elements that are adjacent in the lattice (and not consecutive in the primary sequence) is -1 and zero for the other possible pairs. That is, the HP energy matrix only implies attractions (H with H), and neutral interactions (P with P and P with H). Given a primary sequence, the problem is to search for the folding structure in the lattice that minimizes the energy. The complexity of the problem has been shown to be NP-hard [10,23] and the progress was slow; as Unger points out “minimal progress was achieved in the category of ab initio folding” [22]. Although the HP model is simple, it is powerful enough to capture many properties of actual proteins. It is non-trivial, captures many global aspects of real proteins and still remains the hardness features of the original biological problem [8]. For this reason, many authors have been working on several evolutionary algorithms [22,24] in the direct prediction of the native conformations using the HP model, as we detail in the next section.

Additionally, we must take into account that the energy landscape in this problem presents a multitude of local energy minima separated by high barriers. As Zhao indicates “there are many meta-stable states whose energies are very close to the global minimum and an exceedingly small number of global optimal states. Folding energy landscapes are funnel-like” [24]. For this reason, we will test the capability of Differential Evolution as a method with a better control in the balance between exploration and exploitation with respect to a classical genetic algorithm, as detailed in Section 3. Moreover, we will introduce methods to translate illegal protein conformations to feasible ones, smoothing the landscape (Sect. 3.2). Finally, we will test our proposals with benchmark series (Sect. 4).

2 Previous Work with Evolutionary Algorithms

As we commented, many authors have been working on several evolutionary algorithms, especially genetic algorithms (GA), to determine the final folded conformations using the HP model [22]. One of the main decisions is the encoding of the problem which has to be optimized, in this case, the genotypic encoding of the protein conformation in the lattice. Three basic possibilities can be considered to the representation of the folded sequence in the lattice: Cartesian coordinates and two alternatives with internal coordinates.

In the first case, the location of each amino acid is specified independently with its Cartesian coordinates. With the internal coordinates the embedding of the protein is specified as a sequence of moves taken on the lattice from one amino acid to the next. The first alternative with internal coordinates uses an absolute representation [23] and moves are specified with respect to it. For example, in the case of the square lattice: North, South, East and West. A conformation is expressed as a sequence $\{N, S, E, W\}^{n-1}$, which is the genetic material in the individuals when this representation is used. In the relative representation, relative moves are considered. The reference system is not fixed and the next move depends on the previous move. Now, in the same case as before, three moves are allowed: Forward, Turn Left and Turn Right. The conformations are expressed now as sequences $\{F, L, R\}^{n-1}$. This representation has the advantage of guaranteeing that all solutions are 1-step self-avoiding (because there is no back move).

2.1 Initial Population, Space of Feasible Conformations and Fitness Alternatives

Unger and Moult [23] only considered feasible conformations in the genetic population, that is, embeddings that are self-avoiding paths on the considered lattice. In their work, when operators such as mutation and crossover were applied, the GA iterated until a "nonlethal" (in authors' words) conformation was generated. Similarly, Rylance [17] obtained the initial population using a recoil growth algorithm. Song et al. [19] applied the genetic operators used in [3], with 6 types of mutations; according to the authors, one of them (cornerchange, which mutates one shape to another shape) makes the conformation has more biological significance.

Patton et al. [15] used the relative representation. In addition, unlike the previous authors, they allowed illegal conformations, but using a penalty in the fitness to avoid self-avoiding constraints. This essentially allows the search to proceed through illegal states. With these modifications the results were a bit better in terms of minimization of energy and required computational steps. Krasnogor et al. [12] analyzed the impact of different factors when evolutionary algorithms are used to the problem: the conformational representation, the energy formulation and the way in which infeasible conformations are penalized. Their results supported the use of the relative encoding.

Cotta [2] explored alternatives to the penalty approach. He used a genetic algorithm hybridized with a backtracking algorithm or repair procedure that maps infeasible solutions to feasible conformations. The combination provided slight better results, with respect to a penalty-based approach and to a feasible-space approach. Custódio et al. [4] included a modification in the usual fitness function. The modification was based on the assumption that it may be preferable for a hydrophobic monomer to have a polar neighbor than to be in direct contact with the polar solvent. Their results suggest that the modified model has a greater tendency to form globular structures. Lopes and Scapin [14] used a term in the fitness function, radius of gyration, which indicates how compact a set of amino acids is. Hence, more compact conformations are rewarded.

2.2 Other Alternatives of Optimization

Within the natural or bioinspired computing field different algorithms have been used. For instance, Shmygelska and Hoos [18] presented the use of an ant colony optimization (ACO) algorithm for the 2D HP model and its extension to the 3D HP model, with the inclusion of a local search, that is, the virtual ants further optimized protein conformations. With a pheromone updating procedure, ants folded a protein adding one amino acid at a time based on the pheromone matrix value, which represents previous experience of previous ants, together with the use of heuristic information. The local greedy search introduced long range mutations in the chain. The algorithm scaled worse with sequence length.

Krasnogor et al. [11] also used multimeme algorithms for the problem. The term multimeme means that a set of local searches are used in the individuals of the genetic population. The authors remarked the robustness of the approach across a range of protein structure models and instances. More recently, Lopes and Bitello [13] applied differential evolution for the 2D HP problem. The authors reported that the DE approach was much more consistent than the GA algorithm, since it achieved the maximum number of HH contacts in almost all runs when several benchmark sequences were used. Finally, Cutello et al. [5] proposed an immune algorithm with two special mutation operators, hypermutation and hypermacromutation, which were incorporated into the proposed algorithm to allow effective searching.

3 Methods. Differential Evolution

Differential Evolution (DE) [16] is a population-based search method. DE creates new candidate solutions by combining existing ones according to a simple formula of vector crossover and mutation, and then keeping whichever candidate solution has the best score or fitness on the optimization problem at hand. The central idea of the algorithm is the use of difference vectors for generating perturbations in a population of vectors. This algorithm is specially suited for optimization problems where possible solutions are defined by a real-valued vector. The basic DE algorithm is summarized in the pseudo-code of Algorithm 1.

Algorithm 1. Differential Evolution Algorithm

```
1: Initialize the population randomly
2: repeat
3:   for all individual $x$ in the population do
4:     Let $x_1, x_2, x_3 \in$ population, randomly obtained $\{x_1, x_2, x_3, x$ different from each other.$\}$
5:     Let $R \in \{1, ..., n\}$, randomly obtained {n is the length of the chain.}
6:     for $i = 1$ to $n$ do
7:       Pick $r_i \in U(0,1)$ uniformly from the open range (0,1).
8:       if $(i = R) \vee (r_i < CR)$ then
9:         $y_i \leftarrow x_{1i} + F(x_{2i} - x_{3i})$
10:      else
11:        $y_i = x_i$
12:      end if
13:    end for{$y = [y_1, y_2 ... y_n]$ is a new generated candidate individual}
14:    if $f(y) < f(x)$ then
15:      Replace individual $x$ by $y$
16:    end if
17:  end for
18: until termination criterion is met
19: return $z \in$ population $\backslash \forall t \in$ population , $f(z) <= f(t)$
```

One of the reasons why Differential Evolution is an interesting method in many optimization or search problems is the reduced number of parameters that are needed to define its implementation. The parameters are F or differential weight and CR or crossover probability. The weight factor F (usually in $[0, 2]$) is applied over the vector resulting from the difference between pairs of vectors (x_2 and x_3). CR is the probability of crossing over a given vector of the population (x) and a candidate vector (y) created from the weighted difference of two vectors ($x_1 + F(x_2 - x_3)$). Finally, the index R guarantees that at least one of the parameters (genes) will be changed in such generation of the candidate solution.

The main problem of the genetic algorithm (GA) methodology is the need of tuning of a series of parameters: probabilities of different genetic operators such as crossover or mutation, decision of the selection operator (tournament, roulette,...), tournament size. Hence, in a standard GA it is difficult to control the balance between exploration and exploitation. On the contrary, DE reduces the parameters tuning and provides an automatic balance in the search. As Feoktistov [9] indicates, the fundamental idea of the algorithm is to adapt the step length ($F(x_2 - x_3)$) intrinsically along the evolutionary process. At the beginning of generations the step length is large, because individuals are far away from each other. As the evolution goes on, the population converges and the step length becomes smaller and smaller.

In addition to the usual implementation of DE we used these two strategies: i) Regarding the weighted differences (line 9 in pseudo-code), we used 4 different vectors to calculate the candidate solutions, because, as indicated by Price et al. [16], for large populations four vectors are more effective regarding convergence than the weighted difference of only two vectors. ii) Moreover, the usual implementation of DE chooses the base vector x_1 randomly or as the individual with the best fitness found up to the moment (x_{best}). To avoid the high selective pressure of the latter, the usual strategy is to interchange the two possibilities across generations. Instead of this, we used a tournament to pick the vector x_1, which allows us to easily establish the selective pressure by means of the tournament size.

3.1 Encoding and Fitness

Individuals in DE are real-valued vectors which, in turn, are decoded into a specific fold of an amino acid chain in the square lattice. We used the same real-valued encoding proposed by Lopes and Bitello [13] with relative coordinates. In the bi-dimensional lattices there are only three possible moves: (F)orward, (R)ight and (L)eft. Therefore, the phenotypical representation of a solution is defined over the alphabet F, R, L. The genotype that represents such phenotypical protein conformation is a real-valued vector X. Considering X_{ij} the $j-th$ element of vector X_i, and P the string representing the sequence of moves of the folding, and $\alpha < \beta < \delta < \gamma$ arbitrary constants in R, the genotype-phenotype mapping is defined as follows:

$$\begin{aligned} &If\ \alpha < X_{ij} \leq \beta \ \ then\ P_j = L \\ &If\ \beta < X_{ij} < \delta \ \ then\ P_j = F \\ &If\ \delta \leq X_{ij} < \gamma \ \ then\ P_j = R \end{aligned} \qquad (1)$$

We used $\alpha = -3$, $\beta = -1$, $\delta = 1$ and $\gamma = 3$, as in [13]. If a component of a mutant vector (candidate solution) goes off its limits, as consequence of the DE formula for each candidate solution (line 9 in the pseudo-code), we set the component to a random value in the corresponding interval of the bound limit. Note that the mapping allows several genotypes to represent a single phenotype.

We used initial "stretched" individuals (all elements of the string are F) when creating the initial population. The procedure ensures initial individuals with diversity in the genotypic level, as for the Fs are used random values in the corresponding interval.

Finally, we used the simple fitness function that considers only the maximization of H-H contacts. The infeasible solutions are given a fitness of 0.

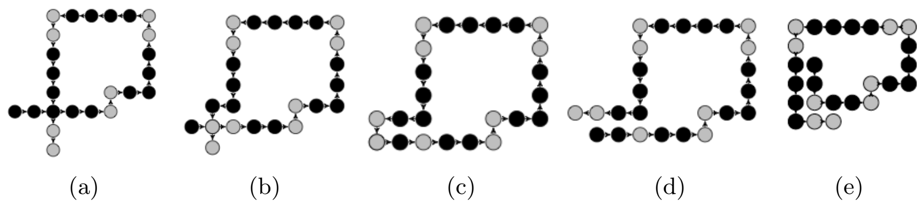

Fig. 1. Repair processes. (a) Original illegal conformation; (b-c) and (d) Intermediate steps and final conformation with the absolute moves procedure; (e) Final conformation with the procedure which uses the Cartesian coordinates.

3.2 Repair Process

This process takes as input an infeasible protein folding (more than one amino acid in the same lattice location) and tries to repair it to a feasible one, trying to maintain most of the original illegal conformation. We implemented two different strategies. The first one works in the Cartesian coordinate space: From an illegal folding, it processes the chain across its amino acids and, if a conflict is encountered, it moves the amino acid that generates the conflict to the nearest free position. In a second stage, the subsequent chain is reconstructed from the new obtained positions, respecting the restrictions of the lattice model.

The second strategy works with the absolute moves (North, South, East and West in the square lattice) to repair the chain conformation. Once a conflict is found, it is changed the absolute move of the amino acid that generates the conflict to obtain a position with no conflict. The procedure is repeated through the rest of the protein conformation. Nevertheless, we did not use a complete search as we did not perform backtracking strategies when there is not any free position after checking all the possible moves. Figure 1 shows an example, where Fig. 1.a is the illegal conformation, Fig. 1.b and Fig. 1.c are intermediate states of this repair procedure and Fig. 1.d is the final feasible conformation. Figure 1.e represents the final rectified conformation from the illegal one of Fig. 1.a using the first procedure with the Cartesian coordinates.

The first strategy tries to obtain legal conformations searching for a similar one in the Cartesian space, while the second tries to maintain the relative conformation of the rest of the chain which has not generated the conflict. In both cases, when there is no possibility of a repair process, the illegal conformation remains in the population (with fitness 0, so it will soon disappear).

Table 1. Benchmarks sequences, for the square lattice, used in the experiments

	Length	**HP Chain**	**E**
S1	20	$HPHP^2H^2PHP^2HPH^2P^2HPH$	-9
S2	24	$H^2P^2HP^2HP^2HP^2HP^2HP^2HP^2H^2$	-9
S3	25	$P^2HP^2H^2P^4H^2P^4H^2P^4H^2$	-8
S4	36	$P^3H^2P^2H^2P^5H^7P^2H^2P^4H^2P^2HP^2$	-14
S5	48	$P^2HP^2H^2P^2H^2P^5H^{10}P^6H^2P^2H^2P^2HP^2H^5$	-23
S6	50	$H^2PHPHPHPH^4PHP^3HP^3HP^4HP^3HP^3HPH^4PHPHPHPH^2$	-21
S7	60	$P^2H^3PH^8P^3H^{10}PHP^3H^{12}P^4H^6PH^2PHP$	-36
S8	64	$H^{12}PHPHP^2H^2P^2H^2P^2HP^2H^2P^2H^2P^2HP^2H^2P^2H^2P^2HPHPH^{12}$	-42

4 Results

We tested the DE implementation hybridized with the repair processes using benchmark sequences employed in the literature. Table 1 shows these sequences with different number of amino acids and the corresponding energy minima (maximum number of HH contacts known up to now).

Regarding the DE implementation, we used standard values: $CR = 0.9$ and $F = 0.9$, whereas the size of the tournament to choose the base vector was 8% of the population, which implies a low selective pressure when choosing such vector to disturb.

Regarding the repair processes, as there is not a clear rule to determine what the best strategy is, we used the two repair processes implemented. The first one, which works with the Cartesian coordinates, is used beginning the conflict checking in the first amino acid of the protein chain as well as starting it from the center of the chain. The second repair strategy, with absolute moves, always begins from the first amino acid. These three possibilities were used with the same probability when the candidate solution used by DE generates an illegal protein conformation.

Figure 2 shows a comparison between the two alternatives, using DE with and without the repair processes. The protein series used was one of the benchmark sequences shown in Table 1 (sequence *S2*). The quality evolutions in the Figure are an average of 5 different evolutions with different initial populations. As the repair processes allow that almost all the individuals in the population are feasible, DE combined with them obtains the optimum configuration in fewer generations. DE without the repair processes needs much more than the 100 generations shown in the Figure to obtain the best configuration in all the runs. For the comparison purpose, we also included in the Figure the evolution of the average quality of the best individual of 5 different runs of a GA with typical parameters (one-point crossover with probability of 0.5, mutations with a probability of 0.02, tournament selection with a size of 4% of the population, and use of the relative encoding).

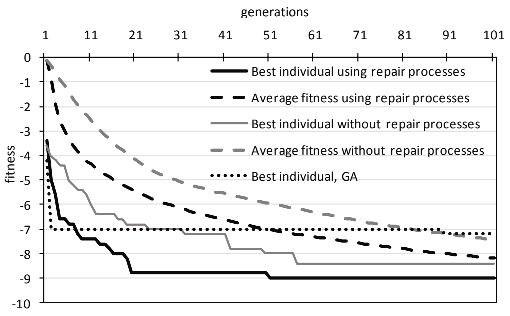

Fig. 2. Comparison of DE evolution with and without the use of the repair processes

In Table 2 we included the comparison of our hybrid DE solution with the results of Unger and Moult with a GA [23], using the benchmark series for the square lattice. In the table, E_{min} represents the maximum number of HH contacts. Previous works summarize the results taking into account independent runs for each protein sequence, and in each column it is specified the best energy value found in the different runs of the corresponding search algorithm. If the best known solution quality was found, it is indicated in bold. The values in parentheses are the minimum number of conformations scanned before the lowest energy values were found in one of the runs. This is the case of the results of Unger and Moult [23] from 5 independent runs. In our case, we included the average energy in the different runs and the average number of

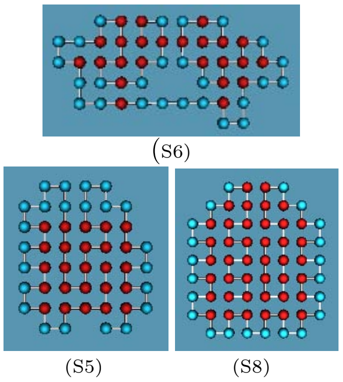

Fig. 3. Optimal configurations for three of the sequences: S5, S6 and S8

Table 2. Comparison of results with the benchmark sequences

Seq.	E_{min}	U&M GA [23]	hybrid DE	L&B DE [13]	ACO [18]
S1	-9	**-9** (30492)	**-9**,-9 (3584, 6362)	**-9**,-9	**-9**
S2	-9	**-9** (30491)	**-9**,-9 (5806, 9292)	**-9**,-9	**-9**
S3	-8	**-8** (20400)	**-8**,-8 (7061, 18828)	**-8**,-8	**-8**
S4	-14	**-14** (301339)	**-14**,-14 (45793, 92579)	**-14**,-13.96	**-14**
S5	-23	-22 (126547)	**-23**,-23 (245943, 532787)	**-23**,-23	**-23**
S6	-21	**-21** (592887)	**-21**,-21(365222, 691989)	**-21**,-21	**-21**
S7	-36	-34 (208781)	-35,-33.57	-35,-34.79	**-36**
S8	-42	-37 (187393)	**-42**,-42 (176313, 340917)	**-42**,-41.87	**-42**

evaluations that obtained the best optimum value (second values after ","). We used 10 independent runs for each sequence.

There is not a clear rule about what the best number of individuals is, so, for each sequence, we used *population size = number of amino acids x 15* (as suggested in [16] and [13]). One of the main factors to be compared is the number of necessary evaluations to find the optimum. The work of Lopes and Bitello did not include the number of evaluations to find their best values [13], so we only included in the Table the best and average energy values reported by the authors. The hybrid DE with the inclusion of the repair procedures reduces significantly the number of conformations that are needed to scan in order to obtain the best energy values, as it can be seen with such best minimum number of necessary evaluations (first values in parentheses). Even the average values of the number of necessary scanned conformations is lower with respect to the reported values of Unger and Moult with a GA [23] (except for S6), taking into account that their reported values are the best in 5 independent runs. Figure 4 shows three examples of the best folded conformations obtained with the hybrid DE. For the sequence $S7$ the minimum was not obtained, although we allowed a maximum number of 10^6 evaluations in the different runs.

In the case of Shmygelska and Hoos [18], using an ant colony optimization (ACO), the authors obtained the best values in all the sequences. Their results are difficult to compare with the evolutionary algorithms, as the authors used 500 runs of the ACO algorithm (in sequences $S1 - S6$) and 100 runs for sequences $S7$ and $S8$, with a population of 100 ants and, additionally, half of the ants performed a local search with a maximum of 1,000 scans through the protein sequence ($S1 - S6$) or a maximum of 10,000 scans ($S7$ and $S8$). As the authors commented, their empirical results indicated that their "rather simple ACO algorithm scales worse with sequence length but usually finds that more diverse ensemble of native states".

5 Conclusions

We applied DE to the optimization of folded protein conformations using the HP lattice model. DE provides a method with a reduced number of defining parameters in its implementation. The differences of vectors used in DE are adequate to search in the multimodal energy landscape inherent to the HP protein folding problem, where the adaptive perturbations or mutations are applied over the segments of the protein conformations that are different in the vectors or individuals of the genetic population. The hybridization of DE with the repair procedures provided a significant reduction of the necessary conformations to obtain the best energy values in different benchmark sequences.

Acknowledgments. This paper has been funded by the Ministry of Science and Innovation of Spain (project TIN2007-64330).

References

1. Anfinsen, C.B.: Principles that govern the folding of proteins. Science 181(96), 223–230 (1973)
2. Cotta, C.: Protein structure prediction using evolutionary algorithms hybridized with backtracking. In: Mira, J., Álvarez, J.R. (eds.) IWANN 2003. LNCS, vol. 2687, pp. 321–328. Springer, Heidelberg (2003)
3. Cox, G.A., Mortimer-Jones, T.V., Taylor, R.P., Johnston, R.L.: Development and optimization of a novel genetic algorithm for studying model protein folding. Theor. Chem. Acc. 112, 163–178 (2004)
4. Custódio, F.L., Barbosa, H.J.C., Dardenne, L.E.: Investigation of the three dimensional lattice HP protein folding model using a genetic algorithm. Genetics and Molecular Biology 27(4), 611–615 (2004)
5. Cutello, V., Nicosia, G., Pavone, M., Timmis, J.: Immune algorithm for protein structure prediction on lattice models. IEEE T. Evol. Comp. 11(1), 101–117 (2007)
6. Dill, K.A.: Dominant forces in protein folding. Biochemestry 29, 7133–7155 (1990)
7. Dor, O., Zhou, Y.: Achieving 80% tenfold cross-validated accuracy for secondary structure prediction by large-scale training. Proteins 66(4), 838–845 (2007)
8. Dill, K.A.: et al. Principles of protein folding: a perspective from simple exact models. Protein. Science 4(3), 561–602 (1995)
9. Feoktistov, V.: Differential Evolution: In Search of Solutions. Springer, Heidelberg (2006)
10. Hart, W.E., Istrail, S.: Robust proofs of NP-hardness for protein folding: General lattices and energy potentials. Journal of Computational Biology 4(1), 1–22 (1997)
11. Krasnogor, N., Blackburne, B.P., Burke, E.K., Hirst, J.D.: Multimeme algorithms for protein structure prediction. In: Guervós, J.J.M., Adamidis, P.A., Beyer, H.-G., Fernández-Villacañas, J.-L., Schwefel, H.-P. (eds.) PPSN 2002. LNCS, vol. 2439, pp. 769–778. Springer, Heidelberg (2002)
12. Krasnogor, N., Hart, W.E., Smith, J., Pelta, D.A.: Protein structure prediction with evolutionary algorithms. In: Proc. GECCO 1999, pp. 1596–1601 (1999)
13. Lopes, H.S., Bitello, R.A.: Differential evolution approach for protein folding using a lattice model. J. of Computer Science and Technology 22(6), 904–908 (2007)
14. Lopes, H.S., Scapin, M.P.: An enhanced genetic algorithm for protein structure prediction using the 2D hydrophobic-polar model. In: Talbi, E.-G., Liardet, P., Collet, P., Lutton, E., Schoenauer, M. (eds.) EA 2005. LNCS, vol. 3871, pp. 238–246. Springer, Heidelberg (2006)

15. Patton, W.P., Punch, W.F., Goldman, E.: A standard genetic algorithm approach to native protein conformation prediction. In: Proceedings of 6th International Conference on Genetic Algorithms, pp. 574–581 (1995)
16. Price, K.V., Storn, R.M., Lampinen, J.A.: Differential Evolution. A Practical Approach to Global Optimization. Springer - Natural Computing Series (2005)
17. Rylance, G.J.: Applications of genetic algorithms in protein folding studies, PhD Thesis. University of Birmingham (2004)
18. Shmygelska, A., Hoos, H.H.: An ant colony optimisation algorithm for the 2D and 3D hydrophobic polar protein folding problem. Bioinformatics 6, 30 (2005)
19. Song, J., Cheng, J., Zheng, T., Mao, J.: A novel genetic algorithm for HP model protein folding. In: Proceedings of the Sixth International Conference on Parallel and Distributed Computing Applications and Technologies, pp. 935–937 (2005)
20. Rosetta system, `http://www.rosettacommons.org`
21. Tramontano, A.: Protein structure prediction. Concepts and applications. Wiley-VCH (2006)
22. Unger, R.: The genetic algorithm approach to protein structure prediction. Structure and Bonding 110, 153–175 (2004)
23. Unger, R., Moult, J.: Genetic algorithms for protein folding simulations. Journal of Molecular Biology 231(1), 75–81 (1993)
24. Zhao, X.: Advances on protein folding simulations based on the lattice HP models with natural computing. Applied Soft Computing 8, 1029–1040 (2008)